SCALABLE VIDEO

By

Ying Lee

A thesis submitted in partial fulfillment of
the requirements for the degree of

Master of Engineering in Electrical
Engineering and Computer Science

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 2000
June 2000

Author _____
Department of Electrical Engineering and Computer Science.
May 17, 2000

Certified by _____
Andrew B. Lippman
Lecturer, Associate Director, Media Laboratory
Thesis Supervisor

Accepted by _____
Arthur C. Smith
Chairman, Department Committee on Graduate Theses

MASSACHUSSETS INSTITUTE OF

TECHNOLOGY

Abstract

SCALABLE VIDEO

by Ying Lee

Chairperson of the Supervisory Committee: Professor Andrew B. Lippman
Lecturer, Associate Director, Media Laboratory

This thesis presents the design and implementation of a scalable video scheme that accommodates the uncertainties in networks and the differences in receivers' displaying mechanisms. To achieve scalability, a video stream is encoded into two kinds of layers, namely the base layer and the enhancement layer. The decoder must process the base layer in order to display minimally acceptable video quality. For higher quality, the decoder simply combines the base layer with one or more enhancement layers. Incorporated with the IP multicast system, the result is a highly flexible and extensible structure that facilitates video viewing to a wide variety of devices, yet customizes the presentation for each individual receiver.

TABLE OF CONTENTS

LIST OF FIGURES

Number Page

ACKNOWLEDGMENTS

The author wishes to thank Professor Andy Lippman for his great insights, guidance, and support for the work done in this thesis.

Thanks to Dean Chen and Dean Christakos for their expertise on the IP multicast system. And special thanks to Stefan Agamanolis for his help with ISIS. More thanks to everyone in the Garden of the Media Lab who made this research quite enjoyable.

Most importantly, thanks to the author's family for their endless love, encouragement, and support.

GLOSSARY

BIFS. Binary format for Scenes.

DAI. DMIF Application Interface.

DCT. Discrete Cosine Transform.

DMIF. Delivery Multimedia Integration Framework.

DNI. DMIF Network Interface.

ES. Elementary Streams in MPEG-4.

IOC. Inter Object Control.

IP. Internet Protocol.

JPEG. The popular image compression standard developed by the Joint Photographers Experts Group.

MPEG. The international video compression standard developed by the Moving Picture Experts Group.

QoS. Quality of Service.

UDP. User Datagram Protocol.

Chapter 1

INTRODUCTION

Background

Today, the power of Internet penetrates more than 86% of American households [1] and enables people to access an enormous wealth of information. The challenge now is to make information easily retrievable for a variety of systems. To reach this goal, information must be customized in accordance with the clients' networks and the features of their devices. Many uncertain parameters exist in a network, such as the speed, load, and bandwidth. Similarly, the wide variety of devices in the digital world ranges from desktops to mobile phones.

An especially thought-provoking task lies in delivering customized video streams over the network because of the huge amount of data involved. If the video content is encoded in high quality, people with low processor power may be frustrated by the wait for data transfer; if the video content is encoded in low quality, people with fast connections and machines may be annoyed by the fuzzy picture frames; finally, if the video content is encoded in medium quality, both low end and high end users may feel discontented.

The solution clearly does not lie in choosing the "perfect" quality rate, but in scaling the streams so that everyone can be pleased. Currently, certain video encoders and decoders provide scalability, such as Windows Media. However,

most of them are not inherently scalable because the common method encodes the same video stream into multiple files of different qualities. Based on the information a client provides, the server then sends the appropriate file. Although video scalability is achieved, this is not a long-term solution because a rise in the number of devices or quality levels may require an increase by magnitudes in the storage space and the number of video streams.

Motivation

Inspired by Dean Chen's powerful IP Multicast system, the scalable video scheme in this research will enhance the whole system's extensibility and flexibility. The basic concept behind the IP Multicast system is to deliver video packets only to clients who choose to receive them. Instead of sending packets of data to every host as in broadcast, multicast provides hosts with the option to ignore certain data packets. As a result, multicast saves both processing power and time wasted on handling unwanted packets. Although unicast is the predominant form of network communication today, it is inefficient because network traffic increases linearly with the number of hosts. In a multicast system, for any number of receivers, only one copy of the original video is required. The system also accommodates a varying number of receivers by utilizing the concept of "smart caches," which enable machines to share data locally. Because computers with cached video contents can function as servers to machines that request this content, the number of users in the system can expand easily. Its main shortcoming is the lack of ability to customize video presentation for uncertainties in both the network and the displaying device.

Today, the value of the scalable video feature is more apparent with the ever-increasing varieties of devices and uncertainties in networks. Due to the speed of technological innovation, the coding scheme must be easily adaptable to the different transmission rates and physical devices that are certain to appear in the future. If a system has to send different copies of the same video over the network to compensate for the uncertainties mentioned earlier, then the network traffic will worsen. On the other hand, with a system that encodes a video stream into separate quality layers, a different subset of the layers can create a distinct picture quality. In this scheme, the sizes of all layers summed together approximate that of the original file. The layering scheme will not only reduce network traffic, but also provide added flexibility. For example, to achieve a new quality level, the decoder simply selects a different subset of layers to process. Without the layering scheme, all existing video contents may need to be encoded again at the new quality level. This scheme also provides enhanced performance. In the multicast system where one server sends data only to subscribers, there are different channels with one layer for each channel. As a result, bad transmission in one channel may not prevent the client from enjoying the movie because other channels also carry information about the movie. Having different channels simulates distributing information to several sources; when one source fails, the remaining ones will continue to function properly.

Goals

The goal of this thesis is to design, implement, and test a scalable video in the multicast system framework. The prototype will demonstrate its ability to multicast videos of different qualities to receivers based on their network and

machine constraints. In the prototype, the encoder should separate a single video file into different layers. The decoder would process two kinds of layers, a base layer that displays the minimally acceptable quality of the original file, and one or more enhancement layers that improve the video quality during the playback. Decoding all layers would enable viewing of the original video quality or the highest possible quality. In the multicast framework, each data channel serves one bit stream produced by the encoder. Then, the receivers can subscribe to a subset of layers based on their processing speed, network bandwidths, and other relevant parameters. As a result, different video qualities would be displayed depending on the receivers' networks and systems while multicasting the same set of layers.

Thesis Overview

Chapter 2 introduces some basic concepts underlying the digital signal compression schemes. After an explanation of image processing, Chapter 3 presents an overview of the latest video-coding standard, MPEG-4, developed by the International Standard Organization (ISO) Motion Picture Expert Group (MPEG). There are two versions of MPEG-4, but the fundamental design and architecture principles apply to both versions. Next, Chapter 4 explores the different design considerations for scalable video coding in the multicast system context. Subsequently, Chapter 5 presents the design decisions and implementation details of the prototype. After that, Chapter 6 discusses potential future extensions to the scalable video schemes and the multicast system. Finally, Chapter 7 concludes the thesis.

