# Novel image compression method using edge-oriented classifier and novel predictive noiseless coding method

C.-H. Lee
L.-H. Chen

*Indexing terms: Edge-oriented classifier, Image compression, Quadtree decomposition, Vector quantisation*

**Abstract:** A new image compression approach is proposed in which variable block size technique is adopted, using quadtree decomposition, for coding images at low bit rates. In the proposed approach, low-activity regions, which usually occupy large areas in an image, were coded with a larger block size and the block mean is used to represent each pixel in the block. To preserve edge integrity, the classified vector quantisation (CVQ) technique is used to code high-activity regions. A new edge-oriented classifier without employing any thresholds is proposed for edge classification. A novel predictive noiseless coding (NPNC) method which exploits the redundancy between neighbouring blocks is also presented to efficiently code the mean values of low-activity blocks and the addresses of edge blocks. The bit rates required for coding the mean values and addresses can be significantly reduced by the proposed NPNC method. Experimental results show that excellent reconstructed images and higher PSNR were obtained.

## 1 Introduction

Vector quantisation (VQ) is a very efficient approach to low bit-rate image compression [1–3]. In VQ image coding, the image to be encoded is first decomposed into vectors (i.e. blocks) and then sequentially encoded vector by vector. Each vector is compared with the codewords in the codebook to find the nearest one. Compression is achieved by transmitting or storing the address of the nearest codeword rather than the codeword itself. The codebook contains the most representative codewords. Many algorithms for codebook design have been proposed. The generalised Lloyd clustering algorithm developed by Linde, Buzo and Gray [4], which is referred to as the LBG algorithm, is the most popular one used to generate the codebook from a set of training vectors. This algorithm is a codebook improvement procedure that iteratively minimises the overall distortion of representing the training vectors by their corresponding codewords.

Direct application of VQ to image coding has a very simple hardware structure, especially the decoder. Nevertheless, due to high computational complexity, most of the VQ techniques encode small and fixed size blocks throughout the coding process. The compression ratio is thus limited. To solve this problem, variable block size image coding techniques [5–9], using the quadtree decomposition method [10], have been adopted to further reduce the bit rates. The quadtree decomposition technique is attractive for low bit-rate image compression, since it uses fewer bits to code low-activity regions which usually occupy large areas in an image.

Edges are very important to human observer in the perception of image quality. Although edge blocks occupy only a small fraction of the image, the reconstructed image quality will be very annoying if the edge blocks are not coded accurately. Many researchers have exploited the edge information for image/video coding [11, 12]. Conventional VQ suffers from the edge degradation problem because the measure criterion MSE for searching the closest codeword does not preserve the edge information accurately. Therefore, many variant classified VQ (CVQ) techniques [13–21] have been developed to solve the poor edge reproduction problem. A CVQ coder consists of a classifier and separate subcodebooks. Each vector is classified into one of the appropriate classes by a classifier. Separate subcodebooks were designed for each of the classes. Subcodebooks were designed independently by the LBG algorithm [4], using training vectors belonging only to the appropriate class. In CVQ, apart from well edge reproduction, the computational complexity can also be reduced since only one appropriate subcodebook has to be searched for the closest codeword to each input vector. However, the design of an edge-oriented classifier based on edge detection in the spatial domain is not a simple task, it is usually necessary to employ several thresholds in the gradient comparison process. Thus, some CVQs in the transform domain have been proposed [19–21]. A good energy-compaction property can be obtained in the transform domain. Therefore, the classification in the transform domain, in which a few transform coefficients are used as edge orientation and location features, is relatively simpler than that in the spatial domain. However, the classification



*IEE Proceedings* online no. 19971326

Paper first received 21st July 1995 and in final revised form 14th April 1997

C.-H. Lee is with the Department of Information Science, Chinese Culture University, Hwa Kang, Yang Min Shan, Taipei 11114, Taiwan, Republic of China

L.-H. Chen is with the Department of Computer and Information Science, National Chiao Tung University, Hsinchu 30050, Taiwan, Republic of China

complexity in transform domain is higher than that in spatial domain.

To achieve high compression ratios and preserve edge integrity, a new image compression approach will be proposed in this paper. Each vector will be decomposed into one of the variable block sizes depending on its activity. Low-activity blocks will be coded with larger block sizes while high-activity blocks with small block sizes. For a low-activity block, the mean value is used to represent each pixel in the block. A high-activity block will be coded using a CVQ technique to preserve the edge integrity property. A simple edge-oriented classifier in the spatial domain will be proposed for block classification. In addition, a novel predictive noiseless coding method is adopted to further reduce the number of bits required for transmitting or storing the mean values of low-activity blocks and the addresses of edge blocks.

