Unequal Error Protection of Embedded Video Bitstreams

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

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

ABSTRACT

This paper presents an unequal error protection of embedded video bitstreams using three dimensional SPIHT (3-D SPIHT) algorithm and Spatio-Temporal Tree Preserving 3-D SPIHT (STTP-SPIHT) algorithm. We have already proved the efficiency and robustness of the STTP-SPIHT in both of noisy and noiseless channels by modifying the 3-D SPIHT algorithm. We demonstrate that the 3-D SPIHT can also be error resilient against channel bit errors by dividing the embedded video bitstreams, and more error resilient when we divide the STTP-SPIHT bitstreams.

Keywords: unequal error protection, video compression, embedded bitstream, error resilience, SPIHT

1. INTRODUCTION

Wavelet zerotree image coding techniques were developed by Shapiro (EZW),[1] and further developed by Said and Pearlman (SPIHT),[2] 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[3,4] that employed a temporal wavelet transform instead of motion compensation, and compared the result with the video compression algorithms which are generally used 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.

To achieve robust video over noisy channels, the work of Sherwood and Zeger[5,6] was extended to progressive video coding by Kim *et al.* They have shown in Ref.[7,8] 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 bit stream. This algorithm is called STTP-SPIHT (Spatio-Temporal Tree Preserving 3-D SPIHT).[10] 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 Ref.[11–13] in the field of image coding area with EZW algorithm.[1]

In Ref.,[14] 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,[9] and improve the overall performance against channel bit errors.

Further author information: (Send correspondence to W.A.P.)
S.C.: E-mail: chos@rpi.edu
W.A.P.: E-mail: pearlw@rpi.edu

Applications of Digital Image Processing XXIV, Andrew G. Tescher, Editor,
Proceedings of SPIE Vol. 4472 (2001) © 2001 SPIE · 0277-786X/01/$15.00

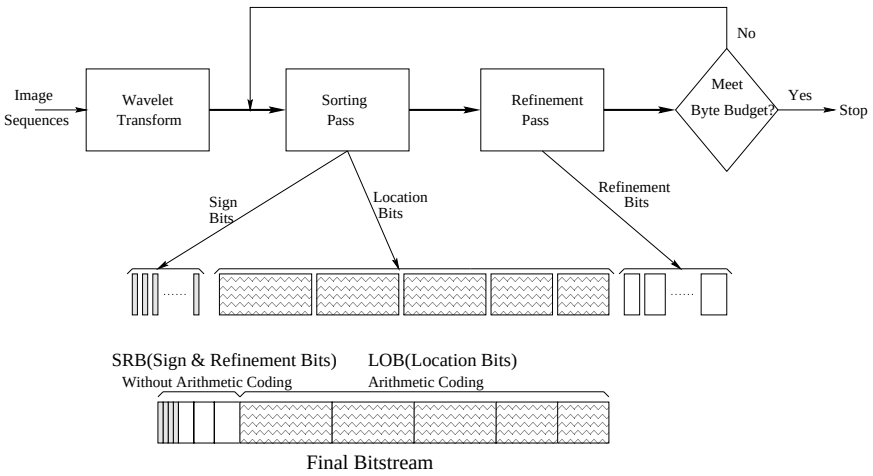
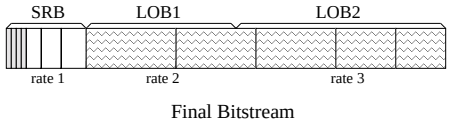


Figure 1. Unequal error protection of the 3-D SPIHT algorithm



$$\Gamma_{LOB1} < \Gamma_{LOB2} < \Gamma_{SRB}$$

Figure 2. Bit rate assignment of the unequal error protection

In this paper, we first show how the 3-D SPIHT encoded video bitstreams can be implemented to unequal error protection by dividing the embedded bitstreams, then present a hybrid coder which combines the STTP-SPIHT algorithm and unequal error protection. This method can effectively protect the very early decoding error, because we more strongly protect the beginning portion of the bitstream. Therefore we are less liable to decode regions of lower resolution compared to other regions.

