

Short Paper

Images Coding Using Two-Pass Side-Match Finite-State Vector Quantization

RUEY-FENG CHANG AND WEN-JIA KUO

Department of Computer Science and Information Engineering

National Chung Cheng University

Chiayi, Taiwan 621, R.O.C.

E-mail: rfchang@cs.ccu.edu.tw

Among the image coding techniques, vector quantization (VQ) has been considered to be an effective method for coding images at low bit rate. The side-match finite-state vector quantizer (SMVQ) exploits the correlations between neighboring blocks (vectors) to avoid large gray level transition across block boundaries. In this paper, an improved SMVQ technique named two-pass side-match finite-state vector quantization (TPSMVQ) is proposed. In TPSMVQ, the size of the state codebook in the first pass is decided by the variances of neighboring blocks. In the second pass, we improve the blocks encoded in the first pass whose variances are greater than a threshold. Moreover, not only the left and upper blocks, but also the lower and right blocks are used to construct the state codebook. In our experiment results, the average improvement of the second pass was up to 1.5 dB in PSNR over the first pass. In comparison to ordinary SMVQ, the average improvement is up to 1.54 dB at nearly the same bit rate.

Keywords: finite-state vector quantization, side-match vector quantization, variable-rate coding

1. INTRODUCTION

In the future, image compression and image sequence coding will become essential for many kinds of applications such as video conferencing, video phones, TV transmission, facsimile transmission, and so on. In the past few years, vector quantization (VQ) has been thought to be an efficient method for image compression [1-4]. The algorithms most commonly used to obtain the codebook used in vector quantization is the iterative clustering algorithm, such as the K-means or the generalized Lloyd clustering algorithm (LBG) proposed by Linde, Buzo and Gray [5]. Each input vector is individually quantized to the closest codeword in the codebook. Compression is then achieved by using the indices or labels of the codewords for the purposes of storage and network transmission. Reconstruction of the image can be finished using table lookup techniques with indices to select the corresponding codewords from the codebook. The major advantage of VQ is that it is easy to implement on the hardware of the decoder.

Finite-state vector quantization (FSVQ) [6-12] has been proposed to exploit the correlations between neighboring blocks to reduce the bit rate. Side-match VQ (SMVQ) [10] is one of the well-known versions of FSVQ. However, better approximation of the input vector may not be found in the state codebook because of low correlations in some situations. For this reason, the quality of the encoded image will be reduced by using the ordinary SMVQ. Hence, in this paper, two-pass side-match finite-state vector quantization (TPSMVQ) is proposed for still image coding.

The organization of this paper is as follows. In Section 2, we review the main features of VQ, FSVQ, and SMVQ. Section 3 describes our TPSMVQ algorithm. Some experimental results are presented in Section 4. In Section 5, some conclusions are given.

2. VECTOR QUANTIZATION TECHNIQUES

A vector quantization can be defined as a mapping from a k -dimensional Euclidean space R^k into a finite subset of R^k called the VQ codebook with size N . The VQ coder can be seen as having two parts: an encoder, which assigns each input vector $\mathbf{x} \in R^k$ to an *index* i that points to the closest codeword $\hat{\mathbf{x}}_i$ in the codebook, and a decoder, which finds the codeword according to the transmitted *index* i .

An FSVQ can be defined as a mapping from $R^k \times S$ to a subset of a super codebook $C = \{\hat{\mathbf{x}}_i : i = 1, \dots, N_s\}$, where $S = \{s_i : i = 1, \dots, M\}$ is the space of the state variables. For each state s_i , we select N_j codewords from the super codebook C as the state codebook SC_{s_i} . In comparison with the size of super codebook, N_j is much smaller than N_s . Hence, the searching time for the codeword of input vector \mathbf{x} can be effectively reduced. The decoder uses the same state function to find the current state s and the corresponding codeword in the same state codebook SC_s by means of the transmitted index.

Side-match vector quantization was proposed by Kim [10] in 1992. SMVQ tries to make the gray level transition across the boundaries of neighboring blocks much smoother. Therefore, it uses the sides of upper and left neighboring blocks to select the codewords from the super codebook as the member of state codebook SC_s for the input vector \mathbf{x} . The corresponding state codebook SC_s of the current input vector \mathbf{x} is composed of those codewords that best match the upper block and the left block relative to \mathbf{x} along the neighboring columns and the neighboring rows, respectively.

