A Doubly Error Resilient Coder of Image Sequences

William A. Pearlman and Yang Hu
PrimaComp, Inc., Niskayuna, NY 12309, USA

Abstract

Many scenarios require single frame random access and error resilience in coding, transmission, and decoding of image sequences. We propose in this paper a SPIHT coder that encodes a sequence of images one frame at a time and uses two methods together to achieve strong error resilience. The first method groups wavelet coefficients into a number of tree blocks and the second one encodes these tree blocks independently using SPIHT with progressive significance maps. Because a substantial part of a progressive significance map contains a fixed-length code stream called the complementary significance map (comp-sig-map), transmission bit errors do not propagate within the comp-sig-map. Furthermore, since the block trees are encoded independently, no error in a given bitstream propagates to another bitstream. Simulations show that the two methods together show stronger error resilience than either one separately. Furthermore, because the groups are formed by interleaved spatial orientation trees, faulty reconstruction from bit errors affecting a given tree-block can be concealed by estimating the true wavelet coefficients from neighboring coefficients in adjacent error-free spatial orientation trees.

Introduction

This paper proposes an efficient image sequence coder endowed with unique built-in error resilience. Gray video, volume medical images, and hyperspectral images are examples of image sequences. For this particular coder, the image frames (or slices or bands) are encoded independently, so that reception bit errors cause damage only in the frame in which they occur. Also, independent frame coding allows rapid random access decoding of single frames. In this work, we use the SPIHT compression algorithm [1] to encode the separate frames. SPIHT supports desirable features such as efficiency, precise rate control, and progressive transmission. For image sequences, it is desirable that the decoded frames have uniform quality, and we shall show how we achieve that objective very simply with SPIHT compression.

One of the drawbacks of SPIHT and variable length coders is their sensitivity to channel errors. The compressed bitstream SPIHT and other set partition coders divide naturally into two components: a significance map that conveys location information; and a value bitstream that conveys intensity information of signs and lower order bits of wavelet coefficients. A single bit error in the significance map can cause a catastrophe in image reconstruction due to error propagation. The earlier the error the worse is the degradation. To overcome this weakness, a number of error resilient coding techniques have been proposed in the literature. Forward error correction (FEC), initially proposed by Sherwood and Zeger [2, 3], is among the first joint source/channel coding schemes using concatenated code, where an outer cyclic redundancy check (CRC) code detects errors and an inner rate compatible punctured convolutional (RCPC) code corrects as many errors as possible. The FEC achieves good performance for a class of binary symmetric channels (BSC) with known error rates. Then it is developed for burst-error-prone channels with more powerful channel codes, such as in [4] with Reed-Solomon code and in [5] with Turbo code. Further improved error resilience is obtained through combining FEC with unequal error protection (UEP) techniques (e.g., in [6, 7]), where more important data are protected by stronger channel codes.

All of the aforementioned techniques employing FEC produce error resilience at the expense of substantial increase in channel transmission rate. Another class of techniques attempts to build natural error resilience into the compressed bitstream without resorting to FEC. Such techniques, which have been adopted by JPEG2000 and the new CCSDS image compression standard, partition the data into groups and encode each group into packets that can be decoded independently [8, 9, 10, 11]. Thanks to independent packetization, error propagation is limited to the packets within which channel errors occur. Error resilient entropy coding (EREC) [12] can be used to reorganize these variable-length packets into fixed-length data slots before multiplexing and transmission. Therefore, the synchronization of the start of each packet can be automatically obtained at the receiver without synchronization words. Self-synchronizing variable length coding [13] is a mechanism that limits the effects of an error to a small portion of the data. Typically, the natural error resilience techniques manage to stop the error propagation at the expense of small loss in coding efficiency. There exist hybrid schemes that protect naturally error resilient bitstreams with FEC. For example, [14, 15] apply channel codes to independently coded packets to further improve the robustness against both random error and packet erasure. The naturally error resilient coder proposed in this paper is likewise compatible with applying error-correcting codes to its compressed bitstreams.

