# Feature-Preserving Interpolative Vector Quantization

**Wu-Ja Lin[1, 2], Wei-Yu Han[1], Yen-Po Lee[1], and Kuang-Shyr Wu[1]**

[1]Department of Computer Science and Information Engineering, Ching Yun University
229, Chien-Hsin Rd., Jung-Li, Taiwan 320, R.O.C., [2]Email: wj.lin@cyu.edu.tw

## Abstract

In this paper, a feature-preserving interpolative vector quantization method is proposed to compress images. The proposed method preserves the average color and color contrast on generating the approximation image and preserves mean, variance, and average radius on quantizing the residual image which is generated by subtracting the approximation image from the source image. The experimental results show that the proposed method usually yields satisfactory results.

**Keywords**: feature-preserving, interpolative vector quantization, mean, variance, average radius.

## 1. Introduction

In the last two decades, vector quantization (VQ) [1-4] has been successfully used in image compression. It is a simple coding scheme and can be easily implemented, especially for the decoder. To encode an image using VQ, every block of the image is represented as a vector and these vectors are quantized into some representative vectors. These representative vectors form a so-called codebook. Then, for each image block, the encoder sends the index of the closest representative vector in the codebook. The decoder decodes the image with the indexed representative vectors. The key to obtain a high quality image is to choose a good codebook. The LBG method [5] proposed by Linde, Buzo, and Gray is a widely used VQ algorithm. It can find a good codebook if a good initial codebook is used.

Instead of applying VQ directly to the source images, Baker and Gray propose a Mean Residual VQ (MRVQ) algorithm [6-7] which removes the block means before quantizing vectors. (The encoder has to transmit the block mean and the indices via a PCM or DPCM encoder.) The MRVQ has been shown to be better than the direct VQ [8]. On the other hand, Hang and Haskell proposed the interpolative VQ (IVQ) [9] which decomposed an image into a pair of low frequency and high frequency images. The low frequency image is obtained by bilinear interpolation and the high frequency image (also called the residual image) is obtained by subtracting the low frequency image from the original image. The residual image was quantized further by the LBG algorithm. The encoder that Hang and Haskell designed sent the intensity values of some pixels of the ``original" image (in order to construct the interpolated image) and the codebook indices. The decoder decoded the image by adding the interpolated image (reconstructed by the received intensity values and the bilinear function) to the quantized residual image (built with the received codebook indices). The IVQ algorithm could obtain good encoded image at low bit-rate.

In this paper, we follow the strategy of IVQ to compress images. We first use an approximation algorithm called Mean-Variance-Preserving (MVP) approximation algorithm [10] to obtain the low frequency image. Then, we apply the Feature-Preserving VQ (FPVQ), which is an application of our analytical clustering method [11], to quantize the residual image. The system diagrams of our encoder and decoder are illustrated in Figures 1 and 2, respectively. The idea of the MVP approximation algorithm is to keep image average color and color contrast. The philosophy of the FPVQ, on the other hand, is to preserve vector mean, variance, and average radius. Both the MVP approximation and the FPVQ method are accomplished using explicit (analytical) formulas.

The organization of this paper is as follows. The formulas provided by the MVP approximation algorithm is introduced in Section 2, and the FPVQ formulas are presented in Section 3. Section 4 shows the experimental results. The conclusion is presented in Section 5.

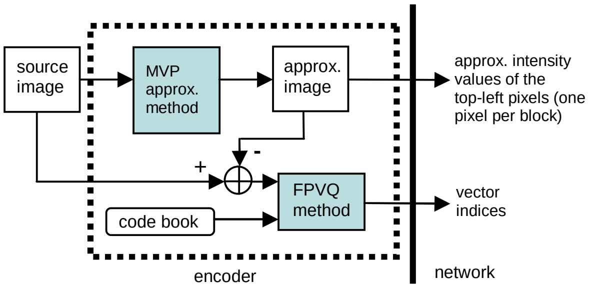

Fig. 1: Encoder of the proposed method.

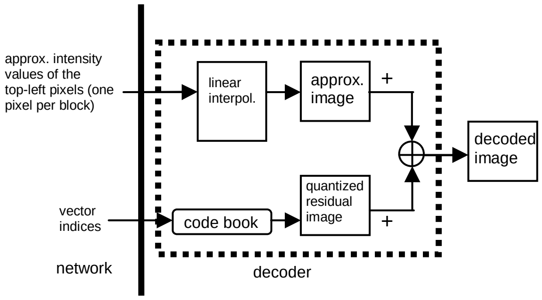

Fig. 2: Decoder of the proposed method

## 2. Mean-Variance-Preserving (MVP) Approximation Algorithm

Before generating the low frequency (approximation) images, we first transform the color space from RGB to YIQ. Then, we apply our approximation algorithm, separately, to the Y, I, and Q images. By combining the approximated Y, I, Q images and transforming from YIQ to RGB, we can generate our approximated image. Let *S* denote Y, I, or Q; and $\hat{S}$ the approximation of *S*. We first divided *S* into $G \times G$ blocks (the block size is $W \times W$). Let $f_{i,j}(1,1)$ denote the intensity value of the top-left pixel of block *(i, j)* in *S* ( $1 \le (i,\ j) \le G$ ); and $\hat{f}_{i,j}(1,1)$ the corresponding intensity value of $\hat{S}$ . The MVP algorithm first calculates $\hat{f}_{i,j}(1,1)$ , $1 \le i,\ j \le G$ , for $\hat{S}$ , and then uses the calculated $\{\hat{f}_{i,j}(1,1)\}_{1 \le i,j \le G}$ and the linear interpolation to fill in the intensity values of the remaining pixels of $\hat{S}$ . We summarize below the overall process of the MVP algorithm:

