GRAY PREDICTION MODEL BASED FINITE STATE VECTOR QUANTIZATION OF IMAGES

Hsuan T. Chang, Ting C. Chang,[†] Yuan B. Chen,[‡] and Yann H. Pan

Department of Electrical Engineering, National Yunlin University of Science and Technology, Touliu Yunlin, 640 Taiwan

[†]*Department of Management Information System, Ling Tung College, Taichung, 408 Taiwan*

[‡]*Department of Electronic Engineering, Chienkuo Technology University, Changhua, 500 Taiwan*

Abstract

Gray theory recently has received a great deal of attention because it has been successfully applied to many disciplines. In this paper, a new class of finite-state vector quantizers (FSVQs) whose next-state function design is based on the gray prediction model (GPM), named the GPMVQs, is proposed. In GPMVQs, the GPM with a single variable and a first-order differential equation is used to design the next-state function for constructing the state codebooks in FSVQs. Four pixels in each row or column of previously coded blocks are the inputs of the GPM and then the corresponding output predicts a pixel in the border of current block. With the western and northern blocks of the current block, seven border pixels are predicted and then used to choose the codewords in the super codebook to construct the state codebook. Simulation results show that the proposed GPMVQs can outperform the side-match vector quantizers, especially for the less-complicated images.

Keywords: gray theory, finite-state vector quantization, gray prediction model, next-state function.

September 7, 2004

1 Introduction

Vector quantization (VQ) is an efficient technique for image compression, especially for the applications of low bit rate coding. It has been dramatically developed since its invention in 1984 [1]. Recently, the finite-state vector quantizers (FSVQs) [2] was proposed to improve the ordinary/memoryless VQ techniques. Figure 1 shows the block diagram of typical FSVQs. Given a vector of dimension k which is indexed by i, the encoder maps a k-dimensional source vector $\underline{x}_i$ into a channel symbol (the index of codeword) q_i from a finite alphabet. Here each channel symbol q_i corresponds to a codeword $\underline{\hat{x}}_i$ in a selected state codebook SC_i. The previously coded blocks $\underline{\hat{x}}_{i-1}, \underline{\hat{x}}_{i-2}, \ldots, \underline{\hat{x}}_{i-j}$ are buffered and used in the next-state function to determine the state s_i of the current block. In the decoder of an FSVQ, all the components are identical to those of the encoder of an FSVQ. If we assume that the channel is noiseless and the initial state of the encoder is known in the decoder, the next-state function of the decoder can keep track of the encoder states. Therefore, the reproduction vectors produced by the decoder are identical to the corresponding reproduction vectors determined by the encoder. Since the codeword for an input vector is found in the state codebook, which is much smaller than the super codebook, the encoding time and bit rate are reduced. Compared with the ordinary VQ, the FSVQs can achieve a higher compression ratio while the image quality is negligibly degraded.

There are several design algorithms of FSVQs [2, 3]. It seems that most of the algorithms are difficult to handle spatial contiguity in images. In attempt to force the encoder to optimize edge contiguity across block boundaries without doing explicit edge detection, Kim proposed the side match vector quantizers (SMVQs) and overlap match vector quantizers (OMVQs) [4] which are two classes of FSVQs. The performances of SMVQs and OMVQs schemes are better than or comparable with that of ordinary VQ. It can be drastically improved when they are used together with variable length noiseless coding. In SMVQs, the boundary pixels neighboring to the northern and western blocks are supposed to be equal. In reality, this criterion is not always satisfied. It only considers the correlation between two adjacent pixels, in which the information may not be enough for prediction in an active area. If more pixels can be used for prediction, better result can be expected.