Chapter 2

SIGNAL COMPRESSION

Compared to analog signals, digital signals can be easily stored and analyzed by computers. They can also be transmitted directly over a digital communication link, such as telecommunications, T1, and optical fiber. Public databases often archive digital signals as well. As a result, digital signals become parts of a wide variety of applications. However, they are not perfect. The main problem lies in their enormous data sizes, which often clog low capacity memory and congest network traffic.

The most effective solution to reduce the large file size lies in data compression, which decreases the storage space for the same amount of data. There are two general types of compression. The first type is called *lossless compression*, which allows perfect recovery of the original digital data. Certain types of information such as executable code and bank statements must be compressed without losing any data because accuracy is the vital. In comparison, other files such as images can tolerate minor losses in data and thus, can be compressed using *losy compression* schemes [7]. To illustrate the performance differences among the formats, consider an uncompressed TIFF image file of size 600 Kbytes. When compressed in lossless GIF format, it is 300 Kbytes, which is half of the original size. With lossy compression such as JPEG, the size would be reduced to 50 Kbytes. The file download time for the three formats is 142 seconds, 71 seconds, and 12 seconds respectively [5]. These are huge differences when several images need to be loaded. The rest of this section describes the two types of

compression schemes in detail. A general understanding of the concepts in signal and image compression is essential for the MPEG-4 video compression standard explained in the next chapter.

Lossless Compression

The main idea behind lossless compression is to obtain a minimized average. This is achieved by coding highly probable symbols into short binary sequences while low probability symbols into long binary sequences. The followings are commonly used algorithms for lossless compression:

- Run-length.

  Data files often contain repeated characters consecutively. For example, an image of the ocean would contain long runs of letters representing blue as the water color. Instead of re-writing the blue character many times, run-length writes it only once along with the number of iterations for the character. For example, if there are seven 31's in a row, instead of storing 31 31 31 31 31 31 31, run-length stores (31 7) representing 31 repeated seven times [5].

- Huffman.

  This compression technique is named after D.A. Huffman who observed that the majority of a file consists of only a small subset of all possible characters. For example, in most written documents, the most commonly used characters comprise more than 95% of the documents, such as

space, carriage return, period, comma, and lower case letters. Based on this observation, Huffman cleverly encode the most frequently occurring characters with very few bits. For characters that are rarely used, Huffman represent them with more bits. By doing so, this algorithm yields the optimum lossless compression by providing the minimum average length over all uniquely decodable characters. However, this scheme requires a priori knowledge of the input [5].

- Arithmetic.

  It is a more sophisticated version of the Huffman code. In this scheme, sequences of characters are represented by individual codes according to their probability of occurrence. Statistics show that this method has 5 to 10% better data compression rate than Huffman coding. Similar to Huffman coding scheme, the Arithmetic technique assumes a priori knowledge of the input probabilities, which is not always true in practice [5].

- Delta.

  The Greek letter delta denotes the change in a variable. Delta encoding refers to techniques that store data as the difference between successive characters rather than directly storing the characters themselves. This coding scheme is especially useful when the values in the original data are smooth or changes slightly between adjacent values. Although this is not true for ASCII texts and executable codes, it is very common when the file represents a signal. Delta encoding has increased the probability that each sample's value will be near zero and decreased the probability that it will be far from zero. As an example, for a smooth signal at around 10K,

delta encoding would have amplitudes that range from 0 to 10 as opposed to the original signal that varies from 10,000 to 10,010 [5].

- LZW.

  This simple and versatile compression is named after Lempel-Ziv, who pioneered the technique for general-purpose data compression. LZW compresses text and executable code to about half of their original size. When used on extremely redundant data files, its compression ratio increases to 5:1. LZW is sometimes called the dictionary code because the key idea is to recursively parse input sequence into blocks of variable size while constructing a dictionary of blocks seen thus far. The dictionary consists of two components, the code number and its corresponding translation. Generally, codes 0 to 225 in the table always represent single bytes from the input file while codes 256 to 4095 correspond to sequences of bytes. The main source of compression comes from representing long stream of codes with a single code number. For example, code 512 may represent a sequence of three bytes: 251 223 423. The same table will be used during decompression to translate the code number back to the original data [5].

Lossy Compressions

Lossy compression is not invertible because information is lost in the coding process. The main advantage in this type of compression is its high reduction ratio. For example, it is typical to achieve 16:1, and the ratio can be 100:1 for

some applications. The main concept in lossy compression is uniform scalar quantization, which rounds numbers to the closest ones in a predefined set.

- JPEG.

  Named after its origin, the Joint Photographers Experts Group (JPEG) is one of the most popular standard for image encoding. It is based on the idea of courser sampling and quantization. To reduce the file size directly by either decreasing the number of bits per sample or discarding some of the samples entirely will obviously yield poor results. The strategy is to perform Discrete Cosine Transform (DCT) on the signals, so the resulting data values will be of different significance. Unlike text documents, smooth data values comprise the majority of pictures. In signal terms, high frequency values represent sharp edges while low frequency value represent smooth data. Thus, the low frequency components of a signal are more important than the high frequency components. As a result, removing 50% of the bits from the high frequency components may only remove 5% of the encoded information [4].

  JPEG compression starts by breaking the image into 8x8 groups. Each group contains 64 pixels, and each pixel can possess different numbers of bits. The groups are treated independently during compression. After transforming and removing data, each resulting group can be represented by different bytes. The followings are the encoding steps.

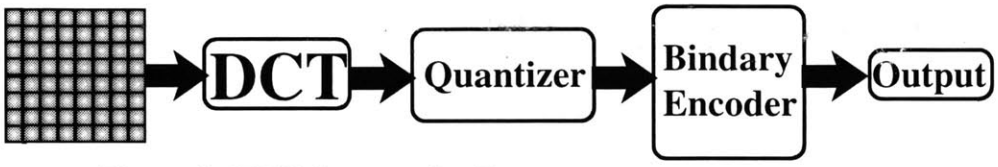

Figure 1: JPEG Compression Process

1. The image is divided into 8x8 pixel blocks.

2. Discrete Cosine Transform is performed on each block.

3. Each 8x8 spectrum is compressed separately. To decrease the size of a file, the number of bits is reduced and some components are eliminated completely. This step is called the uniform scalar quantization because the DCT coefficients are rounded off according to the quantization matrix. Each value in the spectrum is divided by the matching value in the quantization table. Then the result is rounded to the nearest integer. For example, in the sample quantization matrix below, the first value from the left is 1, leaving in the DC value unchanged. The lower-right entry is 16, which means that the original range of –127 to 127 is reduced to only –7 to 7. In other words, the value has been reduced in precision from 8 bits to 4 bits. As a result, this process produces the "lossy" nature of JPEG and also the large compression ratios.

1	1	1	1	1	2	2	4
1	1	1	1	1	2	2	4
1	1	1	1	2	2	2	4
1	1	1	1	2	2	4	8
1	1	2	2	2	2	4	8
2	2	2	2	2	4	8	8
2	2	2	4	4	8	8	16
4	4	4	4	8	8	16	16

Figure 2: A sample quantization table. It reduces the precision from 8 bits to 4 bits.

