ELSEVIER

January 1995

Pattern Recognition Letters

Pattern Recognition Letters 16 (1995) 67–75

Improving block truncation coding by line and edge information and adaptive bit plane selection for gray-scale image compression ☆

Chen-Kuei Yang [1], Wen-Hsiang Tsai *

Department of Computer and Information Science, National Chiao Tung University, Hsinchu, Taiwan 300, ROC

Received 16 February 1994; revised 2 August 1994

Abstract

A new approach to improving block truncation coding for gray-scale image compression is proposed. A set of line and edge bit planes is defined independently of input images and adaptively selected to yield lower bit rates and better reconstructed image quality.

Keywords: Block truncation coding; Image compression; Gray-scale image; Adaptive bit plane selection

1. Introduction

The block truncation coding (BTC) algorithm developed by Delp and Mitchell (1979) is a two-level nonparametric quantizer whose output levels are obtained by preserving the first- and second-order moments of input samples. According to the moment-preserving principle, other coding algorithms for improving the performance of the BTC algorithm have also been proposed (Halverson et al., 1982; Lema and Mitchell, 1984; Hui, 1990). These methods can be employed to compress an input image with less distortion than the original BTC method. Udpikar and Raina (1987) developed a method, which applies vector quantization to BTC results, to truncate the bit plane or the mean vector, or both. Zeng and Neuvo (1993) developed another method which subsamples the BTC bit plane and uses the median filter as the interpolator. These methods improve the bit rate but introduce distortion into the reconstructed image and need additional computational load.

In the 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

$$a=\bar{x}-\bar{\sigma}\sqrt{\frac{q}{m-q}} \quad \text{for } x_i<\bar{x}, \tag{1}$$

$$b=\bar{x}+\bar{\sigma}\sqrt{\frac{m-q}{q}} \quad \text{for } x_i\geqslant\bar{x} \tag{2}$$

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

☆ This work was supported partially by National Science Council, Republic of China under Grant NSC83-0408-E009-010.

* Corresponding author. Email: whtsai@cis.nctu.edu.tw

[1] Also with the Department of Information Management, Ming Chuan College, Taipei.

SSDI 0167-8655(94)00068-9

$$\bar{x}=\frac{1}{m}\sum_{i=1}^{m} x_i, \tag{3}$$

$$\bar{\sigma}=\sqrt{\frac{1}{m}\sum_{i=1}^{m}(x_i-\bar{x})^2} \tag{4}$$

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 the mean and the standard deviation, or a and b, together with a 4×4 bit plane consisting of 1's and 0's indicating whether the pixels are quantized to a or b. Assigning 8 bits to each of a and b results in a bit rate of 2 bits/pixel. To obtain higher compression ratios without increasing computational expenses, it is better to take into account the properties of the human visual system. Certain low-level aspects of vision can be accurately modeled. Some well-known psychophysical and physiological properties of the human visual system have been utilized in the design of certain coding algorithms. For example, it was found that the human eye assigns different attentions to the low- and high-frequency components of images, and this fact leads to the development of the subband coding algorithms (Forchheimer and Kronander, 1987; Chen and Hsing, 1991).

Decomposition of images into low-frequency blocks and blocks containing visually important features (such as edges or lines) suggests visual continuity and visual discontinuity constraints. A block is visually continuous if the values of all the pixels in the block are almost the same. On the contrary, if the variations of the pixel values in the block are noticeable, it is a visually discontinuous block. The mean of a visually continuous block is enough to represent the block. If a block is visually discontinuous and if a strong orientation is associated with it, then it should be coded as a kind of visually important feature (Chen and Bovik, 1990) like edge or line.