Gray theories are proposed by Den in 1982 [6] and have been widely and successfully applied to many applications such as economics, geography, weather, and automatic control [7-10]. There are two major types of applications: (1) short term prediction based on gray model (GM) and (2) multi-purpose decision based on gray relationships. Instead of forming a knowledge base, the GM constructs some differential equations to characterize the controlled system behavior. The GM with a single variable and a first-order differential equation, abbreviated by GM(1,1), predicts the future behavior of the system by using only a few past output data and solving the differential equations. The GM(1,1) has been successfully applied to the motion estimation issue for encoding video data [11] and outperforms other search algorithms. The chip implementation of this algorithm also has been proposed recently [12].

In conventional SMVQs, only the boundary pixels of the previously coded blocks are used for prediction. However, those pixels may not be sufficient to represent the previous blocks. In order to obtain better prediction on the border pixels in the current block, here we propose a gray-theory-based prediction model, which uses four pixels in the same row or column in previously coded blocks to predict the pixels in the border of current block. This scheme is called the GPM-based VQs (GPMVQs) and is considered as a new class of FSVQs. Simulation results show that the proposed gray model is an efficient method in designing the next-state function for FSVQs. The coding performance of the proposed scheme is superior to SMVQs, especially for less complicated images.

2 Gray Prediction Model

In this section we briefly review the GM(1,1) and discuss how to apply GM(1,1) to the next-state function design in FSVQs. First of all, the steps of mathematics analysis for GM(1,1) are shown below: Let $\mathbf{x}^{(0)}$=$\{x^{(0)}(1)$, $x^{(0)}(2)$, $\ldots$, $x^{(0)}(n)\}$ be an original data sequence which contains n data, where $x^{(0)}(i)$ represents the i^{th} system output. The new sequence $\mathbf{x}^{(1)}$=$\{x^{(1)}(1)$, $x^{(1)}(2)$, $\ldots$, $x^{(1)}(n)\}$ is generated with a one-stage accumulated generating operation (AGO) by

$$\text{AGO}\{\mathbf{x}^{(0)}\} = \mathbf{x}^{(1)} = \{\sum_{k=1}^{1} x^{(0)}(k), \sum_{k=1}^{2} x^{(0)}(k), \ldots, \sum_{k=1}^{n} x^{(0)}(k)\}. \tag{1}$$

According to GM(1,1), we have the first-order difference equation $x^{(0)}(k) + az^{(1)}(k) = b$, and thus

$$x^{(0)}(k) = -az^{(1)}(k) + b, \qquad \text{for} \quad k = 1, 2, 3, \ldots, n \tag{2}$$

where a and b are the parameters of GM(1,1) that we want to determine and

$$z^{(1)}(k) = \alpha x^{(1)}(k) + (1-\alpha)x^{(1)}(k-1), \quad \text{for} \quad k \geq 2, \quad \alpha \in [0, 1]. \tag{3}$$

The factor α here is usually given as 0.5 [6], which means that $z^{(1)}(k)$ is affected by $x^{(1)}(k)$ and $x^{(1)}(k-1)$ equally. In certain cases, α values can be varied for different k to pursuit higher predicting accuracy [13, 14].

The matrix form of Equation 2 is represented by

$$X_N = Z\hat{a}, \tag{4}$$

where

$$X_N = \begin{bmatrix} x^{(0)}(2) \\ x^{(0)}(3) \\ x^{(0)}(4) \\ \vdots \\ x^{(0)}(n) \end{bmatrix}, \quad Z = \begin{bmatrix} -z^{(1)}(2) & 1 \\ -z^{(1)}(3) & 1 \\ -z^{(1)}(4) & 1 \\ \vdots & \\ -z^{(1)}(n) & 1 \end{bmatrix}, \quad \hat{a} = \begin{bmatrix} a \\ b \end{bmatrix}. \tag{5}$$

