A Performance Study of Adaptive Video Coding Algorithms for High Speed Networks

A Thesis Submitted to the College of

Graduate Studies and Research

in Partial Fulfillment of the Requirements

for the Degree of

Master of Science

in the

Department of Computational Science

University of Saskatchewan

Saskatoon, Saskatchewan

by

Sanjeev Gupta

February, 1995

PERMISSION TO USE

ACKNOWLEDGEMENTS

My thanks to my supervisor Dr. Carey Williamson for being so supportive and helpful. But for his patience and guidance, this thesis would not have been completed on time.

I also owe thanks to my thesis committee: Dr. Rick Bunt, Dr. Tony Kusalik and Dr Eric Salt (external) for their help in completing my thesis.

I appreciate the efforts of Dr. Mark Keil as Graduate Chairman, and of the departmental secretarial staff for their generous help whenever I needed it. I would also like to thank all my friends, without whom my stay in Saskatoon would not have been such a memorable experience.

Finally, I would like to dedicate this thesis to my family, who were a constant support for me all through the program.

A Performance Study of Adaptive Video Coding Algorithms for High Speed Networks

Candidate: Sanjeev Gupta

Supervisor: Carey L. Williamson

Master of Science Thesis

Presented to the College of Graduate Studies and Research

University of Saskatchewan

February, 1995

ABSTRACT

Video coding algorithms are important for the proliferation of multimedia applications (i.e., those that incorporate video, voice and data) in high speed networks. Video coding algorithms reduce the bandwidth required for video images while maintaining good picture quality. Of particular interest are adaptive video coding algorithms, which can adapt the encoding of a video stream (and thus its bandwidth requirements) dynamically based on the amount of bandwidth available in the network. Such applications can make efficient use of the available bit rate (ABR) service class proposed for B-ISDN.

The goal of this research was to evaluate adaptive video coding algorithms. Therefore there were three main objectives in this research. The first objective was to determine the robustness of adaptive video coding algorithms to changes in network load. The second objective was to identify the domain (network types and load characteristics) in which adaptive video coding algorithms are feasible. The third objective was to evaluate four specific video coding algorithms (JPEG, EPIC, Fractal and MPEG) with respect to their adaptivity on high speed networks.

The simulation results in this thesis show that under the experimental conditions, adaptive video coding algorithms are robust across a wide range of network load levels (20% to 80%). However the domain of applicability of adaptive video coding algorithms is quite narrow: moderately to heavily loaded networks with speeds in the order of 10 Mbps and 100 Mbps. Among the intraframe video coding algorithms studied, the JPEG algorithm shows the most promise for use as an adaptive video coding algorithm.

Contents

List of Figures

List of Acronyms and Abbreviations

ABR Available Bit Rate

ALIC Adaptive Linearly Increasing Compression

AQIC Adaptive Quadratically Increasing Compression

ASFC Adaptive Step Function Compression

ASIC Adaptive Sublinearly Increasing Compression

ATM Asynchronous Transfer Mode

B-ISDN Broadband Integrated Services Digital Network

CBR Constant Bit Rate

CCITT Consultative Committee in International Telegraphy and Telephony

DCT Discrete Cosine Transform

DEC Digital Equipment Corporation

DPCM Differential Pulse Code Modulation

EPIC Efficient Pyramid Image Compression

GOP Group Of Pictures

HDTV High Definition Television

JPEG Joint Photographic Experts Group

IEC International Electro-Technical Commission

ISO International Standards Organization

Kbps Kilobits per second

Mbps Megabits per second

MPEG Motion Picture Experts Group

msec milliseconds

NAHC Non-Adaptive High Compression

NALC Non-Adaptive Low Compression

NTSC National Television System Committee

QOS Quality Of Service

VBR Variable Bit Rate

List of Symbols Used

A Adaptivity

B Network Speed

C Compression Ratio

C_{max} Maximum Compression Ratio Achieved

C_{min} Minimum Compression Ratio Achieved

$\overline{E}$ Average Error per Pixel

F_c Compression Constant

F_d Decompression Constant

I_{max} Maximum Image Quality Achieved

I_{min} Minimum Image Quality Achieved

L_n Number of Discrete Levels of Compression Ratios

N Number of Pixels in the Image

P_i Grayscale Value of Pixel Number i in the Original Image

$\hat{P}_i$ Grayscale Value of Pixel Number i in the Reconstructed Image

R_c Range of Compression Ratios Achieved

R_q Range of Image Quality Achieved

S_c Size of Compressed Image

S_u Size of Uncompressed Image

T End-to-end Transfer Time

T_c Compression Time

$\overline{T_c}$ Average Compression Time

T_d Decompression Time

T_t Transmission Time

ρ Network Utilization at Time of Image Compression

Chapter 1

Introduction

1.1 Motivation

Networks such as the Broadband Integrated Services Digital Network (B-ISDN) provide the technology required for the current and prospective proliferation of video applications and multimedia services [6, 29, 80]. These networks would make available the raw bandwidth necessary for such applications. Furthermore, such networks would be able to satisfy the strict quality of service requirements (e.g., minimal delay, jitter, loss) of multimedia applications [43, 46].

In a multimedia application the video component will have the largest bandwidth requirement. For example, one second of uncompressed video, with 640×480 pixels per frame, 24 bits of colour information per pixel (RGB standard), and a display rate of 30 frames per second (NTSC standard) would have bandwidth requirements of over 220 Mbps. Even if the colour information were reduced to 8 bits per pixel, and the display rate to 15 frames per second, over 36 Mbps of bandwidth would be required. Thus, in order to maintain a rate of 30 frames per second, with 640×480 pixels per frame and 8 bits of colour information per pixel, on a dedicated Ethernet LAN (10 Mbps) a compression factor of at least 7.5:1 is needed.

The traffic characteristics of video sources are considerably different from those of voice or data sources. Video traffic tends to be bursty, depending on the type of encoding [79]. That is, the peak bit rate is much higher than the mean bit rate. The burstiness of the video stream depends on the content of the frame, the quality of image required by the receiver, and the video compression algorithm. Thus, video coding algorithms have a large impact on the bandwidth requirements and the resultant traffic pattern of multimedia applications.

The service requirements of video traffic, in terms of delay, are more stringent than data traffic [28, 79]. Video traffic is highly delay sensitive and can tolerate some amount of loss. On the other hand, data traffic is loss sensitive but can tolerate some amount of delay. While studying video coding algorithms care has to be taken to see that the video coding algorithms do not adversely affect the quality of service requirements (loss and delay) of video traffic. In addition, the video traffic should be provided at least the same quality of service after compression as it would be without compression. For example, since video traffic is highly delay sensitive, video coding algorithms should ensure that the processing at the coding and decoding end do not introduce such significant delay that the video sequence as seen by the end receiver is jerky. Also, the compression and decompression process should ensure that the loss is not significant enough to make the received image perceptually unsatisfactory to the end-user.

Under the present resource constraints (bandwidth and storage) video coding algorithms are important for the integration of multimedia services into computer networks. Such algorithms reduce the number of bits transmitted over the network and thus reduce the bandwidth, the storage and the transmission time required by video applications.

Many video coding algorithms have been proposed in the literature [10, 17, 35, 73, 77]. These algorithms can be basically divided into two groups, Constant Bit Rate (CBR) and Variable Bit Rate (VBR) video coding algorithms. CBR video coding algorithms maintain a constant bit rate regardless of the contents of the frames [11, 38, 78, 81]. Thus, the image quality varies from frame to frame. In VBR video coding algorithms, the size of the compressed frame varies from frame to frame and a constant image quality can be maintained [21, 26, 75].

VBR video coding algorithms are tailor-made for packet switched networks for several reasons [36, 50, 58]. First, packet switched networks provide the flexibility of statistical resource sharing by several VBR video sources [39]. Thus, if a video source does not use up all the bandwidth, another source can be multiplexed into the channel. CBR video algorithms, on the other hand, generally assume circuit switched implementation and make use of fixed bandwidth reservation. Secondly, VBR video coding algorithms can guarantee constant image quality regardless of the scene contents. This is very important for certain applications, like medical imaging, that require a constant and high image quality.

The choice of a video coding algorithm depends somewhat on the services offered by the underlying network. For example, if the network only provides performance guarantees using peak bandwidth allocation, then either CBR or simple VBR coding algorithms may suffice. On the other hand, if the network offers a service class for Available Bit Rate (ABR) (also called Free Bit Rate) traffic [69] then adaptive video coding algorithms are attractive. Rather than reserve bandwidth based on the peak bit rate of the video stream, which may be wasteful, or the mean bit rate, which may be inadequate, adaptive video coding algorithms can dynamically adjust the encoding of a video stream, and thus its bandwidth requirements, to the available capacity in a reservationless network. In addition to making better use of the available bandwidth on the network, adaptivity may make the video coding algorithms robust to changes in the network load by providing more consistent image transmission times.

The choice of a video coding algorithm also affects end-user satisfaction. The end-user satisfaction depends on two factors: the time it takes for the image to get to the user and the quality of the received image. Thus, the lower the time it takes for the image to arrive at the end-user (within a certain user-descernable threshold) and the higher the quality of the image, the better is the end-user satisfaction. In order to transfer perceptually satisfying images, over a period of time, in a feasible amount of time, the video coding algorithm has to adapt to changes in the network load. What this means is that when the network is lightly loaded, the video coding algorithm should provide very little compression and a good quality image should be received by the end user in very little time. But when the network is highly loaded, the video coding algorithm should provide very high compression so that an image of reasonable perceptual quality should be received by the end-user in a feasible amount of time. In other words, video coding algorithms should be adaptive.

Research on adaptive video coding algorithms is beginning to appear in the literature [19, 32]. For example, Garrett *et al.* [19] developed a video coder that changed its encoding scheme to maximize the quality of the received image based on feedback about expected loss rates from the network. Although such research efforts show the potential of adaptive video coding, there has not been much work in identifying the general conditions (e.g., network types, load characteristics) under which adaptive video coding algorithms are preferrable to non-adaptive video coding algorithms. It is this latter issue that this thesis addresses.

1.2 Objectives

There are three main objectives in this research. The first objective is to determine if there is any advantage to adaptive video coding and if adaptive video coding algorithms are robust to changes in network load.

The second objective is to identify the domain (network types and load characteristics) under which adaptive video coding algorithms are feasible. This work will identify the load condition and network types under which adaptive video coding algorithms outperform non-adaptive video coding algorithms.

The third objective is to evaluate four specific video coding algorithms, with widely different compression curves, with respect to their adaptivity on high speed networks. The video coding algorithms under consideration are JPEG [4, 77], EPIC [65] and Fractal [5, 8]–which are all intraframe compression algorithms–and MPEG [40], which is an interframe compression algorithm. The algorithms will be experimentally evaluated on an image sequence to obtain their relative performance characteristics. The evaluation of specific video coding algorithms will provide insight into the desirable features required for a video coding algorithm to be adaptive. The MPEG video coding algorithm needs to be studied in order to determine how interframe video coding algorithms adapt to changes in the network load and if the domain of adaptive interframe video coding algorithms is the same as the domain of adaptive intraframe video coding algorithms.

1.3 Outline of Thesis

The remainder of this document is organized as follows:

Chapter 2 provides background on multimedia communication and the service requirements of the components of multimedia communication. The traffic characteristics of multimedia applications are also presented.

Chapter 3 reviews existing coding techniques and what compression means in the realm of adaptive video coding algorithms. General compression techniques and the application of these techniques to video coding algorithms are presented. The need for video coding algorithms, especially adaptive video coding algorithms, is also discussed.

Chapter 4 presents the analytical work. First, a mathematical model of the end-to-end image transfer problem is derived. Second, a few general cases of video coding algorithms are mapped onto the mathematical model and the characteristics required for a video coding algorithm to be adaptive are identified. Finally, the domain of adaptive video coding algorithms is identified.

Chapter 5 outlines the experimental methodology used. First, the experimental design involved in the simulation study, along with the various factors in the experimental study, are presented. Second, the evaluation criterion for the video coding algorithms and the software implementations of the adaptive video coding algorithms under consideration are presented.

Chapter 6 discusses the experimental results with the four algorithms JPEG, EPIC, Fractal and MPEG on 10 Mbps and 100 Mbps networks.

Chapter 7 presents the conclusions and future research.

Chapter 2

Background

This chapter presents background material on the ongoing research in multimedia communications. The traffic characteristics and the service requirements of multimedia applications are also discussed.

2.1 Multimedia Communication

Multimedia communication refers to the storage, retrieval, representation and dissemination of information in multiple media, such as, text, voice, graphics, images, audio and video [20, 47].

The development of high speed networks such as B-ISDN has made available the bandwidth needed for multimedia applications. Considerable research has been carried out to develop multimedia applications for industry [13, 56, 61], science [31, 37, 44] and the general public [57, 59]. For example, in home shopping networks, instead of displaying dull pictures and a faceless voice, a multimedia system could display the product and a video clip of the person explaining the product. Another application would be in a large corporation with its offices distributed all over the country. With a multimedia application available, it would be possible for the managing director to carry on a multipoint video teleconference with all the regional vice presidents regarding the marketing strategy for the next quarter. Last, but not least, is the use of multimedia applications in the field of medicine [51]. For example, a doctor in Toronto needs the opinion of a neurologist in California concerning an abnormal growth on the patient's brain as seen in a CAT scan. The consultation could be accomplished very easily through a video teleconference between the doctors with both viewing the CAT scan simultaneously [70].

Having discussed some of the applications of multimedia systems, there is a need to identify the communication media on which these applications are feasible. Even though high speed networks, such as B-ISDN, are being projected as the target networks for multimedia applications [12, 41, 48], it would be presumptuous to rule out the suitability of local area networks for multimedia applications. A fully interactive video teleconferencing system has been developed on an Ethernet LAN [61] and a significant amount of research in the area of multimedia applications on LAN's exists in the literature [3, 44, 55, 82].

Thus, multimedia applications will enhance distance communication and will be potentially incorporated in all facets of everyday life in the near future.

2.2 Traffic Characteristics

Multimedia applications support a diverse mix of traffic, the most important being video, voice and data. These traffic streams may have bandwidth requirements ranging from a few kilobits per second to several gigabits per second. In addition, each of these traffic streams may vary from being continuous to being highly bursty [14].

Video applications can be classified as real-time and non-real-time video. Video teleconferencing is an example of real-time video and image transfer is an example of non-real-time video application. Real-time video is bursty [72, 79], with the burstiness (ratio of the peak bit rate to the mean bit rate) depending on the scene contents and the changes in the video stream. Real-time video has very strict delivery time constraints (30 frames/sec for NTSC standard video). On the other hand, non-real-time video applications deliver images to the user on request. Thus, there are long periods of silence followed by an active period in which the image is transferred. Even though there might be a significant change in the successive image contents, very few images are transmitted and so non-real-time video streams have different burst characteristics than real-time video.

Voice traffic has a very low bit rate (1 Kbps-100 Kbps) [79]. A typical conversation has a period where the person speaks followed by long periods of silence (person listens). Thus, voice traffic is somewhat bursty.

The characteristics of data traffic depend to a large extent on the type of application. Variable bit rate data is highly bursty [22], whereas low speed data applications have very low burstiness levels.

2.3 Service Requirements

Most multimedia applications offer three main types of services: video, voice and data. Each of these services requires certain minimum guarantees of the underlying network in terms of delay, loss, jitter, bandwidth and synchronization. The minimum requirement of these services is called the *service requirement* and the service offered by the underlying network is called the *quality of service* (QOS).

Figure 2.1 shows the service requirements for some of the components of a multimedia application on broadband networks. The graph plots the maximum cell delay variation and the probability of cell loss which may be expected for different broadband services.

The service requirements of the different components of a multimedia application are very different. Video and voice traffic are more delay sensitive than data traffic. The service requirements for data traffic, in terms of the delay, depend on the data application. For example interactive data transfer applications like *talk* require lower delay guarantees than non-interactive applications like file transfer. Typical limits of acceptable delay are 1 to 30 msec for interactive video and voice, 10 to 500 msecs for interactive data and 50 msec to 15 secs for file transfer [26]. On the other hand, video traffic can tolerate more bit errors than data traffic, and voice traffic can tolerate the most bit errors. Typical limits of acceptable bit errors on a broadband network are 10^{-10} to 10^{-12} for data, 10^{-8} to 10^{-12} for still images and 10^{-7} to 10^{-12} for moving pictures [6].

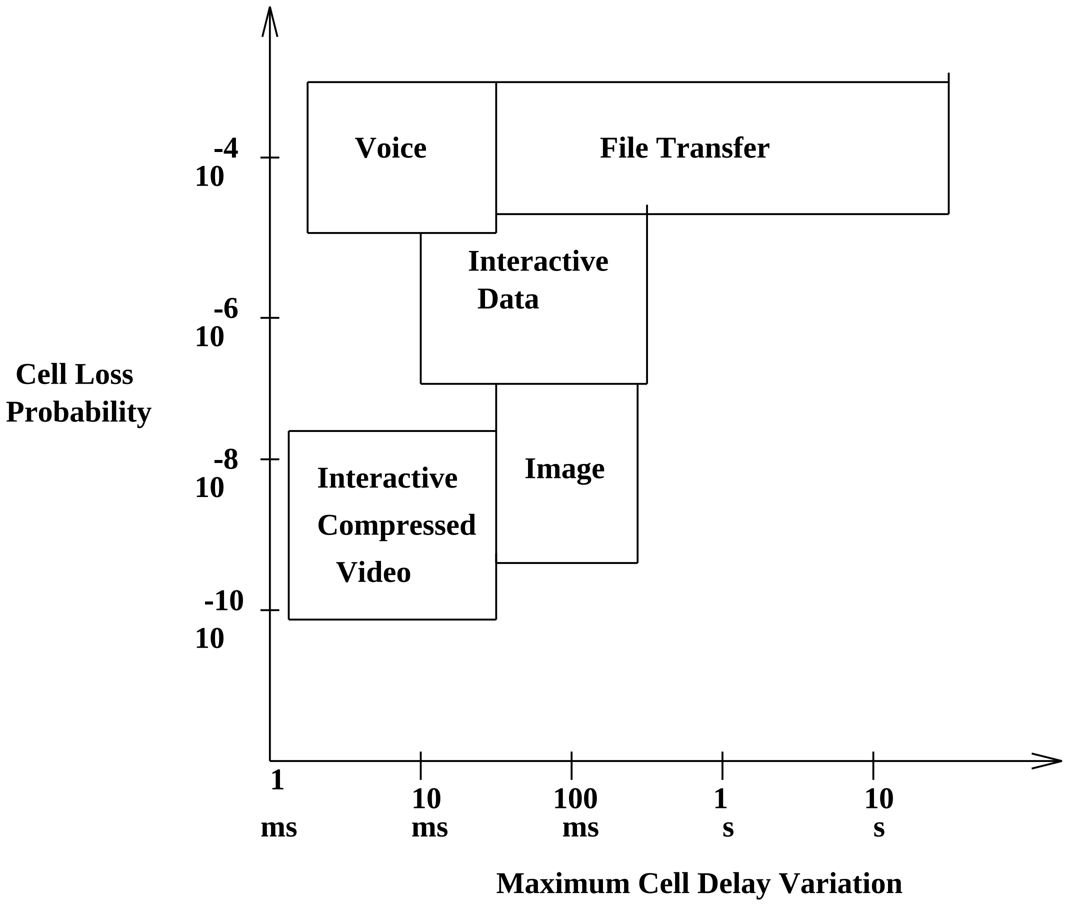


Figure 2.1: Approximate QOS Requirements of Broadband Applications [22]

Another important service requirement of multimedia applications is synchronization. Synchronization in a multimedia environment can be divided into cross-synchronization and synchronization of information retrieval and storage (e.g., combined access and display of separately stored image and text) [42, 49, 67]. Cross-synchronization is used in synchronizing two different media with mutual dependencies (e.g., video and the corresponding audio information in a video clip). Different applications have different synchronization needs and the synchronization time may vary from less than 50 msecs for lip synchronization to several hundred msecs for text and image overlay [43].

Out of the principal media (video, voice and data) in a multimedia application, the video component of a multimedia application has the most stringent service requirements. Thus, there is a need to consider the service requirements of the video component in detail.

- **Bandwidth:** The video component requires very high bandwidth. For example, a sequence of 640×480 pixel per frame, full colour (24 bits), uncompressed NTSC standard video (30 frames per second) [23] requires over 200 Mbps.

- **Delay:** The video component is very sensitive to delay. For example, if the application is trying to display NTSC standard video, the display rate has to be 30 frames/sec or the viewer will find the picture jerky and lacking continuity. Thus, it is essential that the underlying network make available the required packets at the time of playback. Any packet received after the maximum acceptable delay time is assumed to be lost. In broadband networks, the maximum acceptable delay for video packets can vary from 1000 msec for still image transfer to about 50 msec for interactive video [26].

- **Loss:** Even though video can tolerate some amount of loss, the loss must not be large enough to make the image perceptually unsatisfactory to the end-user. Typical acceptable cell (or packet) loss probabilities in video range from 10^{-7} for still images to 10^{-8} for interactive video [28].

- **Jitter:** Jitter is defined as the variations in the delay with respect to the value desired for such delays. For example, if a packet is transmitted at time t_0, the network might guarantee that the packet will arrive at a time $t_0 + x$. In most cases, x is not an exact value, but rather a value plus or minus some small value. The amount by which x can vary is called *jitter* [52]. The variations in delay may be caused by statistical queuing delays introduced at the sender, the receiver and the network, packets that are lost or corrupted by the network, and unsynchronized clocks at the end-user [49, 71]. The video component can tolerate very little jitter. Jitter can be minimized by using prioritized packets and using a lightweight protocol that has no acknowledgments or retransmissions [49].

2.4 Summary

Multimedia applications have components with widely different traffic characteristics and service requirements. Multimedia applications can be loss or delay sensitive, continuous or bursty, and may require high or low bandwidth. In a multimedia application the video component has the most stringent requirements in terms of packet loss, delay, and bandwidth and so is of the greatest interest. If the QOS requirements of video traffic are not met, the video stream as seen by the end-user is jerky and lacks continuity. So it is essential for the network to guarantee the QOS required by video applications.

Chapter 3

Video Coding

This chapter presents some image compression techniques and video coding algorithms that use these image compression techniques. The need for video coding algorithms to adapt to changes in the network load and the means of adapting to changes in the network load are also presented.

