Macroscopic Multistage Image Compression for Robust Transmission over Noisy Channels

P. Greg Sherwood and Kenneth Zeger

University of California, San Diego
La Jolla, CA 92093-0407 USA

ABSTRACT

We propose a macroscopic multistage compression system to provide progressive and robust transmission of images across noisy channels with varying statistics. Each stage encodes the residual image of the previous stage. The choice of source coder and transmission rate at each stage are design parameters. The multistage structure allows the use of efficient unequal error protection channel coding and introduces source redundancy to enable graceful degradation under severe channel conditions. Both bit errors and packet losses are considered. Specific examples are provided to demonstrate the performance of the proposed method.

Keywords: robust image compression, packet erasure, multistage coding

1. INTRODUCTION

Modern communication systems experience channel conditions that change over time due to such things as mobile receivers and transmitters or varying levels of multiuser interference. Handheld wireless devices, for example, must operate in severe channel conditions with limited bandwidth and good power efficiency. It is thus a challenging problem to design image transmission algorithms for these channels that perform well in both good and bad channel conditions.

Typically, in channel coder design the goal is to match the channel coding to the channel, but this may not be feasible when the channel is unknown or varying or when there are multiple simultaneous channels as in a broadcast environment. One option is to allocate transmission rate to code for the worst-case channel conditions, but the cost is lower source coding performance on a clear channel. Alternatively, allocating transmission rate to code for better channels leads to poor performance when the channel is slightly worse, since the decoded error probability typically exhibits a fast transition to poor performance as the channel conditions become noisier than the design conditions.[1]

One method of achieving graceful degradation in a varying channel environment is to use more robust source coders in combination with channel coding. This strategy can be viewed as adding source redundancy instead of channel code redundancy. A number of methods have been developed in the past to improve performance on noisy channels by exploiting source redundancy. The source redundancy is generally either explicitly inserted or else is a byproduct of source coder imperfections. The source redundancy is most commonly used to maintain synchronization, allow smoothing algorithms to reduce large distortions, or detect channel decoding errors.[2–7] Unfortunately, the performance of most robust image coders lags significantly behind that of the best image compression algorithms designed for noiseless transmission.[8] Also, there is generally no convenient method for adjusting the level of source redundancy to better match channel conditions.

Source redundancy has also been used to modify the a priori bit probabilities in the channel decoding algorithm in Refs. 9–12. However, it is often not practical to accurately estimate the probabilities of source coder output sequences. Also in sequence decoding, there may be a significant delay before incorrect candidate sequences can be eliminated by the channel decoding algorithm (because they are improbable source sequences).

In this work, we propose a multistage coding structure for image compression that encodes entire images at each stage rather than the traditional multistage approach of coding relatively small vectors, so we call the method "macroscopic multistage coding" to emphasize the distinction. Each stage encodes the residual of the previous stage. The process of multistage encoding typically results in additional source redundancy in the end-to-end encoded

This work was supported in part by the National Science Foundation.
E-mail: {sherwood,zeger}@code.ucsd.edu

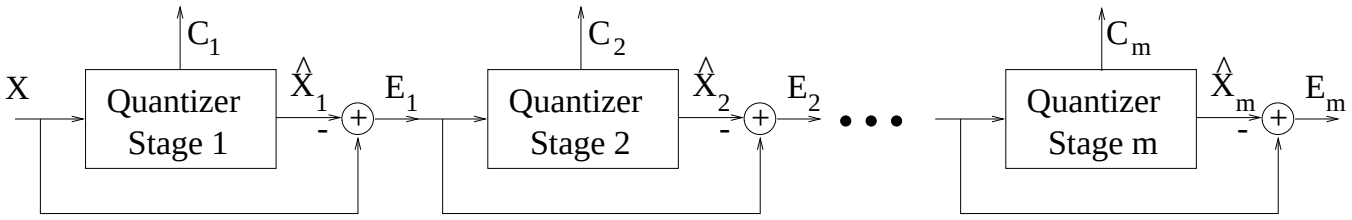

Figure 1. Structure of multistage (residual) source encoder.

bit stream, but the amount of additional redundancy is tunable. The additive structure of the multistage decoder effectively exploits the source redundancy to provide graceful degradation in cases of lost packets or uncorrected bit errors.

