USING RATE-DISTORTION METRICS FOR REAL-TIME INTERNET VIDEO STREAMING WITH TCP

Antonios Argyriou

School of Electrical and Computer Engineering
Georgia Institute of Technology
Atlanta, GA 30332, USA

ABSTRACT

In this paper we explore use of a new rate-distortion metric for optimizing real-time Internet video streaming with the transmission control protocol (TCP). The basic idea is to combat packet delays caused by TCP retransmissions that are essentially interpreted as errors by the streaming application. To this aim, we develop an analytical model of the expected video distortion at the decoder with respect to the expected latency for TCP packets, the channel state, and the error concealment method at the receiver. This metric is exploited with the design of a new algorithm for rate-distortion optimized encoding mode selection for video streaming with TCP (RDOMS-TCP). Real-time video streaming experiments show considerable improvement in PSNR in the range of 2 db over currently proposed TCP-based streaming mechanisms.

1. INTRODUCTION

In today's Internet the TCP protocol that dominates the existing traffic, is considered unsuitable for video streaming applications while its counterpart UDP is usually the protocol of choice. The main reasons for TCP's unusability, are the rapid throughput fluctuations and the reliability mechanism which incurs additional delays for a video bitstream [1]. Despite these problems, the majority of commercial IP-based video streaming systems unexpectedly so employ TCP for transport layer services of encoded video content. In addition, the widespread use of TCP has stimulated research for the development of new mechanisms that facilitate video streaming with TCP. For example in [2], the authors evaluate multimedia streaming with TCP, and conclude that buffering at the client can handle the retransmission delays and the congestion control induced throughput variations of TCP. Another well known approach reported in [3], attempts to provide an approximate CBR channel to the streaming flow that is using TCP, through prioritization over other flows at the last mile connection of the receiver. We must also report a new receiver-driven technique for video streaming, that introduced the idea of receiver-based delay control (RDC) [4], in which receivers delay TCP acknowledgments based on feedback from routers. The most recent work we are aware of, has been reported at [5] in which the authors present an analytical video streaming model for TCP.

In summary, one common characteristic of the aforementioned mechanisms is that they consider modifications and enhancements to TCP or the infrastructure, while they ignore the nature of the content which is a video bitstream. Methods that do consider the nature of video data are source coding network adaptive algorithms [1]. For example an interesting approach is rate-distortion optimization (RDO), which is based on the tradeoffs between quality and bitstream size. Other approaches use RD metrics in order to select the optimal encoding mode at the sender [6, 7]. The authors estimate the end-to-end video distortion for several streaming scenarios and propose heuristics or selecting the optimal encoding mode that minimizes the receiver distortion. Several more sophisticated algorithms have been developed since then.

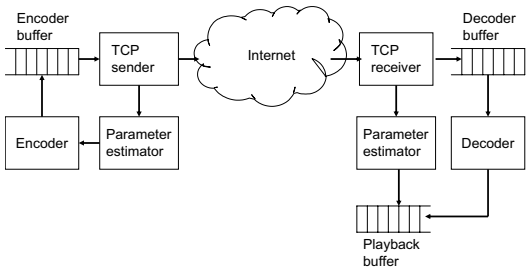

Fig. 1. Proposed media streaming architecture based on TCP.

While all above RD-based algorithms are fairly sophisticated, they ignore one important parameter which is the behavior of the transport protocol. Therefore, two are the goals that we set to achieve with this paper: 1) Derive analytically the expected distortion for a video bistream at the decoder when TCP is used for transport, and 2) select optimal encoding mode for each individual macroblock at the encoder.

2. THE REAL-TIME STREAMING SYSTEM

Figure 1 depicts a simplified block diagram of the end-to-end real-time video streaming system. According to this figure the sender uses an H.263++ real-time encoder that generates the bitstream which is placed in the encoder buffer. Rate control is applied at the encoder according to the TMN8 model, while the TCP protocol is polled in order to obtain the instantaneous available channel rate R_c. While video playback starts, the server continues to send new data to the end of the client playback buffer, while the decoder keeps consuming the data available at the start of the buffer. The network is assumed to generate packet loss according to a two-state Markov chain (the Gilbert-Elliot path model).

