Rate-Distortion Optimized Slicing, Packetization and Coding for Error Resilient Video Transmission*

Enrico Masala[1], Hua Yang[2], Kenneth Rose[2] and Juan Carlos De Martin[3]
[1]Dip. Automatica e Informatica/[3]IEIIT-CNR, Politecnico di Torino,
c.so Duca degli Abruzzi 24, 10129 Torino – Italy
[2]ECE Department, University of California, Santa Barbara, CA 93106
Phone: +39-011-564-5421 Fax: +39-011-564-7099
E-mail: {masala,demartin}@polito.it, {hua,rose}@ece.ucsb.edu

Abstract

This paper presents an algorithm to optimize the tradeoff between rate and expected end-to-end distortion of a video sequence transmitted over a packet network. The approach optimizes the source coding parameters, slicing, network QoS class selection and/or error control coding parameters, and accounts for the effects of compression, packetization, error propagation, and concealment at the decoder. It builds on, and substantially extends the applicability of, the recursive optimal per-pixel estimate (ROPE) technique for end-to-end distortion estimation. A trellis-based algorithm is introduced in order to overcome macroblock interdependencies in the estimation procedure, and allow adaptive slicing. Moreover, we propose a complementary packetization scheme to efficiently arrange the slices into packets for FEC protection while minimizing rate loss due to padding. Simulations demonstrate consistent gains over currently used techniques.

1 Introduction

A fundamental problem in the delivery of video signals over packet networks is that of optimizing the quality perceived by the user, while accounting for all relevant parameters including the effects of compression, packet loss, error propagation, etc. Many different techniques have been proposed to reduce the degradation caused by the unreliability of the network [1]. Examples include various error resilience tools to optimize encoder decisions, channel error control schemes such as FEC and ARQ, as well as error concealment techniques to enhance the signal at the decoder. Better performance is naturally expected from a combination of such techniques. However, the number of parameters involved, and their

*This work was supported in part by the NSF under grants EIA-9986057 and EIA-0080134, the University of California MICRO program, Dolby Laboratories, Inc., Lucent Technologies, Inc., Mindspeed Technologies, Qualcomm, Inc., and CERCOM — Center for Multimedia Radio Communications, http://www.cercom.polito.it

intricate interdependencies present a formidable optimization problem. A typical remedy has been to only consider a small subset of the parameters for simultaneous optimization, as will be further summarized below.

Crucial to many error-resilient transmission methods is accurate estimation at the transmitter of end-to-end distortion. The recursive per-pixel optimal estimate (ROPE) [2] is an optimal low-complexity distortion estimate that naturally accounts for most factors affecting the quality at the receiver. In that work, ROPE was specifically applied to optimize intra/inter mode selection decisions. The work herein takes ROPE as its starting point and extends its applicability to optimize a comprehensive set of coding and networking parameters, including optimization of *adaptive slicing*, a paradigm that was introduced in [3].

To briefly review some of the known attacks on partial coding parameter optimization we note again that [2] seeks optimal intra/inter mode selection and quantization. In [4], the source coding parameters are optimized together with the choice of the best DiffServ classes (for transmission over a DiffServ network). For simplicity, the source coding parameters are considered constant for all macroblocks in a given packet, and the subdivision of the frame into packets is fixed in advance. The optimization of video transmission over a packet network, in conjunction with FEC codes, is examined in [5]. The main limitations are simplifying assumptions on the packetization scheme and the use of a single FEC code for the whole frame. In a very recent paper [6] the authors addressed the above shortcomings, but the subdivision of the frame into slices is nevertheless assumed to be fixed.

We present a rate-distortion optimized solution that seeks the best encoding parameters and protection level for each macroblock, together with *the best slicing for each frame*. The algorithm automatically groups together macroblocks that need similar protection levels, while explicitly taking into account the overhead implied by the creation of additional slices. Moreover, a novel packetization scheme to organize the slices into different packets and efficiently protect them by FEC codes is presented.

The paper is organized as follows. In Section 2 the problem of distortion minimization is reformulated, while accounting for the constraints imposed by the slice creation process, and is then solved by a proposed trellis-based technique. Section 3 presents an efficient algorithm to packetize the slices according to the assigned protection level. Section 4 describes the experimental setup and provides simulation results comparing the approach to current techniques. Conclusions are presented in Section 5.

