Image Coding Via Wavelets

M. Shahshahani
Communications Systems Research Section

The application of two wavelet transforms to image compression is discussed. It is noted that the Haar transform, with proper bit allocation, has performance that is visually superior to an algorithm based on a Daubechies filter and to the discrete-cosine-transform-based Joint Photographic Experts Group (JPEG) algorithm at compression ratios exceeding 20:1. In terms of the root-mean-square error, the performance of the Haar transform method is basically comparable to that of the JPEG algorithm. The implementation of the Haar transform can be achieved in integer arithmetic, making it very suitable for applications requiring real-time performance.

I. Introduction

In an earlier article [1], the author reported on his work on the application of a wavelet transform to image coding. It was noted that the images processed by this method did not suffer from the blockiness which is typical of algorithms based on the discrete cosine transform (DCT) at compression ratios exceeding 20 to 1; however, the edges of the objects in an image were blurred and created an unsatisfactory visual impression. Two approaches were taken to overcome this problem. The wavelet transform used in [1] was essentially one dimensional in nature, and it was applied along rows and columns of the image. In the first approach, it was surmised that the reason for the blurriness of the edges was that the application of the transform along rows and columns did not properly take advantage of the proximity of the pixels. Several modified versions of this transform reflecting the contiguity of the pixels were tested. Improvements in the edges and the general quality of the processed images were observed for certain modified transforms. The second approach was based on a two-dimensional Haar transform. The edges and the general quality of the processed images were in general superior to those of the first method and to those of the Joint Photographic Experts Group (JPEG) DCT-based algorithm at compression ratios exceeding 20 to 1. In terms of the root-mean-square error (RMSE), the performance of this method is essentially comparable to that of the JPEG DCT-based algorithm. This is remarkable, since the quantization and bit-allocation algorithms used in the tests reported here were of a much cruder nature than those utilized in the DCT-based approach. While there is some blockiness in images processed with the Haar transform method, it is not nearly as severe as in images processed with the DCT-based algorithm, and the edges appear as sharp as those obtained by the latter method.

In the application of wavelet transforms, one assumes that the wavelet transform coefficients are encoded by an entropy encoder. This method is analogous to algorithms based on other transforms such as DCT. An advantage of the Haar transform approach is that both the algorithm and its inversion can be implemented in integer arithmetic and are very suitable for applications where real-time performance is required.

II. A Daubechies Filter

The work in [1] was based on the filter $\mathcal{F}$ defined by the matrix

$$\begin{pmatrix} \alpha_0 & \alpha_1 & \alpha_2 & \alpha_3 & 0 & 0 & 0 & \dots & \dots & 0 \\ \alpha_3 & -\alpha_2 & \alpha_1 & -\alpha_0 & 0 & 0 & 0 & \dots & \dots & 0 \\ 0 & 0 & \alpha_0 & \alpha_1 & \alpha_2 & \alpha_3 & 0 & \dots & \dots & 0 \\ 0 & 0 & \alpha_3 & -\alpha_2 & \alpha_1 & -\alpha_0 & 0 & \dots & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \ddots & & & \vdots \\ \alpha_2 & \alpha_3 & 0 & 0 & \dots & \dots & \dots & 0 & \alpha_0 & \alpha_1 \\ \alpha_1 & -\alpha_0 & 0 & 0 & \dots & \dots & \dots & 0 & \alpha_3 & -\alpha_2 \end{pmatrix}$$

where

$$\alpha_0 = \frac{1+\sqrt{3}}{4\sqrt{2}}, \quad \alpha_1 = \frac{3+\sqrt{3}}{4\sqrt{2}}$$

$$\alpha_2 = \frac{3-\sqrt{3}}{4\sqrt{2}}, \quad \alpha_3 = \frac{1-\sqrt{3}}{4\sqrt{2}}$$

