An Orientation Normalized Vector Quantizer for Flow-like Image Coding

Hai Tao and Zhaoqi Bian

Department of Automation, Tsinghua University, Beijing, 100084, P.R. China

Abstract

A new vector quantization image coding scheme called orientation normalized vector quantizer (ONVQ) is presented in this paper. During the encoding process, the orientation field of an image is first calculated and recorded using quad-tree data structure; then, the image is partitioned into digitalized round blocks. These round blocks are normalized to horizontal direction according to their original orientation and vector quantized. The decoder gets reproduction vectors from codebook and restores them to their original orientations. Since blocks of different orientations may use the same codeword, ONVQ needs a much smaller codebook; therefore low bit rates are achieved (0.6-0.8 bpp for 256×256 images). The subjective quality of ONVQ coded images are more satisfying than that obtained by a simple VQ scheme.

1. Introduction

Vector quantization (VQ) technique has been used for several years for low bit-rate image coding and applied in both spatial and transformed domains in various forms. Though some wavelet based transform [5] coding methods with hopeful prospects were presented, VQ techniques are still widely used. In a simple VQ scheme [8] (Fig.1), an image is first partitioned into square blocks of the same size. Since the computational complexity grows greatly as the vector dimension increases, the size of block is usually no larger than 5×5 . Then these blocks are vector quantized. A vector quantizer can be defined as a mapping Q of R^k into a finite subset Y of R^k. Thus $Q:R^k \rightarrow Y$, where $Y = \{ y_i ; i=1, 2, ..., N \}$ is the codebook and N is the number of vectors in Y. The encoder maps an input vector x to the nearest codeword y_i and sends the index i to the decoder in which the same codebook is stored. Thus to generate the reproduction vector for input vector x is just a look-up operation. Many clustering algorithms such as LBG algorithm were proposed to design an optimal codebook from a large training set of representative input vectors. In some VQ schemes, once a codebook is designed, it is used to code all similar images. These schemes are referred as VQ schemes using a fixed codebook. Other schemes design a special codebook for an input image and the codebook is transmitted. In this way, a more suitable codebook is designed. These schemes are called VQ schemes using non-fixed codebook.

The major problem of a simple VQ scheme is that visible blockiness or staircase distortion are usually presented in decoded images and the compression ratio is rather low. Gersho & Ramamurthi proposed classified VQ scheme (CVQ) in which several codebooks were designed to encode different types of input vectors.

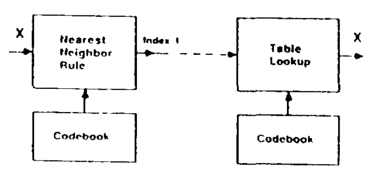

Fig. 1. Diagram of a simple vector quantization scheme.

Though this scheme solves the problem of edge distortion effectively and speeds up the encoding process, low bit rates may not be achieved. One reason is that the interblock correlation is still not exploited. To do this, some kinds of global information of images have to be used.

In mean/shape vector quantizer[1], only the error vector, obtained by subtracting sample mean from the input vector is vector quantized while the sample mean is simplely quantized and transmitted. Thus global amplitude information (sample mean) plays a important role in coding an image.

1063-6919/93 $03.00 © 1993 IEEE

Hierarchical vector quantizer [6] (HVQ) partitions an image into rectangular blocks of various sizes. In the area of high frequency, where gray levels of pixels change rapidly, blocks of size 4×4 or 2×2 are vector quantized by a simple VQ scheme. While in the area of low frequency, where gray levels of pixels change slowly, larger blocks of size 8×8 or 16×16 are used to partition the image. Then transform coding technique is applied. These larger blocks can also be vector quantized after resampling them to smaller blocks. In this way, the global information of frequency has been used to make the coding scheme more efficient.

Other approaches such as the finite state vector quantizer (FSVQ), proposed by Aravind and Gersho [2], are also applied successfully in exploiting the interblock correlation.

An orientation normalized vector quantizer (ONVQ) is proposed in this paper. The new scheme partitions an image into digitalized round blocks and uses the global orientation field information to exploit the interblock correlation. Blocks with different orientations may use the same codewords after orientation normalization and as a result, a smaller codebook is needed. This is especially beneficial in case of a non-fixed codebook is needed. Low bit rates are achieved for flow-like images whose orientation field information can be efficiently recorded. This technique has been successfully applied in an automated fingerprint identification system (AFIS) for fingerprint image data compression.

