Title

Wireless H.264 Video Quality Enhancement Through Optimal Prioritized Packet Fragmentation

Journal

IEEE Transactions on Multimedia, 14(5)

ISSN

1520-9210 1941-0077

Authors

Kambhatla, K. K. R
Kumar, S.
Paluri, S.
et al.

Publication Date

2012-10-01

DOI

10.1109/TMM.2012.2196508

Peer reviewed

Wireless H.264 Video Quality Enhancement Through Optimal Prioritized Packet Fragmentation

Kashyap K. R. Kambhatla, *Student Member, IEEE*, Sunil Kumar, *Senior Member, IEEE*, Seethal Paluri, *Student Member, IEEE*, and Pamela C. Cosman, *Fellow, IEEE*

***Abstract*—We introduce a cross-layer priority-aware packet fragmentation scheme at the MAC layer to enhance the quality of pre-encoded H.264/AVC compressed bitstreams over bit-rate limited error-prone links in wireless networks. The H.264 slices are classified in four priorities at the encoder based on their cumulative mean square error (CMSE) contribution towards the received video quality. The slices of a priority class in each frame are aggregated into video packets of corresponding priority. We derive the optimal fragment size for each priority class which achieves the maximum expected weighted goodput at different encoded video bit rates, slice sizes and bit error rates. Priority-aware packet fragmentation invokes slice discard in the buffer due to channel bit rate constraints on allocating fragment header bits. We propose a slice discard scheme using frame importance and slice CMSE contribution to control error propagation effects. Packet fragmentation is extended to slice fragmentation by modifying the conventional H.264 decoder to handle partial slice decoding. Priority-aware slice fragmentation combined with the proposed slice discard scheme provides considerable PSNR and VQM gains as compared to priority-agnostic fragmentation.**

***Index Terms*—H264, MAC layer, packet fragmentation, priority-aware, PSNR, real-time, video compression, video quality, video slice, video streaming, VQM, weighted goodput.**

I. Introduction

MULTIMEDIA applications such as video streaming, which are delay sensitive and bandwidth intensive, are growing rapidly over wireless networks. Unlike file transfers, a video can tolerate some packet losses. However, existing wireless networks provide only limited and time-varying quality-of-service (QoS) support for these applications.

H.264 AVC is the state-of-the-art video compression standard jointly developed by the ITU and ISO [1]. The compressed video is vulnerable to losses due to channel impairments. Lost video packets induce different levels of quality degradation due to temporal and spatial dependencies in the compressed bitstream. An important problem which affects video quality is error propagation when an error in a reference frame propagates in the decoder to future reconstructed frames which are predicted from that reference frame. This problem led to the design of error-resiliency features such as flexible macroblock ordering (FMO), data partitioning and error concealment schemes in H.264 [1]–[3].

Packet segmentation and reassembly, carried out at the transport layer of the source and gateway nodes to comply with the maximum packet size requirements of intermediate networks [4], [5], cannot efficiently adapt to the varying channel conditions at the intermediate nodes. Furthermore, video streaming uses real-time transport protocol (RTP) and user datagram protocol (UDP). As a result, the transport layer is less important for error protection and bandwidth adaptation. Recent research has demonstrated the promise of cross-layer protocols for supporting the QoS demands of multimedia applications over wireless networks [6]–[8]. For example, van der Schaar *et al.* [7] show the benefits of the joint APP-MAC-PHY approach for transmitting video over wireless networks.

Packet fragmentation at the MAC layer is primarily done to adapt the packet size to the channel error characteristics, in order to improve the successful packet transmission probability and reduce the cost of packet retransmissions. MAC layer fragmentation and retransmission also avoid costly transport layer retransmissions [9], [10]. Fragmentation calls for a trade-off between reducing the number of overhead bits per packet by adopting large fragments and reducing the transmission error rate by using small fragments. However maximum throughput does not guarantee the minimum video distortion at the receiver due to the following reasons—*First*, unlike data packets, loss of H.264 compressed video packets induces different amounts of distortion in the received video. Therefore the fragment size should be adaptive to the packet priority. *Second*, conventional packet fragmentation schemes discard a packet unless all its fragments are received correctly. However, video data is loss tolerant and a packet can be partially decoded even when some of its fragments are lost. Also real-time video transmission is delay-sensitive and retransmission of corrupted fragments may not be feasible.

Manuscript received March 05, 2011; revised June 10, 2011 and October 25, 2011; accepted April 06, 2012. Date of publication April 25, 2012; date of current version September 12, 2012. This material is based upon work that was supported in part by the US Air Force Research Laboratory under agreements FA8750-08-1-0078 and FA8750-11-1-0048 at San Diego State University. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Qibin Sun.

K. K. R. Kambhatla is with the College of Engineering, San Diego State University, San Diego, CA 92182-1326 USA, and also with the Department of Electrical and Computer Engineering, University of California San Diego, La Jolla, CA 92093-0409 USA (e-mail: kkambhat@ucsd.edu).

S. Kumar is with the Department of Electrical and Computer Engineering, San Diego State University, San Diego, CA 92182-1309 USA (e-mail: skumar@mail.sdsu.edu).

S. Paluri is with the Computational Science Research Center, San Diego State University, San Diego, CA 92182-1245 USA (e-mail: spaluri@sciences.sdsu.edu).

P. C. Cosman is with the Department of Electrical and Computer Engineering, University of California San Diego, La Jolla, CA 92093-0407 USA (e-mail: pcosman@eng.ucsd.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMM.2012.2196508

In this paper, we consider H.264 AVC encoded videos with slicing configuration, where each slice can be independently decoded. The video slices are classified in four priority classes based on the distortion contributed by their loss to the received video quality. The slices of a priority class in each frame are aggregated into corresponding video packets whose size is bounded by the network maximum transmission unit (MTU).

The proposed cross-layer approach considers real-time streaming of the pre-encoded H.264 video streams. Under known link conditions, we address the problem of assigning optimal fragment sizes to the individual priority packets at the MAC layer, within the channel bit-rate limitations. In our scenario, optimality is defined by maximizing the expected weighted goodput. The scheme will provide higher transmission reliability to the high priority packets by using smaller fragments, at the expense of 1) allowing larger fragment sizes for the low priority packets, and 2) discarding some low priority packets to meet the channel bit-rate limitations, whenever necessary. The branch-and-bound (BnB) algorithm along with an interval arithmetic method [11]–[13] is used to find the maximum expected weighted goodput and derive the optimal fragment sizes. Other features include a slice discard scheme based on the frame importance and the cumulative mean square error (CMSE) contribution of the slice, a slice fragmentation approach by modifying the conventional H.264 decoder to handle partial slice decoding, and use of various slice sizes. We show that adapting fragment sizes to the packet priority classes reduces the overall expected video distortion at the receiver. Our scheme does not assume retransmission of lost fragments and packets since real-time video transmission is delay-sensitive and retransmission of corrupted fragments may not be feasible. A preliminary shorter version of this paper appeared in [14].

Section II discusses past research on packet fragmentation and packet size optimization. Section III formulates the expected weighted goodput maximization problem and presents the interval arithmetic analysis for determining the fragment sizes. The comparison between the performance of priority-aware and priority-agnostic fragmentation is discussed in Section IV. Experimental results for different combinations of video bit rates, slice sizes and varying channel conditions are discussed in Section V. Finally Section VI discusses the applications of the proposed model and conclusions.

II. Related Work

MAC frame length control has been studied in the past for different purposes such as 1) maximizing wireless network or user or link throughput [15]–[18], 2) optimizing energy efficiency and transmission range [19]–[21], 3) dynamic multi-rate link adaptation [22], and 4) goodput enhancement, delay and retransmission control [23]. Yin *et al.* in [15] determine the feasible packet size which achieves maximum network saturation throughput for a WLAN operating in the IEEE 802.11 distributed coordination function mode. The channel conditions are assumed to be constant throughout the network and the overhead due to retransmission bounds at the MAC layer is taken into consideration. The derived packet size is fixed for every user in the network. Throughput enhancement over time-varying Ricean fading channels in WLANs is studied in [16]. The authors dynamically vary the MAC fragmentation threshold based on the ratio of transmission rate allowed by the receiver (based on its channel conditions) to the sender's packet generation rate. Though lower fragmentation threshold provides better throughput, this scheme does not consider the effect of additional overhead bits and the back-off time on packet delay. A cross-layer design between the MAC and physical layer (PHY) in WLANs is studied in [17]. The optimal packet size is derived for direct sequence spread spectrum and frequency hopping spread spectrum systems by using the channel BER as an interface parameter. The performance of automatic repeat request combined with dynamic packet fragmentation is studied during wireless channel failures (i.e., unavailability of channel) for highly concentrated Gaussian or exponentially distributed data transmitted over distributed computing systems [18]. The authors show that MAC retransmissions of packets of different sizes result in power-law delays and poor utilization of network resources. They propose aggregating small packets into larger ones and using dynamic fragmentation depending on the channel availability period to improve network throughput.

The throughput enhancement in the above schemes [15]–[18] is observed at poor channel conditions at the cost of longer packet delay. Adaptive MAC frame length control has also been used for increasing energy efficiency in [19] and latency control for meeting packet delay requirements in real-time environments on test beds in [20] and [21]. Goodput analysis combined with link rate adaptation for IEEE 802.11a is carried out as a function of the payload size in [22]. This scheme uses an auto rate fallback mechanism which alternates between 1 Mbps and 2 Mbps PHY rates depending on the result of the timeout function and the missed ACK frames. The goodput is computed as the ratio of the payload transmission time to the total time needed to transmit a data packet. Another dynamic fragmentation scheme for goodput enhancement is discussed in [23]. In the above schemes, a successful data packet reception requires the successful transmission/re-transmission of each packet fragment. This introduces considerable delay which may not be suitable for real-time multimedia applications [24].

Recent research has shown the promise of cross-layer protocols in wireless networks [6]–[8]. van der Schaar *et al.* [7] discuss different cross layer solutions and extend the MAC-centric approach to demonstrate that the joint APP-MAC-PHY approach is best suited for transmitting multimedia (e.g., video streaming) over wireless networks.

Lately some cross-layer packet aggregation and fragmentation schemes have been proposed for enhancing H.264 compressed video transmission over wireless networks [25]–[28]. Fallah *et al.* [25] proposed the fragmentation and aggregation of H.264 network abstraction layer units (NALU) in order to enhance the quality of the decoded video stream over IEEE 802.11 WLAN. They showed that the video quality is increased when the fragmentation is done at the application layer through slicing the video compared to when fragmentation is done at the MAC layer. However, they did not combine the application layer slicing and aggregation with MAC layer fragmentation and also did not consider the packet priorities. Connie *et al.*

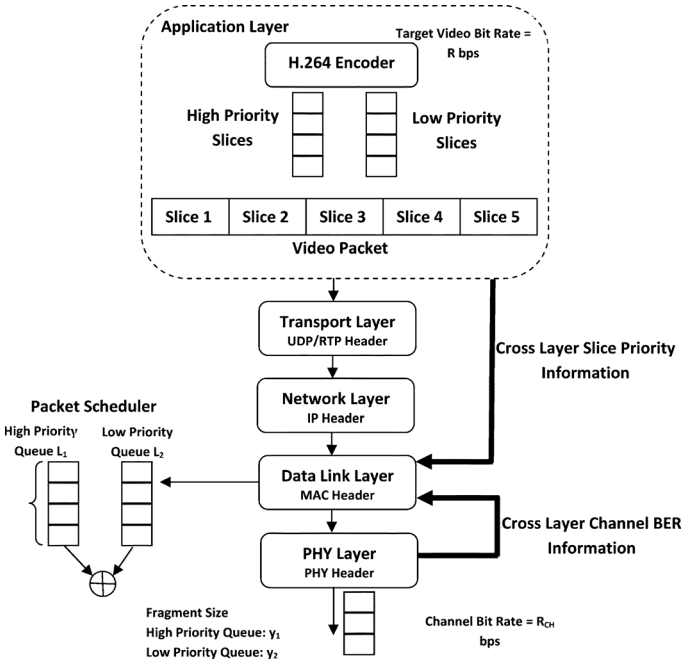


Fig. 1. Cross-layer fragmentation approach.

[26] extended the idea of application layer fragmentation proposed in [25] to 3G UMTS networks in both uplink and downlink transmissions. The Extended Profile of H.264 divides the video data into three partitions (DP A, B and C) with different levels of importance. In order to support QoS requirements of a H.264 bitstream, Ksentini *et al.* [27] mapped the IEEE 802.11e MAC access categories (AC) to these different data partitions (DP). Fallah *et al.* [28] extended the idea in [27] and employed controlled access phase scheduling (CAPS) in the IEEE 802.11e hybrid coordination function controlled channel access (HCCA) mode.