This filter, the discrete version of one of Daubechies' compactly supported wavelets [2], is one dimensional and was applied in the work reported in [1] along rows and columns of an image. Instead, it is possible to reorder the pixels differently to take advantage of their proximity and then apply the filter. There is no unique way of reordering or reconfiguring the pixels. A number of experiments were carried out with widely different results. Certain configurations led to improvement of the edges, while others resulted in significant deterioration of the processed image. The author does not know of any theoretical procedure for obtaining the optimal configuration. The experiments did, however, yield some insight into what a desirable configuration may look like. A method which produced improvements in the processed image is a two-step application of the filter $\mathcal{F}$. The first step is described in Figure 1. Starting with an even-numbered row, the weighted average of the pixels with even coordinates $(2m, k)$, $(2m+1, k)$, $(2m, k+1)$, and $(2m+1, k+1)$ is computed by using the weights α_0, α_1, α_2, and α_3. This weighted average is the transformed value at the pixel with coordinates $(2m, k)$. To compute the transformed values at $(2m+1, k)$, the weighted average of the pixels $(2m, k)$, $(2m+1, k)$, $(2m, k+1)$, and $(2m+1, k+1)$ with weights α_3, $-\alpha_2$, α_1, and $-\alpha_0$ is calculated. The second step is the application of the filter $\mathcal{F}$ along the even-numbered rows after the image is transformed according to the procedure of step 1. The hierarchy of the coefficients is different from that described in [1]. After step 1 of the first iteration of the two-step process, the coefficients in odd-numbered rows are placed at the lowest level of the hierarchy. The coefficients in the even-numbered rows are then transformed according to step 2, and those coefficients corresponding to odd-numbered columns are placed immediately above the lowest level. The next iteration is the application of the same procedure to the subgrid of points with even coordinates. Evenness and oddness are then replaced by divisibility by 4 and congruence to 2 mod 4. The procedure can be repeated in the obvious manner. In the terminology of [1], the coefficients at pixels with coordinates divisible by 2^r form the projection of the data from $\mathcal{L}_{r-1}$ to $\mathcal{L}_r$. Figure 2 is an image of peppers that was processed by this algorithm. The compression ratio is 27:1.

III. The Haar Transform

The Haar transform is defined by the matrix

$$\mathcal{H} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ -1 & 1 & -1 & 1 \\ 1 & -1 & -1 & 1 \\ -1 & -1 & 1 & 1 \end{pmatrix}$$

which is a Hadamard matrix (see [3]). Just as is the case with the Daubechies filter $\mathcal{F}$, this is a one-dimensional filter which may be applied in different ways to the two-dimensional image. The configuration which appears to best take advantage of the proximity of the pixels and produce the most appealing results visually is shown in Fig. 3. The image is subdivided into 2 x 2 blocks. Within each block the pixels are reordered as $(2k, 2m)$, $(2k, 2m+1)$, $(2k+1, 2m)$, and $(2k+1, 2m+1)$, and the matrix $\mathcal{H}$ is applied to each block separately to yield the transformed image which is of the same size. The coefficients in the transformed image are separated into two groups which were designated in [1] as "smooth" and "detail." In the terminology of [1], those coefficients corresponding to the pixels with even coordinates form the smooth group (i.e., the image of the projection from $\mathcal{L}_0$ to $\mathcal{L}_1$), and the remaining are the details (i.e., the image of the projection from $\mathcal{L}_0$ to $\mathcal{E}_0$). The filter $\mathcal{H}$ is then applied to the smooth group in the obvious manner, leading to a hierarchy of transformed data. Figures 4(a) and 4(b) are two images processed by the Haar transform. The compression ratios are 12:1 and 27:1, respectively. Notice that the edges are clearly defined, even with a compression factor of 27.

Notice that the Haar transform is integer valued, and therefore its application can be implemented in integer arithmetic. Since averaging over four pixel values generates large numbers rapidly, it was convenient to divide the values by four, which from the implementation point of view is easily achievable. Naturally, one loses some resolution in this manner; nevertheless, this approach seems to be convenient.

