Title

Multiplexing video streams using dual-frame video coding

Authors

Tiwari, M
Groves, T
Cosman, P C

Publication Date

2008-03-01

Peer reviewed

MULTIPLEXING VIDEO STREAMS USING DUAL-FRAME VIDEO CODING

Mayank Tiwari[1], *Theodore Groves*[2] *and Pamela C. Cosman*[1]

[1]Department of Electrical and Computer Engineering, [2]Department of Economics
University of California, San Diego, La Jolla, CA 92093-0407

ABSTRACT

We consider the transmission of multiple video source streams over a shared channel from a server. Using the rate-distortion curves and dual-frame video coding with high quality long-term reference (LTR) frames, we propose a method to reduce the sum of mean squared error for all the video streams. A simple motion activity detection algorithm was used to determine the amount of high quality given to the LTR frames. Using H.264/AVC, the results show considerable improvement over a baseline scheme where each video stream is provided with equal bitrate.

Index Terms— Dual-frame buffer, H.264/AVC, high quality updating, video compression, long-term reference frame.

1. INTRODUCTION

Transmission of multiple video streams from a central server (or from multiple separate servers but with centralized rate allocation) to multiple destinations over a path with a shared link is a common scenario. Some applications are transmission of digital video over wireless broadband, video-on-demand services, video surveillance, and transmission in a cognitive radio situation where multiple users share the same bandwidth. The total bitrate of multiple video streams is limited by the bandwidth of the central server. Therefore, it is important to efficiently allocate the overall bitrate among the video streams to enhance the overall quality.

Joint bitrate allocation has been widely studied [1–5] to improve the overall quality for multiple video streams. A multi-camera surveillance system was considered in [1] where peak signal-to-noise ratio (PSNR) improvement was shown for transmitting video content only when there is any activity captured by a camera. However, it did not consider the case in which all cameras were capturing an activity simultaneously. A distributed approach with high convergence time for transmitting multiple video streams was considered in [2] where the bitrate allocation was made by link price updated using the subgradient method. A parallel encoder system with large delay and memory requirements was adopted in [3] where multiple streams are encoded with several bitrates, and a combination of bitrates for multiple streams was selected to maximize the PSNR. A superframe concept was used in [4] where one frame each from multiple video streams are combined into a superframe and a quantization parameter (QP) is found to maximize the overall PSNR. In [5], a resource allocation algorithm was proposed to reduce PSNR fluctuation while maintaining high PSNR using fine granularity scalability (FGS). It reduces PSNR fluctuations but also reduces overall PSNR.

In [6], three optimization objectives for transmitting multiple video streams over a shared channel were studied: maximizing overall PSNR, minimizing overall mean squared error (MSE), and minimizing the maximum MSE. Using subjective tests, it was shown that minimizing the overall MSE corresponds best to subjective preferences. Using this result, we chose the performance criterion of minimizing the overall MSE. The results here should not be compared directly with a method that assumes any other performance criterion.

This work was supported by the National Science Foundation, the Center for Wireless Communications at UCSD, and the UC Discovery Grant program of the State of California.

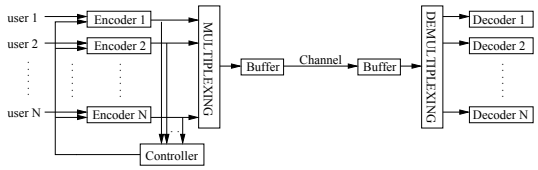


Fig. 1. Bitrate allocation for multiple video streams

In this paper, we compare seven methods of transmitting multiple video streams under the constraint of a fixed total bitrate. Each video stream is given an equal bitrate ("fair" allocation), or alternatively, we allocate bitrate based on instantaneous frame complexity using rate-distortion (R-D) curves. We also combine dual-frame video coding with these two methods. The quality of the long-term reference frame is determined by a simple motion activity detection algorithm. We assume there is a controller that performs bitrate allocation based on the information provided to the controller by individual users as shown in Figure 1. We also compare rate allocation using H.264 JM rate control and the superframe methods described in [4, 7].

The paper is organized as follows: Section 2 discusses dual-frame video coding and the motion detection algorithm. R-D curve fitting and various bitrate allocation methods are discussed in Section 3. Results and conclusions are given in Section 4.

