Research Article

Lossless Image Compression Based on Multiple-Tables Arithmetic Coding

Rung-Ching Chen,[1] Pei-Yan Pai,[2] Yung-Kuan Chan,[3] and Chin-Chen Chang[2, 4]

[1] *Department of Information Management, Chaoyang University of Technology, No. 168, Jifong E. Rd., Wufong Township Taichung County 41349, Taiwan*

[2] *Department of Computer Science, National Tsing-Hua University, No. 101, Section 2, Kuang-Fu Road, Hsinchu 300, Taiwan*

[3] *Department of Management Information Systems, National Chung Hsing University, 250 Kuo-kuang Road, Taichung 402, Taiwan*

[4] *Department of Information Engineering and Computer Science, Feng Chia University, No. 100 Wenhwa Rd., Seatwen, Taichung 407, Taiwan*

Correspondence should be addressed to Yung-Kuan Chan, ykchan@nchu.edu.tw

This paper is intended to present a lossless image compression method based on multiple-tables arithmetic coding (MTAC) method to encode a gray-level image f. First, the MTAC method employs a median edge detector (MED) to reduce the entropy rate of f. The gray levels of two adjacent pixels in an image are usually similar. A base-switching transformation approach is then used to reduce the spatial redundancy of the image. The gray levels of some pixels in an image are more common than those of others. Finally, the arithmetic encoding method is applied to reduce the coding redundancy of the image. To promote high performance of the arithmetic encoding method, the MTAC method first classifies the data and then encodes each cluster of data using a distinct code table. The experimental results show that, in most cases, the MTAC method provides a higher efficiency in use of storage space than the lossless JPEG2000 does.

1. Introduction

With the rapid development of image processing and Internet technologies, a great number of digital images are being created every moment. Therefore, it is necessary to develop an effective image-compression method to reduce the storage space required to hold image data and to speed the image transmission over the Internet [1–16].

Image compression reduces the amount of data required to describe a digital image by removing the redundant data in the image. Lossless image compression deals with reducing

coding redundancy and spatial redundancy. Coding redundancy consists in using variable-length codewords selected to match the statistics of the original source. The gray levels of some pixels in an image are more common than those of others—that is, different gray levels occur with different probabilities—so coding redundancy reduction uses shorter codewords for the more common gray levels and longer codewords for the less common gray levels. We call this process variable-length coding. This type of coding is always reversible and is usually implemented using look-up tables. Examples of image coding schemes that explore coding redundancy are the Huffman coding [4, 5, 7] and arithmetic coding techniques [8, 9].

There exists a significant correlation among the neighbor pixels in an image, which may result spatial redundancy in data. Spatial redundancy reduction exploits the fact that the gray levels of the pixels in an image region are usually the same or almost the same. Methods, such as the LZ77, LZ88, and LZW methods, exploit the spatial redundancy in several ways, one of which is to predict the gray level of a pixel through the gray levels of its neighboring pixels [14].

To encode an image effectively, a statistical-model-based compression method needs precisely to predict the occurrence probabilities of the data patterns in the image. This paper proposes a lossless image compression method based on multiple-tables arithmetic coding (MTAC) method to encode a gray-level image.

A statistical-model-based compression method generally creates a code table to hold the probabilities of occurrence of all data patterns. The type of data pattern significantly affects the encoding efficiency when minimizing storage space. When the data come from different sources, it is difficult to find an appropriate code table to describe all the data. Therefore, this MTAC method categorizes the data and adopts distinct code tables that record the frequencies which the data patterns occur in different clusters.

2. The MTAC Method

The proposed MTAC method contains three approaches: median edge detector (MED) processing, base-switching transformation, and statistical-model-based compressing. This section introduces these three approaches.

2.1. MED Processing Approach

Shannon's entropy equation can estimate the average minimum number of bits needed to encode a data pattern based on the frequency which the data pattern occurs in a data set [11, 12]. Let l be the total number of different data patterns in a data set and p_i the probability of the ith data pattern's occurring in the data set. The entropy rate E of the data set is defined as

$$E = \sum_{i=1}^{l} p_i \times \log_2\left(\frac{1}{p_i}\right). \tag{2.1}$$

It is impossible to encode the data set, in a lossless manner, with a bit rate higher than or equal to E. The bit rate is defined as the ratio of the number of bits holding the compression data to the number of pixels in the compressed image. The higher the entropy rate, the less one can compress it using a statistical-model-based compression method.

