Quality-Driven TCP Friendly Rate Control for Real-Time Video Streaming

Haiyan Luo, Dalei Wu, Song Ci
Department of Computer and Electronics Engineering
University of Nebraska-Lincoln
Email: {hluo, dwu}@bigred.unl.edu, sci2@unl.edu

Antonios Argyriou
Philips Research
Eindhoven, The Netherlands
Email: antonios.argyriou@philips.com

Haohong Wang
Marvell Semiconductors
Santa Clara, CA 95054
Email: haohongw@marvell.com

***Abstract*—TCP Friendly Rate Control (TFRC) has been designed to provide smoother sending rate than TCP for multimedia applications. However, most existing work on TFRC is restricted within exploring the performance of TFRC itself in wired or wireless networks without considering the interaction between TFRC and other network layers. This paper proposes a quality-driven TFRC framework for real-time video streaming, where real-time video coding at the application layer and the packet sending rate at the transport layer are jointly optimized. The proposed framework is formulated to find the optimal video coding parameters and the sending rate to minimize the end-to-end expected video distortion under a given video playback delay constraint. The proposed framework has been implemented and tested by using both H.264/AVC codec and NS-2 simulator. Experimental results demonstrate that the proposed joint optimization framework can significantly improve the received video quality over the existing schemes, especially when delay bound is tight.**

I. Introduction

As the current dominant transport protocols, both UDP and TCP have their drawbacks for video applications. UDP, though simple and efficient, lacks the mechanism to guarantee delivery. Thus, it is up to the receiving application to detect loss or corruption and recover data using error correction techniques. If data is lost, the stream may suffer a dropout. Furthermore, UDP streams are often more difficult to penetrate firewalls. On the other hand, TCP, whose transmission rate is controlled by a congestion window which is halved for every window of data containing a packet drop and increased by roughly one packet per window of data otherwise, is not well-suited for real-time streaming applications, due to its rate variability and additional delays introduced by reliability mechanisms, although it is efficient for bulk data transfer. Nevertheless, congestion control, typically in the form of rate-based congestion control for streaming applications, is still of primary importance for multimedia applications to deal with the diverse and constantly changing conditions of the Internet [1]. In essence, the goal of congestion control algorithms is to prevent applications from either overloading or under-utilizing the available network resources.

TCP-Friendly protocols emerge as a good tradeoff between TCP and UDP, which not only boast slow responsiveness in order to smooth data throughput, but also are TCP-friendly. The definition of "TCP-friendliness" is that a traffic flow, in steady state, uses no more bandwidth in the long term than a conforming TCP flow would use under comparable conditions [2], thus avoiding the starvation or even congestion collapse to TCP traffic when the two types of traffic coexist. In other words, it is characterized by the same behavior with TCP in the long run, but allows smoother throughput fluctuations. As one of the most popular TCP-Friendly protocols, TFRC is an equation-based congestion control mechanism that uses the TCP throughput function derived in [1] to calculate the actual available rate for a media stream.

Since it was formally introduced in [3] and standardized in [8], there has a lot of research [2]–[7] done on TFRC, due to its well-known advantages and bright prospects. However, most of these research activities only focus on the performance of TFRC per se or the comparison with its transport layer peer - TCP. Some work on video streaming using TFRC only focuses on the TFRC performance with pre-encoded video streams as stated in [9], [10]. This paper is the first work to jointly integrate TFRC with real-time video encoding in a cross-layer optimization approach. Our solution aims at setting up a bidirectional interaction between source online encoding and network TFRC. For example, if there is a heavy traffic in the network such that rate control cannot be effectively performed. This information can be used by the video encoder to adapt its coding parameters and decrease the coding rate to alleviate the network congestion, while maintaining a possibly best received video quality at the receiver side. Similarly, in a video sequence to be transmitted, some video clips or video frames with large content varying due to high video mobility might have to be represented by a large amount of data after compressed. The rate control at the transport layer can use this information to increase its sending rate to avoid a large packet loss rate due to possible queue overflow or network congestion.

