ELSEVIER

Signal Processing 82 (2002) 1545–1558

SIGNAL PROCESSING

www.elsevier.com/locate/sigpro

Error resilience and recovery in streaming of embedded video

Sungdae Cho, William A. Pearlman *

Center for Next Generation Video Research, Rensselaer Polytechnic Institute, 110 Eighth Street, Troy, NY 12180-3590, USA

Received 1 September 2001

Abstract

The three-dimensional (3-D) SPIHT coder is a scalable or embedded coder that has proved its efficiency and its real-time capability in compression of video. A forward-error-correcting (FEC) channel (RCPC) code combined with a single automatic repeat request (ARQ) proved to be an effective means for protecting the bitstream. There were two problems with this scheme: the noiseless reverse channel ARQ may not be feasible in practice; and, in the absence of channel coding and ARQ, the decoded sequence was hopelessly corrupted even for relatively clean channels. In this paper, we introduce a new method of partitioning wavelet coefficients into spatio-temporal (s-t) tree blocks to achieve error resilience. Each of these s-t blocks corresponds to the full 3-D image region, because roots of these trees are wavelet coefficients taken at fixed intervals in the root low-frequency subband. Previously, we reported on grouping contiguous root subband coefficients to generate s-t tree blocks that correspond to local 3-D regions. The new procedure brings higher error resilience, since lost coefficients can be concealed with the surrounding coefficients even if some of the coded s-t blocks are totally missing. The bitstreams of the coded s-t blocks are packetized and encoded with a channel code to correct errors and to prevent decoding of erroneous data after errors are detected. Because the separately encoded s-t blocks produce embedded bitstreams, the packets from the bitstreams are interleaved to generate an embedded composite bitstream. The embedded property, whereby successive compressed bits convey successively smaller value information, suggests unequal error protection, where earlier bits are more strongly protected by the channel code than later bits. Therefore, unequal error protection is also incorporated into our video bitstreams to bring an even higher degree of resilience to channel bit errors. Our claims are supported by extensive simulations with decoding of the various 3-D SPIHT bitstreams compared to each other and to MPEG-2. Superiority to MPEG-2 in noiseless and noisy channels, under equal conditions with or without FEC, is clearly demonstrated by the results of these simulations.

Keywords: SPIHT; Error resilient video coding; Unequal error protection; Video compression; Embedded bitstream

1. Introduction

Wavelet zerotree image coding techniques were developed by Shapiro (EZW) [16], and further developed by Said and Pearlman (SPIHT) [14], and have provided unprecedented high performance in image compression with low complexity. Later, Kim et al. extended the idea to the three-dimensional SPIHT (3-D SPIHT) algorithm [10,11] that employed a temporal wavelet transform instead of motion compensation, and compared the result with the video compression algorithms which are generally used

* Corresponding author.
E-mail address: pearlman@ecse.rpi.edu (W.A. Pearlman).

0165-1684/02/$ - see front matter © 2002 Published by Elsevier Science B.V.
PII: S0165-1684(02)00301-8

nowadays, such as MPEG-2 and H.263. They showed promise of a very effective and computationally simple video coding, and also obtained excellent results numerically and visually.

However, wavelet zerotree coding algorithms are, like all algorithms producing variable length codewords, extremely sensitive to bit errors. A single-bit transmission error may lead to loss of synchronization between encoder and decoder execution paths, which would lead to a total collapse of decoded video quality. In a SPIHT compressed bitstream, as will be explained, a single error in a so-called significance map bit causes misinterpretation of all the succeeding bits. Because of the progressive characteristic (embeddedness) of the SPIHT bitstream, an early decoding error (often called decoding failure for convolutional codes) is particularly disastrous.

To achieve robust video over noisy channels, the work of Sherwood and Zeger [17,18] for the SPIHT algorithm was extended to progressive video coding by Kim et al. They have shown in Refs. [12,19] that the robustness is increased when we cascade the 3-D SPIHT coder with rate-compatible punctured convolutional (RCPC) channel coder [9] and automatic repeat request (ARQ) to protect the 3-D SPIHT bitstream from being corrupted by channel errors. This approach increases robustness, but is still susceptible to early decoding failure.

Another approach toward protecting video data from channel bit errors is to modify the 3-D SPIHT algorithm to work independently in a number of so-called spatio-temporal (s-t) blocks that are divided into fixed-length packets and interleaved to deliver a fidelity embedded output bitstream. This algorithm is called spatio-temporal tree preserving 3-D SPIHT (STTP-SPIHT) [4]. As a result, any bit error in the bitstream belonging to any one block does not affect any other block, so that higher error resilience against channel bit errors is achieved. Therefore, any early decoding failure affects the full extent of the GOF in the normal 3-D SPIHT, but in the STTP-SPIHT, the failure allows reconstruction of the associated region with lower resolution only. This algorithm gives excellent result in most cases, but may still experience very early decoding errors, resulting in lower resolution video in specific regions. This method was inspired by Refs. [5–7] in the field of image coding with the EZW algorithm [16].

In Ref. [2], Alatan et al. showed that the embedded image bitstreams can be delivered with error resilience maintained by dividing the bitstreams into three classes. They protect the subclasses with different channel coding rates of the RCPC coder [9], and improve the overall performance against channel bit errors.

In this paper, we use the idea in [4], but a different method for partitioning the wavelet coefficients into s-t block to solve those problems. Instead of grouping adjacent coefficients, we group coefficients at a fixed interval in the lowest subband, depending on the number of s-t blocks S. Then we track the s-t related trees of the coefficients, and merge them together. As a result, the s-t blocks of the STTP-SPIHT correspond to certain local regions, but the s-t blocks of our new grouping method correspond to the full group of frames (GOF) with lower resolution. This grouping method supports error concealment of lost coefficients using surrounding coefficients in the event of decoding failure. We call this algorithm error resilient and error concealment 3-D SPIHT (ERC-SPIHT) algorithm. As with STTP-SPIHT, we separate the subbitstreams into fixed length packets, interleave them to obtain an embedded composite bitstream, and encode them with a RCPC error-correction code with cyclic redundancy check (CRC). This kind of channel code not only corrects errors, but also allows detection of decoding failures, so that decoding can cease in substreams where decoding failures occur. Because the subbitstreams are embedded, the correctly received bits in each subbitstream can be decoded to provide a reconstruction at lower resolution or accuracy. Then, we also show how the 3-D SPIHT encoded video bitstreams can be implemented with unequal error protection by subdividing the embedded bitstreams, producing a hybrid coder which combines the ERC-SPIHT algorithm and unequal error protection. This method can protect against early decoding error with high probability, because we protect more strongly the beginning portion of the bitstream.

The organization of this paper is as follows: Section 2 shows error resilient 3-D SPIHT algorithm. Section 3 shows unequal error protection (UEP) of the 3-D SPIHT algorithm. Section 4 provides simulation results. Section 5 concludes this paper.

