11-29-1993

Simple high-quality lossy image coding scheme

Shaolin Bi

Khalid Sayood

Journal of Electronic Imaging 3(1), 81–91 (January 1994).

Simple high-quality lossy image coding scheme

Shaolin Bi
Khalid Sayood
University of Nebraska-Lincoln
Department of Electrical Engineering
Lincoln, Nebraska 68588-0511

Abstract. *A simple yet efficient image data compression method is presented. This method is based on coding only those segments of the image that are perceptually significant to the reconstruction of the image. Sequences of image pixels whose gray-level differences from the pixels of the previous row exceed two prespecified thresholds are considered significant. These pixels are coded using a differential pulse code modulation scheme that uses a 15-level recursively indexed nonuniform quantizer for the first pixel in a segment and a 7-level recursively indexed nonuniform quantizer for all other pixels in the segment. The quantizer outputs are Huffman coded. Simulation results show that this scheme can obtain subjectively satisfactory reconstructed images at low bit rates. It is also computationally very simple, which makes it amenable to fast implementation.*

1 Introduction

Because of the large amount of data required to represent an image, the past decades have seen the appearance of a large number of image compression algorithms. Many of these algorithms are designed to minimize some mathematically tractable distortion measure. In image coding this is generally some form of average squared error, such as normalized mean squared error (NMSE), signal-to-noise ratio (SNR), and peak signal-to-noise ratio (PSNR). Because the final consumers of the information are humans who do not necessarily use the squared error distortion measure as a performance metric, a design that maximizes the PSNR may not necessarily provide visually acceptable reconstructed images. Unfortunately, a definitive model of the human visual system (HVS) does not yet exist that can be used for developing or evaluating image compression algorithms. This is not to say that there have not been significant contributions to the development of an HVS model, and algorithms that make use of these models.

The best-known model in the compression literature is the modulation transfer function (MTF) model developed by Mannos and Sakrison.[1] This model is used by Griswold[2] to weight a "cosine energy function," which is then used to obtain the bit allocation for a discrete cosine transform (DCT) coding scheme. Nill[3] has shown how to use a transformation of basis, so as to use an MTF model developed using a sine basis with the DCT. The MTF models incorporate global properties of the HVS. Therefore, when they are incorporated into image compression schemes, they provide little local sensitivity. Because images are globally nonstationary, this can be a drawback. This drawback can be overcome to some extent by using the MTF over small sections of the image, such as an 8×8 block.

Compression schemes that operate in the spatial domain avoid this drawback by using the properties of the HVS to drive local adaptation strategies. Netravali and Prasada[4] use the properties of the HVS to develop 2-D masking functions. These masking functions are used on a local level to improve the performance of a differential pulse code modulation (DPCM) system. Ramamoorthy and Jayant,[5] Ramamurthi and Gersho,[6] and Parthasarathy, Thygarajan, and Abut,[7] among others, have developed codebook design techniques for vector quantizers (VQ).

Some of these techniques, such as vector quantization, impose an artificial block structure. This can result in an unnecessary expenditure of resources. For example, a block may be classified as containing an edge and, therefore, need

Paper 93-023 received April 23, 1993; revised manuscript received Nov. 11, 1993; accepted for publication Nov. 29, 1993.
1017-9909/94/$6.00 © 1994 SPIE and IS&T.

additional resources, even though only a small fraction of the pixels in the block actually belong to an edge.

In this paper we present a spatial compression technique that imposes minimal structure while making use of the idea of just-noticeable distortion. The method introduced in this paper processes an image in the spatial domain row by row, from top to bottom. Segments of image pixels of varying lengths called *significant segments* are extracted using two prespecified thresholds. The thresholds are sufficiently robust to changes in image statistics to be considered image independent. These segments represent perceptually significant accumulated gray-level differences in the vertical direction. In the compressed form only the significant segments and their locations are stored/transmitted. The remainder of the image is discarded. At the decoder the image is reconstructed using only the significant segments. In addition to the pixel values in a significant segment, the address and length of the segment must also be specified for reconstructing the image.

