Apart from the encoded data, the subband variances need to be transmitted as side (overhead) information. In addition, for System A the predictor coefficients need to be transmitted while, in System B, the variances of the DCT coefficients and the mean of the dc coefficient need to be transmitted. Note that if the variance information is available in the receiver, the bit allocation procedure can be repeated there and hence no additional information for the parameters of the (UTQ, HC) pairs is necessary.

Assuming that we need two bytes (16 bits) for each real-valued parameter, 36 bytes (16 subbands and two correlation coefficients) need to be transmitted for System A. This corresponds to 0.004 b/p for a 256 × 256 image and 0.001 b/p for a 512 × 512 image. In System B, in addition to the variances of the subbands, the variances of the DCT coefficients and the mean of the dc coefficient need to be transmitted. Therefore, for an $L \times L$ blocksize, the side information is $2(L^2 + 16)$ bytes. For the chosen blocksize of 4 × 4 this amounts to 0.008 b/p and 0.002 b/p for 256 × 256 and 512 × 512 images, respectively. Therefore, for all cases of interest the amount of size information is less than 0.01 b/p—a negligible amount.

V. Transmission Error Effects

In Systems A and B, variable-length coding is used extensively. It is well-known that due to the sequential nature of decoding such codes, channel errors could result in the loss of synchronization and, hence, severe degradation in system performance. Furthermore, predictive coding (used in System A and the W–O scheme) is known to suffer from channel error propagation problems. Finally, in 2-D DCT coding (used in System B), channel errors propagate throughout the block. These facts indicate that the subband coding schemes studied in Section IV may suffer serious difficulties in the presence of transmission (or storage) noise. Of course, due to the extensive use of variable-length coding in Systems A and B, one would expect a greater sensitivity to channel noise in Systems A and B as compared to the W–O scheme.[7] In this section, we will present simulation results for the performance of Systems A and B as well as the W–O scheme in the presence of channel noise.

To prevent the infinite propagation of decoding errors, we have packetized the codeword sequences before transmission. In what follows, we describe the details of the packetization scheme.

A. Packetization Scheme

The main motivation in packetization of codewords is to confine the propagation of channel noise to within a packet. To do so, we must make certain that our packets contain information about a fixed (or known in the receiver) number of pixels so that packet-to-packet propagation of the Huffman decoding error is prevented. Since the pixels are encoded by variable-length codes, the packets cannot be of fixed length. However, in devising our packetization scheme we will try to keep the *average* packet length fixed so that fair comparisons can be made between different systems. It is important to remember that the severity of error propagation is directly related to the packet length.

In the packetization scheme adopted in this work, the packets consist of two parts: i) a *length indicator* indicating the length of the information portion (in bits); and ii) the *information portion* consisting of a sequence of binary codewords. While the length of the length indicator is fixed, that of the information portion of the packet could vary—hence resulting in *variable-length packets*. Furthermore, all codewords transmitted in a packet belong to the same subband.[8] To be more precise, let us consider a packet used for encoding the *i*th subband. Let us suppose that the pixels in this subband are encoded by means of HC's of order n_i at a design bit rate of r_i. Suppose the average length of the information portion of the packet is l_p bits. Then, the number of codewords in this packet is given by:

$$n_{w,i} = \left\lceil \frac{l_p}{r_i n_i} \right\rceil \tag{6}$$

where $\lceil x \rceil$ is used to denote the smallest integer greater than or equal to x. Because l_p is fixed and r_i and n_i are known from the results of the bit allocation procedure, $n_{w,i}$ can be determined in the receiver side. Therefore, in encoding the *i*th subband, each packet contains the information for $n_{p,i} = n_i n_{w,i}$ pixels.[9]

As for the decoding process, the following rules are applied.

1) Decoding of codewords in a packet starts and ends in that packet so that the Huffman decoding error does not propagate beyond the packet boundaries.

2) Decoding of codewords in a packet associated with the *i*th subband terminates when one of the following three conditions is met.

a) $n_{w,i}$ codewords are decoded.

b) The decoding process reaches the end of a packet (known from the length indicator).

c) A bit string which cannot be decoded is encountered. In this case, decoding is stopped immediately, and all pixels that cannot be decoded are reconstructed as zero.

In what follows, we will present simulation results for the performance of the three schemes considered in Section IV when the channel is noisy.

[7] The W–O scheme also uses variable-length codes for encoding some subbands (see [2]); however, the LFS is encoded by fixed-length codes. Since LFS has the highest variance among all subbands, the channel noise effects in the LFS should result in the most dramatic degradations in system performance.

[8] In System B, each of the transform coefficients is treated as a subband. Hence, each packet contains codewords from only one transform coefficient.

[9] All packets used for encoding the *i*th subband contain $n_{p,i}$ pixels except possibly the last one which contains only the remaining pixels.

TABLE VI
PSNR PERFORMANCE RESULTS (IN dB) FOR "LENA"

	Design Bit Rate	0.25 BPP				0.5 BPP				1.0 BPP			
System	Channel BER	1×10^{-2}	1×10^{-3}	1×10^{-4}	1×10^{-5}	1×10^{-2}	1×10^{-3}	1×10^{-4}	1×10^{-5}	1×10^{-2}	1×10^{-3}	1×10^{-4}	1×10^{-5}
System A	AVE-PSNR	9.86	14.24	23.21	30.64	7.46	13.17	23.73	31.89	7.43	12.83	23.92	34.59
	MAX-PSNR	10.69	16.23	31.99	32.00	8.27	15.86	34.66	35.16	8.64	16.72	36.57	38.55
	MIN-PSNR	9.17	11.82	16.48	20.58	6.97	10.29	18.27	21.37	6.65	9.77	17.81	20.29
	STD-PSNR	0.40	1.00	3.08	3.02	0.32	1.16	3.41	4.31	0.34	1.48	4.45	4.96
System B (4 × 4)	AVE-PSNR	14.33	21.12	28.40	31.54	13.90	20.32	29.81	34.77	14.03	20.65	31.01	37.49
	MAX-PSNR	15.40	24.07	31.93	32.19	14.94	24.56	35.04	35.32	15.32	23.82	35.74	38.53
	MIN-PSNR	13.42	18.52	23.55	24.53	13.12	17.33	23.58	28.18	13.03	18.59	23.23	27.88
	STD-PSNR	0.54	1.50	2.49	1.57	0.45	1.62	2.97	1.44	0.51	1.45	3.05	1.88
W-O	AVE-PSNR	22.93	27.62	28.54	28.64	23.08	28.52	29.77	29.93	21.92	29.81	33.33	33.94
	MAX-PSNR	24.09	28.03	28.65	28.65	24.38	29.07	29.94	29.94	23.71	31.08	33.94	34.01
	MIN-PSNR	21.86	27.10	28.36	28.54	20.88	27.29	29.57	29.78	20.16	28.14	32.11	33.12
	STD-PSNR	0.51	0.22	0.06	0.04	0.60	0.32	0.08	0.04	0.72	0.65	0.38	0.17

B. Simulation Results over Noisy Channels

We assume that the channel is a memoryless binary symmetric channel (BSC) with a bit error rate (BER) of P_e. For our simulations in this section, we have considered $P_e = 10^{-2}, 10^{-3}, 10^{-4}$, and 10^{-5}. To make certain that our results are meaningful, for each encoding scheme, bit rate, and channel BER, we have repeated our simulations 50 times and computed the average PSNR (AVE-PSNR), maximum PSNR (MAX-PSNR), minimum PSNR (MIN-PSNR), and standard deviation of the PSNR (STD-PSNR). Simulations are carried out for "Lena" encoded by Systems A and B and the W-O scheme[10] at design bit rates of 0.25, 0.5, and 1 b/p. The simulation results are summarized in Table VI. A few important observations about these results are in order.

1) System A is extremely sensitive to channel errors. Also, the STD-PSNR of System A is fairly large (especially for low BER's) implying that, even at low-channel BER's there is a possibility of severe performance degradation (e.g., ≈ 14 dB difference between AVE-PSNR and MIN-PSNR for System A at 1 b/p with $P_e = 10^{-5}$).

2) System B is also very sensitive to channel errors. However, it performs considerably better than System A. In all cases considered, the MIN-PSNR of System B was significantly larger (4–8 dB) than that of System A; the AVE-PSNR of System B was also larger than that of System A, especially for larger values of the channel BER. Finally, the STD-PSNR of System B is smaller than that of System A.

3) The W-O scheme exhibits the highest degree of robustness in the presence of channel noise. In most cases, the effect of channel noise is negligible for $P_e < 10^{-3}$. Also, contrary to our observation for Systems A and B, the STD-PSNR in this case is very small. In almost all cases, the W-O system performs better than System B with the exception of a few cases where the channel noise is very small ($P_e = 10^{-5}$).

In Fig. 9, an example of reconstructed images from the three systems is presented for an encoding rate of 0.5 b/p and $P_e = 10^{-3}$; these images correspond to those cases in our simulation which result in minimum PSNR. Clearly, the subjective performance of the three systems closely follows the trend suggested by the PSNR results of Table VI.

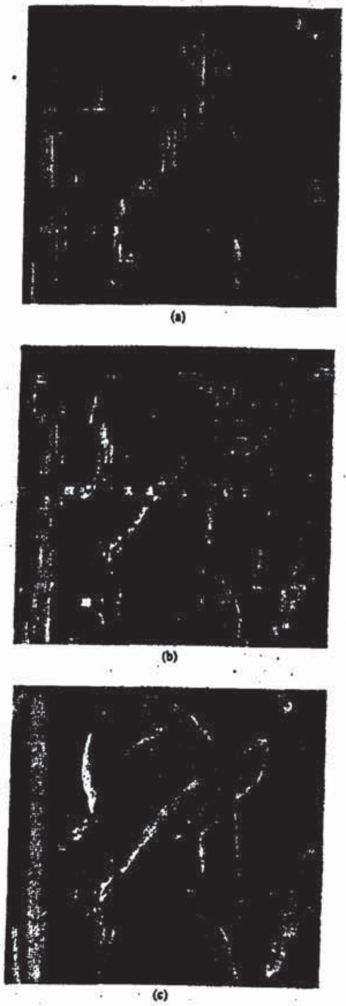
The results of Table VI (also supported by Fig. 9) suggest that Systems A and B, despite their superior performance for noiseless channels, exhibit an unacceptable level of sensitivity to channel errors and hence should not be used over noisy channels (at least over the range of channel BER's considered here). In the next section, we will describe a combined source/channel coding methodology to reduce this severe sensitivity to channel noise.

VI. COMBINED SOURCE/CHANNEL CODING

Systems A and B exhibit a high degree of sensitivity to channel noise because they have been designed to minimize the source coding distortion assuming a noiseless channel. It is a well-known fact that, in general, the more efficient the source coding scheme is, the more sensitive it will be to channel noise unless some corrective measures are taken. Specifically, it is shown in [15] for zero-memory quantizers and in [16] and [17] for predictive coding and transform coding of images that, in the presence of channel noise, increasing the accuracy of a source encoder could result in an overall performance degradation.

One possible method of mitigating the channel error effects is use of error control coding. In this manner, of all bits used for encoding the image, some will be used in source coding while the rest will be kept to provide protection against channel noise. In [16] and [17], an approach in which specific source encoders and channel encoders are combined is considered; in this approach, the rates of the source code and channel code are adjusted so as to minimize the MSE. In [15], an approach for chan-

[10] In all of our simulations for noisy channels, the average packet length is 1024 bits. This is not an optimized length but, we feel, should provide an appropriate tradeoff between the channel error effects and the increase of side information.



(a)

(b)

(c)

Fig. 9. Reconstructed "LENA" at channel BER of 10^{-3}. (a) System A (0.45 b/p. 10.29 dB); (b) System B (0.45 b/p. 17.33 dB); (c) W-O (0.51 b/p. 27.29 dB).

nel-optimized quantization is developed in which source coding accuracy is traded for reduced sensitivity to channel noise.

In this paper, we will consider an approach similar to that of [16] and [17]. In our subband coding systems, an important question is how to distribute the bits among the source coding and channel coding operations for the different subbands so as to minimize the overall distortion caused by quantization noise and channel noise. The main difficulty in doing this stems from the fact that variable-length codes are used in Systems A and B. In this case, the analysis of channel error effects and its impact on the overall distortion is a formidable task (if not impossible).

Another important problem in our systems is that of bit allocation among the different subbands. Clearly, the bit allocation used under the noiseless channel assumption need not be optimal for noisy channels. To be able to determine the optimal bit allocation, we need to be able to determine the distortion-rate performance of the UTQ's followed by a HC and an error correcting code (ECC). Again, due to the inherent problems of packetized variable-length codes, the analytical computation of these distortion-rate performance results is not possible.

In what follows, we will describe a *simulation-based* procedure to determine the best (UTQ, HC, ECC) triple for encoding a memoryless source over a BSC at a given encoding rate. This procedure will lead to the determination of the distortion-rate performance functions that we need for optimal bit allocation among subbands.

A. Selection of (UTQ, HC, ECC) Triple and Bit Allocation

For the discussion in this subsection, we will assume that the source is memoryless with a distribution corresponding to the GGD with parameter α and that the channel is a BSC with a BER given by P_e. Let us suppose, for the time being, that the ECC is to be selected from a prescribed family of ECC's. We will specify this family in the next subsection and provide justification for this choice.

For the given source and channel, consider a (UTQ, HC) pair (as selected in Section IV) with an average bit rate of r_s followed by an ECC with r_c and let $\tilde{d}(r_s, r_c; \alpha, P_e)$ denote the MSE incurred in encoding and transmission of the source. Since the analytical computation of $\tilde{d}$ is impossible, we have resorted to simulation[11] to determine its value for selected choices of r_s, r_c, α, and P_e. Notice that the overall encoding rate is given by $r = r_s/r_c$.

Now consider a fixed encoding rate r. Among the available pairs of (r_s, r_c), there may be several that result in the encoding rate r. Let us denote by (r_s^*, r_c^*) the pair that minimizes $\tilde{d}(r_s, r_c; \alpha, P_e)$; denote this minimum distortion by $d_c(r; \alpha, P_e)$. In other words,

$$d_c(r; \alpha, P_e) = \min_{(r_s, r_c): r_s/r_c = r} \tilde{d}(r_s, r_c; \alpha, P_e). \qquad (7)$$

The function $d_c(r; \alpha, P_e)$ determines the distortion-rate performance of the encoding scheme used for a source with parameter α and a BSC with BER P_e, after the appropriate selection of the (UTQ, HC, ECC) triple is made. Notice that, for a fixed encoding rate r, identifying the

[11] As before, to obtain the MSE we have averaged our simulation results over 50 runs.

TABLE VI
PSNR PERFORMANCE RESULTS (IN dB) FOR "LENA"

	Design Bit Rate	0.25 BPP				0.5 BPP				1.0 BPP			
System	Channel BER	1×10^{-2}	1×10^{-3}	1×10^{-4}	1×10^{-5}	1×10^{-2}	1×10^{-3}	1×10^{-4}	1×10^{-5}	1×10^{-2}	1×10^{-3}	1×10^{-4}	1×10^{-5}
System A	AVE-PSNR	9.86	14.24	23.21	30.64	7.46	13.17	23.73	31.89	7.43	12.83	23.92	34.59
	MAX-PSNR	10.69	16.23	31.99	32.00	8.27	15.86	34.66	35.16	8.64	16.72	36.57	38.55
	MIN-PSNR	9.17	11.82	16.48	20.58	6.97	10.29	18.27	21.37	6.65	9.77	17.81	20.29
	STD-PSNR	0.40	1.00	3.08	3.02	0.32	1.16	3.41	4.31	0.34	1.48	4.45	4.96
System B (4 × 4)	AVE-PSNR	14.33	21.12	28.40	31.54	13.90	20.32	29.81	34.77	14.03	20.65	31.01	37.49
	MAX-PSNR	15.40	24.07	31.93	32.19	14.94	24.56	35.04	35.32	15.32	23.82	35.74	38.53
	MIN-PSNR	13.42	18.52	23.55	24.53	13.12	17.33	23.58	28.18	13.03	18.59	23.23	27.88
	STD-PSNR	0.54	1.50	2.49	1.57	0.45	1.62	2.97	1.44	0.51	1.45	3.05	1.88
W–O	AVE-PSNR	22.93	27.62	28.54	28.64	23.08	28.52	29.77	29.93	21.92	29.81	33.33	33.94
	MAX-PSNR	24.09	28.03	28.65	28.65	24.38	29.07	29.94	29.94	23.71	31.08	33.94	34.01
	MIN-PSNR	21.86	27.10	28.36	28.54	20.88	27.29	29.57	29.78	20.16	28.14	32.11	33.12
	STD-PSNR	0.51	0.22	0.06	0.04	0.60	0.32	0.08	0.04	0.72	0.65	0.38	0.17

B. Simulation Results over Noisy Channels

We assume that the channel is a memoryless binary symmetric channel (BSC) with a bit error rate (BER) of P_e. For our simulations in this section, we have considered $P_e = 10^{-2}$, 10^{-3}, 10^{-4}, and 10^{-5}. To make certain that our results are meaningful, for each encoding scheme, bit rate, and channel BER, we have repeated our simulations 50 times and computed the average PSNR (AVE-PSNR), maximum PSNR (MAX-PSNR), minimum PSNR (MIN-PSNR), and standard deviation of the PSNR (STD-PSNR). Simulations are carried out for "Lena" encoded by Systems A and B and the W–O scheme[10] at design bit rates of 0.25, 0.5, and 1 b/p. The simulation results are summarized in Table VI. A few important observations about these results are in order.

1) System A is extremely sensitive to channel errors. Also, the STD-PSNR of System A is fairly large (especially for low BER's) implying that, even at low-channel BER's there is a possibility of severe performance degradation (e.g., ≈ 14 dB difference between AVE-PSNR and MIN-PSNR for System A at 1 b/p with $P_e = 10^{-5}$).

2) System B is also very sensitive to channel errors. However, it performs considerably better than System A. In all cases considered, the MIN-PSNR of System B was significantly larger (4–8 dB) than that of System A; the AVE-PSNR of System B was also larger than that of System A, especially for larger values of the channel BER. Finally, the STD-PSNR of System B is smaller than that of System A.