The values of a and b can be derived via the least square criterion as,

$$\begin{bmatrix} a \\ b \end{bmatrix} = (Z^T Z)^{-1} Z^T X_N. \tag{6}$$

That is,

$$a = \frac{\sum_{k=2}^{n} z^{(1)}(k) \sum_{k=2}^{n} x^{(0)}(k) - (n-1) \sum_{k=2}^{n} z^{(1)}(k) x^{(0)}(k)}{(n-1) \sum_{k=2}^{n} [z^{(1)}(k)]^2 - [\sum_{k=2}^{n} z^{(1)}(k)]^2} \tag{7}$$

and

$$b = \frac{\sum_{k=2}^{n} [z^{(1)}(k)]^2 \sum_{k=2}^{n} x^{(0)}(k) - \sum_{k=2}^{n} z^{(1)}(k) \sum_{k=2}^{n} z^{(1)}(k) x^{(0)}(k)}{(n-1) \sum_{k=2}^{n} [z^{(1)}(k)]^2 - [\sum_{k=2}^{n} z^{(1)}(k)]^2}. \tag{8}$$

When the values a and b are derived, we substitute them into the solution of Equation 2 and obtain the general solution as follows:

$$\hat{x}^{(1)}(k) = [x^{(0)}(1) - \frac{b}{a}] e^{-a(k-1)} + \frac{a}{b}, \tag{8}$$

in which $\hat{x}^{(0)}(1) = x^{(1)}(1)$, and hence the predicted values, $\hat{x}^{(0)}(k)$ for $k \geq 1$, can be calculated by using the inverse AGO. That is,

$$\hat{x}^{(0)}(k) = \hat{x}^{(1)}(k) - \hat{x}^{(1)}(k-1) = (1 - e^{a})[x^{(0)}(1) - \frac{b}{a}] e^{-a(k-1)}, \quad \text{for} \;\; k \geq 2. \tag{9}$$

Therefore, we can predict the values $\hat{x}^{(0)}(k)$ for $k > n$.

Basically, the major computation overhead is due to the determination of two parameters a and b in Equations (7) and (8), and is dependent on the number of data, n. Once two parameters have been derived, the calculation for the predicted values are much easier. Two factors affect the accuracy of this prediction method: the data type and how many data are used in the prediction model. Monotonically and/or gradually changed data and adequate previous data are helpful in increasing the accuracy of the gray prediction model [13].

3 GPM Based Next-State Function Design

The design of a next state function dominates the coding performance of FSVQs. Here we propose the next-state function that deals with GM(1,1). Figure 2 shows the pixels that contribute the construction of the state codebooks in the proposed GPMVQs. Similar to SMVQs, two previously encoded blocks $\boldsymbol{u}$ and $\boldsymbol{l}$ are used in the next-state function. SMVQs use the boundary pixels only, while the proposed scheme uses the all pixels in the western and northern blocks. In the northern block $\boldsymbol{u}$, four pixels in each column are the inputs of the GM(1,1) (that is, $n = 4$ in Section 2) so that the predicted output, $x(5)$, is the northern boundary pixel in current block. On the other hand, in western block $\boldsymbol{l}$, four pixels in each row are the input of the GM(1,1) and the predicted output is the western boundary pixel in the current block. Therefore, four predicted pixels at both northern and western sides are obtained and correspond to row and column states in their state spaces. Although the GM(1,1) can use more than four pixels for prediction, higher computation load is required simultaneously. Moreover, the previous coded pixels could be different from the original pixels. Thus using the distorted pixels as the inputs of the GM(1,1) could greatly decrease the accuracy on the predicted pixel. Therefore, it is not necessary to use more pixels as the inputs of the GM(1,1).