Simulation results are presented to show the efficiency of the scheme. Two types of simulations were conducted. One set of simulations were conducted to obtain "acceptable" quality images. We call this *level-1 compression.* Another set of simulations were conducted to obtain "transparent" quality images. A reconstructed image is said to be of transparent quality if, when it is displayed side by side with the original, it takes a trained observer more than 20 s to discover a coding distortion.[5] As reported by Ramamoorthy and Jayant,[5] this is a rather stringent requirement and many simple and two-stage VQ schemes could not pass this requirement. We call compression resulting in transparent quality *level-2 compression.*

In the following sections we describe the proposed scheme in more detail. The simulation results are presented in Sec. 6 with a summary in Sec. 7.

2 Extraction of Significant Segments

As mentioned in Sec. 1, we consider significant segments to be those portions of an image that are essential for its reconstruction. A significant segment is defined as follows:

Definition. Given an image of size $M \times M$, where the current pixel value is denoted by $f(i,j)$ and the reconstructed pixel value denoted by $\tilde{f}(i,j)$, where i is the row number and j is the column number, we define a segment of pixels from (i,j) to $(i,j+n-1)$ to be significant if

$$|f(i,k)-\tilde{f}(i-1,k)| \geq T_1, \qquad k=j,\ ...,\ j+n-1 \tag{1}$$

and

$$\sum_{k=j}^{j+n-1} |f(i,k)-\tilde{f}(i-1,k)| \geq T_2\ , \tag{2}$$

where T_1 and T_2 are two thresholds and n is the length of the segment. The segment length n can vary from segment to segment. For computational ease we impose the restriction that a significant segment be confined to a single row.

The first condition guarantees that every pixel in a significant segment has a gray-level-value difference greater than or equal to T_1 from the reconstructed pixel in the row above. The second condition says that the sum of gray-level differences of all the pixels in the segment is greater than or equal to T_2.The first row of pixels in an image is considered to be a significant segment of length M.

While the need for the first condition is self-evident, the need for the second condition is less so. The first condition ensures that the auxillary segments are close, on a point-to-point basis, to a significant segment. Therefore, if the auxillary segments are not available, they can be obtained from the significant segment. The second condition arose as an outcome of our experiments to determine the value of T_1. We found that values of T_1 that permitted a perceptually acceptable or transparent reproduction tended to declare a large number of short segments to be significant, even though when we forced these segments to be auxillary, the perceptual quality did not change. If we put a restriction on the segment length (forbidding small segments) this prevented segments with large (perceptually significant) deviations from the previous row to be declared significant. The second condition effects a compromise between these two situations. If the deviation between the segment being examined and the previous row is large, the second condition could be satisfied regardless of the length of the segment. However, if the deviations are small, the segment length will have to be large for the second condition to be satisfied.

The thresholds T_1 and T_2 are chosen to achieve designated bit rates or reconstructed image qualities. Larger values for T_1 and T_2 give lower bit rates but introduce more distortion in the reconstructed image and vice versa. Because evaluation of lossy image data compression schemes is generally a subjective matter, it is difficult to obtain a mathematical relationship between the values of T_1 and T_2 and the subjective quality of the reconstructed image. Therefore, it is difficult to determine the values of T_1 and T_2 analytically. The values of T_1 and T_2 in our simulations were obtained based on experimentation. Fortunately, we found that T_1 and T_2 were robust to changing image statistics, and as can be seen from the simulation results, for the same values of T_1 and T_2, we obtained reconstructed images with very similar qualities and bit rates.

The address and length of each significant segment must be transmitted to the receiver. This can constitute significant overhead. To reduce this overhead, significant segments that are separated by less than d pixels are connected into one segment. This has the effect of reducing the overall bit rates. For the original images in Figs. 1 and 2 at $T_1=8$ and $T_2=26$, Figs. 3 and 4 are the significant segment maps.

When the decoder has received the coded pixel values in the significant segments along with the location of these segments, the image is reconstructed using the following procedures:

1. The reconstructed pixels in the row above are first filtered using the following low-pass filter and then copied to the corresponding pixels in the current row,

$$\tilde{f}(i,j)=\frac{1}{4}[\tilde{f}(i-1,\ j-1)+2\tilde{f}(i-1,\ j) +\tilde{f}(i-1,\ j+1)]\ . \tag{3}$$

2. If the current pixel being decoded is part of a significant segment, the pixel values obtained from procedure 1 are replaced by the decoded pixel values received.

Fig. 1 "Lena" original image.

Fig. 3 Significant segments in the "Lena" image.

Fig. 2 "Tiffany" original image.

Fig. 4 Significant segments in the "Tiffany" image.