1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32
33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56
57	58	59	60	61	62	63	64

Figure 1: The scanning order of the MED in an image represented by 8×8 pixels.

Let f be the encoded image consisting of $H \times W$ pixels, where H and W are the height and the width of f, respectively. MED [10] estimates the gray level of a pixel by detecting whether there is an edge passing through the pixel. MED scans each pixel in f, starting from the left-top pixel of f, in the order shown in Figure 1. While scanning a pixel $P(i, j)$, MED estimates the gray level $g(i, j)$ of $P(i, j)$ via the gray-levels $g(i, j-1)$, $g(i-1, j-1)$, and $g(i-1, j)$ of the pixels $P(i, j-1)$, $P(i-1, j-1)$, and $P(i-1, j)$, where $P(i, j)$ is the pixel located at the coordinates (i, j) in f. Figure 2 shows the spatial relationships of $P(i, j)$, $P(i, j-1)$, $P(i-1, j-1)$, and $P(i-1, j)$.

For $i = 1$ or $j = 1$, the estimated gray level $\widehat{g}(i, j)$ of $P(i, j)$ is defined as

$$\widehat{g}(i, j) = \begin{cases} g(0, 0), & \text{when } i = 0, \ j = 0, \\ g(0, j) - g(0, j-1), & \text{when } i = 0, \ j = 1 \text{ to } W, \\ g(i, 0) - g(i-1, 0), & \text{when } j = 0, \ i = 1 \text{ to } H. \end{cases} \quad (2.2)$$

In addition, for $i = 1$ to H, and $j = 1$ to W:

$$\widehat{g}(i, j)$$

$$= \begin{cases} \text{Min}(g(i, j-1), g(i-1, j)), & \text{if } g(i-1, j-1) \geq \text{Max}(g(i, j-1)), (g(i-1, j)), \\ \text{Max}(g(i, j-1), g(i-1, j)), & \text{if } g(i-1, j-1) \geq \text{Min}(g(i, j-1), g(i-1, j)), \\ g(i, j-1) + g(i-1, j) - g(i-1, j-1), & \text{otherwise.} \end{cases}$$

$$(2.3)$$

Here, $\text{Max}(g(i, j-1), g(i-1, j))$ and $\text{Min}(g(i, j-1), g(i-1, j))$ are the maximum and the minimum between $g(i, j-1)$ and $g(i-1, j)$, respectively. If $\widehat{g}(i, j) = g(i, j-1)$, MED considers that a horizontal edge passes through $P(i, j)$ or some pixels above $P(i, j)$. When $\widehat{g}(i, j) = g(i-1, j)$, MED perceives that one vertical edge passes through $P(i, j)$ or some pixel on the left of $P(i, j)$.

Let $e(i, j) = g(i, j) - \widehat{g}(i, j)$ be the difference between $\widehat{g}(i, j)$ and $g(i, j)$; we call $e(i, j)$ the estimated error of $g(i, j)$. Similarly, in the decompressing phrase, based on $g(i, j-1)$, $g(i-1, j-1)$, and $g(i-1, j)$, the MTAC method can compute $\widehat{g}(i, j)$ through formulas (2.2) or (2.3); then it can get $g(i, j) = \widehat{g}(i, j) + e(i, j)$. To recover f without loss, the MTAC method needs to save $g(0, 0)$ and the estimated errors of all other pixels in f.

$P(i-1,j-1)$	$P(i-1,j)$	$P(i-1,j+1)$
$P(i,j-1)$	$P(i,j)$	$P(i,j+1)$
$P(i+1,j-1)$	$P(i+1,j)$	$P(i+1,j+1)$

Figure 2: Part of the pixels in an image.

The estimated error $e(i,j)$ is within the interval between -255 and 255. Each $e(i,j)$ can be represented by an 8-bit memory space that describes the absolution value $|e(i,j)|$ of $e(i,j)$ and one bit b that records the sign of $e(i,j)$. All the $|e(i,j)|$s compose a gray-level image f_e, and all the sign bits bs make up a binary image f_s. We call f_e the error image and f_s the sign bit image of f.