III. Proposed Cross-Layer Fragmentation Scheme

A. H.264 Slice and Video Packet Formation

In this paper, we consider videos which are pre-encoded using H.264/AVC with fixed slice size configuration. In this configuration, macroblocks are aggregated into a slice such that their accumulated size does not exceed the pre-defined slice size. However, the chosen slice size represents the upper limit and some slices may be smaller [29].

The network limits the number of bytes that can be transmitted in a single packet based on the MTU bound. The slices formed at the encoder are aggregated into a video packet for transport over IP networks and each of these packets is appended with RTP/UDP/IP headers of 40 bytes [30] as shown in Fig. 1. This aggregation of slices helps to control the amount of network overhead added to the video data. If the video slices are classified in two or more priority classes as explained in Section III-B, the priority slices of each frame are separately aggregated to form packets. The video packets are fragmented at the data link layer and each fragment is attached with 50 byte MAC and PHY layer headers. Fig. 1 illustrates the cross-layer fragmentation approach.

We use a binary symmetric channel $BSC(p_b)$ where p_b is the BER. The data link layer fragments the packets using channel BER information from the PHY layer and slice priority information from the application layer. Here we assume that the data link layer is continuously updated with the channel BER from the PHY layer.

B. Slice Priority

H.264 slices are prioritized based on their distortion contribution to the received video quality. The total distortion of one slice loss is computed using CMSE which takes into consideration the error propagation within the entire GOP. All slices in a GOP are distributed into four priority classes based on their pre-computed CMSE values. Priority 1 slices induce the highest distortion whereas priority 4 slices induce the least distortion to the received video quality. The slice priority value is stored in the 2-bit nal_ref_idc field of the slice header [31]. We combine priority 1 and priority 2 slices into a "high priority" class and priority 3 and priority 4 slices into a "low priority" class to reduce the complexity of the optimization algorithm discussed in Section III-E. However, the original four-level priority information can still be accessed from the nal_ref_idc field of each slice and will be used in our proposed slice discard scheme.

C. Video Packet Fragmentation

Optimal fragment size is determined to maximize the expected weighted goodput which will be explained in Section III-D2. We design two types of packet fragmentation schemes—video packet priority-agnostic and priority-aware. Each of these schemes has two types, *slice fragmentation disabled* and *slice fragmentation enabled*. When slice fragmentation is disabled, the fragment size cannot be smaller than the target slice size and each fragment contains one or more slices in their entirety. This restriction on the fragment size is not needed when slice fragmentation is enabled. As a result, a fragment can contain partial slice data, including the cases of less than one slice and more than one slice. In Fig. 2, we illustrate an example where the computed optimal fragment size is larger than the target slice size but smaller than twice the target slice size. In the slice fragmentation disabled case, Fragment 1 contains only slice 1 since slice 2 is too large to fit entirely in it. Similarly, Fragments 2 and Fragment 3 contain only slice 2 and slice 3, respectively, since the next slice cannot fit in them. Slices 4 and 5 are small enough to be aggregated in Fragment 4. For the slice fragmentation enabled case, Fragment 1 contains slice 1 and the initial portion of slice 2, Fragment 2 contains the remaining portion of slice 2 and the complete slice 3, and Fragment 3 contains slices 4 and 5.

To enable slice fragmentation, the H.264 decoder is modified to perform partial slice decoding [32], [33]. If the first fragment of a slice containing the slice header is lost, the entire slice is discarded. When an intermediate fragment is in error, the macroblock (MB) data of the slice contained in fragments before the corrupted fragment is successfully decoded, and the remaining MB data is concealed at the decoder. We use the concealment scheme implemented in JM 14.2 [29].

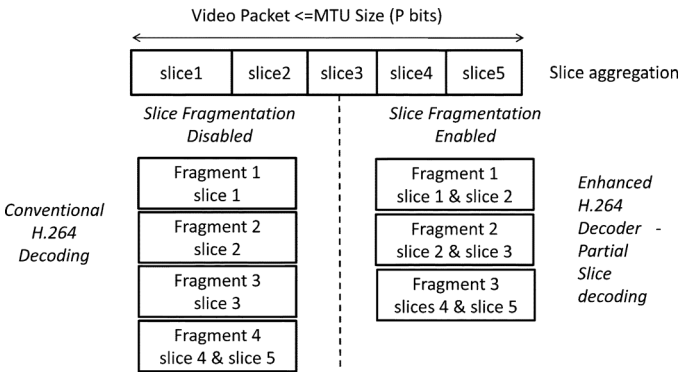


Fig. 2. Illustration of video packet fragmentation with and without slice fragmentation.

D. Problem Formulation for Determining Optimal Fragment Sizes

In conventional packet fragmentation schemes, the data link layer at the receiver expects that erroneous fragments of a packet would be retransmitted, and the entire packet is discarded if any one of its fragments is not received properly. However, retransmission of corrupted fragments may not be feasible in real-time video streaming applications. Since the video bitstream is tolerant to packet losses, the decoder reconstructs the lost packets or fragments using error concealment. Video traffic can also tolerate some low priority slices being discarded to accommodate a higher fragmentation overhead when the overall video bit rate exceeds the channel bit rate. In this section, we discuss the priority-agnostic and priority-aware fragmentation schemes. Here the priority-agnostic fragmentation scheme is based on [20], [25], and [26].

1) Priority-Agnostic Fragmentation: A measure of the reliable transmission of packets over error-prone channels is *goodput*. We define the goodput G as the expected number of successfully received video bits per second (bps) normalized by the target video bit rate R bps. G depends on the fragment success rate (fsr) which is a function of the fragment size (y) and the channel BER (p_b). We assume that each slice is x bits long in our theoretical formulation. A fragment is successfully received iff all the bits of that fragment are received without error. The fsr is expressed as

$$fsr = (1-p_b)^y, \quad y = nx + h. \tag{1}$$

Here, the fragment size is y bits, containing nx bits of slice data (i.e., payload) and h MAC and PHY header bits. For a given value of y, F_{TX} is the corresponding number of fragments transmitted every second and F_{RX} is the corresponding expected number of successfully received fragments. F_{RX} is computed as $F_{RX} = (fsr)(F_{TX})$. We assume that the channel bit rate is R_{CH} bps, the average video bit rate is R bps, and on average $N = R/x$ slices are generated every second. The number of payload bits in a fragment can vary from 1 to P bits, where P represents the MTU size. Therefore, the feasible number of slices in each fragment varies as $n \,\epsilon\, [1/x \; P/x]$. If slice fragmentation is disabled, n is an integer with minimum value of 1. The expected goodput G is computed, after excluding the header bits associated with each fragment, as

$$G = \frac{F_{RX}(y-h)}{R} = \frac{F_{TX}(1-p_b)^y(y-h)}{R}. \tag{2}$$

Here, the objective is to find the optimal fragment size y such that G is maximum:

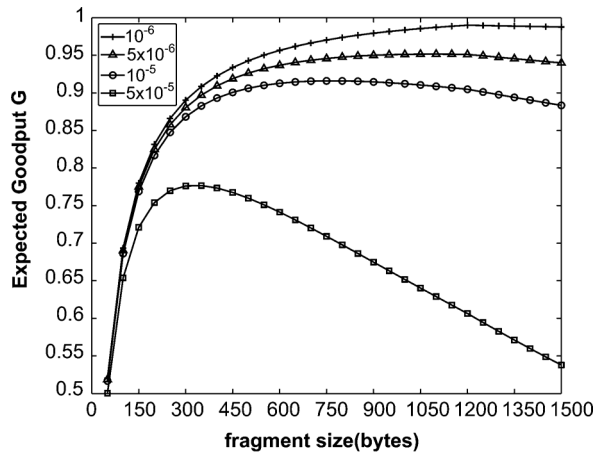
$$y = \arg\max_y G = \arg\max_y \frac{F_{TX}(1-p_b)^y(y-h)}{R}$$
$$F_{TX} = \begin{cases} (\frac{N}{n}); (\frac{N}{n}) \le \frac{R_{CH}}{y} \\ \frac{R_{CH}}{y}; (\frac{N}{n}) > \frac{R_{CH}}{y}. \end{cases} \tag{3}$$

Condition $(N/n) \le R_{CH}/y$ in (3) implies that sufficient bits are available to allocate headers to all the fragments generated in one second. The condition $(N/n) > R_{CH}/y$ implies that for a fragment size of y bits, the requirement for the number of overhead bits exceeds the channel bit rate. Therefore the corresponding number of application layer packets that would be discarded is $\lceil ((N/n) - (R_{CH}/y))\, n/(P/x) \rceil$. The corresponding number of discarded slices (D_S) is

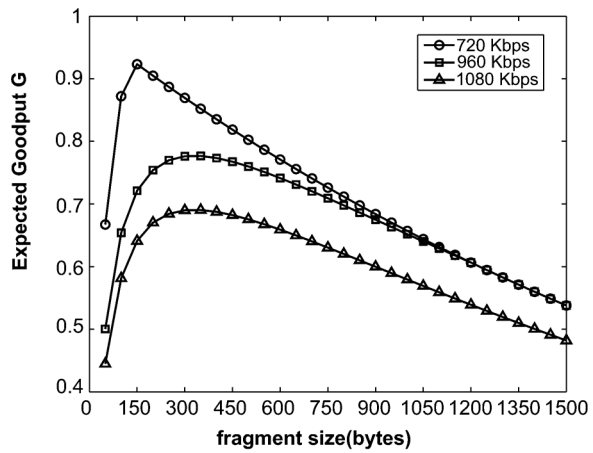
$$D_S = \left\lceil \left(N - \left(\frac{R_{CH}}{y} \right) n \right) \right\rceil. \tag{4}$$

We use a quantized exhaustive search algorithm with a step size of 50 bytes for 30 possible fragment sizes between 50 and 1500 bytes to compute the optimal fragment size in (3). Fig. 3(a) shows the variation in expected goodput G for different fragment sizes and channel BERs for a video encoded at $R = 960$ Kbps with 150 byte slices. The channel bit rate R_{CH} is set to 1 Mbps for all the cases discussed in this paper. The maximum video data in a fragment is limited by P = 1500 bytes. For a fragment of 1500 bytes, the maximum value of G is 55% for $p_b = 5 \times 10^{-5}$ which increases to 98% for a lower channel error rate $p_b = 10^{-6}$, because the fragment success rate increases as the channel BER decreases. The expected goodput also depends on the number of slices discarded. Note that more slices are discarded as the fragment size decreases since the requirement for header bits increases. Therefore, for a fragment size of 150 bytes, though fsr is higher than that for larger fragment sizes, the corresponding G is lower. We observe that the value of G for $p_b = 5 \times 10^{-5}$ is significantly lower than for lower values of p_b. The system achieves a higher value of G at this BER when the encoding bit rate is lower, as shown in Fig. 3(b) for the 720 Kbps video bit rate. There lies an optimal point in each case which trades off the losses due to channel errors with the packet discards. For example, the maximum value of G is achieved at fragment sizes of 300 and 750 bytes for $p_b = 5 \times 10^{-5}$ and 10^{-5}, respectively.

Fig. 3(b) illustrates the variation in G for different fragment sizes and three different encoded video bit rates at $p_b = 5 \times 10^{-5}$. For $R = 720$ Kbps, sufficient bits are available to allocate headers to each fragment. So every slice of the video packet can be transmitted independently in a fragment with maximum $G = 93\%$. However, the maximum achievable G decreases as the encoded video bit rate increases and gets close to R_{CH} (i.e., $R = 960$ Kbps) or exceeds R_{CH} (i.e., $R = 1.08$ Mbps). This is because fewer bits are now available for allocating fragment



(a)



(b)

Fig. 3. Expected goodput G versus fragment size y at (a) R $= 960$ Kbps and different p_b, and (b) $p_b = 5 \times 10^{-5}$ and different R.

headers. More header bits can only be accommodated by discarding some slices. As a result, the maximum value of G decreases to 77% and 69% for video bit rates of 960 Kbps and 1080 Kbps, respectively, when each fragment contains two slices.

