Memory Constrained Wavelet Based Image Coding

Pamela Cosman, *Member, IEEE,* and Kenneth Zeger, *Senior Member, IEEE*

***Abstract*—We present a method for ordering the wavelet coefficient information in a compressed bitstream that allows an image to be sequentially decoded, with lower memory requirements than conventional wavelet decompression schemes. We also introduce a hybrid filtering scheme that uses different horizontal and vertical filters, each with different depths of wavelet decomposition. This reduces decoder memory requirements by reducing the instantaneous number of wavelet coefficients needed for inverse filtering.**

***Index Terms*— Color image coding, data compression, source coding, wavelet transforms, zerotrees.**

I. Introduction

WE CONSIDER the transmission of compressed images or video to inexpensive output devices, such as color printers and wireless videophones, where the amount of on-board memory is tightly constrained. While some existing compression algorithms have low memory requirements, many "high-memory" wavelet-based algorithms have superior distortion versus rate performance, such as embedded zerotree wavelet (EZW) coding, introduced by Shapiro [1] and later refined by Said and Pearlman [2] (SPIHT algorithm). In this letter, we introduce two new techniques to reduce memory requirements for wavelet compression. First, we alter the order of the transmitted information, so that the encoder sends to the decoder only the minimal set of wavelet coefficients needed to compute a given segment of an inverse wavelet transform (IWT) and to produce one output line of the image. The number of coefficients involved in the given partial IWT operation can be minimized by judicious selection of different filter lengths in the two spatial directions, and at different levels of hierarchical decomposition. We describe our memory-efficient coding scheme in terms of an improvement to zerotree algorithms such as EZW and SPIHT. We use the SPIHT zerotree approach as an example quantizer; we alter the wavelet transform and the bitstream ordering, and we omit their entropy coding step, to reduce complexity. A related scheme developed independently appears in [5]; it offers different performance tradeoffs.

II. Bitstream Reordering

A memory-saving approach is to maintain the full-frame wavelet transform, but to rearrange the order of the transmitted bitstream. Even though a full-frame forward wavelet transform is used to obtain the wavelet coefficients array, only a small subset of the wavelet coefficients is required by the IWT to obtain any given output row, without further quantization loss. This process can be described in terms of line-by-line reconstruction in the image spatial domain. We seek the *minimum set* of wavelet coefficients that must be received by the decoder in order to reconstruct a given single horizontal row in the output image. Then we transmit the minimum set of additional wavelet coefficients needed to reconstruct the following row. The decoder expunges any coefficients from the previous set that are not needed in the reconstruction of future rows. This process is iterated for each row until the entire image is reconstructed [3], [4].

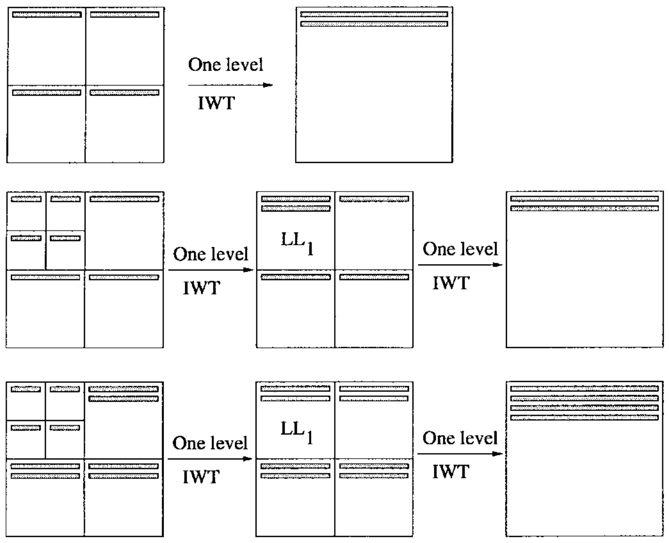

Fig. 1. (a) The decoder performs an inverse wavelet transform (IWT) operation to reconstruct the top row (and the second row) of pixels, using only a single line of coefficients from each of the four bands. (b) The decoder performs one level of IWT to reconstruct the top two rows of the LL_1 subband, using only a single line of coefficients from each of the four smallest subbands. Another round of IWT allows reconstruction of two rows of pixels. (c) Because of the zerotree dependencies, the decoder receives extra coefficients which are not required in the IWT operation that yields the top row. The extra coefficients allow four rows to be reconstructed.

