A Very Low Bit Rate Image Compressor Using Transformed Classified Vector Quantization

Hsien-Wen Tseng
Department of Information Management
Chaoyang University of Technology
168, Jifong East Road, Wufong Township, Taichung County 41349, Taiwan, R.O.C.

Chin-Chen Chang
Department of Computer Science and Information Engineering
National Chung Cheng University
Chiayi, Taiwan 621, R.O.C.
E-mail: ccc@cs.ccu.edu.tw

Keywords: Image compression, JPEG, DCT, DPCM, Classified vector quantization (CVQ), Quadtree

Received: December 8, 2004

In this paper, a very low bit rate image compression scheme is proposed. This scheme is a hybrid method that combines the good energy-compaction property of DCT with the high compression ratio of VQ-based coding. We start by transforming image block from spatial domain to frequency domain using DCT. In order to increase the compression ratio while preserving decent reconstructed image quality, DC coefficients are coded by DPCM and only the most important AC coefficients are coded using classified vector quantization (CVQ). The most important AC coefficients are selected to train the codebook according to the energy packing region of different block classes. Also, this scheme can provide different compression ratios like JPEG does with the same codebooks. The experimental results show that the proposed scheme performs much better than JPEG at low bit rate.

Povzetek: Članek predstavlja kompresijo slik s pomočjo kvantizacije vektorjev.

1 Introduction

As a result of bandwidth and storage limitations, image compression techniques are widely used in data transmission and data storage. In a congested network like the Internet or low bandwidth communication for wireless transmission, image compression at a low bit rate is necessary. One of the most popular and efficient methods for image compression is JPEG [1]. JPEG is an international standard for lossy and lossless compression of images. In lossy JPEG, an N×N image is divided into 8×8 blocks and then a discrete cosine transform (DCT) is performed on each block. The transformed coefficients are quantized and coded using a combination of run-length and Huffman coding. The quality factor (Q) is used to tradeoff image quality with compression ratio. When a high compression ratio is desired, images are highly degraded because of the significant block artifacts. Fig. 1 shows an example of low bit rate image (0.15bpp) compressed by JPEG. It is unacceptable in certain applications. Although JPEG perform poorly in low bit rate, the DCT is accepted as the best approximation of Karhunen-Loeve transform (KLT) [2], the optimum block transform in terms of energy packing efficiency.

Another well-known image compression method is vector quantization (VQ) [3]. It provides many attractive features for image compression. One important feature of VQ is the high compression ratio. The fast decompression by table lookup is another important feature. But edge degradation has been revealed as a serious problem in VQ. For this reason, a classified VQ (CVQ) has been proposed [4]. The CVQ consists of a classifier and separate codebooks for each class. The possible classes are typically: shade, horizontal edge, vertical edge and diagonal edge classes. The design of classifier and codebooks is very important to the CVQ. But it is not a simple task in the spatial domain.

Figure 1: JPEG compressed image
(Bit rate = 0.15bpp, PSNR = 25.91dB)

Several papers [5-7] have proposed CVQ in the transform domain, which are simpler and outperform CVQ in the spatial domain. But they are not as good as JPEG. Meanwhile, in order to minimize the distortion, they have to adopt small block size (4×4) or use the DCT block partition schemes to partition the big block (8×8) into the zonal bands. In this paper, we propose a new scheme to raise the compression ratio by choosing the most important coefficients in the different classified classes, so that reduce the complexity of training codebook and decrease the codebook size. The new scheme is based on the transform domain CVQ, for this reason, it will be called transformed classified vector quantization (TCVQ). Our study will focus on the classifier, the codebook design, and the selection of code vector, as they are very important for reconstructed image quality to preserve the significant features in the image. In addition, we will make the TCVQ provides different compression ratios. Just like the JPEG, one parameter Q (quality factor) specified by the user is used to tradeoff image quality with compression rate.

This paper is organized as follows. Section 2 describes the fundamental concept of DCT and CVQ. The proposed TCVQ scheme is discussed in Section 3. Finally, experimental results are given in Section 4, and conclusions are presented in Section 5.