The BTC algorithm uses a one-bit quantizer to adapt the local properties of the image. The bit planes occupy more than 61% of the output code. A new method for gray-scale image compression using predefined line and edge bit planes to improve the BTC method is proposed. A set of line and edge bit planes is defined according to the visual continuity and discontinuity constraints. The line and edge bit planes are defined independently of the images to be coded. For each input image block, the proposed method follows the steps of the conventional BTC algorithm until a bit plane is generated. Then the bit plane is matched with the predefined bit planes. This results in a best-match predefined bit plane. The set of all the best-match predefined bit planes (each corresponding to an input image block) is classified and those types which occur more frequently in the set are picked out. Assume that n types of predefined bit planes are selected. Then, each of the original bit planes of the input image blocks is matched once again with the selected n predefined bit planes. Each block is finally coded by the values of the mean, the standard deviation, and the index of the best-match predefined bit plane. Experimental results show that the proposed approach is effective for gray-scale image compression.

Advantages of the proposed method include at least the following.

(1) The method is based on the visual continuity and discontinuity constraints found in the human visual system which leads to the reasonable assumption that only edge and line patterns exist in small discontinuous image blocks. And due to the use of edge and line bit planes whose number is much smaller (64 in our study), 4×4 bit planes produced by BTC can be coded by indices with at most 6 bits instead of 16 bits.

(2) By selecting out only those predefined bit planes which occur more frequently and rematching them with the input bit plane, it requires even fewer bits to code the input bit plane without sacrificing too much quality in the reconstructed image. The bit rate is about 1 bit/pixel for 4×4 blocks if all the sixty-four predefined types of bit planes are used, and about 0.8125 bit/pixel for 4×4 blocks if only eight types of the predefined bit planes are used.

(3) By controlling the number n of the selected types of predefined bit planes which occur more frequently, adaptation to the input image can be achieved both in preserving reconstructed image quality and gaining reasonable bit rates.

The remainder of this paper is organized as follows. The proposed line and edge bit planes are defined in Section 2. In Section 3, the adaptive bit plane selection scheme to select the n types of more frequently used bit planes is described. Several experi-

mental results shown to support the feasibility of the proposed method and some discussions are included in Section 4. Finally, some conclusions and suggestions for future study are made in Section 5.

2. Design of line and edge bit planes

With each visual bit plane representing a possible image block content, the number of possible bit planes increases exponentially with the block size. Therefore, a small block size must be selected for designing the desired line and edge bit planes. On the other hand, the line and edge bit planes proposed for use in this study basically are derived from the idea of the twelve types of line edges in a block mentioned in (Ferng and Tsai, 1992). According to the concept of visual discontinuity, it is assumed that the variation of the pixel values in a 4×4 block is noticeable only when there exist at least three or more pixels which form a neighborhood and show a shape of edge or line. Accordingly, sixty-four types of line and edge bit planes for 4×4 blocks are designed and shown in Fig. 1.

Because the ratio of the uniform bit planes to the non-uniform ones in a normal gray-scale image is usually small enough to be ignored and some predefined bit planes may have lower activity in certain images, the number of used predefined bit planes for each input image can be reduced further and adaptively in practical applications without introducing

1100	1111	1100	1111	1111	0011	0011	1100
1100	1111	1110	1111	1111	0111	0111	1110
1100	0000	1111	1110	0111	1111	1110	0111
1100	0000	1111	1100	0011	1111	1100	0011
(1)	(3)	(5)	(7)	(9)	(11)	(13)	(15)
0000	0111	1111	1110	1111	0001	1000	1111
1111	0111	1111	1110	0111	0011	1100	1110
1111	0111	1111	1110	0011	0111	1110	1100
1111	0111	0000	1110	0001	1111	1111	1000
(17)	(19)	(21)	(23)	(25)	(27)	(29)	(31)
1011	1101	1111	1111	0000	1001	0000	1010
1011	1101	0000	1111	1111	1001	1111	1010
1011	1101	1111	0000	1111	1001	0000	1010
1011	1101	1111	1111	0000	1001	1111	1010
(33)	(35)	(37)	(39)	(41)	(43)	(45)	(47)
1100	0011	1000	0001	1000	0001	1001	0000
0110	0110	1100	0011	0100	0010	0110	0000
0011	1100	0110	0110	0010	0100	0110	0000
0001	1000	0011	1100	0001	1000	1001	0000
(49)	(51)	(53)	(55)	(57)	(59)	(61)	(63)

Fig. 1. Sixty-four types of line and edge bit planes (reverse versions not shown).

visible errors, leading to the idea of adaptive bit plane selection described in detail in the next section.