2. DUAL-FRAME CODING AND MOTION DETECTION

In multiple frame prediction, more than one past frame is used in the search for the best match block. At the cost of extra memory storage and extra complexity for searching, multiple frame prediction has been shown to provide a clear advantage in compression performance [8,9].

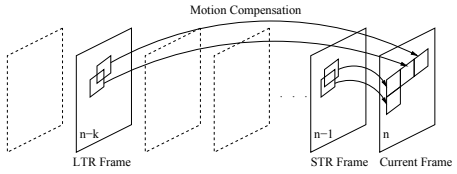


Fig. 2. Dual-frame video coding

In dual-frame video coding [10–13], two frames are used for inter prediction, a short-term reference (STR) and a long-term reference (LTR), as shown in Figure 2. Both encoder and decoder store LTR and STR frames. For encoding frame n, the STR is frame $n-1$ and the LTR is frame $n-k$, for some $k > 1$. The LTR frame can be chosen by jump updating, in which the LTR frame remains the same for encoding N frames, then jumps forward by N frames and again remains the same for encoding the next N frames. In continuous updating, every frame has a turn serving as an STR and as an LTR. In jump updating, every frame serves as an STR, but only every N^{th} frame serves as an

LTR; this allows the use of high quality LTRs, where the LTR frames are allocated more bits than regular frames. This has been shown to enhance the quality of the entire stream [12, 13]. In [12], the assumption was that regular frames could be starved of bits so as to generate high quality LTR frames at even intervals, provided that a long-term average bit rate constraint was met. In this paper, we use the term "regular dual-frame" to mean one in which LTR frames are not assigned a quality higher than that of non-LTR frames. For the video streams and bitrates examined, high quality LTR frames improved the average video quality by 0.6 dB over a regular dual-frame encoder. In [13], dual-frame video coding was considered in a cognitive radio scenario consisting of a low bandwidth channel that is periodically supplemented by the rental of a substantially larger bandwidth for a short interval of time. Here, a high quality frame was formed from the extra bandwidth and used as an LTR frame for some time. In this paper, we will use dual-frame video coding for enhancing the quality of multiple video streams simultaneously.

In dual-frame video coding, a key issue is to allocate an appropriate number of bits to ensure a high quality LTR frame. For a low motion video stream, we should allocate many bits to the LTR frame since the quality of subsequent frames will also be high because they are similar to the LTR frame. For high motion parts of a video stream, it is not desirable to spend many bits on an LTR frame because its higher quality will soon be useless as the subsequent frames rapidly become different from the LTR.

We detect the motion of a video stream by comparing the current and previous frames. We considered a large set of QCIF videos to determine the threshold and amount of extra quality given to an LTR frame. We divide a frame into macroblocks (MBs) of size 16×16 pixels and calculate the sum of absolute differences (SAD) between each MB and the co-located MB in the previous frame. The MB is considered active if the SAD is above a predetermined threshold, otherwise inactive. Examining a wide range of thresholds, we chose a threshold of 500. Figure 3 shows the result of activity detection for the Foreman and Mother Daughter videos. On the x-axis is the frame number and on the y-axis is the percentage of MBs above the threshold. It shows that Foreman is of higher motion than Mother-Daughter.

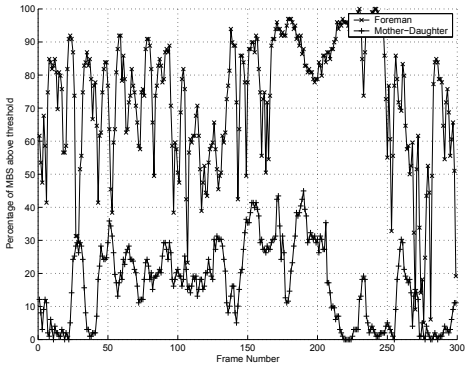


Fig. 3. Percentage of MBs above the motion threshold of 500 for the Foreman and Mother-Daughter video streams.

Let m be the average fraction of active MBs in the 10 frames prior to an LTR frame. The bit allocation for the LTR frame (LTR_bits) is given by

$$LTR\_bits = \begin{cases} 2 \times reg\_bits, & \text{if } m > 0.5 \\ 10 \times reg\_bits, & \text{if } m < 0.1 \\ (12 - 20 \times m) \times reg\_bits, & otherwise, \end{cases} \quad (1)$$

where reg_bits is the average number of bits assigned to a regular frame. These allocations and threshold values were determined experimentally but not carefully optimized for these videos. The results were found to hold for other video streams. Motion activity detection is done in real-time and we do not need to store the video in advance. A similar method in [1] considered binary classification of video activity.