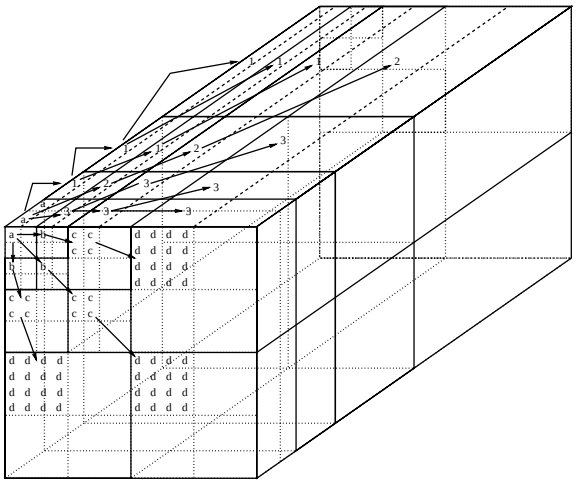

Fig. 1. Structure of the s-t relation of 3-D SPIHT compression algorithm.

2. Error resilient 3-D SPIHT algorithm

2.1. Background

We begin by explaining briefly the tree structure of the wavelet coefficients and the SPIHT coding algorithm that operates on this structure. Fig. 1 shows how coefficients in a 3-D transform are related according to their spatial and temporal domains. Character 'a' represents a root block of pixels ($2 \times 2 \times 2$), and characters 'b', 'c', 'd' denote its successive offspring progressing through the different spatial scales and numbers '1', '2', '3' label members of the same s-t tree linking successive generations of descendants. We used 16 frames in a GOF, therefore we have 16 different frames of wavelet coefficients. We can observe that these frames not only have spatial similarity inside each one of them across the different scales, but also temporal similarity between frames.

The SPIHT algorithm initially searches the lowest s-t subband for the so-called significant coefficients, whose magnitude is no less than a given threshold. The algorithm then searches the trees rooted in the lowest s-t subband for significant coefficients, and, in so doing, finds sets of coefficients whose magnitudes are less than the threshold (i.e., insignificant sets) by a single binary decision that is sent to the bitstream. The tree node that is the root of an insignificant set is put onto a list of insignificant sets (LIS). Whenever single coefficients are found to be insignificant, a '0' is sent to the bitstream and the location of the coefficient enters another list called the list of insignificant points (LIP). When a coefficient significant for the threshold is found, that finding is sent to the bitstream via a '1' along with its sign bit and its location is put onto a list of significant coefficients (LSP). After the algorithm traverses the root subband testing all such trees in this way, the threshold is halved and the process is repeated first by testing for significance at the lowered threshold of all coefficients in the LIP and then for all sets in the LIS. Those coefficients on the LSP at the previous higher threshold are refined in magnitude by sending their lower order magnitude bits in the bit plane (binary expansion) corresponding to the current threshold. The process continues through successive halving of the threshold, until the bit budget is exhausted. The decoder mimics the encoder's execution path, since it receives the significance decision bits which describe it.

The SPIHT bitstream comprises three kinds of bits: significance decision bits for single points or sets (called significance map bits); sign bits; and refinement bits. If errors occur in reception of sign or refinement bits, only the associated coefficients are reconstructed with value inaccuracies. On the other hand, if a significance map bit is in error, then the decoding algorithm deviates from the encoder's execution path and reconstructs the rest of the bitstream completely in error.

In Ref. [4], we reported the STTP-SPIHT compression algorithm. Fig. 2 shows the structure and the basic idea of the STTP-SPIHT compression algorithm. The STTP-SPIHT algorithm divides the 3-D wavelet coefficients into some number S of different groups according to their spatial and temporal relationships, and then to encode each group independently using the 3-D SPIHT algorithm, so that S independent embedded 3-D SPIHT substreams are created. These bitstreams are then interleaved in blocks. Therefore, the final STTP-SPIHT bitstream will be embedded or progressive in fidelity, but to a coarser degree than the normal SPIHT bitstream. In this figure, we show an example of separating the 3-D wavelet transform coefficients into four independent groups, denoted by a, b, c, d, each one of which retains the s-t tree structure of normal 3-D

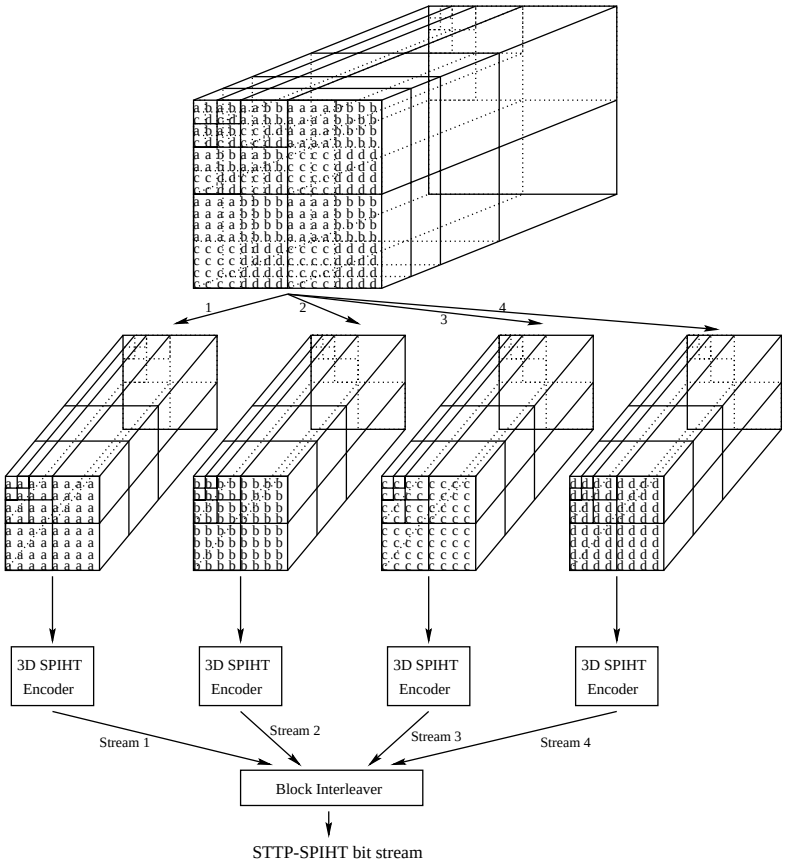

Fig. 2. Structure of the STTP-SPIHT compression algorithm.

SPIHT [10,11], and these trees correspond to the specific regions of the image sequences. The s-t block, which is denoted by a, matches the top-left portion in all frames of the sequence transform. The other s-t blocks correspond to the top-right, bottom-left, bottom-right fractions of the image sequences, and those s-t blocks are denoted by b, c, d, respectively. The normal 3-D SPIHT algorithm is just a case of $S = 1$, and we can flexibly choose S. When we choose the number of substream S for the image size of $X \times Y$, each of X and Y should be divisible by 2^{L+1}, where L is the number of decomposition levels. If the axis is not divisible by the number, we should extend the original image to be divisible. For the 352×240 Football sequence with 3 levels of decomposition, we can choose certain values of S from 1 up to 330.