3. Adaptive bit plane selection

Basically, the sixty-four types of line and edge bit planes for 4×4 image blocks can be classified into three groups based on the amount of activity inside the image:

(1) low-activity bit planes which include features like thin line with low frequency to appear in normal gray-scale images;

(2) medium-activity bit planes which contain smooth contrast edges;

(3) high-activity bit planes which contain basic line and edge patterns appearing most frequently in images.

In order to code the blocks efficiently and keep the high-activity bit planes in use, we adopt an adaptive bit plane selection algorithm described as follows.

Algorithm Adaptive Bit Plane Selection (ABPS)

Step 1. Read in an $M\times N$ gray-scale image G.

Step 2. Partition G into nonoverlapping 4×4 blocks F_j.

Step 3. For each block F_j, perform the following steps of "initial matching":

(1) perform BTC to generate the mean $\bar{x}_j$, the standard deviation $\bar{\sigma}_j$, and the bit plane b_j;

(2) match b_j with the sixty-four predefined line and edge bit planes;

(3) randomly choose one of the best matches if there is more than one best match, and let i_j be the index of the best-match predefined bit plane;

(4) increment by one the value of a counter c_{i_j} corresponding to index i_j, which indicates the activity of the best-match bit plane with index i_j in the input image;

(5) quantize $\bar{x}_j$ to 6 bits and $\bar{\sigma}_j$ to 4 bits (Delp and Mitchell, 1979);

(6) store $\bar{x}_j$, $\bar{\sigma}_j$, and b_j in an array S.

Step 4. Sort all the values of the counters in the decreasing order.

Step 5. Select the bit planes whose corresponding counter values are the top n largest (i.e., which appear more frequently in the input image) and store them in an array BP.

Step 6. For each element b_j in S, perform the following steps of "rematching":

(1) match b_j with the n bit planes stored in BP;

(2) randomly choose one of best matches if there is more than one best match;

(3) output the $\bar{x}_j$, $\bar{\sigma}_j$, and the corresponding index i'_j of the best match (note that index i'_j may be different from index i_j found in Step 3(3)).

The selection of the best-match predefined bit plane in Steps 3 and 6 above is based on template matching using the smallest number of different pixel values as the dissimilarity measure. This measure can be used to select easily the best-match predefined bit plane without causing too much visual difference according to our experimental experience. One reason for this is that the sixty-four predefined bit planes cover most visually obvious patterns existing in a 4×4 pixel block, and so the selection of the best-match bit plane to replace the input image block using the proposed measure will not produce too many changes of bit values.

The reconstruction procedure is the same as the conventional BTC. Because we use the index of the best-match predefined bit plane instead of the original bit plane as the output, we can use 6-bit indices instead of 16-bit bit planes. We can so reduce further the number of used bits when $n<64$ where n is mentioned in Step 5 in the above algorithm.

The overall bit rate can be computed as follows:

$$R=\frac{m+s+\Delta}{16} \tag{5}$$

where m is the number of bits used for the sample mean, s the number of bits used for the standard deviation, and Δ the number of bits used for indexing the selected n bit planes. For example, for $m=6$, $s=4$, and $\Delta=4$, the bit rate R is 0.875 bit/pixel. Note that $\Delta=\lceil\log_2 n\rceil$ where $\lceil\ \rceil$ means the "ceiling" of a real number.

