A Full-Featured, Error Resilient, Scalable Wavelet Video Codec Based on the Set Partitioning in Hierarchical Trees (SPIHT) Algorithm

Sungdae Cho and William A. Pearlman
Center for Next Generation Video Research
Rensselaer Polytechnic Institute
110 Eighth Street
Troy, NY 12180-3590

Corresponding Author:
Prof. William A. Pearlman
Tel: 518-276-6082 Fax: 518-276-8715
E-mail: pearlman@rpi.edu

Abstract

Compressed video bitstreams require protection from channel errors in a wireless 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 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 eliminate the need for ARQ by making the 3-D SPIHT bitstream more robust and resistant to channel errors. We first break the wavelet transform into a number of spatio-temporal tree blocks which can be encoded and decoded independently by the 3-D SPIHT algorithm. This procedure brings the added benefit of parallelization of the compression and decompression algorithms, and enables implementation of region-based coding. Then we demonstrate the packetization of the bitstream 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. In extensive comparative tests, the reconstructed video is shown to be superior to that of MPEG-2, with the margin of superiority growing substantially as the channel becomes noisier. 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.

I. Introduction

Wavelet zerotree image coding techniques were developed by Shapiro (EZW) [2], and further developed by Said and Pearlman (SPIHT) [3], and have provided unprecedented high performance in image compression with low complexity. 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 has shown promise of an effective and computationally simple video coding system without any motion

This material is based upon work supported by the National Science Foundation under Grant No. EEC-9812706, and IBM T. J. Watson Research Center under an IBM Faculty Award. The government has certain rights in this material.

compensation, obtaining excellent numerical and visual results. Later, Kim and Pearlman developed the three dimensional SPIHT (3-D SPIHT) [5] coding algorithm improving on the 3-D EZW system of [4].

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. Numerous sophisticated techniques have been developed over the last several decades to make image transmission over a noisy channel resilient to errors. One approach is to cascade a SPIHT coder with error control coding [7], [8]. The idea is to partition the output bitstream 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 or channel errors that occur early in the transmission, either of which can cause a total collapse of the decoding process.

Another approach to protecting image bitstreams from bit errors is to restructure the node test (NT) of the EZW algorithm. The 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* [9], [10].

Still another 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 [11] for use with the EZW algorithm.

In recent work [6], Alatan *et al.* showed 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 [1], and improve the overall performance against channel bit errors.

To achieve robust video over noisy channels, Kim *et al.* [14], [15] utilized the same RCPC code as Sherwood and Zeger [7], [8] with 3-D SPIHT, and found that a single automatic repeat request (ARQ) was also necessary to assure reliable reception of the bitstream. 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 [11], [12], [13] 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 bitstream. Therefore a bit error in the bitstream belonging to any one block does not affect any other block. We then apply Kim *et al.*'s method [14], [15] of forward error correction, borrowed from Sherwood and Zeger [7], [8], 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 bitstream 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 bitstreams. This less sensitive source coder substantially increases channel error robustness over a wide range of Bit Error Rates (BERs). In addition, we demonstrate that the coder has the functionality of region-based compression/decompression and spatial and/or temporal scalability while retaining error resilience.

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 contains region-based compression of video. Section 5 provides computer simulation results with comparisons to MPEG-2. Section 6 concludes this paper.

II. Error Resilient 3-D SPIHT Video Compression

Variable Length Codes (VLC) are used in current video codecs for higher coding performance. Transmission bit errors for such codes usually result in propagation of errors 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 [11], [12], [13], for partitioning a three-dimensional wavelet transform into independent coding units, so that an error in any one unit does not affect the others. We call this scheme Spatial and Temporal Tree Preserving 3-D SPIHT (STTP-SPIHT).

A. System Overview

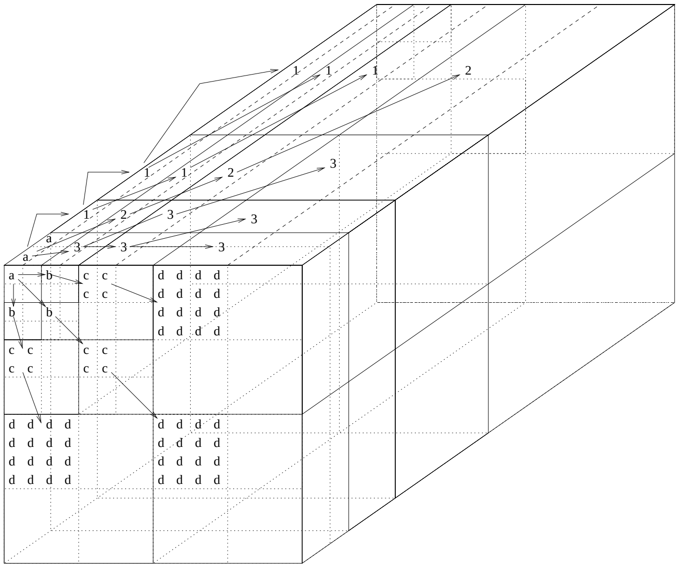


Fig. 1. Structure of the spatio-temporal relation of 3-D SPIHT compression algorithm

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 ($2 \times 2 \times 2$), 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 spatio-temporal tree linking successive generations of descendants. We used 16 frames in a GOF (group of frames), therefore we have 16 different frames of wavelet coefficients. We can observe that these frames have not only spatial similarity inside each one of them across the different scales, but also temporal similarity between frames, which will be efficiently exploited by the Spatial and Temporal Tree Preserving 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 P 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 P 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], and the normal 3-D SPIHT algorithm is just a case of $P = 1$.

Each substream has its own SPIHT header information. These headers are most important for the receiver to decode the substreams correctly, and should be carefully protected from channel errors. Furthermore, as we increase the number of groups P, the header information overhead is also increased,

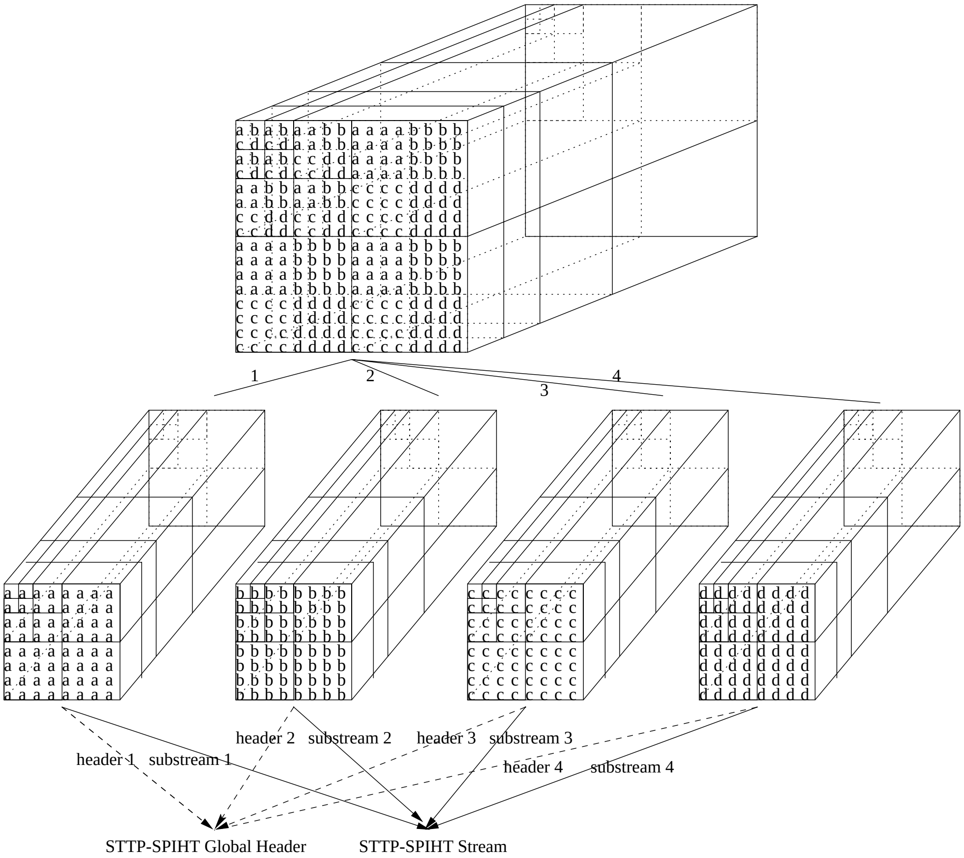
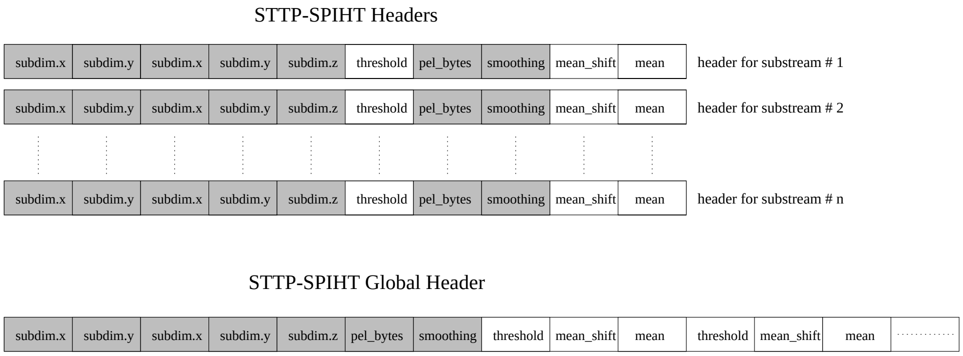


Fig. 2. Structure of the spatio-temporal tree preserving 3-D SPIHT(STTP-SPIHT) compression algorithm

because the size of header information is fixed. In the case of lower bit rates, the increasingly significant size of overhead information compared to data information will be detrimental to video quality.

To minimize these deleterious side effects, we use a global header as shown in Figure 3. As we can see in this figure, the values in the shaded areas (*subdim.x*, *subdim.y*, *subdim.z*, *pel_bytes*, *smoothing*) are the same for all the STTP-SPIHT headers, because each substream corresponds to a region with the same size and bit rate. Therefore we can put the common variables to the beginning as a global header, and continue to write the information that is different in other substreams, namely, *threshold*, *mean_shift*, and *mean*. The final shape of the global header is shown in bottom of Figure 3. Using this idea, we can protect the SPIHT header information more effectively, and in addition to that, we can use more bits to encode video data in the same bit rate.



