Advances in Network-adaptive Video Streaming

Bernd Girod, Jacob Chakareski, Mark Kalman, Yi J. Liang, Eric Setton, and Rui Zhang

Information Systems Laboratory, Department of Electrical Engineering
Stanford University, Stanford CA 94305, USA

ABSTRACT

Internet transmission is characterized by variations in throughput, delay, and packet loss, which can severely affect the quality of multimedia presentations delivered over the network. Still, Internet video streaming has experienced phenomenal growth in the last few years, owing to the extensive research in video coding and transmission. In this paper, we review several recent advances for network-adaptive video streaming that, we believe, will benefit the design of video streaming systems in the future. Employed in different system components, these techniques have the common objective of providing efficient, robust, scalable and low-latency streaming video. They range from purely server or source encoder-based techniques, through transmission schemes that could be implemented either at the sender or at the receiver, to purely client-based techniques. We discuss each of them in detail, presenting also related work and experimental results. We end the paper with a summary of the reviewed techniques and a brief discussion of future research directions.

1 INTRODUCTION

Since the introduction of the first commercial products in 1995, Internet video streaming has experienced phenomenal growth. Over a million hours of streaming media contents are being produced every month and served from hundreds of thousands of streaming media servers. Second only to the number-one Web browser, the leading streaming media player has more than 250 million registered users, with more than 200,000 new installations every day. This is happening despite the notorious difficulties of transmitting data packets with a deadline over the Internet, due to variability in throughput, delay and loss. It is not surprising that these challenges, in conjunction with the commercial promise of the technology, have attracted considerable research efforts, particularly directed towards efficient, robust, scalable and low-latency video coding and transmission [1, 2].

A streaming video system has four major components: 1. The encoder application (often called the "producer" in commercial systems) that compresses video and audio signals and uploads them to the media server. 2. The media server that stores the compressed media streams and transmits them on demand, often serving hundreds of streams simultaneously. 3. The transport mechanism that delivers media packets from the server to the client for the best possible user experience, while sharing network resources fairly with other users. 4. The client application that decompresses and renders the video and audio packets and implements the interactive user controls. For the best end-to-end performance, these components have to be designed and optimized in concert.

The streaming video client typically employs error detection and concealment techniques to mitigate the effects of lost packets [3]. Unless forced by firewalls, streaming media systems do not rely on TCP for media transport but implement their own application level transport mechanisms to provide the best end-to-end delivery while adapting to the changing network conditions. Common issues include retransmission and buffering of packets [4], generating parity check packets [5], TCP-friendly rate control [6], and receiver-driven adaptation for multicasting [7]. New network architectures, such as DiffServ [8] and path diversity transmission in packet networks [9], also fall into this category.

The media server can implement intelligent transport mechanisms by sending out the right packets at the right time, but the amount of computation that it can perform for each media stream is very limited due to the large number of streams to be served simultaneously. Most of the burden for efficient and robust transmission is therefore on the encoder application, which cannot adapt to the varying channel conditions and must rely on the media server for this task. Rate scalable representations are therefore necessary to allow adaptation to varying network throughput without requiring computation at the media server. Multiple redundant representations are an easy way to achieve this task, and they are widely used in commercial systems [4]. To dynamically assemble compressed bit-streams without drift problems, S-frames [10] and, recently, SP-frames [11] have been proposed. Embedded scalable video representations such as FGS [12] would be more elegant for rate adaptation, but they are still considerably less efficient, particularly at low bit-rates. Embedded scalable representations are a special case of multiple description coding of video [13] that can be combined advantageously with packet path diversity [9], a transport technique discussed in Section 4. Finally, the source coder can trade-off some compression efficiency for higher error resilience [14]. For live encoding of streaming video, feedback information can be employed to adapt error resiliency, yielding the notion of network-adaptive source coding. Such schemes have been shown to possess superior performance [15]. For pre-compressed video stored on a media server, these network-adaptive source coding techniques can be achieved by assembling sequences of

appropriately precomputed packets on the fly.

In our opinion, the most interesting recent advances in video streaming technology are those that involve several system components jointly and react to the packet loss and delay, thus performing network-adaptive streaming. In this paper, we review some recent advances in network-adaptive streaming. In Section 2 we discuss network-adaptive packet dependency control, which is an encoder/server technique that achieves low latency and robustness. We then review rate-distortion optimized packet scheduling in Section 3 and packet path diversity in Section 4, as the two most important recent advances in transport mechanisms. Finally, as an example of a new receiver-based technique, we discuss adaptive media playout in Section 5, which reduces the delay introduced by the client buffer and provides rate scalability in a small range. Possible directions of future research are discussed at the end in Section 6.

2 Network-adaptive Packet Dependency Control

Video streaming typically exhibits much higher latencies than that of voice transmission over the Internet. This is due to the strong dependencies among video packets caused by interframe prediction. If a packet containing, say, one frame is lost, the decoding of all subsequent frames depending on the lost frame will be affected. Hence, in commercial systems, time is allowed for several retransmissions in order to guarantee error-free reception of each frame, at the cost of higher latency. Advanced source coding schemes are thus desired to reduce latency while maintaining robustness.

