A LAYERED MULTICAST PACKET VIDEO SYSTEM

A Thesis

by

THOMAS B. BROWN

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

MASTER OF SCIENCE

May 1996

Major Subject: Electrical Engineering

A LAYERED MULTICAST PACKET VIDEO SYSTEM

A Thesis

by

THOMAS B. BROWN

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

MASTER OF SCIENCE

Approved as to style and content by:

Pierce E. Cantrell
Pierce E. Cantrell
(Chair of Committee)

Jerry D. Gibson
Jerry D. Gibson
(Member)

R. Bettati
Riccardo Bettati
(Member)

A. L. Narasimha Reddy
A. L. Narasimha Reddy
(Member)

Alton D. Patton
(Head of Department)

May 1996

Major Subject: Electrical Engineering

ABSTRACT

A Layered Multicast Packet Video System. (May 1996)

Thomas B. Brown, B.S., Texas A&M University

Chair of Advisory Committee: Dr. Pierce E. Cantrell

Software based desktop videoconferencing tools are developed to demonstrate techniques necessary for video delivery in heterogeneous packet networks. Using the current network infrastructure and no network resource reservation, a one-to-many implementation is designed around a two-layer pyramidal video coder. During periods of congestion, the network routers give priority to the base layer, which by itself allows reconstruction of reasonable quality video. Receiver feedback is used to lower the output rate of the encoder's low priority pyramidal layer when all receivers are suffering high packet loss. Each of the two layers is transmitted on a separate multicast channel. Under persistant congestion, an individual receiver will discard the low priority pyramidal layer, which allows the network to prune the multicast tree back and avoid congestion. A new scheme is examined where if the other receivers are agreeable, the source will respond to a receiver pruning its pyramidal layer by lowering its rate and allowing the receiver to quickly rejoin the pyramidal layer at a quality level higher than what the high priority base layer can provide by itself. Another new scheme is described where an agent on the receiver's local router provides spare capacity information to assist the receiver in its decision to rejoin the pyramidal layer.

ACKNOWLEDGMENTS

I would like to take this opportunity to thank my advisor, Dr. Pierce Cantrell, for his direction and giving me the wonderful opportunity to pursue important research. I also wish to thank Dr. Jerry Gibson for stoking the coals of the fire in our group, always keeping the interest level high week after week. Thanks are also due to Dr. Riccardo Bettati and Dr. A. L. Narasimha Reddy for their guidance this year.

It has been my good fortune to work closely with fellow students Sharif Sazzad and Charles Schroeder on joint work in the lab. This research would not have advanced as far as it did without their collaboration.

This research is also indebted to the generous assistance of Dave Berry, Steve McCanne, Ron Frederick, and Bill Fenner.

I owe my lifelong education, confidence, and humility to the sacrifice and love of my parents, and I dedicate this work to my grandfather Elro who was fond of electronics and reading. He passed those two passions down to me.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

FIGURE Page

CHAPTER I

INTRODUCTION

The novelty of Internet videoconferencing has lured a large number of users into making exceptional demands on network resources. The CU-SeeMe videoconferencing system [1] has perhaps empowered the most people with this technology as it is the cheapest solution that runs on the most widely available computer systems.

CU-SeeMe provides a 16 level greyscale picture with a standard window size of 160x120 pixels. Two steps, conditional 8x8 block replenishment and lossless compression, are used in the CU-SeeMe video coding algorithm. Because the original target platform was of the early Macintosh variety, a transform coding technique was not used due to computational complexity. The lossless compression method can achieve an average compression ratio of 1.7 [2].

When sufficient network resources exist between users, CU-SeeMe is a remarkably usable videoconferencing system. Unfortunately, this is not always the case, and the system does not efficiently use limited network resources. The first problem is that for multiparty conferences, a separate stream of data is created from each sender to all receivers. For the worst case, video delivery to n receivers will require n times as much bandwidth as is necessary in a one-to-many conference. A recent addition to the Internet, and a standard in the next-generation Internet Protocol (IPv6) [3], is multicast routing which allows a data stream to be delivered to multiple recipients in an internetwork by generating only one stream at the source and letting the network replicate the stream only at the routers that use divergent paths to reach multicast group members.

The journal model is *IEEE Transactions on Automatic Control*.

Considering a one-to-many video distribution scenario, another problem is that nearly all video coders assume that every receiver has enough network bandwidth to receive the video. While users on uncongested high bandwidth links can view a high rate source, low bandwidth links viewing the same video will receive unacceptable video quality. This is common to nearly all video coders on the Internet. However, a video signal can be divided into a hierarchy of n layers, where the highest priority is given to the most important information and lower priority to successively less important information. Layer i requires the complete reception of layers 1 to $i-1$ for signal reconstruction. Receivers on low bandwidth links can reconstruct a gracefully degraded video signal from this contiguous subset of higher priority layers. The current Internet infrastructure does not support such a scheme as routers do not currently prioritize packets, but as the design goals of IPv6 include support for multimedia traffic, a four bit precedence field is supplied in the IP header [3].

High bandwidth links are usually not found on the edges of a network as more traffic is expected to flow in the core. It is often the case then that the congested router is the router immediately upstream from the receiver (or immediately downstream of the sender). In such a case, if time delays are not excessive, providing priority forwarding in this router will allow a layered video coder to be successfully implemented. A router with priority forwarding will be included in the research testbed.

An appropriate priority scheme must be chosen. Time priority schemes favor higher priority packets by serving them first while space priority schemes discard lower priority packets when the queue fills to a certain level [4]. Time priority may be a better choice for delay-sensitive real-time traffic [4]. If long queuing delays due to high traffic intensity are rare, it may be appropriate to use a space priority scheme as the major goal is to ensure the delivery of the high-priority video stream(s).

Packet loss can result either from buffer overflows in network routers or at a system's network interface, or late packet arrivals at the destination [4]. This research makes no attempt to reduce the packet loss caused by delay. Ignoring the influence of late packet arrivals on packet loss is obviously not appropriate for all cases.

The power of the layered scheme is that it reduces the effects of packet loss during congestion by providing graceful degradation of the video quality. The result is that receivers on congested or low bandwidth links may participate in high bandwidth video streams. The Internet will remain heterogeneous as different types of links are required for different applications. For instance, mobile computing networks tend to be bandwidth limited.

The treatment of multicast routing of hierarchical data has been discussed in the context of bandwidth reservation in [5, 6]. These papers specify an algorithm that delivers to the source a list of destinations and bandwidth available to them under the restriction that each destination receives all of the traffic on a single path. The algorithm then assigns bandwidth to layers under the constraint of a maximum number of layers. As sound as this approach may be, the algorithm requires considerable network overhead as it expects the source to be privy to not only who is listening, but also where those listeners are in the distribution tree. Incremental deployment of such a solution given the Internet's current infrastructure would be challenging. Such an algorithm is not practical since an incrementally deployable solution is desired on an internetwork as large as the Internet. Another interesting but simpler proposal suggests reserving bandwidth for the base layer only, while higher fidelity packets are left to compete with each other [7].

If a source sends each layered video stream on a separate multicast group [8], [9], a receiver can respond to persistent congestion by dropping a layer. This method is referred to as receiver-based congestion avoidance because a receiver causes the multicast distribution tree to be pruned back beyond the point of congestion.

This research extends previous ideas of congestion avoidance with two new approaches. The first is a hybrid scheme, composed of both source-driven and receiver-driven congestion avoidance. If all participants are agreeable, a source reduces its rate in response to a receiver dropping an enhancement layer. The receiver then quickly attempts to rejoin the enhancement layer. The second scheme allows a receiver, by querying an agent on its local router, to make a more informed decision when it is deciding whether to rejoin a layer. Furthermore, a description of end-to-end delivery of layered video using IP multicast is presented.

The research will include the development of experimental one-to-many videoconferencing tools based on the CU-SeeMe coder and the pyramidal coder developed by Sazzad [2].

Even though this research is carried out with 4 bit video, the results may be applied to color video systems as well since the luminance portion of the signal is considered to be representative of the whole video signal [4].

CHAPTER II

LATEST DEVELOPMENTS IN NETWORK INFRASTRUCTURE FOR MULTIMEDIA

The Internet is incrementally evolving into an internetwork that can support multimedia. This has been a rich area for research over the past several years, and the most interesting work is probably still yet to come. This chapter summarizes the important work upon which this research is based.

A. Multicast Pruning

Multicast routing is important for conferences with a large number of participants. A source wishing to multicast data on the Internet will address its packet not to a single machine (unicast), but to a group of machines. No more than one copy of the multicast packet will appear on any link; whereas, unicasting to multiple receivers requires multiple copies to be sent. Pruning is a multicast feature that allows the network to prune multicast trees so that the data is delivered only to the branches necessary to reach all the receivers. Fig. 1 shows a multicast tree for a source W.

To become a participant in a conference, an application (process) joins a host group which is designated by one of a set of reserved class D Internet addresses for multicasting. The process must also specify which interface is to receive the multicast packets. If no other process on the system belongs to the host group already, an unsolicited Internet Group Management Protocol (IGMP) report is sent out the interface [10]. This allows each multicast router to determine which host groups have been joined on each of its links. Knowing which host groups have members on the router's adjoining links allows it to participate in pruning.

