Incorporating Application-Level Knowledge into the MPEG-2 Coding Model

Ketan Mayer-Patel
Department of Computer Science
University of North Carolina, Chapel Hill
kmp@cs.unc.edu

Abstract

Current multimedia standards (in particular video standards) are inflexible and designed for specific applications, often without regard for heterogenous network environments like the Internet. Application-level semantics about the content and context of video information should be reflected in how video is coded and transmitted. Current standards are inflexible because interpretation and management of different elements of the video stream are implicit and intertwined. Encoded video formats can be recast into a more flexible model by separating the generation, management, and transmission of different video elements. A more flexible coding framework will allow applications to better manage network transmission of video stream elements in a manner that reflects application-level knowledge and requirements. Furthermore, the coding model can be adapted to changing network conditions. This paper specifically addresses how reference frame management in the MPEG standards can be recast into a more flexible coding model. By separating the management of reference information from the coding of specific frames, we enable the standard to be more effectively used when application-level knowledge of the video content is present. A number of interesting network strategies are enabled by the proposed revised model.

1 Introduction

Current multimedia standards (in particular video standards) are designed for specific applications and network environments and as a result are inflexible. Assumptions about the networking environment, computing capabilities of the receiver, and the type of video encoded are often made for the sake of achieving greater compression rates. Unfortunately, these video formats are subsequently used in environments where the original assumptions do not hold true (i.e, the Internet), resulting in poor quality, inefficient network usage, and/or severe degradation of application performance in the face of packet loss and limited bandwidth. A priori knowledge about the content of the video stream can only be used in limited ways because video standards are written to support the most general possible video content.

While many efforts have focused on designing and implementing network protocols and mechanisms to facilitate the transmission of streaming packet video, there have been few efforts aimed at creating adaptive coding models that can be dynamically altered to react to changing network conditions. In this paper, we propose changes to the MPEG-2 coding model that enable a number of new application-level networking strategies. These new strategies hinge on the ability of applications to exploit application-level knowledge of video content.

Application-level semantics about the content and context of video information need to be reflected in how video is coded and transmitted. To this end, video standards need to provide ways of adapting the video format to better match the heterogenous needs of different applications and varying network conditions. The cost of these adaptation mechanisms may be a loss of compression, but this cost is recouped in the form of higher effective network utilization (i.e., "goodput"), increased video quality, and graceful degradation in the face of packet loss and bandwidth limitations. Furthermore, a priori knowledge about the content of the video stream can be used to adapt the coding model to actually improve compression. For example, the encoding of a video source that is known to be showing the interior of a particular room can be tuned to reflect the limited set of objects that will appear in the video stream.

Current standards are inflexible because interpretation and management of different elements of the video stream are implicit and intertwined. For example, MPEG-2 encoded video is generally comprised of DCT coefficients and motion vectors for one luminance plane and two chrominance planes. The MPEG-2 syntax intertwines the variable length codes for coefficients and motion vectors in a way that makes it difficult for an application to deal with these elements separately or to deal with any of the three planes separately. Furthermore, the use of temporal redundancy in MPEG-2 is strictly constrained by the specification of three types of frame encodings (i.e., I-, P-, and B-frames). Each frame type creates specific temporal relationships with other frames. No mechanism exists for these temporal relationships to be redefined by the application for its own purposes.

Encoded video formats can be recast into a more flexible model by separating the different video elements and allowing application-level control for how these different video elements are related to each other. Given this flexible coding framework, applications can manage the elements of a video stream in a manner that reflects application-level knowledge and requirements. The encoded video stream becomes a collection of different encoding elements which can be combined in multiple ways to create a video stream encoding congruent with the specific requirements of the application at hand. A number of parallel research efforts are working toward this vision of "object-based" video [6, 5].

The idea of an adaptable, flexible video coding model dovetails with the networking concept of Application Level Framing (ALF) [3] by allowing an application to optimize its video coding model in regard to its networking requirements and changing network conditions. Although separating video encoding elements provides much needed flexibility, it also creates a number of networking challenges. In particular, different encoding elements may have different transport-level protocol requirements, especially in regard to reliability. Using different streams to transport encoding elements, however, creates problems with synchronization and naming across streams.