Packet dependency control has been recognized as a powerful tool to increase error-robustness, and recently, to address the latency issue. Earlier work on this topic includes long-term memory (LTM) prediction for macroblocks for increased error-resilience [16], the reference picture selection (RPS) mode in H.263+ [17] and the emerging H.26L standard [18], and the video redundancy coding (VRC) technique [19].

In our recent work [20], to increase error-resilience and eliminate the need for retransmission, multiple representations of certain frames are pre-stored at the streaming server. A representation can be chosen in which only frames that will be received with high probability are used as reference frames. By doing so, we dynamically control the dependencies among packets to adapt to the varying channel conditions. With increased error-resilience, the need for retransmission can be eliminated. Because buffering is needed only to absorb the packet delay jitter, buffering delay can be reduced to a few hundred milliseconds.

Due to the trade-off between error-resilience and coding efficiency, we apply Optimal Picture Type Selection (OPTS) within an R-D framework, which considers video content, channel loss probability and channel feedback (e.g., ACK, NACK, or time-out). To code a frame, multiple attempts are made, including Intra-coding as well as Inter-coding using different reference frames in the long-term memory. For a particular attempt, v, the associated rate R_v and expected distortion $\overline{D}_v$ are obtained to calculate the cost through a Lagrangian formulation

$$J_v = \overline{D}_v + \lambda R_v. \tag{1}$$

Here v indicates which of the previous frames is used for prediction, where $v = \infty$ denotes an I-frame. To find the expected distortion in (1), a binary tree model is employed which accounts for the channel loss rate and error propagation [20]. Note that the distortion includes both the quantization error and possible decoding mismatch error, which is calculated at the encoder side. For the Lagrange multiplier λ, we use $\lambda = 5e^{0.1Q}(\frac{5+Q}{34-Q})$, which is the same as λ_{mode} in H.26L TML 8 used to select the optimal prediction mode [18, 21], and Q is the quantization parameter used to trade off rate and distortion. The optimal picture type is selected such that the minimal R-D cost J_v is achieved

$$v_{opt} = \arg\min_{v=1,2,\ldots,V,\infty} J_v, \tag{2}$$

where V is the length of the long-term memory.

This rate-distortion optimization applies to both the compression and streaming of live and pre-encoded video. However, a major challenge for streaming pre-encoded video is the potential mismatch due to the loss of the frame used for prediction. This mismatch error might propagate and lead to severe degradation in performance. Past work addressing this problem includes using S-frames [10], and SP-frames, as proposed for H.26L [11, 18]. However, both are achieved at the cost of a significant bit-rate increase.

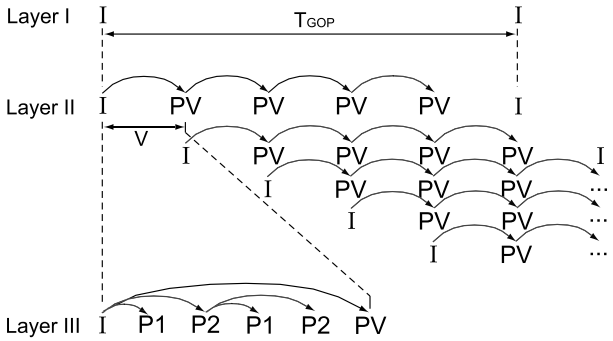

Figure 1: Layered coding structure with coding restrictions. $T_{GOP} = 25, V = 5$.

We use a layered coding structure to avoid the mismatch errors. The coding structure consists of three layers as shown in Fig. 1. *Layer I* contains I-frames that are spaced at intervals of T_{GOP}, the maximum length of a Group of Picture (GOP), which is a multiple of the long term memory length V. They serve as the entry into a GOP. *Layer II* contains only two types of pictures: PV-frames (P frames using $v = V$) and I-frames. They are spaced at intervals of V and serve as the entry into a sub-GOP. Layer I and Layer II pictures are also referred to as *SYNC-frames*, which are only positioned at kV, where $k = 0, 1, 2, \ldots$ (a video sequence starts from 0). The prediction dependency of SYNC-frames in a GOP is shown

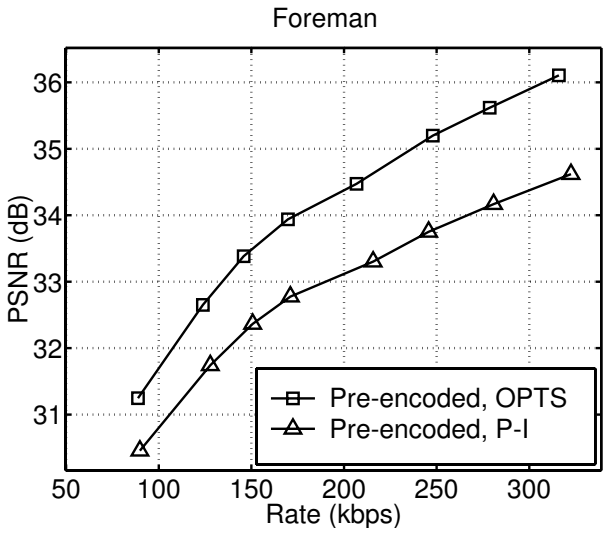

Figure 2: RD performance for *Foreman* sequence, 10% packet loss, 5 frames feedback delay, $V = 5$.

in Fig. 1. T_{GOP}/V versions of SYNC-frames in different phases are pre-stored, with the leading I-frame starting from different positions as shown in Fig. 1. The deployment of the SYNC-frames with multiple phases allows the server to insert an I-frame at *any* SYNC position (kV) to eliminate the mismatch error. The choice between continuing with a PV-frame in a GOP or switching to an I-frame is determined within the OPTS framework considering channel conditions and feedback. Once an I-frame is inserted, it starts a new GOP. *Layer III* pictures are those within a sub-GOP following a SYNC-frame. They can be either P-frames that use a previous frame in the same sub-GOP as reference, or occasionally I-frames - depending on the video content. This limits the prediction dependency within a sub-GOP. Because there are T_{GOP}/V different phases of SYNC-frames, T_{GOP}/V different versions of Layer III pictures must be pre-stored.