In addition, any non-progressive image compression algorithm can be made progressive (by stages), by using the algorithm at each stage in the multistage structure (although some noiseless channel performance is generally sacrificed). Whereas the traditional use of multistage coding is for complexity reduction, our purpose of multistage coding is specifically to improve performance over noisy channels and maintain progressivity. In fact, the stages in our proposed scheme can be high performance image compression schemes themselves, in contrast to plain vector quantization used in ordinary multistage coders. Also, the transmission rate allocated to each stage can be much greater than in more traditional multistage applications. In the following sections, we first give some background on multistage coding, and then describe two specific applications and present numerical results on some severe channel conditions.

2. MULTISTAGE CODING

The basic structure of a multistage (or "residual") coder is depicted in Fig. 1 and is characterized by multiple encoders arranged sequentially so that the quantization error signal from each stage is the input to the next stage. In source coding, this structure is used to reduce the encoder complexity and storage requirements for vector quantization (VQ).[13] For example, a multistage VQ (MSVQ) consisting of m stages with codebook size N_i at stage i, has search complexity (assuming greedy stage by stage encoding) and storage equal to $\sum_{i=1}^{m} N_i$ instead of $\prod_{i=1}^{m} N_i$ (for an equivalent full search vector quantizer). The reproduction vector is a sum of codevectors, one from each stage (i.e., $\hat{X} = \sum_{i=1}^{m} \hat{X}_i$). An example of a predictive image coder based on this structure is Ref. 14. In Ref. 15, design algorithms are presented for channel-matched tree-structured VQ (TSVQ) and MSVQ which improve performance when operating over noisy channels. The role of the multistage structure was complexity reduction and not robustness in that work.

We proposed to use at each stage in Fig. 1 a full-image compression algorithm. This leads to an image coder with the following properties:

1. The coder is progressive at stages.
2. The coder can survive the loss or corruption of one or several stages.
3. The coder performs well if no stages are lost.

The multistage structure can be applied not only to images but to other sources as well. Moreover, there is freedom to allocate the transmission rates in each stage and even to alter the encoding algorithm for each stage. In a more general variation of multistage coding, each stage can depend on the encoded bit streams of previous stages. In this paper, we do not specifically consider this variation, but we note that it offers the possibility of better source coding performance at the expense of increased error sensitivity.

One reason that multistage encoders provide robustness to channel noise is the insertion of known resynchronization points. The transmitted bit stream has multiple independently decodable components which allows all correctly received stages to be used in the reconstruction. As with most other progressive coders, the stages typically do not have equal influences on the decoded quality. Instead, importance essentially decreases monotonically with

stage number. This nonuniform importance allows more efficient channel coding through the use of unequal error protection (UEP).

A bit stream is said to be "progressive" if it can be decoded at more than one transmission rate to yield potentially coarser image qualities at lower transmission rates. One bit stream is said to be "embedded" in a second bit stream if the first bit stream is a prefix of the second bit stream. An image coder is "progressive" if the encoder output bit stream at every transmission rate is progressive. An image coder is said to be "embedded" if the encoder output bit stream for every transmission rate is embedded in the encoder output bit stream for any higher transmission rate. If an image coder is embedded, then it is clearly progressive.

When we refer to the "amount of progressivity" of an image coder, we qualitatively mean the number of different transmission rates that yield progressive bit streams. Similarly, the "amount of embeddedness" refers to the number of different transmission rates whose bit streams are embedded in those of higher transmission rates. Coders that are non-embedded typically produce different bit streams depending on the rate for which they are optimized. Finally we say an image coder has a "serial decoding requirement" if the decoder must decode the bit stream in a sequential and uninterrupted fashion for correct interpretation of the bits.

An interesting feature of our proposed multistage coder is that it simultaneously has progressive, embedded, and non-embedded coding characteristics. As noted previously, the multistage structure naturally induces progressivity whether or not the component coders have this property. Each stage has a constrained transmission rate (by design) regardless of whether the stage encoder is embedded or not. At the same time, the macroscopic multistage structure results in some degree of embedded encoding of the source independent of the type of component source coders. Only the coding in the final stage within the transmission rate constraint changes if the transmission rate does not correspond to the end of a stage.

While the multistage structure allows arbitrary source coders to be used at each stage, there is no guarantee that cascading a particular coder will result in good performance. In fact, if a good image coding algorithm is used in the initial stages, the input to later stages typically deviates statistically from that of natural images. Therefore using the same off-the-shelf image coding algorithm for each stage will not necessarily work well. In Sects. 4 and 5 , two specific examples of multistage coders are discussed.

3. PRELIMINARIES

In this section we carefully explain some terminology that will be useful in the subsequent discussions of the specific coders.