Figure 3 shows two 512×512 gray-level images Airplane and Baboon, and their gray-level histograms. Let $l = 256$, and

$$p_i = \frac{\text{the number of pixels whose gray levels are equal to } i}{\text{the total number of pixels in the image}}. \tag{2.4}$$

According to formula (2.1), the entropy rates of Airplane and Baboon, in theory, reach 6.5 and 7.2 bits/pixel, respectively. From Shannon's limit [11, 12], with such an entropy rate, the minimum number of bits required to describe a pixel in Airplane (resp., Baboon) is 6.5 (resp., 7.2) bits/pixel. Since the numbers of bits are over the acceptable maximum, the MTAC method utilizes MED [10] to decrease the entropy rate of f before encoding f. Figure 4 demonstrates the error images of Airplane and Baboon shown in Figure 3, and the gray-level histograms of the error images. The gray levels of most pixels in the error images are close to 0; the entropy rates of the error images of Airplane and Baboon are 3.6 and 5.2 bits/pixel, respectively, which are far lower than the entropy rates of the original Airplane and Baboon.

Figure 5 shows the sign bit images of Airplane and Baboon. It is clear that both sign bit images are messy, so it is difficult to find a method effectively with which to encode them effectively. To deal with this problem, the MTAC method transforms the error image and sign bit image into a difference image and an MSB (most significant bit) image, respectively. The MTAC method pulls out the MSB of all the $|e(i,j)|$s to create an $H \times W$ binary image f_{MSB}, where the MSB of $|e(i,j)|$ is given to the pixel located at the coordinates (i,j) of f_{MSB}. We call the binary image the MSB image f_{MSB} of f. Meanwhile, the MTAC method concatenates the sign bit b of $e(i,j)$ and the remaining $|e(i,j)|$, whose MSB has been drawn out, by appending b to the rightmost bit of the remaining $|e(i,j)|$ in order to generate another gray-level image. We name the gray-level image the difference image of f. Figure 6 illustrates these actions.

Figure 7 shows the MSB images of Airplane and Baboon. Almost all the pixels on the MSB images are 0. Figure 8 displays the difference images of Airplane and Baboon and their gray-level histograms. Clearly, the gray levels of most pixels in the difference images are equal to 0. The entropy rates of the difference images of Airplane and Baboon are 4.4 and 6.2 bits/pixel, respectively. These entropy rates are higher than the entropy rates of their error images but are much lower than those of the original Airplane and Baboon.
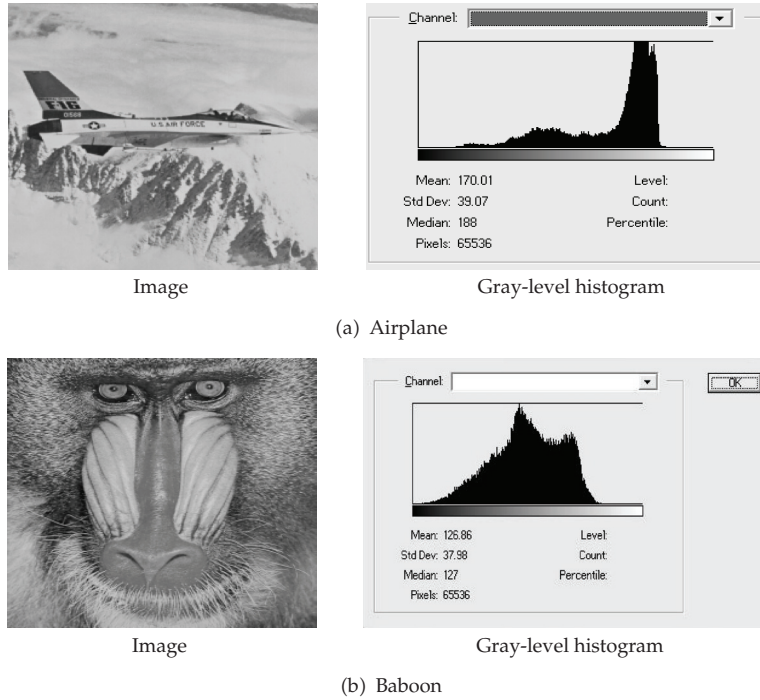
(a) Airplane

(b) Baboon

**Figure 3:** Two gray-level images, Airplane and Baboon**,** and their color histograms.

2.2. Base-Switching Transformation Approach

The gray level of a pixel in a gray-level image is generally represented by an 8-bit memory space. However, it is uneconomical if the gray levels of the pixels in a gray-level image are similar. Hence, the MTAC method adopts the base-switching transformation (BST) algorithm [1, 2] to compress a difference image.

