JOURNAL OF VISUAL COMMUNICATION AND IMAGE REPRESENTATION
Vol. 4, No. 3, September, pp. 225–241, 1993

Image Compression Using Adaptive Multilevel Block Truncation Coding[1]

SHYI-CHYI CHENG*,[2] AND WEN-HSIANG TSAI†

*Institute of Computer Engineering and Information Science and †Department of Computer and Information Science, National Chiao Tung University, Hsinchu, Taiwan 300, Republic of China

Received August 15, 1991; accepted May 11, 1992

The block truncation coding (BTC) algorithm for image compression has the advantages of low computation load and less memory requirement. In this paper, an adaptive image compression algorithm using multilevel BTC is proposed. An input image is partitioned into blocks with variable sizes, and the gray values of each block are adaptively quantized to be one, two, or four levels according to local image statistical characteristics. Depending on the amount of detail or variation among the pixels, an appropriate number of bits are allotted to code the block under the constraint of producing no larger than a given mean square error value. Experimental results show that the proposed method is comparable in computation time with the standard BTC and the absolute moment BTC methods, and the resulting images are much better, being less distorted and having higher compression ratios.

1. INTRODUCTION

Image compression has wide applications in broadcast television, remote sensing, aerial photography, teleconferencing, computer communication, facsimile transmission, etc. Image compression is also useful for reduction of storage which is commonly required for keeping office documents, remote sensed data, medical images, etc. Because of its wide applications, image compression is an important research area of digital image processing.

The block truncation coding (BTC) algorithm [1], unlike other image compression methods such as transformation coding [6] and vector quantization [7], requires less computation effort. It also has the advantage of requiring less memory. The BTC algorithm developed by Delp and Mitchell [1] is a two-level nonparametric quantizer whose output levels are obtained by preserving the first- and second-order moments of input samples. Based on the moment-preserving principle, other coding algorithms for improving the performance of the BTC algorithm have also been proposed [2–4]. These methods can be employed to compress an input image with less distortion than the BTC method. The fidelity measure used to evaluate the compression result is the mean square error (MSE) defined as

$$MSE = \frac{1}{m}\sum_{i=1}^{m}(f_i - g_i)^2, \tag{1}$$

where f_i represents a pixel gray value of the original image, g_i is the estimated value for f_i, and m is the number of pixels in the image. The BTC algorithm was modified by Halverson, Griswold, and Wise [2] to preserve the first-, second-, or even third-order sample moments. Instead of preserving the sample moments, Lema and Mitchell [3] preserve the absolute moments for designing the two-level quantizer. The speed of this method is also faster than that of the BTC method. Furthermore, Hui [4] designed a minimum mean square error quantizer for the BTC algorithm. Based on this quantizer, a block is iteratively quantized to be 1-level, 2-level, or 4-level according to the locality of the image pixel block. In all the methods mentioned above, the size of a block is fixed.

In a conventional BTC algorithm, an input image is partitioned into 4 × 4 blocks, and each block is coded individually. For each pixel with gray value x_i in a certain block with i = 1, 2, . . . , m, there is an output value a or b computed by

$$a = \bar{x} - \bar{\sigma}\sqrt{\frac{q}{m-q}} \quad \text{for } x_i < \bar{x}; \tag{2}$$

$$b = \bar{x} + \bar{\sigma}\sqrt{\frac{m-q}{q}} \quad \text{for } x_i \geq \bar{x}; \tag{3}$$

where $\bar{x}$ is the sample mean and $\bar{\sigma}$ the standard deviation of the block, that is,

$$\bar{x} = \frac{1}{m}\sum_{i=1}^{m} x_i, \tag{4}$$

[1] This work was supported partially by the National Science Council, Republic of China, under Contract NSC80-0404-E009-13.

[2] To whom correspondence should be addressed.

1047-3203/93 $5.00

$$\bar{\sigma} = \sqrt{\frac{1}{m}\sum_{i=1}^{m} x_i^2 - \bar{x}^2}, \tag{5}$$

with m being the total number of pixels in the block, and q the number of pixels with gray values greater than or equal to $\bar{x}$.

Each block is coded by the values of a and b, and a 4×4 bit plane consisting of 1's and 0's indicating where the pixels are quantized to a or b. Assigning 8 bits to each of a and b results in a data rate of 2 bits/pixel. The image can be reconstructed at the receiver end block by block by assigning the values a and b to pixels in accordance with the codes in the bit plane.

As reported in [1], the BTC method tends to produce jagged edges in the reconstructed images. Due to insufficient quantization levels, aliasing effects will appear in the reconstructed images and sharp edges between contrasting pixels present jagged appearances. Areas containing sloping gray levels will turn into false contours because of the inherent quantization noise in the 1-bit (2-level) quantizer.

The previously mentioned modified versions of the BTC method [2–4] improve the resulting image in the sense of producing smaller mean square errors (MSE), but they cannot remove the jagged edges in the reconstructed image. Furthermore, the BTC algorithm and its improved methods suffer from a common deficiency, that is, the low compression ratio.

Performance of the BTC method can be improved substantially by adapting its performance according to image statistics. Essentially four types of adaptation can be made [5]:

1. adaptation of transform,
2. adaptation of bit allocation,
3. adaptation of quantizer levels, and
4. adaptation of block sizes.

The approach proposed in this paper is based on the BTC method, but with adaptation of the block size and the quantizer level according to spatial image statistics. Furthermore, the proposed algorithm can eliminate jagged edges in the reconstructed image and increase the compression ratio. As mentioned above, the jagged edges produced by the conventional BTC algorithm come from the insufficient quantization levels of the edges. Therefore, it is reasonable to introduce more quantization levels to code an edge block for better visual quality. On the contrary, if a block is smooth enough, it is not necessary to code it with two or more quantization levels. In addition, to increase the compression ratio, the size of a uniform block need not be fixed. To explore the local statistical characteristics of a block, a block decomposition procedure is used to divide an input image into blocks of variable sizes, with some of them being edge blocks and the other uniform ones. The result of an input image by performing the proposed decomposition procedure can be represented as a tree. The nodes at the lower levels of the tree correspond to larger blocks, and those at the higher levels correspond to smaller ones. In short, the proposed method essentially is a multilevel BTC scheme which combines the tree organization and the BTC algorithm to yield better compression ratios and produce less distortion.

The remainder of this paper is organized as follows. The proposed multilevel block truncation coding (MLBTC) method is described first in Section 2. Further adaptation of the coded image by the MLBTC method is included in Section 3. The method to reconstruct the coded image is discussed in Section 4. Section 5 includes the experimental results to support the validity of the proposed approach. Finally, some conclusions are drawn in Section 6.

2. PROPOSED MULTILEVEL BLOCK TRUNCATION CODING

The size of blocks used in the conventional BTC method is fixed. Two problems arise naturally: (1) it is difficult to choose a single optimal block size and (2) the different statistical characteristics of distinct blocks cannot be utilized. If the block size is too large for the BTC method, jagged appearances in the edges of the resulting images will be visible. On the contrary, if the block size is too small, the compression ratio will be low. The block size plays an important role in the performance of the BTC algorithm.

In this work variable block sizes are chosen to allow adaptation to local image characteristics. If an image has constant gray values everywhere, a single gray value is adequate to code the image. On the other hand, if the image contains more dispersion in frequency, more gray levels are needed. This observation leads to the following idea. Consider a block as an image itself. Depending on the detail or variation among the pixels, an appropriate number of gray levels should be used to code the block. What is wanted is to code each block with as few gray levels as possible to yield a sufficiently accurate estimate of that block, but with the resulting MSE being constrained to be smaller than a given threshold. Thus an adaptive algorithm is required. For this, a multilevel block truncation coding (MLBTC) algorithm is proposed to break each block, which does not satisfy a given uniformity criterion, into some equal-sized subblocks. The means of these subblocks are computed to constitute a new image, which is then requantized by a 2-level moment-preserving quantizer. Remember that the output of a 2-level moment-preserving quantizer contains two distinct components, namely, the output values a and b

(computed by (1) and (2)), as well as a bit plane consisting of 1's and 0's indicating where the pixels are quantized to be a or b. As a result, the new image will become a binary image with pixel values a and b. As a matter of convenience, the values of a and b will be called the two *quantized means* for the new image. And what has been done so far is essentially that the pixels of each subblock are estimated by its corresponding quantized mean value. If the dispersion of the original gray values of a subblock is found to be large, it is decomposed again. Otherwise, the subblock of the image is coded by its quantized mean. The process continues until each subblock of the input image satisfy the uniformity criterion or is of a preselected smallest size. The 2-level moment-preserving quantizer is finally applied to code all the nonuniform subblocks of the smallest size. Note that there is only one block at the beginning of the algorithm, which is just the input image. After the algorithm stops, the whole image is decomposed into many nonoverlapping blocks with variable sizes. Those of them tested to be uniform are coded by a single gray value, and the others with the preselected smallest size are coded in the same way as that of the BTC method. The information to code the input image by the method mentioned above can be organized as a tree. Figure 1 is given to illustrate the relationship between an intermediate step of the decomposition process and the tree. All the subblocks decomposed from a certain nonuniform block have the same parent node. The BTC method is applied in each decomposition step, and this is why the proposed method is called *multilevel BTC* algorithm. An advantage of the approach is that no a priori knowledge is needed to guide the decomposition process.

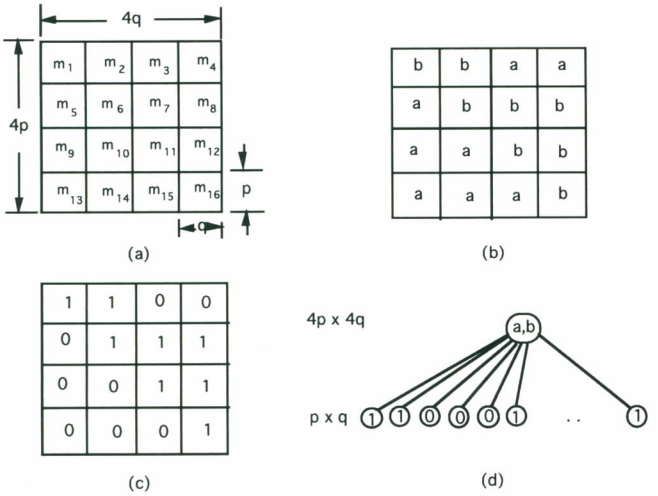


FIG. 1. An example of an intermediate step of the block decomposition process. (a) A $4p \times 4q$ nonuniform block is decomposed into 4×4 subblocks, and the mean (m_1 to m_{16}) of each subblock are computed to constitute a 4×4 new image; (b) 2-level moment-preserving quantization result of the new image; (c) binary representation of (b); (d) tree representation of (b).