The paper is organized as follows. In section 2, the main idea of ONVQ is explained. Section 3 discusses the implementation of ONVQ scheme. In section 4, some experimental results are demonstrated and conclusions are given in section 5.

2. Main Idea

Textures and edges are important parts of images from which many kinds of global information such as orientation field can be extracted. The orientation of an edge is defined as the direction of its tangent line. and the orientation of a texture area is defined as the average orientation of all edges in this area (Fig.2). Traditional VQ schemes simply partition an image into rectangular blocks which means different codewords may be needed for blocks of different orientations. For example, in Fig.3-(a), four codewords should be stored in order to reproduce the four input vectors. If round blocks are used to partition an image, different input vectors may use the same codeword after orientation normalization (Fig.3-(b)). In this way, a much smaller codebook is needed and as a result, low bit rates are achieved. In order to normalize the orientation of blocks, orientation field of the whole image must be calculated and recorded efficiently.

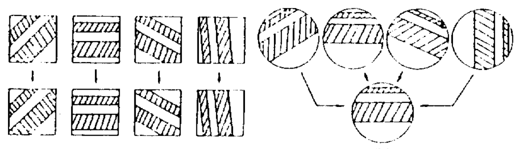

Fig. 3. (a) Four codewords are needed for four rectangular input block vectors. (b) Four round input block vectors use the same codeword.

The decoder restores the orientation field of an image and resamples the horizontal reproduced vectors to their original orientation. Fig.4 shows the diagram of ONVQ scheme.

3. Implementation of the Algorithm

A. Calculation and Coding of Orientation Field

Theoretically, orientation of every round block should be calculated, but it takes a lot of calculation and is not necessary in most cases. In our scheme, a 256×256 image is first partitioned into subimages of 8×8 pixels and the directions of these subimages are calculated and coded. Then the orientation of each subimage is assigned to all the rounds blocks in it.

Several algorithms [4] have been developed for computing the orientation field of flow-like images. The basic idea behind these algorithms is to use an oriented filter, namely the gradient of Gaussian in each point, and perform manipulations on the resulting gradient vector field. We use another kind of scheme in which the orientation angle is directly calculated for a given subimage.

if the pixels in a square block are denoted by $f(i, j)$, $i, j = 0, 1, ..., N-1$, then four oriented sums are calculated as follows

$$dI_0 = \sum_{i=0}^{N-1}\sum_{j=0}^{N-2} |f(i, j) - f(i, j+1)|$$
$$dI_{90} = \sum_{i=0}^{N-2}\sum_{j=0}^{N-1} |f(i, j) - f(i+1, j)|$$
$$dI_{45} = \sum_{i=0}^{N-2}\sum_{j=1}^{N-1} |f(i, j) - f(i+1, j-1)|$$
$$dI_{135} = \sum_{i=0}^{N-2}\sum_{j=0}^{N-2} |f(i, j) - f(i+1, j+1)| \quad (1)$$

if I_0+I_{90} and $I_{45}+I_{135}$ are less than a threshold, the block is regarded as smooth, else, four possible angles are derived from I_0 and I_{90}

$$\alpha_0 = arctg(\frac{dI_0}{dI_{90}}) \qquad \alpha_1 = \pi - \alpha_0$$

$$\alpha_2 = 2\pi - \alpha_0 \qquad \alpha_3 = 2\pi - \alpha_1 \qquad (2)$$

Similarly, other four possible orientation angles can be calculated from I_{45} and I_{135}

$$\beta_0 = arctg(\frac{dI_0}{dI_{90}}) \qquad \beta_1 = \pi - \beta_0$$

$$\beta_2 = 2\pi - \beta_0 \qquad \beta_3 = 2\pi - \beta_1 \qquad (3)$$

if α_i and β_j reinforce each other, which means the angular difference between them is no larger than a threshold θ_t, the orientation of the block is obtained by averaging the two orientation angles :

$$\theta = (\alpha_i + \beta_j)/2 \qquad (4)$$

else, if no α_i reinforces β_j, the block is labeled as non-oriented texture. The orientation of a block, if exists, is then normalized into $[\,0\,,\pi\,)$ and quantized to 14 levels from 0 to 13 while level 14 denotes a non-oriented textured block and level 15 means that the block is smooth.