The BST algorithm partitions a difference image into small nonoverlapping image blocks, each consisting of $m \times n$ pixels. Let g_{min} and g_{max} be the minimal and maximal gray levels of the pixels in an image block B. The difference between g_{min} and the gray level g of each pixel in B can be depicted by $\lceil \log_2(g_{max} - g_{min}) \rceil$ bits. The MTAC method uses a 3-bit memory space S to describe $\lceil \log_2(g_{max} - g_{min}) \rceil$, where

$$
S = \begin{cases} 0, & \text{if } \lceil \log_2(g_{max} - g_{min}) \rceil = 0 \text{ or } 1, \\ \lceil \log_2(g_{max} - g_{min}) \rceil - 1, & \text{otherwise.} \end{cases} \tag{2.5}
$$

For each image block, the BST algorithm needs to hold only g_{min}, S, and the gray-level differences between g_{min} and the gray levels of all the pixels in B. We call the difference between the gray level of a pixel P and g_{min} the gray-level difference of P. Figure 9 is a 4×4 image block B. The $16 \times 8 = 128$ bits of memory space are required to store B. However, in the BST algorithm, g_{max}, g_{min}, and S of B are 137, 122, and 3, respectively. The BST algorithm uses 8 bits, 3 bits, and 4×16 bits to hold g_{min}, S, and the gray-level differences of all the pixels in B; hence, the BST algorithm requires only a total of 75 bits to store B.

Error image   Gray-level histogram

(a) Airplane

Error image   Gray-level histogram
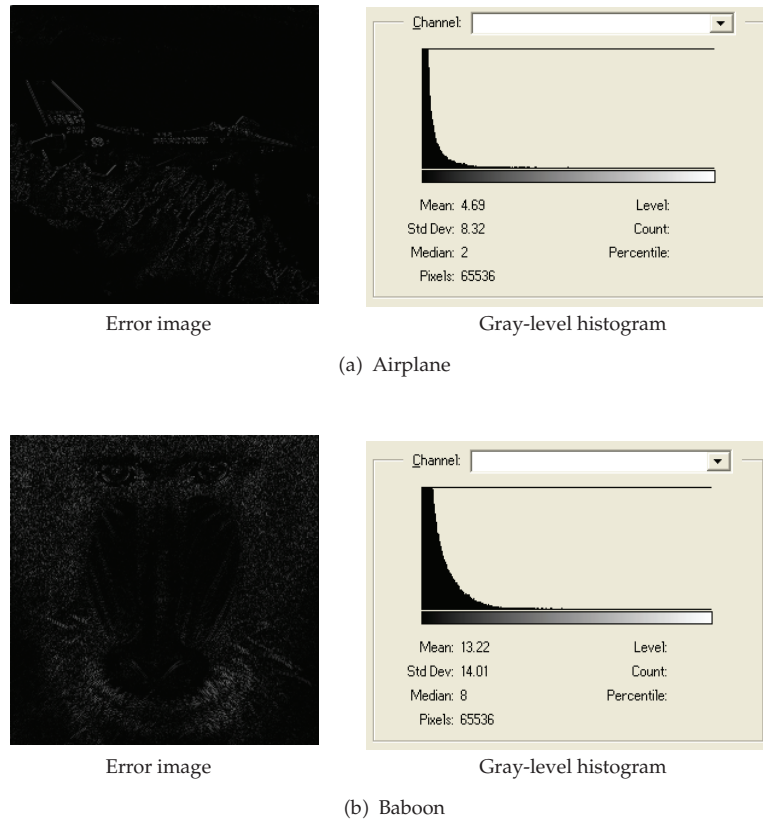
(b) Baboon

Figure 4: The difference images and their histograms of Airplane and Baboon.

2.3. Statistical-Model-Based Compressing Approach

After the MED processing approach, image f is transformed into an MSB image and a difference image. In the base-switching transformation approach, the difference image is segmented into nonoverlapping small image blocks. The MTAC method then writes down g_{min}, S, and the gray-level differences of all the pixels in each image block. However, a few pixels may have big gray-level differences in an image block, so each gray-level difference in this image block requires large number of bits to hold it. For example, the maximal gray-level difference of the image block in Figure 9 is 15; therefore, each gray-level difference can be expressed by at least 4 bits. To remedy this problem, the MTAC method takes arithmetic coding algorithm continuously to compress the data obtained in the MED processing and BST approaches.