IV. Bit Allocation and Quantization

As noted in [1], the best method of quantization is by truncation to the nearest integer, irrespective of the distribution of the coefficients. Bit allocation is done according to the desired compression ratio and the level of the hierarchy in the wavelet pyramid. Figure 5 exhibits RMSE versus compression ratio for what, after much experimentation, appears to be a good choice of bit assignments to the wavelet coefficients obtained via the application of the filter $\mathcal{F}$ as described in Section II. The actual bits assigned to each level are given in Table 1.

The actual bit allocation affects the compression ratio versus root-mean-square error significantly. In the application of the Haar transform, the actual effect of different bit-allocation schemes on RMSE is shown in Fig. 6. Each circled point in the graph corresponds to a particular assignment of bits to different levels of the hierarchy of the wavelet coefficients. It is clear from the graph that certain assignments do not produce desirable effects. The desirable bit-allocation schemes correspond to the lower envelope of the curve. This lower envelope is reproduced in Fig. 7. Figure 8 gives a comparison between the Haar transform algorithm and the standard JPEG DCT-based algorithm on the basis of the root-mean-square error. Note that the performance of the two methods is comparable. This is remarkable, since the latter approach makes use of refined quantization and bit-allocation algorithms, while in the tests reported here, rather crude methods were used. It is therefore reasonable to expect that with further refinement of the Haar method, it will have superior performance even in terms of RMSE. The actual bits assigned to each level of the hierarchy are given in Table 2. For practical applications, the tables and the graphs may be used as guidelines for bit allocation.

V. Conclusion

The Haar transform is a viable alternative to the discrete cosine transform for image compression. With proper bit-allocation, this approach has performance visually superior to that of the commonly used DCT-based JPEG algorithm at compression ratios exceeding 20:1. In terms of RMSE, the performances of the two methods are comparable. It is therefore reasonable to expect that with further refinement of the quantization and bit-allocation algorithms, the performance of the Haar transform will become superior to that of the DCT-based algorithm even in terms of RMSE.

Acknowledgment

The author thanks F. Pollara for providing the data for the application of the JPEG DCT-based algorithm.

References

[1] M. Shahshahani, "Data Compression by Wavelet Transforms," *The Telecommunications and Data Acquisition Progress Report 42-110*, vol. April–June 1992, Jet Propulsion Laboratory, Pasadena, California, pp. 179–187, August 15, 1992.

[2] I. Daubechies, "Orthonormal Bases of Compactly Supported Wavelets," *Communications in Pure and Applied Mathematics*, vol. 41, pp. 909–996, November 1988.

[3] R. A. DeVore, B. Jawerth, and B. J. Lucier, "Image Compression Through Wavelet Transform Coding," *IEEE Transactions on Information Theory*, vol. 38, no. 2, pp. 719–746, March 1992.

Table 1. The assignment of bits resulting in the graph shown in Fig. 5.

Compression ratio	Bit allocation
20.7	2,2,4,4,6,6
25	2,2,3,3,6,6
29.7	2,2,3,3,4,6
36.2	0,2,3,3,4,5
42.8	0,2,3,3,4,5,6,6
46.8	0,0,3,3,4,4,6,6

Table 2. The assignment of bits resulting in the plots in Fig. 8.

Compression ratio	Bit allocation
9.1	4,6,8
24.2	2,4,6,8
27.1	0,4,6,8
29.2	0,4,6,6
35.5	2,3,5,7
37.3	2,3,5,6
39.7	2,3,5,6,6
42.0	0,3,5,7
43.3	2,3,4,6
43.6	0,2,6,6
48.1	0,3,5,6,6
49.0	2,2,4,7
56.7	2,2,3,7
57.5	0,2,4,8
62.4	0,2,4,7
67.9	0,2,4,6

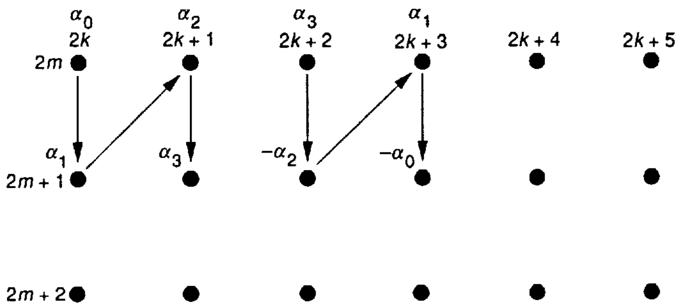

Fig. 1. Step 1 of reconfiguration.

