Flexible Video Decoding: A Distributed Source Coding Approach

Ngai-Man Cheung and Antonio Ortega
Signal and Image Processing Institute, Univ. of Southern California, Los Angeles, CA

***Abstract*— We investigate video compression techniques to address problems that require *flexible video decoding*. In these, the encoder has access to a number of candidate predictors that allow it to exploit source signal correlation, but only a subset of these predictors will be available at the decoder. Crucially, the encoder *does not* know which predictors will be available. Flexible decoding is important in a number of applications including frame-by-frame forward and backward video playback, multiview video, bitstreams switching, robust video transmission, etc. The main challenge to support flexible decoding is that the encoder needs to compress a current frame under the uncertainty on the predictor at decoder. An approach based on conventional "closed loop" prediction, e.g., motion-compensated predictive (MCP) coding in the case of video, could be developed by including multiple possible prediction residues in the bitstream, but this would lead to a considerable coding performance penalty, if all possible predictor combinations are supported, or to drifting, if only some combinations are. Moreover, it is not possible in general to guarantee that decoded versions under different prediction scenarios will be identical. In this paper, we propose a distributed source coding (DSC) based algorithm to tackle the problem. The main novelties of the proposed algorithm are that it incorporates different macroblock modes and significance coding within the DSC framework. This, combined with a judicious exploitation of correlation statistics, allows us to achieve competitive coding performance. Using forward/backward video playback as an example, we demonstrate the proposed algorithm can outperform a solution based on MCP coding.**

I. Introduction

In this paper we investigate video compression algorithms to support *flexible decoding* for a number of emerging applications. In this problem (see Figure 1) the input source X (a video frame) is to be communicated to the decoder and a number of correlated sources $Y_0, Y_1, ..., Y_{N-1}$ (previously decoded video frames) are available at the encoder to serve as candidate predictors for compressing X. But, of these predictors, only *one* will be available at the decoder. Crucially, encoder *does not* know which Y_k will be used at the decoder. Our goal is to develop coding algorithms such that encoder can operate under this kind of uncertainty about decoder operation.

This flexible decoding problem can arise in a number of applications. Consider first a video application where both *forward and backward frame-by-frame playback* are to be supported, which we first investigated in [1]. In this application, the user can choose to play back in either direction, and therefore, when decoding a current frame, either the "past" *or* the "future" reconstructed frame will be available at the decoder to serve as the predictor. However, encoder does not know which one will be present at decoder.

Flexible decoding may also be useful in *multiview video coding*. In this application, the user may navigate between different views during video playback. Therefore, depending on whether the user is staying in the same view (Figure 2.a) or switching views (Figure 2.b), either the previous reconstructed frame of the same view *or* that of another view may be available as predictor for decoding the current frame, respectively. Flexible decoding may also be a useful tool to support *bitstream switching*. In this case, user may choose to switch between different playback qualities, or between different picture sizes. It may also be useful to support flexible decoding to enhance *robustness in transmission*. Here some of the candidate reference frames may arrive at decoder without error and can be used to decode the current frame. However, encoder does not know which ones are error-free.

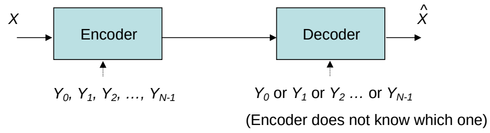

Fig. 1. Problem formulation for flexible decoding. Either one of the candidate predictors $Y_0, Y_1, ..., Y_{N-1}$ will be present at the decoder, but encoder does not know which one.

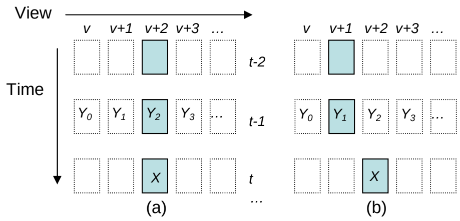

Fig. 2. Flexible decoding in multiview video: (a) User plays back the $(v+2)$-th view. Y_2 is available as predictor for decoding X. (b) User plays back $(v+1)$-th view and switches to the $(v+2)$-th view at time t. In this case, Y_1 is available for decoding X.