The arithmetic coding algorithm [8, 15] is one of the statistical-model-based compressing methods that decide the bit length of a code according to the occurrence frequencies of data patterns. These methods give longer codes to the data patterns that occur more frequently and shorter codes to those that occur less often. Hence, the type of data pattern significantly affects the encoding's efficiency in minimizing storage space. The MTAC method will adopt the arithmetic coding algorithm to compress the MSB image, all the g_{min}s, and all Ss. Since the MSB image, all the g_{min}s, and all Ss have different statistics, the MTAC method will require distinct code tables to record the data patterns of the MSB image, all the
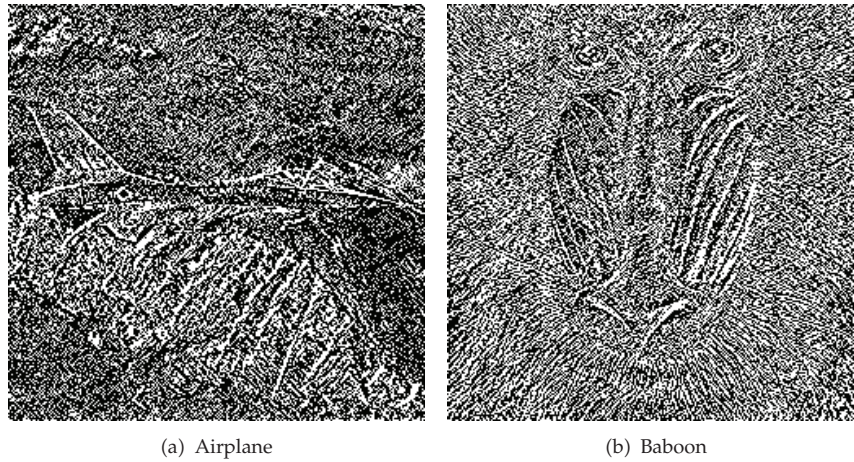
(a) Airplane   (b) Baboon

Figure 5: The sign bit images of the images Airplane and Baboon.

(a) The original $\bar{e}(i,j)$ and b
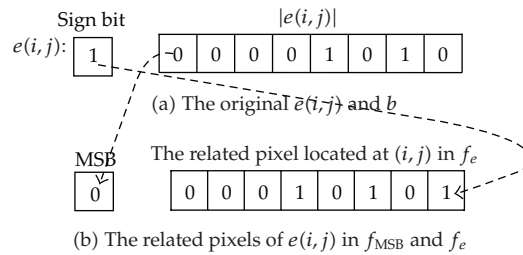
(b) The related pixels of $e(i,j)$ in f_{MSB} and f_e

Figure 6: The pixel values of $e(i,j)$.

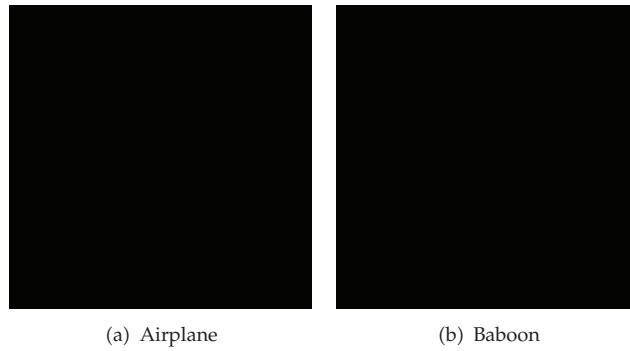g_{min}s, and all Ss. Each data pattern of the MSB image, all the g_{min}s, and all Ss are described by 8-bits, 8-bits, and 3-bits in length, respectively.