STTP-SPIHT Headers

subdim.x	subdim.y	subdim.x	subdim.y	subdim.z	threshold	pel_bytes	smoothing	mean_shift	mean	header for substream # 1
subdim.x	subdim.y	subdim.x	subdim.y	subdim.z	threshold	pel_bytes	smoothing	mean_shift	mean	header for substream # 2
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	
subdim.x	subdim.y	subdim.x	subdim.y	subdim.z	threshold	pel_bytes	smoothing	mean_shift	mean	header for substream # n

STTP-SPIHT Global Header

subdim.x	subdim.y	subdim.x	subdim.y	subdim.z	pel_bytes	smoothing	threshold	mean_shift	mean	threshold	mean_shift	mean	······

Fig. 3. Structure of global header for the STTP-SPIHT compression algorithm

The P sub-bitstreams are then interleaved in appropriate size units (e.g. bits, bytes, packets,

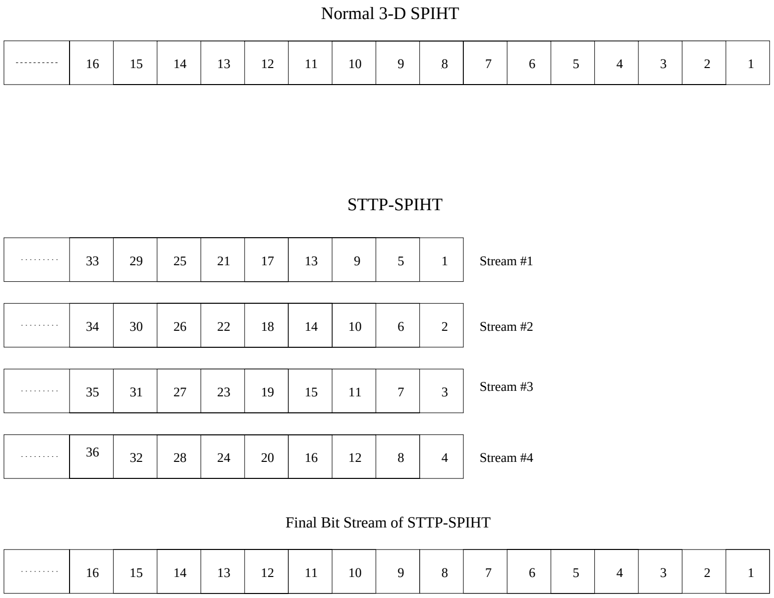


Fig. 4. Comparison of bitstreams between normal 3-D SPIHT and STTP-SPIHT with $P = 4$

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 4 illustrates the assembly of the final bitstreams of 3-D SPIHT and STTP-SPIHT with $P = 4$. The bitstreams are segmented into packets numbered by order of transmission. Encoding proceeds horizontally along the bitstreams, but in STTP-SPIHT, the transmission occurs vertically downward and then from right to left along the bitstreams to accomplish interleaving. Therefore, the final STTP-SPIHT bitstream will be embedded or progressive 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 processed sequentially at the destination. For the STTP-SPIHT, however, the interleaved bitstream will be de-interleaved, and each substream will be processed independently.

By coding the wavelet coefficients with multiple and independent bitstreams, any single bit error affects only one of the P streams, while the others are received unaffected. Therefore the wavelet coefficients represented by a corrupted bitstream are reconstructed at reduced accuracy, while those represented by the error-free streams are reconstructed at the full encoder accuracy.

B. Color STTP-SPIHT

We have considered only gray scale or one color plane, luminance coding. In this section, we will extend STTP-SPIHT to color video coding while still preserving all the properties of the normal 3-D SPIHT. We follow the Kim *et al.*'s [16] color extension method, and apply to each s-t block separately. We treat all color planes of an s-t block as one unit at the coding stage, and generate mixed YUV sub-bitstreams so that we can stop at any point of the bitstream and reconstruct the color video of the best quality at the given bit rate.

In our case, the video sequence is YUV 4:2:2, where the U and V chrominance planes are half the size of the luminance Y plane as shown in Figure 5. The color STTP-SPIHT algorithm is essentially the same as the gray scale STTP-SPIHT except that now we have two more chrominance planes. Figure 5 shows the idea of error resilient color STTP-SPIHT video compression algorithm for YUV 4:2:2, which is to divide the 3-D wavelet coefficients of the three planes into some number P different groups according to their spatial and temporal relationships. We then encode each group independently with

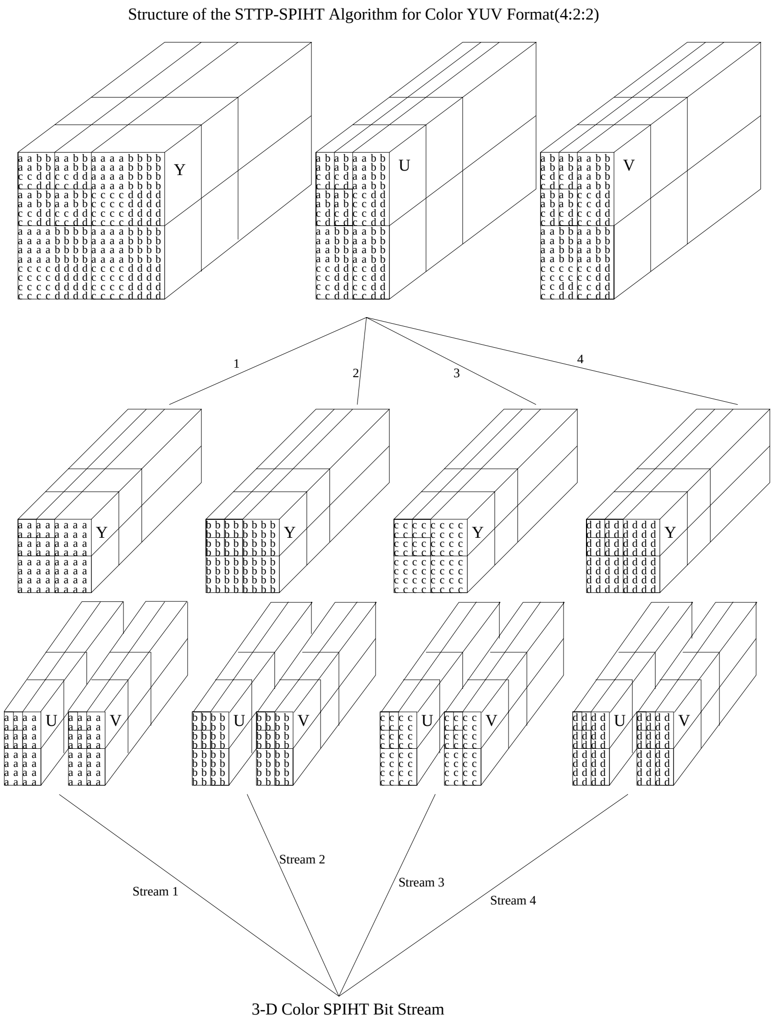

Structure of the STTP-SPIHT Algorithm for Color YUV Format(4:2:2)
Y
U
V
1
2
3
4
Y
Y
Y
Y
U
V
U
V
U
V
U
V
Stream 1
Stream 2
Stream 3
Stream 4
3-D Color SPIHT Bit Stream

Fig. 5. Structure of the STTP-SPIHT Algorithm for Color YUV Format (4:2:2)

the same rate using the 3-D SPIHT algorithm.

C. Scalability of STTP-SPIHT

STTP-SPIHT is scalable in rate, using block interleaving/de-interleaving of the sub-bitstreams. In addition to that, it is highly desirable for STTP-SPIHT to have temporal and/or spatial scalability for today's multimedia applications, such as picture in picture function, video or volumetric image database browsing, and distance learning.

The STTP-SPIHT coder is based on a multiresolution wavelet decomposition, so it should be simple to add the function of multiresolution decoding using the STTP-SPIHT sub-bitstreams. We can just partition the embedded STTP-SPIHT sub-bitstreams into portions according to their subbands, and only decode subbands corresponding to the desired resolution.

Figure 6 shows partitionings of the STTP-SPIHT sub-bitstreams ($P = 4$) according to their corresponding spatial/temporal locations. In this figure, dark areas represent the low resolution of video sequence, and the other part is used for high resolution. As we can see, lower resolution information is usually located at the beginning part of the sub-bitstreams. After coding some point of the video sequence, most of the remaining bit budget is used for coding the higher frequency bands which contain the detail of video not usually visible at reduced spatial/temporal resolution. Figure 7 illustrates this idea for a two layer case of spatially scaled 352×240 “Football”sequence (frame 5) . A low resolution video can be decoded from the first layer only, and the image size is 176×120. If we use three layers,

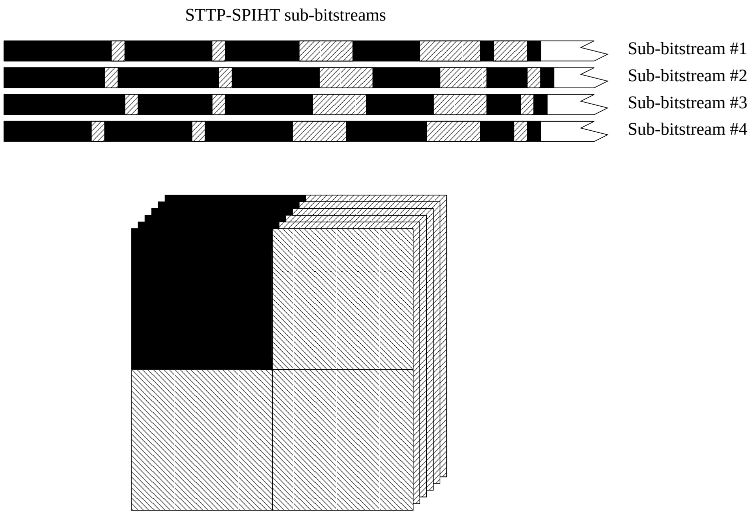


Fig. 6. Partitioning of the STTP-SPIHT sub-bitstreams into portions according to their corresponding spatial locations

the lowest layer's size would be 88 × 60. Figure 8 is a typical example of multiresolution decoding of the "Football" and "Susie" sequences. In this figure, (a) and (b) are spatial half resolution of the sequence, and (c) and (d) are full resolution of the sequence.

The most valuable benefit of resolution scalable decoding is saving of decoding time, because the wavelet transformation consumes most of the decoding time of the process. For example, in a low resolution video of two spatial scales only, one-quarter of the number of wavelet coefficients is transformed, while the spatio-temporal resolution of two spatial and temporal scales involves transformation of one-eighth of the number of wavelet coefficients in the decoder. Therefore the lower resolutions of the sequences need much less time to transform due to the considerably fewer number of wavelet coefficients.