This work focuses on building natural error resilience into the compressed bitstream. We first use the method of Cho and Pearlman [15] of dividing the wavelet transform into a number of groups that are encoded independently. Their groups were cubes of three-dimensional wavelet coefficients, whereas ours will be two-dimensional blocks. Therefore, received bit errors affect only the group in which they occur. No expansion of transmission bandwidth occurs with this scheme, but there is a small loss in compression efficiency due to the overhead of multiple bitstream headers. Every bitstream includes a significance map that is especially vulnerable to propagation of an early bit error. In order to reduce this vulnerability, Hu *et al.* [16] introduced the *progressive significance map* that contains a fixed-length codeword portion in which errors can not propagate. With this double layer of natural error resilience in every frame and no interdependence between frames, we demonstrate an image sequence coder that is efficient

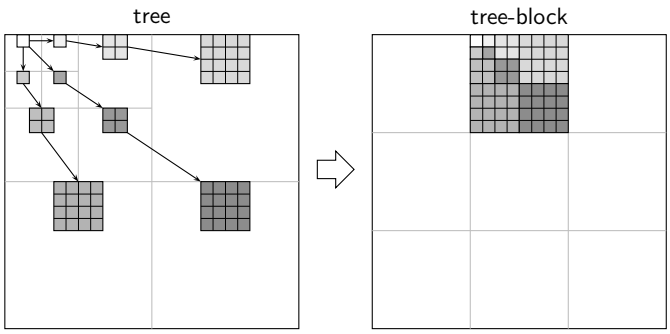


Figure 1. *A spatial orientation tree and its rearrangement into a tree-block placed in the position of the image region it represents.*

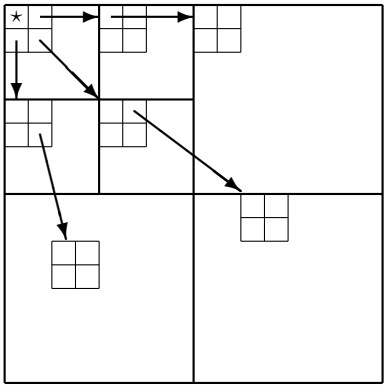


Figure 2. *Examples of parent-offspring dependencies in the spatial-orientation trees. Coefficients in the LL band marked "*" have no descendants.*

in compression and superior in error resilience.

Dispersive Tree-Block Group Coding

A spatial discrete wavelet transform (DWT) is calculated for every image frame. The DWT of a frame is organized into contiguous groups of coefficients called subbands that are associated with distinct spatial frequency ranges. One can view the DWT as a number of non-overlapping spatial orientation trees (SOT's). These trees are rooted in the lowest frequency (LL) subband and branch successively to higher frequency subbands at the same spatial orientation. An example of a single SOT in the wavelet transform domain is shown in Figure 1. This SOT represents the region of the image in the same relative position as its root in the lowest frequency subband. Figure 1 shows the rearrangement of the SOT into a tree block placed in the image region it represents.

In this paper, the SOT or tree-block is encoded using the SPIHT algorithm. The SPIHT algorithm usually operates on a differently configured SOT, shown in Figure 2, where the $2x2$ blocks in the lowest frequency (LL) subband are the seminal elements. The points in the block are roots of SOT's, except for the upper left point, which has no descendants. One can view the tree block corresponding to Figure 1 as the merged SOT's of the $2x2$ block and its upper left point. The sets in SPIHT are offspring sets, which are direct descendants of the roots, and grand-descendant sets, which are descendants of the offspring. The SPIHT algorithm searches for the locations of significant sets. When it deems a set as significant (at a given threshold or bitplane level), it writes a "1" to the significance map; otherwise, it writes a "0". This map determines the execution path of the algorithm, so when received correctly at the decoder, correct decoding will transpire.

The initial threshold of the algorithm corresponds to the top bitplane of the coefficient with highest magnitude. The algorithm searches for significance successively from the highest bitplane to some lower bitplane, depending on the target rate or distortion. The current implementation allows decrease to the bottom or zero bitplane. Using reversible (integer-to-integer) wavelet filters, perfectly lossless decoding is achievable when coding through the bottom bitplane of every frame. Since the coded bits can easily be counted, one can set the bit rate precisely. However, for image sequences, information content varies among the frames, so the same bit rate should not be set for all the frames. Instead, we wish to obtain the same visual quality for every frame. Coding every frame through the same bitplane or through the same fraction of the same bitplane approximately achieves that objective. In this work, where we just wish to demonstrate error resilience, we set the same minimal bitplane for every frame.