3. TWO-PASS SIDE-MATCH FINITE-STATE VECTOR QUANTIZATION

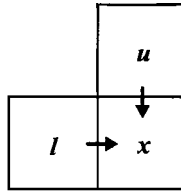
Variable-rate encoding techniques [13-18] have been recognized as being more suitable than fixed-rate encoding techniques. The main advantages of the variable-rate technique are reduction of the bit rate, easy control of image quality, etc. Hence, a variable-rate encoding technique is used in the first pass of our method. Moreover, we can also improve the blocks encoded in the first pass whose variances are greater than a threshold.

In the first pass of our TPSMVQ algorithm, ordinary SMVQ is used except that the size of the state codebook is decided by the variances of neighboring blocks. High-detail blocks are encoded using a larger state codebook, and low-detail blocks are encoded using a smaller one.

The variance of a vector x is defined as

$$\text{var}(x) = \frac{1}{k} \times \sum_{i=0}^{k-1} (x_i - \bar{x})^2, \text{ where } \bar{x} = \frac{1}{k} \times \sum_{i=0}^{k-1} x_i. \quad (1)$$

A threshold TH_1 is used to decide whether it is a high-detail block or not. Fig. 1 shows three classes of blocks. Each class i has a state codebook SC_i . When we encode the input vector, we first find its class and decide the size of the state codebook. Then, we use only the upper and left blocks to predict the codewords of the state codebook by using the ordinary side-match vector quantization method. The encoder selects the closest codeword in the state codebook to be the codeword of the input vector. Fig. 2(a) shows the neighboring blocks used to construct the state codebook in this pass.



$$\begin{aligned} \text{class}(x) &= 1 \text{ if } \text{var}(u) < TH_1 \text{ and } \text{var}(l) < TH_1 \\ \text{class}(x) &= 2 \text{ if } (\text{var}(u) < TH_1 \text{ and } \text{var}(l) > TH_1) \text{ or } \\ &\quad (\text{var}(u) > TH_1 \text{ and } \text{var}(l) < TH_1) \\ \text{class}(x) &= 3 \text{ if } \text{var}(u) > TH_1 \text{ and } \text{var}(l) > TH_1 \end{aligned}$$

Fig. 1. The classifier and three classes of the classifier.

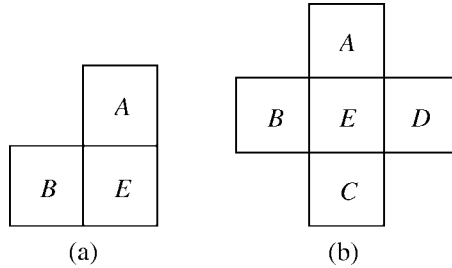


Fig. 2. The neighboring blocks used to construct the state codebook. (a) Block E is the current encoded block, and blocks A and B are the neighboring blocks used to construct the state codebook in the first pass. (b) Block E is the current encoded block, and blocks A , B , C , and D are the neighboring blocks used to construct the state codebook in the second pass.

In the second pass, we improve the quality of the image encoded in the first pass. Those blocks whose variances are greater than TH_2 will be encoded again. The main purpose is to improve the quality of complex blocks. In this pass, we also use the ordinary side-match vector quantization method except that four neighboring blocks instead of two neighboring blocks are used to select the state codebook for the reason that the approximate right and

lower blocks have already obtained in the first pass. Fig. 2(b) shows the neighboring blocks used to construct the state codebook in the second pass of TPSMVQ. Finally, the encoder will select the closest codeword in the state codebook of the second pass to replace the codeword selected in the first pass for the current input vector. Note that only codewords with smaller distortions compared to those obtained in the first pass are transmitted to the decoder in this step. Hence, an overhead bit is needed here for those blocks whose variances are greater than TH_2 to distinguish whether the codeword obtained from the first pass should be replaced or not.

In the proposed TPSMVQ, the first pass and the second pass are both coded from left to right and up to down. In fact, a different scanning method can be adopted in the second pass of TPSMVQ. The backward scanning method has been used in the second pass of our TPSMVQ. In the backward scanning method, images are coded from right to left and down to up. We use only the right and lower blocks to predict the codewords of the state codebook for the second pass instead of using four neighboring blocks as in the previously proposed TPSMVQ. Hence, the correlation of upper block and left block is used in the first pass and the correlation of right block and lower block is used in the second pass. In the ordinary second pass, those blocks whose variances are greater than the threshold TH_2 will be encoded again. Moreover, those blocks whose mean differences MD are greater than a threshold TH_c will be also encoded again. The mean difference MD of block E is defined as $\left| \frac{1}{2}(M_C + M_D) - M_E \right|$, where M_C , M_D , and M_E are the mean values of blocks C , D , and E shown in Fig. 2(b).

