MTEACH: A REMOTE LECTURING SYSTEM

USING MULTICAST ADDRESSING

A Thesis

Submitted to the Faculty

of

Purdue University

by

Vladimir Kljaji

In Partial Fulfillment of the

Requirements for the Degree

of

Master of Science in Electrical Engineering

August 1997

For my family, friends and May. Thank you for your support, help and your love.

ACKNOWLEDGMENTS

I would like to express my gratitude to Professor Edward Delp for being my major advisor, for his guidance throughout the course of this work and especially for his time. This work could not have been done without his support. I would also like to thank Professor Edward Coyle and Professor Michael Zoltowski for their time and for being on my advisory committee.

TABLE OF CONTENTS

Page

Page

Page

LIST OF TABLES

LIST OF FIGURES

Figure *Page*

LIST OF ABBREVIATIONS

AbS	Analysis by Synthesis
ADPCM	Adaptive Pulse Code Modulation
BTC	Block Truncation Coding
CellB	Sun's proprietary video compression standard
CIF	Common Interchange Format
DCT	Discrete Cosine Transform
DES	Data Encryption Standard
FEC	Forward Error Correction
GSM	Global System for Mobile communications (formerly: Groupe Sp—ciale Mobile, now SMG)
ICMP	Internet Control Message Protocol
IDCT	Inverse Discrete Cosine Transform
IETF	Internet Engineering Task Force
IGMP	Internet Group Management Protocol
ISDN	Integrated Service Digital Network
IVS	INRIA Videoconferencing System
JPEG	Joint Photographic Experts Group
LAN	Local Area Network
LPC	Linear Predictive Coding
MBONE	Multicast BackBONE
MJPEG	Motion JPEG
MPEG	Motion Picture Experts Group
NV	Network Video tool
OSI	Open Systems Interconnection

PCM	Pulse Code Modulation
PGP	Pretty Good Privacy encryption program
QCIF	Quarter Common Interchange Format
RAT	Robust Audio Tool
RFC	Request For Comments
RPE-LPC	Regular Pulse Excited - Linear Predictive Coding
RTCP	Real Time Control Protocol
RTP	Real Time Protocol
SCIF	Super Common Interchange Format
SDAP	Session Directory Announcement Protocol
SDP	Session Description Protocol
SDR	Session DiRectory tool
SMG	Special Mobile Group
SMR	Signal to Mask Ratio
TCP	Transmission Control Protocol
TTL	Time To Live
UDP	User Datagram Protocol
VAT	Visual Audio Tool
VIC	VIdeo Conferencing tool
WAN	Wide Area Network
WB	WhiteBoard tool

ABSTRACT

Kljaji , Vladimir M.S.E.E., Purdue University, August, 1997. MTEACH: A Remote Lecturing System Using Multicast Addressing. Major Professor: Edward J. Delp.

Motivation for this thesis comes from the desire to create a complete and functional set of software applications which would support remote lecturing over a campus-wide computer network. Several issues need to be considered - the need for an information system which would enable users to access all the necessary information regarding a lecture (such as time, network address, and the necessary set of multimedia tools), conservation of network bandwidth, low cost and the implementation of “floor control.” What is meant by floor control is session moderation, i.e. enabling and disabling participant's transmissions and preventing unauthorized malicious transmissions.

A highly configurable distributed client-server system (similar to a program guide, or course schedule) for facilitation of management, scheduling, announcement and activation of various real time and stored multimedia events is developed. This system is known as ***MTEACH***. Basic user administration such as adding, modifying, deleting and changing user privileges is also possible. Two existing video and audio conferencing applications have been altered in order to implement the floor control and question asking mechanisms.

Since the underlying network protocol uses multicast addressing scheme, necessary network bandwidth is minimized. Even though ***MTEACH*** works best in a controlled network environment such as a campus network, it can be used on a wider scale provided that all of the network routers support multicast routing. If not, ***MTEACH*** can still work if the lack of multicast is augmented by the use of MBONE (Multicast backBONE) techniques.

Test results have shown that ***MTEACH*** achieved its purpose. Information about lectures can be accessed and changed easily. The implementation of the floor control mechanism along with "question taking" have successfully been implemented without significant network loading. By avoiding the use of dedicated hardware as much as possible, system cost is retained at a minimum.

1. INTRODUCTION

1.1 Review of Teleconferencing and Remote Lecturing Systems

Teleconferencing systems enable interaction between participants located in remote locations. Remote lecturing systems are a subset of teleconferencing systems, where usually one of the participants, the lecturer, has more privileges than the others.

Depending on the type of the interaction, various combinations of audio, video and other media (such as a whiteboard, or still images) can be used. Teleconferencing systems can be point-to-point, point-to-multipoint or the increasingly popular, multipoint-to-multipoint (Figure 1.1). The connections between sites can be one way (simplex), half duplex or two-way (duplex).

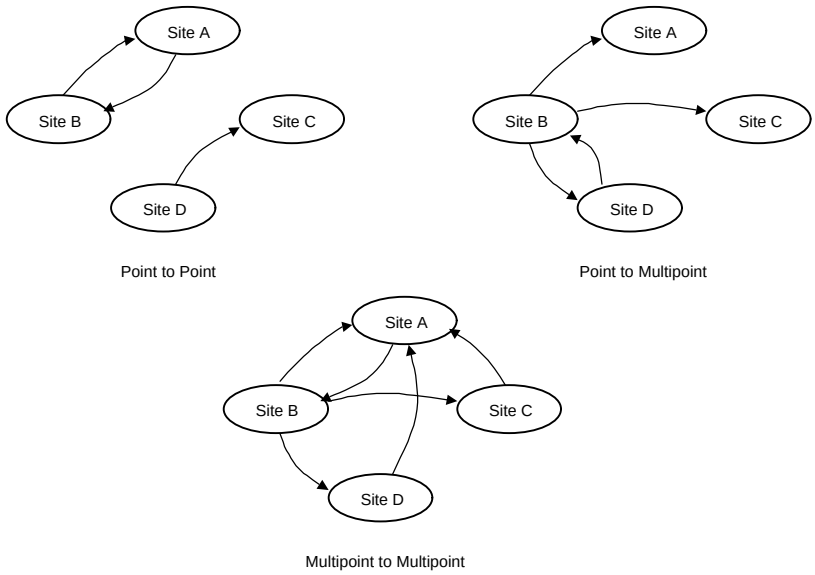

Fig. 1.1. Various Types of Conferencing Systems

Various types of teleconferencing began with the invention of the telephone. The first examples date from the previous century when the sound of a live opera performance was transmitted to a number of subscribers. It was simplex and point to multipoint. One of the first multipoint to multipoint teleconferencing systems was used in the 1930s in Iowa linking housebound and hospitalized students in a school district [1].

Teleconferencing systems have evolved a great deal since then. At first, costly equipment had to be used, e.g. dedicated hardware video compression boards and leased ISDN lines [2], [3]. Later, as computing technology evolved, there has been a move from dedicated hardware to the use of standard workstations and PCs. Examples can be seen in [4], [5]. One interesting example of a modern remote lecturing system can be seen in Stanford's On-Line Distance Learning System, demonstrated at ACM Multimedia '94 [6].

Ultimately, the type of network connection is moving from leased lines to general purpose communication networks, usually LANs and the Internet. A good comparison of quality and performance of such systems is described in [7].

1.2 Motivation

Motivation for this thesis comes from the wish to create a complete and functional system of software applications which would support remote lecturing over a campus-wide computer network. The issues being considered are:

- The need for an information system which would enable users to access all the necessary information with respect to a lecture (such as time, address and the set of necessary multimedia tools),
- Conservation of network bandwidth,
- Low cost,
- Implementation of “floor control.” What is meant by floor control is session moderation, i.e. enabling and disabling participant's transmissions and preventing unauthorized malicious transmissions.

1.3 Proposed Remote Lecturing System

The remote lecturing system requirements are met by ***MTEACH*** in the following way:

- ***MTEACH*** is developed in a modular fashion. A highly configurable distributed client-server system (similar to a program guide or course schedule) for facilitation of management, scheduling, announcement and activation of various real time and stored multimedia events is developed. Basic user administration such as adding, modifying, deleting and changing user privileges is also possible. ***MTEACH*** consists of a distributed server on one side and two client applications on the other. One client application is used for maintaining and running the information system and the other is used by participants to retrieve information about the sessions.

- When considering the reduction of network bandwidth, one usually thinks of utilizing more efficient audio and video compression algorithms. The approach taken in ***MTEACH*** is that of reducing the need of transmitting large amounts of data (namely audio and video), hence the network load would be decreased. This is achieved by the use of a "question asking" mechanism, a similar approach as in [4], where it was referred to as an "interrupt request." The main idea is that if a student has to ask a question, he or she should indicate that by "raising a hand," which in ***MTEACH*** becomes pressing a button. This sends a interrupt to the teacher and the teacher can then decide whether or not the question can be asked. Only then does the student start to transmit data, as opposed to the case when at least audio would have to be transmitted just to ask if the question can be asked. This way, not only is the network load decreased, but the case in which several students might start transmitting at once and creating overall confusion is avoided.

- Numerous conferencing systems have used costly dedicated hardware [7], [1]. They provided high quality audio and video, but results have shown that audio is

more important. Video serves only to establish a teacher's presence and lower quality video can be tolerated [2]. With the rapid improvements in home computing, it is becoming possible to do real time video and audio compression with a relatively cheap computer. New technologies are also enabling much cheaper multipoint conferencing. By choosing the audio and video conferencing tools that support multicast, network loading is avoided. ***MTEACH*** is written entirely in TCL/TK, thus avoiding the need of costly and time consuming rewriting of program code when moving to different hardware platforms. ***MTEACH*** achieves low cost by being highly configurable. ***MTEACH*** can run even on platforms with minimum hardware (a video camera and capture board are needed if video transmission is used, decompression can be done in software, and practically every home computer comes equipped with a sound card).

- Some authors believe that all participants should be treated as peers and, as the number of users increases, social rules would dictate behavior [2]. It is this author's opinion that in the case of remote lecturing, when there is no immediate authority present (students know that the teacher cannot see them), there has to be a way for the teacher to control the session, i.e. to be able to prevent unauthorized transmissions which both load the network and draw students' attention. If there were sufficient resources for the multipoint conferencing where every participant would transmit non-stop, it would be easier to maintain authority since students would know they are seen, but having 30 or 50 faces on one screen can be overwhelming. The implemented control is known as floor control. The teacher can give or take the "floor" from each participant, hence controlling their transmissions.

1.4 Organization of the Thesis

The following three chapters provide the basic foundations for understanding the principles that make remote lecturing over a computer network possible.

Chapter 2 provides an overview of several audio compression standards. Chapter 3 is a review of video compression standards. The standards reviewed in these chapters are chosen because of their use in the two conferencing applications (***VIC*** and ***VAT***) which are used in ***MTEACH***. Chapter 4 gives an insight to the underlying Internet protocols used in the remote lecturing system. It also describes multicasting scheme and the MBONE initiative.

Chapter 5 reviews the current network conferencing applications using multicast and the MBone, and Chapter 6 deals with the implementation of ***MTEACH***.

Appendix A is a comprehensive manual for ***VIC*** and ***VAT*** including the changes made to support ***MTEACH***. Appendix B contains a manual for the session information part of ***MTEACH*** (MADMIN, MSTUDENT and MSERVER). Appendix C provides an example of a typical ***MTEACH*** session. Appendix D illustrates the necessary network bandwidths while transmitting video with ***VIC*** using several video compression standards.

2. OVERVIEW OF AUDIO COMPRESSION STANDARDS

2.1 Introduction

Audio represents an important part of remote interaction. It is possible to have a remote conference system that uses only audio, in fact, very few systems do not use any audio. In remote conferencing, especially in remote lecturing, telephone quality audio is sufficient.

Telephone quality audio is sampled at 8000 samples per second with 8 bits per sample and only one channel (mono), hence the data rate is 64 kbits/s. While this data rate might not be large when compared to video bandwidth requirements, it more than exceeds what current modems can handle (maximum of 33.6 kbits/s), hence it is desirable to reduce the data rate to fit such a channel. There are several compression schemes that can be used to reduce the data rate needed to transmit audio. The techniques reviewed in this chapter are used in the ***VAT*** audio conferencing tool. These include ADPCM (Adaptive Differential Pulse Code Modulation) and GSM (Global System for Mobile communications). GSM is similar to Linear Prediction Coding (LPC).

CD quality audio is sampled at 44,100 samples per second, 16 bits per sample, with two channels (stereo). This requires a data rate of approximately 1.4 Mbits/s[1]. The most popular compression technique used for this type of audio is the MPEG audio compression standard, which will also be briefly reviewed in this chapter.

[1] A data rate of 48,000 samples per second is sometimes used in professional applications.

2.2 PCM

In Pulse Code Modulation (PCM), each audio data sample is represented by a fixed length code word. The number of bits commonly used are 8, 12, 14 and 16 bits per sample. This format is the common starting point for all digital audio systems.

After sampling, each audio sample is quantized [8]. Quantization is conversion from a discrete time-continuous amplitude signal to a discrete time-discrete amplitude signal. There is a loss of information associated with this process. The difference between the original and quantized signals is known as the quantization error or quantization noise. The perceived magnitude of this error can be minimized by selecting the appropriate quantizing scheme. Depending on the application, uniform, logarithmic, nonuniform or vector quantizers can be used.

A uniform quantizer is one in which all the quantizing levels are integer multiples of a constant known as the quantizing step. A uniform quantizer is very easy to implement, but since no assumptions about the quantized signal are made, the performance of such a system can be very poor.

Logarithmic quantizers exploit the logarithmic nature of the human auditory system - the human ear is more sensitive to softer than louder sounds. Therefore, louder sounds can be quantized with fewer bits. A simple method of achieving this is to pass the signal through a compressor with a logarithmic characteristic before quantization. This compressed signal can then be uniformly quantized. At the output of the system the signal is passed through an expander, whose transfer characteristic is the inverse of the compressor's transfer characteristic. Two standardized compressing functions are the μ (mu) and A-law functions. The A-law function is:

$$c(x) = \begin{cases} \dfrac{A|x|}{1+\ln A}\operatorname{sgn}(x), & 0 \le \dfrac{x}{x_{\max}} \le \dfrac{1}{A} \\ x_{\max} \cdot \dfrac{1+\ln\left(\dfrac{A|x|}{x_{\max}}\right)}{1+\ln A}\operatorname{sgn}(x), & \dfrac{1}{A} < \dfrac{x}{x_{\max}} \le 1 \end{cases} \tag{2.1}$$

where x is the input signal, x_{max} is the maximum of x and A is a static parameter. $A = 87.56$ for most telephone systems. The μ law function is:

$$c(x) = x_{max} \frac{\ln\left(1 + m\frac{|x|}{x_{max}}\right)}{\ln(1+m)} \mathrm{sgn}(x) \tag{2.2}$$

where the static parameter m typically takes the value of 255. A-law is used in Europe while μ-law is used in the U.S. By using this scheme, telephone quality speech can be represented by using only 8 bits per sample as opposed to the 13 bits needed with the uniform quantization to achieve the same subjective quality.

Non uniform quantization is best used when the probability distribution function of the input signal is known. The mean square error between the quantized and original signals can then be minimized by matching the quantization levels to the distribution of the signal's amplitude.

When one sample at a time is quantized, the quantizer is referred to as a scalar quantizer. With vector quantization [9], blocks of N samples are treated as an N-dimensional vector and are quantized to predetermined points in the N-dimensional space. Vector quantization can always produce better results than scalar quantization, but it is more sensitive to transmission errors and has a large computational complexity.

2.3 ADPCM

ADPCM [8], [10] quantizes the difference between the current and predicted version of a sample. If the prediction is accurate, the variance of the difference will be smaller than of the original signal and hence it is possible to quantize it with fewer bits. In order to achieve accurate prediction, the signal statistics are continuously tracked and the predictor and the quantizer are changed.

ADPCM encoders vary depending on the prediction method and the way it adjusts to signal changes. Standardized ADPCM encoders are the ITU-T's (former CCITT) G.721

(with a data rate of 32 kbits/s), G.726 and G.727 (data rates of 40, 32, 24 and 16 kbits/s). These operate on the standard 64 kbits/s audio or speech signal.

ADPCM can also be used in hybrid compression schemes. A good example is the G.722 standard, which is a sub-band ADPCM encoding method. The audio signal is divided into two sub-bands and each is then encoded with ADPCM.

2.4 GSM

GSM is actually a digital mobile radio system, originally designed in Europe [11], [12]. This term is also frequently used to describe the type of audio compression used in GSM. The compression technique uses Regular Pulse Excited - Linear Predictive Coder (RPE-LPC) with a long term predictor loop which results in the compression of telephone quality speech to 13 kbits/s.

The RPE-LPC codec belongs to the class of hybrid coders, which attempt to avoid the higher data rates of waveform codecs (no less than 16 kbits/s) and the unnatural sounding speech of source codecs (vocoders). To be more precise, it belongs to the class of time domain Analysis-by-Synthesis (AbS) codecs. Such coders use the same linear prediction filter model of the vocal tract as LPC (Linear Prediction Coefficient) vocoders. Instead of applying a simple two-state voiced/unvoiced model to find the necessary input to this filter, the excitation signal is chosen by attempting to match the reconstructed speech waveform as closely as possible to the original speech waveform. A generalized block diagram of a AbS codec can be seen in Figure 2.1.

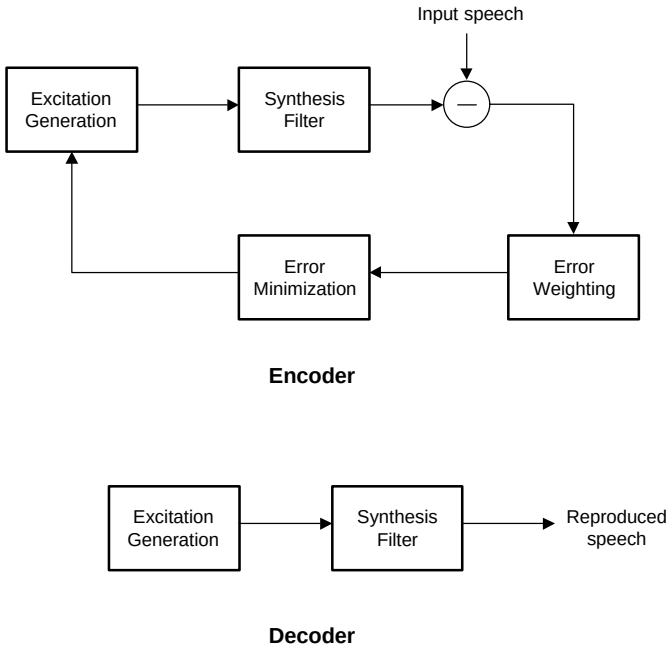

Fig. 2.1. AbS codec block diagram

The input speech is first split into frames, GSM uses 20 ms frames which, at the rate of 8000 samples per second, results in 160 samples in each frame. For each frame, excitation and parameters for synthesis filter are found by minimizing the error between the input and reconstructed speech. Then, the synthesis filter coefficients and excitation are transmitted. At the decoder, each frame is simply reconstructed by passing the excitation through the synthesis filter.

Theoretically, all possible waveforms should be used as the excitation in order to find the one which minimizes the error between the original and reconstructed speech. It is impossible to do this, hence various schemes are used to reduce the number of possible waveforms while retaining the speech quality. A number of non zero pulses is used as excitation and in case of the RPE codec, they are spaced at equal intervals. It is up to the encoder to determine the position and amplitude of the first pulse and amplitudes of the remaining pulses in the sequence. In the case of GSM, the excitation signal is found by decimating the residual signal (obtained by subtracting the reconstructed signal and the

input signal). It is divided in 4 sub-frames, 40 samples each, and each sub-frame is decimated into 13 samples. There can be three possible ways to do this, and the sequence which carries most of the energy is taken to be the excitation signal. In the end, the amplitude of each sample is quantized to 3 bits.

The synthesis filter is usually an all pole filter of the form:

$$H(z) = \frac{1}{A(z)} \tag{2.3}$$

where

$$A(z) = 1 - \sum_{i-1}^{p} a_i z^{-i} . \tag{2.4}$$

$A(z)$ is the prediction error filter determined by minimizing the energy of the residual signal produced when the original speech frame is passed through it. The order p of the filter is typically 10 (GSM uses 8).

GSM compression can be quite computationally demanding, a 486 microprocessor can do it in real time.

2.5 MPEG

MPEG audio compression is used for high quality audio compression. It assumes the input audio signal to be sampled at either 32, 44.1 or 48 KHz, 16 bits/sample, mono or stereo. The target audio data rates for MPEG-1 range from 32 to 224 kbits/s. The standard reviewed in this chapter covers the first MPEG audio standard [13], [14], now known as MPEG-1.

