Managing drift in DCT-based scalable video coding

Amy R. Reibman
AT&T Labs – Research
Florham Park, NJ 07932
amy@research.att.com

Léon Bottou
AT&T Labs – Research
Middletown, NJ 07748
leonb@research.att.com

November 21, 2000

Abstract

When compressed video is transmitted over erasure-prone channels, errors will propagate whenever temporal or spatial prediction is used. Typical tools to combat this error propagation are packetization, re-synchronizing codewords, intra-coding, and scalability. In recent years, the concern over so-called "drift" has sent researchers toward structures for scalability that do not use enhancement-layer information to predict base-layer information and hence have no drift. In this paper, we propose alternative structures for scalability that use previous enhancement-layer information to predict the current base layer, while simultaneously managing the resulting possibility of drift. These structures allow better compression efficiency, while introducing only limited impairments in the quality of the reconstruction.

1 Introduction

Compressed video, which uses predictive coding algorithms and variable-length coding, is sensitive to network impairments since these can cause error propagation. A single bit error or erasure can cause substantial degradation if no action is taken to stop or limit the extent of error propagation. Motion compensation allows the error to propagate both temporally and spatially. Because of this, there has been extensive effort in the video community to design new techniques that limit the extent of error propagation [1]. However, almost all attempts to limit error propagation decrease the coding efficiency, some dramatically so. To ensure the best operation of the video coder in an error-prone channel, the balance between resilience and efficiency must be managed carefully.

Scalable coding algorithms create a partitioning of the compressed bitstream into more and less important parts. This allows a natural combination with different mechanisms to prioritize network transport, for example, marking less important parts for early discard [2], applying unequal error protection [3], or facilitating rate-matching between encoder and network [4] [5]. When used in conjunction with such techniques, scalable video can be very resilient to network-introduced errors. However, this comes at the price of reduced compression efficiency.

In the standards arena, MPEG-2 scalability allows 3 methods of introducing scalability: SNR, spatial, and temporal [2]. The names can be somewhat misleading, since a strict implementation of MPEG-2 SNR scalability (SNRS) uses a prescribed one-loop encoder structure, while MPEG-2 spatial scalability (SS) using identical resolutions in the base and enhancement layer produces another form of SNRS using a two-loop encoder structure. In this paper, we refer to MPEG-2 SNRS as the strict implementation according to the standard, and MPEG-2 SS as either of the algorithms using the MPEG-2 spatial scalability syntax.

H.263 and MPEG-4 scalability [1] have fewer options, limiting the encoder options to those that force no drift if only the base-layer signal is received. (Here, *drift* is defined as the propagation of errors due to partial reception of the less important enhancement-layer information. This is in contrast to the more general term *error propagation*, which we use to include the result of partial reception of the more important base-layer information.) MPEG-4 Finely Granular Scalability (FGS) [4] further restricts the options by requiring that the encoding of the enhancement-layer not use techniques that would introduce drift, although it does have the advantage of bit-plane encoding.

In the research literature, Arnold et. al. [6] consider various loop structures designed to eliminate drift. However, all their coders are less efficient than MPEG-2 SNRS. Taubman and Zakhor [7] use a prediction structure for bit-plane encoder which does not use less important bit-planes of previous subbands to predict more important bit-planes of the current subband. This can reduce compression efficiency but limits error propagation. More recently there is continued effort on developing scalable video coders, all of which focuses on tolerating absolutely no drift [8],[9].

While scalable video coding algorithms are becoming more efficient at compressing the video, they lose compression efficiency because they ignore all enhancement-layer information when predicting the base layer. In particular, experiments show that with MPEG-2 SS, MPEG-4 and H.263 scalability modes all suffer from 0.5-1.5 dB losses for every layer [2], [10]. In the recent development of FGS [4, 11], the design process was very concerned with the complete elimination of all drift. As a result, the algorithm suffers severe compression inefficiency [12]. There have been some published attempts to address the compression inefficiency [12], but these are also based on the principle of tolerating no drift.

The recognition that the impact of errors in a one-layer coder can be probabilistically characterized by the encoder has lead to range of methods, including those of Wenger and Côté [13], Zhang et. al. [14], and Wu et. al. [15]. The impact of errors in a two-layer coder has also been probabilistically characterized in the encoding process by Zhang et. al. in [16]. However, while they consider the impact of errors in the enhancement-layer reconstruction, the encoder they optimize does not have the option of allowing drift to enter into the base-layer reconstruction. Hence, their scheme still has limited compression efficiency compared to a one-layer encoder.