While the above sequence of events is typical a client-server type of streaming application, a set of extra steps are performed by our system. Throughout the streaming session, the streaming server estimates in real-time the end-to-end distortion, as a function of the distortion of the individual macroblocks, the expected end-to-end latency, the error concealment at the decoder, and the state of the transport channel. Performing this task timely and accurately, is one of the primary concerns of this work. According to the algorithm that we will present, the real-time encoder selects an optimal encoding mode for each individual macroblock based on the RD metric that will be calculated.

2.1. TCP Latency

A latency model is needed since we want to explore the rate constraint imposed by the TCP dynamics. Essentially we want to isolate those packets that will violate their playback deadlines and can therefore be characterized as lost for the decoder. This statement can be formally written as follows:

$$\begin{aligned} P_D &= P\{t_s + L \geq t_d\} = 1 - P\{L < \Delta t\} \\ &= 1 - \int_0^{+\infty} F_L(\Delta t - y) f_L(y) dy \end{aligned} \quad (1)$$

where t_s is the time instant the packet was sent while t_d is the playback deadline for this packet. The interesting point here is that because of the TCP's retransmission mechanism we can assume without loosing generality, that after three retransmissions lost packets will be received with probability 1 at the receiver. But these packets will be probably late for their playback as Eq. 1 shows. This means that for TCP, we want essentially to find only one quantity and this the probability of delayed packets.

Instead of re-inventing the wheel, we use a well known latency model for TCP-Reno where the effect of TCP's slow start and congestion avoidance phases is taken into consideration [8]. The total latency to transfer a chunk of d bytes is given as:

$$E[d] = E(T_{HS}) + E(T_{SS}) + E(T_{loss}) + E(T_{delack}) + E(T_{CA})$$

Here, $E(T_{HS})$ represents the latency of the three-way handshake, $E(T_{SS})$ is the expected time spent is slow start, $E(T_{loss})$ is the expected time until the first loss, $E(T_{delack})$ is the expected latency due to the delayed acknowledgements, and finally $E(T_{CA})$ denotes the expected time spent in congestion avoidance. With the help of the last equation, we can calculate Eq. 1, since we assume that no playback buffering is used making thus the value Δt constant.

To model the delivery of packets at the decoder, we adopt a two-state Markov chain where the two states are good and delayed. If we want to capture the probability to transition from one state to another after n sent IP packets, we can rewrite the state transition matrix as:

$$A^n = \begin{pmatrix} \pi_{DD}^n & \pi_{DR}^n \\ \pi_{RD}^n & \pi_{RR}^n \end{pmatrix} \quad (2)$$

where the notation π_{xy}, symbolizes the transition from state x to y. The transition probabilities are calculated using maximum likelihood estimators. Fore example, $\hat{\pi}_{RD} = \frac{n_{RD}}{n_R}$, where n_{RD} is the number of times that a packet was classified as D following a packet classified as R, and n_{DR} is the opposite. Note that for the receiver, it is relatively easy to identify and classify a packet into one of the three aforementioned states, since it has knowledge of whether the packet was delayed or not. Therefore, the probability for a packet to be delayed (i.e lost) after n packets were sent, and were received, is equal to: $P_D = \frac{\pi_{RD}}{\pi_{RD}+\pi_{DR}}$