As an example, suppose we decompose one level using a two-tap Haar filter. Coefficients needed to reconstruct the first output row are shown in Fig. 1(a). With more decomposition levels, memory requirements increase. With a two-level decomposition [see Fig. 1(b)], the decoder can be analyzed in terms of one level of IWT at a time. A single row of coefficients from each of the four smaller subbands allows reconstruction of the top two rows of the LL_1 band. Using only the top row from the LL_1 band, together with the top row from each of the other subbands, allows a second round of

Manuscript received April 15, 1998. This work was supported in part by Hewlett-Packard Company and the National Science Foundation. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. R. M. Mersereau.

The authors are with the Department of Electrical and Computer Engineering, University of California at San Diego, La Jolla, CA 92093-0407 USA (e-mail: cosman@code.ucsd.edu; zeger@code.ucsd.edu).

Publisher Item Identifier S 1070-9908(98)06866-7.

inverse wavelet filtering to reconstruct the top row (and second row) of image pixels to be reconstructed. Now consider the zerotree dependencies. In EZW, each coefficient in the lowest band (LL_2 band) has three children, and each of those has a 2×2 block of children in the next lower directional subband. This additional quantization information is not needed in the IWT to reconstruct the top row of pixels. However, with an EZW-style quantization, one cannot avoid the transmission and storage of this additional information when the top row is reconstructed. For this quantization method, the minimal set of quantized coefficients received by the decoder (for this decomposition level and choice of filter) is shown in Fig. 1(c). These coefficients allow reconstruction of the top row and are in fact sufficient to enable the decoder to reconstruct the top four rows.

The situation is less favorable when longer filters are used. With a four-tap Daubechies filter, two rows of coefficients from each of the four subbands allow reconstruction of the top output row. To reconstruct the second row, no additional coefficients are needed, nor can we purge any coefficients. To reconstruct the third row, one line can be purged from each of the four subbands, and we need to receive a new line from the encoder for each of the four subbands. The memory usage thus remains constant. This forms a "sliding window" in which wavelet coefficients enter the window, are used for a few rounds of inverse filtering to reconstruct some rows, and then exit the window.

This process can be thought of as a reordering of the SPIHT or EZW output bitstream. An ordinary SPIHT encoder (without arithmetic coding) is used initially to encode the input image to the desired transmission rate. The encoder rearranges the progressively ordered bits so that the bits corresponding to the top line of trees are transmitted first, followed by the next line of trees, etc. A small header is sent at the beginning of the image, which describes the threshold T_n and the coordinates (x, y) of the coefficient at which the encoding of the original algorithm (not reordered) terminates. The decoder then evaluates, as it decodes each row in the reordered bitstream, whether the received bits correspond to a threshold of T_n, and, if they do, whether the coefficient coordinates correspond to a position that follows (x, y). If they do, then the decoder deduces that the data corresponds to the next row. While the encoding process can perhaps most easily be understood as a reordering of the zerotree-style bitstream, in actual practice the encoder does not need to encode the full frame and then reorder it. As long as the encoder has some means of determining or estimating the desired terminating threshold, it can run SPIHT-style on the reduced sets of coefficient trees in sequence, as can the decoder.

III. Hybrid Filtering

The examples illustrate that more coefficients are required to produce a single output row when either the filter length increases, or the number of decomposition levels increases. For example, with a 512×512 image undergoing a single level of filtering with a two-tap filter, only $2 \times 512 = 1024$ wavelet coefficients are required to produce an output row (512 pixels). This constitutes 0.4% of the wavelet coefficient array. With a full six levels of decomposition using the same short filter, 64 rows or 12.5% of the array is involved in producing a single output row. With six levels of decomposition with 9-7 biorthogonal filters, the zerotree quantization structure requires approximately 62.5% of the coefficients to be buffered at one time. (Note that the SPIHT encoder groups coefficients together after the last decomposition operation, producing an effect similar to a seventh level of decomposition, and so essentially 100% of the coefficient array is involved in producing a single output row).