III. 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 [14], [15]. Figure 9 illustrates the overall system with an optional ideal return channel for ARQ. In our study, we shall make no use of ARQ. Kim *et al.* [14], [15] cascaded the 3-D SPIHT coder with RCPC using a single request ARQ strategy.

Figure 10 shows the system description of STTP-SPIHT/RCPC coder. In this figure, the RCPC coded stream is a segment of the channel encoded bitstream. We first partition the STTP-SPIHT bitstream 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) [17], [18] 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 detection 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, each segment of N bits of the STTP-SPIHT bitstream is transformed into a segment of $N + c + m$ bits and passed through the rate r RCPC channel encoder, which is a type of punctured convolutional coder with the added feature of rate compatibility.

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

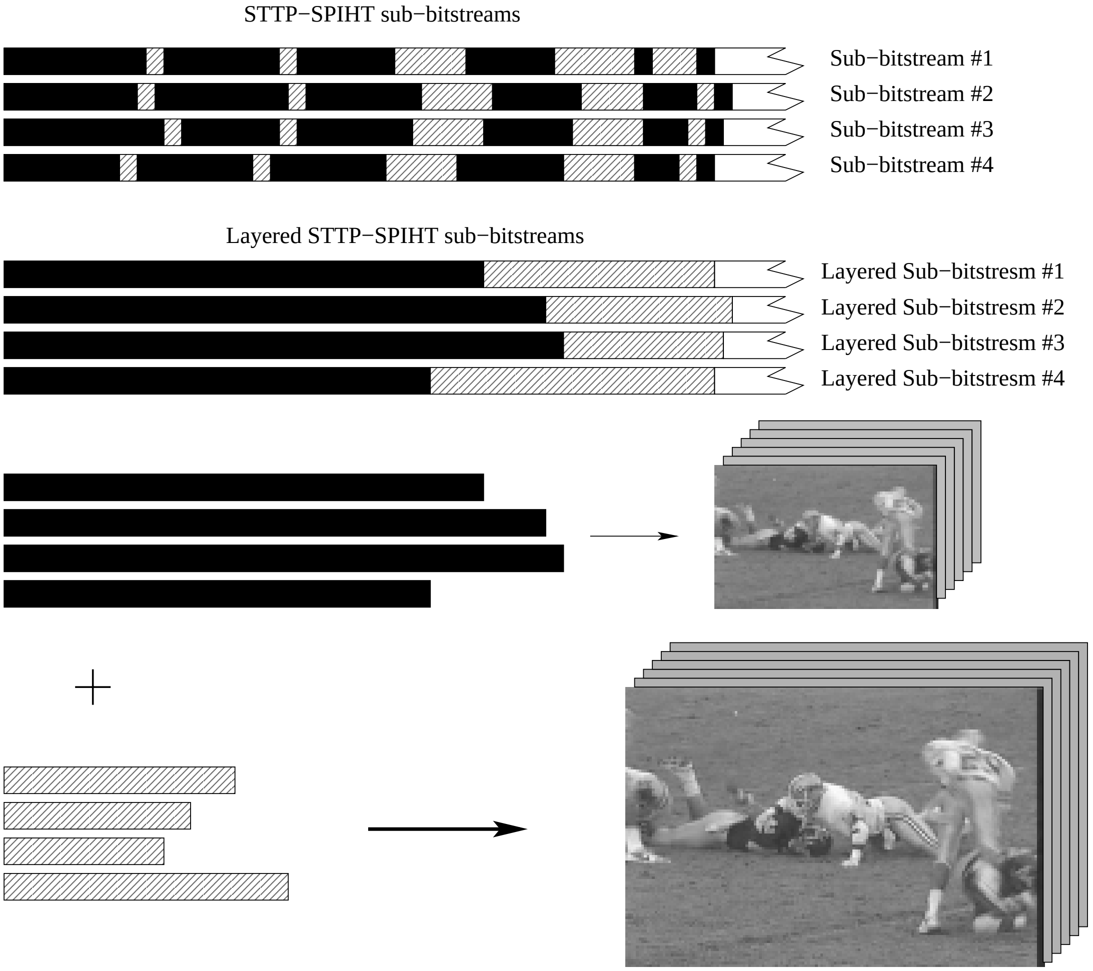


Fig. 7. Multiresolutional decoder uses the higher resolution layer to increase the spatial/temporal resolution of the video.

channel (BSC).

Since the encoder adds redundant bits into 3-D SPIHT bitstream 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 bitstream in bits.

As we saw before, Figure 4 graphically illustrates the final bitstream comparison between the normal 3-D SPIHT and the STTP-SPIHT. For STTP-SPIHT, we interleave by blocks of 200 bits according to Figure 4 to maintain embeddedness.

At the destination, the Viterbi decoding algorithm [19], [20] is used to convert the packets of the received bitstream into a STTP-SPIHT 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 (here, that depth is 100) 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 P sub-bitstreams.

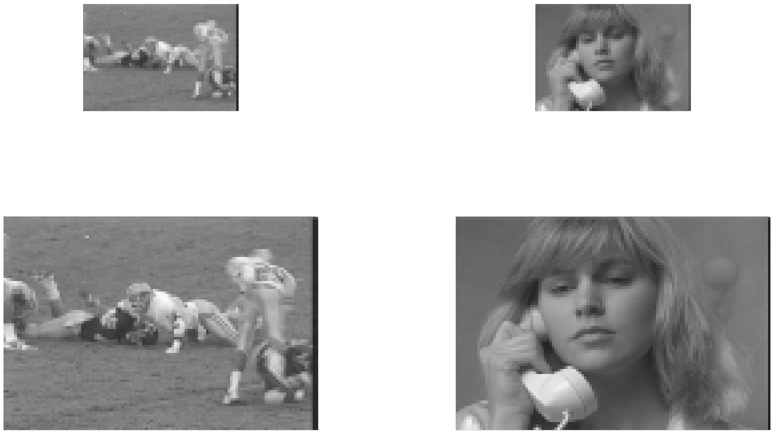


Fig. 8. Multiresolutional decoded sequence with STTP-SPIHT video coder (a) Top-left : spatial half resolution of "Football" sequence (frame 5) (b) Top-right : spatial half resolution of "Susie" sequence (frame 21) (c) Bottom Left : full resolution of "Football" sequence (frame 5) (d) Bottom right : full resolution of "Susie" sequence (frame 21).

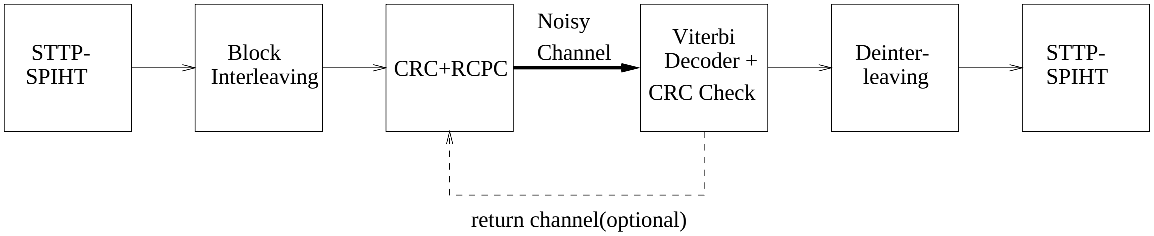


Fig. 9. STTP-SPIHT/RCPC system framework

Figure 11 shows an example of decoding with $P = 16$ 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 that the wavelet coefficients 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

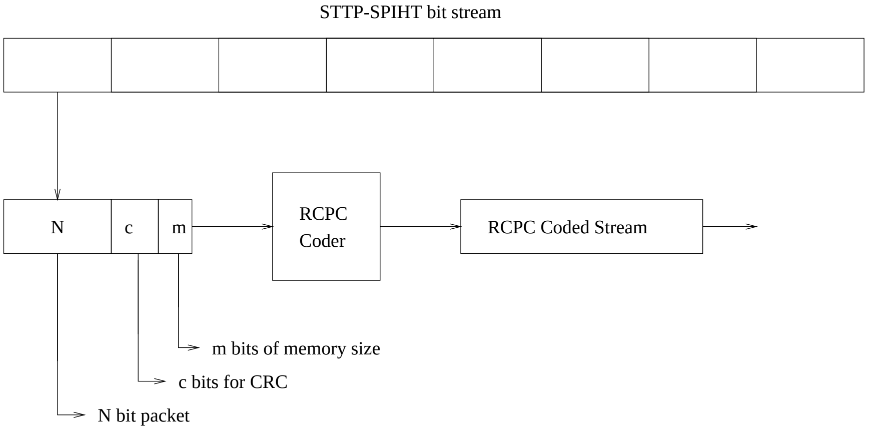


Fig. 10. System description of STTP-SPIHT/RCPC coder

STTP-SPIHT decodes many more clean bits compared to the normal 3-D SPIHT.

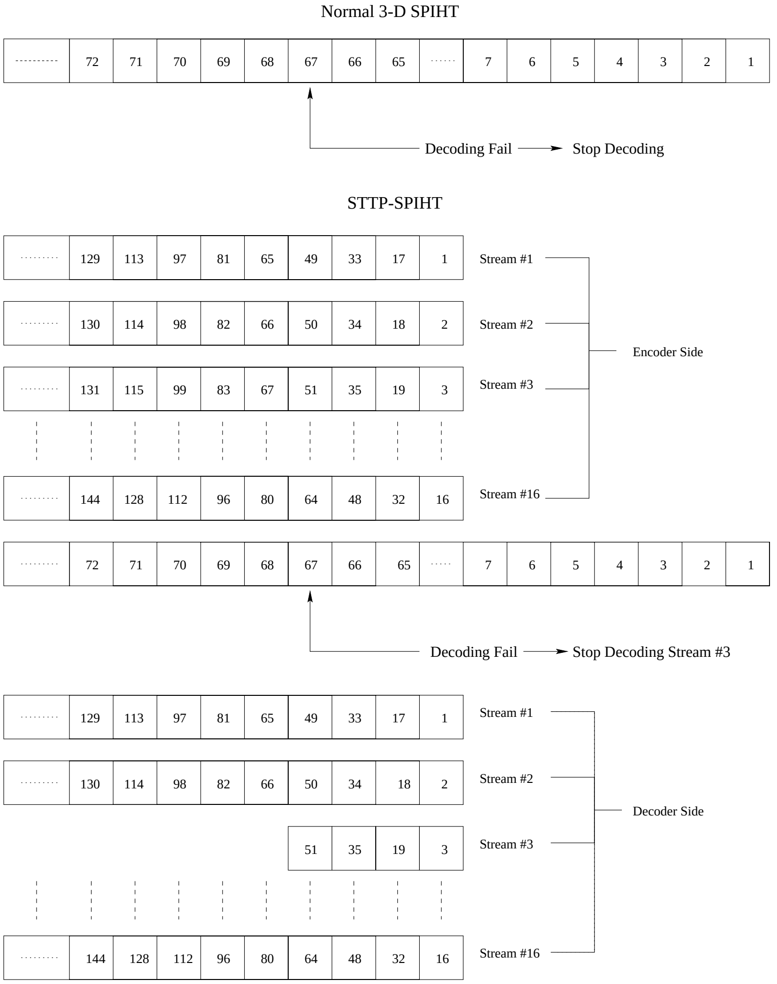


Fig. 11. Example of decoding when decoding failure occurs at packet number 67

IV. Region-Based Compression of Video with STTP-SPIHT

We can take advantage of the STTP-SPIHT coder to get a region-based compression of video sequences. Using the coder, a specific region of interest (ROI) gets reproduced with higher quality than the rest of the image frame.