4. The modified spectrum is then converted from an 8x8 array into a linear sequence following a serpentine pattern as shown in figure 3. This path clusters the high frequency components at the end of the linear sequence.

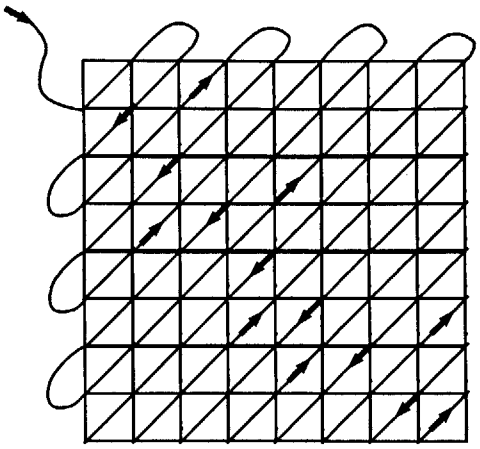

Figure 3: Serpentine Motion. This pattern is used during the transformation of the 8x8 array to a linear sequence.

5. Because the previous step groups sequences of 0's from the eliminated components, those data can be further compressed by run-length encoding and Huffman or Arithmetic coding to produce the final compressed data file.

During decompression, JPEG recovers the qunatized DCT coefficients from the compressed data stream. First, it takes the inverse transform on each block to produce an approximation of the original 8x8 group. Then, these approximated groups are then fitted together to form the uncompressed image. Finally, the resulting image is displayed [4].

- MPEG.

This international standard is characterized by block transformation and motion compensation. Developed by the ISO Moving Picture Experts Group, the essential concept behind MPEG video compression is to remove spatial redundancy within a video frame and temporal redundancy between frames. As in JPEG, DCT-based compression is used to reduce spatial redundancy while motion-compensation is applied

to exploit temporal redundancy. Because images in a video stream usually do not change much within small time intervals, the idea of motion-compensation is to encode a video frame based on other frames temporally close to it [2].

It is the compression standard generally used for digital video sequences in computer video and digital television networks. MPEG also provides compression for the sound track associated with the video. It has numerous advanced features such as playing forward or backward at normal or fast speed. Additionally, the encoded information can be randomly accessed. In other words, an individual frame in the sequence can be easily displayed as a still picture. Established in 1992, the MPEG-1 standard is intended for VHS quality video. The main objective then is to produce reasonable quality images and sound at low bit rates, such as 352x240 images at 1.5 Mbits/second. Emerged in 1994, the MPEG-2 standard puts more emphasis on producing high quality images at higher bit rates, such as 720x485 studio quality images at 45 Mbits/second for HDTV resolution. The most current MPEG-4 standard will be explained in Chapter 3 [2].

MPEG scheme consists of two main types of compressions: within-frame and between-frame. A video stream is simply a sequence of picture frames playing at 29.79 frames per second. Within-the-frame compression encodes frames as if they were ordinary still images. These are called *I-pictures* or intra-coded, and are generally coded with JPEG based standard.

Unless the camera is moving, most images compose of a background that remains constant over several frames. Because of this observation, MPEG applies a sophisticated form of delta encoding to compress the

redundant information between frames. Generally, after compressing one of the frames as an I-picture, MPEG encodes successive frames as predictive-coded or *P-pictures*, where only the pixels that have changed since the previous I-picture are included in the P-pictures [2].

Although MPEG is based on those two compression schemes, its actual implementation is much more complicated. For example, a P-picture can be referenced to a shifted version of an I-picture to account for the motion of objects in the image sequence. To make the coding even more efficient, there are also bi-directional predictive-coded or *B-pictures*. These reference to both a previous and a subsequent I-picture to compensate for regions in the image that gradually change over many frames. The system is further complicated with color and sound.

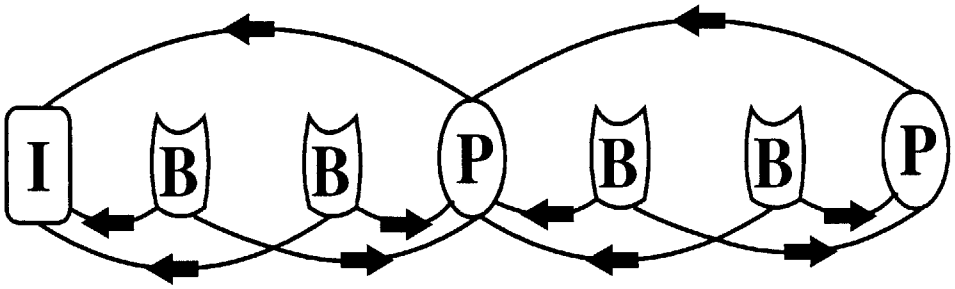

Figure 4. Relations among the different pictures.

The main distortion associated with MPEG occurs when large sections of the image change quickly. Fixed data rate cannot preserve the rapidly changing scenes that require a burst of information. As a result, viewers would notice "blocky" patterns when changing from one scene to the next [7].

- H.263.

This coding algorithm is designed for video conferencing applications. It uses a hybrid of inter-picture prediction, transform coding, and motion compensation to support data rates ranging from 40 Kbits/second to 2 Mbits/second. Similar to MPEG, H.263 uses inter-picture prediction to remove temporal redundancy. Additionally, it utilizes transform coding to exploit spatial redundancy while motion vectors to compensate for objects' displacements between frames. To further minimize the redundancy in the transmitted bit stream, variable length coding is performed.

In a lossy environment, an H.263 system can be adopted to handle the dropping of packets. A decoder can signal its preference for a certain tradeoff between spatial and temporal resolution of the video signal. Temporal sub-sampling is performed by discarding complete picture frames. The decoder can also choose not to update macroblocks in a lost or corrupted packet. In this case, the decoder sends signals to the encoder in regards to the disgarded macroblocks. Responding to the decoder's feedback, the encoder adjusts the local decoder's buffer accordingly to synchronize the local decoder and decoder. This method curtails potential long-term image degradation [3].

H.263 works well in a point-to-point setting. However, in a multicast or broadcast situation, the encoder would not be able to maintain consistent states for multiple receivers. A video coding scheme that is suitable in a broadcast and lossy environment must enable the decoder to recover from a drift with an intra-frame coded picture, such as the MPEG video compression technique explained earlier.

Chapter 3

MPEG-4

Overview

MPEG-4 emphasizes object based coding. It allows user interactivity such as composition, manipulation, and transformation on both synthetic and natural multimedia. Convergence is certain to appear in the future. Hence, MPEG-4 establishes the first standard to present, manipulate, and transport individual media elements. Following the technological trend, single terminal will be replaced by a proliferation of multimedia services over a variety of access networks. MPEG-4's main objective is to develop a multimedia technology that will support the three most common types of services: interactive, broadcast, and conversational (see later description of DMIF). To achieve this goal, MPEG-4 allows independent compression schemes for each audio-visual object. The following outlines the possible objects.