SMVQ selects a state codebook from the super codebook by exploiting the correlations between neighboring blocks to reduce the bit rate. However, correlation between neighboring blocks is not high in a complex image. Thus, the ordinary SMVQ may reduce the image quality of the encoded image. For example, SMVQ may incorrectly encode the edge if the edge is across the lower right corner of the block. The reason is that SMVQ uses only the left block and upper block to predict this block while the edge is across the right block and lower block. Hence, our proposed backward scanning method for the second pass of TPSMVQ can effectively solve this problem. Fig. 3(a) shows an original image that contains an edge across blocks C , D , and E . Fig. 3(b) shows the possibly incorrect result encoded by using the ordinary SMVQ.

Fig. 4 shows the encoder of TPSMVQ. The first pass of the encoder is shown in Fig. 4 (a), and the second pass is shown in Fig. 4(b). The encoding algorithm of the TPSMVQ algorithm can be given as follows.

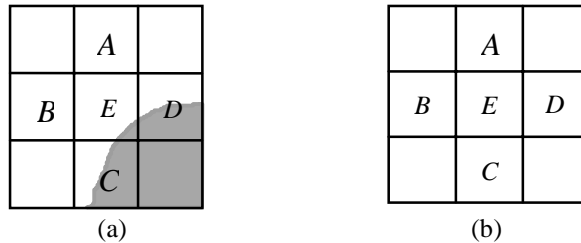


Fig. 3. (a) The original image that contains an edge across blocks C , D , and E . (b) The possibly incorrect result encoded using the ordinary SMVQ.

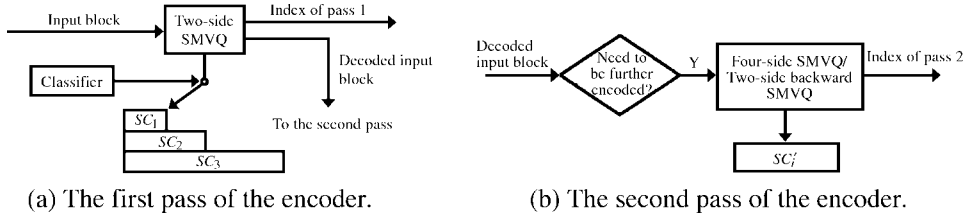


Fig. 4. TPSMVQ encoder.

The first pass:

- Step 1:** Find class i of the input vector x using the block classifier. Decide the size of the state codebook according to class i .
- Step 2:** Select the codewords of the state codebook SC_i from the super codebook C .
- Step 3:** Find the closest codeword \hat{x} from the state codebook SC_i , which will be the codeword of the input vector x . Send index p of \hat{x} to the decoder.
- Step 4:** If another input vector needs to be encoded, go to step 1.

The second pass:

- Step 1:** Let \hat{x} be the codeword of the input vector x obtained in the first pass. If $var(\hat{x}) < TH_2$, then go to step 4.
- Step 2:** Select the codewords of the state codebook SC'_i from the super codebook C . Here, the upper, left, lower, and right neighboring blocks are used to predict the codewords of the state codebook in the ordinary TPSMVQ. Also, only the right and lower blocks are used to predict the codewords of the state codebook in the backward scanning method.
- Step 3:** Find the closest codeword \hat{x}' from the new state codebook SC'_i which will be the new codeword of the input vector x . Send the index p' of \hat{x}' to the decoder if it has smaller distortions compared to that obtained in the first pass.
- Step 4:** If there are still vectors which need to be encoded, go to step 1.

Fig. 5 shows the decoder of TPSMVQ. The first pass of decoder is shown in Fig. 5(a), and the second pass is shown in Fig. 5(b). The steps of the decoding algorithm of the TPSMVQ algorithm are similar to those in the encoder.