2 Problem Formulation and Trellis-Based Solution

Let N be the number of macroblocks in a frame, and let m_i denote the set of source coding parameters and protection level assigned to the macroblock i:

$$m_i = (q_i, t_i, l_i) \tag{1}$$

where q_i is the quantization parameter, t_i specifies whether prediction is used (intra or inter), and l_i represents the protection level. Let $r(q_i, t_i)$ be the source coding rate for macroblock i, including any slice or picture header overhead, and let $E[d(m_i)]$ be the expected distortion computed by the ROPE method [2]. The protection level l_i determines the effective error probability p experienced by the system and used by ROPE in its calculations. If

the transmission is over a DiffServ network, then each l_i corresponds to a DiffServ class and $p = p_{l_i}$. If FEC protection is employed instead, such as a Maximum Distance Separable (MDS) code, then p_{l_i} is computed from the actual channel loss probability p_{ch} as:

$$p_{l_i} = \sum_{j=n-k+1}^{n} \frac{j}{n} \binom{n}{j} p_{ch}^{j} (1 - p_{ch})^{n-j} \tag{2}$$

where $n = n_{l_i}$ and $k = k_{l_i}$ are the parameters of the FEC code associated with protection level l_i.

Our objective is to minimize the expected end-to-end distortion. The frame encoding mode is given by the set of macroblock-specific modes $M = \{m_1, ..., m_N\}$ which determine their coding parameters and protection levels. The expected distortion $E[D(M)]$ of the frame is given by

$$E[D(M)] = \sum_{i=1}^{N} E[d(m_i)]. \tag{3}$$

In a DiffServ network, each class is identified by its loss probability p_{l_j} and the cost per bit c_{l_j}. The frame encoding mode problem is formulated as:

$$\min_{M} E[D(M)] \text{ subject to } C(M) < C_{max} \tag{4}$$

where

$$C(M) = \sum_{i=1}^{N} r(q_i, t_i) \cdot c_{l_i}. \tag{5}$$

If the video sequence is transmitted over a regular (single priority) lossy packet network and protected by FEC codes, the problem can be stated as:

$$\min_{M} E[D(M)] \text{ subject to } R(M) < R_{max} \tag{6}$$

where

$$R(M) = \sum_{i=1}^{N} r(q_i, t_i) \frac{n_{l_i}}{k_{l_i}}. \tag{7}$$

The above constrained minimization problems (4) and (6) are naturally recast in the standard Lagrangian formulation. For example, Equation (6) can be rewritten as:

$$\min_{M} J(M) = \min_{M} E[D(M)] + \lambda \cdot R(M). \tag{8}$$

The solution to (8) is not trivial because of the interdependencies between the macroblocks in a frame. In particular, the same protection level must be shared by the entire content of a packet. Assuming that slices are not split into different packets, we conclude that all macroblocks belonging to the same slice must be assigned the same protection level. In order to assign a different protection level to a macroblock we must generate a new slice and account for the overhead of its header. Moreover, the quantization parameter is encoded differentially with respect to its predecessor in the slice, and the same applies to motion vector prediction, that we assume to be dependent only on previous macroblocks in the same slice.

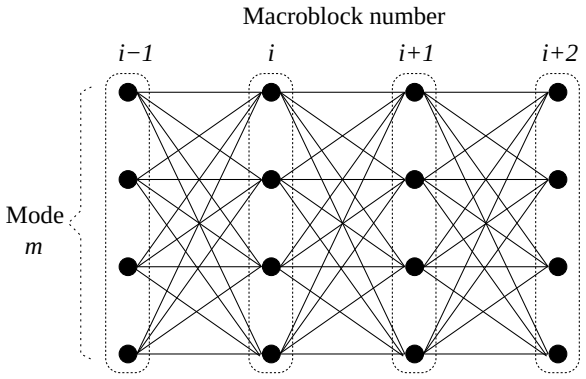

Figure 1: Trellis used to find the optimal coding mode for all the macroblocks in the frame.