This paper describes preliminary work in progress in which the MPEG-2 video coding model is modified to allow flexible application-level control over how reference video data (i.e., video addressed by motion vectors) is defined and managed. These modifications are an example of separating coding elements and providing a means for independently specifying how these elements relate to each other. We believe that this strategy will be employed with increasing frequency with the development of MPEG-4 [5], H.263+ [11], and other new multimedia standards. The goal of this work is to expose and address the networking challenges created by this strategy. The use of MPEG-2 is motivated by the fact that it is a stable standard and a wide variety of hardware and software tools to handle the format exist today. In fact, the choice of MPEG-2 for this work is somewhat arbitrary. The proposed coding model described in this paper *completely* separates reference video data management from the syntax of the encoded video stream. Thus, the ideas and challenges identified in this paper can be as easily applied to other video encoding standards that exploit temporal redundancy in video through the use of interframe encoding (e.g., H.263+). This paper serves to outline how we propose to approach the problem and raise the issues we are trying to target.

The rest of this paper is organized as follows: Section 2 describes the current MPEG-2 reference video model; Section 3 describes several motivating examples for why the current model is inadequate, outlines a proposed revised model, and identifies networking challenges that are raised by the revised model; Section 4 describes related work; and Section 5 summarizes the paper.

2 MPEG-2 Reference Video Model

This section reviews the management of reference video information in the current MPEG-2 standard. The term "reference video" is used to refer to the video frame buffers addressed by motion vectors during motion compensation in the MPEG-2 encoding/decoding process. These frame buffers are also commonly known as the "forward" and "backward" reference frames. We begin by providing a high-level abstract description of the MPEG-2 encoding and corresponding decoding models and the role of reference video information in those models.

Figure 1 shows a block diagram of the MPEG-2 encoding and decoding processes. This diagram is not complete, but it illustrates enough of the coding process for our purposes. Incoming video frames are segmented into 16 pixel by 16 pixel re-

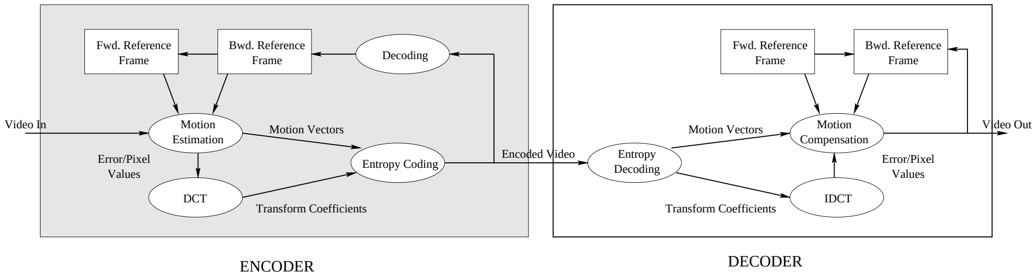

Figure 1: MPEG-2 Encoding and Decoding Models

gions called macroblocks. Possibly associated with each macroblock are a set of motion vectors. Typically, up to two motion vectors are provided. The motion vectors indicate regions within the *forward* and *backward* reference frames which form a prediction for the pixel values of the macroblock. How these reference frames are specified is described below. The prediction is subtracted from the pixel values of the macroblock and the resulting error values are DCT encoded using the Discrete Cosine Transform (DCT). If no motion vectors are provided, then the macroblock is considered to be "intracoded" and the original pixel values are used for the DCT (i.e., no prediction is formed and subtracted). The coefficients from the DCT are quantized, run-length encoded, and along with the motion vectors encoded using variable-length entropy encoding (i.e., Huffman codes). The decoding process is essentially the same in reverse.

Each encoded frame is set to be one of three types: *I*, *P*, or *B*. I-frames are not dependent on any other frame, P-frames are dependent on the previous I- or P-frame, and B-frames are dependent on the previous I- or P-frame as well as the subsequent I- or P-frame. Figure 2 illustrates a typical encoding pattern. Because B-frames may depend on frames that actually occur after them in display order, the frames are transmitted in dependency order and reordered during the decoding process. Because I-frames must be freestanding (i.e., not dependent on other frames), I-frames use no motion compensation whatsoever. In other words, all macroblocks of an I-frame are intracoded. P-frames only use forward prediction and therefore all motion vectors provided point only into the forward reference frame. B-frames use both forward and backward prediction. The I-, P-, and B-frame designation directs how the decoder manages the forward and backward reference frames. Upon encountering an I- or P-frame, the decoder discards the current forward reference frame and installs the current backward reference frame as the new forward reference frame. After decoding, the frame is installed as the new backward reference frame. Decoding B-frames does not affect the reference frames.