Many classes of video sequences contain areas which are more important than others. It is unnecessary and unwise to treat equally all the pixels in video sequence. To minimize the total number of bits, unimportant areas should be highly compressed, thereby reducing transmission time and cost without losing the quality of the video sequence.

One could preserve the features with nearly no loss, while achieving high compression overall by allowing degradation in the unimportant regions termed regionally lossy coding or region-based lossy coding. Compression schemes, which are capable for delivering higher reconstruction quality for the significant portions, are attractive in some cases such as video transmission over channels with highly constrained bandwidth and volumetric medical image areas, where doctors are interested only in a specific portion which might contain a disease.

In the STTP-SPIHT, each substream corresponds to a certain region. Therefore, we can assign more bits to the substream, which has the information of the region of interest, and assign the remaining bits of bit budget to the other substreams. Therefore, the sequence belonging to the background and the ROI's are coded independently at the specified bitrates.

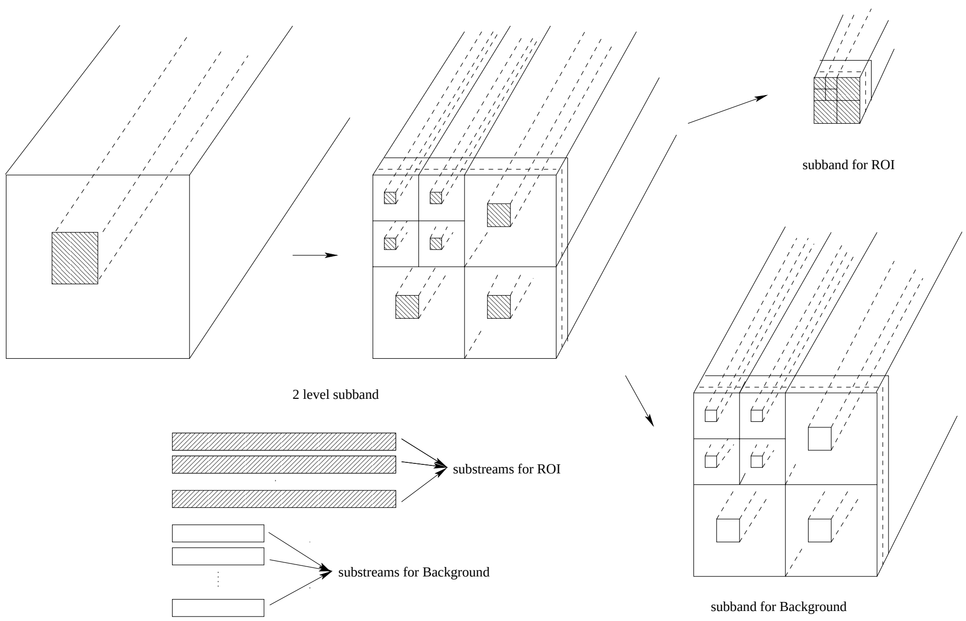


Fig. 12. 3D Encoder System Configuration

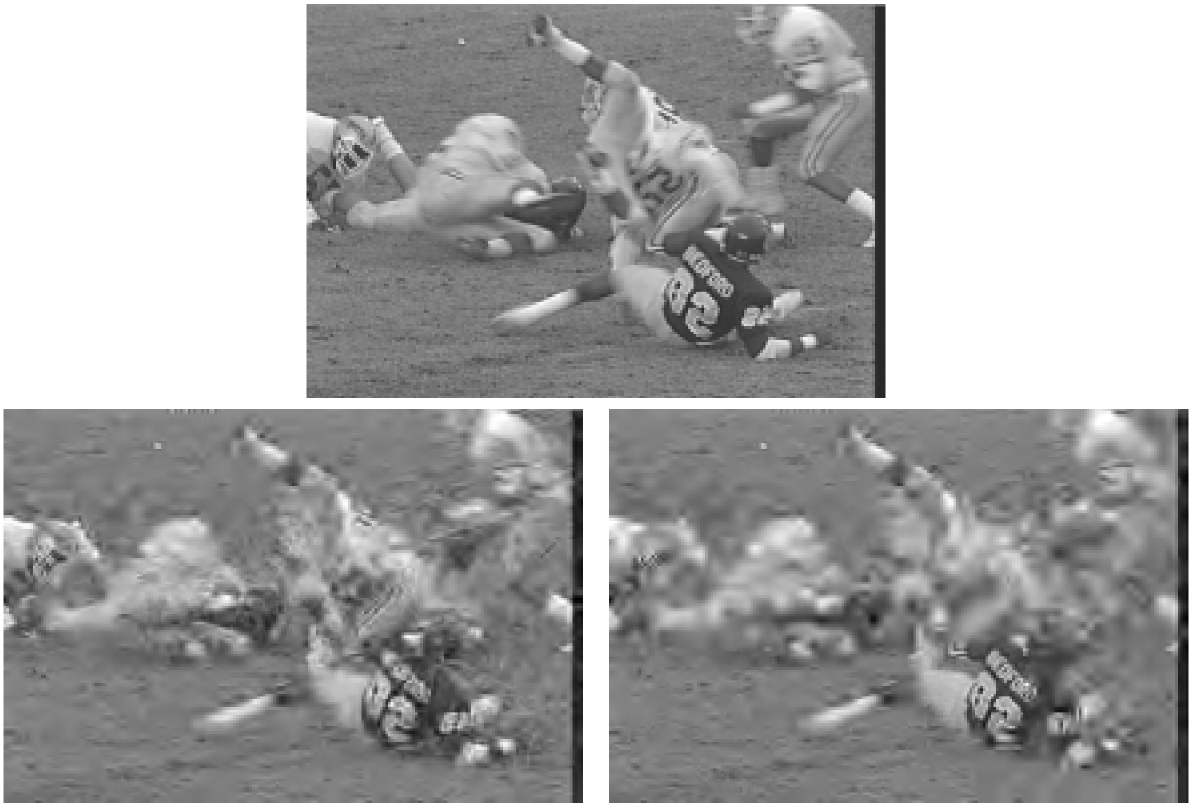


Fig. 13. 352 × 240 “Football” sequence (frame15) (a) Top : Original sequence, (b) Bottom-left : 3-D SPIHT compressed at 0.0845 bpp, (c) Bottom-right : region-based STTP-SPIHT result, requiring an overall rate of 0.0845 bpp

When we encode video sequences for region-based coding, we enter more information to the encoder, such as the axis of top-left and bottom-right positions for the region of interest of the image, desired bit rate for the region of interest, and total bit rate or background bit rate, so that we can decide which bitstreams would correspond to the region. There are two ways to decide the background bit rate. One is to specify the total bit rate and desired bit rate of the ROI, and the other is to specify the ROI bit rate and background's. In the first case, we can always meet the target bit rate because we can assign the remaining bit budget of total bit rate after assigning to the ROI, and the second case, we can handle the quality of background image while maintaining the quality of the ROI.

To the decoder side, there are two kinds of substreams, one for the ROI and the other for the background. The STTP-SPIHT decodes the sub-bitstreams independently. The only difference from the original STTP-SPIHT is that there are two kinds of bit rates specified by the encoder. After their independent decodings, the decoded wavelet coefficients are reordered according to their spatial and temporal relationships, and then the inverse wavelet transform is applied.

In Figure 12, the ROI is shown as the shaded area. We can easily figure out which sub-block of coefficients corresponds to the ROI, because of the spatio-temporal relationships of each block. Then we assign more bits to the block which has the information of the ROI, and the remaining bits to the other blocks. As we can see in this figure, the final substreams for ROI are longer than the other substreams.

Figure 13 shows an example of region-based coded video sequence. In this figure, (a) is the 352×240 original Football sequence (frame 15) and (b) is the decoded image with an overall bit rate of 0.0845 bpp and (c) is region-base coded image with the same overall bit rate of 0.0845 bpp. In (b), the player's back number and name are hard to discern, but in (c), they are much clearer.

V. Results

In this section, we provide simulation results and compare the proposed STTP-SPIHT video codec with MPEG-2 in several aspects such as source coding in noisy and noiseless channels, and combined source and channel coding in noisy channels.

A. Robust Source Coding

In our test of error resilience, we assume that the channel is binary symmetric (BSC) with transition error probability ϵ. For MPEG-2, we use 15 frames in a GOP (Group Of Pictures), and the I/P frame distance is 3 (IBBPBBP...). For the STTP-SPIHT, sixteen frames in a GOF (Group Of Frames) are used, and a dyadic three level transform using 9/7 biorthogonal wavelet filters [21] is applied to the image, and possible robust partitionings are evaluated. We use one global header as shown in Figure 3 instead of using sub-headers for each substream, and we assume the SPIHT global header is not corrupted from bit errors. We interleaved the streams in 200 bit packets to maintain embeddedness. Figure 4 compares the encoded final bitstreams of the normal 3-D SPIHT and the STTP-SPIHT. The receiver de-interleaves 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" (frame number 0-47) and "Susie" (frame number 16-63) sequences. 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 image sequences. All PSNR's reported for noisy channels are averages over fifty (50) independent runs.

The goal of the tests is to demonstrate the inherent resilience of the STTP-SPIHT bitstream to channel errors. Therefore, there has been no attempt to error concealment through postprocessing either for STTP-SPIHT or MPEG-2 in these tests.