2.2. Proposed method

STTP-SPIHT gave us excellent results in both noisy and noiseless channel conditions while preserving all the desirable properties of the 3-D SPIHT [4]. However, this method is also susceptible to early decoding error, and this error results in one or more small regions with lower resolution than the surrounding area. Sometimes, this artifact occurs in an important region. To avoid this, early decoding error should be prevented so as to guarantee a minimum quality of the whole region.

We can use a different method for partitioning the wavelet coefficients into s-t blocks to solve the problem. The 3-D SPIHT compression kernel is independently applied to each tree comprised of the wavelet coefficients in the lowest subband and the spatially

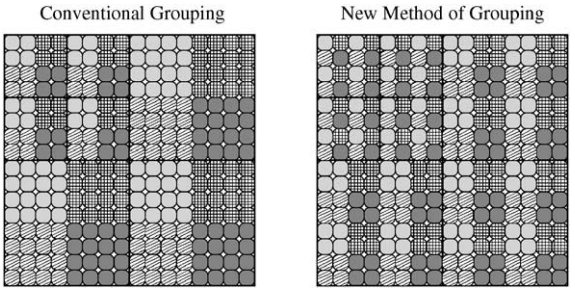

Fig. 3. Graphical illustration of the two methods (STTP-SPIHT and ERC-SPIHT).

related coefficients in the higher frequency subbands. The algorithm produces sign, location and refinement information for the trees in each pass. Therefore, we need to keep the s-t related trees to maintain compression efficiency of the 3-D SPIHT algorithm. However, we do not have to keep together the contiguous wavelet coefficients in the lowest subband, since the kernel is independently applied to each tree rooted in a single lowest subband coefficients and branching into the higher frequency subbands at the same s-t orientation. In our proposed algorithm, therefore, we group the lowest subband coefficients at some fixed interval instead of grouping adjacent coefficients. This interval is determined by the number of s-t blocks S, the image dimensions, and number of decomposition levels. Then, we track the s-t related trees of the coefficients, and merge them together. Fig. 3 graphically compares the two methods. The main advantage of the ERC-SPIHT is maintaining error resilience with coding efficiency. We also assign the same fixed rates to each substream. However, all of the subblocks contain similar information about each other, since each of the subblocks is composed of the coefficients not from a specific region, but from the whole region. Therefore, the fixed assignments of bitrates make more sense to our proposed method. Another nice feature of the ERC-SPIHT is that the very early decoding failure affects the whole region because the decoded coefficients would be spread out to the whole area along with the sequence, and the coefficients are concealed by the other surrounding coefficients which are decoded at a higher rate. When the decoding failure occurs in the same position, the quality of ERC-SPIHT is much better than that of STTP-SPIHT in visually and numerically (peak signal-to-noise ratio (PSNR)) because ERC-SPIHT algorithm itself has the function of error concealment. Therefore, the ERC-SPIHT no longer suffers from small areas which are decoded with a very low resolution.

Fig. 4 shows the recovery capability of ERC-SPIHT in a worst-case example of decoding failure. We used 352×240 "Football" and "Susie" sequence, and coded at 1.0 bit/pixel with ERC-SPIHT ($S = 16$). We assumed the decoding error occurred in the beginning of the substream number 2 (second packet) for the "Football" sequence and the substream number 7 (seventh packet) for the "Susie" sequence, so that one of the substreams is totally missing. As a result, all of the wavelet coefficients which correspond to the missing substreams are set to zeros. When the inverse wavelet transform is applied to the decoder, the corresponding regions are filled with black pixels, because the decoded pixel values are zeros. In this figure, (a) and (c) show the results from the ERC-SPIHT without error concealment, and (b) and (d) show them with error concealment. In this case, we just used the average values of surrounding coefficients for the missing coefficients only in the root subband. As we can see, in the case without error concealment, there are many black spots in the images. However, when we use concealment for the missing coefficients, we can recover the missing areas very well.

3. UEP of embedded bitstreams

3.1. UEP of the 3-D SPIHT

The 3-D SPIHT compression kernel is composed of two passes, a sorting pass and a refinement pass, repeatedly performed until the total bits produced meet the bit budget. From the sorting pass, sign bits and location bits are produced, and from the refinement pass, refinement bits are generated. The location bits are results of significance tests on sets of pixels, including singleton sets, and comprise what is often called the significance map.

As Alatan et al. did in Ref. [2], we classify the bits into two classes according to their bit error sensitivities. The sign bits and refinement bits can be classified as sign and refinement bits (SRB), and the location bits (LOB) can be classified by themselves.

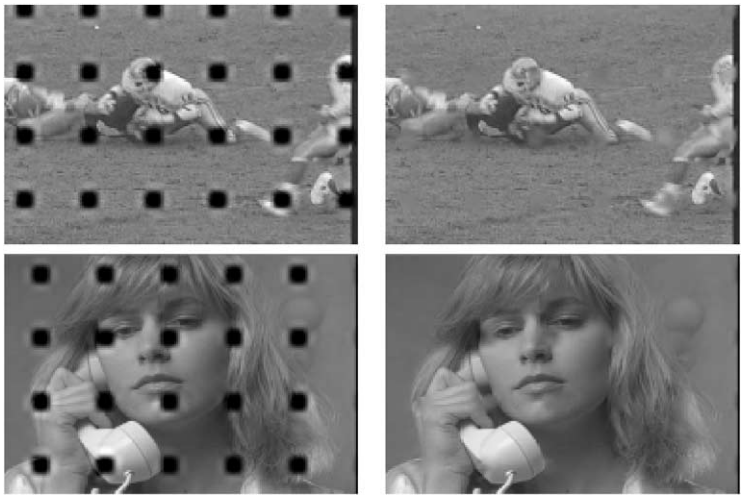

Fig. 4. Coded to 1.0 bit/pixel, with ERC-SPIHT ($P = 16$): (a) top-left: 352×240 "Football" sequence, without concealment (frame 1), PSNR = 17.96 dB; (b) top-right: 352×240 "Football" sequence, with concealment (frame 1), PSNR = 29.74 dB; (c) second row-left: 352×240 original "Susie" sequence (frame 16) without concealment after the seventh substream is missing, PSNR = 19.61 dB; and (d) Second row-right: with concealment (frame 16), PSNR = 33.83 dB.

If any bit error occurs in LOB, then the compressed bit stream is useless after the point where the bit error occurs. However, any bits in the SRB which are affected by channel bit errors do not propagate as long as the LOBs are error free. From our experimental results, the size of the SRB ranges from 20% to 25% of the original bitstream, depending on the rate.