We see that a new row in an image is reconstructed by either copying the low-pass-filtered version of the pixels in the above row or by assigning pixel values in the significant segments. Procedure 1 is used to smooth the reconstructed image. Originally in our simulation the pixel values in the above row were copied directly to the pixels in the current row. We found that there were some noticeable vertical lines in the reconstructed image, as shown in Figs. 5 and 6. The cause of these vertical lines is obvious from the nature of the scheme. Procedure 1 solved this problem to a great extent. Besides smoothing these vertical lines in the reconstructed image, procedure 1 also has the effect of improving the performance of the scheme, i.e., lowering the bit rate or enhancing the reconstructed image quality. We see that Eq. (3) is in fact a predictor of order 3. Better predictors might exist but this one worked better than the others we tried. In level-1 compression the value of T_1 is around the limit of the range of gray-level changes that the HVS can detect. Therefore, some reconstruction artifacts exist in spite of procedure 1. A postsmoothing algorithm was developed to deal with this problem and is described in Sec. 5.

3 Quantization and Coding

Image data in the significant segments must be stored or transmitted for the reconstruction of the image. In our initial simulations each pixel in the significant segments was quantized and coded. However, it was later found that for level-1 compression if only one pixel in two is quantized and coded, i.e., 2:1 subsampling, and the unsampled pixel values estimated, the overall bit rates can be further reduced, or a better reconstructed image can be achieved at the same bit rate. With this approach some horizontal resolution is sacrificed to improve the overall quality of the reconstructed image. Various methods could be used to interpolate the unsampled pixels in the segment. In the current work we use mean interpolation, i.e., the average of the left and right pixels, to estimate the unsampled pixels. We found that this simple approach produced results as good or better than other, more

Fig. 5 Nonsmoothed reconstruction of the "Lena" image.

Fig. 6 Nonsmoothed reconstruction of the "Tiffany" image.

complicated, interpolation schemes. This subsampling cannot be used for level-2 compression or transparent coding.

A DPCM scheme[8] with a 2-D linear predictor[9] is used to code the image data in the significant segments. The first pixel in a significant segment is more important to the prediction of the following pixels in the same segment and its prediction error variance is greater than those of the other pixels in the segment. For level-1 compression, a 15-level max nonuniform quantizer[10] and a 7-level Max nonuniform quantizer are used to quantize the prediction errors for the first pixel and the remaining pixels in the segment, respectively. To reduce the quantization errors while keeping the bit rate relatively low, in level-2 compression, a 15-level recursively indexed[11] Max nonuniform quantizer and a 7-level recursive Max nonuniform quantizer are used for the first pixel and the remaining pixels in the segment, respectively. The distributions of the inputs to the 15- and 7-level quantizers were both supposed to be Gaussian and the quantizer output points in Ref. 10 were used directly. The quantizer input in both cases is the difference between the current pixel value and the predicted value, which is computed as follows:

$$\hat{f}(i,j)=\tilde{f}(i,j)\ , \tag{4}$$

for the first pixel in a segment, and by

$$\hat{f}(i,j)=0.75\tilde{f}(i,j-1)+0.75\tilde{f}(i-1,j)-0.5\tilde{f}(i-1,j-1) \tag{5}$$

for the rest of the pixels in the segment in transparent reconstruction, or by

$$\hat{f}(i,j)=0.75\tilde{f}(i,j-2)+0.75\tilde{f}(i-1,j)-0.5\tilde{f}(i-1,j-2) \tag{6}$$

for level-1 compression with subsampling, where $\hat{f}(i,j)$ is the predicted value of $f(i,j)$ and $\tilde{f}(i,j)$ is the reconstructed pixel value. Equation (5) is obtained from Ref. 12. Other prediction functions were tried and were found inferior to the current ones. All of the pixel values in the first row of the image are coded using a DPCM system with a 15-level nonuniform quantizer.

To match the dynamic range of the Max quantizers, the quantizer input is scaled by a factor λ that is experimentally chosen. A small value of λ will blur the image and a large value λ will cause the bit rates to increase without a corresponding increase in image quality. In our simulations, λ is chosen so that the quantizers have a small amount of overload (several hundreds pixels for the 7-level quantizer and dozens for the 15-level quantizer). The value of λ is fixed for the images used in this simulation.