The familiar tree-structured dependency of wavelet coefficients is shown in Fig. 2. The figure shows a three stage wavelet decomposition of the 512x512 Lena image with the tree structure for one particular spatial region highlighted. Coefficients with large magnitudes have dark gray levels in the image, and each subband has been normalized so the relatively large coefficients in each subband will be visible. The dependency between the subband coefficients is clearly demonstrated by the appearance of recognizable elements of the original image in each subband. Hereafter, a **wavelet tree** will refer to the set of wavelet coefficients with a *root* coefficient in the lowest frequency subband together with its descendants from the other subbands as depicted in Fig. 2.

For the purpose of robust image coding, it is often important to partition an image into pieces prior to encoding. Each piece is treated like an independent image by the encoding algorithm. Partitioning an image can avoid localizing all of the distortions caused by losses in any one piece. A **subimage** refers to a piece (partition element) of an entire image which is coded independently. For the coders that follow, a subimage specifically consists of an ordered set of wavelet trees. Also, the partition of the original image into subimages will be called the **image partition**, and it refers to the partition of an ordered set throughout this paper.

The encoding of a subimage is a segmentation of the total bit stream similar to a stage in the multistage structure. The distinction is that an entire image is coded within a stage whereas a subimage is only a portion of the image and its encoding is always an element of a stage.

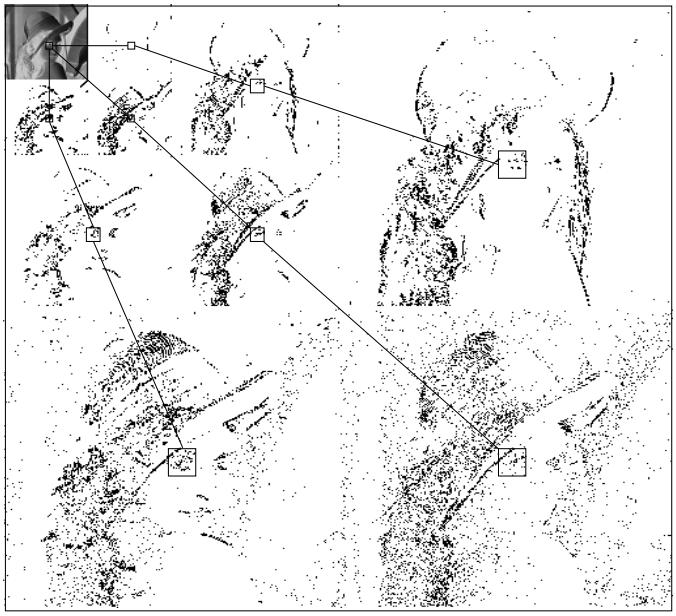

Figure 2. An example wavelet domain image demonstrating the tree-structured dependency of coefficients.

4. A MULTISTAGE ZEROTREE WAVELET CODER

This section describes a multistage coder based on zerotree wavelet coding. A number of high performance image coding algorithms have been proposed in recent years based on exploiting the tree-structured dependency among wavelet subband coefficients.[16–19] Shapiro's Embedded Zerotree Wavelet (EZW)[16] algorithm and the refined algorithm, Set Partitioning In Hierarchical Trees (SPIHT),[17] by Said and Pearlman provide excellent compression performance under noiseless conditions, but are very sensitive to channel errors. Under known and memoryless channel conditions, such as a binary symmetric channel (BSC), it has been shown that channel coding can effectively protect the bit stream and that the combined system has performance superior to most other known methods.[8,20] However, when channel conditions are worse than the design conditions, this method does not degrade very gracefully.[21] In the present paper, we attempt to improve the robustness of image transmission over channels with varying statistics.

One approach to improving the robustness of the EZW or SPIHT coders is to exploit the decreasing importance of the bit stream, as the rate increases, by using UEP channel coding. However, in practice, this results in little improvement due to the combination of the serial decoding requirement of the algorithm and the rapid transition in performance of channel codes as a function of the channel code rate (e.g., see Ref. 20 and Ref. 22). The multistage coding structure reduces the serial decoding requirement by only requiring an uninterrupted bit stream within a stage. The stages tend to have unequal importance (in terms of the decoded image quality) with the first stages being the most important. However, in this case UEP coding is more effective since it is possible to continue decoding beyond the first channel error. A multistage version of the SPIHT coder is discussed in the remainder of this section.

4.1. The Stage Schedule

We refer to a "stage schedule" as the ordered list of transmission rates assigned to the stages of a multistage coder. The rate of the last stage is usually left to absorb any remaining bits in the rate assignment procedure. We assume that the stage schedule is fixed and known at both the encoder and decoder. A possible direction of future research is to study the adaptation of the stage schedule based on the image and the channel conditions, in which case the schedule could be transmitted as overhead.