The organization of this paper is as follows: Section 2 shows unequal error protection of the 3-D SPIHT algorithm. Section 3 shows unequal error protection of the STTP-SPIHT algorithm. Section 4 illustrates the source/channel coding rate. Section 5 provides simulation results. Section 6 concludes this paper.

2. UNEQUAL ERROR PROTECTION OF THE 3-D SPIHT

The 3-D SPIHT compression kernel is composed of 2 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 are often called the significance map.

We can 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 can be classified by themselves (LOB). 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 - 25% of the original bitstream, depending on the rate.

In addition, the 3-D SPIHT algorithm has an important property that is all the compressed bits are positioned in the order of importance. 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. 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 can

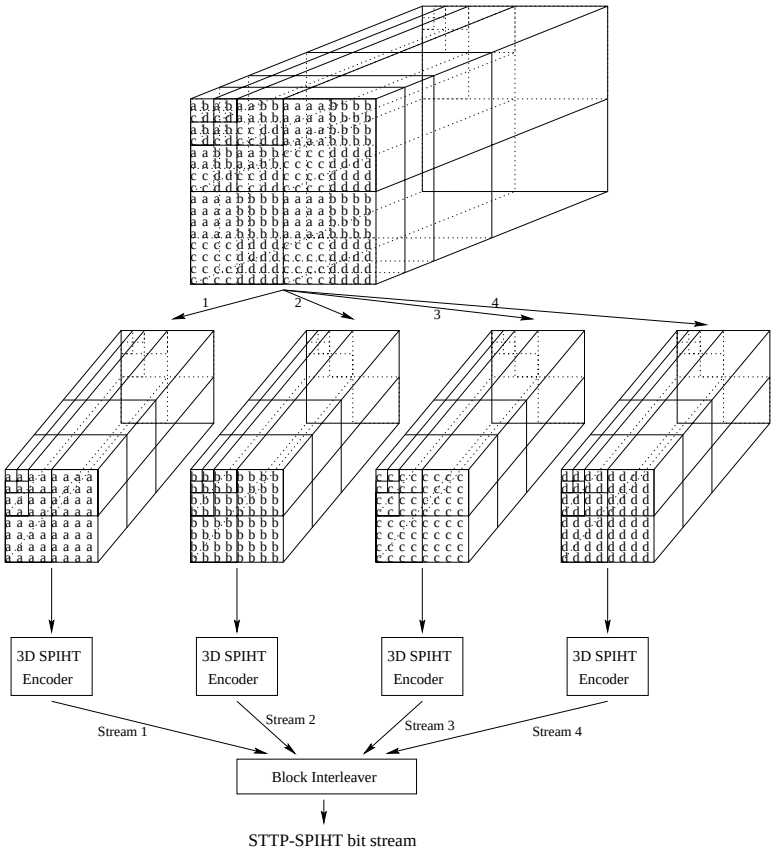


Figure 3. Structure of the Spatio-Temporal Tree Preserving 3-D SPIHT(STTP-SPIHT) compression algorithm

not reconstruct anything from the bitstream. From this idea, we can further sub-partition the LOB class to two classes LOB1 and LOB2, corresponding 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 LOB1 and LOB2, each with stronger protection (lower channel code rate) than SRB, but with LOB1 receiving a lower coder rate (higher protection) than LOB2. Figure 1 and Figure 2 graphically illustrate this idea. Figure 1 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. Figure 2 presents the bitrate assignments according to their bit error sensitivities and importance. As we can see in this figure, LOB1 should be highly protected, because these bits are more important than others in terms of bit sensitivities and the order of importance, then LOB2 and SRB can be protected with successively higher channel coding rates.

As depicted in the Figure 2, we send the SRB bits first, then send LOB bits next. 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 buffer, and these bits are used with LOB bits together. As we mentioned before, the size of SRB bits ranges from 20-25% of the total bitstream for source code rates about 1 bpp (2.53 Mbps). 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 just need one negligible additional item of information, the SRB size.