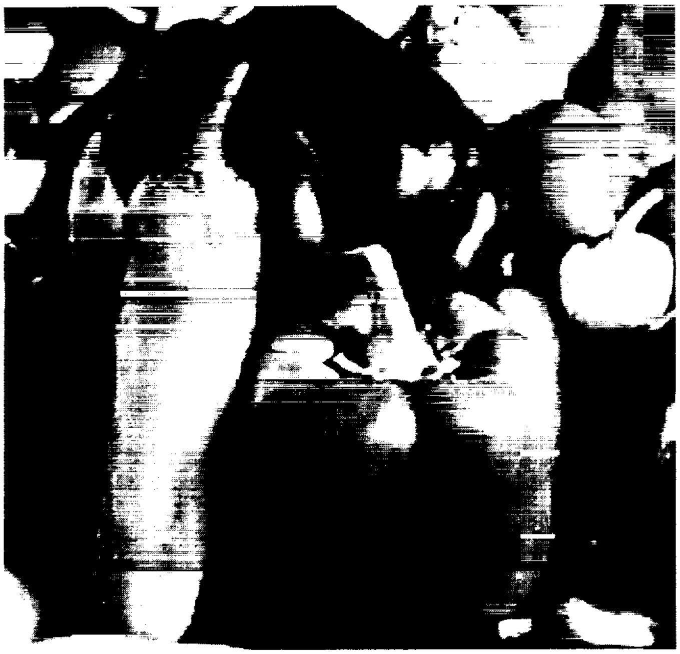

Fig. 2. An image of peppers that was processed by a transform based on the Daubechies filter.

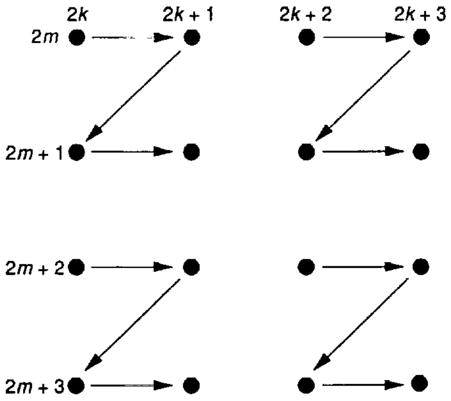

Fig. 3. Configuration of pixels for the Haar transform.