Next, the MTAC method concatenates the gray-level differences of all the image blocks into a binary string GD_S, where the bit length of each gray-level difference in the image blocks is S. For example, the gray-level differences in all the image blocks with $S = 4$ are concatenated into GD_4. The MTAC method then uses an arithmetic coding algorithm to encode each GD_S, where the bit length of each data pattern in encoding GD_S is S. Finally, the MTAC method needs to hold only the height H and the width W of f, all the code tables, and all the compression data generated by the arithmetic coding algorithm.
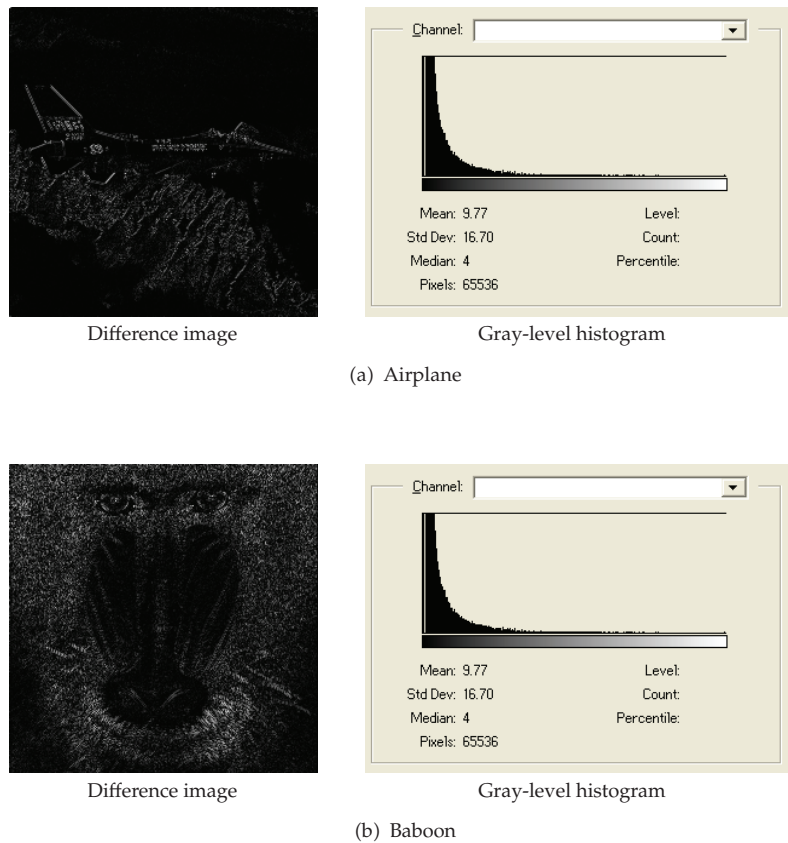
After the statistical-model-based compressing approach has been employed, the MTAC method concatenates W, H, $String_{CODE\_TABLE}$, $String_{MSB}$, $String_{g\,min}$, $String_S$, and $String_{GD}$ into the compression data. Here, $String_{CODE\_TABLE}$ represents all the code tables; $String_{MSB}$, $String_{g\,min}$, $String_S$, and $String_{GD}$ are the compression data of the MSB image, all g_{min}s, Ss, and GD_Ss, respectively.

2.4. Image Decompression

In the decompression phrase, the MTAC method first draws W, H, and $String_{CODE\_TABLE}$ from the compression data. The bit length of each data pattern in the MSB image is 8.

(a) Airplane   (b) Baboon

Figure 7: The MSB images of Airplane and Baboon.

Difference image   Gray-level histogram

Mean: 9.77   Level:
Std Dev: 16.70   Count:
Median: 4   Percentile:
Pixels: 65536

(a) Airplane

Difference image   Gray-level histogram

Mean: 9.77   Level:
Std Dev: 16.70   Count:
Median: 4   Percentile:
Pixels: 65536

(b) Baboon

Figure 8: The difference images and color histograms of Airplane and Baboon.

123	122	125	126
122	125	124	127
124	126	125	128
125	124	129	137

Figure 9: An image block of 4 × 4 pixels.
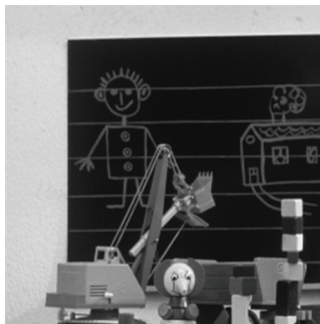
(a) Airplane

(b) Lena

(c) Baboon

(d) Gold

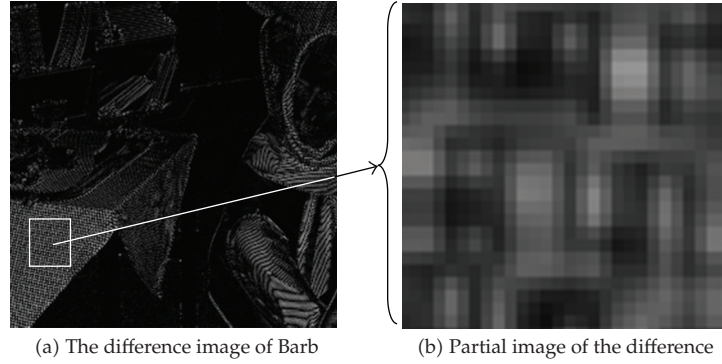(e) Sailboat

(f) Boat

(g) Toy

(h) Barb

(i) Pepper

(j) Girl

Figure 10: The testing images.