3) The W–O scheme exhibits the highest degree of robustness in the presence of channel noise. In most cases, the effect of channel noise is negligible for $P_e < 10^{-3}$. Also, contrary to our observation for Systems A and B, the STD-PSNR in this case is very small. In almost all cases, the W–O system performs better than System B with the exception of a few cases where the channel noise is very small ($P_e = 10^{-5}$).

In Fig. 9, an example of reconstructed images from the three systems is presented for an encoding rate of 0.5 b/p and $P_e = 10^{-3}$; these images correspond to those cases in our simulation which result in minimum PSNR. Clearly, the subjective performance of the three systems closely follows the trend suggested by the PSNR results of Table VI.

The results of Table VI (also supported by Fig. 9) suggest that Systems A and B, despite their superior performance for noiseless channels, exhibit an unacceptable level of sensitivity to channel errors and hence should not be used over noisy channels (at least over the range of channel BER's considered here). In the next section, we will describe a combined source/channel coding methodology to reduce this severe sensitivity to channel noise.

VI. Combined Source/Channel Coding

Systems A and B exhibit a high degree of sensitivity to channel noise because they have been designed to minimize the source coding distortion assuming a noiseless channel. It is a well-known fact that, in general, the more efficient the source coding scheme is, the more sensitive it will be to channel noise unless some corrective measures are taken. Specifically, it is shown in [15] for zero-memory quantizers and in [16] and [17] for predictive coding and transform coding of images that, in the presence of channel noise, increasing the accuracy of a source encoder could result in an overall performance degradation.

One possible method of mitigating the channel error effects is use of error control coding. In this manner, of all bits used for encoding the image, some will be used in source coding while the rest will be kept to provide protection against channel noise. In [16] and [17], an approach in which specific source encoders and channel encoders are combined is considered; in this approach, the rates of the source code and channel code are adjusted so as to minimize the MSE. In [15], an approach for chan-

[10] In all of our simulations for noisy channels, the average packet length is 1024 bits. This is not an optimized length but, we feel, should provide an appropriate tradeoff between the channel error effects and the increase of side information.

pair (r_s^*, r_c^*) is equivalent to determining the optimal balance between the source coding accuracy and the channel error protection.

Having determined the $\hat{d}(r_s, r_c; \alpha, P_e)$ functions by simulation, we have computed the functions $d_c(r; \alpha, P_e)$ for values of α as in Section IV, $P_e = 10^{-2}$, 10^{-3}, and 10^{-4} and a finite number of values of r.[12] An example of the function $d_c(r; \alpha, P_e)$ is provided in Fig. 10 for $\alpha = 0.6$ and three different values of P_e. In this figure, different symbols are used to determine the rate of the optimum channel code used. Also, for comparison purposes, the distortion-rate performance of the (UTQ, HC) pairs of Section IV obtained for a noiseless channel is also included in this figure (dotted curve). The deviation between these performance curves and the one for the noiseless channel is merely the result of the channel noise. Obviously, the deviation is wider for more noisy channels.

Once the channel-optimized distortion-rate performances are determined, the bit allocation procedure of Section IV can be used in a similar manner to obtain the optimum bit allocation among the subbands.

Before we present the simulation results for this combined source/channel coding scheme, in what follows we will describe the class of ECC's we have used in our systems.

B. Error Correction Coding Scheme

The ECC used in our system is a specific form of convolutional codes known as the *rate-compatible punctured convolutional* (RCPC) code. The RCPC code was introduced by Hagenauer [18] as an extension of the punctured convolutional code which was originally introduced by Cain *et al.* [19] mainly for the purpose of obtaining simpler Viterbi decoding for rate $K/N (K \neq 1)$ codes. The main advantage of the RCPC codes is that its rate (and, hence, the error correction capability) can be easily changed by varying the number of punctured bits in the puncturing matrix; therefore, with the same hardware, a variety of channel coding rates can be obtained. This is a desirable characteristic in our system as we wish to vary the rate of the ECC for each subband so as to obtain the best balance between the source coding rate and the channel coding rate. We should mention that this idea was first used in subband coding of speech [20] for adapting the degree of error protection to the error sensitivity of different coder bit streams.

The RCPC code is defined by a *generator tap matrix* of a convolutional code with the constraint length L_c:

$$g = \begin{array}{c} \uparrow \\ N_c \\ \downarrow \end{array} \overset{\leftarrow L_c \rightarrow}{\left(\; g_{ik} \; \right)} \tag{8a}$$

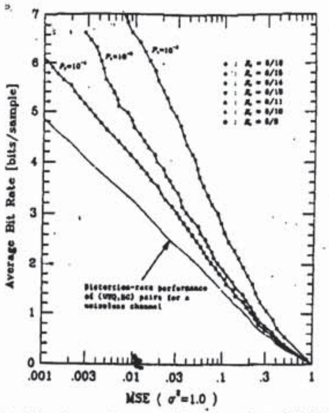


Fig. 10. Rate-distortion performance of the selected set of (UTQ, HC, ECC) triples; $\alpha = 0.6$.

which generates the *mother code* of rate $1/N_c$, and also by *puncturing matrices* with the puncturing period P_c,

$$a(\ell) = \begin{array}{c} \uparrow \\ N_c \\ \downarrow \end{array} \overset{\leftarrow P_c \rightarrow}{\left(\; a_{ij}(\ell) \; \right)}, \qquad \ell = 1, 2, \cdots, (N_c - 1)P_c \tag{8b}$$

which determine the patterns of punctured bits. The nominal rate of the RCPC codes is given by:

$$R_c = P_c/(P_c + \ell), \quad \ell = 1, 2, \cdots, (N_c - 1)P_c \tag{9}$$

which covers the range between $1/N_c$ and $P_c/(P_c + 1)$. In all of our studies, we have used the RCPC code shown in Table I in [18] with $N_c = 4$, $L_c = 5$, and $P_c = 8$.[13] The generator tap matrix g and the puncturing matrices $a(\ell)$ for $\ell = 1, 2, \cdots, 8$ are shown in Figs. 11 and 12, respectively. We have restricted our attention to the BSC with hard-decision decoding; better performance could be obtained on additive Gaussian noise channels with soft-decision decoding.

It should be noted that R_c in (9) is not strictly equal to r_c used in (7) as the rate of the convolutional code. This is because $L_c - 1$ dummy bits should be added to the end of the source encoder output to return the state of the trellis to the all-zero state. Consequently, r_c is given by:

$$r_c = [l_p R_c - (L_c - 1)]/l_p. \tag{10}$$

It should also be noted that when the combined source/channel coding scheme is used, the number of Huffman codewords per packet will be different from that in (6). In this case, the number of codewords in the packet associ-

[12] The actual computation of the $d_c(r; \alpha, P_e)$ function is slightly different from the above. Because there is only a finite number of r_s's and r_c's available, we have actually considered the set of all possible points $\{\hat{d}(r_s, r_c; \alpha, P_e), r_s/r_c\}$ in the distortion-rate plane and selected those that lie on the lower boundary of this set of points.

[13] The performance of the RCPC code will be improved with larger (and still practical) constraint lengths, say, $8 \leq L_c \leq 10$.

$$\mathbf{g} = \begin{bmatrix} 10011 \\ 11101 \\ 10111 \\ 11011 \end{bmatrix}$$

Fig. 11. The generator tap matrix.

$$\mathbf{a}(1) = \begin{bmatrix} 1111\ 0111 \\ 1000\ 1000 \\ 0000\ 0000 \\ 0000\ 0000 \end{bmatrix} \qquad \mathbf{a}(2) = \begin{bmatrix} 1111\ 1111 \\ 1000\ 1000 \\ 0000\ 0000 \\ 0000\ 0000 \end{bmatrix}$$

$$\mathbf{a}(3) = \begin{bmatrix} 1111\ 1111 \\ 1010\ 1000 \\ 0000\ 0000 \\ 0000\ 0000 \end{bmatrix} \qquad \mathbf{a}(4) = \begin{bmatrix} 1111\ 1111 \\ 1010\ 1010 \\ 0000\ 0000 \\ 0000\ 0000 \end{bmatrix}$$

$$\mathbf{a}(5) = \begin{bmatrix} 1111\ 1111 \\ 1110\ 1010 \\ 0000\ 0000 \\ 0000\ 0000 \end{bmatrix} \qquad \mathbf{a}(6) = \begin{bmatrix} 1111\ 1111 \\ 1110\ 1110 \\ 0000\ 0000 \\ 0000\ 0000 \end{bmatrix}$$

$$\mathbf{a}(7) = \begin{bmatrix} 1111\ 1111 \\ 1111\ 1110 \\ 0000\ 0000 \\ 0000\ 0000 \end{bmatrix} \qquad \mathbf{a}(8) = \begin{bmatrix} 1111\ 1111 \\ 1111\ 1111 \\ 0000\ 0000 \\ 0000\ 0000 \end{bmatrix}$$

Fig. 12. The puncturing matrices.

ated with the ith subband is given by:

$$n_{w,i} = [r_{c,i} l_p / (r_{s,i} \cdot n_i)] \tag{11}$$

where $r_{s,i}$ and $r_{c,i}$ are the source and channel coding rates selected from encoding the ith subband and n_i is the same as in (6).

C. Simulation Results

In this section, we will present simulation results for the performance of Systems A and B modified by the combined source/channel coding approach. From now on the channel-optimized versions of System A and System B (with blocksize 4 × 4) will be called System C and System D, respectively. We have studied the performance of Systems C and D at design bit rates of 0.25, 0.5, and 1 b/p for channel BER's of 10^{-2}, 10^{-3}, and 10^{-4}. In all cases, the same packetization scheme with $l_p = 1024$ was used. All subsequent simulation results are based on the 512 × 512 "LENA" image.

To provide some insight into how the encoding rate is divided between the source coding and channel coding operations, in Table VII we have included the average bit rate used for channel coding for different overall encoding rates. Notice that the percentage of bit rate dedicated to error control coding is larger for noisier channels, as one should expect. Also, in this table we have included the PSNR results corresponding to the case that the system is designed for a noisy channel but applied to a noiseless channel. These results provide an upper bound on the system PSNR over noisy channels. The difference between these upper bounds and the PSNR's of Table IV are due to the lower rate used for source coding in Table VII.

The performance of Systems C and D in terms of AVE-PSNR, MIN-PSNR, MAX-PSNR, and STD-PSNR are summarized in Table VIII for different channel BER's and encoding rates. The following important observations can be made.

1) Both Systems C and D provide dramatic improvements over Systems A and B. The improvement of System C over System A is in the range 7–27 dB in AVE-PSNR. The improvement of System D over System B varies between 3 and 21 dB in AVE-PSNR. Typically, these improvements are larger at higher encoding rates and for noisier channels.

2) In all cases for both Systems C and D, the MAX-PSNR coincides with the upper bound on PSNR listed in Table VII. This means that bit errors caused by the noisy channel are sometimes perfectly corrected by the RCPC codes.

3) In almost all cases, System D performs better than System C. Furthermore, System D exhibits a higher degree of robustness against channel noise. Typically, the difference between MAX-PSNR and MIN-PSNR is smaller in System D than in System C; the same holds for STD-PSNR. Since in both systems the same type of channel code is used, this superiority of System D must be due to the inherent robustness of 2-D DCT against transmission noise (similar to our observations in Section V).

4) Systems C and D perform better than the W–O scheme in the presence of channel noise (see Table VI). What is perhaps most interesting is that the performance of Systems C and D over a noisy channel is even better than that of the W–O scheme in the absence of channel noise (with only one exception: System C, 0.25 b/p and $P_e = 10^{-2}$). This has been our justification for not considering a channel-optimized version of the W–O scheme.

In Figs. 13 and 14, we present reconstructed images corresponding to MIN-PSNR and MAX-PSNR obtained from Systems C and D for the design rate of 0.5 b/p at two different values of channel BER, namely $P_e = 10^{-2}$ and 10^{-3}. It is important to mention that the average quality of the reconstructed images in our simulations is usually closer to the image corresponding to MAX-PSNR rather than MIN-PSNR. This is especially true in System D.

D. Channel Mismatch

In designing Systems C and D, it is assumed that the channel BER is known. In many practical situations, the exact value of the BER is not known or the BER varies with time. In such situations, it is important to know the amount of performance loss caused by channel mismatch. Let us denote by PSNR $(P_{e,d}, P_{e,a})$ the AVE-PSNR caused

TABLE VII
DISTRIBUTION OF BITS BETWEEN SOURCE CODING AND CHANNEL CODING FOR "LENA"

System	Design Bit Rate	0.25 BPP			0.5 BPP			1.0 BPP		
	Channel BER	1×10^{-2}	1×10^{-3}	1×10^{-4}	1×10^{-2}	1×10^{-3}	1×10^{-4}	1×10^{-2}	1×10^{-3}	1×10^{-4}
System C	Total Bit Rate	0.26	0.25	0.25	0.49	0.46	0.46	0.91	0.88	0.87
	Channel Bit Rate (%)	0.11 (42%)	0.05 (20%)	0.05 (18%)	0.22 (45%)	0.11 (24%)	0.08 (17%)	0.42 (46%)	0.20 (23%)	0.14 (16%)
	PSNR (Noiseless Case)	29.66	30.92	31.03	32.49	33.86	34.35	35.39	37.14	37.56
System D	Total Bit Rate	0.25	0.24	0.24	0.48	0.46	0.45	0.91	0.89	0.88
	Channel Bit Rate (%)	0.11 (45%)	0.06 (26%)	0.04 (15%)	0.22 (45%)	0.11 (25%)	0.07 (15%)	0.41 (45%)	0.22 (24%)	0.14 (16%)
	PSNR (Noiseless Case)	30.10	31.02	31.65	32.73	34.10	34.56	35.60	37.05	37.59

TABLE VIII
PSNR PERFORMANCE RESULTS (IN dB) FOR "LENA" IN THE PRESENCE OF CHANNEL NOISE

System	Design Bit Rate	0.25 BPP			0.5 BPP			1.0 BPP		
	Channel BER	1×10^{-2}	1×10^{-3}	1×10^{-4}	1×10^{-2}	1×10^{-3}	1×10^{-4}	1×10^{-2}	1×10^{-3}	1×10^{-4}
System C	AVE-PSNR	27.16	29.60	30.91	30.74	32.91	34.30	34.34	35.73	37.50
	MAX-PSNR	29.66	30.92	31.03	32.49	33.86	34.35	35.39	37.14	37.56
	MIN-PSNR	18.98	20.40	27.15	21.60	26.59	33.97	24.87	22.61	37.12
	STD-PSNR	3.28	2.40	0.56	2.96	1.93	0.08	2.52	3.15	0.11
System D	AVE-PSNR	29.96	30.94	31.49	32.38	33.90	34.46	35.24	36.71	37.53
	MAX-PSNR	30.10	31.02	31.65	32.73	34.10	34.56	35.60	37.05	37.59
	MIN-PSNR	28.62	29.46	30.52	26.39	29.00	33.78	32.10	32.88	37.12
	STD-PSNR	0.27	0.25	0.30	0.92	0.72	0.17	0.58	0.66	0.10

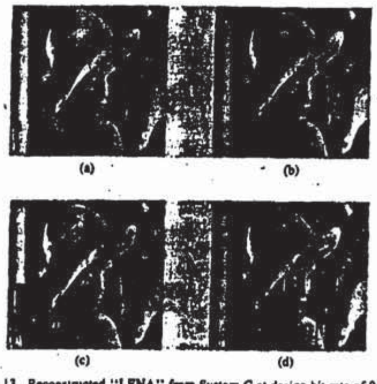


Fig. 13. Reconstructed "LENA" from System C at design bit rate of 0.5 b/p. (a) MAX-PSNR, $P_e = 10^{-2}$. (b) MAX-PSNR, $P_e = 10^{-3}$. (c) MIN-PSNR, $P_e = 10^{-2}$. (d) MIN-PSNR, $P_e = 10^{-3}$.

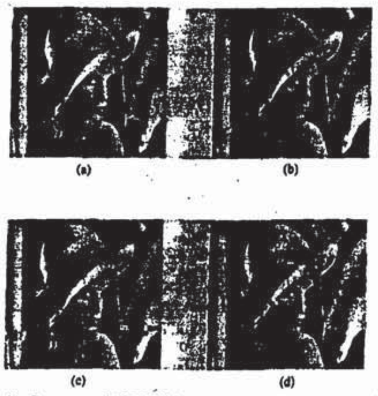


Fig. 14. Reconstructed "LENA" from System D at design bit rate of 0.5 b/p. (a) MAX-PSNR, $P_e = 10^{-2}$ (b) MAX-PSNR, $P_e = 10^{-3}$. (c) MIN-PSNR, $P_e = 10^{-2}$. (d) MIN-PSNR, $P_e = 10^{-3}$.

by a system designed for a channel with BER $P_{e,d}$ and applied to a channel with BER $P_{e,a}$. The PSNR $(P_{e,d}, P_{e,a})$ results for different values of $P_{e,d}$ and $P_{e,a}$ for both Systems C and D are presented in Table IX. These results are for an encoding rate of 0.5 b/p. We have observed that the trend of performance loss is the same for other bit rates.

1) System D is much more robust with respect to channel mismatch than System C.

2) Practically in all cases, the AVE-PSNR of the mismatched case with $P_{e,a} < P_{e,d}$ coincides with the MAX-PSNR of the *matched* case (i.e., when the system is designed and applied to a channel with BER $P_{e,d}$). This implies that, in such cases, all channel errors are corrected by the RCPC codes used in the system.

3) To design the system, overestimating the channel BER is better than underestimating it. For example in

TABLE IX
CHANNEL MISMATCH PERFORMANCE RESULTS AT 0.5 b/p.