## 2 Proposed technique

In the proposed technique, variable block size technique is adopted for image compression. An image to be encoded is first divided into blocks of size $16 \times 16$ and then each block is further decomposed into blocks of sizes $8 \times 8$ and $4 \times 4$, using a quadtree decomposition technique, depending on the activity of the block. In the quadtree decomposition, each block is repeatedly divided into four equal quadrants if the variance of the block is above a predefined threshold $T_v$. On the other hand, the decomposition process will stop if the variance of the block is below $T_v$, this block is regarded as a low-activity block and all the pixels in the block will be represented by the block mean. The variance of an $l \times l$ block is given by

$$\sigma^2 = \frac{1}{l \times l} \sum_{i=1}^{l} \sum_{j=1}^{l} (x_{ij} - \mu)^2 \qquad (1)$$

where $x_{ij}$ is the grey value of the $(i, j)$th pixel in the block and $\mu$ is the block mean. In this paper, the block mean has been uniformly scalar quantised. If the smallest block size $4 \times 4$ is reached and its variance is above the threshold $T_v$, it is regarded as a high-activity block. An edge-oriented classifier is then provided to classify the high-activity block into one of the four edge classes: horizontal, vertical and two diagonal edges (45° and 135°), depending on its edge orientation. The proposed classifier will be described in the next Section. After block classification, this block will then be encoded by the subcodebook specifically designed for the class it belongs to. These edge codebooks are designed independently by the LBG algorithm [4] using training vectors belonging only to the appropriate class. The search algorithm adopted here is the mean-pyramid method [22] which can greatly reduce the computation time in codebook design as well as encoding phases. The mean-pyramid method uses the mean pyramids of codewords to reject many codewords which are definitely not candidates for the closest codeword to the input vector. This process starts from the top level down to the bottom level to gradually reject more and more codewords. Since a lot of codewords have been rejected, the search process can be sped up greatly.

In the proposed technique, the quadtree structure code is transmitted to the decoder as side information to convey the size and position of the decomposed blocks. Besides the quadtree structure code, the class indexes and codeword addresses of edge blocks and the mean values of low-activity blocks have to be transmitted to reconstruct the image. To further use the redundancy between neighbouring blocks, the mean values of low-activity blocks and the addresses of edge blocks will be efficiently coded using a novel predictive noiseless coding (NPNC)) method. This method will be described in Section 4. Since the pixels in each low-activity block are represented by the block mean, the blocking effect between the boundaries of two blocks occurs unavoidably. To remove the blocking effect, a simple smoothing filter is provided for low-activity blocks and will be described in Section 5. In the next Section, we will first describe the edge-oriented classifier design method in the spatial domain.

## 3 Edge-oriented classifier design

As mentioned previously, the classifier based on edge detection in the spatial domain [13] needs to employ several thresholds in the gradient comparison process. Nevertheless, it is difficult to determine these thresholds, CVQ in the transform domain have been proposed [19–21] to avoid this problem. However, the classification complexity in transform domain is higher than that in spatial domain. In this Section, a simple approach to block classification will be proposed. The classifier determines the class of an edge block, based on the edge orientation, in the spatial domain and without employing any thresholds. All edge blocks are divided into four classes: horizontal, vertical and two diagonal edges (45° and 135°). This is done by convolving the edge block with the following four convolutional masks:

$$\text{mask } 1 = \begin{bmatrix} -2 & -1 & 1 & 2 \\ -2 & -1 & 1 & 2 \\ -2 & -1 & 1 & 2 \\ -2 & -1 & 1 & 2 \end{bmatrix}$$

$$\text{mask } 2 = \begin{bmatrix} 2 & 2 & 2 & 2 \\ 1 & 1 & 1 & 1 \\ -1 & -1 & -1 & -1 \\ -2 & -2 & -2 & -2 \end{bmatrix}$$

$$\text{mask } 3 = \begin{bmatrix} 5 & 2 & 1 & 0 \\ 2 & 1 & 0 & -1 \\ 1 & 0 & -1 & -2 \\ 0 & -1 & -2 & -5 \end{bmatrix}$$

$$\text{mask } 4 = \begin{bmatrix} 0 & 1 & 2 & 5 \\ -1 & 0 & 1 & 2 \\ -2 & -1 & 0 & 1 \\ -5 & -2 & -1 & 0 \end{bmatrix}$$

As we can see from the four masks, these masks can determine the block shape. The absolute values of the four convolution results are defined as $AC_1$, $AC_2$, $AC_3$ and $AC_4$. These four values are used to determine the block class according to the following rules:

*Rule 1* (class 1, vertical class): $AC_1 > AC_2$ and $|AC_1 - AC_2| > |AC_3 - AC_4|$,