3. METHODS FOR MULTIPLEXING VIDEO STREAMS

In this section, we describe the allocation of bits to multiple video streams simultaneously. We have N video streams and B kbps of total bitrate available to transmit these video streams. The simplest bitrate allocation is to divide bits equally among the video streams and among the frames. If a video stream is f frames per second, then each frame gets $\frac{B}{N \times f}$ bits. We call this method **EqualBits** and it could be called a "fair" allocation. Given the number of bits to encode a frame of a video stream, the reference software model of H.264 [14] will search and choose the best prediction mode to reduce the MSE.

In this paper, the curve-fitting model for the R-D curve of a frame in video stream n is given by

$$D_n(R_n) = a_n + \frac{b_n}{R_n} \quad (2)$$

where R_n is the number of bits and D_n is the distortion for a frame i in video stream n. a_n, b_n are the coefficients for generating this curve-fitting model. Other curve-fitting models are available in the literature [15]. We use the least squares approach to find a_n and b_n. We take 14 R-D measurement points using different QPs. The complexity of generating R-D curves can further be reduced by using the method described in [16] and is not included in this paper.

Given the R-D curve-fit for a frame in each video stream, the sum of MSE for all the video streams can be minimized using standard optimization techniques like the Lagrangian multiplier. Consider the i^{th} frame of each video stream. The optimization problem can be formulated as

$$\min_{R_n} \sum_{n=1}^{N} D_n(R_n) \qquad s.t. \sum_{n=1}^{N} R_n \leq \frac{B}{f} \quad (3)$$

Using Lagrange multipliers, the bit allocation for video j is

$$R_j = \frac{\sqrt{b_j}}{\sum_{n=1}^{N} \sqrt{b_n}} \times \frac{B}{f}, \qquad \forall j \in 1, 2, ..., N \quad (4)$$

The bit allocation achieved by Eq. 4 essentially finds a point in each R-D curve where the slope is the same for all the curves. We denote this method of bitrate allocation as **EqualSlope** [6]. This minimizes, on a per frame basis, the sum of MSEs for all the video streams. This method gives extra bits to a video stream that is experiencing high motion by taking some bits from the low motion video streams. Both EqualBits and EqualSlope allocation require neither any input buffer to store raw frames to be encoded nor any output buffer to store the encoded bitstream that needs to be transmitted (because the multiplexed bitstreams achieve the constant target output rate for each frame).

Both the methods above use only STR frames for motion compensation. By using dual-frame video coding, we can further reduce the sum of MSE by exploiting the LTR frame. Let L_n be the number of bits assigned to an LTR frame for the n^{th} video stream and let the LTR update interval be K. Note that L_n varies with the motion activity of a video stream, as described in the previous section. Using EqualBits, each video stream should get $\frac{K \times B}{N \times f}$ bits for K frames. In the LTR extension to EqualBits, out of this pool of bits, L_n bits are assigned to an LTR frame. We define

$$r_n = \frac{\frac{K \times B}{N \times f} - L_n}{K - 1} \quad (5)$$

We allocate r_n bits to each of the remaining $K-1$ frames in that group of K frames in the stream. The LTR frames are assigned to each video stream in a round-robin fashion where any particular video stream is assigned an LTR frame at a regular interval and between two LTR frames for any given video stream, there will be one LTR frame for each of the other video streams. This may also be deemed a "fair" allocation. Although the number of bits L_n given to the LTR for stream n may be higher than that for some other stream, the number of bits r_n given to the other $K-1$ frames in that group of K frames for stream n will be correspondingly lower, so each video stream is allocated an equal number of bits over K frames. Some of the bits are taken from the regular frames in order to raise the quality of the LTR frame. This method is called **LTR_EqualBits**.