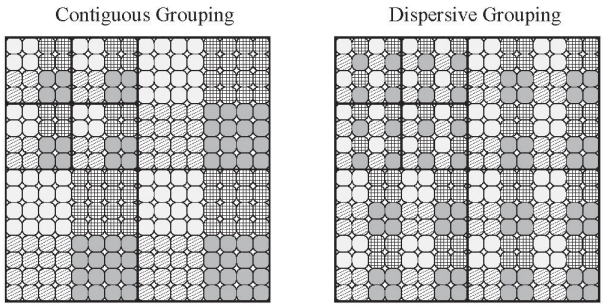


Figure 3. *Example of $S = 4$ contiguous and dispersive groupings in a $16x16$ DWT.*

The usual operation of SPIHT is to visit all the SOT's rooted in the lowest frequency subband at the same given threshold to test significance of the sets. In this way, we get a single compressed bitstream from the encoder. Within this bitstream is a significance map in which a single bit error will cause catastrophic decoding error in the rest of the bitstream. We want the encoder to produce a number of independent sub-bitstreams, so that no error in a given sub-bitstream affects decoding in another sub-bitstream. Therefore we divide the coefficients in the lowest frequency subband into a number of groups, denoted by S, where the merged SOT's of every group are encoded independently. $S = 1$ indicates normal SPIHT, which encodes the LL subband as a single group. Figure 3 shows two possible group formations for $S = 4$ groups or sub-bitstreams in a $16x16$ DWT. The fixed intervals in the right hand dispersive grouping are 2 in each direction and so is designated as a 2x2 grouping. In our experiments, we chose a $4x4$ grouping, where the fixed intervals are 4 in each direction.

The left-hand graphic shows groups formed from contiguous points, which can be single coefficients or $2x2$ blocks, while the right-hand one shows dispersive groups formed by gathering points at fixed intervals. Suppose a bit error occurs early in the sub-bitstream belonging to the top right contiguous group. Then the upper right quadrant of the decoded image will show serious

degradation. If the other three sub-bitstreams are received correctly, then the other three quadrants of the decoded image will be reconstructed without error. However, suppose a bit error occurs early in the sub-bitstream belonging to the dispersive group formed by the darker solid gray coefficients. The decoded image will show a $4x4$ array of degraded pixels (or small square regions) in the same relative positions in the decoded image as the same-colored coefficients in the lowest frequency subband. The errors in these isolated pixels can be effectively concealed by estimating the true values of their transform coefficients from their surrounding coefficients decoded from the sub-bitstreams of the other groups. This additional capability of error concealment makes the dispersive group formation very attractive for use in error-prone channel environments.

In general, the number of groups or sub-bitstreams S depends on the dimensions of the lowest frequency subband, which in turn depends on the image dimensions and the number of wavelet decompositions. It is hard to derive a general formula for the number of possible groups, but if the image has M rows and M columns, and the number of decompositions is D, then there are $M^2/2^{2D}$ coefficients in the lowest frequency subband. The largest fixed interval of $M/2^{2D}$ gives the maximum $S = M/2^{D+1}$. Typically we find that $S = 16$ provides satisfactory levels of error resilience and compression efficiency. The penalty for using too many sub-bitstreams is the amount of overhead needed for so many headers.