2 DCT and CVQ

2.1 DCT

There are many transform schemes that transform a set of pixels from their normal spatial representation to a frequency domain. Some of these include Karhunen-Loeve Transform (KLT), Discrete Fourier Transform (DFT), and Discrete Cosine Transform (DCT). Among these, DCT is widely used for image compression because it provides a good energy-compaction property. The DCT transformation generally results in the signal energy being distributed among a small set of transformed coefficients only. In general, image compression method like JPEG uses quantization to discard the transformed coefficients with the least information. Quantization usually results in some distortion of original image. However, this distortion can be controlled within an acceptable level.

Given an image consisting of the N×N pixels *f(x,y)*, its two-dimensional DCT produces the N×N array of numbers *F(i,j)* given by

$$F(i,j)=\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}4f(x,y)\cos\left(\frac{(2x+1)i\pi}{2N}\right)\cos\left(\frac{(2y+1)j\pi}{2N}\right),$$

where 0 <= *i*, *j* <= N-1. The JPEG standard divides the entire image into 8×8 blocks and then performs a two-dimensional DCT on each block. In the DCT matrix, the coefficient *F*(0,0) is called the "DC coefficient," and the remaining 63 coefficients are called the "AC coefficients." The image information (energy) is usually concentrated in the low frequency region (the top left corner). The high frequency region is located in the bottom right corner. Here are some examples of the results of applying the DCT to different class blocks [8].

- A shade block produces a DCT matrix with energy concentrated in the top left corner.
- A horizontal edge block produces a DCT matrix with energy concentrated in the left side.
- A vertical edge block produces a DCT matrix with energy concentrated in the upper side.
- A diagonal edge block produces a DCT matrix with energy concentrated in the diagonal region.

2.2 CVQ

Edge is a very significant feature perceptually in an image. A truthful coding that preserves the edge information is of importance. Unfortunately, edge degradation has been revealed as a serious problem in VQ. This is because the codebook design is based on conventional distortion measure such as the mean square error (MSE). The MSE is the square of the Euclidean distance between input vector and output vector. The best-matched code vector is chosen using a minimum MSE value. However, in the edge blocks, the pixel values vary quickly from pixel to pixel, the MSE hardly possesses any edge preserving property. Edge degradation then happens.

In order to alleviate the edge degradation in conventional VQ, a classified VQ (CVQ) was introduced by Ramamurthi and Gersho [4]. In CVQ, the image is divided into blocks, and the blocks are classified into various classes. Then the blocks belonging to a class are coded only with code vectors belonging to the same class in order to preserve the perceptual feature associated with each class. Various codebooks are designed for each class, so as to preserve the perceptual feature associated with each class. The block diagram of CVQ is shown in Fig. 2.

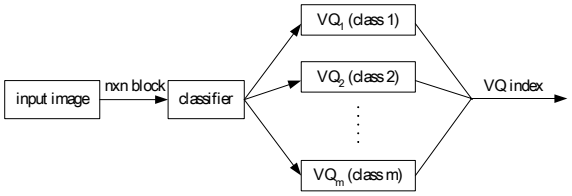


Figure 2: The block diagram of CVQ

Needless speaking, the classifier which classifies the image blocks, is important to the CVQ. However, the design of classifier based on edge detection in the spatial domain is not a simple task, because it usually is interfered by the image background. But by taking advantage of the good energy-compaction property of the DCT, we can simplify the block classification problem.

In [9], Kim et al. proposed an image coding scheme which employs both the DCT and the CVQ. An input image is divided into 4×4 blocks, and the blocks are transformed using the DCT and are classified into four edge-oriented classes. The classification employs one appropriate threshold T and two most important AC coefficients F(0,1) and F(1,0). If F(0,1) < T and F(1,0) < T, then the block is classified into the shade block. Otherwise, if F(0,1) ≥ T, F(1,0) ≥ T, and

max(F(0,1),F(1,0))/min(F(0,1),F(1,0)) < 2, then the block is classified into the diagonal edge block. In case the above two conditions are not met and F(1,0) ≥ F(0,1), then the block is classified into the horizontal edge block. Otherwise, the block is classified into the vertical edge block. Besides, four weighted tree-structured codebooks according to the four classes are designed by using the binary tree-structured self-organizing feature maps (BTSOFM) [10]. The BTSOFM is a modified tree search vector quantization (TSVQ) [11]. Unlike the conventional TSVQ, which has the 16-dimensional intermediate codevectors throughout the tree, the dimension of the intermediate codevectors in Kim et al.'s scheme varies from 1 to 16 as it goes down the tree. The dimensional reduction results from the characteristics of the DCT coefficients and the edge-oriented classification. By applying only most important DCT coefficients to be coded to the input of the tree, they can yield significant reduction of computation. Hence a high speed vector quantization with good reconstructed image quality is proposed.