We can also incorporate dual-frame video coding in the EqualSlope method. Again, the bit allocation for LTR frames is as described in LTR_EqualBits. Using Eq. 4, for any frame where no stream has an LTR, the bits allocated to video j are

$$R_j = \frac{\sqrt{b_j}}{\sum_{n=1}^{N}\sqrt{b_n}} \times \sum_{n=1}^{N} r_n, \quad \forall j \in 1,2,...,N \tag{6}$$

where r_n is defined by Eq. 5. If, at any time instant, a video stream k has an LTR frame, then that video stream receives L_k bits and the remaining video streams will receive EqualSlope allocation, i.e.

$$R_j = \frac{\sqrt{b_j}}{\sum_{n=1,n\neq k}^{N}\sqrt{b_n}} \times \sum_{n=1,n\neq k}^{N} r_n, \quad \forall j \in 1,2,...,N, j \neq k \tag{7}$$

This method uses dual-frame video coding with high quality LTR frames and EqualSlope allocation for regular frames and is denoted by **LTR_EqualSlope**. Note that both LTR_EqualBits and LTR_EqualSlope do not require any input buffer to store raw frames but they require a fixed size output buffer to store the encoded bitstream for transmission if the transmission channel has constant bitrate.

We expect EqualSlope will perform better than EqualBits because, in EqualSlope, bits are allocated to different video streams based on their complexity. We expect that LTR_EqualBits will perform better than EqualBits and LTR_EqualSlope will perform better than EqualSlope because of the advantages of dual-frame video coding.

For comparison, we included the results for H.264 JM reference software rate control [14] for individual video streams and implemented the superframe methods described in [4, 7]. With JM rate control (**H.264_RC**), a target bitrate was assigned and the encoder was allowed to use the R-D optimization method to efficiently encode each video stream separately. The superframe method was implemented in two different ways: **SF_EqualBits** and **SF_RC**. In SF_EqualBits, the bits used for each superframe are constant and the encoder finds a QP that meets the constraint. In SF_RC, the target bitrate for the whole superframe stream was assigned and then H.264 encodes superframes with rate control (bit constraint for individual superframes is waived).

4. RESULTS

The simulation was performed using the baseline profile of H.264/AVC [17] reference software JM 9.6 [14], modified for our work. All the video streams used for the simulations are QCIF (176×144 pixels) at 30 frames per second and of length 300 frames. The first frame is an I-frame and the remaining frames are P-frames. We considered a lossless channel of $N \times 15$ kbps, for N video streams. Note that it does not include the first frame which is separately INTRA-coded for each video stream. For LTR_EqualBits and LTR_EqualSlope, the distance between two LTR frames in a video stream is set at 25 frames and there are a total of 12 LTR frames.

Table 1 shows the results of multiplexing video streams using all seven methods described in section 3. The table shows the average MSE and its corresponding PSNR for the video streams. For any video stream in Table 1, the PSNR is calculated from the MSE, where the MSE is the average MSE of the whole video stream. The average PSNR is calculated from the MSE averaged within and across all the video streams. We can see that, for a high motion video such as Foreman, EqualSlope reduces its MSE by assigning more bits, and the low motion video such as Mother-Daughter receives fewer bits and thus there is an increase in its MSE. The MSE reduction for the high motion video is much larger than the MSE increase for the low motion video when compared to the EqualBits case. When high quality LTR frames are used, we see MSE reduction in all the videos compared to EqualBits. When the EqualSlope technique is applied along with LTR frames, we see that high motion videos further reduce their MSE compared to LTR_EqualBits while there is some increase in MSE for low motion videos. But when we compared the overall MSE, we found that LTR_EqualSlope performs the best. In this case, LTR_EqualBits performs marginally better than EqualSlope. Figure 4 shows the MSE versus frame number for two of these four multiplexed video streams. The seven curves in each figure represent different methods of multiplexing video streams. As can be seen in figure, LTR_EqualSlope performs the best overall, providing the best MSE for low motion video streams such as container, while being nearly the best for high motion video streams such as Carphone or Foreman. The quality pulsing caused by the LTR methods is apparent in the MSE plot of Figure 4(b) but is not perceptually noticeable when viewing the video.

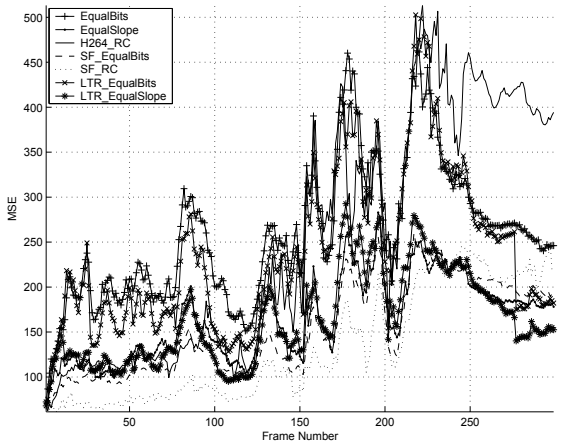
Table 1. MSE and PSNR for multiplexed video streams