In our scheme, a label is assigned to each 8×8 block in a 256×256 image; therefore a label matrix $L(i,j)$ i,j=0, 1, ..., 31 is created. To record the matrix, quad-tree structure is used. A quad-tree is a hierarchical data structure, where each node, unless it is a leaf, generates four children. It has been shown that the quad-tree can be used efficiently to describe the variable block size structure with a very small overhead. In this paper, for every four adjacent matrix elements, a decision rule is tested to see if they reinforce each other. If the test is positive, the elements are merged into 2×2 blocks. This process is repeated until the maximum block size of interest is attained.

B. Orientation Normalization of Round-like Blocks

Some kinds of round blocks shown in Fig.5 can be used to partition an image. Blocks like Fig.5-(b) can overlay an image without overlapping in the way shown in Fig.6. Blocks like Fig.5-(c) can also cover the whole image, but with a little overlapping. Our paper demonstrates the ONVQ scheme using Fig.5-(b) blocks while using a larger round blocks is similar. During encoding process, a round block is first resampled to horizontal orientation according to its original direction. To do this, some kind of interpolation must be performed. For bandlimited signals, ideal error-free interpolation is performed when coefficients taken from a sinc function are used. Some experiments showed that bilinear interpolation is good enough in many cases. This technique is used in our scheme.

A round block with pixels $x_0, \ldots, x_{11}$ is shown in Fig.7-(a), $x_{12}, \ldots, x_{15}$ are neighboring pixels. θ is the original orientation of the block. To resample the block to $x_0', \ldots, x_{11}'$ at angle θ (Fig.7-(b)) using bilinear interpolation technique. For example, if x_0' falls in the square area formed by x_0, x_2, x_3, x_{12}, the interpolation formula for x_0' is

$$x_0' = (1-p)qx_0 + p(1-q)x_2 + pqx_3 + (1-p)(1-q)x_{12} \qquad (5)$$

where p and q are interpolation coefficients and are functions of θ. if we define

$$X_a = \left[\, x_0\,, x_1\,, \ldots, x_{11} \,\right]^T$$

$$X_b = \left[\, x_{12}\,, x_{13}\,, x_{14}\,, x_{15} \,\right]^T \qquad (6)$$

and

$$X' = \left[\, x_0'\,, x_1'\,, \ldots, x_{11}' \,\right]^T \qquad (7)$$

which denotes the pixels of the round block after orientation normalization. The bilinear interpolation process can be expressed as :

$$x' = R(\theta)\cdot\begin{bmatrix} X_a \\ X_b \end{bmatrix} \qquad (8)$$

Where $R(\theta)$ is a 12×16 bilinear interpolation matrix. During the decoding process, the decoder first finds the reproduction vector in codebook according to the codeword index. Suppose the reproduced vector is $Y = \left[\, y_0\,, y_1\,, \ldots, y_{11} \,\right]^T$. As shown previously, Y is of horizontal orientation. Then , by using bilinear interpolation technique again, the horizontal block Y is resampled to its original orientation. If the resampled block is denoted as Y', the process can be expressed as

$$Y' = G(\theta)\cdot Y \qquad (9)$$

Where $G(\theta)$ is a 12×12 bilinear interpolation matrix function of θ. This process is shown in Fig.8. The effect of resampling in which bilinear interpolation is performed is in somewhat like that of a low-pass filter. This introduces additional error to the coding system. Suppose the error caused by the vector quantizer is

$$e = Y - X' \qquad (10)$$

then

$$Y' = G(\theta)\cdot Y$$

$$= G(\theta)\cdot(X' + e)$$
$$= G(\theta)\cdot\left[R(\theta)\cdot X + e \right]$$
$$= G(\theta)\cdot R(\theta)\cdot X + G(\theta)\cdot e \tag{11}$$