3. UNEQUAL ERROR PROTECTION OF THE STTP-SPIHT

Figure 3 shows the structure and the basic idea of the STTP-SPIHT compression algorithm. STTP-SPIHT algorithm divides 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

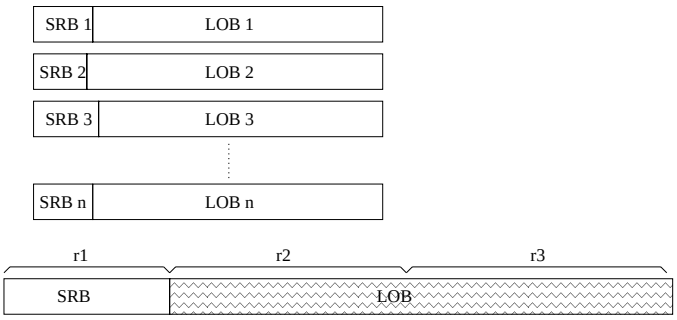


Figure 4. Bitstream of STTP-SPIHT algorithm

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 spatio-temporal tree structure of normal 3-D SPIHT,[3,4] and these trees correspond to 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 $P = 1$, and we can flexibly choose P. When we choose P, the number of coefficients of x and y axis of sub-dimension should be divisible by 16 for 3 level decomposition, and divisible by 8 for 2 level decomposition. If the axis is not divisible by those numbers, we should extend the original image to be divisible. For 352×240 Football sequences, we can choose P from 1 up to 330.

STTP-SPIHT gave us excellent results in both noisy and noiseless channel conditions while preserving all the desirable properties of the 3-D SPIHT.[10] However, this method is also susceptible to early decoding error, and this error results in small region with lower resolution than the surrounding area. Sometimes, this artifact occurs in an important region. To avoid this, early decoding error should be prevented to guarantee a minimum quality of the whole region.

We can still partition the STTP-SPIHT sub-bitstreams according to their bit sensitivities and the order of importance. Figure 4 shows us this idea. Each substreams are divided into SRB and LOB, denoted by SRB1 - SRBP, and LOB1 - LOBP. As we used in the 3-D SPIHT algorithm, we send all the SRBs first, and then send LOBs. Unlike in the case of 3-D SPIHT, we used block interleaving/deinterleaving scheme for LOB area to maintain progressiveness. The overhead of this method is the information bits which are saved in each sub-bitstream header to convey its SRB size.

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 interleaved bitstream, and decodes the bitstreams with SRBs together. Therefore this bitstream protects early bitstream more strongly with little loss of progressiveness.

4. SOURCE/CHANNEL CODING RATE

Figure 5 shows the system description of 3D/STTP-SPIHT with RCPC coder. The functions of block interleaving and deinterleaving are needed only for the STTP-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 cyclic redundancy code (CRC)[15,16] 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.

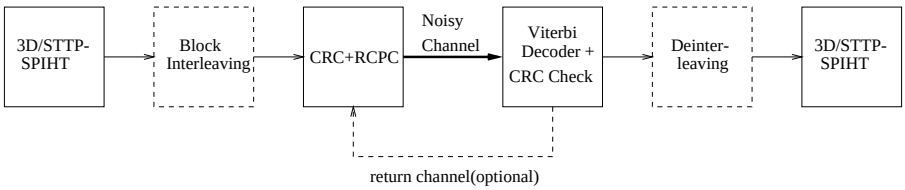


Figure 5. 3D/STTP-SPIHT with RCPC system framework

The effective source coding rate R_{eff} for the original 3-D SPIHT 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. 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 protection, we first get the $R_{eff_{SRB}}$ and M_{SRB} according to the Equation(1). Then $R_{eff_{LOB1}}$ and $R_{eff_{LOB2}}$ can be calculated by

$$\frac{r_{LOB1}}{N+c+m} R_{LOB1} + \frac{r_{LOB2}}{N+c+m} R_{LOB2} = M - M_{SRB}, \tag{2}$$

where $R_{LOB1} + R_{LOB2} = R_{total}$ - R_{SRB}. Therefore, we can get R_{LOB1} and R_{LOB2}. 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 BER (Bit Error Rate) is 0.01.