Because the sixty-four 16-bit bit planes are predefined and fixed, they can be provided in advance at both the sender and the receiver ends. Therefore, it is unnecessary to include the number of bits used for storing the n bit planes in the overall bit rate computed by Eq. (5). Instead, only the bits used to index the selected n bit planes are needed, whose number is specified by Δ in Eq. (5).

4. Experimental results and discussions

The proposed approach has been tested on an IRIS Indigo workstation on several gray-scale images. Each gray-scale image has 8 bits/pixel and is 512×512 in size. To evaluate the performance of the proposed approach, we use a new criterion in this study, called the average bit plane assignment error (ABPAE), which is described in the following. For convenience of illustration, first define the bit difference of an input image block F_j as follows:

$$\mathrm{BD}_j=\sum_{x=1}^{4}\sum_{y=1}^{4}|b'_j(x,y)-b_j(x,y)| \tag{6}$$

where $b'_j(x,y)$ is the value of the bit plane generated for F_j by the proposed method ABPS at (x,y) and $b_j(x,y)$ the value of the bit plane generated for F_j by the BTC method at (x,y). Then, the bit plane assignment error of F_j is defined as follows:

Table 1
The ABPAE values and bit rates of applying the conventional BTC and the proposed ABPS to some images using 64, 32, 16, and 8 predefined bit planes

Image name	A 8 bit planes	B 16 bit planes	C 32 bit planes	D 64 bit planes	E conventional BTC
Lenna	0.363	0.249	0.145	0.047	0.000
pepper	0.427	0.322	0.194	0.084	0.000
jet	0.427	0.322	0.096	0.021	0.000
mandrill	0.435	0.287	0.162	0.040	0.000
candy	0.202	0.001	0.000	0.000	0.000
house	0.144	0.000	0.000	0.000	0.000
bit rate	0.8125 b/pixel	0.875 b/pixel	0.9375 b/pixel	1 b/pixel	2 b/pixel

Table 2
The MAE values and bit rates of applying the conventional BTC and the method ABPS to some images using 64, 32, 16, and 8 predefined bit planes

Image names	A 8 bit planes	B 16 bit planes	C 32 bit planes	D 64 bit planes	E conventional BTC
Lenna	4.55	4.16	3.96	3.85	3.06
pepper	5.05	4.40	4.17	4.07	3.08
jet	4.77	4.19	3.88	3.82	3.07
mandrill	13.18	12.22	11.36	11.01	7.85
candy	2.25	1.63	1.55	1.55	1.20
house	2.26	1.79	1.79	1.79	1.49
bit rate	0.8125 b/pixel	0.875 b/pixel	0.9375 b/pixel	1 b/pixel	2 b/pixel

Fig. 2. The original images of "Lenna" left and "mandrill" right, both with size 512×512.

$$\mathrm{BPAE}_j = \begin{cases} 1, & \text{if } \mathrm{BD}_j > 4, \\ 0, & \text{otherwise.} \end{cases} \tag{7}$$

According to our experimental results, we notice that two bit planes with their corresponding bit difference value less than 4 usually cannot be distinguished by the human vision system. Therefore, in (7) above, when BPAE_j is set to 1, it means that the bit plane assignment is unsuitable because the bit difference value is larger than 4.

Finally, the ABPAE of an input image is defined as follows:

$$\mathrm{ABPAE} = \frac{1}{B} \sum_{j=1}^{B} \mathrm{BPAE}_j \tag{8}$$

where B is the number of blocks in the input image.