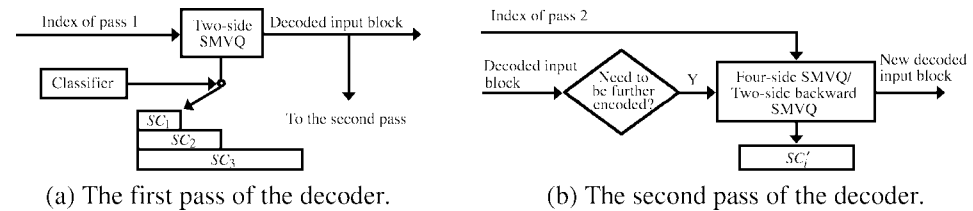


Fig. 5. TPSMVQ decoder.

Note that the state codebook of the second pass is different from that in the first pass. The advantage of TPSMVQ is that the bit rate for low-detail blocks can be reduced in the first

pass. The reason is that low-detail blocks are encoded using a smaller state codebook. Moreover, the quality of high-detail blocks can be improved after the second pass encoding.

4. EXPERIMENTS AND RESULTS

Computer simulations of TPSMVQ have been performed for several 512×512 monochrome still images with 256 gray levels. The quality of the encoded image was evaluated using the peak signal-to-noise ratio (PSNR). The PSNR is defined as

$$\text{PSNR} = 10 \log_{10} \frac{255^2}{\text{MSE}} \text{dB}. \quad (2)$$

Note that the mean-square error for an $m \times m$ image is defined as

$$\text{MSE} = \left(\frac{1}{m} \right)^2 \sum_{i=1}^m \sum_{j=1}^m (x_{ij} - \hat{x}_{ij})^2, \quad (3)$$

where x_{ij} and \hat{x}_{ij} denote the original and quantized gray levels, respectively.

The super codebooks of TPSMVQ, ordinary SMVQ, and ordinary VQ were generated using the LBG algorithm from a training set of five 512×512 still images including Lena, Lenna, Barbara, Boat, and Toys. The images Peppers, Airplane, and Zelda were outside the training set. The block size was 4×4 , resulting in a total of 16384 vectors to be encoded in an image. The threshold TH_1 for the first pass ranged from 3 to 200, and the threshold TH_2 for the second pass ranged from 5 to 200. The threshold TH_e for the mean difference was 50. The size of the super codebook was 256. In the first pass, the state codebook size for each class was 8, 16, and 32, respectively. The size of the state codebook in the second pass was 64.

The experiment results are shown in Tables 1-5. Table 1 shows the results of TPSMVQ, ordinary SMVQ, and ordinary VQ for almost the same bit rate. In our experimental results, the average improvement of the second pass was up to 1.5 dB in PSNR compared to the first pass. The improvement of TPSMVQ over ordinary VQ was 3.465 dB for the same bit rate for the image Lena. The improvement of TPSMVQ over ordinary SMVQ was 2.156 dB for the same bit rate for the image Lena. Table 2 shows the results before and after the second pass. Table 3 shows the number of blocks encoded for each image in the second pass. The derailment rate and mean-square error of the blocks in the first pass are shown in Table 4. The derailment rate is defined as the percentage of codewords selected by SMVQ that are not the best codewords in the super codebook for the input vectors. A good SMVQ which can obtain the exact codewords should have a lower derailment rate. We find that the derailment rate of complex blocks is much higher than that of smooth ones. Also, the mean-square error of complex blocks is much higher than that of smooth ones. Hence, the quality of the encoded image can be improved by the second pass encoding of the complex blocks. Table 5 shows the result of TPSMVQ using the backward scanning method in the second pass. Compared with TPSMVQ, the improvement is up to 0.932 dB for nearly the same bit rate for the image Peppers.

Table 1. The bit rate (bit/pixel) and PSNR (dB) of TPSMVQ, ordinary SMVQ, and ordinary VQ.

Inside/ outside	Image 512×512	TPSMVQ		Ordinary AMVQ		Ordinary VQ	
		PSNR (dB)	Bit-Rate (bit/pixel)	PSNR (dB)	Bit-Rate (bit/pixel)	PSNR (dB)	Bit-Rate (bit/pixel)
Inside	Lena	30.633	0.250	28.477	0.250	27.168	0.250
Inside	Lenna	28.579	0.259	27.507	0.250	26.589	0.250
Inside	Barbara	24.547	0.259	23.984	0.250	23.058	0.250
Inside	Boat	28.302	0.251	27.057	0.250	25.504	0.250
Inside	Toys	29.677	0.250	27.331	0.250	26.012	0.250
Outside	Peppers	29.604	0.258	27.247	0.250	27.021	0.250
Outside	Airplane	29.195	0.249	27.717	0.250	24.953	0.250
Iytsude	Zelda	32.776	0.246	31.626	0.250	29.108	0.250

Table 2. The results before and after the second pass.