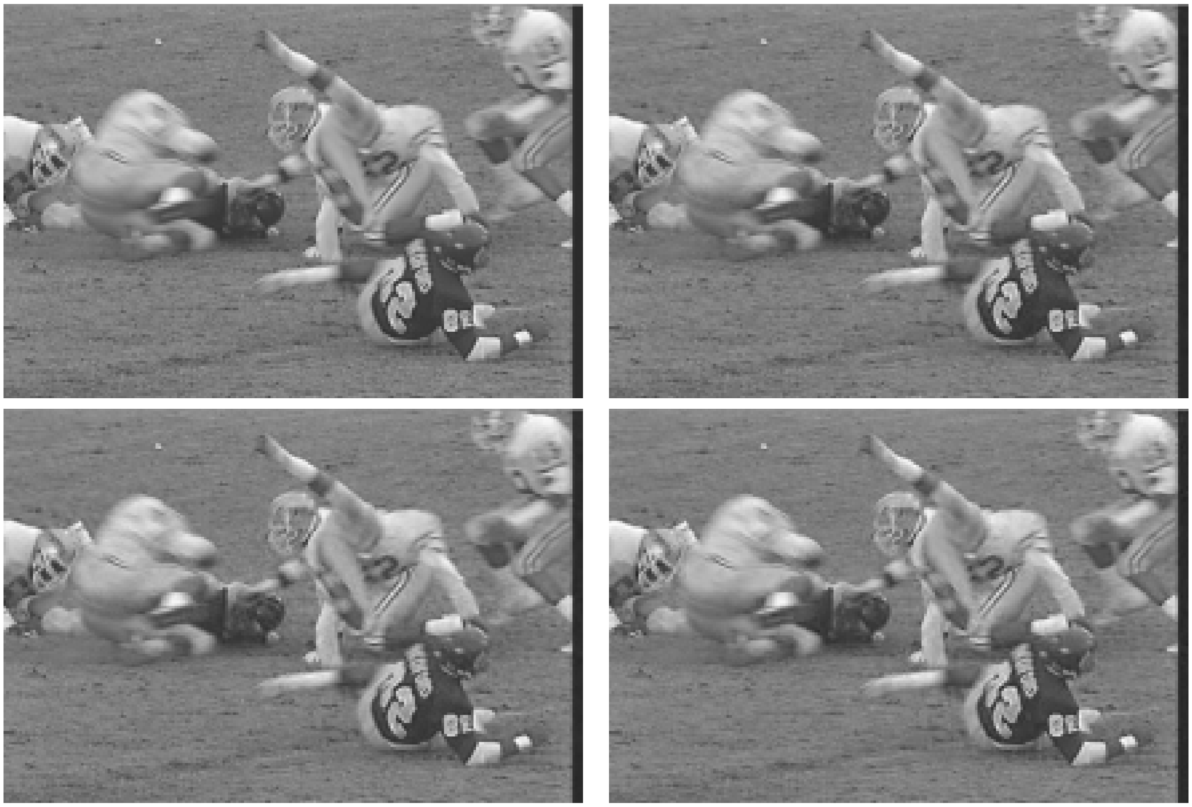


Fig. 14. 352 × 240 "Football" sequence (frame14) coded to 1.0 bit/pixel without any bit error. (a) Top-left : Normal 3-D SPIHT, PSNR = 34.19 dB, (b) Top-right : STTP-SPIHT (P = 4), PSNR = 33.96 dB, (c) Bottom-left : STTP-SPIHT (P = 16), PSNR = 33.50 dB, (d) Bottom-right : MPEG-2, PSNR = 32.91 dB

In our simulation of error resilient video transmission without error correction coding, we decoded the STTP-SPIHT and MPEG-2 bitstreams to the end of the received bitstreams regardless of channel bit errors, since there was no mechanism to announce channel errors. Figure 14 shows the football sequences without any bit error using normal 3-D SPIHT (a), STTP-SPIHT (P = 4) (b), STTP-SPIHT (P = 16) (c), MPEG-2 (d), where the PSNR's are 34.19 dB, 33.96 dB, 33.50 dB and 32.91 dB respectively. The successively lower PSNRs for the STTP-SPIHT are mainly due to the presence of successively more overhead bits needed to demarcate sub-bitstreams. Figure 15 shows the effect in the presence of bit errors (BER = 10^{-4}) without error correction. The first three images (a), (b), (c) represent the sequences used with the STTP-SPIHT algorithm with P = 16, P = 55 and P = 110, where the PSNR's are 17.93 dB, 21.66 dB and 25.80 dB respectively and the visual results noticeably improve as the number of blocks P increase, and image (d) shows the sequence with the MPEG-2 algorithm with corresponding PSNR is 16.88 dB and many blocks badly corrupted by bit errors.

In Figure 16, BER = 10^{-5} is used with STTP-SPIHT with P = 16 (a), P = 55 (b) and P = 110 (c) and MPEG-2 algorithm (d), and the corresponding PSNRs are 27.00 dB, 30.94 dB, 31.12 dB and 28.08 dB respectively. Here, at the lower error rate, even P = 16 blocks offers a decent reconstruction and P = 55 and 110 blocks achieves a very good reconstruction, in contrast to the MPEG-2 decoded sequence where some blocks are completely obliterated.

Figure 17 represents the frame by frame comparison of PSNR's of "Football" sequence with different BERs and coded with 1.0 bit/pixel. The solid line on top shows the PSNR values of STTP-SPIHT (P = 16) without any bit errors, and the second and the third solid lines mean PSNR values with channel bit errors, and the corresponding BERs are 10^{-5} and 10^{-4} respectively. The dotted lines represent STTP-SPIHT with P = 55, and the dashed line indicates the MPEG-2 coded sequence. As we can see, the PSNRs for STTP-SPIHT without bit errors are similar to those of the MEPG-2, but in the

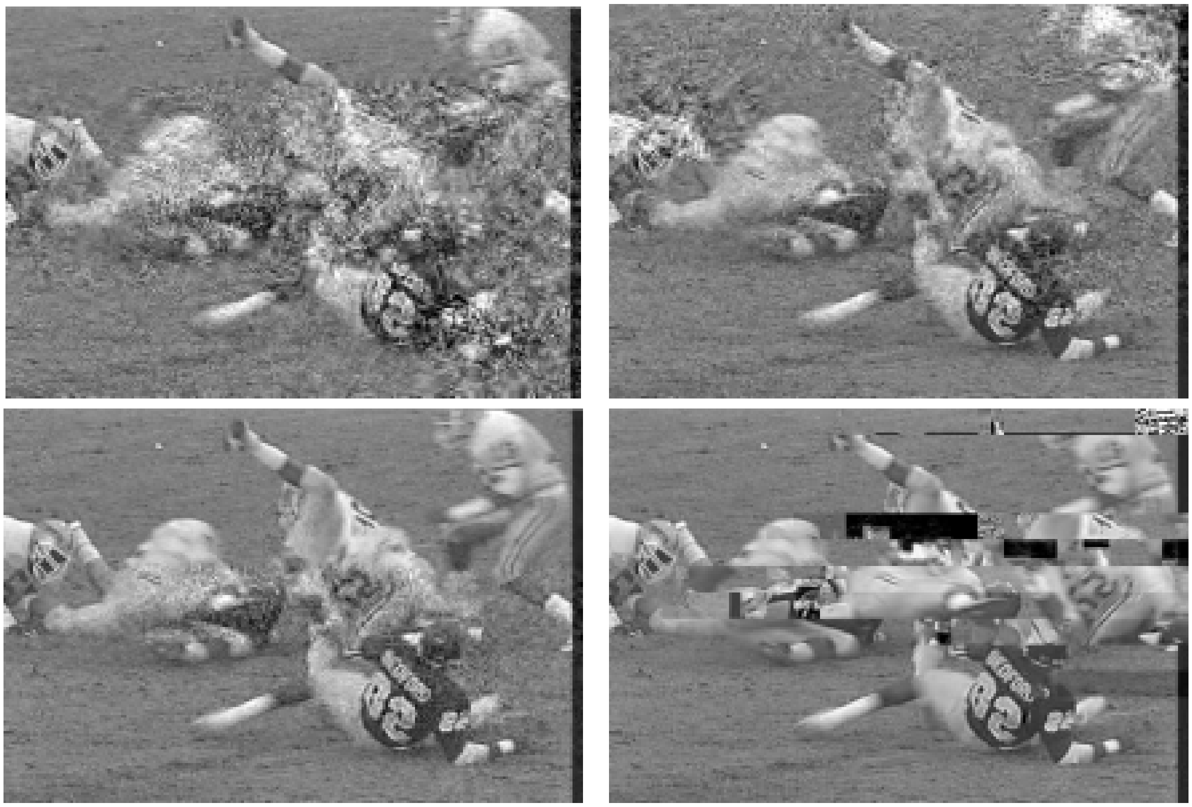


Fig. 15. 352 × 240 “Football” sequence (frame15) coded to 1.0 bit/pixel with BER = 0.0001. (a) Top-left : STTP-SPIHT ($P = 16$), PSNR = 17.93 dB, (b) Top-right : STTP-SPIHT ($P = 55$), PSNR = 21.66 dB, (c) Bottom-left : STTP-SPIHT ($P = 110$), PSNR = 25.80 dB, (d) Bottom-right : MPEG-2, PSNR = 16.88 dB

case of channel bit errors, the PSNR values of STTP-SPIHT ($P = 55$) are much higher than those of the MPEG-2.

Figure 18 illustrates the comparison of resulting average PSNR of “Football” sequence with wide range of BERs (0 - 10^{-3}) and different number of blocks P (1, 4, 10, 16, 55, 110, 330), and coded with 1.0 bit/pixel. In this figure, when BER is very high (10^{-3}), the average PSNR value of the STTP-SPIHT with $P = 330$ is still 2.41 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 0.77 dB, 0.27 dB, 0.08 dB, 0.76 dB, 1.00 dB, 1.34 dB with number of blocks $P = 4, 10, 16, 55, 110, 330$, respectively. However, if the BER is larger than 10^{-6}, the PSNR differences are much larger, ranging from 0.98 dB to 11.95 dB depending on the number of blocks P, and the BERs.

Table I shows the actual PSNR values for MPEG-2 and the differences among MPEG-2, normal 3-D SPIHT and STTP-SPIHT (P = 4, 10, 16, 55, 110, 330) of Figure 18. Table II also shows the average PSNR values and the PSNR differences of the “Susie” sequence. As we can see, the performance of the normal 3-D SPIHT is better than that of STTP-SPIHT in error free condition, then degrades rapidly as the BER becomes higher. In the error free condition, the performance of STTP-SPIHT gets worse as P increases due to the header overhead in each sub stream. Nevertheless, as the BER becomes higher, STTP-SPIHT with large P outperforms both normal 3-D SPIHT and MPEG-2 by significant margins.