Progressive Significance Maps

Encoding the wavelet transform into independent and separate sub-bitstreams assures that transmission errors affect only the sub-bitstream in which they occur. That property provides a certain level of error resilience. But every significance map within the sub-bitstreams is still vulnerable to catastrophic error propagation caused by a received bit error. Hu, Pearlman, and Li [16] introduced the structure of the *progressive significance map*, wherein a major portion is not vulnerable to catastrophic error due to its being written wth fixed-length codewords. The progressive significance map comprises two segments: the first, called the *sum map*, indicating the numbers of significant sets among the four emanating from the branch nodes in the SOT; and the second, called the *complementary map*, indicating the location patterns of those significant sets. The complementary map is written with fixed length codewords, so that bit errors there cause only local damage in the decoded image. Bit errors in the sum map cause incorrect numbers of significant coefficients within the group, so these errors propagate. The sum map is arithmetic coded to save bits for storage or transmission. The size of the arithmetic-coded sum map is typically about 60% of the size of the conventional (arithmetic-coded) significance map, so we have reduced the footprint of the vulnerability by 40% in every sub-bitstream. A progressive significance map in every sub-bitstreams adds another layer of error resilience to our coding method.

Results

We wish to demonstrate the property of error resilience, while trying to emulate a realistic simulated transmission scenario as much as possible. Toward this objective, we must assume error free headers in the codestream, else decoding can not proceed. The order of the segments in every bitstream (or sub-bitstream) is sum map, complementary map, sign bits, and refinement bits. We wish to protect an early portion of the sum map from errors to prevent catastrophic error propagation that would result from errors early in the bitstream. Recall that only sum map errors result in error propagation, while errors in the other segments cause only local damage. Therefore, we shall keep error free a portion that is the first 5% of every significance map and add bit errors randomly to the remaining 95%. The following sign and refinements bits are kept error free, so that we can assess the impact of errors in the significance maps. We assume the transmission model of the binary symmetric channel, so shall inject bit errors randomly into the remaining parts of every bitstream at a specified error rate. We have chosen the error rate of 0.001 for all of our tests. The quality of the reconstruction of the error-laden image depends mainly on the position of the first error. The earlier the error, the worse is the quality; the later the error the better is the quality. The position of the first error is only weakly dependent on the error rate, so one error rate suffices to demonstrate error resilience. Due to the randomness of the error positions, we conduct 10 trials of injecting bit errors and decoding the codestream and report the average of the PSNR's. It turned out that the standard deviations in all cases were small fractions of a dB, so ten trials were more than sufficient.

We present here results of simulations with three types of grayscale image sequences: (1) the CT_SKULL volume medical image; (2) the CUPRITE hyperspectral image; and (3) the SALESMAN gray video sequence. All are stored as 8 bits per pixel. The pixels of CT_SKULL and SALESMAN are originally 8 bits, but CUPRITE was scaled from its original 16 bits to 8 bits, so that we could compare PSNR's in the same range. In all cases, we used 64 frames (slices or bands) in our tests. We chose to compare four coding modes: SPIHT with conventional or regular significance map (R); SPIHT with dispersive 4x4 groups and conventional significance map (R/4x4); SPIHT with progressive significance map (P); and SPIHT with dispersive 4x4 groups and progressive significance map (P/4x4). We did not attempt error concealment in order to report the effect of the dispersive groups alone on error resilience.

The results of our simulations appear in Figures 4, 5, and 6. The trends are the same for the three sequences; they just differ in the PSNR's obtained for the same rates due to their different degrees of compressibility. We note first that for no bit errors, the four curves for the different SPIHT modes are quite close together, indicating that there is small loss in efficiency, 2% for progressive map and 8-9% for dispersive 4x4 groups, suffered for the two error resilient methods. One also notes that the dispersive 4x4 grouping method alone attains average gains in PSNR between 0.5 to 2.5 dB over regular or prog-map SPIHT for error rate of 0.001. In fact, for all three sequences the top performer is P/4x4, SPIHT with 4x4 dispersive groups producing 16 sub-bitstreams with progressive significance maps. In some cases, the gains of P/4x4 are as much as 5 dB.