Inside/outside	Image 512×512	First Pass	Second Pass	Improvement
		PSNR (dB)	PSNR (dB)	PSNR (dB)
Inside	Lena	28.676	30.633	1.957
Inside	Lenna	27.448	28.579	1.131
Inside	Barbara	23.655	24.547	0.892
Inside	Boat	26.953	28.302	1.349
Inside	Toys	27.282	29.677	2.395
Outside	Peppers	27.784	29.604	1.820
Outside	Airplane	27.817	29.195	1.378
Outside	Zelda	31.621	32.776	1.155

Table 3. The number of blocks encoded in the second pass.

Inside/outside	Image 512×512	Number of Blocks
Inside	Lena	1017
Inside	Lenna	1026
Inside	Barbara	1528
Inside	Boat	1264
Inside	Toys	1075
Outside	Peppers	1194
Outside	Airplane	1159
Outside	Zelda	684

Table 4. The derailment rate and mean-square error of the blocks in the first pass.

Inside/ outside	Image 512×512	Derailment Rate		Mean-Square Error	
		complex	smooth	complex	smooth
Inside	Lena	36.13	8.25	338.802	35.778
Inside	Lenna	37.59	19.04	328.786	74.359
Inside	Barbara	48.54	31.84	520.339	222.501
Inside	Boat	43.90	16.46	428.010	67.326
Inside	Toys	34.49	11.77	437.905	46.678
Outside	Peppers	31.78	8.71	322.269	44.156
Outside	Airplane	42.11	9.74	415.753	44.131
Outside	Zelda	20.01	5.64	136.607	20.113

Table 5. The bit rate (bit/pixel) and PSNR (dB) of TPSMVQ and TPSMVQ using the backward scanning method in the second pass.

Inside/ outside	Image 512×512	TPSMVQ		Backward-TPSMVQ	
		PSNR (dB)	Bit-Rate (bit/pixel)	PSNR (dB)	Bit-Rate (bit/pixel)
Inside	Lena	30.633	0.250	30.729	0.250
Inside	Lenna	28.579	0.259	28.798	0.262
Inside	Barbara	24.547	0.259	24.705	0.265
Inside	Boat	28.302	0.251	28.455	0.252
Inside	Toys	29.677	0.250	29.948	0.252
Outside	Peppers	29.604	0.258	30.536	0.259
Outside	Airplane	29.195	0.249	29.304	0.250
Outside	Zelda	32.776	0.246	32.808	0.246

Fig. 6 shows typical results of the image Lena: 27.168 dB at 0.250 bit per pixel using ordinary VQ, 28.477 dB at 0.250 bit per pixel using SMVQ, and 30.633 dB at 0.251 bit per pixel using TPSMVQ. Fig. 7 shows the results of the image Peppers: 29.604 dB at 0.258 bit per pixel using TPSMVQ and 30.536 dB at 0.259 bit per pixel using the backward scanning method in the second pass of TPSMVQ.

5. CONCLUSIONS

In this paper, a new technique called TPSMVQ has been proposed for encoding still images. The encoding method is divided into two passes in our TPSMVQ. In the first pass, TPSMVQ selects a state codebook to encode the input vector according to its characteristics. The second pass of TPSMVQ can improve the quality of image encoded in the first pass. Not