B. Combined Source and Channel coding

In our simulation of error resilient video transmission with error correction capability, both the STTP-SPIHT and MPEG-2 bitstreams were protected identically. As in previous works [7], [8], [14], [15], we protected the 200 bit packets with the CRC, $c = 16$ bit parity check using generator polynomial

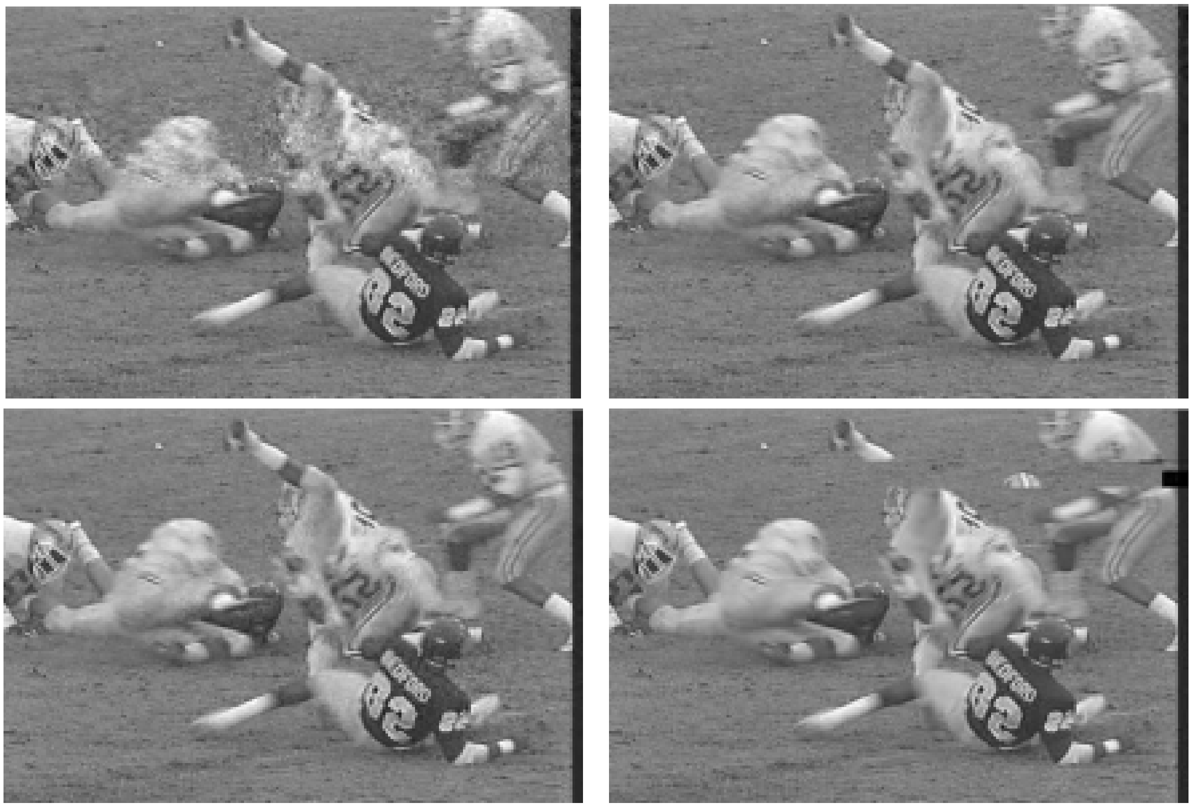


Fig. 16. 352 × 240 “Football” sequence (frame14) coded to 1.0 bit/pixel with BER = 0.00001. (a) Top-left : STTP-SPIHT ($P = 16$) , PSNR = 27.00 dB, (b) Top-right : STTP-SPIHT ($P = 55$), PSNR = 30.94 dB, (c) Bottom-left : STTP-SPIHT ($P = 110$), PSNR = 31.12 dB, (d) Bottom-right : MPEG-2, PSNR = 28.08 dB

BER	0	10^{-6}	10^{-5}	10^{-4}	10^{-3}
MPEG-2 $*$	33.27	32.07	27.14	17.47	9.17
Normal 3-D SPIHT	+0.93	-4.02	-8.43	-6.26	-1.94
STTP-SPIHT ($P = 4$)	+0.77	-1.99	-6.77	-5.06	-0.99
STTP-SPIHT ($P = 10$)	+0.27	+0.22	-3.65	-1.67	+0.98
STTP-SPIHT ($P = 16$)	+0.08	-0.29	-0.20	+1.13	+1.78
STTP-SPIHT ($P = 55$)	-0.76	-0.29	+1.31	+2.62	+6.02
STTP-SPIHT ($P = 110$)	-1.00	-0.15	+0.88	+4.33	+7.61
STTP-SPIHT ($P = 330$)	-1.34	-0.14	+2.24	+8.79	+11.95

$*$ omits frames of failed decoding

TABLE I

Comparison of average PSNR(dB) of “Football” sequence with different BERs and number of blocks P (1, 4, 10, 16, 55, 110, 330) with STTP-SPIHT and MPEG-2 at total transmission rates of 2.53 Mbps. (no channel code, and coded to 1.0 bit/pixel)

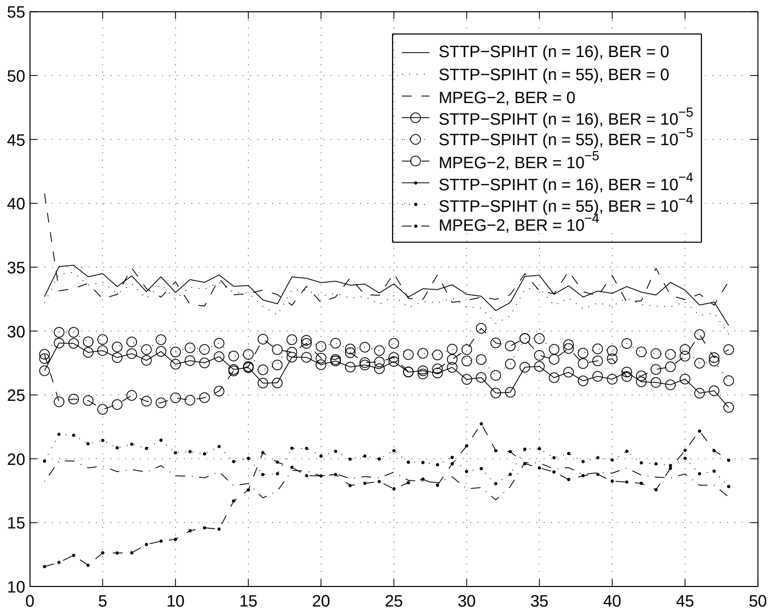


Fig. 17. Comparison of frame by frame PSNR(dB) of "Football" sequence with different BERs (0, 10^{-5} and 10^{-4}) and coded to 1.0 bit/pixel with STTP-SPIHT and MPEG-2 without channel code.

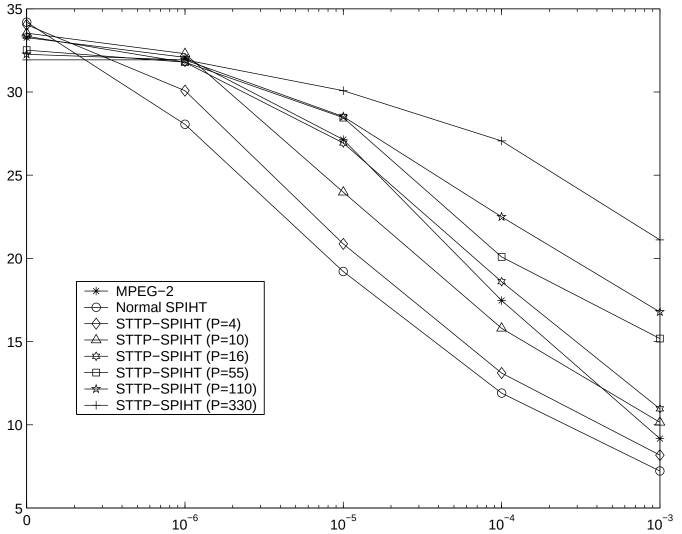


Fig. 18. Comparison of average PSNR(dB) of "Football" sequence with different BERs and number of blocks P (1, 4, 16, 55, 110, 330), and coded to 1.0 bit/pixel with STTP-SPIHT without channel code.

$g(x) = X^{16}+X^{14}+X^{12}+X^{11}+X^{8}+X^{5}+X^{4}+X^{2}+1$, and RCPC channel coder with constraint length $m = 6$. We focused on 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 $352 \times 240 \times 16$ frames, the size of the bitstream is $R_{total} = 1,351,680$ bits (equivalently total transmission rate of 1.0 bpp with $352 \times 240 \times 16$ 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" and "Susie" sequences of SIF (352×240) format. For the STTP-SPIHT, we stop decoding for the substream where decoding failure occurs. In our test, the path search depth was

BER	0	10^{-6}	10^{-5}	10^{-4}	10^{-3}
MPEG-2 $*$	42.90	41.16	31.32	18.78	9.27
Normal 3-D SPIHT	+1.76	-3.29	-3.85	-4.64	-1.78
STTP-SPIHT ($P = 4$)	+1.47	-1.90	-1.72	-1.60	-0.19
STTP-SPIHT ($P = 10$)	+1.29	+1.78	+1.77	+2.40	+2.31
STTP-SPIHT ($P = 16$)	+0.23	+0.11	+3.04	+3.79	+3.97
STTP-SPIHT ($P = 55$)	-0.69	+0.36	+4.18	+7.03	+10.85
STTP-SPIHT ($P = 110$)	-1.18	-0.68	+7.13	+10.56	+13.04
STTP-SPIHT ($P = 330$)	-1.92	-1.03	+7.98	+17.13	+19.29

$*$ omits frames of failed decoding

TABLE II

Comparison of average PSNR(dB) of "Susie" sequence with different BERs and number of blocks P (1, 4, 10, 16, 55, 110, 330) with STTP-SPIHT and MPEG-2 at total transmission rates of 2.53 Mbps. (no channel code, and coded to 1.0 bit/pixel).

set to 100, and if none of these paths is satisfied by the CRC, then the decoder stops decoding for the substream. 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. For some trials with MPEG-2 in noisy conditions, decoding failed for several consecutive frames. For example in Figure 17, the bottom curve shows several frames at the beginning of the sequence with very low PSNR's. This phenomenon never occurred in any of the STTP-SPIHT runs. In the tables, we decided to omit failed decoding frames in the PSNR calculations for MPEG-2, and mark those PSNR's with asterisks.