If a router is the "parent router" of a link for a given source, it will be responsible

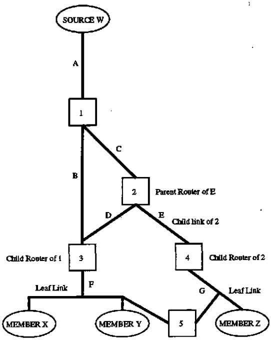

Fig. 1. Relationship between Parents, Children, and Leaves.

for forwarding host group packets for that source. The parent router on the link is the router on the link who has the shortest distance to the source. The shortest distance is determined in the case of Distance Vector Multicast Routing Protocol (DVMRP) [11] routing by the periodic exchange of (source, distance) information by a link's routers. Because of this relationship, it is evident that DVMRP performs minimum hop routing when all the link "distances" are the same, which is the assumption for Fig. 1. In Fig. 1, router 2 is the parent for link E. Link D does not have a parent router as no data flows through it. Instead, data flows through link B to reach link F. Router 3 is the parent of link F, but if member Z was a source, for example, router 5 would be the parent router for Z's stream. Similarly, a "child link" belongs to its parent router. Link E is a child of router 2, but link D is not a child in W's source tree.

When a source begins sending data to a multicast group address, the first packet

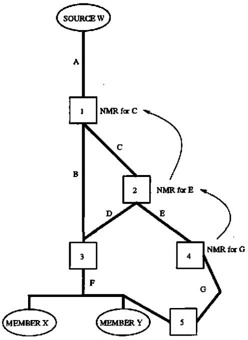

Fig. 2. Pruning a Tree on the First Packet.

is generally broadcast across the entire shortest path tree with the exception of leaf links that do not have members of the group. Leaves are simply child links that no other router uses to reach the source. Links F and G are leaf links with respect to source W. Although no other router uses link D to reach the source, D is not a leaf because it is not a child. Data will initially flow through all links except for D since D is not part of the shortest path tree. However, if there were no member Z at the beginning, then no data would initially be transmitted on link G. This initial delivery method is called Truncated Reverse Path Broadcasting (TRPB). No more than one copy of this first packet appears on any link as each link will only receive the packet from its parent router. A router may also truncate a branch if the incoming packet's time-to-live (TTL) value, which is decremented at each router, is less than the threshold configured at the router [12]. This is simply an administrative way to contain a conference within a certain number of hops.

Fig. 2 shows the same tree without a group member Z. The first packet will not appear on link G since it is a leaf link. When the first multicast packet reaches a router for which all child links are leaves and have no members of the group, a Non-Membership-Report (NMR) for the (source, group) pair is generated. Thus, an NMR is generated at router 4. The report is copied and passed to the next router upstream. If the next upstream router receives NMRs from all of its child routers and there are no group members on any of its child links, it copies the NMR and passes it upstream to the next router. Since router 4 is router 2's only child router, router 2 will pass an NMR up to router 1 where the prune stops. Thus, the pruning process propagates nonmembership information up the tree far enough so that subsequent packets only go as far as they need to go to reach group members [13].

B. Real-Time Transport Protocol

The Real-Time transport protocol (RTP) [14] was developed to provide a lightweight transport protocol for time-dependent data and to provide interoperability among conferencing tools in order to promote experimentation with audio and video delivery to a large number of participants [15]. RTP is is an application-level protocol composed of a data protocol and a control protocol (RTCP). The application has the responsibility of encapsulating the data payload with the RTP header and providing the RTCP control functionality. As it is network independent, it may fit above UDP/IP, or ATM's adaptation layer, or real time protocols like the Real-Time Message Transport/Internetwork Protocol (RMTP/RTIP) [16]. It does require the layer below it to provide multiplexing of the data and control channels. For example, UDP multiplexes data and control using two ports.

RTP leaves the responsibility of support for Quality of Service (QoS) guarantees,

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|V=2|P|X| CC=0  |M|   PT = 27   |        sequence number        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         RTP timestamp                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           synchronization source (SSRC) identifier            |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
|            contributing source (CSRC) identifiers             |
|                             ....                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Fig. 3. RTP Data Header

if any, to the network layers underneath it. It does not provide admission control or resource reservation services. Instead, it performs functions common to nearly all media delivery applications. Fig. 3 shows the RTP data header.

A sequence number field is provided to allow a receiver to keep the data in order. For video, the timestamp field is used by a receiver to associate packets with a given video frame. All packets for a given video frame will have the same timestamp. The payload type (PT) field allows a receiver to identify which decoder it needs to use to reconstruct the media, and a one bit marker (M) field's use depends on the media. For audio, it signals the beginning of a talk spurt, but for video it identifies the last packet for a video frame which causes the receiver to render the image. If the last packet of a video frame is lost, the receiver will render the frame when a packet with a new timestamp arrives.

Each media stream is assigned a session source identifier (SSRC). For example, an audio stream from a source will get a unique SSRC and a video stream from the same source will get its own unique SSRC value also. SSRCs are unique across a conference. An audio and video stream from the same source are bound by a canonical name (CNAME) identifier in an RTCP SDES packet which allows a receiver to synchronize

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|V=2|P|    SC   |  PT=SDES=202  |             length            |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
|                          SSRC/CSRC_1                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           SDES items                          |
|                              ...                              |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
|                          SSRC/CSRC_2                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           SDES items                          |
|                              ...                              |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

Fig. 4. RTCP Source Description Identifier (SDES)

the two streams. The CNAME is globally unique across all conferences and is usually the email address of the sender. The SDES packet format is shown in Fig. 4.

The control protocol's primary function is to provide reception quality feedback information. Packet loss, delay, and jitter statistics are fed back to the source, which uses the information to control its output rate. A participant who has recently sent data will periodically send a sender report (SR) followed by a list of receiver reports (RR), one for each source in the conference. Stacking multiple RTCP reports into one packet is what is referred to as a compound packet.

Also sent in each compound packet is an SDES report with the CNAME identifier since new participants in the conference need to receive this as soon as possible to allow synchronization of the media streams. Fig. 5 shows a sender report. The sender report gives an indication of the amount of bandwidth transmitted by providing an octet count and a packet count.

Receivers who are not sending a data stream send only RTCP RRs compounded with the SDES packet. Reception statistics from senders and receivers should be sent

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|V=2|P|    RC   |   PT=SR=200   |             length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         SSRC of sender                        |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
|              NTP timestamp, most significant word             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|             NTP timestamp, least significant word             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         RTP timestamp                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     sender's packet count                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      sender's octet count                     |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
|                 SSRC_1 (SSRC of first source)                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| fraction lost |       cumulative number of packets lost       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           extended highest sequence number received           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      interarrival jitter                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         last SR (LSR)                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   delay since last SR (DLSR)                  |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
|                 SSRC_2 (SSRC of second source)                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                               ...                             :
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
|                  profile-specific extensions                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Fig. 5. RTCP Sender Report

as often as the bandwidth constraints allow. The specification suggests that the total bandwidth may take no more than twenty percent of the total conference bandwidth. RTCP transmission intervals are calculated each time a new participant is sensed. The minimum interval allowed is five seconds.

One last important characteristic to note is that in a multicast conference, the data and control are sent using the same multicast address and sent to everyone. This allows a third party monitor to detect faults or bottlenecks in the distribution, but since the data is sent along with the control, the data stream will appear on the monitor's link even if it does not want it.

C. Priority Forwarding

Descriptions have been offered on how building a source tree is achieved and how end-to-end transport can be provided. What remains to be determined is how the network layer is going to provide a quality of service. One means of improving the quality of service is to prioritize important packets. In this research, important base layer multicast packets will be given priority over pyramidal difference layer packets to improve the quality of service during congestion. Whereas building routing tables has traditionally been achieved by a process running above the network kernel on a Unix machine, multicast forwarding occurs in the IP kernel.

Fig. 6 outlines the process used to forward and prioritize a multicast packet in the IP kernel. A network interface such as Ethernet, ISDN, or FDDI generates a hardware interrupt which calls its device driver which ultimately places the packet on the IP input queue and schedules a software interrupt. Since hardware interrupts have a higher priority than software interrupts, several packets may be left on the IP input queue before the software interrupt takes them off [10]. When the software

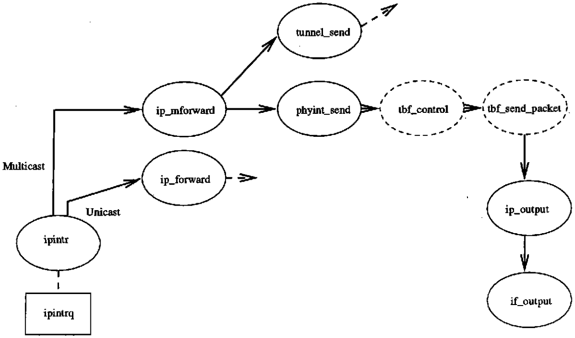

Fig. 6. Multicast Forwarding.

interrupt occurs, the kernel calls *ipintr* which dequeues the packet, verifies it, and processes any options [10]. If the packet has reached its final destination, it is passed up to the transport layer, otherwise, *ipintr* attempts to forward the packet. If it is a multicast packet and if this is a multicast router, *ipintr* tries to forward the packet by calling *ip_mforward* which determines the output virtual interface, and if there are no tunnels, it calls *phyint_send* which actually makes a copy of the outgoing datagram and then decrements the TTL value [10]. Until recently, *phyint_send* would pass the datagram directly to *ip_output*, but now it passes through *tbf_control* so that tokens can be counted to see if the multicast rate limit has been exceeded. If the rate limit has not been exceeded, the packet is finally passed to *ip_output*. If the multicast rate limit is exceeded, *tbf_control* will call *tbf_dq_sel* to discard a member of the queue based on a precedence value [17].