MPEG offers three independent compression layers:

- Layer I is the simplest, and it is suitable for target data rates larger than 128 kbits/s per channel.
- Layer II is more complex and its target data rates are approximately 128 kbits/s per channel.

- Layer III is the most complex but offers the best audio quality and is suitable for target rates of approximately 64 kbits/s.

Figure 2.2 shows the basic encoding scheme.

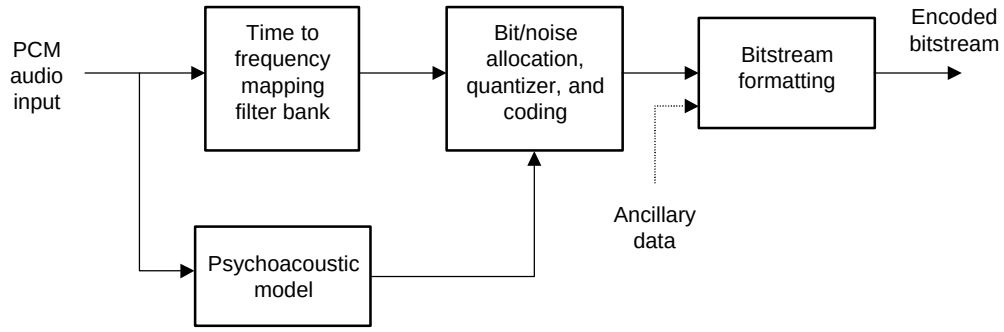

Fig. 2.2. MPEG Audio Compression Block Diagram

The input signal is split into frames, ranging from 384 samples for layer I, to 1,152 samples for layer III. Each frame is then divided into equal frequency sub-bands (32). Bits are allocated for encoding each sub-band, along with the scale factor. The role of the scale factor is to help fully utilize the quantizer range by scaling the samples. The samples are then quantized. If encoding with layer III, quantized samples are further encoded using a Huffman variable length code.

At the decoder (Figure 2.3), samples are dequantized (Huffman decoded, if necessary), scaled and put into the synthesis filter bank. The system is designed to reduce aliasing effects. The overall frequency response, provided that no quantizing has occurred, yields a ripple of no more than 0.07 dB.

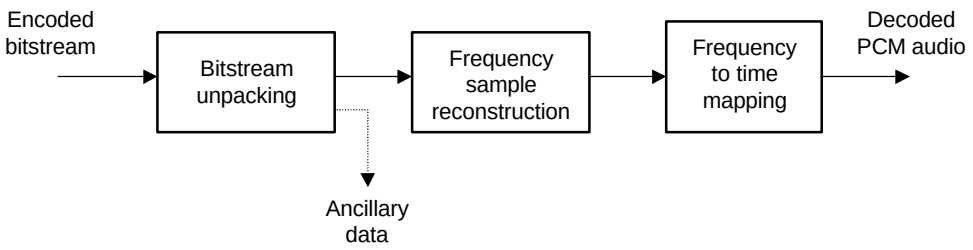

Fig. 2.3. MPEG Audio Decompression Block Diagram

The most intricate part of the MPEG audio compression algorithm is the way bits are assigned to sub-bands. This is done by the use of psychoacoustic models. This is one of the main differences between the MPEG audio layers. Depending on the signal statistics and the human auditory model, the human ear's weaknesses are explored in order to achieve compression. The main weakness is that in the presence of stronger tone, softer tones (and noise) at and around the stronger tone's frequency are masked, i.e. they become inaudible. This fact is represented by Signal to Mask Ratio (SMR), which depends on the frequency of the subband, the number of quantization bits and the characteristics of the encoded signal. Layer I has fairly simple psychoacoustic modeling, while Layers II and III have increasingly complicated models.

There are no audio conferencing applications to date which use MPEG audio, mainly due to its computational complexity and required bandwidth.

2.6 Summary

Three qualitatively different audio compression schemes have been briefly reviewed, with the following characteristics:

- ADPCM - can be used for both speech and high quality audio compression. It operates on the audio signal in time domain, it is of low computational complexity but yields relatively low compression ratios.
- GSM - a standard based on the hybrid RPE-LPC algorithm, used for speech compression. It has relatively high computational complexity, but it yields better compression ratios.

- MPEG-1 - a standard used for compression of high quality audio, operates in the frequency domain. It has high computational complexity but yields the highest compression ratios, as well as the best quality.

Currently, only the first two schemes are used in software based audio conferencing applications.

3. OVERVIEW OF VIDEO COMPRESSION STANDARDS

3.1 Introduction

Transmitting a sequence of images uncompressed over a network requires a large amount of network bandwidth. Consider the QCIF format (176 x 144 pixels). If captured in true color (24 bits per pixel) at 10 frames per second, requires a data rate of 6 Mbits per second to be transmitted uncompressed (see Table 3.1). This data rate would require more than a half of a standard LAN (10 Mb/s) in which case two way communication would be impossible. Things get even more complicated if one would try to fit this video stream into a standard ISDN line (128 kb/s). Hence, the need for video compression.

Table 3.1
Bandwidth Requirements for CommonVideoconferencing Formats

PICTURE FORMAT	REQUIRED BANDWIDTH FOR UNCOMPRESSED VIDEO, AT 30 FRAMES/S, 24 BITS/PIXEL [Mb/s]
QCIF (176 x 144)	18.25
CIF (352 x 288)	72.99
SCIF (704 x 576)	291.96
NTSC (640 x 525)	241.92

When compressing video, the fact that there is a great deal of redundancy between the frames is exploited. There are two types of redundancy - spatial and temporal.

Several video compression standards are briefly reviewed: CellB, Motion JPEG and MPEG, and H.261. Modified versions of CellB and H.261 are used in the ***VIC*** application (see Chapter 5 and Appendix A).

Finally, in Section 3.5, the conditional replenishment algorithm used in the ***VIC*** implementation (and modification) of CellB and H.261 standards is described. This algorithm is necessary when doing unreliable video transmissions since one cannot rely on motion prediction. That is because the video frame from which the estimation has been obtained may not arrive at the destination.

3.2 CellB

CellB compression is Sun Microsystem's proprietary compression designed for use in videoconferencing applications [15]. Its main advantages are approximately the same execution time for compression and decompression.

CellB does not use interframe encoding apart from skipping frames. Intraframe encoding is based on Block Truncation Coding (BTC), devised by Delp and Mitchell [16].

3.2.1 BTC algorithm

There are several variations to the original BTC algorithm [17]. The original version is described here.

A grayscale picture is divided into non-overlapping 4 x 4 (n x n in general) pixel blocks. Pixels in each block are divided into high and low intensity pixels by using the threshold, t:

$$t = \bar{x} = \frac{1}{N} \sum_{i,j} x(i, j) \tag{3.1}$$

where N is the total amount of pixels in the block and $x(i, j)$ is the grayscale value of pixel at position i, j.

The mask $T(i, j)$ is formed:

$$T(i, j) = \begin{cases} 1, & if \quad x(i, j) \geq t, \\ 0, & if \quad x(i, j) < t \end{cases} \tag{3.2}$$

Also, two representative levels, a and b are obtained as follows:

$$a = \bar{x} - s\sqrt{\frac{N_1(t)}{N_0(t)}} \tag{3.3}$$

$$b = \bar{x} + s\sqrt{\frac{N_0(t)}{N_1(t)}} \tag{3.4}$$

where $N_0(t)$ is the number of pixels with intensities less than t, $N_1(t) = N - N_0(t)$ and $s^2 = \frac{1}{N}\left\{\sum_{i,j}[x(i, j) - \bar{x}]^2\right\}$.

For representing the pixel block, only the mask T and the levels a and b are used. Since there are 16 pixels in the block, the mask takes 16 bits and two representative levels take 8 bits each (grayscale), so the 4 x 4 block of 8 bits per pixel (128 bits) is compressed to 32 bits.

When decompressing the block, each pixel in the block is represented with either level a or level b depending on the mask T.

CellB uses this technique on 24 bit images converted to the YC_bC_r color space. Each 4 x 4 pixel block (cell) is represented by 32 bits yielding a 1:12 compression ratio). Sixteen bits represent the mask and are computed the same way as in the basic BCT algorithm. Then, the average chrominance C_b and C_r values are obtained. An index to a table of 256 vectors is found. Each vector consists of two bytes - one byte for C_b and one for C_r with its mean closest to the mean of the original C_b and C_r. The index is obviously 8 bits. The last 8 bits are filled with the index to the table of vectors consisting of one pair of

luminance (Y) values. These two values represent the above mentioned representative levels of luminance. Their mean is the closest to the mean of the computed luminance values for the block.

When decompressing the block, each pixel takes the luminance value depending on the mask, the luminance vector, and chrominance values depending on the chrominance vector.

3.3 JPEG Based Compression Schemes

Both Motion JPEG (MJPEG) and MPEG rely on the still image JPEG compression standard. Developed by the Joint Photographic Experts Group [18], [19], [20]. With data rates from 0.25 bits/pixel for moderate to good quality, to 1.5 - 2.0 bits/pixel for images usually indistinguishable from the original, JPEG is one of the best lossy compression standards targeted for continuous-tone (24 bits per pixel) images.

3.3.1 JPEG standard

JPEG has four modes of operation:

- Sequential encoding - each image block is encoded in a single pass, from left to right and from top to bottom.
- Progressive encoding - an image is encoded in several passes, which results in multiple pass decoding, suitable when having long transmission times so that the viewer can watch the image "build up" from coarser to finer details.
- Lossless encoding - no loss of information.
- Hierarchical encoding - an image is encoded at multiple resolutions. This results in being able to access lower resolution versions of the image without having to decode the full resolution image.

Only the first mode, sequential encoding (sometimes referred to as the "Baseline Sequential Codec"), is discussed here since it is the most popular of the four modes and is used in MJPEG.

The basic compression scheme is shown in Figure 3.1. The image is divided into 8 x 8 pixel blocks and each block is separately transformed using DCT (Discrete Cosine Transform). The DCT coefficients are then quantized and entropy coded.

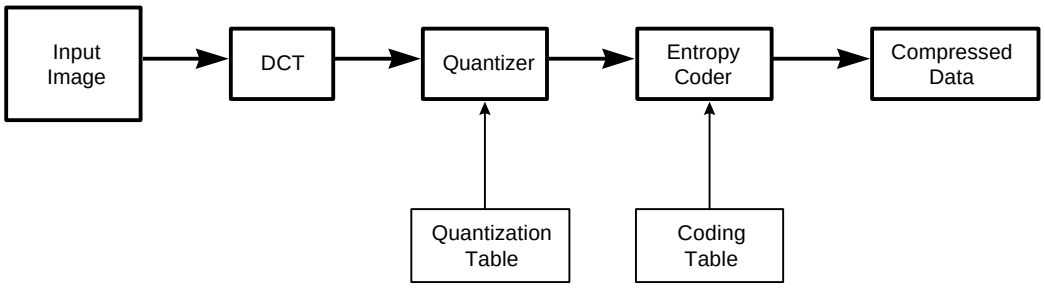

Fig. 3.1. JPEG Compression Block Diagram

The DCT is defined by the following equation:

$$F(u,v) = \frac{1}{4}C(u)C(v)[\sum_{x=0}^{7}\sum_{y=0}^{7} f(x,y)\cdot\cos\frac{(2x+1)up}{16}\cos\frac{(2y+1)vp}{16}] \qquad (3.5)$$

and the IDCT (Inverse DCT) is defined by:

$$f(x,y) = \frac{1}{4}[\sum_{u=0}^{7}\sum_{v=0}^{7} C(u)C(v)F(u,v)\cdot\cos\frac{(2x+1)up}{16}\cos\frac{(2y+1)vp}{16}] \qquad (3.6)$$

where: $C(u),C(v) = \frac{1}{\sqrt{2}}$ for $u,v = 0$,

$C(u),C(v) = 1$ otherwise.

The DCT effectively translates 64 two-dimensional points into 64 two-dimensional spatial frequencies. The relative amount of the two-dimensional spatial frequencies contained in the input signal is represented by the DCT coefficients. The coefficient with zero frequency in both dimensions is known as the "DC coefficient" and the remaining coefficients are the "AC coefficients." Since the color values usually slowly vary from

pixel to pixel, the higher frequency coefficients are typically zero or very small so they can be discarded. This is the main way in which compression is achieved.

Once transformed, the DCT coefficients are quantized. Uniform quantization is used and the quantization thresholds are defined by a quantization table. The table is provided by the user (or the application). Quantization is the main reason for lossy compression, but it is used since it achieves further compression by assigning no more than the necessary number of bits to each coefficient.

After the quantization, the quantized values are entropy coded. It should be noted that the DC coefficient of each block is encoded separately from the other 63 coefficients. The reason for this is that the DC coefficient represents the average value of all the pixels in the block and, does not vary significantly from block to block. Hence, it can be encoded as the difference between the DC coefficient in the previous block and the one in the current block. The rest of the coefficients are first arranged according to the "zig-zag" sequence (Figure 3.2), which places lower frequency coefficients before higher ones.

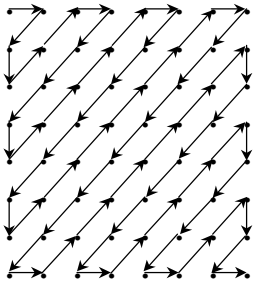

Fig. 3.2. Zig-zag sequence

Lower frequency coefficients are typically larger than higher frequency coefficients so this ordering facilitates the entropy coding. Either Huffman or arithmetic coding can be used. Huffman coding is faster, though arithmetic coding produces 5 - 10% better compression.

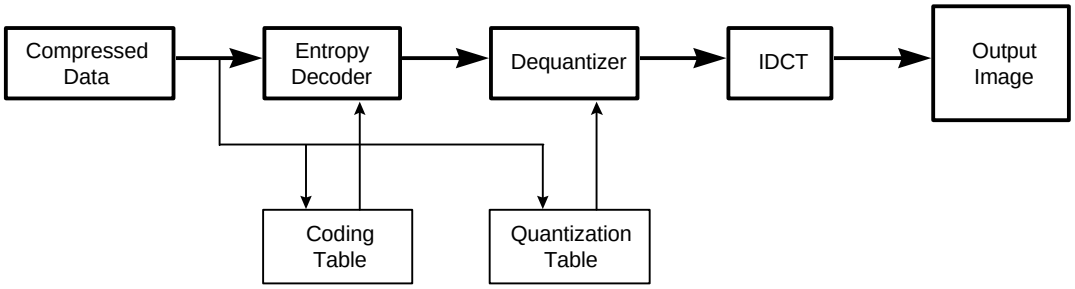

Fig. 3.3. JPEG Decompression Block Diagram

Decompression starts with the entropy decoder, after which the coefficients are dequantized and finally the IDCT is performed (Figure 3.3).

3.3.2 MJPEG and the MPEG standard

Motion JPEG is achieved by encoding or decoding a sequence of images separately in real time using JPEG compression. It can be done in software, but in order to achieve 30 fps, dedicated hardware must be used. No MJPEG standard exists. Its advantage is that it only uses JPEG compression and no motion estimation. The disadvantage is that each frame is encoded separately and hence the compression is not very high compared to schemes that utilize temporal redundancy.

MPEG (Motion Picture Experts Group), on the other hand, is an international video compression standard [18], [21], [22], [15]. It has evolved from CCITT's H.261 standard. It defines the coding of both video and associated audio.

The first version of MPEG, MPEG1 will be reviewed. The latest MPEG standard is MPEG2, while MPEG4 is currently being drafted. MPEG1 is suitable for use in remote lecturing applications because its target data rate is 1.5 Mbits/s. MPEG2 targets commercial broadcast quality and has higher data rates of 5 - 10 Mbits/s. MPEG4 will be the most suitable for the use in remote lecturing since it targets low bit rates (for example, 28.8 k modem videoconferencing).

MPEG compression uses the spatial and temporal redundancy in the video stream in order to achieve compression. Spatial redundancy is exploited, nearly the same way as the JPEG baseline sequential standard. These frames are known as I frames. For inter-

frame compression, motion compensation is used. Macroblocks (16 x 16 pixels) can be forward (Figure 3.4), backward or bi-directional predicted (Figure 3.5).

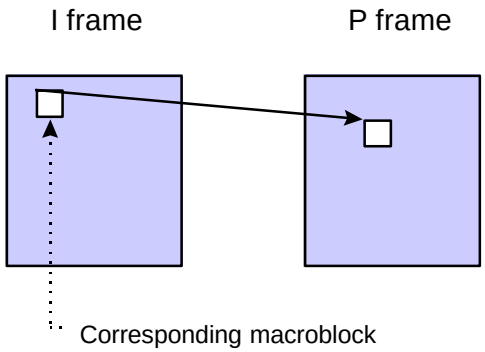

Fig. 3.4. Forward Prediction

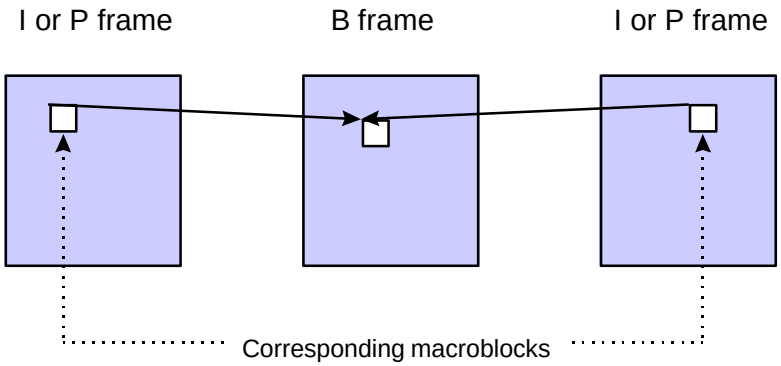

Fig. 3.5. Bidirectional prediction

Apart from I frames, there are:

- P frames, where each macroblock can be either intraframe or forward or backward interframe encoded
- B frames, where each macroblock can be either intra, or interframe forward, backward or bidirectionaly encoded.

One of the previous (forward prediction), future (backward prediction), or both (bi-directional prediction) frames is searched for the macroblock with the best match with the current one. This is known as motion compensation. This part of the MPEG algorithm takes most of the computational time. Depending on the available computational re-

sources, the search can be in steps of one or half pixel. The search window can also be defined. Once the macroblock with best match is found, the difference between the blocks is encoded using the DCT. The motion vector is computed and encoded with a variable length code.

MPEG compression is designed so that it is possible to randomly access the compressed video data. B frames are impractical for random access since it is necessary to have one of the previous and one of the future frames in order to decode it. Also, having many of B frames in a sequence decreases the correlation between the two reference frames (and the compression suffers). On the other hand, B frames achieve very high compression, so it is good to have many of them. It has been found that it is reasonable to put reference frames (I and P frames) at approximately 1/10th second intervals. One possible combination might be: I B B P B B P B B … I B B P B B.

The decoding process consists of demultiplexing the video stream (separating the motion vectors, compressed blocks and ancillary data), dequantization, IDCT and if necessary adding the decoded difference with the corresponding macroblock in the previous and/or future frames (depending on the type of frame).

MPEG's quality is comparable to or somewhat better than VHS video. As an example, MPEG is used to record and later retransmit course lectures is the Stanford's Multimedia Instructional Network.

Some of the arguments supporting the use of MPEG as presented in a demo during ACM Multimedia '94 [23]:

- Good quality real-time video encoders have recently appeared on the market.
- Relatively inexpensive high quality MPEG video decoders for the PC are now available.
- Video servers capable of serving multiple 1.5 Mb/s MPEG video streams simultaneously have been on the market for more than a year.
- Robotically manipulated tertiary storage systems based on tape cartridges or optical disks with storage capacities up to several terabytes are also available.
- New networking technologies and products such as Ethernet switching hubs and ATM will provide the necessary network bandwidth to support video traffic. For

wide area communications, switched-56Kb/s and switched-T1 services are available and are fully adequate to support real-time video streams.

Prices of such equipment have drastically dropped since. For example, PC based MPEG video decoder cards can be purchased for as low as $25.

3.4 H.261

This video compression standard is defined in the ITU-T's Recommendation H.261 titled "Video Codec for Audiovisual Services at p x 64 kbits" [18], [24], [25], [15]. It is designed for transmission of video over ISDN lines with target data rates of p x 64 kbits/s where p can be between 1 and 30. The MPEG1 video standard was based on H.261 and since H.261 is, in a way, a subset of MPEG1, only the differences between the two techniques are discussed here.

The main differences are:

- In H.261, interframe and intraframe encoding is applied at the macroblock level. In MPEG, all macroblocks in I pictures are interframe encoded.
- There is no bidirectional (or backward) prediction in H.261.
- The search window for motion prediction in H.261 can not exceed 15 pixels in both directions.
- Only CIF and QCIF picture formats are allowed, CIF is optional.

A H.261 decoder block diagram can be seen in Figure 3.6.