Fig. 4 shows the amount of discarded data for different video encoding rates at $p_b = 5 \times 10^{-5}$. As the video encoding rate increases, more slices are generated every second. When the encoding rate is 720 Kbps, sufficient bits are available to allocate fragment headers and hence no slice is discarded when fragment size $\geq$ 150 bytes. When R increases to 960 Kbps, the amount of discarded data increases. When the encoding rate (1080 Kbps) exceeds R_{CH}, 14.1 Kbytes worth of slice data is discarded every second even for a 1500 byte fragment size (i.e., no fragmentation). Though one may be inclined to choose a large fragment size to reduce the number of discarded slices, it also decreases the fragment success rate as explained in Fig. 3 and shown in (1).

2) Priority-Aware Fragmentation: We extend the fragmentation scheme to make it adaptive to the individual packet priority classes. We assign smaller fragment sizes to higher priority packets to increase their transmission success probability. The video packets are divided in two priorities and the link layer scheduler shown in Fig. 1 transmits all the high priority

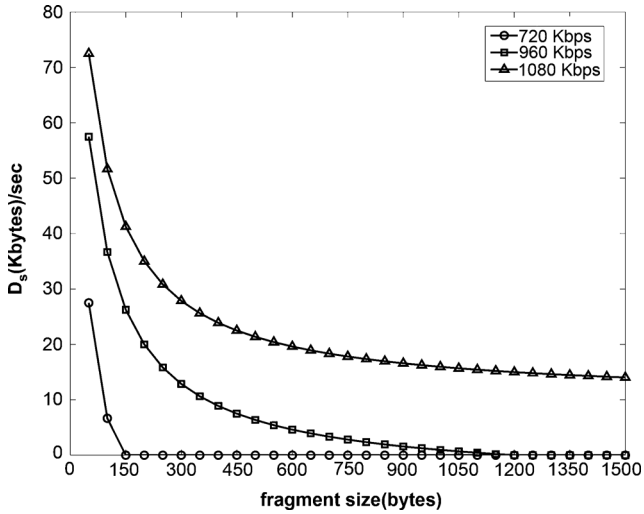


Fig. 4. Slice data discarded per second at $p_b = 5 \times 10^{-5}$.

fragments before low priority fragments during every second. We define a new performance parameter called the expected weighted goodput G_W, which is computed as a linear combination of individual priority goodput :

$$G_W = w_1 g_1 + w_2 g_2. \tag{5}$$

The weights w_1 and w_2 capture the relative distortion contribution per bit from the individual slice priorities. w_1 is computed as the ratio of the mean CMSE of the high priority slices to the mean CMSE of all slices in the pre-encoded video, and $w_2 = 1 - w_1$. We used the median of all slice CMSE values as the CMSE threshold for assigning slice priority. The weights depend on this threshold, video content and encoding parameters such as target encoding rate R and slice size x. We define $n_1,\ n_2\ \epsilon\ [1/x\ P/x]$ as the number of slices that are aggregated into each fragment of the high priority and low priority packets if slice fragmentation is disabled. The corresponding fragment sizes would be $y_1 = n_1 x + h$, $y_2 = n_2 x + h$ bits. Let N be the total number of slices generated in one second, and l_1 and l_2 be the corresponding numbers of high priority and low priority slices generated per second. During each second it is difficult to predict the number of packets in each priority queue at the data link layer. If video during some period of time has high motion activity, it would have more slices with CMSE values greater than the threshold. As a result, there will be more high priority packets. In any given second, the number of high priority slices can vary from $[0\ N]$ and the expected number is $N/2$. Hence a truncated normal distribution, which is symmetric about $N/2$ and spanning from 0 to N is considered here:

$$p(l_1 = k) = \frac{K_1}{\sqrt{2\pi}} e^{-(k-N/2)^2/2}$$
$$\text{for } k = 0, 1, 2, 3, \ldots, N \text{ and } l_1 + l_2 = N \tag{6}$$

where K_1 is a normalization constant to make this a proper probability mass function. Now we find $\bar{y} = [y_1\ y_2]$ which maximizes G_W averaged over all possible queue lengths from $[0\ N]$

$$\max_{\bar{y}} G_W = \max_{\bar{y}} \sum_{l_1} p(l_1)(w_1 g_1 + w_2 g_2). \tag{7}$$

Here, $p(l_1)$ is the probability of a given high priority queue length l_1. The individual priority goodputs g_1 and g_2 for a given high priority queue length l_1 and corresponding low priority queue length l_2 are therefore computed using the expected goodput formula expressed in (3):

$$g_1 = \begin{cases} (1-p_b)^{y_1}; & \left(\frac{l_1}{n_1}\right) \le \frac{R_{CH}}{y_1} \quad (a) \\ \frac{\left(\frac{R_{CH}}{y_1}\right)(y_1-h)(1-p_b)^{y_1}}{R}; & \left(\frac{l_1}{n_1}\right) > \frac{R_{CH}}{y_1} \quad (b) \end{cases} \tag{8}$$

$$g_2 = \begin{cases} (1-p_b)^{y_2}; & \left(\frac{l_2}{n_2}\right) \le \frac{\left(R_{CH}-\left(\frac{l_1}{n_1}\right)y_1\right)}{y_2} \quad (a) \\ \frac{\left[\frac{\left(R_{CH}-\left(\frac{l_1}{n_1}\right)y_1\right)}{y_2}\right](y_2-h)(1-p_b)^{y_2}}{l_2 x}; & \\ \left(\frac{l_2}{n_2}\right) > \frac{\left(R_{CH}-\left(\frac{l_1}{n_1}\right)y_1\right)}{y_2} \quad (b). \end{cases} \tag{9}$$

The low priority goodput g_2 is computed from the bits remaining to be allocated after all the high priority fragments have been transmitted during each second. Condition (a) in (8) and (9) implies that sufficient bits are available to allocate fragment headers when high and low priority fragments are transmitted at sizes y_1 and y_2. Condition (b) in (8) implies that all the low and some high priority slices should be discarded to accommodate the overhead demand and satisfy the channel bit rate constraint while transmitting at a fragment size y_1. The slice discard scheme is discussed later in Section V-C. Further, Condition (b) in (9) implies that there are sufficient bits to transmit all high priority fragments at size y_1, but not for transmitting all low priority fragments at size y_2. Therefore, some low priority slices should be discarded. Combining (7)–(9) and substituting $l_2 = N - l_1$, we formulate the objective function to maximize G_W as shown in (10) at the bottom of the page.

Fig. 5 shows G_W and the number of discarded slices during one second for the CIF Foreman video encoded at $R = 960$ Kbps over a channel with $R_{CH} = 1$ Mbps at $p_b = 10^{-5}$. The weights $(w_1,\ w_2) = (0.89,\ 0.11)$ used were derived for the Foreman video sequence. The mean CMSE value of high priority slices contributes 89% of the received video distortion whereas the mean CMSE value of low priority slices contributes only 11%. The optimal fragment sizes are determined in terms of the number of 150 byte slices that can be aggregated into each priority fragment. In Fig. 5(a), $(n_1, n_2) = (2, 5)$ and $[(y_1, y_2) = (300{,}750) + h]$ are the optimal high and low priority fragment sizes which achieve the maximum goodput of 0.954. This is achieved at the cost of discarding 56 low priority slices per second as shown in Fig. 5(b). As the fragment size decreases, the fragment success rate increases but the number of discarded slices also increases due to higher fragment overhead. When $(n_1, n_2) = (1, 1)$, more than 175 slices are discarded as shown in Fig. 5(b) and the corresponding G_W decreases to 0.92 in Fig. 5(a). Also when $(n_1, n_2) = (10, 1)$ i.e., low priority packets are transmitted at smaller fragment sizes and high priority packets are transmitted at larger fragment size, the corresponding G_W reaches its minimum value of 0.87 and 85 low priority slices are discarded.

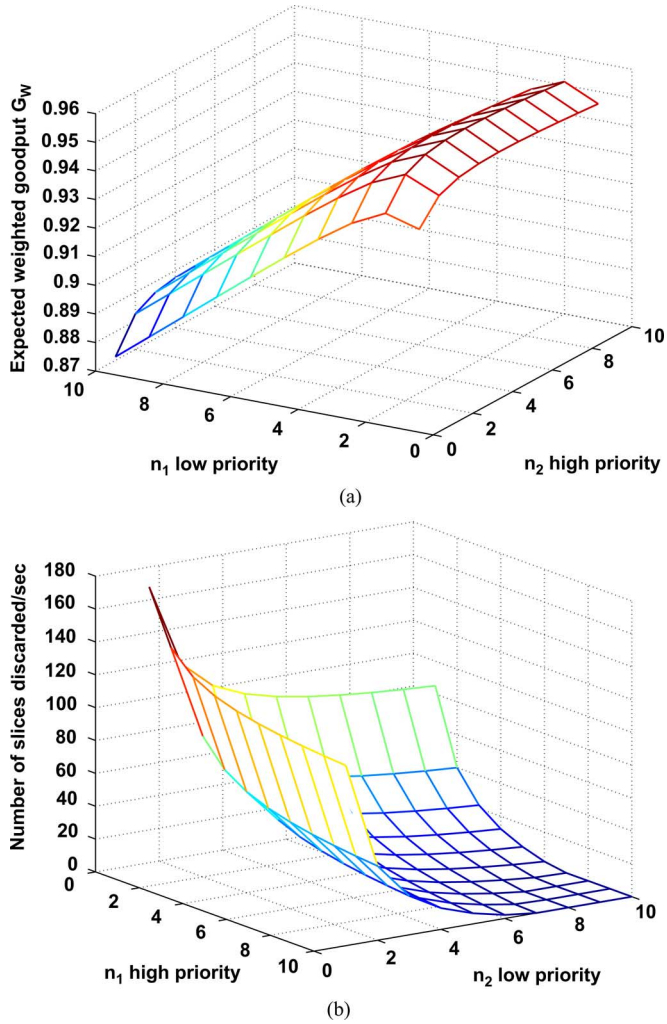


Fig. 5. (a) Expected weighted goodput and (b) slices discarded for Foreman video encoded at $R = 960$ Kbps and $x = 150$ bytes.

E. Branch-and-Bound (BnB) Optimization Using Interval Arithmetic Analysis

We used the BnB technique along with interval arithmetic analysis to solve the priority-aware expected weighted goodput optimization problem [11]. BnB is a global optimization technique used for non-convex problems, especially in discrete

$$\arg\max_{\bar{y}} G_W = \arg\max_{\bar{y}} \sum_{l_1} p(l_1) \begin{cases} \frac{w_1 n_1 x}{y_1 R} R_{CH}(1-p_b)^{y_1} = f_1(n_1);\ C_1 : \left(\frac{l_1}{n_1}\right) \ge \frac{R_{CH}}{y_1} \\ w_1(1-p_b)^{y_1} + \frac{\frac{w_2 n_2}{y_2}\left(R_{CH}-\left(\frac{l_1}{n_1}\right)y_1\right)(1-p_b)^{y_2}}{N-l_1} = f_2(n_1,n_2)\ ; \\ \qquad\qquad C_2 : \left(\frac{l_1}{n_1}\right) y_1 + \left(\frac{N-l_1}{n_2}\right) y_2 > R_{CH},\ \left(\frac{l_1}{n_1}\right) < \frac{R_{CH}}{y_1} \\ w_1(1-p_b)^{y_1} + w_2(1-p_b)^{y_2} = f_3(n_1,n_2);\ C_3 : \left(\frac{l_1}{n_1}\right) y_1 + \left(\frac{N-l_1}{n_2}\right) y_2 \le R_{CH} \end{cases} \tag{10}$$

and combinatorial optimization. The original domain of the optimization variables is divided into smaller sub-regions, and interval arithmetic analysis is performed in each sub-region to compute the lower and upper bounds. The interval arithmetic analysis uses inclusion functions $f_1(n_1)$, $f_2(n_1, n_2)$, and $f_3(n_1, n_2)$ derived from our main objective function in (10) to compute the bounds. Depending on the computed bounds, a decision is made on whether a sub-region is retained or pruned [11]–[13]. In (10), the number of slices in the high and low priority fragments (i.e., n_1, n_2) and the conditions C_1, C_2, and C_3 define the search region.