The challenge is for the encoder to compress the current frame such that it can be recovered using any of the error-free references.

In order to support flexible decoding within conventional motion-compensated predictive (MCP) video coding systems (e.g., MPEG, H.26X), the encoder may send all the possible prediction residues $\{Z_i; i = 0 \text{ to } N-1\}$ to the decoder, where $Z_i = X - Y_i$ (following the notations in Figure 1), so that X can be recovered no matter which Y_i is available at the decoder. Each Z_i would correspond to a P-frame in these video coding standards. There are two main problems with such an approach. First, coding performance is degraded because multiple prediction residues are included in the bitstream. Specifically, the overhead to support flexible decoding increases with the number of candidate predictors. Second, this approach may cause drifting. This is because, in practical video compression standards, quantized versions of Z_i, $\hat{Z}_i$, are sent to the decoder. Therefore, the reconstructed sources $\hat{X}_i = \hat{Z}_i + Y_i$ are not identical when different Y_i are used as predictors. Drifting may occur when $\hat{X}_i$ is used as reference for decoding future frames.

The H.264 video compression standard has defined SP- and SI-frames to support functionalities such as random access or error recovery that were originally supported by I-frames [2]. Essentially SP-frames follow the MCP coding approach we just discussed, but with modifications such that $\hat{X}_i$ can be identically reconstructed from different Y_i's using its corresponding Z_i (here Z_i corresponds to a primary or secondary SP-frame). This is achieved by using a different prediction loop from those in conventional P-frames (e.g., SP-frames compute the prediction residue in the transform domain whereas P-frames would compute that in the pixel domain [2]). However, this causes some penalty in coding performance, and the compression efficiency of SP-frames is in general worse than that of P-frames [2]. To support flexible decoding, different SP-frames bits (each corresponding to a different Y_i) need to be generated and sent to the decoder, similar to the conventional MCP coding, and therefore, H.264 SP-frames would incur a comparable amount of overhead as that in conventional MCP coding. It should be noted that most H.264 SP-frame applications assume the availability of feedback from the decoder (e.g., [3]), so that the encoder does know which predictor is available at the decoder and transmits only one of the Z_i. In short, H.264 SP-frames were not originally designed for the flexible decoding problem, where there exists uncertainty at the encoder about predictor availability at the decoder.

In this paper, we propose to address the general flexible video decoding problem using a distributed source coding (DSC) approach [4], [5]. Specifically, we propose a DSC-based video encoding algorithm where the encoder has access to the various predictors, Y_k, which will play the role of side information (SI) at the decoder, but there is uncertainty as to which one will be used for decoding. One of the main challenges for DSC-based applications has proven to be achieving competitive compression efficiency [6]. To address this challenge, our proposed algorithm incorporates novel macroblock modes and significance coding into the DSC framework. This, along with careful exploitation of correlation statistics allows to achieve significant performance improvements. Using forward/backward video playback as an example, we demonstrate the proposed algorithm can outperform, in terms of coding efficiency, technique based on MCP coding technique based on the ideas discussed above. Moreover, the proposed algorithm incurs only a small amount of drifting. In particular, DSC-coded macroblocks lead to the same reconstruction no matter which predictor candidate Y_k is used.