As shown in Figures 2(a) and 2(b), the pixels, $\{x_{1,j}|j = 1,2,3,4\}$ and $\{x_{i,1}|i = 1,2,3,4\}$ are predicted by the pixels $\{u_{1,j}, u_{2,j}, u_{3,j}, u_{4,j}|j = 1,2,3,4\}$ in the northern block and $\{l_{i,1}, l_{i,2}, l_{i,3}, l_{i,4}|i = 1,2,3,4\}$ in the western block using the GM(1,1), respectively. Note that the pixel $x_{1,1}$ can be the average value of two pixels predicted by the left row and the upper column. The error between the predicted pixels $\{x_{1,j}|j = 1,2,3,4\}$ and $\{x_{i,1}|i = 1,2,3,4\}$ and the corresponding pixels of every codeword in the super codebook is calculated. Then the calculated errors of all the codewords in the super codebook are sorted with an incremental order. If the GM(1,1) predicts well, the codewords with smaller errors represent better candidates for encoding the current block. The state codebooks are selected according to sizes we choose, and can be designed once the super codebook has been constructed (i.e., off-line design) to speed up the encoding and decoding processes. Consider the contents of the blocks. SMVQs use the border pixel only to predict the boundary pixel in the current block. The proposed GPMVQs employ four previous pixels and predicts the next pixel following the property in previous pixels.

The image contents may contain edges, which are abrupt changes of pixels values in images. The accuracy of GM(1,1) could greatly decrease when the input pixel values vary a lot. To avoid incorrect prediction, the input pixel values are examined. If the difference between any two consecutive values in a row or column is greater than a threshold value, then the GM(1,1) will not be employed to predict the boundary pixel. Instead, the side match algorithm is used.

4 Experimental Results

In computer simulation, six images shown in Fig. 3(a)–(f) are used to test the proposed GPMVQs. The super codebooks of various sizes are generated by applying the LBG algorithm [5] on four test images: F-16, Building, Harbour, and Peppers. Then the state codebooks are generated by the use of GM(1,1) with $\alpha = 0.5$. The proposed GPMVQs are compared with the SMVQs. Both the proposed GPMVQs and SMVQs predict only the boundary pixels. In predicting each boundary pixel, the former use four previously coded pixels and the later use only one pixel. Other classes of FSVQs, for example, the gradient-match vector quantizers [15], predict not only boundary pixels, but also the inner pixels, are not considered here. Therefore, here only the quantizers that predict the boundary pixels are compared.

The peak signal-to-noise ratio (PSNR) of a decoded image is defined as

$$\mathrm{PSNR} = 10 \log_{10} \frac{512^2 \times 255^2}{\sum \sum_{i,j=1}^{512} [f(i,j) - f'(i,j)]^2} \text{ dB.} \tag{14}$$

Tables 1 shows the rate-PSNR performance of SMVQs and GPMVQs for the six test images The super codebook size is 1024. The results for various state codebook sizes for SMVQs and GPMVQs are provided. Note that the Lena and Lady images are outside the training images, while the others are inside the training images. As shown in this table, the performance of the proposed GPMVQs is better than those of SMVQs for the smooth images such as the Lena and Lady images, especially for the cases of small state codebooks. The 1.25 dB PSNR improvement is obtained for the Lady image when the state codebook is of size 32. As shown in Figure 3(a), a lot of grayscale values on the face are gradually changed. Therefore, superior prediction results of GMP for the Lady image are obtained. On the other hand, the performance improvement for the complicated images such as the Building and Harbour images is not obvious. This indicates that applying the GM(1,1) on the next-state function design for FSVQs is more worthwhile for the less complicated images, especially for the images whose most pixel values are gradually changed. Note that for complicated image blocks, a threshold value 30 for current column/row is used to determine whether the GM(1,1) will be used to predict the boundary pixel. The simulation results for the Building and Harbour images show that the proposed GPMVQs have comparable results.

Although the proposed GPMVQs outperforms the SMVQs, the required computation increases accordingly. Fortunately, the computation required for GM(1,1) prediction can be performed when the state codebooks are off-line designed. Therefore, the encoding and decoding time of the proposed GPMVQ is comparable with that of the SMVQs. In addition to SMVQs, there are many other classes of FSVQs [3,15,17,18]. Some of them can greatly improve the coding performance for SMVQs. A special application of FSVQs on improving fractal image compression was proposed in Ref. [16]. To improve the proposed GPMVQs, the following two directions can be considered. If the accuracy of the GM(1,1) model can be improved, the coding performance of the proposed GPMVQs can be further enhanced. On the other hand, adaptive selecting the complicated blocks as the initial blocks in FSVQs [19] also has great potential to improve the proposed method.