Each sub-region is defined by the lower and upper bounds of n_1 and n_2 as $[\underline{n}_1 \quad \bar{n}_1]$ and $[\underline{n}_2 \quad \bar{n}_2]$ and these are used to compute the bounds of G_W in that sub-region using the above inclusion functions. The lower and upper bounds of function $f_1(n_1)$ satisfying condition C_1 are derived as

$$\underline{f}_1 = \left(\frac{w_1 \underline{n}_1 x}{(\underline{n}_1 x + h) R} R_{CH} (1 - p_b)^{\bar{n}_1 x + h} \right)$$
$$\bar{f}_1 = \left(\frac{w_1 \bar{n}_1 x}{(\bar{n}_1 x + h) R} R_{CH} (1 - p_b)^{\underline{n}_1 x + h} \right).$$

This is because $f_1(n_1)$ in (10) can be expressed as a product of two functions, $A(n_1) = (w_1 n_1 x)/(n_1 x + h) R R_{CH}$ and $B(n_1) = (1 - p_b)^{(n_1 x + h)}$. Here $A(n_1)$ is minimum at $\underline{n}_1$ and $B(n_1)$ is minimum at $\bar{n}_1$ and both are positive. Therefore, the lower bound $\underline{f}_1 = A(n_1)|_{\underline{n}_1} \times B(n_1)|_{\bar{n}_1}$. Similarly $A(n_1)$ is maximum at $\bar{n}_1$ and $B(n_1)$ is maximum at $\underline{n}_1$ and both are positive. Therefore, the upper bound $\bar{f}_1 = A(n_1)|_{\bar{n}_1} \times B(n_1)|_{\underline{n}_1}$. Similarly the lower and upper bounds of functions $f_2(n_1, n_2)$ and $f_3(n_1, n_2)$ are also derived. The lower (LB) and upper (UB) bounds of the weighted goodput G_W in a sub-region are computed as the expected lower and upper bounds of the inclusion functions in that sub-region over queue length l_1 varying from $[0\, N]$.

In the BnB algorithm, the overall search region of the variables n_1 and n_2 and the sub-regions generated from them form a tree. The sub-regions contain the estimated LB and UB of the weighted expected goodput G_W. Depending on the values of these bounds, a decision is made to either retain or prune a sub-region. The following steps are applied iteratively until the maximum G_W value is obtained:

1) In the set of unexpanded sub-regions, find the maximum "LB", say $LB_{\max}$, and prune all those sub-regions whose UB is less than $LB_{\max}$.
2) From the set of unpruned and unexpanded sub-regions, select the sub-region with maximum "UB" and further spawn it to form two more sub-regions.
3) Modify the set of unexpanded sub-regions and repeat step 1.

The BnB algorithm reduces the number of times (i.e., 36 times for $R = 960$ Kbps, $p_b = 5 \times 10^{-5}$ and 150 byte slice size) the expected weighted goodput values have to be computed as compared to the exhaustive search case (i.e., 100 times for all combinations of $n_1 \in [1\ 10]$ and $n_2 \in [1\ 10]$).

Table I shows the optimal (n_1, n_2) values derived for different encoding rates and channel BERs. (n_1, n_2) remains the same at $R = 720$ Kbps for different channel BER's since sufficient bits are available to allocate headers to each fragment. At $R = 960$ Kbps and 1080 Kbps, the (n_1, n_2) decreases as the channel BER increases in order to increase the fragment success rate. Also as the encoding rate increases towards the channel transmission rate $R_{CH} = 1$ Mbps for a given BER, (n_1, n_2) increases in order to limit the increase in the number of slices discarded. The computation time for the BnB algorithm is directly dependent on the number of slices generated during every second. The exhaustive search takes 76 ms to determine the optimal point whereas the BnB algorithm takes 56 ms on a Core 2 Duo 2.6-GHz Intel processor with 4 GB of RAM. This value is obtained when the distribution of slices in the priority queues (l_1, l_2) is assumed unknown and we need to compute the expected G_W. However the encoder would compute the distribution of slices in the priority queues during each second. When this information is available, the BnB algorithm takes only 13 ms to compute the optimal point whereas the exhaustive search takes 28 ms.

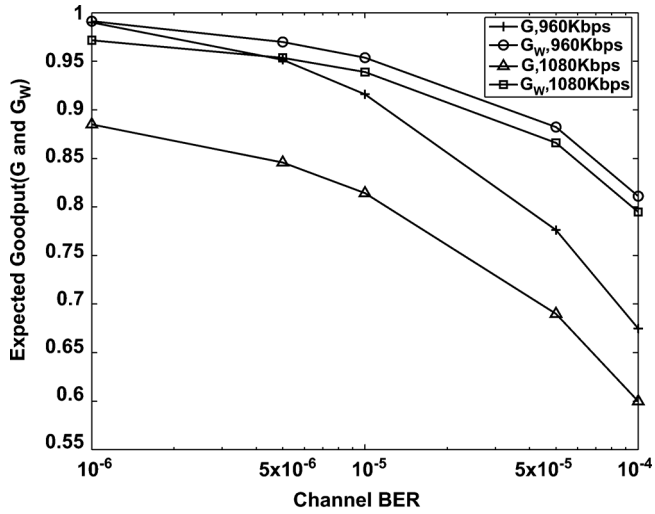
TABLE I
OPTIMAL NUMBER OF SLICES IN HIGH AND LOW PRIORITY FRAGMENTS

BER	720 Kbps	960 Kbps	1080 Kbps
10^{-6}	(1,1)	(5,10)	(6,10)
5×10^{-6}	(1,1)	(2,7)	(3,7)
10^{-5}	(1,1)	(2,5)	(2,5)
5×10^{-5}	(1,1)	(1,2)	(1,2)
10^{-4}	(1,1)	(1,2)	(1,2)

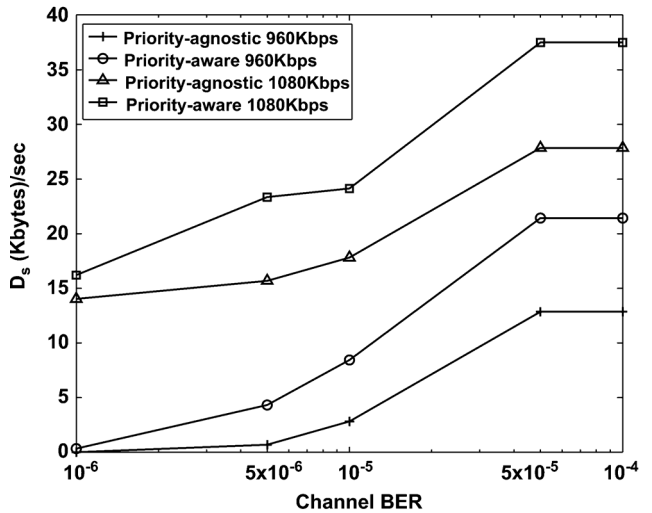
IV. PRIORITY-AGNOSTIC VERSUS PRIORITY-AWARE FRAGMENTATION

In this section, we compare the goodput and slice discard rates of the priority-agnostic and priority-aware schemes analytically, for two bit rates of 960 Kbps and 1080 Kbps over a 1024 Kbps channel. In the next section, we compare the performance of priority-agnostic and priority-aware schemes using simulations with video sequences. The maximum expected goodput G in priority-agnostic fragmentation and G_W in priority-aware fragmentation shown in Fig. 6(a) are computed using (2) and (10), respectively. The slice size is 150 bytes and the weights w_1 and w_2 for the priority-aware fragmentation are derived for the Foreman video sequence as discussed in Section III-D2. For 960 Kbps the weights $(w_1,\ w_2) = (0.89,\ 0.11)$ and for 1080 Kbps $(w_1,\ w_2) = (0.9,\ 0.1)$. The corresponding optimal fragment sizes for priority-aware fragmentation were listed in Table I. The number of Kbytes/sec discarded in Fig. 6(b) in order to achieve maximum goodput is computed using (4).

As shown in Fig. 6(a), priority-aware fragmentation achieves a goodput gain of 14% over priority-agnostic fragmentation at $R = 960$ Kbps and $p_b = 10^{-4}$, even when it discards 8.6 Kbytes of additional data per second as shown in Fig. 6(b). However, the performance of both fragmentation schemes starts converging as the channel BER decreases from 10^{-4} to 10^{-6}. This is expected because, when the channel is good enough, slices are not discarded [see Fig. 6(b)] and packets are not lost, and so fragmentation does not need to use priority information. In fact the priority-agnostic case uses 8 slices per fragment at BER of 10^{-6} with no slice data discarded whereas the priority-aware case uses 5 slices per fragment for high priority and 10 slices per fragment for low priority discarding only 320 bytes



(a)

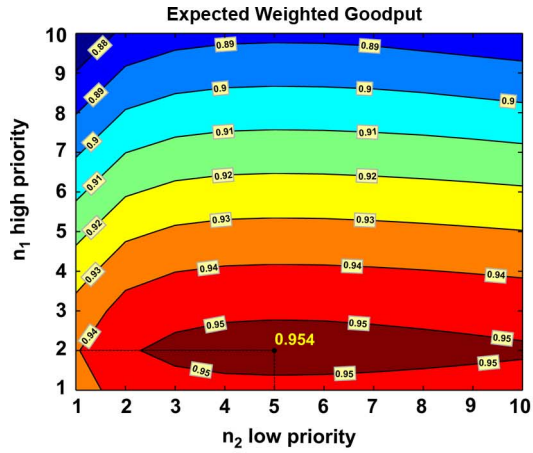


(b)

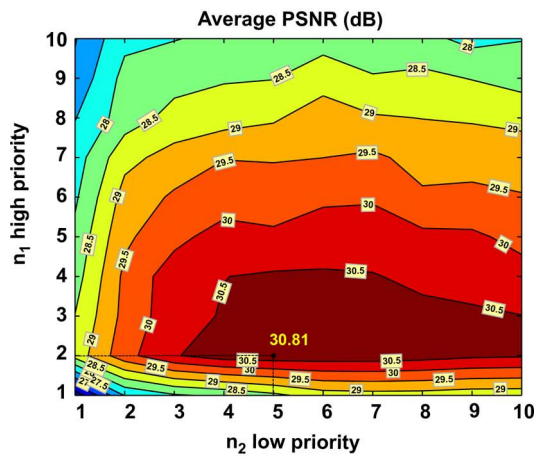
Fig. 6. (a) Expected goodput and (b) slice discard comparisons between priority-agnostic and priority-aware fragmentation.

of low priority slice data. Similarly, Fig. 6(a) shows that the priority-aware fragmentation achieves a goodput gain of 20% over priority-agnostic fragmentation at $R = 1080$ Kbps and $p_b = 10^{-4}$. We discard 9.6 Kbytes of additional data to achieve this gain as shown in Fig. 6(b). Unlike $R = 960$ Kbps, the priority-aware fragmentation achieves a goodput gain of 9% over priority-agnostic fragmentation at lower BER ($p_b = 10^{-6}$) for $R = 1080$ Kbps in Fig. 6(a). Note that 1) priority-aware fragmentation uses 6 and 10 slices per fragment for the high and low priority packets, respectively, as compared to 10 slices per fragment in the priority-agnostic case, and 2) discards slightly more low priority slice data (i.e., 16.2 Kbytes) as compared to discarding 14 Kbytes in priority-agnostic fragmentation as shown in Fig. 6(b).

Though the priority-aware fragmentation provides goodput gain by increasing the transmission reliability of higher priority packets, we have also investigated if this G_W gain corresponds to better video quality. We illustrate the results for Foreman encoded at 960 Kbps with a slice size of 150 bytes and transmitted over a 1 Mbps channel with BER = 10^{-5}. The expected goodput values for $n_1 \epsilon [1\ 10]$, $n_2 \epsilon [1\ 10]$ were shown



(a)



(b)

Fig. 7. (a) Expected weighted goodput G_W and (b) average PSNR at $R = 960$ Kbps, $R_{CH} = 1$ Mbps, and $p_b = 10^{-5}$.

in Fig. 5(a). We have also computed the video quality [in terms of average PSNR (dB)] for these values of n_1 and n_2. Fig. 7(a) and (b) illustrates the contour plots displaying the isolines (line connecting the points of equal value) of G_W and video PSNR. The distance between the isolines is equivalent to the gradient which represents the improvement in the corresponding values of G_W and video PSNR. The plots show that higher G_W generally corresponds to higher video PSNR values. For example, the dark red region in the contour plots represents the highest value of $G_W = 0.954$ for $(n_1, n_2) = (2, 5)$ which also corresponds to the highest PSNR of 30.81 dB. Similarly, the dark blue region in the contour plots represents the lowest value of $G_W = 0.87$ for $(n_1, n_2) = (10, 1)$ which also corresponds to the lowest PSNR of 25.65 dB. We have observed a similar behavior for other video sequences (e.g., CIF Silent video) and encoding rates.