Fig. 5 shows bit error sensitivity of a compressed bitstream encoded by the 3-D SPIHT algorithm only with no subsequent arithmetic coding. In this figure, we compressed the first 16 frames of the 352×240 monochrome "Football" sequence, and put 1 bit error to the compressed bitstream from beginning to the end of the bitstream. In this figure, the x-axis represents the bit error position, and the y-axis means the average PSNR of decoded sequence. As we can see, only one bit error in the beginning of the bitstream affects the whole bitstream, and makes the bitstream meaningless. In addition to that, we can see in (b)–(d) many positions where the average PSNRs are very high compared to the other positions. This class is called SRB, since these bits represent sign and refinement information of wavelet coefficients. The other class is also called LOB, because these bits contain the information of location of the wavelet coefficients and should be synchronized between encoder and decoder.

In addition, the 3-D SPIHT algorithm has an important property that all the compressed bits are positioned in the order of their contribution to value. This means that SPIHT produces a purely embedded or progressive bitstream, meaning that the later bits in the bitstream refine earlier bits, and the earlier bits are needed for the later bits to be useful. Fig. 6 represents average PSNR values versus bit-rates for the $352 \times 240 \times 16$ monochrome "Football" sequence. From this figure, we can see that the average PSNR value very rapidly increases when bit-rates are lower than 0.05 bpp, and most of the bits in this bit-rate are LOBs. Above this rate, PSNR increases at much more gradually with bit-rate. This result implies that the very beginning of the bitstream should be more strongly protected against channel bit errors than later portions of the bitstream.

For this reason, even if we have only the beginning part of the bitstream, we can still get a rough rendition of the source. But if we lose just a small portion at the beginning part of the bitstream containing LOB bits, then we cannot reconstruct anything from the bitstream. From this idea, we can further subpartition the

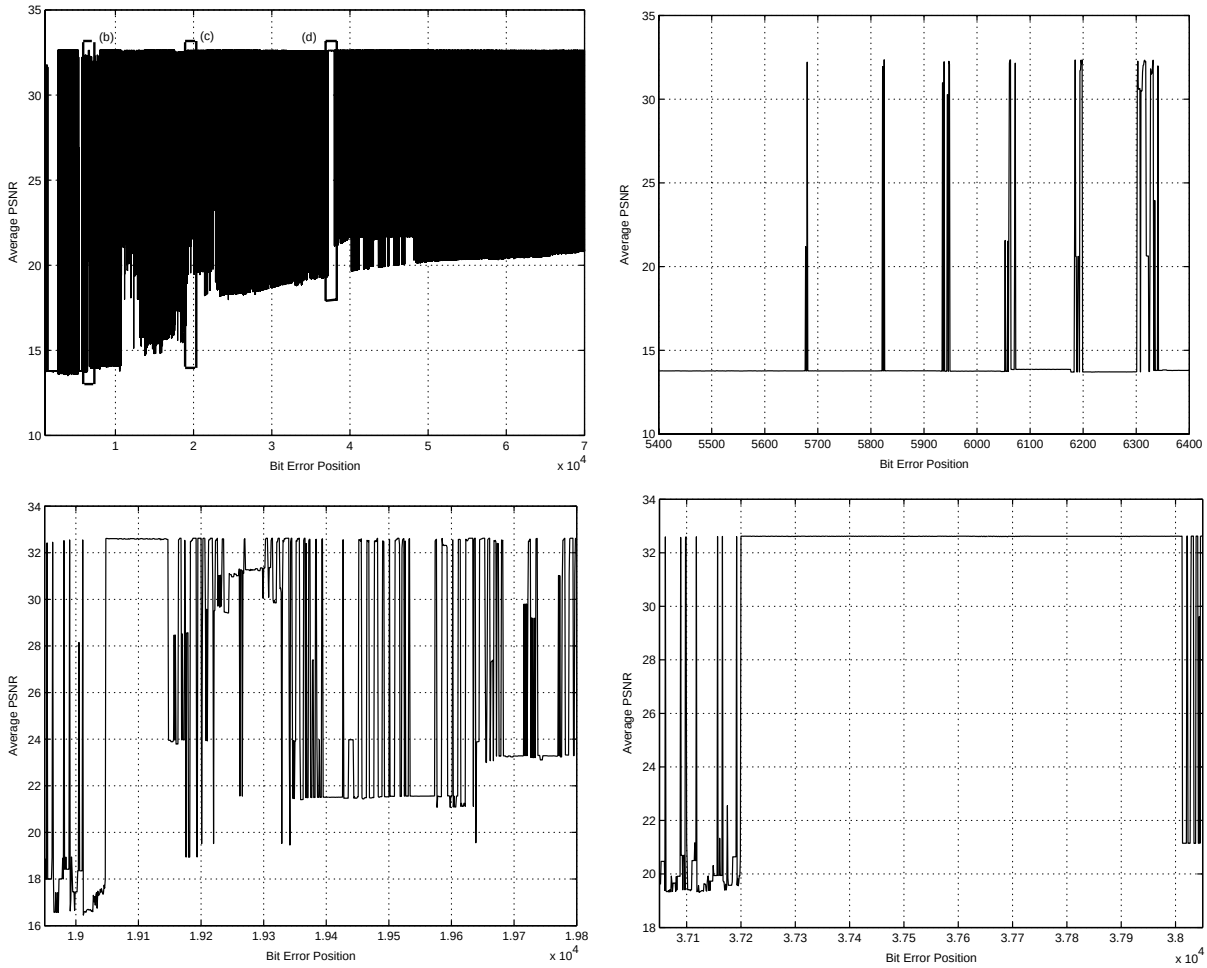

Fig. 5. Average PSNR versus bit error position to represent bit error sensitivity of the 3-D SPIHT compressed bitstream of 352×240 monochrome "Football" sequences (first 16 frames) coded 1.0 bpp (bit per pixel): (a) top-left: average PSNRs of 70,000 bit; (b) top-right: average PSNRs for the first sorting pass of 1000 bit sample; (c) bottom-left: average PSNRs for second refinement pass; and (d) bottom-right: average PSNRs for the third refinement pass.

LOB class into two classes LOB-a and LOB-b as Alatan et al. did in Ref. [2]. Each class corresponds to the earlier and later parts, respectively, of the bitstream.