Fixed sets of Huffman codes are used for the 15- and 7-level quantizers for the same categories of reconstructed images. These Huffman codes were obtained from the combined frequencies of the quantizer outputs in images "Lena" and "Tiffany." Table 1 shows the output points, the frequencies that the output points are used, and the corresponding Huffman codewords for level-1 compression. The Huffman codewords were obtained similarly for level-2 compression.

4 Data Structure

Along with the actual DPCM-coded pixel data, for every significant segment we also have to send some overhead information, which informs the decoder of the position and length of the significant segment. This information constitutes a large part of the total information needed to represent an image. For example, in monochrome images, overhead information constitutes nearly half of the total information in the level-1 compression and it takes about 0.35 bit/pixel in level-2 compression. We describe how we encode overhead information in the following.

The position of a significant segment is given by the distance from the beginning of the current significant segment to the end of the previous significant segment. Here we introduce a concept of an empty segment. If $L(L=2^m$, m is an integer) consecutive pixels are not part of a significant segment, these L consecutive pixels are labeled as an empty segment. An empty segment is represented by a 0. If the distance between two significant segments is D, then there

Table 1 Maximum quantizer outputs, frequency of outputs, and Huffman codewords for a level-1 compression.

7-Level Quantizer				15-Level Quantizer			
Quantizer Output	Frequency Lena	Frequency Tiffany	Codeword	Quantizer Output	Frequency Lena	Frequency Tiffany	Codeword
0.0	13343	16947	0	0.0	1018	653	0001
0.5606	4942	4996	11	0.2739	3919	4228	1
-0.5606	4470	4197	100	-0.2739	2032	2168	01
1.188	1105	818	1011	0.5548	601	699	0000
-1.188	982	743	10100	-0.5548	470	636	0010
2.033	144	39	101011	0.8512	114	337	00110
-2.033	253	141	101010	-0.8512	96	106	001111
				1.175	16	65	0011100
				-1.175	19	33	00111010
				1.546	0	10	001110111
				-1.546	13	8	0011101100
				2.007	0	0	0011101101011
				-2.007	3	2	001110110100
				2.681	0	2	00111011011
				-2.681	0	0	0011101101010

are $K_1 = \lfloor D/L \rfloor$ empty segments in it and K_1 bits of 0's will be used to represent these K_1 empty segments. The remaining value of $D-(K_1 \times L)$ is represented by an m-bit binary number that is preceded by a 1 to indicate the end of a sequence of empty segments. The remaining distance can also be Huffman coded. However, from our simulations we found that the performance improvement obtained is minimal.

To see how the coding works, consider the following example. Suppose $L=16(m=4)$ and $D=71$, then D would be represented by the codeword 000010111. In this codeword the first four 0's represent four empty segments at a length of 16 each. The first 1 after the four 0's is a marker to show that the next 4 bits represent the remainder of the distance, which in this case is seven. Therefore, $D=4\times 16+7=71$. As mentioned in Sec. 2, segments with distance between them less than or equal to d are connected together to form a single segment. Therefore, we know that all of the distances between segments are greater than or equal to $d+1$ and, hence, we can always subtract $d+1$ from the distance. This will further reduce the number of bits required to represent the distance. For reasons of computational convenience, only segments in the same row were connected in our simulation.

The length of the significant segment has to be specified to separate the binary numbers representing the position of the segment from the binary numbers representing the pixel data. The length of the segment is represented by a t-bit binary word. If we specify t to be too large, then during the times that specifying the length of a significant segment would have required fewer bits we would be wasting resources; however, specifying t to be too small would limit the length of the significant segment. This would lead to breaking up a long segment into smaller segments, which in turn would result in an increase in the overhead. To avoid this problem we employed recursive indexing[11] to represent the length of the significant segments. The recursive indexing scheme functions as follows: Given t bits we can represent values between 0 and 2^t-1. If the actual value we need to represent is less than 2^t-1, we send the t-bit codeword corresponding to that value. If however the value is greater than or equal to 2^t-1, we send the codeword corresponding to 2^t-1, and subtract 2^t-1 from the value. If the remainder is less than 2^t-1, then we send the t-bit code corresponding to the difference. If not, we repeat the process. This process is repeated until the remainder is less than 2^t-1. For example, suppose $t=3$ and we wished to encode a value of 5. Because 5 is less than 7 we would simply transmit the codeword 101. A value of 9 however would be encoded as 111010. The receiver, on seeing the 111 codeword, recognizes that the index will be 7 + whatever comes after, and keeps adding the values corresponding to the 3-bit codewords until it obtains a value less than 7.