- Audio (single or multi-channel) or video (arbitrarily shaped or rectangular)
- Natural (audio or video) or synthetic (text, graphics, animated faces, or synthetic music)

- 2D (Web pages) or 3D (spatial sound, 3D virtual world)
- Streamed (video movie) or downloaded (audio jingle)

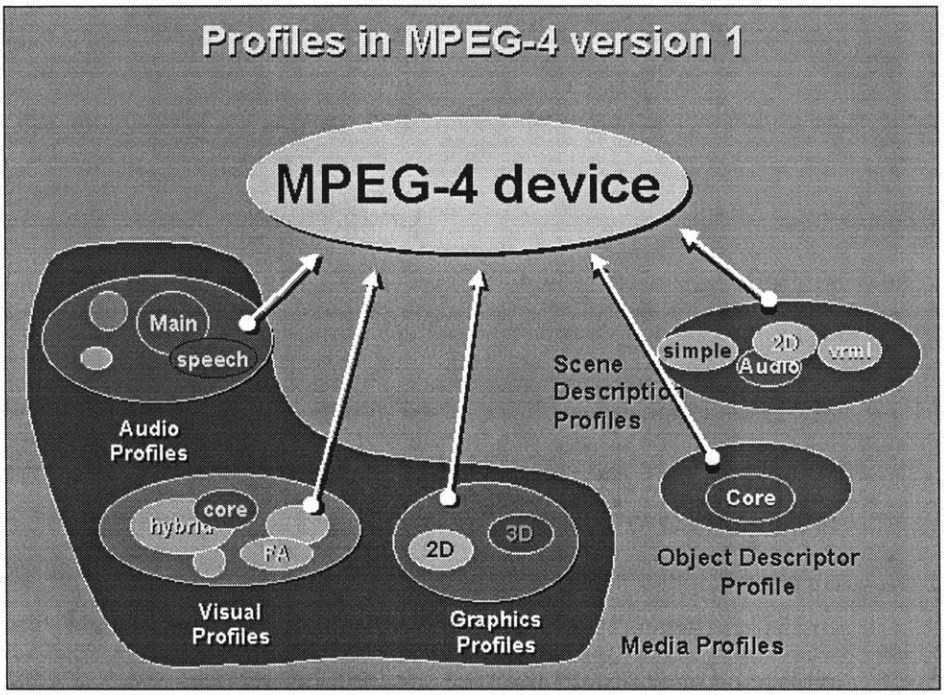

Figure 5. MPEG-4 Components [9].

An audio-visual scene is composed of one or more audio-visual objects arranged according to a scene description. Even though the coding scheme may be different for the individual objects, they can still interact with each other within the same scene. A scene description may include the following details.

1. Spatial and temporal relationship between the audiovisual objects.
2. Behavior and interactivity of the audio-visual objects and scenes.
3. Protocols to modify and animate the scene in time.

Furthermore, this scheme allows for handling hierarchically encoded data, associating meta-information about the content, and providing the intellectual property rights associated with each element. The protocol used for a scene description is called the Binary Format for Scenes (BIFS) [9], which sets rules to manipulate the MPEG-4 objects. For this research, the architecture and video schemes in the MPEG-4 system are of particular interest.

Architecture

The diagram below illustrates the important modules in delivering MPEG-4 video.

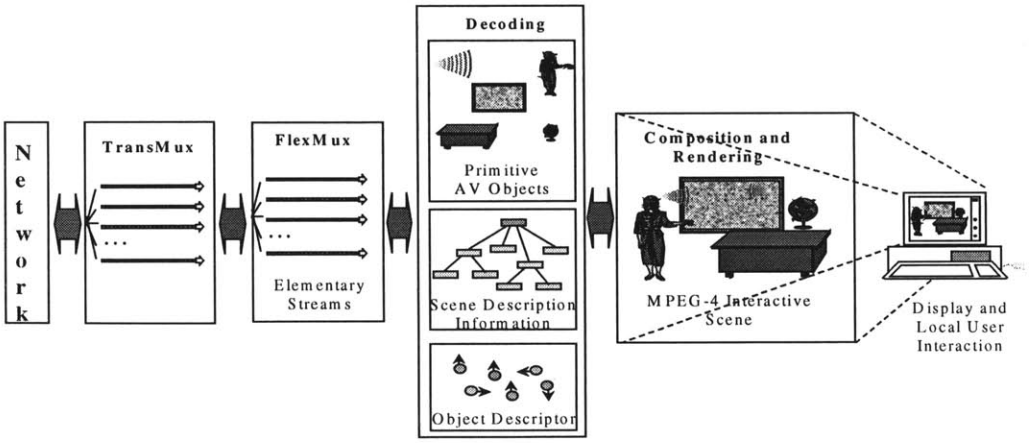

Figure 6. Video Stream Delivery in MPEG-4 [8].

The synchronized delivery of streaming information from source to destination is specified in terms of the synchronization and the delivery modules containing a

two-layer multiplexer. The first multiplexing layer is managed according to the DMIF specification. This multiplex may be embodied by the MPEG-defined "FlexMux" tool, which allows grouping of Elementary Streams (ES) with a low overhead. Multiplexing at this layer may be used to group ES with similar Quality of Service (QoS) requirements to reduce the number of network connections and the end-to-end delay. The second layer is the transport multiplexing or "TransMux" layer. It offers transport services matching the requested QoS. Only the interface to this layer is specified by MPEG-4. The concrete mapping of the data packets and control signaling must be done in collaboration with organizations that have jurisdiction over the respective transport protocol. Any suitable existing transport protocol stack may become a specific TransMux instance, such as the Real Time transport Protocol (RTP), User Datagram Protocol (UDP), Internet Protocol (IP), or Asynchronous Transfer Mode (ATM). The choice is left to the end user or service provider. As a result, MPEG-4 can be used in a wide variety of operation environments [8].

The module that controls the integration of the different protocols is the Delivery Multimedia Integration Framework (DMIF). The flexibility of DMIF is evident in various situations. For example, developers can write an application once and run it with different delivery technologies by simply plugging in more DMIF instances. Thus, the functions of commercial multimedia applications would no longer rely on uncertain guesses of the network technology of tomorrow.

DMIF also abstracts application from delivery of data and provides a uniform walkthrough to access content. In other words, the applications that rely on DMIF for communication do not need to know the underlying communication method. DMIF specifies the delivery layer details and presents applications with a simple interface with MPEG-4. The development of DMIF was strongly motivated by the fact that many different standards exist. Each delivery

technology, for example, has it own peculiarities, and each environment has its own API. The environment can be local files, broadcast sources, and interactive servers [9].