We can utilize these facts to get higher error resilience against channel bit errors. We separate the SRB and LOB in the original bitstream, and transmit the SRB first with lowest error protection (highest channel code rate), then LOB-a and LOB-b, each with stronger protection (lower channel code rate) than SRB, but with LOB-a receiving a lower coder rate (higher protection) than LOB-b. The reason to send SRB bits first is that the decoder needs sign bits once LOB bits indicating significance are encountered. Figs. 7 and 8 graphically illustrate this idea. Fig. 7 shows the structure of the unequal error protection 3-D SPIHT (UEP-SPIHT) and how the bits are classified and combined together. As we can see, we do not use the arithmetic coding for SRB bits to avoid error propagation among the bits. Fig. 8 presents the bit-rate assignments according to their bit error sensitivities and importance. As we can see in this figure, LOB-a should be highly protected, because these bits are more important than others in terms of bit sensitivities and the order of importance, then LOB-b and

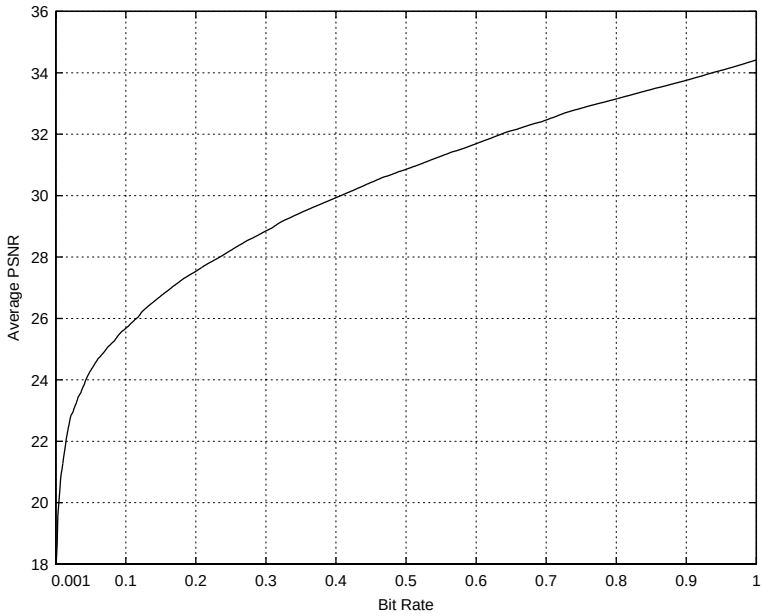

Fig. 6. Average PSNRs versus bit-rates for $352 \times 240 \times 16$ monochrome "Football" sequences.

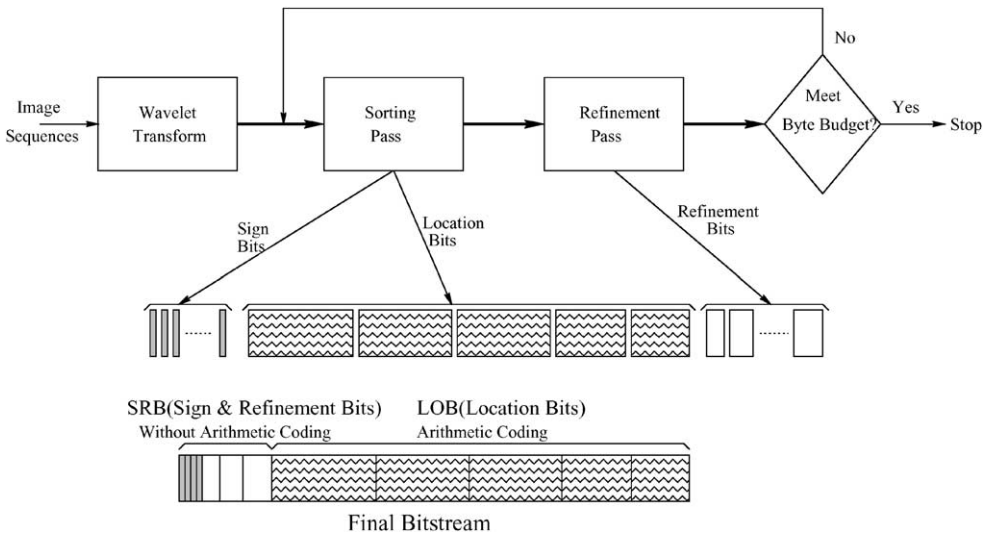

Fig. 7. UEP of the 3-D SPIHT algorithm.

SRB can be protected with successively higher channel coding rates.

As depicted in Fig. 8, we send the SRB bits first, then send the LOB bits. This means that while sending SRB bits, this bitstream is not progressive. However, after sending SRB bits, this bitstream is purely progressive, since all the SRB bits are stored in a buffer, and the sign bits in this buffer are accessed when LOB significance bits are encountered. As we mentioned before, the SRB segment ranges from 20% to 25% of the total bitstream for source code rates about 1 bpp (2.53 Mbps). The SRB size is relatively

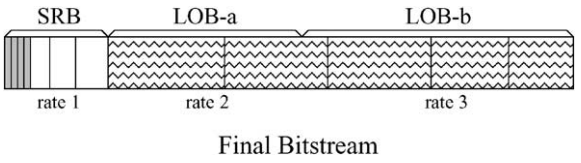

Fig. 8. Bit rate assignment of the UEP.

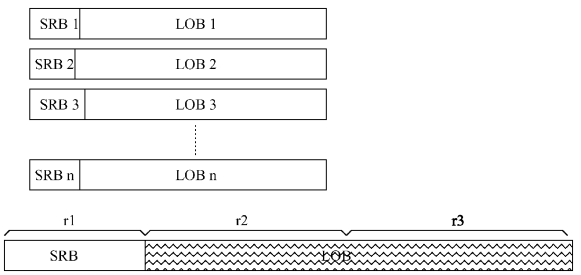

Fig. 9. Bitstream of STTP-SPIHT algorithm.

smaller at smaller bit-rates. Therefore, we get higher error resilience against channel bit errors while we sacrifice the progressiveness to a small extent. In the UEP-SPIHT header, we need just one negligible additional item of information, the SRB size.

3.2. UEP with the ERC-SPIHT

ERC-SPIHT gives excellent results in both noisy and noiseless channel conditions while preserving all the desirable properties of the 3-D SPIHT. These results are shown later in Table 2. However, this method still stops the decoding process for the substream wherein the first decoding error occurs. When such a decoding error occurs, we must discard the following bits, but can effectively conceal the affected region. Furthermore, the higher protection of the early part of the bitstream in the unequal error protection scheme makes the potentially disastrous early decoding error much less likely to occur.

To implement unequal error protection to ERC-SPIHT, we partition the subbitstreams according to their bit sensitivities and the order of importance. Fig. 9 shows us this idea. Every substream is divided into SRB and LOB segments, denoted by SRB1-SRB*n*, and LOB1-LOB*n*, where each is divided into its LOB-a and LOB-b segments. As we used in the 3-D SPIHT algorithm, we send all the SRBs first, and then send LOBs. In order to restore progressiveness to the composite bitstream, we used a packet interleaving/deinterleaving scheme for the LOB area to maintain progressiveness. The overhead of this method is the information bits which are saved in each subbitstream header to convey its SRB size. The decision of number of packets for each class is explained in a later section.

To decode the bitstream, the decoder reads the header first, and put the SRBs to buffer areas according to the information of the SRBs' size as the bits are arriving. Once all the SRBs have arrived, the decoder deinterleaves the LOBs according to the packet size, since the LOBs are sent as an interleaved bitstream, and decodes the bitstreams with SRBs together. Therefore, early portions of this bitstream are protected strongly with little loss of progressiveness.