DSC has been studied extensively for enabling low-complexity video encoding, e.g., [7], [8]. However, there are significant differences between low-complexity encoding and flexible decoding, as summarized in Table I, which will lead us to a different solution. DSC has also been proposed to address compression of image-based rendering data/light fields to provide random access [9], [10]. This prior work, however, assumes that the encoder has knowledge of predictor status at decoder, notably through using feedback, while in our case the encoder needs to operate with unknown predictor status. A recent work [11] has proposed a DSC-based approach to address the problem of robust video transmission by allowing a video block to be decoded using more than one predictor blocks. While the general problem setting and philosophy is similar to ours, different assumptions are made. In particular, this work assumes encoder knows the probability that each predictor will be used, as determined by the packet erasure probability (whereas we assume all predictors are equally-likely to be used). This information is exploited to reduce the coding rate. In addition, the specific tools used are different from those proposed here. Our previous work [1] has also proposed to apply DSC to enable forward/backward video playback. This paper presents, however, a considerably different and significantly more efficient algorithm to address the general flexible decoding problem. Among the key improvements are the introduction of macroblock modes and significance coding, a different approach to exploit the correlation between source and side-information, a different way to partition the input symbols and estimate the source bit's conditional probability, and a minimum MSE dequantization.

This paper is organized as follows. In Section II we discuss how DSC can address flexible decoding. In Section III we present the proposed compression algorithm. Section IV presents the experimental results and Section V concludes the work.

II. Flexible decoding based on DSC: intuitions

In conventional MCP coding, the encoder computes a prediction residual $Z = X - Y$, between source X and predictor Y, and sends it to decoder (Figure 3.a). DSC approaches the same compression problem taking a "virtual communication channel" viewpoint [6], [12]. Specifically, X is viewed as an input to a channel with *correlation noise* Z, and Y is the output of the channel (Figure 3.b). Therefore, to recover X from Y, encoder would send parity information to the decoder. Compression is achieved by using fewer bits in the parity information than the number of bits that would be needed to send X directly. Note that this parity information does not depend on a specific Y being observed, and the encoder does not need Y to generate the parity information - only the statistics of Z are required for determining the amount of parity information to be sent. The decoder will be able to recover X as long as a sufficient amount of parity information has been received.

To understand how DSC can tackle flexible decoding, consider N virtual channels which each corresponds to a predictor candidate Y_i (Figure 3.c). Each channel is characterized by the correlation noise $Z_i = X - Y_i$. To recover X from any of these channels, the encoder could send an amount of parity information corresponding to the *worst* Z_i. Doing so, X can be recovered no matter which Y_i is available at the decoder. Note that encoder only needs to know the statistics of all the Z_i to determine the amount of parity information, and this is feasible since X and all Y_i are accessible at encoder. In particular, the encoder does not need to know which Y_i is actually present at decoder. Comparing with the MCP approach where the overhead to handle flexible decoding increases with N, in the DSC approach, the overhead depends mainly on the worst-case Z_i.

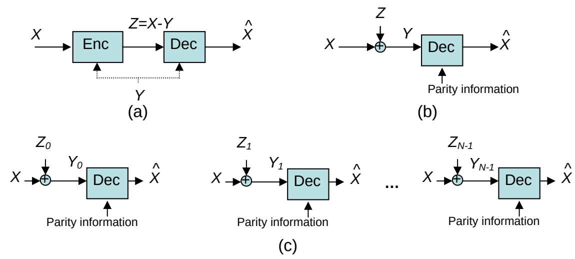

Fig. 3. Compression of input source X: (a) MCP coding; (b) DSC from the virtual channel viewpoint; (c) DSC approach to the flexible decoding problem.

III. Proposed algorithms

Figure 4 depicts the proposed video encoding algorithms to support flexible decoding based on DSC.

A. Motion estimation and macroblock classification

Each macroblock (MB) M in the current frame first undergoes standard motion estimation w.r.t. each candidate reference frame f_i, and the corresponding motion information (one per reference frame, f_i) is included in the bitstream. Denote A_i the best motion-compensated predictor for M obtained in f_i. If the difference between M and A_i is sufficiently small M may be classified to be in a skip mode w.r.t. f_i. In such cases, the overhead in including multiple prediction residues could be small, and M would be encoded using conventional MCP coding (similar to standard H.26X algorithms) w.r.t. the candidate reference frames which do not have skipping. However, for the majority of the macroblocks, there would be no skipping w.r.t. all f_i, and we would encode them using DSC.