Table 2 Data structure for a segment.

Position	Length	First Pixel	Following Pixels
K_1+m+1 bits	$K_2 t$ bits	1 to 14 bits	1 to 6 bits / each

Note that in level-1 reconstruction the segment pixels are subsampled as described in Sec. 3, and, hence, the real length of the segment is about two times the number of the sampled pixels. If we define the minimum length of the segment to be $l_{\min}$, we can also subtract $l_{\min}$ from the binary number representing the length of the segment as we do for the position of a segment. Suppose the length of a segment is l, it would be represented by $K_2 = \lfloor (l-l_{\min})/(2^t-1) \rfloor + 1$ binary numbers, and the number of bits to be used is $K_2 t$. Note also that in transparent quality image compression if we restrict the length of significant segments to be even, we can also use this method to reduce the overhead information.

Table 2 shows the data structure and bits required to represent a segment. Table 2 shows that a significant segment is represented by three pieces of information: the position of the segment, the length of the segment, and the actual pixel data in the segment. If recursive quantizers are used, the pixel data will be different from the ones shown in the table. For example, suppose we have a significant segment with length 12 that is at a distance of 63 from the last significant segment. If we define $L=16$, $d=8$, $t=3$, and $l_{\min}=1$, then there are $(63-9)/16=3$ empty segments. The remaining distance is $63-9-3\times 16=6$, so the position of the segment can be represented by 00010110. By subsampling, there are $12/2=6$ pixels left in the segment. Now the length of the segment can be represented by $\lfloor (6-1)/7 \rfloor + 1 = 1$ binary number of 3 bits, i.e., 101.

The coding of the overhead information, as described above, is basically run-length coding with some small tricks in coding the "run" and the "length." Besides the method of coding the overhead information described above, we tried the method described in Ref. 13, which differentially addresses the significant segments. We also tried to use a special "end-of-segment" symbol to indicate the ending of the segment data. Both of these produced worse results than the method described above.

5 Image Postsmoothing

As mentioned in Sec. 2, because the values of threshold T_1 are close to the range of gray-level changes that human eyes can detect in level-1 compression, some reconstruction artifacts are noticeable, especially in low-activity (quasiconstant) regions. The effect is that in these regions if you look at the image carefully you can notice the gray-level changes

caused by the significant segments and low-pass filtering. A postsmoothing algorithm is developed to solve this problem. Usually smoothing will eliminate image details to some extent that will, in turn, degrade the subjective quality of the image. Our method, introduced below, tries to avoid this problem.

We know that significant segments in our scheme represent gray-level changes and they are usually located on edges and contours of objects in the image. To preserve the details of the image, only those pixels that are not on segments can be smoothed, while pixels on segments should not be touched. But segments also exist in some quasiconstant regions such as the forehead and the cheeks of the face, where abrupt gray-level changes would be very noticeable in the reconstructed images. Because the density of segments in these regions is lower than the density in other regions with higher gray-level activities, we can identify the pixels in these regions as ''scattered'' segment pixels and smooth them. Labeling is carried out in a moving local window: Suppose the current pixel is in the center of the window. If the following two conditions are both satisfied, that is, the current pixel is on a segment and the number of pixels not on segments in the local window exceeds a threshold T_3, the current pixel is labeled as ''scattered'' and is smoothed.

A 3×3 window and $T_3=6$ is used in our simulation. Smoothing is implemented by assigning the average gray-level values of the pixels in the window to the current pixel. We can see that this is a conservative smoothing procedure at $T_3=6$, i.e., the current pixel is smoothed only when there is at most one other pixel on a significant segment in the window. Smoothing can be carried out several times for the same reconstructed image. It should be pointed out that postsmoothing is only carried out in low-rate (level-1) compression. In level-2 or transparent coding the reconstructed image is so good that postsmoothing is unnecessary.