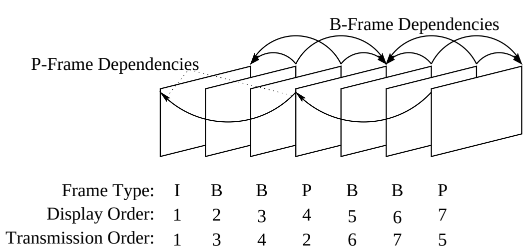

Figure 2: Relationship between I-, P-, and B-frames.

Reference frame management in the MPEG-2 standard is an example of how different elements of the video coding model are inflexibly intertwined. The specification of what is installed as reference frame information is tied to the encoding of the frames themselves. The dependencies created between I-, P-, and B-frames are rigidly specified.

3 Revised MPEG-2 Reference Video Model

This section describes a revised reference frame management model that allows application-level knowledge to be incorporated into how reference frame data is handled. We motivate how a revised reference model may be useful with examples, outline our ideas for a new reference frame management model, and describe some of the network challenges that we expect to address to support the new model. The central feature of our revised model is that reference frame management is made separate and explicit from the encoding of specific frames. Thus, in addition to the encoded frames,

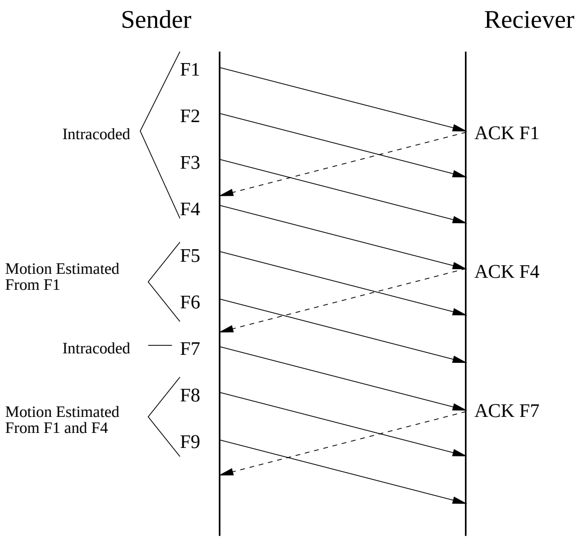

Figure 3: Using reference management to mitigate effects of packet loss.

reference frame management "instructions" will be sent to designate that a frame may be used as a reference frame in the future and to install specific frames previously sent as either the forward or backward reference frame.

3.1 Motivating Examples

By separating reference frame management from other elements of the decoding model, we can incorporate application-level knowledge about the video source and network environment. Minimizing the impact of packet loss is an example of how an application can use this ability. In the current MPEG reference model, the encoding of an I-frame immediately installs it as a reference frame for the immediately preceding B-frames as the backward reference and the subsequent P- and B-frames as the forward reference. Packet loss that occurs while transmitting the I-frame to the receiver results in severe errors for all of the frames that depend on it. Furthermore, since P-frames create a dependency chain, these errors are propagated until the next I-frame. Typically this means that 15 frames will have severe errors in them.

If the coding model is changed to allow the application to delay installation of a reference frame, feedback from the receiver can be used to determine if a reference frame was successfully received before it is used as a reference frame. Figure 3 illustrates this scheme. In this example, frames F1 through F4 are intracoded (i.e., do not use motion compensation). F1 is transmitted along with reference management commands that indicate that F1 may be used as a reference frame in the future and should be acknowledged. After the acknowledgement for F1 is received by the sender, F5 and F6 can be encoded with motion compensation using F1 as a reference frame. F5 and F6 are transmitted along with reference management commands that instruct the receiver to install F1 as a reference frame. In a similar fashion, after F4 has been acknowledged, F8 and F9 can use both F1 and F4 as reference frames. Periodically, new reference frame candidates are transmitted (e.g., F4 and F7). This scheme is more robust to packet loss because the installation and use of reference frames is driven by explicit acknowledgements from the receiver.

In our example, we disguise the fact that frames are transmitted as a series of packets and not just a single packet. By incorporating knowledge about how frames are packetized, more sophisticated strategies for dealing with packet loss can be implemented. If only a single packet of an reference frame is lost, the application can proceed by explicitly installing the reference frame and avoiding the use of motion vectors that point into the area of the frame represented by the lost packet.