Layer III frames are pre-encoded offline based on the video content and the expected channel condition, while SYNC-frames are dynamically assembled during delivery to respond to the channel feedback. The sub-GOP SYNC-frame determines which set of Layer III pictures to send. Pre-stored multiple versions of the bitstreams enable mismatch-free assembly, though achieved at the cost of extra disk storage. The layered coding also provides temporal scalability for rate control and facilitates interactive functions such as random access, fast-forward or fast-reverse.

Simulation results in [20] compare the performance of 1) the proposed network-adaptive OPTS scheme, and 2) a simple scheme that uses normal P-frames with periodic I-frames to combat packet loss (referred to as the *P-I* scheme), where T_{GOP}/V phases of the sequence are also pre-stored, and feedback-induced I-frame insertion at certain positions is allowed.

Fig. 2 shows the R-D performance of sending 230 frames of *Foreman* sequence, respectively, at 30fps over the channels with 10% loss rate. Modified H.26L TML 8.5 [18] is implemented. Distortion at different channel loss rates is shown in Fig. 3 for *Foreman* encoded at approximately the same 200 Kbps using different schemes. The gain in PSNR ranges from 0.5 dB to 1.6 dB, depending on the channel loss rate.

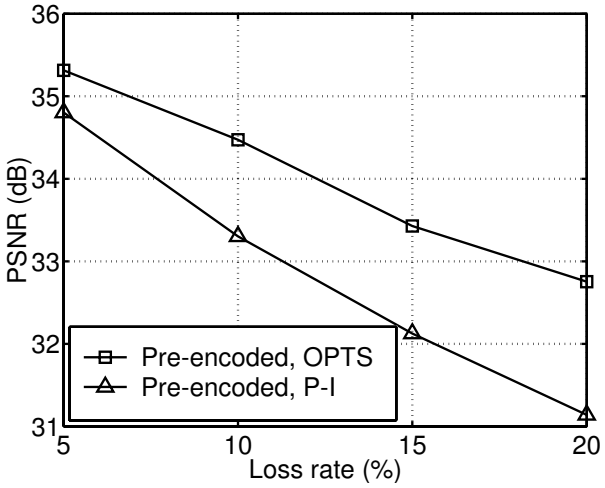

Figure 3: Distortion at different channel loss rates. *Foreman* sequence. The bitrate is 200 Kbps.

Although no retransmission is used, good quality is maintained for typical video sequences sent over lossy channels. Thus the high robustness achievable through packet-dependency control can be used to eliminate the need for retransmission, leading to latencies similar to those for Internet voice transmission.

3 R-D Optimized Packet Scheduling

The second advance that we review in this paper is a transport technique. Because playout buffers are finite, and because there are constraints on allowable instantaneous transmission rates, retransmission attempts for lost packets divert transmission opportunities from subsequent packets and reduce the amount of time that subsequent packets have to successfully cross the channel. A streaming media system must make decisions, therefore, that govern how it will allocate transmission resources among packets.

Recent work of Chou et al. provides a flexible framework to allow Rate-Distortion Optimized (RaDiO) control of packet transmissions [22, 23]. The system can allocate time and bandwidth resources among packets in a way that minimizes a Lagrangian cost function of expected rate and distortion. For example, consider a scenario in which uniformly sized frames of media are placed in individual packets, and one packet is transmitted per discrete transmission interval. A RaDiO streaming system decides which packet to transmit at each opportunity based on the packets' deadlines, their transmission histories, the channel statistics, feedback information, the packets' interdependencies, and the reduction in distortion yielded by each packet if it is successfully received and decoded.

The framework assumes that opportunities to transmit packets occur at discrete intervals in time. At each transmission opportunity, the algorithm determines which

packets to transmit by optimizing its transmission choices for the current opportunity jointly with a complete plan for transmissions it will most likely make in the future. The plan comprises a transmission policy π_l for each packet dictating whether packet l is transmitted or not at each transmission opportunity, depending on feedback. A transmission policy induces an error probability, $\epsilon(\pi_l)$, where an error is defined as the event that a packet does not arrive at the receiver by the playout time. The policy also induces an expected number of times that the packet is transmitted, $\rho(\pi_l)$. The algorithm decides how it will allocate transmissions among the set of packets under consideration by finding the set of corresponding policies which minimize the cost function for those packets $J = D + \lambda R$, where $R = \sum_l \rho(\pi_l) B_l$ is the expected rate that will be incurred where B_l is the size in bytes of packet l, and D is the expected reproduction distortion of the media stream, given the transmission choices contained in the set of policies. D is given by:

$$D = D_0 - \sum_l \Delta D_l \prod_{l' \preceq l} (1 - \epsilon(\pi_{l'})). \qquad (3)$$

D_0 is the reproduction distortion of the group of media frames if no packets arrive, ΔD_l is the expected amount of distortion that is removed if packet l is decodable by its deadline, and the product term $\prod_{l' \preceq l}(1 - \epsilon(\pi_{l'}))$ is the probability that packet l is decodable. $l' \preceq l$ refers to the set of packets l' which must be present to decode packet l. The scheme assumes that loss probabilities for packets are independent, and that expected distortion reductions ΔD_l can be factored independently of the loss probabilities for individual packets.