We propose to solve the minimization problem using a trellis, shown in Figure 1, where the metric is the Lagrangian cost J, and each node at stage i represents a particular choice of the coding mode of macroblock i. For the first macroblock, the Lagrangian cost is computed for all possible parameter combinations or mode choices for m_1. Then, for each node at stage i, the algorithm evaluates all the paths leading to that node from any admissible mode m_{i-1} for the previous macroblock, and stores the one that produces the minimum Lagrangian cost:

$$J_t(m_i) = \min_{m_{i-1}}\{J_t(m_{i-1}) + [E[d(m_i)] + \lambda \cdot r(m_i, m_{i-1})]\} \tag{9}$$

Note that the rate $r(m_i, m_{i-1})$ of the i-th macroblock depends not only on m_i, but also on the choices for the previous macroblock m_{i-1}, due to the dependencies explained earlier. A slice header is inserted either if the protection level in m_i differs from the one chosen in m_{i-1} or if a larger variation in the quantization parameter is advantageous, but disallowed by the differential encoding rule. In either case, the additional rate due to the slice header is accounted for by $r(m_i, m_{i-1})$. The winning node m_N in the last stage, corresponding to the last macroblock in the frame, determines the optimal path in the trellis and the solution M for the whole frame, including its subdivision into slices.

When a macroblock is coded as skipped, the previous node is restricted to have the same quantization level. The q value of the current node represents the value maintained at the decoder to interpret the next differential encoding, rather than the one used to encode that macroblock. A variation of q is still allowed in case of a skipped macroblock, at the expense of encoding a zero motion vector and zero coefficients, thus keeping the system standard compliant. Large variations of q are allowed too, with the insertion of a new slice header, because they could lead to savings in the subsequent macroblocks.

The trellis approach offers considerable computational savings. First, the known benefits of dynamic programming ensure that the complexity only grows linearly with the number of macroblocks or stages. Note also that transformation operations, as well as quantization and VLC coding results are shared among various nodes and need not be repeated for all nodes. Finally, note that operations that are not shared can be carried out in parallel.

If the subdivision of the frame into slices is predefined, the proposed method is still applicable. In this case the optimization can be performed in two separate steps, optimizing

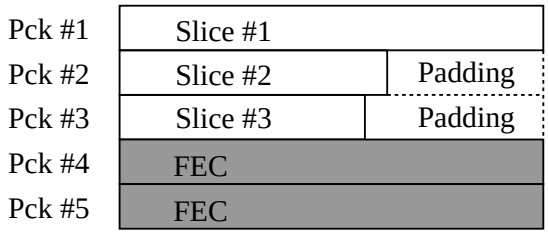

Figure 2: Cross-packet redundancy (padding scheme).

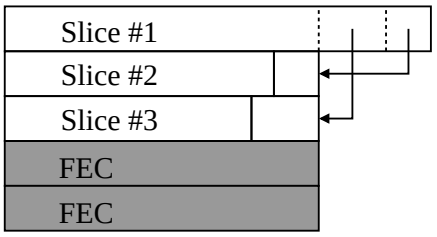

Figure 3: Packet size equalization scheme with slice splitting.

first the macroblock coding parameters and then the protection level of each slice. Note that the optimal choice of parameters for each slice is completely independent of other slices. Similarly, if the protection level is chosen at the frame level, the second step optimization is performed on the whole frame.

3 Packetization Algorithm

The slices produced by the coding algorithm need to be placed in packets prior to transmission. We will assume here that the Real-Time Protocol (RTP) [7] is used. In the case of a DiffServ network, each slice is put into a single packet and assigned to the class corresponding to the chosen protection level. In the case of ordinary packet networks, the common approach is to apply FEC across packets [8] as in Figure 2. Some padding is inserted to equalize the size of the data packets before computing the redundancy packets. We assume that each slice corresponds to one packet in this case as well.

The above approach, to which we refer as *padding scheme*, suffers from several shortcomings. First, the lack of uniformity in data packet size results in redundancy packets that are larger than what is assumed by the optimization algorithm, thus compromising the optimality of the solution. Moreover, the ratio of the redundancy to actual source bits is worse than that what is intended by the nominal code parameters, and implies waste. Secondly, it is extremely difficult to integrate the exact computation of the final redundancy amount in the optimization algorithm, because it depends on mode decisions for all the macroblocks in the frame. Finally, the algorithm produces a number of slices to protect with a certain FEC code that usually differs from the k value of the code. In this case, a code with the required k value and a similar protection level is used, but this also reduces the optimality of the solution.

To circumvent the above shortcomings, we propose to use the following *packet size equalization* scheme. The slices are split in order to produce packets of equal size, so that no padding is required, as in Figure 3.

