spend an extra amount of computation (around 0.16×10^6 additions) on the thresholding operation, but we achieve a large saving in computation (around 2.16×10^6 additions) by withholding those blocks, whose MAD values at the full resolution level are less than the predefined accuracy criterion, from further processing. This is where the computational savings comes from.

The frames of the "Train" sequence are 720×288 pixels, and only the central portion, 640×256 pixels, is processed. With the operational parameters listed in Table I (the criterion value three), about 52% of the total blocks are stopped at the top level. The processing time has been reduced by about 17% by the new algorithm, compared with Method 1. The PSNR, the error image entropy, and the vector entropy are almost the same.

The frames of the "Football" sequence are 720×480 pixels, and only the central portion, 640×384 pixels, is processed. With the operational parameters listed in Table I (the criterion value four), about 38% of the total blocks are stopped at the top level. The processing time is about 14% less than that required by Method 1, while the PSNR, the error image entropy, and the vector entropy are almost the same.

As discussed in the previous section, the experiments with a single accuracy criterion of value three also produce a similar, good performance for all three different image sequences.

In summary, it is clear that with the three different testing sequences, our algorithm works faster than the fastest existing top-down multiresolution block matching algorithm while achieving almost the same quality of the reconstructed image.

IV. Conclusions

The existing multiresolution block matching techniques, such as the top-down pyramid technique, propagate all the motion vectors estimated at a lower resolution level to the next higher resolution level for refinement regardless of whether the computed motion vector gives a satisfactory motion compensation or not. Based on this observation, we presented a new thresholding multiresolution block-matching algorithm so that motion vectors computed at the lower resolution level will be treated differently. According to the motion compensation performance, those blocks satisfying the predefined accuracy criterion are withheld from further processing, and a large amount of computation is saved. Three experiments with quite different motion complexities have shown that the proposed algorithm works well. It greatly reduces the processing time from 14% to 20%, compared with the fastest existing multiresolution technique, while maintaining almost the same quality of the reconstructed image.

Acknowledgment

The authors appreciate the reviewers' detailed and valuable constructive comments.

References

[1] H. G. Musmann and P. Pirsch, "Advances in picture coding," *Proc. IEEE*, vol. 73, no. 4, pp. 523–548, Apr. 1985.

[2] D. L. Gall, "MPEG: a video compression standard for multimedia applications," *Commun. AMC*, vol. 34, pp. 46–58, Apr. 1991.

[3] D. Tzovaras, M. G. Strintzis, and H. Sahinolou, "Evaluation of multiresolution block matching techniques for motion and disparity estimation," *Signal Processing: Image Commun.*, vol. 6, pp. 56–67, 1994.

[4] F. Dufaux and F. Moscheni, "Motion estimation techniques for digital TV: A review and a new contribution," *Proc. IEEE*, vol. 83, no. 6, pp. 858–876, 1995.

[5] M. H. Chan, Y. B. Yu, and A. G. Constantindes, "Variable size block matching motion compensation with applications to video coding," *Proc. Inst. Elec. Eng.*, vol. 137, pt. I, no. 4, pp. 205–212, Aug. 1990.

[6] F. Dufaux and M. Kunt, "Multigrid block matching motion estimation with an adaptive local mesh refinement," in *SPIE Proc. Visual Communications and Image Processing '92*, vol. 1818, Boston, MA, Nov. 1992, pp. 97–109.

Layered DCT Still Image Compression

Jiankun Li, Jin Li, and C.-C. Jay Kuo

***Abstract*— Motivated by Shapiro's embedded zerotree wavelet (EZW) coding and Taubman and Zakhor's layered zero coding (LZC), we propose a layered discrete cosine transform (DCT) image compression scheme, which generates an embedded bit stream for DCT coefficients according to their importance. The new method allows progressive image transmission and simplifies the rate-control problem. In addition to these functionalities, it provides a substantial rate-distortion improvement over the JPEG standard when the bit rates become low. For example, we observe a bit rate reduction with respect to the JPEG Huffman and arithmetic coders by about 60% and 20%, respectively, for a bit rate around 0.1 b/p.**