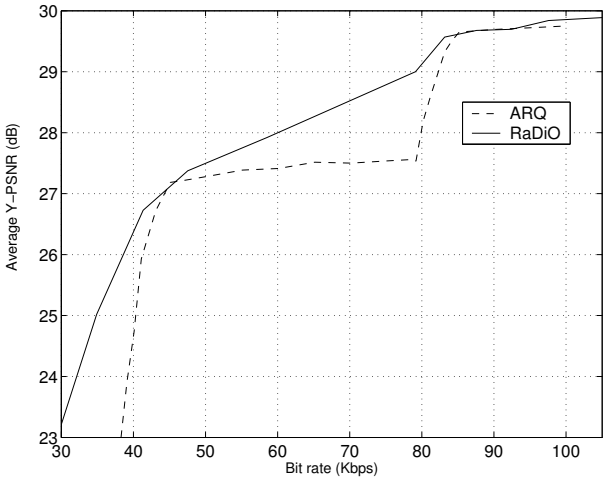

Figure 4: PSNR vs. transmitted bit-rate for a streaming system that uses ARQ and for a streaming system that uses RaDiO transmission scheduling. The results are for an H.263+, SNR scalable encoding of *Foreman.*

Fig. 4 shows the rate-distortion improvement achieved for an H.263+ two-layer SNR scalable encoding of the *Foreman* sequence, with a frame rate of 10 fps, a GOP consisting of one I-frame followed by 9 P-frames, a mean source rate of 32 kbps for base layer alone, and a mean source rate of 64 kbps when combined with the enhancement layer. The results are for a channel model that assumes independent packet losses with loss probability 0.2, and independent, Γ-distributed end-to-end delays with mean delay 50 ms. The traces in the figure plot PSNR as a function of rate for an ARQ system and a RaDiO system. In the ARQ system, the client requests retransmissions for packets that do not arrive by a time interval after they are expected, and the server transmits these requested packets with priority as long as the requested packet may still reach the client in time for playout. When the capacity of the channel falls below the source rate for the enhanced stream, the ARQ system sends only the base layer packets. Both the ARQ and the RaDiO system use an initial pre-roll delay of 400 ms. In Figure 4 we see that by continuously optimizing its packet transmission choices, the optimized system makes use of the SNR and temporal scalability of the source encoding to finely tune the source rate to the available channel capacity, yielding substantial gains.

The framework put forth in [22] is flexible. Using the framework, optimized packet schedules can be computed at the sender [22] or at the receiver [23]. The framework is also applicable to streaming over heterogenous networks, where the last hop to the client is wireless [24]. The authors have also presented simplified methods to compute approximately optimized policies that require low computational complexity. Furthermore, the framework, as shown in [23], appears to be robust against simplifications to the algorithm and approximations to ΔD_l, the information characterizing the value of individual packets with respect to reconstruction distortion. Low complexity is important for server-based implementation, while robustness is important for receiver-based implementations, where the receiver makes decisions.

In a recent work [25], we have extended the RaDiO framework to the scenario of proxy driven streaming. The proxy, located at the edge of the backbone network, coordinates the communication between the media server and the client using a hybrid receiver/sender driven streaming in a RaDiO framework. This improves the end-to-end performance when compared to a sender or receiver driven RaDiO system and in addition relieves the backbone network from the traffic load created by retransmissions of media packets lost in the last hop to the client. The framework enables the proxy to determine at every instant which packets, if any, it should either request from the media server or retransmit directly to the client, in order to meet a constraint on the average transmission rate on the last hop while minimizing the average end-to-end distortion. Figure 5 shows the performance of sender and proxy driven RaDiO systems for streaming SNR and temporal scalable representation of *Foreman*. The expected distortion is measured in PSNR (dB), while the expected rate is measured in Kbps. It can be seen from Figure 5 that the proxy based system outperforms the sender-driven system, with gains reaching up to 1.5 dB.

