Error Resilient Compression and Transmission of Scalable Video

Sungdae Cho[a] and William A. Pearlman[a]

[a]Center for Digital Video and Media Research
Department of Electrical, Computer, and Systems Engineering
Rensselaer Polytechnic Institute
Troy, New York 12180-3590

ABSTRACT

Compressed video bitstreams require protection from channel errors in a wireless channel and protection from packet loss in a wired ATM channel. The three-dimensional (3-D) SPIHT coder has proved its efficiency and its real-time capability in compression of video. A forward-error-correcting(FEC) channel (RCPC) code[1] combined with a single ARQ (automatic-repeat-request) 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 first show how to make the 3-D SPIHT bitstream more robust to channel errors by breaking the wavelet transform into a number of spatio-temporal tree blocks which can be encoded and decoded independently. This procedure brings the added benefit of parallelization of the compression and decompression algorithms. Then we demonstrate the packetization of the bit stream and the reorganization of these packets to achieve scalability in bit rate and/or resolution in addition to robustness. Then we encode each packet with a channel code. Not only does this protect the integrity of the packets in most cases, but it also allows detection of packet decoding failures, so that only the cleanly recovered packets are reconstructed. This procedure obviates ARQ, because the performance is only about 1 dB worse than normal 3-D SPIHT with FEC and ARQ. Furthermore, the parallelization makes possible real-time implementation in hardware and software.

Keywords: video compression, video transmission, robust source coding, error resilient transmission, combined source-channel coding, 3-D wavelet transform, embedded wavelet coding

1. INTRODUCTION

Wavelet zerotree image coding techniques were developed by Shapiro (EZW),[2] and further developed by Said and Pearlman (SPIHT),[3] which provided higher performance and lower complexity image compression. Improved two-dimensional (2-D) zero-tree coding (EZW) by Said and Pearlman[3] has been extended to three dimensions (3-D EZW) by Chen and Pearlman[4] and shows promise of an effective and computationally simple video coding system without any motion compensation, and obtained excellent numerical and visual results. Later, Kim and Pearlman developed the three dimensional SPIHT (3-D SPIHT)[5] coding algorithm. The SPIHT algorithm not only outperforms EZW but also provides an alternative explanation and implementation of the EZW principle. However, wavelet zerotree coding algorithms are 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.

Numerous sophisticated techniques have been developed over the last several decades to make image transmission over a noisy channel resilient to errors. The one approach is to cascade a SPIHT coder with error control coding.[6,7] The idea is to partition the output bit stream from the SPIHT coder into consecutive blocks of length N. Then to each block c checksum bits and m zero bits are added to the end to flush the memory and terminate the decoding trellis at the zero state. The resulting block of $N + c + m$ bits is then passed through a rate r rate-compatible punctured convolutional (RCPC) coder.[1] However, this technique has the disadvantage of still being vulnerable to packet erasures that occur early in the transmission, and this packet erasure can cause a total collapse of the decoding process.

Another approach to protecting image bit streams from bit errors is to restructure the node test (NT) of the EZW algorithm. One approach is to remove dependent coding and classify the coding bit sequence into subsequences that can be protected differently using RCPC codes according to their importance and sensitivity. This type of technique was used by Man *et al.*[8]

The other approach is to make image transmission resilient to channel errors by partitioning the wavelet transform coefficients into groups and independently processing each group. This method was first reported by Creusere[9] for use with the EZW algorithm.

To achieve robust video over noisy channels, Kim *et al.*[10] utilized the same RCPC code as Sherwood and Zeger[6,7] with 3D SPIHT, and found that a single automatic repeat request(ARQ) was also necessary to adequately protect the bit stream. ARQ, however, may not be feasible in certain scenarios and has the unfortunate consequence of increasing traffic on already congested channels.

In this paper, we first extend Creusere's work[9,11,12] to the 3-D SPIHT coder. We modify the 3-D SPIHT algorithm to work independently in a number of so-called spatio-temporal (s-t) blocks, composed of packets that are interleaved to deliver a fidelity embedded output bit stream. Therefore a bit error in the bit stream belonging to any one block does not affect any other block. We then apply Kim *et al.*'s method[10] of forward error correction, borrowed from Sherwood and Zeger,[6,7] to every packet. Now we can detect decoding failures in any one packet and stop decoding, so that the rest of the block's bitstream will not corrupt the correct bits already decoded up to that point. Because this bit stream is embedded and an s-t block corresponds to a s-t region of video, the already decoded bits contribute a less accurate rendition of the region, while other regions corresponding to clean s-t blocks are reconstructed according to the rate of their bit streams. This less sensitive source coder substantially increases channel error robustness over a wide range of BER's (Bit Error Rate).

The organization of this paper is as follows: Section 2 shows how to make the 3-D SPIHT bitstream more robust to channel errors by breaking the wavelet transform into a number of spatio-temporal tree blocks. Section 3 shows error resilient video transmission against channel bit errors. Section 4 provides computer simulation results. Section 5 concludes this paper.