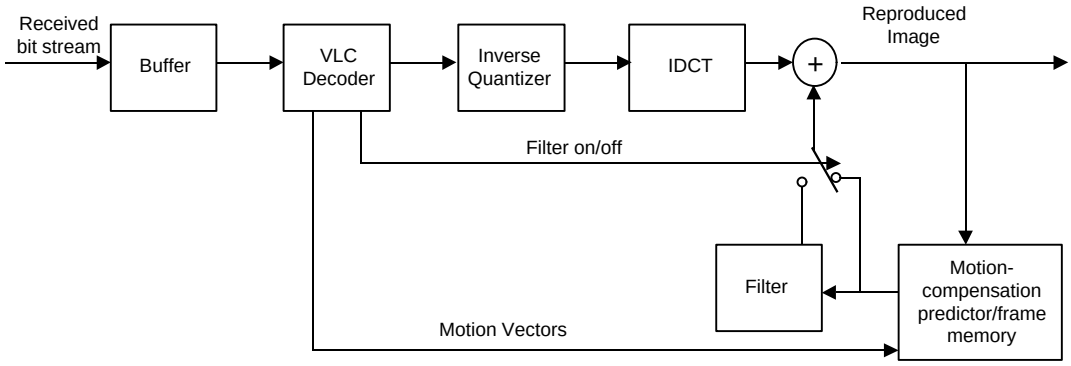

Fig. 3.6. H.261 Decoder Block Diagram

After passing through the receive buffer, the incoming bit stream is decoded in the VLC (Variable Length Code) decoder, which distributes the information to the inverse quantizer, filter on/off switch and motion-compensation predictor/frame memory. After passing through the inverse quantizer, data is inverse DCT transformed and then added to the corresponding previous frame data blocks, as defined by the motion vectors and the motion-compensation memory. Before being added to the current data blocks, previous frame blocks can be low pass filtered, which removes the high frequency noise when needed [24].

There are several interesting characteristics and considerations in the H.261 standard.

- The standard defines only the decoder. The encoder should be compatible with the decoder.
- Only the closest previous frame is used in prediction in order to reduce the coding delay.
- It attempts to balance hardware complexities of the encoder and decoder, which is necessary for real time communication.

A common H.261 encoder implementation can be seen in Figure 3.7.

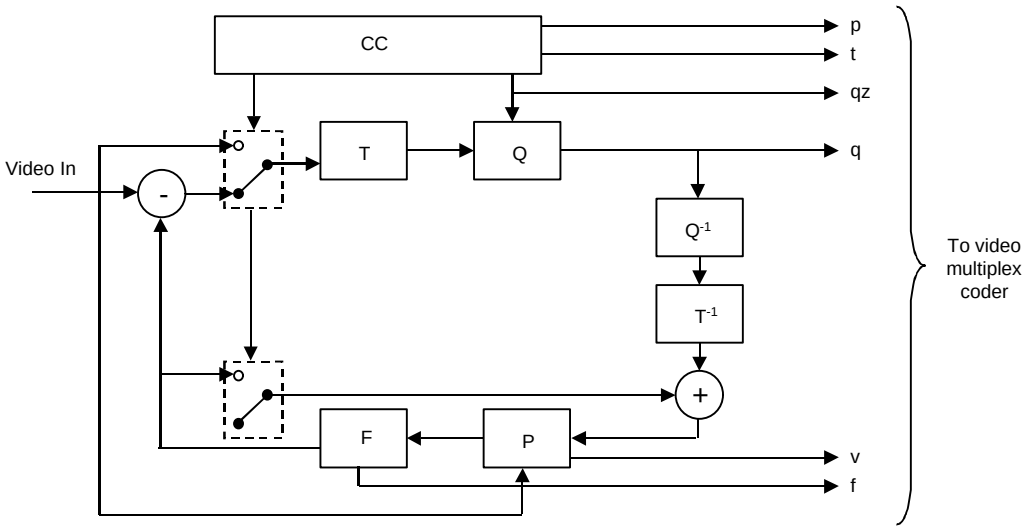

Fig. 3.7. H.261 Encoder Block Diagram

- CC - Coding Control, decides if DCT transformer (T) would transform the input block or the difference between the input and previous block.
- T - DCT transform
- Q - Quantizer
- F - Low Pass Loopback Filter
- P - Picture Memory with motion-compensated variable delay
- p - Flag for inter/intra block
- t - Flag for transmitted or not
- qz - Quantizer indication - which quantizer is used
- q - Quantizing index for transform coefficients
- v - Motion Vector
- f - Switching on/off of the loop filter

H.261 is not as computationally intensive as MPEG, though in order to achieve lower data rates, picture quality suffers.

Its successor is H.263, which targets even lower bit rate video coding.

3.5 Conditional Replenishment Algorithm

All of the previously described compression methods take advantage of temporal redundancy in the video stream. In order to exploit it, each new frame has to be encoded by also using the information from previous (and/or future) frames. If an unreliable transmission protocol is used, as usually is the case with real time audio and video, there is no guarantee that the previous (or future) frames are going to arrive at the destination. Hence, it might be impossible to decode a large part of the video stream in case of even moderate data loss. The best thing to do is not rely on temporal redundancy, i.e. do not use motion prediction techniques. However, the data rate will suffer. A conditional replenishment algorithm uses techniques so that the effect of not using motion prediction is alleviated.

Here is how it works in the ***VIC*** video conferencing tool. A video frame is divided into 8 x 8 pixel blocks and only the blocks that change with respect to the previous ones in time are transmitted. This can save a considerable amount of bandwidth, especially in the case of a still background (as in remote lecturing). The problem with this scheme is that the receiver might accumulate many stale blocks from packet loss or from joining an in-progress session. This is addressed by running a background refresh process which cycles through all the blocks continuously transmitting them at some low rate.

Once the conditional replenishment step determines that a block is to be transmitted, the block must be coded. How it is coded depends upon the compression standard used.

A faster variation of this algorithm is known as "simple conditional replenishment." In this case, block updates are sent uncompressed which requires more bandwidth. This approach has very high image quality but works very poorly over low bandwidth networks. Even on high bandwidth networks, slower end-systems have a hard time keeping up with the data rates associated with processing uncompressed video.

It is interesting that in the ***VIC*** implementation of the H.261 standard with conditional replenishment, the blocks are coded as intra-mode macroblock updates using an H.261 compliant syntax. H.261 codecs typically cannot produce this type of bit stream, but decoders have no problem decoding it since the syntax is fully compliant.

4. NETWORK PROTOCOLS

4.1 Introduction

In the previous two chapters, various audio and video data compression schemes have been described. Once the data is compressed, it is ready to be transmitted over the network. There exist several different ways in which this can be done. Reliable or unreliable, point to point or point to multipoint.

In this chapter, some of the most commonly used protocols in network conferencing are discussed. ***MTEACH*** uses the family of Internet Protocols (IP) for data transfer. Two widely used protocols, namely TCP and UDP, are described in Section 4.3. In Section 4.4, the RTP protocol suitable for real time applications is briefly described and in Section 4.5 the multicast network service and its applications is discussed.

4.2 IP

The IP (Internet Protocol) is the underlying protocol from the TCP/IP protocol suite. This also includes TCP, UDP, ICMP and IGMP [26]. Its official specification is RFC 791 [27].

IP is an unreliable and connectionless protocol. Unreliable means that it is a best effort service - if there is congestion or a link is down, IP simply stops transmitting data, without storing it for later retransmission. It only tries to send back the ICMP (Internet Control Message Protocol) message about what happened. ICMP is a protocol that serves to communicate error messages and other important conditions [26]. IP also calculates the checksum of the data it is transporting, so that a possible transmission error can be detected. Connectionless means no information is maintained about the order in which

datagrams arrive - if two consecutive datagrams are traveling over different paths in the network, they can arrive out of order to their destination, without IP detecting it.

Apart from other identifiers, each IP datagram contains the source and destination network address and TTL (Time To Live) field. TTL is decreased with each router that the datagram passes through. Once it reaches 0 the datagram is discarded unless it arrived to its destination.

IP routing is simple, especially in the case when the destination is directly connected to the host. In this case, the IP datagram is sent directly to the destination. Otherwise, the datagram is sent to the default router, which handles further redirections. When a datagram is received, IP searches through the routing table stored in the memory. It first checks whether the received datagram has one of its own destination addresses or a broadcast address. If so, the datagram is delivered to the protocol module specified in the protocol field of the IP header. If not, the datagram is forwarded according to the routing table (if the host is not configured to act as a router, the datagram is discarded).

4.3 UDP

UDP (User Datagram Protocol) is a simple protocol, where each output operation produces exactly one UDP datagram, which is encapsulated in the IP datagram and then transmitted (Figure 4.1) [26].

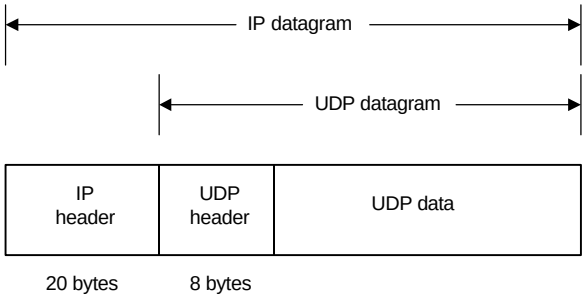

Fig. 4.1. UDP encapsulation

The original specification for UDP is RFC 768 [28]. The UDP protocol is unreliable - data packets are sent from one host to the other but there is no acknowledgment nor re-

transmission in case of lost or damaged data. This protocol is the protocol of choice in audio and video conferencing applications. The reason is that the loss of data is not a problem with the audio and video. Short skipping of audio or a video frame block missing here and there can be tolerated. Of course, with an increase in network congestion, these losses can become intolerable.

4.4 TCP

TCP (Transmission Control Protocol), [26], provides a connection oriented, reliable data flow between two hosts. TCP's original specification is RFC 793 [29]. Two applications using TCP have to establish a connection between them before they can exchange data. The reliability is achieved by doing several things:

- Data is broken into best sized chunks that are passed to IP in order to be transmitted. The chunk of data sent to IP is known as a segment (Figure 4.2).

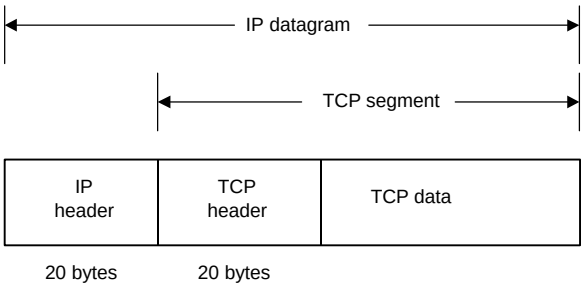

Fig. 4.2. Encapsulated TCP segment

- After sending a segment, TCP maintains a timer, waiting for the reception acknowledgement from the other end. If the timer times out, the segment is retransmitted.
- When the data is received, an acknowledgement is sent.
- TCP maintains a checksum of its header and data.
- Since IP datagrams can arrive out of order, TCP reorders the data in proper order.
- Duplicate datagrams are discarded.

- TCP provides flow control, so that the slower side does not run out of receiving buffers.

Each TCP segment contains the source and destination port number to identify the sending and receiving application. These values, along with IP's source and destination address identify the connection.

TCP is used for communication between the session data servers and clients, as described in Chapter 6.

4.5 Real Time Protocol

Real Time Protocol (RTP) is designated for use with real time applications. As originally described in RFC 1889 [30], “RTP provides end-to-end network transport functions suitable for applications transmitting real-time data, such as audio, video or simulation data, over multicast or unicast network services.” It does not provide any assurance of timely or error free delivery. It relies on the lower layer protocols to do so. UDP is usually used (other protocols can be used too) as a supporting lower layer protocol.

RTP provides packet numbering, time stamping and additional information about the sender. These features are especially useful in real time applications. The ***VIC*** and ***VAT*** conferencing tools are examples of the use of RTP (see Section 5.3). Figure 4.3 shows the RTP packet header.

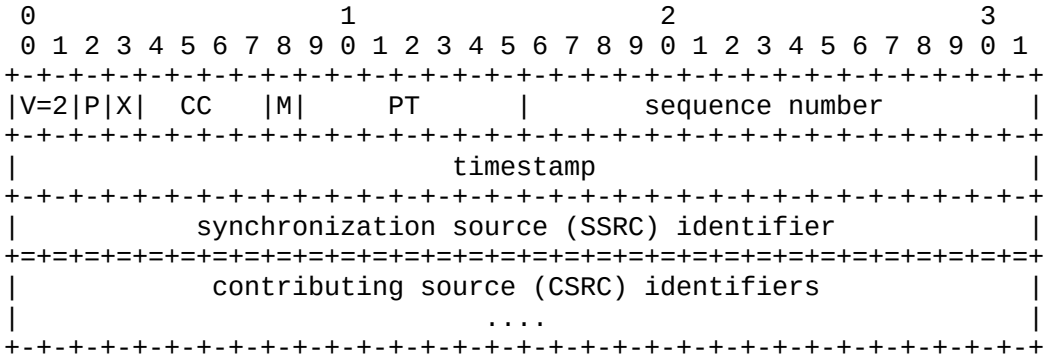

Fig. 4.3. The RTP header

RTP header's main features are:

- PT - payload type. It identifies the format of the payload,
- sequence number - increments by one for each packet sent,
- timestamp - increments by the number of sampling ticks for each data block. Several blocks can have the same timestamps if they are all generated at the same time,
- SSRC - identifies the synchronization source. It is generated randomly so that no two sources have the same identifier. If so, a collision resolving mechanism should be applied, as in [30],
- CSRC - number of these is given by the CC field and those are the SSRC identifiers of the contributed sources, used when mixing several sources into one stream.

RTCP (Real Time Control Protocol) is closely linked to RTP. RTP is used to transport the data packets over the network, whereas RTCP is used to transport control data. Control data carries the information about the quality of service (number of lost packets, time delay) and about the other session participants (real name, email address, phone number, location, etc.). For example, ***VIC*** and ***VAT*** make use of it for session participants. RTCP and RTP protocols use a pair of ports. RTP uses the even port and RTCP the next higher odd one.

One important feature of RTP is its support for mixers and translators. Mixers can be used in the case when a conference participant is connected to the rest of the group via a slow link so it is not able to receive multiple data streams. It is useful if those multiple streams could be mixed into one single stream, which could fit into the link's bandwidth. RTP supports the use of such mixers by being able to merge RTP headers from the packets from multiple sources into one single header. Translators can be used when some of the participants are behind a firewall. Two translators are needed - one outside the firewall which translates the incoming packets and sends them to the other translator located inside the firewall. The other one then translates the data packets back to their original address.

4.6 Multicasting and the MBone

4.6.1 Introduction

Multicasting represents a way of sending information to a group of recipients without having to send the same data packet more than once. It is an optimal way of addressing a group of recipients, as opposed to unicasting and broadcasting which are inefficient. Multicasting was conceived by Steven Deering at Stanford University and is described in [31].

4.6.2 Multicasting

IP supports three types of addresses: unicast, multicast and broadcast [26]. A unicast address defines one and only one recipient, a broadcast address defines a range of receiving addresses. A multicast address defines a group of recipients. If a data packet was sent to several people at once using unicast, one would have to send the same packet once for each receiver. If there exists a part of the path to two or more receivers which is the same, one packet would be sent through that part twice or more. This unnecessarily loads the network. If that one packet was broadcasted in hope that it would reach all the receivers, the surrounding network would be flooded with data packets, and all users would receive them, whether they wanted them or not. Thus, multicast is the optimal way to send data to a group of recipients.

To ensure that a packet passes through a route no more than once and to manage the group membership, IGMP (Internet Group Management Protocol) is used [31]. As with the other protocols from the TCP/IP suite, IGMP is encapsulated within the IP datagram. The routers can keep track of group membership by:

- A host sends an IGMP report when the first process joins a group. If multiple processes join same group, a report is sent only the first time the group is joined.
- No report is sent when the process leaves a group, even if that is the last process in a group. In this case, the next time a IGMP query arrives, no response is sent.
- Multicast routers send the IGMP query at regular intervals to see if any hosts still have processes belonging to any groups.

- The host responds to a IGMP query by sending one report for each group that still has at least one process.

There are still problems involving multicast in wide area networks that need to be addressed (section 9.13 of [31]).

One group is defined by its address (not by the list of its members) and its members can join or leave the group. Multicast addresses are defined as class D Internet addresses. This means that the first four bits of the 32-bit address are 1110, which leaves the other 28 bits to define the multicast address. Thus, multicast addresses can assume ranges from 224.0.0.0 to 239.255.255.255 (some of these are reserved). A group is joined by sending a group membership request. It is important to note that a host does not have to be a member of the group in order to transmit data to that group. Depending of the level of conformance, a host can either:

- support no multicasting (level 0)
- support sending but not receiving multicast packets (level 1)
- have full support for multicasting (level 2)

So far, the most frequently used transport protocol used in multicast is UDP. TCP cannot be used since it is a connection oriented protocol. This makes multicasting unreliable - there is no guarantee that the data packets are going to arrive to their destinations in correct order, if they arrive at all. There have been some efforts at implementing a reliable multicast protocol [32], [33], but none of the implementations is as widely used as UDP.

4.6.3 MBone

Multicasting is a useful way of communicating with multiple recipients but it does not work if the network routers do not support multicasting. Addressing this problem is the purpose of the MBone [34], [35], [36], [37], [38]. It enables multicast traffic even if some routers do not support multicast.

MBone stands for Multicast backBone and is, in essence, a hack. It originated in 1992 from an experiment to transmit live audio and video during the meetings of the Internet Engineering Task Force (IETF). It is a virtual network that runs on top of the Internet and

allows the flow of multicast packets. The MBone addresses the following problem: how to transfer multicast packets over a path in which routers that do not support multicast packets are located (one router that does not recognize multicast traffic is enough to stop the traffic)? Introducing custom software on each side of the path solves the problem. The multicast routing daemon (usually known as “mrouted”) encapsulates all multicast packets into unicast before transmitting them over the problematic path. The same daemon receives the packets on the other side of the path and converts them back to their original state. Paths set up like this are usually referred to as “multicast tunnels,” since multicast packets are tunneled through the unicast path.

A large number of multicast tunnels are set up all over the world, with some estimates of approximately 20,000 users in 30 countries [39]. The MBone will gradually disappear as more and more routers begin to support multicast.

5. CURRENT MULTICAST CAPABLE NETWORK CONFERENCING APPLICATIONS

5.1 Introduction

Currently most popular network conferencing applications using multicast and the MBONE are reviewed in this chapter [35], [40]. These tools run on a large number of platforms, ranging from UNIX workstations (Sun, SGI, HP), to standard home PCs.

5.2 SDR

SDR (Session DiRectory) is used for scheduling and announcing multimedia events on the MBone [41]. To do this, SDR uses the Session Directory Announcement Protocol (SDAP). Once the user enters all information he/she wishes to announce, SDR periodically multicasts session announcement packets describing the session. The packets are multicast on a well known multicast address and port. Sessions are described using the Session Description Protocol (SDP) [42], [43].

When a user receives an SDP packet, they decode it and display the information. It is interesting that the interval between repeats of the same announcement message depends on the overall number of advertised sessions (within the scope where the session is advertised), so that the bandwidth taken by the advertisements remains approximately constant. For example, the more announcements one particular SDR tool "hears," the less frequently it is going to advertise its own announcements. If an announcement is not received for a period of time (usually ten times more than the expected period between the announcements), the receiver considers it canceled. Figure 5.1 shows SDR's main window.

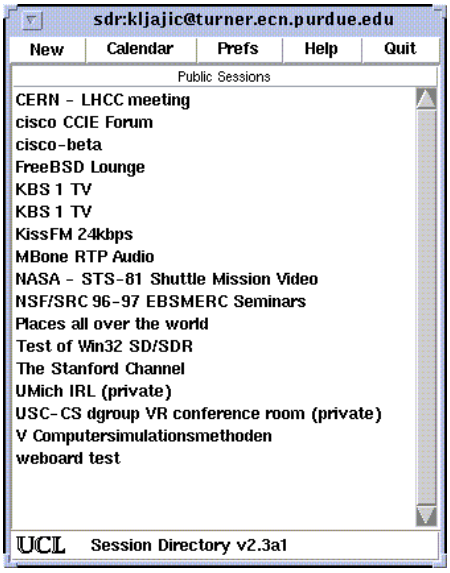

Fig. 5.1. SDR Main Window

Other SDR features include bandwidth allocation for a specified TTL range or administrative scope (defined by a range of multicast addresses, default TTL and a name), a calendar of current and upcoming sessions and a small integrated web browser.

5.3 VIC and VAT

VIC (Video Conferencing Tool), [44], and ***VAT*** (Visual Audio Tool) are video and audio conferencing applications (respectively) developed by the Network Research Group at the Lawrence Berkeley National Laboratory in collaboration with the University of California, Berkeley [45], [40], [46]. They can use either multicast or unicast network protocols. In both cases UDP is used.