V. Experimental Results and Discussion

A. Simulation Setup

This section evaluates the performance of the baseline system, and priority-agnostic and priority-aware fragmentation. The baseline system does not include slice prioritization and the packets are transmitted at the network limited MTU size of 1500 bytes. Two CIF resolution (352×288) video

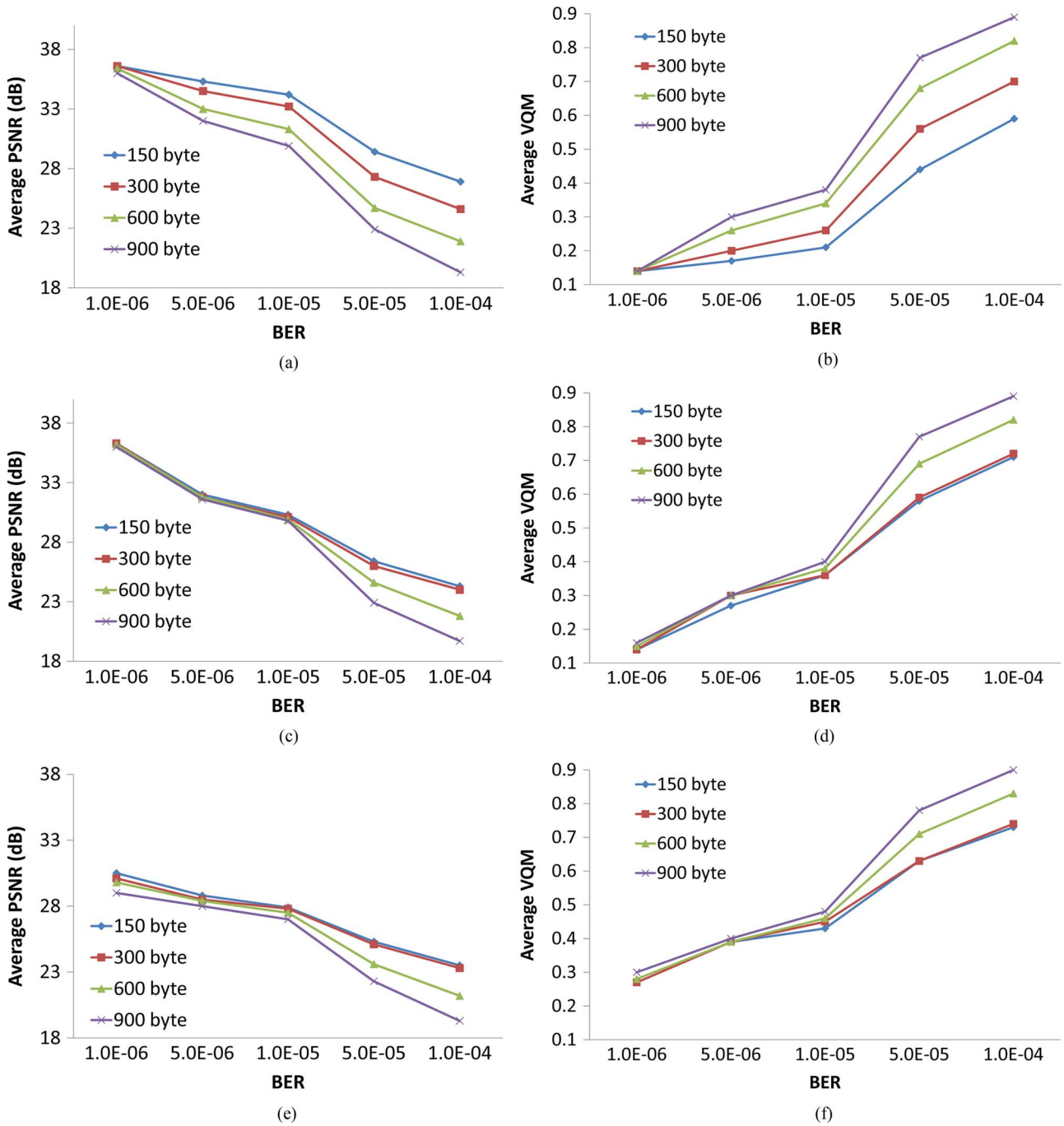


Fig. 8. Average PSNR achieved by priority-agnostic fragmentation for Foreman encoded at (a) 720 Kbps, (c) 960 Kbps, and (d) 1080 Kbps and corresponding average VQM at (b) 720 Kbps, (d) 960 Kbps, and (e) 1080 Kbps.

sequences Foreman and Silent are used in our experiments, where Silent has lower motion activity than Foreman. They are encoded using H.264/AVC JM 14.2 reference software [29] for a GOP length of 20 frames with GOP structure IDR B P B . . . P B IDR at 30 frames/sec and encoding rates of 720 Kbps, 960 Kbps, and 1080 Kbps. Slice sizes of 150, 300, 600, and 900 bytes are used and the slices are formed using dispersed mode FMO with two slice groups. Two reference frames are used for predicting the P and B frames, with error concealment enabled using temporal concealment and spatial interpolation. The error concealment in a frame depends on the frame type and the type of losses encountered. If the entire (IDR, P, or B) frame is lost, motion copy is performed wherein the motion vectors from the best suited previous reference frame are copied and all the macroblocks are predicted. If some slices of a predicted (P or B) frame are lost, the decoder verifies the availability of sufficient motion information for the lost macroblocks. If motion information is available, the motion copy is performed else co-located macroblocks of the previous reference frame are directly copied. If some slices of an IDR frame are lost, the corresponding macroblocks are concealed using spatial interpolation. Error concealment is enabled for all the schemes evaluated in this section. The channel transmission rate is 1 Mbps and the PHY and MAC layer header h is set to 50 bytes.

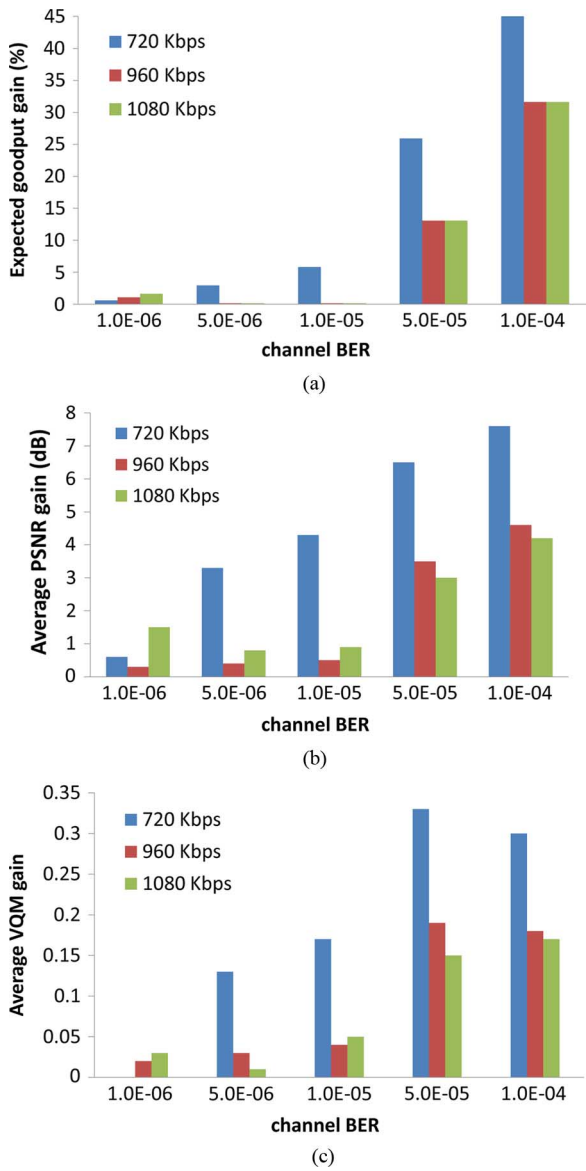


Fig. 9. Gains achieved by 150 byte slices over 900 byte slices in priority-agnostic fragmentation for Foreman, in terms of (a) expected goodput, (b) average PSNR, and (c) average VQM.

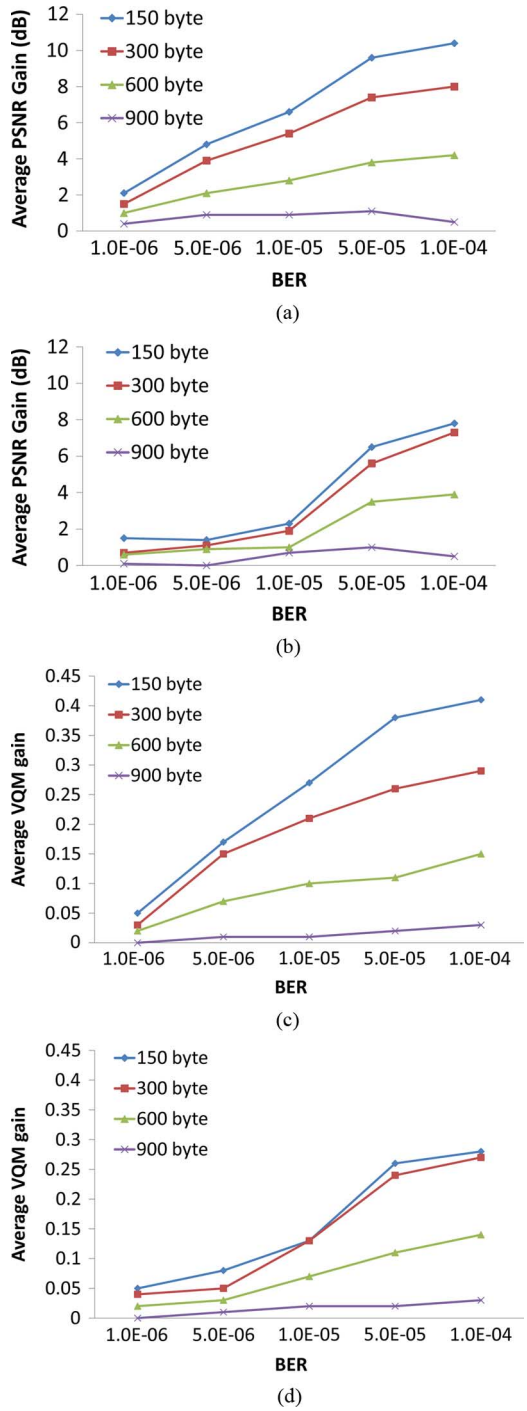


Fig. 10. Average PSNR gain achieved by priority-agnostic fragmentation over baseline system for Foreman encoded at (a) 720 Kbps and (b) 960 Kbps and corresponding average VQM gain at (c) 720 Kbps and (d) 960 Kbps.

Three video encoding rates are chosen to study the following cases 1) At 720 Kbps video bit rate, the 1 Mbps channel can support the fragment overhead; 2) at 960 Kbps, the channel may not have sufficient bits to accommodate fragment overhead without slice discard; and 3) at 1080 Kbps, the channel cannot even support the encoded video rate. For both test sequences, the PSNR increases with encoding rate as well as slice size. As the slice size increases, more macroblocks are encoded in each slice; this can more effectively exploit the spatial correlation in the neighboring macroblocks. A 900 byte slice provides a 0.4–0.5 dB PSNR gain compared to a 150 byte slice size. Similarly, when the encoding rate is increased from 720 Kbps to 960 Kbps, the PSNR increases by 1–1.5 dB. In addition to computing PSNR the subjective quality of the resultant videos is also evaluated using the perceptually based video quality metric (VQM) discussed in [34] and [35]. VQM is reported as a single number for the entire sequence and has a nominal output range from zero to one, where zero (one) represents no (maximum) perceived impairment.