Two questions about the above decomposition procedure should be answered. One is how to judge whether a tested block is uniform or not, and the other is why the pixels of a uniform block are estimated by its quantized mean rather than by its actual mean. A new criterion proposed to decide whether a block $\Re$, in which the pixel coordinates are normalized to be within the region of $\{x \in (0, 1), y \in (0, 1)\}$, is uniform is defined as

$$\sqrt{(\bar{x} - 0.5)^2 + (\bar{y} - 0.5)^2} \leq \tau, \tag{6}$$

where $(\bar{x}, \bar{y})$ are the coordinates of the center of gravity of the pixel gray values inside $\Re$. Note that if $\Re$ is uniform, the value of the left-hand side term of Eq. (6) will be very close to zero [10]. The selection of the uniformity threshold value τ is a trade-off between the compression ratio and the distortion degree in the reconstructed image. It is difficult to define an optimal value of τ to obtain the largest compression ratio without introducing false contours. According to our simulations, when the value of τ is set to be in the interval (0, 0.01), reasonable good results can be obtained.

The values of $(\bar{x}, \bar{y})$ of $\Re$ can be found from the values of the three mass moments M_0, M_x, and M_y of $\Re$. The relations are as follows:

$$\bar{x} = \frac{M_x}{M_0}, \tag{7}$$

$$\bar{y} = \frac{M_y}{M_0}. \tag{8}$$

The three mass moments of $\Re$ can be computed by

$$M_0 = \int \int_{\Re} f(x, y)dxdy, \tag{9}$$

$$M_x = \int \int_{\Re} xf(x, y)dydx, \tag{10}$$

$$M_y = \int \int_{\Re} yf(x, y)dxdy. \tag{11}$$

By assuming that $f(x, y)$ takes a constant value over each grid, the integral in (9), (10), or (11) becomes a weighted sum of the gray values in an $n \times m$ block as follows:

$$M_0 = \frac{1}{nm} \sum_x \sum_y f(x, y), \tag{12}$$

$$M_x = \frac{1}{2n^2m} \sum_x \sum_y xf(x, y), \tag{13}$$

$$M_y = \frac{1}{2nm^2} \sum_x \sum_y yf(x, y). \tag{14}$$

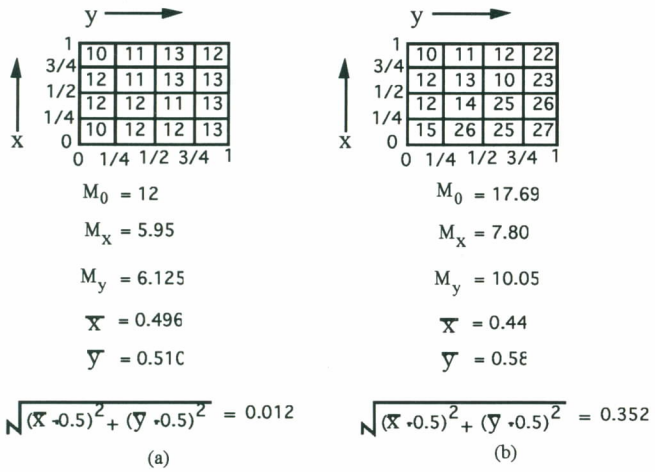


FIG. 2. Testing uniformity criterion on 4 × 4 blocks with threshold $\tau = 0.02$. (a) A uniform block with computed uniformity $< \tau$; (b) a nonuniform block with computed uniformity $> \tau$.

The effectiveness of the criterion described by (6) is illustrated by an example shown in Fig. 2.

To answer the second question, we can state at least two advantages of using the quantized mean instead of the actual mean: (1) the quantized mean can reduce more of the blocky effect and (2) the compression ratio can be increased because the value of a quantized mean can be retrieved from the information stored in its parent node in the decomposition tree. These two advantages are demonstrated by a simple example shown in Fig. 3. Figure 3a includes a noisy rectangle in a background. If the pixels of each uniform block is coded by its actual mean value, the resulting blocky effect will be obvious, as shown in Fig. 3d. However, this effect can be reduced by replacing the actual mean by its corresponding quantized mean, as shown in Fig. 3e. In addition, 128 bits are needed to code the simple image shown in Fig. 3a if all the uniform blocks are coded by the actual means (assuming 8 bits per mean); however, only 32 bits (for a, b, and a 16-bit bit plane) are needed to represent the original image if the actual means are replaced with the quantized means.

Given an $n \times n$ image with n as a power of 2, a concise description of the proposed MLBTC algorithm is given below. A more detailed adaptive algorithm is described in the next section.

MLBTC ALGORITHM.

1. Compute the mean of the input image as an initial approximation of the image.
2. Compute the values of M_0, M_x, and M_y of the image, and test the uniformity of the image according to the rule defined in by (6). If the uniformity criterion is satisfied, then label the image as a *good* block and stop decomposing the block; otherwise, label it as a *bad* block and continue.

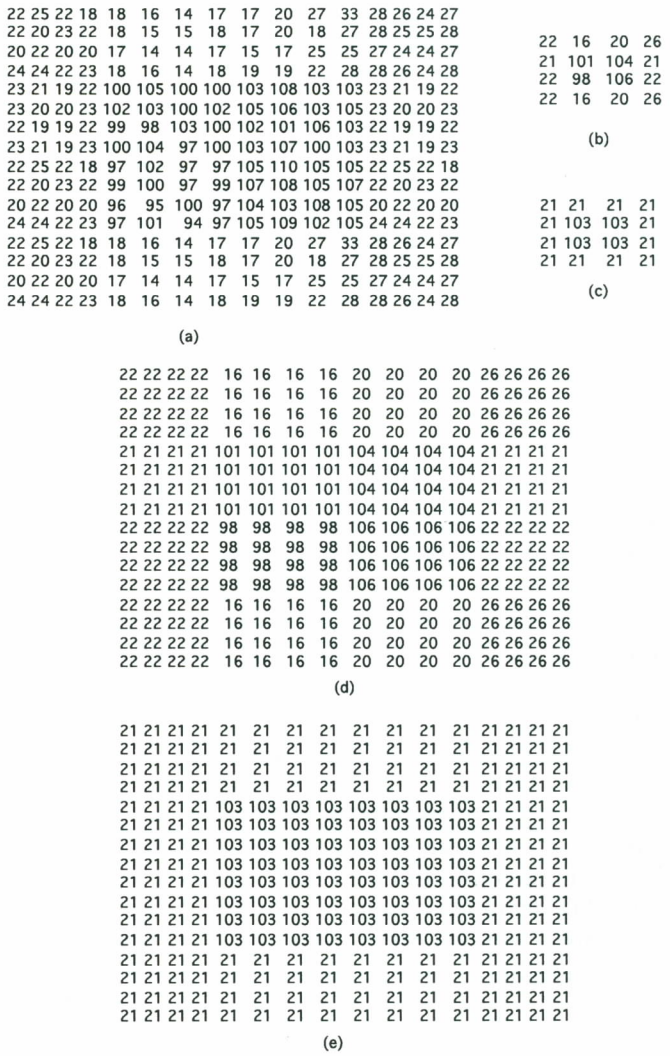


FIG. 3. An illustrative example showing the advantages of using the quantized mean over the actual mean. (a) Original image with a noisy rectangle in a background; (b) means for 4 × 4 decomposition of (a); (c) binary representation of (b) using the quantized means; (d) the final image by replacing pixel values of each subblock with its actual mean; (e) the result of replacing the actual mean by its corresponding quantized mean in each subblock.

3. Check the size of the *bad* block. If it is a block with a preselected smallest size, relabel it as a *good* block, apply the 2-level moment-preserving quantizer to the block, and stop decomposing it. Otherwise, continue.
4. Break the *bad* block into a set of equal-sized blocks.
5. Calculate the means of all the subblocks and quantize them by the 2-level quantizer. Use the quantized mean of each subblock as the approximation of the entire subblock.
6. Repeat Steps 2 through 5 for each of the subblocks.

A tree is built after the algorithm stops. The root of the tree represents the whole image, and the leaf nodes, which correspond to the *good* blocks output by the algorithm, break the pixels of the image into different nonoverlapping blocks. The nodes at the lower levels correspond to the larger block sizes. All leaf nodes not located at the highest level are uniform blocks and are coded by single gray values because they satisfy the uniformity criterion. This is the key point that the compression ratio can be increased by the proposed MLBTC method. It is also observed that the leaf nodes at the highest level of the tree are of the preselected smallest block size, and some of them, which are tested to be uniform are also coded by single gray values, but the others are coded by the 2-level moment-preserving quantizer. The larger the number of the smallest nonuniform blocks, the lower the compression ratio.

The jagged appearances in the edges of the coded image are still unsolved by the MLBTC method. Further improvement should be done to upgrade the resulting visual quality. For convenience, the tree built by the MLBTC algorithm is called the MLBTC tree.

3. ADAPTIVE MULTILEVEL BLOCK TRUNCATION CODING

In this section, we describe the method we proposed to improve the tree built by the MLBTC algorithm to remove the jagged appearances of the edges in the resulting image. As mentioned in Section 1, the jagged edges occur due to insufficient quantization levels. Based on this fact, a question arises: how to decide whether a block labeled *good* is quantized with insufficient quantization levels or not? If this question can be answered completely, the problem of producing jagged edges in the coded image can be removed. Unfortunately there lacks a good definition of a sufficiently quantized block. A possible judgment is: if a block is quantized sufficiently, the MSE between the original gray values and the quantized ones of the block might be small. However, a threshold value is introduced here and the problem is changed to how to determine the threshold value automatically. Note that a solution to this new problem should not be time-consuming in order not to destroy the advantages of the standard BTC method.