The proposed framework is formulated as to select the optimal video coding parameters and the transport layer sending rate to minimize the average video distortion with a predetermined delay bound. To provide a smooth video playback experience, the video delay bound refers to the video frame delay deadline. Different video frames are associated

978-1-4244-2324-8/08/$25.00 © 2008 IEEE.

with different delay deadlines, which are determined by both the video decoding and display settings at the application layer. To solve the above formulated problem and implement the proposed framework, we design a controller at the source node as shown in [11]. The responsibility of the controller is to: 1) communicate with each layer and obtain the corresponding information. For example, it retrieves the expected video distortion from the encoder and the network conditions from lower layers; 2) perform optimization and dynamically determine optimal values of the corresponding parameters of each layer; and 3) pass the determined values back to each layer. For real-time source coding, the estimated distortion (caused by quantization, packet loss, and error concealment) at the encoder is calculated by the "recursive optimal per-pixel estimate" (ROPE) method [12], which provides an accurate video-quality-based optimization metric to the controller. The proposed framework is implemented and tested on H.264/AVC JM 12.2 [13]. Compared to peer work, the novelty of this work is that *this is the first quality-driven cross-layer optimization framework integrating TFRC with real-time video encoding.*

The remainder of this paper is organized as follows. In Section II, we present the system model and the network specification in our framework. In Section III, we elaborate the problem formulation, followed by the proposed solution using dynamic programming. The experiment implementation details and the corresponding simulation results are presented in Section IV. Finally, Section V concludes the paper.

II. Problem Statement

A. System Model

The system model of the proposed framework is shown in Figure 1, which consists of a 3-layer structure and a controller. The controller is the core of the proposed framework, which is equipped with all possible values of the key parameters of each layer. With the feedback information from the network such as RTT, queue length, packet loss rate, etc., the controller performs optimization and chooses the optimal set of parameter values in a cross-layer approach, to achieve the best video distortion/delay performance.

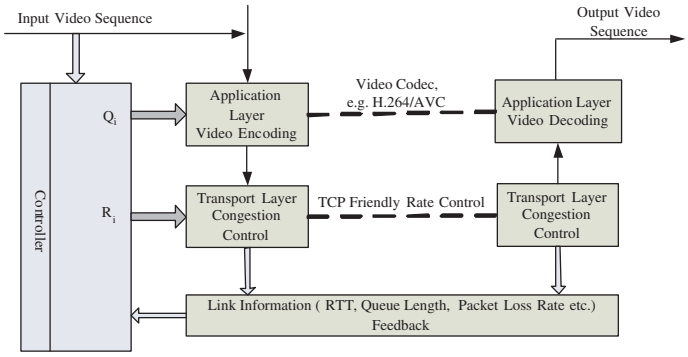

Fig. 1. The system model for optimized real-time video communications based on TFRC.

Clearly, the tradeoff among the selected parameters in different layers are mixed. For instance, to maintain a reasonable delay, the source coding might choose a coarser parameter, which increases the vulnerability of coded frames and will cause unacceptable video quality. However, if the sender using larger sending rate, retransmission trials might increase which will result in excessive delay. Therefore, cross-layer optimization approach is a natural solution to improve the overall performance.

B. Network Specification

1) The Application Layer: In video coding, the rate-distortion rule indicates that setting a finer coding parameter will directly improve the coded video quality. However, the resulting large data rate might increase transmission delay, and increase the chance of getting corrupted by transmission error and data loss due to network congestion. In our framework, we consider the video coding parameters Q of each video unit. A video unit can be a video frame, a video slice or a macroblock (MB), which depends on specific complexity requirement. Q can be the quantization step size (QP), the intra/inter prediction mode, or their combinations. Such coding parameters have significant impacts on the rate-distortion performance of video coding systems.