3. EXPECTED DECODER DISTORTION

Let us know calculate the accurate per pixel distortion at the decoder, and see how can we use the result that we derived in the previous section. Let M_i^n be the coded macroblock at location i of frame n, and let also $M_i^n \in X_k$ symbolize the fact that a coded macroblock is contained in network TCP/IP packet X_k. Let also f denote the pixel value at the encoder, and $\tilde{f}$ the reconstructed value at the decoder, and $\hat{f}$ the encoder estimation of the reconstructed pixel value at the decoder. If we denote as η_i^n as the TCP packet that contains MB M_i^n, and as K the number of packets that packetize the first I frame of the series, then the value $\eta_i^n - K - 1$, will give the number of state transitions (Eq. 2) that happened until we reach packet η_i^n. Therefore, the probabilities for M_i^n to be received will be given as $\tilde{P}_R^{(i,n)} = \pi_{RR}^{\eta_i^n - K - 1}$, while the probability to be delayed by $\tilde{P}_D^{(i,n)} = \pi_{RL}^{\eta_i^n - K - 1}$. Essentially, this means that the probability to receive M_i^n correctly, equals to the probability of receiving successively $\eta_i^n - K - 1$ packets.

We also want to be able to account for a simple error-concealment method that is employed at the decoder, which substitutes the lost M_i^n with M_i^{n-1}. Therefore we have to calculate the probabilities that M_i^n was delayed given that M_i^{n-1} was received ($\tilde{P}_{RD}$), and the event that were both de-

layed ($\tilde{P}_{DD}$). For $\tilde{P}_{RD}$ we have:

$$\tilde{P}_{RD} = \begin{cases} 0 & \text{if } n = 0 \\ \pi_{RR}^{(\eta_i^{n-1}-K+1)} \pi_{RD}^{(\eta_i^n - \eta_i^{n-1})} & \text{if } n > 0 \end{cases} \quad (3)$$

The above equations can be interpreted as follows: The probability to loose M_i^n, precisely after a successful MB delivery is given as the probability of receiving the M_i^{n-1} correctly with probability $\pi_{RR}^{(\eta_i^{n-1}-K+1)}$, times the probability to have another chain of RRRRL... events which depends on the id of the packets used for M_i^n and M_i^{n-1}, and their relative distance ($\eta_i^n - \eta_i^{n-1}$) in the network packet.

The next step is the calculation of the expected value of a reconstructed pixel at the receiver. The following two equations show the expected pixel values for intra-coded and inter-coded macroblocks respectively:

$$E[\hat{f}_{ij}^n] = \tilde{\pi}_R^{(i,n)} \tilde{f}_{ij}^n + \hat{\pi}_{RD}^{(i,n)} \hat{f}_{ij}^{n-1} + \tilde{\pi}_{DD}^{(i,n)} \hat{f}_{ml}^n$$

$$E[\hat{f}_{ij}^n] = \tilde{\pi}_R^{(i,n)} (\tilde{e}_{ij}^n + \tilde{f}_{uv}^{n-1}) + \tilde{\pi}_{RD}^{(i,n)} \hat{f}_{ij}^{n-1} + \tilde{\pi}_{DD}^{(i,n)} \hat{f}_{ml}^n$$

The pixel value, will be equal to the reconstructed value at the encoder times the probability to receive the MB correctly $\tilde{\pi}_R^{(i,n)} \tilde{f}_{ij}^n$, plus the probability to loose this MB and so use the reconstructed value of the same pixel of the previous frame $E[\hat{f}_{ij}^{n-1}]$ (error concealment is used). We have to add the probability that both the previous and the current MB are lost and another pixel from the current frame ($E[\hat{f}_{ml}^{n-1}]$) is used. The same logic is used for the second equation with the one difference that the value the IDCT residue ($\tilde{e}_{ij}^n$) must added to the reconstructed pixel values.