A proposed solution is to define the threshold value as the median of all the MSE values of the blocks which are labeled *good* in the MLBTC tree. One reason to choose the median is that it is easy to compute. Another reason is that it is effective, as shown by our experimental results. More specifically, if the MSE of a *good* block B is larger than this threshold value, B is judged to be quantized insufficiently, and a further process is applied to B. Assume that the highest level of the MLBTC tree is h. Two cases should be considered.

(1) If the level of B in the MLBTC tree is less than h, that is, if the size of B is larger than the preselected smallest value, then B is relabeled as a *bad* block, and is decomposed and requantized according to the rules mentioned in the MLBTC algorithm. B will then become a nonleaf node in the resulting MLBTC tree.

(2) If the level of B is h, that is, if B is of the preselected smallest size, then it is requantized with double of its originally quantized levels to improve the visual quality of the resulting image.

That is, if a block is originally approximated by a single gray value, it is requantized by a 2-level quantizer, and if the block is originally quantized by a 2-level quantizer, it is requantized by a 4-level quantizer, etc. A simple example to illustrate the aforementioned adaptation process is given in Fig. 4.

This adaptation process can be performed more than once to decrease the MSE value of the resulting image to a desired value. However, it should be noted that the compression ratio will also be decreased when the adaptation process is performed to improve the visual quality of the resulting image. A trade-off should be taken between the resulting estimation accuracy and the compression ratio. In this study the quantization levels for each block are limited to be at most 4 for the reason of yielding higher compression effect.

There exists another unsolved problem in the above algorithm, namely, how to quantize the gray values of a block into 4 gray levels? Some algorithms [8, 9] proposed

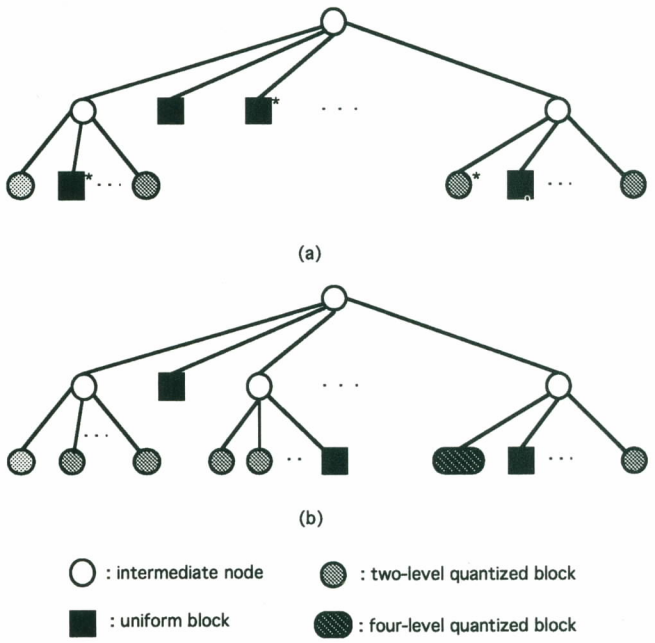


FIG. 4. Adaptation of the MLBTC tree. (a) MLBTC tree before adaptation; (b) MLBTC tree after adaptation. Those nodes marked by * are judged to be insufficiently quantized and are requantized.

for multilevel thresholding might be used to achieve this goal. Unfortunately, the computation time of these methods are long and hence are unsuitable here. A method proposed in this study is as follows.

FOUR-LEVEL QUANTIZATION ALGORITHM.

1. Find the mean m and deviation d of the block.
2. Compute the values of a and b from m and d according to Eqs. (2) and (3). Consider the values of a, b, and m to divide the gray value distribution of the block into four intervals ([0, a], [a, m], [m, b], and [b, 255]) as shown in Fig. 5.
3. Requantize interval [0, a] to be a, [a, m] to be $(a + m)/2$, [m, b] to be $(b + m)/2$, and [b, 255] to be b.

The idea behind the proposed 4-level quantizer is to introduce more quantization levels into the transition region of an edge to eliminate more of the jagged edges in the resulting image. An advantage of the above 4-level quantizer is its low computation effort; only the original mean and deviation needed to be recorded to reconstruct the four quantized gray levels. The experimental results described in the next section show that the jagged edges in the final coded image can be eliminated effectively. The proposed image compression algorithm, called adaptive multilevel block truncation coding (AMLBTC), can now be fully described as follows.

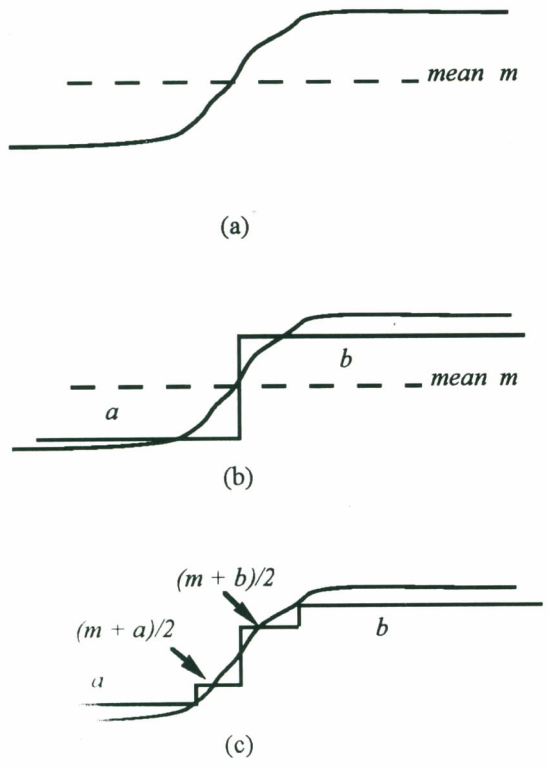


FIG. 5. Illustration of proposed 4-level quantizer. (a) Original noisy edge; (b) 2-level quantization; (c) 4-level quantization.

AMLBTC ALGORITHM.

1. Perform the MLBTC algorithm with an MLBTC tree T as the output.
2. Compute the median θ of all the MSE values of the *good* blocks in T.
3. Compare the MSE e of each *good* block B with θ. If $e > \theta$, then two cases are considered as follows. Assume that the highest level of the T nodes is h and that the level of B is h'.
   (1) If $h' < h$, then relabel B as *bad* and decompose it according to the rules mentioned in the MLBTC algorithm. A subtree is built with B as the root node T. Merge it with the original MLBTC tree.
   (2) If $h' = h$, then requantize B with twice of its original quantized levels using the four-level quantization algorithm.
4. Repeat Steps 2 and 3 until the overall MSE value of the input image is less than a predefined threshold, or until the lowest preselected compression ratio has been achieved.

The lower bound of the overall MSE value for the proposed AMLBTC algorithm is achieved when all the *good* blocks are of the smallest size and are 4-level quantized. Hence, the predefined threshold for the overall MSE mentioned in Step 4 should be chosen to be no less than this bound to stop the AMLBTC algorithm. The selection of the proper predefined threshold value depends on the maximum allowable distortion.

The advantages of the proposed AMLBTC algorithm over the standard BTC method include: (1) the distortion is reduced dramatically and (2) the compression ratio is improved greatly. Furthermore, the proposed AMLBTC method is simple, and the computation load is almost the same as that of the standard BTC method.

4. MLBTC TREE CODING AND IMAGE RECONSTRUCTION

In this section, we describe the method we proposed to code the information stored in an MLBTC tree, which can be followed by the receiver end to reconstruct the coded image. In this study, an MLBTC tree is coded in two stages. In the first stage, the nodes of the tree are visited and coded according to their types by depth-first tree traversal. The purpose of performing this stage is to code the organization of the tree. The resulting code stream can address the nodes properly in the tree. In the second stage, according to the same tree traversal sequence as Stage 1, the information stored in the nodes of the MLBTC tree is sent. The overall code stream for an MLBTC tree so consists of two parts, namely, the tree organization and the information stored in the nodes.

More specifically, there are four types of nodes, which

contain necessary information for reconstruction, in an MLBTC tree as described in the following.

1. Intermediate node: the information stored in a node of this type is the quantized means (a and b) of its child nodes.
2. Uniform block: only one bit (0 or 1) is stored in each node of this type, which can be used to retrieve the corresponding quantized mean value (a or b) from its parent node.
3. Two-level quantized block: three components are stored in each node in this type, namely, one bit which works like that stored in the type-2 node, the standard deviation of the block, and a bit plane which works like that used in the BTC method. The rule to reconstruct this kind of node is the same as the BTC algorithm, except that the actual mean now is replaced by the quantized one.
4. Four-level quantized block: the information stored in each node of this type is the same as that stored in the type-3 node, except that the number of bits for a bit plane is two times that of the type-3 node.

For convenience, the total number of bits to code both the tree organization and the information stored in all of the intermediate nodes in the MLBTC tree is called the overhead of the tree. Figure 6 is given to illustrate the coding method. The number of branches of each node in the tree as shown in Fig. 6a is 4. The tree organization is coded first. The four types of nodes in the tree, namely, intermediate node, uniform block, 2-level quantized block, and 4-level quantized block are coded as 0, 1, 01, and 00, respectively. Then the tree organization for the tree in Fig. 6 is coded as follows:

$$0_{n1},\ 0_{n2},\ 1_{n3},\ 1_{n4},\ 01_{n5},\ 01_{n6},\ 1_{n7},\ 0_{n8},\ 1_{n9},\ 01_{n10},\ 00_{n11},\ 1_{n12},\ 1_{n13}.$$

The subscripts are included just for the purpose of indicating the nodes represented by the codes. Because the number of branches of each node in an MLBTC tree is preselected, the highest level of the tree can be known in advance for a given-sized input image. Based on this fact, the tree to be reconstructed from its code stream can be designed to be unique. Let h be the highest level of an MLBTC tree T. If the present scanned code is b and the level of T to be explored is l, then the rules to reconstruct the tree T from its tree-organization code C are as follows.

1. If $b = 0$, then two cases are considered.
   Case 1. If $l < h$, an intermediate node is generated and the value of l is increased by one. The next node to be generated will be the leftmost child of the present node (depth-first tree traversal sequence).
   Case 2. If $l = h$, the next symbol of C is scanned to judge which type (2-level quantized block or 4-level quantized one) the present node to be generated is. If the present node is not the rightmost child of its parent node, the right sibling of the present node will be the one to be generated. The value of l is not changed. However, if the present node is the rightmost child of its parent node, then the node control of generating next node is returned to its parent node. The value of l is decreased by one.
2. If $b = 1$, a uniform block is generated. The same discussion as that in the Case 2 of Rule 1 can be made to decide the next node to be generated.