For a N-frame video sequence $\{f_1, f_2, \ldots, f_N\}$ to be encoded and transmitted from the source node to the destination node. In recent video coding standards, such as H.264 [13], a slice could either be as small as a group of macroblocks (MBs), or as large as an entire video frame. Each slice header acts as a resynchronization marker, which allows the slices to be independently decodable, and to be transported out of order, and still be decoded correctly at the decoder. Let I_n be the total number of slices in video frame f_n, and $J_{n,i}$ the total number of pixels in the ith slice of frame f_n. Let $\xi_{n,i}$ denote the ith slice of frame f_n, $f_{n,i}^j$ the original value of pixel j in the slice $\xi_{n,i}$, $\hat{f}_{n,i}^j$ the corresponding encoder reconstructed pixel value, $\tilde{f}_{n,i}^j$ the corresponding reconstructed pixel value at the decoder, and $E[d_{n,i}^j]$ the expected distortion at the receiver of pixel j in slice $\xi_{n,i}$. As in most existing literature [12], we use the expected mean-squared error (MSE) as distortion metric. Then the total expected distortion $E[D]$ for the entire video sequence can be calculated by summing up the expected distortion over all the pixels

$$E[D] = \sum_{n=1}^{N} \sum_{i=1}^{I_n} \sum_{j=1}^{J_{n,i}} E[d_{n,i}^j] \tag{1}$$

where

$$\begin{aligned} E[d_{n,i}^j] &= E[(f_{n,i}^j - \tilde{f}_{n,i}^j)^2] \\ &= (f_{n,i}^j)^2 - 2f_{n,i}^j E[\tilde{f}_{n,i}^j] + E[(\tilde{f}_{n,i}^j)^2]. \end{aligned} \tag{2}$$

Since $\tilde{f}_{n,i}^j$ is unknown to the encoder, it can be thought as a random variable. To compute $E[d_{n,i}^j]$, the first and second moments of $\tilde{f}_{n,i}^j$ need to be calculated based on the used packetization scheme and error concealment scheme, and the feedback information on the end-to-end packet loss rate. The

978-1-4244-2324-8/08/$25.00 © 2008 IEEE.
This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE "GLOBECOM" 2008 proceedings.

detailed calculation of the first and second moments can be found in [12], [14].

2) The Transport Layer: At the transport layer, we use TFRC mechanism to avoid the drawbacks of using TCP and to provide smooth throughput for real-time video applications. TFRC is designed for applications that use a fixed packet size, and vary their sending rate in packets per second in response to congestion [8]. In TFRC, the average sending rate is given by the following equation:

$$R_{TFRC} = \frac{s}{RTT\sqrt{\frac{2\rho}{3}} + t_{RTO}(3\sqrt{\frac{3\rho}{8}})\rho(1+32\rho^2)} \quad (3)$$

where s is the packet size in bytes, RTT is the round-trip time estimate, t_{RTO} is the value of the retransmission timer, ρ is the packet loss rate and R_{TFRC} is the transmit rate in bytes/second.

In our framework, we propose to optimize the sending rate from a set of optional rates. For each given sending rate, we can derive an inverse function of the equation (3) for ρ:

$$\rho = \overline{f}(RTT, R_{TFRC}) \quad (4)$$

Therefore, the equivalent TFRC packet loss rate ρ can be estimated using the inverse function with the measured RTT. The transmission delay of slice $\xi_{n,i}$ can be expressed as

$$T_{n,i} = \left\lceil \frac{L_{n,i}}{s} \right\rceil \frac{s}{R_{TFRC}} \quad (5)$$

where $\lceil \frac{L_{n,i}}{s} \rceil$ is the number of the s-byte packets into which the total $L_{n,i}$ bytes of the slice $\xi_{n,i}$ are fragmented.

III. Problem Formulation and Solution Procedure

A. Problem Formulation

In this work, we jointly optimize the source coding parameter and the sending rate within a delay-distortion framework. The goal is to minimize the perceived video distortion within the given slice delay bound. To provide a smooth video display experience to users, each frame f_n is associated with a frame delay deadline $T_n^{\max}$. Thus, all the slices of video frame f_n have the same delay constraint $T_n^{\max}$. Let $\vec{Q}_{n,i}$ and $\vec{R}_{n,i}$ be the source coding parameter and the TFRC sending rate for slice $\xi_{n,i}$. Therefore, the problem at hand can be formulated as

$$\min_{\{\vec{Q}_{n,i}, \vec{R}_{n,i}\}} \sum_{n=1}^{N} \sum_{i=1}^{I_n} E[D_{n,i}]$$
$$s.t. : Max\{T_{n,1}, T_{n,2}, ...T_{n,I_n}\} \leq T_n^{max}. \quad (6)$$