Table III shows the comparison of average PSNRs with STTP-SPIHT, normal 3-D SPIHT and MPEG-2 at total transmission rates of 1 Mbps and 2.53 Mbps of "Football" and "Susie" sequences with bit error rates (BER) of 0, 0.01 and 0.001. In noiseless conditions, the PSNRs of STTP-SPIHT are similar to those of MPEG-2 or 0.1 - 0.8 dB higher. However, in the noisy channel, we can see that the average PSNRs of the STTP-SPIHT with "Football" sequence are about 2 - 3 dB higher at 2.53 Mbps and 0.3 - 0.9 dB higher at 1 Mbps than those of the MPEG-2, and in the case of "Susie" sequence the average PSNRs of the STTP-SPIHT are about 2 - 3 dB higher at 2.53 Mbps and 1 - 2 dB higher at 1 Mbps than those of the MPEG-2. When we compare with the normal 3-D SPIHT with ARQ, the average PSNRs of STTP-SPIHT are just 1 - 2 dB lower. However, the ARQ strategy is often inapplicable to real time scenarios.

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

Figure 19 shows the comparison of 352 × 240 "Football" and "Susie" sequence with FEC and BER = 0.01. Typical reconstructions of "Football" and "Susie" sequence at total transmission rate of 2.53 Mbps and channel bit error rates of 0.01 with $P = 16$ and MPEG-2 are shown in Figure 19 (c), (d) and (g), (h). As we can see, in Figure 19 (b) and (f), the normal 3-D SPIHT stops decoding when decoding failure occurs, but in Figure 19 (c) and (g), the STTP-SPIHT stops decoding for the substream in which decoding failure occurs. Therefore any early decoding failure affects the full extent of the GOF in the normal 3-D SPIHT. However, the MPEG-2 decoded sequence in Figure 19 (d) and (h), the decoding failure affects some block, and the block is filled with some other picture's block. Note in Figure 19 (c) and (g), the early decoding failure in STTP-SPIHT allows reconstruction of a small region at the bottom, right of center (c), and at the top, right of center (g), with lower resolution only,

	Sequence	football		Susie	
	BER(Bit Error Rate)	0		0	
1Mbps	STTP-SPIHT ($P = 10$)	28.47		39.53	
	STTP-SPIHT ($P = 16$)	28.32		38.63	
	MPEG-2	28.23		39.64	
	3-D SPIHT	29.11		39.92	
2.53 Mbps	STTP-SPIHT ($P = 10$)	33.54		44.19	
	STTP-SPIHT ($P = 16$)	33.35		43.13	
	MPEG-2	33.27		42.90	
	3-D SPIHT	34.20		44.66	
	BER(Bit Error Rate)	0.01	0.001	0.01	0.001
1Mbps	STTP-SPIHT ($P = 16$)/RCPC	26.04	26.61	35.84	37.27
	MPEG-2/RCPC $*$	25.13	26.31	33.28	36.66
	3-D SPIHT/RCPC	24.21	26.55	34.02	37.10
	3-D SPIHT/RCPC+ARQ	26.63	27.72	36.74	37.91
2.53 Mbps	STTP-SPIHT ($P = 16$)/RCPC	29.61	30.77	39.67	40.27
	MPEG-2/RCPC $*$	27.36	28.98	36.66	38.87
	3-D SPIHT/RCPC	24.50	28.20	34.46	37.64
	3-D SPIHT/RCPC+ARQ	32.10	32.80	41.71	43.23

$*$ omits frames of failed decoding

TABLE III

Comparison of average PSNRs (dB) of "Football" and "Susie" sequences with STTP-SPIHT, normal 3-D SPIHT and MPEG-2 at total transmission rates of 1 Mbps and 2.53 Mbps with bit error rates (BER) of 0, 0.01, 0.001

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.

Figure 20 compares 352×240 "Football" sequence between STTP-SPIHT with $P = 16$ and MPEG-2 sequence at total transmission rate of 2.53 Mbps and channel bit error rates of 0.01. In this figure, (a), (c), (e), and (g) are typical results from STTP-SPIHT, and the frame numbers are 3, 7, 10, and 15 respectively, and (b), (d), (f), and (h) are the MPEG-2 decoded sequence with the same frames as STTP-SPIHT's. As we can see, the decoding failure in the STTP-SPIHT affects a small region with lower resolution only. However, in the MPEG-2 decoded sequence, the decoding failure produces distortions in random positions, and the positions are marked with black pixels or filled with another frame's image.

Table IV shows the comparison of average PSNRs with STTP-SPIHT, normal 3-D SPIHT and MPEG-2 at total transmission rates of 1 Mbps and 2.53 Mbps of $352 \times 240 \times 48$ YUV 4:2:2 color "Football" sequence with bit error rates (BER) of 0, 0.01 and 0.001. We downloaded the color RGB football sequence from [22] and converted it to YUV 4:2:2 format using the standard filtering and downsampling. This sequence contains a different scene from that of gray "Football" sequence. We can find that the average PSNRs for Y of STTP-SPIHT are 0.5 to 1 dB higher and U and V of STTP-SPIHT are 1 to 3 dB higher than that of MPEG-2 in noisy channel conditions.

VI. Conclusions

We have implemented a parallel SPIHT coding of video and have shown 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. The embedded sub-bitstreams of the

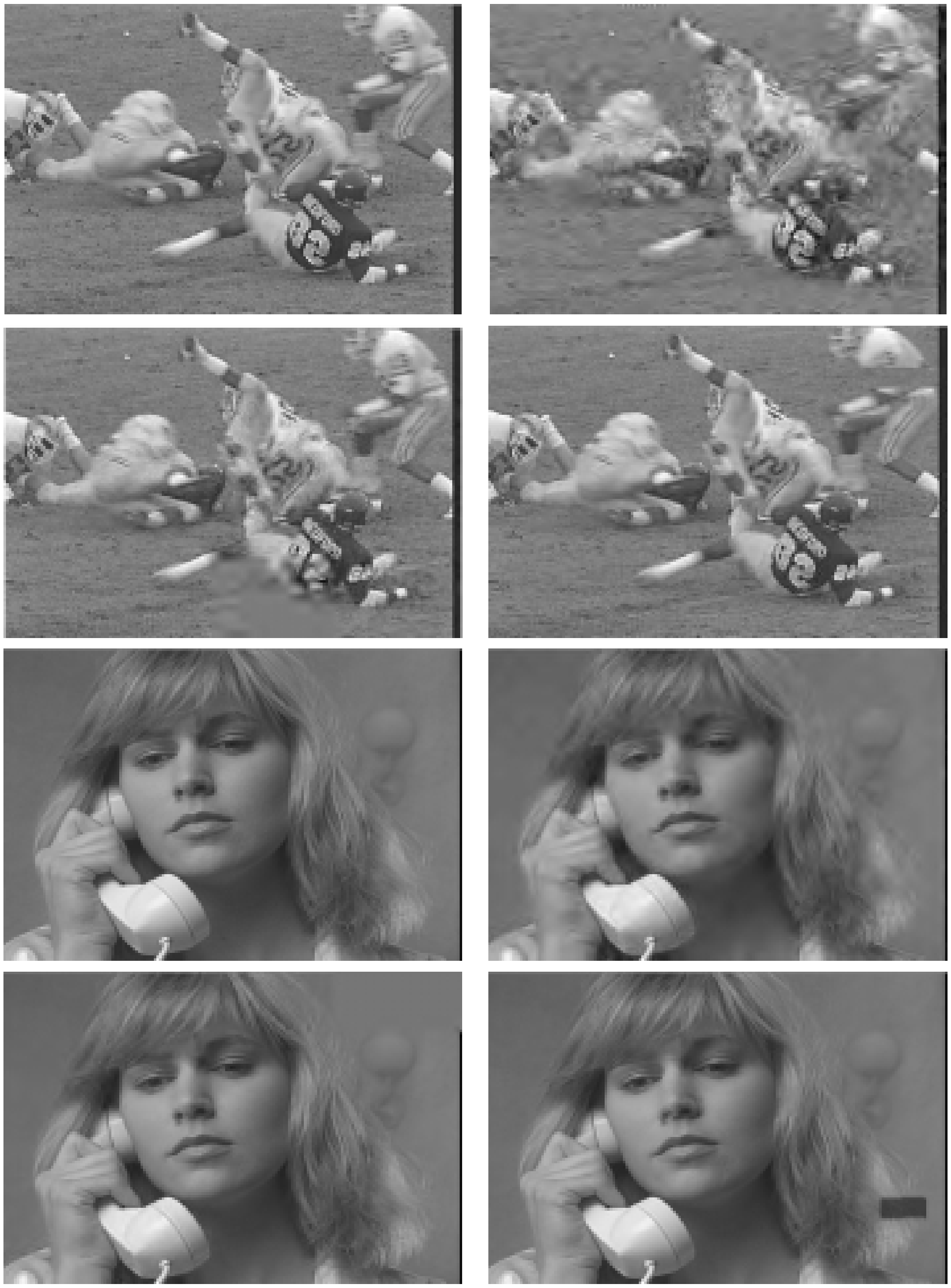


Fig. 19. (a) Top-left : 352 × 240 original "Football" sequence (frame15), (b) Top-right : 352 × 240 "Football" reconstruction using normal 3-D SPIHT/FEC with BER = 0.01. PSNR = 24.41dB, (c) Second row-left : 352 × 240 "Football" reconstruction (frame15) using STTP SPIHT(P=16)/RCPC with BER = 0.01. PSNR = 29.35dB, (d) Second row-right : MPEG-2/RCPC with BER = 0.01, PSNR = 27.45 dB, (e) Third row-left : 352 × 240 original "Susie" sequence (frame29), (f) Third row-right : 352×240 "Susie" reconstruction using normal 3-D SPIHT/RCPC with BER = 0.01. PSNR = 32.68 dB, (g) Bottom-left : 352 × 240 "Susie" reconstruction (frame29) using STTP-SPIHT(n=16)/RCPC with BER = 0.01. PSNR = 38.53dB, (h) Bottom-right : MPEG-2/RCPC with BER = 0.01, PSNR = 32.84 dB. Total transmission rate is set to 2.53 Mbps.