Figure 6b is an example to demonstrate the reconstruction procedure.

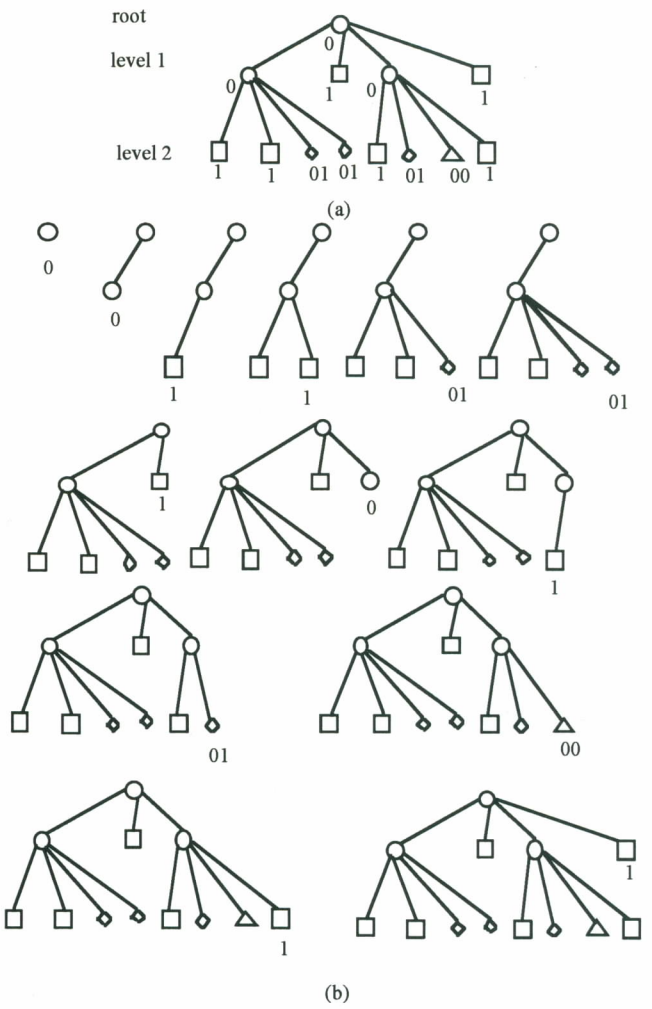


FIG. 6. An example of MLBTC tree coding. (a) An MLBTC tree with tree organization coded as 00110101101010011.; (b) reconstruction sequence of the MLBTC tree from the code stream of (a). The notations ○, □, ◇, △ represent intermediate nodes, uniform blocks, 2-level quantized blocks, and 4-level quantized blocks, respectively.

Following the code stream for the tree organization, the information stored in each node is then sent. Let the information stored in the four types of nodes mentioned above be denoted as ○, □, ◇, and △, respectively. Then this part of information is coded as the following string:

$$\bigcirc_{n1}, \bigcirc_{n2}, \square_{n3}, \square_{n4}, \diamond_{n5}, \diamond_{n6}, \square_{n7}, \bigcirc_{n8}, \square_{n9}, \diamond_{n10}, \triangle_{n11}, \square_{n12}, \square_{n13}.$$

5. EXPERIMENTAL RESULTS

To check the feasibility of the overall algorithm proposed in Section 3, a series of experiments were conducted on many images of size 512 × 480. All the images were digitized to be 8 bits per pixel. Several of them were processed to compare the performance of the BTC, AMBTC, and the proposed AMLBTC methods. For the standard BTC and the AMBTC methods, the images were scanned and divided into nonoverlapping 4 × 5 blocks with each block coded individually. By allowing 6

TABLE 1
Bit Rates Spent for Blocks of Variable Sizes

Block size	Type	Number of pixels	Bits consumption	Bit rate (bits/pixel)
256 × 240	Uniform	61440	1	0.000016
64 × 60	Uniform	3840	1	0.00026
16 × 15	Uniform	240	1	0.004
	Uniform		1	0.05
4 × 5	2 levels	20	26	1.3
	4 levels		46	2.3

FIG. 7. Example of reconstructed images for three different methods (I). (a) Original image GIRL with 8 bits/pixel; (b) image reconstructed image by BTC with *MSE* 8.70 and 1.6 bits/pixel; (c) image reconstructed by AMBTC with *MSE* 8.07 and 1.6 bits/pixel; (d) image reconstructed by AMLBTC with *MSE* 3.07 and 1.6 bits/pixel. The *MSE* value of the proposed AMLBTC is much lower than those of the other two methods under the same compression ratio.

bits to code each of the mean and standard deviation of each block, the bit rate for both methods is 1.6 bits/pixel. The block sizes for the higher levels of the MLBTC tree built by the proposed AMLBTC method are chosen to be 256×240, 64×60, 16×15, and 4×5. The bits needed to code the blocks at each level of the tree are shown in Table 1. Accordingly, the larger the number of the non-uniform blocks with the smallest preselected size, the lower the compression ratio. In order to facilitate performance comparison, the final MLBTC tree is properly adjusted (by tuning the value of the uniformity threshold τ) such that the resulting bit rate of the proposed AMLBTC method is approximately 1.6 bits/pixel. Figures 7 through 12 show some of the reconstructed images for the three methods of the same bit rate. The MSE values of the reconstructed images were computed and listed in Table 2. It can be seen that by using the AMLBTC method, there is an average drop of 73% (or 68%) in the mean square errors of the reconstructed images, compared with the standard BTC (or AMBTC) method. Note that the jagged edges of the reconstructed images are visible in the results of both the standard BTC and the AMBTC methods, but are almost completely eliminated in those of the AMLBTC method.

In terms of the compression ratio, the AMLBTC method is also superior to the standard BTC and the AMBTC methods. If no adaptation of bit allocation is given to the sample means and deviations, the bit rate for the standard BTC or AMBTC method is limited to be 2 bits per pixel. This disadvantage is improved by the

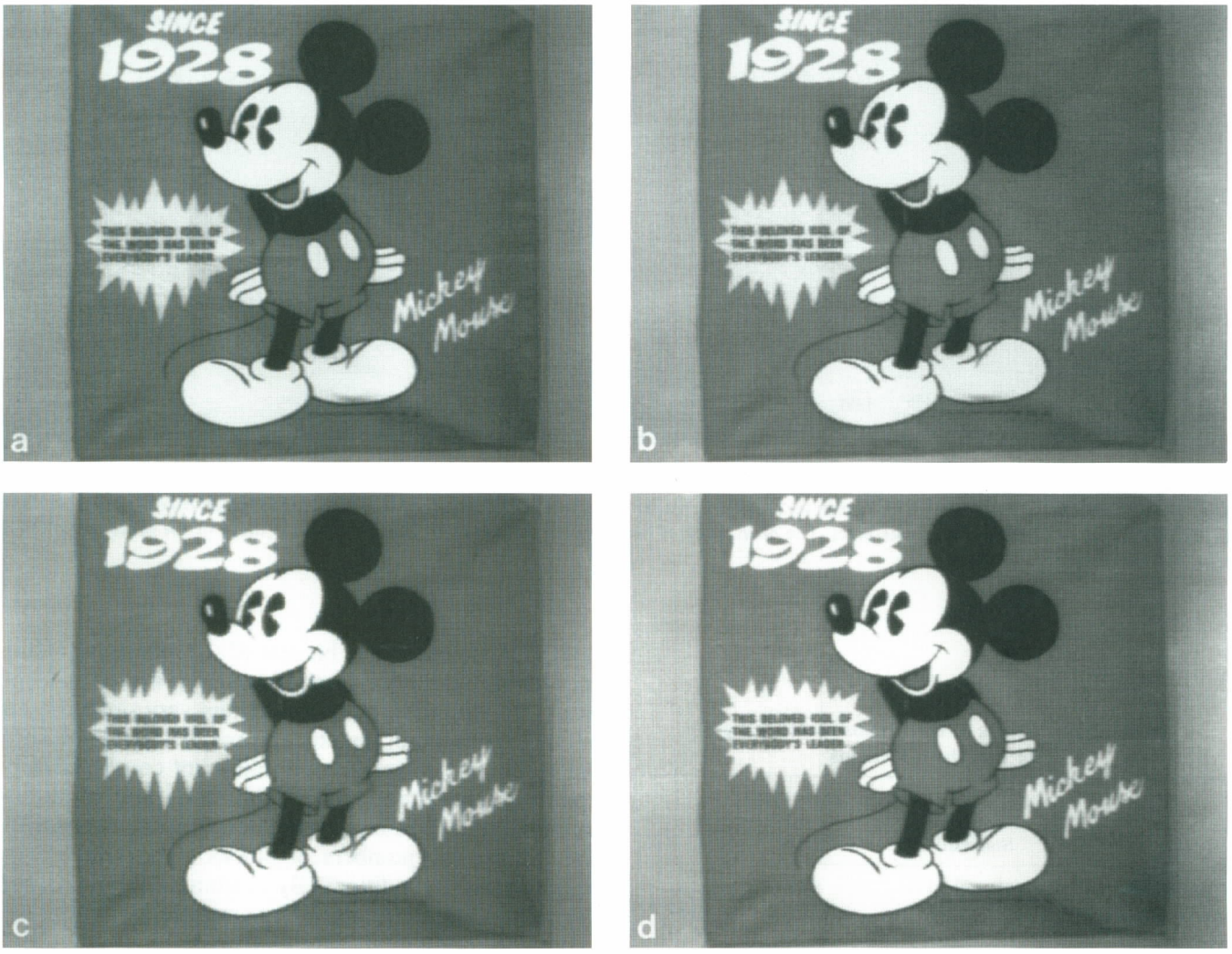


FIG. 8. Example of reconstructed images for three different methods (II). (a) Original image Mickey Mouse with 8 bits/pixel; (b) image reconstructed image by BTC with *MSE* 25.43 and 1.6 bits/pixel; (c) image reconstructed by AMBTC with *MSE* 23.72 and 1.6 bits/pixel; (d) image reconstructed by AMLBTC with *MSE* 6.96 and 1.6 bits/pixel. The *MSE* value of the proposed AMLBTC is much lower than those of the other two methods under the same compression ratio. Reproduced by permission of the publisher. © The Walt Disney Company.