System		System C				System D			
$P_{e,a}$	$P_{e,d}$	1×10^{-2}	1×10^{-3}	1×10^{-4}	0.0	1×10^{-2}	1×10^{-3}	1×10^{-4}	0.0
1×10^{-2}	AVE-PSNR	30.74	18.71	9.32	7.46	32.38	25.79	18.78	13.90
	MAX-PSNR	32.49	22.65	11.42	8.27	32.72	28.37	20.82	14.94
	MIN-PSNR	21.60	16.31	7.66	6.97	26.39	22.87	16.76	13.12
	STD-PSNR	2.96	1.29	0.87	0.32	0.92	1.33	1.17	0.45
1×10^{-3}	AVE-PSNR	32.49	32.91	29.16	13.17	32.73	33.90	32.54	20.32
	MAX-PSNR	32.49	33.86	34.10	15.86	32.73	34.10	34.49	24.56
	MIN-PSNR	32.49	26.59	20.91	10.29	32.73	29.00	23.47	17.33
	STD-PSNR	0.0	1.93	4.76	1.16	0.0	0.72	2.42	1.62
1×10^{-4}	AVE-PSNR	32.49	33.86	34.30	23.73	32.73	34.10	34.46	29.81
	MAX-PSNR	32.49	33.86	34.35	34.66	32.73	34.10	34.56	35.04
	MIN-PSNR	32.49	33.86	33.97	18.27	32.73	34.10	33.78	23.58
	STD-PSNR	0.0	0.0	0.08	3.41	0.0	0.0	0.17	2.97
0.0	PSNR	32.49	33.86	34.35	35.16	32.73	34.10	34.56	35.32

System D, $|\text{PSNR}\ (10^{-2}, 10^{-3}) - \text{PSNR}\ (10^{-3}, 10^{-3})| = 1.17$ dB, while $|\text{PSNR}\ (10^{-3}, 10^{-2}) - \text{PSNR}\ (10^{-2}, 10^{-2})| = 6.59$ dB.

E. Side Information

As in Section IV, we need to evaluate the amount of side information necessary to transmit to the receiver for Systems C and D. In addition to the amount of side information evaluated in Section IV, the following two items need to be transmitted:

1) the length indicator of the packets, and

2) the additional bits for error protection of side information.

It is easy to show that the packet length can never be larger than 2^{14} bits; hence, 14 bits are enough to encode the length indicator.[14] As for additional bits for protection of side information, we assume that a rate 1/3 RCPC code is powerful enough to render the side information error-free when the channel error probability is less than 10^{-2}. Under this assumption, the amount of side information of Section IV grows by a factor of three amounting to 0.013 and 0.023 b/p for Systems C and D, respectively, for an image of size 256 × 256; for an image of size 512 × 512, the side information reduces by a factor of four. The length indicator of a packet should also be protected because this information is indispensable for channel decoding. Assuming that 14 bits is used for the length indicator, after error protection 42 bits or six bytes are needed for the length indicator. This increases the bit rate by $100 \times (6 \times 8/1024) = 4.7\%$ corresponding to an increase of 0.012, 0.023, and 0.047 b/p for design average bit rates of 0.25, 0.5, and 1 b/p, respectively. In Table X, we have summarized the increases of the average bit rate incurred by the side information. These numbers have to be added to the total bit rates tabulated in Table VII to calculate the actual overall bit rates.

TABLE X
THE AMOUNT OF SIDE INFORMATION (b/p)

Design Rate				
Image Size	System	0.25	0.5	1.0
256 × 256	System C	0.025	0.036	0.060
	System D	0.035	0.046	0.070
512 × 512	System C	0.015	0.026	0.050
	System D	0.018	0.029	0.053

VII. SUMMARY AND CONCLUSIONS

In this paper, we have developed new schemes for subband image coding over noiseless and noisy channels. For the noiseless channel situation, we have developed two encoding schemes. The difference between the two schemes is in the coding of the lowest frequency subband: the first scheme uses DPCM while the second uses 2-D DCT coding. Both schemes use zero-memory quantization for other subbands. An important feature of these schemes is that the output of all quantizers are entropy-coded. The justification for using entropy-coded quantization resides in the statistical results on the shape of the distribution of subbands, which suggest a significant gain for entropy-coded quantization over conventional Lloyd–Max quantization followed by fixed-length coding. Both schemes perform better than the nonadaptive scheme in [2] (and, hence, other schemes against which comparisons were made in [2]). The difference is significant both subjectively and objectively. The objective performances of the DCT- and DPCM-based schemes are more-or-less the same although, at low bit rates, the DCT-based scheme offers a subjectively noticeable improvement over the DPCM-based scheme.

[14] In all of our simulations, the length of the packet has been less than 2^{11} bits.

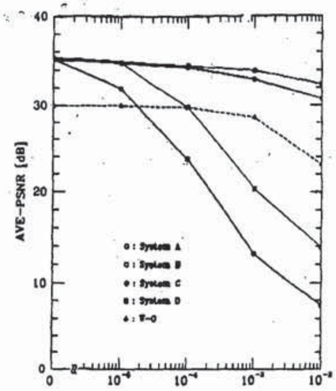


Fig. 15. AVE-PSNR performance of Systems A-D and W-O for different channel BER's at a design rate of 0.5 b/p.

For noisy channels, due to the extensive use of variable-length codes and the concomitant error propagation problems, our schemes exhibit an unacceptable level of sensitivity to channel noise. To combat this difficulty, we have developed a combined source/channel coding scheme in which the schemes designed for the noiseless channel are combined with appropriately designed rate-compatible punctured convolutional codes. Simulation results for a variety of encoding rates and channel bit error rates indicate that the channel-optimized schemes perform dramatically better than their counterparts designed for the noiseless channel (of course, at the cost of some added complexity). They also perform better than the scheme developed in [2].

Fig. 15 illustrates the performance of the W-O scheme as well as Systems A, B, C, and D at the encoding rate of 0.5 b/p. Clearly, Systems A and B exhibit a great sensitivity to channel noise, despite their very good performance for noiseless channels. Systems C and D both perform better than W-O. In fact, in most cases, their worst performance (at $P_e = 10^{-2}$) is still better than the best performance (at $P_e = 0.0$) of W-O. System D performs better than System C and exhibits a better robustness against channel noise. In view of these results, we conclude that the best scheme among those considered here is System D. The DCT blocksize used in System D is 4 × 4, for which the complexity of implementation is quite manageable.

Possible avenues for further research include: i) the study of intraband entropy-constrained VQ [21] for encoding the subbands; ii) the development of an extension of System B in which an *adaptive* 2-D DCT coding, similar to that of [22], is used for encoding the LFS, and iii) the study of system performance for bursty and fading channels.

Acknowledgment

The authors would like to thank the anonymous reviewers for their helpful comments.

References

[1] R. E. Crochiere, S. A. Webber, and J. L. Flanagan, "Digital coding for speech in subbands," *Bell Syst. Tech. J.*, vol. 55, pp. 1069–1085, Oct. 1976.

[2] J. W. Woods and S. D. O'Neil, "Subband coding of images," *IEEE Trans. Acoust., Speech, Signal Proces.*, vol. ASSP-34, pp. 1278–1288, Oct. 1986.

[3] H. Gharavi and A. Tabatabai, "Sub-band coding of monochrome and color images," *IEEE Trans. Circ. Syst.*, vol. 35, pp. 207–214, Feb. 1988.

[4] P. H. Westerink, J. Biemond, and D. E. Boekee, "Sub-band coding of images using predictive vector quantization," in *Proc. ICASSP*, Apr. 1987, pp. 1378–1381.

[5] P. H. Westerink, D. E. Boekee, J. Biemond, and J. W. Woods, "Subband coding of images using vector quantization," *IEEE Trans. Commun.*, vol. 36, pp. 713–719, June 1988.

[6] R. J. Safranek, K. Mackey, N. S. Jayant, and T. Kim, "Image coding based on selective quantization of the reconstruction noise in the dominant sub-band," in *Proc. ICASSP*, Apr. 1988, pp. 765–768.

[7] C. Kim, J. Bruder, M. J. T. Smith, and R. M. Mersereau, "Subband coding of color images using finite state vector quantization," in *Proc. ICASSP*, Apr. 1988, pp. 753–756.

[8] J. D. Johnston, "A filter family designed for use in quadrature mirror filter banks," *Proc. ICASSP*, Apr. 1980, pp. 291–294.

[9] M. J. T. Smith and S. L. Eddins, "Subband coding of images with octave band tree structures," in *Proc. ICASSP*, Apr. 1987, pp. 1382–1385.

[10] N. Farvardin and J. W. Modestino, "Optimum quantizer performance for a class of non-Gaussian memoryless sources," *IEEE Trans. Inform. Theory*, vol. IT-30, pp. 485–497, May 1984.

[11] P. H. Westerink, J. Biemond, and D. E. Boekee, "Evaluation of image sub-band coding schemes," in *Proc. EURASIP*, Sept. 1988, pp. 1149–1152.

[12] R. C. Reininger and J. D. Gibson, "Distributions of the two-dimensional DCT coefficients for images," *IEEE Trans. Commun.*, vol. COM-31, pp. 835–839, June 1983.

[13] Y. Shoham and A. Gersho, "Efficient bit allocation for arbitrary set of quantizers," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 36, pp. 1445–1453, Sept. 1988.

[14] N. S. Jayant and P. Noll, *Digital Coding of Waveforms.* Englewood Cliffs, NJ: Prentice-Hall, 1984.

[15] N. Farvardin and V. Vaishampayan, "Optimal quantizer design for noisy channels: An approach to combined source-channel coding," *IEEE Trans. Inform. Theory*, vol. 33, pp. 827–838, Nov. 1987.

[16] J. W. Modestino and D. G. Daut, "Combined source-channel coding of images," *IEEE Trans. Commun.*, vol. COM-27, pp. 1644–1659, Nov. 1979.

[17] J. W. Modestino, D. G. Daut, and A. L. Vickers, "Combined source-channel coding of images using the block cosine transform," *IEEE Trans. Commun.*, vol. COM-29, pp. 1261–1274, Sept. 1981.

[18] J. Hagenauer, "Rate-compatible punctured convolutional codes (RCPC codes) and their applications," *IEEE Trans. Commun.*, vol. 36, pp. 389–400, Apr. 1988.

[19] J. B. Cain, G. C. Clark, Jr., and J. M. Geist, "Punctured convolutional codes of rate $(n - 1)/n$ and simplified maximum likelihood decoding," *IEEE Trans. Inform. Theory.*, vol. IT-25, pp. 97–100, Jan. 1979.

[20] R. V. Cox, J. Hagenauer, N. Seshadri, and C.-E. Sundberg, "A subband coder designed for combined source and channel coding," in *Proc. ICASSP*, Apr. 1988, pp. 235–238.

[21] P. A. Chou, T. Lookabaugh, and R. M. Gray, "Entropy-constrained vector quantization," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 37, pp. 31–42, Jan. 1989.

[22] W.-H. Chen and C. H. Smith, "Adaptive coding of monochrome and color images," *IEEE Trans. Commun.*, vol. COM-25, pp. 1285–1292, Nov. 1977.

[23] N. Tanabe and N. Farvardin, "Subband image coding using entropy-coded quantization over noisy channels," Tech. Rep. UMIACS-TR-89-86, Univ. Maryland, Aug. 1989.

Naoto Tanabe was born in Kyoto, Japan, on February 2, 1955. He received the B.E. and M.E. degrees in electrical engineering from the University of Tokyo in 1977 and 1979, respectively.

Since joining Mitsubishi Electric Corporation in 1979, he has been engaged in research and development on image communication systems, especially for facsimile. He was a Visiting Scholar at the University of Maryland, College Park, during 1988–1989. He is a Senior Research Engineer at the Information and Communication Systems Group of the Mitsubishi Electric Corporation. His current research interests include image coding and processing with application to image communication terminals.

Mr. Tanabe is a member of the Institute of Electronics, Information and Communication Engineers of Japan.

Nariman Farvardin was born in Tehran, Iran, on July 15, 1956. He received the B.S., M.S., and Ph.D. degrees in electrical engineering from Rensselaer Polytechnic Institute, Troy, NY, in 1979, 1980, and 1983, respectively.

During 1980–1981 and 1982–1983 he was a Research Assistant in the Electrical, Computer, and Systems Engineering Department of Rensselaer Polytechnic. He received an IBM fellowship during the 1981–1982 academic year. Since January 1984, he has been with the Department of Electrical Engineering, University of Maryland, College Park, where he is currently an Associate Professor and holds a joint appointment with the Institute for Advanced Computer Studies. During the 1990–1991 academic year, he was a Visiting Professor at École Nationale Superieure des Telecommunications, Paris, France. His research interests include information theory, digital communications, and signal processing with application to speech/image coding and transmission. He was the Associate Editor for Quantization, Speech/Image Coding of the IEEE TRANSACTIONS ON COMMUNICATIONS from 1987 to 1990. In 1987, he received the Presidential Young Investigator Award from the National Science Foundation.

Reversible image compression via multiresolution representation and predictive coding

Amir Said and William A. Pearlman

Department of Electrical, Computer, and Systems Engineering
Rensselaer Polytechnic Institute, Troy, New York 12180

ABSTRACT

In this paper a new image transformation suited for reversible (lossless) image compression is presented. It uses a simple pyramid multiresolution scheme which is enhanced via predictive coding. The new transformation is similar to the subband decomposition, but it uses only integer operations. The number of bits required to represent the transformed image is kept small though careful scaling and truncations. The lossless coding compression rates are smaller than those obtained with predictive coding of equivalent complexity. It is also shown that the new transform can be effectively used, with the same coding algorithm, for both lossless and lossy compression. When used for lossy compression, its rate-distortion function is comparable to other efficient lossy compression methods.

1. INTRODUCTION

In general, reversible (or lossless) image compression is required whenever some processing (magnification, filtering, subtraction, etc.) should be applied to the image. Most lossy compression techniques are designed for the human visual system and may destroy some of the information required during processing. It is also indicated for images obtained at great cost, such as medical images, when it is unwise to discard any information that later may be found to be necessary.

Some of the most effective methods for reversible compression use linear predictive coding,[1, 2] which is the method adopted in the JPEG Still Picture Compression Standard.[3] This form of compression is usually defined for a single resolution, in a way that the image can be recovered only in its entirety—a characteristic limits the application of those methods.

There are several advantages in a multiresolution representation of the image. One of them is the possibility of *progressive-resolution transmission,* where the image is recovered at increasingly finer resolutions. This, in turn, allows a *multi-use* scheme, where users with devices of different

0-8194-1369-0/93/$6.00

resolutions can access the same compressed image file and efficiently recover the image only up to the resolution the device is capable of using (displaying, printing, etc).

Another advantage of the multiresolution representation is related to the coding efficiency.[4] Single-resolution predictive coding requires a statistical model of the image, and may be based on some assumptions, like stationary distributions, which are seldom encountered in real images. This problem can be alleviated with more complex adaptive models, but at the expense of a much larger computational effort. On the other hand, the multiresolution representation has a recursive structure, in such a way that a high order and complex transformation can be done with a sequence of low order and simple transformations.

One important aspect of reversible compression is that, in a sense, all bits of the image representation are equally important, because all have to be recovered. Lossy compression methods use only the numerical pixel values to remove correlations and to select the most relevant information. For that reason, some of the most effective transformations for lossy compression (e.g., DCT, PR-QMF subband decomposition, etc.) are reversible only if its coefficients are perfectly represented as real numbers. Truncating the values in the transformed image may render the method inefficient for reversible compression because too many bits are used to code the fractional part.[2] Adapting these transforms to use only integer operations does not solve the problem because they still increase the number of bits required for an exact representation.

Since the pixel's numerical value is important in the image's context, a good transformation for reversible compression may use arithmetic operations to reduce the correlations, but it must deal carefully with the truncation process, and must limit the maximum number of bits required to represent each pixel in the transformed image.

In this paper we propose a new integer transformation for reversible image compression that addresses these problems. It uses a simple pyramid multiresolution scheme enhanced with predictive coding. We call it S+P transform. It differs from other methods[5] because the prediction is used *during* (instead of after) the sequence of recursive transformations, and hence can use information that is not available after the image is transformed. Numerical results show that the S+P transform allows more compression than single-resolution linear predictive coding methods of small complexity, while keeping the advantages of the multiresolution representation, and can be calculated with a very small computational effort.

It is also shown that the S+P transform it is well suited for ***progressive-fidelity transmission***, where, for a single resolution, the image quality is gradually improved up to perfect reconstruction. This approach uses the same coding algorithm for lossy and lossless compression, without requiring, for example, the lossy-plus-residual approach.[1] The results show that its rate-distortion function is comparable to other efficient lossy compression methods.

2. MULTIRESOLUTION REPRESENTATION

We begin with a pyramid transformation known as the S (Sequential) transform,[1, 6] which is similar to the Haar multiresolution image representation.[7] There are different definitions of the S transform

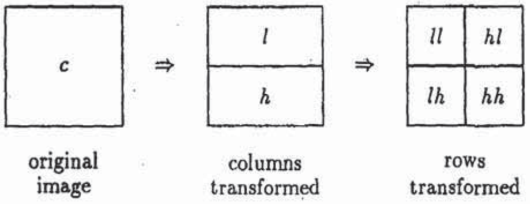


Figure 1: Multiresolution sequential transformation.

in the literature, but most differ only in some implementation details.

A sequence of random integers $c[n]$ with length N, can be perfectly represented by the two sequences with length $N/2$ defined by:

$$\begin{aligned} l[n] &= \lfloor (c[2n] + c[2n+1])/2 \rfloor , \\ h[n] &= c[2n] - c[2n+1], \end{aligned} \tag{1}$$

where $\lfloor \cdot \rfloor$ corresponds to downward truncation. The sequences $l[n]$ and $h[n]$ form the S transform of $c[n]$.

The inverse transformation is:

$$\begin{aligned} c[2n] &= l[n] + \lfloor (h[n] + 1)/2 \rfloor , \\ c[2n+1] &= c[2n] - h[n]. \end{aligned} \tag{2}$$

The advantage of this representation is that, if the correlation coefficient of $c[2n]$ and $c[2n+1]$ is larger than 1/3, then the average variance of $l[n]$ and $h[n]$ is smaller than the variance of $c[n]$.[4] Since the adjacent pixels in an image are highly correlated, we apply (1) to the sequence of image pixels to reduce its first-order entropy. In this case, $h[n]$ normally has small variance, while the variance of $l[n]$ is approximately equal to the variance of $c[n]$.

The two-dimensional transformation is done by applying the transformation (1) sequentially to the rows and columns of the image, as shown in Figure 1. The components corresponding to $l[n]$ then are the mean of 2×2 pixel blocks of the image. They form another image with half the resolution, and with statistical properties that are similar to those of the original image. Hence, the same transformation can be recursively applied to this lower resolution "mean image" (*ll* in Figure 1) to form a multiresolution hierarchical pyramid.[4]

Note that the maximum number of bits required to represent each pixel in the *ll* images does not change with each transformation. In addition, the S transform is so simple that it is easy to find the truncation that allows perfect reconstruction. There is no data expansion in this transformation, i.e., it can be done "in place," and use the same number of pixels of the original image. Except for the truncations in (1) and (2), this transformation is equal to a subband decomposition. For that reason we borrow some of its terminology. For instance, we call $l[n]$ and $h[n]$ the lowpass and highpass components.