*Rule 2* (class 2, horizontal class): $AC_2 > AC_1$ and $|AC_2 - AC_1| > |AC_3 - AC_4|$,

*Rule 3* (class 3, 45° diagonal class): $AC_3 > AC_4$ and $|AC_3 - AC_4| > |AC_1 - AC_2|$,

*Rule 4* (class 4, 135° diagonal class): $AC_4 > AC_3$ and $|AC_4 - AC_3| > |AC_1 - AC_2|$

In such a way, the edge class can be determined without employing any thresholds. Once the edge block has been classified, the closest codeword to the input vector (block) will be found in the corresponding subcodebook. The address of this codeword together with the class number are then transmitted to the decoder. In the next Section, we will provide an efficient way to code the mean values of low-activity blocks and codeword addresses of edge blocks.

## 4 Encoding of mean values and addresses

Huffman coding can be introduced to further reduce the bit rate for coding the mean values of low-activity blocks and addresses of edge blocks if their probability distribution is known. In Huffman coding, the more frequently occurred values can be encoded with shorter codes and less frequently occurred ones with longer codes. However, Huffman coding needs a two-pass process. In the first pass, it computes probabilities of all values. The second pass converts the values into Huffman codes according to their probabilities. This will reduce processing speed since it needs scanning the indexes twice and will delay the transmission of codes until all the codes are generated such that the probability distribution can be calculated. In addition, it is necessary to transmit the probabilities of all values to the decoder to construct the corresponding Huffman tree, this will reduce the compression ratio. Therefore, we will provide a novel predictive noiseless coding (NPNC) method to efficiently code the mean values of low-activity blocks and the codeword addresses of edge blocks. In general, the required number of bits can be reduced by utilising the redundancy between adjacent blocks and a noiseless coding method.

### *4.1 Proposed novel noiseless coding method*

The basic idea of the proposed novel noiseless coding method is very simple. Suppose we have to transmit a $d$-bit binary code for each number. If this number is very small, it will have a long run length of 0s prefix in its binary code representation. Therefore, it suffices to transmit the length $l$ of the 0s prefix and the remaining $(d - l - 1)$ suffix bits. An example would be

binary code: 00000101

length of 0s prefix: 5

suffix: 01

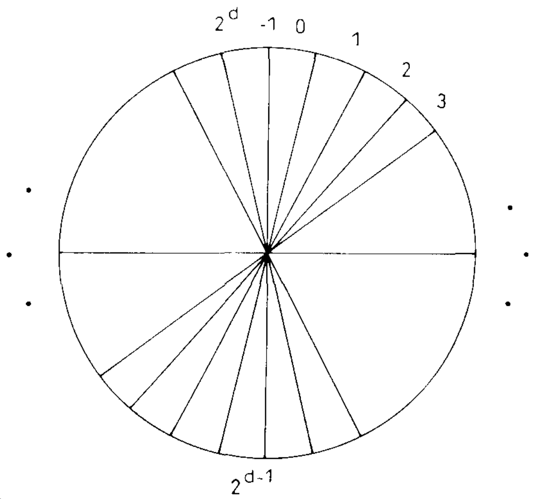


**Fig. 1** *Arrangement of mean values sequentially in the rotating wheel*

The length of 0s prefix can be efficiently coded using a variable length coding method. Let $(m_o, m_p)$ represent the pair of original value and predicted one, each value ranges from 0 to $(2^d - 1)$. Therefore, the difference between these two values will be in the range from $-(2^d - 1)$ to $(2^d - 1)$. To reduce the range of the difference, all values from 0 to $(2^d - 1)$ are arranged sequentially in a rotating wheel as shown in Fig. 1. In such an arrangement, the difference $m_d$ between $m_o$ and $m_p$ will be defined as

$$m_d = \min(2^d - |m_o - m_p|, |m_o - m_p|) \qquad (2)$$

An extra bit is used to indicate the difference between $m_o$ and $m_p$ is on the clockwise direction or on the counterclockwise direction. Then, the difference between these two values can be efficiently coded using the proposed novel noiseless coding method. In the following two subsections, we will describe how to predict the mean values of low-activity blocks and codeword addresses of edge blocks.

### *4.2 Encoding of mean values*

There is a high correlation between the mean values of adjacent blocks. Based on this fact, a predictor is first adopted to estimate the mean values using the pixels of the neighbouring blocks. Nasrabadi and Feng suggested a method to predict the mean value of a $4 \times 4$ block [23] by using the previously encoded vertical and horizontal pixels bordering the current block as shown in Fig. 2. This method is very simple to estimate the mean value of the current block. In the NPNC method, this predictive method was extended to predict the mean values of low-activity blocks of block size $s \times s$ ($s = 4$, 8, or 16). The original scalar quantised mean value is then compared with the predicted mean value. The difference is finally efficiently coded using the noiseless coding method described above.