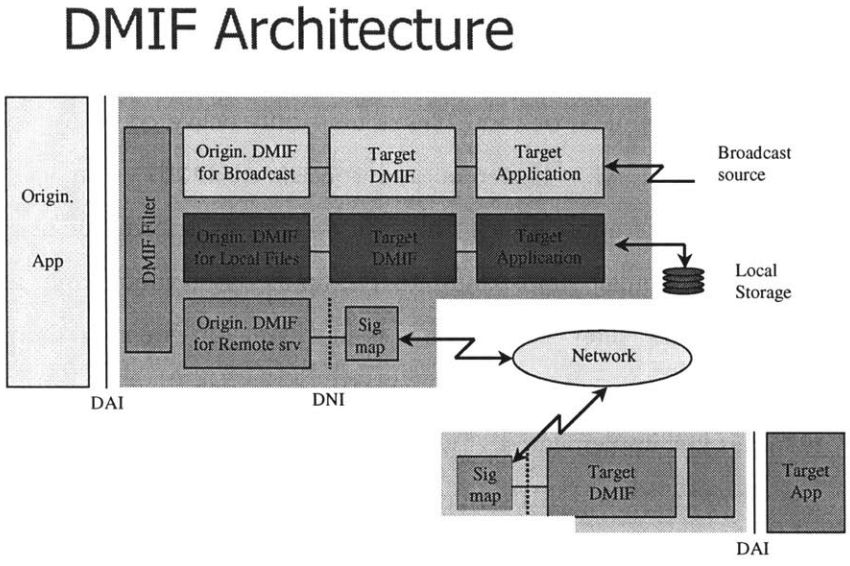

Figure 7. DMIF Architecture. [8]

In the Remote Interactive scenario, the DMIF layer is completely application-unaware. The introduction of an additional interface, the DMIF Network Interface (DNI), emphasizes the kind of information DMIF peers must exchange. An additional module ("sig map") handles the mapping of the DNI primitives to signaling messages used on the specific Network. Note that the DNI primitives are only specified for information purposes. Because a DNI interface need not be present in an actual implementation, the above figure represents the DNI as internal to the shaded box.

An application may also access data through the DMIF Application Interface (DAI), irrespective of whether the data comes from a broadcast source, local storage, or a remote server. In all scenarios, the application only interacts through a uniform interface or the DAI. Different DMIF instances will then translate the local application requests into specific messages and deliver them to the remote application. This process takes care of the peculiarities of the delivery technology involved. Similarly, data entering the terminal from remote servers, broadcast networks, or local files is uniformly delivered to the local application through the DAI. Different and specialized DMIF instances are indirectly invoked by the application to manage the various specific delivery technologies. The details remain transparent to the application, which only interacts with a single "DMIF filter." This filter is in charge of directing the particular DAI primitive to the right instance. DMIF does not specify this mechanism, but assumes it is implemented. The shaded gray boxes in the above figure emphasize the borders of a DMIF implementation. While the DMIF communication architecture defines a number of modules, the actual implementations are only required to preserve their appearance at those borders.

MPEG-4 Video

Above the DMIF protocol lies the MPEG-4 audio-video system. It integrates conventional and object oriented coding to achieve high flexibility. Conventional coding provides baseline functions, such as compression, error resilience, and scalability, while object oriented coding provides extended features of content-based coding and still texture coding. Each video object in a scene is coded and transmitted separately. MPEG-4 is fully compatible with baseline H.263 and

almost compatible with MPEG-1 and MPEG-2. In order to build a high-quality prototype, an understanding of MPEG-4's coding efficiency, scalability, and synchronization of streaming data is critical [9].

Coding efficiency

MPEG-4 improves the motion estimation and compensation for rectangular and arbitrarily shaped objects significantly. This is accomplished with a few new techniques

1. Global motion estimation predicts the global motion for an object with a small number of parameters. It is based on trajectory coding, texture coding, and warping for prediction errors.

2. Quarter Pel Motion Compensation enhances the precision of the motion compensation scheme. This comes at the cost of small syntactical and computational overhead. However, a more accurate motion description gives small prediction error, which eventually leads to better visual quality.

3. Shape-adaptive DCT significantly improves the accuracy of texture coding for arbitrarily shaped objects. This algorithm is based on predefined orthonormal sets of one-dimensional DCT basis functions.

Evaluations show that the combination of the above techniques saves up to 50% of a bit stream, compared to MPEG-1. [5]

Video Scalability

Scalability refers to the ability to decode only a part of a bit stream and reconstruct images or image sequences with reduced quality, spatial resolution, or temporal resolution. The MPEG-4 standard supports the coding of images and video objects with spatial and temporal scalability, both with conventional rectangular as well as with arbitrarily shaped objects. For decoding still images, the MPEG-4 standard will provide spatial scalability of up to 11 levels of granularity and quality scalability up to the bit level. For video sequences an initial maximum of 3 levels of granularity will be supported, but work is ongoing to raise this level to 9 [9].

In order to achieve a balance between coding efficiency and fine-granularity requirements, MPEG-4 adopted a hybrid scalability structure characterized by a DCT motion compensated base layer and a fine granular scalable enhancement layer. Under this structure, the server can transmit part of or the entire over-coded enhancement layer to the receiver. Therefore, this structure enables the streaming system to adapt to varying network conditions. In the multicast system, the base layer must carry a minimally acceptable quality of video to be reliably delivered. The enhancement layer would improve upon the base layer video and fully utilize the estimated available bandwidth. For example, the enhancement layer could be all or part of the difference between the base layer and the original image data. To achieve the goal of scalability, the following are the different components that need to be modified.

1. Encoder complexity scalability. It allows encoders to generate different layers of valid and meaningful bit streams for a given texture, image or video. The fundamental idea in achieving scalability is to separate a high quality video into different layers, which consist of a base layer with minimally acceptable quality and one or more enhancement layers.

2. Decoder complexity scalability. For a given texture, image or video, the decoder can produce different levels of complexity. The higher the decoded quality, the more layers and the larger the subset the decoder applies.

3. Spatial scalability. The decoder can choose a subset of the total bit stream generated by the encoder to reconstruct and display textures, images and video objects at reduced spatial resolution. For textures and still images, a maximum of 11 levels of spatial scalability will be supported. For video sequences, a maximum of 3 levels will be supported.

4. Temporal scalability. Similar to spatial scalability, the decoder decodes only a subset of the total bit stream to achieve reduced temporal resolution. A maximum of 3 levels will be supported.

5. Quality scalability. The bit stream can be parsed into several layers of different bit rate, and the combination of a subset of the layers can still be decoded into a meaningful signal. The bit stream parsing can occur either during transmission or in the decoder. The reconstructed quality is related to the number of layers used for decoding and reconstruction.

Scalability is a highly desirable feature for several purposes, especially for applications that require progressive coding of images, video sent over heterogeneous networks, and low bandwidth receivers. For example, a client with slow network connection and processor speed may not be capable of displaying the full resolution or full quality images or video sequences. As a result, rather than sending the whole file, the scalable system transmits only the subset of file that the client is capable of decoding [9].

Synchronization of streaming data

As described earlier, each stream is characterized by a set of descriptors for configuration information, such as the required decoder resources and the precision for encoded timing information. Synchronization of elementary streams is achieved through time stamping of individual access units within elementary streams. The synchronization layer manages the identification of such access units and the time stamping. This layer also identifies the type of access unit in elementary streams independent of the media type, recovers the time base of a media object or a scene description, and synchronizes among them. The syntax of this layer is configurable in a large number of ways, allowing use in a broad spectrum of systems.