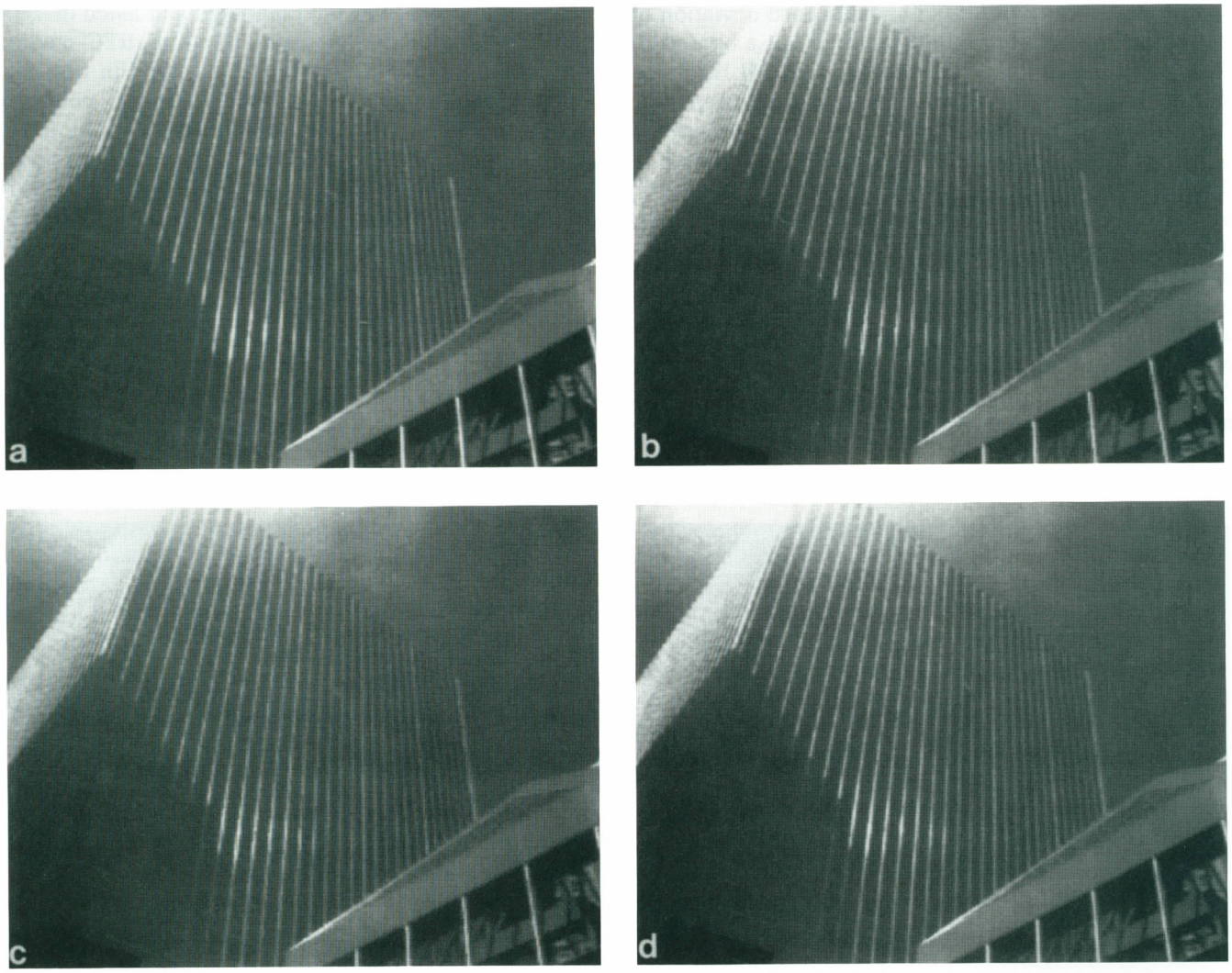


FIG. 9. Example of reconstructed images for three different methods (III). (a) Original image building with 8 bits/pixel; (b) image reconstructed image by BTC with *MSE* 31.94 and 1.6 bits/pixel; (c) image reconstructed by AMBTC with *MSE* 22.36 and 1.6 bits/pixel; (d) image reconstructed by AMLBTC with *MSE* 6.21 and 1.6 bits/pixel. The *MSE* value of the proposed AMLBTC is much lower than those of the other two methods under the same compression ratio.

TABLE 2
Mean Square Errors of the Reconstructed Images for the Three Methods

Image	Mean square error		
	Standard BTC	AMBTC	AMLBTC
Girl	8.70	8.09	2.97
Mickey Mouse	25.43	23.72	6.47
Building	31.94	22.36	6.21
Monkey	24.45	16.24	8.26
Brain	11.14	10.37	3.55
Chess	10.86	10.12	3.75
Average	18.75	16.24	5.20

AMLBTC method. The bit rate of the reconstructed images for the AMLBTC method can be as low as 0.50 bits/pixel while the MSE value is still kept very small, as seen in our experimental results. Figures 13 through 18 show the reconstructed images with different compression ratios. The computed statistics for these images are also listed in Table 3. The items below columns 3 through 7 in Table 3 are the numbers of blocks with variable sizes under different conditions. By combining Tables 1 and 3, the compression ratios can be computed as

$$\textit{compression ratio} = \frac{512 \times 480 \times 8}{\textit{overhead} + \Sigma_{\textit{block size}} N_{\textit{block size}} \times B_{\textit{block size}}}, \quad (15)$$

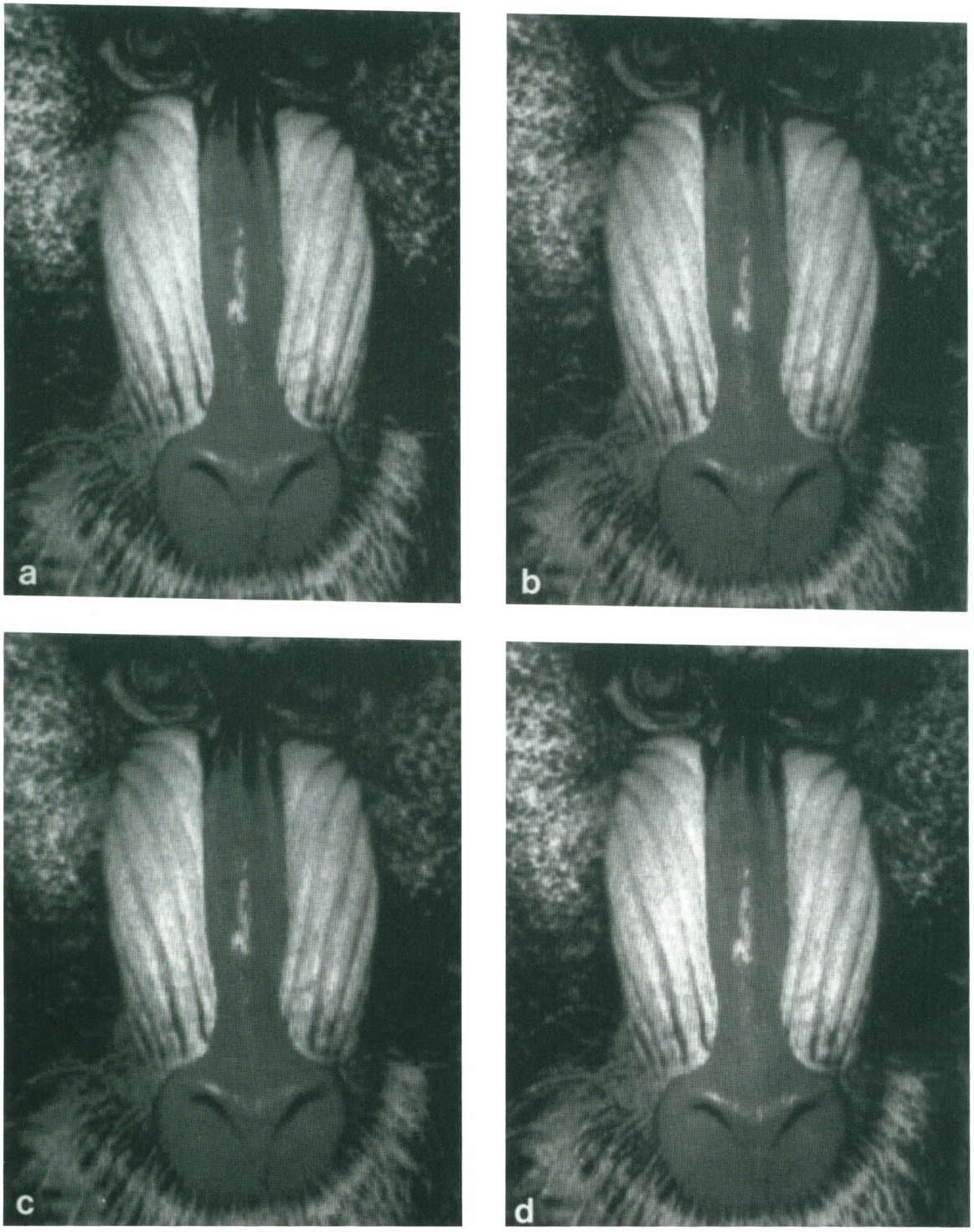


FIG. 10. Example of reconstructed images for three different methods (IV). (a) Original image Monkey with 8 bits/pixel; (b) image reconstructed image by BTC with *MSE* 22.45 and 1.6 bits/pixel; (c) image reconstructed by AMBTC with *MSE* 22.76 and 1.6 bits/pixel; (d) image reconstructed by AMLBTC with *MSE* 8.26 and 1.6 bits/pixel. The *MSE* value of the proposed AMLBTC is much lower than those of the other two methods under the same compression ratio.