B. Priority-Agnostic Fragmentation

Priority-agnostic packet fragmentation ignores the packet priorities and uses the optimal fragment size derived by maximizing the expected goodput G as discussed in Section III-D1. The average video PSNR and average VQM achieved by the priority-agnostic fragmentation for the Foreman sequence are shown in Fig. 8. As in the baseline system, the average PSNR

TABLE II
(A) PSNR AND (B) VQM GAINS OF PRIORITY-AWARE OVER PRIORITY-AGNOSTIC FRAGMENTATION WITH MODIFIED DROP-TAIL SLICE DISCARD FOR FOREMAN AT 720, 960, AND 1080 KBPS

BER	10^{-6}	5×10^{-6}	10^{-5}	5×10^{-5}	10^{-4}
150 byte	0.5, 0.3, 0.1	0.3, 0.9, 0.3	0.2, 1.1, 0.4	0.2, 0.4, 0.4	0.1, 1, 1.2
300 byte	0.5, 0.2, 0.5	1.1, 1.1, 0.4	1.2, 1.3, 0.3	1.7, 0.2, 0.4	2.1, 0.7, 1
600 byte	0.7, 0.2, 0.8	2.6, 1, 0.5	2.7, 0.9, 0.4	4.2, 1.3, 1.4	4.2, 2.7, 2.7
900 byte	1, 0.3, 1.4	3.1, 0.7, 0.3	4.2, 0.5, 0.3	5.1, 2.7, 2.5	5.9, 4.3, 4.2

(a)

BER	10^{-6}	5×10^{-6}	10^{-5}	5×10^{-5}	10^{-4}
150 byte	0.01, 0.01, 0.05	0.01, 0.03, 0.08	0.01, 0.08, 0.09	0.01, 0.08, 0.06	0.01, 0.09, 0.09
300 byte	0.01, 0.01, 0.05	0.03, 0.06, 0.08	0.05, 0.08, 0.1	0.1, 0.07, 0.05	0.1, 0.07, 0.07
600 byte	0.01, 0.01, 0.06	0.08, 0.06, 0.08	0.12, 0.06, 0.09	0.22, 0.11, 0.1	0.22, 0.16, 0.15
900 byte	0.02, 0.01, 0.07	0.12, 0.03, 0.07	0.2, 0.05, 0.09	0.29, 0.18, 0.16	0.25, 0.22, 0.2

(b)

in Fig. 8 decreases when the channel BER increases. For 720 Kbps video, the video quality is purely determined by the impact of channel errors as sufficient bits are available to allocate a header to each fragment. At high channel BER, the fragment success probability, fsr, expressed in (1), decreases, resulting in more errors. The deterioration in expected goodput G due to increasing channel BER was shown in Fig. 3(a). For higher video bit rates 960 and 1080 Kbps and larger slice sizes, the fragment sizes may be higher and some slices are also discarded to meet the channel bit rate constraint which leads to more video quality deterioration. For example, a video bit rate of 1080 Kbps requires 27 Kbytes of slice data to be discarded every second as shown in Fig. 4 in order to achieve a maximum expected goodput G of only 0.67 by transmitting 300 byte fragments as shown in Fig. 3(b). The fragments formed from smaller slice sizes, though, provide better PSNR performance as compared to fragments formed from larger slice sizes in Fig. 8. This is because smaller slice size allows a finer aggregation of video data into fragments. For example, each fragment contains eight and two 150 byte slices at BER of 10^{-6} and 10^{-4}, respectively, as compared to only one 900 byte slice.

The VQM plots illustrated in Fig. 8 agree with the trends observed in average PSNR values. A high level of perceived impairment can be observed at high channel BERs, high video bit rates, and large slice sizes. Fig. 9 compares the expected goodput, video PSNR, and VQM gains achieved by fragments formed from 150 byte slices over those formed from 900 byte slices for the Foreman video sequence. The gains generally increase with BER for each video bit rate. At 720 Kbps and BER of 10^{-4}, a large goodput gain of more than 45%, a PSNR gain of 7.6 dB, and VQM gain of 0.3 is achieved. Similarly 31% gain in goodput is achieved at 960 Kbps and 1080 Kbps video bit rates with corresponding PSNR gains of 4.6 dB and 4.2 dB and VQM gains of 0.18 and 0.17, respectively.

The PSNR and VQM gain achieved by the priority-agnostic fragmentation over the baseline system is shown in Fig. 10. Considerable gains are achieved for 150, 300, and 600 byte slice sizes due to enhanced fragment success rate in priority-agnostic fragmentation as compared to the baseline system. The gain for these slice sizes generally increases with BER for 720 and 960 Kbps video bit rate. Also the smaller slice sizes achieve larger PSNR gains as they can achieve finer aggregation of video data in fragments. For example, a PSNR gain of 10.4 dB is achieved for a 720 Kbps video encoded using 150 byte slices at a channel BER of 10^{-4}. For 900 byte slices, the gain is only up to 1 dB. At a low channel BER of 10^{-6}, different slice sizes achieve less than 2 dB gain, since very few fragments are corrupted by error. Fig. 10(c) and (d) shows that priority-agnostic fragmentation significantly improves the perceptual video quality as compared to the baseline transmission.

C. Priority-Aware Fragmentation

We adapt the fragment size to slice priorities as explained in Section III-D2, i.e., larger fragment size is used for the low priority slices, along with slice fragmentation. Priority-aware fragmentation thus maximizes the weighted expected goodput G_W by increasing the transmission reliability of high priority packets.

In order to discard the excess slices as discussed in Section III-D, we first use a modified drop-tail based slice discard scheme which first discards the lowest priority (i.e., S=4) slices, followed by S=3 and S=2 slices, for one second interval. Table II(a) and (b) shows the PSNR and VQM gains for the Foreman video achieved by the priority-aware fragmentation over the priority-agnostic fragmentation when both schemes use the modified drop-tail mechanism for slice discard.

In order to further increase the PSNR values as well as the corresponding gain achieved by the priority-aware fragmentation, we also propose a slice discard scheme for dispersed mode FMO with two slice groups as follows:

Step 1) Consider the B-frame starting from the end of the GOP and drop slices from Slice Group 1 (SG1) corresponding to the two lowest priorities (i.e., S=3 and 4).
Step 2) Repeat Step 1 for the next B-frame until slices from all the B-frames are discarded, as needed.
Step 3) Repeat Steps 1 and 2 for SG2 slices of B frames.
Step 4) Consider the P-frame starting from the end of GOP and discard the lowest priority (i.e., S=4) slices from SG1.
Step 5) Repeat Step 4 for the next P-frames.
Step 6) Drop the lowest priority (i.e., S=4) slices from SG1 of the IDR frame.
Step 7) Drop high priority slices (i.e., S=1 and 2) from SG1 and SG2 of B-frames starting from the end of the GOP.

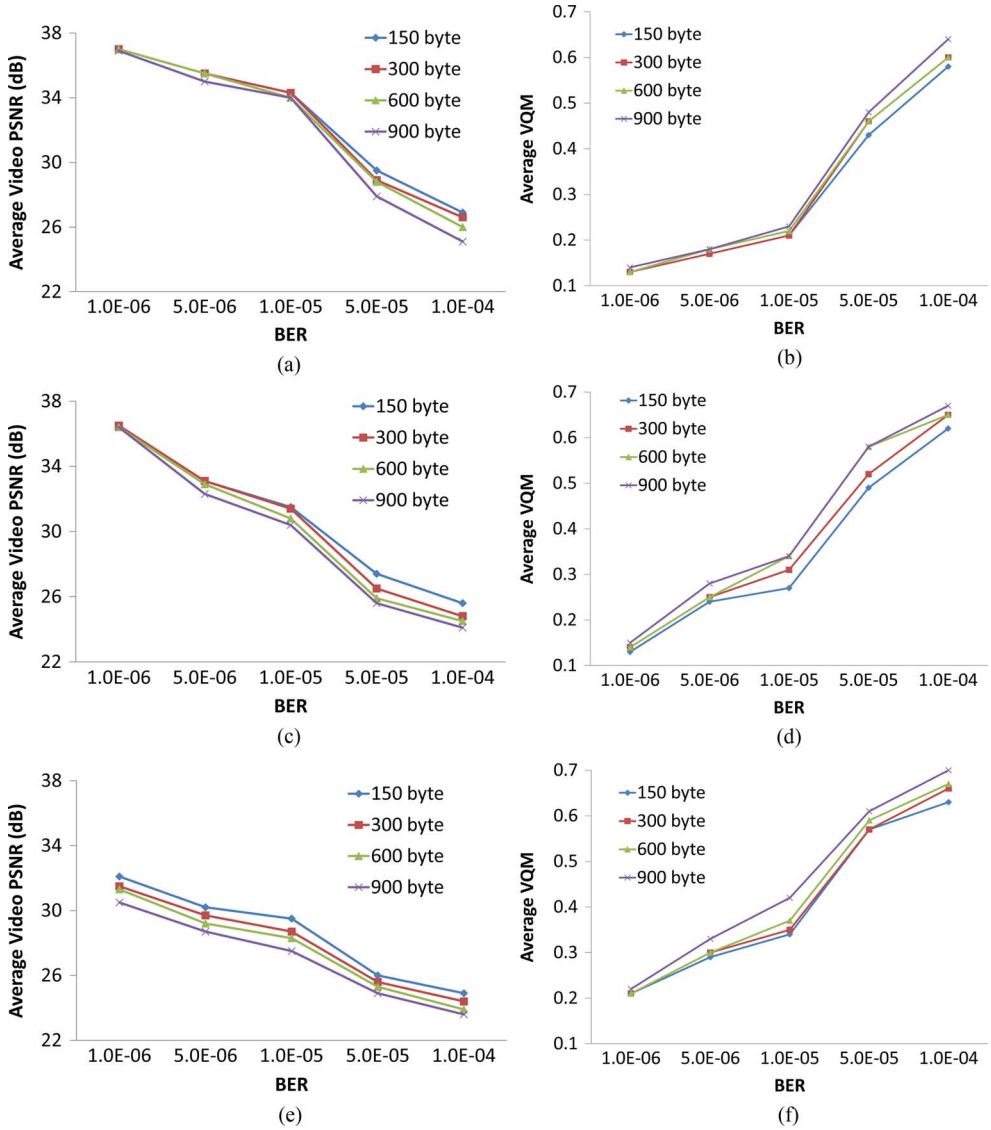


Fig. 11. Average PSNR achieved by priority-aware fragmentation for Foreman encoded at (a) 720 Kbps, (c) 960 Kbps, and (e) 1080 Kbps and corresponding average VQM at (b) 720 Kbps, (d) 960 Kbps, and (f) 1080 Kbps.

Since dispersed mode FMO is used, if some low priority slices from a slice group are discarded, the lost MBs will be concealed from spatially adjacent MB's belonging to the other slice group. Since error concealment may not be effective when spatially adjacent MBs are discarded, we discard the lowest priority slices from only one slice group in P and IDR frames. Since B-frame slices do not cause error propagation and can be effectively concealed, our scheme allows the discard of slices from both B-frame slice groups. Unlike the proposed slice discard scheme, the modified drop-tail based scheme does not consider the slice group, frame type and frame location information. Table III shows the additional gain, both in terms of PSNR and VQM, achieved by the proposed slice discard scheme over the modified drop-tail scheme in the priority-aware fragmentation. We have achieved similar gains for the Silent video sequence. Note that no additional gain is achieved for 720 Kbps video as no slices are discarded for this bit rate.