The use of a multicast rate limit will provide a convenient laboratory simulation of congested conditions so that congestion avoidance algorithms may be evaluated. The priority mechanism is used when the rate limit is exceeded. Another reason the rate limit is used is because it is hard for the forwarding mechanism to know when congestion is dropping packets, but it is easy for it to know when a rate limit is making it drop packets [18]. This is because the kernel doesn't realize it is congested until it has reached the network interface output processing routine *if_output*.

D. Resource Reservation

Resource reservation is a method used to earmark network resources for a conference. The Resource Reservation Protocol (RSVP) is a popular experimental implementation. This research assumes that there is no such facility. The necessity of a resource reservation protocol can not be assumed. If resources are to be reserved, then there will be those who are refused service. For the service to be acceptable to customers, the network must say "no" rarely. If the network is able to say "no" rarely, then reservations may not be necessary, and degraded service might be acceptable [19]. If degraded, non-reservation service is acceptable, layered media will enjoy an additional importance on the Internet.

CHAPTER III

NETWORK SUPPORT FOR A TWO-LAYER PYRAMIDAL CODER

In this chapter, network support required for two-layered video is described. As a background for some of the congestion avoidance design decisions, a brief description of the the experimental pyramidal coder is necessary and will be presented first. Specical considerations needed for the delivery of layered video follow. The design of both a source-based congestion avoidance scheme with receiver feedback and a receiver-based congestion avoidance scheme are then described. A method that allows a receiver to quickly rejoin the pyramidal layer when it notices that the sender has lowered its quality is then described. Finally, a method that increases the probability of a successful rejoin is described.

A. Description of Encoder

The pyramidal coder uses both 320x240 and 160x120 resolution video frames as input to produce the pyramidal difference stream. The 8x8 conditional replenishment blocks produced by the first stage of the 160x120 resolution CU-SeeMe stream are upscaled to 16x16 blocks and subtracted from the corresponding 16x16 blocks from the 320x240 image. The original pixel values are represented with 4 bits, but as the pyramidal difference values range from -15 to +15; they are represented with 5 bits [2].

A run length coding can be used to provide a lossless compression for the pyramidal difference layer, but the pyramidal difference values around zero can also be mapped to zero to achieve a reduction in the bit-rate at the expense of a reasonable reduction in quality. A quality parameter, q, is defined to control how many values are mapped to zero. For example, if $q = 2$, then the difference values -2, -1, +1, and +2 are all mapped to zero. Since most of the difference values generated by the coder

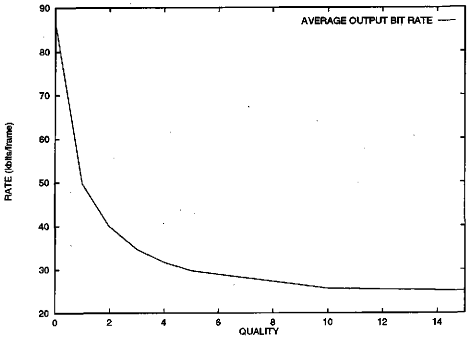

Fig. 7. Rate versus Quality Parameter for 320x240 Image [2].

are concentrated around zero, this example mapping actually reduces the bit-rate of the pyramidal difference stream by about two thirds [2]. This quality parameter is useful because it can be used to provide dynamic bit-rate adjustment.

Fig. 7 shows the typical dependence of the complete 320x240 video bit-rate on the quality parameter at 2 frames/s. The contribution of the base layer stream to the rate is constant for all values. Tests suggest that $q = 3$ is a good choice as it provides good quality at a relatively low bit-rate [2].

B. End-to-End Data Delivery

Several special considerations exist for delivering layered video on a multicast IP network. The application-level processing done at the receiver is more complicated than the packetization done at the source primarily due to the special cases that arise when packets are lost or processed out of order at the receiver application. Fig. 8 shows the packetization process that occurs at the source.

The frame capture, compression, and packetization process is initiated as a result of a software alarm. The frame capture function immediately sets another alarm for the next frame. The alarm interval is the reciprocal of the frames/s setting chosen by the user. After the frame is grabbed, a 320x240 image and a 160x120 image are constructed by decimation. The four most significant bits of the luminance are used to produce the 16 level greyscale video. If the user has selected to send only the 160x120 resolution, then only the 160x120 image is produced and compressed. This option saves valuable processing time and allows a higher frame rate. The RTP timestamp, which will be used for all the packets for the current frame, is generated and the compression module is called.

In the compression module, each 8x8 block in the video frame is compared to the one in the previous frame, and if its measure of change as calculated by the CU-SeeMe compression method is greater than a threshold, the block will be selected for transmission. If the decision is made to transmit a block, the 8x8 block is losslessly compressed, the corresponding 16x16 difference block is constructed and compressed, and each is added to their respective send buffer. If the addition of a block increases its buffer's length to more than 1000 bytes, the packetization module is called, the contents of the buffer are transmitted, and the buffer is cleared. Compression and packetization are tightly coupled in this way because it is desirable to send packets that end on block boundaries. This allows the receivers to salvage blocks from packets that follow a lost packet for a given frame. Otherwise, if a packet starts in the middle of a block, there is no simple and efficient method to determine where the first complete block begins.

Transmitting a buffer after it grows beyond a threshold of 1000 bytes after the

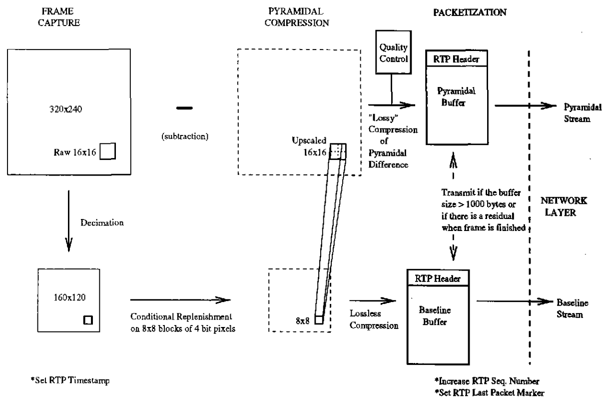

Fig. 8. Frame Capture, Compression, and Packetization for Pyramidal Coder.

addition of a block keeps the maximum packet size within the Ethernet maximum transmittable unit (MTU) size of 1500 bytes. Preventing a packet from becoming fragmented makes it less susceptible to information loss as IP will discard the entire packet if any of the fragments are lost [20]. Keeping separate threshold values for each stream might be appropriate since 16x16 blocks are larger than 8x8 blocks.

At the end of the data of a video frame for each layer, the encoder attaches a 255 to let the decoder know where the data for a video frame ends. A 255 is an illegal value and is not a possible value that can be generated by the coder. Not only will the last packet of a video frame for each stream end with a 255, but it will also have its "last packet" marker bit set in the RTP field. The "last packet" marker serves a different function. While the 255 byte tells the decoder where the data ends, the RTP marker bit is designed to let the application know when to display the video frame. Alternatively, the decoder could have simply used the return values from the *recvfrom* calls to determine the buffer size of each layer.

Each stream is transmitted using a different multicast address so that the multicast tree created may have different prunes for the base layer stream and the pyramidal stream as some receivers will not have the network resources to receive the pyramidal stream. Each stream is considered its own RTP session and consequently maintains its own RTP session source identifier (SSRC) so that each will have its own packet sequence number space. Packet loss calculations at the receiver are simpler in this arrangement than they would be if both streams belonged to the same RTP session, sharing the same sequence number space.

It is important to keep in mind that the base layer stream is sent with a higher priority than the pyramidal stream, but like assigning multicast addresses and session source identifiers, this occurs during initialization of the network channel. Also, it is convenient to send the base layer and pyramidal streams on different port numbers as well. As the port number is used in the advertisement for MBONE sessions, this scheme allows the application to immediately know which layer is which.

However, if the system design was changed to allow multiple sources, this arrangement of using separate RTP sessions and separate UDP ports would prevent the receiver from decoding video from a source until it received an RTCP CNAME binding for that source [21]. Since reports are sent no more frequently that once every five seconds, this would be inconvenient. McCanne suggests using only one source identifier for all the layers in a video stream so the receiver may immediately decode and view the video without having to wait for the CNAME binding [21]. The aim of this research does not require multiple sources, so a separate source identifier for each layer will be kept.