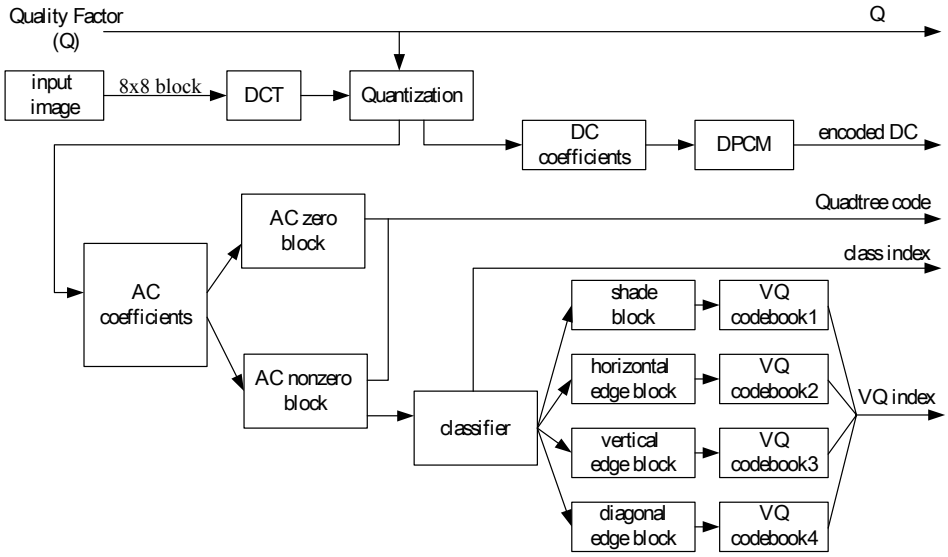


Figure 3: The block diagram of TCVQ

3 TCVQ

Kim et al. [9] exploit the DCT, the CVQ, and the TSVQ for fast encoding, whereas we will utilize the DCT and the CVQ for low bit rate coding. The proposed transformed classified vector quantization (TCVQ) is a hybrid image compression method that employs the energy-compaction property of DCT and the high compression ratio of CVQ. The quality factor (Q) is specified by user to tradeoff image quality with compression ratio. In order to achieve high compression ratio, the TCVQ uses two stages of compression. In the first stage, image is transformed from the spatial domain to the frequency domain and then quantized. These quantized nonzero AC coefficients are compressed again using the CVQ in the second stage.

Fig. 3 shows the block diagram of the TCVQ. The main TCVQ compression steps are outlined below, and some critical steps are then described in detail later.

1. The image is divided into 8×8 blocks. Each block is separately transformed using DCT. The transformed coefficients are then quantized to eliminate the high-frequency components. This elimination will not lead to significant degradation on image because the high energy components will be concentrated in the low frequency region. It also reduces the dimension of transformed coefficients and helps us to lower the vector size of codebook later. This quantization is done in the same way JPEG is, so we use the default quantization table of JPEG. Meanwhile, the quality factor (Q) is specified by the user, which is used to tradeoff image quality with compression ratio.
2. The DC coefficient is proportional to the average of the block entries. Experience shows that in a continuous-tone image, the DC coefficients of adjacent blocks are close, so we encode the DC coefficients using DPCM. It outputs the first coefficient, followed by differences of the DC coefficients of consecutive blocks.
3. As for the AC coefficients, most AC coefficients are zeros after being quantized. Here, we define "AC zero block" as an 8×8 block, which only contains zeros in AC coefficients. The residual blocks are defined as "AC nonzero blocks." These "AC zero blocks" are not necessary to encode, but remember its position in image only. If we label the "AC zero block" with "0" and the "AC nonzero block" with "1", we can get a bitmap table to represent the image. This bitmap table is then encoded using quadtree compression and sent to the decoder as side information for the reconstruction of image.
4. The "AC nonzero blocks" are encoded using CVQ.