Both of these applications (tools) work using the same underlying principles. One address and port are bound for outgoing data (data port) and the same address and port + 1 are used for the control data (information about the other participants). The data is captured, encoded and transmitted over the network and/or received from the network, decoded and displayed/played. As long as any of these programs execute, they send their control data over the control port. Control data contains informations such as the name,

source id, email address, tool that is used and additional notes. In addition, there exists a conference bus, which is a well known multicast address (224.255.222.239) over which all the applications on one computer that support it can exchange data (the packets have TTL of 0 - loopback interface). Ports start with 57007, which is considered to be channel 0 and is shared within all applications, and continue upwards in increments of one for each additional conference bus. This bus can be used for various purposes, one of which are voice switched windows in ***VIC*** (it takes cues from ***VAT*** so that only window with the current speaker is highlighted).

Apart from simple UDP (using multicast), these applications also support the RTP protocol and ATM transfers (in unicast mode).

VIC's receive and decode block diagram is shown in Figure 5.2 [44]:

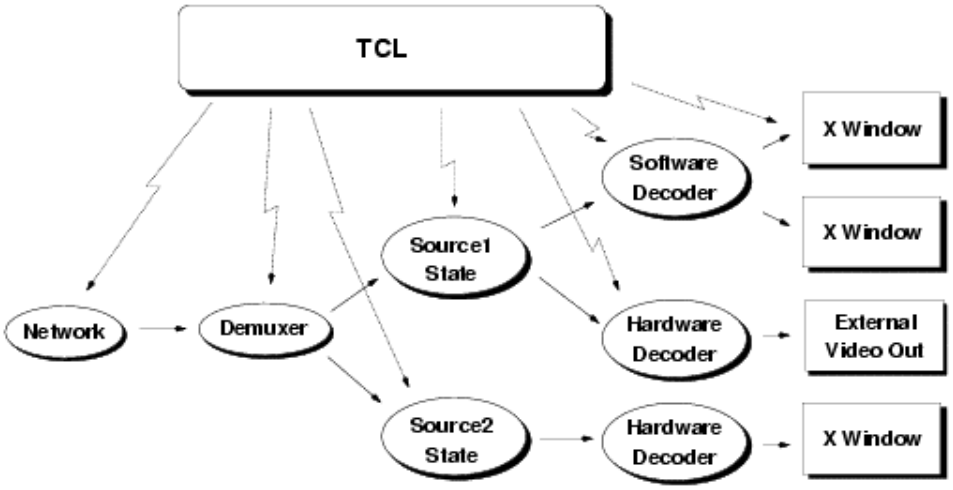

Fig. 5.2. VIC Receive/Decode Block Diagram

A similar architecture is used for the receive path.

VIC has an interesting feature in that it is possible to use JPEG compression for H.261. That can be substantially faster if hardware support for JPEG compression exists. The idea is that the frame is JPEG compressed in hardware, then Huffman decoded and then encoded in H.261. Since the DCT actually takes most of the time when compressing the frame (motion compensation is not used), this can increase the encoding speed.

For more details about supported video and audio compression techniques, see Appendix B.

5.4 Other Applications

5.4.1 WB

WB is the shared whiteboard tool developed by the authors of ***VIC*** and ***VAT*** to provide a shared display function for teleconferencing over the Internet [47], [40]. Users can draw, type or import graphics on the screen which is shared between the session participants.

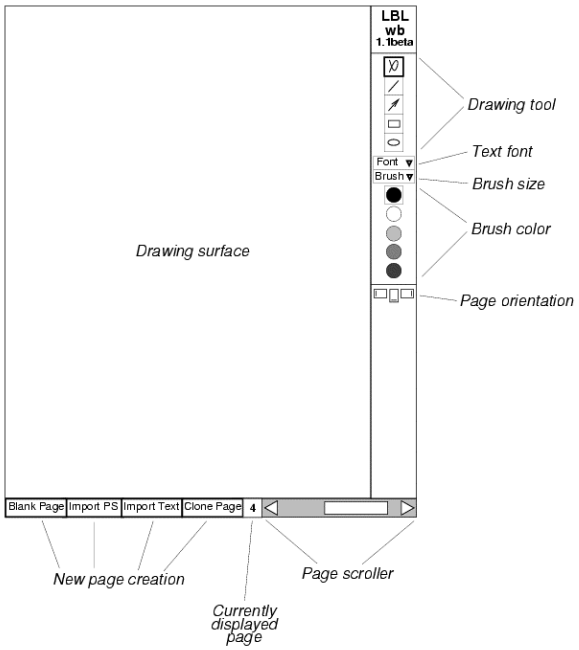

Fig. 5.3. The Whiteboard Window

By representing shapes on the screen as objects rather than bitmaps, WB reduces the necessary bandwidth.

5.4.2 RAT

RAT stands for Robust Audio Tool and was developed in the Department of Computer Science, University College London, U.K, to provide audio teleconferencing over the Internet [48], [49].

Fig. 5.4. RAT Main Window

RAT is similar to existing audio tools (such as ***VAT***) and is RTP compliant. RAT offers some extra functionality:

- Redundancy (packet loss protection) - RAT can transmit a lower bandwidth version of the audio packets so that if the original is lost, the low bandwidth version is used.
- Adaptive Scheduling protection - current general purpose computing facilities, such as UNIX workstations do not provide support for real-time services in their scheduling algorithms. One solution to this problem is to retard the timing in the audio tool, so that all packets can still be played. RAT takes a different approach. It maintains the relative timing interval dictated by the network conditions; this approach minimizes the end-to-end delay of the system, and minimizes the gaps caused by late scheduling of the audio process by the host operating system.
- Improved Statistics - such as measured loss rate.
- Lip synchronization option - a version of ***VIC*** is modified to cooperate with RAT and produce lip-synchronized playback.

- Improved hands-free performance option - by using a silence detection mechanism, there is no need of pressing the "Talk" button.

5.4.3 NV

NV (Network Video), [50], [51], [35], is ***VIC***'s predecessor, developed by Xerox Palo Alto Research Center. It supports two custom compression algorithms - Haar wavelets or DCT. It is the program on which the more complex and effective programs (***VIC***) were based.

5.4.4 IVS

IVS (INRIA Videoconferencing System) was developed by INRIA, France [52], [53], [54]. It is a system which can transmit audio and video over the Internet using multicast or unicast. For audio, PCM or ADPCM can be used, and H.261 is used for video. It also supports the RTP protocol. IVS comes with a set of useful software utilities, namely IVSD (IVS Server), IVS_RECORD (records the conference to a file), IVS_REPLAY (replays recorded conference) and IVS_GW (IVS gateway for use with security fire-walls).

5.5 Summary

VIC and ***VAT***, supported by SDR and WB, are currently the most popular MBONE conferencing tools. Other tools, such as RAT, are emerging and might become increasingly popular over the years.

6. IMPLEMENTATION OF MTEACH

6.1 Introduction

In the previous chapters it has been shown how the two most common means of communication, audio and video, can be efficiently transmitted over a computer network (LAN, ISDN, or WAN). These principles can be used in a remote lecturing system. Such a system has to fulfill two requirements. One is the existence of the lecture information system and the other is the existence of a set of appropriate multimedia tools (applications) which would facilitate the interaction between the students and the lecturer.

With the wealth of tools for remote cooperation, information retrieval, education and, generally, communication there is a need for a system which would manage the tools and if necessary adopt new ones. Such a system will differ in certain aspects depending on its particular use, for example, video on demand or multipoint communication. This particular system, known as ***MTEACH*** (Multimedia TEACHing), has been developed with the needs of a university campus in mind. Its primary purpose is to be able to maintain and enable access to various campus sessions. Sessions can be either course lectures (meant to be interactive) or broadcasts such as local university radio station.

This chapter describes in detail ***MTEACH***. The general structure is described and the details of the client and server implementations are discussed. How the ***VIC*** and ***VAT*** tools have been altered for ***MTEACH*** in order to allow session moderation is also discussed. This is also known as “floor control,” whose implementation is explained in Section 6.3.2. Section 6.3.3 describes the implementation of the mechanism which enables students to ask questions without loading the network. Finally, system test results are presented. The complete manuals for the MADMIN, MSTUDENT and MSERVER applications can be found in Appendix B.

6.2 MTEACH System Overview

6.2.1 Session information system

The basic idea is that the system should provide several basic functions:

- maintain a database of all the multimedia sessions (courses and broadcasts)
- enable database access to eligible users
- enable easy session and user privileges administration
- provide the necessary amount of security
- facilitate the starting of multimedia sessions
- be configurable so that new type of multimedia tools can be added without changing the program code
- be distributed so that the network is evenly loaded

With these functions in mind, the following system elements have been developped:

1) Server (MSERVER), which maintains the session and user database and provides the information upon request. Once started, the server resides on the designated computer and handles user requests.

2) Administrative program tool (MADMIN), which has all the features necessary to add, change and delete the sessions and users from the database. Depending on the given username and password the server gives administrator access, moderator access or refuses access. The program (and the server) allows only certain operations when executed in the moderator mode.

3) User program (MSTUDENT) available to end users, which helps users to retrieve the session information as well as join sessions in progress.

The last two programs are basically client programs activated by users, moderators (professors) or administrators.

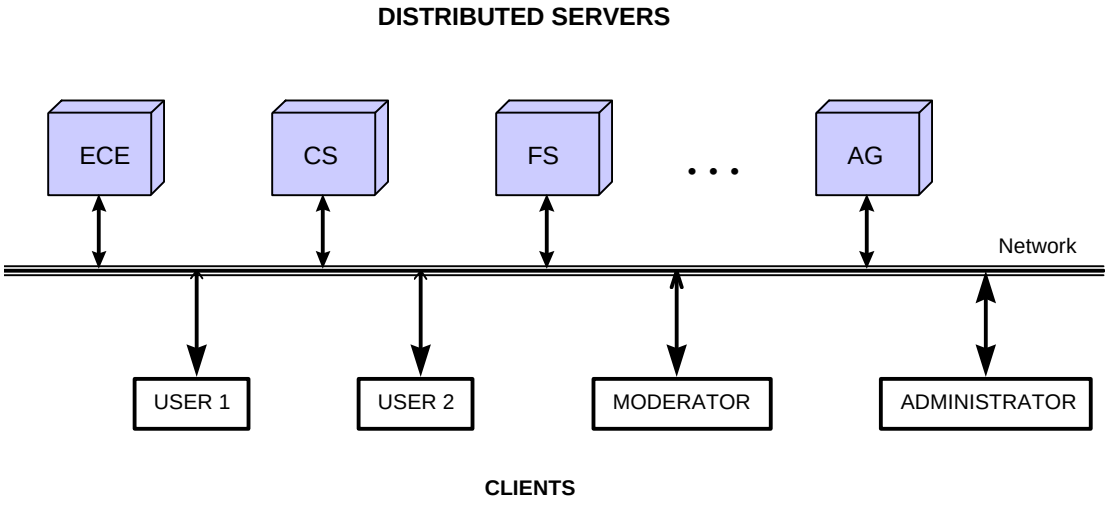

Fig. 6.1. System Block Diagram

In order to access or change any information in the database it is necessary to provide the server with the proper username and password. Each server maintains its own user and session database. The system block diagram is shown in Figure 6.1.

As can be seen from the diagram, the proposed scheme is to distribute servers across departments (Electrical and Computer Engineering, Computer Science, Food Science, …, Agriculture). Clients can connect to any of the servers and, if they provide the server with the proper username and password, can be granted access to the session information. For normal users, the flow of information is mostly from server to the user. The only time users provide servers with information is when they change their passwords (denoted with a small arrow in the diagram). Moderators can, in addition, change informations about the sessions they moderate and the administrators can change any data in the databases, both user, moderator and session.

6.2.2 Floor control

In every centralized system there is a need for the master control. In the case of remote lecturing, the professor (or moderator) should have the capability to control the

course of lecture. This is referred to as “floor control.” The moderator can give or take “the floor” to or from each participant.

The following are the implementation issues concerning the servers, clients, implemented floor control and the ***MTEACH*** system as a whole.

6.3 Implementation

The entire system has been implemented using the TCL scripting language along with the TK toolkit, and the C and C++ programming languages. The reason for using TCL is that it is fairly easy to produce the graphical user interfaces. Another important reason is its portability - it can be ported to every major platform with no code changes.

6.3.1 The client - server system

The servers maintain four separate databases. These are the global session, restricted session, the moderator and the user database. The reason for having separate databases is the ease of maintenance. The databases are accessed only through the server (Figure 6.2).

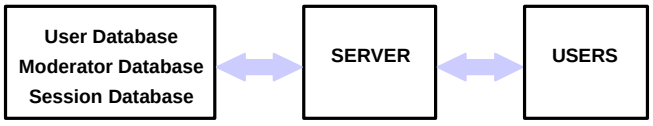

Fig. 6.2. Server Block Diagram

The global session database contains data for the sessions that are accessible to all users (guest access is possible). The restricted session database contains data for the sessions available only to authorized users. Each session entry, regardless of the session type, contains the session name, media tool list (the command line arguments for their execution) and the session information. The media tool list has separated arguments for the user and moderator. There can also be multiple entries if several tools for the same media type exists (for example, both ***VAT*** and RAT can be used for audio conferencing). This is useful because users might not have the same conferencing programs. Each user

or moderator database entry contains the user name, password and the list of restricted sessions in which they are allowed to participate (or moderate).

Each of these four databases can be remotely maintained by the administrator through the use of the MADMIN (Multimedia ADMINistration) application. There are two more files accessed by the server which cannot be changed using MADMIN. These are the administrative password and the list of media types and appropriate media tools. Administrative password cannot be changed remotely for security reasons (even though it can be encrypted). By reconfiguring the list of media types the whole system can be easily reconfigured for the use of new multimedia tools. Each entry in the list contains the media type name (e.g. Audio, or Video) along with the corresponding program tools (***VAT***, ***VIC***,…). This list is not changed frequently so it is not necessary to be able to change it with MADMIN. The server can be started on any computer on the network and it does not have any special requirements.

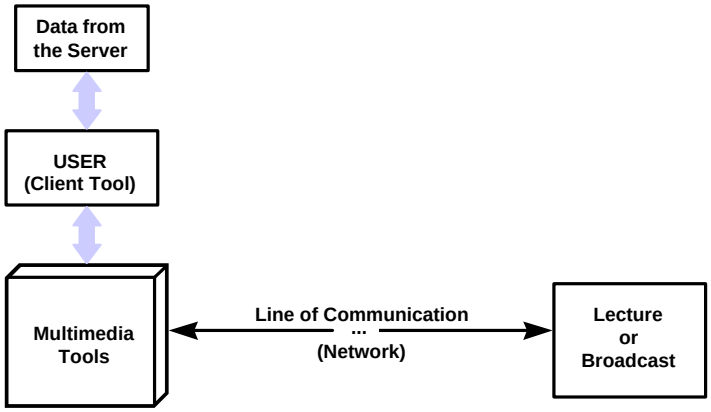

Fig. 6.3. Client interaction

What happens once the client program fetches the session data from the server can be seen in Figure 6.3. The client tool gets the command line for execution of the multimedia tools used in the chosen session. Then it attempts to execute one program for each media type. If not found, it tries to execute another program for the same media type, if it exists. MADMIN (if authorized) can execute both moderator and user command lines while MSTUDENT can execute only user command lines. Once the proper conferencing pro-

gram is executed, it establishes the line of communication through the network and thus joins or starts the lecture, broadcast or some other multimedia event. In this way, the user can access not only real time events, but also stored recordings if the arguments for the appropriate player for media on demand are supplied.

There are two types of client tools (programs): administrative (MADMIN - Figure 6.4) and user (student, MSTUDENT - Figure 6.5). The main differences between the two are that with MADMIN, sessions can be started in a moderation mode and can also be changed, as opposed to MSTUDENT where users can only join the sessions or change their passwords. When changing or adding the session, all major session parameters can be defined (Figure 6.6).

Fig. 6.4. MADMIN

Fig. 6.5. MSTUDENT

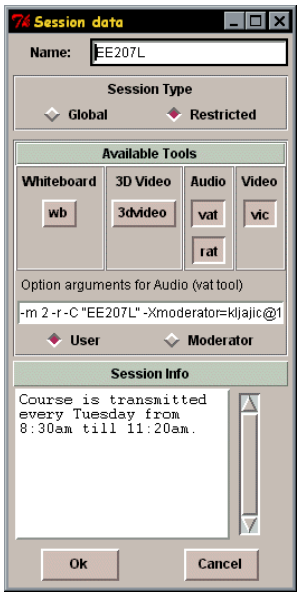

Fig. 6.6. The session parameters

There is one more difference between the two types of session management tools - the way they connect to the server. In order to avoid network loading, MSTUDENT connects to the server only briefly, just to fetch the session data, after which it disconnects. MADMIN remains connected as long as necessary (until the user terminates the connection). Since there must be no errors in the transmitted and received data, the TCP protocol has been used.

6.3.2 Floor control implementation in VIC and VAT conferencing tools

Since lectures over the Internet usually consist of audio and video media (the whiteboard is sometimes used) special attention has been given to the two most common conferencing tools found on the MBone - the Video Conferencing Tool (***VIC***) and the Visual Audio Tool (***VAT***) (see Chapter 5).

Both ***VIC*** and ***VAT*** have been modified to better serve the purpose of ***MTEACH*** because there are several problems associated with their use.

The first problem is that all users in a session have the same rights, meaning that anybody can start an audio or video transmission at any time. Since that can be both annoying and cause network loading, all users have to behave nicely, which is sometimes not entirely possible when a large group of undergraduate students is involved. The other

problem is two way communication between the students and the professor. When asking questions, one would have to start transmitting at least audio, which might interfere with the professor's transmission and/or other student's transmissions.

Fig. 6.7. Modified VIC screen shot

Two conditions must be satisfied - the existence of a moderator who moderates the session (this can be the same person who is lecturing), i.e. one who would be able to enable or disable other people's transmissions, and the existence of a mechanism for students to ask questions without loading the network. The former condition is floor control.

Floor control can be implemented as a separate application (the floor control manager) which could "orchestrate" all the running applications that support the conference bus or it can be incorporated within one of the existing applications (Figure 6.8). If more applications supporting the bus emerge, a separate floor control application would be a better solution because it would be easier to upgrade it and debug it as a separate program. However, since ***VIC*** and ***VAT*** are currently the only publicly available conferencing programs (with source code) that support the conference bus, it is necessary to modify one of these applications so that floor control can be implemented. In this case, ***VAT*** is the application of choice. The reason is that the audio media is probably going to be used in most lectures.

When initially started, these modified applications have the transmission disabled on the student's part. All sources are also muted. Every new audio source (meaning a ***VAT*** tool joining the session) is compared with the moderator identifier provided via MSTUDENT. If a match is found, that source is unmuted. In general, every new source that is detected by some application is queried whether it should be muted or not by sending the "source query" signal to the floor control manager through the conference bus. The manager compares it to the previously supplied moderator identifier and returns a "source enabled" or a "source disabled" message.

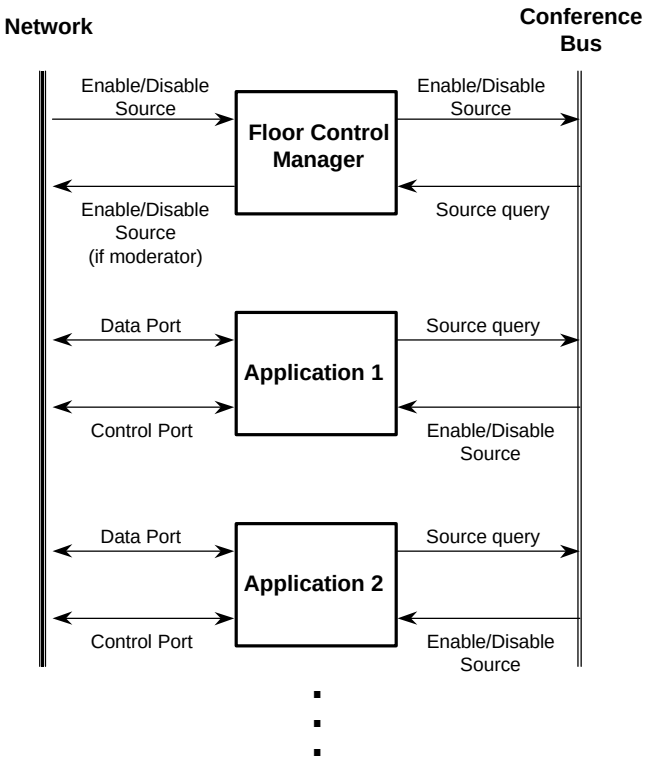

Fig. 6.8. Floor Control Block diagram

The source can also be muted or unmuted by the floor control manager if it receives such a message over the network (the message must come from the session moderator, otherwise it is discarded). This mechanism can be observed in Figure 6.8 where the generalized system is shown. In case of this particular implementation, the ***VAT*** tool has been modified to also serve as the floor control manager.