As mentioned previously, the packet processing for layered video at the receiver is relatively complicated. However, the fact that the source-rooted tree algorithms in use on the Internet deliver all multicast packets from the same source along the same tree [22] will simplify the reconstruction of video frames at the receiver. First, it will not be necessary to reorder the multicast packets. Since they all follow the same path, they will arrive in order. Second, no buffering will be needed to measure packet loss. If packet number $n + 1$ arrives before packet number n, there is no hope that packet number n may still arrive.

Since the receiving application receives the two layers on two different multicast addresses, it will read the data from the two layers on two different socket descriptors. Although IP can be relied upon to deliver multicast packets in order, multicast packets can not be expected to arrive at the receiving application in the order they were sent when two different sockets are used. Specifically, the experimental source always sends the last base layer packet for a video frame before the last pyramidal packet. This order will be maintained throughout the path in the internetwork from source to receiver and also in the receiver IP input queue, but when the IP kernel module decides that the multicast packet should be delivered to the transport layer (always UDP for multicast), it calls the UDP input processing function which places the UDP datagram on the appropriate socket buffer [10]. The receiver application knows that it has packets waiting on each socket, but it does not know that the packet in the base layer socket buffer arrived before the packet in the pyramidal socket buffer. This is very much like the shared human experience that occurs at the grocery store when choosing the shortest of two lines often results in a longer wait before service.

From the previous discussion, it will be assumed that packet payload data may immediately be added to its respective receive buffer without reordering, but care must be taken to handle the cases where the last packets of each stream are processed out of order or do not arrive at all. The algorithm to process an incoming packet, whether it is a base layer or a pyramidal packet, will now be examined. For simplicity, the following description assumes that the 320x240 resolution image size is being viewed and thus the receiver is accepting both streams. However, the experimental tool supports viewing 160x120 resolution image size as well as 80x60 resolution. Each one of the two streams uses a separate multicast address.

If a packet is waiting in one of the socket buffers, a software interrupt will call a general packet processing routine as depicted in Fig. 9. The packet is read into a buffer in application memory space. The timestamp in the RTP header is checked to see if it belongs to a previous video frame that has already been rendered. This may happen if the first pyramidal packet of the next frame is processed before the last base layer packet of the current frame. There is a potential for ghosting in this situation. Ghosting occurs when new pyramidal information is updated on old base layer blocks when the new base layer blocks have been lost. Fig. 10 shows this special condition.

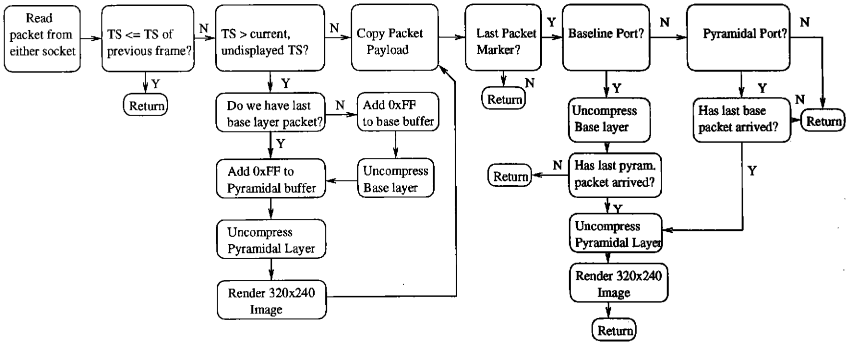

Fig. 9. Flow Diagram for Packet Processing at Receiver.

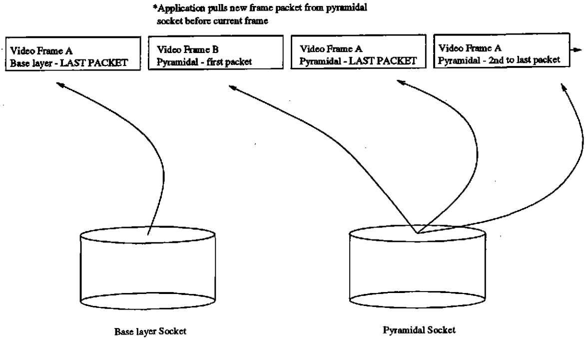

Fig. 10. Packet Arrives After Its Frame Has Been Rendered.

As noted in the second chapter, if a packet for a new video frame appears (one with a new timestamp) before the old video frame has been rendered, the application assumes it has lost the packet with the RTP "last packet" marker bit set, and it immediately renders the old video frame before it processes the new packet.

If it is assumed that the internetwork priority mechanism always discards the pyramidal packets for a given video frame before the base layer packets, the case will never occur when the last base layer packet for the current frame is lost and the first base layer packet of the next frame is processed before the last pyramidal packet of the current frame.

If the current packet does indeed belong to the previous rendered frame, it is discarded and the routine is finished processing the packet.

Next, if the current packet's RTP timestamp does not represent a previously rendered frame, the possibility that it represents a new frame while the current frame has not yet been rendered is checked. This occurs when the last pyramidal packet is lost. Before the packet may be processed any further, a 255 byte is added to the end of of the compressed pyramidal data buffer (so that the decompression routine will know where the data ends since the original 255 byte has been lost), and the current video frame is immediately decompressed and rendered. It is also possible that all the pyramidal packets have been lost and the last base layer packet, too. In this case, a 255 byte must be added to the compressed base layer data buffer as well, and the buffer must be decompressed here as a last base layer packet did not previously trigger the base layer decompression routine. After the decompression and rendering of the previous video frame, the packet is allowed to be processed.

The packet payload is now permitted to be added to the appropriate compressed data buffer. If the RTP "last packet" marker bit is not set, the routine is finished. If it is set, the port is examined to determine what layer the packet belongs to.

If it is a base layer packet, the data in the baseline compressed buffer is uncompressed. If the corresponding last pyramidal packet has not arrived, the routine is finished. Otherwise, if the last pyramidal packet has already arrived, then the pyramidal difference data is uncompressed and the 320x240 image is rendered.

Otherwise, if upon examining the port, it is a pyramidal packet and the last base layer packet has already arrived, the pyramidal data is uncompressed and the image is rendered. If the last medium packet has not already arrived, the routine does not uncompress and render the image.

Figs. 11 and 12 demonstrate the improvement in video quality in the presence of congestion. The first frame shows the serious effects of packet loss for a 320x240 CU-SeeMe multicast stream using the nv videoconferencing tool [23]. The second frame shows a 320x240 pyramidal stream with priority given to the base layer stream. These two frames were subjected to a roughly fifty percent packet loss by setting a multicast

Fig. 11. Packet Loss Effects for nv.

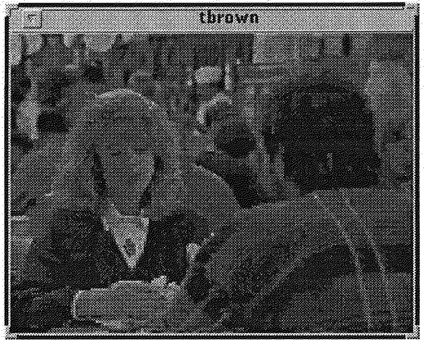

Fig. 12. Packet Loss Effects for CafeMocha.

Fig. 13. Packet Loss Effects for CafeMocha with No Priority.

rate limit on a router. Fig. 13 shows the effects of packet loss on a pyramidal 320x240 stream with no priority given to the base layer stream. The ghosting effects are due to new pyramidal information being updated on old base layer blocks when the new base layer blocks have been lost. The dependence on the priority mechanism is evident. Note that although the two layers were delivered on two different multicast channels, this was not necessary to provide the improved quality. Both streams could very well have been sent on one multicast channel as long as the base layer stream was given priority.

End-to-end delivery of two-layered video data has now been addressed, and consequently receivers enjoy the benefit of receiving gracefully reduced video quality in the presence of packet loss as long as there is enough bandwidth to at least receive the base layer. However, the network is a shared resource, and the system must not be allowed to annex as much of this resource for itself as it wants. The remaining sections in this chapter attempt to find a balance between providing good quality video to the session members and avoiding congestion at the links in the multicast tree.

C. Source-based Congestion Avoidance

The design of a congestion avoidance scheme for video delivery without the use of resource reservation has been attempted in [24, 25, 26, 27]. Neither [26] nor [27] address how their schemes would be used in a multicast environment. None of these schemes use layered video and consequently suffer serious quality degradation in the presence of packet loss. All of these schemes use a feedback mechanism so that the sender can reduce its output bit-rate in response to packet loss measured at the receivers. In [24], it is noted that this type of scheme has to contend with the possible problem of feedback delay being too high to report the current state of the network.

After congestion is alleviated by reducing the source rate, the source will receive favorable packet loss reports from the receiver(s) and raise its output rate, which tends to cause congestion again. A filter is used in [25] to avoid these oscillations in the source output rate. The idea is to put less weight on new packet loss reports. A filtered loss rate λ represented by the following equation

$$\lambda = (1-\alpha)\lambda + \alpha * b \qquad\qquad \alpha \in (0,1) \tag{3.1}$$

is used to produce a more stable system. The filtered value is composed of the newly received value b weighted by α and the previous filtered value weighted by the compliment of α. Reducing α increases the influence on previous values and makes the system less responsive to changes. Short term congestion is less likely to cause the source to reduce its rate, and the source will not raise its rate unless uncongested conditions have persisted for a while. A congestion threshold is defined to be the packet loss percentage that results in unacceptable video quality. When λ exceeds the congestion threshold allowed, there is a multiplicative decrease to aggressively reduce the bandwidth. When it falls into the unloaded region, there is an additive increase. There is a region between the congested and unloaded region in which the source takes no action.