In the real implementation, the sync layer has a minimum set of tools for consistency checking and padding to convey time-based information. Each packet consists of one or a fragment of an access unit. The time stamped access units form the only semantic structure of elementary streams that is visible on this layer. Time stamps transfers the nominal decoding and composition time for an access unit. The sync layer requires reliable error detection and framing of each individual packet from the underlying layer, which can be accomplished by using the FlexMux. The sync layer retrieves elementary streams from SL-packetized streams.

Chapter 4

DESIGN CONSIDERATIONS

To design a scalable video scheme for the multicast system, the following issues must be carefully considered.

- Integration of the scalable video scheme into the multicast system.
- File format of the scalable video.
- Complexities in the encoder and decoder.

In order to seamlessly integrate the scalable video scheme into the current IP multicast system, it is imperative to understand how the packets are delivered in this network.

IP Multicast Video

The IP multicast video system consists of a group of hosts on an IP network that are multicast enabled. After receiving a video clip, each host may in turn multicast it to receivers that join later. In other words, each receiver may potentially act as a server by establishing a new session in which one receiver host becomes a server for the particular video clip requested. The arbitration

mechanism is implemented to suit the particular applications, yet remains transparent to the users. Every multicast session may include multiple data streams. In the modified multicast system with scalable video support, 2 multicast channels will be used for each video program.

- Data Channel.

  The data channel contains "packetized" streams with fixed size. Each packet is stamped with a sequence number and multicast with the User Datagram Protocol (UDP). At fixed intervals, datagrams will be sent to the receivers with adjusted bit rate. The number of repair requests directly controls the transfer rate on the data channel. When the source server detects that the number of packets lost exceeds a certain threshold, it will halt sending datagrams, and let the repair process take over.

- Repair Channel

  On the receiver ends, the packets may not arrive in order on the data channel. A main function of the receivers is to re-order the packets and detect any missing datagrams. On the repair channel, requests for each missing packet or NACKs are multicast after waiting for a random period. During this waiting period, if a receiver host notices a NACK from another host for the same missing packet, it will suppress its own NACK. This scheme combines both the Automatic Repeat Request and the NACK suppression techniques. As a result, the number of duplicate requests will be minimized on the repair channel [3].

All hosts on the system subscribe to the Inter Object Control (IOC) channel. Messages containing information about the different available sessions will be multicast on the IOC channel. An application on the server host can directly

initiate a session.. The server host may also start a session upon receiving file request messages over the IOC channel. Whenever it creates a new session , the IOC channel announces its existence through multicasting a session description packet. To participate in a session, the receiver subscribes to the appropriate data and repair multicast channels associated with the session of interest. Currently, videos are compressed in MPEG-2 format and packetized automatically in the multicast system for transmission [3].

The above explanation indicates that modifying the system architecture is unnecessary because the IP multicast video system abstracts away the network layer from the application layer. The only adjustment occurs in the way a host subscribes to a channel. Instead of a single data channel and repair channel for each program, hosts that request high quality video will have to subscribe to multiple data and repair channels. In order for receivers to select the appropriate channels, messages sent by the IOC channel must contain sufficient information about the channels, such as the kinds of layers they contain.

Types of scalability

Scalability can be achieved in several different ways, but they all share one fundamental concept: layering. Generally, the video stream is separated into different layers, each containing a subset of the total video data. The sum of all layers will provide enough information necessary to playback the original quality.

Three aspects of scalability deserve consideration: resolution, rate, and granular scalability.

➢ Resolution Scalability

Resolution scalability means that the decoder can display picture frames of different sizes. The larger frame-sized video would contain the smaller picture frames as subsets along with some other information.

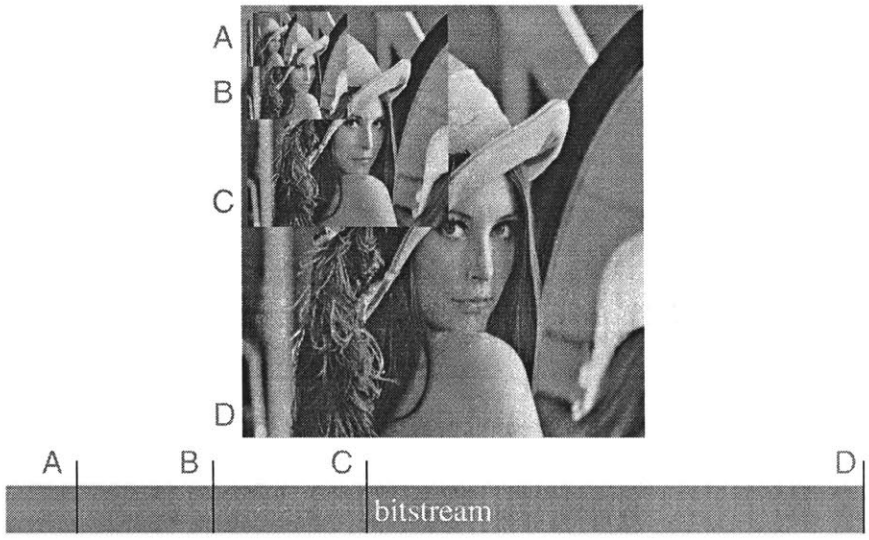

Figure 8. Resolution Scalability [9].

➢ Rate Scalability

Again, the same video stream is separated into different layers of different bits. Based on its network and processor speed, the decoder extracts specific sets of layers to display. The number of layers is proportional to the quality of the video.

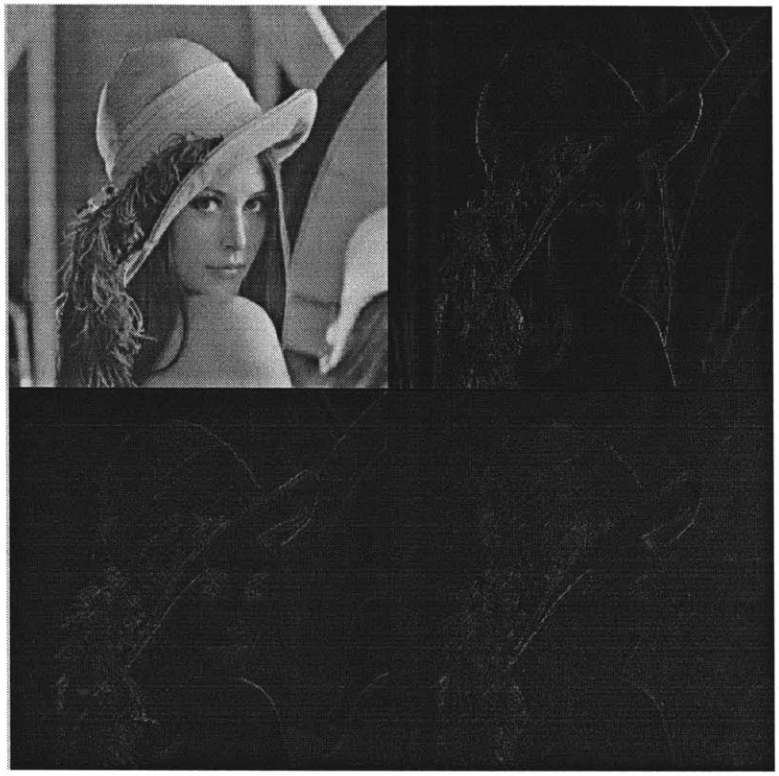

Figure 9. Rate scalability. The upper left corner represents the amount of information contained in the LL layer, and the lower right corner shows the HH layer.