For each FEC code (n,k) used in the frame, a group of packets is generated as follows:

1. Let S be the set of the slices to be protected by that code, and let b be their total size.

2. Let $b_p = b/k$ be the packet size where k packets are used to packetize S.

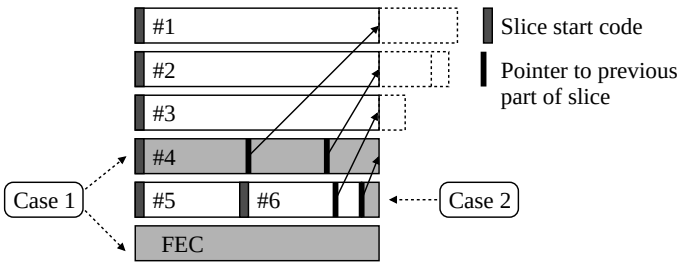

Figure 4: Example of packet decoding. The shaded parts cannot be decoded because either they are not received (case 1) or they depend on previous unavailable parts (case 2).

3. While tracking the available space in each packet, assign each slice in S (consecutively) to a packet, and reduce its available packet space accordingly. If the slice is larger that the largest available space, split it and mark the remaining part unassigned.

4. For each unassigned part of slice, assign it to the largest available space. Precede it with a pointer to the packet containing the previous slice part. Split it again if needed, and repeat until the slice has been completely packetized.

5. For each packet, store the length in macroblocks of the last slice that begins in the packet, so that its end can be easily detected during the decoding process.

6. Compute the $n - k$ redundancy packets.

To reconstruct the slices, the following procedure is used. The redundancy packets are exploited as much as possible to recover the losses. Then, for each available packet belonging to a certain group:

1. Decode the first part if it starts with a slice start code, until reaching the end of the slice or packet.

2. If a new start code is detected at the end of the slice, repeat the preceding item.

Then, for each incompletely decoded packet, and each partial slice in it:

3. Read the pointer to the previous packet, and continue the decoding of the uncompleted slice. If the referred packet is missing, proceed with the next incompletely decoded packet. The length in macroblocks of the slice determines its end.

4. Repeat the preceding item until possible.

To apply the previous procedure, the receiver must identify the packets belonging to the same group. Therefore, a consecutive RTP sequence number is assigned to the packets of each group. To distinguish the various groups, the least significant bits of the RTP sequence number of the first packet in the group is stored in every packet. Part of the timestamp field provided by the RTP could be used for this purpose.

The initial part of each slice is easily decoded if the corresponding packet has been received. Then, if all the packets in the group are present, the receiver can decode all slices.

The end of a slice is determined either by detecting a following slice start code or by means of its length in macroblocks, inserted in the packet by the encoder with a method analogous to the group information. If several packets are missing, some slices may be incompletely decoded either because a portion is missing (e.g., when the pointer indicates a missing packet) or because that portion is located after an undecodable fragment of another slice.

Figure 4 shows an example of the decoding procedure. The fourth and last packets are not received. The FEC code is not sufficient to recover the loss of the data packet. The slices affected by the losses are Slice 4 and part of Slice 1 and 2. The last piece of Slice 2 is not decodable because it depends on a previous part included in a lost packet (case 2.)

The proposed packetization method introduces some interdependencies among parts of certain slices, thus it modifies the actual effective loss probability of some macroblocks. This compromises the optimality of the solution, but causes significantly less degradation than the commonly used padding scheme, as is confirmed by the simulation results in Section 4.

Finally, it is worth noting that with both packetization schemes, the correct macroblock loss probability that takes into account the dependencies can be computed after the packetization. That probability is effectively used by ROPE when updating the first and second moments of each decoder reconstructed pixel.

4 Results

The proposed optimization and packetization algorithms have been implemented on the basis of the UBC H.263+ codec [9], modified to support Annex K (Slice Structured Mode). The results presented are on the standard QCIF sequence *foreman*, encoded at 30 fps and 300 kbit/s, unless otherwise noted. Analogous results, not shown here, were obtained for other standard sequences such as *carphone* and *news*. The rate control algorithm is the one used in [2].

4.1 Trellis-Based Slicing and Coding Optimization

The first set of experiments tests the performances of the proposed adaptive slicing and coding optimization algorithm, which jointly optimizes the coding mode of each macroblock, its protection level and the subdivision of the frame into slices.

The *foreman* video sequence is transmitted over a DiffServ network implementing the QoS classes shown in Table 1. Each slice is put into one packet, marked and transmitted according to the QoS class assigned to the slice by the algorithm.

Table 1: Characteristics of the used DiffServ classes.