A smaller value of the ABPAE means that the bit plane assignment causes fewer errors. Table 1 shows the ABPAE values of some experiments conducted on six images "Lenna", "pepper", "jet", "mandrill", "candy", and "house" using 64, 32, 16, and 8 predefined bit planes (i.e., with the value of n mentioned in Step 5 in Algorithm ABPS being 64, 32, 16, and 8). It can be seen that the ABPAE values of the column of using the 64 predefined bit planes (Column D) are smaller than 0.084; that means the bit planes generated by the proposed method are visually almost identical to the bit planes generated by the BTC method. However, from the last row of Table 1, it is seen that the compression ratio obtained from the proposed method using the 64 predefined bit planes is two times that of the BTC method. Also, it can be observed that for images with uniform regions like "candy" and "house", the proposed method can both preserve the reconstructed image quality and gain reasonable compression ratios due to the adaptive selection of the n predefined bit planes (with $n<64$) mentioned in Step 5 of Algorithm ABPS. That is, it requires fewer bits to code the selected bit planes without sacrificing too much quality in the reconstructed image.

We also use the mean absolute error (MAE) as another criterion to evaluate the quality of the reconstructed image. The MAE is defined as follows:

Fig. 3. The reconstructed "Lenna" using 64, 32, 16, and 8 predefined bit planes in to-left, top-right, bottom-left, and bottom-right, respectively.

$$\text{MAE}=\frac{1}{MN}\sum_{M}\sum_{N}|f_1(x,y)-f_2(x,y)| \tag{9}$$

where $M\times N=512\times 512$ and $f_1(x,y)$ and $f_2(x,y)$ are the pixel gray-values of the reconstructed and original images, respectively, at image coordinates (x,y). A smaller value of the MAE means that the reconstructed image is less distorted.

Table 2 shows the MAE values obtained from performing the proposed algorithm ABPS and the conventional BTC method on the above-mentioned images using 64, 32, 16, and 8 predefined bit planes. It can be seen that the more used bit planes, the smaller the MAE values. For each image, there exists possibly an optimal number of bit planes that yields the highest compression ratio with visual quality still being kept reasonable in the reconstructed image.

Fig. 2 shows the original images of "Lenna" and "mandrill" of size 512×512. Figs. 3 and 4 show the reconstructed images of "Lenna" and "mandrill", respectively, after using the ABPS as the compression algorithm. Each figure shown includes the reconstructed images using 64, 32, 16, and 8 predefined bit planes in top-left, top-right, bottom-left, and bottom-right, respectively. Note that even when only 8 predefined bit planes are used (resulting in a good bit rate of 0.8125), the reconstructed image quality is still acceptable. This is due to the adaptive selection of fewer but more frequently occurring bit planes (see Step 6 of Algorithm ABPS). Only a few zigzag edges are noticeable which are due to the aliasing effect resulting from block truncation using two gray-levels for each block as well as the use of fewer bit planes.

Fig. 5 shows the "error images" of "Lenna" which includes the gray-value differences between the original and the reconstructed images using 64, 32, 16, and 8 predefined bit planes in top-left, top-right, bottom-left, and bottom-right, respectively. The errors are magnified five times since otherwise the pictures would be almost fully black. This means that the difference values are very small. Also, the errors are distributed in the images. These facts indicate that the proposed approach is effective. As can be seen, the errors are distributed in the images especially on the