The cost of using the scheme described above is that the temporal distance between a reference frame and frames that depend on the reference frame is at least one round trip time. This temporal distance makes the reference frame less useful because the content of the reference frame is no longer as closely correlated to the content of the dependent frames. This may be especially true in the case of high motion. The advantage of the revised model is that the application which has knowledge of the kind of video being transmitted can adjust its strategies accordingly. The original reference frame management model can always be implemented using the new model.

The scheme described above is essentially the same as the *Reference Picture Selection* mode of H.263+ [11]. This mode is also known as *NEWPRED*. An important distinction between our proposed model and NEWPRED is that we are attempting to decouple reference picture management from the rest of the coding model completely. NEWPRED relies on H.263-specific structure and syntax for naming reference frames and does not allow for out-of-band specification of reference data. Our vision is to construct video codecs that are assembled from distinctly specified modules. In this case, one that encodes picture data and one that manages reference video data. This approach enables the use of a wide variety of possible schemes including the NEWPRED-like scheme described above. Other examples of how a decoupled reference frame management scheme might be used include:

- *Calculating an "optimal" set of reference frame data for video sources with well-known content characteristics.* An off-line process may be able to determine an optimal set of reference frame data to be used for the encoding of a particular video source. The reference data set may then be distributed prior to streaming the video. The video can be encoded assuming that the receiver has already obtained the necessary reference data. This approach trades off start-up latency (i.e., the time required to obtain the reference data set) for better compression and robustness to packet loss.

- *Caching reference frame data across different communication sessions.* If the video source is part of a telepresence or distance learning environment, reference data may be cached across different sessions because the same set of participants are periodically receiving video streams that have similar content from one session to another.

- *Using model-based techniques to create reference data sets that can be "layered" to accommodate parallax with camera movement.* If the application has knowledge of the 3-D geometry of the scene represented by the video, reference frames can be constructed from pieces of reference data that are associated with a particular depth and position. As the camera moves, the reference data can be appropriately warped and layered to form good reference frames for the new camera position.

3.2 Revised Model

Figure 4 shows the proposed new reference frame model. The coding model illustrated in Figure 1 is separated into two components. One component represents the the MPEG-2 process for encoding picture data (i.e., the existing MPEG-2 coding model). The second component embodies a new process called the "Reference Manager." The reference manager is responsible for implementing any application-specific reference data management schemes. We expect that different applications will provide appropriate implementations of the reference manager. On the encoding side, the reference manager produces reference management commands that indicate to the reference manager on the decoding side on how reference frame data is to be manipulated. The encoding reference manager may also send out-of-band reference data. The reference manager on the decoding side is responsible for producing any required feedback to indicate which pieces of reference frame information were successfully received. The reference frames within the MPEG-2 encoding process are relabeled "R1" and "R2" instead of "forward" and "backward" because their use is no longer necessarily in-line with the original labels. Additional memory available to the reference manager processes is used as a cache for reference frame data that may be specified but not yet installed into one of the reference frame buffers. This additional memory is labeled "Reference Cache" in Figure 1.

One feature of the revised model is that the syntax of MPEG-2 encoded frames remains intact. The use of reference frame data by dependent frames remains the same (i.e., syntax of motion vectors, number of reference frames available, etc.). Breaking the standard as little as possible is important to expedite the development of experimental applications that use the new revised model. Because the management of the reference frame buffers themselves is now under the control of a new mechanism not specified in the standard, the new model is not backwards compatible with existing MPEG-2 software and hardware. But, by leaving much of the syntax of encoded frames intact, existing software decoders can be more easily modified to support the new revised model.

3.3 Networking Challenges

In the description of the revised model we are working toward, we have purposefully avoided specifying exactly how reference frame data and management commands are specified and transported. The challenges raised by the transport-level requirements of these encoding elements are complex. Good solutions will require careful experimentation. Among the several challenges that must be addressed are:

- *The interface between the reference management process and the coding process needs to be defined.* To realize our goal of completely separating the coding process from the management of reference data, a standard interface