6.3.3 Question asking

Once the session starts, students can press the question button if they wish to ask question. The moderator can see who wants to speak from the participant list. Those who want to speak will have their names highlighted. The moderator can then enable that student, which would enable the student's application transmission controls and also unmute student in other participant's applications (which is done by the floor control manager). The moderator can also disable the student after he is done with the transmission (question/answer). Figures 6.9 and 6.10 represent screen shots of the modified ***VAT*** tool. On the left is ***VAT*** running in moderator mode and on the right is ***VAT*** running in student mode.

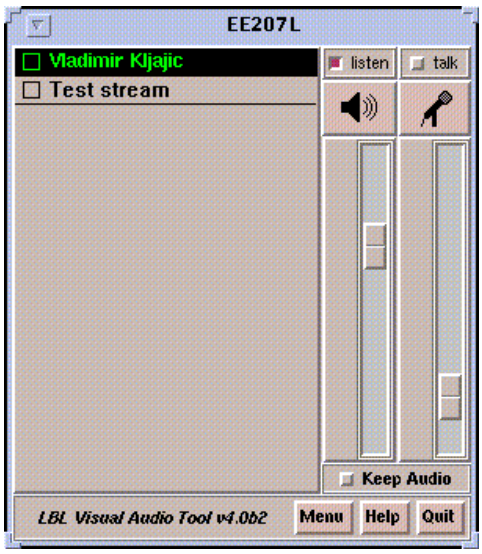

Fig. 6.9. Modified VAT (moderator mode)

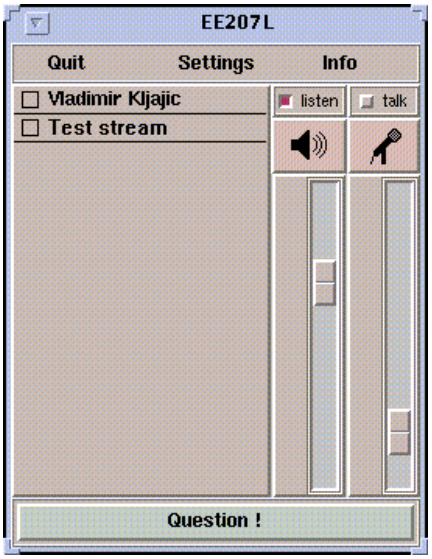

Fig. 6.10. Modified VAT (student mode)

This mechanism is conducted over the application's control port. Every action (question, enabling or disabling the student) produces one multicast packet which is transmitted to all participants. Since the packets are never longer than approximately 20 bytes, network load is avoided.

6.3.4 Security issues

Every time when there exists data which only certain users are authorized to use there is an issue of security. In the case of ***MTEACH*** there are two basic types of information.

The first is information about the “place” and time when the session will occur. By place, we mean the multicast group address (or unicast) and the port number. If the user does not know when and where, he cannot receive the desired data. This system prevents end users from knowing where the session takes place. Only administrators and moderators can see what address and port combination is being used for a certain session. One place where users can see the address and port number is the conferencing tool. In particular case of ***VIC*** and ***VAT***, they have been modified to not show the address and port when started in the lecturing mode. The address and port hiding measures only make finding the information more complicated. It is fairly easy for a knowledgeable user to find the address and port number once the session has started.

The other type of information is the session data itself. This includes audio, video or data packets. Once the session address, port and time are known, it is easy to obtain the session data unless it is encrypted. Most of the conferencing tools have encryption options. In the case of ***VIC*** and ***VAT***, DES encryption is used (though it is not usually compiled with it because of the US export laws). The encryption key is entered and the entire session is encrypted and decrypted using that key. The key is unique for all the session participants and has to be agreed upon by some other means before the session starts (for example, PGP might be used to transfer the key).

There is one more security issue to discuss - prevention of intruder transmissions. This could occur if there was a user who knows the time and place of the session and who wants to do intentional (and unauthorized) transmission of data to all session participants. The implemented floor control system prevents this from happening, because the users have the name of the session moderator and their applications (***VIC*** and ***VAT***) would “listen” only to that particular source, and only to controls from that source. In effect, all of the applications (***VIC*** and ***VAT***) take commands from the floor control manager which in turn accepts enable/disable commands only from a certain source. The only way for transmitted data to be received by users is that it comes from the proper source. The

source in this case is identified by the CNAME (canonical name) identifier used in the RTP protocol so the only way to have an intrusion is if someone deliberately represented itself as the moderator, which would mean tampering with the CNAME. The CNAME is unique and it is usually either the complete user's email address or just the host name in case of a single user hosts (the host name part in the email address should be in dotted quad format, though that is not necessary).

6.4 Summary

This chapter covered the main part of this thesis – implementation of the ***MTEACH*** system.

Session information can be retrieved by using the MSTUDENT application, while MADMIN is used by the session administrator and the session moderators to change and invoke the sessions.

Floor control is implemented in the ***VIC*** and ***VAT*** conferencing tools, with ***VAT*** also acting as a floor control manager.

A question asking mechanism serves to reduce network load when asking questions during the course of a lecture. It also provides the moderator with an easy way to see who is asking the question and enable the participant to transmit the question.

The entire system is not heavily secured. Some security measures exist only as a pre-caution to disable accidental misuse. If used for transmitting secret information, further security measures would have to be deployed.

7. CONCLUSION

7.1 Results

MTEACH has been tested on Sun SPARC workstations and partially on PC based computers. Test results have shown that the initial goals were met:

- Information about the lectures can be easily accessed and changed. Servers can be easily mounted on various computer systems and both servers and clients can be run across multiple platforms with no additional programming.

- The network bandwidth necessary for the operation of this system does not exceed 100 kbits/s in the case of a head-and-shoulder frame with no more than 6 frames per second (H.261 using CIF picture format). In case of large (SCIF) picture format transmission, used to transmit the professor's notes (similar to an overhead projector), the CellB standard was used at 1 frame per second. The bandwidth was approximately 200 kbits/s. See the Appendix D for more information on the video bandwidth necessary for transmissions with the ***VIC*** tool. If GSM compressed audio is transmitted, 13 kbits/s should be added to the overall data rate.

- The system is low-cost, with video capture boards and cameras as the only special purpose hardware.

- Floor control is successfully implemented.

There were several problems associated with the transmission and receiving of audio and video. Sometimes, received audio is not synchronized with the video. This can be annoying, especially when following a lecture. The solution to this would be to enable ***VIC*** and ***VAT*** to synchronize their playback. While ***VAT*** has certain functions that would facilitate this implementation, ***VIC*** does not. Again, as for floor control, if more than two media types are used, it might be feasible to design a sync control manager.

The other problem that was observed was occasional instability of ***VIC*** and ***VAT***. This is due to the fact that they have not yet been tested enough, but considering their price (free), a couple of crashes per week can be tolerated. Since they crash only if the data transmission formats are changed often, there is no concern that they might crash during lecture.

7.2 Future Research

There is a lot of room for improvement in this system. The client and server part of ***MTEACH*** might be modified to support transfer of encrypted data. The PGP (Pretty Good Privacy) system of encryption, which is publicly available, might work quite well. As mentioned before, a separate application could be built to serve as a floor control manager. That would be feasible in large scale multimedia distribution systems, with many different type of media used.

The reliability of the audio and video transmission can be improved at expense of increasing the necessary network bandwidth. Either FEC (Forward Error Correction) could be used, or the data could be filtered with a low pass filter and the low frequency version would be sent along with the normal one. Since the packet containing the low frequency part is smaller, the chances of it arriving at the destination are better than that of the normal, larger, packet. In case the large packet does not arrive within a specified time frame, the low frequency version would be used instead. This system is already in use in the RAT tool (see Chapter 5).

Another improvement is data scaling. Since different paths in the network have different bandwidths (and hence in general, every user has a different access speed), the data is usually transmitted at the speed of slowest link. An approach to solve this problem is to

scale the data in such a way so that every user receives the maximum amount of data allowed by the network bandwidth. This is usually done by filtering the audio or video, thus separating them in several layers. The path with the highest bandwidth would receive all the layers, and paths with lower bandwidths would receive less layers, depending on their actual bandwidth. Applications (or devices) known as transceivers would be in charge of data scaling. The RTP protocol already supports them and ***VAT*** can act as a transceiver. More about data (in this case video) scaling can be found in [55].

BIBLIOGRAPHY

[1] C. H. Olgren and L. A. Parker, *Teleconferencing Technology and Applications*, Artech House Inc., 1983.

[2] C. An, B. T. Barcelo, J. A. Inkrott, A. R. Snowdon, K. J. Trojniar, "A Multimedia Distance Learning Trial Using ISDN BRI," *AT&T Technical Journal*, vol. 72, no. 1, January/February 1993, pp. 15-21.

[3] W. J. Clark, "Multipoint Multimedia Conferencing," *IEEE Communications Magazine*, May 1992, pp. 44-50.

[4] T. Algra, "A PC-Based Real-Time Multimedia Tele-Education System," *Proceedings of* the *IEEE Global Telecommunications Conference*, vol. 2, November 28 - December 2 1994, pp. 881-885.

[5] J. Robinson, E. Rubinov, C. Toulson, B. Prasada, S. Sabri, N. Goldberg and G. Vonderweidt, "A Multimedia Interactive Conferencing Application for Personal Workstations," *IEEE Transactions on Communications*, vol. 39, no. 11, November 1991, pp. 1698-1707.

[6] F. Tobagi, "On-Line Distance Learning System," *WWW Page*, <*http://minas.stanford.edu/project/project.html*>, Stanford University.

[7] J. Harju, V. Kosonen, C. Li, "Quality and Performance of a Desktop Video Conferencing System in the Network of Interconnected LANs," *Conference on Local Computer Networks*, October 1994, pp. 365-371.

[8] C. McElroy, "Speech Coding," *WWW Page*, <*http://wwwdsp.ucd.ie/*speech*/tutorial/speech_coding/speech_tut.html*>, DSP Research Group, University College, Dublin.

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

[10] J. Woodard, "Speech Coding," *WWW Page*, <*http://www.mobile.ecs.soton.ac.uk/speech_codecs/index.html*>.

[11] S. M. Redl, M. K. Weber, M. W. Oliphant, *An Introduction to GSM*, Artech House, Norwood, MA., 1995.

[12] J. Scourias, "Overview of the Global System for Mobile Communications," *WWW Page, <http://ccnga.uwaterloo.ca/~jscouria/GSM/index.html>*, 1997.

[13] ISO/IEC International Standard IS 11172-3, "Information Technology - Coding of Moving *Pictures* and Associated Audio for Digital Storage Media at up to about 1.5 Mbits/s - Part 3: Audio."

[14] D. Pan, "A Tutorial on MPEG/Audio Compression," *IEEE Multimedia*, vol. 2, no. 2, Summer 1995, pp. 60-74.

[15] *Solaris XIL 1.1 Imaging Library Programmer's Guide*, SunSoft, November 1993.

[16] E. J. Delp and O. R. Mitchell, "Image Compression Using Block Truncation Coding," *IEEE Transactions on Communications*, vol. COM-27, September 1979, pp. 1335-1342.

[17] H. B. Mitchell, N. Zilverberg, M. Avraham, "A comparison of different block truncation coding algorithms for image compression," *Signal Processing: Image Communication*, vol. 6, no. 1, March 1994, pp. 77-82.

[18] R. Aravind, G. L. Cash, D. L. Duttweiler, H. Hang, B. G. Haskell, A. Puri, "Image and Video *Coding* Standards," *AT&T Technical Journal*, January/February 1993.

[19] ISO/IEC JTC1 Draft International Standard 10918-1, "Digital Compression and Coding of *Continuous*-tone Still Images, Part 1, Requirements and Guidelines," November 1991.

[20] G. K. Wallace, "The JPEG Still Picture Compression Standard," *Communications of the ACM*, vol. 34, no. 4, April 1991, pp. 30-44.

[21] Committee Draft of Standard ISO11172, Coding of moving pictures and associated audio. *ISO*/MPEG 90 176, December 1990.

[22] D. LeGall, "MPEG: A Video Compression Standard for Multimedia Applications," *Communications of the ACM*, vol. 34, no. 4, April 1994, pp. 47-58.

[23] F. Tobagi, "STANFORD MULTIMEDIA INSTRUCTIONAL NETWORK: An On-Line Distance Learning System Using Digital Video and Multimedia Networking Technologies," *WWW Paper*,

<http://minas.stanford.edu/project/SMIN.ps>, Stanford University, October 1994.

[24] ITU-T Recommendation H.261, Video Codec for Audiovisual Services at p x 64 kbits, March 1993.

[25] M. Liou, "Overview of the px64 kbit/s Video Coding Standard," *Communications of the* ACM, vol. 34, no. 4, April 1991, pp. 60-63.

[26] R. Stevens, *TCP/IP Illustrated Volume 1*, Addison-Wesley, Reading, Mass., 1994.

[27] J. B. Postel, "Internet Protocol," RFC 791, September 1981.

[28] J. B. Postel, “User Datagram Protocol,” RFC 768, August 1980.

[29] J. B. Postel, “Transmission Control Protocol,” RFC 793, September 1981.

[30] H. Schulzrinne, S. Casner, R. Frederick and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications," RFC 1889, January 1996.

[31] S. Deering, “Host Extensions for IP Multicasting,” RFC 1112, August 1989.

[32] R. Talpade, M. H. Ammar, "Single Connection Emulation: An Architecture for Providing a Reliable Multicast Transport Service," *Proceedings of the 15th IEEE Intl Conference on Distributed* Computing *Systems*, Vancouver, June 1995, pp. 144-151.

[33] TASC Inc, "RAMP: A Reliable Adaptive Multicast Protocol," *WWW Page*, *<http://www.tasc.com/simweb/papers/RAMP/ramp.htm>*, TASC Inc.

[34] S. Casner and S. E. Deering, “First IETF Internet Audiocast,” *Computer Communication* Review, vol. 22, no. 3, July 1992, pp. 92-97.

[35] V. Kumar, "MBone (or IP Multicast) Information Web," *WWW Page*, *<http://www.mbone.com>*, 1997.

[36] M. Macedonia and D. Brutzman, "MBone Provides Audio and Video Across the Internet," *Computer*, vol. 27, no. 4, April 1994, pp.30-36.

[37] MBone PC Guide, *WWW Page*, *<http://cosmos.kaist.ac.kr/salab/project/mbone-ng/mbonepc.html>*, System Architecture Laboratory at CS department at KAIST (Korean Advanced Institute of Science and Technology).

[38] K. Savetz, N. Randall and Y. Lepage, *MBONE: Multicasting Tomorrow's Internet*, IDG Books WorldWide, Foster City, CA., 1996.

[39] V. Jacobson, "The MBone - Interactive Multimedia on the Internet," *Slides from a presentation given* at *U.C. Berkeley (and on the MBone), <ftp://ftp.ee.lbl.gov/talks/vj-ucb-feb17.ps.Z>*, February 1995.

[40] MICE NSC User documentation, *WWW Page, <http://www-mice-nsc.cs.ucl.ac.uk/mice-nsc/tools/>*, MICE National Support Center, England.

[41] M. Handley, "The SDR Session Directory," *WWW Page, <http://ugwww.ucs.ed.ac.uk/mice/archive/sdr.html>*, Department of Computer Science, University College London, April 1996.

[42] M. Handley, V. Jacobson, "SDP: Session Description Protocol," INTERNET DRAFT, November 1995.

[43] M. Handley, J. Crowcroft, *The World Wide Web: Beneath the Surf*, UCL Press, London, 1995.

[44] S. McCanne and V. Jacobson, "VIC: A Flexible Framework for Packet Video," *ACM Multimedia*, November 1995, San Francisco, CA, pp. 511-522.

[45] LBNL Audio Conferencing Tool (VAT), *WWW Page, <http://www-nrg.ee.lbl.gov/vat/>*, Lawrence Berkeley Laboratories, CA.

[46] UCB/LBNL Video Conferencing Tool (VIC), *WWW Page, <http://www-nrg.ee.lbl.gov/vic/>*, Lawrence Berkeley Laboratories, CA.

[47] LBNL Whiteboard Tool (WB), *WWW Page, <http://www-nrg.ee.lbl.gov/wb/>*, Lawrence Berkeley Laboratories, CA.

[48] V. Hardman, A. Sasse, M. Handley and A. Watson, "Reliable Audio for Use over the Internet," *Proceedings of Inet'95*, Honolulu, Hawaii, June 1995.

[49] The RAT (Robust Audio Tool) Project, *WWW Page, <http://www-mice.cs.ucl.ac.uk/mice/rat/index.html>*, MICE National Support Center, England.

[50] R. Frederick, "Experiences with real-time software video compression," *Proceedings of the Sixth International Workshop on Packet Video*, Portland, OR, September 1994.

[51] R. Frederick, "Network Video (NV)," Xerox Palo Alto Research Center, *On-Line Software, <ftp://ftp.parc.xerox.com/net-research>*.

[52] A. M. Fleming, "MICE-NSC IVS User Guide," *WWW Page*, <*http://www.mcg.gla.ac.uk/projects/mice-nsc/ivsmanual.html*>, MICE National Support Center, England.

[53] T. Turletti, "IVS's Home Page," *WWW Page*, <*http://zenon.inria.fr:8003/rodeo/personnel/Thierry.Turletti/ivs.html*>, Sophia Antipolis, France.

[54] T. Turletti, "The INRIA Videoconferencing System (IVS)," *ConneXions - The Interoperability Report Journal*, vol. 8, no 10, October 1994, pp. 20-24.

[55] S. McCanne, "Scalable Compression and Transmission of Internet Multicast Video," *Ph.D. Thesis*, University of California Berkeley, UCB/CSD-96-928, December 1996.

APPENDIX A
VIC AND VAT MANUALS

A.1 VAT Manual (VAT version 4.0b2)*

SYNOPSIS

vat [-aAcdEjJklLMnnsSv] [-F device] [-f audiofmt] [-g geometry]
[[-m host/port[/fmt]] [-N sessionname] [-C conferencename]
[-K key] [-P priority] [-t ttl] [-U socket] [-u script] [dest]

DESCRIPTION

VAT allows users to conduct host-to-host or multihost audio teleconferences over an internet (multihost conferences require that the kernel support IP multicast). On most architectures, no hardware other than a microphone is required - sound I/O is via the built-in audio hardware. On DEC systems, an AudioFile server must be running.

OPTIONS

(Note that all the parameters set by the following flags can also be controlled by X resources (which all have `reasonable' defaults) so one should not need to give ***VAT*** any flags in the usual case. The flags only exist to temporarily override some resource.)

-a Enable automatic gain control on the output (speaker).

-A Enable automatic gain control on the input (mike).

---

* As presented in the ***VAT*** UNIX manual pages, with additional screen shots.

-c Start up in `conference mode' (see description below).
(This flag is the opposite of -l.)

-C Use conference as the title of this ***VAT*** window. If no -C flag is given, the destination address and port are used as the window title.

-d Start up with the VU meters disabled (this can be changed using the `Disable Meters' checkbox on the auxiliary controls panel.

-E Include a checkbox in the auxiliary controls panel for specifying that echo cancellation is being performed externally (i.e., in hardware). This option can also be effected by setting the X resource Vat.externalEchoCanel to "true."

-f Sets the output audio packet format to fmt. (Note that it is not necessary to set an input format since ***VAT*** will accept any packet format it knows about). Currently available audio formats are:

pcm 78Kb/s 8-bit mu-law encoded 8KHz PCM (20ms frames)
pcm2 71Kb/s 8-bit mu-law encoded 8KHz PCM (40ms frames)
pcm4 68Kb/s 8-bit mu-law encoded 8KHz PCM (80ms frames)
idvi 46Kb/s Intel DVI ADPCM (20ms frames)
dvi2 39Kb/s Intel DVI ADPCM (40ms frames)
dvi4 36Kb/s Intel DVI ADPCM (80ms frames)
gsm 17Kb/s GSM (80ms frames)
lpc4 9Kb/s Linear Predictive Coder (80ms frames)

(The idvi encoding was contributed by Jack Jansen of CWI. The gsm coder was contributed by Carsten Bormann of the Technische Universitaet Berlin. The lpc coder was contributed by Ron Frederick of Xerox PARC).

The default audio format can be set with the audioFormat X resource. It defaults to pcm.

-g Override the default window geometry with geometry, which should be a standard X geometry string.

-j Start up with audio output to the external audio jack. This flag can be defaulted by setting the X resource Vat.speaker to false.

-J Start up with audio output muted.

-k Keep all sites in the `Conference Hosts' panel. Normally sites that exit are deleted from the panel. With -k, sites that exit are grayed-out but not deleted.

-K Use key as the encryption key for this ***VAT*** session.