6 Simulation Results

We ran two types of simulations for both monochrome and color images, in which color images were processed in the Y, I, and Q planes, separately. In the first case we found thresholds T_1, T_2, and the scale factor λ for which we obtained images of ''acceptable'' quality. These images had PSNR values of around 30 dB. The PSNR for images of size $M\times M$ is defined as

$$\mathrm{PSNR(dB)}=10\ \log_{10}\frac{(255)^2}{1/M^2\ \sum_{i,j=1}^{M}[f(i,j)-\tilde{f}(i,j)]^2}\ , \qquad (7)$$

and the bit rate is defined as

$$\mathrm{bits/pixel}=\frac{\text{total bits used}}{M^2}\ . \qquad (8)$$

In all of the simulations, the image sizes were 512×512. We chose the empty segment length $L=16$, connecting gap $d=4$. To recursively represent the length of the significant segments, we chose the number of binary bits t to be two for the low-rate case and three for the transparent case. For a given quality we used the same coder parameters for all images; that is, the coder parameters are quality dependent, not image dependent. This allows for a straightforward implementation of the compression scheme. In the case of color images, we used two different values of T_2 (T_{21} and T_{22}) for the even and odd rows for the I and Q planes. We found that allowing slightly higher distortion in every other row of the I and Q planes did not perceptually degrade the overall quality while lowering the average rate.

Fig. 7 Low-rate reconstruction of the "Lena" image (rate = 0.51 bit/pixel).

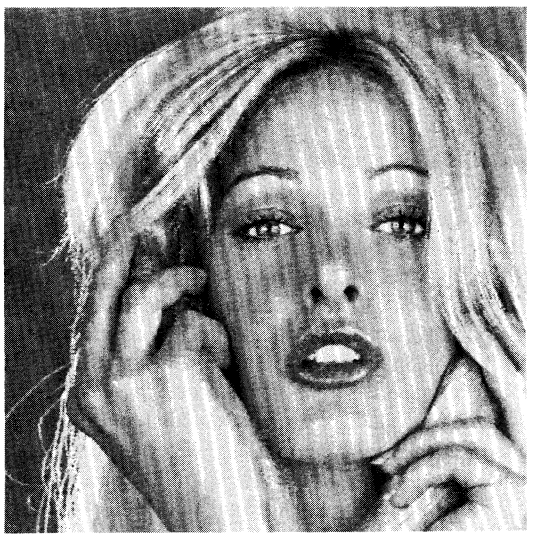


Fig. 8 Low-rate reconstruction of the "Tiffany" image (rate = 0.55 bit/pixel).

The monochrome images coded using the acceptable criterion, or level-1 compression, are shown in Figs. 7 and 8. Another set of images and their level-1 compression are shown in Figs. 9 through 12. A set of color images (printed here in black and white) and their level-1 compression are shown in Figs. 13 and 14. The monochrome images were coded using an average rate of approximately 0.52 bit/pixel while the average rate for the color images ranged from 1.08 bits/pixel for the popular ''Lena'' image to 1.34 bits/pixel for the high-detail ''Peppers'' image, as shown in Tables 3 and 4. The PSNRs for the reconstructed images in level-1

Fig. 9 "F-16" original image.

Fig. 11 Low-rate reconstruction of the "F-16" image (rate = 0.52 bit/pixel).

Fig. 10 "Peppers" original image.

Fig. 12 Low-rate reconstruction of the "Peppers" image (rate = 0.51 bit/pixel).

compression are computed before the postsmoothing is performed.

Thresholds and scale factors were obtained to get transparent quality images. The results are shown in Tables 5 and 6 and Figures 15 and 16. To obtain transparent quality images, we did not subsample the significant segments. However, we restricted the length of the significant segments to be even and the minimum length to be two. In this way, we can still use half of the bits to represent the length of the segments just as we did in the subsampled cases.

Because this is a differential encoding scheme in the spatial domain, we compared our results to a DPCM system with entropy coding. The proposed system significantly outperforms entropy-coded DPCM. For example, taking the "Lena" image, DPCM with Huffman coding gives a bit rate of 2.57 bits/pixel and a PSNR of 36.23 dB with a 15-level max nonuniform quantizer, and a bit rate of 1.26 bits/pixel and PSNR of 30.50 dB with a 7-level max nonuniform quantizer. The qualities of the reconstructed images in these two cases are similar to those of the transparent and low-rate reconstructed images by our scheme. However, the rates for the transparent quality and low-rate images are 1.15 and 0.51 bits/pixel, respectively. Thus it takes roughly twice the bit rates for entropy-coded DPCM to obtain reconstructed images of similar quality as the proposed scheme.