3.3. Source/channel coding rate

Fig. 10 shows the system description of 3D/ERC-SPIHT with RCPC coder. The functions of packet interleaving and deinterleaving are needed for the ERC-SPIHT and STTP-SPIHT, but not for regular 3-D SPIHT. Before RCPC encoding, we partition the bitstream into equal length segments of N bits. Each segment of N bits is then passed through a CRC [3,13] parity checker to generate c parity bits. In a CRC, binary sequences are associated with polynomials of a certain polynomial $g(x)$ called the generator polynomial. Hence, the generator polynomial determines the error control properties of a CRC.

Next, m bits, where m is the memory size of the convolutional coder, are padded at the end of each $N + c + m$ bits of the segment and passed through the rate r RCPC channel coder, which is a type of punctured convolutional coder with the added feature of rate compatibility.

The effective source coding rate R_{eff} for the original 3-D SPIHT is given by

$$R_{\text{eff}} = \frac{Nr}{N + c + m} R_{\text{total}}, \tag{1}$$

where the unit of R_{eff} and R_{total} can be either bits/pixel, bits/s, or the length of bitstream in bits. The total number of packets M is calculated by R_{eff}/N, where R_{eff} is the bitstream length. In the case of unequal error

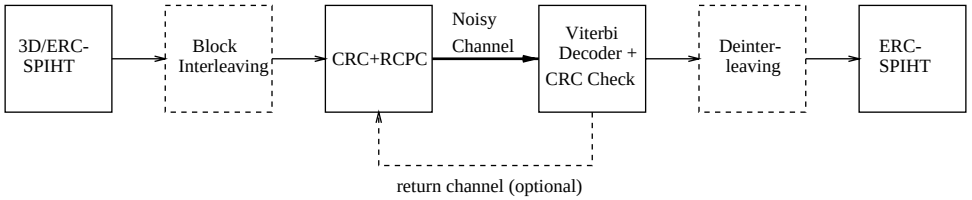

Fig. 10. 3-D/ERC-SPIHT with RCPC system framework.

Table 1
Relative percentages of source/channel (RCPC) bits at total transmission rate of 2.53 Mbps when BER is 0.01

		r	Source (%)	Channel (%)
3-D SPIHT		2/3	60	40
	SRB	4/5	14	5.5
UEP-SPIHT	LOB-a	4/7	14	13.5
	LOB-b	2/3	32	21

protection, we first get the $R_{\text{eff}_{\text{SRB}}}$ and M_{SRB} according to Eq. (1). Then, $R_{\text{eff}_{\text{LOB-a}}}$ and $R_{\text{eff}_{\text{LOB-b}}}$ can be calculated by

$$\frac{r_{\text{LOB-a}}}{N+c+m}R_{\text{LOB-a}} + \frac{r_{\text{LOB-b}}}{N+c+m}R_{\text{LOB-b}} = M - M_{\text{SRB}}, \tag{2}$$

where $R_{\text{LOB-a}} + R_{\text{LOB-b}} = R_{\text{total}} - R_{\text{SRB}}$. Therefore, we can get $R_{\text{LOB-a}}$ and $R_{\text{LOB-b}}$. Table 1 shows the relative percentage of source coding bits and channel coding bits using this equation for $352 \times 240 \times 48$ monochrome "Football" sequences at total transmission rate of 2.53 Mbps when the bit error rate (BER) is 0.01. In this table, we assigned the same bit budget to the normal 3-D SPIHT as in [12,19] according to the channel coding rate. For the ERC-SPIHT with unequal error protection, we use the same bit budget as in the case of normal 3-D SPIHT. We first assign same channel coding rate to the LOB-b ($r_{\text{LOB-b}}$), then assign one level lower rate to the LOB-a ($r_{\text{LOB-a}}$), and one level higher rate to the SRB (r_{SRB}). Then, ($R_{\text{LOB-a}}$), ($R_{\text{LOB-b}}$), and (R_{SRB}) are calculated by Eqs. (1) and (2).

4. Results

In our test of error resilience, we assume that the channel is binary symmetric (BSC). For MPEG-2, we use 15 frames in a group of pictures (GOP), and the I/P frame distance is 3 (IBBPBBP…). For the 3-D/ERC-SPIHT, 16 frames in a GOF are used, and a dyadic three-level transform using 9/7 biorthogonal wavelet filter [1] is applied to the image sequences. For the 3-D SPIHT and MPEG-2, we send the bitstream sequentially in 200 bit packets, and for the ERC-SPIHT, we interleave the streams in 200 bit packets to maintain embeddedness, and the receiver deinterleaves the bitstream to a series of substreams, each one of which is decoded independently. The algorithm is then tested using the $352 \times 240 \times 48$ monochrome "Football" sequences. The distortion is measured by the PSNR:

$$\text{PSNR} = 10\log_{10}\left(\frac{255^2}{\text{MSE}}\right) \text{ dB}, \tag{3}$$

where MSE denotes the mean squared error between the original and reconstructed image sequences. All PSNRs reported for noisy channels are averages over 50 independent runs.

Fig. 11 represents the frame by frame comparison of PSNRs of "Football" and "Susie" sequences coded with 1.0 bit/pixel (2.53 Mbps) without bit errors and forward error correcting. The solid line on top shows the PSNR values of normal 3-D SPIHT with "Susie" sequence, and the second solid line means PSNR value with "Football" sequence. The dashed dot lines represent ERC-SPIHT with $S = 16$, and dashed lines mean STTP-SPIHT with $S = 16$, and dotted lines show the MPEG-2 coded sequence. As we can see, there are just small losses (0.1–0.4 dB) with ERC-SPIHT with partitioning to 16 subgroups, and this curve follows the normal 3-D SPIHT very closely. In the "Susie" sequence, the PSNRs in STTP-SPIHT with $S = 16$ is about 1 dB worse than those of the normal 3-D SPIHT in every frame. This difference of PSNRs is mainly due to the inefficient rate allocation among