2. ERROR RESILIENT 3-D SPIHT VIDEO COMPRESSION

Variable Length Codes (VLC) are used in current video codecs for higher coding performance, transmission bit errors usually result in propagation throughout the decoded file. In zerotree algorithms such as SPIHT, when a single bit error occurs in a bit conveying significance of a coefficient or set of coefficients, the result is loss of synchronization between the encoder and decoder, giving erroneously decoded data beyond the point of the error. Therefore, a major concern of the designer is the control of errors so that reliable transmission can be obtained. We describe now a scheme, borrowed from Creusere's work with images,[9,11,12] for partitioning a three-dimensional wavelet transform into independent coding units, so that an error in any one unit does not affect the others.

Figure 1 shows how coefficients in a three-dimensional (3-D) transform are related according to their spatial and temporal domains. Character 'a' represents a root block of pixels (2x2x2), 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 spatio-temporal tree linking successive generations of descendants. We used 16 frames in a GOF(group of frames), therefore we have 16 different wavelet coefficient frames. We can observe that these coefficient frames have not only spatial similarity inside each frame across the different scale, but also temporal similarity between two frames, which will be efficiently exploited by the Spatial and Temporal Tree Preserved SPIHT (STTP-SPIHT) algorithm.

As shown in Figure 2 the basic idea of the error resilient 3-D SPIHT video compression algorithm is to divide the 3-D wavelet coefficients into some number n different groups according to their spatial and temporal relationships, then to encode each group independently using the 3-D SPIHT algorithm, so that n independent embedded 3-D SPIHT substreams are created. 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 spatio-temporal tree structure of normal 3-D SPIHT.[5]

The n bit streams are then interleaved in appropriate size units (e.g. bits, bytes, packets, etc.) prior to transmission so that the embedded nature of the composite bitstream is maintained. Therefore we can stop decoding at any compressed file-size or let run until nearly lossless reconstruction is obtained, which is desirable in many applications including HDTV.

Figure 3 illustrates the assembly of the final bit streams of 3-D SPIHT and STTP-SPIHT with $n = 4$. The bit streams are segmented into packets numbered by order of transmission. Encoding proceeds horizontally along the bit streams, but in STTP-SPIHT, the transmission occurs vertically downward and then from right to left along the bit streams to accomplish interleaving. Therefore, the final STTP-SPIHT bitstream will be embedded or progressive

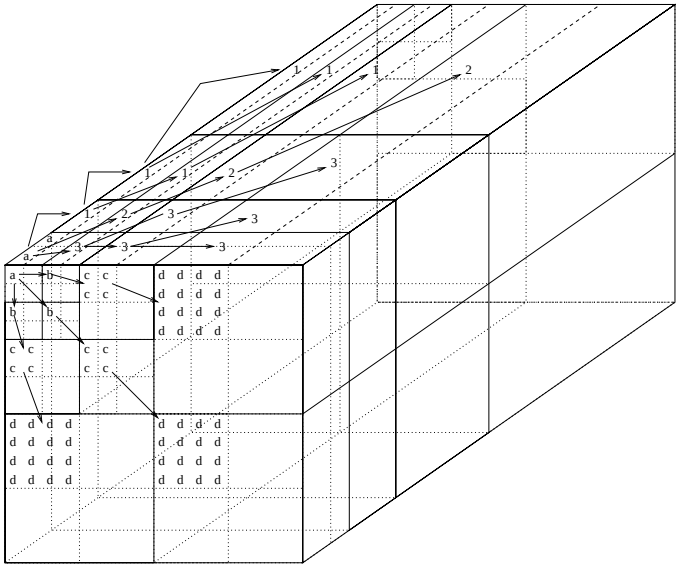

Figure 1. Structure of the spatio-temporal relation of 3-D SPIHT compression algorithm