We also compared the proposed scheme with the Joint Photographic Experts Group (JPEG) implementation developed by the Independent JPEG Group. The PSNR performance at lower rates (around 0.5 bit/pixel) was 2 to 3 dB higher for the JPEG implementation. However, the performance at level-2 rates was more comparable. The PSNR for the "Tiffany" and "Peppers" images was higher for the proposed system, and the PSNR for "Lena" and "F-16" was higher for the JPEG implementation. In the subjective comparisons, the level-1 results for JPEG were slightly better

Fig. 13 Original color images presented in black and white: (a) "Lena," (b) "Tiffany," (c) "F-16," and (d) "Peppers."

than those for the proposed system, while the level-2 results were very similar. This performance advantage is obtained at the cost of more complexity. In our view, the JPEG algorithm has a substantially more complex implementation, and in multimedia applications in which special purpose hardware may not be available, its implementation may be problematic. The proposed algorithm is a spatial domain algorithm that is easily amenable to low-complexity implementation.

7 Summary

We have presented a simple and efficient image data compression method in this paper. Segments of image pixels are obtained by an extraction procedure. Segment pixels are coded with a DPCM scheme that uses a 15-level nonuniform quantizer for the first pixel in a segment and a 7-level nonuniform quantizer for the remaining pixels. Simulation results show that this method can achieve very satisfactory reconstructed images at low bit rates. The other advantages of this method are low complexity, fast implementation, and image independence.

Acknowledgment

This work was supported by a grant from the NASA Lewis Research Center, Cleveland, Ohio (NAG 3-806).

References

1. J. L. Mannos and D. J. Sakrison, "The effect of a visual fidelity criterion on the encoding of images," *IEEE Trans. Inf. Theory* **IT-20,** 525–536 (July 1974).
2. N. C. Griswold, "Perceptual coding in the cosine transform domain," *Opt. Eng.* **19,** 306–311 (May/June 1980).
3. N. B. Nill, "A visual model weighted cosine transform for image compression and quality assessment," *IEEE Trans. Commun.* **33,** 551–557 (June 1985).
4. A. N. Netravali and B. Prasada, "Adaptive quantization of picture signals using spatial masking," *Proc. IEEE* **65,** 536–548 (April 1977).
5. V. Ramamoorthy and N. S. Jayant, "High quality image coding with a model testing vector quantizer and a human visual system model,"

Fig. 14 Low-rate reconstructions of color images presented in black and white: (a) "Lena" (rate = 1.08 bits/pixel), (b) "Tiffany" (rate = 1.20 bits/pixel), (c) "F-16" (rate = 1.07 bits/pixel), (d) "Peppers" (rate = 1.34 bits/pixel).

Table 3 Simulation results for monochrome images using level-1 compression ($T_1 = 8$, $T_2 = 26$, $\lambda = 0.02$, $t = 2$).

Image	Bits/pixel		No. of Segments	PSNR(dB)
	Huffman Coding	No Huffman Coding		
Lena	0.51	0.69	7529	29.58
Tiffany	0.55	0.75	8229	29.73
F-16	0.52	0.72	6746	30.73
Peppers	0.51	0.68	7817	30.42

Table 4 Simulation results for color images using level-1 compression ($\lambda = 0.02$, $t = 2$; Y plane: $T_1 = 8$, $T_2 = 26$; I plane: $T_1 = 5$, $T_2^1 = 25$, $T_2^2 = 25$; Q plane: $T_1 = 4$, $T_2^1 = 15$, $T_2^2 = 25$).

Image	Bits/pixel and PSNR(dB)						
	Total Rate	Y Plane		I Plane		Q Plane	
	Rate	Rate	PSNR(dB)	Rate	PSNR(dB)	Rate	PSNR
Lena	1.08	0.49	30.37	0.32	33.56	0.27	36.10
Tiffany	1.20	0.45	31.13	0.31	33.88	0.44	33.92
F-16	1.07	0.52	31.26	0.21	34.96	0.34	35.05
Peppers	1.34	0.46	31.47	0.42	32.58	0.46	34.17

in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, pp. 1164–1167 (April 1988).
6. B. Ramamurthi and A. Gersho, "Image vector quantization with a perceptually based cell classifier," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, 32.10.1–32.10.4 (1984).
7. K. S. Thygarajan, S. Parthasarathy, and H. Abut, "A matrix quantizer incorporating the human visual model," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, 141–144 (1985).
8. C. C. Cutler, "Differential quantization for television signals," U.S. Patent No. 2 605 361 (July 29, 1952).
9. K. Sayood and S. M. Schekall, "Use of the ARMA predictors in the differential encoding of images," *IEEE Trans. Acoust., Speech, Sig. Process.* **36,** 1791–1795 (November 1988).
10. J. Max, "Quantizing for minimum distortion," *IRE Trans. Inf. Theory* **IT-6,** 7–12 (March 1960).