Because the low resolution (ll) images are formed with mean values (a form of lowpass filtering), the degradation of those image's quality due to aliasing effects will occur only after several transformations—an advantage over the unfiltered subsampling used, for example, in the HINT (Hierarchical INTerpolation) coding method.[2, 11]

The S transform is very simple, can be quickly calculated, and significantly reduces the first-order entropy. However, it is too simple to eliminate the correlation between the highpass components. On the other hand, one advantage of predictive coding is that it does not have to be linear for perfect reconstruction, and the prediction value can be truncated at will. Since the S transform components *are* correlated, we can use a predictive coding method to further reduce the first-order entropy of the transformed image, while keeping the transformation reversible.

3. S+P TRANSFORM

In the S+P transform (**S**-transform + **P**rediction) we use, during each one-dimensional transformation, some values of $l[n]$ and $h[n]$ to estimate of the value of a given $h[n]$. Calling the estimate $\hat{h}[n]$, the difference

$$h_d[n] = h[n] - \left\lfloor \hat{h}[n] \right\rfloor, \tag{3}$$

replaces $h[n]$, forming a new transformed image with smaller first-order entropy. No estimation is subtracted from $l[n]$ because it is later transformed with the same method.

Defining

$$\Delta l[n] = l[n-1] - l[n], \tag{4}$$

the general form of the estimator is:

$$\hat{h}[n] = \alpha_1 \, \Delta l[n] + \alpha_2 \, \Delta l[n+1] - \alpha_3 \, h[n+1]. \tag{5}$$

With this formulation all terms have zero mean. To simplify the notation, we disregard, for now, the image borders. Note that (5) is linear, while (3) is not.

During the inverse S+P transformation the pixels are visited in the inverse order, so that the information required to calculate the prediction have already been recovered. So, the inverse transformation algorithm is like the transformation algorithm running "backwards," and the prediction is added instead of subtracted. After $l[n]$ and $h[n]$ are recovered the inverse S transform (2) is calculated.

The predictor coefficients that minimize the variance of $h_d[n]$ can be found by solving the Yule-Walker equations.[2] This solution will not necessarily minimize the entropy of $h_d[n]$, but we found that it is practically optimal. In fact, we discovered that there can be several solutions that can practically minimize the entropy. This freedom of choice is important because it allows us to select good rational coefficients, and calculate the S+P transform with fast integer operations and

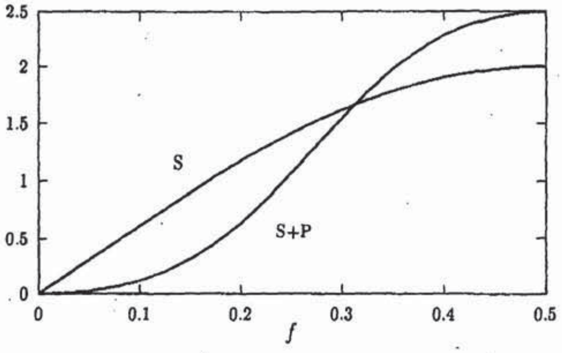


Figure 2: Frequency response of the S and S+P transforms.

bit-shifts instead of multiplications or divisions. This way the S+P transform can maintain the speed advantage of the S transform.

The interpretation of the S+P transform is easier in the frequency domain. If we disregard the truncations, we can combine (1), (3), (4) and (5), and see that $h_d[n]$ can be regarded as the output of a FIR filter, to the input sequence $c[n]$. The z-transform of the filter's response is:

$$\begin{aligned} F(z) &= \frac{1}{2}\Big\{-\alpha_1 - \alpha_1\, z^{-1} + (\alpha_1 - \alpha_2 + 2)\, z^{-2} + (\alpha_1 - \alpha_2 - 2)\, z^{-3} + \\ &\qquad (\alpha_2 + 2\alpha_3)\, z^{-4} + (\alpha_2 - 2\alpha_3)\, z^{-5}\Big\}. \end{aligned} \tag{6}$$

Figure 2 shows the frequency response of a particular S+P estimator compared to the S transform. Since most of an image's energy is concentrated in the low frequency components, it is clear that a filter with higher attenuation in the low frequencies will reduce the variance of $h_d[n]$. On the other hand, there is an inevitable amplification of the higher frequencies. This is due to the structure of (6), and is required to keep the transformation reversible.

Even though our estimator has only three parameters, a good deal of modeling can be done in the rcsponse (6). Figure 3 shows the frequency response of the set of predictors in Table 1. In principle, the choice for the best predictor depends on the image's characteristics. Smooth and noiseless images are better compressed using the filter with largest attenuation on the low frequencies. Noisy and very detailed images require smaller amplification of the high frequencies. However, we next show that the predictor choice is not critical, and that there are good "universal" predictors, i.e., those that are effective for a broad class of images (e.g., portraits, landscape, medical, etc.).

We tested the selected predictors with the set of images in Table 2. The first four are well known, and can be seen in the references. The images CT 1 to 4 are medical (tomography)

parameter	predictor				
	A	B	C	D	E
α_1	1/4	4/16	2/8	3/16	3/16
α_2	1/4	5/16	3/8	8/16	9/16
α_3	0	2/16	2/8	6/16	8/16

Table 1: Parameters of the set of selected predictors.

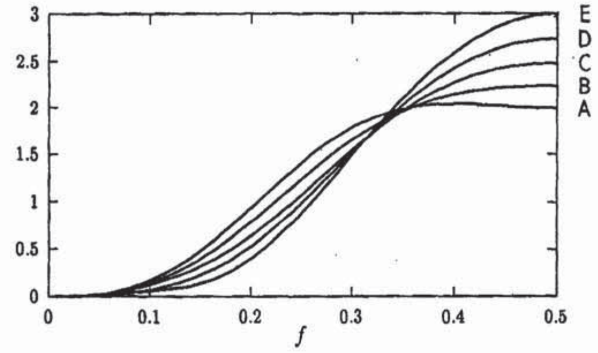


Figure 3: Frequency response of the selected set of predictors.

Name	origin	dimension	bits/pixel
Girl, Couple	USC	256 × 256	8
Lena, Mandrill	USC	512 × 512	8
CT 1 to 4	CT	512 × 512	12

Table 2: Set of test images.

images. The first-order entropies of the transformed images are shown in Table 3. Those entropies are calculated as the weighted mean of the entropies in each of the pyramid quadrants. This way they are more accurate estimates of the entropy-coding bit rates when different adaptive models are used for each quadrant.

To give a reference for comparisons Table 3 also shows the first-order entropies obtained with the JPEG[3] third-order predictors #4 and #7. We can see that the difference between the S (no predictor column) and S+P transforms is significant, but the S+P entropy in some cases is not much smaller than JPEG's. Nevertheless, the multiresolution advantages should also be taken into account. In addition, we show in the next section that, using the pyramid structure, the S+P transform can be compressed to rates smaller than the first-order entropy, and it is clearly superior to JPEG rates.

4. CODING RESULTS

We selected the predictor C for our coding tests because it can be calculated as

$$\hat{h}[n] = \frac{1}{8}\ \{\, 2\ (\Delta l[n] + \Delta l[n+1] - h[n+1]) + \Delta l[n+1]\ \}, \tag{7}$$

and hence use bit shifts instead of multiplications and divisions. In the image borders we used the predictors

$$\begin{aligned} \hat{h}[1] &= \frac{1}{4}\ \{\ \Delta l[2] + \Delta l[3]\ \}, & (8) \\ \hat{h}[N/2] &= \frac{1}{4}\ \{\ \Delta l[N/2-1] + \Delta l[N/2]\ \}. & (9) \end{aligned}$$

In our tests the CPU times to calculate the S and S+P transforms of a 512 × 512 image were 0.4 s and 0.5 s, respectively (SUN-SPARC 10 workstation). All rates presented in this section are not entropy estimates, but are calculated from the size of the compressed files. In all tests the S+P transform pyramids have five levels.

In our first sequence of tests we evaluated the progressive-resolution transmission scheme. The adaptive arithmetic coding algorithm of Witten et al.[8] was used to entropy-code the different resolution images. An adaptive Markov model with five states (contexts) was used, where the state is defined by the value of the previous pixel in a row (e.g., state 1 if $0 \leq hh_d[n-1] < 4$, where hh_d represents a pixel value in the completely transformed image). The model was reset before coding each part of the pyramid. The Markov model does not change the speed of the arithmetic encoder, and can explore the remaining dependencies between transform pixels.

Table 4 shows the results of those tests. The same file was used to progressively recover images at increasingly finer resolutions. The actual bit rate required to code each image was divided by the number of pixels in the original image to give the bit/pixel rates in the table. Hence, those bit/pixel rates represent the contribution of the low resolution images. Note that the full-resolution rates can be smaller than the first-order entropies of Table 3. The average coding and decoding times were 4.2 s and 4.8 s, respectively, for 512 × 512 images.

In the second set of tests we used the zero-tree[9, 10] compression algorithm. This coding method uses the magnitude ordering of the different pyramid levels to achieve larger compression. The sequence of information coded by this method is similar to bit-plane coding, but, instead of visiting all pixels in the image to code a bit-plane, it uses a tree structure to avoid visiting the pixels with small magnitude. This way it can detect and more efficiently code the low-activity regions in the transformed image. It is simple, fast and, because it works with individual bits, can be used for reversible compression.

Two versions of the zero-tree coding algorithm (called I and II) were used, and both can compress the image only in full-resolution. Version I does not allow any form of progressive transmission, while version II is designed for progressive-fidelity transmission. The difference between the two versions is that version I uses the transformed image as previously defined, while version II implicitly scales the transformed image so that the transformation is nearly orthogonal—a requirement to minimize the mean squared-error distortion.

Table 5 shows the compression achieved with both versions to code the test images. The best results, relative to the first-order entropy, were obtained for the set of medical images (CT). The average coding and decoding times of version I were 12.0 s and 12.8 s, respectively, for 512 × 512 images.

In the progressive-fidelity transmission scheme the decoder initially sets the reconstructed image to zero and update its pixel values using the coded message. All results were obtained from the same file: the decoder can decide at which rate to stop, and then it calculates the inverse S+P transform to obtain a lossy version of the image. If it continues decoding to the end of the file then the image is recovered exactly. The results obtained with the image "Lena" are shown in Table 6 (the decoding CPU times do not include the inverse transformation). The average CPU time of version II to code and decode a 512 × 512 image up to perfect reconstruction were 10.5 s and 11.0 s, respectively.

Those rate *versus* PSNR results are excellent, considering the speed of the S+P transform and decoding algorithm. They are slightly inferior to methods like subband coding with adaptive vector quantization,[12] but superior to some other vector quantization coding methods.[13] Figure 4 shows the lossy version of the image "Lena" coded with this method, at rate 0.3 bit/pixel. Like the images coded via subband decomposition, there are no blocking artifacts, and, even though a bit-plane approach was used, the inverse transformation completely eliminates the "banding" artifacts usually present in bit-plane coded images.

REFERENCES

1. M. Rabbani and P.W. Jones, *Digital Image Compression Techniques*, SPIE Opt. Eng. Press, Bellingham, Washington, 1991.
2. G.R. Kuduvalli and R.M. Rangayyan, "Performance analysis of reversible image compression techniques for high-resolution digital teleradiology," *IEEE Trans. Med. Imaging*, vol. 11, pp. 430–445, Sept. 1992.

3. G.K. Wallace, "The JPEG still picture compression standard," *Comm. ACM*, vol. 34, pp. 30–44, April 1991.

4. R.P. Rao and W.P. Pearlman, "On entropy of pyramid structures," *IEEE Trans. Inform. Theory*, vol. 35, pp. 407–412, March 1991.

5. J.W. Woods and S.D. O'Neil, "Subband coding of images," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 34, pp. 1278–1288, Oct. 1986.

6. V.K. Heer and H-E. Reinfelder, "A comparison of reversible methods for data compression," *Proc. SPIE*, vol. 1233 Med. Imag. IV, pp. 354–365, 1990.

7. E.H. Adelson, E. Simoncelli, and R. Hingorani, "Orthogonal pyramid transforms for image coding," *Proc. SPIE*, vol. 845, Cambridge, MA, pp. 50–58, Oct. 1987.

8. I.H. Witten, R.M. Neal, and J.G. Cleary, "Arithmetic coding for data compression," *Commun. ACM*, vol. 30, pp. 520–540, June 1987.

9. J.M. Shapiro, "An embedded wavelet hierarchical image coder," *Proc. IEEE Int. Conf. on Acoust., Speech, Signal Processing,* San Francisco, CA, pp. IV 657–660, March 1992.

10. A Said and W.A. Pearlman, "Image compression using the spatial-orientation tree," *Proc. IEEE Int. Symp. on Circuits and Systems,* Chicago, IL, pp. 279–282, May 1993.

11. P. Roos, A. Viergever, and M.C.A. van Dijke, "Reversible intraframe compression of medical images," *IEEE Trans. Med. Imaging,* vol. 7, pp. 328–336, Sept. 1988.

12. Y.H. Kim, and J.W. Modestino, "Adaptive entropy coded subband coding of images," *IEEE Trans. Image Processing,* vol. 1, pp. 31–48, Jan. 1992.

13. S-W. Wu and A. Gersho, "Lapped block decoding for vector quantization of images," *Proc. SPIE*, vol. 1818 Visual Comm. and Image Processing, pp. 488–499, 1992.

image	S+P predictor						JPEG	
	none	A	B	C	D	E	4	7
Girl	5.00	4.56	4.51	4.48	4.50	4.55	4.92	4.75
Couple	4.45	4.23	4.17	4.13	4.15	4.20	4.23	4.37
Lena	4.77	4.41	4.35	4.32	4.33	4.38	4.81	4.61
Mandrill	6.12	5.96	5.90	5.88	5.92	5.98	6.39	6.04
CT 1	4.29	3.74	3.63	3.54	3.45	3.44	3.77	4.11
CT 2	6.60	6.00	5.89	5.82	5.75	5.73	5.94	6.40
CT 3	5.62	5.10	5.01	4.94	4.89	4.88	5.32	5.47
CT 4	5.88	5.25	5.12	5.02	4.96	4.95	5.33	5.70

Table 3: First-order entropy of different predictors (bit/pixel).

image	resolution		
	128 × 128	256 × 256	512 × 512
Girl	1.18	4.58	—
Couple	1.12	4.08	—
Lena	0.31	1.11	4.29
Mandrill	0.36	1.48	5.87
CT 1	0.35	1.12	3.32
CT 2	0.51	1.72	5.68
CT 3	0.39	1.41	4.78
CT 4	0.43	1.47	4.86

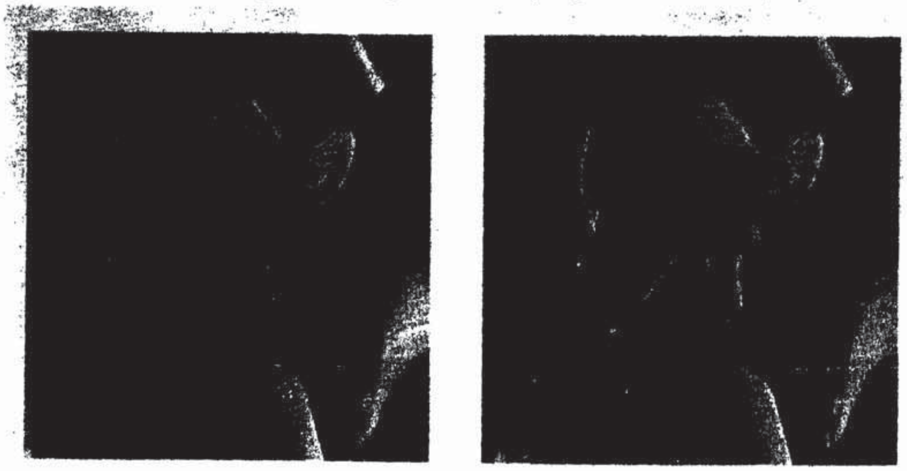
Table 4: Progressive-resolution transmission rates (bit/pixel).

Version	Image							
	Girl	Couple	Lena	Mandrill	CT 1	CT 2	CT 3	CT 4
I	4.41	3.92	4.20	5.69	2.94	5.27	4.40	4.46
II	4.45	4.08	4.25	5.77	3.08	5.53	4.65	4.70

Table 5: Zero-tree method compression rates (bit/pixel).

	rate (bit/pixel)						
	0.2	0.3	0.4	0.5	0.6	0.7	0.8
MSE	58.6	36.5	30.7	22.4	20.3	16.6	12.8
PSNR (dB)	30.5	32.5	33.3	34.6	35.1	35.9	37.1
decoding time (s)	0.52	0.79	1.12	1.45	1.78	2.00	2.38

Table 6: Progressive-fidelity transmission (image "Lena" 512 × 512).



original "Lena"

0.3 bit/pixel, PSNR = 32.5 dB

Figure 4: Lossy reproduction of the image "Lena."

-Jim Meany

Embedded Image Coding Using Zerotrees of Wavelet Coefficients

Jerome M. Shapiro

***Abstract*—The embedded zerotree wavelet algorithm (EZW) is a simple, yet remarkably effective, image compression algorithm, having the property that the bits in the bit stream are generated in order of importance, yielding a fully embedded code. The embedded code represents a sequence of binary decisions that distinguish an image from the "null" image. Using an embedded coding algorithm, an encoder can terminate the encoding at any point thereby allowing a target rate or target distortion metric to be met exactly. Also, given a bit stream, the decoder can cease decoding at any point in the bit stream and still produce exactly the same image that would have been encoded at the bit rate corresponding to the truncated bit stream. In addition to producing a fully embedded bit stream, EZW consistently produces compression results that are competitive with virtually all known compression algorithms on standard test images. Yet this performance is achieved with a technique that requires absolutely no training, no pre-stored tables or codebooks, and requires no prior knowledge of the image source.**

The EZW algorithm is based on four key concepts: 1) a discrete wavelet transform or hierarchical subband decomposition, 2) prediction of the absence of significant information across scales by exploiting the self-similarity inherent in images, 3) entropy-coded successive-approximation quantization, and 4) universal lossless data compression which is achieved via adaptive arithmetic coding.