3.1 Compression

Compression is a technique that generally reduces the size of an image by coding patterns and redundancies in the image [25]. Compression techniques can be classified as lossless or lossy. *Lossless compression* algorithms do not lose any information during the compression process, so the decompressed image looks exactly like the original. On the other hand, *lossy compression* tries to reduce the size of the image by eliminating redundancies and taking advantage of the non-linearity of human vision [4]. Thus, some data is lost during compression. The amount of data lost depends on the complexity of the original image and the compression ratio [25]. Lossy compression algorithms provide a much higher level of compression (50:1) than lossless compression algorithms (5:1) [77]. But lossless compression has lower compression times than lossy compression.

Compression consists of two distinct components, modeling and coding. The modeling process decides on the symbols to output for a particular input. The coding process then produces the appropriate code based on the probabilities output by the modeling process. Figure 3.1 shows the entire compression process. Throughout this research the term "video coding algorithm" refers to the entire compression process.

The compression process can be symmetric or asymmetric [64, 66]. A compression process is said to be *symmetric* if it takes the same amount of time to encode the image as it does to decode the image. An *asymmetric* compression process, on the other hand, spends more time compressing the image in order to attain lower decompression times. Asymmetric compression processes are advantageous when the images being transmitted are precompressed and stored. Since the decompression is very fast the image can be decoded very quickly.

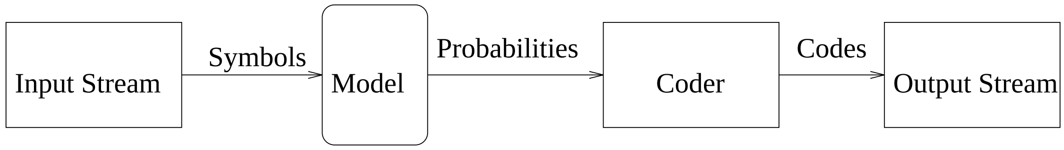


Figure 3.1: The Compression Process [68]

3.2 Image Compression Techniques

The most commonly used image compression techniques are *spatial domain*, *transform* and *quantization*. One or more of these methods may also be combined to give *hybrid* techniques [17].

Spatial Domain Coding: This coding technique compresses images by considering them in the spatial domain. Spatial domain coding techniques look at neighbouring regions, of the region under consideration, in order to compress it. One of the spatial domain coding methods is *pulse code modulation* (PCM)[16]. In a PCM scheme, the incoming image (analog values) is sampled and quantized. Thus, the coded signal is nothing more than the digital representation of the input image. Another method is *differential pulse code modulation* (DPCM). In a DPCM scheme, the preceding quantized pixel value is used to predict the value of the current pixel. The prediction error is then rounded off to the nearest integer and encoded. At the receiver, the received code is decoded and the prediction error is added to the integer prediction (formed as in coding) to reproduce the original pixel [54]. Spatial domain coding methods are very simple and fast but can provide only very low compression ratios (4:1) [16].

Transform Coding: Transform coding compresses an image file in two steps. First, a mathematical transform is applied to the image file and a set of coefficients for the transform are obtained [2, 45]. Then those coefficients that represent redundant information are discarded and only the coefficients representing significant information are encoded [16]. Transform coding is suitable for compressing both still images and real-time video [16]. Transform coding techniques have high processing requirements but are able to provide higher levels of compression than predictive techniques.

The transform coding techniques compress the image as follows [54]. The image is first divided into smaller size blocks. Each of these blocks is then transformed using a transformation matrix. The coefficients of the transformation matrix are then encoded. The transforms most often considered for image coding are discrete fourier (DFT) and discrete cosine (DCT).

Hybrid Coding: This coding technique combines *transform* and *spatial domain* (DPCM) coding techniques [16, 27]. The hybrid coding technique works by first transform coding the image. Each transformation coefficient is then DPCM coded and transmitted. This technique combines the simplicity of DPCM and robustness of transform coding and results in a technique that is less sensitive to transmission errors than DPCM, but is more sensitive to transmission errors than transform coding. Hybrid coders for real-time image compression have been developed [16].

Quantization: Quantization refers to the digitization of the image. The quantizer takes in an analog video signal and transforms it into the corresponding digital representation. The digital representation is then compressed by applying various digital processing techniques. The quantization step works as follows: the amplitude of the input video signal is compared to a set of precomputed discrete levels. An input signal value falling between two discrete values is then quantized to a value within the quantization band. Each of these quantized samples is assigned a binary code and the digital signal is made up of a sequence of these binary codes.

3.3 Need for Video Coding Algorithms

Video applications have very high bandwidth requirements and some networks may not be able to support these video applications unless the video stream is compressed. Because of the stringent service requirements (delay, loss, jitter) of video traffic, care must be taken to see that video coding algorithms do not adversely affect the service requirements of video traffic. For example, since video traffic is delay sensitive, the video coding algorithm should not introduce more delay than the maximum acceptable delay.

The advantages of video coding algorithms are as follows:

- **Reduced Bandwidth Requirements:** Video coding algorithms reduce the number of bits to be transferred, therefore the amount of bandwidth required for the application is reduced. Compressed NTSC standard video requires 1.5 Mbps instead of 200 Mbps and compressed HDTV requires 100 Mbps instead of 1.3 Gbps [83].

- **Reduced Queueing Delay:** The reduction in the number of bits reduces the queueing delays at the end users, thus reducing delay and jitter. Also, since the number of bits to be transmitted is reduced, the transfer time is also reduced. These reductions in time help in reducing the end-to-end delay.

- **Reduced Storage Requirements:** The reduction in storage is probably one of the most important advantages of video coding algorithms, especially for transmission of stored video and images. One uncompressed NTSC standard frame (640×480 pixels, with 24 bits of colour information per pixel) requires 0.9 Megabytes of storage. A one minute NTSC standard video clip at a display rate of 30 frames/sec requires 1.6 Gigabytes of storage. Even with a compression ratio of 10:1, the storage problem would be much more manageable.

Video coding algorithms have the following disadvantages. First, there is the processing overhead of compressing and decompressing the image. Second, most of the video coding algorithms are lossy (i.e., some information is lost during coding). Thus, some information is lost even before transmission begins. Care must be taken to see that the processing overheads at the end users do not delay the video traffic too much and that the loss is not significant enough to affect the end user's satisfaction with the received image.

3.4 Video Coding Algorithms

Various coding schemes are available for video [15, 21, 53, 76]. This section presents some of the video coding algorithms based on the image compression techniques discussed in the previous section.

Video coding algorithms can be classified as *intraframe* and *interframe* coding algorithms. Intraframe coding algorithms encode an image based on the redundant information present within the frame. The coding algorithms look for areas in the image that resemble other parts, or for information that can be dropped without significantly affecting image quality. The most widely used intraframe coding algorithms are Joint Photographic Experts Group algorithm (JPEG) [77], recursive block coding, and Efficient Pyramid Image Coding (EPIC) [1]. Intrafame coding algorithms typically have compression ratios in the range 2:1 to 50:1 [4].

An interframe coding algorithm looks for redundancies between successive frames. Since a stream of video consists of successive frames with very little information change from one frame to the next, interframe coding algorithms are ideal for video. Since interframe coding algorithms encode the difference between successive frames, they have higher compression ratios than intraframe coding algorithms. Typical compression ratios for interframe coding are 50:1 to 200:1 [4]. An interframe coding algorithm is used in the MPEG algorithm [40] and the H.261 standard [9].

A few of the intraframe video coding algorithms present in the literature that are evaluated in this research are JPEG [77], EPIC (sub-band coding) [1], and fractal coding [8]. The only interframe coding algorithms that is evaluated in this research is the MPEG [40] video coding algorithm.

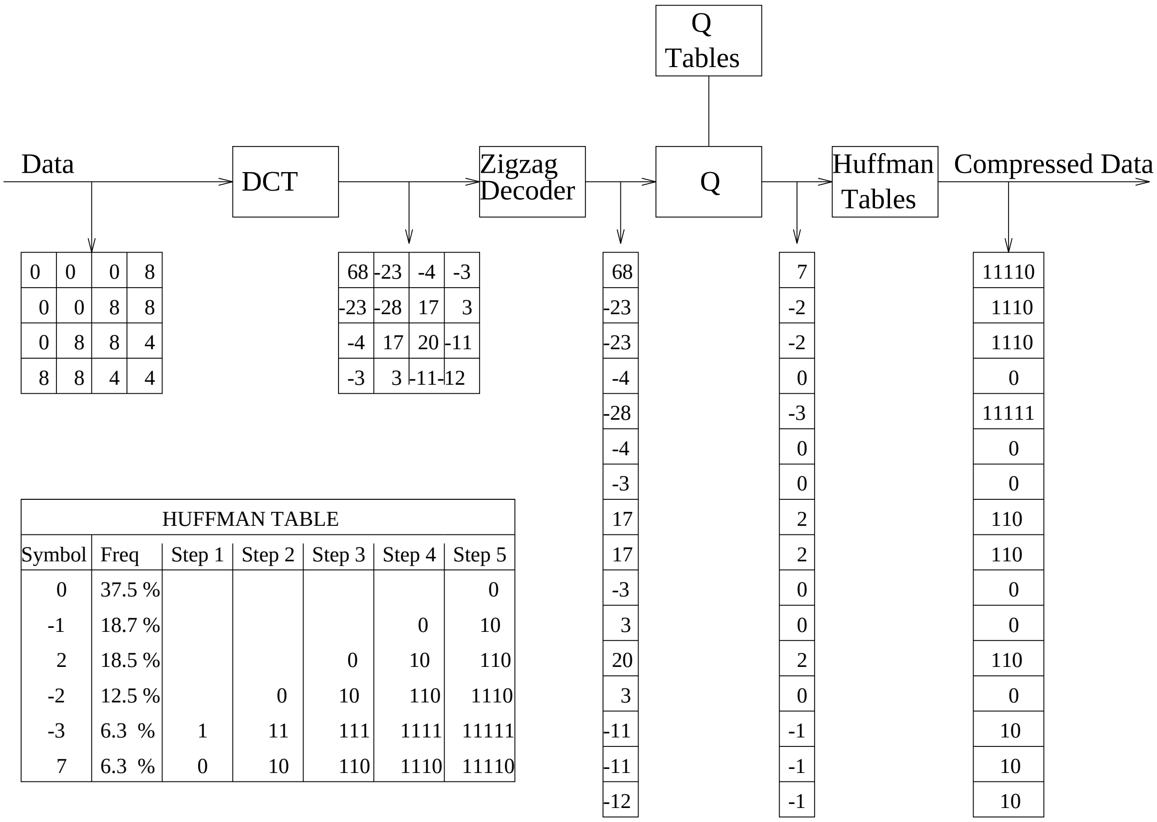
3.4.1 JPEG

The Joint Photographic Experts Group proposed a standard for still picture compression. A JPEG video coder [4, 77], based on this standard, was developed by a research team under the auspices of the International Standard Organization (ISO). The JPEG video coding algorithm provides good compression ratios (20:1) while maintaining high picture quality [77].

The JPEG algorithm is an example of transform-based image coding. Figure 3.2 shows a simple JPEG encoder. The JPEG encoder works as follows: First, the JPEG algorithm breaks the input image up into non-overlapping blocks of 8×8 pixels. Second, each of these blocks is transformed using a discrete cosine transform (DCT), resulting in a transformed matrix with 64 coefficients. These 64 coefficients represent the frequency contents of the block. The coefficients give the relative strength of signal terms increasing from left to right and top to bottom. Third, the coefficients of the DCT are quantized. The quantization step varies with frequency and higher frequency components matter less than the lower frequency ones. So higher frequency coefficients may be quantized with a large step size thereby reducing the number of bits required to represent the coefficients. Fourth, the coefficients are reordered in a one-dimensional array by reading the two-dimensional array in a zigzag route. This way the quantized coefficients can be arranged approximately in order of ascending frequencies. The last step involves encoding the coefficients using Huffman coding [16]. Huffman coding is a statistical encoding scheme that takes advantage of the non-uniform distribution of the frequency components so that short representations are used for components that occur often, and longer representations are used for components that are used less often. The JPEG decoding process is the reverse of the encoding process.

In the JPEG encoder shown in Figure 3.2, a 4x4 block is considered for simplicity.

The decoder is not shown since decoding is the reverse process of encoding. The inputs to the decoder are the Huffman codes. The Huffman codes are formed at the JPEG encoder as shown in Figure 3.2. The frequency components, listed in descending order of relative frequency of occurence, are assigned a sequence of bits to form codes as follows. The two groups that occur least often are selected and a 0 bit is assigned to one and a 1 bit is assigned to the other. Next the two groups, -3 and 7, are treated as one group and are assigned a 1 bit in the second bit place and a 0 bit is assigned to -2 in the first bit place. This process is repeated until Huffman codes for all the components are generated. Decoding these codes gives the quantized coefficients. The coefficients of the transform are obtained by inverse quantization. An inverse zigzag operation on the linear array followed by inverse DCT gives the reconstructed block.



HUFFMAN TABLE						
Symbol	Freq	Step 1	Step 2	Step 3	Step 4	Step 5
0	37.5 %					0
-1	18.7 %				0	10
2	18.5 %			0	10	110
-2	12.5 %		0	10	110	1110
-3	6.3 %	1	11	111	1111	11111
7	6.3 %	0	10	110	1110	11110

Figure 3.2: Elements of a JPEG Encoder [3]

3.4.2 EPIC

The EPIC video coding algorithm is based on a sub-band decomposition of the input image and Huffman coding. The EPIC encoder and decoder (codec) is shown in Figure 3.3.

Sub-band coding [54] is a form of transform coding. Sub-band coding breaks up an image into its component frequencies by employing bandpass filters. If a bandpass filter is designed for a particular frequency band, then the bandpass filter allows the specified frequencies to pass through and removes all other frequencies [54]. The image, after passing through the bandpass filters, is broken up into bands of frequency components. For example, if two bandpass filters were used then there would be two frequency components, each containing half of the information in the original image. Both of these components are then encoded using transform coding techniques like

DCT or DPCM. At the receiver, the signals are decoded and the decoded components are interpolated to give the reconstructed image.

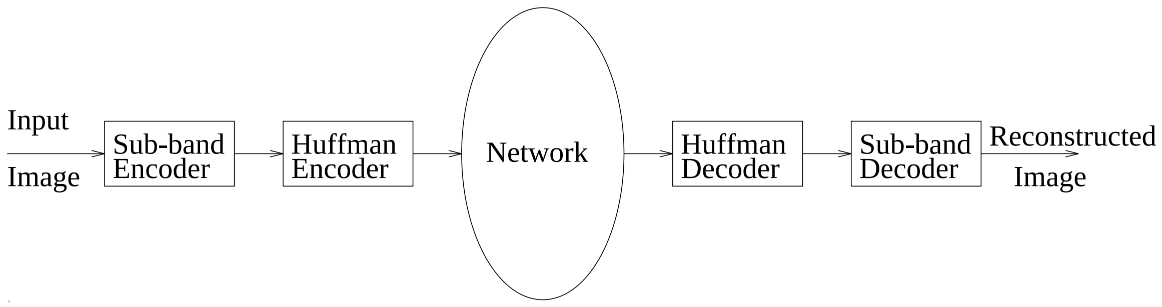


Figure 3.3: Elements of an EPIC Encoder and Decoder [2]

Figure 3.4 illustrates the sub-band encoder and decoder (codec).

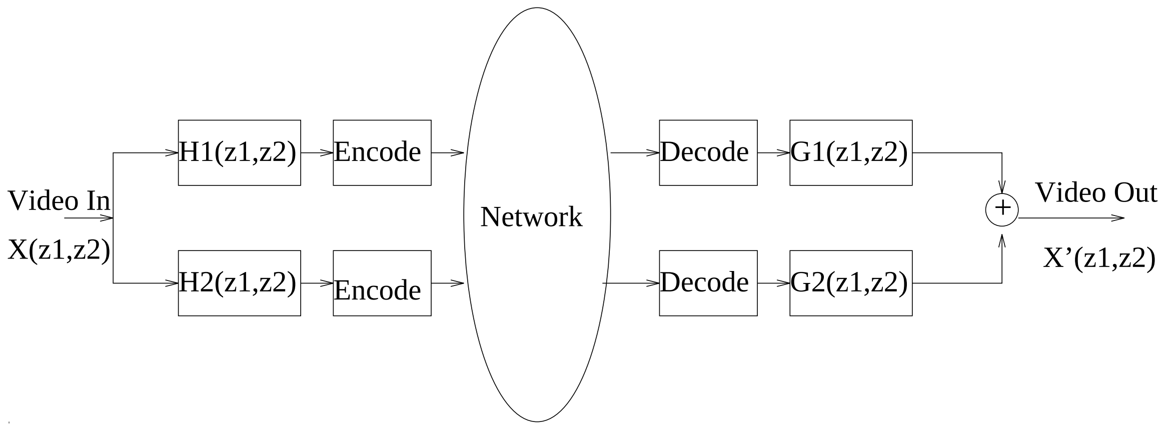


Figure 3.4: Sub-band Encoder and Decoder [68]

The image is broken up into two frequency components by the bandpass filters H1(z1,z2) and H2(z1,z2). These frequency components are then encoded and transmitted. At the receiver the signals are decoded and interpolated to give the decoded image. Sub-band coding has high processing requirements and is very complex. This is because the image first has to be broken up into frequency components and each of these frequency components has to be transformed independently. The compression ratio achieved using sub-band coding depends on the image contents. Significant research has been done with sub-band coding [34, 35, 65]. Adelson *et al.* developed a sub-band pyramid coder for still images [1] and Karlsson demonstrated the use of sub-band coding in video conferencing applications [33].

The EPIC algorithm breaks the image up into frequency bands using bandpass filters and each of these frequencies is transformed using a discrete cosine transform (DCT). The transformed frequency components are then Huffman encoded to give the Huffman codes. The Huffman codes are the inputs to the decoder and the EPIC decoding process is the reverse of the encoding process.

3.4.3 Fractal Compression

The fractal compression method [8] is based on the assumption that every part of an image looks like some other part of an image. The fractal coding algorithm [8] is a type of transform coding technique.

The fractal coding technique is shown in Figure 3.5 and works as follows. The image is broken up into small non-overlapping regions called domains. The image is also broken up into larger overlapping regions called ranges. For each of the domain regions, the algorithm tries to find a range larger than the domain which when transformed closely matches the domain region. The range region can be shrunk, rotated or its intensity and contrast changed to match the domain. The coefficients of the transform and the domain regions chosen are stored in a Fractal Image Format (FIF) file. The decoding process is started with an arbitrary image and the FIF file. The reverse of the encoding process is applied to the image in order to obtain the reconstructed image.

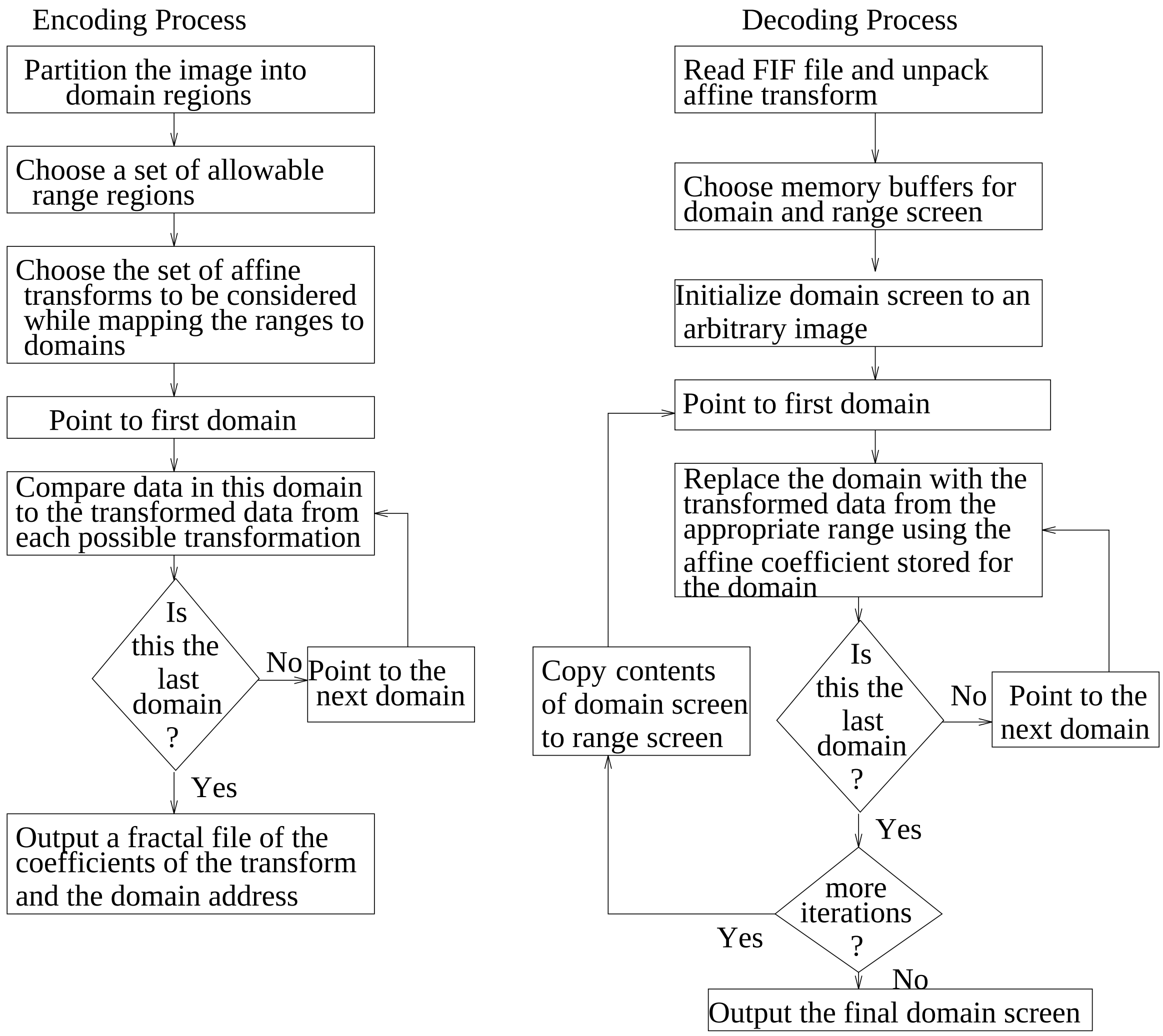


Figure 3.5: Fractal Coding-Decoding Processes [5]

The fractal coding method has the following advantages. First, the decoded image is resolution independent [8, 18]. What this means is that a FIF file of a 64×64 image can be used to reconstruct an image of any size. The resolution independence property results in the high compression ratios quoted for fractal compression (as high as 10,000:1 [8]). Second, the FIF file does not have to increase in proportion to the size of the reconstructed image because fractally coded images are resolution independent [5]. Third, compression ratios can be improved by taking more time to compress the image, without adding to the decompression time or decreasing the quality of the image [5], because the compression process can spend more time in looking for better mappings for the ranges. The only disadvantage to fractal coding is its complexity and long compression times (it takes 45 minutes to compress a single 128×128 pixel grayscale image on a DEC 3000).