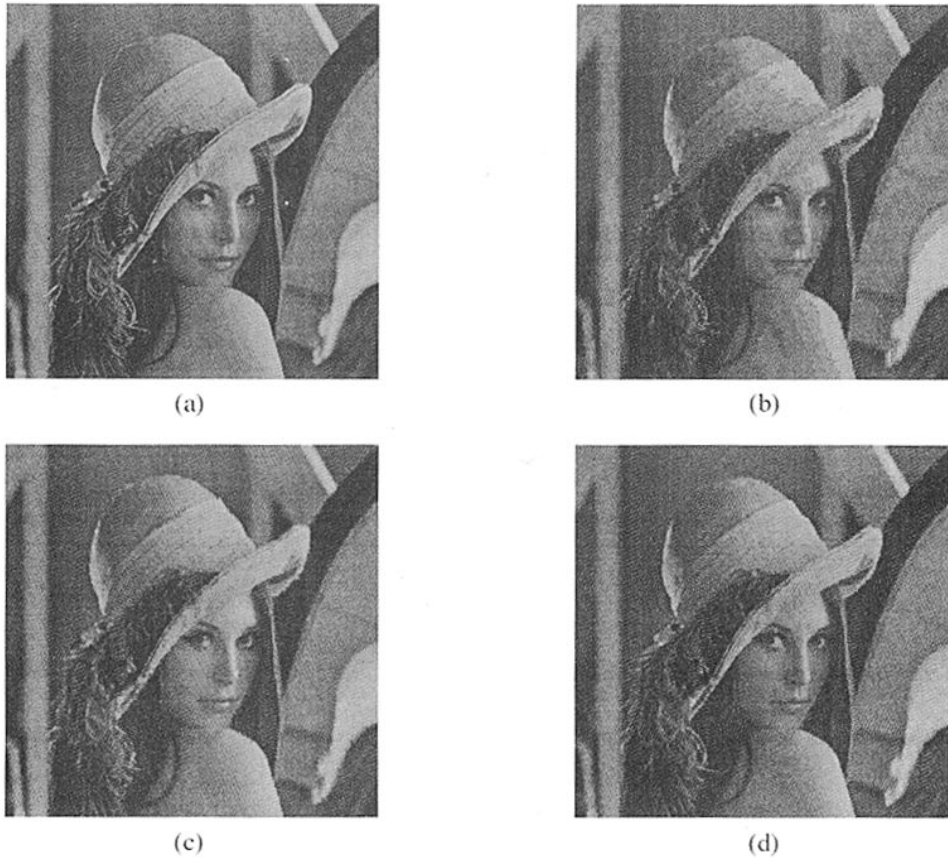


Fig. 6. (a) The original image Lena. (b) The encoded image Lena using the ordinary VQ at 0.250 bit/pixel. (c) The encoded image Lena using the ordinary SMVQ at 0.250 bit/pixel. (d) The encoded image Lena using the TPSMVQ at 0.250 bit/pixel.

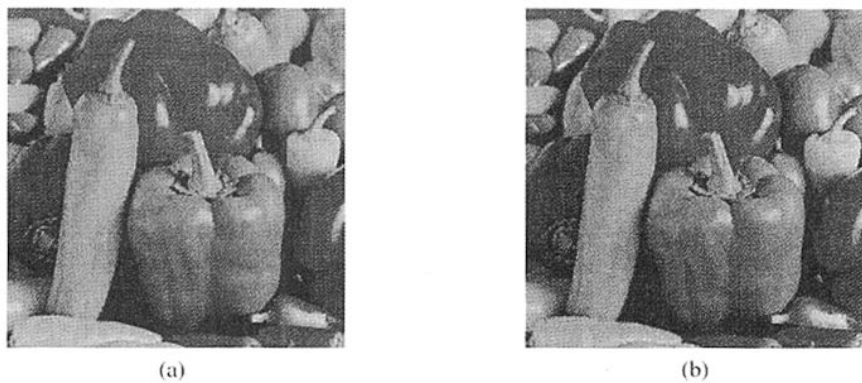


Fig. 7. (a) The encoded image Peppers using TPSMVQ at 0.258 bit/pixel with PSNR 29.604 dB. (b) The encoded image Peppers using TPSMVQ with the backward scanning method at 0.259 bit/pixel with PSNR 30.536, which is about 0.932 dB better than TPSMVQ in PSNR.

only the correlations of two neighboring blocks, but also those of the other two neighboring blocks are used to predict the state codebook in this step. Also noteworthy noticeable thing is that different scanning methods can also be considered when encoding images. According to the discussion in the above sections, we conclude that our TPSPMVQ algorithm is a highly efficient and effective image compression scheme. We also find that it is superior to the ordinary SMVQ and VQ.