Note that choosing between MCP and DSC for a given macroblock can be achieved using rate-distortion (RD) based mode selection (as in H.264): The RD costs of MCP and DSC are computed and the one achieving the minimum RD cost is selected. Such RD optimized mode decision algorithm can achieve a better coding performance, at the expense of requiring higher encoding complexity. In our comparison with H.263 (Section IV) we did not use this RD

TABLE I

COMPARE DSC-BASED LOW-COMPLEXITY ENCODING AND FLEXIBLE DECODING.

	DSC-based low-complexity encoding [7], [8]	**DSC-based flexible decoding**
Key objective	Low complexity video encoding for mobile video, video sensors, etc.	Generate robust bitstream to facilitate flexible decoding for forward/backward video playback, multiview video, etc.
Encoder complexity	Most target applications require low-complexity, real-time encoding.	Not primary issue. Most target applications use off-line encoding.
Encoder access to SI	SI not accessible by encoder due to complexity constraint.	Encoder has access to all the SI candidates. However, the exact one to be used at decoder is unknown to encoder.

optimized mode decision. As will be discussed, we implemented our proposed algorithms mainly based on H.263 coding tools.

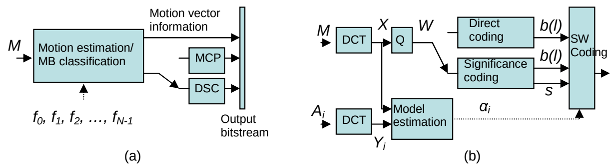

Fig. 4. (a) Proposed encoding algorithms; (b) Encoding macroblock M using DSC. Here "Q" denotes scalar quantization.

B. Direct coefficient coding (DCC)

For those macroblocks M to be encoded with DSC, we first apply standard 8×8 DCT to the pixel data to obtain the vector of transform coefficients X, and we then quantize X to obtain the quantization index W (Figure 4.b). This is similar to intra-frame coding in standard H.26X algorithms. Denote Y_i the DCT coefficient in A_i corresponding to W (recall A_i is the best motion-compensated predictor from each f_i). We compress W by exploiting its correlation with the worst case Y_i, so that it can be recovered with any Y_i that may be present at the decoder.

The quantized values of the K lowest frequency DCT coefficients (along a zig-zag scan order) are encoded with *direct* coefficient coding (DCC), and for the rest we use *significant* coefficient coding (SCC). In DCC, we form the k-th frequency coefficient vector by grouping together the kth $(0 \leq k \leq K-1)$ frequency coefficients from all the 8×8 blocks in a frame (except those in skip modes). Then each of these vectors is converted into a bit-plane representation, and the bit-planes are passed to a Slepian-Wolf (SW) coder, where inter-frame correlation is exploited to compress the bit-planes losslessly.

C. Significant coefficient coding (SCC)

The quantized values of the k-th highest frequency coefficients, $k \geq K$, are encoded using SCC. Specifically, we first use a *significance bit* s to signal if the quantized value of a coefficient is zero ($s = 0$) or not ($s = 1$), so that only the value of a non-zero coefficient needs to be sent to the decoder. The significance bits for the kth frequency coefficients from all the 8×8 blocks in a frame (except those in skip modes) are grouped together to form a significance bit-plane to be compressed by the SW coder. On the other hand, the non-zero coefficients are grouped together to form coefficient vectors where all the DCT frequencies are combined, as we found that the correlation statistics of non-zero coefficients are similar at different frequencies.