B. Solution Procedure

For simplicity, let us denote by $\mathbf{v_w} = \{Q_{n,i}, R_{n,i}\}$ the parameter vector of slice $\xi_{n,i}$, where $1 \leq w \leq N \times I$ is the index of the slice $\xi_{n,i}$ over the whole video clip. According to (6), any selected parameter vector $\mathbf{v_w}$ which results in a single-slice delay larger than the constraint T_n^{max} can not belong to the optimal parameter vector $\mathbf{v_w}^* = \{Q_{n,i}^*, R_{n,i}^*\}$. Therefore, we can make use of this fact by redefining the distortion as follows:

$$E[D'_{n,i}] = \begin{cases} \infty : & T_{n,i} > T_n^{max} \\ E[D_{n,i}] : & T_{n,i} \leq T_n^{max} \end{cases} \quad (7)$$

In other words, the average distortion of a slice with a delay larger than the delay bound is set to infinity. This means that, if a feasible solution exists, the parameter vector which minimizes the average total distortion, as defined in (6), will not result in any slice delay larger than T_n^{max}. Therefore, the minimum distortion problem can be transformed into an unconstrained optimization problem using the above redefinition of the single-slice distortion.

Most decoder concealment strategies introduce dependencies between slices. For example, if the concealment algorithm uses the motion vector of the previous MB to conceal the lost MB, then it would cause the calculation of the expected distortion of the current slice to depend on its previous slices. Without losing the generality, we assume that due to the concealment strategy, the current slice will depend on its previous a slices ($a \geq 0$). To solve the optimization problem, we define a cost function $G_k(\mathbf{v_{k-a}}, ...\mathbf{v_k})$, which represents the minimum average distortion up to and including the kth slice, given that $\mathbf{v_{k-a}}, ..., \mathbf{v_k}$ are decision vectors for the $(k-a)$th to kth slices. Let $\mathbf{O}$ be the total slice number of the video sequence, and we have $\mathbf{O} = N \times I$. Therefore, $G_{\mathbf{O}}(\mathbf{v_{O-a}}, ..., \mathbf{v_O})$ represents the minimum total distortion for all the slices of the whole video sequence. Thus, solving (6) is equivalent to solve

$$\min_{\mathbf{v_{O-a}}, ..., \mathbf{v_O}} G_{\mathbf{O}}(\mathbf{v_{O-a}}, ..., \mathbf{v_O}) \quad (8)$$

The key observation for deriving an efficient algorithm is the fact that given $a+1$ decision vectors $\mathbf{v_{k-a-1}}, ...\mathbf{v_{k-1}}$ for the $(k-a-1)$th to $(k-1)$th slices, and the cost function $G_{k-1}(\mathbf{v_{k-a-1}}, ..., \mathbf{v_{k-1}})$, the selection of the next decision vector $\mathbf{v_k}$ is independent of the selection of the previous decision vectors $\mathbf{v_1}, \mathbf{v_2}, ..., \mathbf{v_{k-a-2}}$. This means that the cost function can be expressed recursively as

$$G_k(\mathbf{v_{k-a}}, .., \mathbf{v_k}) =$$
$$\min_{\mathbf{v_{k-a-1}}, ..., \mathbf{v_{k-1}}} \{G_{k-1}(\mathbf{v_{k-a-1}}, ..., \mathbf{v_{k-1}}) + E[D_k]\} \quad (9)$$

The recursive representation of the cost function above makes the future step of the optimization process independent from its past step, which is the foundation of dynamic programming.

The problem can be converted into a graph theory problem of finding the shortest path in a directed acyclic graph (DAG) [15]. The computational complexity of the algorithm is $\mathbf{O}(\mathbf{O} \times |\mathbf{v}|^{a+1})$ (where $|\mathbf{v}|$ the cardinality of $\mathbf{v}$), which depends directly on the value of a. For most cases, a is a small number, so the algorithm is much more efficient than an exhaustive search algorithm with exponential computational complexity. More information about the algorithm can be found in [16] and [11].