MSE	Carphone	Foreman	Mother-D.	Container	Avg MSE
EqualBits	167.49	259.74	41.43	98.94	141.90
EqualSlope	144.86	164.55	63.97	113.08	121.62
H.264_RC	154.16	251.39	34.30	47.04	121.72
SF_EqualBits	137.79	153.03	79.62	127.75	124.55
SF_RC	126.60	132.78	69.07	117.06	111.38
LTR_EqualBits	149.39	238.54	33.25	40.36	115.38
LTR_EqualSlope	133.94	164.27	45.98	47.98	98.04

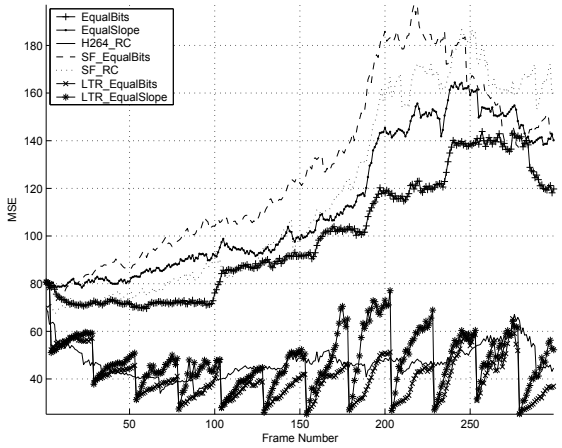
PSNR	Carphone	Foreman	Mother-D.	Container	PSNR(Avg MSE)
EqualBits	25.89	23.99	31.96	28.18	26.61
EqualSlope	26.52	25.97	30.07	27.60	27.28
H.264_RC	26.25	24.13	32.78	31.41	27.28
SF_EqualBits	26.74	26.28	29.12	27.07	27.18
SF_RC	27.11	26.90	29.74	27.45	27.66
LTR_EqualBits	26.39	24.36	32.91	32.07	27.51
LTR_EqualSlope	26.86	25.98	31.50	31.32	28.22

As expected, H.264_RC performs better than EqualBits because, with an output buffer, there is more freedom in assigning the bits to various frames according to their relative complexity in H.264_RC. But its performance is quite similar to EqualSlope where the complexity of video streams is exploited simultaneously at the frame level. SF_EqualBits performs much better than EqualBits because it also allocates bits at the frame level based on complexity. Although both EqualSlope and SF_EqualBits assign bits to each video stream based on complexity, the performance of EqualSlope is still better than that of SF_EqualBits due to the fact that EqualSlope gives the optimal bit allocation based on the R-D curve of each video stream (Eq. 3 and Eq. 4) at the frame level which may not result in the same QP as SF_EqualBits. Similarly, SF_RC performs better than H.264_RC because it not only uses the output buffer for variable bitrate but also distributes the bits within a superframe based on complexity. With the use of LTR frames and EqualSlope, LTR_EqualSlope outperforms both H.264_RC and SF_RC by 19% and 12% respectively. In terms of PSNR, LTR_EqualSlope outperforms SF_RC by 0.56 dB and H.264_RC by 0.94 dB. When comparing individual video streams we find that, for high motion video streams, SF_RC reduces the MSE by a huge margin, but at the expense of a large increase in the MSE for low motion video streams. On the other hand, LTR_EqualSlope reduces the MSE

of high motion videos by a huge margin but it marginally increases the MSE of low motion videos.



(a) Foreman video stream



(b) Container video stream

Fig. 4. MSE variation with frame number for multiplexed video streams

During simulation trials it was observed that if we allocate many bits to LTR frames, then the performance of LTR_EqualBits tends to be very close to LTR_EqualSlope. When more bits are given to LTR frames, then fewer bits are left for other frames. Therefore, when we equalize the slope for these other frames, there are not many bits to adjust and we get a very small advantage from doing EqualSlope. If we give fewer bits to LTR frames, then the effect of LTR frames is small. In that case, the performance of LTR_EqualBits is close to that of EqualBits. Therefore, it is necessary to moderate the amount of extra quality given to LTR frames to achieve better performance.

