Variable-Rate Hierarchical Vector Quantization with Variance Classification

H.C. Chao,[1] Wei-Ming Chen,[2] C.Y. Lin,[3] J.L. Chen,[3] S.F. Linn[3]

[1] Department of Electrical Engineering, National Dong-Hwa University, Hualien, Taiwan, Republic of China 97401. E-mail: Hcc@mail.ndhu.edu.tw

[2] Department of Computer Science and Information Engineering, National Chung Cheng University, Chia-Yi, Taiwan, Republic of China. E-mail: wmchen@mail.ndhu.edu.tw

[3] Department of Computer Science and Information Engineering, National Dong-Hwa University, Hualien, Taiwan, Republic of China. E-mail: Lchen@mail.ndhu.edu.tw; david@mail.ndhu.edu.tw

ABSTRACT: Because the bit rate can be adjusted depending on the network traffic situation, the variable bit rate is more suitable than a fixed rate for transmitting images on the network. In this paper, a new variable rate hierarchical classified vector quantization (VHCVQ) scheme is proposed. This technique is suitable for progressive transmission. In our two-stage VHCVQ scheme, we use the differences between the first-stage input vectors and the first-stage quantized vectors, called residual vectors, to code the image in the second stage. Usually the values between these differences are small and each is close to one another. Using the difference to code the image may reduce the codebook size and the bit rate as well. In our simulation, using these differences reduced the bit rate by about 0.075 bpp. © 2000 John Wiley & Sons, Inc. Int J Imaging Syst Technol, 11, 138–143, 2000

Key words: VQ; codeword; codebook; quantization

I. INTRODUCTION

Vector quantization (VQ; Linde et al., 1980; Chou et al., 1999) is used widely in image coding systems. A vector quantizer includes two components: an encoder and decoder. The image to be quantized is first partitioned into nonoverlapping blocks. The input vectors are then quantized to the closest codewords that are measured by several distortion metrics such as mean-square error (MSE). After the closest codeword is found, the codeword index is stored. The decoder then uses that index to retrieve the codeword from the lookup table and output it as the reconstructed vector. Image compression is achieved using the indices of the codewords for storage and transmission.

VQ is a block-coding method that exploits all statistical dependencies within a given block. In ordinary VQ, using a larger block size can achieve a higher compression, but a very large codebook is required for a fixed picture quality. The large-sized codebook leads to a large number of comparisons in order to find the best codeword. This reduces the efficiency of the encoder. This is why an ordinary VQ usually uses a 4×4-pixel block size.

Hierarchical vector quantization (HVQ; Gersho and Shoham, 1984; Chang et al., 1985) is proposed to overcome the block size barrier in the ordinary VQ. In HVQ, the image is coded in several stages, starting with a relatively large block size, and then dividing that block into smaller subblocks to perform the next VQ stage. Because HVQ codes images in several stages, the first-stage codebook size need not be very large. We can view the characteristics (e.g., the variance value or whether or not it has edges) of the block to decide if the next coding stage is required. Not all blocks require next-stage coding, so the bit rate can be reduced.

Classified VQ (CVQ) is an effectual technique to encode an image at a low bit rate. It uses a classifier to classify each input vector into one of several different classes by features such as the vector's variance (Chang and Chen, 1995), edge direction, or other heuristic characteristics. Each class has its own codebook called a class codebook.

The organization of this paper is as follows. Section II lists some basic definitions and provides the overview of the proposed method. The variable rate hierarchical classified vector quantization (VHCVQ) encoder and decoder are described briefly in Section III. Section IV describes the simulation results and conclusions are presented in Section V.

II. BASIC DEFINITIONS

The variance for a vector X with dimension k is defined as

$$\mathrm{var}(X) = \frac{1}{k} \sum_{i=0}^{k-1} (X_i - \bar{X})^2 \qquad \bar{X} = \frac{1}{k} \sum_{j=0}^{k-1} X_j$$

The block with low variance is usually referred to as a low-detail block. A high variance block usually means a highly detailed block. Typically, if an edge passes through a block, the variance of the

Correspondence to: H.C. Chao

Table I. VHCVQ simulation results with or without using the difference in the second stage.

Image	With Difference	Without Difference
Inside		
Lena		
Bit rate	0.3154	0.3939
PSNR[a]	30.783	30.137
Airplane		
Bit rate	0.2523	0.2910
PSNR	29.435	29.083
Boat		
Bit rate	0.3162	0.3879
PSNR	29.203	28.538
Lake		
Bit rate	0.3168	0.3794
PSNR	27.725	27.543
Baboon		
Bit rate	0.3536	0.4374
PSNR	23.203	22.983
Man		
Bit rate	0.3367	0.4160
PSNR	28.306	27.520
Outside		
Camera		
Bit rate	0.3151	0.3568
PSNR	27.966	29.148

[a] Peak signal-to-noise ratio.

block increases. We can also predict an edge or nonedge block by using its variance.