Our work is based on the observation that if one can effectively manage error propagation in both a one-layer encoder and a two-layer encoder that does not allow the introduction of base-layer drift, it is possible to extend these techniques to an encoder that does allow the introduction of drift into the base layer. Such an encoder will have greater compression efficiency for higher bit-rates, with only slightly degraded resilience for the lower bit-rates.

In particular, despite the predominance of arguments in the literature maintaining that systems should not be designed to allow drift, there is some evidence that drift need not be eliminated completely. In Aravind et. al. [2], the MPEG-2 SNRS with 0.1% cell losses was invisible even when the base bit-rate was only 25% of the total bit-rate. Further, in [17], the loop structure with only partial mismatch control provided the best single-channel reconstruction quality for a given redundancy in a motion-compensated multiple description video coder.

In this paper, we consider structures for managing drift in a scalable DCT-based motion-compensated video coder. For a comprehensive management of drift, five features are necessary. Partial management of drift is possible with different subsets of these five features.

- First, there should be a means to introduce drift incrementally. This is straightforward to achieve by bit-plane encoding or by creating an embedded bitstream. This will only be effective, however, if used in conjunction with a mechanism in the transport provides more reliable delivery of the more important bit-planes to the receiver. Examples can be found in [3] and [5].

- Second, there should be a way for the encoder to measure the drift being potentially introduced, so it knows when drift is becoming significant.

- Third, there should be encoding options that can allow drift (ie, allow errors in the enhancement-layer to propagate into the base layer), while simultaneously keeping the amount of drift under control.

- Fourth, there should be a means to drastically reduce or eliminate drift without the need for a full I-frame.

- Fifth, there should be a system-level optimization, designed to maximize expected quality across all expected receivers. Inherent to this optimization, there must be some (possibly inaccurate) knowledge on the part of the encoder as to how many errors the channel will introduce, and how those errors will be introduced (gradually bit-plane by bit-plane, or suddenly when an entire packet of high-priority data is lost).

In this paper, we purposely don't consider structures like B-frames or P'-frames (which are similar to B-frames without forward prediction, and which are enabled by Reference Picture Selection (RPS) mode of annex N in H.263+ [1]), even though these structures naturally reduce drift by having fewer predictions made from partially correct data. Instead we focus on ways to manage drift within the predictive framework of P-frames. B- and P'-frames can easily be incorporated into our system-level structure, and indeed, a P'-frame is one way to limit the temporal extent of error propagation without an I-frame, even for a one-layer encoder.

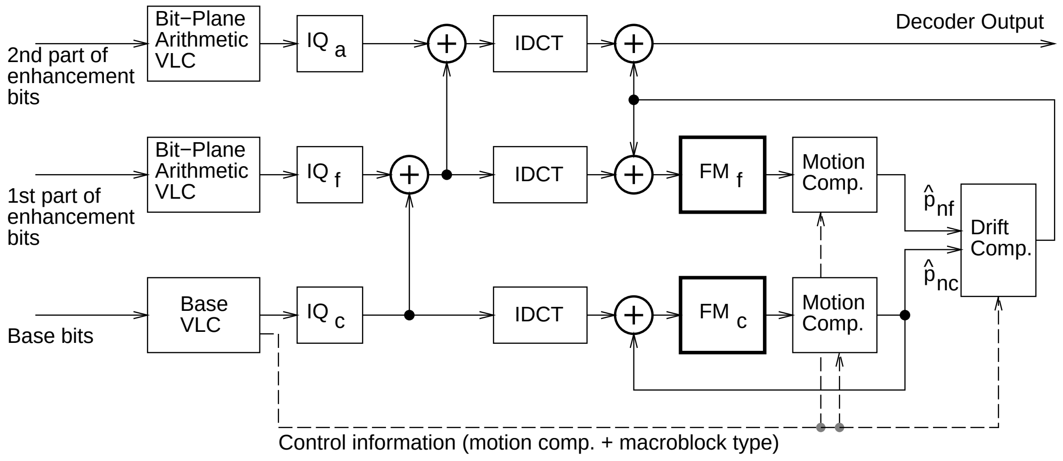


Figure 1: Two-loop decoder with drift control

While we explicitly consider only hybrid block-based DCT approaches to layering here, the basic ideas could also be applied to wavelet video coders that use some form of motion-compensated prediction.