5 Conclusion

A new class of FSVQs whose design of the next-state function is based on GM(1,1), named the GPMVQs, is proposed in this paper. Four pixels in previously coded blocks are applied to GM(1,1) to predict the border pixel in the current block. The predicted pixel follows the property in previously coded pixels and thus it is a promise prediction. The simulation results show that the proposed GPMVQs can outperform SMVQs for the less-complicated images. For complicated images, their performances are still comparable.

Acknowledgment

This research was partially supported by the National Science Council, Taiwan, under contract NSC 92-2213-E-224-047.

References

[1] R.M. Gray, "Vector quantization," *IEEE ASSP Mag.*, pp. 4–29, Apr. 1984.

[2] Gersho and R.M. Gray, *Vector Quantization and Signal Compression*, Chapter 14, Boston, MA: Kluwer, 1992.

[3] N. Nasrabadi and S. Rizvi, "Next-state functions for finite-state vector quantization," *IEEE Trans. on Image Processing*, vol. 4, no. 12, pp. 1592–1601, Dec. 1995.

[4] T. Kim, "Side match and overlap match vector quantization for images," *IEEE Trans. on Image Processing*, vol. 1, no. 2, pp. 170–185, Apr. 1992.

[5] Y. Linde, A. Buzo, and R. Gray, "An algorithm for vector quantization design," *IEEE Trans. on Communications*, vol. 28, no. 1, pp. 84–95, Jan. 1980.

[6] J. Deng, "Control problems of grey system," *System Control Letters*, vol. 5, pp. 288–294, 1982.

[7] J. Deng, "Introduction to grey system," *The Journal of Grey System*, vol. 1, pp. 1–24, 1989.

[8] Y.P. Huang and C.C. Huang, "The integration and application of fuzzy and grey modeling methods," *Fuzzy Sets and Systems*, vol. 78, no. 1, pp. 107–119, 1996

[9] Y.P. Huang and T.M. Yu, "The hybrid grey-based models for temperature prediction," *IEEE Trans. on Syst. Man, Cybern.*, vol. 27, no. 2, pp. 284–292, 1997

[10] S.-J. Huang and C.-L. Huang, "Control of an inverted pendulum using grey prediction," *IEEE Trans. on Industry Applications,* vol. 36, no. 2, pp. 452–458, 2000

[11] J.M. Jou, P.Y. Chen, and J.M. Sun, "The gray prediction search algorithm for block motion estimation," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 9, no. 6, pp. 843–848, Sep. 1999.

[12] J.M. Jou, Y.-H. Shiau, P.-Y Chen, and S.-R. Kuang, “A low-cost Gray prediction search chip for motion estimation,” *IEEE Trans. on Circuits and Systems I: Fundamental Theory and Applications,* vol. 47, no. 7, pp. 928–938, July 2002.

[13] K.L. Wen, T.C. Chang, Hsuan T. Chang, and M.L. You, “The adaptive alpha in GM(1,1) model,” *in Proceedings of 1999 IEEE International Conference on System, Man, and Cybertics*, vol. 1, Oct. 1999, pp. 304–308.

[14] K.L. Wen, T.C. Chang, Hsuan T. Chang, and M.L. You, “Inverse approach to find an optimun alpha for grey prediction model,” *in Proceedings of 1999 IEEE International Conference on System, Man, and Cybertics*, vol. 1, Oct. 1999, pp. 309–313.

[15] Hsuan T. Chang, “Gradient match vector quantizers for images,” *Optical Engineering*, vol. 39, no. 8, pp. 2046–2057, Aug. 2000.

[16] Hsuan T. Chang, “Gradient match and side match fractal vector quantizers for images,” *IEEE Trans. on Image Processing*, vol. 11, no. 1, pp. 1–9, Jan. 2002.