In the two-stage VHCVQ scheme, we used the difference in an 8×8 input vector and its quantized vector to code the image in the second stage. The difference for an input vector and a quantized vector with dimension k is defined as:

$$D_i = X_i - \tilde{X}_i \quad \text{for} \quad i = 0 \text{ to } k - 1.$$

Where D is the difference, X is the input vector, and $\tilde{X}$ is the quantized vector.

The D_i value is the error for the quantized vector. Usually, the value of D_i is small and the difference in value for each vector is close. Using this difference to code an image may reduce the codebook size and the bit rate.

The value of D_i is usually close to 0; it may be a plus or a minus value. Before training the codebooks from the difference vectors, the vectors could be normalized in order to keep all the residual vectors positive. In order to transform all values of D_i into plus values, the algorithm added 128 to each D_i. When decoding, each 4×4 value codeword should be minus 128. Considering the difference as a factor, an image can be coded under a low bit rate. Table I shows the VHCVQ performance with or without the difference.

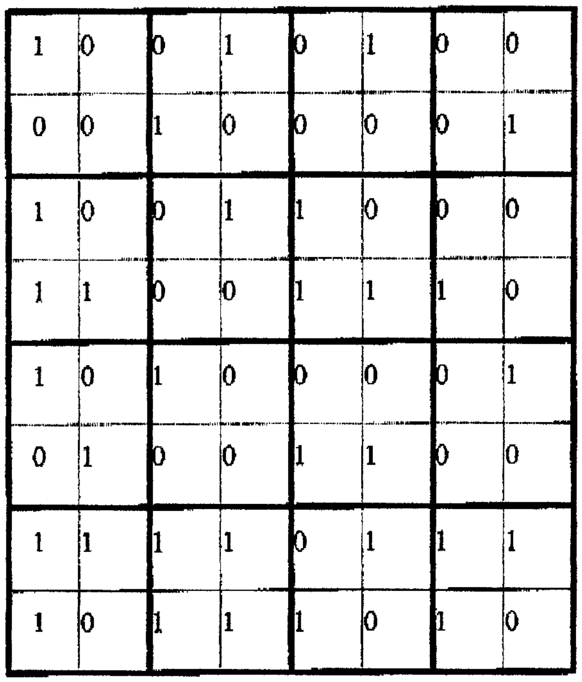
A. An Overview of the Algorithm. In the proposed method, we adopted a two-stage HVQ scheme. The first stage uses a larger block (8×8 pixels). If the variance of the quantized block is lower, it should be considered a low-detail block and no second stage coding is needed. However, if the variance of the quantized block is higher, as in a high-detail block, it will be divided into four 4×4 subblocks. Our proposed classified technique then follows for further 4×4 block coding.

B. Classified Method. In our two-stage VHCVQ scheme, we used a two-stage classification method. In the first stage, a codebook with 256 codewords was used and the block size was 8×8. All of the 8×8 quantized blocks were classified into four classes (class 0–class 3) using their variance. Before classifying, three thresholds, TH_1, TH_2, and TH_3, must be given. These thresholds are chosen to separate all 8×8 vectors into four equal groups (class 0–class 3). The vectors in class 0 are the low variance, low-detail blocks. It is not necessary to perform second-stage coding. Class 1–class 3 are higher variance, higher-detail blocks. The vectors in these classes require second-stage VQ.

After classifying all of the 8×8 codewords, each codeword will be divided into four 4×4 subblocks. Computing the variance of each subblock, if the variance of the subblock is smaller than the given threshold, TH, this subblock will be set to 0, otherwise it will be set to 1. Note that the subblocks were quantized at the first stage (divided by an 8×8-coded vector). The quantized results (vector index) were also transmitted to the client. For this reason, Figure 1 need not be transmitted to the client. To choose the TH value, all of the 4×4 subblocks for the 8×8 codewords should be sorted using their variances. The median is chosen as the TH value.

Second-stage classification now begins. In first-stage classification, all codewords were classified into one of the four classes. If a codeword belongs to class 1, class 2, or class 3, it will be divided into four subblocks. Each subblock will be subdivided into one of 32 classes.

In Figure 2, block A is the current encoding block; B, C, D, and E are its neighboring blocks. If A's neighboring block is empty, it



1	0	0	1	0	1	0	0
0	0	1	0	0	0	0	1
1	0	0	1	1	0	0	0
1	1	0	0	1	1	1	0
1	0	1	0	0	0	0	1
0	1	0	0	1	1	0	0
1	1	1	1	0	1	1	1
1	0	1	1	1	0	1	0

Figure 1. Each 4×4 block of the quantized image is mapped into a 0 or 1 value: 1 represents a high variance and 0 stands for a low variance.