(a) The difference image of Barb   (b) Partial image of the difference

Figure 11: The difference image of Barb and its partial image.

Table 1: Entropies of ten original images and their error images.

Image	Entropy rate of original image	Entropy rate of error image
Airplane	6.529	3.567
Baboon	7.224	5.091
Lena	7.432	3.681
Toy	6.748	3.123
Gold	7.452	3.853
Sailboat	7.248	4.126
Boat	6.975	3.595
Barb	7.647	4.557
Pepper	7.570	3.535
Girl	7.260	3.935

Hence, the MTAC method can reconstruct the MSB image based on String$_{\text{MSB}}$ by using the arithmetic decoding method. Since f consists of $H \times W/9$ image blocks, the MTAC method will decompress the $(H \times W/9)g_{\min}$s from String$_{g \min}$ using the arithmetical decoding method, where the bit length of a data pattern is 8. Similarly, it can decode $(H \times W/9)S$s from String$_S$, where each data pattern is described by 3 bits. How many data patterns are in each GD$_S$ can be easily computed via Ss. Hence, each GD$_S$ can be decoded as well.

3. Experiments

The purpose of this section is to investigate the performance of the MTAC method by experiments. In these experiments, ten 256×256 gray-level images Airplane, Lena, Baboon, Gold, Sailboat, Boat, Toy, Barb, Pepper, and Girl, shown in Figure 10, are used as test images. The first experiment explores the effect of the MED processing approach on reducing the entropy rate of the compressed image. Table 1 lists the entropy rates of the ten original test images and the entropy rates of their error images. The experimental results show that most of the entropy rates of the error images are close to half those of the original test images.

In experiment 2, the arithmetic coding method is used to encode the sign bit images of the ten test images, where the bit length of each data pattern is 8 bits. Table 2 shows the sizes of the original sign bit images and their compression data obtained by the arithmetic coding

Table 2: The size of the sign bit images and their compression data.

Image	Size of original image (bytes)	Size of compression data (bytes)
Airplane	8192	8069
Baboon	8192	8256
Lena	8192	8187
Toy	8192	8135
Gold	8192	8169
Sailboat	8192	8310
Boat	8192	8152
Barb	8192	8222
Pepper	8192	8201
Girl	8192	8296

Table 3: Entropy rates of the difference images.

Image	Entropy rate
Airplane	4.368
Baboon	6.048
Lena	4.527
Toy	3.900
Gold	4.742
Sailboat	4.997
Boat	4.396
Barb	5.442
Pepper	4.390
Girl	4.801

method where the bit length of each data pattern is 8 bits. The experimental results illustrate the difficulty of obtaining a good performance in compressing the sign bit images. Since the sign bit images are very messy, the arithmetic coding method cannot effectively encode it; even for most error images, the sizes of their compression data are larger than the sizes of their original images; that is, there hardly exist the coding and spatial redundancies in the error images. Table 3 lists the entropy rates of the difference images of the ten test images where $m \times n$ are set to 3×3. Although the entropy rates of the difference images are higher than those of the error images, they are much lower than those of the original images are.

The last experiment compares the performance of the MTAC method with that of the lossless JPEG2000 [3, 6, 13]. Table 4 displays the bit rates obtained by the MTAC method and the lossless JPEG2000 in encoding the test images. Table 4 reveals that the MTAC method is more efficient in storage space use than the lossless JPEG2000, except image Barb. Figure 11 shows that huge gray-level variations among most adjacent pixels in the partial magnified difference image of Barb. The MTAC method obtains better performance in terms of storage space use than that of the lossless JPEG2000 in encoding an image with small gray-level variations among adjacent pixels but performs worse in compressing an image with great gray-level variations among adjacent pixels. Hence, the MTAC method performs worse in terms of storage space use than the lossless JPEG2000 does when encoding Barb.