[17] H.-C. Wei, P.-C. Tsai, and J.-S. Wang, “Three-sided side match finite-state vector quantization,” *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 10, no. 1, pp.51–58, Jan. 2000.

[18] S. B. Yang and L. Y. Tseng, “Smooth side-match classified vector quantizer with variable block size,” *IEEE Transactions on Image Processing,* vol. 10, no. 5, pp. 677–685, May 2001.

[19] Hsuan T. Chang and Y.H. Pan, “Adaptive initial blocks for improving finite state vector quantization of images,” *Optical Engineering*, vol. 43, no.4, pp.830–837, Apr. 2004.

Table 1: The PSNR comparison between SMVQs and the proposed GPMVQs. The super codebook is of size 1024 and the state codebooks are of sizes 16, 32, 64, 128, 256, and 512.

Image			Lena	Lady	Harbour	Building	Peppers	F-16
State codebook size	16	SMVQ	**26.61**	**31.70**	**22.72**	**23.16**	**23.16**	**26.14**
		GPMVQ	**26.89**	**32.11**	**22.47**	**23.22**	**23.22**	**26.28**
		bpp	**0.256**					
	32	SMVQ	**28.49**	**33.08**	**23.51**	**24.91**	**28.84**	**27.76**
		GPMVQ	**28.84**	**34.33**	**23.29**	**24.81**	**29.05**	**28.13**
		bpp	**0.317**					
	64	SMVQ	**29.94**	**35.34**	**24.10**	**26.26**	**30.14**	**29.44**
		GPMVQ	**30.14**	**35.58**	**23.98**	**26.28**	**30.35**	**29.62**
		bpp	**0.379**					
	128	SMVQ	**31.14**	**36.37**	**24.69**	**27.52**	**31.50**	**30.89**
		GPMVQ	**31.23**	**36.67**	**24.59**	**27.49**	**31.70**	**30.81**
		bpp	**0.440**					
	256	SMVQ	**31.90**	**37.34**	**25.19**	**28.40**	**32.57**	**31.90**
		GPMVQ	**31.90**	**37.34**	**25.14**	**28.35**	**32.57**	**31.80**
		bpp	**0.502**					
	512	SMVQ	**32.22**	**37.34**	**25.44**	**28.74**	**32.82**	**32.33**
		GPMVQ	**32.22**	**37.34**	**25.44**	**28.74**	**32.82**	**32.33**
		bpp	**0.563**					