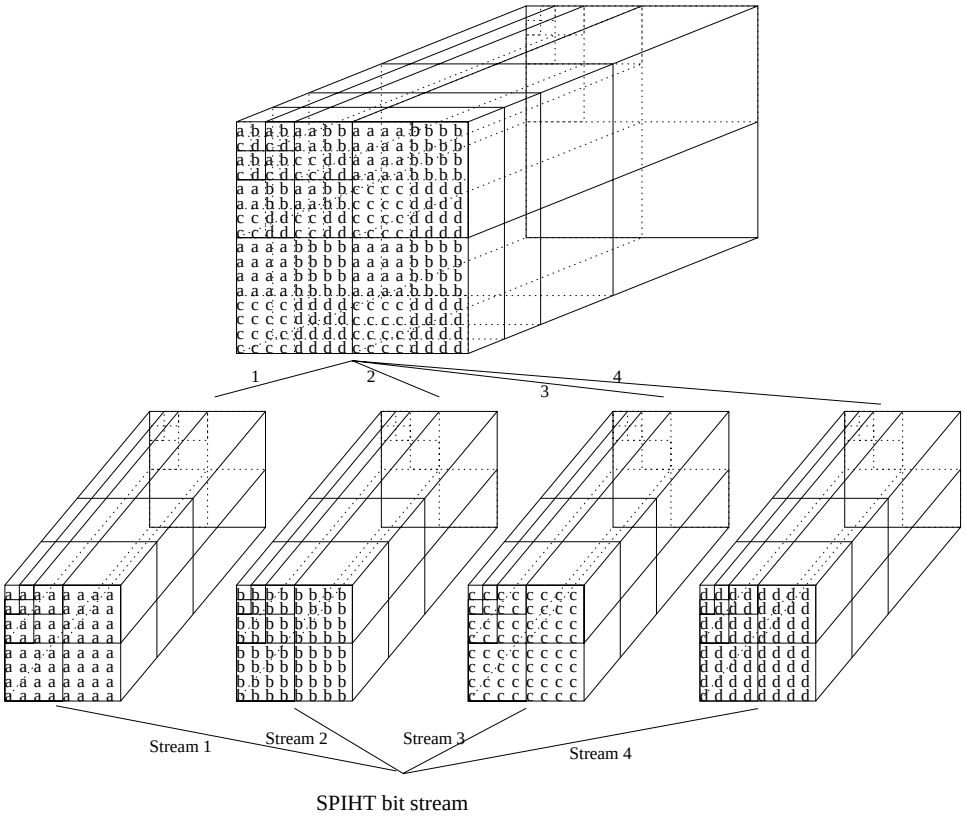

Figure 2. Structure of the spatio-temporal tree preserved 3-D SPIHT(STTP-SPIHT) compression algorithm

in fidelity, but to a coarser degree than the normal SPIHT bitstream. After transmitting the packets, the stream of normal 3-D SPIHT will be used by itself at the destination. For the STTP-SPIHT, however, the interleaved bit stream will be de-interleaved, and each substream will be processed independently and maintain more resilience against bit errors on the channel.

By coding the wavelet coefficients with multiple and independent bit streams, any single bit error affects only

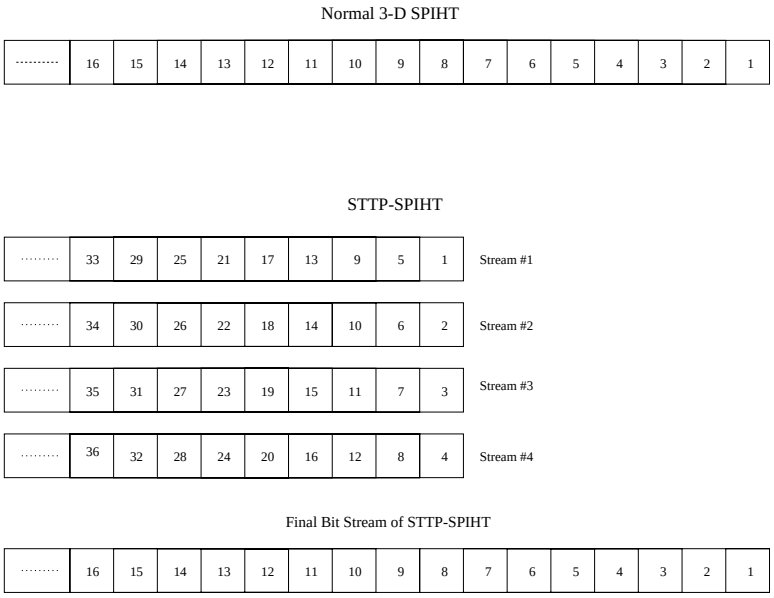

Figure 3. Comparison of bit streams between normal 3-D SPIHT and STTP-SPIHT with $n = 4$

one of the n streams, while the others are received unaffected. Therefore the wavelet coefficients represented by the corrupted bit stream are reconstructed at reduced accuracy, while those represented by the other streams are reconstructed at the full encoder accuracy.

3. ERROR RESILIENT VIDEO TRANSMISSION

In this section, we combine the block interleaving scheme of STTP-SPIHT with the forward error correcting code of Kim *et al.*'s work.[10] Figure 4 illustrates the overall system with an optional ideal return channel for ARQ. In our study, we shall make no use of ARQ. Kim *etal.*[10] cascaded the 3-D SPIHT coder with RCPC using a single request ARQ strategy.