-l Start up in `lecture mode' (see description below). This flag can be defaulted by setting the X resource Vat.lectureMode to true.

-m Serve as a `mixer' for host host which is listening on port port optionally using audio packet format fmt. The mix function allows ***VAT*** to function as `application level gateway'. There are three uses for this function:

Bandwidth Adaptation

Say you want to participate in a conference at work that is using 64Kb/s PCM but you to join over the 9.6Kb/s line to your home. If there is a machine "work" at work that can proxy for your machine "home" at home, this could be set up as:

At work: vat -m home/5678/lpc4

At home: vat work/5678/lpc4

Concentration

If a conference has two communities connected by a low speed link, it's desirable to mix multiple conversations into a single data stream rather than shipping them over the link individually. If, say, the gateway machines were "foo" and "bar" and the conference were on the default multicast address, this would be set up as:

On foo: vat -m bar/5678

On bar: vat -m foo/5678

(Note that the port used by the mixers has to be different than the port used by the conference).

Format Conversion

Say organization A uses Etherphone audio packet format for internal conferencing but wants to have a joint conference with organization B that uses G.278 packet format. If a machine AtoB at A can be used to convert/concentrate traffic to B via a peer BtoA, the two conferences can be combined with:

on A's AtoB: vat -m BtoA/5678/pcm Aaddr/Aport/etherphone

on B's BtoA: vat -m BtoA/5678/pcm Baddr/Bport/g278

Note that the two conference environments need have nothing in common - they can use different audio formats, addresses and/or ports. The only agreement needed is over the peer-to-peer communication between AtoB and BtoA.

-M Start up with audio input unmuted.

-n Force `native' ***VAT*** packet format and default address. (This flag is the opposite of -v).

-N Use session, in lieu of your user name and local host, to identify you to other sites. If no -N flag is given, the X resource Vat.sessionName is used.

-P Use priority as this ***VAT*** window's priority for obtaining the audio device. All ***VAT*** windows have a priority which is typically set by the X resource Vat.defaultPriority (defaults to 100) but this can be overridden by a -P flag. If a window requests the audio (because new network data arrived or the mike has been unmuted) and the window currently holding the audio is either lower priority or hasn't used audio for Vat.idleHoldTime seconds, the audio holder immediately gives it up. Otherwise the new window's request is ignored. (Vat.idleHoldTime provides hysteresis to prevent `thrashing' when two conferences go active at about the same time; the priority provides a way to distiguish `background' windows, say a radio station broadcast or a `directory' window of people reachable via ***VAT***, from `foreground' activity like a particular audio conference so ***VAT*** can make better decisions on what should get the audio).

-s Start up with audio output to the internal speaker. (This flag is the opposite of -j.)

-S Make new sites come up `suppressed' (the check box next to the sitename will be checked and you will have to click on it to hear the site speak). This flag is intended for something like meeting audiocasts where a moderator wants to have control over who is able to speak. This flag can also be set by the Vat.muteNewSites X resource.

-t Set the multicast ttl (time-to-live) to ttl. (The ttl is ignored if the destination address is not an IP multicast address). If no -t flag is given, the value of the X resource Vat.defaultTTL is used.

-U Use the unix-domain stream socket specified by socket for audio I/O. Some process should bind to and listen on this socket before ***VAT*** is run. The data is raw 8khz mulaw samples.

-u Use scrip as the tcl program to build the user inter face, rather than the built-in script. You can also override ***VAT***'s built-in tcl script with your own tcl code in $HOME/.vat.tcl.

-v Use a packet format and default multicast address that is compatible with ISI's vt (voice teleconferencing tool). ***VAT*** and vt can interoperate but vt does not participate in ***VAT***'s `session' protocol so the names of conference participants running vt will not show up in the ***VAT*** `Conference Hosts' window (IP addresses for vt participants appear instead). In vt compatibility mode, the default destination host or multicast group is controlled by the X resource Vat.defaultVTHost rather than Vat.defaultHost. The -v flag can be defaulted by setting the X resource VTMode to true.

Note: In addition to invoking the "quit" button, typing 'q', 'Q', ctrl-C or ctrl-D anywhere in the window will terminate ***VAT***.

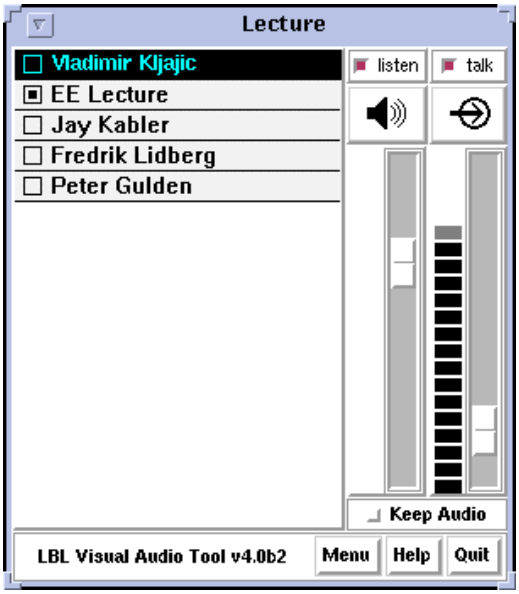

Fig. A.1. VAT main window

The ***VAT*** window is divided into two parts: the right has controls for the local audio and the left has a status display of the hosts participating in the current conference. The audio controls consist of two sliders that control the mike and playback gain, a button to toggle output between the built in speaker and the headphone jack, buttons to mute/unmute either the mike or speaker, and buttons to control acquistion of the audio hardware. To adjust the sliders, either click at the desired position or click and drag the slide button to the desired position. Just to the left of each slider is a VU meter. A rule of thumb is to adjust the mike and speaker gain sliders so the peak readings on the meter are about 80% of full scale. To change the audio output line (i.e., speaker, headphone, line-out, etc.) click on the speaker icon (it should change to a headphone icon). Additional clicks will round-robin among the available lines. If there is only one option, the button will be disabled. Similarly, click on the mike icon to select among the input lines. To toggle mike/speaker muting, click on the appropriate mute button. The button will be highlighted when muting.

The Conference Hosts window lists every site currently participating in the conference, with your site always listed first. The participant name is displayed in a box that is highlighted whenever that site is speaking and grayed-out if a "session" message from

the site hasn't been received for at least 30 seconds (each ***VAT*** sends "session" message every 6 seconds) - this usually indicates that the site has lost connectivity or that ***VAT*** has been aborted or stopped. There is a checkbox to the left of each participant name. Clicking on the box will cause audio from that participant to be discarded instead of played (for example, this might be used to suppress a site that is generating echoes). Names, by default, appear in the form user@host, but can be arbitrarily specified by each participant (i.e., with -N).

Multiple VAT Windows

One host can be running an arbitrary number of ***VAT*** sessions (presumably with different destination addresses). However, since most workstations have only one set of audio hardware, only one of those sessions will be able to access the mike and speaker. For the most part, the ***VAT*** sessions will automatically follow the action. If you unmute the mike or press the "Keep" button, the audio device will be acquired by that session and the session that previously held the audio will relinquish it. ***VAT*** displays its title bar in an oblique font when the audio is not being held.

A ***VAT*** session will also acquire the audio if there is input from the network. But to prevent a background ***VAT*** session from stealing the audio from the foreground session, you can toggle the "Keep" button. When the "Keep" button is highlighted, ***VAT*** will relinquish the audio only if there is a user demand in another window (i.e., unmuting the mike or selecting the Keep button).

The audio can be explicitly released by clicking on the "Release" button. Note however that if no other ***VAT***s are active and there is input from the net with idleHoldTime seconds, the ***VAT*** session that just released the audio will re-acquire it.

Participants in a multi-site conference often want to have 'side conversations' that don't bother the rest of the conference participants. ***VAT*** has some support for establishing side conversations: If you middle-click on the name of some site in the conference hosts window, a new ***VAT*** window will be created that talks only to that participant (it sends unicast datagrams rather than multicast). If that other participant also middle-clicks

on your site, you can have a private conversation between just your two sites using the newly created ***VAT*** windows. Note: due to a 'bug' in the way most systems implement multicast, if you create a new window aimed at a particular participant but they haven't created a window aimed at you, they will hear you speaking in the main conference window and may not realize that your audio is being sent only to them and not multicast. One can view this either as a feature (it provides a semi-private channel you can use to ask someone to set up a side conversation) or a bug (it often leads to strange, one-sided conversations where one side multicasts and the other doesn't).

Note that a site that's running as a 'mixer' (-m flag) for some other site should not run multiple ***VAT*** windows: The audio hardware is needed to provide the timing for the mixing function and the window doing the mix will cease to function if it loses control of the audio hardware.

Auxiliary Controls

Clicking on the "Menu" label at the bottom of the ***VAT*** window will cause a panel of auxiliary controls to open.

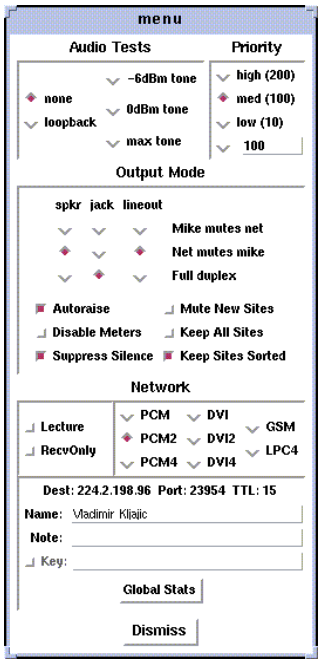

Fig. A.2. VAT Auxiliary Controls window

The Audio Tests buttons will enable some audio test modes. These should not be selected during a conference. The loopback mike button will cause input from the mike to be sent to the local speaker/jack. This might be useful for checking levels and debugging cable problems but the 20ms delay from input to output makes talking in this mode almost impossible. The three tone buttons will generate one of three reference tones through the local speaker. Level setting should generally be done with the -6dBm tone.

The Output Mode buttons control what ***VAT*** will do to avoid feedback/echo from the mike to the speaker. In mike mutes net mode, ***VAT*** will mute the speaker whenever it thinks that you are talking, while in net mutes mike mode, ***VAT*** will mute the mike whenever input from the network arrives. In full duplex mode, ***VAT*** will assume that feedback can't happen and do nothing to avoid it. In echo cancel mode, ***VAT*** will attempt to eliminate echoes by doing some fancy signal processing. (EchoCancel requires the BSD sound driver - it is disabled when running ***VAT*** under Sun OS because the Sun driver does not provide any mechanism to time correlate audio output and input). The internal speaker should only be used in `speakerphone' or `echo cancel' mode - selecting `headphone' mode for it will result in your site injecting a lot of unpleasant echoes into the conference. The headphone jack should be set to `FullDuplex' mode if you have headphones plugged into it and `MikeMutesNet' or `EchoCancel' mode if you have an external amp and speaker plugged into it.

There are two type-in boxes (see figure) at the bottom of the Auxiliary Controls panel. The one labeled `Name' can be used to change the session name announced to other sites. The one labelled `Key' can be used to specify an encryption key (see next section).

Encryption

Since ***VAT*** conversations are typically conducted over open IP networks there is no way to prevent eavesdropping, particularly for multicast conferences. To add some measure of privacy, ***VAT*** allows the audio packet streams to be DES encrypted. Pre-

sumably only sites sharing the same key (and, of course, the NSA) will be able to decrypt and listen to the encrypted audio.

Encryption is enabled by entering an arbitrary string in the key box (this string is the previously agreed upon encryption key for the conference - note that key distribution should be done by mechanisms totally separate from ***VAT***). Encryption can be turned off by entering a null string (just a carriage return or any string starting with a blank) in the key box.

X RESOURCES

The following are the names and default values of X resources used by ***VAT***. This list is incomplete. Consult the tcl code in vat.tcl for the complete set.

Vat.lectureMode:	false
Vat.inputPort:	Mike
Vat.outputPort:	Speaker
Vat.speakerMode:	Speakerphone
Vat.jackMode:	Headphone
Vat.mikeGain:	32
Vat.speakerGain:	180
Vat.jackGain:	180
Vat.mikeAGC:	false
Vat.mikeAGCLevel:	32
Vat.speakerAGC:	false
Vat.speakerAGCLevel:	32
Vat.maxPlayout:	6
Vat.muteNewSites:	false
Vat.siteDropTime:	30

Vat.maxPlayout is the maximum `play out' delay, in seconds, that can be tolerated. I.e., ***VAT*** dynamically adapts to delays introduced in the network by delaying the play out of a remote site's audio packets. The range of adaptation is limited by the size of a buffer in ***VAT*** and this parameter essentially sets the size of that buffer. Setting maxPlay-out larger than 10 seconds will probably result in poor ***VAT*** and system behavior due to excessive paging activity.

VAT has two different modes of adapting the playout delay, one more suitable for an interactive, multi-party discussion or conference and the other more suitable for listening to a speech or lecture. The two modes differ in how quickly they `forget' the delay ***VAT*** introduces to adapt to transient network congestion: In Conference mode ***VAT*** attempts to minimize the delay (since large delays make interactive conversations difficult) but this usually results in more lost packets when the delay becomes too short handle the next congestion event. In Lecture mode ***VAT*** attempts to minimize lost packets by reducing delays very slowly. This results in the clearest playback but interactivity may suffer.

Conference mode is the default when ***VAT*** starts up unless the - l flag is given or the X resource lectureMode is set to true. There are radio buttons in the network section of the Auxiliary Controls panel to switch between Conference and Lecture modes. The switch can be made at any time and takes effect immediately.

Type-in Boxes

Since there's no standard for X keyboard input we had to roll our own and ***VAT***'s type-in boxes need some explanation:

To enter something new in a type-in box, position the mouse in the box and click to insert a cursor. To modify the existing contents of the box, click with the left mouse but-ton where you want to append (or click-and-drag if you want to replace) then start to type. If you hit carriage return, your new input will be acted on. If you hit escape or bell (^G) your new input will be discarded. If the string is too long for all of it to be visible, the right mouse button can be used to drag invisible portions into view. Hold down the button and drag the text the direction you want it to move.

While typing, a tiny subset of emacs-like line editing commands are available: ^H or DEL (delete previous character), ^D (delete character), ^F (forward char), ^B (backward char), ^A (beginning of line), ^E (end of line), ^W (delete previous word), ^U (delete everything).

Statistics

Clicking on a name with the left mouse button will bring up a window with packet statistics for that site. The window will remain mapped as long as the button is depressed. There are three of numbers. The last column is the aggregate statistics since ***VAT*** started, and the first column is the difference between the last update time and the current time. The middle column is a smoothed average of the first column with units depend on the statistic. The smoothing filter is a exponentially weighted average with gain Vat.statTimeConst. All of the averages (except total packets) are displayed as a percent of number of packets sent (which is actually an estimate based on the number of packets received plus those estimated to be missing).

The statistics are updated every second or so while the window is mapped. If you hold down the shift-key when you bring up the statistics window, it will be mapped permanently and will include stripchart. The stripchart plots the selected statistic against time.

Altenatively, the stats for all the sites can be sent to standard out by typing 's' or 'S' anywhere in the ***VAT*** window. The first entry is the total packets received from the site since starting ***VAT*** and since the last time you requested stats. E.g., `3515/62' means 3515 packets have been received from the site and 62 received since the last time you printed stats. The next four entries are various error counters. Each consists of two numbers separated by a slash. The first number is a count of how many times the error occurred since the last time you requested stats. The second is the same info expressed as percentage of packets received. E.g., `9/3.0' in the `drop' column would mean 9 packets or 3% of the packets received since the last stats output had been dropped. The `lost' column counts packets that didn't make it (this is computed by keeping track of gaps in the packet se-

quence number space). The `drop' column counts packets that arrived but were dropped because the current playout delay was too short to resequence the packet. The `dup' column counts duplicate packets that were discarded (you shouldn't see any of these unless the packets had the misfortune to traverse the T1 NSFNet backbone - the worthless IBM token ring hardware in each NSS loves to duplicate packets). The `order' column counts packets that arrived out of order (this is not an error but it does indicate routing strangeness and will always increase the playout delay). Following the error stats is the current playout delay for the site and the site name.

AUTHORS

Van Jacobson (van@ee.lbl.gov) and Steven McCanne (mccanne@ee.lbl.gov), both of Lawrence Berkeley Laboratory, University of California, Berkeley, CA.

Jack Jansen (Jack.Jansen@cwi.nl) of Stichting Mathematisch Centrum, Amsterdam, the Netherlands, contributed the Intel DVI ADPCM codec.

Ron Frederick (frederic@parc.xerox.com) of Xerox PARC, Palo Alto, CA, contributed the LPC codec which is based on an implementation done by Ron Zuckerman (ronzu@isu.comm.mot.com) of Motorola which was posted to the Usenet group comp.dsp on 26 June 1992.

Carsten Bormann (cabo@cs.tu-berlin.de) and Jutta Degener (jutta@cs.tu-berlin.de) of the Communications and Operating Systems Research Group (KBS) at the Technische Universitaet Berlin contributed the GSM codec.

Steve Casner (casner@isi.edu) of ISI, Los Angeles, CA, and Steve Deering (deering@parc.xerox.com) of Xerox PARC have invested tremendous effort in making ***VAT*** work on a scale far beyond the authors' wildest expectations and have contributed greatly to ***VAT***'s development, both directly (via careful analysis of bugs and useful suggestions) and indirectly (via setting up several global conferences that severely pushed the envelope of ***VAT***'s capabilities).

A.1.1 Changes to VAT

SYNOPSIS

vat [-m 0|1|2]

OPTIONS

-m New ***VAT*** modes:

0 - the original ***VAT***

1 - teacher mode. ***VAT*** interface looks the same, but when the user name is clicked, the option menu of its enabling or disabling appears. The user interface is reconfigured slightly (instead of buttons at the bottom, there are now menus at the top. The color scheme is also changed).

2 - student mode. ***VAT*** interface changes to enable question asking (Figure A.3).

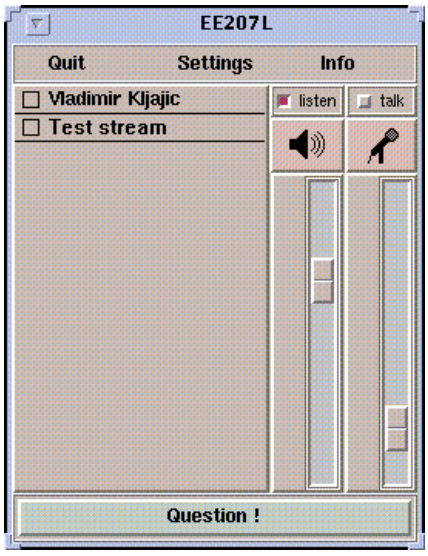

Fig. A.3. VAT Main Window (Student Mode)

X RESOURCES

-Xmoderator - if set, defines a conference moderator
e.g. -Xmoderator=kljajic@128.46.138.173

OPERATION

Teacher Mode

When in this mode, ***VAT*** can send or receive special type of 'lecturing' messages. When it receives message from a user indicating he/she has a question, the color of the student's name changes to cyan, with a black background (if the user decides to withdraw the question, the message is sent and the color returns to normal). If a user's name is clicked, a popup menu appears displaying choices of enabling a user's transmission, disabling it, or getting more information about the user. Once enabled, the user's name becomes green indicating that the user can now transmit.

VAT enables the conference bus and listens to ***VIC***. When ***VIC*** is started in either the student or the teacher mode, it sends a "hello" message over the bus, to which ***VAT*** replies and ***VIC*** then reconfigures the interface appropriately (see the ***VIC*** section).

There are menus the top of the window. Settings replaces the old "Menu" button while "Info" menu contains the "About" information and "Help," replacing the "Help" button.

Student Mode

Again, ***VAT*** can send or receive lecturing messages. In this case, the user interface reconfigures itself - there is a menu at the top, the same as in the teacher mode, and a new "Question" button at the bottom of the window. The default text within that button is "Question?," but when it is depressed it becomes "Question!," changing color, and indi-

cating that the question message has been sent. If depressed again, it toggles back to the default state, sending a question cancellation message.

When the question message from another user is received, his/her name becomes cyan, and when the user is enabled by the teacher, it turns green. If the Xmoderator resource is specified, only messages coming from the moderator (enabling or disabling users) are accepted. If enabled by the moderator, the transmit button is automatically invoked, thus starting the transmission immediately.

All new sources are muted by default, unless they are specified in the Xmoderator resource or an "enable" message for their activation comes from the moderator (teacher).

Note: When ***VAT*** starts in the student mode, the transmit button is disabled until an "enable" message from the teacher arrives. It is invoked and enabled after that, until the "disable" message arrives.

Floor Control

Since ***VAT*** acts as a floor control manager, it manages ***VIC*** in both new modes (teacher and student). When ***VIC*** is run in the teacher or student mode, it mutes all new video sources by default and unmutes them only when ***VAT***'s command is issued over the conference bus.