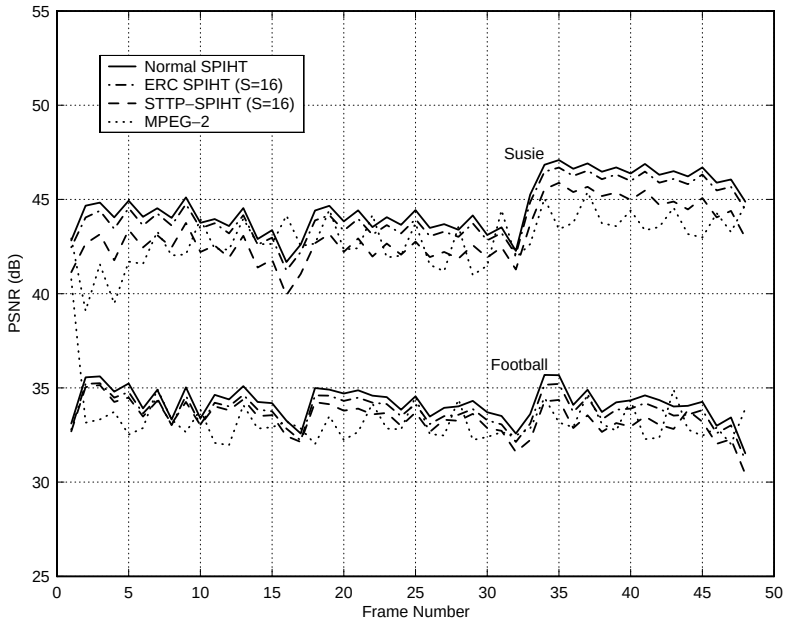

Fig. 11. Comparison of frame by frame PSNR (dB) of the "Football" and "Susie" sequences coded to 1.0 bit/pixel with normal 3-D SPIHT, ERC-SPIHT ($S = 16$), STTP-SPIHT ($S = 16$), and MPEG-2 without channel bit errors.

Table 2
Comparison of average PSNRs among 3-D SPIHT, ERC-SPIHT ($S = 4, 10, 16, 55$), STTP-SPIHT ($S = 4, 10, 16, 55$), and MPEG-2 at total transmission rate of 2.53 Mbps in noiseless channel

	Football		Susie	
MPEG-2	33.27		42.90	
3-D SPIHT	34.20		44.66	
	STTP-SPIHT	ERC-SPIHT	STTP-SPIHT	ERC-SPIHT
$S = 4$	34.04	34.04	44.37	44.50
$S = 10$	33.54	33.97	44.19	44.44
$S = 16$	33.35	33.79	43.13	44.26
$S = 55$	32.51	33.35	43.13	43.65

substreams, because the "Susie" sequence contains a large portion of still background extending to the full GOF. The rates of these substreams should be allocated unequally to reflect the unequal amount of activity in different substream regions. Therefore, ERC-SPIHT with $S = 16$ successfully follows the PSNR values of the normal 3-D SPIHT with just small PSNR differences.

Table 2 shows the comparison of average PSNRs among 3-D SPIHT, ERC-SPIHT ($S = 4, 10, 16, 55$), STTP-SPIHT ($S = 4, 10, 16, 55$), and MPEG-2 at total transmission rates of 2.53 Mbps of "Football" and "Susie" sequences with BER of 0. As we can see in this table, the average PSNRs of ERC-SPIHT with $S = 16$ are 0.44–1.13 dB higher than those of the STTP-SPIHT with $S = 16$, and still 0.07–0.25 dB higher than those of the STTP-SPIHT with $S = 10$.

In our simulation of error resilient video transmission with error correction capability, all of the normal 3-D SPIHT, ERC-SPIHT, STTP-SPIHT and MPEG-2 bitstreams were protected identically. As in previous works [12,17,19], we protected the 200 bit packets with the CRC, $c = 16$ bit parity check while generator polynomial $g(x) = X^{16} + X^{14} + X^{12} + X^{11} + X^8 + X^5 + X^4 + X^2 + 1$, and RCPC channel coder with constraint length $m = 6$. We focused on BER of $\varepsilon = 0.01$ and 0.001, because the BERs of most wireless communication channels are $\varepsilon = 0.01$–0.001. We set the total transmission rate R_{total} to 2.53 Mbps, $r = 2/3$ for $\varepsilon = 0.01$ and $8/9$ for $\varepsilon = 0.001$ for the equal error protection (EEP). In our case of UEP, we use a certain r only for LOB-b, and use the one level higher rate available to us for the SRB, and one level lower rate for the LOB-a with the same transmission rate of

2.53 Mbps. In the case of $\varepsilon = 0.01$, the RCPC rate for LOB-b $= 2/3$, and the rate for the SRB $= 4/5$, and the rate for the LOB-a $= 4/7$. Once we decide the channel coding rate, we can easily calculate the bit budget for the three classes using Eqs. (1) and (2).

At the destination, the Viterbi decoding algorithm [8,15] is used to convert the packets of the received bitstream into a 3-D/ERC-SPIHT and MPEG-2 bitstream. In the Viterbi algorithm, the "best Path" chosen is the one with the lowest path metric that also satisfies the checksum equations. In other words, each candidate trellis path is first checked by computing a $c = 16$ bit CRC. When the check bits indicate an error in the block, the decoder usually fixes it by finding the path with the next lowest metric. However, if the decoder fails to decode the received packet within a certain depth of the trellis, it stops decoding for that 3-D/ERC-SPIHT stream. For MPEG-2, which is not embedded and needs the full bitstream to see the whole frames, when decoding failure occurs, we can use one of two schemes. One is just to use the corrupted packet, and the other is to put all 0's to the corrupted packet. In this paper, we use the corrupted packet itself.

Fig. 12 shows the comparison of 352×240 "Football" sequence with RCPC and BER $= 0.01$. In this figure, (a) is the 352×240 original Football sequence (frame 15). Typical reconstructions of "Football" sequence at total transmission rate of 2.53 Mbps and channel BER of 0.01 with $S = 16$ and MPEG-2 are shown in Figs. 12(b)–(d). As we can see, in Fig. 12(b), the ERC-SPIHT stops decoding for the substream where decoding failure occurs. Therefore, any early decoding failure affects the whole region, and the coefficients are concealed by the other surrounding coefficients which are decoded at higher rates. In this example, the very early decoding error occurs in the 10th substream, but it is very hard to discern the affected region. However, the MPEG-2 decoded sequence in Fig. 12(c), the decoding failure affects some block, and the block is filled with some other picture's block. In the case of unequal error protection ERC-SPIHT, the probability of very early decoding error is very low because this method strongly protects the earlier part of the bitstream, and usually gives a nice result. Fig. 12(d) shows a typical example of a frame in the decoded sequence (frame 15).

Table 3 shows the comparison of average PSNRs among 3-D SPIHT, UEP-SPIHT, and UEP/ERC-SPIHT for 352×240 monochrome "Football" sequence at total transmission rate of 2.53 Mbps. As we can see, the average PSNR of UEP-SPIHT is about 2–5 dB higher than those of the equal error protection 3-D SPIHT (EEP-SPIHT). In the hybrid method, when BER is 0.01 the average PSNR is 30.03 dB, which is a little higher than that of equal error protection of ERC-SPIHT, and when BER is 0.001 the average PSNR is 31.75 dB. In addition to the higher PSNR, we have prevented very early decoding error using the lower channel coding rate for the LOB-a. The merit over the unequal error-protection of 3-D SPIHT is that we can get consistent results. In other words, the unequal error-protected 3-D SPIHT still stops decoding whenever decoding failure occurs, however, the unequal error protected ERC-SPIHT prevents very early decoding error effectively, and still operates until S decoding failures occur.