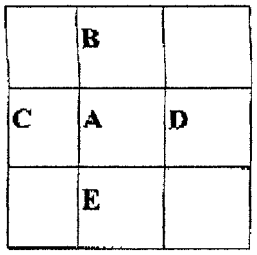


Figure 2. 4 × 4 blocks in the second stage.

will be set to 0. Blocks A, B, C, D, and E can be 0 or 1, so there are 32 (2^5) states, each state having its own codebook state. The training vectors for each codebook state is generated using the same classification method.

Because the variance of the current encoding block (block A) is more important than the others, these 32 states are broken down into two parts: (I) block A = 0 and (II) block A = 1. These two parts have different codebook sizes. The codebook sizes are greater when block A = 1 (Table II).

The classification method also takes the variance of the neighboring blocks into consideration. These blocks can usually be used to predict the direction of the edges (Chang and Cheng, 1996). As we mentioned earlier, high-variance blocks usually have edges passing through them. For example, block A may have a vertical edge when blocks A, B, and E have high variance. Block A may have a horizontal edge if blocks A, C, and D have a high variance. In the proposed classification method, the blocks with the same edge direction will usually be classified in the same class.

III. ENCODER AND DECODER ALGORITHM

A. Encoder Algorithm. **Step 1:** For each 8 × 8 input vector X, find the nearest codeword, $\bar{X}_j$, transmit index j to the decoder.

Step 2: If there is another 8 × 8 input vector required for encoding, go to step 1, otherwise go to Step 3.

Step 3: Find the class i for each quantized block. If $i = 0$, go to Step 5, otherwise go to Step 4.

Step 4: Use difference, $D = X - \bar{X}$, and divide it into four 4 × 4 subblocks, d_1, d_2, d_3, d_4, select the state codebook for each block by our classified method; find the nearest codeword $\tilde{d}_{k1}$, $\tilde{d}_{k2}$, $\tilde{d}_{k3}$, $\tilde{d}_{k4}$ for each block; transmit index k_1, k_2, k_3, and k_4 to the decoder.

Step 5: If there is another 4 × 4 vector required for encoding, go to step 3, otherwise STOP.

Table II. Second-stage state codebook size.

Value of Block A	Codebook Size
Class 1	
A = 0	8
A = 1	16
Class 2	
A = 0	16
A = 1	32
Class 3	
A = 0	16
A = 1	64

Table III. Simulation results.

Image	VHCVQ	SMVQ	VHCVQ	SMVQ
Lena				
Bit rate	0.3154	0.3154	0.1784	0.1924
PSNR	30.783	29.200	29.274	25.738
Airplane				
Bit rate	0.2523	0.3154	0.1893	0.1924
PSNR	29.435	27.918	28.257	24.124
Boat				
Bit rate	0.3162	0.3154	0.1892	0.1924
PSNR	29.203	27.621	27.633	24.592
Lake				
Bit rate	0.3168	0.3154	0.2070	0.1924
PSNR	27.725	26.205	26.722	23.223
Baboon				
Bit rate	0.3536	0.3154	0.2136	0.1924
PSNR	23.203	22.017	21.973	20.704
Man				
Bit rate	0.3367	0.3154	0.1853	0.1924
PSNR	28.306	27.751	26.873	24.923
Camera				
Bit rate	0.3151	0.3154	0.1716	0.1924
PSNR	27.966	26.987	27.073	24.412

B. Decoder Algorithm. **Step 1:** Use the index, j, to find the reconstructed vector, $\tilde{X}_j$, from the codebook.

Step 2: If there is another 8 × 8 vector for decoding, go to Step 1, otherwise go to Step 3.

Step 3: Find the class i for each quantized block, $\tilde{X}$. If $i = 0$, go to Step 5, otherwise go to Step 4.

Step 4: Find the state codebooks and use the indices k_1, k_2, k_3, k_4 to find the 4 × 4 reconstructed difference vectors $\tilde{d}_{k1}$, $\tilde{d}_{k2}$, $\tilde{d}_{k3}$, $\tilde{d}_{k4}$ from the state codebooks. Combine these four vectors into an 8 × 8 vector, $\tilde{D}_k$; add $\tilde{D}_k$ to $\tilde{X}_j$.

Step 5: If there is another 4 × 4 vector required for decoding, go to Step 3, otherwise STOP.

IV. SIMULATION RESULTS

The simulation results were processed on the SUN ULTRA SPARC workstation. All images used in these experiments were 512 × 512 monochrome still images, each pixel containing 256 gray levels. We used 10 pictures as the training set. Those were Lena, Airplane, Boat, Lake, Goldhill, Baboon, Barbara, Girl, Couple, and Zelda. Besides these 10 pictures, Man and Camera were not included in the training set, but were used to test the performance. Performance can be evaluated using the PSNR defined as:

Table IV. Comparison between JPEG and VHCVQ.

	Lena		Boat	
Compression Method	PSNR	Compression Rate	PSNR	Compression Rate
JPEG	30.974	33.55	29.136	30.84
VHCVQ	30.785	25.36/0.86 (lossless compression) = 29.49	29.203	25.30/0.89 (lossless compression) = 28.42