Fig. 11 shows the expected received video PSNR and VQM for priority-aware fragmentation with the proposed slice discard scheme for the Foreman video sequence. The variation of average video PSNR with BER and slice size is similar to our observations for the priority-agnostic fragmentation shown in Fig. 8. In particular, we notice in Fig. 11(a) and (c) that the use of slice fragmentation in priority-aware fragmentation has improved the PSNR performance of large slice sizes and narrowed the gap between large and smaller slices sizes as compared to

TABLE III
(A) PSNR AND (B) VQM GAIN DUE TO PROPOSED SLICE DISCARD FOR FOREMAN AT 960 KBPS (1080 KBPS)

BER	10^{-6}	5×10^{-6}	10^{-5}	5×10^{-5}	10^{-4}
150 byte	0.1 (1.7)	0.4 (1.3)	0.3 (1.4)	0.8 (0.5)	0.5 (0.4)
300 byte	0.2 (1.1)	0.4 (1.0)	0.2 (0.8)	0.5 (0.3)	0.3 (0.3)
600 byte	0.2 (0.9)	0.3 (0.5)	0.2 (0.6)	0.2 (0.5)	0.2 (0.2)
900 byte	0.3 (0.3)	0.2 (0.6)	0.3 (0.4)	0.2 (0.3)	0.3 (0.3)

(a)

BER	10^{-6}	5×10^{-6}	10^{-5}	5×10^{-5}	10^{-4}
150 byte	0.01 (0.04)	0.01 (0.03)	0.01 (0.03)	0.02 (0.01)	0.02 (0.01)
300 byte	0.01 (0.03)	0.01 (0.03)	0.01 (0.02)	0.01 (0.01)	0.01 (0.01)
600 byte	0.01 (0.02)	0.01 (0.01)	0.01 (0.02)	0.01 (0.01)	0.01 (0.01)
900 byte	0.01 (0.01)	0.01 (0.02)	0.01 (0.01)	0.01 (0.01)	0.01 (0.01)

(b)

TABLE IV
(A) PSNR AND (B) VQM GAINS OF PRIORITY-AWARE OVER PRIORITY-AGNOSTIC FRAGMENTATION (WITHOUT SLICE FRAGMENTATION) FOR FOREMAN AT 960 KBPS (1080 KBPS)

BER	10^{-6}	5×10^{-6}	10^{-5}	5×10^{-5}	10^{-4}
150 byte	0.3 (1.7)	1 (1.5)	1.3 (1.7)	1.2 (0.8)	1.4 (1.5)
300 byte	0.1 (1.5)	1 (1.2)	1 (0.8)	0.6 (0.6)	0.2 (0.6)
600 byte	0.1 (1.3)	0.8 (0.7)	0.9 (0.8)	0.2 (0.6)	0.2 (0.5)
900 byte	0.2 (0.7)	0.7 (0.6)	0.5 (0.6)	0.2 (0.5)	0.2 (0.7)

(a)

BER	10^{-6}	5×10^{-6}	10^{-5}	5×10^{-5}	10^{-4}
150 byte	0 (0.07)	0.05 (0.09)	0.05 (0.1)	0.08 (0.06)	0.08 (0.09)
300 byte	0 (0.06)	0.03 (0.09)	0.04 (0.09)	0.05 (0.05)	0.02 (0.02)
600 byte	0 (0.06)	0.02 (0.09)	0.03 (0.08)	0.02 (0.02)	0 (0.01)
900 byte	0 (0.05)	0.01 (0.06)	0.02 (0.05)	0.01 (0.01)	0 (0.01)

(b)

TABLE V
PSNR GAINS OF PRIORITY-AWARE OVER PRIORITY-AGNOSTIC FRAGMENTATION (WITHOUT SLICE FRAGMENTATION) FOR SILENT AT 960 KBPS (1080 KBPS)

BER	10^{-6}	5×10^{-6}	10^{-5}	5×10^{-5}	10^{-4}
150 byte	0.2 (2.1)	0.3 (2.5)	0.9 (2.3)	1.6 (0.8)	2 (2.1)
300 byte	0.2 (2.5)	0.7 (2.7)	0.8 (2.1)	1.1 (0.8)	0.8 (0.8)
600 byte	0.4 (2.1)	0.5 (2.3)	1.1 (2.1)	0.3 (0.6)	0.3 (0.4)
900 byte	0.2 (2.3)	0.1 (1.5)	0.4 (2)	0.3 (0.5)	0.3 (0.5)

(a)

BER	10^{-6}	5×10^{-6}	10^{-5}	5×10^{-5}	10^{-4}
150 byte	0 (0.07)	0.01 (0.09)	0.04 (0.09)	0.1 (0.05)	0.1 (0.1)
300 byte	0 (0.09)	0.03 (0.1)	0.03 (0.07)	0.04 (0.05)	0.03 (0.03)
600 byte	0 (0.07)	0.03 (0.1)	0.04 (0.08)	0.01 (0.02)	0.01 (0.01)
900 byte	0 (0.07)	0.01 (0.06)	0.01 (0.05)	0 (0.01)	0 (0.01)

(b)

the priority-agnostic fragmentation (in Fig. 8). It also has reduced the amount of visual impairment as measured by VQM compared to the priority-agnostic fragmentation [see Figs. 11(b) and (d)]. However as discussed in Section III-C, the loss of the first fragment containing the slice header causes the subsequent fragments of a packet to be discarded at the receiver even though they were successfully received. This still keeps the performance of large slices slightly inferior as compared to 150 byte slices when the channel error rate is high and the resulting fragment success rate is low. Slice fragmentation of a large slice also causes more slices to be discarded from the buffer when the video is encoded at high bit rates and results in adjacent slices of the frame to be dropped. This causes the large slices to underperform as compared to the 150 byte case even at low channel error rates at 1080 Kbps as shown in Fig. 11(e).

Tables IV and V show the video PSNR and VQM gains achieved by priority-aware over priority-agnostic fragmentation without using slice fragmentation for 960 Kbps Foreman and Silent video sequences. The corresponding gain for 1080 Kbps video is shown in brackets. The priority-aware fragmentation scheme adapts the fragment sizes to the individual packet priority levels which results in better received video quality. Note that both schemes have the same PSNR performance at a video bit rate of 720 Kbps because the margin of 304 Kbps for 1024 Kbps (i.e., 1 Mbps) channel bit rate is sufficient to transmit one slice per fragment. At a given channel BER, it is also observed that higher gain is achieved at smaller slice sizes. Increasing the slice size decreases the flexibility in choosing

TABLE VI
AVERAGE PSNR GAIN OF PRIORITY-AWARE (WITH SLICE FRAGMENTATION) OVER PRIORITY-AGNOSTIC FRAGMENTATION FOR FOREMAN (SILENT) AT (A) 720 KBPS, (B) 960 KBPS, AND (C) 1080 KBPS

BER	10^{-6}	5×10^{-6}	10^{-5}	5×10^{-5}	10^{-4}
150 byte	0.5 (0.4)	0.3 (0.2)	0.2 (0.1)	0.2 (0.1)	0.1 (0.2)
300 byte	0.5 (0.3)	1.1 (0.9)	1.2 (1)	1.7 (1.8)	2.1 (1.8)
600 byte	0.7 (0.6)	2.6 (2.1)	2.8 (2.5)	4.2 (5.3)	4.2 (5.2)
900 byte	1 (0.9)	3.1 (3.4)	4.2 (4)	5.1 (6.8)	5.9 (7.5)

(a)

BER	10^{-6}	5×10^{-6}	10^{-5}	5×10^{-5}	10^{-4}
150 byte	0.4 (0.7)	1.3 (1.3)	1.4 (1.7)	1.2 (1.7)	1.5 (2.2)
300 byte	0.4 (0.3)	1.5 (1.6)	1.5 (1.3)	0.7 (1.2)	1 (2)
600 byte	0.4 (0.6)	1.3 (0.9)	1.1 (1.2)	1.5 (2.4)	2.9 (4.2)
900 byte	0.6 (0.3)	0.9 (0.8)	0.8 (1.1)	2.9 (4.5)	4.6 (6.7)

(b)

BER	10^{-6}	5×10^{-6}	10^{-5}	5×10^{-5}	10^{-4}
150 byte	1.8 (1.9)	1.6 (2.5)	1.8 (2.2)	0.9 (0.6)	1.6 (2)
300 byte	1.6 (2.5)	1.4 (2.6)	1.1 (2.6)	0.7 (0.8)	1.3 (1.5)
600 byte	1.7 (2.1)	1 (2.2)	1 (2.1)	1.9 (2.1)	2.9 (3.7)
900 byte	1.7 (2.3)	0.9 (2)	0.7 (2.3)	2.8 (3.6)	4.5 (5.8)

(c)

TABLE VII
AVERAGE VQM GAIN OF PRIORITY-AWARE (WITH SLICE FRAGMENTATION) OVER PRIORITY-AGNOSTIC FRAGMENTATION FOR FOREMAN (SILENT) AT (A) 720 KBPS, (B) 960 KBPS, AND (C) 1080 KBPS

BER	10^{-6}	5×10^{-6}	10^{-5}	5×10^{-5}	10^{-4}
150 byte	0.01 (0.01)	0.01 (0.01)	0.01 (0.01)	0.01 (0.01)	0.01 (0.01)
300 byte	0.01 (0.01)	0.03 (0.04)	0.05 (0.04)	0.1 (0.13)	0.1 (0.11)
600 byte	0.01 (0.01)	0.08 (0.07)	0.12 (0.12)	0.22 (0.21)	0.22 (0.18)
900 byte	0.02 (0.03)	0.12 (0.13)	0.2 (0.18)	0.29 (0.31)	0.25 (0.28)

(a)

BER	10^{-6}	5×10^{-6}	10^{-5}	5×10^{-5}	10^{-4}
150 byte	0.02 (0.01)	0.04 (0.05)	0.09 (0.08)	0.1 (0.11)	0.11 (0.11)
300 byte	0.02 (0.01)	0.07 (0.06)	0.09 (0.09)	0.08 (0.09)	0.08 (0.1)
600 byte	0.02 (0.01)	0.07 (0.05)	0.07 (0.08)	0.12 (0.12)	0.17 (0.19)
900 byte	0.02 (0.01)	0.04 (0.03)	0.06 (0.06)	0.19 (0.19)	0.23 (0.26)

(b)

BER	10^{-6}	5×10^{-6}	10^{-5}	5×10^{-5}	10^{-4}
150 byte	0.09 (0.08)	0.11 (0.1)	0.12 (0.11)	0.07 (0.05)	0.1 (0.11)
300 byte	0.08 (0.08)	0.11 (0.11)	0.12 (0.1)	0.07 (0.05)	0.08 (0.09)
600 byte	0.08 (0.07)	0.09 (0.1)	0.11 (0.1)	0.11 (0.12)	0.16 (0.19)
900 byte	0.08 (0.09)	0.09 (0.1)	0.10 (0.09)	0.18 (0.18)	0.21 (0.24)

(c)

the fragment sizes as each fragment contains one or more slices in their entirety. For example, the fragment size can be either 600 bytes or 1200 bytes for a 600 byte slice size. Moreover a 900 byte slice allows us only 1 slice/fragment at 1500 bytes MTU. This restricts the PSNR gain that can be achieved by priority-aware over priority-agnostic fragmentation. A similar trend is also observed for the VQM gain in Tables IV(b) and V(b). Smaller slice sizes provide relatively more improvement in the perceptual video quality at the receiver.

This inflexibility can be mitigated by allowing the fragmentation of slices by suitably modifying the decoder to handle partial slice data. This further increases the PSNR gain that can be achieved by priority-aware fragmentation scheme as discussed below. Table VI shows the PSNR gains achieved by priority-aware (with slice fragmentation) over priority-agnostic fragmentation for Foreman and Silent sequences at 720, 960, and 1080 Kbps. Table VII shows the VQM gains achieved corresponding to the PSNR gains in Table VI. The priority-aware

(a)

(b)

(c)

Fig. 12. 126th frame of Foreman encoded at 720 Kbps and BER of 5×10^{-5} in (a) Baseline system: Average PSNR = 20.3 dB, Average VQM = 0.83, (b) Priority-agnostic fragmentation: Average PSNR = 25.2 dB, Average VQM = 0.68, and (c) Priority-aware fragmentation: Average PSNR = 29.9 dB, Average VQM = 0.46.

(a)

(b)

(c)

Fig. 13. 126th frame of Silent encoded at 720 Kbps and BER of 5×10^{-5} in (a) Baseline system: Average PSNR = 22 dB, Average VQM = 0.81, (b) Priority-agnostic fragmentation: Average PSNR = 28.5 dB, Average VQM = 0.68, and (c) Priority-aware fragmentation: Average PSNR = 33.3 dB Average VQM = 0.47.

fragmentation achieves better PSNR and VQM values as compared to the priority-agnostic fragmentation for all the slice sizes and BERs. For Foreman, a maximum PSNR gain of 5.9 dB and corresponding VQM gain of 0.25 is achieved at 720 Kbps and 900 byte slice size at channel BER of 10^{-4} as shown in Tables VI(a) and VII(a), respectively. Similarly, 4.6 dB PSNR gain and corresponding VQM gain of 0.23 is achieved for a video encoded at 960 Kbps for the same slice size and channel

BER in Tables VI(b) and VII(b), respectively. The maximum gains for the Silent video sequence are 7.5 dB at 720 Kbps [Table VI(a)] and 6.7 dB at 960 Kbps [Table VI(b)] for 900 byte slice size and channel BER of 10^{-4}. Also, the corresponding VQM gains are 0.28 in Table VII(a) and 0.26 in Table VII(b). Similar gains are also observed when the video encoding rate (1080 Kbps) exceeds the channel bit rate of 1 Mbps.

Fig. 12 shows the 126th frame of Foreman encoded at 720 Kbps using a slice size of 600 bytes at a BER of 5×10^{-5}. Fig. 13 shows the 126th frame of Silent using these same specifications. The average VQM values are also shown in Figs. 12 and 13.