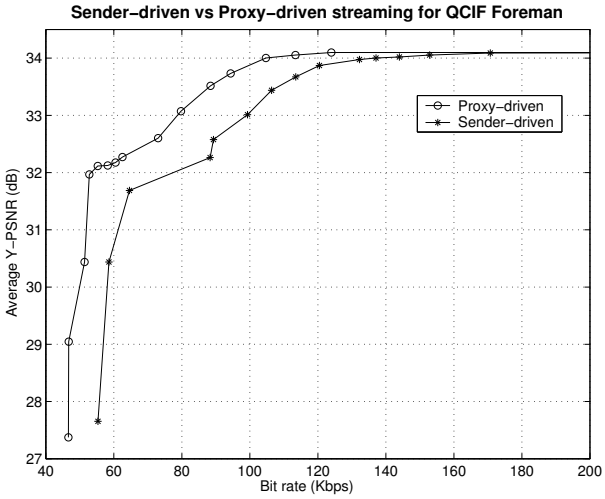

Figure 5: Average Y-PSNR vs. expected rate (Kbps) on the last hop, for sender and proxy driven RaDiO systems. The results are for an H.263+, SNR and temporal scalable encoding of *Foreman.*

4 PACKET PATH DIVERSITY

Diversity techniques have been studied for many years in the context of wireless communication. A number of studies has shown that there is an analogous situation in Internet communication: in 30-80% of the cases there is an alternate path that performs significantly better than the default path between two hosts. Performance is measured in terms of round-trip-time, loss rate and bandwidth. These studies motivate the introduction of packet path diversity in [26], where the authors propose two methods for sending packets through different paths and a system that combines multiple description coding of video with path diversity. The experimental results presented show the potential benefits of the proposed system.

There are a number of important questions that need to be addressed in order to allow an effective use of packet path diversity. How can packets be forced to travel along different paths? How many paths should be used and how should they be chosen? How should the load be shared among the chosen paths, and how to use potential feedback from the receiver to improve communication?

A framework for Internet video streaming from multiple senders simultaneously to a single receiver is proposed in [27] to exploit path diversity and achieve higher robustness. In [9] the authors have exploited the concept of path diversity in the context of multiple description streaming of video in content delivery networks. In [28] multiple description coding (MDC) and layered coding are studied and compared under the multi-path transport environment. Using multiple paths for video transmission over ad-hoc networks is studied in [29], where a new reference picture selection scheme is proposed. In [30] we have combined transmission over multiple paths and R-D optimized reference picture selection to provide improved protection against channel burst errors while keeping high coding efficiency. The R-D optimized reference picture selection we use is constrained in such a way that two almost independent streams are created. In this way, we can stop error propagation and enable an improved state recovery in the case of losses. We combine this encoding with a sending strategy that uses feedback. With the assumption of a two-state Markov channel model we track the burst errors and avoid sending packets over the path that is expected to be in a bad state. This dynamic switching reduces packet loss rate and facilitates state recovery to stop error propagation in case of losses. Overall our algorithm demonstrates significant gain over existing schemes.

Furthermore, we have recently extended the RaDiO framework from Section 3 to streaming over multiple time-varying channels. An additional dimension for packet scheduling is introduced, as media packets might be transmitted over different channels at every transmission opportunity. Here, the sender or the receiver using an instance of the RaDiO framework computes optimal packet schedules considering all the available channels jointly. Packets may be transmitted on multiple channels simultaneously.

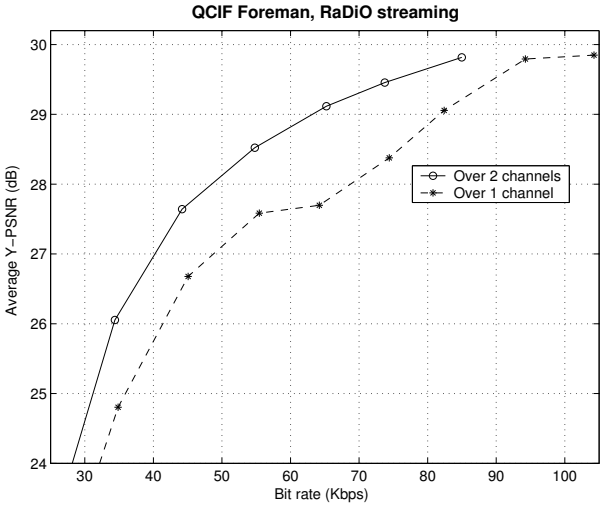

Figure 6: PSNR vs. transmitted bit-rate for RaDiO streming over a single channel and over two channels simultaneously. The results are for an H.263+, SNR scalable encoding of *Foreman.*

In Figure 6 we show experimental results that illustrate the distortion vs. rate performance of RaDiO streaming of video over one channel and over two channels simultaneously. For the two channel case the rate is the overall rate of the two channels. In both cases the initial preroll delay is 600 ms. The *Foreman* sequence used for the experiments from Figure 4 is also employed here. The channels are statistically independent and modeled using a two state discrete-time Markov model. The model is specified in terms of the probability of being in State 2, $\pi_2 = 0.8$, and an expected duration of stay in State 2, $\tau_2 = 2$ s, given that a transition has been made to State 2. Furthermore, the states are characterized with different probability of packet loss, respectively 0.03 and 0.15, and different parameters for the distribution of the transmission delay, in case of no loss. The delay is modeled as

obeying a shifted Gamma distribution, with a shift of 25 ms. In State 1 the mean and variance of the transmission delay are 75 ms and 50 ms respectively, while for State 2 these quantities are 275 ms and 250 ms.

It can be seen from Figure 6 that RaDiO streaming over two channels performs consistently better than RaDiO streaming over a single channel, for the same constraint on the expected transmission rate and given the selected channel parameters. The difference in performance is substantial over the whole range of available transmission rates. A maximum gain close to 1.5 dB is observed for the source encoding of the video used in the experiments.