I. Introduction and Problem Statement

THIS paper addresses the two-fold problem of 1) obtaining the best image quality for a given bit rate, and 2) accomplishing this task in an embedded fashion, i.e., in such a way that all encodings of the same image at lower bit rates are embedded in the beginning of the bit stream for the target bit rate.

The problem is important in many applications, particularly for progressive transmission, image browsing [25], multimedia applications, and compatible transcoding in a digital hierarchy of multiple bit rates. It is also applicable to transmission over a noisy channel in the sense that the ordering of the bits in order of importance leads naturally to prioritization for the purpose of layered protection schemes.

A. Embedded Coding

An embedded code represents a sequence of binary decisions that distinguish an image from the "null," or all gray, image. Since, the embedded code contains all lower rate codes "embedded" at the beginning of the bit stream, effectively, the bits are "ordered in importance." Using an embedded code, an encoder can terminate the encoding at any point thereby allowing a target rate or distortion metric to be met exactly. Typically, some target parameter, such as bit count, is monitored in the encoding process. When the target is met, the encoding simply stops. Similarly, given a bit stream, the decoder can cease decoding at any point and can produce reconstructions corresponding to all lower-rate encodings.

Embedded coding is similar in spirit to binary finite-precision representations of real numbers. All real numbers can be represented by a string of binary digits. For each digit added to the right, more precision is added. Yet, the "encoding" can cease at any time and provide the "best" representation of the real number achievable within the framework of the binary digit representation. Similarly, the embedded coder can cease at any time and provide the "best" representation of an image achievable within its framework.

The embedded coding scheme presented here was motivated in part by universal coding schemes that have been used for lossless data compression in which the coder attempts to optimally encode a source using no prior knowledge of the source. An excellent review of universal coding can be found in [3]. In universal coders, the encoder must *learn* the source statistics as it progresses. In other words, the source model is incorporated into the actual bit stream. For lossy compression, there has been little work in universal coding. Typical image coders require extensive training for both quantization (both scalar and vector) and generation of nonadaptive entropy codes, such as Huffman codes. The embedded coder described in this paper attempts to be universal by incorporating all learning into the bit stream itself. This is accomplished by the exclusive use of adaptive arithmetic coding.

Intuitively, for a given rate or distortion, a nonembedded code should be more efficient than an embedded code, since it is free from the constraints imposed by embedding. In their theoretical work [9], Equitz and Cover proved that a successively refinable description can only be optimal if the source possesses certain Markovian

Manuscript received April 28, 1992; revised June 13, 1993. The guest editor coordinating the review of this paper and approving it for publication was Prof. Martin Vetterli.

The author is with the David Sarnoff Research Center, Princeton, NJ 08543.

IEEE Log Number 9212175.

1053-587X/93$03.00 © 1993 IEEE

properties. Although optimality is never claimed, a method of generating an embedded bit stream with no apparent sacrifice in image quality has been developed.

B. Features of the Embedded Coder

The EZW algorithm contains the following features

- A discrete wavelet transform which provides a compact multiresolution representation of the image.
- Zerotree coding which provides a compact multiresolution representation of *significance maps*, which are binary maps indicating the positions of the significant coefficients. Zerotrees allow the successful prediction of insignificant coefficients across scales to be efficiently represented as part of exponentially growing trees.
- Successive Approximation which provides a compact *multiprecision* representation of the significant coefficients and facilitates the embedding algorithm.
- A prioritization protocol whereby the ordering of importance is determined, in order, by the precision, magnitude, scale, and spatial location of the wavelet coefficients. Note in particular, that larger coefficients are deemed more important than smaller coefficients regardless of their scale.
- Adaptive multilevel arithmetic coding which provides a fast and efficient method for entropy coding strings of symbols, and requires no training or prestored tables. The arithmetic coder used in the experiments is a customized version of that in [31].
- The algorithm runs sequentially and stops whenever a target bit rate or a target distortion is met. A target bit rate can be met *exactly*, and an operational rate-vs.-distortion function (RDF) can be computed point-by-point.

C. Paper Organization

Section II discusses how wavelet theory and multiresolution analysis provide an elegant methodology for representing "trends" and "anomalies" on a statistically equal footing. This is important in image processing because edges, which can be thought of as anomalies in the spatial domain, represent extremely important information despite that fact that they are represented in only a tiny fraction of the image samples. Section III introduces the concept of a *zerotree* and shows how zerotree coding can efficiently encode a significance map of wavelet coefficients by predicting the absence of significant information across scales. Section IV discusses how successive approximation quantization is used in conjunction with zerotree coding, and arithmetic coding to achieve efficient embedded coding. A discussion follows on the protocol by which EZW attempts to order the bits in order of importance. A key point there is that the definition of importance for the purpose of ordering information is based on the magnitudes of the uncertainty intervals as seen from the viewpoint of what the decoder can figure out. Thus, there is no additional overhead to transmit this ordering information. Section V consists of a simple 8 × 8 example illustrating the various points of the EZW algorithm. Section VI discusses experimental results for various rates and for various standard test images. A surprising result is that using the EZW algorithm, terminating the encoding at an arbitrary point in the encoding process does not produce any artifacts that would indicate where in the picture the termination occurs. The paper concludes with Section VII.

II. Wavelet Theory and Multiresolution Analysis

A. Trends and Anomalies

One of the oldest problems in statistics and signal processing is how to choose the size of an analysis window, block size, or record length of data so that statistics computed within that window provide good models of the signal behavior within that window. The choice of an analysis window involves trading the ability to analyze "anomalies," or signal behavior that is more localized in the time or space domain and tends to be wide band in the frequency domain, from "trends," or signal behavior that is more localized in frequency but persists over a large number of lags in the time domain. To model data as being generated by random processes so that computed statistics become meaningful, stationary and ergodic assumptions are usually required which tend to obscure the contribution of anomalies.

The main contribution of wavelet theory and multiresolution analysis is that it provides an elegant framework in which both anomalies and trends can be analyzed on an equal footing. Wavelets provide a signal representation in which some of the coefficients represent long data lags corresponding to a narrow band, low frequency range, and some of the coefficients represent short data lags corresponding to a wide band, high frequency range. Using the concept of *scale*, data representing a continuous tradeoff between time (or space in the case of images) and frequency is available.

For an introduction to the theory behind wavelets and multiresolution analysis, the reader is referred to several excellent tutorials on the subject [6], [7], [17], [18], [20], [26], [27].

B. Relevance to Image Coding

In image processing, most of the image area typically represents spatial "trends," or areas of high statistical spatial correlation. However "anomalies," such as edges or object boundaries, take on a perceptual significance that is far greater than their numerical energy contribution to an image. Traditional transform coders, such as those using the DCT, decompose images into a representation in which each coefficient corresponds to a fixed size spatial area and a fixed frequency bandwidth, where the bandwidth and spatial area are effectively the same for all coefficients in the representation. Edge information tends to disperse so that many non-zero coefficients are required to represent edges with good fidelity. However, since the edges represent relatively insignificant energy with respect to the entire image, traditional transform coders, such as those using the DCT, have been fairly successful at medium and high bit rates. At extremely low bit rates, however, traditional transform coding techniques, such as JPEG [30], tend to allocate too many bits to the "trends," and have few bits left over to represent "anomalies." As a result, blocking artifacts often result.

Wavelet techniques show promise at extremely low bit rates because trends, anomalies, and information at all "scales" in between are available. A major difficulty is that fine detail coefficients representing possible anomalies constitute the largest number of coefficients, and therefore, to make effective use of the multiresolution representation, much of the information is contained in representing the *position* of those few coefficients corresponding to significant anomalies.

The techniques of this paper allow coders to effectively use the power of multiresolution representations by efficiently representing the positions of the wavelet coefficients representing significant anomalies.

C. A Discrete Wavelet Transform

The discrete wavelet transform used in this paper is identical to a hierarchical subband system, where the subbands are logarithmically spaced in frequency and represent an octave-band decomposition. To begin the decomposition, the image is divided into four subbands and critically subsampled as shown in Fig. 1. Each coefficient represents a spatial area corresponding to approximately a 2 × 2 area of the original image. The low frequencies represent a bandwidth approximately corresponding to $0 < |\omega| < \pi/2$, whereas the high frequencies represent the band from $\pi/2 < |\omega| < \pi$. The four subbands arise from separable application of vertical and horizontal filters. The subbands labeled LH_1, HL_1, and HH_1 represent the finest scale wavelet coefficients. To obtain the next coarser scale of wavelet coefficients, the subband LL_1 is further decomposed and critically sampled as shown in Fig. 2. The process continues until some final scale is reached. Note that for each coarser scale, the coefficients represent a larger spatial area of the image but a narrower band of frequencies. At each scale, there are three subbands; the remaining lowest frequency subband is a representation of the information at all coarser scales. The issues involved in the design of the filters for the type of subband decomposition described above have been discussed by many authors and are not treated in this paper. Interested readers should consult [1], [6], [32], [35], in addition to references found in the bibliographies of the tutorial papers cited above.

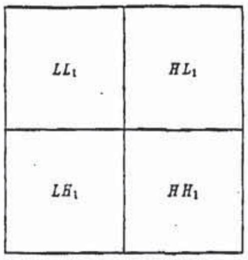


Fig. 1. First stage of a discrete wavelet transform: The image is divided into four subbands using separable filters. Each coefficient represents a spatial area corresponding to approximately a 2 × 2 area of the original picture. The low frequencies represent a bandwidth approximately corresponding to $0 < |\omega| < \pi$ 2, whereas the high frequencies represent the band from π '2 $< |\omega| < \pi$. The four subbands arise from separable application of vertical and horizontal filters.

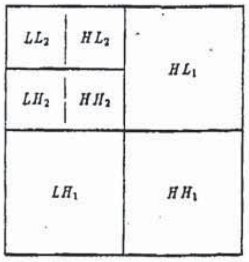


Fig. 2. A two-scale wavelet decomposition: The image is divided into four subbands using separable filters. Each coefficient in the subbands LL_2, LH_2, HL_2 and HH_2 represents a spatial area corresponding to approximately a 4 × 4 area of the original picture. The low frequencies at this scale represent a bandwidth approximately corresponding to $0 < |\omega| < \pi/4$, whereas the high frequencies represent the band from $\pi/4 < |\omega| < \pi/2$.

It is a matter of terminology to distinguish between a transform and a subband system as they are two ways of describing the same set of numerical operations from differing points of view. Let x be a column vector whose elements represent a scanning of the image pixels, let X be a column vector whose elements are the array of coefficients resulting from the wavelet transform or subband decomposition applied to x. From the transform point of view, X represents a linear transformation of x represented by the matrix W, i.e.,

$$X = Wx. \tag{1}$$

Although not actually computed this way, the effective filters that generate the subband signals from the original signal form basis functions for the transformation, i.e., the rows of W. Different coefficients in the same subband represent the projection of the entire image onto translates of a prototype subband filter, since from the subband point of view, they are simply regularly spaced different outputs of a convolution between the image and a subband filter. Thus, the basis functions for each coefficient in a given subband are simply translates of one another.

In subband coding systems [32], the coefficients from a given subband are usually grouped together for the purposes of designing quantizers and coders. Such a grouping suggests that statistics computed from a subband are in some sense representative of the samples in that subband. However this statistical grouping once again implicitly de-emphasizes the outliers, which tend to represent the most significant anomalies or edges. In this paper, the term "wavelet transform" is used because each wavelet coefficient is individually and deterministically compared to the same set of thresholds for the purpose of measuring significance. Thus, each coefficient is treated as a distinct, potentially important piece of data regardless of its scale, and no statistics for a whole subband are used in any form. The result is that the small number of "deterministically" significant fine scale coefficients are not obscured because of their "statistical" insignificance.

The filters used to compute the discrete wavelet transform in the coding experiments described in this paper are based on the 9-tap symmetric quadrature mirror filters (QMF) whose coefficients are given in [1]. This transformation has also been called a QMF-pyramid. These filters were chosen because in addition to their good localization properties, their symmetry allows for simple edge treatments, and they produce good results empirically. Additionally, using properly scaled coefficients, the transformation matrix for a discrete wavelet transform obtained using these filters is so close to unitary that it can be treated as unitary for the purpose of lossy compression. Since unitary transforms preserve L_2 norms, it makes sense from a numerical standpoint to compare all of the resulting transform coefficients to the same thresholds to assess significance.

III. Zerotrees of Wavelet Coefficients

In this section, an important aspect of low bit rate image coding is discussed: the coding of the positions of those coefficients that will be transmitted as nonzero values. Using scalar quantization followed by entropy coding, in order to achieve very low bit rates, i.e., less than 1 bit/pel, the probability of the most likely symbol after quantization—the zero symbol—must be extremely high. Typically, a large fraction of the bit budget must be spent on encoding the *significance map*, or the binary decision as to whether a sample, in this case a coefficient of a 2-D discrete wavelet transform, has a zero or nonzero quantized value. It follows that a significant improvement in encoding the significance map translates into a corresponding gain in compression efficiency.

A. Significance Map Encoding

To appreciate the importance of significance map encoding, consider a typical transform coding system where a decorrelating transformation is followed by an entropy-coded scalar quantizer. The following discussion is not intended to be a rigorous justification for significance map encoding, but merely to provide the reader with a sense of the relative coding costs of the position information contained in the significance map relative to amplitude and sign information.

A typical low-bit rate image coder has three basic components: a transformation, a quantizer and data compression, as shown in Fig. 3. The original image is passed through some transformation to produce transform coefficients. This transformation is considered to be lossless, although in practice this may not be the case exactly. The transform coefficients are then quantized to produce a stream of symbols, each of which corresponds to an index of a particular quantization bin. Note that virtually all of the information loss occurs in the quantization stage. The data compression stage takes the stream of symbols and attempts to losslessly represent the data stream as efficiently as possible.

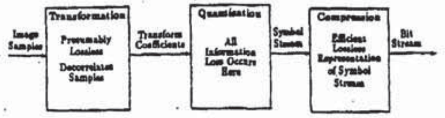


Fig. 3. A generic transform coder.

The goal of the transformation is to produce coefficients that are decorrelated. If we could, we would ideally like a transformation to remove all dependencies between samples. Assume for the moment that the transformation is doing its job so well that the resulting transform coefficients are not merely uncorrelated, but statistically independent. Also, assume that we have removed the mean and coded it separately so that the transform coefficients can be modeled as zero-mean, independent, although perhaps not identically distributed random variables. Furthermore, we might additionally constrain the model so that the probability density functions (PDF) for the coefficients are symmetric.

The goal is to quantize the transform coefficients so that the entropy of the resulting distribution of bin indexes is small enough so that the symbols can be entropy-coded at some target low bit rate, say for example 0.5 bits per pixel (bpp.). Assume that the quantizers will be symmetric midtread, perhaps nonuniform, quantizers, although different symmetric midtread quantizers may be used for different groups of transform coefficients. Letting the central bin be index 0, note that because of the symmetry, for a bin with a nonzero index magnitude, a positive or negative index is equally likely. In other words, for each nonzero index encoded, the entropy code is going to require at least one-bit for the sign. An entropy code can be designed based on modeling probabilities of bin indices as the fraction of coefficients in which the absolute value of a particular bin index occurs. Using this simple model, and assuming that the resulting symbols are independent, the entropy of the symbols H can be expressed as

$$H = -p \log_2 p - (1 - p) \log_2 (1 - p) + (1 - p)[1 + H_{NZ}], \qquad (2)$$

where p is the probability that a transform coefficient is quantized to zero, and H_{NZ} represents the conditional entropy of the absolute values of the quantized coefficients conditioned on them being nonzero. The first two terms in the sum represent the first-order binary entropy of the significance map, whereas the third term represents the conditional entropy of the distribution of nonzero values multiplied by the probability of them being nonzero. Thus, we can express the true cost of encoding the actual symbols as follows:

$$\text{Total Cost} = \text{Cost of Significance Map} + \text{Cost of Nonzero Values.} \quad (3)$$

Returning to the model, suppose that the target is $H = 0.5$. What is the minimum probability of zero achievable? Consider the case where we only use a 3-level quantizer, i.e. $H_{NZ} = 0$. Solving for p provides a lower bound on the probability of zero given the independence assumption

$$p_{\min}(H_{NZ} = 0, H = 0.5) = 0.916. \quad (4)$$

In this case, under the most ideal conditions, 91.6% of the coefficients must be quantized to zero. Furthermore, 83% of the bit budget is used in encoding the significance map. Consider a more typical example where $H_{NZ} = 4$, the minimum probability of zero is

$$p_{\min}(H_{NZ} = 4, H = 0.5) = 0.954. \quad (5)$$

In this case, the probability of zero must increase, while the cost of encoding the significance map is still 54% of the cost.

As the target rate decreases, the probability of zero increases, and the fraction of the encoding cost attributed to the significance map increases. Of course, the independence assumption is unrealistic and in practice, there are often additional dependencies between coefficients that can be exploited to further reduce the cost of encoding the significance map. Nevertheless, the conclusion is that no matter how optimal the transform, quantizer or entropy coder, under very typical conditions, the cost of determining the positions of the few significant coefficients represents a significant portion of the bit budget at low rates, and is likely to become an increasing fraction of the total cost as the rate decreases. As will be seen, by employing an image model based on an extremely simple and easy to satisfy hypothesis, we can efficiently encode significance maps of wavelet coefficients.

B. Compression of Significance Maps using Zerotrees of Wavelet Coefficients

To improve the compression of significance maps of wavelet coefficients, a new data structure called a *zerotree* is defined. A wavelet coefficient x is said to be *insignificant* with respect to a given threshold T if $|x| < T$. The zerotree is based on the hypothesis that if a wavelet coefficient at a coarse scale is insignificant with respect to a given threshold T, then *all* wavelet coefficients of the same orientation in the same spatial location at finer scales are likely to be insignificant with respect to T. Empirical evidence suggests that this hypothesis is often true.

More specifically, in a hierarchical subband system, with the exception of the highest frequency subbands, every coefficient at a given scale can be related to a set of coefficients at the next finer scale of similar orientation. The coefficient at the coarse scale is called the *parent*, and all coefficients corresponding to the same spatial location at the next finer scale of similar orientation are called *children*. For a given parent, the set of all coefficients at all finer scales of similar orientation corresponding to the same location are called *descendants*. Similarly, for a given child, the set of coefficients at all coarser scales of similar orientation corresponding to the same location are called *ancestors*. For a QMF-pyramid subband decomposition, the parent-child dependencies are shown in Fig. 4. A wavelet tree descending from a coefficient in subband $HH3$ is also seen in Fig. 4. With the exception of the lowest frequency subband, all parents have four children. For the lowest frequency subband, the parent-child relationship is defined such that each parent node has three children.