The maximum stage length (i.e., the number of packets or total rate in a stage) is constrained by the serial decoding requirement within a stage and the effective packet error rate, after the channel decoder, under the worst channel conditions of interest. If the probability of a packet error is p under these conditions, then there is little reason to create a stage much longer than $1/p$ packets since error-free packets beyond the first packet error are

discarded within a stage. While each additional stage can offer improvement in robustness under poor channel conditions, the performance typically decreases under good channel conditions. As the probability of packet error under the worst channel conditions drives the maximum stage length down, loss in source coding performance under good conditions leads to the necessity of finding methods to improve performance. The tradeoff between performance on good channels and on bad channels is determined by our choice of stage schedules, the source coders used at each stage, and the channel coding for each stage.

4.2. Improving Performance

While the SPIHT algorithm performs well on natural images it is not particularly suited for coding residual images. The residual image at the output of the first stage (coded by SPIHT) tends to have coefficients in lower frequency subbands with peak magnitudes about the same as those in higher frequency subbands. Also, the state information of SPIHT is lost, so extra bits must be spent at the start of each stage to describe the locations of the large coefficients. (In ordinary SPIHT this is implicit in the bit stream). One way to reduce the performance loss in using SPIHT at every stage of a multistage encoder, is to use fewer levels of wavelet decomposition in later stages. The motivation for this approach is that the coefficients in the coarser scales have generally been refined beyond their most significant bit and most of the tree-structured dependency has been exploited at these scales. Also the large coefficients at fine scales would not be as deeply nested in the tree-structure so fewer bits are needed to code their locations. This method is useful regardless of the stage length (i.e., the transmission rate of a stage), but it is not sufficient to completely remove the problem associated with very short stages.

A problem associated with using very short stages is that extra bits are spent coding coefficients that have already been refined in earlier stages since the state information is reset at the beginning of the stage. This is similar to using a repetition channel code where the code rate is too small (i.e., it is over-coded). An effective solution to this problem is to partition the image within a stage into subimages (see Sect. 3). This method allows the stage length to increase because serial decoding is only required within a subimage. The visual quality of the reconstructed image can be improved under packet loss conditions by ensuring the subimages consist of spatially diverse sets of wavelet trees. This way, the distortion due to packet loss will be distributed throughout the image. This goal is accomplished by choosing a good ordering of the image wavelet trees prior to partitioning into subimages. The coder mentioned in Sect. 5 uses a similar method except the image is partitioned into many more subimages (i.e., the number of subimages equals the number of packets). Also with the multistage SPIHT coder, the partitioning only occurs in later stages with fewer levels of wavelet decomposition and only in those cases where packet loss rates significantly limit stage lengths.

4.3. Overhead information

Some overhead information is required for proper decoding of the the multistage version of SPIHT beyond that required for the original version. First the starting threshold value needs to be sent for each stage or partition. In the worst case this represents about 5 bits per stage, but can be reduced by coding differentially from a predicted value which would probably be a static value based on the stage schedule. In the experimental results that follow, the full 5 bit encoding was used.

The decoder must also be able to identify the stage and intrastage position of each correctly received packet. The amount of explicit coding required for this depends on the details of the system and the packet loss mechanism. For situations where all packets are received and the only loss is due to channel decoding failures, no extra information is necessary since the correct location of each packet is evident by counting received packets. If some packets never reach the receiver (e.g., misrouting or buffer overflow in a wireline network), then each packet must include enough overhead bits to resolve the uncertainty. For example, if the probability of N or more consecutive packet erasures is acceptably low over the channel conditions of interest, then a packet index of $\lceil \log_2(N) \rceil$ bits suffices to determine the intrastage position of each received packet. If packets can arrive out of order, the maximum delay would affect the number of bits required to resolve the correct position. In some systems packet index information is included in the packet header along with routing information regardless of the packet payload, so explicit coding by the source coder would be reduced or eliminated in this case.

The experimental results in Sect. 4.4 focus on systems where packet losses are due to channel decoding failures, as can happen in wireless links. Therefore no additional information is necessary to identify the correct position of a received packet. Sect. 5.3 considers the case where packets are lost due to misrouting or buffer overflow, as in a wireline network.