		r	Source	Channel
3-D SPIHT		2/3	60%	40%
UEP-SPIHT	SRB	4/5	14%	5.5%
	LOB1	4/7	14%	13.5%
	LOB2	2/3	32%	21%

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

5. RESULT

In our test of error resilience, we assume that the channel is binary symmetric (BSC). Sixteen frames in a GOF (Group Of Frames) are used, and a dyadic three level transform using 9/7 biorthogonal wavelet filters[17] is applied to the image sequences. For the 3-D SPIHT, we send the bitstream sequentially in 200 bit packets, and for the STTP-SPIHT, we interleave the streams in 200 bit packets to maintain embeddedness, and 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" sequences. The distortion is measured by the peak signal to noise ratio (PSNR)

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

where MSE denotes the mean squared error between the original and reconstructed image sequences. All PSNR's reported for noisy channels are averages over twenty (20) independent runs.

We used the same set of parameters for the CRC parity checker and RCPC channel coder[10,7,8]: $N = 200$, $c = 16$, and $m = 6$. 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. We set the total transmission rate R_{total} to 2.53 Mbps, $r = 2/3$ for $\epsilon = 0.01$ and 8/9 for $\epsilon = 0.001$. In our case of unequal error protection, we use a certain r only for LOB2, and use the one level higher rate available to us for the SRB, and one level lower rate for the LOB1 with the same transmission rate of 2.53 Mbps. In the case of $\epsilon = 0.01$, the RCPC rate for LOB2 = 2/3, and the rate for the SRB = 4/5, and

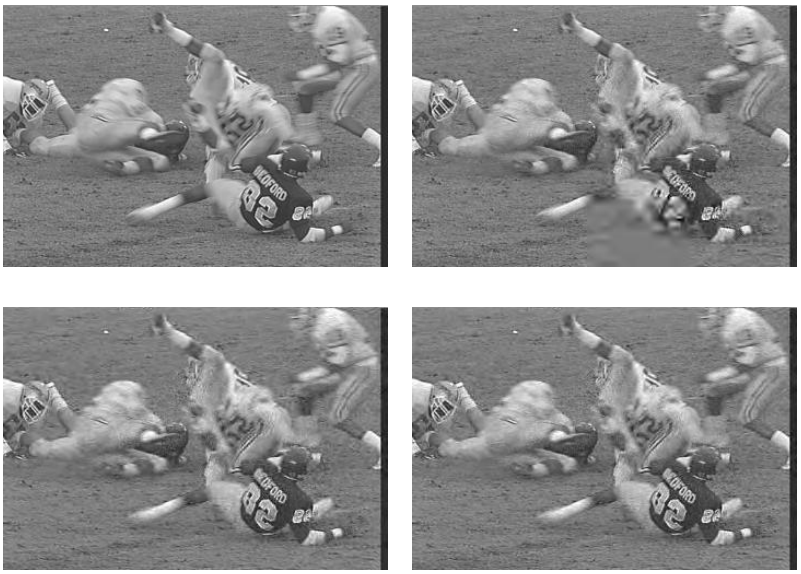


Figure 6. 352×240 "Football" sequence (frame15) (a)Top-left : Original sequence (b)Top-right : using STTP-SPIHT (n=16)/RCPC with BER = 0.01. PSNR = 29.35 dB (c)Bottom-left : using 3-D SPIHT with unequal error protection with BER = 0.01. PSNR = 29.06 dB (d)Bottom-right : using STTP-SPIHT with unequal error protection with BER = 0.01. PSNR = 30.14 dB

the rate for the LOB1 = 4/7. Once we decide the channel coding rate, we can easily calculate the bit budget for the three classes using Equation(1) and (2).

At the destination, the Viterbi decoding algorithm[18,19] is used to convert the packets of the received bit-stream into a 3D/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.