5 Adaptive Media Playout

Adaptive Media Playout (AMP) is a new technique that can be used to reduce latencies in streaming systems that rely on buffering at the client to protect against random packet losses and delays. In these systems, the client buffers an amount of data before playout begins. While the likelihood of a buffer underflow, an event which causes media playout to halt, decreases as more data is stored, the latency experienced by the user increases. The client must therefore trade off buffer underflow probability and latency.

By giving the client some control over the rate at which its playout process consumes data, AMP allows reduced buffer sizes and latencies for a given buffer underflow probability. With AMP, the client alone controls its data consumption rate by varying the speed at which it plays out media. For video, the client simply adjusts the duration of display of each frame. For audio, the client performs signal processing to scale the signal in time while preserving its pitch [31]. Informal subjective tests have shown that slowing the playout rate of video and audio up to 25% is often un-noticeable, and that time-scale modification is subjectively preferable compared to halting playout or errors due to missing data [31, 32].

The buffering latency most noticeable to the user is pre-roll delay, the time that it takes for the buffer to fill with data and for playout to begin after the user makes a request. In today's commercial streaming products, these delays typically range from 5 to 15 seconds. An AMP-enabled client can reduce pre-roll delays by beginning playout with fewer frames of media stored in the buffer. Using slowed playout, the client can decrease the initial consumption rate so that the buffer occupancy increases over time. When a sufficient amount of media is buffered, playout can continue at normal speed.

Another noticeable latency is the constant buffering delay during live streaming or in two-way communication. By allowing both slowed and faster-than-normal playout, an AMP-enabled client can reduce the mean delay experienced by the user, for a given probability of underflow. Slowed playout, used during bad channel periods to decrease the likelihood of an underflow, causes delay to increase over time. Therefore, during good channel periods the client must play media faster than normal to eliminate any delay that has accumulated. We have found that fast and slow playout allows a mean latency that is smaller than the constant delay that the user would experience in the case of fixed playout speed, for a given probability of underflow. The application was explored for the case of two-way voice communication in [31] and has been extended to video.

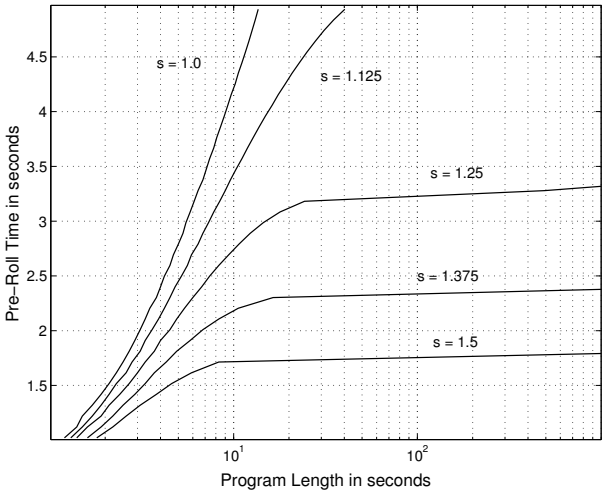

Figure 7: Required pre-roll time as a function of program length for 99% reliability, if frame periods are stretched by factor s whenever buffer level falls below 10 seconds of media. Channel model parameters are $\lambda_G = 1.09$, $\lambda_B = 0.4$, $T_G = 20sec$,and $T_B = 2sec$.

The Markov chain analysis and simulation results shown in Figs. 7 and 8 illustrate the latency reductions achievable with AMP. In both figures, the results are for a model of the packet network that randomly transitions between a good and a bad state. When the channel is in the good state, the mean number of frames that can be delivered to the client per frame period of the media sequence is given by λ_G. In the bad state, the number is given by λ_B. The mean durations of the good and bad states are T_G and T_B. We model the durations of the channel states and frame inter-arrival times as exponentially distributed random variables.

Fig. 7 illustrates pre-roll delay reduction with AMP. It is a plot of the required pre-roll time for a pre-stored media clip of a given length such that the media clip will play without interruption 99% of the time. These results are for an AMP policy which dictates that whenever the client buffer contains fewer than 10 seconds of media, playout frame periods will be stretched by a factor s. We see in the plot that by allowing playout to be slowed, less pre-roll time is needed. For instance, with slowdown factor $s = 1.25$, a 3.5 second pre-roll time will allow reliable streaming of programs longer that 1000 seconds, compared to the 7 second program allowable for the same pre-roll time without AMP.

Fig. 8 illustrates AMP's ability to reduce the mean buffering delay during live streaming. The figure plots Mean Time Between Buffer Underflows (MTBBU) as a function of mean buffering delay. In this case the AMP policy dictates that frame periods are scaled by factor s when the number of frames in the buffer falls below a threshold, and by factor f when the number of frames in