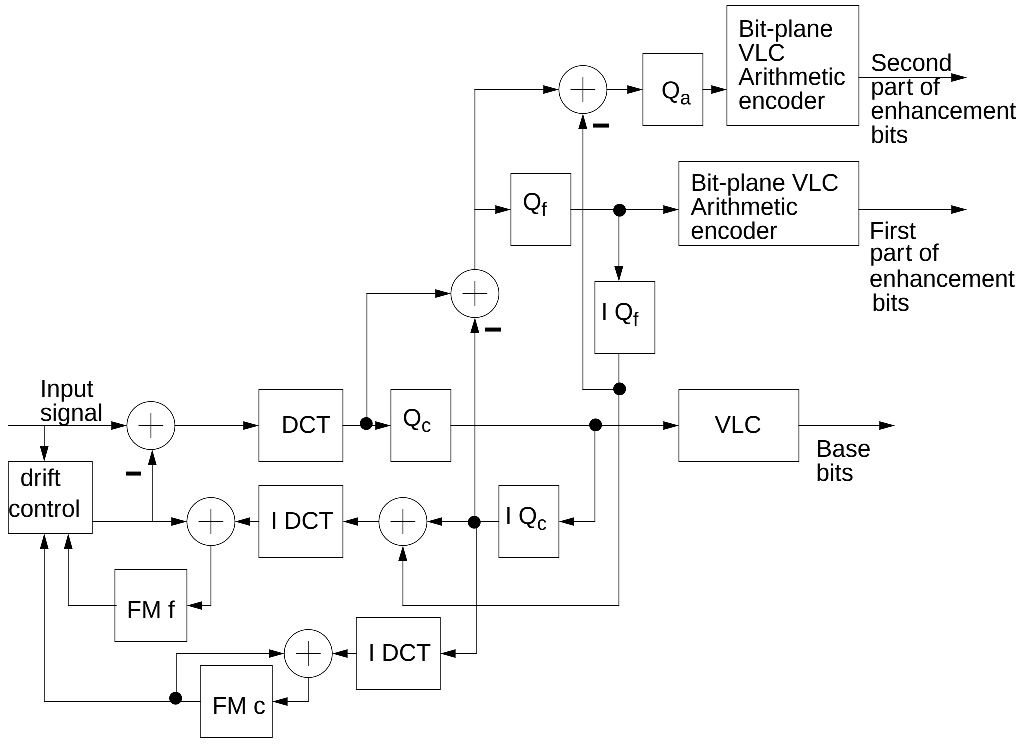
Section 2 presents the general scalable coder with drift control. In section 3 we present alternate optimization criteria for the scalable video encoder. Section 4 demonstrates that our coder performs better than three alternative encoders across most of the range of channel bit-rates, even though our coder suffers marginal performance degradation at the lowest bit-rates compared to the no-drift encoder alternatives.

2 Scalable DCT coder with drift control

The scalable DCT decoder with drift control shown in Figure 1 incorporates all five components necessary for effective drift management. It takes three levels of input. The base bits, with bit rate R_{nc}, are assumed to be always available. The first part of the enhancement bits, with bit-rate $R_{nf} - R_{nc}$, may not be received by the decoder, but if received, are used to predict the next frame. The second part of the enhancement bits, with bit-rate $R_{na} - R_{nf}$, may not be received, and is never used to predict the next frame.

Both the decoder and the encoder maintain two frame memories. The *coarse frame memory* depends only on the base bits and never drifts. The *fine frame memory* is updated by first combining both motion compensated frame memories, and then applying the base bits and the first part of the enhancement bits. The fine memory drifts when some of these enhancement bits are lost.

Let $\hat{p}_{nc}$ and $\hat{p}_{nf}$ be motion-compensated predictions from the coarse and fine memories for macroblock n. For each macroblock, the drift compensation box on Figure 1 combines the coarse and fine predictions according to a macroblock type information. The first option eliminates drift by taking the coarse prediction $\hat{p}_{nc}$ only (as in FGS).