A.2 VIC Manual (VIC version 2.8)*

SYNOPSIS

vic [-A app] [-B kbps] [-C conference] [-c renderer] [-D device] [-d display]
[-f format] [-F fps] [-H] [-I channel] [-K key] [-M colorfile] [-m mtu]
[-N sessionname] [-o clipfile] [-P] [-r dir] [-s] [-t ttl] [-U interval]
[-u script] [-X resource= value] dest/port[/fmt/ttl]

DESCRIPTION

VIC is an experimental video conferencing tool that allows groups of users to transmit video to each other over an IP Multicast network ("***VIC***" is a contraction of VIdeo Conference). A host must be equipped with a camera and frame digitizer to send video, but no special hardware is required to receive and display it. Audio is handled by a separate application.

The conference is carried out on the address specified by dest. Point-to-point conferences are initiated by supplying a standard IP address, while multiparty conferences use a Class D group address. Port specifies the UDP port to use; fmt overrides the default video encoding (see below); and, ttl specifies the IP time-to-live (see below).

Video is coded in a variety of formats. The default format depends on the host video capture hardware, but can be overridden. ***VIC*** will take advantage of certain hardware compression and decompression units, if present, but since decompression hardware is not always available, all supported coding formats can be decoded in software.

* As presented in the VIC UNIX manual pages, with additional screen shots.

OPTIONS

-A Use the RTPv1 packet format that is compatible with the app argument, which may be nv for Xerox PARC's network video tool, or ivs for the INRIA Video-conference System.

-B Set the maximum value of the bandwidth slider to kbps kilobits per second. If the conference address is a multicast address, the maximum bandwidth is limited by the ttl argument using the conventional MBONE bitrate/ttl limits.

-C Use conference, as the title of this ***VIC*** window. If the -C flag is ommitted, the destination address and port are used as the window title.

-c Use the method indicated by renderer to convert video (typically represented in 24-bit YUV color space) to a format suitable for rendition on the local display.

Renderer may be one of the following:

best - the ``highest quality" renderer for the display

dither - a simple 4x4 ordered dither

ed - a simple, error-diffusion dither (i.e., Floyd-Steinburg dither) quantize straight quantization

gray - 32-levels of gray

true - full 24-bit YUV to RGB translation truegray full 24-bit Y to RGB translation.

The dither, ed, quantize, and gray switches require a color-mapped display (i.e., a display that supports PseudoColor X visuals), while the true and truegray switches require a full-color display (i.e., a display that supports TrueColor or DirectColor X visuals).

-D Use device for video capture. This option is used when more than one video capture device is installed in a host. The argument may be one of the following:

videopix - Sun VideoPix card

parallax - Parallax Xvideo card (jpeg board for sparcs)

jv - Lance Berc's jvdriver server (which supports the DEC J300 and the experimental jvideo boards)

mme - Microsoft's MME multimedia server (on DEC alphas)

tx - DEC tx/pip frame grabber

svideo - SGI starter video library

vl - SGI VL

xil - Sun XIL library

rtvc - Sun /dev/rtvc* (`mme' and `parallax' are not yet implemented).

Without -D, a default device is assumed which can be overridden with the Vic.device resource.

-d Connect to the X server indicated by the display argument.

-f Use the video coding indicated by the format argument for transmission. Format may be one of:

jpeg - Motion JPEG

mpeg - MPEG (not yet implemented)

h261 - CCITT H.261

nv - Xerox PARC Network Video

scr - simple conditional replenishment
cellb - Sun CellB

Not all encodings are compatible with all frame grabbers. For instance, you need JPEG compression hardware in order to source a JPEG stream (e.g., a DEC J300 or SunVideo).

-F Set the maximum value of the frame rate slider to fps frames per second.

-H Use hardware decoding if available. ***VIC*** uses software decoding by default.

-I Use the small integer channel, which must be non-zero, as the channel identifier for group interprocess communication on the local host. This value should be consistent across each group of applications that belong to a single conference, and should be unique across conferences. The session directory tool (sd) will allocate appropriate values. (***VIC*** and ***VAT*** currently use this mechanism to co-ordinate voice activated video switching. ***VAT*** version 2.17, or later, is required).

-K Use key as the encryption key for this ***VIC*** session.

-M Use colorfile as the base lookup table for the errordiffusion or quantizing color rendering algorithms. This file is generated from a colormap using ppmtolut(1). The input to ppmtolut is a ppm(5) file, which contains a single pixel of each color in the colormap (the geometry of the pixmap is irrelevant).

The error-diffusion code can utilize any colormap in which the chrominance level of each color falls on the lattice 0, 16, 32, ... 240. mkcube(1) is a simple utility for generating such colormaps with varying color densities. Note that this option can also be used in conjunction with the ordered dither, but doing so is not advisable. The reason is that an

ordered dither relies on colors uniformly spaced throughout the (5x5x5) RGB color cube, so overriding this default colormap probably will not produce improved results.

-m Set the packet transmission size to mtu bytes, but limited to 1024 bytes (per the application protocol). The default is 1024.

-N Use session, in lieu of your user name and local host, to identify you to other sites. If -N is ommitted, the X resource Vic.sessionName is used.

-o Dump the RTP video stream sourced from the local host to a file.

-P Use a private X colormap.

-r* Use RTIP, instead of IP, to deliver the video data stream. (RTIP is a real-time connection-oriented communication protocol designed by the Tenet group at UC Berkeley. See http://tenet.berkeley.edu for more information). In this case, the RTCP session control is disabled, and only point-to-point communication is allowed.

RTIP is a simplex protocol requiring connection establishment in both directions. The dir argument indicates the direction, and must be one of send, recv, or duplex. If recv, is specified, the application blocks at startup until the connection is established (e.g., ***VIC*** with a -r send option is initiated on another RTIP host, and directed at our client.) If duplex is specified, two simplex channels, one in each direction are established.

-s Don't use shared buffers with the X server.

---

* Note: This protocol is apparently abandoned in version 2.8 of VIC and is now used in the ***MTEACH*** system implementation.

-t Set the multicast ttl (time-to-live) to ttl. (The ttl is ignored if the destination address is not an IP multicast address.) If no -t flag is given, the value of the X resource Vic.defaultTTL is used. A ttl of 1 restricts traffic to the local net; a ttl of 0 restricts the traffic to the local host (e.g., only loopback works, which is useful for testing).

-U Use interval as the update period, in seconds, for the thumbnail sized images of each video source.

-u Use script as the tcl program to build the user interface, rather than the compiled-in script. This is only useful during development.

-X Override the X resource Vic.resource with value.

OPERATION

The main ***VIC*** window (see Figure A.4) provides an abbreviated summary of all sources that are actively transmitting video to the conference address. If no sources are active, the text ``No Network Sources" is displayed in the window. Otherwise, each source has a panel composed of a thumbnail image, identification text, some bit and frame rate statistics, a "mute" button, a "color" button, and a "stats" button.

Fig. A.4. VIC Main Window

The first line of the identification text contains the RTP NAME attribute of the corresponding source, which for ***VIC***, is set using -N, Vic.sessionName, or manually entered in the control menu (see below). The second line of text contains the RTP CNAME attribute and format of transmitted video. If the NAME and CNAME are identical (or very similar), or if the CNAME does not contain a numeric IP address, the second line will instead list the source's IP address (along with the video format). The third line contains filtered frame and bit rate statistics, and a loss indicator. These rates may differ from the actual sender's rate because of network packet drops or loss due to local socket buffer overflows because of CPU saturation. The gain of the low-pass filter used to smooth the statistics is controlled by the Vic.filterGain resource. Note that smoothing can be effectively disabled by setting Vic.filterGain to 1.

Loss Computation

The number of missing packets is computed as the difference between the total number of packets received and the total number of packets sent (which is inferred from sequence numbers). At each sampling interval, a loss percentage is computed by dividing the number of packets missing into the number of packets received during that interval. This percentage is then low-passed filtered (again using the Vic.filterGain constant), which is what finally appears as the parenthesized loss indicator.

Mute & Color

The "mute" button, when selected, causes ***VIC*** to ignore video from the corresponding host. In general, you want to disable any site your not interested in to shed load. Also, it is a good idea to mute your own looped-back transmission to make the encoding process run faster.

The toggle button labeled "color" controls the color disposition of the output. When enabled (by default), video is displayed in color; otherwise, it is displayed in grayscale.

Using grayscale reduces the CPU load (for machines without TrueColor displays) because color dithering is costly. The "color" button does not affect your transmitted video.

Statistics

Clicking on the "stats" button brings up a top level window containing network and video coding statistics for the corresponding source. These statistics are updated in real-time once per second.

The window consists of three panels. The top panel lists the RTP NAME attribute, coding format and geometry, and times of reception of the most recent control and data packets. The middle panel lists the actual statistics, which depend on the underlying coding format. (For example, only H.261 streams have a Bad-GOB stat.) The statistics are displayed in three columns. The first column is the change since the last sampling period (i.e., change over the last second); the middle column is a smoothed version of the first column (smoothing controlled by Vic.statsFilter); and the last column is the accumulated value since startup.

The bottom panel contains a stripchart that displays the statistic from the middle panel that is selected with the radio button. The stripchart plots one point per sampling interval.

Viewing Windows

The thumbnail image is not updated in real-time, but rather every few seconds (the default update interval can be overridden with the X resource Vic.stampInterval or -U). Leftclicking on the image will open a larger viewing window of the corresponding source. Along the bottom of the window are some additional controls and the corresponding site name. The "Dismiss" button will close the window, as will typing a `q' into the window. The popup menu labeled "Size" is used to set the window size, while the menu labeled "Mode" changes the switching mode of the window. By default, the switching mode is "locked," which means that the window is locked onto the indicated

video source. In "browse" mode, ***VIC*** cycles through the set of active video sources, switching participants every Vic.switchInterval seconds. Additionally, when in "browse mode," you can cycle through the participants by hand using the `>' and `<' keys. The last mode is "voice-activated." When running in tandem with vat(1), voice-activated switching causes the video window to switch to whoever is talking (see -I).

You can run multiple voice-activated windows simultaneously, which will cause the remote participants who have spoken recently to be displayed.

The Control Menu

Clicking on the "Menu" button in the lower righthand corner of the main ***VIC*** window will bring up a control panel (see figure), which is composed of three subpanels: transmission controls, encoder controls, and session controls. The transmission controls include a toggle button label "Transmit," which opens the video capture device and begins transmission. The "Lock" toggle button prevents any external agents from automatically initiating or terminating transmission. (For example, a "video silence suppression" algorithm might remotely turn off transmission if there are no interested receivers). The "Release" button terminates the transmission if active, and explicitly closes the capture device (so it may be opened by another application if the device is exclusive access). If the device cannot be opened (e.g., because no capture device is present or the device server isn't running), then a dialog box containing an error message will appear in response to invoking the Transmit button.

Adjacent to the transmission buttons are rate control sliders. The bit rate is limited with the top slider while the frame rate is limited with the bottom slider. ***VIC*** uses the more constraining control to limit the output transmission rate. The frame rate slider ranges from 1 to 30 frames/sec, while the bit-rate slider ranges from 10 to Vic.maxbw kilobits/sec. The actual capture (and encode) rates are displayed above the two sliders.

The "Encoder" panel contains controls for selecting the coding format, video image size, coding quality level, device ports, signal type, and device. Not all options are supported by all devices. The upper lefthand panel contains a list of supported coding for-

mats, which may be changed at any time. Formats that are not supported by the underlying device (or by software compression) are grayed out and disabled.

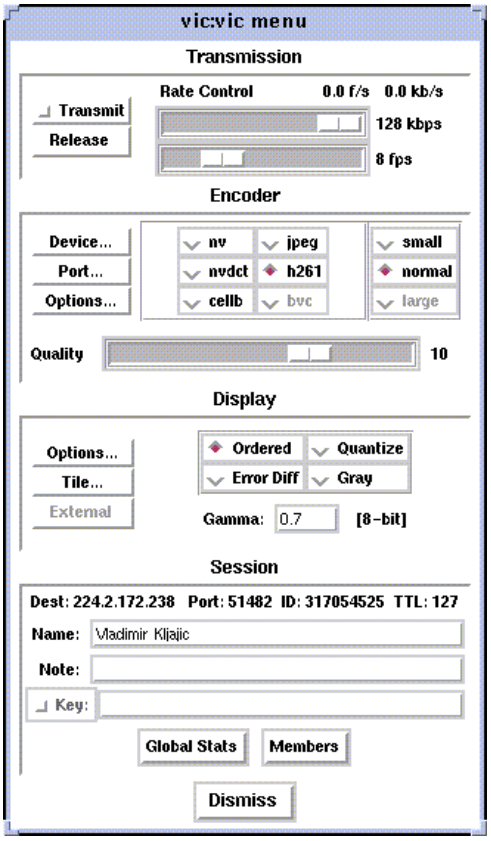

Fig. A.5. VIC Control Panel Window

The video image size is controlled by selecting generic "small," "normal," and "large" formats. The actual size depends on the coding format and underlying signal type. In general, NTSC images are 640x480 (lg), 320x240 (norm), or 160x120 (sm); PAL images are 768x576 (lg), 384x288 (norm), or 192x144 (sm); and H.261 images are converted from their native signal size to CIF size 352x288 (norm) or QCIF size 176x144 (sm). If a size is not supported by the underlying hardware, the corresponding button will be disabled.

To the right of the size selector is the device selector. Typically, a single binary contains support for only one device type, but eventually there will be support for multiple types (for example, VideoPix, SunVideo, and Parallax on a Sparcstation).

If the selected coding format supports a quality adjustment, then the quality slider will be enabled and the corresponding quality "value" displayed next to the slider. The semantics of the quality setting depend on the particular coding format, but in general, higher quality settings are obtained by moving the slider to the right. For nv format, the setting controls the size of the dead-zone region of the Haar transform coefficient quantizer. For motion JPEG, the setting corresponds to the Independent JPEG group's 1-100 compression value. Finally, for H.261, the value corresponding to the GQUANT and MQUANT quantizers from the CCITT standard (this is the nominal quantizer -- if the quantizer is too small to adequately represent the dynamic range of a block, then a larger quantizer is used for that block).

Adjacent to the quality slider are two pull-down menu buttons. The "Port" button selects among the analog input jacks to the capture device (for example, a VideoPix has two composite inputs and an S-Video input). The "Type" button selects the analog video types, which is one of auto, NTSC, PAL, or SECAM. The "auto" setting attempts to determine the signal type from the actual input (provided the hardware supports this).

The "Session" panel controls conference address information, some type-in boxes, and other session controls.

The first line of the panel lists the numeric IP address UDP port of the conference, the RTP source identifier of the local instance, and the multicast TTL. There are two typein boxes labeled `Name' and `Key'. The `Name' box can be used to change the RTP session name announced to other sites. The `Key' box contains a session key for encryption described below. Below the type in boxes are toggle switches for controlling session behavior. The "Mute New Sources" button, when selected, causes sources that transmit video to come up "muted."

Encryption

Since ***VIC*** conversations are typically conducted over open IP networks, there is no way to prevent eavesdropping, particularly for multicast conferences. To add some measure of privacy, ***VIC*** allows the video packet streams to be DES encrypted. Presumably

only sites sharing the same key (and, of course, the NSA) will be able to decrypt and watch the encrypted video.

Encryption is enabled by entering an arbitrary string in the key box (this string is the previously agreed upon encryption key for the conference - note that key distribution should be done by mechanisms totally separate from ***VIC***). Encryption can be turned off by entering a null string (just a carriage return or any string starting with a blank) in the key box.

Tiling

Along the bottom of the control menu are several buttons. The button labeled "Tile" is a pull-down menu which allows you to specify the number of columns to use for displaying the thumbnail summaries of each active source.

The default is single column. The number of columns can also be specified by typing a single digit into the main window.

Session Member Listing

Clicking on the "Members" button brings up a top level window with a scrollable list of all the participants in the session. This list includes participants that are not actively sending video.

Colormap Optimization

The "Colors" button invokes a dynamic optimization of the color map used by the error diffusion or ordered dither algorithms. The distribution of colors for all "unmuted" sources is collected and handed off to a separate process to compute an improved colormap. ***VIC*** forks off histtolut(1), which must be in your execution path, to perform the computation. Because this optimization is computationally intensive, it may take a non-

negligible time to complete. During this time, the "Colors" button is disabled and grayed out.

CODING FORMATS

VIC supports a variety of video coding formats and it's a good idea to be familiar with the tradeoffs among formats before deciding which to use for a transmission. All of the formats (except Motion JPEG) utilize a block-based conditional replenishment algorithm, where the video image is divided up into 8x8 blocks and only those blocks that change are transmitted. By coding each block independently of the past, the decoding process is made robust to packet loss. Because block updates are driven by scene activity, receivers might accumulate many stale blocks from packet loss or from joining an in-progress session. This is circumvented by running a background refresh process which cycles through all the blocks continuously transmitting them at some low rate. The efficacy of this approach has been nicely demonstrated by Ron Frederick in NV.

Once the conditional replenishment step determines that a block is to be transmitted, the block must be coded. How it is coded depends upon the selected format. For the nv format, the block is transformed to a frequency domain representation via the 8x8 Haar transform. The Haar coefficients are quantized with a simple dead-zone only quantizer (i.e., coefficients that fall below some threshold are truncated to zero; otherwise, they are unchanged). The coefficients are then run-length encoded. Unlike traditional transform coders, there is no Huffman or arithmetic coding step (which typically yields a factor of two in compression gain -- but because of the dead-zone only quantizer, entropy coding would be less effective here).

For H.261, the blocks are coded as intra-mode macroblock updates using an H.261 compliant syntax. Note that ***VIC*** never uses motion-compensated macroblock types since this type of coding is very susceptible to packet loss. H.261 codecs typically do not have provisions for producing this type of bit stream, which we call "Robust-H.261," but decoders have no problem decoding it since the syntax is fully compliant. (Most H.261 codecs have an "intra" operating mode, but this is typically very inefficient because every

block of every frame is coded.) The RobustH.261 and nv encoders are both transform coders and are in fact quite similar. The differences are: (1) H.261 uses a discrete cosine transform (DCT) instead of a Haar transform; (2) H.261 uses a linear quantizer instead of dead-zone only quantizer; and (3) H.261 applies Huffman coding to the runlength encoded symbols.

For the "simple conditional replenishment" (scr) format, the block update is sent uncompressed. This approach has very high image quality but works very poorly over low bandwidth networks. Even on high bandwidth networks, slower end-systems have a hard time keeping up with the data rates associated with processing uncompressed video.

For the CellB format, blocks are encoded according to Sun Microsystems CellB syntax. CellB is a block truncation coding technique that gives fairly good compression gain, but the resulting quality is somewhat poor. The simplicity of the CellB codec results in a fast software implementation.

Finally, for Motion JPEG format, entire frames are coded via the JPEG still image standard. Motion JPEG is suitable only in high bandwidth environments and is supported only with capture devices that support hardware JPEG compression. ***VIC*** can, however, decode Motion JPEG in software.

Coding Format Tradeoffs

As in nv, ***VIC*** limits its transmission bandwidth by using a variable frame rate. When scene activity is high, the video becomes harder to code and the frame rate slows. Under this scheme, higher compression gain turns into higher frame rates. Because overall perceived quality depends very much on scene content, it's not always clear which coding scheme is best. For example, it's better to view slides at a low frame rate and high image quality, whereas most people prefer viewing a human speaker at a higher frame rate at the expense of lower image quality. The Haar transform in the nv algorithm tends to code edges, and hence text, better than the DCT in H.261. On the other hand, for typical scenes, the Robust-H.261 encoder tends to outperform the nv encoder by a factor of two

to four (Frederick has reported a similar factor of two by replacing the Haar transform by the DCT in the nv coding algorithm).

MONITOR GAMMA

Because computer monitors are not designed to display generic composite video and because analog video standards bias source signals with a display gamma correction, most computer monitors are not properly calibrated for displaying digital video signals. In other words, cameras adjust for a gamma response that is not typically present in computer monitors. For color mapped rendering (i.e., error diffusion or ordered dither), ***VIC*** allows you to specify a gamma correction factor that is tailored to your monitor. You can choose a proper gamma using the test image, gamma.gif, included in the ***VIC*** distribution. View the image from several feet away and choose the bar which appears to have a uniform gray level. The number printed below this bar is the gamma of your display. Take this number, divide it by 2.2 (the gamma correction built into an analog video signal), and use this result for ***VIC***'s gamma correction (i.e., Vic.gamma).

This gamma calibration procedure is due to Robert Berger (rwb@J.GP.CS.CMU.EDU), who provides an excellent discussion of monitor gamma in http://www.cs.cmu.edu:8001/afs/cs/user/rwb/www/gamma.html. The gamma.gif calibration image is redistributed with the permission of Robert Berger.

X RESOURCES

See ***VIC*** man page.