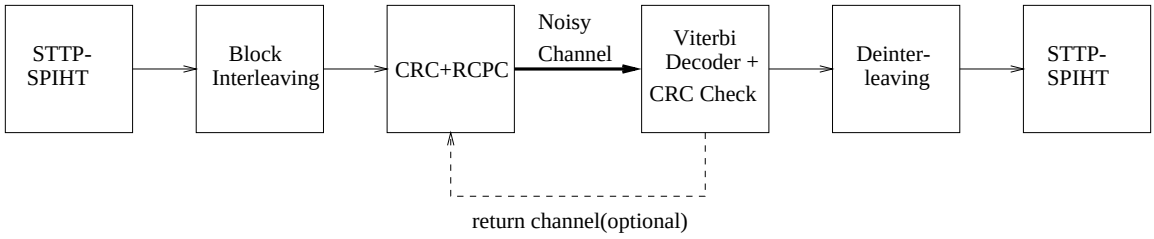

Figure 4. STTP-SPIHT/RCPC system framework

Figure 5 shows the system description of STTP-SPIHT/RCPC coder. In this figure, the RCPC coded stream is a segment of the channel encoded bit stream. We first partition the STTP-SPIHT bit stream into equal length segments of N bits. In our case, $N = 200$ bits. Each segment of size N bits is then passed through a cyclic redundancy code (CRC)[13,14] parity checker to generate $c = 16$ parity bits. In a CRC, binary sequences are associated with polynomials and codewords are selected such that the associated codeword polynomials *v(x)* of *N+c* bits segments are multiples 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 $N + c$ bits segment to flush the memory of the RCPC coder. Hence, early N bits of the STTP-SPIHT bit-stream are transformed into $N + c + m$ bits of the segment and passed through the rate r RCPC channel encoder, which is a type of punctured convolutional coder with the added feature of rate compatibility.

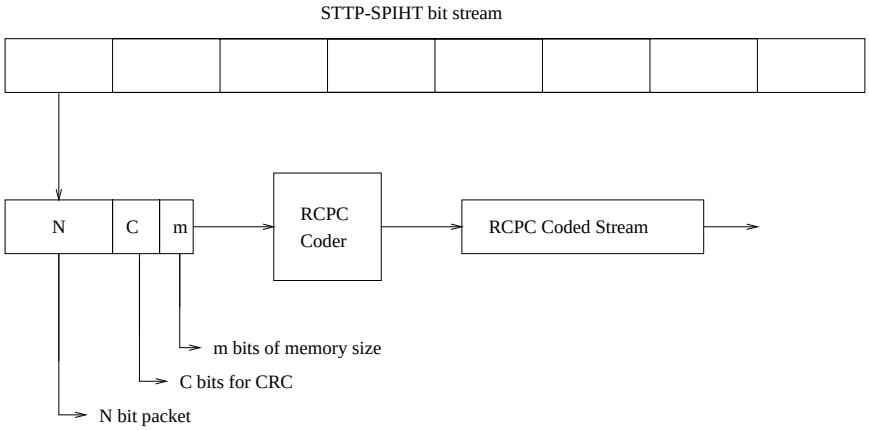

Figure 5. System description of STTP-SPIHT/RCPC coder

The RCPC rate r is defined as $k/n < 1$, where k is the number of input bits entering the RCPC coder and n is the number of corresponding output bits. Hence, the rate can be interpreted as the number of information bits entering the encoder per transmitted symbol.

Finally, the RCPC coded stream is then transmitted over the computer simulated binary symmetric channel (BSC).

Since the encoder adds some redundancy bits into 3-D SPIHT bit-stream according to the rate r of the RCPC, the effective source coding rate R_{eff} is less than the total transmission rate R_{total}, and is given by

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

where a unit of R_{eff} and R_{total} can be either bits/pixel, bits/sec, or the length of bit-stream in bits.

As we saw before, Figure 3 graphically illustrates the final bit stream comparison between the normal 3-D SPIHT and the STTP-SPIHT. For STTP-SPIHT, we interleaved by block as in Figure 3 to maintain embededdness.

At the destination, the Viterbi decoding algorithm[15,16] is used to convert the packets of the received bit-stream into a STTP-SPIHT bit-stream. 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 stream. The decoding procedure continues until either the final packet has arrived or a decoding failure has occurred in all n sub-bitstreams.

Figure 6 shows an example of decoding with $n = 16$ when decoding failure occurs at packet number 67. In that case, the normal 3-D SPIHT stops decoding at that point, but the STTP-SPIHT stops decoding only for the stream number 3, and continues to decode the packets of the other streams. After decoding, the normal 3-D SPIHT has only 66 clean packets, but the STTP-SPIHT has more clean packets, because the normal 3-D SPIHT just stops decoding at the first decoding failure but the STTP-SPIHT can accept up to 16 decoding failures in the worst case. In our example, substream number 3 has a decoding failure, and shorter length of bitstream after decoding and de-interleaving compared to other substreams. The result is the wavelet coefficients of low resolution in substream number 3 are surrounded by the other coefficients of higher resolution in the other substreams. Therefore the reproduction quality of the STTP-SPIHT is much better than that of the normal 3-D SPIHT, because STTP-SPIHT decodes many more clean bits compared to the normal 3-D SPIHT.