The amount of data is too large to record here, so we present here the results of the CT_SKULL sequence decoded through the last threshold of $2^3 = 8$ for each test case, namely the points for this threshold in Fig. 4. Recall that for a given case, stopping the decoder after a given threshold yields approximately the same distortion for each frame. The numerical results for regular (R) and progressive (P) significance maps, with 1x1 (non-dispersive)and 4x4 dispersive groups for the error rate of 0.001 and 5% bsig-

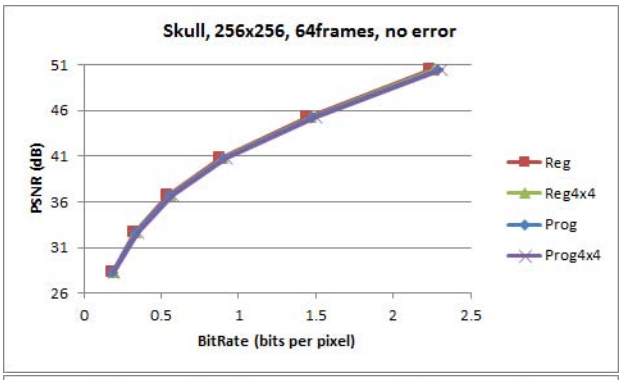
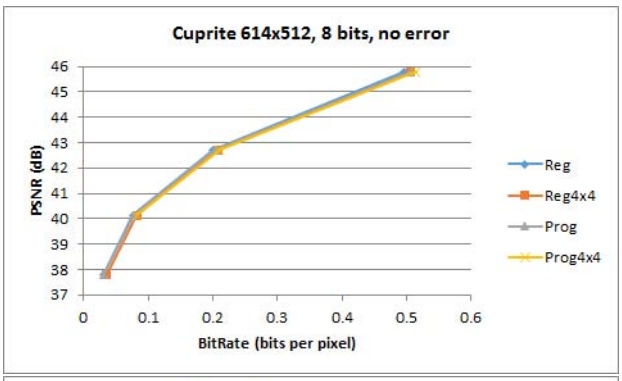


Figure 4. *CT_SKULL plots of PSNR vs. Rate for SPIHT error resilience modes.*



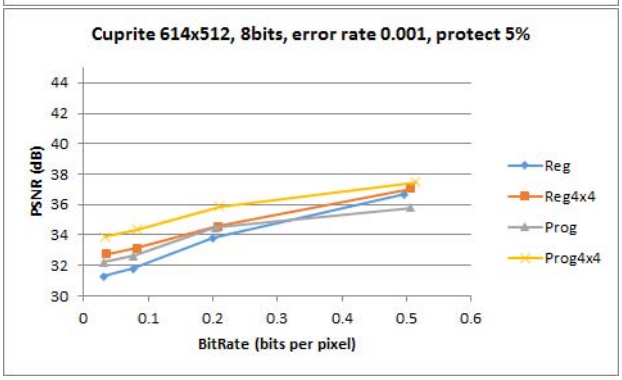
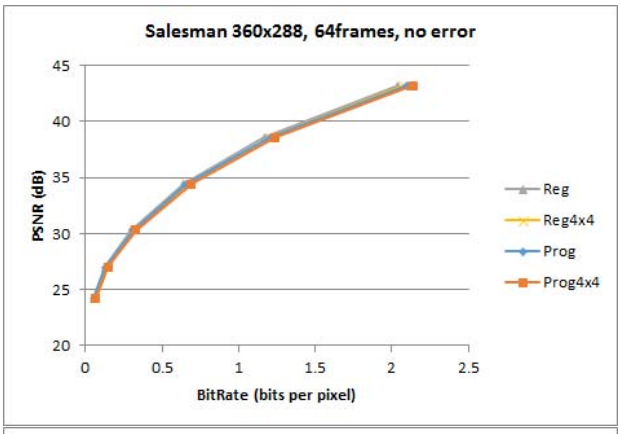