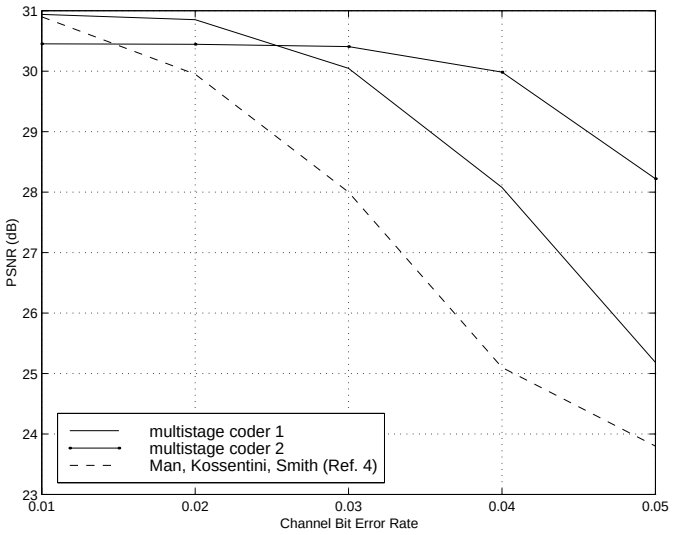

Figure 3. Performance results for the 512x512 Lena image at 0.25 bpp total transmission rate sent over a BSC.

4.4. Experimental Results

In this section we focus on transmission over a BSC with channel error rates between 0.01 and 0.05. The performance is measured by the mean decoded MSE, computed over at least 100 trials on a BSC of the given error rate. The channel conditions were selected to allow comparison with results in Ref. 4.

The output of each stage was protected with a rate-compatible punctured convolutional (RCPC)/cyclic redundancy code (CRC) concatenated channel code as described in Ref. 20. A different RCPC rate was selected for each stage to exploit the unequal importance of their output bit streams. Each RCPC code was based on a memory 6 mother code, and a 16-bit CRC was used for error detection and as part of the list decoding process.[20] The packets consisted of 220 information bits.

The multistage coders described in this section use the version of SPIHT with arithmetic coding. Experiments indicated that arithmetic coding provides about 0.2 dB better PSNR under good channel conditions. The gain associated with arithmetic coding is less than usual because the arithmetic coder did not have time to adapt the model during the short stages (or subimages). These short stages were necessary for robustness under the severe channel conditions. It might be possible to obtain better source coding performance if the models are initialized with more accurate probability distributions. Results for two coders are provided to demonstrate two possible tradeoffs between performance under the good channel conditions versus the performance under bad conditions.

The first coder consisted of 3 stages with the first stage length equal to 10 packets, the second stage length equal to 15 packets, and the third stage assigned the remaining rate. The number of levels of wavelet decomposition was also changed at each stage with 6 levels in the first stage, 5 levels in the second stage, and 4 levels in the third stage. Finally for stages 2 and 3 the image was partitioned prior to coding. For stage 2, the wavelet trees were divided uniformly into 3 sets, and each set was coded independently to produce 5 packets each. Similarly, in stage 3, the set of all wavelet trees was divided uniformly to give $\lceil$stage packets/4$\rceil$ partition elements with each coded to produce 4 packets (except perhaps the last partition element). The RCPC code rates for the three stages were 1/2, 4/7, and 2/3 respectively. The second multistage coder had essentially the same structure except the channel code rates were lower for the last two stages. The RCPC rates for coder 2 were 1/2, 8/15, and 4/7.

Figure 3 shows the performance of the two multistage coders as well as the results from Ref. 4 for comparison. We note that the results from Ref. 4 used different channel coding than the coders presented here (i.e., RCPC coding alone vs. RCPC/CRC), so those results would probably improve if the same RCPC/CRC were used. However, the comparison shows that both multistage methods are performing well. To demonstrate the visual quality, two example decoded images from multistage coder 2 are shown in Figs. 4(a)- 4(b) for channel bit error rates (BERs) 0.01 and 0.05.

(a) BER=0.01, PSNR = 30.42 dB. (b) BER=0.05. PSNR = 29.24 dB.

Figure 4. Example decoded images for multistage coder 2 for channel BER=0.01 and 0.05.

5. A PACKETIZED MULTISTAGE IMAGE CODER

This section describes a multistage version of the Packetized Zerotree Wavelet (PZW) coder presented in Ref. 23. The PZW coder modifies zerotree coding in order to be more robust to lost packets due to buffer overflow or misrouting (i.e., lost packets never reach the receiver). Specifically PZW partitions an image into subimages as explained in Sect. 3 and encodes each subimage with a zerotree algorithm so that the bits exactly fill one packet. The order of the wavelet trees is fixed according to a dither pattern matrix (selected to reduce the visual effect of lost packets) and the exact image partition is determined for each input image. Header information is included with each packet to allow independent decoding.