If the total coding error of the system is E and $G(\theta)\cdot R(\theta)=\left[F(\theta)\ ,\ S(\theta) \right]$ where $F(\theta)$ is a 12×12 matrix and $S(\theta)$ is a 12×4 matrix, then

$$E = Y' - X_a \tag{12}$$
$$= G(\theta)\cdot e + \left\{ \left[F(\theta) - I \right]\cdot X_a + S(\theta)\cdot X_b \right\}$$

The first part of the expression denotes the error caused by vector quantizer. It can be easily proved that

$$\| G(\theta)\cdot e \| \le \| e \| \tag{13}$$

Where $\| \cdot \|$ denotes Euclidean norm. The second part of the total error is caused by resampling process. In most cases, it is found relatively small compared with the error caused by vector quantizer. A method to reduce this part of error is normalizing orientations of blocks to an orientation possessed by most blocks, which means fewer blocks are needed to be resampled. Another remedy for this problem is to perform simple VQ for those blocks with resampling error exceeding a threshold.

4. Experiments

In this section, some experimental results are demonstrated. A fingerprint image is coded by ONVQ algorithms with codebooks of different sizes and the results are compared with those of a simple VQ scheme. Non-fixed codebooks created by LBG algorithm are used in our scheme. Fig.9(a) represents the original image "Fingerprint" of resolution 256×256 by 8 bits (32 level halftones made from laser printer) in which the orientation field is the most important global information. Another character of this image is the gray levels of pixels changing rapidly almost everywhere which makes it difficult to use frequency information to exploit interblock correlation. The orientation field recorded by quad-tree structure is shown in Fig.9(b). Objective and subjective qualities of decoded images are evaluated. The mean square error (MSE) is a widely accepted objective performance measure. it was defined as

$$MSE = \frac{1}{T}\sum_{j=1}^{T}(x_j - \hat{x}_j)^2 \tag{14}$$

where T denotes the total number of pixels in an image and x_j and $\hat{x}_j$ denote the original and reproduced pixels.

Bit rate is another important performance measure of image compression system. The bit rate of a simple VQ scheme with non-fixed codebook is calculated as

$$b_{VQ} = \frac{C_n K \log_2 G_l}{T} + \frac{\log_2 C_n}{K} = b_c + b_i \tag{15}$$

where

C_n number of codewords in the codebook

K dimension of the image vector

G_l number of grey level for every pixels in a codeword

T number of pixels in an image

while the bit rate of ONVQ is given by

$$b_{OAVQ} = \frac{C_n K \log_2 G_l}{T} + \frac{\log_2 C_n}{K} + \frac{q}{T}$$
$$A = b_c + b_i + b_q B \tag{16}$$

where q denotes the number of bits coding the angular information of the whole image.

ONVQ using different codebook sizes of 64,128 and 256 is compared with a simple VQ algorithm. Several observations can be obtained from the experimental results given in TABLE I.

- A smaller codebook is needed for ONVQ to get the same MSE as simple VQ. The MSE of ONVQ using 64 codevectors is even lower than that of a simple VQ scheme using 128 codewords. This is more notable when the codebook size is reduced.
- Bit rates only increased slightly when orientation field is recorded. In TABLE I, the b_q is found very small compared with b_c and b_i. This is especially true for those images whose orientation fields change slowly.

TABLE I

VQ & ONVQ			VQ	ONVQ	
C_n	b_c (bpp)	b_i (bpp)	MSE	b_q (bpp)	MSE
64	0.098	0.5	304.1	0.0083	191.9
128	0.197	0.58	196.4	0.0083	171.0
256	0.295	0.67	167.1	0.0083	159.6

The images coded with 128 codewords by ONVQ and simple VQ are shown in Fig.9(c)(d). Parts of the decoded images are enlarged and shown in Fig.9(e)(f). From these images, it was found the staircase distortion is greatly weakened by ONVQ. This makes the subjective quality of ONVQ coded images more satisfying than that obtained by using simple VQ.