VI. Conclusion

An efficient priority-aware MAC layer packet fragmentation scheme was proposed to improve the quality of pre-encoded H.264 bitstreams transmitted over unreliable error-prone wireless links. The optimal fragment sizes for the respective priority levels are derived using the branch-and-bound technique combined with multidimensional arithmetic interval methods. Further slice fragmentation was used to boost the PSNR and VQM quality performance of videos, especially encoded at large slice sizes. The performance in terms of expected received video quality was compared to 1) the traditional baseline model where each packet is transmitted onto the channel at the network limited MTU size, and 2) priority-agnostic fragmentation using a single optimal fragment size. It was shown that maximizing the expected goodput or expected weighted goodput provides large gains in received video quality.

The cross-layer priority information exchange between the video layer and MAC layer allowed us to design a slice discard scheme which enabled us to reduce the impact of lost slices on the received video quality. The fact that these gains are achieved without error correction techniques makes it all the more interesting to evaluate the above strategies using unequal error protection for different priority levels.

References

[1] T. Wiegand, G. J. Sullivan, G. Bjøntegaard, 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.

[2] S. Kumar, L. Xu, M. K. Mandal, and S. Panchanathan, "Error resiliency schemes in H.264/AVC standard," *Elsevier J. Vis. Commun. Image Represent., special issue on Emerging H.264/AVC Video Coding Standard*, vol. 17, no. 2, pp. 183–185, Apr. 2006.

[3] T. Stockhammer, M. M. Hannuksela, and T. Wiegand, "H.264/AVC in wireless environments," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 657–673, Jul. 2003.

[4] A. S. Tanenbaum, *Computer Networks*. Englewood Cliffs, NJ: Prentice Hall, 2002.

[5] B. A. Forouzan, *Data Communications and Networking*, ser. Forouzan Networking Series. New York: McGraw-Hill, 2007.

[6] S. Shakkotai, T. Rappaport, and P. Karlsson, "Cross-layer design for wireless networks," *IEEE Commun. Mag.*, vol. 41, no. 10, pp. 74–80, Oct. 2003.

[7] M. van der Schaar and N. S. Shankar, "Cross-layer wireless multimedia transmission: Challenges, principles and new paradigms," *IEEE Wireless Commun. Mag.*, vol. 12, no. 4, pp. 50–58, Aug. 2005.

[8] E. Setton, T. Yoo, X. Zhu, A. Goldsmith, and B. Girod, "Cross-layer design of ad hoc networks for real-time video streaming," *IEEE Wireless Commun.*, vol. 12, no. 4, pp. 50–58, 2005.

[9] Y. Sun, I. Sheriff, E. M. Belding-Royer, and K. C. Almeroth, "An experimental study of multimedia traffic performance in mesh networks," *ACM WiTMeMo*, pp. 25–30, 2005.

[10] Y. Wang, K. Yu, Y. Liu, and H. Zhang, "Effects of MAC retransmission on TCP performance in IEEE 802.11 based ad-hoc networks," in *Proc. 59th IEEE VTC*, Feb. 2004, vol. 4, pp. 2205–2209.

[11] G. Alefeld and G. Mayer, "Interval analysis: Theory and applications," *Elsevier J. Computat. Appl. Math.*, vol. 121, pp. 421–464, Aug. 1999.

[12] K. Ichida and Y. Fujii, "An interval arithmetic method for global optimization," *Springer-Verlag J. Comput.*, vol. 23, pp. 85–97, Aug. 1979.

[13] H. Munack, "On global optimization using interval arithmetic," *Springer-Verlag J. Comput.*, vol. 48, pp. 319–336, Apr. 1992.

[14] K. K. R. Kambhatla, S. Kumar, and P. Cosman, "Prioritized packet fragmentation for H.264 video," in *Proc. IEEE Int. Conf. Image Processing*, Sep. 11–14, 2011, pp. 3233–3236.

[15] J. Yin, X. Wang, and D. P. Agrawal, "Optimal packet size in error-prone channel for IEEE 802.11 distributed coordination function," in *Proc. IEEE WCNC*, 2004, pp. 1654–1659.

[16] B. S. Kim, Y. Fang, and T. F. Wong, "Throughput enhancement through dynamic fragmentation in wireless LANs," *IEEE Trans. Veh. Technol.*, vol. 54, no. 4, pp. 1415–1425, Jul. 2005.

[17] F. Zheng and J. Nelson, "Cross-layer adaptive design for the frame length of IEEE 802.11 networks," in *Proc. IEEE Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks, WiOPT*, 2008, pp. 437–442.

[18] P. R. Jelenkovic and J. Tian, "Dynamic packet fragmentation for wireless channels with failures," in *Proc. ACM MobiHoc*, May 2008, pp. 73–82.

[19] X. Wang, J. Yin, and D. P. Agrawal, "Analysis and optimization of the energy efficiency in the 802.11 DCF," *Springer Sci. J. Mobile Netw. Appl.*, vol. 11, pp. 279–286, 2006.

[20] P. Lettieri and M. B. Srivastava, "Adaptive frame length control for improving wireless link throughput, range and energy efficiency," in *Proc. IEEE INFOCOM*, 1998, vol. 2, pp. 564–571.

[21] D. Rajan and C. Poellabauer, "Adaptive fragmentation for latency control and energy management in wireless real time environments," in *Proc. IEEE Int. Conf. Wireless Algorithms, Systems and Applications, WASA*, 2007, pp. 158–168.

[22] D. Qiao, S. Choi, and K. G. Shin, "Goodput analysis and link adaptation for IEEE 802.11a wireless LANs," *IEEE Trans. Mobile Comput.*, vol. 1, no. 4, pp. 278–292, Dec. 2002.

[23] Y. Chang, B. Kwon, and J. A. Copeland, "Dynamic optimal fragmentation for goodput enhancement in WLANs," in *Proc. IEEE Int. Conf. Testbeds and Research Infrastructure for the Development of Networks and Communities, TridentCom*, 2007, pp. 1–9.

[24] U. Horn, K. Stuhlmuller, M. Link, and B. Girod, "Robust internet video transmission based on scalable coding and unequal error protection," *Image Commun., special issue on Real-time video over the Internet*, vol. 15, no. 1–2, pp. 77–94, Sep. 1999.

[25] Y. P. Fallah, K. Darrell, S. Avideh, K. Faizal, and P. Nasiopoulos, "A cross layer optimization mechanism to improve H.264 video transmission over WLANs," in *Proc. IEEE CCNC*, 2007, pp. 875–879.

[26] A. T. Connie, P. Nasiopoulos, V. C. M. Leung, and Y. P. Fallah, "Video packetization techniques for enhancing H.264 video transmission over 3G networks," in *Proc. IEEE CCNC*, 2008, pp. 800–804.

[27] A. Ksentini and M. Naimi, "Toward an improvement of H.264 video transmission over IEEE 802.11e through a cross layer architecture," *IEEE Commun. Mag.*, vol. 44, no. 1, pp. 107–114, Jan. 2006.

[28] Y. P. Fallah, P. Nasiopoulos, and H. Alnuweiri, "Efficient transmission of H.264 video over multirate IEEE 802.11e WLANs," *EURASIP J. Wireless Commun. Netw.*, pp. 1–14, Jan. 2008.

[29] *H.264/AVC Reference Software JM14.2*, ISO/IEC Std. [Online]. Available: http://iphome.hhi.de/suehring/tml/download/.

[30] *RoHC: Robust header Compression*, RFC:3095 Std., Jul. 2001.

[31] S. Wenger, M. M. Hannuksela, T. Stockhammer, M. Westerlund, and D. Singer, *RFC 3894—RTP Payload Format for H.264 Video,* Network Working Group Std., 2005.

[32] L. Superiori, O. Nemethova, and M. Rupp, "Performance of a H.264/AVC error detection algorithm based on syntax analysis," in *Proc. Int. Conf. Advances in Mobile Computing and Multimedia*, Dec. 2006.

[33] S. Kumar, A. Janarthanan, M. M. Shakeel, S. Maroo, J. D. Matyjas, and M. Medley, "Robust H.264/AVC video coding with priority classification, adaptive NALU size and fragmentation," in *Proc. IEEE MILCOM*, 2009, pp. 1702–1707.

[34] S. Wolf and M. H. Pinson, Video Quality Measurement Techniques, National Telecommunications and Information Administration (NTIA), U.S. Department of Commerce, 2002.

[35] M. H. Pinson and S. Wolf, "A new standardization method for objectively measuring video quality," *IEEE Trans. Broadcast.*, vol. 50, no. 3, pp. 312–322, 2004.

Kashyap K. R. Kambhatla (S'06) received the B.Tech. degree in electronics and communication engineering from Jawaharlal Nehru Technological University, Hyderabad, India, in 2004 and the M.S. degree in electrical and computer engineering from Clarkson University, Potsdam, NY, in 2006. He is currently pursuing the Ph.D. degree in electrical and computer engineering jointly awarded by the College of Engineering, San Diego State University, and Jacob's School of Engineering, University of California, San Diego.

During the summer of 2006, he worked on video telephony in 3G EV-DO Rev. A and B cellular networks as a research intern in the Next-Gen Wireless Technology Group at Sprint Advanced Technology Labs, Burlingame, San Francisco, CA. His research interest is in the area of real-time cross-layer quality-of-service (QoS) aware video streaming over ad-hoc wireless communication networks through packet size and physical layer FEC code rate adaptation. He also works on network optimization problems and developing efficient medium access control protocols for reliable video transmission.

Mr. Kambhatla is a member of Phi Kappa Phi.

Sunil Kumar (M'98–SM'04) received the B.E. degree in electronics engineering from S.V. National Institute of Technology, Surat, India, in 1988 and the M.E. and Ph.D. degrees in electrical and electronics engineering from Birla Institute of Technology and Science, Pilani, India, in 1992 and 1997, respectively.

From 1997–2002, he was a postdoctoral fellow and adjunct faculty in the Electrical Engineering Systems Department at the University of Southern California, Los Angeles. From 2002 to 2006, he was an Assistant Professor in Electrical and Computer Engineering at Clarkson University, Potsdam, NY. Since 2006, he has been an Associate Professor and Thomas G. Pine Faculty Fellow in the Electrical and Computer Engineering Department at the San Diego State University, San Diego, CA. His research interests include QoS-aware cross layer wireless network protocols and robust multimedia compression techniques, including H.264. He has published more than 100 research articles in refereed journals and conferences, including two book/book chapters. His research has been supported by NSF, U.S. Air Force Research Lab, Department of Energy, California Energy Commission, and industry.

Dr. Kumar was a guest editor of the April 2006 special issue of *Elsevier Journal of Visual Communication and Image Representation* on "Emerging H.264/AVC Video Coding Standard". He has organized workshops and technical sessions in several international conferences.

Seethal Paluri (S'07) received the B.S. degree in electrical and computer engineering from the University of the West Indies, Trinidad and Tobago, in 2005 and the M.S. degree in electrical engineering from the University of Southern California, Los Angeles, in 2008. She is currently pursuing the Ph.D. degree in computational sciences with major in mathematics and minor in electrical engineering, jointly awarded by San Diego State University and Claremont Graduate University.

Her research interests are in the areas of error resilient image and video compression, error concealment techniques, and region of interest (ROI) based coding. She also works on developing statistical modeling techniques for video quality prediction and prioritization for reliable real-time quality-of-service (QoS) aware streaming over wireless networks.

Pamela C. Cosman (S'88–M'93–SM'00–F'08) received the B.S. degree with Honor in electrical engineering from the California Institute of Technology, Pasadena, in 1987 and the M.S. and Ph.D. degrees in electrical engineering from Stanford University, Stanford, CA, in 1989 and 1993, respectively.

She was an NSF postdoctoral fellow at Stanford University and a Visiting Professor at the University of Minnesota during 1993–1995. In 1995, she joined the faculty of the Department of Electrical and Computer Engineering at the University of California, San Diego, where she is currently a Professor and Vice Chair. She was the Director of the Center for Wireless Communications from 2006 to 2008. Her research interests are in the areas of image and video compression and processing, and wireless communications.

Dr. Cosman is the recipient of the ECE Departmental Graduate Teaching Award, a Career Award from the National Science Foundation, a Powell Faculty Fellowship, and a Globecom 2008 Best Paper Award. She was a guest editor of the June 2000 special issue of the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS on "Error-resilient image and video coding" and was the Technical Program Chair of the 1998 Information Theory Workshop in San Diego. She was an associate editor of the IEEE COMMUNICATIONS LETTERS (1998–2001) and an associate editor of the IEEE SIGNAL PROCESSING LETTERS (2001–2005). She was the Editor-in-Chief (2006–2009) as well as a Senior Editor (2003–2005, 2010–present) of the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS. She is a member of Tau Beta Pi and Sigma Xi.