Each "AC nonzero block" is classified into one of four classes: shade block, horizontal edge block, vertical edge block, and diagonal edge block. Here, we use a simple classification algorithm, which is modified from the algorithm proposed by Kim et al [9]. Our algorithm employs eight AC coefficients of the input block as edge oriented features. We will describe it in detail later. Now the codebooks for these "AC nonzero blocks" are separately designed according to various classes. We employ four different codebooks, each designed for one of four classes. The four codebooks have different vector and codebook sizes according to the properties of different classes. We will also discuss the design of codebook in detail later.

5. The last step integrates all the information (encoded DC, Quadtree code, VQ index, and class index) generated above, and outputs the results.

3.1 The Classification Algorithm

We classify the "AC nonzero blocks" into one of four classes: shade block, horizontal edge block, vertical edge block, and diagonal edge block. The horizontal edge block makes the energy concentrated in the left region of the transformed matrix. The vertical edge block makes the energy concentrated in the upper region of the transformed matrix. In the case of the diagonal block, the energy is concentrated in the diagonal region of the transformed matrix. If all of the AC coefficients are relatively small, the block must be a shade block. Fig. 4 shows the relationship between the edge orientations and the two values of V and H, where V is the maximum value of absolute values of C_1, C_5, C_6, C_7, and H is the maximum value of absolute values of C_2, C_3, C_8, C_9.

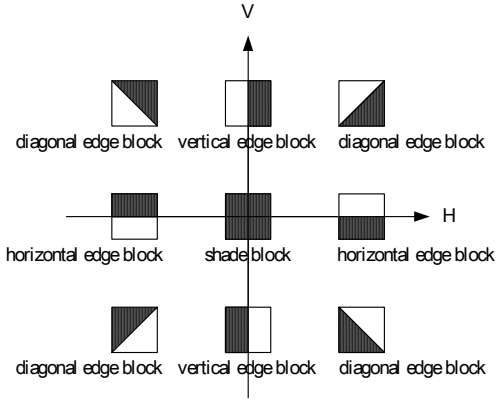
The block classification algorithm is shown as follows.

1. Compute V = max($|C_1|$, $|C_5|$, $|C_6|$, $|C_7|$) and H = max($|C_2|$, $|C_3|$, $|C_8|$, $|C_9|$).
2. Classify the block B into one of the four classes.
   If (V < threshold Γ) and (H < threshold Γ)
   B = shade block;
   else if (V >= threshold Γ) and (H >= threshold Γ) and (max(V, H)/min(V,H) <2)
   B = diagonal block;
   else if (H >= V)
   B = horizontal edge block;
   else
   B = vertical edge block;

C_0	**C_1**	**C_5**	**C_6**	C_{14}	C_{15}	C_{27}	C_{28}
C_2	C_4	**C_7**	C_{13}	C_{16}	C_{26}	C_{29}	C_{42}
C_3	**C_8**	C_{12}	C_{17}	C_{25}	C_{30}	C_{41}	C_{43}
C_9	C_{11}	C_{18}	C_{24}	C_{31}	C_{40}	C_{44}	C_{53}
C_{10}	C_{19}	C_{23}	C_{32}	C_{39}	C_{45}	C_{52}	C_{54}
C_{20}	C_{22}	C_{33}	C_{38}	C_{46}	C_{51}	C_{55}	C_{60}
C_{21}	C_{34}	C_{37}	C_{47}	C_{50}	C_{56}	C_{59}	C_{61}
C_{35}	C_{36}	C_{48}	C_{49}	C_{57}	C_{58}	C_{62}	C_{63}

H = max($|C_2|$, $|C_3|$, $|C_8|$, $|C_9|$)
V = max($|C_1|$, $|C_5|$, $|C_6|$, $|C_7|$)

(a) Calculation of H and V



(b) The relationship of edge orientations

Figure 4: DCT coefficients and block classification

3.2 The Design of Codebooks

One of the most important tasks in a CVQ application is the design of codebook. In this paper, we will design four codebooks according to the four classes. All the codebooks are designed in the frequency domain. The image is divided into 8×8 blocks and each block is separately transformed using DCT. Then each block has 63 AC coefficients. However, it is too large and more time demanding if we take all the 63 AC coefficients into the code vector of training set. Hence, the transformed coefficients will be truncated to eliminate the least information components. It also reduces the dimension of code vector in the codebook.