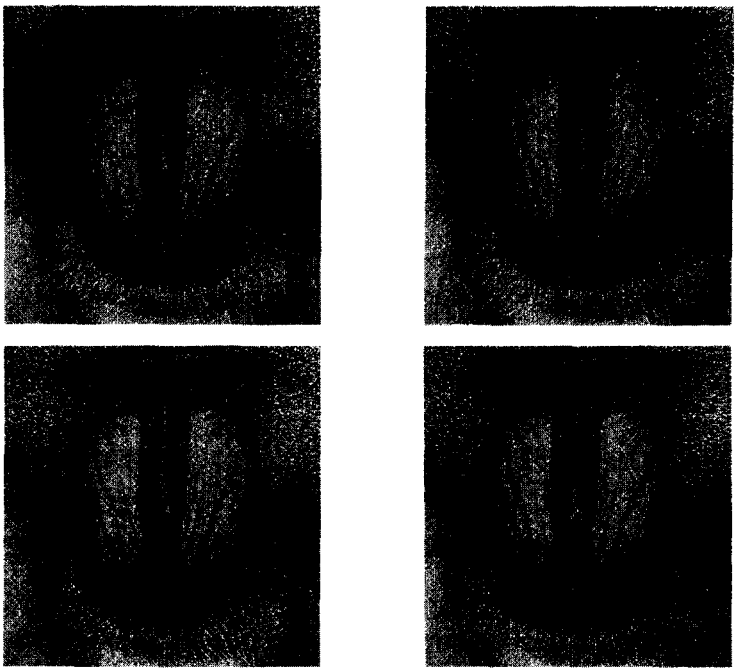

Fig. 4. The reconstructed "mandrill" using 64, 32, 16, and 8 predefined bit planes in top-left, top-right, bottom-left, and bottom-right, respectively.

line and edge locations.

In some cases, it is desired to yield automatically the number n of bit planes selected for use in rematching (see Steps 5 and 6 of Algorithm ABPS) according to the distributions of the counter values c_i mentioned in Step 3(4) in ABPS. One way to achieve this is to insert an additional step between Step 4 and Step 5 of ABPS as follows:

Step 4.5. For automatic selection of the value of n,

(1) set variables $m=0$ and $Sum=0$;

(2) let the sorted counters be indexed by $i=1$ through 64 with $i=1$ for the largest counter value;

(3) set $i=1$;

(4) set $Sum=Sum+c_i$;

(5) set $m=m+1$;

(6) if the value of Sum is smaller than 90% of the total number of the image blocks, then set $i=i+1$ and go to (4), else go to (7);

(7) set n to 2^k, where k is such that $2^{k-1}<m\leq 2^k$.

In Step 4.5(6), when the value of Sum is larger than 90% of the total number of image blocks, it means that 90% of the blocks will be rematched to identical predefined bit planes in Step 6 of ABPS, and the resulting value of the ABPAE may be kept smaller than 0.1 as can be seen from Column D of Table 1.

Table 3 shows the results of automatic selection of n by Step 4.5 of ABPS. It can be seen that forty-four to forty-nine bit planes are required when 90% of the image blocks are included, and the values of the ABPAE are close to 0.1 except those of the images "candy" and "house". This means that images with randomized image contents usually require sixty-four predefined bit planes if good quality of the reconstructed image is desired. However, images like "candy" and "house", which have uniform regions, usually require fewer bits to represent the corresponding indices of the selected bit planes because the number of the selected bit planes is smaller.

Finally, it is mentioned that the idea of the proposed approach is only suitable for smaller blocks and not applicable for bigger blocks (8×8 blocks, for example). Applying the BTC method to bigger blocks will cause blocky artifacts to appear on the reconstructed image, and the same situation will appear