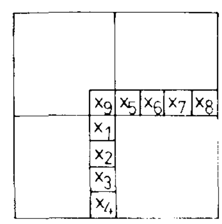


**Fig. 2** *Predicted mean $\mu_p = 1/9 \sum_{i=1}^{9} x_i$*

### *4.3 Encoding of addresses*

The idea used to code the addresses of edge blocks is the same as that used to code the mean values. In general, two vectors with large mean value or variance difference will have a large distance. Based on this idea, the codebook will be reordered according to the difference between the mean value and variance of each codeword and the predicted ones. For each edge blocks, the predicted mean value and variance were estimated by the proposed edge type guided predictor described below. Then, the codebook was reordered in an nondecreasing order of the mean value and variance difference. In such a way, the codewords with similar mean values and variances to those of the edge block can be efficiently coded using the proposed noiseless coding method if the predicted values are close to the original ones. Let $\mu_p$ and $\sigma_p^2$ be the predicted mean value and variance of the current block. In the proposed edge type guided predictor, the block mean value and variance is predicted by several bordering pixels of the neighbouring blocks depending on the edge type of the current edge block. Variant edge types will use variant sets of neighbouring pixels to predict the mean values and variances. Let $\mu_{pv}$, $\mu_{ph}$, $\mu_{pd45}$, and $\mu_{pd135}$ be the predicted mean values for the vertical, horizontal, 45° diagonal and 135° diagonal edge blocks, respectively. Similarly, let $\sigma^2_{pv}$, $\sigma^2_{phv}$, $\sigma^2_{pd45}$, and $\sigma^2_{pd135}$ be the predicted variances.

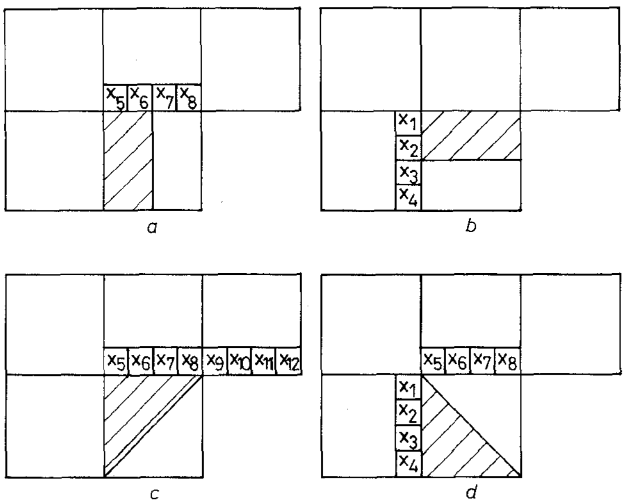

**Fig. 3** *Pixels of neighbouring blocks used to predict the current block mean value*
*a* Vertical edge block
*b* Horizontal edge block
*c* 45° diagonal edge block
*d* 135° diagonal edge block

These values can be formulated as follows:

$$\mu_{pv} = \frac{1}{4}\sum_{i=5}^{8} x_i \qquad \sigma^2_{pv} = \frac{1}{4}\sum_{i=5}^{8}(x_i - \mu_{pv})^2 \qquad (3)$$

$$\mu_{ph} = \frac{1}{4}\sum_{i=1}^{4} x_i \qquad \sigma^2_{ph} = \frac{1}{4}\sum_{i=1}^{4}(x_i - \mu_{ph})^2 \qquad (4)$$

$$\mu_{pd45} = \frac{1}{8}\sum_{i=5}^{12} x_i \qquad \sigma^2_{pd45} = \frac{1}{8}\sum_{i=5}^{12}(x_i - \mu_{pd45})^2 \qquad (5)$$

$$\mu_{pd135} = \frac{1}{8}\sum_{i=1}^{8} x_i \qquad \sigma^2_{pd135} = \frac{1}{8}\sum_{i=1}^{8}(x_i - \mu_{pd135})^2 \qquad (6)$$

where the $x_i$s are the grey values of the previously decoded pixels bordering the current block as shown in Fig. 3. The mean difference between the predicted mean value $\mu_p$ and the mean value $\mu_i$ of codeword $y_i$ is defined as

$$\mu_d(i) = \mu_p - \mu_i$$

and the variance difference is defined as

$$\sigma_d(i) = \sigma_p - \sigma_i$$

Further, the predicted difference is defined as

$$p_d(i) = \mu_d(i) + \sigma_d(i)$$

For each codeword $y_i$, the mean value difference $\mu_d(i)$ and variance difference $\sigma_d(i)$ between predicted values and actual values are calculated. The predicted difference $p_d(i)$ is then calculated and the codebook is reordered in an non-decreasing order of $p_d(i)$. The address of the nearest codeword to the edge block is then found and coded by the proposed noiseless coding method.