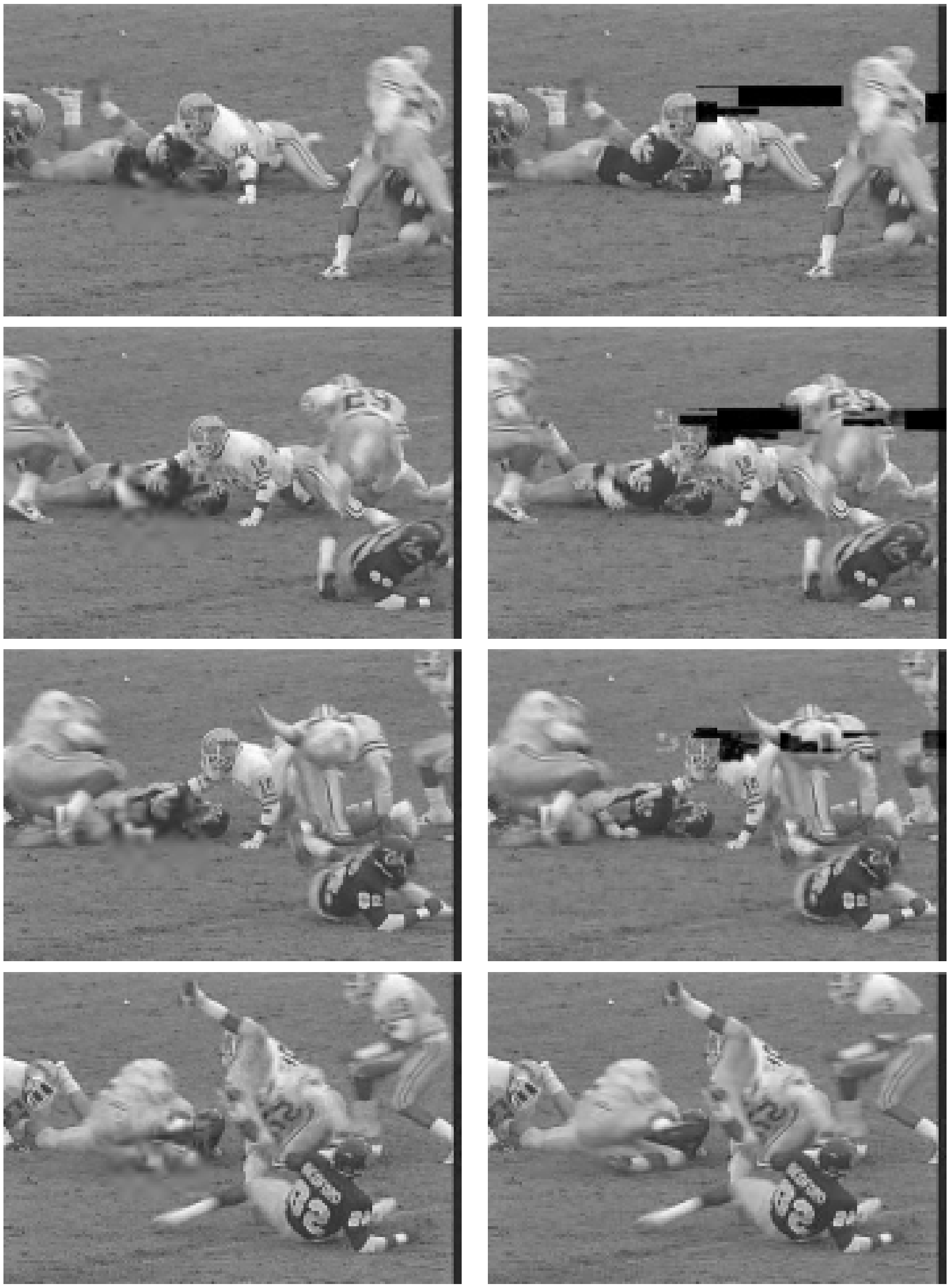


Fig. 20. (a) Top-left : 352×240 “Football” reconstruction (frame 3) using STTP-SPIHT ($P = 16$)/RCPC with BER = 0.01. PSNR = 30.82 dB, (b) Top-right : 352×240 “Football” reconstruction (frame 3) using MPEG-2/RCPC with BER = 0.01. PSNR = 21.24dB, (c) Second row-left : frame 7, PSNR = 29.85 dB , (d) Second row-right : frame 7, PSNR = 20.72 dB, (e) Third row-left : frame 10, PSNR = 30.49 dB, (f) Third row-right : frame 10, PSNR = 21.48 dB, (g) Bottom-left : frame 15, PSNR = 29.35 dB, (h) Bottom-right : frame 15, PSNR = 29.42 dB. Total transmission rate is set to 2.53 Mbps.

	Component	Y		U		V	
	BER(Bit Error Rate)	0		0		0	
1Mbps	STTP-SPIHT	29.19		38.65		36.55	
	MPEG-2	28.56		38.03		35.98	
	3-D SPIHT	29.77		38.55		36.64	
2.53 Mbps	STTP-SPIHT	34.51		40.81		39.61	
	MPEG-2	35.29		39.96		38.75	
	3-D SPIHT	35.18		40.95		39.81	
	BER(Bit Error Rate)	0.01	0.001	0.01	0.001	0.01	0.001
1Mbps	STTP-SPIHT/RCPC	26.08	28.42	36.63	38.03	34.14	35.96
	MPEG-2/RCPC ∗	25.95	26.17	34.82	35.03	33.04	33.39
	3-D SPIHT/RCPC	24.53	27.93	36.37	37.57	32.85	35.51
	3-D SPIHT/RCPC+ARQ	27.38	28.61	37.57	38.03	35.21	35.95
2.53 Mbps	STTP-SPIHT/RCPC	29.03	29.97	38.41	38.67	36.47	37.93
	MPEG-2/RCPC ∗	28.57	29.26	37.33	36.46	36.00	35.41
	3-D SPIHT/RCPC	24.83	28.84	36.56	38.13	33.24	36.15
	3-D SPIHT/RCPC+ARQ	31.95	33.66	39.55	40.38	37.95	39.08

∗ omits frames of failed decoding

TABLE IV

Comparison of average PSNRs (dB) of $352 \times 240 \times 48$ color "Football" sequence (YUV 4:2:2) with STTP-SPIHT ($P = 4$), normal 3-D SPIHT and MPEG-2 at total transmission rates of 1 Mbps and 2.53 Mbps with bit error rates (BER) of 0, 0.01, 0.001

STTP-SPIHT algorithm allow the reorganization of the full bitstream to achieve fidelity and frame rate scalability, but at a coarser level than in the full frame 3D-SPIHT algorithm. The STTP-SPIHT algorithm has proved to be much more robust and resilient to channel random bit errors than MPEG-2, which it outperforms even in noiseless channels.

STTP-SPIHT uses no motion estimation/compensation as does MPEG-2, so is not susceptible to temporal propagation errors. Other features demonstrated are region-of-interest encodings and decodings and multiresolution decoding. In fact, the system is a fully operational color video software codec with a parallel architecture suitable for hardware realization.

ACKNOWLEDGMENT

The authors wish to express their gratitude to Young Seop Kim for his advice and encouragement, and his idea of how to incorporate region-of-interest coding into the overall system.

References

[1] J. Hagenauer, *Rate-compatible punctured convolutional codes (RCPC codes) and their applications*, IEEE Transactions on Communications, vol. 36, pp. 389-400, April 1988.

[2] J. M. Shapiro, *Embedded image coding using zerotrees of wavelet coefficient*, IEEE Transactions on Signal Processing, vol. 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, vol. 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, March 1997.

[6] A. A. Alatan, M. Zhao, and A. N. Akansu *Unequal Error Protection of SPIHT Encoded Image Bit Streams*, IEEE Journal on Selected Areas In Communications, vol. 18, pp. 814-818, June 2000.

[7] P. G. Sherwood and K. Zeger *Progressive Image Coding for Noisy Channels*, IEEE Signal Processing Letters, vol. 4, pp. 189-191, July 1997.

[8] P. G. Sherwood and K. Zeger *Progressive Image Coding on Noisy Channels*, Proc. DCC, pp. 72-81, April 1997.

[9] H. Man, F. Kossentini, and M. J. T. Smith *Robust EZW Image Coding for Noisy Channels*, IEEE Signal Processing Letters, vol. 4, no. 8, pp. 227-229, August 1997.

[10] H. Man, F. Kossentini, and M. J. T. Smith *A Family of Efficient and Channel Error Resilient Wavelet/Subband Image Coders*, IEEE Transactions on Circuits and Systems for Video Technology, vol. 9, no. 1, pp 95-108, February 1999.

[11] C. D. Creusere *A New Method of Robust Image Compression Based on the Embedded Zerotree Wavelet Algorithm*, IEEE Transactions on Image Processing, vol. 6, no. 10, pp. 1436-1442, October 1997.

[12] C. D. Creusere *Robust image coding using the embedded zerotree wavelet algorithm*, Proc. Data Compression Conference, pp. 432, March 1996.

[13] C. D. Creusere *A family of image compression algorithms which are robust to transmission errors*, Proc. SPIE, vol. 2668, pp. 82-92, January 1996.

[14] Z. Xiong, B.-J Kim, and W. A. Pearlman *Progressive video coding for noisy channels*, In Proc. IEEE International Conference on Image Processing (ICIP '98), vol. 1, pp. 334-337, October, 1998.

[15] B.-J Kim, Z. Xiong, and W. A. Pearlman, and Y. S. Kim *Progressive video coding for noisy channels*, Journal of Visual Communication and Image Representation, vol. 10, pp. 173-185, 1999.

[16] B.-J Kim, Z. Xiong, and W. A. Pearlman *Low bit-rate scalable video coding with 3-D Set partitioning in Hierarchical Trees (3-D SPIHT)*, IEEE Trans. Circuits and Systems for Video Technology, vol. 10, pp. 1374-1387, December 2000.

[17] T. V. Ramabadran and S. S. Gaitonde *A Tutorial on CRC Computations*, IEEE Micro, vol. 8, pp. 62-75, August 1998.

[18] G. Castagnoli, J. Ganz, and P. Graber *Optimum Cyclic Redundancy-Check Codes with 16-Bit*, IEEE Transactions on Communications, vol. 38, pp. 111-114, January 1990.

[19] G. D. Forney, Jr. *The Viterbi algorithm*, Proc. IEEE, vol. 61, pp. 169-176, January 1994.

[20] N. Seshadri and C. Sundberg *List Viterbi decoding algorithm with applications*, IEEE Transactions on Communications, vol. 42, pp. 111-114, 1994.

[21] M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies *Image coding using wavelet transform*, IEEE Transactions Image Processing, vol. 1, pp. 205-220, 1992

[22] www.image.cityu.edu.hk/imagedb/