Figure 3.6 shows how the fractal decoding algorithm successively adds details to the reconstructed image with each iteration. Figure 3.6 (a) shows the starting image for the decoding process. The starting image was assumed to be blank. The fractal decoding process was applied to this blank image and the resultant image was used as the input to the fractal decoding process for the second iteration. This process was repeated with the output image of an iteration being used as the input image for the next iteration. Figure 3.6 (b-d) show the output images of the fractal decoding process after the third, fifth and tenth iteration.

3.4.4 MPEG

The Motion Picture Expert Group working under the auspices of the International Standard Organization (ISO) and the International Electro-Technical Commision (IEC) proposed a standard for full motion video. An algorithm based on the standard was developed by a research team and is called the MPEG video coding algorithm [40]. MPEG provides high levels of compression (50:1-200:1)[4] by using both interframe and intraframe coding technques.

The MPEG group proposed a standard for both the video and audio part of a video stream. Since this study is concerned with video coding, only the video part of the standard is presented in this section.

Figure 3.6: An Example of the Fractal Decoding Process (a) Initial Image (b) After Third Iteration (c) After Fifth Iteration (d) After Tenth Iteration (Reconstructed Image).

MPEG considers a video stream as a set of still images with strict display time constraints (30 frames or images per second). The MPEG encoder expects the input images to be in a format known as YUV format. The YUV format is an efficient way of representing the image. Each pixel in an image is represented by a YUV triplet, where the Y value represents the luminance (black and white) level and the U/V values represent the chrominance (colour) level. The advantage of using the YUV format is that the U and V components can be encoded at a lower resolution since the human eye is less sensitive to the chrominance component than to the luminance component.

The basic concept of MPEG is to predict motion from frame to frame in the temporal direction, and then to use DCT's (discrete cosine transforms) to organize the redundancy in the spatial directions. The DCT's are done on 8x8 blocks, and the motion prediction is done in the luminance (Y) channel on 16x16 blocks. During encoding, a match for the 16×16 block to be encoded is searched for in a previous or future frame. The DCT coefficients (of either the actual data, or the difference between the block to be encoded and the close match) are quantized. The DCT coefficients, the motion vectors, and the quantization parameters are then Huffman coded using fixed tables. Also, the motion vectors and the DCT components are DPCM coded.

MPEG considers three types of frames: the Intrapicture frames (I-frames), Predicted frames (P-frames) and Interpolated frames (B-frames for bidirectional prediction). The I-frames serve as random access points into the video stream and also as a reference to encode and decode P-frames and B-frames. I-frames provide the least compression of the three frames and are usually encoded using the hybrid coding technique followed in JPEG.

The P-frames are coded with reference to a past frame (I or P) and will generally be used as a reference in the future. P-frames are compressed using interframe compression method (motion compensation) and provide compression ratios that are higher than the compression ratio of I-frames but lower than the compression ratio of B-frames. The B-frames are coded with reference to a past and a future frame. Thus B-frames require both a past and a future reference and can never be used as a reference. The B-frames are also encoded using motion compensation. The relationship between the three frame types is shown is Figure 3.7.

Motion compensation is used in coding B-frames and provides very high compression. Motion compensation is made up of a motion-compensated prediction and motion-compensated interpolation. Motion-compensated prediction assumes that the current frame can be modeled as the translation of the frame at some previous time. The motion-compensated interpolation encodes a subsignal with 1/2 or 1/3 of the frame rate of the original video stream and at the decoding end the video stream is obtained by interpolation of the low resolution signal and adding a correction factor.

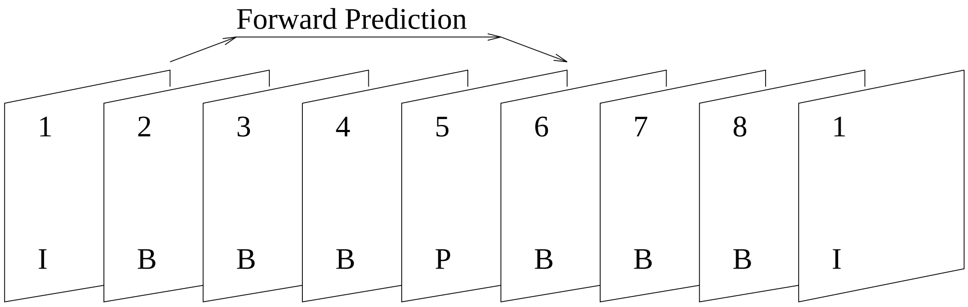


Figure 3.7: Interframe Coding [40]

Bidirectional prediction not only provides very high compression, but also reduces error by averaging over the past and future frames and reduces propagation of prediction error. Even though B-frames provide very high compression, use of too many B-frames between reference frames (P- frames and I-frames) decreases the correlation between the B-frames and reference frames. The MPEG standard proposed the use of a reference frame at about a 1/10th second interval.

The sequence of frames from one I-frame to the next is called a Group Of Pictures (GOP). A video sequence is made of many such GOPs. If some of the GOP information is lost in the network, then the playback is continued from the I-frame of the next GOP instead of waiting for the retransmission of the lost GOP information.

3.5 Need for Adaptivity

A video coding algorithm is said to be adaptive if it can vary the encoding of a video stream (and thus its bandwidth requirements) dynamically based on the amount of bandwidth available in the network. This means that, regardless of the network load, the video coding algorithm should be able to compress the image such that a perceptually satisfying image is received by the end-user in a reasonable amount of time. Thus, the coding algorithm must be able to highly compress the image when the network is highly loaded and be able to compress the image less when the network is not loaded.

In addition to making better use of the available bandwidth on the network, adaptivity may be able to make video coding algorithms more robust: at heavy network loads the adaptive video coding algorithm compresses the image more, thereby reducing the time taken for the image to reach the end user, and at low network loads the adaptive algorithm compresses the image less but since the network is lightly loaded the image transmission time is low. Thus adaptive video coding algorithms can guarantee certain maximum bounds on the image transfer time and minimize the jitter, making them more robust than non-adaptive algorithms.

While studying the need for adaptive video coding algorithms it is essential to understand that in networks with bandwidth reservation for guaranteed QOS there is no need for adaptive video coding algorithms. If peak bit rate allocation is always used then the network can always guarantee the QOS, regardless of the network load. However peak bandwidth allocation is often inefficient, especially for VBR coding. Furthermore, users of high speed networks would rather pay for the bandwidth that they actually use than the bandwidth that they reserve. Thus one of the basic assumptions while studying adaptive video coding algorithms is that the network does not provide simple peak bandwidth reservation for guaranteed QOS. Instead, the network offers Available Bit Rate (ABR) services.

With this fundamental assumption in mind, then, video coding algorithms need to be adaptive for the following reasons.

- When the network is heavily loaded, the primary purpose of the video coding algorithm is to get the image to the receiver in at least a reasonable amount of time. The user might be willing to settle for an image of lower quality (e.g., black and white instead of colour) if it is received within a reasonable amount of time. In such a situation, the video coding algorithm might compress the image further, at a sacrifice of some image quality.

- For intermediate levels of network load, depending on the level of congestion, the video coding algorithm has to decide whether the end-user desires a perceptually inferior image in less time or a perceptually superior image in more time. Thus, depending on the network load, the video coding algorithm has to compress the image adaptively so that as the network congestion level increases the compression of the image also increases.

- At a state of low network load, the primary purpose of the video coding algorithm is to get an image of very high quality to the receiver. The receiver will only settle for an image of very high quality. So the video coding algorithm might provide very low compression. A low compression ratio is satisfactory in this condition because the bandwidth required is available on the network.

3.6 Means of Adapting to Changes in Network Load

There are several ways in which video coding algorithms can adapt to changes in the network load.

- **Reduce Colour Information**: The video coding algorithm can reduce the number of bits used to represent the colour information of each pixel when the network load increases. Thus, if the colour information is dropped from 24 bits (full colour) to 8 bits (grayscale) for an image, then a bandwidth saving of a factor of 3 is obtained, but the decoded image will be less perceptually satisfying than the full colour image.

- **Reduce Image Size**: The video coding algorithm can reduce the size of the image when the network load increases. Thus, an image of 640×480 pixels could be reduced to 160×120 pixels, obtaining a bandwidth saving of a factor of 16, but the decoded image will be less perceptually satisfying than the full size image.

- **Reduce Frame Rate**: The display rate of a video stream could be reduced under conditions of high load. For example, if the frame rate of a video stream was reduced from 30 frames per second to 15 frames per second, then a bandwidth saving of a factor of 2 is obtained. The received image sequence may lack continuity and be jerky.

- **Change Algorithm Parameters**: The parameters of the video coding algorithm under consideration could be changed depending on the network load to provide higher or lower compressions. For example, the number of quantization levels in the JPEG algorithm could be changed. Each setting of the parameter results in a different compression ratio (level of the compression algorithm) for the compression algorithm, and may change the image quality. Thus when the network is lighty loaded the parameters of the video coding algorithm could be set to provide low compression ratios (and a good quality image). But as the network load increases the parameter setting could be changed dynamically according to the network load level in order to provide increasing compression ratios (and decreasing image quality).

3.7 Summary

Video applications have very high bandwidth requirements. Video coding algorithms are needed to reduce the bandwidth and storage requirements of video applications. If the video coding algorithm is not adaptive, then the compressed image will take a long time to be transfered at high loads and the perceptual quality may be reduced unnecessarily under low network loads, resulting in low user satisfaction. Thus, the video coding algorithm needs to adapt the encoding of a video stream and thus its bandwidth requirements dynamically based on the amount of bandwidth available in the network.

Chapter 4

Analytical Study

The main objective of adaptive video coding algorithms is to provide the end-user with an image of good perceptual quality in a small amount of time regardless of the network load. Thus there is a need to minimize the time it takes for the image to reach the end-user. This minimization of the transfer time is called the image-transfer problem and a mathematical formulation of the end-to-end transfer time will help to understand and address the image-transfer problem.

This chapter is organized as follows. First, the derivation of a simple mathematical representation of the image-transfer problem is described. Second, a few canonical cases of video coding algorithms are studied to determine if there is any advantage to adaptivity. Third, the type of networks that can support adaptive video coding algorithms are identified.

4.1 Assumptions

Several assumptions have been made in order to arrive at the representation of the image-transfer problem for the adaptive video coding algorithms.

1. The network is a shared resource whose load may change unpredictably with time (i.e., there is traffic from other users on the network).

2. A lightweight transfer protocol with no acknowledgments and no retransmissions (e.g., ATM) is assumed so that user-level performance is limited only by the capacity of the network. That is, protocol overheads are ignored.

3. It takes negligible time to display an image at the receiver once the image is decompressed.

4. All packets transmitted on the network are 128 bytes in size.

5. For the intraframe compression algorithms, a single image transmitted multiple times in succession provides a reasonable approximation of a video sequence. A single image was used instead of a video clip to minimize storage requirements since a very large amount of storage is required for full motion video. For example, one second of full motion uncompressed NTSC standard video (30 frames/sec) with 640×480 pixels/frame and 24 bits of colour information per pixel requires about 27 Mbytes of storage. The analytical work evaluated only intraframe compression algorithms and simulated a video sequence by transmitting a single image repeatedly.

4.2 The Image Transfer Problem

The time (T) it takes for the end-user to receive an image depends on the compression time (T_c), the transmission time (T_t) of the compressed image, and the time to decompress the image (T_d). The transmission time is the time from when the sender starts transmitting the first bit of the compressed image to the time the last bit is received by the receiver. By definition, this time includes retransmission, queuing and channel access time on the network. Thus,

$$T = T_c + T_t + T_d \tag{4.1}$$

The compression time (T_c) depends on the compression algorithm used, the size of the uncompressed image S_u, the contents of the uncompressed image, and the speed of the machine used. For simplicity T_c is represented by a fixed value F_c depending only on the compression algorithm. That is, the uncompressed image size S_u is assumed to be constant, and since a non-changing sequence of images is assumed, the effects of image contents are ignored. Machine speed is also ignored.

The decompression time (T_d) depends on the compression algorithm used, the size of the compressed image S_c, the contents of the decompressed image and the speed of the machine used. For the sake of simplicity, T_d is represented by a value F_d depending only on the compression algorithm used. Image size, image contents and machine speed are again ignored.

The expected transmission time T_t depends on three main factors: the speed of the network (B bits per second), the utilization ($0 \leq \rho \leq 1$) of the network, and the size of the compressed image (S_c bits).
Mathematically,

$$T_t = \frac{S_c}{(1-\rho)B} \tag{4.2}$$

For example, a compressed image of 20 Kilobytes transmitted over a 10 Mbps network link that has a network utilization of 0.3 requires

$$T_t = \frac{163840}{(1-0.3) \times 10 \times 10^6} = 0.0234 \; secs \tag{4.3}$$

In order to validate the mathematical model of the transmission time, a simple simulation was conducted to compare the transmission time for an image of size 128×128 pixels and 8 bits of colour information, compressed by the JPEG algorithm and transmitted on a 10 Mbps network, for the mathematical model and a simulation. The mathematical model used Equation 4.2. The simulation model used empirically obtained compression information for the JPEG algorithm on a single image. The (simulated) transmission time for the image on a 10 Mbps network was determined for different levels of (simulated) cross traffic on the network. The results are plotted in Figure 4.1. From Figure 4.1 it is observed that the transmission times for the two models were in close agreement. Results for the simulation model are consistently lower than the mathematical model, but only by about 0.0002 secs ($< 5\%$).

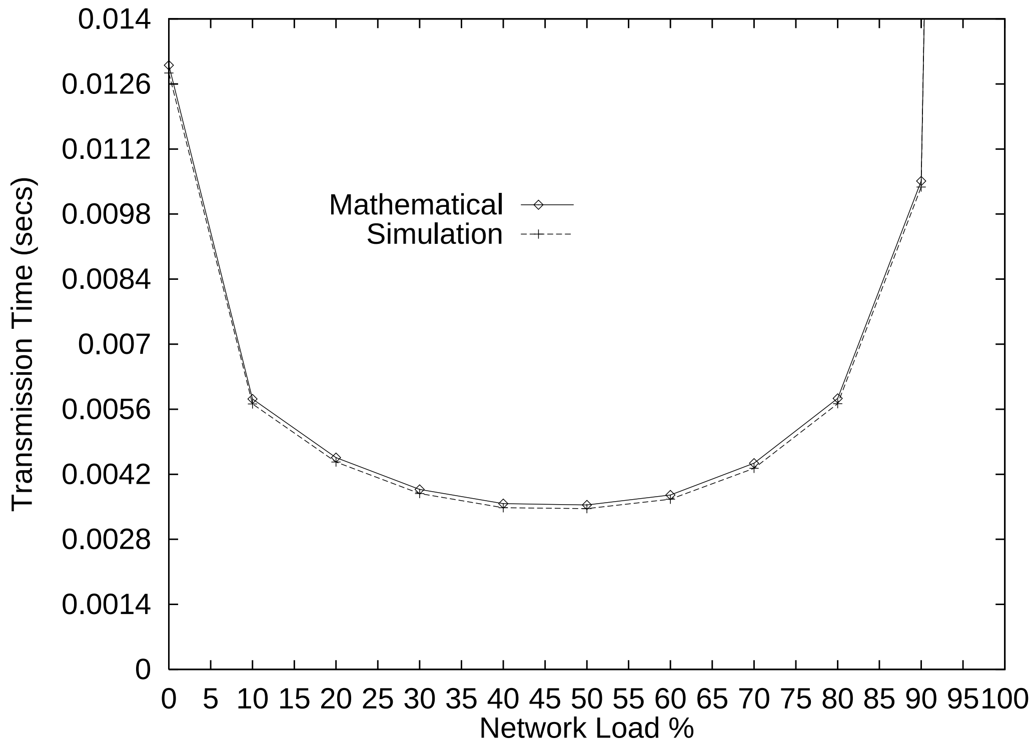


Figure 4.1: Transmission Time Curves for the Mathematical Model and Simulation Model of the Adaptive JPEG Algorithm on a 10 Mbps Network

The size of the compressed image (S_c) depends on the size of the original image (S_u), and the compression ratio (C_r) offered by the compression algorithm, where the compression ratio is the ratio of the size of the original image to the size of the compressed image. Mathematically,

$$S_c = \frac{S_u}{C_r} \tag{4.4}$$

Therefore Equation 4.1 becomes

$$T = \frac{S_u}{(1-\rho)BC_r} + F_c + F_d \tag{4.5}$$

For example, the JPEG algorithm running on a DEC 3000, compressing an image with 128×128 pixels and 8 bits of colour information per pixel at a compression ratio of 10:1 has $F_c = 0.75$ secs, $F_d = 0.50$ secs (based on actual measurements). If the image is transmitted on a 10 Mbps Ethernet LAN at a utilization of 0.3, the predicted end-to-end transfer time is

$$T = \frac{131072}{(1-0.3) \times 10 \times 10^6 \times 10} + 0.75 + 0.5 = 1.252 \; secs. \tag{4.6}$$

4.3 A Study of Hypothetical Cases of Video Coding Algorithms

This section presents an analytical study of a few hypothetical video coding algorithms. A study of some hypothetical video coding algorithms will help in determining if adaptive video coding algorithms are robust to changes in network load and the load conditions under which adaptive video coding algorithms are advantageous.

4.3.1 Generic Cases of Video Coding Algorithms

Figure 4.2 shows the compression profiles of six hypothetical video coding algorithms. The compression profile shows how the compression ratio offered by the video coding algorithm changes as parameter settings are changed. The compression profiles of the generic cases of video coding algorithms represented could be the plots of a simple reduction in the frame rate of a video sequence as the network load is increased, reduction in the size of the image as the network load increases, or changing the algorithm parameters (and thus changing the compression ratio) as the network load changes.

The compression profiles of the hypothetical video coding algorithms were obtained by fitting values to some arbitrarily chosen curves. Each hypothetical algorithm is assumed to have a single generic parameter, which can range from a minimum setting of 0 (offering the least compression) to a maximum of 100 (offering the most compression). The compression profile illustrates the compression behaviour of the generic algorithms as a function of this parameter.

As seen in Figure 4.2, several hypothetical video coding algorithms with different

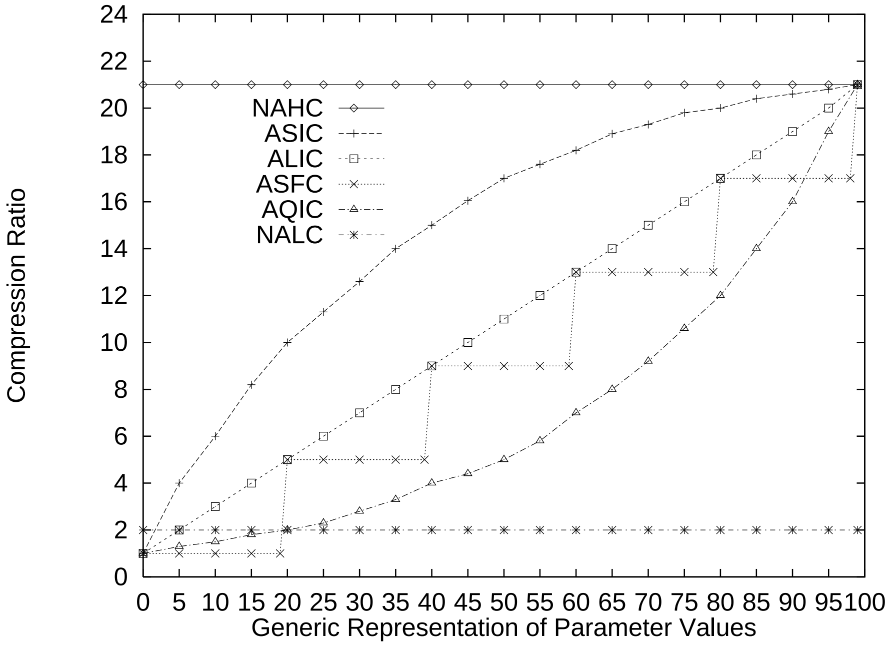


Figure 4.2: Compression Profiles of Hypothetical Video Coding Algorithms

compression profiles are considered. The hypothetical video coding algorithms under consideration are explained as follows.

- Non-Adaptive Low Compression (NALC): This represents a non-adaptive video coding algorithm with a low compression ratio. An example of a video coding algorithm with such a compression profile would be JPEG operating at close to its minimum compression factor. Studying a NALC video coding algorithm is useful in order to show that even though non-adaptive video coding algorithm with low compression ratios provide images of very high quality, adaptive video algorithms may be more robust than non-adaptive video coding algorithms.

- Non-Adaptive High Compression (NAHC): This represents a non-adaptive video coding algorithm with a high compression ratio. An example of a video coding algorithm with such a compression profile would be JPEG operating at its maximum compression ratio. There is a need to study NAHC algorithms in order to evaluate if adaptive video coding algorithms provide any performance gains over non-adaptive video coding algorithm with a high compression ratio.

- Adaptive Linearly Improving Compression (ALIC): This represents an adaptive video compression algorithm with a linear increase in the compression ratio as its parameters are changed. Thus, as the network load increases the video coding algorithm can adapt to provide higher compression ratios with a deterioration in the image quality. An example of a video coding algorithm having a linearly increasing compression profile is the EPIC algorithm (Figure 4.3). Another example would be the linear reduction in frame rate for a video clip as the network load increases. It is interesting to study a video coding algorithm with an ALIC profile in order to see if it outperforms a non-adaptive video coding algorithm operating at the maximum and minimum compression ratios of the ALIC profile and if video coding algorithms with ALIC profiles are suitable for adaptivity.

- Adaptive Step Function Compression (ASFC): This represents a video coding algorithm with a small number of discrete levels in its compression ratios. The algorithm has a step function for its compression profile. An example of a video coding algorithm with an ASFC profile is the fractal video coding algorithm (Figure 4.3). Another example would be the reduction of a colour image to a black and white image or the reduction in the size of the image (through a progression of predetermined sizes) as the parameter settings are changed. A study of a video coding algorithm with an ASFC profile would help in determining if a video coding algorithm with very few levels can be considered for adaptivity. The study will also show how a video coding algorithm with an ASFC profile compares to a video coding algorithm with an ALIC profile having the same average slope.