The effects of adjusting the output rate with the quality parameter for the coder used in this research can be seen in Fig. 7. The coder can aggresively reduce its bandwidth by increasing the number of values mapped to zero. Remember that changing q from 0 to 2 reduces the bandwidth by approximately two-thirds. If congestion still persists at $q = 2$, it would make sense to set $q = 15$ which would cause the coder to refrain from sending the pyramidal layer. Bandwidth can slowly be added back when the network is uncongested by reducing q until q reaches 2 or 3. Adding bandwidth by reducing q at this point causes a drastic increase in the output bit-rate and is likely to return the network into a congested state.

In general, [24] and [25] make the bandwidth adjustment decision by providing the "greatest good for the greatest many" in a conference with many receivers. For example, if a majority of the receivers are suffering high packet loss, the source will reduce its bandwidth. Using a two-layered coder allows the fair use of a different policy. The pyramidal bit-rate will only be reduced if all receivers are either reporting high packet loss or have dropped the pyramidal layer. This is fair because all of the receivers now have the privelege of receiving a minimum quality with the base layer stream. The pyramidal stream can now be used to provide high quality video to those who have the resources. They no longer have to suffer a reduction in quality if most of receivers in the conference are connected to low bandwidth links.

However, the use of two-layered video does not make source-based congestion avoidance obselete. It can still be used if all the receivers are suffering packet loss. Moreover, the source should stop transmitting the pyramidal layer if no receivers are requesting it. This optimization prevents unnecessary bandwidth from being wasted on the source's subnet. This can be achieved by setting $q = 15$.

D. Receiver-based Congestion Avoidance

As mentioned in the previous section, there are some serious drawbacks to the source-based congestion avoidance method. The implementations that use the source-based congestion avoidance scheme are actually expecting it to perform two functions. Not only is the scheme used for congestion avoidance, but at the same time it is used for improving the quality of service by reducing packet loss. When this scheme is used in a conference with many participants, it can provide neither of these functions well. It has already been shown that using a layered video approach improves the quality of service by gracefully reducing the quality in the presence of packet loss. A receiver-based scheme will now be examined for its suitability in providing congestion avoidance in a multicast environment.

A receiver-based scheme requires a layered coder and allows each receiver of the multicast session to take action for itself when it determines the path between it and the source is congested. Like the source-based scheme, a receiver measures packet loss for an indication of congestion, but instead of having to communicate this to the source, the receiver is now empowered to quickly make its own decision. By dropping the highest layer in the hierarchy, the receiver cuts the rate of data flow on its subnet and those pruned. Fig. 14 shows receiver-based congestion avoidance with a two-layered coder. Member Z has left the pyramidal group and the multicast tree has been pruned back from Member Z to Router 1. All traffic between Router 1 and

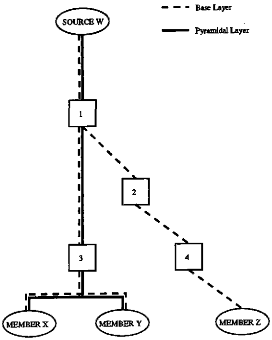

Fig. 14. Pruning the Pyramidal Layer.

Member Z does not have to compete with the bandwidth generated by the source W's pyramidal stream.

Although priority forwarding in the routers will handle congestion at the bottleneck, the receiver-based approach allows traffic to be discarded further upstream from the congested link in the source tree [28]. Although the upstream links from the congested router are not congested, better use may be made of the links by Transmission Control Protocol (TCP) streams that have backed off for the enhancement layer [22]. Also, if a layer is only contributing a small number of high quality blocks to the base image, it is not helping the picture, and it may be discarded.

Dropping a layer when all routers toward the source have priority forwarding does not improve the quality of the video at the receiver. However, if there is a router in the path that does not prioritize packets and it is the bottleneck, then dropping a layer would most likely improve the quality by allowing more base layer packets to be forwarded.

As noted in [29], a receiver should aggressively drop a layer. However, the receiving application should give the priority forwarding mechanism a chance to absorb short term congestion and should drop the pyramidal layer only during longer term congestion. This strategy prevents transient surges in the network from unnecessarily compromising the receiver's video quality. Also, repetitively joining a layer might be burdensome to the network. With DVMRP, routers remember which NMRs they have sent. A join immmediately includes a group (layer) in the conference by sending out cancellation messages to undo the effect of the NMRs [13]. Each cancellation message must be positively acknowledged. So, an appropriate interval to sample lost packets needs to be determined which attempts to balance the desire to be agressive and the desire to provide the best quality to the receiver.

As a layer should be dropped as soon as the receiver decides the congestion is not short term, it should be conservative in its attempt to add it. After an appropriately selected rejoin interval, the receiver may try to join the layer again. Packet loss can be sampled again to determine if the link is still congested. However, it may well be the case that the extra traffic added by the pyramidal layer might constantly be sufficient to put the bottleneck link in a congested state. This is especially true if the bottleneck link is the last link in the multicast tree. In this case, no choice of rejoin interval will be sufficient. The problem is that measuring packet loss is an unsatisfactory method to infer the spare capacity needed to allow a successful join.

If two receivers are on the same subnet, their criteria for joining and leaving a layer should be designed to allow them to agree on when to leave and when to join. If one of the receivers has decided that the network is congested and drops the highest layer in the hierarchy, the multicast group will not be pruned unless all other receivers on the subnet make the same decision. If each measures congestion by calculating the percentage of RTP sequence numbers that did not arrive, they will have the same basis to make a decision. They can synchronize their actions by joining and leaving on the reception of an RTCP sender report since the sender report will appear on the network interface of each at the same time. However, the granularity does not need to be this fine, and it would be natural for them to take action when they are generating an RTCP receiver report as this is where the receiver makes its packet loss calculations to share with the other conference members. RTCP receiver reports are designed to be unsynchronized to prevent periodic surges in the network when a large number of participants are involved, but as they are required to be sent periodically, using this event to make a decision will provide a certain degree of granularity for a consensus prune.

If a receiver upstream of another receiver decides to drop the pyramidal layer, the downstream receiver will make the same decision unless the upstream receiver is bound by processing speed. It is typical in this case for the UDP input buffer to overflow.

An important and unique problem exists when using RTP sequence numbers to make join and leave decisions. No packets for the layer are received during the time it is dropped. Consequently, the first loss sample calculated by a receiver following a rejoin will be corrupted as the received packet count will be artificially low. One way to prevent the receiver from accidently dropping the layer due to the corrupted statistic is to ignore the first sample following a rejoin and wait for the next one before making a leave decision.

E. Quick Recovery Scheme

One of the problems with RTP is that it does not allow a participant to monitor RTCP control messages without receiving the corresponding data stream. This is the result of the specification of sending both streams on the same IP multicast address. A third-party monitoring application that wishes to diagnose faults or bottlenecks in the multicast distribution is one example of an application that does not wish to receive the data stream while receiving the control messages from all the participants.

It would also be useful for a receiver to continue to receive statistics on the pyramidal layer after it has dropped it. If the receiver, after dropping the pyramidal layer, is privy to the rate being transmitted on the pyramidal channel, it should be able to make a better estimate of its ability to rejoin. One way of accomplishing this is to diverge from the RTP specification. By sending each layer's RTCP stream on the multicast address of the base stream, the receiver will have access to the statistics of each layer as long as it is receiving the base stream. The sender's payload data rate can be estimated by inspecting the sender's octet count in the sender reports [14].

Now, after a receiver drops the pyramidal layer, it can monitor the pyramidal stream and detect when the source lowers its rate. At this time, it may prematurely rejoin the pyramidal stream. That is, it may now join earlier than usual since it has an indicator that it may be able to receive the stream. If the source is using source-based congestion avoidance, and the other receivers in the conference are agreeable to it, the source will lower its rate soon after the receiver has dropped the layer, and the receiver may quickly recover at a quality higher than the base layer stream alone can provide. Although this hybrid scheme may help in some cases, it has a couple of obvious problems. First, if there are many receivers, it is most likely that some will have sufficient network capacity and will prefer that the source not lower its rate to accomodate the receiver who has dropped a layer. Second, even if the other receivers have agreed to let the source drop its rate to provide a reasonable level of quality (perhaps $q = 3$), the bottleneck may still exist.

F. An Agent for the Rejoin Decision

What the receiver really needs to know is how much capacity is available at the bottleneck. The bottleneck is often, but not always, the last link in the multicast tree. An agent could live on the routers attached to the receiver's link. The agent on the router toward the source could provide its spare capacity to the receiver. If the receiver is receiving RTCP messages for the pyramidal layer on the base channel, it can subtract the rate of the pyramidal stream from the spare capacity statistic to get a very good idea if it should rejoin the pyramidal group if the bottleneck is indeed the local link.