5. Conclusion

In this paper, we introduced a new VQ scheme in which the orientation information is used to control the coding process. An image is first partitioned into round blocks. Those blocks with strong orientation information are resampled to the same direction before they are vector quantized. As described in section 2, several blocks of different orientations may use the same codeword; therefore a smaller codebook is needed. For those blocks with weak or no orientation information, tow special codebooks are designed (level 14 and level 15). As a result, ONVQ could code these blocks at least as efficiently as a CVQ scheme. ONVQ performs very well for coding flow-like images since their orientation fields can be efficiently recorded while if the image orientation fields change too rapidly, it does not save much more bits than using a simple VQ scheme. Another disadvantage of ONVQ is that the resampling operation introduces additional error as a low-pass filter does. Several remedies for this problem have been proposed in this paper.

References

[1] S. E. Budge and R. L. Baker, *Compression of color digital images using vector quantization in product codes*, in Proc. IEEE ICASSP, pp. 129-132, Apr. 1985.

[2] A. Aravind and A. Gersho, *Low-rate image coding with finite-state vector quantization*, in Proc. IEEE ICASSP, Mar. 1986, pp. 137-140

[3] Nasser M. Nasrabadi and Robert A. King, *Image Coding Using Vector Quantization: A Review*, in IEEE Trans. Commun., vol. COM-36, pp. 957-971, Aug. 1988.

[4] A. Ravishankar Rao and Brian G. Schunck, *Computing oriented texture fields*, CVGIP: Graphical Models and Image Processing, vol. 53, pp. 157-185, 1991.

[5] Stephane G. Mallat, *A theory for multiresolution signal decomposition: the wavelet representation*, IEEE Trans. PAMI, vol. 11, No. 7, pp. 674-693, July 1989.

[6] C. Y. Chiu, C. T. Chen and Didier J. Le Gall, *Quadtree-structured finite state vector quantization for still images*, Presented at the 1989 conf. on info. sciences & syst., the Johns Hopskin univ., Baltimore, MD., Mar. 1989.

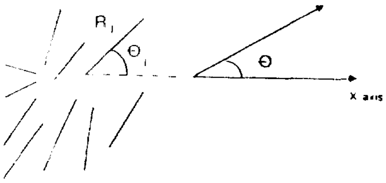

Fig. 2. The definition of orientation of a texture area.

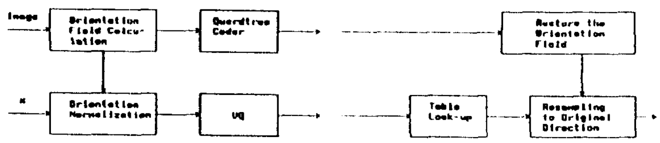

Fig. 4. The diagram of ONVQ.

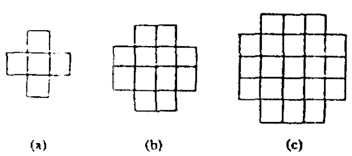

Fig. 5. Three kinds of digitalized round blocks.

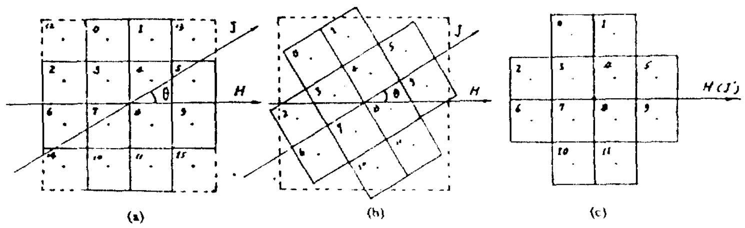

Fig. 7. The resampling process of the ONVQ encoder.

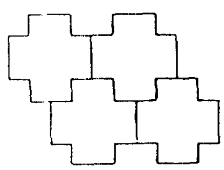

Fig. 6. Digitalized round blocks overlay an image without overlapping.

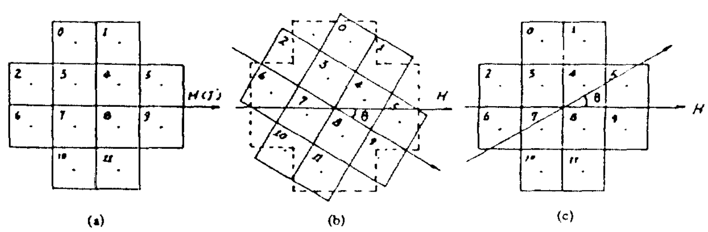

Fig. 8. The resampling process of the ONVQ decoder.