***Index Terms*— Arithmetic coder, JPEG, layered coding, progressive transmission.**

I. Introduction

The JPEG [1] compression standard is widely adopted in still image coding. The process consists of three stages: the block discrete cosine transform (DCT), uniform quantization with a quantization table, and the Huffman (used in the baseline system) or the arithmetic (used in the extended system) entropy coding. In JPEG, one DCT coefficient has to be completely encoded before the coding of the next coefficient. More recently, a concept known as *embedded* or *layered* coding was proposed by Shapiro [3] and further developed by Taubman and Zakhor [4] in the context of wavelet coding. In contrast with the coefficient-by-coefficient approach, they adopted a new approach, in which each coefficient is successively quantized into a certain number of bits. The most significant bits of all coefficients are grouped together to form one layer and encoded first. Then, we move to the layer of the second significant bits and so on. The coding order is consistent with the importance of each bit so that the encoder and the decoder can stop at any time. The embedding property is essential for progressive image transmission. It also greatly simplifies the rate-control problem and allows unequal error protection for robust image transmission.

In this letter, we generalize the layered coding to image compression methods based on the block DCT transform. Even though the generalization is not difficult, we feel that it is valuable to have

Manuscript received April 5, 1996; revised November 12, 1996. This paper was recommended by Associate Editor J. Brailean. This work was supported by the Integrated Media Systems Center, a National Science Foundation Engineering Research Center, and the National Science Foundation Presidential Faculty Fellow (PFF) Award ASC-9350309.

The authors are with the Integrated Media Systems Center and the Department of Electrical Engineering-Systems, University of Southern California, Los Angeles, CA 90089-2564 USA.

Publisher Item Identifier S 1051-8215(97)02238-6.

its detailed implementation available. This is the main objective of the letter. Besides providing more functionalities, the new coder gives a substantially better rate-distortion performance than the JPEG standard, especially at low bit rates.

Even though there exist strong similarities between our proposed method and progressive JPEG, we would like to point out two major differences. First, DCT coefficients are quantized to integer indices and then progressively encoded in progressive JPEG. Thus, its coding distortion is bounded by the quantization step. In our scheme, DCT coefficients are quantized successively so that there is no lower bound on coding distortion due to quantization. Theoretically speaking, an arbitrary level of accuracy can be achieved. Second, our approach predicts the position of significant coefficients across bit layers, which achieves a substantially better rate-distortion performance than progressive JPEG.

The work is organized as follows. We introduce the concept of layered coding in Section II. The detailed implementation is presented in Section III. Experimental results are given in Section IV.

II. Overview of Layered Coding of DCT Coefficients

The proposed layered DCT coding adopts the 8×8 block DCT transform which is the same as JPEG. Its main differences are in the quantization and the entropy coding schemes. For a given 8×8 DCT block, we arrange the 63 ac coefficients in a zigzag order and denote them by $C_1, C_2, \cdots, C_{63}$. Suppose that the ac coefficients after quantization take a value ranging from $-32\,767$ to $32\,767$ so that each of them requires 16 b for representation (including the sign bit). They are labeled with $S, B_1, \cdots, B_{15}$, where S is the sign bit, B_1 the most significant bit (MSB), and B_{15} the least significant bit (LSB). Values in such a bit matrix form a set of *intermediate symbols*. JPEG encodes these symbols in a coefficient-by-coefficient manner with an entropy coder. A two-dimensional (2-D) run-level Huffman coder is adopted in the baseline system while the arithmetic coder is used in the extended system. For more details of JPEG, we refer to [1]. One major issue with JPEG is the rate control problem. It is, in general, difficult to estimate the number of coding bits generated for a given Q factor. Note also that truncating the compressed bit stream at an arbitrary point is equivalent to deleting the bottom portion of the image, since it is encoded sequentially from one block to another.