Fig. 15 shows the streams receivers use to obtain statistics to make the rejoin decision. Since a router's capacity would only be needed rarely, it would only be broadcast on the local subnet in response to a query from the receiver. The query is multicast with a TTL of 1 and to a "well-known" multicast address. The query contains a single (source, group) pair if there is one source so that the correct router will respond to the receiver. If there are multiple sources, the situation becomes much more complicated unless each source uses a different set of multicast addresses for transmission. This is currently the only way to selectively reject different sources.

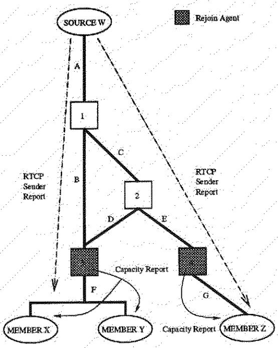

Fig. 15. Receivers Use Sender's Octet Count and Agent's Capacity Report.

CHAPTER IV

SIMULATION RESULTS

In this chapter, the results of different congestion avoidance schemes are presented and analyzed while keeping in mind that each method must also balance the desire to provide good quality video to the receivers. After showing the performance of Bill Fenner's multicast priority mechanism [17] used in these experiments, results for three congestion avoidance methods are examined.

A. Evaluation of Space Priority Scheme

The multicast priority mechanism was described on p. 11. It was designed to let MBONE users give priority to audio over video as degradation of audio due to packet loss is more irritating than degradation of video. It is desired to use this same mechanism to give priority to the base layer. When the rate of multicast bandwidth across the router exceeds the multicast rate limit set by the administrator, packets with lower priority are discarded in favor of higher priority packets. It is important to scrutinize the priority forwarding mechanism to make sure it provides this service which is crucial to the use of layered video.

A roughly two minute video sequence was chosen and multicast at four frames per second. No audio was transmitted. The multicast rate limit of a FreeBSD router was set at 100 kb/s. The sequence was played four times; twice with assigning both layers at the same priority, and then twice with the base layer given a higher priority than the pyramidal layer. Packet loss was measured at the receiver through the use of RTP sequence numbers. An independent confirmation of the loss statistics was provided by "netstat -gs." When a low priority packet is displaced by a high priority packet, it is reported as "datagrams selectively dropped." When a packet arrives and

Table I. Performance of Multicast Priority Forwarding, 100kb/s rate limit.

PRIORITY	BASE PACKETS LOST (lost/total)	PYRAMIDAL PACKETS LOST (lost/total)
off	527/926	1074/2204
off	522/917	1038/2187
on	1/848	1269/2019
on	0/858	1281/2044

the queue is full, it is reported as "datagrams dropped due to queue overflow [18]."

As can be seen in Table I, for the first two trials that do not prioritize packets, more than half of the base layer packets are lost. This causes unacceptable deterioration in the video quality. Between the next two trials, only one base layer packet was lost. The careful accountant will notice an apparent loss of packets as around 500 lost base packets turn into half that many pyramidal packets on the next two trials. This is partially caused by a shorter video sequence being transmitted as the total packet count is lower. In addition, the average pyramidal packet length is larger than the average base layer packet length because more pyramidal packets are generated per video frame. *All* of the pyramidal packets, except for the last one of the frame, are guaranteed to be greater than 1000 bytes.

It is evident that the priority scheme is very good. If there is a lot of motion in the scene, the rate for the base layer could easily exceed the router's 100 kb/s rate limit at four frames per second. This almost surely explains the lost base layer packet in the third trial.

B. Results of Source-based Congestion Avoidance

In the previous chapter, it was proposed that for two-layered video delivery, it is fair for a source to refrain from lowering its rate unless all receivers have either dropped the pyramidal layer or are losing a significant number of packets. This is justified because all receivers now benefit from a minimum 160x120 resolution on the base layer. The experiments in this section will use one receiver. Even the results of a simple scheme such as this will allow some interesting observations. The video sequence that is used for each trial is about ten minutes of the movie "When Harry Met Sally [30]." It begins with nearly two minutes of very low motion as Harry and Sally are sitting still watching "Casablanca," and throughout the rest of the sequence, there is a wide variance of motion. The sequence ends when an older woman says "I'll have what she's having."

The video is sent at 3 frames/s. Audio is also sent using VAT [31] PCM2 speech coding. Equal priority is given to the audio and the base layer. A lower priority is given to the pyramidal layer. The receiver is one hop away from the sender, separated by a FreeBSD router with a multicast rate limit set at 250 kb/s. With no feedback, results are shown in Fig. 16 for the congestion measured in lost packets/second at the router. Measurements are taken at 3 second intervals. Quality $q = 0$ is maintained for the entire sequence. Fig. 17 shows the bandwidth generated at the source for the 160x120 resolution stream and the bandwidth for the entire 320x240 resolution. These bandwidth measurements do not include bandwidth generated by any of the protocol headers. Availability of the program which runs on the router to measure discarded packets is mentioned in Appendix A.

The first observation that can be made is that with a low rate limit of 250 kb/s, the combination of the audio and video causes fairly consistent congestion

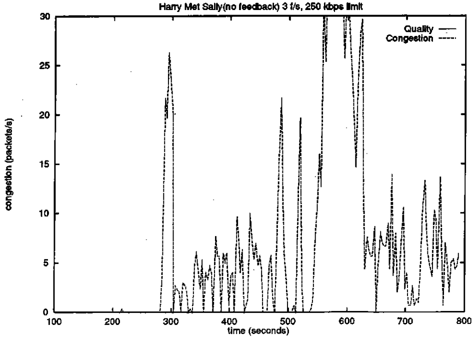

Fig. 16. Packet Loss and Quality, No Feedback.

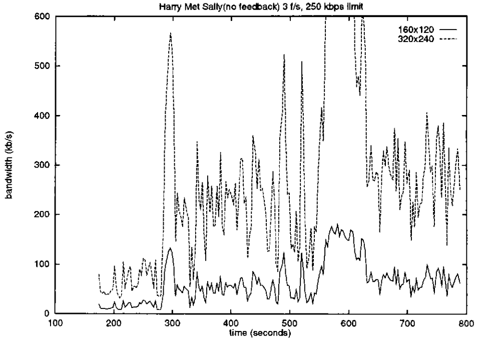

Fig. 17. The 160x120 and 320x240 Bandwidth, No Feedback.

starting right before the two minute mark with the initial spike which happens to be a scene change. It is also important to note that throughout these tests, there is no contention with other multicasts for the 250 kb/s of bandwidth. The receiver has 250 kb/s dedicated for itself. Also important is that in the second half of the sequence, there is an interval greater than 30 seconds where the combination of the audio and the base layer alone congests the router. This peak region for the base layer is observable in Fig. 17 between timestamps 580 and 610. Even if the pyramidal layer is not transmitted, the video quality will be seriously degraded. This suggests that, ideally, a rate ceiling should be set for the base layer, and when this rate is exceeded, the encoder should attempt to smooth its output to stay within the ceiling.

The first optimization to be attempted is for the source to lower the rate of the pyramidal layer by incrementing q when it receives a packet loss statistic of 20 percent or greater from the receiver. Receiver reports for these experiments are generated every 5 seconds, so the packet loss interval will also be 5 seconds. With less than twenty percent packet loss, the video quality has been perceived by the author to be fairly good. When the source receives a packet loss statistic of zero percent, it will decrement q by one. If it is in between zero and twenty percent, it does nothing. Figs. 18 and 19 show the results of this optimization.

In this run, $\alpha = 1$. All the weight is given to the current loss value. A simple observation to be made is that the output rate of the base layer did not change from the original run and will not change for any of the remaining runs. The congestion has been reduced somewhat, but the source is making the decision to increase its output rate too quickly. When q drops to one or zero, the coder is raising its rate drastically, and it is sending the router back into a congested state. Increasing the output bit-rate by simply decreasing q is counterproductive. Bandwidth needs to be added much more slowly to produce a stable system. This might be achieved by

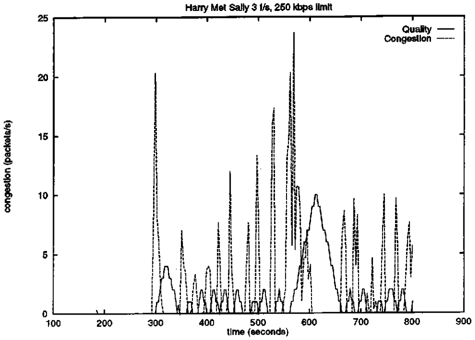

Fig. 18. Packet Loss and Quality, $\alpha = 1$.

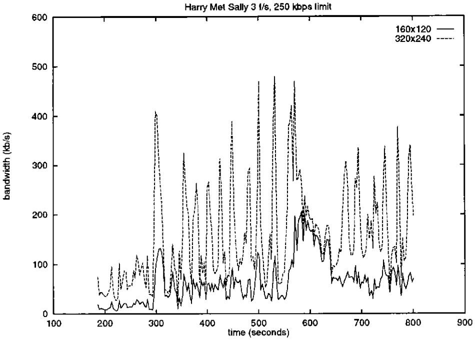

Fig. 19. The 160x120 and 320x240 Bandwidth, $\alpha = 1$.

shaping the traffic at the source. Also notice the initial spike again. This wasted bandwidth can not be fixed with a feedback scheme, but it might be fixed with traffic shaping at the source.