- Granular Scalability

  With this coding scheme, an image can be decoded progressively. Progressive decoding means that the decoder can display the image after receiving only a small amount of data. As it receives more data , the quality of the decoded image improves. When all data arrives, the system displays the original quality. Granular scalability also includes different kinds of layers. A base layer contains information to display minimally acceptable quality while an enhancement layer utilizes the estimated

available bandwidth to improve the base layer. This method is primarily used on still images.

Methods to achieve scalability

There are several methods to achieve video scalability and separate the same bit stream into different layers.

- Spatial decomposition

For a video stream composed of a sequence of frames, a two-dimensional spatial analysis is performed on each frame. One level of the 2-D spatial analysis partitions a video frame into four sub-bands, namely the LL, LH, HL, and HH. LL represents horizontally Low-passed and vertically Low passed sub-band while HH stands for horizontally High-passed and vertically High passed sub-band. As shown in figure 9, the LL strongly resembles the original image and can be further analyzed using another 2-D spatial analysis, and the high frequency sub-bands contain mostly edge information.

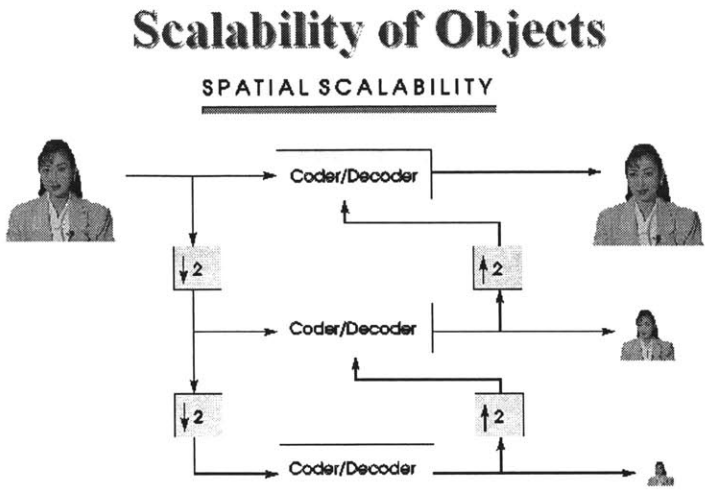

Figure 10. Spatial Scalability [9].

➢ Temporal decomposition

This technique explores the redundancy in adjacent frames. Because consecutive frames contain highly similar data, there is a lot of redundant information. For devices with low bandwidth, one may choose to decode only even numbered video frames.

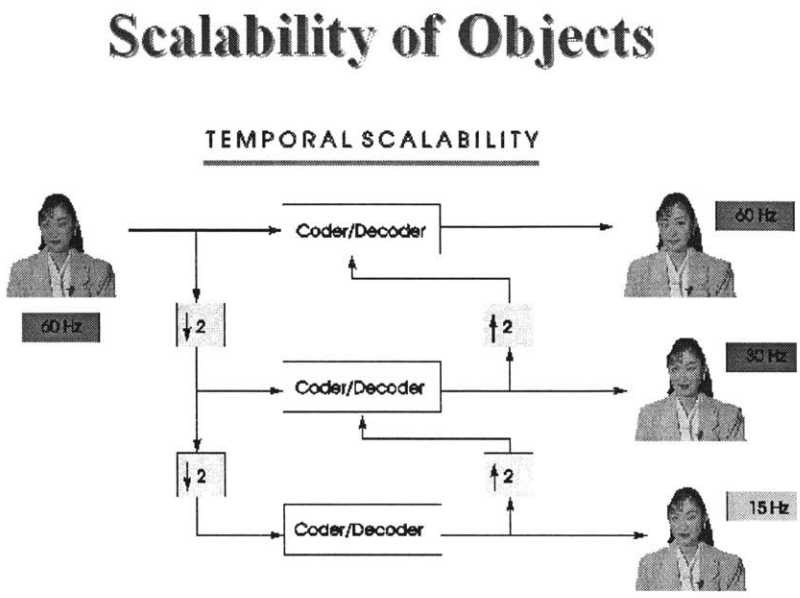

Figure 11. Temporal Scalability [9].

Encoders and Decoders

The functionalities in the encoder and decoder must be allocated appropriately for high performance. The main function of an encoder is to compress the bit streams so that it can be transmitted easily over the network. After the data arrives at the receiver port, a decoder's responsibility is to decompress the information and display it properly.

- Encoder

The encoder is located on the server side. It first encodes a video content. Then the IP Multicast system sends the video packets on the data channel. To achieve scalability, the encoder must be able to separate a video stream into different layers. Then, the multicast system sends each layer on a separate data channel.

- Decoder

The client has a decoder that displays the video packets received. Without the scalable system, the decoders on different client machines decode exactly the same stream if they subscribe to the same program. In a scalable video system, the decoder must be able to display different qualities. This objective is achieved if the decoder can simultaneously decode streams from different channels. For example, for a program that has three layers and the client wishes to see the best quality, the decoder simply integrates the different layers to output a high quality stream.

Chapter 5

SCALABLE VIDEO PROTOTYPE

Overview

The prototype demonstrates the improved flexibility and extensibility of the IP Multicast system with scalable video support. Two modules form the foundation for the system: an encoder and a decoder. On the server side, the encoder separates a video stream into two layers; on the receiver side, the decoder integrates layers from different multicast data channels to form a single video stream.

Two types of layer will result from the encoding process, a base layer and an enhancement layer. If necessary, the encoder can produce more enhancement layers with a few simple extensions. In the currently implementation, the enhancement layer contains the differences between the base layer and the original picture. Thus, decoding the base layer and the enhancement layer will yield the original video quality.

The scalable video scheme is based on the ideas found in ISIS movie constructions. ISIS is a powerful multimedia programming language developed by Stefan Agamanolis in the Media Lab. Named after the Egyptian goddess of

fertility, ISIS is especially tailored to support the development of demanding responsive media applications.

ISIS Video Format

For each video stream, ISIS creates a directory with the same name as the movie. This directory contains four files: audio, info, offset, and data [10].

- Audio

  This file contains the audio data for the movie. The audio is generally encoded in PCM format and has two channels.

- Info

  The "info" file contains important information describing the video and the audio file. For example, it has the version number for ISIS, the name of the audio and video files, the coding scheme, frame size, and the file size. Although the video playback does not require this file, it contains useful information for the user and the other video related functionalities provided by ISIS.

- Offset

  Because the video file store raw data of picture frames consecutively, the offset file provides information to locate any arbitrary frames. Note that there is no offset file for audio because the audio buffer size is always constant.