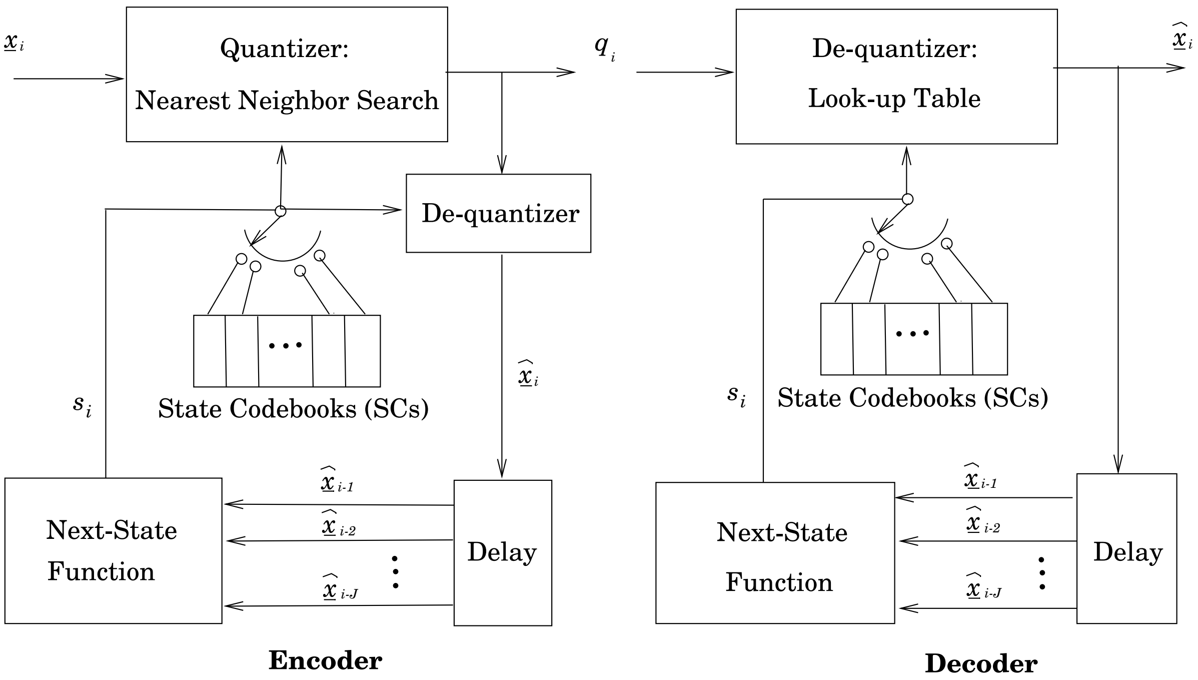
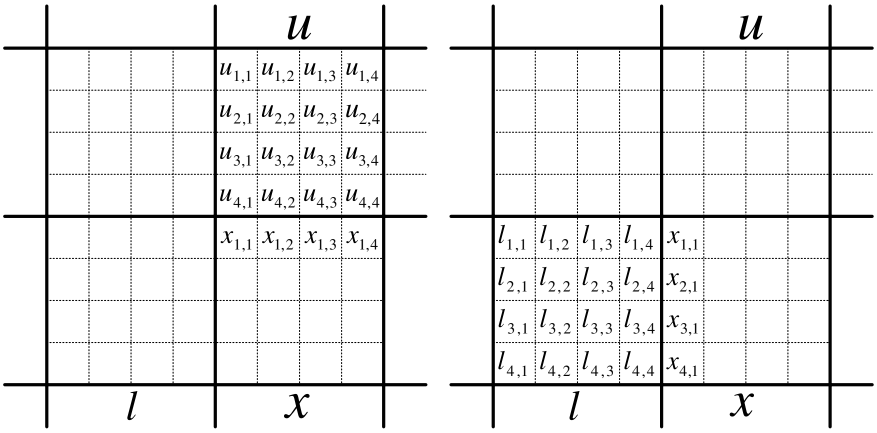
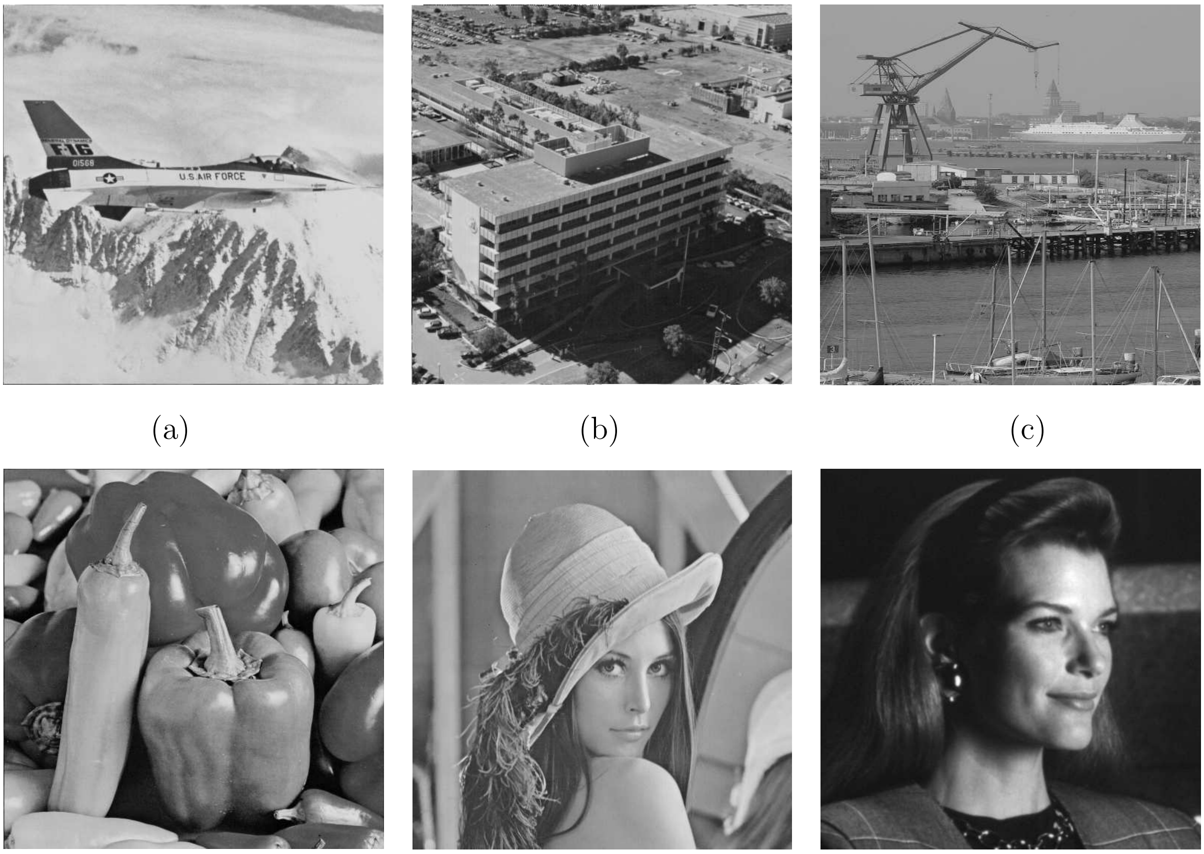