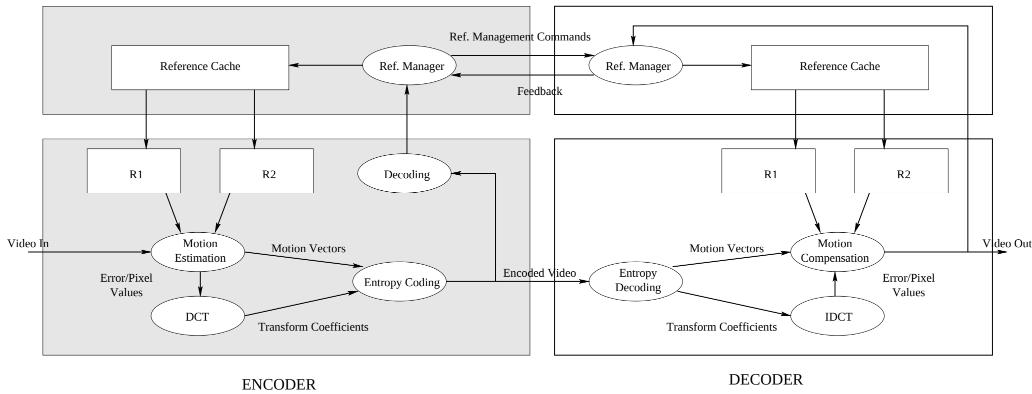

Figure 4: Revised MPEG-2 coding model.

between these two processes must be defined. This interface must allow the coding process to specify the amount and structure of addressable reference memory available and provide methods for installing and uninstalling specific portions of that reference memory.

- *A naming scheme for reference data needs to be developed.* Reference management commands will need a method of defining portions of encoded video frames that may be used as reference data. Thus, frames and subregions within frames need to be uniquely identified. The naming scheme must also support out-of-band specification of reference data (i.e., reference frame data that is not actually part of the encoded video stream). Another desired feature of the naming scheme is the ability to create persistent names which can be used across sessions. Using persistent names will allow applications to cache reference data from a video source for use in the future. Finally, the naming scheme needs to also be as compact as possible to minimize the loss of compression suffered by using the scheme.

- *A reference management command set needs to be defined.* The set of operations available will determine in what ways applications can leverage the ability to manage reference data. A sparse set of reference management commands will not provide enough flexibility, while a complicated command set will make decoders slow and complex.

- *Transport protocols for reference management commands and data need to be defined.* The key challenge with transporting reference management commands and data is the variable "lifetime" associated with these elements. Unlike video frames which have very short lifetimes (i.e., typically 1/30th of a second for B-frames and up to 1/2 second for P- and I-frames), reference commands and data may have very long lifetimes depending on how an application uses them. The relative reliability associated with these elements must be proportional to these lifetimes.

- *The use of network resources must be coordinated among multiple streams* In the revised model a video stream is no longer a single bitstream. Instead, two separate, but associated, streams are created, one for encoded picture data and one for reference management commands and out-of-band reference data. The transport-level network resources used by these two streams (i.e., bandwidth, etc.) must be coordinated.

- *A feedback protocol needs to be developed.* Many of the strategies that are possible with the revised model depend on feedback from the decoder. The form of this feedback and how it communicated back to the sender needs to be defined. We intend that the feedback protocol be usable in both point-to-point streaming applications as well as in multicast applications in which receivers send feedback probabilistically.

- *Open-loop strategies that avoid explicit feedback need to be developed.* In broadcast applications in which audience size is massive, feedback will be infeasible. Open-loop strategies that still take advantage of explicit reference frame management need to be developed for these applications.

For many of these challenges, existing work can be used as a starting point and adapted to these issues. In particular, the current RTP payload formats for MPEG-2 may be used. The Scalable Naming and Announcement Protocol (SNAP) is a possible framework for developing the required naming scheme. The Congestion Manager framework provides a framework for coordinating the use of network resources among multiple streams [1]. We are most interested in exposing how existing protocols fail to meet these needs.

4 Related Work

A variety of other research efforts are also developing the idea of flexible coding models that are adapted to specific application needs. Most notably, the work of Bove, et. al [6], has focused on the concept of "object-based" video representations. In their work, video is described by a language that combines different coding elements. Their work has concentrated on developing a video description language and building hardware to implement their coding model.

The H.263+ encoding standard [11] specifies extension modes for alternative reference frame management. These modes include specifications for back channel feedback messages. These extension modes, however, are strictly tied to the H.263 syntax and rely on H.263-specific coding structures. Additionally, out-of-band reference video data is not supported.