Class number	Residual loss probability p	Cost (μcent/bit)
1	0.10	1.00
2	0.05	2.00
3	0.025	4.00

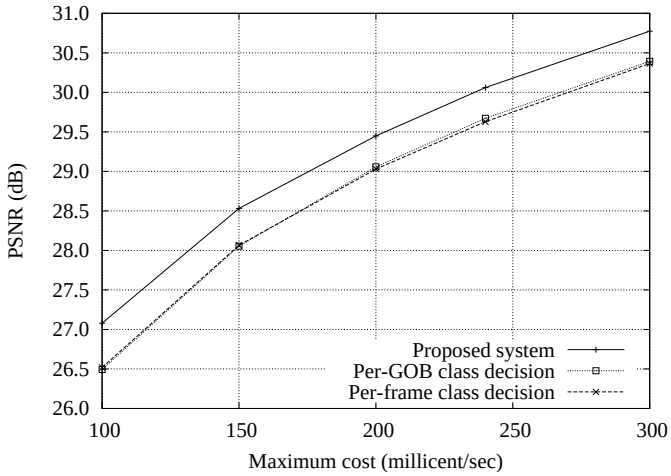

Figure 5: PSNR versus the maximum cost constraint, for the proposed optimization algorithm, in comparison with per-frame and per-GOB class decision schemes.

Figure 5 shows the performance of the proposed trellis-based algorithm versus the maximum allowed cost, in comparison with two other schemes that select the QoS class for the whole frame or for each single slice, respectively. In both cases, each slice corresponds to one GOB of the H.263 standard (fixed slicing). These methods can be viewed as restricted special cases of the proposed scheme, as pointed out at the end of Section 2. The gain due to joint optimization of the coding and protection parameters at the macroblock level is clear. Of the two fixed slicing methods, the choice of one protection level for the whole frame delivers the worst performances. But selecting the protection level for each fixed slice only provides a very modest performance improvement.

The advantage of the proposed joint optimization at the macroblock level resides in its ability to effectively exploit all the classes while automatically considering the slicing overhead in the decision process. Consequently, it uses the more expensive classes only where they are truly beneficial in the rate-distortion sense, hence the gain relative to less sophisticated methods.

4.2 Packet Size Equalization

This second set of experiments tests the performance of the proposed packetization scheme. The following MDS codes have been used in all the experiments: (7,6) (6,5) (5,4) (4,3) (3,2) (5,3) (6,4) (7,5). Sending packets without protection is another option.

Figure 6 shows the performance of the proposed slice-splitting scheme and the commonly used padding scheme, versus the packet loss rate. The upper curve represents the ideal upper bound, where no real packetization is performed, and each slice is directly subjected to the effective loss probability that the FEC code is designed to provide. The middle curve represents the performance of the proposed packetization scheme. We note the significant performance loss of the padding scheme, especially for low and medium packet loss probabilities. We attribute this effect to the employment of longer redundancy packets, which causes a bitrate waste that significantly compromises performances, especially when

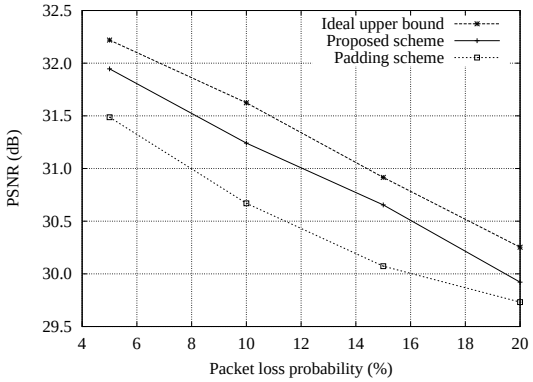

Figure 6: PSNR versus packet loss probability for various packetization methods.

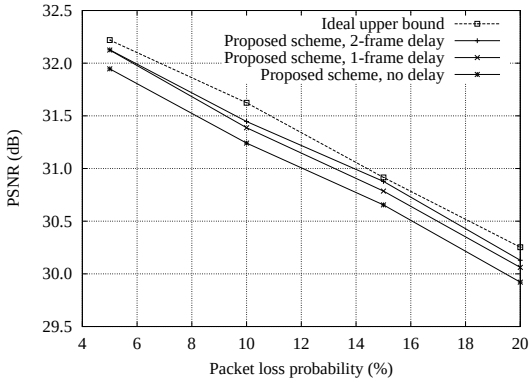

Figure 7: PSNR versus packet loss probability for various packetization delays.

the channel is quite good. The proposed packetization scheme has superior performances because it completely avoids this pitfall.