SCC is introduced as an alternative to DCC to reduce the number of source bits to be handled in SW coding. Specifically, assume DCC leads to L_k bitplanes for the the kth frequency coefficient vector. Therefore, each kth frequency coefficient contributes L_k source bits in DCC, regardless of whether the coefficient is zero or not. With SCC, a zero coefficient contributes one source bit (significance bit), while a non-zero coefficient contributes approximately $1+L_k$ bits. If p_k is the probability that the kth frequency coefficient will be zero, then the expected number of source bits using SCC is $1 \times p_k + (1 + L_k) \times (1 - p_k)$, and SCC can lead to source bits saving (compared with DCC) if this expected number is less than L_k, or equivalently if $p_k > \frac{1}{L_k}$ holds. Therefore, the value of K (where SCC starts) can be determined using this equation and some statistics of the video sequences.

D. Bit-plane compression

Bit-planes extracted from the K coefficient vectors produced in DCC along with those produced in SCC are compressed by a SW coder, starting from the most significant bit-planes. Denote a bit in the bit-plane at l-th level of significance by a binary r.v. $b(l)$, where $l = 0$ corresponding to the least significant level. That is, $b(l)$ is the l-th significant bit in the quantization index W. Binary r.v. $b(l)$ is to be compressed using Y_i and decoded bits $b(l+1), b(l+2), \ldots$ as side information. Specifically, this is performed by a low density parity check (LDPC) based SW encoder, which computes the *syndrome* bits from the original bit-planes and sends to the decoder [13].

E. Model and conditional probability estimation

SW decoding needs the conditional probability $p(b(l)|Y_i, b(l+1), b(l+2), \ldots)$ estimated from SI to aid recovering $b(l)$. The probability can be estimated as follows. The encoder estimates the conditional p.d.f. $f_{X|Y_i}(x|y_i)$ for each coefficient vector and for each candidate predictor. Assuming a system model $X = Y_i + Z_i$, and under the assumption of independence of Y_i and Z_i, we have

$$f_{X|Y_i}(x|y_i) = f_{Z_i}(x - y_i) \quad (1)$$

We assume Z_i is Laplacian distributed, i.e., $f_{Z_i}(z_i) = \frac{1}{2}\alpha_i e^{-\alpha_i |z_i|}$, and estimate the model parameters α_i at the encoder using maximum likelihood estimation (MLE) and send to the decoder. Note that in flexible decoding problem, the encoder can access to all the candidate SIs. Therefore, the model parameters can be readily estimated. This is not the case in typical DSC applications, where there are constraints on accessing side-information at the encoder making model estimation a non-trivial problem [14].

Given all the model parameters α_i, the decoder can estimate the conditional probability for any Y_i available at decoder using the following procedure (Figure 5). Denote $\tilde{W}$ the numerical value of the concatenation of the sequence of the decoded bits $b(l+1), b(l+2), \ldots$, i.e., $\tilde{W} = b(l+1) \times 2^0 + b(l+2) \times 2^1 + \ldots$. Given the decoded bits, the quantization index W can range only from $\tilde{W} \times 2^{l+1}$ to $\tilde{W} \times 2^{l+1} + 2^{l+1} - 1$. When $W \in [W_r, W_s]$, $b(l) = 0$, and when $W \in [W_t, W_u]$, $b(l) = 1$, where W_r, W_s, W_t, W_u are given by (in the cases when $\tilde{W} \geq 0$):

$$W_r = \tilde{W} \times 2^{l+1} \quad ; \quad W_s = \tilde{W} \times 2^{l+1} + 2^l - 1;$$
$$W_t = \tilde{W} \times 2^{l+1} + 2^l \quad ; \quad W_u = \tilde{W} \times 2^{l+1} + 2^{l+1} - 1. \quad (2)$$

Equations for $\tilde{W} < 0$ are similar. Therefore, the decoder can estimate the probabilities that $b(l)$ will be zero and one by integrating $f_{X|Y_i}(x|y_i)$ over the intervals $[X_r, X_s]$ and $[X_t, X_u]$ respectively, where $[X_r, X_s]$ is the inverse quantization mapping of $[W_r, W_s]$, and $[X_t, X_u]$ is that of $[W_t, W_u]$.