$\underline{x}_i$
Quantizer:
Nearest Neighbor Search
q_i
De-quantizer:
Look-up Table
$\widehat{\underline{x}}_i$
De-quantizer
$\widehat{\underline{x}}_i$
s_i
State Codebooks (SCs)
s_i
State Codebooks (SCs)
Next-State
Function
$\widehat{\underline{x}}_{i-1}$
$\widehat{\underline{x}}_{i-2}$
$\widehat{\underline{x}}_{i-J}$
Delay
Next-State
Function
$\widehat{\underline{x}}_{i-1}$
$\widehat{\underline{x}}_{i-2}$
$\widehat{\underline{x}}_{i-J}$
Delay
Encoder
Decoder

Figure 1: The block diagram of FSVQ.


u
u_{1,1} u_{1,2} u_{1,3} u_{1,4}
u_{2,1} u_{2,2} u_{2,3} u_{2,4}
u_{3,1} u_{3,2} u_{3,3} u_{3,4}
u_{4,1} u_{4,2} u_{4,3} u_{4,4}
x_{1,1} x_{1,2} x_{1,3} x_{1,4}
l
x
u
l_{1,1} l_{1,2} l_{1,3} l_{1,4} x_{1,1}
l_{2,1} l_{2,2} l_{2,3} l_{2,4} x_{2,1}
l_{3,1} l_{3,2} l_{3,3} l_{3,4} x_{3,1}
l_{4,1} l_{4,2} l_{4,3} l_{4,4} x_{4,1}
l
x

(a) (b)

Figure 2: The pixels in previously coded blocks are used to estimate the boundary pixels of current block $\boldsymbol{x}$: (a) for the northern block $\boldsymbol{u}$ and (b) for the western block $\boldsymbol{l}$.



(a) (b) (c)

(d) (e) (f)

Figure 3: The test images: (a) F-16, (b) Building, (c) Harbour, (d) Peppers, (e) Lena, and (f) Lady.