A scanning of the coefficients is performed in such a way that no child node is scanned before its parent. For an N-scale transform, the scan begins at the lowest frequency subband, denoted as LL_N, and scans subbands HL_N, LH_N, and HH_N, at which point it moves on to scale $N - 1$, etc. The scanning pattern for a 3-scale QMF-pyramid can be seen in Fig. 5. Note that each coefficient within a given subband is scanned before any coefficient in the next subband.

Given a threshold level T to determine whether or not a coefficient is significant, a coefficient x is said to be an element of a *zerotree* for threshold T if itself and *all* of its descendents are insignificant with respect to T. An element of a zerotree for threshold T is a *zerotree root* if it is not the descendant of a previously found zerotree root for threshold T, i.e., it is not *predictably insignificant* from the discovery of a zerotree root at a coarser scale at the same threshold. A zerotree root is encoded with a special symbol indicating that the insignificance of the coefficients at finer scales is completely predictable. The significance map can be efficiently represented as a string of symbols from a 3-symbol alphabet which is then entropy-coded. The three symbols used are 1) zerotree root, 2) isolated zero, which means that the coefficient is insignificant but has some significant descendant, and 3) significant. When encoding the finest scale coefficients, since coefficients have no children, the symbols in the string come from a 2-symbol alphabet, whereby the zerotree symbol is not used.

As will be seen in Section IV, in addition to encoding the significance map, it is useful to encode the sign of significant coefficients along with the significance map. Thus, in practice, four symbols are used: 1) zerotree root, 2) isolated zero, 3) positive significant, and 4) negative significant. This minor addition will be useful for embedding. The flow chart for the decisions made at each coefficient are shown in Fig. 6.

Note that it is also possible to include two additional symbols such as "positive/negative significant, but descendants are zerotrees" etc. In practice, it was found that

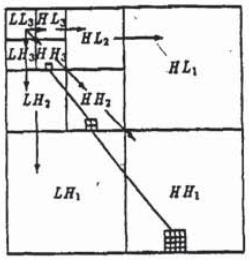


Fig. 4. Parent-child dependencies of subbands: Note that the arrow points from the subband of the parents to the subband of the children. The lowest frequency subband is the top left, and the highest frequency subband is at the bottom right. Also shown is a wavelet tree consisting of all of the descendents of a single coefficient in subband *HH*3. The coefficient in *HH*3 is a zerotree root if it is insignificant and *all* of its descendants are insignificant.

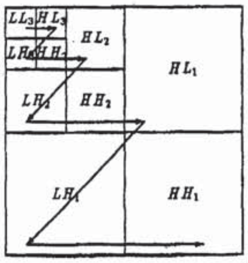


Fig. 5. Scanning order of the subbands for encoding a significance map: Note that parents must be scanned before children. Also note that all positions in a given subband are scanned before the scan moves to the next subband.

at low bit rates, this addition often increases the cost of coding the significance map. To see why this may occur, consider that there is a cost associated with partitioning the set of positive (or negative) significant samples into those whose descendents are zerotrees and those with significant descendants. If the cost of this decision is C bits, but the cost of encoding a zerotree is less than $C/4$ bits, then it is more efficient to code four zerotree symbols separately than to use additional symbols.

Zerotree coding reduces the cost of encoding the significance map using self-similarity. Even though the image has been transformed using a decorrelating transform the *occurrences* of insignificant coefficients are not independent events. More traditional techniques employing transform coding typically encode the binary map via some form of run-length encoding [30]. Unlike the zerotree symbol, which is a single "terminating" symbol and applies to all tree-depths, run-length encoding requires a symbol for each run-length which much be encoded. A technique that is closer in spirit to the zerotrees is the end-of-block (EOB) symbol used in JPEG [30], which is also a "terminating" symbol indicating that all remaining DCT coefficients in the block are quantized to zero. To see why zerotrees may provide an advantage over EOB symbols, consider that a zerotree represents the insignificance information in a given orientation over an approximately square spatial area at all finer scales up to and including the scale of the zerotree root. Because the wavelet transform is a hierarchical representation, varying the scale in which a zerotree root occurs automatically adapts the spatial area over which insignificance is represented. The EOB symbol, however, always represents insignificance over the same spatial area, although the number of frequency bands within that spatial area varies. Given a fixed block size, such as 8 × 8, there is exactly one scale in the wavelet transform in which if a zerotree root is found at that scale, it corresponds to the same spatial area as a block of the DCT. If a zerotree root can be identified at a coarser scale, then the insignificance pertaining to that orientation can be predicted over a larger area. Similarly, if the zerotree root does not occur at this scale, then looking for zerotrees at finer scales represents a hierarchical divide and conquer approach to searching for one or more smaller areas of insignificance over the same spatial regions as the DCT block size. Thus, many more coefficients can be predicted in smooth areas where a root typically occurs at a coarse scale. Furthermore, the zerotree approach can isolate interesting non-zero details by immediately eliminating large insignificant regions from consideration.

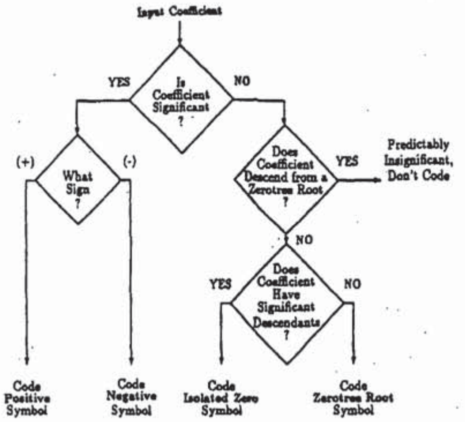

Input Coefficient
Is Coefficient Significant ?
YES
NO
What Sign ?
(+)
(-)
Does Coefficient Descend from a Zerotree Root ?
YES
Predictably Insignificant, Don't Code
NO
Does Coefficient Have Significant Descendants ?
YES
NO
Code Positive Symbol
Code Negative Symbol
Code Isolated Zero Symbol
Code Zerotree Root Symbol

Fig. 6. Flow chart for encoding a coefficient of the significance map.

Note that this technique is quite different from previous attempts to exploit self-similarity in image coding [19] in that it is far easier to predict insignificance than to predict significant detail across scales. The zerotree approach was developed in recognition of the difficulty in achieving meaningful bit rate reductions for significant coefficients via additional prediction. Instead, the focus here is on reducing the cost of encoding the significance map so that, for a given bit budget, more bits are available to encode expensive significant coefficients. In practice, a large fraction of the insignificant coefficients are efficiently encoded as part of a zerotree.

A similar technique has been used by Lewis and Knowles (LK) [15], [16]. In that work, a "tree" is said to be zero if its energy is less than a perceptually based threshold. Also, the "zero flag" used to encode the tree is not entropy-coded. The present work represents an improvement that allows for embedded coding for two reasons. Applying a deterministic threshold to determine significance results in a zerotree symbol which guarantees that no descendant of the root has a magnitude larger than the threshold. As a result, there is no possibility of a significant coefficient being obscured by a statistical energy measure. Furthermore, the zerotree symbol developed in this paper is part of an alphabet for entropy coding the significance map which further improves its compression. As will be discussed subsequently, it is the first property that makes this method of encoding a significance map useful in conjunction with successive-approximation. Recently, a promising technique representing a compromise between the EZW algorithm and the LK coder has been presented in [34].

C. Interpretation as a Simple Image Model

The basic hypothesis—if a coefficient at a coarse scale is insignificant with respect to a threshold then all of its descendants, as defined above, are also insignificant—can be interpreted as an extremely general image model. One of the aspects that seems to be common to most models used to describe images is that of a "decaying spectrum." For example, this property exists for both stationary autoregressive models, and non-stationary fractal, or "nearly-1/f" models, as implied by the name which refers to a generalized spectrum [33]. The model for the zerotree hypothesis is even more general than "decaying spectrum" in that it allows for some deviations to "decaying spectrum" because it is linked to a specific threshold. Consider an example where the threshold is 50, and we are considering a coefficient of magnitude 30, and whose largest descendant has a magnitude of 40. Although a higher frequency descendant has a larger magnitude (40) than the coefficient under consideration (30), i.e., the "decaying spectrum" hypothesis is violated, the coefficient under consideration can still be represented using a zerotree root since the whole tree is still insignificant (magnitude less than 50). Thus, assuming the more common image models have some validity, the zerotree hypothesis should be satisfied easily and extremely often. For those instances where the hypothesis is violated, it is safe to say that an *informative*, i.e., unexpected, event has occurred, and we should expect the cost of representing this event to be commensurate with its self-information.

It should also be pointed out that the improvement in encoding significance maps provided by zerotrees is specifically *not* the result of exploiting any linear dependencies between coefficients of different scales that were not removed in the transform stage. In practice, the linear correlation between the values of parent and child wavelet coefficients has been found to be extremely small, implying that the wavelet transform is doing an excellent job of producing nearly uncorrelated coefficients. However, there is likely additional dependency between the *squares* (or magnitudes) of parents and children. Experiments run on about 30 images of all different types, show that the correlation coefficient between the square of a child and the square of its parent tends to be between 0.2 and 0.6 with a string concentration around 0.35. Although this dependency is difficult to characterize in general for most images, even without access to specific statistics, it is reasonable to *expect* the magnitude of a child to be smaller than the magnitude of its parent. In other words, it can be reasonably conjectured based on experience with real-world images, that had we known the details of the statistical dependencies, and computed an "optimal" estimate, such as the conditional expectation of the child's magnitude given the parent's magnitude, that the "optimal" estimator would, with very high probability, predict that the child's magnitude would be the smaller of the two. Using only this mild assumption, based on an inexact statistical characterization, given a fixed threshold, and conditioned on the knowledge that a parent is insignificant with respect to the threshold, the "optimal" estimate of the significance of the rest of the descending wavelet tree is that it is entirely insignificant with respect to the same threshold, i.e., a zerotree. On the other hand, if the parent is significant, the "optimal" estimate of the significance of descendants is highly dependent on the details of the estimator whose knowledge would require more detailed information about the statistical nature of the image. Thus, under this mild assumption, using zerotrees to predict the insignificance of wavelet coefficients at fine scales given the insignificance of a root at a coarse scale is more likely to be successful in the absence of additional information than attempting to predict significant detail across scales.

This argument can be made more concrete. Let x be a child of y, where x and y are zero-mean random variables, whose probability density functions (PDF) are related as

$$p_x(x) = ap_y(ax), \qquad a > 1. \tag{6}$$

This states that random variables x and y have the same PDF shape, and that

$$\sigma_y^2 = a^2\sigma_x^2. \tag{7}$$

Assume further that x and y are uncorrelated, i.e.,

$$E[xy] = 0. \tag{8}$$

Note that nothing has been said about treating the subbands as a group, or as stationary random processes, only that there is a similarity relationship between random variables of parents and children. It is also reasonable because for intermediate subbands a coefficient that is a child with respect to one coefficient is a parent with respect to others; the PDF of that coefficient should be the same in either case. Let $u = x^2$ and $v = y^2$. Suppose that u and

v are correlated with correlation coefficient ρ. We have the following relationships:

$$E[u] = \sigma_x^2 \tag{9}$$

$$E[v] = \sigma_y^2 \tag{10}$$

$$\sigma_u^2 = E[x^4] - \sigma_x^4 \tag{11}$$

$$\sigma_v^2 = E[y^4] - \sigma_y^4. \tag{12}$$

Notice in particular that

$$\sigma_v^2 = a^4 \sigma_u^2. \tag{13}$$

Using a well known result, the expression for the best linear unbiased estimator (BLUE) of u given v to minimize error variance is given by

$$\hat{u}_{\text{BLUE}}(v) = E[u] - \rho \frac{\sigma_u}{\sigma_v} (E[v] - v) \tag{14}$$

$$= \frac{1 - \rho}{a^2} \sigma_y^2 + \rho \frac{v}{a^2}. \tag{15}$$

If it is observed that the magnitude of the parent is below the threshold T, i.e., $v = y^2 < T^2$, then the BLUE can be upper bounded by

$$\hat{u}_{\text{BLUE}}(v | v < T^2) < \frac{1 - \rho}{a^2} \sigma_y^2 + \rho \frac{T^2}{a^2}. \tag{16}$$

Consider two cases a) $T \geq \sigma_y$ and b) $T < \sigma_y$. In case (a), we have

$$\hat{u}_{\text{BLUE}}(v | v < T^2) \leq \frac{T^2}{a^2} < T^2, \tag{17}$$

which implies that the BLUE of x^2 given $|y| < T$ is less than T^2, for *any* ρ, including $\rho = 0$. In case (b), we can only upper bound the right hand side of (16) by T^2 if ρ exceeds the lower bound

$$\rho \geq \frac{1 - \frac{a^2 T^2}{\sigma_y^2}}{1 - \frac{T^2}{\sigma_y^2}} \triangleq \rho_0. \tag{18}$$

Of course, a better nonlinear estimate might yield different results, but the above analysis suggests that for threshold exceeding the standard deviation of the parent, which by (6) exceeds the standard deviation of all descendants, if it is observed that a parent is insignificant with respect to the threshold, then, using the above BLUE, the estimates for the magnitudes of all descendants is that they are less than the threshold, and a zerotree is expected regardless of the correlation between squares of parents and squares of children. As the threshold decreases, more correlation is required to justify *expecting* a zerotree to occur. Finally, since the lower bound $\rho_0 \rightarrow 1$ as $T \rightarrow 0$, as the threshold is reduced, it becomes increasingly difficult to expect zerotrees to occur, and more knowledge of the particular statistics are required to make inferences. The implication of this analysis is that at very low bit rates, where the probability of an insignificant sample must be high and thus, the significance threshold T must also be large, expecting the occurrence of zerotrees and encoding significance maps using zerotree coding is reasonable without even knowing the statistics. However, letting T decrease, there is some point below which the advantage of zerotree coding diminishes, and this point is dependent on the specific nature of higher order dependencies between parents and children. In particular, the stronger this dependence, the more T can be decreased while still retaining an advantage using zerotree coding. Once again, this argument is not intended to "prove" the optimality of zerotree coding, only to suggest a rationale for its demonstrable success.

D. Zerotree-Like Structures in Other Subband Configurations

The concept of predicting the insignificance of coefficients from low frequency to high frequency information corresponding to the same spatial localization is a fairly general concept and not specific to the wavelet transform configuration shown in Fig. 4. Zerotrees are equally applicable to quincunx wavelets [2], [13], [23], [29], in which case each parent would have two children instead of four, except for the lowest frequency, where parents have a single child.

Also, a similar approach can be applied to linearly spaced subband decompositions, such as the DCT, and to other more general subband decompositions, such as wavelet packets [5] and Laplacian pyramids [4]. For example, one of many possible parent-child relationship for linearly spaced subbands can be seen in Fig. 7. Of course, with the use of linearly spaced subbands, zerotree-like coding loses its ability to adapt the spatial extent of the insignificance prediction. Nevertheless, it is possible for zerotree-like coding to outperform EOB-coding since more coefficients can be predicted from the subbands along the diagonal. For the case of wavelet packets, the situation is a bit more complicated, because a wider range of tilings of the "space-frequency" domain are possible. In that case, it may not always be possible to define similar parent-child relationships because a high-frequency coefficient may in fact correspond to a larger spatial area than a co-located lower frequency coefficient. On the other hand, in a coding scheme such as the "best-basis" approach of Coifman *et al.* [5], had the image-dependent best basis resulted in such a situation, one wonders if the underlying hypothesis—that magnitudes of coefficients tend to decay with frequency—would be reasonable anyway. These zerotree-like extensions represent interesting areas for further research.

IV. Successive-Approximation

The previous section describes a method of encoding significance maps of wavelet coefficients that, at least empirically, seems to consistently produce a code with a lower bit rate than either the empirical first-order entropy,

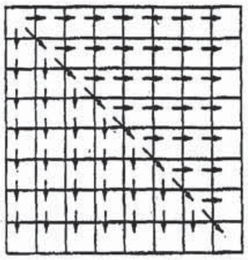


Fig. 7. Parent–child dependencies for linearly spaced subbands systems, such as the DCT. Note that the arrow points from the subband of the parents to the subband of the children. The lowest frequency subband is the top left, and the highest frequency subband is at the bottom right.

or a run-length code of the significance map. The original motivation for employing successive-approximation in conjunction with zerotree coding was that since zerotree coding was performing so well encoding the significance map of the wavelet coefficients, it was hoped that more efficient coding could be achieved by zerotree coding more significance maps.

Another motivation for successive-approximation derives directly from the goal of developing an embedded code analogous to the binary-representation of an approximation to a real number. Consider the wavelet transform of an image as a mapping whereby an amplitude exists for each coordinate in scale-space. The scale-space coordinate system represents a coarse-to-fine "logarithmic" representation of the domain of the function. Taking the coarse-to-fine philosophy one-step further, successive-approximation provides a coarse-to-fine, multiprecision "logarithmic" representation of amplitude information, which can be thought of as the range of the image function when viewed in the scale-space coordinate system defined by the wavelet transform. Thus, in a very real sense, the EZW coder generates a representation of the image that is coarse-to-fine in both the domain and range simultaneously.

A. Successive-Approximation Entropy-Coded Quantization

To perform the embedded coding, successive-approximation quantization (SAQ) is applied. As will be seen, SAQ is related to bit-plane encoding of the magnitudes. The SAQ sequentially applies a sequence of thresholds $T_0, \cdots, T_{N-1}$ to determine significance, where the thresholds are chosen so that $T_i = T_{i-1}/2$. The initial threshold T_0 is chosen so that $|X_j| < 2T_0$ for all transform coefficients x_j.

During the encoding (and decoding), two separate lists of wavelet coefficients are maintained. At any point in the process, the *dominant* list contains the *coordinates* of those coefficients that have not yet been found to be significant in the same relative order as the initial scan. This scan is such that the subbands are ordered, and within each subband, the set of coefficients are ordered. Thus, using the ordering of the subbands shown in Fig. 5, all coefficients in a given subband appear on the initial dominant list prior to coefficients in the next subband. The *subordinate* list contains the *magnitudes* of those coefficients that have been found to be significant. For each threshold, each list is scanned once.