Bit-plane VLC Arithmetic encoder
Second part of enhancement bits
Q_a
Bit-plane VLC Arithmetic encoder
First part of enhancement bits
Q_f
$I\ Q_f$
Input signal
DCT
Q_c
VLC
Base bits
drift control
I DCT
$I\ Q_c$
FM f
I DCT
FM c

Figure 2: Two-loop encoder with drift control

The second option allows drift by taking the fine prediction $\hat{p}_{nf}$ only (as in MPEG-2 SNRS). The third option reduces – but does not eliminate – drift by averaging both predictions $(\hat{p}_{nc} + \hat{p}_{nf})/2$. For simplicity, we only consider here these three options; introducing new combinations would naturally extend our scheme.

The scalable DCT encoder (Figure 2) tracks both frame memories under the assumption that all bits are received by the decoder. The encoder makes several decisions that affect the decoder drift in the fine memory. The first decision is the selection of a combination mode for the drift compensation. The second decision involves the number of bit-planes that might be used in the prediction loop; this is accomplished by adjusting the quantization Q_f relative to the final quantization Q_a. A third technique could be to apply a filter to the prediction from the coarse loop in order to smooth the discontinuities associated with prediction blocks that straddle macroblock boundaries; we do not explore this latter possibility in this paper.

Different images have different trade-offs between efficiency and resilience as a function of these drift control decisions. The encoder must make these decisions and send this information to the decoder. The encoder makes these choices on a macroblock basis with the goal of optimizing the total system performance as described in section 3.

To minimize the influence of drift in general, we use an embedded coder (described in section 4) to compress each individual frame. This allows more significant enhancement layer bit-planes to be received and decoded even if the network does not have sufficient bandwidth to send the entire enhancement layer. The base-layer VLC also relies on arithmetic bit-plane coding, but could also be implemented using the usual Huffman method. Macroblock type information and motion vectors are included in the base layer. We use the same motion vectors in both the base and enhancement layers.

3 Encoder optimization

The traditional (often implicit) optimization when designing a scalable coder is to minimize the maximum possible distortion at the decoder, subject to the constraint that the channel rate R is $\bar{R}_c \leq R \leq \bar{R}_a$. Typically, both $\bar{R}_c$ and $\bar{R}_a$ are known, although neither the instantaneous channel rate nor the average channel rate in some time interval is known. This maximum distortion is achieved for the minimum rate $\bar{R}_c$. Thus, optimizing using this criterion leads to a prediction process that does not tolerate any drift in the base layer. However, this also results in larger than necessary distortions at rates near $\bar{R}_a$. We explore here some alternate criteria for optimization, to achieve better compression at high rates without sacrificing too much quality at lower rates.

One optimization criterion is to minimize the distortion at the highest rate, subject to constraint that the drift at the lowest rate is kept below some value. This can be expressed as $\min\{D_a\}$ subject to $D_c \leq \bar{D}_c$ and the rate constraints

$$R_c \leq \bar{R}_c \text{ and } R_a \leq \bar{R}_a. \tag{1}$$

Here, R_c and R_a are the rates associated with the base bits, and all the bits, respectively, and D_c and D_a are the distortions of the associated reconstructions.

An alternate optimization criterion is to minimize the distortion averaged across all possible channel rates, subject to the rate constraints in (1). Determining the average distortion, however, requires knowledge of the probability distribution of the rates. This information is typically not available. However, a simple approximation is to minimize a weighted distortion $w_c D_c + (1 - w_c) D_a$, subject to the rate constraints in (1). The choice of w_c is influenced by the application.

These two constrained optimizations can be solved by converting them to unconstrained Lagrangian optimizations. The unconstrained optimization problems will have two or three Lagrange parameters, and can be solved using techniques similar to those in [13, 14, 16].

We take a simpler approach at this time. With the desire to keep the amount of drift limited, we decide how to code each macroblock based on the sum of the absolute error of the three possible predictions: S_{nc}, S_{nf} and $S_{n(c+f)}$. If the drift for this macroblock will be larger than some threshold, $S_{nc} \geq T_{drift}$, we choose to code the macroblock using the prediction $(p_{nc} + p_{nf})/2$. Otherwise, we choose the prediction that has the minimum sum of the absolute error.

4 Results

We compare the performance of our proposed coder to three systems: a one-loop encoder with no drift control, the encoder of Arnold et. al. [6], and a FGS encoder [4]. All encoders use an identical embedded DCT coder for compression.