4. RESULTS

4.1. Robust Source Coding

We assume that the channel is binary symmetric (BSC) with error probability ϵ as shown in Figure 7. 16 frames in a GOF are used, and a three level transform using 9/7 biorthogonal wavelet filter[17] is applied to the image, and

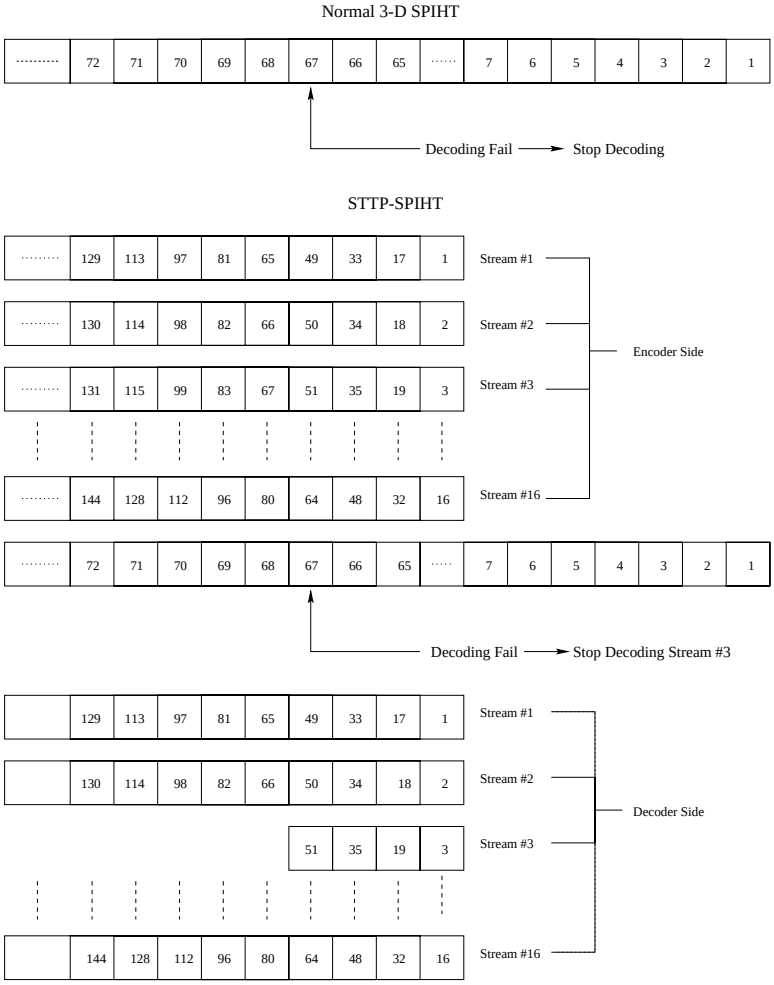

Figure 6. Example of decoding when decoding failure occurs at packet number 67

possible robust partitionings are evaluated. We put the SPIHT headers to each of the substreams to delineate the streams, and we assume the SPIHT headers are not corrupted from bit errors. We interleaved the streams in 200 bit packets to maintain embededdness. Figure 3 compares the encoded final bit streams of the normal 3-D SPIHT and the STTP-SPIHT. The receiver de-interleaves the bit stream to a series of substreams, each one of which is decoded independently. The algorithm is then tested using the 352 x 240 football sequence. The distortion is measured by the peak signal to noise ratio (PSNR)

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

where MSE denotes the mean squared error between the original and reconstructed images.

In our simulation of error resilient video transmission without error correction coding, we decoded the 3-D SPIHT and STTP-SPIHT bit streams to the end of the received bitstreams regardless of channel bit errors, since there was no mechanism to announce channel errors. Figure 8 shows the football sequences without any bit error using normal 3-D SPIHT (left), STTP-SPIHT ($n = 4$) (middle), STTP-SPIHT ($n = 16$) (right), and the PSNR's are 33.26 dB, 33.01 dB and 32.39 dB respectively. The successively lower PSNR's are due to the presence of successively more overhead bits needed to demarcate the sub-bitstreams. Figure 9 shows the effect in the presence of bit errors (BER = 10^{-4}) without error correction. The left image is the result of the normal 3-D SPIHT algorithm, where the PSNR is 11.19 dB and the visual result is awful. The following two images represent the sequences used with the STTP-SPIHT algorithm with $n = 4$ and $n = 16$, where the PSNR's are 14.72 and 17.93 dB respectively and the visual results

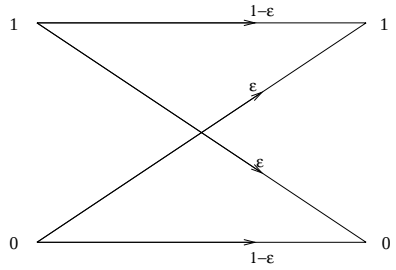

Figure 7. Transition probability diagram of binary symmetric channel

Figure 8. 352 x 240 "Football" sequence (frame15) coded to 1.0 bit/pixel without any bit error. Normal 3-D SPIHT, PSNR = 33.26 dB (left), STTP-SPIHT ($n = 4$), PSNR = 33.01 dB (middle), STTP-SPIHT ($n = 16$),PSNR = 32.39 dB (right)

noticeably improved for $n = 16$. In Figure 10, BER = 10^{-5} is used with normal 3-D SPIHT and STTP-SPIHT algorithm with $n = 4$ and $n = 16$, and the corresponding PSNRs are 18.25 dB, 23.67 dB, and 27.00 dB respectively. Here, at the lower error rate, where the normal 3D-SPIHT reconstruction is still badly corrupted, even $n = 4$ blocks offers a decent reconstruction and $n = 16$ blocks achieves a very good reconstruction.