In conclusion, we considered seven different ways to allocate bitrate for multiple video streams, two of which are new. We considered the R-D curve properties for performing bitrate allocation. We found that MSE reduction can be achieved using dual-frame video coding with high quality LTR frames. One novel idea in this paper was the use of motion activity detection to determine the amount of high quality for LTR frames. Multiplexing video streams using high quality LTR frames with a fixed amount of extra bits going to the LTRs regardless of motion activity was found to give poor performance. The second novel idea for multiplexing video streams was to combine the high quality LTR frames in dual-frame video coding with EqualSlope. This new method was shown to outperform existing methods for multiplexing video streams.

5. REFERENCES

[1] X. Zhu, E. Setton, and B. Girod, "Rate allocation for multi-camera surveillance over an ad hoc wireless network," in *Proc. Picture Coding Symposium (PCS)*, Dec. 2004.

[2] X. Zhu and B. Girod, "Distributed rate allocation for multi-stream video transmission over ad-hoc networks," in *Proc. IEEE Int. Conf. on Image Processing*, vol. 2, Sept. 2005, pp. 157–160.

[3] J. Kammin and M. Sakurai, "Video multiplexing for the MPEG-2 VBR encoder using a deterministic method," in *Proc. IEEE International Conf. on Automated Production of Cross Media Content for Multi-Channel Distribution*, 2006, pp. 221–228.

[4] L. Wang and A. Vincent, "Bit allocation and constraints for joint coding of multiple video programs," *IEEE Trans. on Circuits and Systems for Video Tech.*, vol. 9, no. 6, pp. 949–959, Sept. 1999.

[5] G. Su and M. Wu, "Efficient bandwidth resource allocation for low-delay multiuser video streaming," *IEEE Trans. on Cir. and Sys. for Video Tech.*, vol. 15, no. 9, pp. 1124–1137, Sept. 2005.

[6] M. Kalman and B. Girod, "Optimal channel-time allocation for the transmission of multiple video streams over a shared channel," in *Proc. IEEE Workshop on Multimedia Signal Processing*, Oct. 2005, pp. 1–4.

[7] J. Yang, X. Fang, and H. Xiong, "A joint rate control scheme for H.264 encoding of multiple video sequences," *IEEE Trans. on Cons. Electronics*, vol. 51, no. 2, pp. 617– 623, May 2005.

[8] M. Gothe and J. Vaisey, "Improving motion compensation using multiple temporal frames," *Proc. IEEE Pacific Rim Conference on Communications, Computers and Signal Processing*, vol. 1, pp. 157–160, May 1993.

[9] M. Budagavi and J. D. Gibson, "Multiframe video coding for improved performance over wireless channels," *IEEE Trans. on Image Processing*, vol. 10, no. 2, pp. 252–265, Feb. 2001.

[10] T. Fukuhara, K. Asai, and T. Murakami, "Very low bit-rate video coding with block partitioning and adaptive selection of two time-differential frame memories," *IEEE Trans. on Circuits and Systems for Video Tech.*, vol. 7, no. 1, pp. 212–220, Feb. 1997.

[11] T. Wiegand, X. Zhang, and B. Girod, "Long-term memory motion-compensated prediction," *IEEE Transactions on Circuits and Systems for Video Tech.*, vol. 9, no. 1, pp. 70–84, Feb. 1999.

[12] V. Chellappa, P. Cosman, and G. Voelker, "Dual-frame motion compensation with uneven quality assignment," *Proc. IEEE Data Compression Conference*, pp. 262–271, Mar. 2004.

[13] M. Tiwari and P. Cosman, "Dual-frame video coding with pulsed quality and a lookahead buffer," *Proc. IEEE Data Compression Conference*, pp. 372–381, Mar. 2006.

[14] "H.264/AVC ref. software," http://iphome.hhi.de/suehring/tml.

[15] K. Stuhlmüller, N. Färber, M. Link, and B. Girod, "Analysis of video transmission over lossy channels," *IEEE Journal on Selected Areas in Comm.*, vol. 18, no. 6, pp. 1012–1032, June 2000.

[16] L. Lin and A. Ortega, "Bit-rare control using piecewise approximated rate-distortion characteristics," *IEEE Trans. on Cir. and Sys. for Video Tech.*, vol. 8, no. 4, pp. 446–459, Aug. 1998.

[17] T. Wiegand, G. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Transactions on Circuits and Systems for Video Tech.*, vol. 13, no. 7, pp. 560–576, July 2003.