During a *dominant pass*, coefficients with coordinates on the dominant list, i.e., those that have not yet been found to be significant, are compared to the threshold T_i to determine their significance, and if significant, their sign. This significance map is then zerotree coded using the method outlined in Section III. Each time a coefficient is encoded as significant, (positive or negative), its magnitude is appended to the subordinate list, and the coefficient in the wavelet transform array is set to zero so that the significant coefficient does not prevent the occurrence of a zerotree on future dominant passes at smaller thresholds.

A dominant pass is followed by a *subordinate pass* in which all coefficients on the subordinate list are scanned and the specifications of the magnitudes available to the decoder are refined to an additional bit of precision. More specifically, during a subordinate pass, the *width* of the effective quantizer step size, which defines an uncertainty interval for the true magnitude of the coefficient, is cut in half. For each magnitude on the subordinate list, this refinement can be encoded using a binary alphabet with a "1" symbol indicating that the true value falls in the upper half of the old uncertainty interval and a "0" symbol indicating the lower half. The string of symbols from this binary alphabet that is generated during a subordinate pass is then entropy coded. Note that prior to this refinement, the width of the uncertainty region is exactly equal to the current threshold. After the completion of a subordinate pass the magnitudes on the subordinate list are sorted in decreasing magnitude, to the extent that the decoder has the information to perform the same sort.

The process continues to alternate between dominant passes and subordinate passes where the threshold is halved before each dominant pass. (In principle one could divide by other factors than 2. This factor of 2 was chosen here because it has nice interpretations in terms of bit plane encoding and numerical precision in a familiar base 2, and good coding results were obtained).

In the decoding operation, each decoded symbol, both during a dominant and a subordinate pass, refines and reduces the width of the uncertainty interval in which the true value of the coefficient (or coefficients, in the case of a zerotree root) may occur. The reconstruction value used can be anywhere in that uncertainty interval. For minimum mean-square error distortion, one could use the centroid of the uncertainty region using some model for the PDF of the coefficients. However, a practical approach, which is used in the experiments, and is also MINMAX optimal, is to simply use the center of the uncertainty interval as the reconstruction value.

The encoding stops when some target stopping condition is met, such as when the bit budget is exhausted. The encoding can cease at any time and the resulting bit stream contains all lower rate encodings. Note, that if the bit stream is truncated at an arbitrary point, there may be bits at the end of the code that do not decode to a valid symbol since a codeword has bene truncated. In that case, these bits do not reduce the width of an uncertainty interval or any distortion function. In fact, it is very likely that the first L bits of the bit stream will produce exactly the same image as the first $L + 1$ bits which occurs if the additional bit is insufficient to complete the decoding of another symbol. Nevertheless, terminating the decoding of an embedded bit stream at a specific point in the bit stream produces exactly the same image that would have resulted had that point been the initial target rate. This ability to cease encoding or decoding anywhere is extremely useful in systems that are either rate-constrained or distortion-constrained. A side benefit of the technique is that an operational rate vs. distortion plot for the algorithm can be computed on-line.

B. Relationship to Bit Plane Encoding

Although the embedded coding system described here is considerably more general and more sophisticated than simple bit-plane encoding, consideration of the relationship with bit-plane encoding provides insight into the success of embedded coding.

Consider the successive-approximation quantizer for the case when all thresholds are powers of two, and all wavelet coefficients are integers. In this case, for each coefficient that eventually gets coded as significant, the sign and *bit position* of the most-significant binary digit (MSBD) are measured and encoded during a dominant pass. For example, consider the 10-bit representation of the number 41 as 0000101001. Also, consider the binary digits as a sequence of binary decisions in a binary tree. Proceeding from left to right, if we have not yet encountered a "1," we expect the probability distribution for the next digit to be strongly biased toward "0." The digits to the left and including the MSBD are called the *dominant bits*, and are measured during dominant passes. After the MSBD has been encountered, we expect a more random and much less biased distribution between a "0" and a "1," although we might still expect $P(0) > P(1)$ because most PDF models for transform coefficients decay with amplitude. Those binary digits to the right of the MSBD are called the ***subordinate bits*** and are measured and encoded during the subordinate pass. A zeroth-order approximation suggests that we should expect to pay close to one bit per "binary digit" for subordinate bits, while dominant bits should be far less expensive.

By using successive-approximation beginning with the largest possible threshold, where the probability of zero is extremely close to one, and by using zerotree coding, whose efficiency increases as the probability of zero increases, we should be able to code dominant bits with very few bits, since they are most often part of a zerotree.

In general, the thresholds need not be powers of two. However, by factoring out a constant mantissa, M, the starting threshold T_0 can be expressed in terms of a threshold that is a power of two

$$T_0 = M2^E, \tag{19}$$

where the exponent E is an integer, in which case, the dominant and subordinate bits of appropriately scaled wavelet coefficients are coded during dominant and subordinate passes, respectively.

C. Advantage of Small Alphabets for Adaptive Arithmetic Coding

Note that the particular encoder alphabet used by the arithmetic coder at any given time contains either 2, 3, or 4 symbols depending whether the encoding is for a subordinate pass, a dominant pass with no zerotree root symbol, or a dominant pass with the zerotree root symbol. This is a real advantage for adapting the arithmetic coder. Since there are never more than four symbols, all of the possibilities typically occur with a reasonably measurable frequency. This allows an adaptation algorithm with a short memory to learn quickly and constantly track changing symbol probabilities. This adaptivity accounts for some of the effectiveness of the overall algorithm. Contrast this with the case of a large alphabet, as is the case in algorithms that do not use successive approximation. In that case, it takes many events before an adaptive entropy coder can reliably estimate the probabilities of unlikely symbols (see the discussion of the zero-frequency problem in [3]). Furthermore, these estimates are fairly unreliable because images are typically statistically nonstationary and local symbol probabilities change from region to region.

In the practical coder used in the experiments, the arithmetic coder is based on [31]. In arithmetic coding, the encoder is separate from the model, which in [31], is basically a histogram. During the dominant passes, simple Markov conditioning is used whereby one of four histograms is chosen depending on 1) whether the previous coefficient in the scan is known to be significant, and 2) whether the parent is known to be significant. During the subordinate passes, a single histogram is used. Each histogram entry is initialized to a count of one. After encoding each symbol, the corresponding histogram entry is incremented. When the sum of all the counts in a histogram reaches the maximum count, each entry is incremented and integer divided by two, as described in [31]. It should be mentioned, that for practical purposes, the coding gains provided by using this simple Markov conditioning may not justify the added complexity and using a single histogram strategy for the dominant pass performs almost as well (0.12 dB worse for Lena at 0.25 bpp.). The choice of maximum histogram count is probably more critical, since that controls the learning rate for the adaptation. For the experimental results presented, a maximum count of 256 was used, which provides an intermediate tradeoff between the smallest possible probability, which is the reciprocal of the maximum count, and the learning rate, which is faster with a smaller maximum histogram count.

D. Order of Importance of the Bits

Although importance is a subjective term, the order of processing used in EZW implicitly defines a precise ordering of importance that is tied to, in order, *precision*, *magnitude*, *scale*, and *spatial location* as determined by the initial dominant list.

The primary determination of ordering importance is the numerical *precision* of the coefficients. This can be seen in the fact that the uncertainty intervals for the magnitude of all coefficients are refined to the same precision before the uncertainty interval for any coefficient is refined further.

The second factor in the determination of importance is *magnitude*. Importance by magnitude manifests itself during a dominant pass because prior to the pass, all coefficients are insignificant and presumed to be zero. When they are found to be significant, they are all assumed to have the same magnitude, which is greater than the magnitudes of those coefficients that remain insignificant. Importance by magnitude manifests itself during a subordinate pass by the fact that magnitudes are refined in descending order of the center of the uncertainty intervals, i.e., the decoder's interpretation of the magnitude.

The third factor, *scale*, manifests itself in the *a priori* ordering of the subbands on the initial dominant list. Until the significance of the magnitude of a coefficient is discovered during a dominant pass, coefficients in coarse scales are tested for significance before coefficients in fine scales. This is consistent with prioritization by the decoder's version of magnitude since for all coefficients not yet found to be significant, the magnitude is presumed to be zero.

The final factor, *spatial location*, merely implies that two coefficients that cannot yet be distinguished by the decoder in terms of either precision, magnitude, or scale, have their relative importance determined arbitrarily by the initial scanning order of the subband containing the two coefficients.

In one sense, this embedding strategy has a strictly nonincreasing operational distortion-rate function for the distortion metric defined to be the sum of the widths of the uncertainty intervals of all of the wavelet coefficients. Since a discrete wavelet transform is an invertible representation of an image, a distortion function defined in the wavelet transform domain is also a distortion function defined on the image. This distortion function is also not without a rational foundation for low-bit rate coding, where noticeable artifacts must be tolerated, and perceptual metrics based on just-noticeable differences (JND's) do not always predict which artifacts human viewers will prefer. Since minimizing the widths of uncertainty intervals minimizes the largest possible errors, artifacts, which result from numerical errors large enough to exceed perceptible thresholds, are minimized. Even using this distortion function, the proposed embedding strategy is not optimal, because truncation of the bit stream in the middle of a pass causes some uncertainty intervals to be twice as large as others.

Actually, as it has been described thus far, EZW is unlikely to be optimal for *any* distortion function. Notice that in (19), dividing the thresholds by two simply decrements E leaving M unchanged. While there must exist an optimal starting M which minimizes a given distortion function, how to find this optimum is still an open question and seems highly image dependent. Without knowledge of the optimal M and being forced to choose it based on some other consideration, with probability one, either increasing or decreasing M would have produced an embedded code which has a lower distortion for the same rate. Despite the fact that without trial and error optimization for M, EZW is probably suboptimal, it is nevertheless quite effective in practice.

Note also that using the width of the uncertainty interval as a distance metric is exactly the same metric used in finite-precision fixed-point approximations of real numbers. Thus, the embedded code can be seen as an "image" generalization of finite-precision fixed-point approximations of real numbers.

E. Relationship to Priority-Position Coding

In a technique based on a very similar philosophy, Huang *et al.* discusses a related approach to embedding, or ordering the information in importance, called priority-position coding (PPC) [10]. They prove very elegantly that the entropy of a source is equal to the average entropy of a particular ordering of that source plus the average entropy of the position information necessary to reconstruct the source. Applying a sequence of decreasing thresholds, they attempt to sort by amplitude all of the DCT coefficients for the entire image based on a partition of the range of amplitudes. For each coding pass, they transmit the significance map which is arithmetically encoded. Additionally, when a significant coefficient is found they transmit its value to its full precision. Like the EZW algorithm, PPC implicitly defines importance with respect to the magnitudes of the transform coefficients. In one sense, PPC is a generalization of the successive-approximation method presented in this paper, because PPC allows more general partitions of the amplitude range of the transform coefficients. On the other hand, since PPC sends the value of a significant coefficient to full precision, its protocol assigns a greater importance to the least significant bit of a significant coefficient than to the identification of new significant coefficients on next PPC pass. In contrast, as a top priority, EZW tries to reduce the width of the largest uncertainty interval in all coefficients before increasing the precision further. Additionally, PPC makes no attempt to predict insignificance from low frequency to high frequency, relying solely on the arithmetic coding to encode the significance map. Also unlike EZW, the probability estimates needed for the arithmetic coder were derived via training on an image database instead of adapting to the image itself. It would be interesting to experiment with variations which combine advantages of EZW (wavelet transforms, zerotree coding, importance defined by a decreasing sequence of uncertainty intervals, and adaptive arithmetic coding using small alphabets) with the more general approach to partitioning the range of amplitudes found in PPC. In practice, however, it is unclear whether the finest grain partitioning of the amplitude range provides any coding gain, and there is certainly a much higher computational cost associated with more passes. Additionally, with the exception of the last few low-amplitude passes, the coding results reported in [10] did use power-of-two amplitudes to define the partition suggesting that, in practice, using finer partitioning buys little coding gain.

V. A Simple Example

In this section, a simple example will be used to highlight the order of operations used in the EZW algorithm. Only the string of symbols will be shown. The reader interested in the details of adaptive arithmetic coding is referred to [31]. Consider the simple 3-scale wavelet transform of an 8 × 8 image. The array of values is shown in Fig. 8. Since the largest coefficient magnitude is 63, we can choose our initial threshold to be anywhere in (31.5, 63]. Let $T_0 = 32$. Table I shows the processing on the first dominant pass. The following comments refer to Table I:

1) The coefficient has magnitude 63 which is greater than the threshold 32, and is positive so a positive symbol is generated. After decoding this symbol, the decoder knows the coefficient in the interval [32, 64) whose center is 48.

2) Even though the coefficient 31 is insignificant with respect to the threshold 32, it has a significant descendant two generations down in subband *LH*1 with magnitude 47. Thus, the symbol for an isolated zero is generated.

3) The magnitude 23 is less than 32 and all descendants which include (3, −12, -14, 8) in subband *HH*2 and all coefficients in subband *HH*1 are insignificant. A zerotree symbol is generated, and no symbol will be generated for any coefficient in subbands *HH*2 and *HH*1 during the current dominant pass.

4) The magnitude 10 is less than 32 and all descendants (-12, 7, 6, -1) also have magnitudes less than 32. Thus a zerotree symbol is generated. Notice that this tree has a violation of the "decaying spectrum" hypothesis since a coefficient (-12) in subband *HL*1 has a magnitude greater than its parent (10). Nevertheless, the entire tree has magnitude less than the threshold 32 so it is still a zerotree.

5) The magnitude 14 is insignificant with respect to 32. Its children are (-1, 47, -3, 2). Since its child with magnitude 47 is significant, an isolated zero symbol is generated.

63	−34	49	10	7	13	−12	7
−31	23	14	−13	3	4	6	−1
15	14	3	−12	5	−7	3	9
−9	−7	−14	8	4	−2	3	2
−5	9	−1	47	4	6	−2	2
3	0	−3	2	3	−2	0	4
2	−3	6	−4	3	6	3	6
5	11	5	6	0	3	−4	4

Fig. 8. Example of 3-scale wavelet transform of an 8 × 8 image.

TABLE I

Processing of First Dominant Pass at Threshold $T = 32$. Symbols are POS for Positive Significant, NEG for Negative Significant, IZ for Isolated Zero, ZTR for Zerotree Root, and Z for a Zero when There are no Children. The Reconstruction Magnitudes are Taken as the Center of the Uncertainty Interval

Comment	Subband	Coefficient Value	Symbol	Reconstruction Value
(1)	LL3	63	POS	48
	HL3	-34	NEG	-48
(2)	LH3	-31	IZ	0
(3)	HH3	23	ZTR	0
	HL2	49	POS	48
(4)	HL2	10	ZTR	0
	HL2	14	ZTR	0
	HL2	-13	ZTR	0
	LH2	15	ZTR	0
(5)	LH2	14	IZ	0
	LH2	-9	ZTR	0
	LH2	-7	ZTR	0
(6)	HL1	7	Z	0
	HL1	13	Z	0
	HL1	3	Z	0
	HL1	4	Z	0
	LH1	-1	Z	0
(7)	LH1	47	POS	48
	LH1	-3	Z	0
	LH1	-2	Z	0

6) Note that no symbols were generated from subband *HH*2 which would ordinarily precede subband *HL*1 in the scan. Also note that since subband *HL*1 has no descendants, the entropy coding can resume using a 3-symbol alphabet where the IZ and ZTR symbols are merged into the *Z* (zero) symbol.

7) The magnitude 47 is significant with respect to 32. Note that for the future dominant passes, this position will be replaced with the value 0, so that for the next dominant pass at threshold 16, the parent of this coefficient, which has magnitude 14, can be coded using a zerotree root symbol.

During the first dominant pass, which used a threshold of 32, four significant coefficients were identified. These coefficients will be refined during the first subordinate pass. Prior to the first subordinate pass, the uncertainty interval for the magnitudes of all of the significant coefficients is the interval [32, 64). The first subordinate pass will refine these magnitudes and identify them as being either in interval [32, 48), which will be encoded with the symbol "0," or in the interval [48, 64), which will be encoded with the symbol "1." Thus, the decision boundary is the magnitude 48. It is no coincidence that these symbols are exactly the first bit to the right of the MSBD in the binary representation of the magnitudes. The order

TABLE II
PROCESSING OF THE FIRST SUBORDINATE PASS. MAGNITUDES ARE PARTITIONED INTO THE UNCERTAINTY INTERVALS [32,48) AND [48,64), WITH SYMBOLS "0" AND "1" RESPECTIVELY

Coefficient Magnitude	Symbol	Reconstruction Magnitude
63	1	56
34	0	40
49	1	56
47	0	40

of operations in the first subordinate pass is illustrated in Table II.

The first entry has magnitude 63 and is placed in the upper interval whose center is 56. The next entry has magnitude 34, which places it in the lower interval. The third entry 49 is in the upper interval, and the fourth entry 47 is in the lower interval. Note that in the case of 47, using the center of the uncertainty interval as the reconstruction value, when the reconstruction value is changed from 48 to 40, the reconstruction error actually increases from 1 to 7. Nevertheless, the uncertainty interval for this coefficient decreases from width 32 to width 16. At the conclusion of the processing of the entries on the subordinate list corresponding to the uncertainty interval [32, 64), these magnitudes are reordered for future subordinate passes in the order (63, 49, 34, 47). Note that 49 is moved ahead of 34 because from the decoder's point of view, the reconstruction values 56 and 40 are distinguishable. However, the magnitude 34 remains ahead of magnitude 47 because as far as the decoder can tell, both have magnitude 40, and the initial order, which is based first on importance by scale, has 34 prior to 47.

The process continues on to the second dominant pass at the new threshold of 16. During this pass, only those coefficients not yet found to be significant are scanned. Additionally, those coefficients previously found to be significant are treated as zero for the purpose of determining if a zerotree exists. Thus, the second dominant pass consists of encoding the coefficient -31 in subband $LH3$ as negative significant, the coefficient 23 in subband $HH3$ as positive significant, the three coefficients in subband $HL2$ that have not been previously found to be significant (10, 14, -13) are each encoded as zerotree roots, as are all four coefficients in subband $LH2$ and all four coefficients in subband $HH2$. The second dominant pass terminates at this point since all other coefficients are predictably insignificant.