One can save in memory by shortening the filters or decomposing fewer levels, at the expense of significant SPIHT performance reduction. However, in our scheme, neither large numbers of levels of decomposition nor long filter lengths *in the horizontal direction* cause increased memory requirements, since one horizontal row always is reconstructed at a time, matching the direction in which the paper exits the printer. To satisfy (and exploit) these constraints, we introduce a hybrid filter—a wavelet transform that uses either different numbers of decomposition levels or different filter lengths, or both, in the different spatial directions. In one example, we perform the full six levels of decomposition using the 9-7 filters in the horizontal direction and in the vertical direction we decompose three levels with the 9-7 filters, and an additional three levels with the 2-tap Haar filters. The total number of levels of decomposition in each direction is six in this example, but the buffering is much less than if the 9-7 filters were used in both directions. The total number of levels of decomposition does not have to be the same in the two directions. There exist many different possibilities, each presenting its own trade-off between distortion, rate, and buffering requirements. It is also possible to allow the transforms in the two directions to be of different types (e.g., horizontal wavelet transform and vertical block DCT), giving rise to other trade-offs such as visual artifacts due to "blocking."

Fig. 2 shows a block diagram of the decoding mechanism used with a hybrid filtering structure in the vertical direction with six levels of wavelet reconstruction. The first three levels are performed using Haar filters and the latter three levels use 9-7 biorthogonal filters. It is assumed that all six levels of decomposition in the horizontal direction use the 9-7 biorthogonal filters (not shown). In addition to the filters, extra storage elements are shown (labeled "D") preceding the high pass filters (and the first lowpass filter).

When the filters' memories are filled, to produce 64 new outputs (i.e., vertical lines) at the end of the filterbank, one must provide one scalar input at $L1$ and $H1$, two inputs at $H2$, four at $H3$, eight at $H4$, 16 at $H5$, and 32 at $H6$. For some quantization schemes, only the memory of these filters is necessary to decode the image from the wavelet domain. In general, the memory requirements can be divided into the inverse filtering operations and the quantization operations. Memoryless scalar quantization (SQ) requires no extra storage, whereas predictive SQ and context-based adaptive entropy coded SQ require extra coefficients to be stored for prediction or context during reconstruction. With zerotrees, information from different bit planes must be temporarily stored until entire

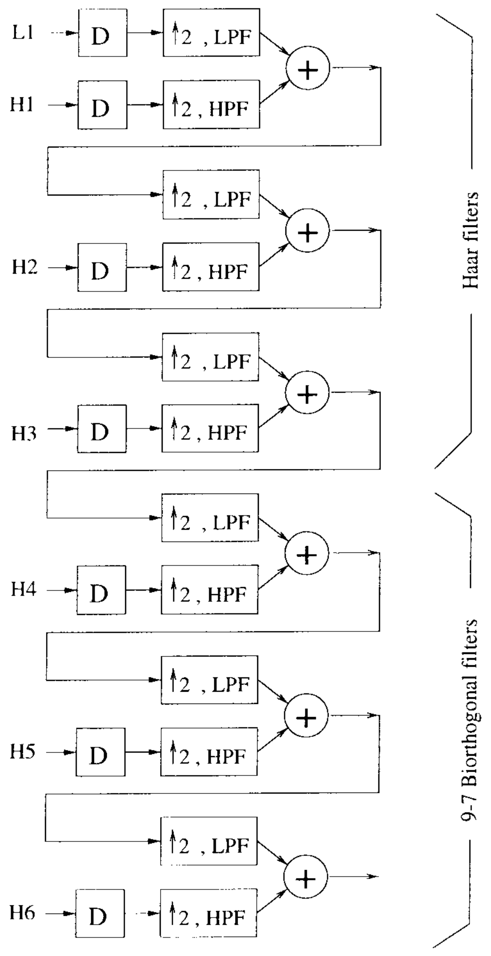

Fig. 2. Block diagram of hybrid filter wavelet decoder for vertical direction.

wavelet coefficients can be deduced; then they can (in the proper order) be filtered to produce output image values. The "D" boxes in this case represent the process of accumulating bit plane information and delaying the input until wavelet coefficients are known. For EZW coding, for example, the memory requirement is the sum of the filter memories and the delay memories. A substantial savings in memory over the usual full-image EZW decoding can still be achieved.