Our multistage modification of PZW, obtains improved robustness by distributing information about any spatial region into multiple packets (instead of just one) and by introducing unequal packet importance which allows improved performance by exploiting efficient erasure correction coding. As a side benefit, the multistage PZW has some degree of progressivity.

Each stage generally uses a different number of levels of wavelet decomposition similar to the multistage SPIHT coders in Sect. 4, with the number levels typically decreasing monotonically with stage number. As previously mentioned, changing the number of levels allows improved coding of the residual images by the zerotree algorithm. A random permutation is selected for each stage and fixed at the encoder and decoder to determine the order of wavelet trees for the stage. Random permutations at each stage sufficiently create spatially diverse subimages as well and reduce the chance that two spatial regions are in the same subimage at two different stages. The specific permutations selected slightly affect the compression performance of the algorithm, but experimental trials indicate that this affects the decoded PSNR by less than 0.1 dB for typical coding rates and 0.2-0.3 dB for short initial stages. The effect of the specific permutations also diminishes as more stages are used which is due, in part, to selecting good partitions at each stage.

We partition the set of all wavelet trees in an image, ordered according to the permutation, so the number of subimages equals the number of packets for that stage. Since the encoded subimage length is small and each stage is partitioned prior to encoding, selecting a good partition is more important to performance than with the multistage SPIHT coder. Because the partition is not fixed, each packet contains information about the trees contained within it. To reduce the dependency between the packets, the partition information is encoded so that the set of trees in a packet can be determined without information from other packets. This amounts to encoding the starting tree index which requires $\log_2(M)$ bits (where M is the number of wavelet trees) and the number of trees in the packet (usually 4-5 bits is sufficient). The next section considers the problem of finding a good partition.

5.1. Finding a Good Image Partition

As previously mentioned, the problem of finding a good partition of the image at each stage increases as the numbers of partitions and stages increase. The process of independently encoding subimages results in some variation in the rate allocated to individual wavelet trees compared to the case without image partitioning which usually results in a loss of performance. It is thus of interest to find the best partition of wavelet trees into subimages which are each encoded with a rate constraint corresponding to the packet length. Typically one would minimize the reconstructed MSE, but other distortion measures can also be used.

This problem can be solved with dynamic programming[24] by noting that given n wavelet trees are assigned to the first m subimages, the best partition of the remaining trees is independent of the specific initial partition. Therefore, the cumulative number of trees assigned to subimages can serve as the state in a trellis diagram, and the subimage (or packet in this case) number can serve as the trellis stage index. Typically, there are further constraints imposed which limit the search space such as the minimum and maximum number of trees per subimage.

The distortion function for each subimage is a complicated function of the packet size and the set of wavelet trees in this case, but the values can be determined by running the encoding algorithm. Partial distortion values associated with a particular set of wavelet trees are used several times during the dynamic programming algorithm, so complexity can be reduced by storing these values in a table for later use. Also, simpler but coarser approximations to the desired metric can be used to trade off complexity and performance within the same dynamic programming framework.

Our multistage PZW coder uses this dynamic programming approach to optimize the partition selected at each stage. Experiments indicate that MSE optimized partitions in later stages can recover some of the performance loss in earlier stages caused by partitioning.

5.2. Reducing Distortion at the Decoder

Packet erasures cause distortions in the spatial locations corresponding to the wavelet trees within the discarded packets. In PZW,[23] the decoded distortion is reduced by interpolating missing wavelet tree roots from neighboring coefficients in the low frequency subband.

Figures 5(a) and 5(b) show examples of the typical distortions that result due to missing tree information. Missing trees at coarser scales cause distortions over larger spatial regions and interpolation is not effective. At finer scales, the missing trees cause large distortions in small spatial areas and the distortion can typically be reduced by interpolation. The third image in Fig. 5(c) shows a smoothed version of the image in Fig. 5(b), where the improvement in quality can easily be seen, and numerically the PSNR is improved from 21.21 dB to 25.53 dB.

Interpolation is really only effective in the multistage PZW for reducing distortions caused by lost packets from the first stage. Also if the missing wavelet tree roots are from a coarse scale (i.e., many transform levels), then interpolating the value from neighbors does not work better than using the low frequency subband mean (transmitted separately). However, using an initial stage with many levels of decomposition can be useful for robustness when UEP channel coding is feasible. Also if a short first stage is desirable for the purpose of increasing progressivity, then a coarse scale can be useful for achieving good source coding performance within that stage.