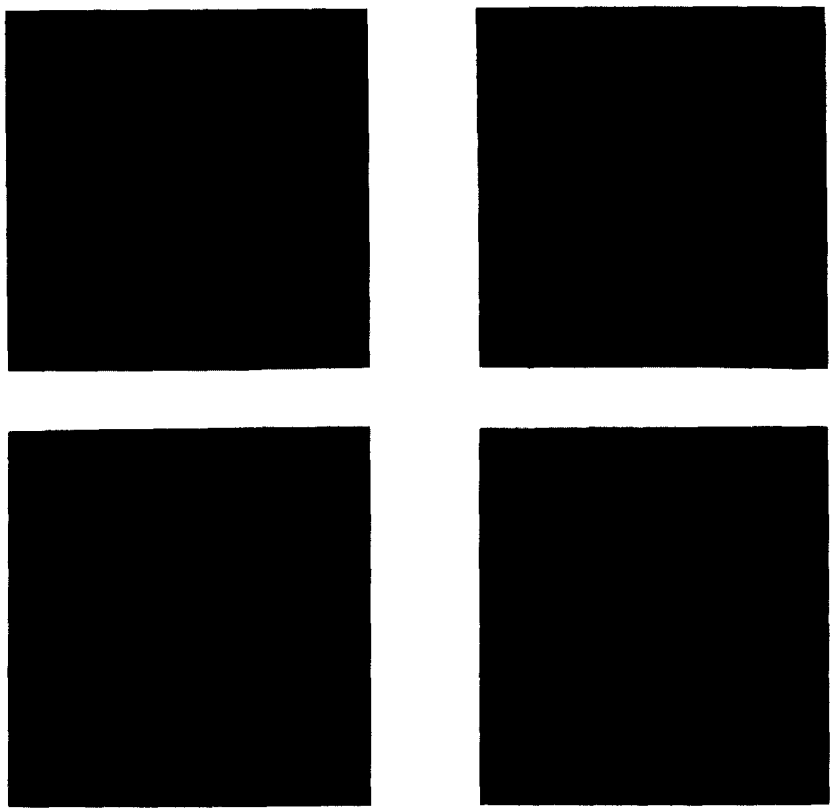

Fig. 5. The error images of "Lenna" of the reconstructed images using 64, 32, 16, and 8 predefined bit planes in top-left, top-right, bottom-left, and bottom-right, respectively, compared with the original image (five times the difference values).

Table 3
The value of n selected automatically by Step 4.5 of ABPS

Image	90% of blocks	ABPAE	m	n
Lenna	14807	0.088	47	64
pepper	14805	0.096	48	64
jet	14773	0.055	44	64
mandrill	14749	0.075	49	64
candy	15151	0.010	15	16
house	14833	0.075	13	16

when the proposed approach is applied to bigger blocks.

5. Conclusions and suggestions for future study

A new approach to improving the conventional BTC has been proposed. The approach is based on the use of a set of predefined bit planes and an adaptive bit plane selection scheme. Compression results with good bit rates and reasonable reconstructed image quality have been obtained. Compared with those methods (Udpikar and Raina, 1987; Zeng and Neuvo, 1993) using vector quantization, our method requires no training phase but still yields good quality in reconstructed images. To lower the MAE values, one of the methods of Halverson et al. (1982), Lema and Mitchell (1984) or Hui (1990) can be applied. Also, the proposed method can be extended for color image compression and at least a 3 bits/pixel data rate can be achieved.

References

Chen, D. and A.C. Bovik (1990). Visual pattern image coding. *IEEE Trans. Commun.* 38, 2137–2146.

Chen, C.T. and T.R. Hsing (1991). Digit coding technique for visual communications. *J. Visual Communication and Image Representation* 2 (1), 1–16.

Delp, E.J. and O.R. Mitchell (1979). Image compression using block truncation coding. *IEEE Trans. Commun.* 27, 1335–1342.

Ferng, S.Y. and W.H. Tsai (1992). Dynamic image compression using moment-preserving technique. *Proc. 1992 National Symposium on Telecommunications,* Taiwan, 190–195.

Forchheimer, R. and T. Kronander (1987). Image coding – from waveforms to animation. *IEEE Trans. Acoust. Speech Signal Process.* 37, 2008–2023.

Halverson, D.R., N.C. Griswold and G.L. Wise (1982). A generalized block truncation coding algorithm for image compression. *IEEE Trans. Acoust. Speech Signal Process.* 32, 664–668.

Hui, L.H. (1990). An adaptive block truncation coding algorithm for image compression. *Proc. IEEE Internat. Conf. Acoust. Speech Signal Process.*, New Mexico, 2233–2236.

Lema, M.D. and O.R. Mitchell (1984). Absolute moment block truncation coding and its application to color images. *IEEE Trans. Commun.* 32, 1148–1157.

Udpikar, V. and J.P. Raina (1987). BTC image coding using vector quantization. *IEEE Trans. Commun.* 35, 352–356.

Zeng, B. and Y. Neuvo (1993). Interpolative BTC image coding with vector quantization. *IEEE Trans. Commun.* 41 (16), 1436–1438.