In this research, we demonstrate that by adopting a different coding order for intermediate symbols, we convert the conventional JPEG coder into an embedded coder called the *layered DCT coder*. Consider the grouping of all bits B_i of coefficients C_j, $1 \le j \le 63$, into layer L_i. Now, the new coder first encodes the most significant layer L_1, then layers L_2, L_3, and so on. Within each layer L_i, the coding follows the coefficient order, i.e., starting with coefficient C_1, then $C_2, C_3, \cdots, C_{63}$. This coding order is illustrated in Fig. 1. One important advantage of the layer-by-layer coding is that the resulting bit stream has the embedding property. Since the output bit stream is organized in an order of decreasing importance, rate-control can be easily achieved by simply truncating the bit stream according to the desired coding budget, or the desired coding quality. The layered DCT coding is also suitable for progressive transmission, for the more important coding bit is always transmitted prior to the less important bit. Another advantage of the layer-by-layer coding is that the new scheme provides a better rate-distortion tradeoff especially at low bit rates, which will be demonstrated in Section IV.

III. Detailed Implementation

The detailed implementation of the proposed layered DCT compression algorithm is described in this section.

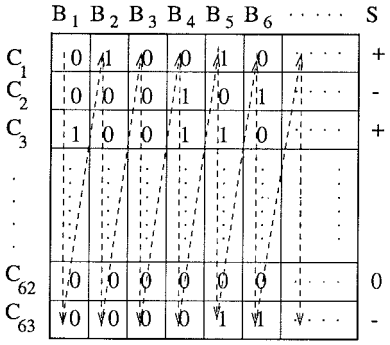

Fig. 1. Bit scanning orders for the layered DCT coding scheme.

Step 1—Block DCT Transform and Coefficient Scaling: As done in JPEG, the input image is partitioned into 8×8 blocks, and the block DCT transform is applied to each block. The DCT coefficients C'_j are scaled with a standard JPEG quantization table Q, i.e., $C_j = \frac{C'_j}{Q_j}$, $j = 0, \cdots, 63$. The scaling is performed to emphasize the visual importance of low frequency components.

Step 2—Coding of DC Coefficients: DC coefficients of neighboring blocks are highly correlated. They can be either encoded with a differential layer coding [4] or encoded in the same way as the JPEG arithmetic coder. It is observed that the performance of the two schemes are very similar. In the experiment reported in Section IV, the latter approach is adopted.

Step 3—Successive Quantization of AC Coefficients: The main differences between the proposed method and JPEG are in the quantization scheme and the entropy coder. After the DCT transform and scaling, JPEG applies a one-step quantization which maps each DCT coefficient to a value in a finite index set. The value is then converted to an intermediate symbol and encoded by an entropy coder. In the proposed new scheme, we adopt a successive quantization procedure which is achieved not in one step but with several successive steps. Roughly speaking, at layer i, the DCT coefficient is only quantized up to the precision of the *significant threshold* T_i. Then, the quantization result of layer i is refined with a smaller significant threshold T_{i+1} at layer $i+1$.

To initialize the process of successive quantization, we set all coefficients as insignificant and search the whole image for the maximum absolute value of the ac coefficients, which is denoted by T_0. Then, we apply the significant identification rule with significant threshold $T_1 = T_0/2$ to construct layer L_1. That is, for each scaled ac coefficient C_j, if its magnitude is greater than T_1, we use symbol "1" to encode its significance, and then record its sign S. Otherwise, we generate symbol "0" to indicate that it is still insignificant. Note that only the significant coefficient needs a sign. In terms of mathematics, we have

$$\begin{array}{llll} C_j > T_1, & B_{j,1} = 1, & S_j = \text{`+'}, & E_{j,1} = C_j - 1.5 \cdot T_1, \\ C_j < -T_1, & B_{j,1} = 1, & S_j = \text{`}-\text{'}, & E_{j,1} = -C_j - 1.5 \cdot T_1, \\ \text{otherwise}, & B_{j,1} = 0, & & E_{j,1} = C_j \end{array}$$

where symbol $E_{j,1}$ in the last column denotes the quantization residue at layer L_1. For each advanced layer L_{i+1}, $i = 1, 2, \cdots$, we refine the significant threshold by half, i.e., $T_{i+1} = T_i/2$, and quantize all ac coefficients accordingly. If the ac coefficient to be coded is insignificant in all previous layers, the significance identification rule is applied. Otherwise, the refinement quantization rule is applied. These two rules can be summarized as follows.