- Shade block codebook
  The shade block is a smooth block with no edge passing through it. The energy is concentrated in the low frequency region of the transformed matrix, which is located in the upper-left corner of DCT matrix. Hence, the code vector for the shade block codebook is (C_1, C_2, C_3, C_4, C_5, C_6, C_7, C_8, C_9) (see Fig. 5). Then the 9-dimensional VQ codebook is used to vector quantize the shade blocks.
- Horizontal edge block codebook
  The horizontal block has the transformed matrix with a left nonzero region. The energy is concentrated in the left region of the transformed matrix. Hence, the code vector for the horizontal block codebook is (C_1, C_2, C_3, C_4, C_5, C_7, C_8, C_9, C_{10}, C_{11}, C_{19}) (see Fig. 6). Then the 11-dimensional VQ codebook is used to vector quantize the horizontal edge blocks.
- Vertical edge block codebook
  The vertical block has the transformed matrix with an upper nonzero region. The energy is concentrated in the upper region of the transformed matrix. Hence, the code vector for the vertical block codebook is (C_1, C_2, C_3, C_4, C_5, C_6, C_7, C_8, C_{13}, C_{14}, C_{16}) (see Fig. 7). Then the 11-dimensional VQ codebook is used to vector quantize the vertical edge blocks.
- Diagonal edge block codebook

The diagonal edge block is the most complicated block. So we need more dimensional code vector to represent the block. The energy is concentrated in the diagonal region of the transformed matrix. Hence, the code vector for the diagonal block codebook is (C_1, C_2, C_3, C_4, C_5, C_7, C_8, C_{11}, C_{12}, C_{13}, C_{17}, C_{18}, C_{23}, C_{24}, C_{25}) (see Fig. 8). Then the 15-dimensional VQ codebook is used to vector quantize the diagonal edge blocks.

C_0	**C_1**	**C_5**	**C_6**	C_{14}	C_{15}	C_{27}	C_{28}
C_2	**C_4**	**C_7**	C_{13}	C_{16}	C_{26}	C_{29}	C_{42}
C_3	**C_8**	C_{12}	C_{17}	C_{25}	C_{30}	C_{41}	C_{43}
C_9	C_{11}	C_{18}	C_{24}	C_{31}	C_{40}	C_{44}	C_{53}
C_{10}	C_{19}	C_{23}	C_{32}	C_{39}	C_{45}	C_{52}	C_{54}
C_{20}	C_{22}	C_{33}	C_{38}	C_{46}	C_{51}	C_{55}	C_{60}
C_{21}	C_{34}	C_{37}	C_{47}	C_{50}	C_{56}	C_{59}	C_{61}
C_{35}	C_{36}	C_{48}	C_{49}	C_{57}	C_{58}	C_{62}	C_{63}

Figure 5: The code vector for shade block codebook

C_0	**C_1**	**C_5**	C_6	C_{14}	C_{15}	C_{27}	C_{28}
C_2	**C_4**	**C_7**	C_{13}	C_{16}	C_{26}	C_{29}	C_{42}
C_3	**C_8**	C_{12}	C_{17}	C_{25}	C_{30}	C_{41}	C_{43}
C_9	**C_{11}**	C_{18}	C_{24}	C_{31}	C_{40}	C_{44}	C_{53}
C_{10}	**C_{19}**	C_{23}	C_{32}	C_{39}	C_{45}	C_{52}	C_{54}
C_{20}	C_{22}	C_{33}	C_{38}	C_{46}	C_{51}	C_{55}	C_{60}
C_{21}	C_{34}	C_{37}	C_{47}	C_{50}	C_{56}	C_{59}	C_{61}
C_{35}	C_{36}	C_{48}	C_{49}	C_{57}	C_{58}	C_{62}	C_{63}