Table 3
Comparison of average PSNR in dB over 50 independent trials among 3-D SPIHT, UEP-SPIHT, and UEP/ERC-SPIHT at total transmission rate of 2.53 Mbps

BER	0.01	0.001
EEP 3D-SPIHT/RCPC	24.5	28.2
EEP ERC-SPIHT/RCPC	29.90	31.75
UEP 3-D SPIHT/RCPC	29.43	30.18
UEP ERC-SPIHT/RCPC	30.03	31.77
MPEG/RCPC	26.35	28.98

5. Conclusions

We have introduced a new partitioning method of wavelet coefficients to support error resilience and error concealment (ERC-SPIHT). The results show that this different method of grouping wavelet coefficient tree roots at a fixed interval and encoding these interleaved tree blocks independently provides high degrees of error resilience, error recovery, and coding efficiency. We also integrated UEP into the 3-D SPIHT and the ERC-SPIHT compressed bitstreams. The result is very promising while sacrificing progressiveness to a small degree. The most remarkable thing is that we effectively mitigate the early decoding error for the embedded bitstream. In this paper, all our work is done with channel bit errors over the binary symmetric channel (BSC). In addition, our method could be used in packet erasure channels, because a decod-

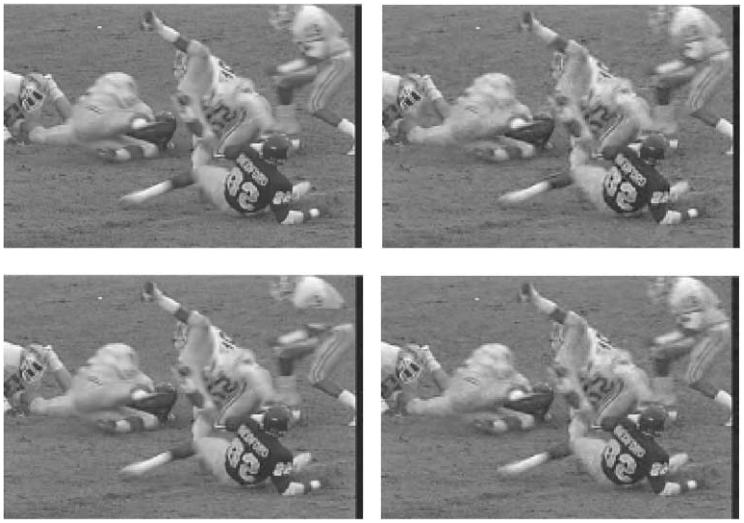

Fig. 12. 352 × 240 "Football" sequence (frame15): (a) top-left: original sequence; (b) top-right: using ERC-SPIHT ($n = 16$)/RCPC with BER = 0.01, PSNR = 29.64 dB; (c) bottom-left: using MPEG-2/RCPC with BER = 0.01, PSNR = 27.45 dB; and (d) bottom-right: using ERC-SPIHT with UEP with BER = 0.01, PSNR = 30.14 dB.

ing failure is equivalent in effect to a packet erasure in our scheme. For a bursty packet erasure channel, since subbitstreams are interleaved by packets, a burst of packet erasures or decoding failures would cause termination of decoding in a number of successive subbitstreams. In that case and in the case of higher packet erasure rate, the number S of subbitstreams should increase to minimize the effect of packet erasures on the reconstructed video. Further research is needed to evaluate our methods and modify them, if necessary, for fading and packet erasure channels.

References

[1] M. Antonini, M. Barlaud, P. Mathieu, I. Daubechies, Image coding using wavelet transform, IEEE Trans. Image Process. 1 (1992) 205–220.

[2] A. Aydin Alatan, M. Zhao, A.N. Akansu, Unequal error protection of SPIHT encoded image bit streams, IEEE J. Selected Areas Commun. 18 (June 2000) 814–818.

[3] G. Castagnoli, J. Ganz, P. Graber, Optimum redundancy check codes with 16-bit, IEEE Trans. Commun. 38 (January 1990) 111–114.

[4] S. Cho, W.A. Pearlman, A full-featured error resilient scalable video codec based on the Set Partitioning in Hierarchical Trees (SPIHT) algorithm, IEEE Trans. Circuits Systems Video Technol. 12 (March 2002) 157–171.

[5] C.D. Creusere, A family of image compression algorithms which are robust to transmission errors, Proc. SPIE 2668 (January 1996) 82–92.

[6] C.D. Creusere, Robust image coding using the embedded zerotree wavelet algorithm, Proceedings of the Data Compression Conference, March 1996, p. 432.

[7] C.D. Creusere, A new method of robust image compression based on the embedded zerotree wavelet algorithm, IEEE Trans. Image Process. 6 (10) (October 1997) 1436–1442.

[8] G.D. Forney Jr., The Viterbi algorithm, Proc. IEEE 61 (January 1994) 169–176.

[9] J. Hagenauer, Rate-compatible punctured convolutional codes (RCPC codes) and their applications, IEEE Trans. Commun. 36 (April 1988) 389–400.

[10] B.-J. Kim, W.A. Pearlman, An embedded wavelet video coder using three-dimensional set partitioning in hierarchical trees, Proceedings of the Data Compression Conference, March 1997, pp. 251–260.

[11] B.-J. Kim, Z. Xiong, W.A. Pearlman, Low bit-rate scalable video coding with 3D set partitioning in hierarchical trees (3D SPIHT), IEEE Trans. Circuits Systems Video Technol. 10 (December 2000) 1374–1387.

[12] B.-J. Kim, Z. Xiong, W.A. Pearlman, Y.S. Kim, Progressive video coding for noisy channels, J. Visual Commun. Image Representation 10 (1999) 173–185.

[13] T.V. Ramabadran, S.S. Gaitonde, A tutorial on CRC computations, IEEE Micro 8 (August 1988) 62–75.

[14] A. Said, W.A. Pearlman, A new, fast and efficient image codec based on set partitioning in hierarchical trees, IEEE Trans. Circuits Systems Video Technol. 6 (June 1996) 243–250.

[15] N. Seshadri, C. Sundberg, List Viterbi decoding algorithm with applications, IEEE Trans. Commun. 42 (1994) 111–114.
[16] J.M. Shapiro, Embedded image coding using zerotrees of wavelet coefficient, IEEE Trans. Signal Process. 41 (December 1993) 3445–3462.
[17] P.G. Sherwood, K. Zeger, Progressive image coding on noisy channels, Proceedings of the DCC, April 1997, pp. 72–81.
[18] P.G. Sherwood, K. Zeger, Progressive image coding for noisy channels, IEEE Signal Process. Lett. 4 (July 1997) 189–191.
[19] Z. Xiong, B.-J. Kim, W.A. Pearlman, Progressive video coding for noisy channels, Proceedings of the IEEE International Conference on Image Processing (ICIP '98), Vol.1, October 1998, pp. 334–337.