The concept of tuning transport-level network protocols to account for application-level semantics of the transmitted data was first clearly articulated by Clark and Tennenhouse as "Application Level Framing" [3]. This concept was applied to the communication of video information by McCanne with the development of Intra-H.261 [8, 7]. His work demonstrated the effectiveness of joint source/channel coding in which the video coding model reflects network-level knowledge about packet loss rates and optimal packet sizes. The development of the Real Time Protocol (RTP), the associated Real Time Control Protocol (RTCP), and specific payload for video formats within RTP were heavily influenced by ALF. We expect to implement our communication protocol within the RTP framework.

The Scalable Naming and Announcement Protocol (SNAP) provides an interesting possible approach to the naming issues identified in our work [10]. This protocol allows participants to build source-based hierarchical namespaces. The protocol builds on an underlying announce/listen mechanism (i.e., SRM [4]) to communicate the namespace to other participants in a scalable manner.

The MPEG-4 standard [5] provides extensive support for synchronizing and multiplexing different flows related to a single multimedia presentation. The system and synchronization layers of MPEG-4 may prove useful as a framework for relating the reference management and encoded picture data streams. However, transport-level issues (e.g., bandwidth management, reliability, packetization, etc.) are not addressed.

Bolot, et. al, have explored the use of forward error correction (FEC) to provide adaptive relative reliability for the delivery of multimedia data (in particular, audio) [2]. Podolsky, et. al, are approaching the problem of relative reliability by developing retransmission schemes within the RTP framework specifically designed for multimedia data types [9].

Balakrishnan, et. al, are exploring the problem of coordinating the use of network resources by multiple flows [1]. Their work has concentrated on coordinating the congestion management actions of separate instances of different protocols. The framework they have developed provides an API to incorporate application-level preferences in managing bandwidth among flows.

5 Summary

This paper outlined a proposed revision of the MPEG-2 coding model to allow explicit application control over reference frame management. Several motivating examples were given to illustrate how such control could be used to adapt video coding and transmission to the specific needs of an application and incorporate application knowledge about the content and context of the video stream. Two key features that the proposed model will support is a flexible naming scheme for referring to reference video frame data and a set of management commands that can be used by applications to implement different management strategies. A variety of anticipated network challenges were identified. One key challenge is the development of a streaming protocol that transmits elements (i.e., reference management commands and data) with reliability requirements relative to its expected lifetime and the lifetimes of other related elements.

Providing application-level control over reference frame management is an example of the direction that emerging multimedia standards are moving toward to accommodate heterogenous application requirements and the dynamic network environment of the Internet. The networking issues identified in this paper are likely to be faced by these emerging standards. By exploring these issues within a specific context, we hope to gain insight on viable techniques that can be applied more generally.

References

[1] Hari Balakrishnan, Hariharan S. Rahul, and Srinivasan Seshan. An integrated congestion management architecture for internet hosts. *Proceeding of ACM SIGCOMM*, September 1999.

[2] J.-C. Bolot, S. Fosse-Parisis, and D. Towsley. Adaptive fec-based error control for internet telephony. *Proceeding of InfoComm 1999*, March 1999.

[3] D.D. Clark and D.L. Tennenhouse. Architectural considerations for a new generation of protocols. *Proc. ACM SIGCOMM 1990, Computer Communication Review*, 20(4):200–208, September 1990.

[4] S. Floyd, V. Jacobson, C.-G. Liu, S. McCanne, and L. Zhang. A reliable multicast framework for light-weight sessions and application level framing. *IEEE/ACM Transactions on Networking*, December 1997.

[5] Working Group 11 ISO/IEC Joint Committee 1, Subcommittee 29. *Overview of the MPEG-4 Standard*, December 1999.

[6] V. M. Bove Jr. Multimedia based on object models: Some whys and hows. *IBM Systems Journal*, 35:337–348, 1996.

[7] S. McCanne. *Scalable Compression and Transmission of Internet Multicast Video*. PhD thesis, University of California, Berkeley, 1996.

[8] S. McCanne and V. Jacobson. vic: a flexible framework for packet video. *Proceedings of ACM Multimedia '95*, pages 511–522, 1995.

[9] M. Podolsky, S. McCanne, and M. Vetterli. Soft arq for layered streaming media. *Journal of VLSI Signal Processing Systems for Signal, Image and Video Technology, Special Issue on Multimedia Signal Processing*, April 2000.

[10] S. Raman and S. McCanne. Scalable data naming for application level framing in reliable multicast. *Proceeding of the ACM Multimedia Conference 1998*, 1998.

[11] International Telecommunication Union. *H.263: Video Coding for low bit rate communication*, February 1998.