- Adaptive Quadratically Improving Compression (AQIC): This represents an adaptive video compression algorithm with a quadratic increase in its compression ratio as its parameters are changed. An example of a video coding algorithm with an AQIC profile is the JPEG algorithm (Figure 4.3). Even though a video coding algorithm with an AQIC profile provides lower compression ratios than a video coding algorithm with an ALIC profile (with the same end points and thus the same average slope), it would be interesting to see the relative performance in terms of user satifaction.

- Adaptive Sublinearly Improving Compression (ASIC): This represents an adaptive video compression algorithm with a logarithmic increase in its compression ratio as its parameters are changed. While no specific example of a video coding algorithm with an ASIC profile is known, the ASIC profile is useful as a hypothetical case. The ASIC profile (with the same end points as the ALIC profile) provides higher compression ratios than the ALIC profile over most of the range, even though both have the same average slope. Again, an algorithm with an ASIC profile is of interest because it will help in determining the characteristics that an adaptive video coding algorithm should have.

Figure 4.3 shows the compression profiles of the three intraframe video coding algorithms under consideration in this research (JPEG, EPIC and Fractal). The profiles were determined by different settings of the algorithm parameters (described in Chapter 3) for each value of the network load.

The JPEG algorithm has an almost quadratically increasing compression profile. The EPIC video coding algorithm has a linear increase in its compression ratio as as the network load level increases. The fractal video coding algorithm has a step function for its compression profile. Thus the hypothetical coding algorithms considered are reasonable approximations of real coding algorithms.

4.3.2 Assumptions

Several assumptions were made while studying the hypothetical video coding algorithms. These assumption are as follows:

- An image sequence with images of size 640×480 pixels and 24 bits of colour information per pixel.

- A 10 Mbps network.

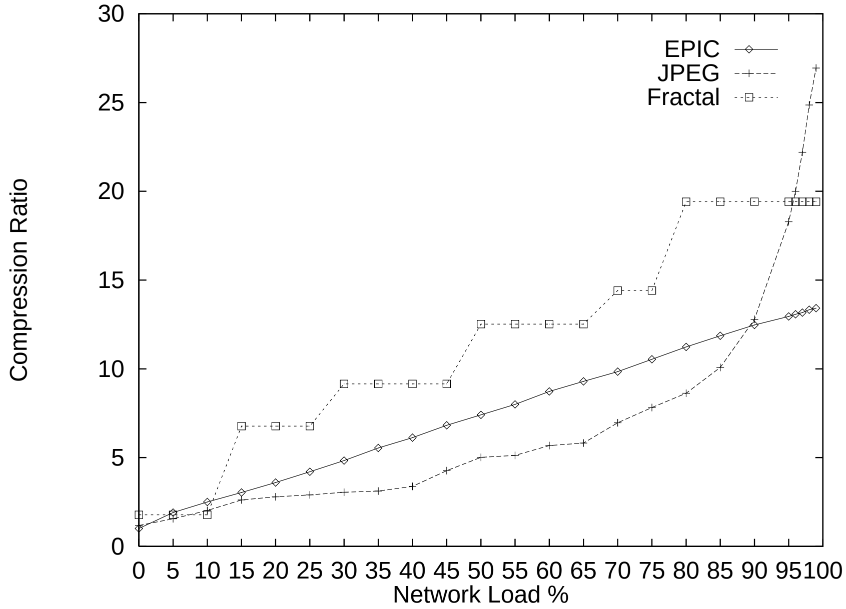


Figure 4.3: Compression Profiles for JPEG, EPIC and Fractal Video Coding Algorithms

- A 1-1 direct mapping between the generic parameter space of the video coding algorithm and the network load (expressed as a percentage). What this means is that at a network load of 10% the parameter setting for the video coding algorithm is 10.

4.3.3 Evaluation of Hypothetical Video Coding Algorithms

Figure 4.4 shows the transmission time curves for the hypothetical video coding algorithms. The values for the transmission time are calculated using the formula for transmission time derived in Section 4.2.

The NALC and NAHC algorithms represent non-adaptive video coding algorithms. From Figure 4.4, it is seen that under low load conditions, the transmission time for the NALC algorithm is comparable to that of other algorithms, but as the load gets to be greater than 20%, the transmission time for the NALC algorithm is much higher than the other algorithms. The transmission time is much higher for the NALC algorithm because the network load has increased, but the size of the compressed image is the same. At network load levels of 80% or greater, the performance

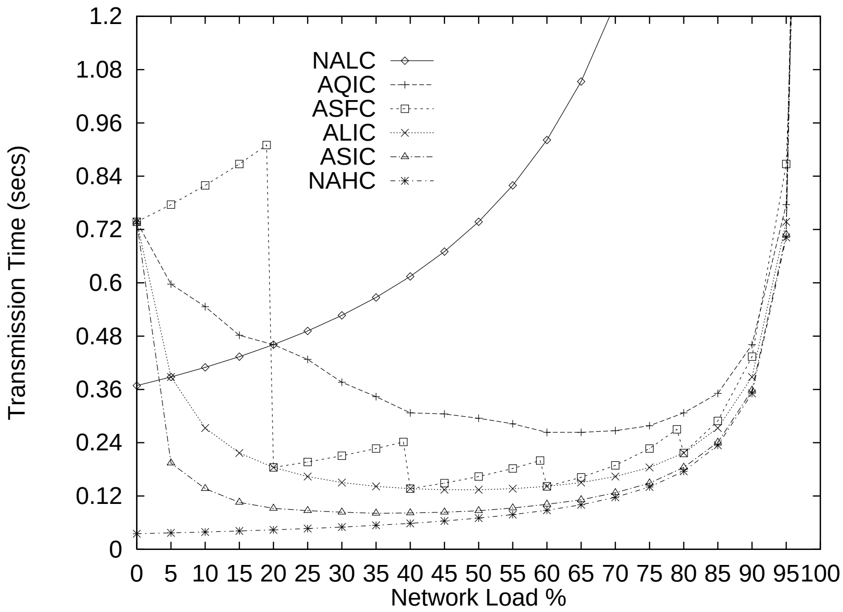


Figure 4.4: Transmission Time Curves for Hypothetical Video Coding Algorithms

of the NALC algorithm is too poor for the algorithm to be considered at these loads. Over the entire range, the NAHC algorithm has the lowest transmission times because it offers the highest compression. Even though the NAHC algorithm has the lowest transmission times, the quality of the received image is the worst (higher compression implies lower image quality) among all the hypothetical algorithms. Thus there is a tradeoff between the image quality and transmission time and the point of balance of the image quality and transmission time depends on the video application and the user. For example, an NAHC algorithm may be advantageous for time critical applications that require low transmission times.

The four adaptive algorithms have similar qualitative behaviour to each other, with a general decrease in transmission time as network load increases from low to moderate, and an increase in transmission time as the load increases from moderate to very high. The decrease in transmission times for low levels of network load is because the algorithm adapts to the increase in the network load and compresses the image more. The ASFC algorithm has the highest transmission times for load levels of 0%-20% because its compression ratio at these loads is lower than the compression ratio of the other algorithms at these loads. For network loads greater than 20%, the ASFC, ALIC and ASIC algorithms have similar values for the transmission times because the compression ratios offered by these algorithms are very similar. The transmission time for AQIC is slightly higher because of its lower compression in this range.

Overall, the adaptive video coding algorithms are are able to guarantee a maximum bound on the image transmission time and minimize jitter over a fairly wide range of network loads. Thus adaptive video coding algorithms are more robust than non-adaptive video coding algorithms. The transmission times for all the algorithms begin to increase very rapidly at loads of 90% or greater. Even though the images are highly compressed, it takes a long time for the sender to get the bandwidth required to transmit the compressed images.

4.4 Domain of Adaptive Video Coding Algorithms

Having identified the general characteristics for a video coding algorithm to be adaptive, it is necessary to determine the network types on which adaptive video coding algorithms are feasible. The network types and speeds required for adaptive video coding algorithms is called the domain of adaptive video coding algorithms. In order to identify the domain of adaptive video coding algorithms, the following network types are considered:

- 100 Kbps: This link speed is representative of some of today's telecommunication links (64 Kbps) and Internet links.
- 1 Mbps: Today's T1 (1.5 Mbps) links, as used in the new CA*net backbone, are examples of networks operating at about 1 Mbps.
- 10 Mbps: An example of a network operating at 10 Mbps is the Ethernet LAN.
- 100 Mbps: These are representative of Fiber Distributed Data Interface (FDDI) networks (100 Mbps) and B-ISDN (150 Mbps).
- 1 Gbps: These are representative of the gigabit networks of the future.

With an NTSC standard image of 640×480 pixels with 24 bits of colour information per pixel, the five networks were evaluated for the NALC and ALIC compression profile from Figure 4.2. The transmission times for the NALC and ALIC compression profiles, obtained using Equation 4.2, are shown in Tables 4.1 and 4.2, respectively. The network utilization column denotes the utilization of the network at the time the image is compressed. Thus the network utilization could vary from 0.0 to 1.0, but the case of 1.0 network utilization is not considered because there would be no available bandwidth for the image to be transmitted.

Table 4.1: Transmission Time in Seconds for Non-Adaptive Video Coding (NALC)

Network Utilization	100 Kbps	1 Mbps	10 Mbps	100 Mbps	1 Gbps
0.0	72.000	7.200	0.720	0.072	0.007
0.1	80.000	8.000	0.800	0.080	0.008
0.2	90.000	9.000	0.900	0.090	0.009
0.3	102.857	10.285	1.028	0.102	0.010
0.4	120.000	12.000	1.200	0.120	0.012
0.5	144.000	14.400	1.440	0.144	0.014
0.6	180.000	18.000	1.800	0.180	0.018
0.7	240.000	24.000	2.400	0.240	0.024
0.8	360.000	36.000	3.600	0.360	0.036
0.9	720.000	72.000	7.200	0.720	0.072
0.99	7200.000	720.000	72.000	7.200	0.720

Table 4.2: Transmission Time in Seconds for Adaptive Video Coding (ALIC)

Network Utilization	100 Kbps	1 Mbps	10 Mbps	100 Mbps	1 Gbps
0.0	72.000	7.200	0.720	0.072	0.007
0.1	20.000	2.000	0.200	0.020	0.002
0.2	15.000	1.500	0.150	0.015	0.001
0.3	12.857	1.285	0.128	0.012	0.001
0.4	12.000	1.200	0.120	0.012	0.001
0.5	12.000	1.200	0.120	0.012	0.001
0.6	12.857	1.285	0.128	0.012	0.001
0.7	15.000	1.500	0.150	0.015	0.001
0.8	20.000	2.000	0.200	0.020	0.002
0.9	36.000	3.600	0.360	0.036	0.003
0.99	342.857	34.285	3.428	0.342	0.034

These two tables help to illustrate which network types can support video applications. Low speed networks such as the 100 Kbps and 1 Mbps networks do not possess the raw bandwidth needed to support video applications. For example, even at a network utilization of 0.5, an image (as described previously) compressed using the ALIC compression profile, requires 1200 msecs to be transmitted over a 1 Mbps network. On the other hand, gigabit networks can provide the bandwidth requirements of video traffic even without compression. For example, even at a network utilization of 0.5, an image compressed using the NALC compression profile, requires only 14 msecs to be transmitted over a 1 Gbps network. Thus, 100 Kbps, 1 Mbps and gigabit networks can be eliminated from the domain of adaptive video coding algorithms. Therefore, the networks of interest are the 10 Mbps and 100 Mbps networks.

With the compression scenario under consideration, 10 Mbps networks cannot support real-time video. However, 10 Mbps networks can support still image transfer applications or black and white video streams with smaller images. Thus, an adaptive video coding algorithm running on a 10 Mbps network can adapt its compression ratio to the network load whenever an image is requested by the user.

Networks of 100 Mbps can support NTSC standard real-time video even under the ALIC compression scenario considered. Adaptive video coding algorithm running on a 100 Mbps network can guarantee a maximum bound on the image transmission time (33 msec/frame) across a wide range of network loads making the algorithms more robust.

4.5 Summary

The results of the analytical study suggest that it is useful to have video coding algorithms adapt to changes in the network load condition. Three specific observations about adaptive video coding algorithms were made in this chapter:

- Adaptive video coding algorithms are robust and are able to guarantee a maximum bound on the image transmission time across a wide range of network loads.

- Adapting the video coding algorithm to the changes in the network load is advantageous across a wide range of network loads (20%-80%).

- Adaptive video coding algorithms for NTSC standard video are best suited for network speeds on the order of 100 Mbps. On 10 Mbps networks, adaptive video coding algorithms can be used for still image transfer.

Chapter 5

Experimental Methodology

This chapter presents the experimenal methodology used in this research. The experimental factors considered and performance metrics used are also presented.

5.1 Experimental Design

In this section, a subset of the video coding algorithms are identified as potential candidates for adaptive video coding algorithms. The three intraframe algorithms under consideration are JPEG [4], EPIC [65], and the fractal coding algorithm [5, 8]. The MPEG algorithm [40] is the only interframe coding algorithm considered in this research.

These coding algorithms provide a reasonable representation of the hypothetical cases considered in the previous chapter. JPEG has an almost quadratically increasing compression curve similar to the AQIC profile (Figure 4.1). EPIC's compression curve is linearly increasing and resembles the ALIC profile. Fractal compression has a step function for its compression curve and resembles the ASFC profile. The fractal coding algorithm was chosen because it is resolution independent. Thus, it can provide very high compression ratios at very high loads with very good image quality. Subband coding forms the basis of the EPIC video coding algorithm. JPEG is based on the hybrid compression technique and fractal compression is based on the transform compression technique. The MPEG algorithm is based on the interframe coding method.

For the intraframe compression algorithms (JPEG, EPIC and fractal), the experiments are carried out on a non-changing image sequence with each image corresponding to the type of image seen in video teleconferencing applications. The test image used is a standard image for evaluation of video coding algorithms [45, 63]. The image is a grayscale image with 128×128 pixels and 8 bits of information per pixel (Figure 5.1). The compressed image was broken up into packets of size 128 bytes for (simulated) transmission.

The interframe compression algorithm (MPEG) was evaluated using a sequence of 10 frames of 352×240 pixels in the YUV format (12 bits per pixel) obtained with the MPEG coder-decoder [60] from Berkeley.

Figure 5.1: The Test Image

A simulation model was built to test the adaptivity of the video coding algorithms. The simulator, shown in Figure 5.2, is very simple with a sender, which compresses the image and transmits it over the network, and a receiver, which decompresses and displays the compressed image received. The network is loaded with cross-traffic from a sender-receiver pair. The sender at the time of compression gets the network load level and compresses the image accordingly. The network load at the time of compression is the level of cross traffic and during image transmission the sum of the cross traffic and the image transmission from the sender.

The simulator requires the following input specifications:

1. The specification of the video coding algorithm to be used at the sender. The compression is done at the sender by adjusting the parameters of the algorithm depending on the network load.

2. The specification of the bandwidth and propagation delay of the underlying network.

3. The specification of the traffic model and load level to be used for the cross-traffic.

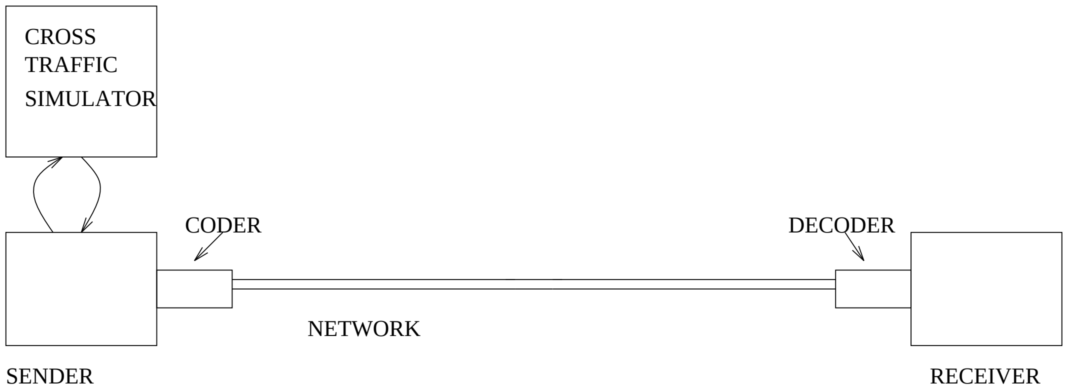


Figure 5.2: The Simulation Model

Simple assumptions have been made in constructing the simulator. First, the sender can monitor the cross traffic model at all times, thus knowing of any changes in the network load as soon as the changes occur. Second, once the image is compressed and the transmission is started, even if the load changes, the transmission of the current image is completed. Third, the Poisson model was chosen for the cross traffic. In a Poisson model, the packet interarrival times are independent and exponentially distributed with a mean $1/\lambda$, where λ is the mean packet arrival rate [30].

The entire simulation is carried out an a shared Sparc10 machine. The presence o f other users on the same machine does not affect the transmission time and picture quality, but does affect the compression and decompression time. The experiments are reduplicated twice and the results averaged.

5.2 Experimental Factors

The following factors were considered in the experiments, with levels as indicated:

- Network Speeds: Two network speeds were considered for the experiments: 10 Mbps and 100 Mbps. Simulations with these network speeds will indicate the feasibility of adaptive video coding algorithms on these networks and justify the analytical evaluation in the previous chapter. These simulation experiments would also help in determining the feasibility of extending adaptive video coding algorithms to real-time video on these networks.

- Video coding algorithms: Four algorithms were chosen for the experimental study: JPEG, EPIC, fractal and MPEG.

- Cross traffic scenarios: The traffic scenarios shown in Figures 5.3 and 5.4 are to be used in the simulation. The network load is plotted against the time of the day. The value of each time unit is 0.0075 secs (i.e., transmitting 30 images per second and one image is compressed every 4 time units). This particular value for the time was chosen so that there are changes in network load during the transmission of the image.

  Four traffic scenarios are considered:

  - Scenario GI: there is a Gradual Increase in the network load. The network load increases in small steps. The load curve is linear and increases in increments of 5 percent for every time unit.
  - Scenario GD: there is a Gradual Decrease in the network load. The network load decreases in small steps. The load curve is linear and decreases in decrements of 5 percent for every time unit.
  - Scenario RI: there is a Rapid Increase in the network load from a state of low load to being almost fully loaded. Thus the network load goes from 0 percent in the first time unit to 99 percent in almost the next time unit.

– Scenario RD: there is a Rapid Decrease in the network load. The network goes from a state of being almost fully loaded to a state of being very lightly loaded. Thus, the network load decreases from 99 percent in the first time unit to 0 percent in almost the next time unit.

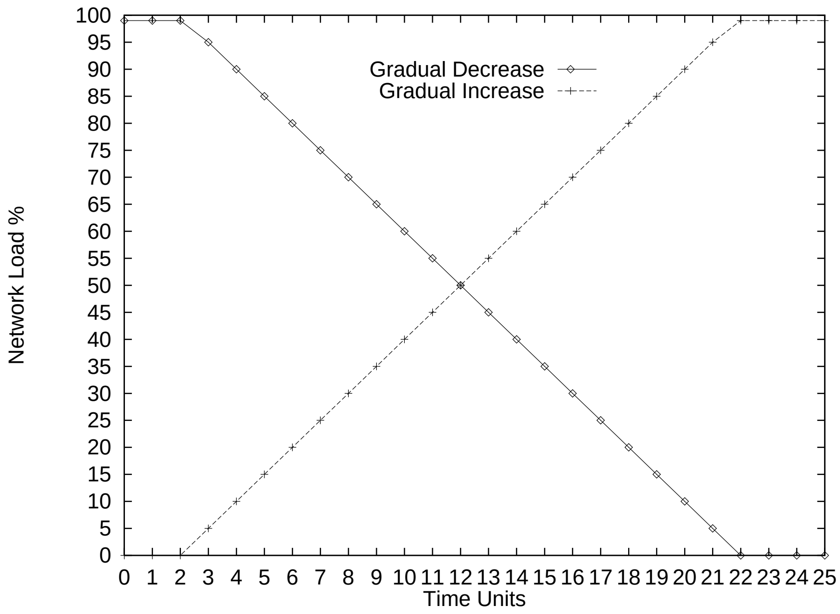


Figure 5.3: Gradual Change in Cross Traffic

5.3 Performance Metrics

The performance metrics used in the experiments are explained as follows:

- Compression Ratio: Compression ratio is defined as the ratio of the size of the original image (in bytes) to the size of the compressed image. The compression ratio indicates the level of compression that the video coding algorithm offers. The higher the compression ratio, the better.

- Processing Time: A certain amount of time is spent in encoding and decoding the image and constitutes the processing time. The lower the processing time, the better will be the end-to-end performance of the video coding algorithm. The processing time for an image depends to a large extent on the video coding

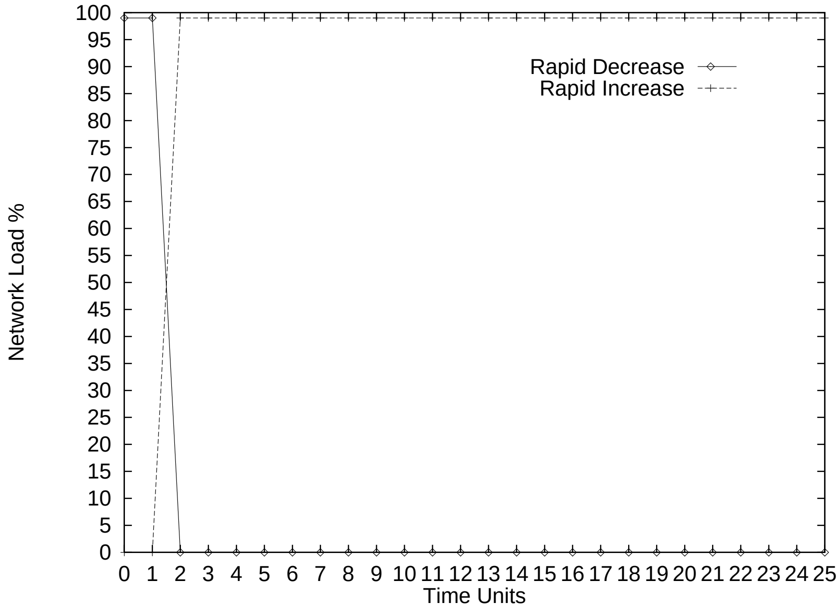


Figure 5.4: Rapid Change in Cross Traffic

algorithm used and the contents of the image. The compression and decompression times are measured using the Unix *gettimeofday()* function. This function measures the (wall-clock) time (to the nearest microsecond) spent executing user instructions in the encoding and decoding process.