Figure 11 illustrates the comparison of resulting average PSNR of "Football" sequence with wide range of BERs(10^{-9} - 10^{-3}) and different number of blocks n (1, 4, 16, 55, 110, 330), and coded with 1.0 bit/pixel. In this figure, when BER is high(10^{-4} - 10^{-3}), the average PSNR value of the STTP-SPIHT with $n = 330$ is still 0.79 dB higher than that of the normal 3-D SPIHT in the case of 100 times lower BER. In an error-free or very low bit error condition, the PSNR differences are 0.17 dB, 0.59 dB, 1.16 dB, 1,38 dB, 1,95 dB with number of blocks $n = 4, 16, 55, 110, 330$, respectively, and these differences remain below BER = 10^{-7}. However, if the BER is higher than 10^{-7}, the PSNR differences are much large, ranging from 1.9 dB to 19.76 dB depending on the number of blocks n, and the BERs.

Table 1 shows the actual PSNR values of Figure 11. Table 2 shows the PSNR differences between normal 3-D SPIHT and STTP-SPIHT ($n = 4$, 16, 55, 110, 330). As we can see, the performance of the normal 3-D SPIHT is

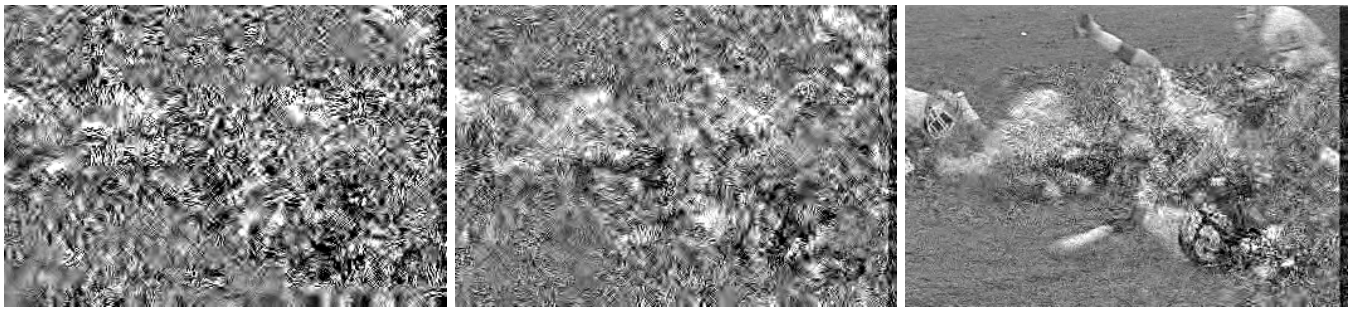

Figure 9. 352 x 240 "Football" sequence (frame15) coded to 1.0 bit/pixel with BER = 0.0001. Normal 3-D SPIHT, PSNR = 11.19 dB (left), STTP-SPIHT ($n = 4$), PSNR = 14.72 dB (middle), STTP-SPIHT ($n = 16$),PSNR = 17.93 dB (right)

Figure 10. 352 x 240 "Football" sequence (frame15) coded to 1.0 bit/pixel with BER = 0.00001. Normal 3-D SPIHT, PSNR = 18.25 dB (left), STTP-SPIHT ($n = 4$), PSNR = 23.67 dB (middle), STTP-SPIHT ($n = 16$),PSNR = 27.00 dB (right)

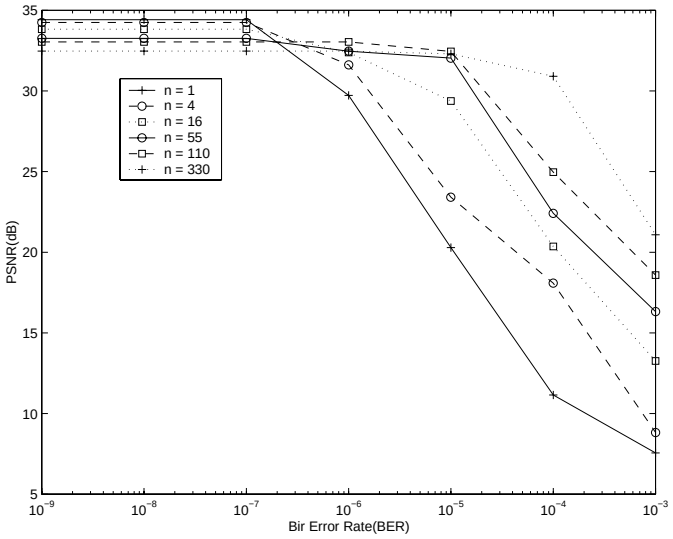

Figure 11. Comparison of average PSNR(dB) of "Football" sequence with different BERs and number of blocks n (1, 4, 16, 55, 110, 330), and coded to 1.0 bit/pixel with STTP-SPIHT without channel code.

better than that of STTP-SPIHT below BER = 10^{-7}, then degrades rapidly as the BER becomes higher. In lower error conditions (BER <= 10^{-7}), the performance of STTP-SPIHT gets worse as n increases due to the header overhead in each substream, because we put the 3-D SPIHT header into each substream to delineate the substreams.