Figure 5. *CUPRITE plots of PSNR vs. Rate for SPIHT error resilience modes.*



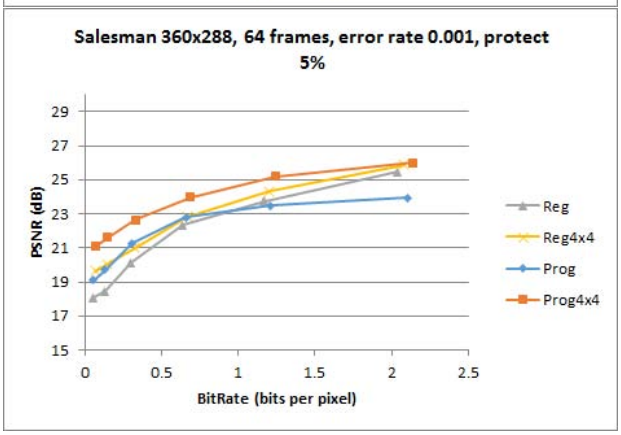


Figure 6. *SALESMAN plots of PSNR vs. Rate for SPIHT error resilience modes.*

Table 1. Average rates and PSNR's for test cases of CT_SKULL

Map Type	Dispersive Spacing	Error Rate	Protect %	Ave. Rate (bpp)	PSNR
R	1 x 1	0	-	0.881	40.807
R	1x1	0.001	5	0.881	21.314
R	4x4	0.001	5	0.952	24.217
P	1x1	0.001	5	0.902	22.735
P	4x4	0.001	5	0.975	27.251

nificance map protection are shown in Table 1. Also included there for reference is the case of zero error for R map and non-dispersive (1x1) groups. The PSNR would be exactly the same for the other three cases, but the average rates will be somewhat higher. Note especially that the 4x4 grouping gives about a 3 dB gain in PSNR over the non-dispersive grouping using the regular map and a 6 dB gain using the progressive map. Generally the progressive map does better in error resilience at lower rates and hence shorter bitstreams. The 4x4 grouping generates 16 short bitstreams, so does quite well for such a scenario. In order to show a typical example of the visual effects of the error resilience in these cases, we have extracted their decoded third frames of the CT_SKULL sequence and display them in Fig. 7. The third frame results are fairly typical of the trends in the other frames. Gains in fidelity are clearly visible for the 4x4 grouping (16 bitstreams) compared to the 1x1 grouping (single bitstream) for both the regular (R) and progressive (P) significance maps. Utilizing progressive maps brings considerable gains in fidelity to both groupings. The P/4x4 reconstructed frame (lower right image) is markedly superior in fidelity to the other error-degraded reconstructions.

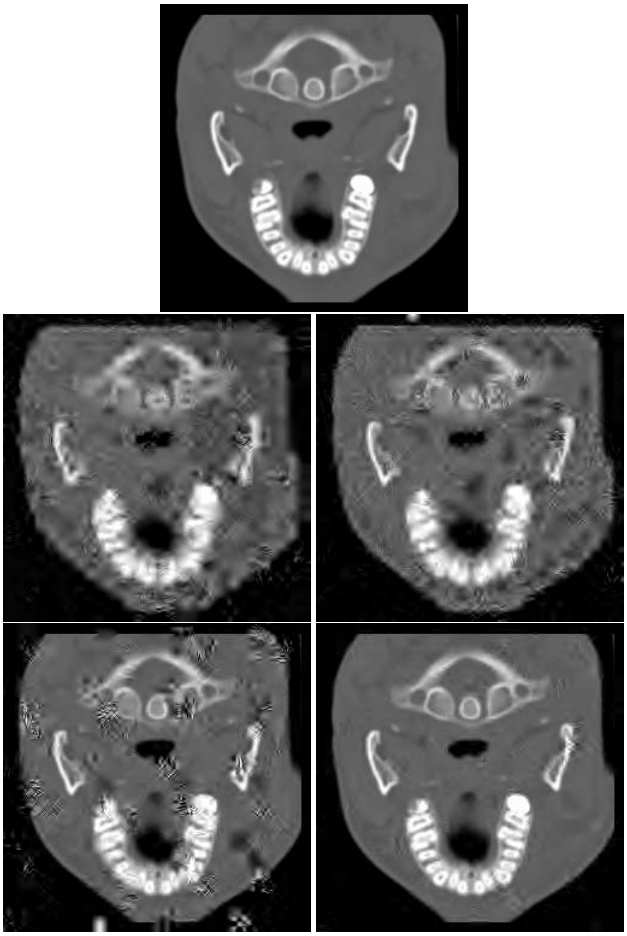


Figure 7. *Reconstructed 3rd frames of CT_SKULL decoded through last threshold of 8. Top is R/1x1, 0 error rate; middle row is R/1x1 and R/4x4, with 0.001 error rate and (significance map) protection of 5%; bottom row is P/1x1 and P/4x4 with 0.001 error rate and protection of 5%.*