In addition, we consider a scenario where some delay is allowed at the transmitter. In this case, the packetization is performed after encoding a few frames. Thus, when performing the packetization, the number of slices to be placed into packets is higher and provides more flexibility to find a better allocation in which the amount of dependency due to the slice splitting is reduced. Figure 7 shows the performance increase allowing respectively a zero, one or two-frame delay before placing the slices into packets. The performance of the system using a two-frame delay is close to the ideal upper bound.

Finally, Figure 8 shows the relative benefits offered by each aspect of the proposed technique, and in particular the crucial role played by packet equalization in the context of FEC protection (and in contradistinction with the case of DiffServ transmission). Four cases resulting from the various combination of slicing and packetization techniques are shown. The inefficiency of padding for packetization causes less damage in the case of fixed slicing which involves a high number of slices with smaller variation in length in bits. However it severely compromises the performance of adaptive slicing. When inefficient padding is replaced with the proposed packetization scheme, adaptive slicing optimization achieves additional performance gains over fixed slicing, as expected.

5 Conclusions

We presented a rate-distortion optimized algorithm to find the best encoding parameters and protection level for each macroblock in order to minimize the expected end-to-end distortion. The proposed algorithm is able to optimally create a set of slices for each frame, while accounting for the overhead that slice creation incurs. Furthermore, for improved protection by FEC codes, a packet size equalization scheme is presented to efficiently arrange the slices into packets. Simulation results show that the performance gains with respect to the currently used techniques are consistent over various network conditions. Finally, a higher delay variant of the packetization algorithm is shown to further improve the overall performance, and to approach the ideal upper bound.

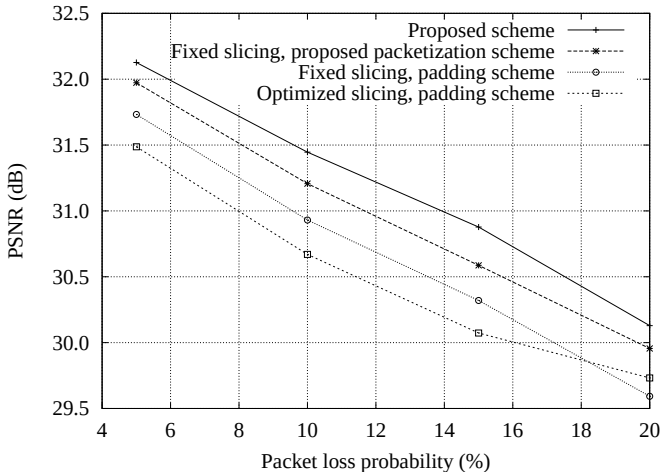

Figure 8: PSNR versus packet loss probability for various transmission schemes.

References

[1] D. Wu, Y.T. Hou, and Y.-Q. Zhang, "Transporting real-time video over the internet: challenges and approaches," *Proceedings of the IEEE*, vol. 88, no. 12, pp. 1855–1875, December 2000.

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

[3] E. Masala, D. Quaglia, and J.C. De Martin, "Adaptive picture slicing for distortion-based classification of video packets," in *Proc. IEEE Workshop on Multimedia Signal Processing*, Cannes, France, October 2001, pp. 111–116.

[4] C.E. Luna, Y. Eisenberg, R. Berry, T.N. Pappas, and A.K. Katsaggelos, "Joint source coding and packet marking for video trasmission over DiffServ networks," in *Tyrrhenian International Workshop on Digital Communications (IWDC)*, Capri, Italy, September 2002.

[5] F. Zhai, R. Berry, T.N. Pappas, and A.K. Katsaggelos, "A rate-distortion optimized error control scheme for scalable video streaming over the internet," in *Proc. IEEE Int. Conf. on Multimedia & Expo*, Baltimore, MD, July 2003, pp. 125–128.

[6] F. Zhai, Y. Eisenberg, C.E. Luna, T.N. Pappas, R. Berry, and A.K. Katsaggelos, "Packetization schemes for forward error correction in internet video streaming," in *Proc. 41st Allerton Conf. on Communication, Control and Computing*, October 2003.

[7] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A transport protocol for real-time applications," *RFC 1889*, January 1996.

[8] M. Gallant and F. Kossentini, "Rate-distortion optimized layered coding with unequal error protection for robust internet video," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, no. 3, pp. 357–372, March 2001.

[9] H.263+ codec, "http://spmg.ece.ubc.ca," .