BER	10^{-9}	10^{-8}	10^{-7}	10^{-6}	10^{-5}	10^{-4}	10^{-3}
Normal 3-D SPIHT	34.42	34.42	34.42	29.72	20.29	11.15	7.56
STTP-SPIHT(n=4)	34.25	34.25	34.25	31.62	23.41	18.08	8.82
STTP-SPIHT(n=16)	33.83	33.83	33.83	32.39	29.37	20.36	13.26
STTP-SPIHT(n=55)	33.26	33.26	33.26	32.47	32.04	22.42	16.32
STTP-SPIHT(n=110)	33.04	33.04	33.04	33.04	32.45	24.97	18.58
STTP-SPIHT(n=330)	32.47	32.47	32.47	32.47	32.30	30.91	21.08

Table 1. Comparison of average PSNR(dB) of "Football" sequence with different BERs and number of blocks n (1, 4, 16, 55, 110, 330), and coded to 1.0 bit/pixel with STTP-SPIHT (no channel code).

4.2. Combined Source and Channel coding

In our simulation of error resilient video transmission with error correction capability, we used the same set of parameters for the CRC parity checker and RCPC channel coder[7,6,10]: $N = 200$, $c = 16$, and $m = 6$. We focused on

BER	10^{-9}	10^{-8}	10^{-7}	10^{-6}	10^{-5}	10^{-4}	10^{-3}
Normal 3-D SPIHT	0	0	0	0	0	0	0
STTP-SPIHT(n=4)	-0.17	-0.17	-0.17	+1.9	+3.12	+6.93	+1.26
STTP-SPIHT(n=16)	-0.59	-0.59	-0.59	+2.67	+9.08	+9.21	+5.7
STTP-SPIHT(n=55)	-1.16	-1.16	-1.16	+2.75	+11.75	+11.27	+8.76
STTP-SPIHT(n=110)	-1.38	-1.38	-1.38	+3.32	+12.16	+13.82	+11.02
STTP-SPIHT(n=330)	-1.95	-1.95	-1.95	+2.75	+12.01	+19.76	+13.52

Table 2. PSNR differences between normal 3-D SPIHT and STTP-SPIHT (no channel code) for $n = 4, 16, 55, 110, 330$ of "Football" sequence with different BERs, and coded to 1.0 bit/pixel

bit error rates (BER) of $\epsilon = 0.01$ and 0.001, because the BER's of most wireless communication channels are $\epsilon = 0.01$ - 0.001. The corresponding rates and R_{eff} are calculated from Equation(1). In our case, we set the total transmission rate R_{total} to 2.53 Mbps, $r = 2/3$ for $\epsilon = 0.01$ and 8/9 for $\epsilon = 0.001$. For example, if we use a 352 x 240 x 16 frames, the size of the bit-stream is $R_{total} = 1,351,680$ bits (equivalently total transmission rate of 1.0 bpp with 352 x 240 x 16 frames), therefore we have effective number of packets $M_1 = R_{eff}/N = \frac{Nr}{(N+c+m)N} R_{total} = (2/3)/222 \times 1351680 \approx$ 4060 packets for $\epsilon = 0.01$, and $M_2 = R_{eff}/N = \frac{Nr}{(N+c+m)N} R_{total} = (8/9)/222 \times 1351680 \approx 5413$ packets for $\epsilon =$ 0.001.

We tested "Football" sequence of SIF (352×240) format. Table 3 shows the comparison of average PSNR between STTP-SPIHT and normal 3-D SPIHT at the same total transmission rate of 2.53 Mbps. We can see that the average PSNR of the STTP-SPIHT in the case of $\epsilon = 0.01$ is about 5 dB higher than that of the normal 3-D SPIHT, and in the case of $\epsilon = 0.001$ the average PSNR of the STTP-SPIHT is about 3.5 dB higher than that of the normal 3-D SPIHT. When we compare with the normal 3-D SPIHT with ARQ, the average PSNR is almost same. However, the ARQ strategy is often inapplicable to real time scenarios.

BER	STTP-SPIHT / FEC	3-D SPIHT / FEC	3-D SPIHT / FEC+ARQ
0.01	29.75 dB	24.5 dB	32.1 dB
0.001	31.75 dB	28.2 dB	32.8 dB

Table 3. Comparison of average PSNR between STTP-SPIHT and normal 3-D SPIHT at total transmission rate of 2.53 Mbps

Since the STTP-SPIHT bit stream is embedded, the decoder can request for more information (additional STTP-SPIHT/RCPC bit-stream) to improve the video quality from the transmitter whenever more channel bandwidth is available.