## 5 Smoothing algorithm

Since the pixels in each low-activity block are represented by the block mean, the blocking effect between the boundaries of two blocks occurs unavoidably. To remove the blocking effect, a simple smoothing filter is provided. Since only low-activity blocks are smoothed, the high-activity blocks will not be blurred and edges will be preserved. The smoothing filter uses three various mask sizes (3 × 3, 5 × 5, and 9 × 9) for the post-processing of three different block sizes (4 × 4, 8 × 8, and 16 × 16). It adopts a simple average operation over the pixels in the area of the mask. The response of the smoothing operation is given by

$$x' = \frac{1}{W}\sum_{i=1}^{l}\sum_{j=1}^{l}(w_{ij} \times x_{ij}) \qquad (7)$$

where $x'$ represents the smoothed grey value of the present pixel at which the centre of the mask is located, $x_{ij}$ denotes the grey level of $(i, j)$th pixel in the mask, $w_{ij}$ denotes the weight of $(i, j)$th pixel defined as follows:

$$w_{ij} = \begin{cases} 1 & \text{if } x_{ij} \in \text{low-activity block} \\ 0 & \text{if } x_{ij} \in \text{high-activity block} \end{cases}$$

and

$$W = \sum_{i=1}^{l}\sum_{j=1}^{l} w_{ij}$$

With the above operation, the grey level of each pixel in low-activity blocks only changes slightly and is not affected by any edge pixels.

## 6 Experimental results

The performance of the proposed method has been evaluated on several images. A training set of twelve 512 × 512 monochrome images with 256 grey levels were used to generate edge codebooks. In codebook design phase, each image is divided into 4 × 4 blocks. Each block is then checked to see if it is a high-activity block. If a block is a high-activity block, it is further

**Table 1: Comparison of classifiers**

| Codebook size | Lena | | | | Jet | | | |
|---|---|---|---|---|---|---|---|---|
| | Proposed classifier | | BTC classifier [17] | | Proposed classifier | | BTC classifier [17] | |
| | bpp | PSNR | bpp | PSNR | bpp | PSNR | bpp | PSNR |
| 64 | 0.201 | 30.70 | 0.216 | 30.47 | 0.207 | 29.56 | 0.219 | 29.27 |
| 128 | 0.212 | 31.20 | 0.231 | 31.08 | 0.220 | 30.23 | 0.237 | 29.98 |
| 256 | 0.228 | 31.69 | 0.247 | 31.59 | 0.237 | 30.80 | 0.254 | 30.57 |
| 512 | 0.244 | 32.12 | 0.264 | 31.98 | 0.254 | 31.12 | 0.271 | 31.07 |
| 1024 | 0.261 | 32.45 | 0.282 | 32.35 | 0.273 | 31.54 | 0.291 | 31.43 |

**Fig. 4** *Reconstructed images*
*a* Original image Lena
*b* JPEG, 0.243 bpp, PSNR = 30.94
*c* VBSVQ, 0249, bpp, PSNR = 31.03, codebook size 512
*d* Quadtree map
*e* Proposed method, 0.244 bpp, PSNR = 32.12, codebook size 512

classified into one of the four edge types using the proposed classifier. Each edge codebook is designed independently by the LBG algorithm using the blocks which belongs to the edge class as the training set. Two $512 \times 512$ images (Lena and Jet) shown in Fig. 4*a* and Fig. 5*a* were used in the simulations. To show the robustness of the proposed method, the two images are outside the training set.

Table 1 compares the proposed classifier with the BTC classifier proposed in [17] in terms of the bit rates (bpp) as well as PSNR for the two test images. From this Table, we can see the proposed classifier can