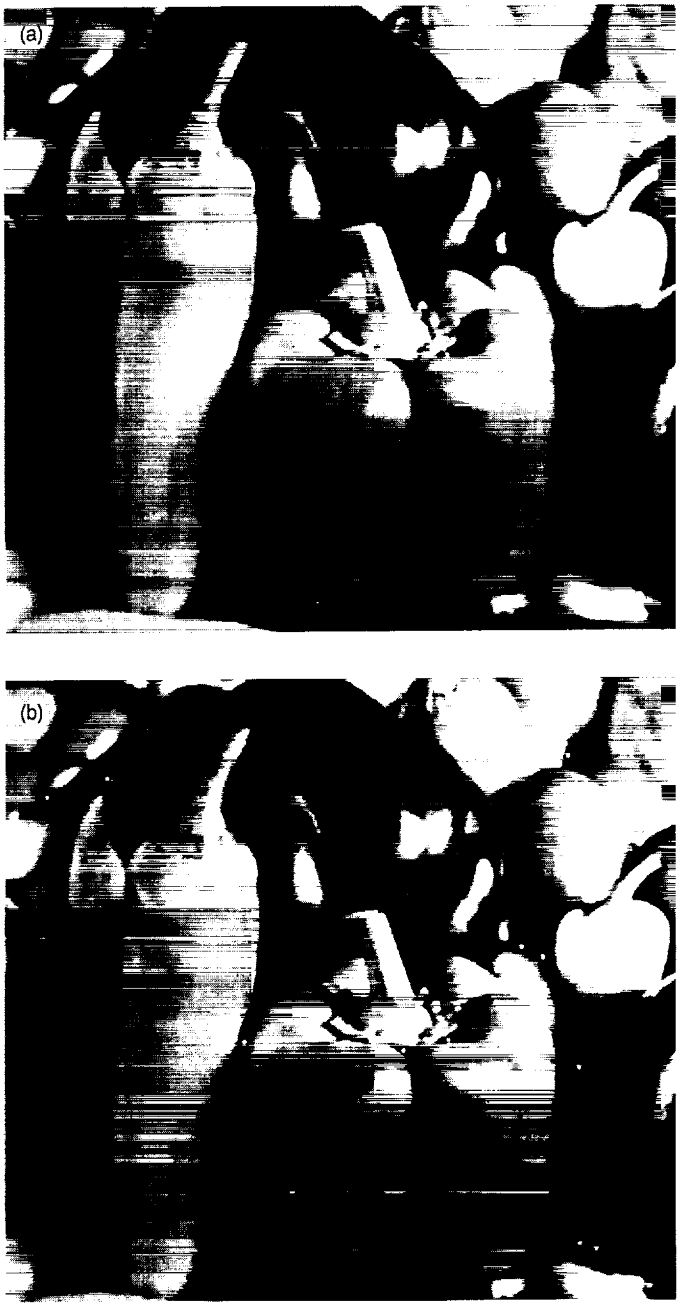

Fig. 4. Images processed by the Haar transform when (a) the compression ratio is 12:1 and (b) the compression ratio is 27:1.

ORIGINAL PAGE
BLACK AND WHITE PHOTOGRAPH

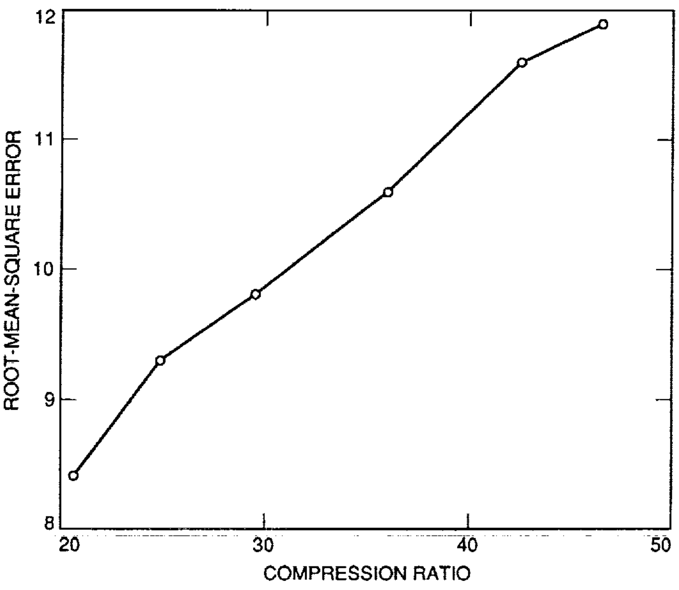

Fig. 5. Root-mean-square error versus compression ratio for good bit assignments. (See Table 1 for actual bits assigned.)