5.3. Experimental Results

To test the performance of the multistage version of PZW, the channel was selected to match the conditions in Ref. 23. Packets were erased independently with a probability corresponding to the given packet loss rate, the packet size was 384 bits, and the total transmission rate was 0.209 bpp.

In this channel model, lost packets do not arrive at the decoder, and it is assumed that the source coder must provide explicit coding to determine a packet's position in the transmitted sequence if necessary. Therefore, overhead bits are sent with each packet to identify the stage number, which consisted of one bit (i.e., two stages) for the coders in this section.

The PZW algorithm distributes the image information as uniformly as possible among the packets so the maximum loss associated with a packet erasure is minimized. For the multistage coder, the packets have unequal importance due to the progressive nature of the encoding structure. Therefore explicit erasure coding is used to protect the important first stage packets. Specifically Reed-Solomon (RS) codes are used across packets (similar to the method in Ref.[25]) so that erased packets can be recovered if the capabilities of the code are not exceeded. This method

(a) Effect of missing trees at coarse scale.

(b) Effect of missing trees at finer scale.

(c) Same image as (b) after smoothing.

Figure 5. Example images demonstrating the effect of erased trees on decoded image quality.

allows more levels of wavelet decomposition in the first stage for better source coding efficiency while alleviating the problem that smoothing algorithms would be ineffective (as mentioned in Sect. 5.2).

In order to correctly decode the RS code on the first stage packets, the positions of the missing packets must be identified. Therefore a packet index needs to be sent as overhead so the position within the stage can be identified for each received packet. The packet index bits and the starting tree number contained within each packet for partition information are somewhat redundant. The starting tree can be predicted from the packet index and stage schedule and only the error needs to be coded. Also, the starting tree index of the first packet is redundant since it is always zero, and only the starting tree index is necessary in the final stage packet since the remainder of the trees must be included.

The multistage encoder uses a straight-forward encoding of header bits without the index prediction. It is composed of two stages with a first stage length of 12 packets and 6 levels of wavelet decomposition followed by a second stage using 4 levels of decomposition. The 12 first stage packets are protected by a systematic (20,12) RS code which was selected to provide good performance at the highest erasure rate of interest, 0.2, while not sacrificing error-free performance too much. Using this RS code at the highest erasure rate, the first stage packets are available for decoding more than 99% of the time. The overhead in each packet for this coder includes: 1 bit for the stage number, 5 bits for packet index (only in first stage), 5 bits for starting threshold value, 6 bits first stage and 10 bits second stage for the starting tree index, 4 bits first stage and 5 bits second stage for the tree count. Table 1 shows performance results for the multistage coder and PZW over a range of packet erasure rates. The values in the table are mean decoded MSE converted to PSNR, and for the multistage coder, they are representative of the range of values that occurs as different permutations are selected for each stage's tree order (i.e., the values in the table are within about 0.2 dB of the performance of a randomly selected permutation set).

As the numerical results show, the multistage coder is robust even under high loss conditions while only sacrificing about 0.4 dB in PSNR under lossless channel conditions. These results give an indication of the performance possible with the multistage version of PZW. Implementing more efficient encoding of the packet overhead can yield an improvement of about 0.3 dB in PSNR. Also, the coder can be adjusted for a different tradeoff between lossless and high loss channel conditions. Example decoded images are provided in Figs. 6(a)-6(c) for three erasure rates to demonstrate the visual quality.

6. CONCLUSION

A macroscopic multistage encoding structure has been proposed as a method of creating robust source coders. An important benefit is that it allows the amount of source redundancy to be tuned to some degree. As a side benefit,

Coder	No erasures	1% erasures	10% erasures	20% erasures
PZW[23]	32.19	31.33	26.29	24.63
Multistage PZW	31.8	31.6	30.2	28.4

Table 1. Performance results for 512x512 Lena sent over a packet erasure channel. Values are PSNR (dB) converted from mean MSE.

(a) No erasures. PSNR = 31.96 dB. (b) 10% erasures. PSNR = 30.50 dB. (c) 20% erasures. PSNR = 28.45 dB.

Figure 6. Example decoded images for multistage PZW under three erasure conditions.

the structure results in some progressivity even if the underlying coders are not progressive. The method was shown to work well under severe channel conditions with little sacrifice in performance under good channel conditions.