- Transmission Time: Transmission time is defined as the difference between the time the sender starts transmitting the first bit of the compressed image and the time the last bit is received by the receiver. The transmission time depends on the size of the compressed image, the network load level and the cross traffic.

- Image Quality: There are two means of evaluating the quality of an image: subjective evaluation and objective evaluation. The subjective evaluation is a human judgement of the perceptual quality of the image. Objective evaluation measures give a quantitative evaluation of the quality of the reconstructed image by comparing it to the original image.

  In this research the average error per pixel is used as the quantitative measure of the image quality. The average error per pixel ($\overline{E}$) is computed by summing up the absolute differences in the intensities of all corresponding pixels in the original and reconstructed image and dividing the sum by the total number of pixels in the image. Therefore the higher the average error per pixel, the lower the image quality, and vice versa. Mathematically.

$$\overline{E} = \frac{\sum_{i=0}^{N-1} \mid P_i - \hat{P}_i \mid}{N}$$

where P_i and $\hat{P}_i$ are the grayscale values of pixel number i in the original and reconstructed image, respectively, and N is the total number of pixels in the image.

5.4 Implementation of the Algorithms

This section explains the implementation of the specific video coding algorithms under consideration.

JPEG: The JPEG algorithm coded by the Independent JPEG group [77] was used in the experimental study. The JPEG algorithm is implemented in the C programming language. The algorithm is as described in Section 3.3.1. The implementation had to be modified such that the quantization levels were varied as the network load changed in order to make the JPEG algorithm adaptive to network load changes.

EPIC: The EPIC video coding algorithm developed at MIT media labs by Adelson *et al.* [1] was used for evaluating the algorithm. The implementation is as discussed in Section 3.3.2.

Fractal Image Coding: The algorithm suggested by Anson [5] was used to implement fractal image coding. In our implementation of the fractal image coding algorithm, the entire image is broken up into non-overlapping range blocks of 4×4 pixels. The domain blocks are overlapping and have 8×8 pixels. The transformations are limited to rotation and flip. Thus there are only 16 transformations (4 flips, 4 rotations). This limitation of the ranges to being only of size 4×4 pixels limits the amount of granularity the coding technique can capture. For example, in the encoded image in Figure 5.5 (a), the teeth seem to be one white smudge. So a modification to the fractal compression algorithm as suggested by Fisher [18] was made to better encode the image.

The modified fractal compression algorithm uses quad-tree partitioning to break down the ranges. In the quad-tree partitioning scheme, an image is successively broken up into four quadrants until a specified condition (error of the quadrant less than a minimum error) is met or the quadrants are too small to be divided any further. For the image considered, the ranges were assumed to be of size 8×8 pixels with the domains of size 16×16 pixels. The domains are then mapped onto the ranges so that the error is less than a certain specified minimum error. If no domain can be mapped onto the range, then the range is broken up into four quadrants and the process repeated. The process is repeated until the error is less than the specified minimum error or the range cannot be further subdivided. An image compressed using this method is shown in Figure 4.5 (b). The fractal compression method with quad-tree is seen to be better and the teeth are no longer a white smudge (the $\overline{E}$ is lower for the fractal compression method with quad-tree than the original fractal compression method).

a  b 

Figure 5.5: Decoded Image (a) Original Fractal Compression ($\overline{E} = 17.84$ (b) Fractal compression with Quad-Tree ($\overline{E} = 14.62$)

MPEG: The MPEG video coding algorithm developed by Rowe *et al.* [60] at U C Berkeley was used in the experimental study. The MPEG video coding algorithm is implemented in the C programming language on the X windows system [62]. The MPEG codec consists of a video coder which accepts an image sequence in YUV format and compresses each image as either an I, B or P frame as specified in a parameter file. Other parameters that can be changed are the number of frames in the GOP sequence, the quantization levels of each frame and the order of frames in a GOP sequence. Varying these parameters results in compression levels of 4:1 to 400:1. The decoder takes a compressed video stream, uncompresses it and displays the video sequence on the X windows system using 8 bits of colour information per pixel. The decoder had to be modified slightly in order to evaluate the quality of received image using the average error per pixel measure.

5.5 Summary

This chapter presented the experimental design used in the reserach. The metrics used to evaluate the performance of video coding algorithms and the implementations of the adaptive video coding algorithms under consideration were also presented.

Chapter 6

Experimental Results

This chapter presents the experimental results of the adapative interframe and intraframe video coding algorithms under consideration. The purpose of this chapter is to explore the following: (1) to see if adaptive video coding algorithms are robust to changes in network load; (2) to see if there is any advantage to adaptivity on a granularity smaller or larger than a per frame basis; (3) to observe the relative performance of some intraframe video coding algorithms with widely different characteristics; (4) to see if adaptive interframe video coding algorithms have similar characteristics and performance as adaptive intraframe video coding algorithms.

6.1 Results for Adaptive JPEG Implementation

The JPEG algorithm was studied under the conditions mentioned in Section 5.2 and the results are described as follows.

Figure 6.1 shows the compression time curves for the JPEG algorithm. The mean compression time is 0.75 secs and the compression time remains close to the mean across a wide range of network loads, indicating that the quantization step (the adaptive part of the algorithm) has minimal impact on the running time of the overall JPEG algorithm.

Figure 6.2 shows the decompression time curves for the JPEG algorithm. Once again the fluctuations in the decompression time are due to the decompression al-

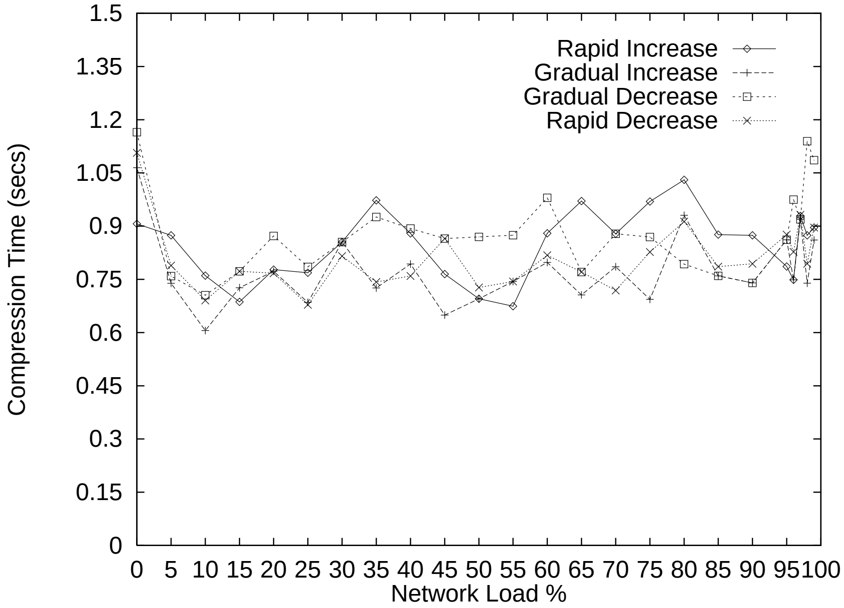


Figure 6.1: Compression Time for Adaptive JPEG Video Coding Algorithm

gorithm running on a machine that is running other processes as well. The decompression times for the JPEG algorithm are lower than the compression times because JPEG is an asymmetric compression algorithm.

Figure 6.3 shows the compression profile of the JPEG algorithm. The quantization level of the JPEG algorithm was increased as the network load increased resulting in higher compression ratios. The compression ratios range from 1:1 to 28:1 and increase almost quadratically as the quantization parameter is changed. The compression ratio offered by the adaptive JPEG algorithm depends solely on the network load at the time of image compression. Therefore the four cross traffic scenarios all have the same compression ratio at any particular network load and the compression curves for the four cross traffic scenarios coincide.

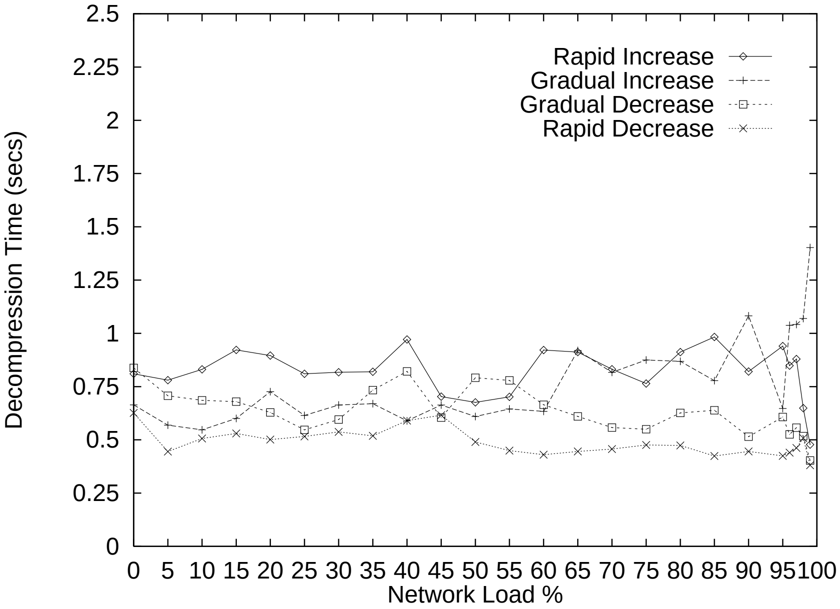


Figure 6.2: Decompression Time for Adaptive JPEG Video Coding Algorithm

Figure 6.4 shows the average error per pixel curve for the JPEG algorithm. The average error per pixel increases as the network load level increases because the image is being compressed more, and more information is lost. The average error per pixel curves for the four cross traffic scenarios coincide because the four cross traffic scenarios provide the same level of compression as each other at any network load level and images that are compressed the same have the same average error per pixel.

Figure 6.5 shows the transmission times for the adaptive JPEG algorithm on a 10 Mbps network. The transmission time is found to remain below the 33 msecs/frame delivery time for real-time video over a wide range of network loads for the RD, GD and GI cross traffic scenarios. The variation in the transmission time is also reduced: the transmission time remains almost constant even though the network load increases because of the adaptivity of the algorithm (i.e., JPEG compresses the image more at higher load levels).

From Figure 6.5, it is observed that the transmission time for the RI traffic scenario is much higher than the other three cross traffic scenarios for almost all network loads. The reason for the transmission time being so high is that the image is compressed at a low load (low compression) and the very next instant the network becomes almost

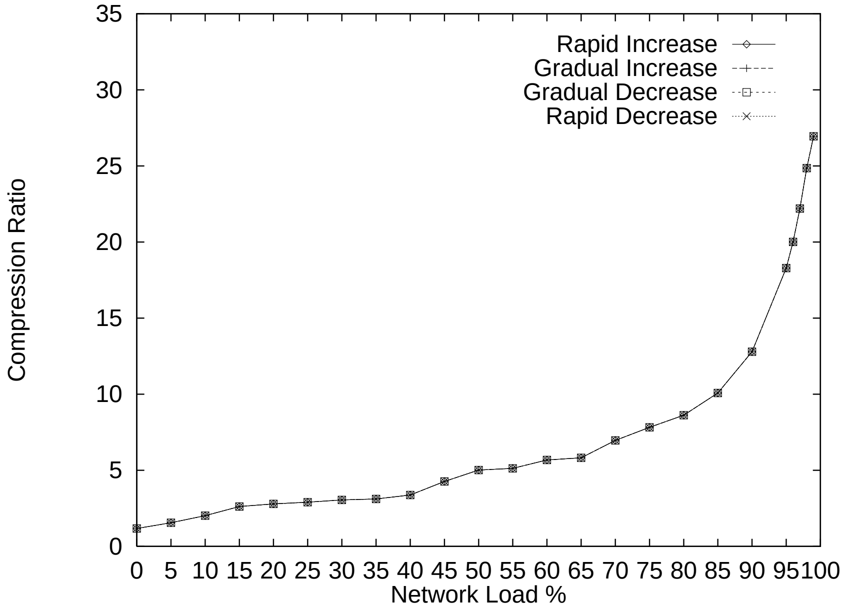


Figure 6.3: Compression Profile for Adaptive JPEG Video Coding Algorithm

fully loaded. Thus, the sender is trying to transmit a large image in a heavily loaded network resulting in a large transmission time. For the GD and RD scenarios, the transmission time is very low and remains relatively constant for load levels of 0%-95% because the image is compressed at a high load (and therefore higher compression) and the transmission takes place at a lower load. For the GI scenario the transmission time is the same as the GD and RD scenarios and remains constant for load levels of 0%-75%. The transmission time for the GI scenario is higher than the GD and RD scenarios for load levels greater than 75% because the JPEG algorithm provides the same level of compression in the GI, GD and RD scenarios but the load increases, during transmission, in the GI scenario (as opposed to decreasing in the GD and RD scenarios).

For the JPEG algorithm on a 100 Mbps network only the curves for the transmission time (Figure 6.5) are shown because the compression and decompression time, compression ratio and average error per pixel are the same as on a 10 Mbps network. The transmission time curves for JPEG on a 100 Mbps network are similar to the transmission time curves for JPEG on a 10 Mbps network, but are lower by an order of magnitude on the 100 Mbps network.

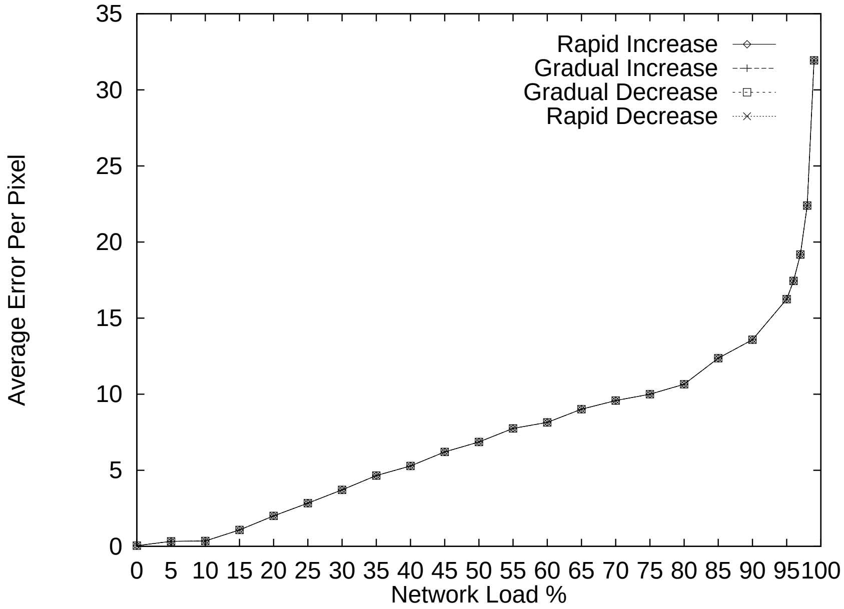


Figure 6.4: Error Curve for Adaptive JPEG Video Coding Algorithm

The adaptive JPEG video coding algorithm provided poor performance for the transmission time for the RI scenario. For the rest of the traffic scenarios (GI, GD and RD) the adaptive JPEG video coding algorithm is able to guarantee a maximum bound on the image transfer time for moderate to heavy network loads. Thus the adaptive JPEG algorithm is robust and is able to keep the transmission time below the maximum bound while maintaining reasonable image quality for moderately to heavily loaded networks provided that the load does not rapidly increase.

6.2 Revised JPEG Implementation

The implementation of the adaptive JPEG video coding algorithm in Section 6.1 adapts to the changes in the network load at the time of compression of each new frame (interframe adaptivity). Thus, the quantization levels of the adaptive JPEG video coding algorithm are adjusted according to the network load level at compression time.

Another approach to adaptivity would be to adapt to changes in a network on a granularity smaller than a per frame basis (i.e., on an intraframe basis). The suggested

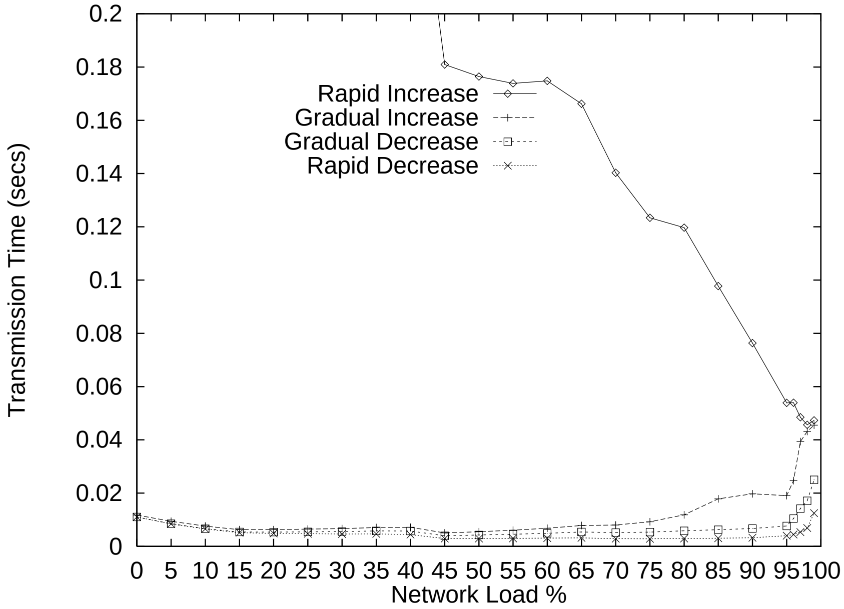


Figure 6.5: Transmission Time Curves for Adaptive JPEG Video Coding Algorithm on a 10 Mbps Network

modification involves breaking up the image into smaller parts and compressing and transmitting each part separately. This scheme is called the revised scheme. Thus, at the time of compression of each part of the image, the quantization levels of the adaptive JPEG video coding algorithm are adjusted according to the network load level.

The revised scheme would make the algorithm more adaptive as follows:

- If the network load suddenly increases (scenario RI) after the image compression was started and if the entire image was compressed and transmitted, then it would take a long time for the image to reach the end-user since the image is compressed at a low load (low compression ratio). But, if the image is broken up into say four quadrants, then the first quadrant will have a low compression ratio, but the other quadrants will have very high compression ratios since they are compressed at higher loads. This will reduce the overall transmission time for the whole image.

- If the network load suddenly decreases (scenario RD) after the image compression was started, and the entire image was compressed and transmitted, then it

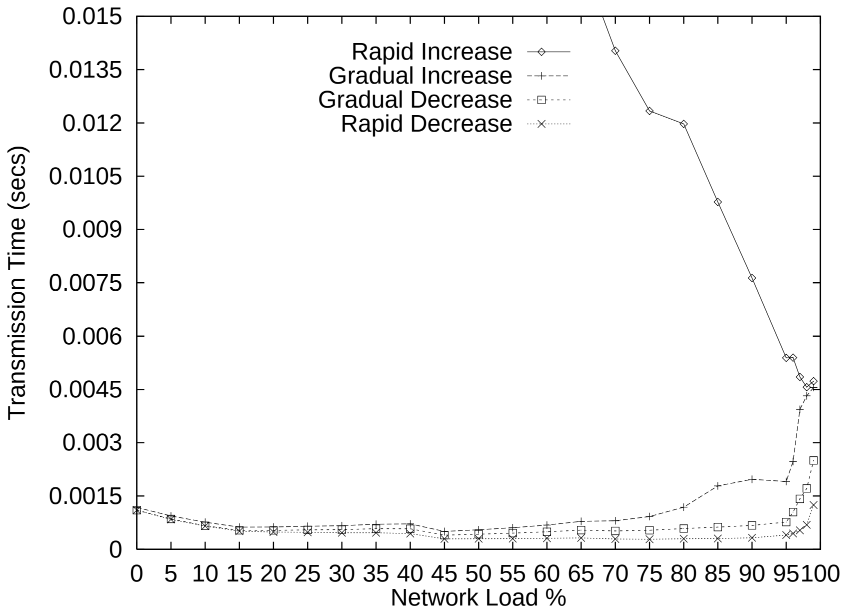


Figure 6.6: Transmission Time Curves for Adaptive JPEG Video Coding Algorithm on a 100 Mbps Network

would be compressed at a higher compression ratio and will have lower image quality. But, if the image is broken up into four quadrants, the first quadrant would be compressed at a higher load (higher compression ratio) but the other three quadrants would be compressed at a lower load and thus the overall image should have a better quality.

The adaptive JPEG algorithm had to be modified to work with the revised scheme. The modifications involved breaking up the image into four quadrants and compressing and transmitting each quadrant separately. At the receiver, each encoded quadrant is decoded and the image is reconstructed.

6.3 Results for Revised JPEG Implementation

In order to test the revised scheme, experiments were conducted under the same conditions as the original scheme. The results of the experiments with the revised JPEG implementation, with the four cross traffic scenarios and Poisson cross traffic on a 10 Mbps network are shown in Figures 6.7 to 6.9.

From Figure 6.7, it is observed that the compression ratios of the quadrants is lower than the compression ratio of the whole image (Figure 6.3). This phenomenon occurs because JPEG is an intraframe compression algorithm that tries to look for and encode similar frequencies with the same code. Thus in a smaller region there are fewer repetitions of frequencies and the compression ratio is lower. Thus, when the image is broken up into four quadrants, the algorithm gives lower compression ratios than if the entire image is compressed as a whole.

The average error per pixel (Figure 6.8) for the four cross traffic scenarios increases as the network load increases because at higher network loads the image is compressed more and therefore the image quality is lower. The average error per pixel for the revised JPEG algorithm with the RI cross traffic scenario is much higher than the adaptive JPEG algorithm (Figure 6.4) with the RI cross traffic scenario because the network becomes almost fully loaded after the first quadrant is compressed and the remaining three quadrants are compressed at a very high compression ratio. The revised JPEG algorithm provides almost the same average error per pixel as the adaptive JPEG for the RD, GD and GI cross traffic scenarios.