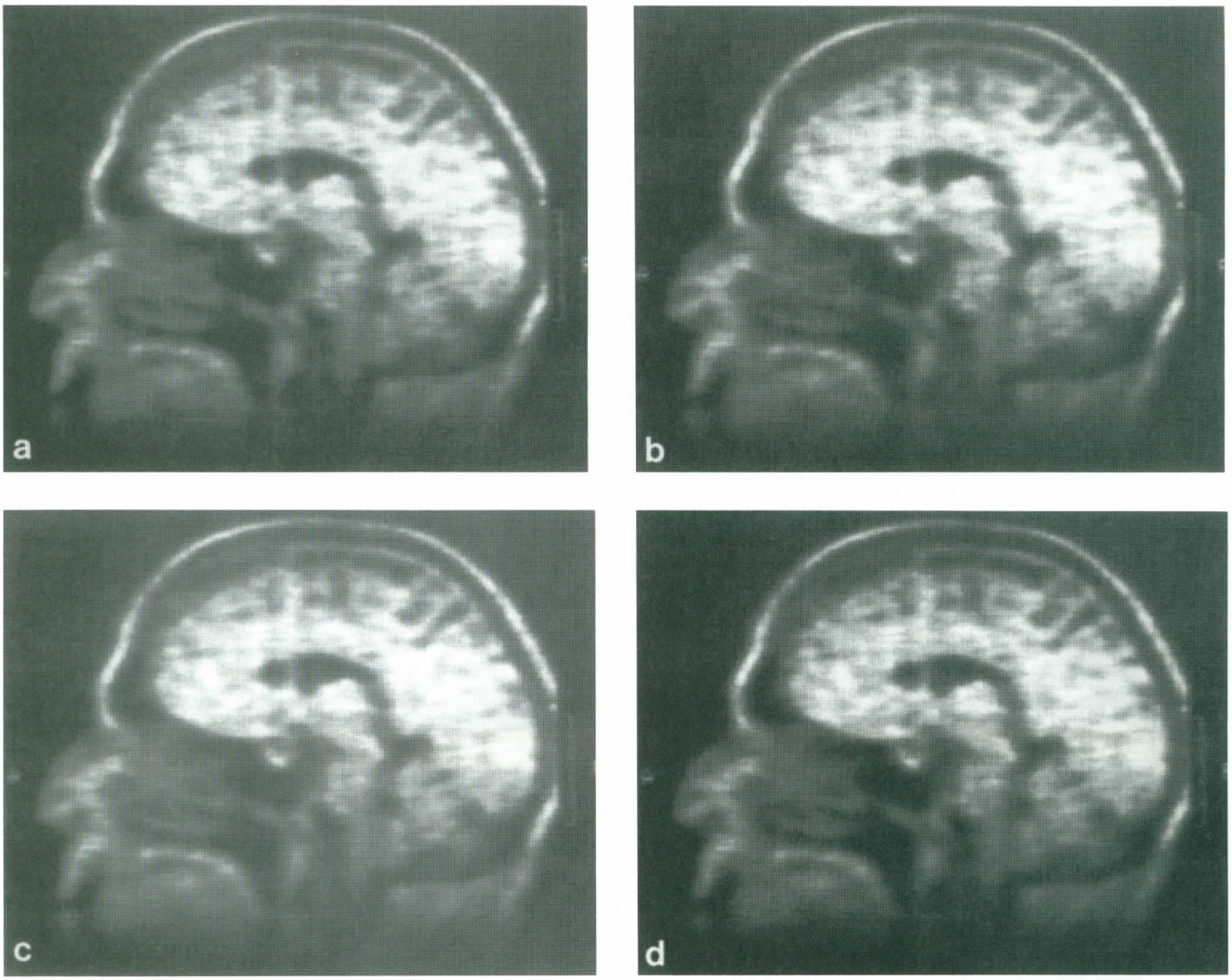


FIG. 11. Example of reconstructed images for three different methods (V). (a) Original image Brain with 8 bits/pixel; (b) image reconstructed image by BTC with *MSE* 11.14 and 1.6 bits/pixel; (c) image reconstructed by AMBTC with *MSE* 10.37 and 1.6 bits/pixel; (d) image reconstructed by AMLBTC with *MSE* 3.55 and 1.6 bits/pixel. The *MSE* value of the proposed AMLBTC is much lower than those of the other two methods under the same compression ratio.

TABLE 3
Computed Statistics of the Three Input Images

Image	τ	Number of blocks					Overhead	*MSE*	Comp. ratio
		64*60	16*15	♣4*5	♡4*5	♠4*5			
Girl	0.008	0	189	7284	1795	943	13846	6.65	16.17
	0.004	0	104	5793	3812	1435	17375	4.30	9.854
Mickey Mouse	0.008	6	436	3190	1350	1364	9610	14.71	16.74
	0.004	0	229	4885	2970	1685	15283	8.81	10.63
Building	0.008	4	307	3265	3117	1459	13431	15.47	11.35
	0.004	0	188	3336	4584	2112	17816	9.40	7.93
Monkey	0.008	0	160	4565	5197	606	17259	19.39	10.06
	0.004	0	59	2323	8137	1120	21925	12.84	6.57
Brain	0.008	0	311	5398	2337	821	12802	9.80	15.62
	0.004	0	192	3177	5276	1531	17879	5.30	8.23
Chest	0.008	0	349	5231	1952	917	12057	11.60	16.52
	0.004	0	168	3297	5137	1838	18335	5.60	7.85

Note. The notations ♣, ♡, and ♠ represent uniform, 2-level quantized, and 4-level quantized smallest-sized blocks, respectively.

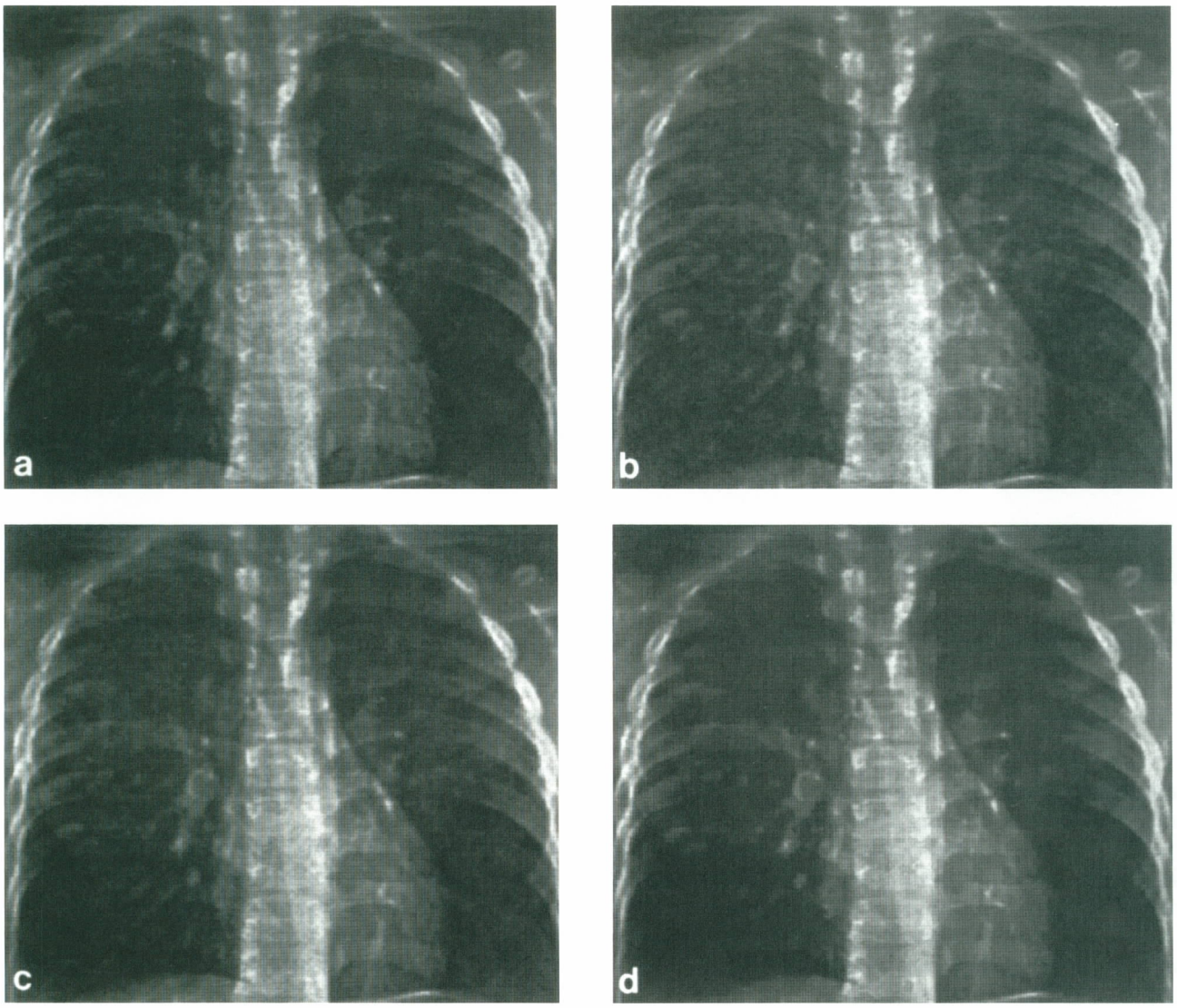


FIG. 12. Example of reconstructed images for three different methods (VI). (a) Original image Chest with 8 bits/pixel; (b) image reconstructed image by BTC with *MSE* 10.86 and 1.6 bits/pixel; (c) image reconstructed by AMBTC with *MSE* 10.12 and 1.6 bits/pixel; (d) image reconstructed by AMLBTC with *MSE* 3.75 and 1.6 bits/pixel. The *MSE* value of the proposed AMLBTC is much lower than those of the other two methods under the same compression ratio.