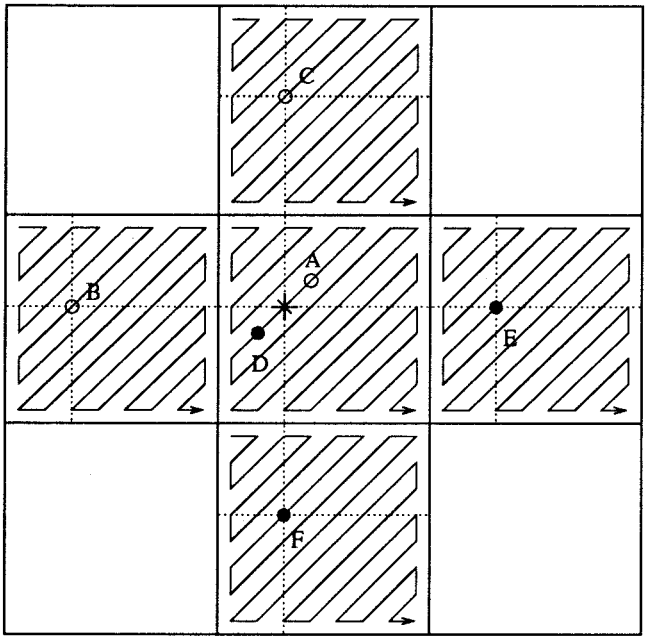

Fig. 2. Illustration of significant symbol coding context, where $*$ is the current coding position, $\circ$ is the significant symbol of current layer, and $\bullet$ is the significant symbol in the previous layer.

1) Rule of significance identification:

$E_{j,i-1} > T_i,\ B_{j,i} = 1, S_j = `+',\ E_{j,i} = E_{j,i-1} - 1.5 \cdot T_i,$
$E_{j,i-1} < -T_i,\ B_{j,i} = 1, S_j = `-',\ E_{j,i} = -E_{j,i-1} - 1.5 \cdot T_i,$
otherwise, $B_{j,i} = 0,\ E_{j,i} = E_{j,i-1}.$

2) Rule of refinement quantization:

$$E_{j,i-1} \geq 0, \quad B_{j,i} = 1, \quad E_{j,i} = E_{j,i-1} - T_i,$$
$$E_{j,i-1} < 0, \quad B_{j,i} = 0, \quad E_{j,i} = E_{j,i-1} + T_i.$$

The generated symbols $B_{j,i}$ and S_j provide a binary representation of the coefficient C_j normalized by T_0 as shown in Fig. 1. These symbols are generated with a decreasing order of importance and will be coded layer by layer in the next step.

Step 4—Context Adaptive Arithmetic Coding: All symbols generated in successive quantization are binary, which can be coded efficiently by an arithmetic coder. Besides, we can predict the location of the significant coefficient with a context adaptive arithmetic coder used in the JPEG extended system. We also encode the refinement symbol and the sign with a specific context. The context adaptive arithmetic coder is a highly efficient entropy coder. Its average coding rate is close to the entropy of the source with low computational complexity. Its implementation only requires addition and shifting operations. There is no need for training or assuming the initial probability distribution so that it is parameter free. The source probability distributions p_0 and p_1 are estimated on the fly and implemented with a look-up table. The source probability distribution is represented as an 8-b (1-byte) status of the coder, which includes 7 b for the probability table and 1 b for the most frequently appearing symbol (MFS). The small coder status enables the construction of a parallel coder for a compound source. We refer to [2] for more details.

For significance identification, our coding context consists of six bits "A" to "F" which are illustrated in Fig. 2. For each circle position, we use 1 b to represent the current significance status of the symbol. Among the 6-b context, "A" and "D" are frequency prediction points inside the current coding block, "B," "C" and "E," "F" are the spatial prediction points of the neighboring blocks. "A," "B," and "C" are coded before the current coding coefficient "*" and therefore belong to the current coding layer. In contrast, "D," "E," and "F" are coded after the current coding coefficient "*" and belong to the previous coding layer. The 6-b context classifies the coding of current coefficient "*" into $2^6 = 64$ categories, and a separate adaptive arithmetic coder is assigned to each category. The layer-by-layer coding turns out to be more efficient than the coefficient-by-coefficient JPEG coding, especially at low bit rates. This can be explained by the following reason. For the layer-by-layer coding, we only predict whether a bit is significant or not. Compared to the prediction of the coefficient value, this task is easier and can be done more accurately.