978-1-4244-2324-8/08/$25.00 © 2008 IEEE.
This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE "GLOBECOM" 2008 proceedings.

IV. Experimental Results

In this section, video coding is performed using the H.264/AVC JM 12.2 codec. The first 100 frames of the QCIF sequence "Foreman" are coded at frame rate 30 frames/second. All frames except the first are encoded as P frames. We assume that a video frame consists of only one slice. To avoid prediction error propagation, a 10% macroblock level intra-refreshment is used. When a packet is lost during transmission, we use the temporal-replacement error concealment strategy: the motion vector of a missing MB can be estimated as the median of motion vectors of the nearest three MBs in the preceding row. If that row is also lost, the estimated motion vector is set to zero. The pixels in the previous frame, which are pointed to by the estimated motion vector, are used to replace the missing pixels in the current frame. In our experiments, we consider the QP of each slice as the video coding parameter to be optimized. The optional values of QP are 4, 6, 8, 10, 14, 18, 24, 30, 38, and 46. The expected video quality at the receiver is measured by the peak signal-to-noise ratio (PSNR).

For the network simulation, the use multihop topology which is shown in the Figure 2. Each node has a queue with a queue length of 100 packets for packet buffering. As in [2], we set the default value of t_{RTO} as $t_{RTO} = 4 \times RTT$ in our NS-2 simulator. We also set the link capacity as 2 Mbps and the optional TFRC sending rates are 1.0 Mbps, 1.2 Mbps, 1.4 Mbps, 1.6 Mbps, 1.8 Mbps, 1.9 Mbps. Figure 3

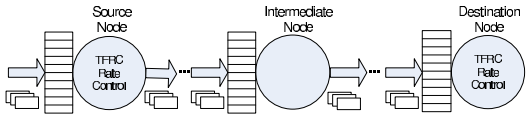

Fig. 2. The network model used in our experiments.

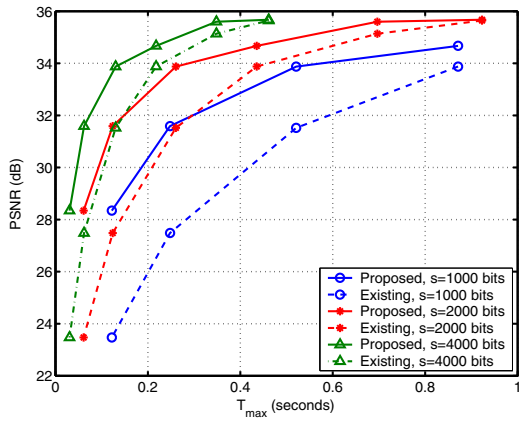

Fig. 3. PSNR-delay performance comparisons with different packet sizes.

shows the PSNR-delay performance comparisons between the proposed joint optimization scheme and the existing scheme with video coding and sending rate being separated. The hop number of the network is 4. In the implementation of existing schemes, we encode the video with fixed QP values. Each mark on dashed lines corresponds to a fixed QP value, and large values of QP correspond to small values of $T_n^{\max}$. The values of $T_n^{\max}$ are set equal to the resulting maximum frame delay of overall 100 frames coded by the corresponding QP. To guarantee comparison fairness, we set the values of $T_n^{\max}$ in the proposed joint optimization scheme equal to those in the existing schemes. From the Figure 3, we can observe that with equal frame deadlines, the proposed joint optimization scheme has a superior PSNR performance over the existing scheme. This is because the video coding parameter and the sending rate are jointly optimized, leading to a good trade-off between the resulting video distortion and end-to-end delay. Moreover, the tighter the frame delay deadline, the larger the obtained PSNR performance gain. This means our proposed scheme is especially suitable for real-time video communications. We also observe that, with the same video frame delay deadline, both the proposed scheme and the existing scheme have a better PSNR performance with the increase of packet sizes. This is because larger packet sizes lead to larger transport-layer sending rates. In return, larger sending rates lead to finer video coding with fixed video frame delay deadlines. Although larger packet sizes may cause larger packet loss rates, the resulting finer video coding has a greater impact on the video quality over the resulting packet loss rate. Therefore, Figure 3 suggests that large packet sizes are preferred in delay-constrained video communications while packet sizes make little difference for video communications without strict delay requirements. The RTT at the transport layer is estimated by