- Data

  This file contains the raw compressed data for the video frames one after another. To locate the starting point of a new frame, the offset file must be used [10].

To support scalable video, two files in the original ISIS video format have to change. The "info" file must contain information on both the base layer and the enhancement layer. Instead of one data file, there would two data files, one for the base layer and the other one for the enhancement layer. Consistent file labeling must exist to prevent confusion of the content in a data file for the decoder.

Scalable Video Encoder

Both the encoder and the decoder process the video streams on a frame-by-frame basis. The following steps outline the process that produces two layers from encoding the original video content.

1. Divide the frame into 8x8 pixel blocks.

2. Take the Discrete Cosine Transform on each block.

3. Separate the transformed frames into four sub-bands, LL, LH, HL, and HH. The LL sub-band is the base layer because it contains the most important information on the frame. The rest sub-bands together comprise the enhancement layer, which holds mostly edge information

on the video frames. For the next three steps, the encoder performs them separately on the two layers.

4. Compress each layer by reducing the number of bits and eliminating some components.

5. Convert the 8x8 array representation of blocks into a linear sequence following a serpentine pattern for each quality layer. This pattern groups the high frequency bits together.

6. Compress the sequence of characters by the Arithmetic coding to form the final compressed file.

After performing the above steps on all frames in the video, two compressed data files result. The IP Multicast can send each layer on one data channel, and broadcast the existence of the base channel and the enhancement channel over the IOC.

Scalable Video Decoder

The decoder reverses the compression and transformation done by the encoder. During decompression, the followings list detailed steps completed on a frame-by-frame basis.

1. Recover the quantized coefficients from the compressed data stream in each layer or data channel.

2. Take the inverse Discrete Cosine Transform on each block to produce an approximation of the original 8x8 group. There should be two groups, one for the base layer and the other one for the enhancement layer.

3. Combine the resulting two 8x8 groups to form one 8x8 group.

4. Display the resulting frame, which should closely approximate the original uncompressed video frames.

When a host has low network connection speed, it may only wish to receive the base layer. To decode only one data channel, omit step 3 from the above scheme.

Demonstration Scenario

The prototype should demonstrate the receiver's flexibility to decode any number of layers, which results in video of different qualities. The following scenario best achieves this objective. First, a viewer subscribes to a video program that has two channels. As the program starts, the receiver only decodes the base layer to display a minimally acceptable quality. When the viewer pauses the video, the decoder processes both the base layer and the enhancement layer. This results in a much clearer picture frame. If the viewer clicks "pause" again, the playback returns to the lower quality video by decoding only the base layer.

This scenario demonstrates the idea well because it shows the flexibility of the scalable video in the multicast framework. Decoding only the base layer simulates situation of transmitting data through low bandwidth networks. For

receivers using high bandwidth networks and fast computers, the decoder processes both layers, as illustrated by pausing the video. Sending data packets over the network has numerous uncertainties. The number of channels a host receives is heavily dependent on the network traffic, its processor load, and other factors explained in the previous chapter. For the same receiver, it is possible that sometimes it receives both channel and sometimes it only receives one channel.

Chapter 6

CONCLUSION

Summary

This research increases the flexibility and extensibility of IP Multicast Video by incorporating the scalable video scheme into the design. The system is capable of accommodating uncertainties in the network and the differences in receivers' displaying mechanisms. Hence, the resulting multicast system offers video viewing to a wide variety of devices, yet customizes the presentation for each individual receiver.

The fundamental concept lies in encoding a video stream into different layers. Each decoder must process the base layer because it contains data necessary to display minimally acceptable video quality. To achieve higher quality, the decoder combines the base layer with one or more enhancement layers. The two important modules used in this scheme are the encoder and the decoder. On the server side, the encoder divides a video stream into a base layer and an enhancement layer. To generate more levels of scalability, the encoder can be easily extended to include more than one enhancement layer. On the receiver side, the decoder integrates layers from different data channels to form the high quality video stream. Because the multicast system abstracts the transportation of

data packets in the network layer from the application layer, the modified encoder and decoder seamlessly incorporate in the original multicast system framework.

This research also develops a prototype to illustrate the added flexibility and extensibility of the multicast system. For playback of video, only the base layer is decoded. However, when a viewer pauses the video clip, both the base layer and the enhancement layer are processed for a much higher quality still image. Because there are numerous uncertainties in the network parameters and receiver devices, this prototype demonstrates the system's capability of decoding different data channels even for the same receiver.

Future Extensions

As technology evolves rapidly, new devices will offer improved portability and ease of use. The multicast system should certainly be broadened to multicast data to different devices. Potential extensions to the system are as follows:

- Incorporate the MPEG-4 video coding scheme into the Multicast system. The second version of MPEG-4 will be released later this year. As described in Chapter 3, MPEG-4 is highly scalable and allows various data formats, natural or synthetic.

- Extend the multicast system to support interactive applications. The challenge of an interactive application lies in the requirement of two-way communications. Because of the nature of interactivity, end-to-end delay must be very low. Although delays in certain networks can be quite long,

both scalable data streams and technological advancements will permit fast responses in the near future.

- Multicast to different devices. Because Personal Digital Assistants (PDA) and cell phones are quite popular, these low bandwidth devices are already Internet enabled and will soon be video enabled. With the scalable video feature, minor adjustments should permit the system to multicast video streams to different devices of varying bandwidths. The United States lags behind Europe and Japan in adapting mobile devices because of the lack of a unified standard. Currently, the biggest challenge for the US is to agree on a single standard. Once this challenge is overcome, the IP multicast system with scalable video can be fully exploited.

BIBLIOGRAPHY

[1] Hayder Radha, Yingwei Chen, Kavitha Parthasarathy, and Robert Cohen. *Scalable Internet video using MPEG-4*. Signal Processing: Image Communication. Philips Research NY, 1999.

[2] MPEG-4 Requirements Group, MPEG-4 requirements document, Dublin, Ireland, October 1998.

[3] Dean Chen. *IP Multicast Video with Smart Network Caches.* Master's Thesis, Massachusetts Institute of Technology, 1998.

[4] Joseph B. Stampleman. *Scalable Video Compression.* Master's Thesis. Massachusetts Institute of Technology, 1998.

[5] Smith, Steven W. Data Compression. http://www.dspguide.com/datacomp.htm. 1997

[6] Guillaume Boissiere. *Personalized Multicast.* Master's Thesis, Massachusetts Institute of Technology, 1998.

[7] Smith, Chris. *Theory and the Art of Communications Design.* State of the University Press, 1997.

[8] Avaro, Olivier. MPEG-4 Overview. http://www.cselt.it/mpeg/documents/ibc_tutorial/olivier/ppframe.htm. July, 1999.

[9] Sikora, Thomas. Video Tools in MPEG-4. http://www.cselt.it/mpeg/documents/vancouver_seminar/index.htm. July, 1999.

[10] Agamanolis, Stefan. ISIS. http://isis.www.media.mit.edu. May, 2000.