Table 5 Simulation results for monochrome images using level-2 compression ($T_1 = 4$, $T_2 = 15$, $\lambda = 0.04$, $t = 3$).

Image	Bits/pixel		No. of Segments	PSNR(dB)
	Huffman Coding	No Huffman Coding		
Lena	1.15	1.65	9247	33.54
Tiffany	1.30	1.83	9630	35.89
F-16	1.06	1.60	7809	36.38
Peppers	1.29	1.75	9955	35.84

Table 6 Simulation results for color images using level-2 compression ($\lambda = 0.04$, $t = 3$; Y plane: $T_1 = 3$, $T_2 = 10$; I plane: $T_1 = 3$, $T_2^1 = 10$, $T_2^2 = 25$; Q plane: $T_1 = 3$, $T_2^1 = 10$, $T_2^2 = 25$).

Image	Bits/pixel and PSNR(dB)						
	Total Rate	Y Plane		I Plane		Q Plane	
	Rate	Rate	PSNR(dB)	Rate	PSNR(dB)	Rate	PSNR
Lena	2.33	1.17	34.35	0.68	36.87	0.48	37.95
Tiffany	2.51	1.19	37.10	0.66	37.43	0.66	36.92
F-16	1.91	1.17	37.06	0.42	39.03	0.32	38.92
Peppers	3.01	1.41	36.57	0.90	35.85	0.70	36.95

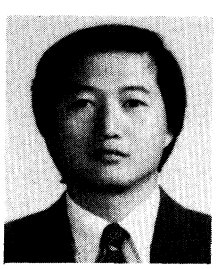
(a) (b) (c) (d)

Fig. 15 Transparent quality reconstructions: (a) "Lena" (rate = 1.15 bits/pixel), (b) "Tiffany" (rate = 1.30 bits/pixel), (c) "F-16" (rate = 1.06 bits/pixel), (d) "Peppers" (rate = 1.29 bits/pixel).

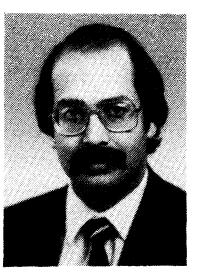
(a) (b) (c) (d)

Fig. 16 Transparent quality reconstructions of color images presented in black and white: (a) "Lena" (rate = 2.33 bits/pixel), (b) "Tiffany" (rate = 2.51 bits/pixel), (c) "F-16" (rate = 1.91 bits/pixel), (d) "Peppers" (rate = 3.01 bits/pixel).

11. K. Sayood and S. Na, "Recursively indexed quantization of memoryless sources," *IEEE Trans. Inf. Theory* **IT-38,** 1602–1609 (Sep. 1992).
12. M. Rabbani and P. W. Jones, *Digital Image Compression Techniques,* Vol. **TT7,** SPIE Optical Engineering Press (1991).
13. B. G. Haskell, "Differential addressing of clusters of changed picture elements for interframe coding of videotelephone signals," *IEEE Trans. Commun.* **COM-24,** 140–144 (January 1976).



Shaolin Bi received his BE in electrical engineering from the Shanghai Institute of Railway Technology, China, in 1983. He received his MS degree in electrical engineering from the Graduate School, Academia Sinica, China, in 1986. He is currently a PhD student in the Department of Electrical Engineering at the University of Nebraska-Lincoln. His research interests include data compression, source coding, image processing, and communications networks.



Khalid Sayood received his undergraduate education at the Middle East Technical University, Ankara, Turkey, and the University of Rochester, New York. He received his BS and MS degrees from the University of Rochester and the PhD degree from Texas A&M University, College Station, in 1977, 1979, and 1982, respectively, all in electrical engineering. He joined the University of Nebraska–Lincoln in 1982 where he is a professor of electrical engineering. His current research interests include data compression, joint source/channel coding, communication networks, and biomedical applications. He is a member of Eta Kappa Nu and Sigma Xi.