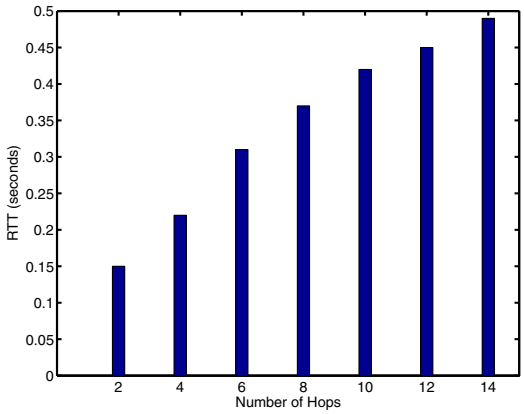

Fig. 4. Measured RTT's vs. the number of hops.

measuring the time elapsed between sending a data packet and receiving the acknowledge. Figure 4 depicts the RTT estimate of the network with different number of hops when the packet size is 4000 bits. Figure 5 shows the corresponding PSNR-delay performance comparisons with different number of hops between the proposed joint optimization scheme and the existing scheme with video coding and sending rate being separated. As shown in the figure, the number of hops has a significant impact on the PSNR-delay performance of the proposed scheme as well as that of the existing scheme. This is because smaller number of hops leads to smaller RTT's.

978-1-4244-2324-8/08/$25.00 © 2008 IEEE.

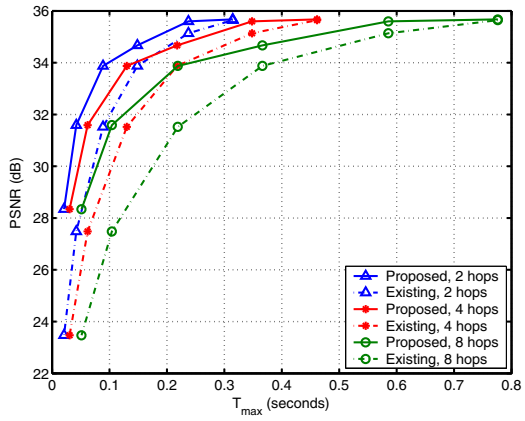

Fig. 5. PSNR-delay performance comparisons with different number of hops.

Within a fixed video frame delay deadline, the smaller RTT allows finer video coding, which brings a better received video quality. The same conclusion of Figure 3 goes to Figure 5 that the proposed scheme has a significant video quality improvement over the existing scheme under tighter video frame delay deadlines. Table I shows the statistics of the

TABLE I
STATISTICS OF QP AND TFRC SENDING RATE

Scheme	$\mathbf{T_n^{max} = 0.031s}$ PSNR=28.3 (dB)		$\mathbf{T_n^{max} = 0.062s}$ PSNR=31.6 (dB)	
	QP	R_{TFRC} (Mbps)	QP	R_{TFRC} (Mbps)
Existing	46	1.4	38	1.4
Proposed	46 (21) 38 (60) 30 (19)	1.2 (5), 1.4 (65) 1.6 (26) 1.8 (4)	38(28) 30 (56) 24 (16)	1.2 (3), 1.4 (43) 1.6 (32),1.8 (15) 1.9 (7)

(a) Original (b) Optimized (c) Existing

Fig. 6. Frame 82 from the reconstructed video.

selections of QP and TFRC sending rate in our proposed joint optimization scheme over the first 100 frames with the corresponding $T_n^{\max}$. The hop number is 4 and the packet size is 4000 bits. The values in brackets mean the frequency (also the number of frames) of the corresponding QP and the TFRC sending rate being used in coding the 100 video frames, respectively. This table indicates the dynamic variations of QP and TFRC sending rate required to adapt the network traffic conditions. To illustrate the quality of video frames, we plot a sample video frame from the original video in Figure 6 (a) and compare it to the reconstructed video frames under our proposed scheme and the existing scheme for $T_n^{\max} = 0.062s$, respectively. The frame under our proposed scheme has a visual quality very close to the original frame, while the frame under the existing scheme is considerably blurry.