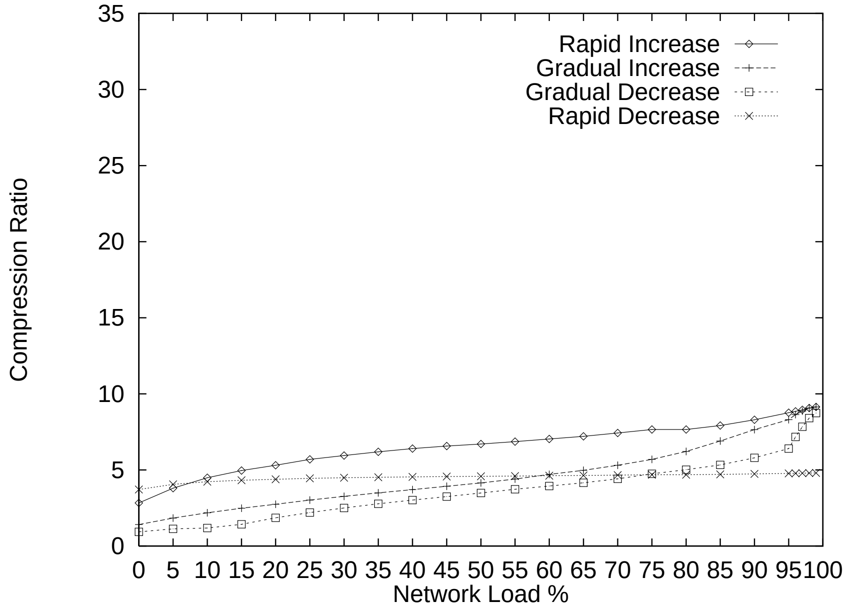


Figure 6.7: Compression Profiles for Revised JPEG Algorithm

The transmission times for the revised JPEG algorithm (Figure 6.9) with the RD and GD traffic scenarios are observed to be almost the same as the transmission times for the adaptive JPEG algorithm (Figure 6.5). The revised JPEG algorithm has much lower transmission time for the RI traffic scenario than the adaptive JPEG algorithm. The adaptive JPEG algorithm has slightly lower transmission times for the GI traffic scenario for loads greater than 80%. The revised JPEG algorithm has a much lower transmission time for the RI traffic scenario because even though the first quadrant is compressed at a lower load (lower compression), the other three quadrants are compressed at much higher network loads (higher compression). Thus, the revised JPEG algorithm is transmitting a much smaller image than the adaptive JPEG algorithm resulting in a lower transmission time.

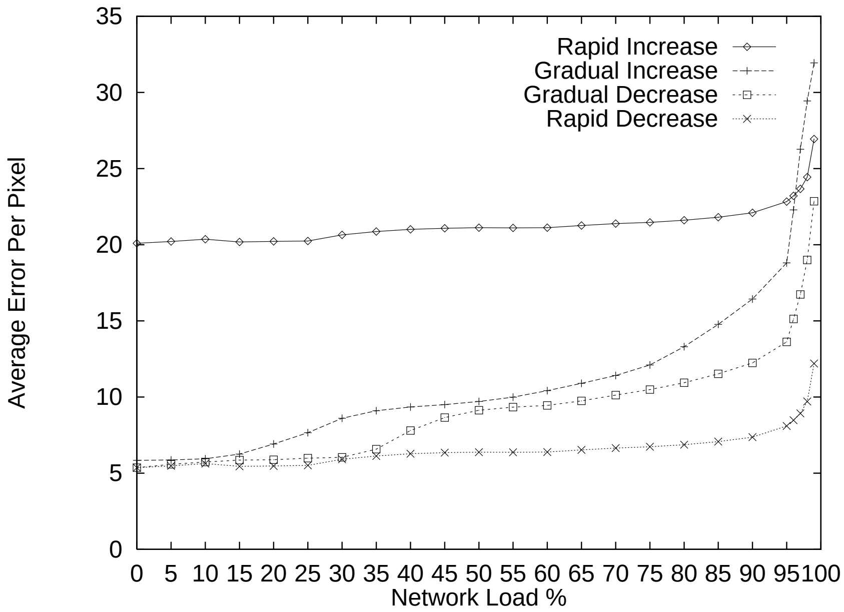


Figure 6.8: Error Curves for Revised JPEG Algorithm

Overall, the revised JPEG algorithm does not provide any substantial gain in the transmission time or image quality over the adaptive JPEG algorithm. Since there are processing overheads to breaking up an image into four quadrants and reconstructing the image, the slight gain in performance is more than offset by the processing overheads. Therefore adaptivity at a granularity smaller than a per frame basis is not deemed worthwhile. Thus the performance of the revised JPEG algorithm on a 100 Mbps network and the revised scheme for the other video coding algorithms (on both 10 Mbps and 100 Mbps networks) is not evaluated.

An adaptive JPEG algorithm adapting on a granularity larger than a per frame basis was also studied. The performance of the JPEG algorithm adapting on a larger granularity than a per frame basis was found to be much worse than the adaptive JPEG algorithm. For example, even adapting once every two frames provided very poor levels of performance because in the case of increasing load, one out of every two frames is compressed less. Therefore adaptivity at a granularity larger than a per frame basis is also not deemed worthwhile.

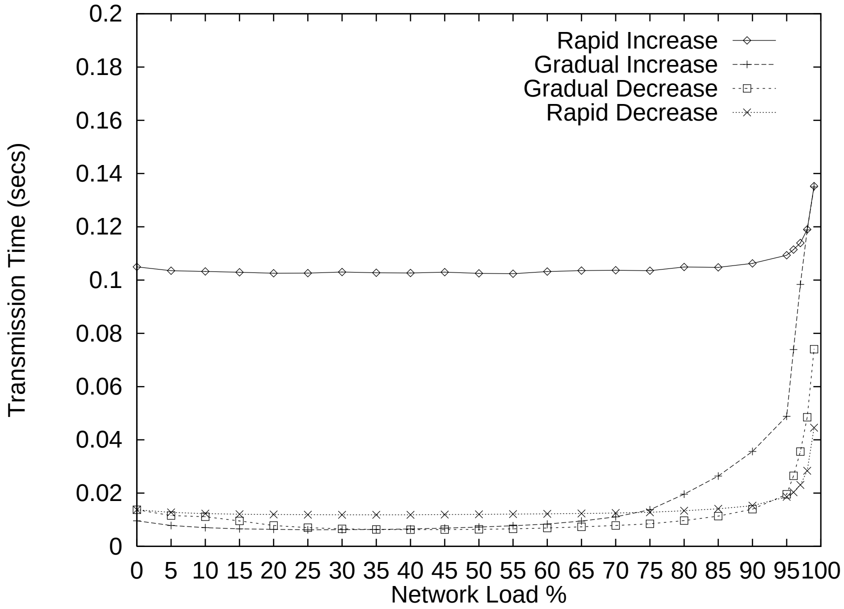


Figure 6.9: Transmission Time Curves for Revised JPEG Algorithm on a 10 Mbps Network

6.4 Results for EPIC Implementation

The EPIC video coding algorithm was evaluated to determine its performance on 10 Mbps and 100 Mbps networks.

Figure 6.10 shows the compression time curves for the EPIC algorithm. The compression times for the EPIC algorithm are slightly higher than the JPEG algorithm. The mean compression time is 1.1 secs, and compression times do not vary by much across a wide range of network loads. The presence of spikes in the compression time curves are attributed to the machine running the compression process being a shared machine.

Figure 6.11 shows the decompression time curves for the EPIC algorithm. The decompression times for the EPIC algorithm are slightly higher than the JPEG algorithm and the EPIC algorithm is also asymmetric.

Figure 6.12 shows the compression profile of the EPIC algorithm. The quantization levels of the sub-bands are increased as the network load increases to give higher compression ratios. The compression ratio increases almost linearly as the network load increases, with the compression ratio ranging from 1:1 to 14:1. The compression ratios provided by the adaptive EPIC algorithm are lower than the compression ratios of the adaptive JPEG algorithm.

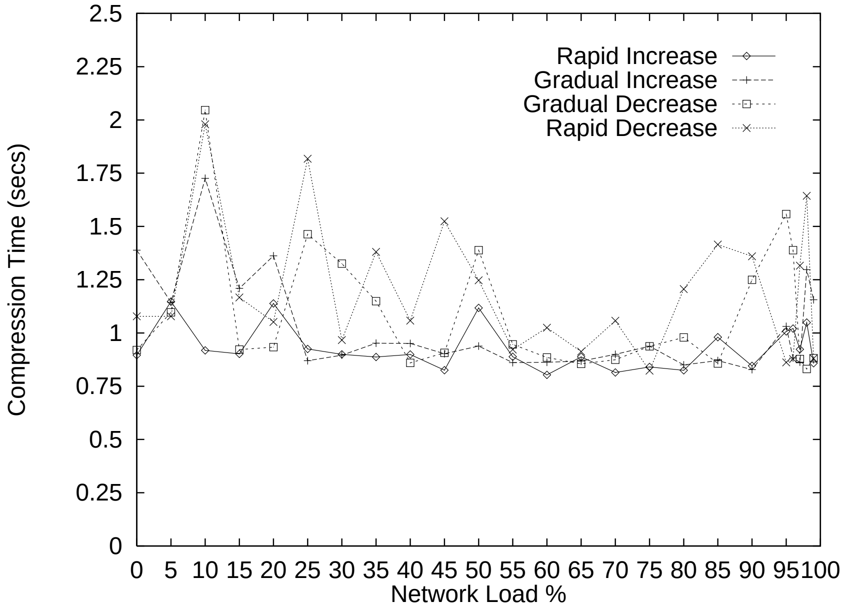


Figure 6.10: Compression Time for Adaptive EPIC Video Coding Algorithm

Figure 6.13 shows the average error per pixel curve for the EPIC algorithm. The average error per pixel increases almost linearly as the network load level increases from 0% to 25% and then flattens out slightly. The average error per pixel increases as the network load level increases because the image is being compressed more, and more information is lost.

Figure 6.14 shows the transmission time behaviour for the adaptive EPIC algorithm for the four cross traffic scenarios on a 10 Mbps network. From Figure 6.14, it is observed that the transmission time results are very similar to those of the JPEG algorithm (Figure 6.3) in terms of guaranteeing a maximum bound on the image transmission time for a wide range of loads provided the network load does not change rapidly. The adaptive EPIC algorithm is able to keep the transmission time fairly constant. The transmission times for the EPIC video coding algorithm are slightly higher than the JPEG algorithm for network load levels of 65%-99% because the EPIC algorithm provides lower compression ratios than the JPEG algorithm for similar network load levels.

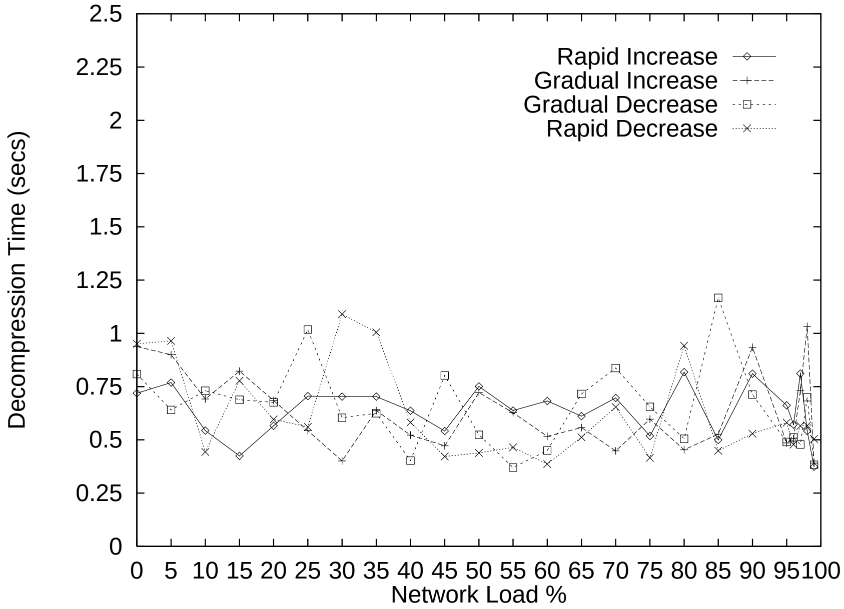


Figure 6.11: Decompression Time for Adaptive EPIC Video Coding Algorithm

The transmission time remains almost constant for the GD, RD and GI cross traffic scenarios. For the GD and RD cross traffic scenarios the image is compressed at a high network load (and therefore high compression ratio) and transmitted over a network that is relatively lightly loaded. For the RD cross traffic scenario the load levels do not change much after the image is compressed and since the image is compressed more at higher network loads the transmission time remains almost constant.

For the experiments on a 100 Mps network only the curve for the transmission time (Figure 6.15) is shown because the compression ratio and average error per pixel are the same as on a 10 Mbps network. The transmission time curves on the 100 Mbps network are lower than the transmission time curves on the 10 Mbps network by an order of magnitude.

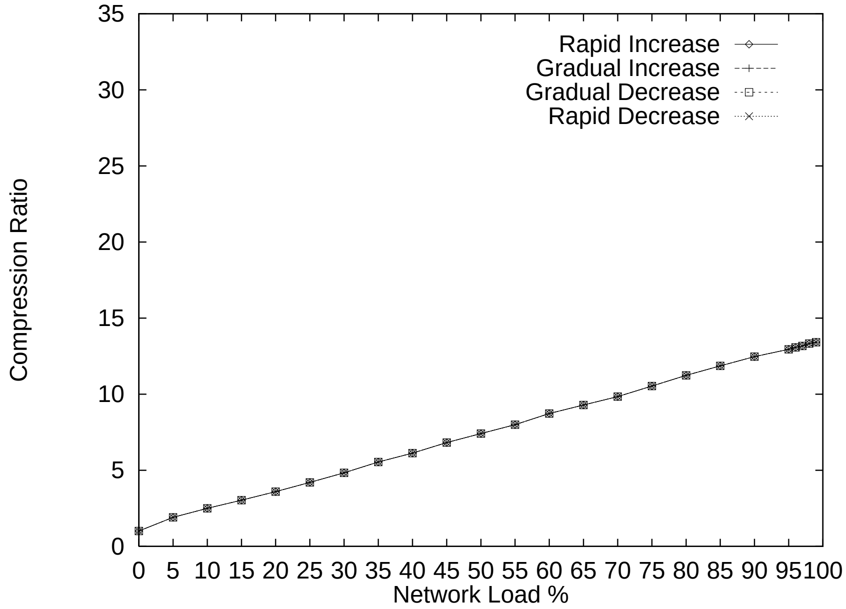


Figure 6.12: Compression Profile of Adaptive EPIC Video Coding Algorithm

6.5 Results for Fractal Implementation

The fractal video coding algorithm was studied to evaluate its performance relative to the JPEG and EPIC algorithms.

Figure 6.16 shows the compression time curves for the fractal algorithm. The compression time for the fractal algorithm is much higher than the JPEG and EPIC algorithm because the fractal algorithm takes about 45 minutes to compress an image. The curves for the four cross-traffic scenarios almost coincide because the variations in the compression time are negligible when compared to the total compression time.

Figure 6.17 shows the decompression time curves for the fractal algorithm. The decompression time for the fractal algorithm is almost double the decompression time for the JPEG and EPIC algorithms. The fractal algorithm is an assymetrcic algorithm with a mean decompression time of 1.5 secs.

Figure 6.18 shows the compression profile of the fractal algorithm. The acceptable error value for the fractal algorithm was increased as the network load increased to provide higher levels of compression. The reconstructed image became almost unrecognizable at error levels of 6 or greater and fractional values for error did not affect the compression ratio by much, so only 6 distinct compression ratios could be

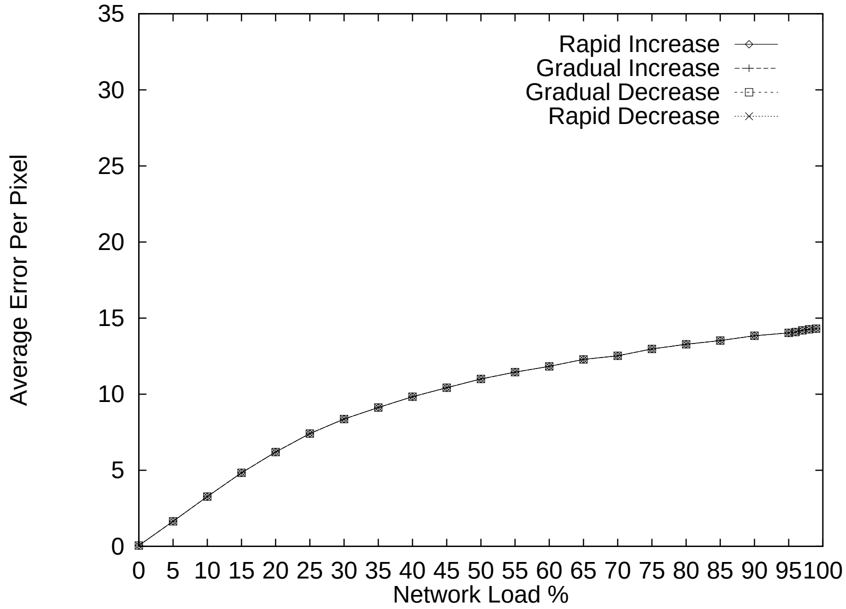


Figure 6.13: Error Curve for Adaptive EPIC Video Coding Algorithm

obtained. The compression ratio increases in a step function as the network load increases, and ranges from 1:1 to 18:1.

Figure 6.19 shows the average error per pixel curve for the fractal algorithm. The average error per pixel increases in a step function as the network load level increases because the compression profile for the adaptive fractal algorithm is a step function and higher levels of compression result in higher average error per pixel.

Figure 6.20 shows the transmision time results for the fractal algorithm on a 10 Mbps network. From Figure 6.20, it is observed that the fractal algorithm is able to guarantee a maximum bound (33 msecs/frame) on the image transmission time for a wide range of network load levels (except for the RI cross traffic scenario). The adaptive fractal algorithm keeps the transmission time fairly constant across a wide range of network loads. The transmission time for the fractal algorithm is lower than the EPIC and JPEG algorithm for network load levels of 15%-90% because the fractal algorithm offers higher compression ratios than the EPIC and JPEG algorithm for these network load levels. The transmission time curves for the fractal algorithm on a 100 Mbps network are not shown because the curves are similar to the transmission time curves on a 100 Mbps network, and only an order of magnitude lower on the

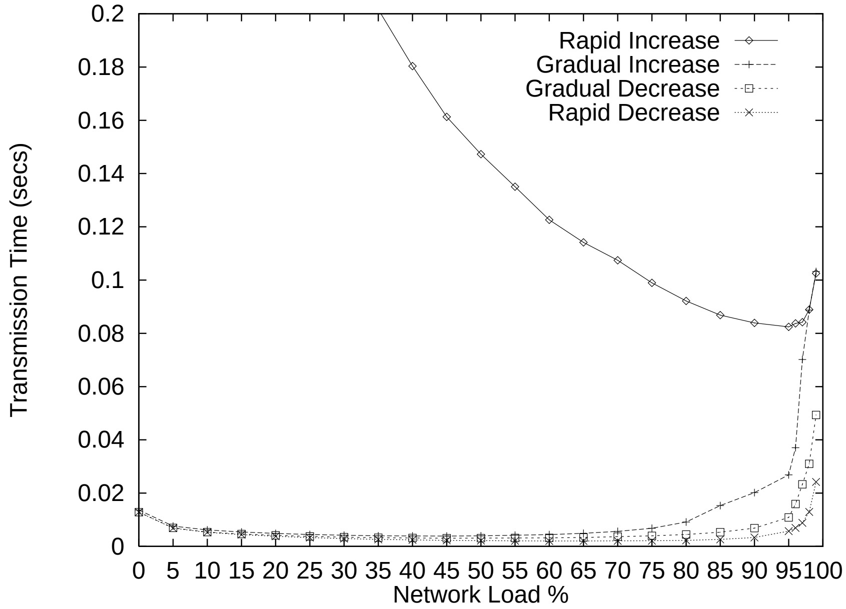


Figure 6.14: Transmission Time Curves for Adaptive EPIC Video Coding Algorithm on a 10 Mbps Network

100 Mbps network.

Even though the fractal algorithm is robust to changes in network load (by providing transmission times below a certain threshold), the time taken to compress and decompress an image are very high. So the fractal coding algorithm is not very responsive to changes in network load and is a poor choice for adaptivity.

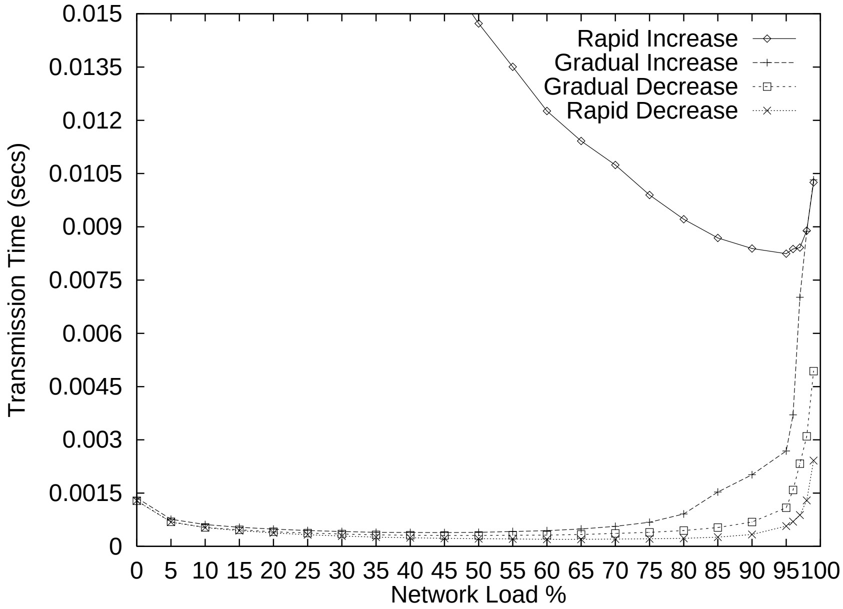


Figure 6.15: Transmission Time Curves for Adaptive EPIC Video Coding Algorithm on a 100 Mbps Network

6.6 Results for MPEG Implementation

The MPEG video coding algorithm was evaluated in order to determine how interframe video coding algorithms adapt to changes in network load and if adaptive interframe video coding algorithms have the same characteristics as adaptive intraframe video coding algorithms.

Figure 6.21 shows the compression time curves for the MPEG algorithm. The compression times for the MPEG algorithm are slightly higher than the compression times for the JPEG algorithm but lower than the compression times for the EPIC algorithm.

Figure 6.22 shows the decompression time curves for the MPEG algorithm. The decompression times for the MPEG algorithm are much lower than the decompression times for the EPIC and JPEG algorithms.

Figure 6.23 shows the compression profile of the MPEG algorithm. There are two ways for the MPEG algorithm to adapt to changes in the network load. First, the frame can be encoded as a P-frame, B-frame or I-frame. Second, the quantization level can be varied for each frame.