Figure 12 shows the 352 x 240 original "Football" sequence (frame15) (left). Typical reconstructions of "Football" sequence at total transmission rate of 2.53 Mbps and channel bit error rates of 0.01 with $n = 4$ and $n = 16$ are shown in Figure 12 (middle and right). As we can see, in Figure 12 (middle), the normal 3-D SPIHT stops decoding when decoding failure occurs, but in Figure 12 (right), the STTP-SPIHT stops decoding for the substream which decoding failure occurred. Therefore any early decoding failure affects the full extent of the GOF in the normal 3-D SPIHT. Note in Figure 12 (right), the early decoding failure allows reconstruction of a small region at the bottom, right of center, with lower resolution only, as only bits belonging to a low resolution were received correctly before cessation of decoding. Full resolution regions, where all the bits were correctly decoded, surround this reduced-resolution region.

5. CONCLUSIONS

We have shown here how robustness and resilience to transmission errors can be achieved in an embedded video compression algorithm with little increase in its complexity and little loss in noiseless channel performance. In addition to that, the STTP-SPIHT algorithm doesn't lose any properties of the normal 3-D SPIHT algorithm. The STTP-SPIHT algorithm presented here is much more robust against bit errors than that of the normal 3-D SPIHT algorithm. The resilience of the coded sequence to transmission errors increases with the number of s-t blocks. In

Figure 12. 352 x 240 original "Football" sequence (frame15) (left) 352 x 240 "Football" reconstruction using normal 3-D SPIHT/FEC with BER = 0.01. PSNR = 24.41dB (middle), 352 x 240 "Football" reconstruction (frame15) using STTP SPIHT(n=16)/RCPC with BER = 0.01. PSNR = 29.35dB (right). Total transmission rate is set to 2.53 Mbps.

the experiments sixteen s-t blocks provided large gains in error resilence with only minor degradation in error-free performance due to increased header information overhead.

REFERENCES

1. J. Hagenauer, "Rate-compatible punctured convolutional codes (rcpc codes) and their applications," *IEEE Transactions on Communications* **36**, pp. 389–400, April 1988.
2. J. M. Shapiro, "Embedded image coding using zerotrees of wavelet coefficient," *IEEE Transactions on Signal Processing* **41**, pp. 3445–3462, December 1993.
3. A. Said and W. A. Pearlman, "A new, fast and efficient image codec based on set partitioning in hierarchical trees," *IEEE Trans. on Circuits and Systems for Video Technology* **6**, pp. 243–250, June 1996.
4. Y. Chen and W. A. Pearlman, "Three-dimensional subband coding of video using the zero-tree method," *SPIE Visual Communications and Image Processing* , pp. 1302–1309, March 1996.
5. B.-J. Kim and W. A. Pearlman, "An embedded wavelet video coder using three-dimensional set partitioning in hierarchical trees," *Proc. of Data Compression Conference* , pp. 251–260, 1997.
6. P. G. Sherwood and K. Zeger, "Progressive image coding for noisy channels," *IEEE Signal Processing Letters* **4**, pp. 189–191, July 1997.
7. P. G. Sherwood and K. Zeger, "Progressive image coding on noisy channels," *Proc. DCC* , pp. 72–81, April 1997.
8. H. Man, F. Kossentini, and J. T. Smith, "Robust image coding for noisy channels," *IEEE Signal Processing Letters* **4**(8), August 1997.
9. C. D. Creusere, "A new method of robust image compression based on the embedded zerotree wavelet algorithm," *IEEE Transactions on Image Processing* **6**(10), pp. 1436–1442, October 1997.
10. B.-J. Kim, Z. Xiong, and W. A. Pearlman, "Progressive video coding for noisy channels," *In Proceedings ICIP 98* , 1998.
11. C. Creusere, "Robust image coding using the embedded zerotree wavelet algorithm," *Proc. Data Compression Conference* , p. 432, (Snowbird, UT), March 1996.
12. C. Creusere, "A family of image compression algorithms which are robust to transmission errors," *Proc. SPIE* **2668**, pp. 82–92, January 1996.
13. T. V. Ramabadran and S. S. Gaitonde, "A tutorial on crc computations," *IEEE Micro* **8**, pp. 62–75, August 1998.
14. G. Castagnoli, J. Ganz, and P. Graber, "Optimum cyclic redundancy-check codes with 16-bit," *IEEE Transactions on Communications* **38**, pp. 111–114, January 1990.
15. J. G.D. Forney, "The viterbi algorithm," *Proc. IEEE* **61**, pp. 169–176, Fabruary 1994.
16. N. Seshadri and C. Sundberg, "List viterbi decoding algorithm with applications," *IEEE Transactions on Communications* **42**, pp. 111–114, 1994.
17. M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies, "Image coding using wavelet transform," *IEEE Trans. Image Processing* **1**, pp. 205–220, 1992.