After calculating the expected pixel values, the distortion will be given as the mean absolute differences (MAD) of the pixel values in frame n, for either Intra (I) or Inter (IR) coded MBs:

$$MAD(M_i^n) = \frac{\sum_{j=1}^{256} |f_{ij}^n - E[\hat{f}_{ij}^n]|}{256} \quad (4)$$

3.1. Encoding Mode Selection

The question that rises now is how to utilize this analytical model of the expected decoder distortion. What we claim is that the more accurate distortion estimate at the encoder, can lead to a better allocation of the available channel rate through selection of the encoding mode of each individual macroblock. Even though this principle has been demonstrated before [6], in this paper we are the first to consider the effect of a transport protocol in the behavior of the real-time encoder. During the streaming session, one of the first tasks of the encoder is to allocate a bit budget to the next frame waiting to be encoded. This task called *rate control*, is performed by a simple algorithm in H.263. Our task is to allocate the bit budget per frame to each of the macroblocks that are about to be encoded, by selecting an intra or predictive encoding mode.

To formalize this problem, consider a group of m macroblocks that belong to frame n, i.e. $\Phi^n = (M^n, ..., M_m^n)$. Also consider the encoding vector for these macroblocks $\Theta^n = (\theta^n, ..., \theta_m^n)$, where $\theta \in \{intra, inter\}$. If there is a number of F frames waiting to be encoded, with $\Phi = (f, ..., f_F)$, the objective is to:

$$\min E[D(\Phi, \Theta)] \quad \text{such that} \quad R(\Phi) \leq R_c \quad (5)$$

R_c is the current rate constraint imposed by TCP. The expected distortion for a macroblock $s \in (1...m)$, that belongs to frame n will be $D[M_s^n] = MAD(M_s^n)$, that was derived in Eq. 4. Now, the objective of the Lagrangian relaxation problem we define, is to select the optimal vector θ^* so that the overall distortion is minimized. The Lagrangian optimization problem is therefore formally defined for all the macroblocks up to be encoded:

$$\min \sum_{i=1}^{mn} J_k = \sum_{i=1}^{mn} E[D(\Phi, \Theta)] + \lambda \sum_{i=1}^{mn} R \quad (6)$$

The selection of the optimal Lagrange multiplier λ, can be selected using several alternatives. We followed the same approach as in [7], in order to simplify comparison. So for a frame n this parameter is set as:

$$\lambda_n = \frac{2B_n + (g - B_n)}{B_n + (g - B_n)} \lambda_{n-1} \quad (7)$$

where B_n denotes the current occupancy of the encoder buffer.

4. EXPERIMENTS

For the experiments in this paper we used a client server configuration Both the sender and the receiver are linux boxes while the middlebox is a freeBSD machine that acts as a router. The Dummynet software was used in the middlebox in order to emulate various link configurations in terms of packet loss rate, bandwidth and delay. The QCIF FOREMAN and AKIYO sequences, were used for real-time encoding with H.263 encoder at various bitrates for the streaming experiments. The video units were packetized into RTP packets and the sent to TCP. Due to the short duration of the sequences (150 frames), they were repeated and fed as input to the encoder. The capacity of the bottleneck link between the two routers is set to 250Kbps and delay at 10ms . The results were obtained by running the same scenario 100 times and averaging the PSNR values of the same experiments.

Figure 2 illustrates the main benefit of our algorithm. It depicts PSNR at the decoder as a function of the frame number. When we used the RD adaptive algorithm ROPE, that does not consider TCP, a more generous encoding rate was produced leading to a slightly higher PSNR when no packet loss takes place. While frame number 60 was being encoded,

packet drops were caused by congestion, and TCP retransmissions followed. In this case with RDOMS-TCP, the short-term future latency is expected to be high (due to congestion) even if no packet loss is observed, increasing thus the probability of delayed packets. However, with TCP-ROPE the algorithm considers the bad channel state but for few dropped packets (we use TCP). Therefore the encoder does not scale the quantizer as much as it should do based on the expected latency but simply tries to match the available TCP rate. However, this results into more delayed packets for the subsequent frames which means low decoded PSNR.