REFERENCES

1. R. M. Gray, "Vector quantization," *IEEE Acoustics, Speech and Signal Processing Magazine*, 1984, pp. 4-29.
2. M. Goldberg, P. R. Boucher and S. Shlien, "Image compression using adaptive vector quantization," *IEEE Transactions on Communication*, Vol. 34, No. 2, 1986, pp. 180-187.
3. N. M. Nasrabadi and R. A. King, "Image coding using vector quantization: A review," *IEEE Transactions on Communication*, Vol. 36, No. 8, 1988, pp. 957-971.
4. H. M. Hang and B. G. Haskell, "Interpolative vector quantization of color images," *IEEE Transactions on Communication*, Vol. 36, No.4, 1986, pp. 465-470.
5. Y. Linde, A. Buzo and R. M. Gray, "An algorithm for vector quantizer design," *IEEE Transactions on Communication*, Vol. 28, No. 1, 1980, pp. 84-95.
6. M. O. Dunham and R. M. Gray, "An algorithm for the design of label-transition finite-state vector quantizers," *IEEE Transactions on Communication*, Vol. 33, No. 1, 1985, pp. 83-89.
7. J. Forster, R. M. Gray and M. O. Dunham, "Finite-state vector quantization for waveform coding," *IEEE Transactions on Information Theory*, Vol. 31, No.3, May 1985, pp. 348-359.
8. R. Aravind and A. Gersho, "Low-rate image coding with finite-state vector quantization," in *Proceeding of International Conference on Acoustics, Speech and Signal Processing*, 1986, pp. 137-140.
9. R. Aravind and A. Gersho, "Image compression based on vector quantization with finite memory," *Optical Engineering*, Vol. 26, No.7, 1987, pp. 570-580.
10. T. Kim, "Side match and overlap match vector quantizers for images," *IEEE Transactions on Image Processing*, Vol. 1, No. 2, 1992, pp. 170-185.
11. Y. Hussian and N. Farvardin, "Variable-rate finite-state vector quantization of images," in *Proceeding of International Conference on Acoustics, Speech and Signal Processing*, 1991, pp. 2301-2304.
12. N. M. Nasrabadi, C. Y. Choo and Y. Feng, "Dynamic finite-state vector quantization of digital images," *IEEE Transactions on Communication*, Vol. 42, No. 5, 1994, pp. 2145-2154.
13. Y. S. Ho and A. Gersho, "Variable-rate multistage vector quantization for image coding," in *Proceeding of International Conference on Acoustics, Speech and Signal Processing*, 1988, pp. 1156-1159.
14. E. Daly and T. R. Hsing, "Variable bit rate vector quantization of video images for packet-switched networks," in *Proceeding of International Conference on Acoustics, Speech and Signal Processing*, 1988, pp. 1160-1163.
15. Y. S. Ho and A. Gersho, "Variable-rate contour-based interpolative vector quantization for image coding," in *Proceeding of Global Communication Conference*, 1988, pp. 750-754.

16. R. F. Chang and W. T. Chen, "Image coding using variable-rate side-match finite-state vector quantization," *IEEE Transactions on Image Processing*, Vol. 2, No. 1, 1993, pp. 104-108.
17. R. F. Chang and W. M. Chen, "Adaptive quadtree-based side-match finite-state vector quantization," in *Proceeding of SPIE Conference Visual Communication Image Processing*, 1994, pp. 377-388.
18. R. F. Chang and W. M. Chen, "Adaptive edge-based side-match finite-state classified vector quantization with quadtree map," *IEEE Transactions on Image Processing*, Vol. 5, No. 2, 1996, pp. 378-383.

Ruey-Feng Chang (張瑞峰) was born in Taichung, Taiwan, on August 25, 1962. He received the B.S. degree in electrical engineering from National Cheng Kung University, Taiwan, Republic of China, in 1984, and the M.S. degree in computer and decision science and the Ph.D. degree in computer science from National Tsing Hua University, Taiwan, Republic of China, in 1988 and 1992, respectively. He is currently an Associate Professor of the Department of Computer Science and Information Engineering, National Chung Cheng University, Taiwan, Republic of China. His research interests include still image coding, video sequence coding, and packet video. Dr. Chang is a member of IEEE, ACM, SPIE, and Phi Tau Phi.

Wen-Jia Kuo (郭文嘉) was born in Changhua, Taiwan, on April 30, 1971. He received the B.S. degree in applied mathematics from National Chung Hsing University, Taiwan, Republic of China, in 1993, and the M.S. degree in computer science and information engineering from National Chung Cheng University, Taiwan, Republic of China, in 1995, respectively. He is currently a Ph.D. student of the Department of Computer Science and Information Engineering, National Chung Cheng University, Taiwan, Republic of China. His research interests include image processing, image coding, and computer network.