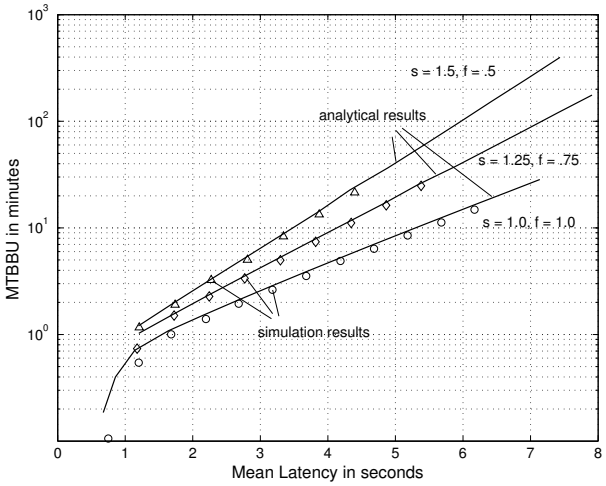

Figure 8: Mean time between buffer underflows as a function of mean latency when frame periods are scaled by factor s when the buffer level is below a threshold and scaled by factor f when the level exceeds the threshold. Channel model paramters are $\lambda_G = 1.33$, $\lambda_B = 0$, $T_G = 28.5sec$,and $T_B = 1.5sec$.

the buffer exceeds the threshold. In the plot we see, for example, that when $s = 1.25$ and $f = 0.75$, a mean latency of 5.25 seconds yields an MTBBU of 25 minutes compared to an MTBBU of 10 minutes for the same delay without adaptation ($s = f = 1.0$).

AMP can also be employed as a form of rate-scalability, allowing clients to access streams which are encoded at a higher source rate than their connections would ordinarily allow [33]. Furthermore, we have recently extended the RaDiO framework of Section 3 to include adaptive media playout, such that each packet is optimally scheduled, along with a recommended individual playout deadline. For that, the distortion measure is extended by a term that penalizes time-scale modification and delay [34].

6 Conclusions

We have discussed several recent results in network-adaptive video streaming. While the reviewed techniques affect the streaming process at different points, their common goal is to improve streaming performance in the presence of network-introduced errors. First, network-adaptive packet dependency control at the source encoder can greatly improve the error-resilience of streaming video and reduce latency. Second, rate-distortion optimized packet scheduling, a transport technique, provides a flexible framework to determine the best packet to send and/or to request given the network behavior, the packets' deadlines, their transmission histories, the distortion reduction associated with sending each packet, and the inter-packet dependencies. Packet path diversity is another transport technique that improves the delivery mechanism of multimedia packets by employing multiple channels over which data may be transmitted. Finally, by controling the rate at which the client consumes data, adaptive media playout can be used to reduce receiver buffering and therefore average latency, and to provide limited rate scalability.

Additional improvement in performance is achieved if some of the reviewed techniques are combined, as discussed and shown throughout the paper. It is our belief that considering in a joint fashion some of the issues that these techniques address separately now, can yield even further benefits. This is certainly one promising direction for future research. Others include issues such as distributed or streaming, contrasting the current sender or receiver driven architecture, and intelligent streaming proxies located at the junction of two or more heterogenous networks.

References

[1] M. R. Civanlar, A. Luthra, S. Wenger, and W. Zhu (eds.), *Special Issue on Streaming Video, IEEE Trans. on Circ. and Syst. for Video Techn.*, vol. 11, no. 3, March 2001.

[2] C. W. Chen, P. Cosman, N. Kingsbury, J. Liang, and J. W. Modestino (eds.), *Spec. Issue on Error Resilient Image and Video Transmission, IEEE Jour. on Sel. Areas in Comm.*, vol. 18, no. 6, June 2001.

[3] Y. Wang, and Q. Zhu, "Error control and concealment for video communication: a review," *Proc. of the IEEE*, vol. 86, no. 5, pp. 974-997, May 1998.

[4] G. J. Conklin, G. S. Greenbaum, K. O. Lillevold, A. F. Lippman, and Y. A. Reznik, "Video coding for streaming media delivery on the Internet," *IEEE Trans. on Circ. and Syst. for Video Techn.* vol. 11, no. 3, pp. 269-281, March 2001.

[5] W. Tan, and A. Zakhor, "Video multicast using layered FEC and scalable compression," *IEEE Trans. on Circ. and Syst. for Video Techn.*, vol. 11, no. 3, pp. 373-387, March 2001.

[6] W. Tan, and A. Zakhor, "Real-time Internet video using error resilient scalable compression and TCP-friendly transport protocol," *IEEE Trans. on Multimedia*, vol. 1, no. 2, pp. 172-186, June 1999.

[7] S. McCanne, M. Vetterli, and V. Jacobson, "Low-complexity video coding for receiver-driven layered multicast," *IEEE Journal on Selected Areas in Communications*, vol. 15, no. 6, pp. 983-1001, August 1997.

[8] J. Shin, J. Kim, and C.-C. J. Kuo, "Quality-of-service mapping mechanism for packet video in differentiated services network," *IEEE Transactions on Multimedia*, vol.3, no.2, pp. 219-231, June 2001.

[9] J. Apostolopoulos, T. Wong, W. Tan, and S. Wee, "On multiple description streaming with content delivery networks," *IEEE Infocom*, July 2002.

[10] N. Färber, and B. Girod, "Robust H.263 compatible video transmission for mobile access to video servers," in *Proc. IEEE Int. Conf. on Image Proc.*, vol. 2, pp. 73-76, Santa Barbara, October 1997.

[11] M. Karczewicz, and R. Kurceren, "A proposal for SP-frames," Proposal to H.26L, Jan. 2001.