Figure 3 presents PSNR as a function of the channel packet loss probability for real-time target encoding rate of 256Kbps and 64Kbps respectively. We compare our results with a mechanism that implements an RD optimal mode selection policy for streaming with UDP [7], and the methodology reported at [3], that also considers streaming with TCP. We see that when the target bitrate was 256Kbps, the RDOMS/UDP approach outperforms both the other two. However, the benefit of the proposed RDOMS algorithm, come into place when TCP is used for transport. It clearly outperforms by 2-2.5 db, a purely TCP based streaming approach, which does not consider the protocol behavior. More important, for higher packet loss rate, the performance in terms of PSNR is increasing. When the target bitrate was set to 64Kbps, PSNR presents the same trend, but this time the effect of the algorithms is not so significant due to the low bitrate injected to the network.

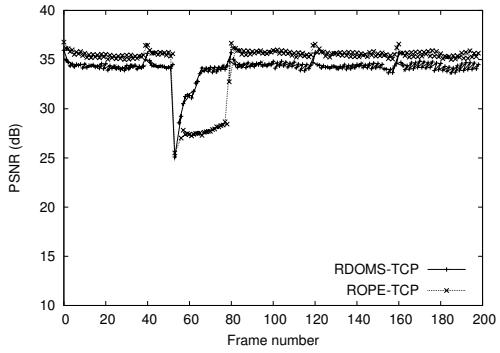

Fig. 2. PSNR as a function of the frame number at the decoder. Target encoding rate is 512Kbps.

5. CONCLUSIONS

With this paper we wanted to demonstrate a novel mechanism for improving performance of TCP video streaming based on the use of an RD metric. We initially developed an analytical model of the expected video distortion at the decoder with respect to the TCP latency, the channel state, and a simple error concealment method at the receiver. Based on this model we proposed an algorithm for RD optimized mode selection

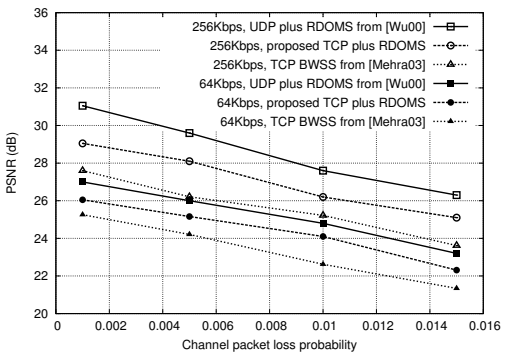

Fig. 3. PSNR as a function of the end-to-end packet loss probability at the decoder.

(RDOMS) for video streaming with TCP. Experimental results for real-time video streaming showed PSNR improvement of nearly two db over currently proposed TCP-based streaming mechanisms.

6. REFERENCES

[1] Yao Wang, Jrn Ostermann, and Ya-Qin Zhang, *Video Processing and Communications*, Prentice Hall, 2002.

[2] C. Krasic, K. Li, and J. Walpole, "The Case for Streaming Multimedia with TCP," in *Workshop on Interactive Distributed Multimedia Sytems (IDMS)*, 2001.

[3] Puneet Mehra and Avideh Zakhor, "TCP-Based Video Streaming Using Receiver-Driven Bandwidth Sharing," in *International Packet Video Workshop*, 2003.

[4] Pai-Hsiang Hsiao, H.T. Kung, and Koan-Sin Tan, "Video over TCP with Receiver-based Delay Control," in *ACM NOSSDAV*, 2001.

[5] Bing Wang, Jim Kurose, Prashant Shenoy, and Don Towsley, "Multimedia Streaming via TCP: An Analytic Performance Study," in *ACM Multimedia*, 2004.

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

[7] Dapeng Wu et al., "An End-to-End Approach for Optimal Mode Selection in Internet Video Communication: Theory and Application," *IEEE JSAC*, vol. 18, no. 6, pp. 977–995, June 2000.

[8] Neal Cardwell et al., "Modeling TCP Latency," in *INFOCOM*, 2000.