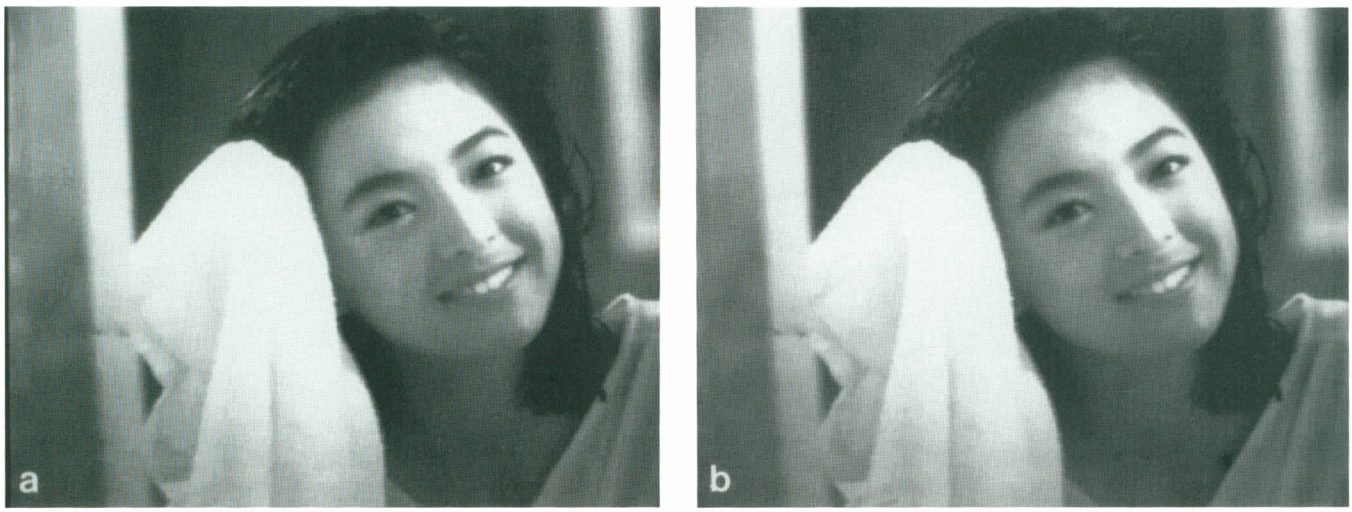


FIG. 13. Proposed AMLBTC for image Girl with different compression effects. (a) Image reconstructed with *MSE* 6.65 and 0.495 bits/pixel; (b) image reconstructed with *MSE* 4.30 and 0.811 bits/pixel.

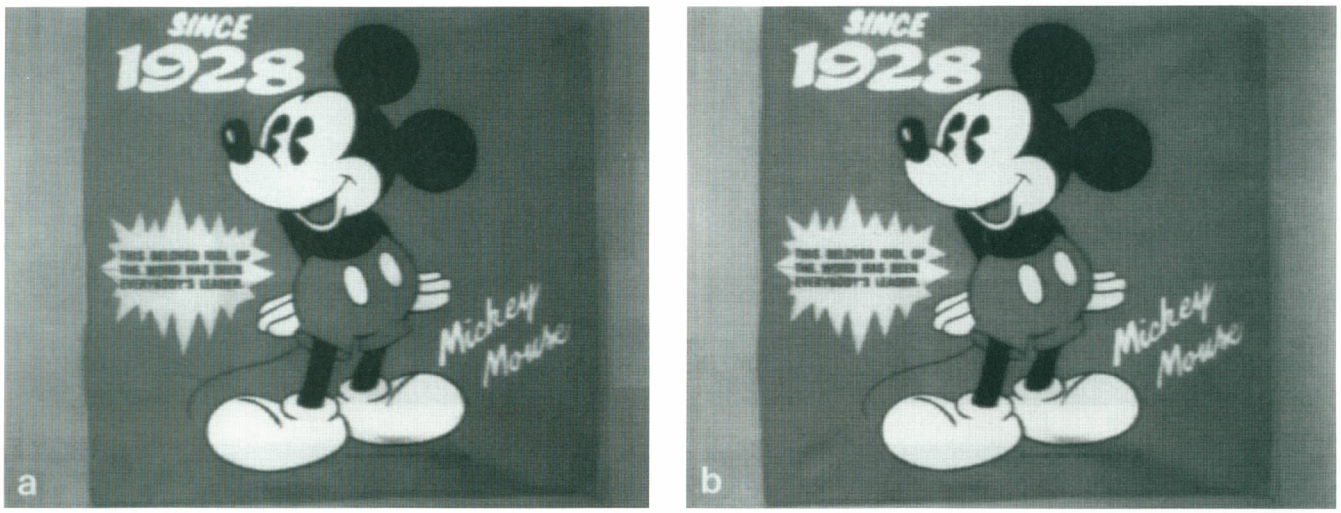


FIG. 14. Proposed AMLBTC for image Mickey Mouse with different compression effects. (a) Image reconstructed with *MSE* 14.71 and 0.477 bits/pixel; (b) image reconstructed with *MSE* 8.81 and 0.752 bits/pixel. Reproduced by permission of the publisher. © The Walt Disney Company.

where the notation $N_{block\ size}$ represents the number of blocks with the size of *block size*, which can be found in Table 3, and $B_{block\ size}$ is the number of bits needed to code a block with the size of *block size*, which can be found in Table 1. The possible values of *block size* are 64 × 60, 16 × 15, and 4 × 5 as shown in Table 3. For higher compression ratios, it can be seen that the visual quality of the reconstructed images are still acceptable.

6. CONCLUSION

A new method for image compression, called adaptive multilevel block truncation coding (AMLBTC) has been proposed, and the performance has been shown to be better than those of the standard BTC and AMBTC methods. The proposed method produces results with less distortion and higher compression ratios. The AMLBTC re-

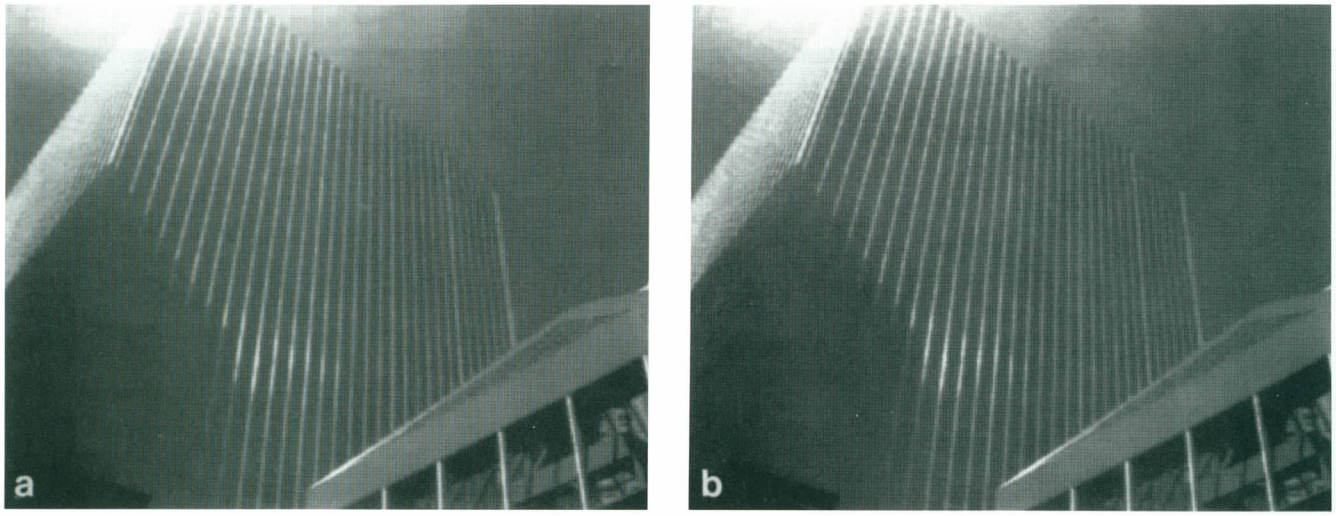


FIG. 15. Proposed AMLBTC for image Building with different compression effects. (a) Image reconstructed image with *MSE* 15.47 and 0.704 bits/pixel; (b) image reconstructed with *MSE* 9.40 and 1.008 bits/pixel.

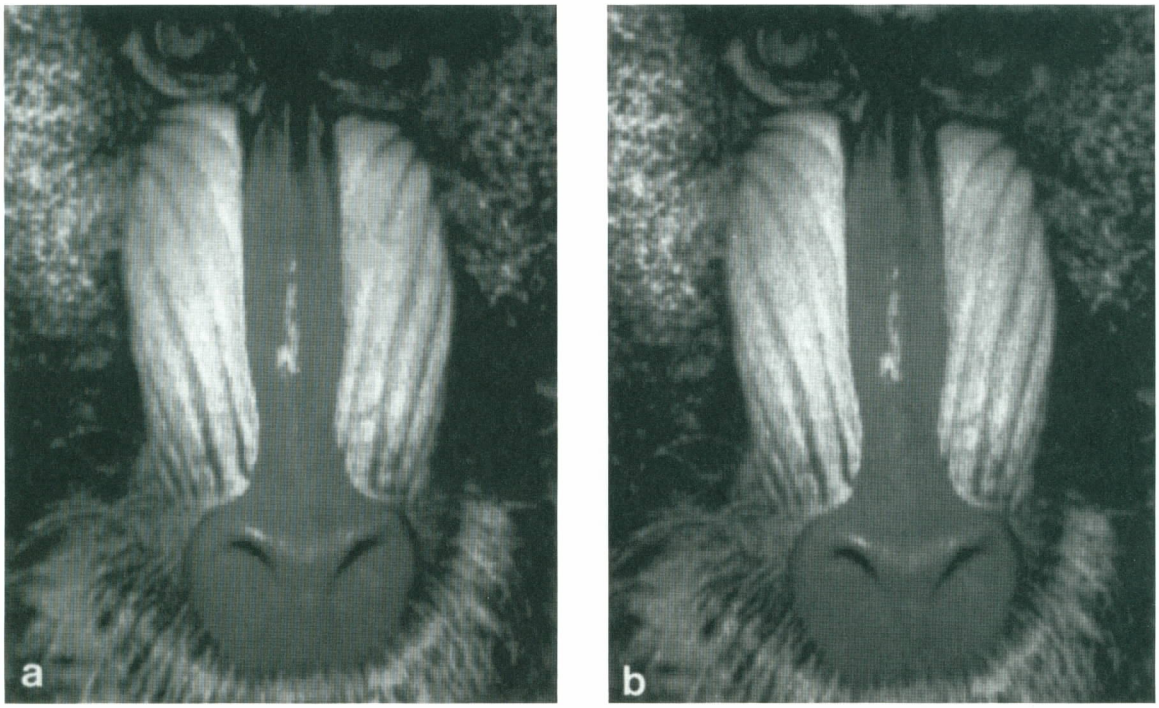


FIG. 16. Proposed AMLBTC for image Monkey with different compression effects. (a) Image reconstructed with *MSE* 19.39 and 0.795 bits/pixel; (b) image reconstructed with *MSE* 12.84 and 1.217 bits/pixel.

mains relatively simple in computation and can be practically implemented by computer software. Also, the proposed method has the same general characteristics as those of the BTC, including low storage requirement and extremely simple coding and decoding steps. Only gray-scale images are discussed; however, the coding scheme is equally applicable to color images after minor modification.