Figure 6: The code vector for horizontal edge block codebook

C_0	**C_1**	**C_5**	**C_6**	**C_{14}**	C_{15}	C_{27}	C_{28}
C_2	**C_4**	**C_7**	**C_{13}**	**C_{16}**	C_{26}	C_{29}	C_{42}
C_3	**C_8**	C_{12}	C_{17}	C_{25}	C_{30}	C_{41}	C_{43}
C_9	C_{11}	C_{18}	C_{24}	C_{31}	C_{40}	C_{44}	C_{53}
C_{10}	C_{19}	C_{23}	C_{32}	C_{39}	C_{45}	C_{52}	C_{54}
C_{20}	C_{22}	C_{33}	C_{38}	C_{46}	C_{51}	C_{55}	C_{60}
C_{21}	C_{34}	C_{37}	C_{47}	C_{50}	C_{56}	C_{59}	C_{61}
C_{35}	C_{36}	C_{48}	C_{49}	C_{57}	C_{58}	C_{62}	C_{63}

Figure 7: The code vector for vertical edge block codebook

C_0	**C_1**	**C_5**	C_6	C_{14}	C_{15}	C_{27}	C_{28}
C_2	**C_4**	**C_7**	C_{13}	C_{16}	C_{26}	C_{29}	C_{42}
C_3	**C_8**	**C_{12}**	**C_{17}**	**C_{25}**	C_{30}	C_{41}	C_{43}
C_9	**C_{11}**	**C_{18}**	**C_{24}**	C_{31}	C_{40}	C_{44}	C_{53}
C_{10}	C_{19}	**C_{23}**	C_{32}	C_{39}	C_{45}	C_{52}	C_{54}
C_{20}	C_{22}	C_{33}	C_{38}	C_{46}	C_{51}	C_{55}	C_{60}
C_{21}	C_{34}	C_{37}	C_{47}	C_{50}	C_{56}	C_{59}	C_{61}
C_{35}	C_{36}	C_{48}	C_{49}	C_{57}	C_{58}	C_{62}	C_{63}

Figure 8: The code vector for diagonal edge block codebook

Furthermore, to make the TCVQ can provide different compression ratios, each code vector of the codebooks possesses original AC coefficients. It means the transformed coefficients in these codebooks are not quantized. The user specifies a quality factor (Q) when they encode an image. TCVQ uses the quality factor (Q) and the default quantization table (Table I) of JPEG to quantize the DCT coefficients and the codebooks. Then each block is coded with an index value of the closest code vector in the quantized codebook. In the same way, the TCVQ uses the quality factor (Q) and the table to quantize the codebook while decoding image. The code vector in the quantized codebook is used to reproduce the DCT matrix. The missing components are padded with zero value. Finally, the DCT matrix is inversely transformed using IDCT to obtain an approximated image.

The traditional VQ-based compression method needs many various sizes of codebooks to achieve the request of various compression ratios. Take a 256 grey scale image with a block size of n×n for example. Choosing a codebook with size M, then the compression ratio in traditional VQ is $8n^2/\log_2 M$. If we need the higher compression ratio, the M must be reduced. It is necessary to train a new codebook with smaller size. But TCVQ uses only the same codebook at various compression ratios. Larger quality factor (Q) causes the number of "AC zero block" to increase, thus the compression ratio is increased.

Table I. The default quantization table of JPEG

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

4 Experimental Results

For performance evaluation, we designed four different sized codebooks by using LBG algorithm [12]. The shade block codebook is a 9-dimentional VQ with codebook size 64. The horizontal edge block codebook is an 11-dimentional VQ with codebook size 128. The vertical edge block codebook is an 11-dimentional VQ with codebook size 128. The diagonal edge block codebook is a 15-dimentional VQ with codebook size 256. The codebook size of different codebooks depends on the complexity of the blocks. The shade block is smooth and the codebook size is small. The diagonal edge block is too complicated and the codebook size is large. This kind of classification can also reduce the bit rate of image because regular image usually has a large number of smooth blocks. Besides, the threshold Γ defined by the block classification algorithm in section 3.1 is chosen to be 45, which is experimentally set and work for all images.