Note that each Y_i exhibits different levels of correlation with respect to $b(l)$. To ensure that $b(l)$ can be recovered with any of the predictor candidates Y_i, the encoder sends R syndrome bits to the decoder, where $R = \max\ R_i$, and R_i is number of syndrome bits required to recover $b(l)$ when Y_i is used as predictor. By doing so, each bit-plane can be exactly recovered no matter which Y_i is available at the decoder, and therefore, W can be losslessly recovered and X reconstructed to the same value when any of the Y_i is used as predictor. This eliminates drifting in DSC-coded macroblock.

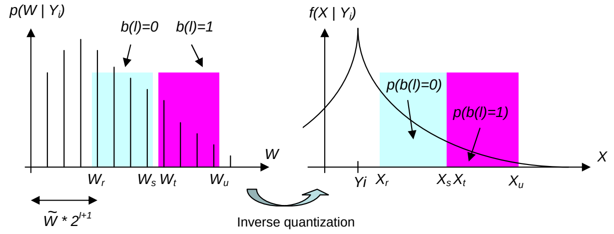

Fig. 5. Estimate the conditional probability $p(b(l)|Y_i, b(l+1), b(l+2), \ldots)$.

IV. Experimental results and discussion

This section presents simulation results using forward/backward playback as an example. We compare with a H.263 inter-frame coding system where both forward predicted P-frames and backward predicted P-frames are included, i.e., P-frames are duplicated. Note that this can be seen as a MCP-based solution for the forward/backward playback application (i.e., analogous to SP frames for this particular case). As discussed, such system may incur drifting, since in general the reconstructed forward and backward predicted P-frames are not identical. We compare the systems with GOP sizes equal to 15. Our implementation of the proposed algorithm is based on H.263 coding tools, e.g., half-pixel accuracy motion estimation, H.263 mode decision, etc. We also include all the overhead in communicating the macroblock mode, model parameters and the encoding rate of each bit-plane. We test the systems with sequences Coastguard and Stefan, which have considerable amounts of motion and picture details. Figure 6 shows the comparison results. As shown in the figure, the proposed algorithm outperforms the scheme with duplicate P-frames, with about 1-2dB gain in the medium/high picture quality range (33-36dB). We also show the results of H.263 intra-frame coding and inter-frame coding (same GOP sizes). Note that inter-frame coding cannot support flexible decoding. The results are shown here for reference only.

We also compare the approaches in terms of drifting with the following experiment: in forward decoding, a backward predicted frame is used for frame number 1 and as a reference for decoding the following frame. This is similar to what would happen when decoding direction switches. As shown in the results in Figure 7, the proposed algorithm incurs a negligible amount of drifting.

Additional results on multiview video (reported in [15]) further demonstrate the competitive performance of the proposed approach.

V. Conclusions and future work

We have proposed a video compression algorithm to support flexible decoding, based on DSC. The proposed algorithm integrates macroblock mode and significance coding to improve coding performance. In addition, we have discussed how correlation models can be used to estimate the conditional probability of the source bits at the decoder. Simulation results using forward/backward playback demonstrate the proposed DSC-based algorithm can outperform the MCP approach. In addition, the proposed system suffers only a small amount of drifting, as all the DSC-coded macroblocks would be identically reconstructed. Future work includes investigating improved model estimation methods.

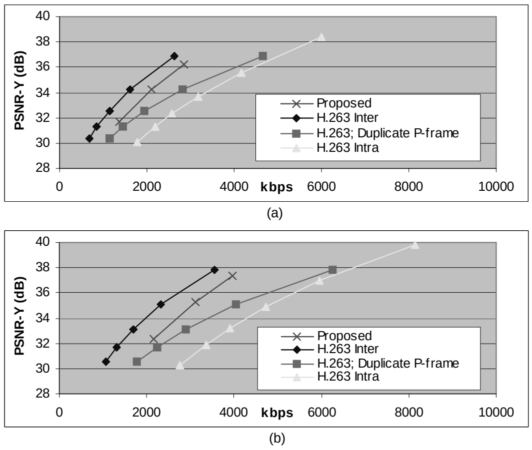

Fig. 6. Simulation results: (a) Coastguard; (b) Stefan. The sequences are in CIF format encoded at 30 fps, and results are reported for the first 30 frames.