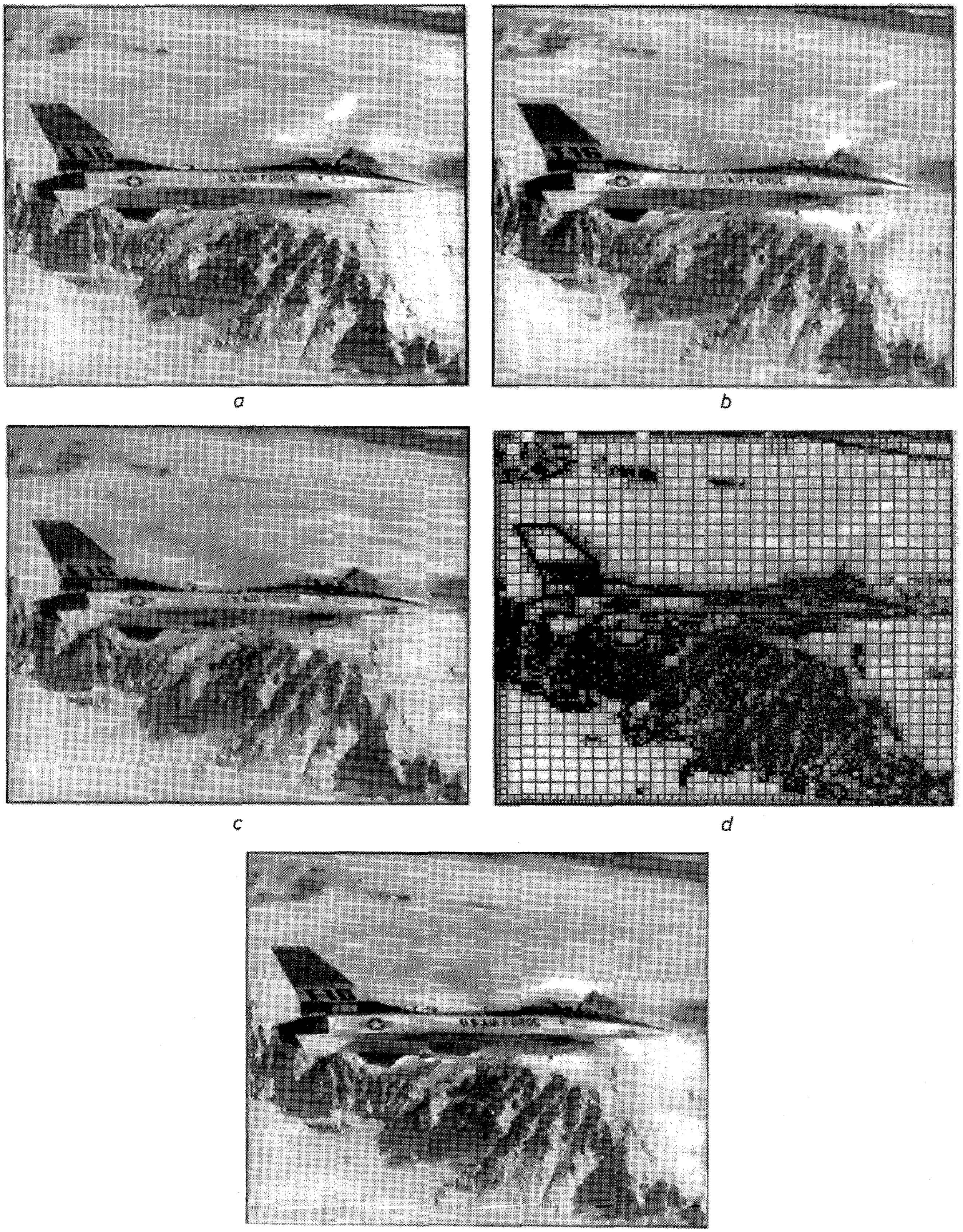


**Fig. 5** *Reconstructed images*
*a* Original image Jet
*b* JPEG, 0.267 bpp, PSNR = 30.24
*c* VBSVQ, 0.261 bpp, PSNR = 30.30, codebook size 512
*d* Quadtree map
*e* Proposed method, 0.254 bpp, PSNR = 31.12, codebook size 512

achieve lower bit rates and higher PSNR than the BTC classifier proposed in [17].

Table 2 gives the composition of output bits for the two test images, including quadtree, mean value, classification, and address bits.

The number of bits required to transmit or store the mean values of all blocks using the proposed NPNC method is compared with that using fixed length coding method, and Huffman coding method. Table 3 shows the comparison results with various codebook sizes as well as various test images. From this Table, we can see that the bit rates are significantly reduced by the proposed NPNC method.

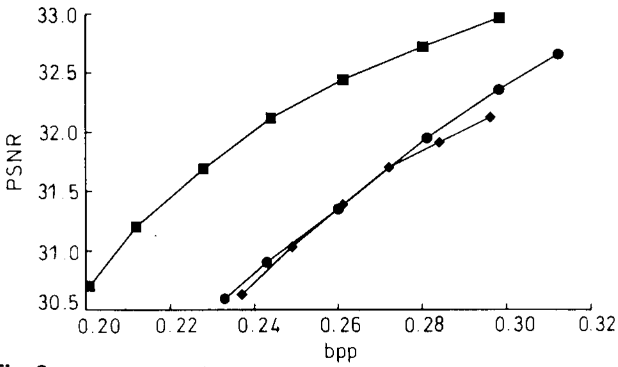


**Fig. 6** *Comparison of performance for JPEG, VBSVQ, and proposed image coding method in terms of PSNR against bpp for encoding the image Lena*
● JPEG
■ proposed method
◆ VBSVQ

The number of bits required to transmit or store the addresses of all edge blocks using the proposed NPNC method is compared with that using fixed length coding method, Huffman coding method, and average method. The average method uses all of the neighbouring pixels as shown in Fig. 2 to predict the mean and variance values of an edge block. Table 4 shows the comparison results with various codebook sizes as well as various test images. From this Table, we can see that the required bit rates can be reduced by the proposed NPNC method. This Table also shows that the proposed edge type guided predictor outperforms that proposed in [23]. It is worth noting that the occurrence probability of each mean value that needs to be transmitted to the decoder is not included in Tables 3 and 4.