V. CONCLUSIONS

In this paper, we have considered the joint optimization of real-time coding and the transport-layer sending rate based on the TFRC protocol. In our framewok, the video coding process at the application layer and the transmission rate control at the transport layer interact with and adapt to each other in a quality-driven approach. The optimal video coding parameters and TFRC sending rate are obtained by using dynamic programming. Experiments were conducted by using the H.264/AVC JM 12.2 codec and NS-2 simulator. Experimental results indicate that the proposed joint optimization framework can effectively achieve a good trade-off between the video compression efficiency and the network congestion control. The results also show that the proposed framework can significantly improve the video quality over the existing schemes, especially when delay bound is tight.

REFERENCES

[1] J. Yan, K. Katrinis, M. May, and B. Plattner, "Media- and TCP-Friendly Congestion Control for Scalable Video Streams," *IEEE Transactions on Multimedia*, vol. 8, pp. 196–206, Apr. 2006.
[2] S. Floyd and K. Fall, "Promoting the use of end-to-end congestion control in the Internet," *IEEE/ACM Trans. Networking*, vol. 7, pp. 458–472, Aug 1999.
[3] S. Floyd, M. Handley, J. Padhye, and J. Widmer, "Equation-Based Congestion Control for Unicast Applications," *Pro. ACM SIGCOMM*, Aug 2000.
[4] A. Argyriou, "A Joint Performance Model of TCP and TFRC with Mobility Management Protocols," *Wiley Wireless Communications and Mobile Computing (WCMC) Special Issue on Mobile IP*, vol. 6, pp. 547–557, Aug 2006.
[5] M. Li, C. Lee, E. Agu, M. Claypool, and R. Kinicki, "Performance Enhancement of TFRC in Wireless Ad Hoc Networks," *Distributed Multimedia Systems (DMS)*, September 2004.
[6] Z. Fu, X. Meng, and S. Lu, "A Transport Protocol For Supporting Multimedia Streaming in Mobile Ad Hoc Networks," *IEEE JSAC*, December 2004.
[7] K. Chen and K. Nahrstedt, "Limitations of Equation-based Congestion Control in Mobile Ad Hoc Networks," *Workshop on Wireless Ad Hoc Networking (WWAN)*, March 2004.
[8] J. P. M. Handley, S. Floyd and J. Widmer, "TCP friendly rate control (TFRC): Protocol specification," *TRFC 3448*, vol. 8, Jan. 2003.
[9] X. Zhu and B. Girod, "Media-Aware Multi-User Rate Allocation over Wireless Mesh Networks," in *IEEE OpComm-06*, Sep. 2006, pp. 1–8.
[10] ——, "Distributed Rate Allocation for Video Streaming over Wireless Networks with Heterogeneous Link Speeds," in *ISMW-07*, Aug. 2007, pp. 296–301.
[11] D. Wu, S. Ci, and H. Wang, "Cross-layer optimization for video summary transmission over wireless networks," *IEEE J. Select. Areas Commun.*, vol. 25, no. 4, pp. 841–850, may 2007.
[12] R. Zhang, S. L. Regunathan, and K. Rose, "Video Coding with Optimal Inter/Intra-Mode Switching for Packet Loss Resilience," *IEEE J. Select. Areas Commun.*, vol. 18, no. 6, pp. 966–976, Jun. 2000.
[13] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC Video Coding Standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 560–576, Jul. 2003.
[14] D. Wu, S. Ci, and H. Wang, "Cross-Layer Optimization for Packetized Video Communications over Wireless Mesh Networks," in *IEEE ICC*, China, May 2008.
[15] G. M. Schuster and A. K. Katsaggelos, *Rate-Distortion Based Video Compression: Optimal Video Frame Compression and Object Boundary Encoding*. Norwell, MA: Kluwer, 1997.
[16] K. R. A. Ortega, "Rate-distortion methods for image and video compression," *Signal Processing*, vol. 15, pp. 23–50, Nov 1998.

978-1-4244-2324-8/08/$25.00 © 2008 IEEE.
This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE "GLOBECOM" 2008 proceedings.