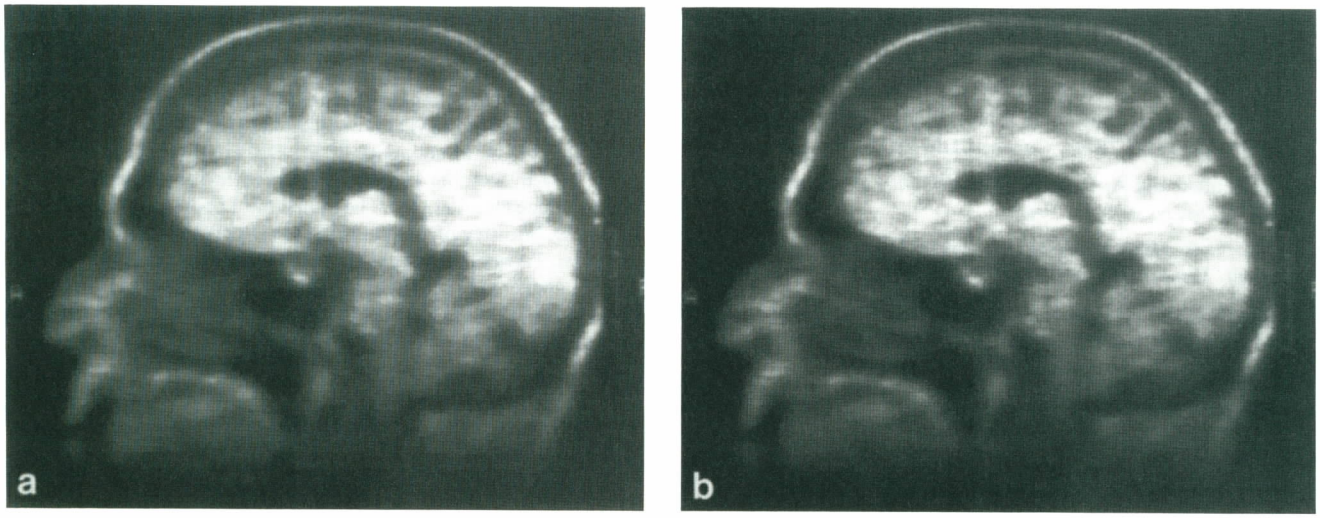


FIG. 17. Proposed AMLBTC for image Brain with different compression effects. (a) Image reconstructed with *MSE* 9.80 and 0.512 bits/pixel; (b) image reconstructed with *MSE* 5.30 and 0.972 bits/pixel.

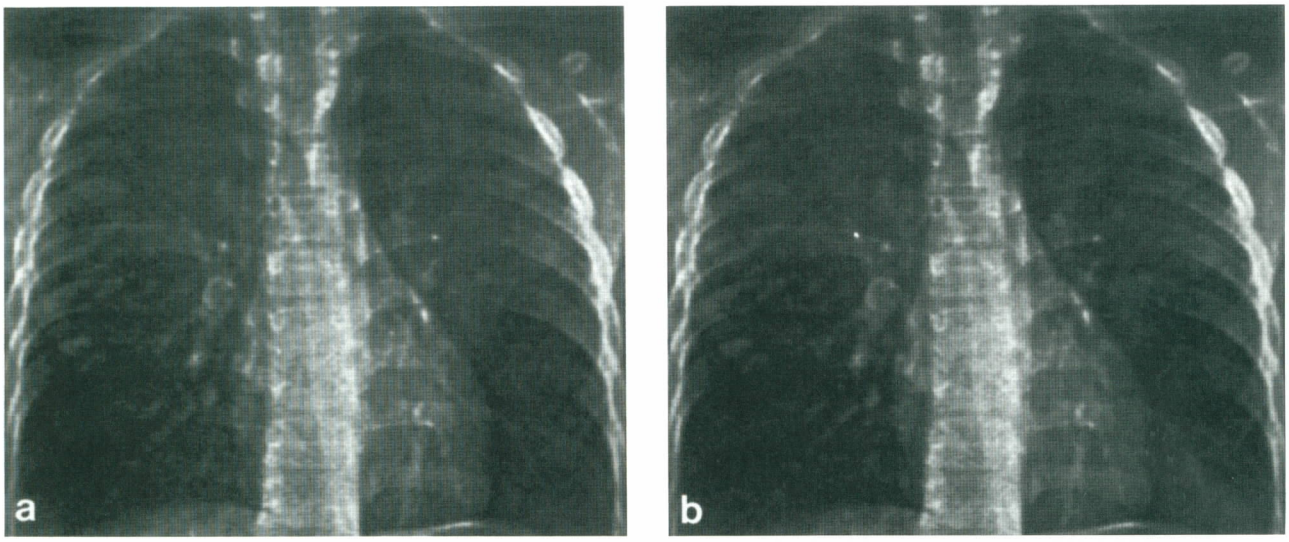
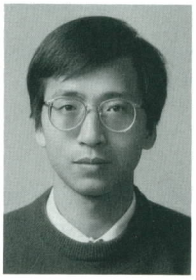


FIG. 18. Proposed AMLBTC for image Chest with different compression effects. (a) Image reconstructed image with *MSE* 11.60 and 0.484 bits/pixel; (b) image reconstructed with *MSE* 5.60 and 1.019 bits/pixel.

REFERENCES

1. E. J. Delp and O. R. Mitchell, Image compression using block truncation coding, *IEEE Trans. Comm.* **COM-27**(9), Sept. 1979, 1335–1342.
2. D. R. Halverson, N. C. Griswold, and G. L. Wise, A generalized block truncation coding algorithm for image compression, *IEEE Trans. Acoust. Speech Signal Process.* **ASSP-32**(3), June 1984, 664–668.
3. M. D. Lema and O. R. Mitchell, Absolute moment block truncation coding and its application of color images, *IEEE Trans. Comm.* **COM-32**(10), Oct. 1984, 1148–1157.
4. L. Hui, An adaptive block truncation coding algorithm for image compression, *IEEE International Conference on ASSP, New Mexico, April, 1990,* pp. 2233–2236.
5. P. Wintz, Transform picture coding, *Proc. IEEE* **60**(7), July 1972.
6. N. S. Jayant, and P. Noll, *Digital Coding of Waveforms*, Prentice–Hall, Englewood Cliffs, New Jersey, 1985.
7. Y. Linde, A. Buzo, and R. M. Gray, An algorithm for vector quantizer design, *IEEE Trans. Comm.* **COM-28**(1), Jan. 1980, 84–95.
8. A. Rosenfeld and A. C. Kak, *Digital Picture Processing, II*, Academic Press, New York, 1982.
9. W. H. Tsai, Moment-preserving thresholding: A new approach, *Comput. Vision Graphics Image Process.* **29,** 1985, 377–393.
10. H. Flanders and J. J. Price, *Calculus with Analytic Geometry*, Academic Press, New York, 1978.



SHYI-CHYI CHENG was born in Changhua, Taiwan, Republic of China, on September 28, 1963. He received the B.S. degree in electrical engineering from National Tsing Hua University, Hsinchu, Taiwan, Republic of China, in 1986, and the M.S. degree in electronics and the Ph.D degree in computer science and information engineering, both from National Chiao Tung University, Hsinchu, Taiwan, in 1988 and 1992, respectively. During his study at National Chiao Tung University, he worked as a research assistant in the Computer Vision Laboratory of the Department of Computer and Information Science. He was a winner of the 1988 AceR Long-Term outstanding M. S. Thesis Scholarship and the 1992 Xerox Foundation Ph.D. Dissertation Study Scholarship. From July to October in 1992, he served in the Chinese Army as an officer. Since October, 1992, he joined the Laboratories of Telecommunications, Ministry of Communications, Republic of China, at Chungli, Taoyuan, Taiwan, as a research member. His current research interests include image coding, digital signal processing, data compression, and communications. Mr. Cheng is a member of the Chinese Image Processing and Pattern Recognition Society.

WEN-HSIANG TSAI was born in Tainan, Taiwan, Republic of China, on May 10, 1951. He received the B.S. degree from National Taiwan University in 1973, the M.S. degree from Brown University in 1977, and his Ph.D. degree from Purdue University in 1979, all in electrical engineering. Since November 1979, he has been on the faculty of the Institute of Computer Science and Information Engineering at National Chiao Tung University, Hsinchu, Taiwan. From 1984 to 1986, he was an assistant director and later an associate director of the Microelectronics and Information Science and Technology Research Center at National Chiao Tung University. He joined the Department of Computer and Information Science at National Chiao Tung University in August 1984, acted as the Head of the Department from 1984 to 1988, and is currently a professor there. He serves as a consultant to several research institutes and industrial companies. His current research interests include computer vision, image processing, pattern recognition, and neural networks. Dr. Tsai is an Associate Editor of *Pattern Recognition*, the *International Journal of Pattern Recognition and Artificial Intelligence*, the *Journal of the Chinese Institute of Engineers*, and the *Journal of Information Science and Engineering*. He was elected as an Outstanding Talent of Information Science of the Republic of China in 1986. He was the winner of the 13th Annual Best Paper Award of the Pattern Recognition Society of U.S.A. He obtained the 1987 Outstanding Research Award and the 1988–1989, 1990–1991, and 1992–1993 Distinguished Research Awards of the National Science Council of the Republic of China. He was the recipient of the 1989 Distinguished Teaching Award of the Ministry of Education of the Republic of China. He was also the winner of the 1990 Outstanding Paper Award of the Computer Society of the Republic of China. Dr. Tsai has published more than 65 papers in well-known international journals. Dr. Tsai is a senior member of the IEEE and a member of the Chinese Image Processing and Pattern Recognition Society, the Computing Linguistics Society of the Republic of China, and the Medical Engineering Society of the Republic of China.