Table 4: Bit rates (bits/pixel) obtained by the MTAC and lossless JPEG 2000.

Image	Method	
	MTAC	Lossless JPEG2000
Airplane	4.14	4.35
Lena	4.20	4.25
Baboon	6.10	6.11
Gold	4.71	4.90
Sailboat	4.88	5.10
Boat	4.24	4.44
Toy	3.90	4.16
Barb	*5.28*	5.14
Pepper	4.34	4.43
Girl	4.63	4.73

4. Conclusions

This paper proposes the MTAC method to encode a gray-level image f. The MTAC method contains the MED processing, BST, and statistical-model-based compressing approaches. The MED processing approach reduces the entropy rate of f. The BST approach decreases the spatial redundancy of the difference image of f based on the similarity among adjacent pixels. The statistical-model-based compressing approach further compresses the data generated in the MED processing and BST approaches, based on their coding redundancy. The data patterns of the data produced by the MED processing approach and the BST approach have different bit lengths and distinct occurrence frequencies. Hence, the MTAC method first classifies the data into clusters before compressing the data in each cluster using the arithmetic coding algorithm via separated code tables.

The experimental results reveal that the MTAC method usually gives a better bit rate than the lossless JPEG2000 does, particularly for the images with small gray-level variations among adjacent pixels. However, when the gray-level variations among adjacent pixels in an image are very large, the MTAC method performs worse in terms of bit rate.

References

[1] T. J. Chuang and J. C. Lin, "On the lossless compression of still image," in *Proceedings of the International Computer Symposium-On Image Processing and Character Reognition (ICS '96)*, pp. 121–128, Kaohsiung, Taiwan, 1996.

[2] T. J. Chuang and J. C. Lin, "New approach to image encryption," *Journal of Electronic Imaging*, vol. 7, no. 2, pp. 350–356, 1998.

[3] S. C. Diego, G. Raphaël, and E. Touradj, "JPEG 2000 performance evaluation and assessment," *Signal Processing: Image Communication*, vol. 17, no. 1, pp. 113–130, 2002.

[4] Y. C. Hu and C.-C. Chang, "A new lossless compression scheme based on huffman coding scheme for image compression," *Signal Processing: Image Communication*, vol. 16, no. 4, pp. 367–372, 2000.

[5] D. A. Huffman, "A method for the construction of minimum redundancy codes," in *Proceedings of the IRE*, vol. 40, pp. 1098–1101, 1952.

[6] ISO/IEC FCD 155444-1, Information Technology-JPEG 2000 Image Coding System, 2000.

[7] D. E. Knuth, "Dynamic Huffman coding," *Journal of Algorithms*, vol. 6, no. 2, pp. 163–180, 1985.

[8] G. G. Langdon Jr., "An introduction to arithmetic coding," *IBM Journal of Research and Development*, vol. 28, no. 2, pp. 135–149, 1984.

[9] J. Rissanen and G. G. Langdon Jr., "Arithmetic coding," *IBM Journal of Research and Development*, vol. 23, no. 2, pp. 149–162, 1979.

[10] S. A. Martucci, "Reversible compression of HDTV images using median adaptive prediction and arithmetic coding," in *Proceedings of the IEEE International Symposium on Circuits and Systems*, vol. 2, pp. 1310–1313, New York, NY, USA, 1990.

[11] C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, pp. 379–423, 1948.

[12] C. E. Shannon, "Prediction and entropy of printed English," *Bell System Technical Journal*, vol. 30, pp. 50–64, 1951.

[13] A. N. Skodras, C. A. Christopoulos, and T. Ebrahimi, "JPEG 2000: the upcoming still image compression standard," *Pattern Recognition Letters*, pp. 1337–1345, 2001.

[14] U. Topaloglu and C. Bayrak, "Polymorphic compression," in *Algorithms and Database Systems*, P. Yolum, et al., Ed., vol. 3733 of *Lecture Notes in Computer Science*, pp. 759–767, Springer, New York, NY, USA, 2005.

[15] T. Welch, "A technique for high-performance data compression," *IEEE Computer*, vol. 17, no. 6, pp. 8–19, 1984.

[16] M. U. Celik, G. Sharma, and A. M. Tekalp, "Gray-level-embedded lossless image compression," *Signal Processing: Image Communication*, vol. 18, no. 6, pp. 443–454, 2003.