Although we can expect the drastic rate increase from $q = 2$ to $q = 0$ to continue to be a source of instability, it might be interesting to reduce α from one to other values. Figs. 20 and 21 show $\alpha = 0.75$. As can be seen, giving weight to previous values seems to be helpful in this situation as the source does not hurry as much to increase its rate. However, no amount of trickery using this method will overcome the instability. But this method might indeed work well with a traffic shaping scheme.

Again, the role of a source-based congestion scheme is limited in a multicast environment as it can not accommodate many participants. A much better alternative in multicast conferences is a receiver-driven scheme in which each receiver can determine for itself if its network can support the bandwidth required. If the network can not support it, the receiver asks the network to discard any enhancement layers it cannot support.

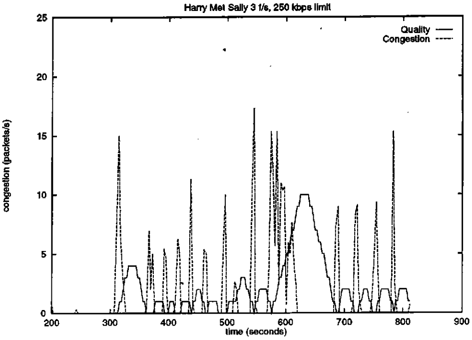

Fig. 20. Packet Loss and Quality, α = 0.75.

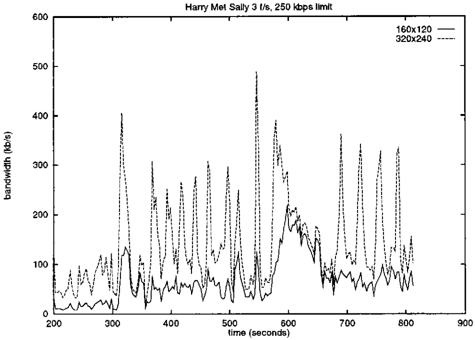

Fig. 21. The 160x120 and 320x240 Bandwidth, α = 0.75.

C. Results of Receiver-based Congestion Avoidance

As mentioned in the previous chapter, there is a problem when a receiver rejoins the pyramidal layer. The first loss sample after the rejoin will be artificially high because the received packet count was constant while the pyramidal layer was dropped, and the total packet count continued to increase. Figs. 22 and 23 show the effects of this problem.

Before discussing the problem, it must be mentioned that in order to counteract the system's instability, the source was modified to wait for eight consecutive favorable loss reports before it moves q from 2 to 1 or from 1 to 0, but it is still easy to move q from 3 to 2.

Each time the receiver rejoins the pyramidal group, it immediately leaves it the next time it samples packet loss. This indirectly, but unmistakenly, shows up in the quality graph in Fig. 22. The source responds to the artificially large packet loss statistic by incrementing q to 3. The router shows hardly any congestion since it is not trying to forward the pyramidal layer most of the time, but the receiver almost never gets the chance to receive good quality after it drops the pyramidal layer once. But more importantly, the lack of congestion is due to the new kludge of making the source wait for eight favorable loss reports before dropping q to one or zero. The receiver waits to rejoin after 10 packet loss samples, which in this case is 50 seconds.

Figs. 24 and 25 show these effects fixed. The quality does not bounce from 2 to 3 roughly every 50 seconds like the previous run. However, when the source lowers q to 0, the receiver must drop the layer again and wait a while. Again, remember that since the router is often not forwarding the pyramidal layer, the pyramidal stream is not contributing to the congestion.

The receiver of layered video is prepared to deal with indefinite congestion, but

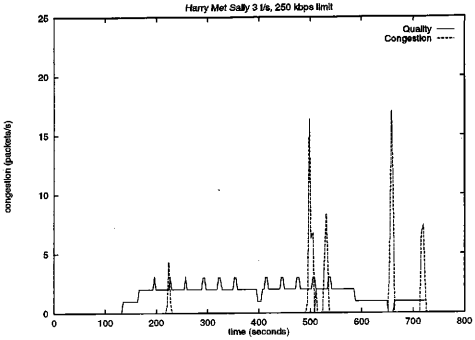

Fig. 22. Bad Rejoin Effects - Packet Loss and Quality.

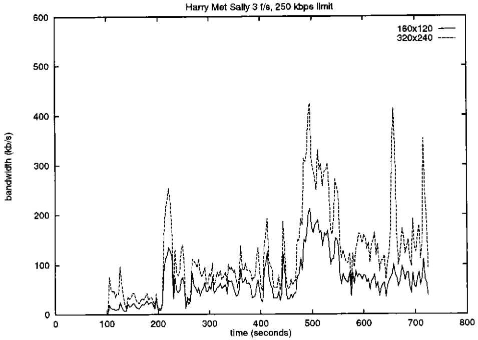

Fig. 23. Bad Rejoin Effects - Bandwidth.

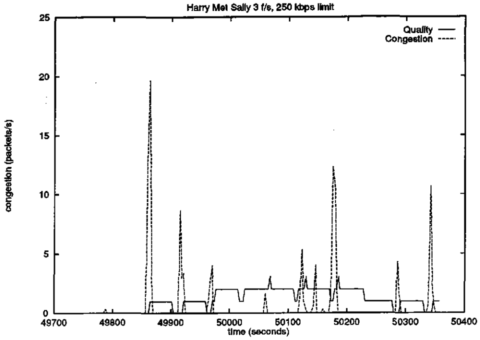

Fig. 24. Rejoin Effects Fixed - Packet Loss and Quality.

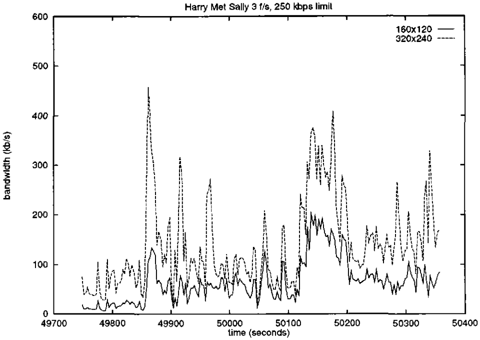

Fig. 25. Rejoin Effects Fixed - Bandwidth.

as the network is a shared resource, the receiver should act with some sense of urgency when it reports high packet loss so that it will not prevent others from getting useful work done. The interval over which to sample packet loss before making the decision to drop a layer is a question of how much pain the source causes the users on the network. This research will present results for a very conservative approach. The decision to drop the pyramidal layer will be made after each loss sample. A better idea of where the breaking point might occur would seem to require a more complex testbed and network simulations.

D. Results of Quick Recovery Scheme

In the receiver-based congestion avoidance experiments, the receiver drops a layer and the source responds by incrementing q. Now, the rate is at a level acceptable to the receiver, but the receiver has been maintaining a policy of waiting 50 seconds before attempting to rejoin the pyramidal group. If the receiver sees that the source has lowered its rate, it can try to rejoin the layer early. One way a receiver can detect the lowered rate is by simply inspecting the octet count in the RTCP sender report. To make the experiment even simpler, the source will append its quality value in the NAME field when it sends out an RTCP message. If the receiver sees the value of q increase, it will rejoin. The receiver will stop attempting to rejoin using this scheme when q increases from 3 to 4. At this point, the source is not lowering its rate enough to significantly increase the probability that the receiver will be able to successfully rejoin. If it successfully joins, the receiver will be able to enjoy a higher quality of video than it otherwise would be able to afford.

Figs. 26 and 27 show the results of the quick recovery scheme. On this trial, the times that the receiver joined and dropped the pyramidal layer are shown. With

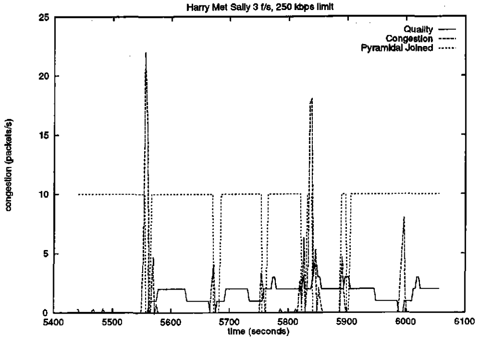

Fig. 26. Quick Recovery: Packet Loss and Quality.

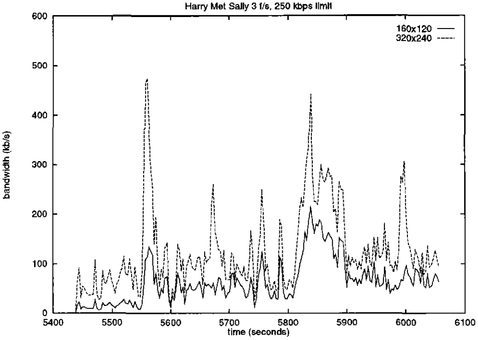

Fig. 27. Quick Recovery - Bandwidth.

the exception of the time slice from 5842 to 5888, the pyramidal stream was rejoined exactly ten seconds after it was dropped. The exception was due to q increasing to four. The fact that it would not have been able to successfully rejoin during this period even if it tried validates the decision to yield at $q = 4$. The video quality obtained during the video sequence was significantly improved since the receiver was able to make use of the pyramidal stream when it could. The video image more closely approximated the true 320x240 image since no more than 3 pyramidal difference values were mapped to zero instead of 15.