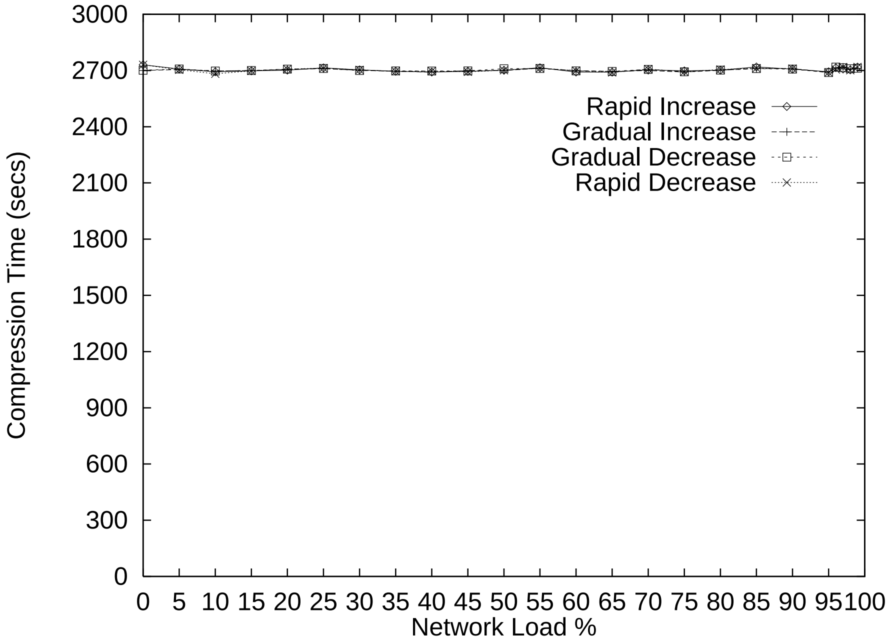


Figure 6.16: Compression Time for Adaptive Fractal Video Coding Algorithm

The MPEG algorithm offers very high compression with compression ratios ranging from 1:1 to 440:1. In this experiment, at load levels of 0%-40% the frame is encoded as an I-frame (offering low compression and high image quality), and only the quantization levels is varied. At load levels of 40%-80% the frame is encoded as a P-frame (offering intermediate compression and intermediate image quality), and only the quantization level is varied. At load levels of 80%-99% the frame is encoded as a B-frame (offering high compression and low image quality), and only the quantization level is varied.

Figure 6.24 shows the average error per pixel curve for the MPEG algorithm. The average error per pixel increases as the load level increases from 0% to 45% and then almost levels off. The average error per pixel almost levels off for load levels greater than 45% because the algorithm is using interframe compression, and there is not much further deterioration in the image quality.

Figure 6.25 shows the transmission times for the MPEG algorithm on a 10 Mbps network. The transmission times for the MPEG algorithm are much lower than the JPEG, EPIC and Fractal algorithms because the compression ratio provided by the MPEG algorithm is much higher. The transmission time is found to decrease as

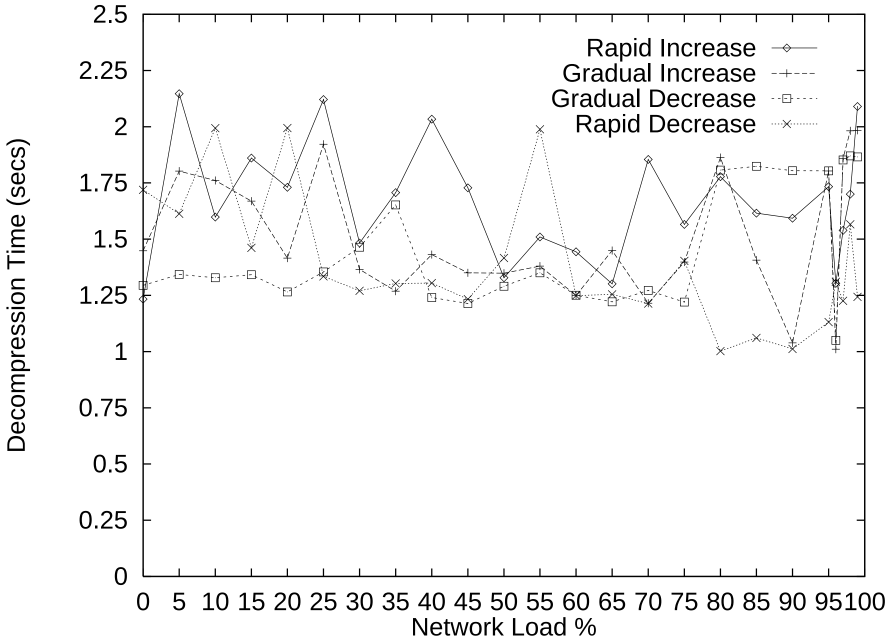


Figure 6.17: Decompression Time for Adaptive Fractal Video Coding Algorithm

the network load increases because the compression ratio provided by the MPEG algorithm increases significantly as the network load increases.

Even though the MPEG algorithm provides transmission times that are much lower than the other algorithms, the image quality for the MPEG algorithm is very poor at load levels of 60% to 99%. Thus there is a tradeoff between the transmission time and the received picture quality and the balance between the two depends on the application and the demands of the end user.

6.7 Summary

From the experimental study it is observed that:

- Adaptive video coding algorithms are robust because they are quite effective in guaranteeing certain maximum bounds on the image transfer time across a wide range of network loads.

- Among the intraframe compression algorithms tested, the adaptive JPEG algorithm provides the most promise for adaptivity.

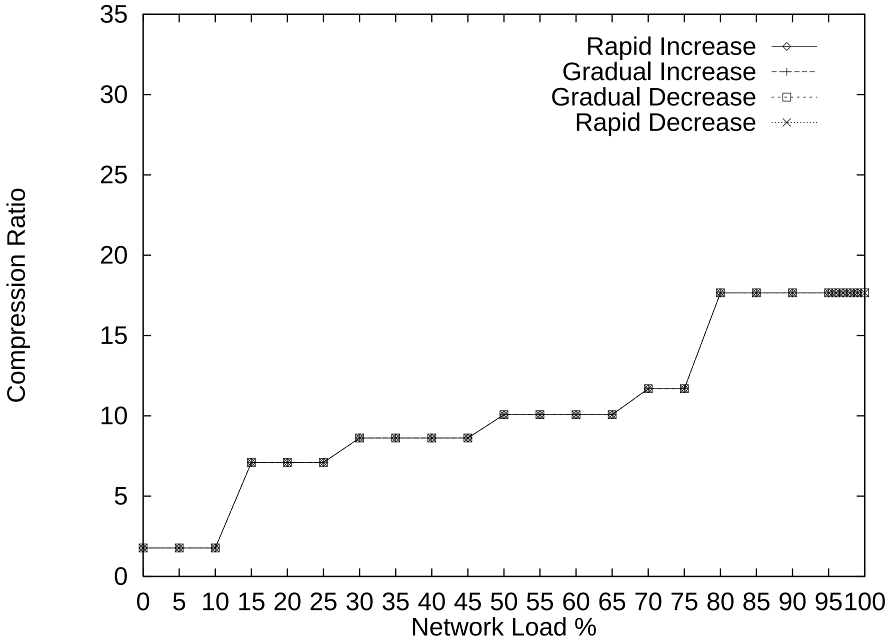


Figure 6.18: Compression Profile for Adaptive Fractal Video Coding Algorithm

- The revised JPEG scheme with intraframe adaptivity provides slightly lower transmissions than the adaptive JPEG scheme with interframe adaptivity, but the overheads are more significant than the gains.

- Adaptivity on a granularity smaller or larger than a per frame basis does not offer significant advantage over adaptivity on a per frame basis.

- The fractal video coding algorithm is not suitable for adaptivity because of the very high compression and decompression time. Thus adaptive video coding algorithms need to have very low compression and decompression times along with reasonable compression levels.

- Adaptive interframe video coding algorithms have characteristics similar to adaptive intraframe video coding algorithm.

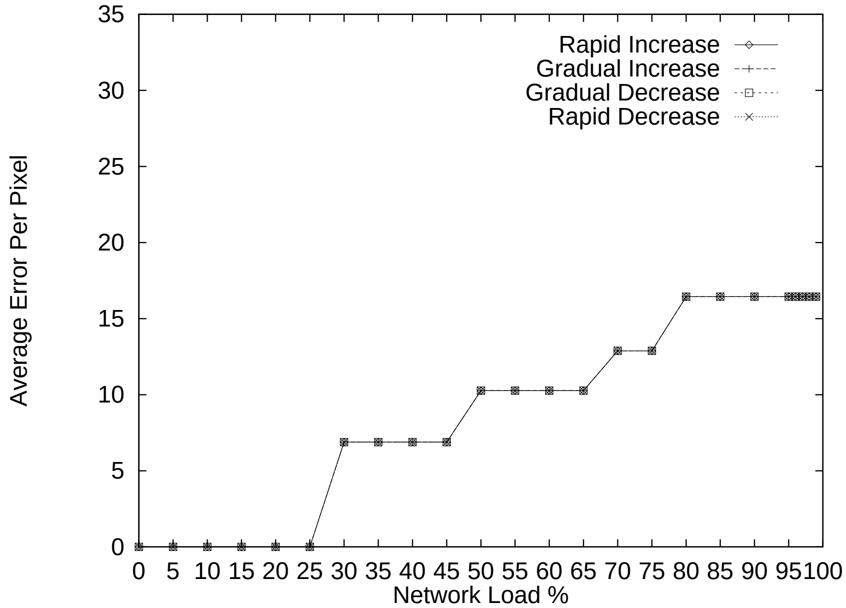


Figure 6.19: Error Curve for Adaptive Fractal Video Coding Algorithm

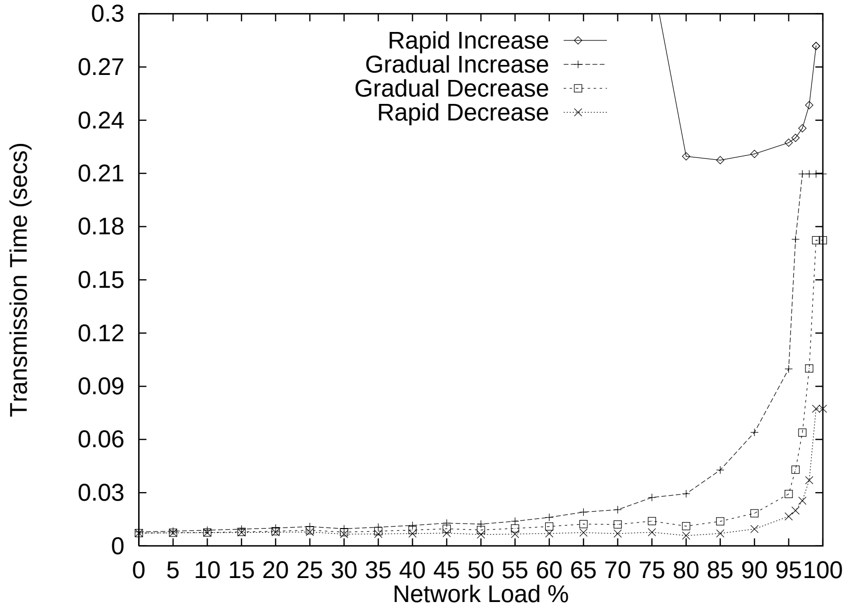


Figure 6.20: Transmission Time Curves for Adaptive Fractal Video Coding Algorithm on a 10 Mbps Network

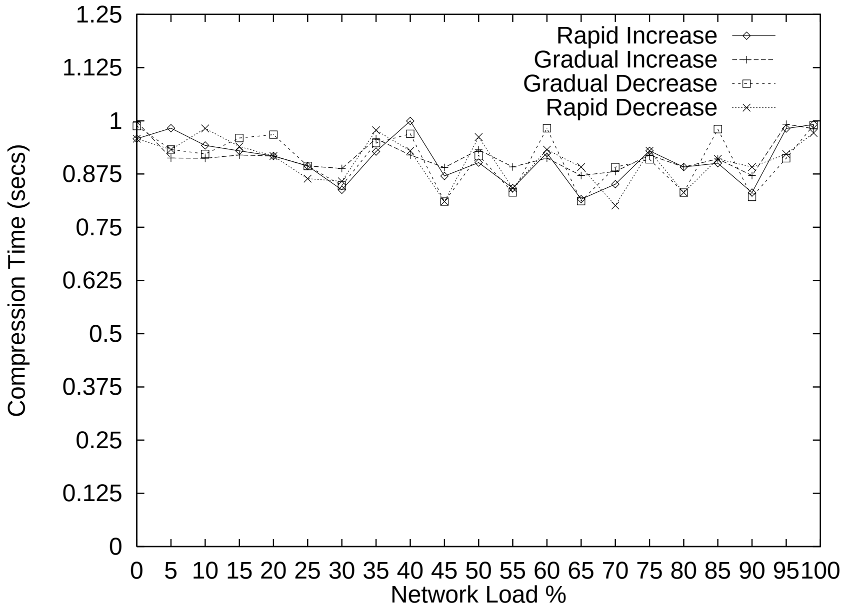


Figure 6.21: Compression Time for Adaptive MPEG Video Coding Algorithm

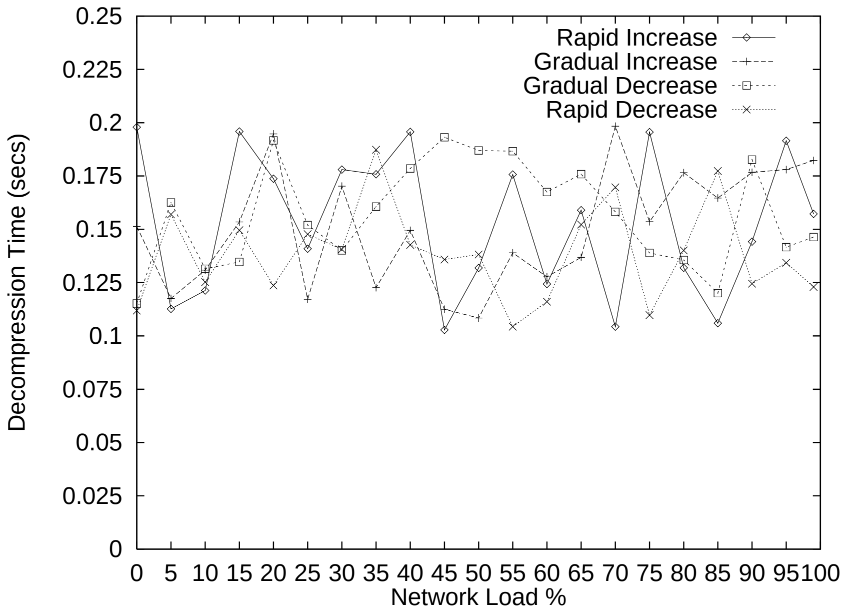


Figure 6.22: Decompression Time for Adaptive MPEG Video Coding Algorithm

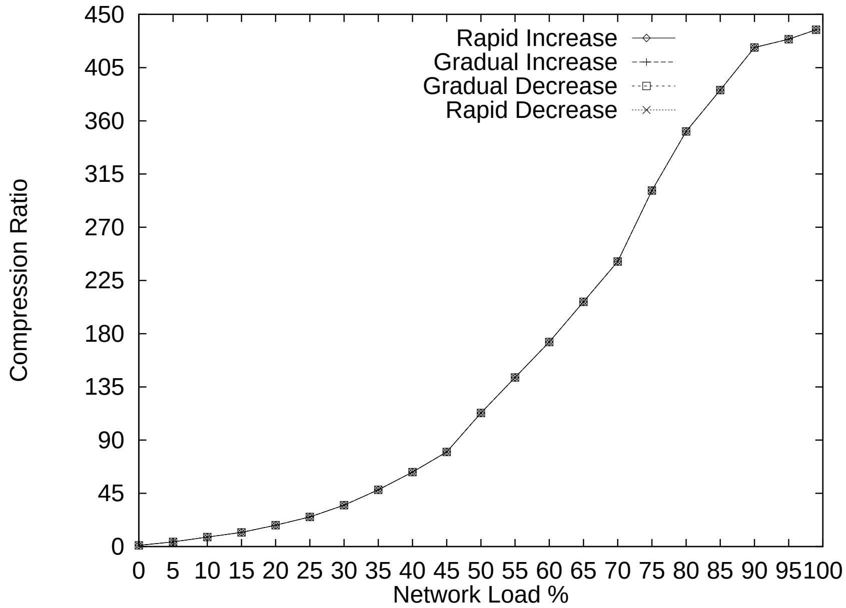


Figure 6.23: Compression Profile for Adaptive MPEG Video Coding Algorithm

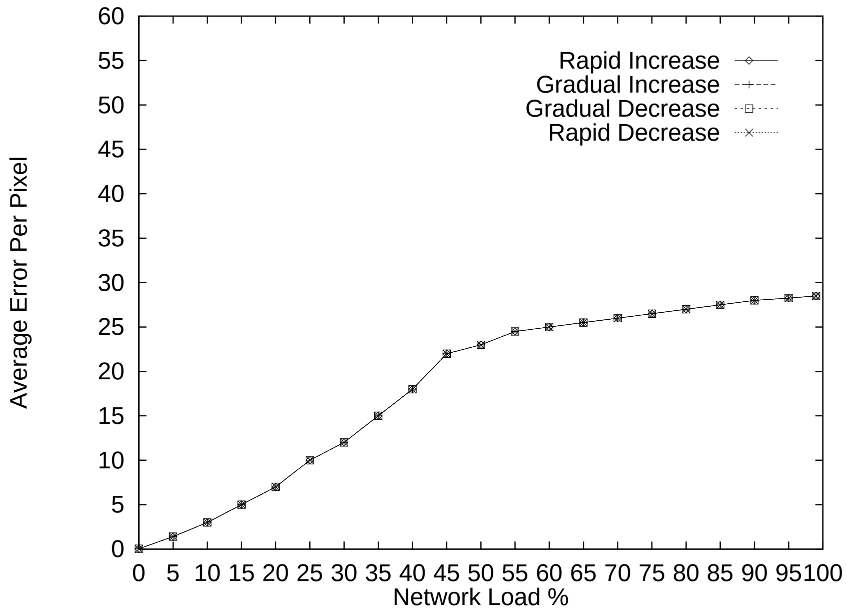


Figure 6.24: Error Curve for Adaptive MPEG Video Coding Algorithm

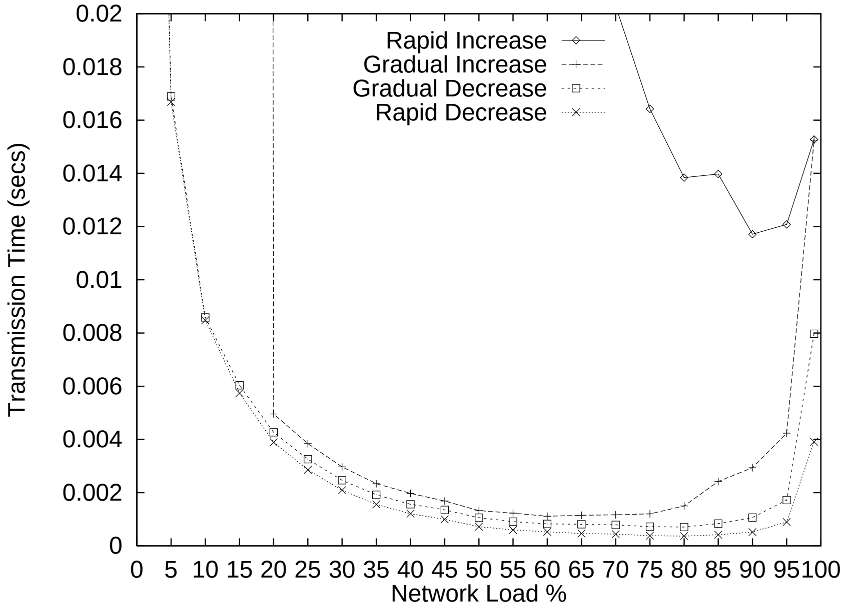


Figure 6.25: Transmission Time Curves for Adaptive MPEG Video Coding Algorithm on a 10 Mbps Network

Chapter 7

Summary and Conclusions

This chapter presents a summary of the work done in this thesis and the conclusions drawn from this work. Future research directions are also suggested.

7.1 Summary

This thesis studies adaptive video coding: the ability to adjust the encoding rate of a video stream (and therefore the bandwidth required) dynamically based on the amount of bandwidth available in the network.

There were three main objectives to this research. The first objective was to determine if adaptive video coding algorithms are robust to changes in network load. The second objective was to determine the load conditions and types of networks on which adaptive video coding algorithms are feasible. The third objective was to evaluate four specific algorithms with respect to their adaptivity on high speed networks and to see if adaptive interframe video coding algorithms have similar characteristics as adaptive intraframe video coding algorithms.

Some progress has been made on each of these objectives. A simple mathematical representation of the end-to-end image transfer problem was derived in order to better understand the effect of network load changes on time taken to receive an image. Six hypothetical video coding algorithms with compression profiles similar to existing algorithms were studied analytically in order to determine the load conditions and network speeds on which adaptive video coding algorithms are feasible. Experiments were conducted with four existing video coding algorithms (JPEG [77], EPIC [1], Fractal [8] and MPEG [40]) to evaluate their adaptivity and determine if adaptive interframe video coding algorithms have the same characteristics as adaptive intraframe video coding algorithms.

The analytical study of the hypothetical video coding algorithms led to the following observations:

- Adaptive video coding algorithms are robust across a wide range of network loads (20% to 80%) and can provide more consistent image transmission time than non-adaptive video coding algorithms by reducing jitter and guaranteeing certain maximum bounds on the image transfer time. A non-adaptive video coding algorithm may provide lower transmission times than an adaptive version of the same algorithm (by compressing the image at the highest possible compression), but the image is often of lower quality.

- Under the test conditions, adaptive video coding algorithms are advantageous at moderate to heavy levels of network load. At low network load (less than 20%) the adaptive video coding algorithms do not outperform non-adaptive video coding algorithms by enough to merit the complexity of adaptivity. For very high network loads (greater than 80%) the performance of adaptive and non-adaptive video coding algorithms is about the same (and very poor).

- With the compression scenario under consideration, adaptive video coding algorithms are best suited for 10 Mbps and 100 Mbps networks. Networks of speed lower than 10 Mbps do not possess the raw bandwidth required to support video applications. Gigabit networks possess more than enough bandwidth to support video applications even without significant compression.