AUTHOR

Steven McCanne (mccanne@ee.lbl.gov), University of California, Berkeley and Lawrence Berkeley Laboratory, Berkeley, CA, and Van Jacobson (van@ee.lbl.gov), Lawrence Berkeley Laboratory, Berkeley, CA.

A.2.1 Changes to VIC

SYNOPSIS

vic [-r 0|1|2]

OPTIONS

-r 0 - original ***VIC***

1 - ***VIC*** in teacher mode. No visible changes. All new sources are muted by default, but it is possible to unmute them (the mute button is enabled).

2 - ***VIC*** in student mode. The user interface is simplified (Figure A.6). All new sources are muted by default, the mute button is disabled and only a command from ***VAT*** can enable the source.

Note: -r is used instead of -m, as in ***VAT***, because the -m option is used in ***VIC*** and in order to maintain compatibility with standard versions, the unused switch had to be utilized.

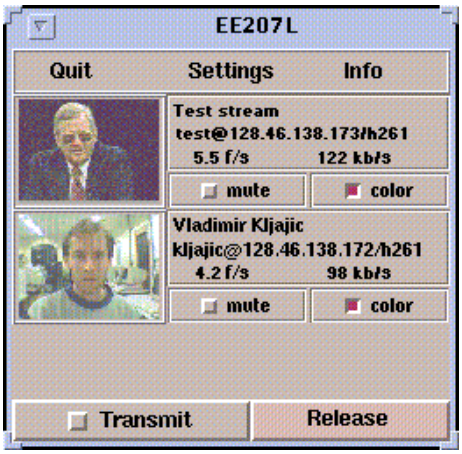

Fig. A.6. VIC Main Window (Student Mode)

OPERATION

Teacher Mode

Apart from the color palette fix, no user interface changes were done to ***VIC***. All new video sources are muted by default and are automatically unmuted when the user is enabled by using ***VAT*** (***VAT*** sends "enable" message over the conference bus). The source can also be unmuted manually.

Student Mode

The user interface is reconfigured, with a menu bar on the top, replacing the "Quit," "Menu" ("Settings" menubutton) and "Help" buttons from the standard main window. The "Transmit" and "Release" buttons are now on the main window instead in the auxiliary control window, as well as the "Look Who's Watching" menubutton, which is used instead of the "Global Stats" from the auxiliary control window.

Transmit is enabled by default, until the ***VAT*** application running in student mode is detected over the conference bus, in which case it becomes disabled and it remains so until the "enable" message from ***VAT*** is received.

The mute button is disabled until the source user is enabled by the teacher (session moderator), at which point it becomes unmuted and enabled.

APPENDIX B
MADMIN, MSTUDENT AND MSERVER MANUALS

B.1 MADMIN Manual

SYNOPSIS

MAdmin.tk [*server_name/port*]

OPTIONS

server_name/port - can be specified, this will be the default server when MADMIN starts. If not specified, the user is prompted with a menu with a list of servers to choose from (the list is read from the file *Servers.dat*).

OPERATION

MADMIN is a part of the ***MTEACH*** system which serves for the session administrators and moderators (teachers) as a way to connect to the server, retrieve or modify its data and start or join the multimedia session.

When started, after selecting the server, the user can click on the "No connection.." button, after which he/she is prompted for the username and password. The user is uniquely defined by his username and password. Depending on the information that server sends afterwards, three cases exist:

- The user is granted administrator access. In this mode, the user can change or add every possible option in the session and user database.
- The user is granted moderator access. In this mode, the user can check which sessions he/she moderates, change the session information field and check which us-

ers are allowed to participate in his/her session. He/she can also see the settings for the session (which application, what are the command line arguments) but cannot change them.

- The user is not allowed to access the server.

The main window looks as follows:

Fig. A.7. MADMIN Main Window

On the top, the menu commands are as follows:

- Session menu:
  - Start Session - starts the specified applications for the selected session, with the moderator command line arguments. The moderator command line arguments may or may not be the same as the user command line arguments.
  - Join Session - starts applications for the selected session with the user command line arguments.
  - Disconnect - disconnects from the server. Once established, the connection with the server is maintained until disconnected.

    - Quit - quits program and terminates the server connection.
- Settings:
    - Change Server - pops up a menu where one can select the server. In this menu, it is possible to add new servers (with ports to connect to) by pressing "New."
    - Change Password - prompts the user for the new password.
- Help: pretty much straightforward.

Under the main menu, the title bar displays the currently selected session. The session or user can be selected by clicking on its name in the main list window. If double clicked, the session information pops up (if the users are displayed, it shows the sessions that a user is entitled to or moderates). Right click to remove the information window.

There are two buttons underneath the title bar:

- List Sessions - this button behaves differently depending on the user privileges. If administrator privileges are given, it displays two selections when clicked:
    - Get Global Sessions - downloads the list of sessions which can be accessed by any user (and guests, which login with a "guest" username and no password).
    - Get Restricted Sessions - downloads the list of sessions available only to selected users. Again, this button behaves differently: in administrative mode, it will download all restricted sessions existing on the server, whereas in the moderator mode, it will download only the sessions that the current user is moderating.
- List Users - if in admin mode, shows two selections:
    - Get User List - downloads the list of all users (students) from the server.
    - Get Moderator List - downloads the list of all moderators (teachers) from the server.

If in moderator mode, clicking on "List Sessions" the list of moderated sessions appears and by clicking on "List Users" the list of users attending all moderated sessions (by that moderator) appear.

Underneath the main list window, there are 3 buttons:

- Add - disabled in moderator mode. In administrative mode, three selections are possible:
  - Add Session - displays the blank session information window and a new session can be created.
  - Add User - displays the user information window and a new user can be added. The user information window consists of username, password and sessions that can be attended or moderated (separated by comma). Note that if the moderator, for example, wishes to attend certain sessions other than the moderating ones, he/she would have to be added as normal users.
  - Add Moderator - displays the user information window and a new moderator can be added.
- Change - displays the session information window (Figure A.8), with the selected session information (or user information window with the selected user information).

Fig. A.8. MADMIN Session Information Window

The session Information window consists of the following:

- Name - session name field
- Session Type - can be Global, which can then be accessed by all (username "guest") or Restricted, which can be accessed only by users who have this session name in their session list.
- Available Tools - this menu is auto configurable. Every time when connection with the server is established, the server sends a list of all the media types and appropriate applications (executables) that it supports. This menu configures accordingly to show all possible media tools (applications). If the button is depressed, this means that the tool is in use for the current session. There can be multiple tools for one media type. When the session is being started or joined, MAdmin attempts to start those tools one by one until it succeeds. This is good if the administrator is not sure which tool should be used and which tool is actually used by the end user. He/she can then input several tools in the server list.
- Option arguments entry field - when clicked on a particular media tool (application), this field displays its command line arguments with which it is going to be invoked. The option arguments are used either when the moderator selects "Start Session" or when the user joins the session, depending on the "User" or the "Moderator" selection.
- Session Info - information about the session.

When "Ok" is selected, the information is sent to the server which either adds or updates the session.

In moderator mode, only the "Session Info" field can be changed.

- Delete - deletes the session or the user or the moderator, depending on the selection.

The field at the bottom of the main window displays either "No connection," or "Connected to _server_name_," depending on the status of the connection. If clicked, it

either tries to establish the connection with the server or disconnects (it is a shortcut for the "Disconnect" menu button).

Note: Once MADMIN is started, it looks for the file named *Servers.dat*. If found, it tries to load the server information from that file. The file format is:

server's_internet_address port

server's_internet_address port

…

This list of servers is going to be displayed when MADMIN is started. If *Servers.dat* is not found, the user is prompted to input the server name and the port.

B.2 MSTUDENT Manual

SYNOPSIS

MStudent.tk [*server_name/port*]

OPTIONS

server_name/port - can be specified, it is the default server when mstudent starts. If not specified, user is prompted with a menu with a list of servers to choose from (the list is read from the file *Servers.dat*).

OPERATION

MSTUDENT is a part of the ***MTEACH*** system which serves the students (or end users) so that they can connect to the server, retrieve basic session information or change their passwords, and join the multimedia session. When started, after selecting the server, the user is prompted for the username and password. The user is uniquely defined by its username and password. The user can either be granted the access in which case the ses-

sion data available to him/her is automatically retrieved and the connection is terminated or the user is not allowed to access the server.

The main window looks as follows:

Fig. A.9. MSTUDENT Main Window

Menus are as follows:

- Session:
  - Join Session - the appropriate multimedia applications are invoked, thus joining the session. Shortcut is the big button on the main window.
  - Quit - quits the application.
- Settings:
  - Change Server - pops up a menu where one can select the server to connect to. In that menu, it is possible to add new servers (with ports to connect to) by pressing "New."
  - Change Password - prompts the user for the new password.
- Help: pretty much straightforward.

Clicking on the session in the main list window selects it, double clicking invokes the information window displaying session information (right click to remove it). Note that only the global sessions (accessed with the "guest" username) or the sessions that the user is registered to are retrieved from the server and displayed.

Note: Once MSTUDENT is started, it looks for the file named *Servers.dat*. If found, it tries to load the server information from that file. The file format is:

server's_internet_address port

server's_internet_address port

…

This list of servers is going to be displayed when MSTUDENT is started. If *Servers.dat* is not found, the user is prompted to input the server name and the port.

B.3 MSERVER Manual

SYNOPSIS

MServer.tk [port #][log 0|x]

OPTIONS

[port #][log 0|x] - port number specifies the port to which the server is listening (defaults to 23232). The *log* argument opens the file *Log.dat* in which the connections to the server are logged (time, date and where from - connections established and terminated), if the argument after *log* is different than 0. *log* defaults to 0.

OPERATION

MSERVER is the part of the ***MTEACH*** system. It maintains the database of users, moderators, administrator and multimedia sessions. It also provides access to the databases to the authorized users. It does not have a graphical user interface so it can be started remotely.

Once started, it replies to requests from the MADMIN and MSTUDENT applications until it is terminated (Ctrl-C). It maintains 6 files (databases), in ASCII:

- *Adm.dat* - this is one of the two files that should be created before MSERVER is started. It should contain only one word - the administrative password. Administrator username is "administrator" by default.
- *Moderats.dat* - it contains information about all the moderators. It is of the same format as the following file.
- *Students.dat* - contains information about all the students (end users) on the server. The file format is:

  Name

  Password

  List of sessions, separated by comma (and possibly spaces)

  **

  Name

  …

  **

  It should be in ASCII.
- *Globsess.dat* - contains information about the global sessions. Has the same format as the following file.
- *Restrict.dat* - contains information about the restricted sessions. The format:

  Session name

  List of complete command lines for the multimedia applications that can be invoked for this session. If different command line arguments are used

for users and moderators, the user line should go first and then in the next line, the moderator command line is preceded by m:

Example:

vat -m 2 -r 224.22.22.22/57593

m: vat -m 1 -r 224.22.22.22/57593

rat -m 2 224.22.22.22/57593

m: rat -m 1 224.22.22.22/57593

vic -r 2 224.22.22.22/23456

m: vic -r 1 224.22.22.22/23456

One asterisk sign (*).

Session information, can be multiple lines.

Two asterisk signs (**).

New session…

- *Types.dat* - this is the other file that should be created manually and cannot be changed remotely. It contains the information about all possible types of media and supporting applications. Example:

  .Audio
  vat
  rat
  .Video
  vic
  .Whiteboard
  wb
  .3D Video
  3dvideo

  First, the media name is indicated preceded by a period, and in the next line(s) names of the applications that support it (executable names).

SECURITY

The directory where MSERVER resides should be protected by resetting all group and other permissions.

APPENDIX C
EXAMPLE OF A TYPICAL MTEACH SESSION

C.1 Setting Up the Session Parameters

There are three types of users in ***MTEACH***:

- Administrator - sets up and maintains the server and its session and the user databases. Can both start and join all sessions.
- Moderator - can see all the session's parameters but can only change the session information field for the session he/she moderates (the text describing the session). Can start his/her own session and join other sessions to which he/she has access to.
- Student - can see only the session information and can join the sessions to which he/she has access to.

Before the server is run for the first time, the administrator has to make two files which will reside in the same directory as MSERVER.

The first file is *Adm.dat* which is a file containing the administrative password. It is created by using a ASCII text editor and typing in one word, the password. For example, the file *Adm.dat* would contain one line of ASCII as follows:

secret_password

The second file is *Types.dat* (see Section B.3 for the explanation of its format). For example, *Types.dat* would contain:

.Audio
vat
rat

.Video
vic
.Whiteboard
wb
.3D Video
3dvideo

This file creates Audio, Video, Whiteboard and 3D Video media types for ***VAT*** and **RAT** tools used for audio, ***VIC*** for video and **3dvideo** for 3D video media.

Once these files are created, the MSERVER can be started (with or without the extra parameters - see Section B.3). An example of starting the server:

MServer.tk port 12067 log 1

This would start the server listening on port 12067 and logging the events (default log file name is *Log.dat*). If the *port* and *log* parameters are not specified, *port* defaults to 23232 and *log* to 0 (no logging). Note that in order to run a TCL/TK script file, you have to have TCL and TK installed on the computer running MSERVER (only TCL is sufficient for running MSERVER).The server is now ready to receive requests from the client tools.

The administrator can now execute the MADMIN tool to set up new sessions. An example of starting the MADMIN:

MAdmin.tk foo.ecn.purdue.edu/12067

This would connect MADMIN to the server running on the machine foo.ecn.purdue.edu listening on port 12067. If no parameters are given, MADMIN tries to open the file *Servers.dat* which contains the server machine names and ports. An example of *Servers.dat* would contain the following lines and should be created manually:

foo.ecn.purdue.edu 12067

boo.cc.purdue.edu 23000

fee.ecn.purdue.edu 56790

If *Servers.dat* is found, the user is presented with a list of these servers to choose from. If not, the user is prompted for the server name and the port number. To actually connect to the server, "Click here to connect" button should be invoked.

The administrator is then prompted for the username and the password. The administrator should enter "administator" as the username and the password should be the one previously entered in the file *Adm.dat*. If the password is correct, the bottom main window entry (a button) turns into "Connected to foo.ecn.purdue.edu."

A new session is now created with MADMIN by clicking on the *Add* button, selecting *Session* and then providing the necessary session information. Here is a sample list of actions which would create a test session named "Test Session":

- Click on *Add* button.
- Select *Session*. A new window will appear.
- Click on the *Name* entry box.
- Enter *Test Session*.
- Let us assume it is going to be a restricted type of session. Click on the *Restricted* radio button in the *Session Type* field.
- Assuming that the ***VAT*** application is going to be used for audio and ***VIC*** for video, click on the *vat* button located in the *Available Tools/Audio* field. The title of the entry field underneath is going to change to "Option arguments for Audio (vat tool):." We will then enter the arguments for the students (users) first, hence click on the *User* radio button to indicate that. Now click in the entry field above and enter the command line arguments for ***VAT*** (everything that would be needed to execute ***VAT***). An example:

-m 2 -C "Test Session" -r -Xmoderator=*user@host* 224.22.22.22/57593

where *user* is the username of the session moderator and *host* is his/her host Internet address (in dotted quad format). In this example, ***VAT*** is started in the student mode (the option "-m 2") with its window entitled "Test Session" (the -C option), using the RTP protocol (the -r option), transmitting/receiving on the multicast address 224.22.22.22 on port 57593 and with the session moderator specified after "Xmoderator" argument.

Now we click on the *Moderator* radio button to indicate that we would like to enter the command line arguments for the moderator. For example:

-m 1 -C "Test Session" -r 224.22.22.22/57593

The argument -*m 1* starts ***VAT*** in moderating mode.

Audio is set up properly now. For the video, click on the *vic* button located in the *Available Tools/Audio* field. Enter the option argument entry field in the same manner as for the audio. For example, for the user, the arguments can be:

-r 2 222.22.22.22/23456

and for the moderator:

-r 1 222.22.22.22/23456

The video is now set up properly. Additional arguments can be entered, such as the type of audio or video compression or maximal allowed bandwidth (see Section A.1).

- Click on the *Session Info* box and enter any relevant session information. For example:

This session is running every day from 13:00 - 14:30. Session moderator is John Doe.

Once everything is set, *Ok* button should be invoked which uploads the information to the server.

A similar procedure should be used for entering new users (moderators or students). First click on *Add*, then select *Moderator* or *User*, and enter in the data (see Section B.1). For example, click *Add/Moderator* and then enter the moderator's username (does not have to be the same as the user's email address), password and the names of the sessions he/she moderates. If the moderator is moderating "Test Session" and "EE207" session, one would enter:

Test Session, EE207

in the Sessions entry field. In the case of students, this field should be filled with the list of the sessions that he/she is allowed to attend. **Note** that the **passwords** and the **session names are case sensitive**, whereas **usernames are not**.

For reviewing the entered sessions or users, *List Sessions* (and then *Get Global sessions* or *Get Restricted sessions*) or *List Users* (and then *Get User List or Get Moderator List*) should be invoked. You can then double click on the session or user name for information (right click to remove the information window).

The session now has the proper information set up, along with the accompanying users and the moderator.

C.2 Starting or Joining the Session

The difference between starting and joining a session is that when starting a session, the multimedia applications are invoked with the moderator command line arguments by the moderator and when joining a session, they are invoked with the user command line arguments by the users.

When the moderator or the administrator wishes to start a session, he/she can start the MADMIN tool in the same way as described in the previous section, select the server where the session information is located and log in.

- Click on the *List Sessions* button. If you are logged in as the administrator, you can then select to list either the global or the restricted sessions. If logged in as the moderator, the list of the global sessions, the moderating sessions and the sessions to which you have the access to will be only displayed.
- Click on the session name and then select *Session/Start Session* from the menu. That will invoke all the necessary multimedia tools.

To join the session select *Session/Join Session* instead of *Session/Start Session*.

Students would invoke MSTUDENT first:

MStudent.tk foo.ecn.purdue.edu/12067

This would automatically connect them to the server running on the machine foo.ecn.purdue.edu. If no arguments are supplied, MSTUDENT attempts to open the file *Servers.dat* which serves the same purpose as described above. The rest of the student login procedure is similar to the administrator or the moderator login, with the difference being that after the password is accepted, the global session and restricted session lists are downloaded automatically (only restricted session to which the student is permitted to enter are downloaded).

The student can then select the session to join and click on the *Join Session* button. This will start all the necessary multimedia applications.

C.3 Asking Questions with the modified VAT and VIC

To ask a question, one should press the *Question?* button located at the bottom of the main ***VAT*** window. To enable or disable a student's transmission the moderator should select the appropriate student by clicking on the student's name.

For detailed information on how to ask questions or how to enable a student to ask a question, see Section A.1.1.

APPENDIX D

NECESSARY NETWORK BANDWIDTHS FOR DIFFERENT VIDEO COMPRESSION ALGORITHMS USING VIC*

Test Environment:

Hardware

CPU: Intel Pentium 133MHz

RAM: 64 MB

Video Card: Matrox Millenium 4MB

Video Grabber: Omnimedia Tailsman Sequence-P1S

Ethernet Card: 3COM 3C509 Combo (10Mbps Ethernet was used)

Software

OS: FreeBSD 2.2

VIC 2.7b2

Settings

Maximum bandwidth: 3Mbps

Maximum frame rate: 30fps

* Note: the tests were conducted at System Architecture Laboratory in the Computer Science department at KAIST (Korean Advanced Institute of Science and Technology) [37]

Results

Note : LARGE (640x480), MEDIUM (320x240), SMALL (160x120)

Table D.1
H.261 (quality 10)

	LARGE	MEDIUM	SMALL
Full motion	(not supported)	5~10 fps 300~400 kbps	25~30 fps 200~300 kbps
Little motion	(not supported)	25~30 fps 200~300 kbps	25~30 fps 200~300 kbps

Table D.2
CELLB (quality 10)

	LARGE	MEDIUM	SMALL
Full motion	5 ~10 fps 2~2.5 Mbps	25 ~30 fps 2~2.5 Mbps	25 ~30 fps 500~800 kbps
Little motion	10 ~15 fps 1.5~2 Mbps	25 ~30 fps 500~700kbps	25 ~30 fps 200~300kbps

Table D.3
NVDCT (quality 2)

	LARGE	MEDIUM	SMALL
Full motion	1~5 fps 500~800kbps	5~10 fps 500~800kbps	25~30 fps 500~800kbps
Little motion	5~10 fps 500~700kbps	25~30 fps 300~500kbps	25~30 fps 200~300kbps

Table D.4
NV (quality 2)

	LARGE	MEDIUM	SMALL
Full motion	1~5 fps 1~1.5Mbps	15~20 fps 1~1.5Mbps	25~30 fps 700~1000kbps
Little motion	5~10 fps 1~1.5Mbps	25~30 fps 1~1.5Mbps	25~30 fps 300~500kbps

NV and NVDCT produced the best quality, followed by H.261 and finally CellB.