Table 2 shows the comparison of average PSNRs among 3-D SPIHT, UEP-SPIHT, and UEP/STTP-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

BER	3-D SPIHT	UEP-SPIHT	UEP/STTP-SPIHT
0.01	24.5 dB	29.43 dB	30.04 dB
0.001	28.2 dB	30.18 dB	31.49 dB

Table 2. Comparison of average PSNR among 3-D SPIHT, UEP-SPIHT, and UEP/STTP-SPIHT at total transmission rate of 2.53 Mbps

the hybrid method, when BER is 0.01 the average PSNR is 30.04 dB, which is a little higher than that of equal error protection of STTP-SPIHT, and when BER is 0.01 the average PSNR is 31.49 dB. In addition to the higher PSNR, we have prevented very early decoding error using the lower channel coding rate for the LOB1. The merit over the unequal error protection of the 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 STTP-SPIHT prevents very early decoding error effectively, and still operates until P decoding failures occur.

6. CONCLUSION

We have implemented unequal error protection for the 3-D SPIHT and the STTP-SPIHT algorithm. The result is very promising while sacrificing small degree of progressiveness. The most remarkable thing is we can avoid the early

decoding error for the STTP-SPIHT.

REFERENCES

1. J. M. Shapiro, "Embedded image coding using zerotrees of wavelet coefficient," *IEEE Transactions on Signal Processing* **41**, pp. 3445–3562, December 1993.
2. 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.
3. 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.
4. B.-J. Kim, Z. Xiong, and W. A. Pearlman, "Low bit-rate scalable video coding with 3d set partitioning in hierarchical trees (3d spiht)," *IEEE Trans. Circuits and Systems for Video Technology* **10**, pp. 1374–1387, December 2000.
5. P. G. Sherwood and K. Zeger, "Progressive image coding for noisy channels," *IEEE Siginal Processing Letters* **4**, pp. 189–191, July 1997.
6. P. G. Sherwood and K. Zeger, "Progressive image coding on noisy channels," *Proc. DCC '97* , pp. 72–81, April 1997.
7. B.-J. Kim, Z. Xiong, and W. A. Pearlman, "Progressive video coding for noisy channels," *Proc. 1998 IEEE Int. Conf. On Image Processing (ICIP'98)* , October 1998.
8. B.-J. Kim, Z. Xiong, W. A. Pearlman, and Y. S. Kim, "Progressive video coding for noisy channels," *Journal of Visual Communication and Image Representation* **10**, pp. 173–185, 1999.
9. J. Hagenauer, "Rate-compatible punctured convolutional codes (rcpc codes) and their applications," *IEEE Transactions on Communications* **36**, pp. 389–400, April 1988.
10. S. Cho and W. A.Pearlman, "Error resilient compression and transmission of scalable video," *Applications of Digital Image Processing XXIII, Proceedings SPIE* **4115**, pp. 396–405, 2000.
11. C. D. Creusere, "A family of image compression algorithms which are robust to transmission errors," *Proc. SPIE* **2668**, pp. 82–92, January 1996.
12. C. D. Creusere, "Robust image coding using the embedded zerotree wavelet algorithm," *Proc. Data Compression Conference* , p. 432, March 1996.
13. C. D. Creusere, "A new method of robust image compression based on the embedded zerotree wavelet algorithm," *IEEE Transactions on Image Processing* **6**, pp. 1436–1442, October 1997.
14. A. A. Alatan, M. Zhao, and A. N. Akansu, "Unequal error protection of spiht encoded image bit streams," *IEEE Jounral on Selected Areas In Communications* **18**, pp. 814–818, June 2000.
15. T. V. Ramabadran and S. S. Gaitonde, "A tutorial on crc computations," *IEEE Micro* **8**, pp. 62–75, August 1988.
16. 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.
17. M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies, "Image coding using wavelet transform," *IEEE Transactions Image Processing* **1**, pp. 205–220, 1992.
18. G. D. Forney, "The viterbi algorithm," *Proc. IEEE* **61**, pp. 169–176, February 1984.
19. N. Seshadri and C. Sundberg, "List viterbi docoding algorithm with applications," *IEEE Transactions on Communications* **42**, pp. 313–323, 1994.