[12] M. van der Schaar, and H. Radha, "A hybrid temporal-SNR fine-granular scalability for Internet video," *IEEE Trans. on Circ. and Syst. for Video Techn.*, vol. 11, no. 3, pp. 318-331, March 2001.

[13] Y. Wang, M. Orchard, V. Vaishampayan, and A. R. Reibman, "Multiple description coding using pairwise correlating transforms," to appear in *IEEE Trans. on Image Processing*.

[14] R. Zhang, S. L. Regunathan, and K. Rose, "Video coding with optimal inter/intra-mode switching for packet loss resilience," *IEEE Journal on Sel. Areas in Comm.*, vol. 18, no. 6, pp. 966-976, June 2000.

[15] B. Girod, and N. Färber, "Wireless video," in A. Reibman, M.-T. Sun (eds.), Compressed Video over Networks, Marcel Dekker, 2000.

[16] T. Wiegand, N. Färber, and B. Girod, "Error-resilient video transmission using long-term memory motion-compensated prediction," *IEEE Journal on Sel. Areas in Comm.*, vol. 18, no. 6, pp. 1050–1062, June 2000.

[17] ITU-T Recommendation H.263 Version 2 (H.263+), *Video coding for low bitrate communication*, Jan. 1998.

[18] ITU-T Video Coding Expert Group, *H.26L Test Model Long Term Number 8*, July 2001, online available at ftp://standard.pictel.com/video-site/h26L/tml8.doc.

[19] S. Wenger, G. D. Knorr, J. Ott, and F. Kossentini, "Error resilience support in h.263+,"*IEEE Trans. on Circ. and Syst. for Video Techn.*, vol. 8, no. 7, pp. 867–877, Nov. 1998.

[20] Y. J. Liang and B. Girod, "Rate-distortion optimized low-latency video streaming using channel-adaptive bitstream assembly," in *Proc. IEEE Int. Conf. on Mult. and Expo*, Lausanne, Switzerland, Aug. 2002.

[21] T. Wiegand and B. Girod, "Lagrange multiplier selection in hybrid video coder control," in *Proc. IEEE Int. Conf. on Image Proc.*, vol. 3, pp. 542–545, Thessaloniki, Greece, Oct. 2001.

[22] P. A. Chou and Z Miao, "Rate-distortion optimized streaming of packetized media," *IEEE Trans. on Multimedia*, February 2001. Submitted. http://research.microsoft.com/~pachou.

[23] P. A. Chou and A. Sehgal, "Rate-distortion optimized receiver-driven streaming over best-effort networks," in *Proc. Packet Video Workshop*, Pittsburg, PA, April 2002.

[24] J. Chakareski and P. A. Chou, "Application layer error correction coding for rate-distortion optimized streaming to wireless clients," in *Proc. IEEE Int. Conf. on Acoust., Speech and Sig. Proc.*, Orlando, USA, May 2002.

[25] J. Chakareski, P. A. Chou, and B. Girod, "Rate-distortion optimized streaming from the edge of the network," *IEEE Workshop on Mult. Sig. Proc.*, St. Thomas, US Virgin Islands, Dec. 2002. Submitted.

[26] J. G. Apostolopoulos, "Reliable Video Communication over Lossy Packet Networks using Multiple State Encoding and Path Diversity," in *Proc. SPIE Vis. Comm. and Image Proc.*, pp. 392-409, Jan. 2001.

[27] T. Nguyen and A. Zakhor, "Distributed Video Streaming with Forward Error Correction," in *Proc. Packet Video Workshop*, Pittsburgh, USA, April 2002.

[28] Y. Wang, S.S. Panwar, S. Lin, and S. Mao, "Wireless video transport using path diversity: multiple description vs. layered coding," in *Proc. IEEE Int. Conf. on Image Proc.*, Rochester, USA, Sept. 2002.

[29] S. Lin, S. Mao, Y. Wang, and S. Panwar, "A reference picture selection scheme for video transmission over ad-hoc networks using multiple paths," in *Proc. IEEE Int. Conf. on Mult. and Expo*, Aug. 2001.

[30] Y. J. Liang, E. Setton and B. Girod, "Channel-adaptive video streaming using packet path diversity and rate-distortion optimized reference picture selection," *IEEE Workshop on Mult. Sig. Proc.*, St. Thomas, US Virgin Islands, Dec. 2002. Submitted.

[31] Y. J. Liang, N. Färber, and B. Girod, "Adaptive playout scheduling using time-scale modification in packet voice communication," in *Proc. IEEE Int. Conf. on Acoust., Speech and Sig. Proc.*, Salt Lake City, May 2001.

[32] E. G. Steinbach, N. Färber, and B. Girod, "Adaptive playout for low Latency video streaming," in *Proc. IEEE Int. Conf. on Image Proc.*, Thessaloniki, Greece, Oct. 2001.

[33] M. Kalman, E. Steinbach, and B. Girod, "Adaptive playout for real-time media streaming," in *Proc. IEEE Int. Symp. on Circ. and Syst.*, Scottsdale, AZ, May 2002. Invited Paper.

[34] M. Kalman, E. Steinbach, and B. Girod, "R-D optimized media streaming enhanced with adaptive media playout," in *Proc. Int. Conf. on Mult. and Expo*, Lausanne, Switzerland, Aug. 2002.