Table II. JPEG and TCVQ performances

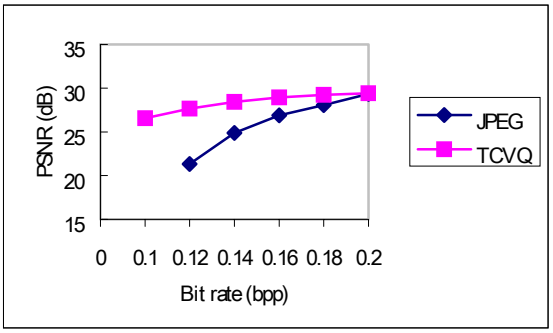
Bit rate (bpp)	PSNR (dB) JPEG	TCVQ
0.20	29.32	29.40
0.18	28.07	29.24
0.16	26.89	28.94
0.14	24.89	28.41
0.12	21.33	27.64
0.10	--	26.56

(a) Lena

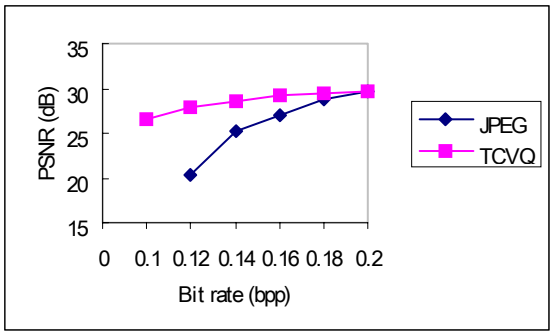
Bit rate (bpp)	PSNR (dB) JPEG	TCVQ
0.20	29.67	29.63
0.18	28.70	29.40
0.16	27.05	29.15
0.14	25.12	28.58
0.12	20.42	27.87
0.10	--	26.64

(b) Pepper

Two 512×512 images (Lena and Pepper) outside the training set were used for testing. The training set was obtained from eight different 512×512 images. All the images are 256 grey scale images. DCT is performed with a block size of 8×8.



(a) Lena



(b) Pepper

Figure 9: Bit rate versus PSNR using JPEG and TCVQ

We employ the bit per pixel (bpp) to estimate the transmission bit rate. It is defined as bpp = P/B, where P is the total number of pixels in an image and B is the total number of transmitted bits for this image. As a measure of reconstructed image quality, the peak signal-to-noise ratio (PSNR) in dB is used, which is defined as follows:

$$\text{PSNR} = 10 \times \log_{10} \frac{255^2}{\text{MSE}} \text{dB},$$

where MSE is the mean-square error. For an N×N image, its MSE is defined as

$$\text{MSE} = (\frac{1}{\text{N}})^2 \times \sum_{i=1}^{N} \sum_{j=1}^{N} (\alpha[i,j] - \beta[i,j])^2 .$$

Here, $\alpha[i,j]$ and $\beta[i,j]$ denote the original and decoded gray levels of the pixel [i,j] in the image, respectively. A larger PSNR value means that the encoded image preserves the original image quality better.

In general, the compression rate of VQ-based image compressor is obviously strictly dependent on the codebook size. For example in Kim et al.'s scheme [9], the bit rate is 0.625 bpp when using a 4×4 block with codebook size 1024. However, the size of the codebook at low bit rate, such as 0.25 bpp, is reduced to 16. It is quite obvious that the reconstructed image quality is degraded significantly. Thus a general VQ-based image compressor is inadequate for coding at low bit rates.

(a) Lena (b) Pepper

Figure 10: Original test images

For comparison, TCVQ versus baseline JPEG (using default Huffman table) are tabulated. The PSNR values of TCVQ and JPEG at different bit rates for Lena and Pepper images are given in Table II. The JPEG program is taken from public domain license: Stanford University Portable Video Research Group. The PSNR plots shown in Fig. 9 illustrate the performance of the TCVQ against JPEG for different bit rates.

In Table II and Fig. 9, it can be seen that TCVQ outperforms JPEG. The quality of JPEG image is degraded very quickly at low bit rate. On the contrary, TCVQ maintains a stable quality at low bit rate. Fig. 10 shows the original test images. Figures 11 and 12 show the reconstructed images for the two algorithms, with different bit rates.