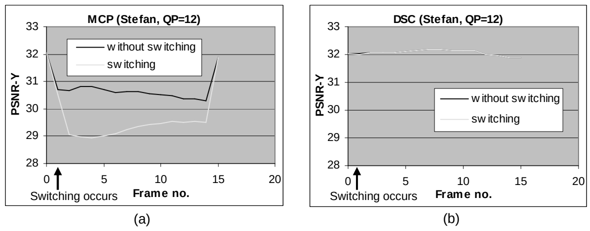

Fig. 7. Drifting experiment using Stefan sequence: (a) MCP; (b) DSC. The figure shows the PSNR of the reconstructed frames in the first GOP. Note that with DSC, the PSNR are almost the same in the switching and non-switching cases.

References

[1] N.-M. Cheung, H. Wang, and A. Ortega, "Video compression with flexible playback order based on distributed source coding," in *Proc. Visual Communications and Image Processing (VCIP)*, 2006.

[2] M. Karczewicz and R. Kurceren, "The SP- and SI-frames design for H.264/AVC," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 13, July 2003.

[3] X. Zhou, W.-Y. Kung, and C.-C. J. Kuo, "A robust H.264 video streaming scheme for portable devices," in *Proc. Int'l Symp. Circuits and Systems*, 2005.

[4] D. Slepian and J. Wolf, "Noiseless coding of correlated information sources," *IEEE Trans. Information Theory*, vol. 19, pp. 471–480, July 1973.

[5] S. Pradhan and K. Ramchandran, "Distributed source coding using syndromes (DISCUS): Design and construction," in *Proc. Data Compression Conference (DCC)*, 1999.

[6] B. Girod, A. Aaron, S. Rane, and D. Rebollo-Monedero, "Distributed video coding," *Proceedings of the IEEE, Special Issue on Advances in Video Coding and Delivery*, vol. 93, no. 1, pp. 71–83, Jan. 2005.

[7] R. Puri and K. Ramchandran, "PRISM: a new robust video coding architecture based on distributed compression principles," in *Proc. Allerton Conf. Communications, Control, and Computing*, Oct. 2002.

[8] A. Aaron, R. Zhang, and B. Girod, "Wyner-Ziv coding of motion video," in *Proc. Asilomar Conf. Signals, Systems, and Computers*, Nov. 2002.

[9] A. Jagmohan, A. Sehgal, and N. Ahuja, "Compressed of lightfield rendered images using coset codes," in *Proc. Asilomar Conf. Signals, Systems, and Computers*, 2003.

[10] A. Aaron, P. Ramanathan, and B. Girod, "Wyner-Ziv coding of light fields for random access," in *Proc. Workshop on Multimedia Signal Processing (MMSP)*, 2004.

[11] J. Wang, V. Prabhakaran, and K. Ramchandran, "Syndrome-based robust video transmission over networks with bursty losses," in *Proc. Int'l Conf. Image Processing (ICIP)*, 2006.

[12] Z. Xiong, A. Liveris, and S. Cheng, "Distributed source coding for sensor networks," *IEEE Signal Processing Magazine*, pp. 80–94, Sept. 2004.

[13] A. Liveris, Z. Xiong, and C. Georghiades, "Compression of binary sources with side information at the decoder using LDPC codes," *IEEE Communications Letters*, Oct. 2002.

[14] N.-M. Cheung, H. Wang, and A. Ortega, "Correlation estimation for distributed source coding under information exchange constraints," in *Proc. Int'l Conf. Image Processing (ICIP)*, 2005.

[15] N.-M. Cheung and A. Ortega, "Distributed source coding application to low-delay free viewpoint switching in multiview video compression," June 2007, submitted to *PCS07*.