The experiments with existing video coding algorithms led to the following observations:

- Adaptivity on a time scale shorter than a per frame basis does not offer significant improvement in performance over adaptivity on a per frame basis. Therefore adaptivity on a granularity smaller than a per frame basis is not worth the complexity.

- Adaptive video coding algorithms are robust and are able to keep the transmission time below the delivery time constraints of real-time video (33 msec/frame).

- Adaptive interframe video coding algorithms have similar characteristics as adaptive intraframe video coding algorithms.

7.2 Conclusions

There are three main conclusions from this research.

First, adaptive video coding algorithms are advantageous whenever the network is a shared resource, with other users competing for the bandwidth. In this situation, adaptive video coding algorithms provide robustness by keeping the transmission times below certain limits while providing better picture quality than non-adaptive algorithms.

Second, with the compression profile considered, the useful domain of adaptive video coding algorithms is fairly narrow. Networks of speeds 10 Mbps and 100 Mbps are targeted for adaptive video coding algorithms. Networks of speeds lower than 10 Mbps do not possess the raw bandwidth required for video applications, and gigabit networks can support video applications even without much compression. Also adaptive video coding algorithms are advantageous only at moderate to heavy levels of network load (20% to 80%). At very heavy and low network loads, the performance of adaptive video coding algorithms does not merit the additional complexity of adaptivity.

Third, the algorithms under consideration (with the exception of the fractal video coding algorithm) are good at adaptivity. All the algorithms under consideration are able to maintain the network transmission time below certain levels across a wide range of network load. The fractal video coding algorithm is not suitable for adaptivity because of the high compression and decompression times which make the algorithm extremely sluggish in responding to network load changes. Thus in addition to providing high levels of compression, with good image quality, adaptive video coding algorithms should have low compression and decompression times.

7.3 Future Research

The main thrust of this research was to determine if there is any advantage to adaptive video coding, and the conditions under which adaptive video coding algorithms are feasible. Many issues associated with adaptive video coding algorithms still need to be resolved.

The first such issue is how and when to trigger adaptation. One of the assumptions made throughout this research was that the information regarding network load is available to the encoder instantaneously. This assumption was made to simplify the experimental design, but the issue of how the encoder gets information on the network load is critical. There are two approaches to solving this problem: predictive load estimation and feedback-based load information.

The predictive approach estimates the network load at any instant from the network load at the previous time instant. The advantage of predictive load estimation is that the design is simple and the encoder has the load information almost instantaneously. The disadvantage to the predictive approach is that the estimation of network load may not be very accurate.

The feedback-based approach uses some type of feedback from the network regarding network load level. The advantage of the feedback approach is that the information on the network load is more reliable than the predictive approach. The disadvantage is that if there is congestion in the network, the feedback information would be delayed and the encoder will not have a realistic estimate of the network load level. It would also be of interest to see how the feedback and predictive load estimation approaches work on a real B-ISDN network.

A second issue is how to integrate adaptive video coding with congestion control. It would be interesting to explore how a congestion control scheme like traffic shaping works with adaptive video coding. Another approach (priority based) could assign a higher priority to important packets and lower priority to less important packets from the adaptive video coding algorithm. At points of congestion on the network the lower priority packets can be dropped making sure that the higher priority packets reach the receiver.

A third issue is to study alternatives to adaptation for non-real-time video applications. For example, while telecasting a movie if enough storage is available at the receiver then the movie could be telecast at times of low network loads and buffered at the receiver. The tradeoff is that the user has to request the movie in advance (not interactively). The issue of who is to pay for the storage also has to be settled.

A final issue is hardware support for adaptive video coding. Clearly, hardware support is required if encoding and decoding are to be done in real-time. However, software approaches offer the flexibility on which adaptive coding relies. How to strike the correct balance between hardware and software support for adaptive video coding is still an open question.

Bibliography

[1] E. Adelson and E. Simoncelli, "Subband image coding with three-tap pyramids", In *Proceedings of Picture Coding Symposium*, Cambridge, Massachusetts, pp. 1-3, September 1990.

[2] E. Adelson, E. Simoncelli, and R. Hingorani, "Orthogonal pyramid transform for image coding", *Proceedings of SPIE*, Vol. 8, No. 5, pp. 50-58, Cambridge, MA, October 1987.

[3] S. Ahuja and J. Ensor, "Coordination and control of multimedia conferencing", *IEEE Communications Magazine*, Vol. 30, No. 5, pp. 38-43, May 1992.

[4] P. Ang, P. Ruetz and D. Auld, "Video compression makes big gains", *IEEE Spectrum*, Vol. 28, No. 10, pp. 16-19, October 1991.

[5] L. Anson, "Fractal image compression", *Byte*, Vol. 18, No. 11, pp. 195-202, October 1993.

[6] H. Armbruster and K. Wimmer, "Broadband multimedia applications using ATM networks: High performance computing, high-capacity storage, and high-speed communication", *IEEE Journal on Selected Areas in Communications*, Vol. 10, No. 9, pp. 1382-1396, December 1992.

[7] S. Bakshi and Y. Yang, "Shape-from-shading for non-lambertian surfaces", to be published in *International Conference on Image Processing*, Austin, Texas, November 1994.

[8] M. Barnsley and A. Sloan, "A better way to compress images", *Byte*, Vol. 13, No. 1, pp. 215-223, January 1988.

[9] CCITT, Recommendation H.261 prepared by Study Group XV and approved under Resolution No. 2, December 1990.

[10] Y. Chen, K. Sayood and D. Nelson, "A robust coding scheme for packet video", *IEEE Transactions on Communications*, Vol. 40, No. 9, pp. 1491-1502, September 1992.

[11] H. Chin, J. Goodge, R. Griffiths, and D. Parish, "Statistics for video signals for viewphone-type pictures", *IEEE Journal on Selected Areas in Communications*, Vol. 7, No. 5, pp. 827-832, June 1989.

[12] C. Chow and M. Adachi, "Achieving multimedia communications on a heterogeneous network", *IEEE Journal on Selected Areas in Communications*, Vol. 8, No. 3, pp. 348-359, April 1990.

[13] W. Clark, "Multipoint multimedia conferencing", *IEEE Communications Magazine*, Vol. 30, No. 5, pp. 44-50, May 1992.

[14] J. Crowcroft, S. Hailes, M. Handlry, A. Jena, D. Lewis and I. Wakeman, "Some multimedia traffic characteristics and measurement results", Department of Computer Science, University College London, England, Unpublished manuscript, 1992.

[15] J. Darragh and R. Baker, "Fixed distortion subband coding of images for packet switched networks", *IEEE Journal on Selected Areas in Communications*, Vol. 7, No. 5, pp. 789-800, June 1989.

[16] M. Ekstrom, *Digital Image Processing Techniques*, Academic Press, Inc., 1984.

[17] P. Farrelle, *Recursive Block Coding for Image Data Compression*, New York, Springer-Verlag, 1990.

[18] Y. Fisher, "Fractal image compression", *SIGGRAPH course notes*, 1992.

[19] M. Garrett and M. Vetterli, "Joint source/channel coding of statistical multiplexed real-time services on packet switched networks", *IEEE/ACM Transactions on Networking*, Vol. 1, No. 1, pp. 71-80, February 1993.

[20] N. D. Georganas, "Guest editorial: Multimedia communications", *IEEE Journal on Selected Areas in Communications*, Vol. 8, No. 3, pp. 322-323, April 1990.

[21] M. Ghanbari, "Two-layer coding of video signals for VBR networks", *IEEE Journal on Selected Areas in Communications*, Vol. 7, No. 5, pp. 771-781, June 1989.

[22] H. Gilbert, O. Aboul-Magd and V. Phung, "Developing a cohesive traffic management strategy for ATM networks", *IEEE Communications Magazine*, Vol. 29 , No. 10, pp. 36-45, October 1991.

[23] M. Gilge and R. Gusella, "Motion video coding for packet switching networks-an integrated approach", International Computer Science Institute, University of California, Berkeley, Technical Report, No. 91-065, December 1991.

[24] J. Green, "Evolution of DVI system software", *Communications of the ACM*, Vol. 35, No. 1, pp. 52-56, January 1992.

[25] C. Guglielmo, "Lossy compression standards have extra squeeze appeal: JPEG, MPEG and related technologies maximizing compression", *MacWeek*, Vol. 7, No. 24, pp. 30-31, June 1993.

[26] I. Habib and T. Saadawi, "Multimedia traffic characteristics in broadband networks", *IEEE Communications Magazine*, Vol. 30, No. 7, pp. 48-54, July 1992.

[27] A. Habibi, "Hybrid coding of pictorial data", *IEEE Transactions on Communications*, Vol. 22, No. 5, pp. 614-624, May 1974.

[28] D. Hong and T. Suda, "Congestion control and prevention in ATM networks", *IEEE Network Magazine*, Vol. 5, No. 4, pp. 10-15, July 1991.

[29] T. Hoshi, K. Mori, Y. Takahashi, Y. Nakayama, and T. Ishizaki, "B-ISDN multimedia communication and collaboration platform using advanced video workstations to support cooperative work", *IEEE Journal on Selected Areas in Communications*, Vol. 10, No. 9, pp. 1403-1412, December 1992.

[30] R. Jain, *The Art of Computer Systems Performance Analysis*, John Wiley & Sons, Inc., 1991.

[31] N. Jayant, "High quality networking of audio-visual information", *IEEE Communications Magazine*, Vol. 31, No. 9, pp. 84-95, September 1993.

[32] H. Kanakia, P. Mishra and A. Reibman, "An adaptive congestion control scheme for real-time packet video transport", *Proceedings of ACM SIGCOMM '93*, pp. 20-31, September 1993.

[33] G. Karlsson, "Sub-band coding for packet video", Ph.D. Thesis, Department of Electrical Engineering, Columbia University, 1989.

[34] G. Karlsson and M. Vetterli, "Sub-band coding of video signals for packet switched networks", *Proceedings of SPIE*, Cambridge, MA, pp. 7-11, October 1987.

[35] G. Karlsson and M. Vetterli, "Three dimensional sub-band coding of video", *Proceedings of IEEE ICASSP*, New York, pp. 1100-1103, April 1988.

[36] 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.

[37] A. Karmouch, L. Orozco-Barbosa, N. Georganas, and M. Goldenberg, "A multimedia medical communications system", *IEEE Journal on Selected Areas in Communications*, Vol. 8, No. 3, pp. 325-339, April 1990.

[38] R. Kishimoto, Y. Ogata, and F. Inumaru, "Generation interval distribution characteristics of variable rate video coding data streams in an ATM network",

IEEE Journal on Selected Areas in Communications, Vol. 7, No. 5, pp. 833-841, June 1989.

[39] F. Kishino, K. Manabe, Y. Hayashi, and H. Yasuda, "Variable bit-rate coding of video signals for ATM networks", *IEEE Journal on Selected Areas in Communications*, Vol. 7, No. 5, pp. 801-805, June 1989.

[40] D. LeGall, "MPEG: a video compression standard for multimedia applications", *Communications of the ACM*, Vol. 34, No. 4, pp. 46-58, April 1991.

[41] W. Leung, T. Baumgartner, Y. Hwang, M. Morgan and S. Tu, "A software architecture for workstations supporting multimedia conferencing in packet switching networks", *IEEE Journal on Selected Areas in Communications*, Vol. 8, No. 3, pp. 380-390, April 1990.

[42] T. Little and A. Ghafoor, "Synchronization and storage models for multimedia objects", *IEEE Journal on Selected Areas in Communications*, Vol. 8, No. 3, pp. 413-427, April 1990.

[43] T. Little and A. Ghafoor, "Multimedia synchronization protocols for broadband integrated services", *IEEE Journal on Selected Areas in Communications*, Vol. 9, No. 9, pp. 1368-1381, December 1991.

[44] S. Loeb, "Delivering interactive multimedia documents over networks", *IEEE Communications Magazine*, Vol. 30, No. 5, pp. 52-59, May 1992.

[45] M. Marcellin, P. Sriram, and K. Tong, "Transform coding of monochrome and colour images using trellis coded quantization", *IEEE Transcations on Circuits and Systems for Video Technology*, Vol. 3, No. 4, pp. 270-276, August 1993.

[46] P. Mouchtaris, "Traffic analysis for a custom video architecture", *IEEE Journal on Selected Areas in Communications*, Vol. 10, No. 9, pp. 1421-1426, December 1992.

[47] H. Mouftah, "Multimedia communications: An overview", *IEEE Communications Magazine*, Vol. 30, No. 5, pp. 18-19, May 1992.

[48] S. Nakano, N. Nagai, T. Sawada, and O. Miyagishi, "Network control method and human-machine interface design for ISDN multimedia terminal", *IEEE Journal on Selected Areas in Communications*, Vol. 8, No. 3, pp. 360-367, April 1990.

[49] C. Nicolaou, "An architecture for real-time multimedia communication systems", *IEEE Journal on Selected Areas in Communications*, Vol. 8, No. 3, pp. 391-400, April 1990.

[50] M. Nomura, T. Fuji, and N. Ohta, "Basic characteristics of variable rate video coding in ATM environment", *IEEE Journal on Selected Areas in Communications*, Vol. 7, No. 5, pp. 752-760, June 1989.

[51] L. Orozco-Barbosa, A. Karmouch, N. Georganas and M. Goldberg, "A multimedia interhospital communication system for medical consultation", *IEEE Journal on Selected Areas in Communications*, Vol. 10, No. 7, pp. 1145-1157, September 1992.

[52] C. Paratridge, *Gigabit Netwoking*, Addison-Wesley Publishing Company, 1994.

[53] G. Polyzos, "Hierarchical coding of images and continuous media for transmission over packet-switched networks", USCD Technical Report CS93-1, Computer Science Laboratory, Department of Computer Science and Engineering, University of California at San Diego, January 1993.

[54] W. Pratt, *Digital Image Processing*, John Wiley and Sons, Inc., 1978.

[55] P. Rangan, "Video conferencing, file storage, and management in multimedia computer systems", *Computer Networks and ISDN Systems*, Vol. 25, No. 8, pp. 901-919, August 1993.

[56] P. Rangan and D. Swinehart, "Software architecture for integration of video services in the Etherphone system", *IEEE Journal on Selected Areas in Communications*, Vol. 9, No. 9, pp. 1395-1404, December 1991.

[57] M. Ransom and D. Spears, "Applications of public gigabit networks", *IEEE Network*, Vol. 6, No. 2, pp. 30-40, March 1992.

[58] J. Roberts, "Variable-bit-rate traffic control in B-ISDN", *IEEE Communications Magazine*, Vol. 29, No. 9, pp. 50-56, September 1991.

[59] J. Rosenberg, R. Kraut, L. Gomez, and C. Buzzard, "Multimedia communications for users", *IEEE Communications Magazine*, Vol. 30, No. 5, pp. 20-36, May 1992.

[60] L. Rowe, K. Patel and B. Smith, "MPEG video software encoder", Computer Science Division-EECS, University of California at Berkeley, November 1992, (Sofware implementation and documentation).

[61] S. Sakata, "Development and evaluation of an in-house multimedia desktop conference system", *IEEE Journal on Selected Areas in Communications*, Vol. 8, No. 3, pp. 340-347, April 1990.

[62] R. W. Scheifler and J. Gettys, "The X Window System", *ACM Transactions on Graphics*, Vol. 5, No. 2, pp. 79-109, April 1986.

[63] H. Shimizu, M. Mera, and H. Tani, "Packet communication protocol for image services on a high-speed multimedia LAN", *IEEE Journal on Selected Areas in Communications*, Vol. 7, No. 5, pp. 782-799, June 1989.

[64] B. Simon, "How lossy compression shrinks image files", *PC Magazine*, Vol. 12, No. 13, pp. 371-376, July 1993.

[65] E. Simoncelli, "Orthogonal sub-band image transforms", M. Sc. Thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, 1988.

[66] W. Sirotta, "Compression engines", *PC World*, Vol. 11, No. 11, pp. 66-72, November 1993.

[67] R. Steinmetz, "Synchronization properties in multimedia systems", *IEEE Journal on Selected Areas in Communications*, Vol. 8, No. 3, pp. 401-412, April 1990.

[68] A. Syed, "Quantative evaluation of video coding algorithms for ATM-based networks", M.Sc. Thesis, Department of Computational Science, University of Saskatchewan, October 1993.

[69] T. Tanaka, H. Haseyama, K. Manabe, F. Kishino, "Bit-rate control algorithm on video packet coding and its characteristics", *Signal Processing: Image Communications*, Vol. 3, No. 4, pp. 321-331, September 1991.

[70] S. Terves, E. Hashem, B. Majumdar, K. Mitchell and D. Michaud, "Multimedia communications in medical imaging", *IEEE Journal on Selected Areas in Communications*, Vol. 10, No. 7, pp. 1145-1157, September 1992.

[71] L. Trajkovic and S. Golestani, "Congestion control for multimedia services", *IEEE Network Magazine*, Vol. 6, No. 5, pp. 20-26, September 1992.

[72] J. Turner, "Managing bandwidth in ATM networks with bursty traffic", *IEEE Network Magazine*, Vol. 6, No. 5, pp. 50-58, September 1992.

[73] C. Turner and L. Peterson, "Image transfer: an end-to-end design", *Proceedings of ACM SIGCOMM '92*, pp. 258-268, August 1992.

[74] O. Vega and Y. Yang, "A heuristic approach to recover shape from shading", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 15, No. 6, pp. 592-597, June 1993.

[75] W. Verbiest and L. Pinnoo, "A variable bit rate video codec for asynchronous transfer mode networks", *IEEE Journal on Selected Areas in Communications*, Vol. 7, No. 5, pp. 761-770, June 1989.

[76] I. Wakeman, "Packetized video: optimal for interaction between the user, network and codec", Department of Computer Science, University College London, England, Unpublished manuscript, 1992.

[77] G. Wallace, "The JPEG still image picture compression standard", *Communications of the ACM*, Vol. 34, No. 4, pp. 31-44, April 1991.

[78] M. Wernik, O. Aboul-Magd, and H. Gilbert, "Traffic management for B-ISDN Services", *IEEE Network Magazine*, Vol. 6, No. 5, pp. 10-19, September 1992.

[79] G. Woodruff and R. Kositpaiboon, "Multimedia traffic management principles for guaranteed ATM network performance", *IEEE Journal on Selected Areas in Communications*, Vol. 8, No. 3, pp. 437-445, April 1990.

[80] D. Wright, "Strategic impact of broadband telecommunications in insurance, publishing and health care", *IEEE Journal on Selected Areas in Communications*, Vol. 10, No. 9, pp. 1369-1381, December 1992.

[81] M. Zafirovic-Vutotic and I. Niemegeers, "Multimedia communication systems: Upper layers in the OSI reference model", *IEEE Journal on Selected Areas in Communications*, Vol. 10, No. 9, pp. 1397-1402, December 1992.

[82] C. Ziegler and G. Weiss, "Multimedia conferencing on local area networks", *IEEE Computer*, Vol. 23, No. 9, pp. 52-61, September 1990.

[83] W. Zou "Digital HDTV compression techniques", *IEEE Transactions on Broadcasting*, Vol. 37, No. 4, pp. 131-133, December 1991.

Glossary

ABR Available Bit Rate, one of the proposed service classes for ATM/B-ISDN. The user is guaranteed a certain minimum bit rate and is assigned whatever additional bandwidth is available whenever it is available.

Adaptivity The ability of a video coding algorithm to change the encoding of a video stream (and thus its bandwidth requirements) dynamically based on the amount of bandwidth available in the network.

Asymmetric Compression Compression strategy in which compression time is much higher than the decompression time.

ATM Asynchronous Transfer Mode is a connection-oriented, packet switching and multiplexing technique for B-ISDN networks in which all packets are of a constant size of 53 bytes.

Bandpass Filter A filter that allows a specific frequency (the one it is designed for) through and blocks all other frequencies.

B-ISDN A high speed digital network ($\geq$ 150 Mbps) capable of supporting voice, video and data applications in a flexible, cost effective and integrated manner.

Burstiness The ratio of the peak bit rate of a traffic source to the mean bit rate.

Chrominance The colour level in an image.

Codec An assembly comprising an encoder and a decoder in the same equipment.

Coding Conversion of an analog function to a digital form using a specific set of rules.

Compression Profile The dynamics of the change in the compression ratio by varying factors like frame rate, colour information, image size or the parameters of a video coding algorithm.

Compression Ratio The ratio of the size of original image to the size of compressed image.

Compression Time Time to compress the image.

Decompression Time Time to decompress the image.

Fractal A video compression algorithm proposed by Adelson *et al.* [1].

Frame One image in a sequence of images.

Full Colour 24 bits of colour information per pixel.

Full Motion Video Video displayed at 30 frames per second.

Grayscale 8 bits of colour information per pixel.

H.261 Standard A standard proposed by CCITT for video teleconferencing [9].

Huffman Coding A statistical encoding scheme that uses short representations for components that occur more often, and longer representations for components that occur less often.

Hybrid Coding Combination of transform coding and spatial domain coding.

Jitter the variations in the delay with respect to the value desired for such delays. For example, if a packet is transmittedat time t_0, the network might guarantee that the packet will arrive at a time $t_0 + x$. In most cases, x is not an exact value, but rather a value plusor minus some small value. The amount by which x can vary is called *jitter* [52].

Lossless Compression Compression without any loss of information, such that the reconstructed image is identical to the original image.

Lossy Compression Compression with some loss of information, such that the reconstructed image is not identical to the original image.

Luminance The black and white levels in an image.

Multimedia A diverse mix of media like video, voice and data.

End-to-end Transfer Time Time taken to compress, transmit and decompress the image.

NTSC Committee which proposed the standard for video transmission for television at 30 frames per second in North America.

Pixels (Picture Elements) The smallest resolution in any graphical image.

Real-Time Pertaining to the actual time during which a physical process transpires.

Sampling The process of sensing a waveform's amplitude at specific instants of time. Sampling is usually done periodically.

Spatial Domain Coding A compression technique that looks at neighbouring regions of the region under consideration.

Sub-band Coding The process of breaking up an image into frequency bands and coding each frequency band separately.

Symmetric Compression The compression and decompression processes take about the same time.

Synchronization Two signals are synchronous if their corresponding significant instants have a desired phase relationship (CCITT).

Traffic The messages sent and received over one or a group of communication channels.

Transform Coding Process which encodes an image by applying a mathematical transform and encoding the coefficients of the transform.

Transmission Time Time taken for an image to be transmitted over the network.

Video Clip A sequence of successive images.

Video Coding Algorithms Algorithms that reduce the size of an image by encoding patterns and redundancies in the image.

YUV An image format that represents each pixel in terms of its luminance (Y) and chrominance (U/V) levels.