TABLE I
PERFORMANCE COMPARISON FOR LENA AND BOAT

Image	PSNR	Rate of JPEG			Layered DCT			
		Huff.	Prog.	Arith.	Rate	J-Huff.	J-Prog	J-Arith.
Lena	24.23	0.125	0.086	0.054	0.041	67.2%	52.3%	24.1%
	30.84	0.25	0.234	0.198	0.192	23.2%	17.9%	3.0%
	34.76	0.5	0.497	0.463	0.453	9.4%	8.8%	2.4%
	37.92	1.0	0.977	0.954	0.940	6.0%	3.8%	1.5%
Boat	24.26	0.15	0.127	0.078	0.064	57.3%	49.6%	18.1%
	27.54	0.25	0.225	0.184	0.181	27.6%	19.5%	1.6%
	31.02	0.5	0.481	0.437	0.491	1.8%	-2.1%	-12.4%
	34.56	1.0	0.981	0.931	0.902	9.8%	8.1%	3.1%

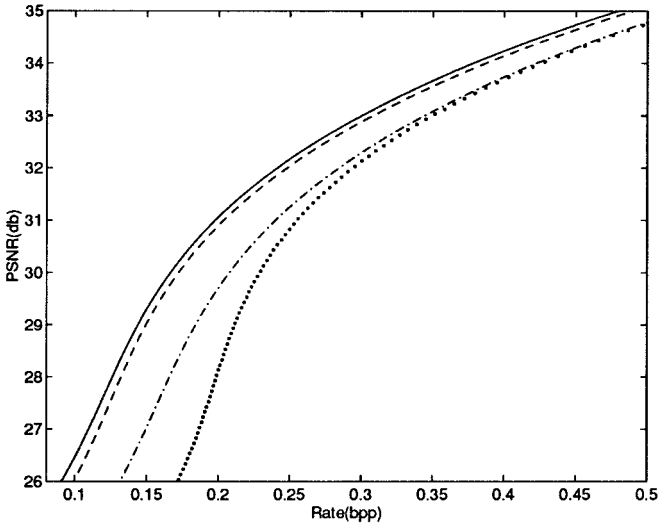

Fig. 3. Rate-distortion performance of Lena using (a) JPEG Huffman coder (dotted line), (b) JPEG progressive coder (dashed-dotted line), (c) JPEG arithmetic coder (dashed line), and (d) the proposed layered DCT coder (solid line).

IV. EXPERIMENTAL RESULTS

Experiments are conducted to compare the new layered DCT coder with the JPEG standard using the Huffman, progressive Huffman, and the arithmetic entropy coders. The images used in the experiments are Lena and Boat of size 512×512. For fair comparison, we strip the header of the JPEG coded bit stream. We use the same quantization factor Q for JPEG Huffman and arithmetic coders, which results in the same coding image quality for both coders. Due to the embedding property of the proposed layered DCT coder, we can truncate the coded bit stream when it reaches the same PSNR values as JPEG. By doing so, we can compare the bit rates for the four coding schemes with the same quality. The results are shown in Table I. The coding bit rate reduction with respect to JPEG Huffman, progressive Huffman, and arithmetic coders are listed in the last three columns of the table. The rate-distortion tradeoff curves for Lena are also depicted in Fig. 3.

We see from the experimental results that the layered DCT coder significantly outperforms the JPEG Huffman coder, progressive Huffman coder, and also outperforms the JPEG arithmetic coder in most cases. The improvement is more substantial when the bit rates become lower. For example, it outperforms the other three coders by about 60%, 50%, and 20%, respectively, when the bit rate is around 0.1 b/p.