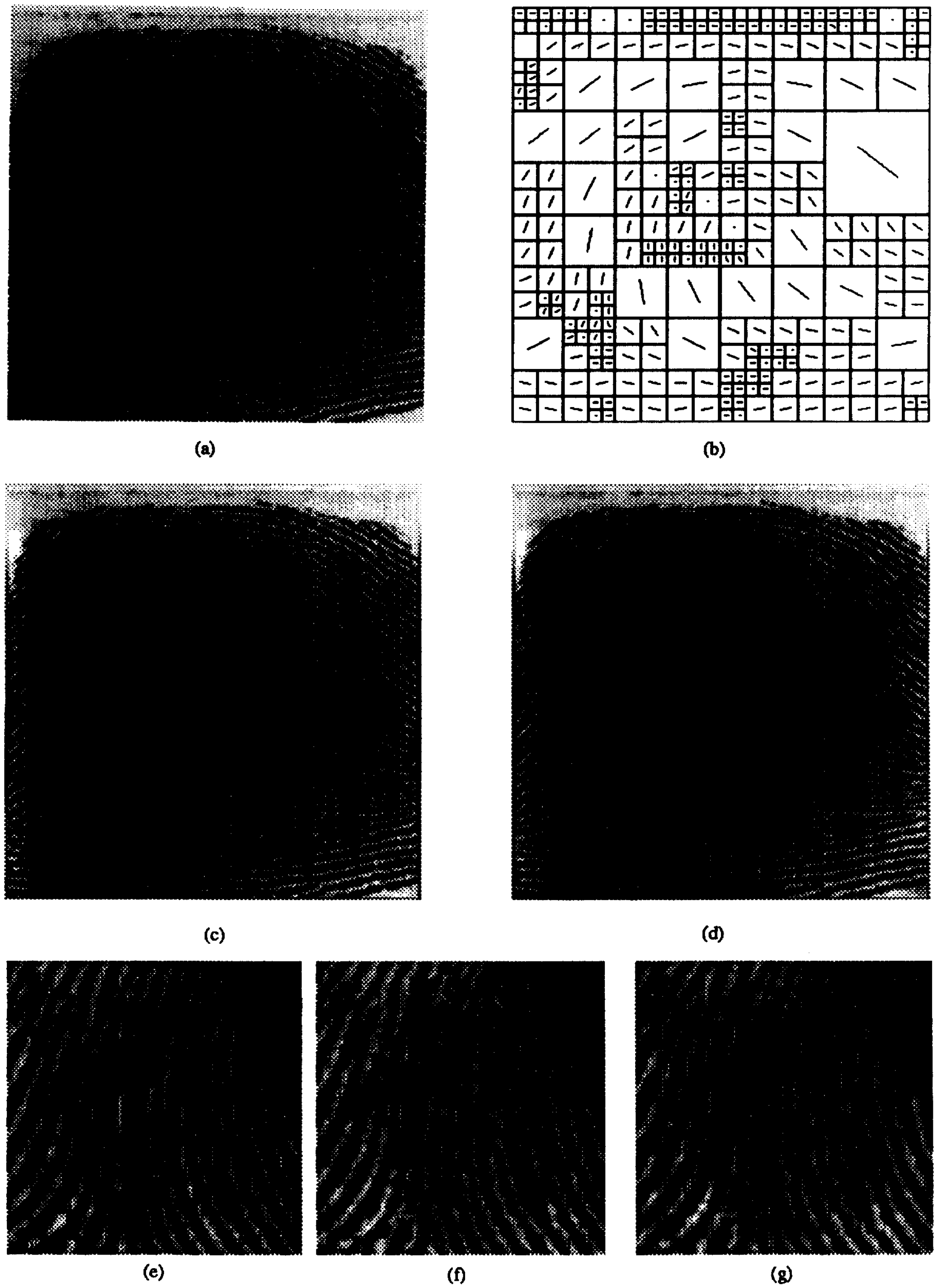

Fig. 9. (a) Original 256×256 image "Fingerprint". (b) The orientation field recorded by quad-tree structure. (c) Simple VQ with 128 codewords. (d) ONVQ with 128 codewords. (e) A part of Original image. (f) A part of image in (c). (g) A part of image in (d).