Conclusions

We have developed a SPIHT image sequence coder with superior error resilience properties. The method of encoding dispersive groups of wavelet coefficients in spatial orientation tree blocks using SPIHT produces gains in error resilience either with regular or progressive significance maps. Using dispersive group coding that produces multiple bitstreams, each with the progressive significance map, gives the highest resistance to channel bit errors in the range of rates tested.

Acknowledgements

We gratefully acknowledge the support of the Office of Naval Research under Contract No. N00014-05-1-0507.

References

[1] A. Said and W. A. Pearlman, "A new, fast, and efficient image codec based on set partitioning inhierarchical trees," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 6, no. 3, pp. 243–250, June, 1996.

[2] P. Sherwood and K. Zeger, "Progressive image coding on noisy channels," in *In Proc. Data Compression Conferenc (DCC)*, pp. 72–81, March, 1997.

[3] P. Sherwood and K. Zeger, "Progressive image coding for noisy channels," *IEEE Signal Processing Letters,*, vol. 4, no. 7, pp. 189–191, July, 1997.

[4] P. Sherwood and K. Zeger, "Error protection for progressive

image transmission over memoryless and fading channels," *IEEE Trans. on Communications,*, vol. 46.

[5] N. Thomos, N. V. Boulgouris, and M. G. Strintzis, "Wireless image transmission using turbo codes and optimal unequal error protection," *IEEE Trans. on Image Processing*, vol. 14, no. 11, pp. 1890–1901, Nov. 2005.

[6] A. Alatan, M. Zhao, and A. Akansu, "Unequal error protection of spiht encoded image bit streams," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 6, pp. 814–818, 2000.

[7] R. Hamzaoui, V. Stankovic, and Z. Xiong, "Optimized error protection of scalable image bit streams [advances in joint source-channel coding for images]," *IEEE Signal Processing Magazine*, vol. 22, no. 6, pp. 91–107, Nov. 2005.

[8] C. Creusere, "A new method of robust image compression based on the embedded zerotree wavelet algorithm," *IEEE Trans. Image Processing*, vol. 6, no. 10, pp. 1436–1442, Oct. 1997.

[9] J. Rogers and P. Cosman, "Wavelet zerotree image compression with packetization," *IEEE Signal Processing Letters*, vol. 5, no. 5, pp. 105–107, 1998.

[10] T. Kim, S. Choi, R. Van Dyck, and N. Bose, "Classified zerotree wavelet image coding and adaptive packetization for low-bit-rate transport," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 11, no. 9, pp. 1022–1034, Sep. 2001.

[11] S. Yang and T. Cheng, "Error-resilient spiht image coding," *Electronics Letters*, vol. 36, no. 3, pp. 208–210, 2000.

[12] D. Redmill and N. Kingsbury, "The erec: an error-resilient technique for coding variable-length blocks of data," *IEEE Transactions on Image Processing*, vol. 5, no. 4, pp. 565–574, 1996.

[13] W. Lam and A. Reibman, "Self-synchronizing variable-length codes for image transmission," in *Proc. ICASSP*, pp. 477–480, March, 1992.

[14] P. Cosman, J. Rogers, P. Sherwood, and K. Zeger, "Combined forward error control and packetized zerotree wavelet encoding for transmission of images over varying channels," *IEEE Trans. Image Processing*, vol. 9, no. 6, pp. 982–993, June, 2000.

[15] S. Cho and W. Pearlman, "Multilayered protection of embedded video bitstreams over binary symmetric and packet erasure channels," *Journal of Visual Communication and Image Representation*, vol. 16, no. 3, pp. 359–378, 2005.

[16] Y. Hu, W. A. Pearlman, and X. Li, "Progressive significance map and its application to error-resilient image transmission," *IEEE Trans. on Image Processing*, vol. 21, no. 7, pp. 3229–3238, 2012.