4.1 Embedded DCT coder description

The DCT coefficients are first divided by the finest quantizer Q_a. The absolute value of the integer quotient can always be represented using twelve bits. The coder processes each block of 64 coefficients by iterating on the twelve bit-planes. It maintains three binary adaptive Z-coder [18] for each layer: one for the base bits, one for the first part of the refinement bits, and one for the second part of the refinement bits. The upper bit-planes are encoded using the Z-coder associated with the base bits. The lower bit-planes are encoded using a Z-coder associated with refinement bits, according to the choices of Q_c/Q_a and Q_f/Q_a.

Each bit-plane iteration processes each coefficient by coding whether the binary representation of its absolute value contains a 0 or a 1 for the current bit-plane. The sign bits are coded just after coding the first 1 of each coefficient. Such coefficients are named *significant*. Previous bit-plane coders [19, 20] take advantage of the wavelet transform structure by coding decisions addressing the significance of entire coefficient groups. The same result is achieved for zig-zag ordered DCT coefficients by coding a stopping decision after each significant coefficient. A positive stopping decision indicates that none of the remaining coefficients will be significant after processing this bit-plane. When it is known that some remaining coefficients are already significant, there is no need to code the stopping decision.

The Z-coder represents probability distributions using context variables. These variables are adapted after coding each binary decision. Each bit-plane is encoded using its own set of context variables so that it can be decoded without knowledge of the less-important bit-planes. Context variables might be reset after coding any number of blocks, for instance after each frame, or after each network packet.

4.2 Comparisons

We use each encoder to create a single encoding, containing the base layer and 4 bit-planes of enhancement-layer information. Each coder uses a fixed, identical, quantizer in the base layer. In the current implementation, our proposed coder sets $Q_f = Q_a$.

To obtain the performance comparisons in Figures 3 and 4 for the sequences *Hall monitor* and *Foreman*, respectively, we successively discard enhancement-layer bit-planes and decode the remainder. The x-axis shows the decoded bit-rate, and the y-axis shows the PSNR of the resulting decoder reconstruction as bit-planes are discarded. Also shown is the performance of the one-loop encoder with no loss (solid line). This provides an upper bound on the performance of the scalable coders.

The FGS coder performs poorly, especially for the mostly-still Hall sequence. The one-layer decoder with drift suffers a 2.3-2.5 dB degradation at the lowest bit-rate, compared to the drift-free Arnold and FGS decoders. Our proposed coder suffers about 1.1-1.2 dB performance degradation at the lowest bit-rate, but outperforms the other decoders almost everywhere. Our coder outperforms the Arnold coder for bit-rates greater than about 60-80 kilobits per second, and it outperforms the one-loop coder for all except the highest bit-rates. Table 1 shows the PSNR averaged across different channel rates, assuming a uniform distribution of rates between the

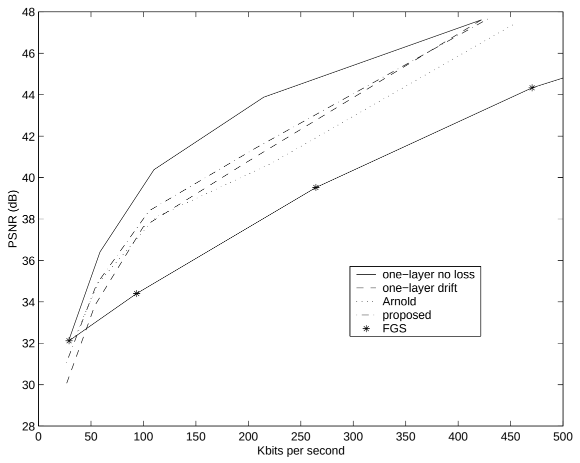


Figure 3: PSNR vs. rate for sequence *Hall monitor*.

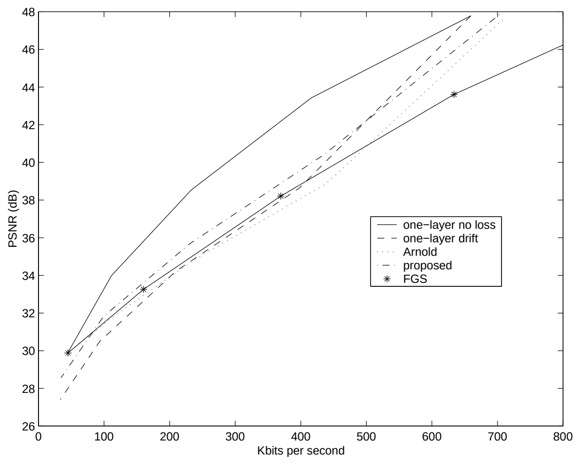


Figure 4: PSNR vs. rate for sequence *Foreman*.

smallest and the largest rate of the one-loop encoder. Although our implementation currently uses a simple optimization strategy, it still outperforms the other coders across the range of channel rates by 0.3-0.4 dB.