REFERENCES

1. G. Zemor and G. D. Cohen, "The threshold probability of a code," *IEEE Transactions on Information Theory* **41**, pp. 469–477, Mar. 1995.
2. J. Cai, C. W. Chen, and Z. Sun, "Error resilient image coding with rate-compatible punctured convolutional codes," in *Proc. IEEE Int. Symp. Circuits and Systems, 1998*, vol. 4, pp. 110–113, 1998.
3. H. Li and C. W. Chen, "Joint source and channel optimized TCQ with layered transmission and RCPC," in *Proc. ICIP 98*, 1998.
4. H. Man, F. Kossentini, and M. J. Smith, "A class of EZW image coders for noisy channels," in *Proceedings ICIP 97*, vol. 3, pp. 90–93, 1997.
5. A. E. Mohr, E. A. Riskin, and R. E. Ladner, "Recovering from bit errors in scalar-quantized discrete wavelet transformed images," in *Proc. ICIP 98*, 1998.
6. P. G. Sherwood and K. Zeger, "Error protection of wavelet coded images using residual source redundancy," in *Thirty-First Asilomar Conference on Signals, Systems and Computers*, vol. 2, pp. 980–984, 1997.
7. I. Kozintsev, J. Chou, and K. Ramchandran, "Image transmission using arithmetic coding based continuous error detection," in *Proceedings DCC '98. Data Compression Conference*, J. Storer and M. Cohn, eds., pp. 339–348, 1998.
8. P. G. Sherwood and K. Zeger, "Progressive image coding for noisy channels," *IEEE Sig. Proc. Letters* **4**, pp. 189–191, July 1997.
9. K. Sayood and J. C. Borkenhagen, "Use of residual redundancy in the design of joint source/channel coders," *IEEE Trans. on Comm.* **39**, pp. 838–846, June 1991.
10. K. Sayood, F. Liu, and J. D. Gibson, "A constrained joint source/channel coder design," *IEEE Journal on Selected Areas in Communications* **12**, pp. 1584–1593, Dec. 1994.

11. J. Hagenauer, "Source-controlled channel decoding," *IEEE Trans. on Comm.* **43**, pp. 2449–2457, Sept. 1995.
12. S. Emami and S. L. Miller, "DPCM picture transmission over noisy channels with the aid of markov model," *IEEE Transactions on Image Processing* **4**, pp. 1473–1481, Nov. 1995.
13. A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*, Kluwer Academic Publishers, 1992.
14. S. A. Rizvi and N. M. Nasrabadi, "Predictive residual vector quantization," *IEEE Transactions on Image Processing* **4**, pp. 1482–1495, Nov. 1995.
15. N. Phamdo, N. Farvardin, and T. Moriya, "A unified approach to tree-structured and multistage vector quantization for noisy channels," *IEEE Transactions on Information Theory* **39**, pp. 835–850, May 1993.
16. J. M. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients," *IEEE Trans. on Sig. Proc.* **41**, pp. 3445–3462, Dec. 1993.
17. A. Said and W. A. Pearlman, "A new, fast, and efficient image codec based on set partitioning in hierarchical trees," *IEEE Trans. on CSVT* **6**, pp. 243–250, June 1996.
18. Z. Xiong, K. Ramchandran, and M. T. Orchard, "Space-frequency quantization for wavelet image coding," *IEEE Transactions on Image Processing* **6**, pp. 677–693, May 1997.
19. E. Ordentlich, M. Weinberger, and G. Seroussi, "A low-complexity modeling approach for embedded coding of wavelet coefficients," in *Proceedings DCC '98. Data Compression Conference*, J. Storer and M. Cohn, eds., pp. 408–417, 1998.
20. P. G. Sherwood and K. Zeger, "Progressive image coding on noisy channels," in *Proceedings DCC '97. Data Compression Conference*, J. A. Storer and M. Cohn, eds., pp. 72–81, 1997.
21. P. C. Cosman, J. K. Rogers, P. G. Sherwood, and K. Zeger, "Image transmission over channels with bit errors and packet erasures," in *Thirty-Second Asilomar Conference on Signals, Systems and Computers*, 1998.
22. J. Lu, A. Nosratinia, and B. Aazhang, "Progressive source-channel coding of images over bursty error channels," in *Proc. ICIP 98*, 1998.
23. J. Rogers and P. Cosman, "Wavelet zerotree image compression with packetization," *IEEE Sig. Proc. Letters* **5**, pp. 105–107, May 1998.
24. D. P. Bertsekas, *Dynamic Programming: Deterministic and Stochastic Models*, Prentice Hall, 1987.
25. P. G. Sherwood and K. Zeger, "Error protection for progressive image transmission over memoryless and fading channels," in *Proc. ICIP 98*, 1998.