JPEG, variable block size VQ (VBSVQ) [8] are used to compare with the proposed image coding method. Fig. 6 shows the relationship of the bit rates in terms of bits per pixel (bpp) and peak signal-to-noise ratios (PSNR) for the encoded image Lena with these three methods. Define the mean square error (MSE) as

$$\mathrm{MSE} = \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}(x_{ij} - y_{ij})^2$$

where $x_{ij}$ is the original pixel grey level and $y_{ij}$ is the reconstructed pixel grey level, MN is the total number

**Table 2: Composition of output bits**

| Codebook size | Test image | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Lena | | | | | Jet | | | | |
| | Quadtree | Mean | Class | Address | Total | Quadtree | Mean | Class | Address | Total |
| 64 | 10684 | 12911 | 9030 | 20160 | 52785 | 10284 | 11804 | 9512 | 22589 | 54189 |
| 128 | 10684 | 12721 | 0930 | 23136 | 55571 | 10284 | 11695 | 9512 | 26117 | 57608 |
| 256 | 10684 | 12699 | 9030 | 27253 | 59666 | 10284 | 11562 | 9512 | 30648 | 62006 |
| 512 | 10684 | 12675 | 9030 | 31565 | 63936 | 10284 | 11483 | 9512 | 35246 | 66525 |
| 1024 | 10684 | 12568 | 9030 | 36157 | 68437 | 10284 | 11439 | 9512 | 40252 | 71487 |

**Table 3: Comparison of the number of bits required to code the mean values of all blocks**

| Codebook size | Test image | | | | | |
|---|---|---|---|---|---|---|
| | Lena | | | Jet | | |
| | Fixed | Huffman | NPNC | Fixed | Huffman | NPNC |
| 64 | 22524 | 11249 | 12911 | 19278 | 10517 | 11804 |
| 128 | 22524 | 11056 | 12721 | 19278 | 10383 | 11695 |
| 256 | 22524 | 10986 | 12699 | 19278 | 10257 | 11562 |
| 512 | 22524 | 10965 | 12657 | 19278 | 10160 | 11483 |
| 1024 | 22524 | 10838 | 12568 | 19278 | 10125 | 11439 |

**Table 4: Comparison of the number of bits required to code the addresses of all edge blocks**

| Codebook size | Test image | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | Lena | | | | Jet | | | |
| | Fixed | Huffman | Average | NPNC | Fixed | Huffman | Average | NPNC |
| 64 | 27090 | 18092 | 22358 | 20160 | 28536 | 20816 | 23783 | 22589 |
| 128 | 31605 | 21791 | 25943 | 23136 | 33292 | 24809 | 27572 | 26117 |
| 256 | 36120 | 25558 | 30241 | 27253 | 38048 | 28834 | 32207 | 30684 |
| 512 | 40635 | 29493 | 34655 | 31565 | 42804 | 32895 | 37049 | 35246 |
| 1024 | 45150 | 33061 | 39270 | 36157 | 47560 | 36667 | 42140 | 40252 |

of pixels in the image. The PSNR is then defined as

$$\mathrm{PSNR} = 10\log_{10}\left(\frac{255^2}{\mathrm{MSE}}\right)\ \mathrm{dB}$$

The reconstructed images are shown in Figs. 4 and 5. From these Figures, we can see that the reconstructed images of the proposed technique will have higher PSNR and better visual quality.

## 7 Conclusions

In this paper, a new image compression approach will be proposed. Variable block size technique is adopted, using quadtree decomposition, for coding images at low bit rates. In the proposed approach, low-activity regions which usually occupy large areas in an image were coded with a larger block size and the block mean is used to represent each pixel in the block. To preserve edge integrity, classified vector quantisation (CVQ) technique is used to code high-activity regions. A new edge-oriented classifier without employing any thresholds is proposed for edge classification. Simulation results show that the proposed classifier outperforms the BTC classifier proposed in [17]. In addition, a novel predictive noiseless coding (NPNC) method which exploits the redundancy between neighbouring blocks is also presented to efficiently code the mean values of low-activity blocks and the addresses of edge blocks. The bit rates required for coding the mean values and addresses can be significantly reduced by the proposed NPNC method. Experiment results show that excellent reconstructed images and higher PSNR have been obtained at low bit rates.