IV. Results and Conclusions

The 24–bit color Lena image was compressed to 0.24 bits/pixel; a numerical comparison of the results with various hybrid filters is given in Table I. In all cases, horizontal filtering is identical to that in SPIHT; vertical filtering, however, uses n levels of decomposition with 9-7 filters and 6-n further levels with Haar filters. Here we report a color peak signal-to-noise ratio (PSNR) by averaging the three component colors' mean square errors (MSE's) to produce MSE_{avg} and then taking a logarithm as: $\text{PSNR} = 10 \log_{10}(255^2/\text{MSE}_{avg})$. The performance loss associated with substituting two or three levels of Haar filtering vertically is slight. While our work has been discussed primarily in the context of decoder operations, it is generally applicable to memory savings in encoder operations as well. That is, for many quantization and entropy coding strategies, filtering operations can be done incrementally at the encoder as well, employing each new line as it becomes available. We have recently extended the work in this paper by using quadtree-guided wavelet compression [6] for reduced memory coding, since only one or two levels of decomposition are required, and the extra storage (for prediction) is small [7].

TABLE I
PSNR Results for Color Lena Image at 0.24 Bits/Pixel, Using Various Hybrid Filters. The Lower Bound on Memory Usage Is in Terms of the Percentage of the Wavelet Coefficient Array That Must Be Buffered at any One Time by the Decoder in Order to Accomplish the Inverse Filtering for the Given Filtering Operations. The Actual Memory Requirements Will Be Higher, Depending on Whether Low Band Coefficients Are Grouped to Form a Seventh Level of Decomposition, and Also Because the Decoder Needs to Store Lists of Coefficient Sets

Decomposition Levels with 9-7 filters	Decomposition Levels with Haar filters	PSNR (dB)	Lower bound on Memory Usage
6	0	30.77	62.5 %
5	1	30.76	37.5 %
4	2	30.73	25.0 %
3	3	30.64	18.7 %
2	4	30.42	15.6 %
1	5	30.05	14.1 %
0	6	29.60	12.5 %

Acknowledgment

The authors would like to thank S. Tjarnlund, M. Pettersson, and T. Frajka for their help in running numerical experiments and C. Chrysafis and A. Ortega for providing a preprint of [5].

References

[1] J. M. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients," *IEEE Trans. Signal Processing,* vol. 41, pp. 3445–3462, Dec. 1993.
[2] A. Said and W. A. Pearlman, "A new, fast, and efficient image codec based on set partitioning in hierarchical trees," *IEEE Trans. Circuits Syst. Video Technol.,* vol. 6, pp. 243–250, June 1996.
[3] M. Pettersson and S. Tjärnlund, "Color image compression for color printers using wavelet transform," Master's thesis, Dept. Electr. Eng., Linköping Univ., S-581 83 Linköping, Sweden, Nov. 1997.
[4] P. C. Cosman and K. Zeger, "Memory constrained wavelet-based image coding," in *Conf. Rec. First Ann. UCSD Conf. Wireless Commun.,* Mar. 1998, pp. 54–60.
[5] C. Chrysafis and A. Ortega, "Line based, reduced memory, wavelet image compression," in *Proc. Data Compression Conf.,* Snowbird, UT, Mar. 1998.
[6] C.-Y. Teng and D. Neuhoff, "Quadtree-guided wavelet image coding," in *Proc. Data Compression Conf.,* 1996, pp. 406–415.
[7] P. Cosman, T. Frajka, and K. Zeger, "Image compression for memory constrained printers," in *Proc. 1998 Int. Conf. Image Processing,* Chicago, IL, Oct. 1998, to be published.