References

[1] M.-T. Sun and A. R. Reibman, ed., *Compressed Video over Networks*, Marcel Dekker, Inc, New York, NY, 2001.

[2] R. Aravind, M. R. Civanlar, and A. R. Reibman, “Packet loss resilience of MPEG-2 scalable video coding algorithms”, *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, no. 5, pp. 426-435, October 1996.

[3] P. A. Chou, A. Mohr, A. Wang, and S. Mehrotra, “FEC and pseudo-ARQ for receiver-driven layered multicast of audio and video”, *Data Compression Conference*, pp. 440-449, Mar. 2000.

seq.	One-loop	Arnold	Proposed	FGS
Hall	41.04	40.61	41.39	38.14
Fore	38.11	37.59	38.54	37.58

Table 1: PSNR averaged across channel assuming uniform distribution

[4] H. Radha et. al. "Fine-granular-scalable video for packet networks", *Packet Video Workshop*, New York NY, April 1999.

[5] R. Rejaie, M. Handley, D. Estrin, "Quality adaptation for congestion controlled video playback over the Internet", *Proceedings of ACM SIGCOMM '99* , Cambridge, MA., September 1999.

[6] J. F. Arnold, M. R. Frater, and Y. Wang, "Efficient drift-free Signal-to-Noise Ratio scalability", *IEEE Trans. Circuits and Systems for Video Technology*, vol 10, no. 1, pp. 70-82, Feb. 2000.

[7] D. Taubman and A. Zakhor, "Multirate 3-D subband coding of video", *IEEE Trans. Image Processing*, vol. 3, no. 5, pp. 572-588, Sept. 1994.

[8] U. Benzler, "Spatial scalable video coding using a combined subband-DCT approach", *IEEE Trans. Circuits and Systems for Video Technology*, vol. 10, no. 7, pp. 1080-1087, October 2000.

[9] M. Comanski, A. Luczak, and S. Mackowiak, "Spatio-temporal scalability for MPEG video coding", *IEEE Trans. Circuits and Systems for Video Technology*, vol. 10, no. 7, pp. 1088-1093, October 2000.

[10] K. Rose and S. L. Regunathan, "Towards optimality in scalable predictive coding", preprint.

[11] M. van der Schaar and H. Radha, "A novel MPEG-4 based hybrid temporal-SNR scalability for Internet video", *IEEE International Conference on Image Processing*, Vancouver CA, Oct. 2000.

[12] F. Wu, S. Li, and Y.-Q. Zhang, "DCT-prediction based progressive fine granularity scalable coding", *IEEE International Conference on Image Processing*, Vancouver CA, Oct. 2000.

[13] S. Wenger and G. Côté, "Using RFC2429 and H.263+ at low to medium bit-rates for low-latency applications", in *Packet Video Workshop '99*, New York, NY, April 1999.

[14] 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.

[15] D. Wu, Y. T. Hou, B. Li, W. Zhu, Y.-Q. Zhang, and H. J. Chao, "An end-to-end approach for optimal mode selection in Internet video communication: Theory and application", *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 6, pp. 977-995, June 2000.

[16] R. Zhang, S. L. Regunathan, and K. Rose, "Switched error concealment and robust coding decisions in scalable video coding", *IEEE International Conference on Image Processing*, Vancouver CA, Oct. 2000.

[17] A. R. Reibman, H. Jafarkhani, Y. Wang, and M. Orchard, "Multiple description coding for video using motion compensated prediction", *International Conference on Image Processing*, Kobe Japan, Oct. 1999.

[18] Bottou, L. and Howard, P. and Bengio, Y.", "The Z-Coder Adaptive Binary Coder", *Proc. IEEE Data Compression Conference 1998*, Snowbird, UT, 1998.

[19] Shapiro, J. M., "Embedded image coding using zerotrees of wavelets coefficients", *IEEE Transactions on Signal Processing*, vol. 41, pp. 3445-3462 December, 1993.

[20] Said, A. and Pearlman, W. A.", "A New, Fast, and Efficient Image Codec Based on Set Partitioning in Hierarchical Trees", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, no. 3, pp. 243-250, June, 1996.