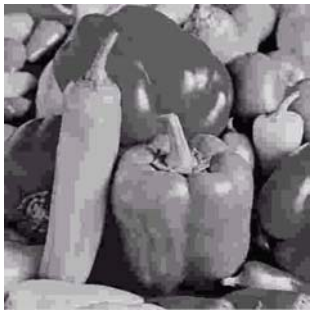
(a) JPEG compressed image, Bit rate = 0.18bpp, PSNR = 28.07

(b) TCVQ compressed image, Bit rate = 0.18bpp, PSNR = 29.24

(c) JPEG compressed image, Bit rate = 0.14bpp, PSNR = 24.89

(d) TCVQ compressed image, Bit rate = 0.14bpp, PSNR = 28.41

Figure 11: Test of Lena image



(a) JPEG compressed image, Bit rate = 0.18bpp, PSNR = 28.70

(b)TCVQ compressed image, Bit rate = 0.18bpp, PSNR = 29.40

(c) JPEG compressed image, Bit rate = 0.14bpp, PSNR = 25.12

(d) TCVQ compressed image, Bit rate = 0.14bpp, PSNR = 28.58

Figure 12: Test of Pepper image

5 Conclusions

This paper has proposed a new scheme TCVQ for image coding at very low bit rate. The TCVQ takes the advantages of both the DCT and the CVQ while preserving good reconstructed image quality. The codebook uses a smaller dimension for its code vector, reducing the complexity of the sample space and decreasing the size of the codebook. Besides, the TCVQ can provide different compression ratios even though it is a VQ-based compression method. The codebook is quantized for adapting to different compression ratios while encoding or decoding image. The only complex operation compared with JPEG is the codebook design, which is set up offline. The codebook lookups for image decoding is simple and fast. Evaluation of the compression performance of the TCVQ reveals its superiority over JPEG at low bit rate.

References

[1] W. Pennebaker and J. Mitchell (1993) *JPEG Still Image Data Compression Standard*. Van Nostrand Reinhold, New York.

[2] N. Ahmed, T. Natarjan, and K. R. Rao (1974) Discrete cosine transform, *IEEE Trans. Comput.* C-23, pp. 90-93.

[3] N. Nasrabadi and R. King (1988) Image coding using vector quantization: A review, *IEEE Trans. Commun.* COM-36, pp. 957-971.

[4] B. Ramamurthi and A. Gersgo (1986) Classified vector quantization of images, *IEEE Trans. Commun.* COM-34, pp. 1105-1115.

[5] J. W. Kim and S. U. Lee (1989) Discrete cosine transform - classified VQ technique for image coding, *Proc. IEEE ICASSP*, pp. 1831-1834.

[6] D. S. Kim and S. U. Lee (1991) Image vector quantizer based on a classification in the DCT domain, *IEEE Trans. Commun.* COM-39, pp. 549-556.

[7] J. W. Kim and S. U. Lee (1992) A transform domain classified vector quantizer for image coding, *IEEE Trans. Circuits and Systems for Video Technology*, Vol. 2, No. 1, pp. 3-14.

[8] D. Salomon (1997) *Data Compression – The Complete Reference*, Springer-Verlag, New York.

[9] B. H. Kim, T. Y. Kim, J. W. Lee, and H. M. Choi (1996) DCT-based high speed vector quantization using classified weighted tree-structured codebook, *Proceedings of IEEE International Conference on Systems, Man, and Cybernetics*, Vol. 2, pp. 935-940.

[10] T. Chiueh, T. Tang, and L. Chen (1994) Vector quantization using tree-structured self-organizing feature maps, *IEEE Journal on Selected Areas in Communications*, Vol. 12, No. 9, pp.1594-1599.

[11] A. Buzo, Jr. A. Gary, R. Gary, and J. Markel (1980) Speech coding based upon vector quantization, *IEEE Trans. Acoustics, Speech, and ignal Processing*, Vol. 28, pp. 562-574.

[12] Y. Linde, A. Buzo, and R. M. Gray (1980) An algorithm for vector quantizer design, *IEEE Trans. Commun.*, COM-28, pp. 84-95.