## 8 Acknowledgment

This research was supported in part by the National Science Council of R. O. C. under contract NSC-84-0404-E008-038.

## 9 References

1 NASRABADI, N.M., and KING, R.A.: 'Image coding using vector quantization: a review', *IEEE Trans.*, 1988, **COM–36**, (8), pp. 957–971
2 GERSHO, A., and GRAY, R.M.: 'Vector quantization and signal compression' (Kluwer, Boston, 1992)
3 GRAY, R.M.: 'Vector quantization', *IEEE ASSP Mag.*, 1984, **1**, pp. 4–29
4 LINDE, Y., BUZO, A., and GRAY, R.M.: 'An algorithm for vector quantizer design', *IEEE Trans.*, 1980, **COM–28**, (1), pp. 84–95
5 NASRABADI, N.M., LIN, S.E., and FENG, Y.: 'Interframe hierarchical vector quantization', *Opt. Eng.*, 1989, **28**, (7), pp. 717–725
6 STROBACH, P.: 'Quadtree-structured recursive plane decomposition coding of images', *IEEE Trans.*, 1991, **39**, (6), pp. 1380–1397
7 BERGER, L., MARIOT, J.P., and LAUNAY, C.: 'A new formulation for fast image coding using quadtree representation', *Patt. Recog. Lett.*, 1992, **13**, pp. 425–432
8 VAISEY, J., and GERSHO, A.: 'Image compression with variable block size segmentation', *IEEE Trans. Signal Process.*, 1992, **40**, (8), pp. 2040–2060
9 LEE, M.H., and GREBBIN, G.: 'Classified vector quantization with variable blocksize DCT models', *IEE Proc., Vis. Image Signal Process.*, 1994, **141**, (1), pp. 39–48
10 SAMET, H.: 'The quad-tree and related hierarchical data structures', *ACM Comput. Surv.*, 1984, **16**, (2), pp. 188–260
11 ERYURTLU, F., KONDOZ, A.M., and EVANS, B.G.: 'A subband image coding algorithm using vector quantization'. Proceedings of the IEEE international symposium on *Multimedia technologies*, Southampton, UK, 1993
12 MOHSENIAN, N., and NASRABADI, N.M.: 'Edge-based subband VQ techniques for images and video', *IEEE Trans. Circuits Syst. Video Technol.*, 1994, **4**, (1), pp. 53–67
13 RAMAMURTHI, B., and GERSHO, A.: 'Classified vector quantization of images', *IEEE Trans.*, 1986, **COM–34**, (11), pp. 1105–1115
14 KUBRICK, A., and ELLIS, T.: 'Classified vector quantization of images: codebook design algorithms', *IEE Proc. I, Commun. Speech Vis.*, 1990, **137**, (6), pp. 379–386
15 PO, L.M., and CHAN, C.K.: 'Directionally classified image vector quantization using Walsh Hadamard subspace distortion', *Electron. Lett.*, 1991, **27**, (21), pp. 1964–1967
16 NGAN, K.N., and KOH, H.C.: 'Predictive classified vector quantization', *IEEE Trans. Image Process.*, 1992, **1**, (3), pp. 269–280
17 LO, K.L., and CHAM, W.K.: 'New predictive classified vector quantization scheme for image compression', *Electron. Lett.*, 1994, **30**, (16), pp. 1280–1282
18 ABBAS, H.M., and FAHMY, M.M.: 'Classified vector quantization using variance classifier and maximum likelihood clustering', *Pattern Recognit. Lett.*, 1994, **15**, pp. 49–55
19 KIM, D.S., and LEE, S.U.: 'Image vector quantizer based on a classification in the DCT domain', *IEEE Trans. Commun.*, 1991, **39**, (4), pp. 549–556
20 KIM, J.W., and LEE, S.U.: 'A transform domain classified vector quantizer for image coding', *IEEE Trans. Circuits Syst. Video Technol.*, 1992, **CS–2**, (1), pp. 3–14
21 HWANG, C., VENKATRAMAN, S., and RAO, K.R.: 'Human visual system weighted progressive image transmission using lapped orthogonal transform/classified vector quantization', *Opt. Eng.*, 1993, **32**, (7), pp. 1524–1530
22 LEE, C.H., and CHEN, L.H.: 'A fast search algorithm for vector quantization using mean pyramids of codewords', *IEEE Trans. Commun.*, 1995, **43**, (2/3/4), pp. 1697–1702
23 NASRABADI, N.M., and FENG, Y.: 'Image compression using address vector quantization', *IEEE Trans. Commun.*, 1990, **38**, (12), pp. 2166-2173