The subordinate list now contains, in order, the magnitudes (63, 49, 34, 47, 31, 23) which, prior to this subordinate pass, represent the three uncertainty intervals [48, 64), [32, 48) and [16, 31), each having equal width 16. The processing will refine each magnitude by creating two new uncertainty intervals for each of the three current uncertainty intervals. At the end of the second subordinate pass, the order of the magnitudes is (63, 49, 47, 34, 31, 23), since at this point, the decoder could have identified 34 and 47 as being in different intervals. Using the center of the uncertainty interval as the reconstruction value, the decoder lists the magnitudes as (60, 52, 44, 36, 28, 20).

The processing continues alternating between dominant and subordinate passes and can stop at any time.

VI. EXPERIMENTAL RESULTS

All experiments were performed by encoding and decoding an actual bit stream to verify the correctness of the algorithm. After a 12-byte header, the entire bit stream is arithmetically encoded using a single arithmetic coder with an adaptive model [31]. The model is initialized at each new threshold for each of the dominant and subordinate passes. From that point, the encoder is fully adaptive. Note in particular that there is no training of any kind, and no ensemble statistics of images are used in any way (unless one calls the zerotree hypothesis an ensemble statistic). The 12-byte header contains 1) the number of wavelet scales, 2) the dimensions of the image, 3) the maximum histogram count for the models in the arithmetic coder, 4) the image mean and 5) the initial threshold. Note that after the header, there is no overhead except for an extra symbol for end-of-bit-stream, which is always maintained at minimum probability. This extra symbol is not needed for storage on computer medium if the end of a file can be detected.

The EZW coder was applied to the standard black and white 8 bpp. test images, 512 × 512 "Lena" and the 512 × 512 "Barbara," which are shown in Figs. 9(a) and 11(a). Coding results for "Lena" are summarized in Table III and Fig. 9. Six scales of the QMF-pyramid were used. Similar results are shown for "Barbara" in Table IV and Fig. 10. Additional results for the 256 × 256 "Lena" are given in [22].

Quotes of PSNR for the 512 × 512 "Lena" image are so abundant throughout the image coding literature that it is difficult to definitively compare these results with other coding results.[1] However, a literature search has only found two published results where authors generate an actual bit stream that claims higher PSNR performance at rates between 0.25 and 1 bit/pixel [12] and [21], the latter of which is a variation of the EZW algorithm. For the "Barbara" image, which is far more difficult than "Lena," the performance using EZW is substantially better, at least numerically, than the 27.82 dB for 0.534 bpp. reported in [28].

The performance of the EZW coder was also compared to a widely available version of JPEG [14]. JPEG does not allow the user to select a target bit rate but instead allows the user to choose a "Quality Factor." In the experiments shown in Fig. 11, "Barbara" is encoded first using JPEG to a file size of 12 866 bytes, or a bit rate of 0.39 bpp. The PSNR in this case is 26.99 dB. The EZW encoder was then applied to "Barbara" with a target file

[1] Actually there are multiple versions of the luminance only "Lena" floating around, and the one used in [22] is darker and slightly more difficult than the "official" one obtained by this author from RPI after [22] was published. Also note that this should not be confused with results using only the green component of an RGB version which are also commonly cited.

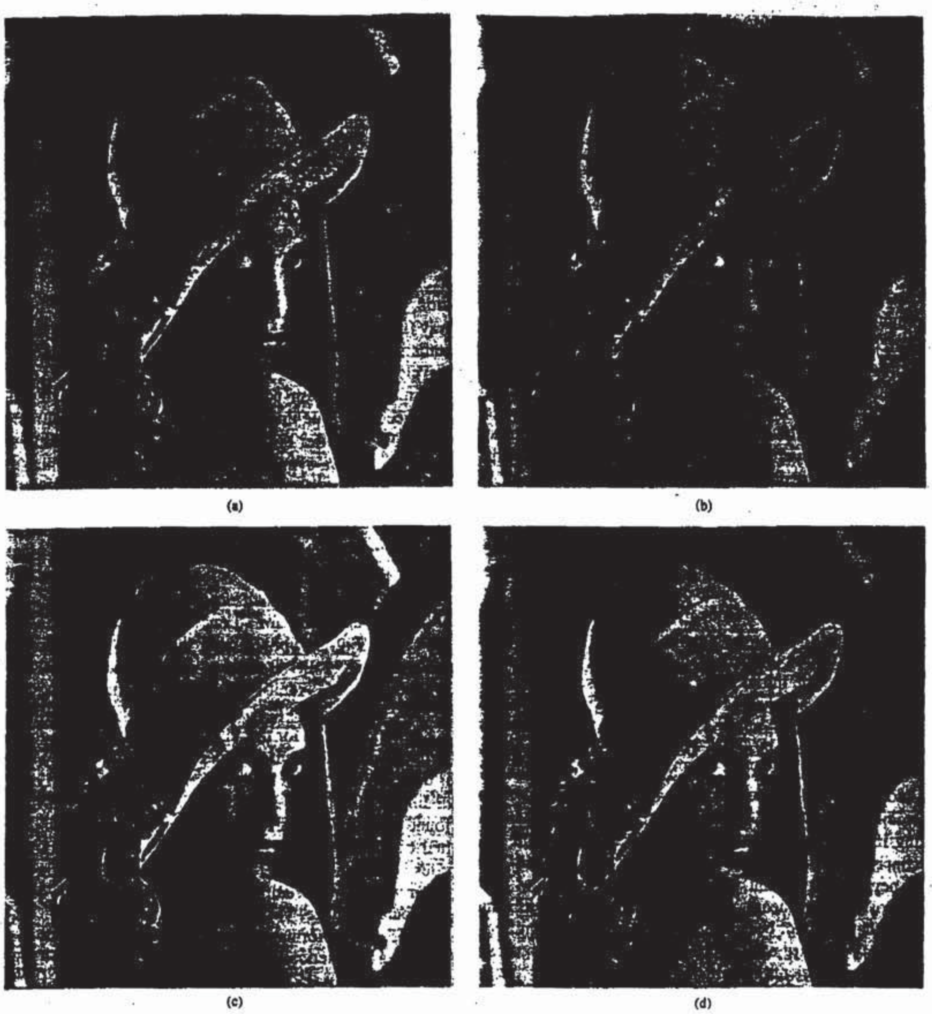


Fig. 9. Performance of EZW Coder operating on "Lena." (a) Original 512 × 512 "Lena" image at 8 bits/pixel (b) 1.0 bits/pixel, 8:1 Compression, PSNR = 39.55 dB, (c) 0.5 bits/pixel 16:1 Compression, PSNR = 36.28, (d) 0.25 bits/pixel, 32:1 Compression, PSNR = 33.17 dB, (e) 0.0625 bits/pixel, 128:1 Compression, PSNR = 27.54 dB, (f) 0.015625 bits/pixel, 512:1 Compression, PSNR = 23.63 dB.

size of exactly 12 866 bytes. The resulting PSNR is 29.39 dB, significantly higher than for JPEG. The EZW encoder was then applied to "Barbara" using a target PSNR to obtain exactly the same PSNR of 26.99. The resulting file size is 8820 bytes, or 0.27 bpp. Visually, the 0.39 bpp. EZW version looks better than the 0.39 bpp. JPEG version. While there is some loss of resolution in both, there are noticeable blocking artifacts in the JPEG version. For the comparison at the same PSNR, one could probably argue in favor of the JPEG.

Another interesting figure of merit is the number of significant coefficients retained. DeVore *et al.* used wavelet transform coding to progressively encode the same image [8]. Using 68 272 bits, (8534 bytes, 0.26 bpp.), they re-

(e)

(f)

Fig. 9. *(Continued.)*

TABLE III

Coding Results for 512 × 512 Lena Showing Peak-Signal-to-Noise (PSNR) and the Number of Wavelet Coefficients that were Coded as Nonzero

# bytes	R	Compression	MSE	PSNR (dB)	signif. coef
32768	1.0	8:1	7.21	39.55	39446
16384	0.5	16:1	15.32	36.28	19385
8192	0.25	32:1	31.33	33.17	9774
4096	0.125	64:1	61.67	30.23	4950
2048	0.0625	128:1	114.5	27.54	2433
1024	0.03125	256:1	188.2.7	25.38	1253
512	0.015625	512:1	281.7	23.63	616
256	0.0078125	1024:1	440.2	21.69	265

TABLE IV

Coding Results for 512 × 512 Barbara Showing Peak-Signal-to-Noise (PSNR) and the Number of Wavelet Coefficients that were Coded as Nonzero

# bytes	R	Compression	MSE	PSNR (dB)	signif. coef
32768	1.0	8:1	19.92	35.14	40766
16384	0.5	16:1	57.57	30.53	20554
8192	0.25	32:1	136.8	26.77	10167
4096	0.125	64:1	257.1	24.03	4522
2048	0.0625	128:1	318.5	23.10	2353
1024	0.03125	256:1	416.2	21.94	1259
512	0.015625	512:1	546.8	20.75	630
256	0.0078125	1024:1	772.5	19.54	291

tained 2019 coefficients and achieved a RMS error of 15.30 (MSE = 234, 24.42 dB), whereas using the embedded coding scheme, 9774 coefficients are retained, using only 8192 bytes. The PSNR for these two examples differs by over 8 dB. Part of the difference can be attributed to fact that the Haar basis was used in [8]. However, closer examination shows that the zerotree coding provides a much better way of encoding the positions of the significant coefficients than was used in [8].

An interesting and perhaps surprising property of embedded coding is that when the encoding or decoding is terminated during the middle of a pass, or in the middle of the scanning of a subband, there are no artifacts produced that would indicate where the termination occurs. In other words, some coefficients in the same subband are represented with twice the precision of the others. A possible explanation of this phenomena is that at low rates, there are so few significant coefficients that any one does not make a perceptible difference. Thus, if the last pass is a dominant pass, setting some coefficient that might be significant to zero may be imperceptible. Similarly, the fact that some have more precision than others is also imperceptible. By the time the number of significant coefficients becomes large, the picture quality is usually so good that adjacent coefficients with different precisions are imperceptible.

Another interesting property of the embedded coding is that because of the implicit global bit allocation, even at extremely high compression ratios, the performance scales. At a compression ratio of 512 : 1, the image quality of "Lena" is poor, but still recognizable. This is not the case with conventional block coding schemes, where at such high compression ratios, there would be insufficient bits to even encode the DC coefficients of each block.

The unavoidable artifacts produced at low bit rates using this method are typical of wavelet coding schemes coded to the same PSNR's. However, subjectively, they are not nearly as objectionable as the blocking effects typical of block transform coding schemes.

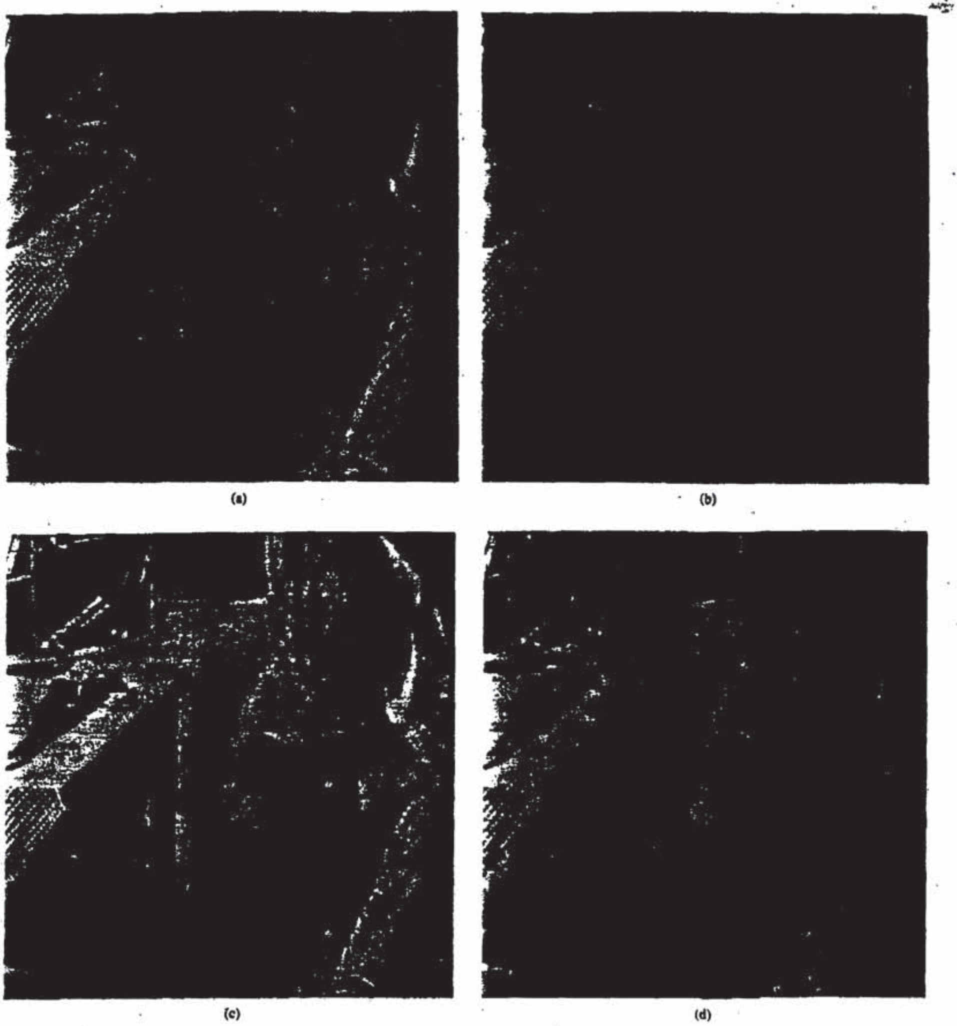


Fig. 10. Performance of EZW Coder operating on "Barbara" at (a) 1.0 bits/pixel, 8:1 Compression, PSNR = 35.14 dB (b) 0.5 bits/pixel, 16:1 Compression, PSNR = 30.53 dB, (c) 0.125 bits/pixel, 64:1 Compression, PSNR = 24.03 dB, (d) 0.0625 bits/pixel, 128:1 Compression, PSNR = 23.10 dB.

VII. Conclusion

A new technique for image coding has been presented that produces a fully embedded bit stream. Furthermore, the compression performance of this algorithm is competitive with virtually all known techniques. The remarkable performance can be attributed to the use of the following four features:

- a discrete wavelet transform, which decorrelates most sources fairly well, and allows the more significant bits of precision of most coefficients to be efficiently encoded as part of exponentially growing zerotrees,
- zerotree coding, which by predicting insignificance across scales using an image model that is easy for

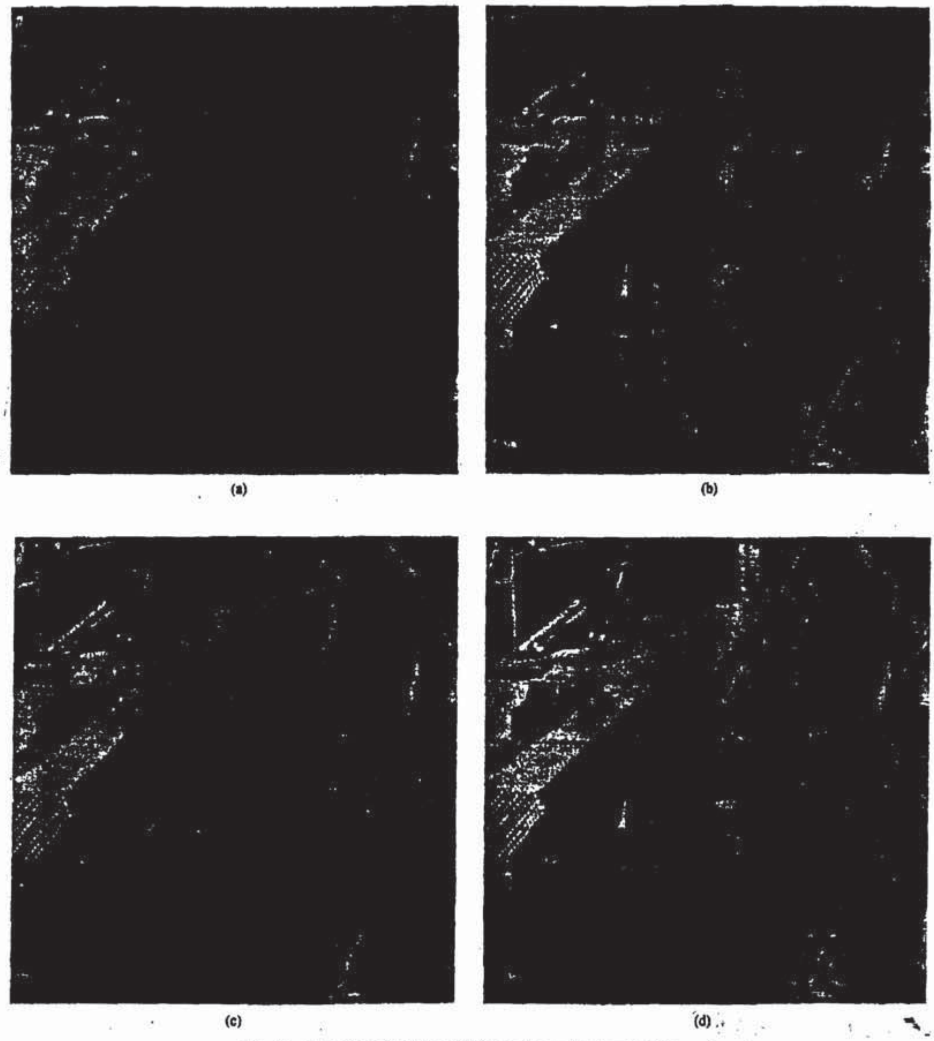


Fig. 11. Comparison of EZW and JPEG operating on "Barbara" (a) Original 512 × 512 (b) EZW at 12 866 bytes, 0.39 bits/pixel, 29.39 dB, (c) EZW at 8820 bytes, 0.27 bits/pixel, 26.99 dB, (d) JPEG at 12 866 bytes, 0.39 bits/pixel, 26.99 dB.

most images to satisfy, provides substantial coding gains over the first-order entropy for significance maps,

- successive-approximation, which allows the coding of multiple significance maps using zerotrees, and allows the encoding or decoding to stop at any point,
- adaptive arithmetic coding, which allows the entropy coder to incorporate learning into the bit stream itself.

The precise rate control that is achieved with this algorithm is a distinct advantage. The user can choose a bit rate and encode the image to *exactly* the desired bit rate. Furthermore, since no training of any kind is required,