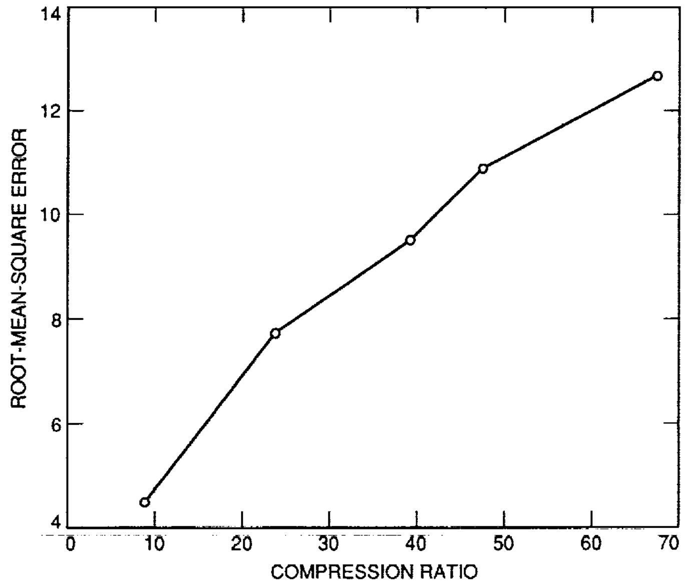

Fig. 7. Lower envelope of the curve in Fig. 6.

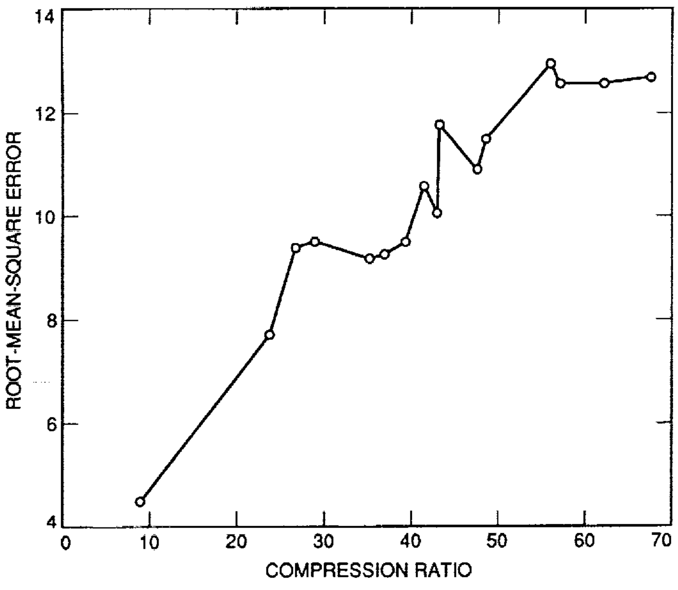

Fig. 6. The effect on root-mean-square error of different bit-allocation schemes resulting from application of the Haar transform.

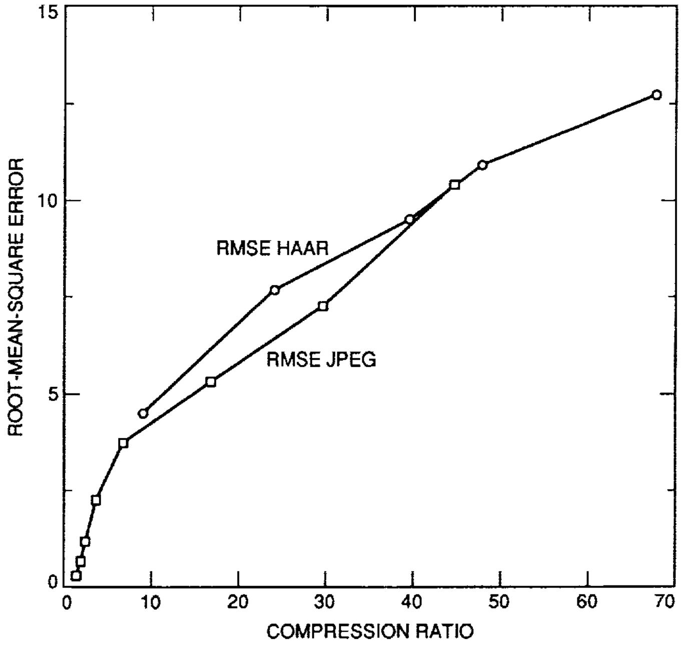

Fig. 8. Root-mean-square error comparison of the Haar transform and JPEG DCT-based algorithms.