CHAPTER V

CONCLUSION

Integrating multimedia into an internetwork that was not designed for it is an exciting challenge, and much has been learned. Although there are many questions still to be answered, it is almost certain that hierarchical video will play an important role as networks will continue to be heterogeneous. Networks will need to be modified to provide priority service to handle multimedia traffic. Priority service is a requirement for hierarchical video. Source-based congestion avoidance schemes are good for unicast conferences, but are extremely limited in use for multicast. Receiver-based congestion avoidance appears to be a legitimate method for long-term congestion avoidance.

In this research, networking improvements were implemented for a popular video coder. The delivery of layered video was examined and implemented, and two new approaches to balancing video quality and congestion avoidance were described.

Packet processing at the receiver is rather complicated with two-layer video as packets appear on two different sockets. Although IP multicast delivers all packets for a multicast group in order, pyramidal packets may be processed out of order with respect to base layer packets at the application level. Although this causes a wide variety of problem conditions that must be serviced, a working implementation was developed that demonstrates appealing video even during heavy packet loss.

It was noted that with receiver-based congestion avoidance, the method that is defined to report packet loss in the RTP specification breaks down when applied to a multicast stream that is dynamically pruned and joined throughout the conference. After a receiver rejoins a pyramidal layer, its first loss sample can be thrown out to prevent the receiver from thinking that packets it did not ask for were actually dropped due to congestion.

There is a tradeoff involved with a receiver-based congestion avoidance scheme. The receiver wishes to drop the pyramidal layer as soon as possible if there is going to be persistant congestion, but it does not want to drop it at all if the congestion turns out to be short-term. How long a receiver can wait before it makes the decision to drop a layer depends on how much pain the unpruned source is causing the other users on the network.

A hybrid scheme was shown to have an advantage over the receiver-based scheme alone if the other receivers were agreeable to a slight degradation in their quality. Whereas a receiver-based scheme alone is powerless in its ability to let a receiver who has dropped a layer receive better quality, the hybrid scheme allows the receiver to quickly upgrade its video quality and keep it for the long duration that it would otherwise be degraded.

Finally, a scheme that seems to be promising was described which requires not only output rate reports from the sender, but free capacity reports from the local router to determine if the receiver can rejoin with a higher probability of success. Although the method described only works when the bottleneck is the local link, this is a common situation. Consequently, it is an interesting area for further research.

Another area for further research stems from the degraded quality of service that was observed in the video sequence when base layer packets were lost. It would be helpful to impose a limit on the output rate of the base layer. Smoothing the output rate of the base layer to stay underneath the limit while still providing good video seems challenging.

REFERENCES

[1] R. Cogger, CU-SeeMe, Cornell University. Software available via anonymous ftp at cu-seeme.cornell.edu.

[2] S. Sazzad, "Design of a Three Resolution Coder," M.S. Thesis, Texas A&M University, pp. 56–64, Dec. 1995.

[3] S. Deering, R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification," Internet draft, March 17, 1995.

[4] C. Aras, J. Kurose, D. Reeves, H. Schulzrinne, "Real-Time Communication in Packet-Switched Networks," *Proceedings of the IEEE*, vol. 82, pp. 122–139, Jan. 1994.

[5] N. Shacham, "Multicast Routing of Hierarchical Data," *Conference Record of the International Conference on Communications (ICC)*, Chicago, Illinois, pp. 1217–1221, IEEE, June 1992.

[6] N. Shacham, "Multipoint Communication by Hierarchically Encoded Data," *Proceedings of the Conference on Computer Communications (IEEE Infocom)*, vol. 3, Florence, Italy, pp. 2107–2114 (9A.4), IEEE, May 1992.

[7] G. Karlsson, M. Vetterli, "Packet Video and Its Integration into the Network Architecture," *IEEE Journal on Selected Areas in Communications*, vol. 7, no. 5, pp. 739–51, June 1989.

[8] I. Wakeman, "Packetized Video - Options for Interaction Between the User, the Network and the Codec," *The Computer Journal*, vol. 36, no. 1, pp. 55–67, 1993.

[9] S. Deering, "Internet Multicast Routing: State of the Art & Open Research Issues," MICE Seminar, Stockholm, October 19, 1993.

[10] G. Wright and W. R. Stevens, *TCP/IP Illustrated*, vol. 2. Reading, Massachusetts, Addison Wesley Publishing Company, 1995.

[11] D. Waitzman, C. Partridge, S. Deering, "Distance Vector Multicast Routing Protocol," RFC 1075, Nov 1988.

[12] B. Fenner, "IP Multicast Routing Daemon." Manual page, Nov. 1995.

[13] S. Deering and D. Cheriton, "Multicast Routing in Datagram Internetworks and Extended LANs," *ACM Transactions on Computer Systems*, vol. 8, pp. 85–110, May 1990.

[14] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications," Internet draft, March 1995.

[15] S. McCanne and V. Jacobson, "vic: A Flexible Framework for Packet Video," *Proc. of ACM Multimedia '95*, Nov. 1995.

[16] A. Banerjea, D. Ferrari, B. Mah, M. Moran, D. Verma, and H. Zhang, "The Tenet Real-Time Protocol Suite: Design, Implementation, and Experiences," Technical Report TR-94-059, International Computer Science Institute, Berkeley, California, November 1994.

[17] B. Fenner, FreeBSD source code, "/usr/src/sys/netinet/ipmroute.c," 1995.

[18] B. Fenner, Private Communication, Xerox Palo Alto Research Center, Palo Alto, California, Nov. 1995.

[19] S. Deering, "Reservations or No Reservations," INFOCOM '95 Panel Session, Apr. 1995.

[20] W. R. Stevens, *UNIX Network Programming*, Englewood Cliffs, New Jersey: Prentice-Hall, 1991.

[21] S. McCanne, Private Communication, Lawrence Berkeley National Laboratory, Berkeley, California, Feb. 1996.

[22] idmr@cs.ucl.ac.uk mailing list. Archive available by anonymous FTP from cs.ucl.ac.uk:/darpa/idmr-archive.Z

[23] R. Frederick, Network Video (nv), Xerox Palo Alto Research Center. Software available ftp://ftp.parc.xerox.com/net-research.

[24] J. Bolot and T. Turletti, "A Rate Control Mechanism for Packet Video in the Internet," *IEEE Infocom*, Toronto, Canada, June 1994.

[25] I. Busse, B. Deffner, and H. Schulzrinne, "Dynamic QoS Control of Multimedia Applications Based on RTP," *Computer Communications*, Jan. 1996.

[26] S. Chakrabarti and R. Wang, "Adaptive Control for Packet Video," *IEEE Multimedia International Conference*, May 1994.

[27] H. Kanakia, P. Mishra, and A. Reibman, "An Adaptive Congestion Control Scheme for Real-Time Packet Video Transport," *ACM Computer Communications Review*, pp. 20–31, Sept. 1993.

[28] T. Turletti and J. Bolot, "Issues With Multicast Video Distribution in Heterogeneous packet networks," *Proc. 6th International Workshop on PACKET VIDEO*, Portland, Oregon, Sept 26–27, 1994, pp. F3.1–3.4.

[29] S. McCanne and M. Vetterli, "Joint Source/Channel Coding for Multicast Packet Video," *IEEE International Conference on Image Processing*, pp. 776–85, Oct., 1995.

[30] "When Harry Met Sally," Columbia Pictures, Hollywood, California, 1989.

[31] V. Jacobson and S. McCanne, Visual Audio Tool (vat), Lawrence Berkeley Laboratory. Software available ftp://ftp.ee.lbl.gov/conferencing/vat.

[32] W. R. Stevens, *TCP/IP Illustrated*, vol. 1. Reading, Massachusetts, Addison Wesley Publishing Company, 1994.

APPENDIX A

MULTICAST VIDEOCONFERENCING - CAFEMOCHA

CafeMocha is a one-to-many implementation of a pyramidal coder which divides video into 3 separate streams of information and distributes them using three multicast channels. The medium resolution video is coded using the CU-SeeMe compression algorithm and is sent on one multicast address. The large resolution video uses 2 multicast addresses; the medium channel plus an enhancement channel.

It is not available on-line for public consumption, but may be obtained through the Multimedia Communications and Networking Lab through the email address students@www-mcnl.tamu.edu.

VITA

Thomas B. Brown was born on September 3, 1969 in Houston, Texas. He received the degree of Bachelor of Science in Electrical Engineering from Texas A&M University (College Station, TX) in May 1992. He can be reached through his email address tbrown@vcsun2.tamu.edu.

Mr. Brown can be reached at 3312 Chapparal Ft. Worth, Texas 76109.

The typist for this thesis was Thomas B. Brown.