The reason that our scheme is superior to the JPEG standard, especially in the low bit-rate range, is attributed to the context adaptive arithmetic coder and intercoefficient prediction. Compared with the 2-D run-level Huffman coder, the arithmetic coder is more efficient in the coding of both stationary and transient sources. The arithmetic coder does not limit the length of the codeword to an integer, so that symbols with a probability close to one can be effectively encoded. Furthermore, the probability state machine and intercoefficient prediction also ensure the arithmetic coder tracks the transition of the source within five to six sample points so that it encodes the irregularity of DCT coefficients efficiently. In low bit rates, DCT coefficients are frequently quantized to zero, with sparse nonzero high-frequency coefficients. The probability of zero DCT coefficient is close to one, which is ideal for the arithmetic coder. When the bit rates become high, the probability distribution of the coefficient tends to be uniform, and the advantage of the arithmetic coder over the Huffman coder is less obvious. Nevertheless, the rate-distortion performance of the layered DCT coder still outperforms that of JPEG for most cases.

In addition to the superior rate-distortion performance, the layered DCT coder possesses the embedding property which makes progressive image transmission and rate-control easier to attain.

REFERENCES

[1] W. B. Pennebaker, *JPEG Still Image Data Compression Standard*. New York: Van Nostrand Reinhold, 1993.
[2] D. L. Duttweiler and C. Chamzas, "Probability estimation in arithmetic and adaptive Huffman entropy coders," *IEEE Trans. Image Processing*, vol. 4, pp. 237–246, Mar. 1995.
[3] J. M. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients," *IEEE Trans. Signal Processing*, vol. 41, no. 12, pp. 3445–3462, 1993.
[4] D. Taubman and A. Zakhor, "Multirate 3-D subband coding of video," *IEEE Trans. Image Processing*, vol. 3, no. 3, pp. 572–588, 1994.

Optimal Wiener Interpolation Filters for Multiresolution Coding of Images

Thomas Sikora

***Abstract*—A design approach is presented which allows the optimization of coefficients for symmetric and separable finite impulse response (FIR) interpolation filters for multiresolution coding schemes. The interpolation filters serve as optimal inverse filters in the Wiener sense and are designed to match the characteristics of the specific filters used for decimation as well as for the statistics of "typical" images to be reconstructed. Applied to the coding of test images in a four-level progressive pyramid scheme, the optimal interpolation filters generated substantially improved rate-distortion results compared to conventional filters.**

***Index Terms*— Hierarchical coding, image coding, image decimation, image interpolation, inverse filtering, Wiener filtering.**

I. INTRODUCTION

The progressive coding and transmission of still images is an important field in interactive image communications [1]–[5]. This technique allows the progressive build-up of images with successively refined quality at the receivers display. Thus, for large images requiring high bandwidth for transmission, basic information about the image content is transmitted within a short period of time, so that it may be possible for the viewer to make a decision whether further transmission is required.

For progressive transmission, images are usually decomposed into a multiresolution representation. Fig. 1 illustrates such a basic decomposition into four levels of a Gaussian pyramid. The original image is represented at the highest Level 0 of the pyramid. Each successive level of the pyramid contains versions of the original image with successively reduced horizontal and vertical resolution. Usually, a lower resolution pyramid image is generated from the image at the next higher level of the pyramid by low-pass filtering and subsampling by a factor of two in each image dimension. In many applications, low-pass filters with few taps are employed to retain low implementation complexity.

For progressive coding and transmission of images in a multiresolution approach, the Laplacian image pyramid introduced by Burt and Adelson in [2] has attracted particular attention. Using this scheme, the lowest resolution level image is encoded first. The decoded version of this lowest level image is next interpolated to serve as a prediction of the image content at the next higher resolution level. Only the residual information (interpolation error) at the higher resolution level is coded and transmitted to the receiver. This technique is repeated to encode and transmit the information of each higher level in the pyramid. The overall rate-distortion performance of the progressive Laplacian image coding scheme relies heavily on the accuracy of the interpolation filters in the different levels of the pyramid [5]. The purpose of this paper is to address the problem of designing optimal interpolation filters for progressive multiresolution coding schemes.

Optimal sample rate conversion and adaptive interpolation has been investigated by a number of authors [6]–[11]. Steele and Benjamin have reported in [12] an adaptive interpolation approach

Manuscript received November 21, 1995; revised November 4, 1996. This paper was recommended by Associate Editor Y. Wang.

The author is with the Department Image Processing, Heinrich-Hertz-Institute (HHI) Berlin GmbH, 10587 Berlin, Germany.

Publisher Item Identifier S 1051-8215(97)02239-8.