Step 1. Calculate $\hat{f}_{i,j}(1,1)$ , $1 \le i,\ j \le G$ , by requiring that the { $\hat{f}_{i,j}(1,1)$ , ..., $\hat{f}_{i,j}(1,W)$ ; $\hat{f}_{i,j+1}(1,1)$ } in $\hat{S}$ preserve the mean *m* and variance $\sigma^2$ of the { $f_{i,j}(1,1)$ , ..., $f_{i,j}(1,W)$ ; $f_{i,j+1}(1,1)$ } in *S*.

Step 2. Obtain $\hat{f}_{i,j}(1,x)$ and $\hat{f}_{i,j}(x,1)$ , where $x \in \{2,\dots,W\}$ by linear interpolation, i.e., using the line segment connecting $\hat{f}_{i,j}(1,1)$ and $\hat{f}_{i,j+1}(1,1)$ , and using the line segment connecting $\hat{f}_{i,j}(1,1)$ and $\hat{f}_{i+1,j}(1,1)$ .

Step 3. Calculate $\hat{f}_{i,j}(r,s)$ , $2 \le r,\ s \le W$ , using the line segment connecting $\hat{f}_{i,j}(1,s)$ and $\hat{f}_{i+1,j}(1,s)$ .

Let $\hat{f}(x) = \hat{f}_{i,j}(1,x),\ x = 1,\dots,W$ , and $\hat{f}(W+1) = \hat{f}_{i,j+1}(1,1)$ . The formulas to calculate $\hat{f}(1)$ and $\hat{f}(W+1)$ are

$$\hat{f}(1) = 2m - \hat{f}(W+1), \tag{1}$$

$$\hat{f}(W+1) = m \pm \sqrt{\frac{3W}{W+2}\sigma^2}. \tag{2}$$

To determine the sign of Equation (2), we can examine the sign of

$$f(W+1) - f(1), \tag{3}$$

where $f(1) = f_{i,j}(1,1)$ and $f(W+1) = f_{i,j+1}(1,1)$. If the sign of Equation (3) is positive, we choose a plus sign for Equation (2); otherwise, we choose a minus sign for Equation (2).

## 3. Feature-Preserving Vector Quantization (FPVQ)

In this section, we introduce how the FPVQ algorithm is used to generate a codebook and quantize the residual image. (Note that a residual image is generated by subtracting the approximated image generated by the MVP algorithm from the original image.) Suppose there are *h* training residual images, each residual image contains $G \times G$ blocks and the block size is $W \times W$. Each block is treated as a vector; therefore, we have an *N*-point *d*-dimensional data set $T = \{(x_1^i,\dots,x_d^i)\}_{i=1}^N$, where *N*= $h \times G \times G$ and *d*= $W \times W$ . To generate a codebook for *T*, our FPVQ algorithm hierarchically splits *T*. If the codebook is expected to have *k* codewords, then there will be *k-1* bisections. The *k* representative vectors of the *k* obtained clusters form the codebook. For each bisection, the FPVQ will preserve the mean, variance, and the average radius of the vector set being bisected. Without the loss of generality, we only show the first one of the *k-1* bisections. In other words, we only show how to bisect the *N*-point *d*-dimensional data set *T* into two subsets $T_A$ and $T_B$ . Let $C_A = \{x_1^A,\dots,x_d^A\}$ and $C_B = \{x_1^B,\dots,x_d^B\}$ denote the cluster representatives of $T_A$ and $T_B$ ; and $p_A$ and $p_B$ the corresponding population ratios. To make $C_A$ and $C_B$ (weighted by $p_A$ and $p_B$ , respectively) preserve the mean, variance, and average radius of *T*, we require that

$$p_A + p_B = 1 = 100\,\% \tag{4}$$

$$\begin{cases} p_A x_1^A + p_B x_1^B = \dfrac{1}{N}\sum_{i=1}^{N} x_1^i = \overline{x_1} \\ \vdots \\ p_A x_d^A + p_B x_d^B = \dfrac{1}{N}\sum_{i=1}^{N} x_d^i = \overline{x_d} \end{cases} \tag{5}$$

$$\begin{cases} p_A(x_1^A - \overline{x_1})^2 + p_B(x_1^B - \overline{x_1})^2 = \dfrac{1}{N}\sum_{i=1}^{N}(x_1^i - \overline{x_1})^2 = \overline{\Delta x_1^2} \\ \vdots \\ p_A(x_d^A - \overline{x_d})^2 + p_B(x_d^B - \overline{x_d})^2 = \dfrac{1}{N}\sum_{i=1}^{N}(x_1^i - \overline{x_d})^2 = \overline{\Delta x_d^2} \end{cases} \tag{6}$$

$$\begin{aligned} & p_A\sqrt{(x_1^A - \overline{x_1})^2 + \dots + (x_d^A - \overline{x_d})^2} \\ & + p_B\sqrt{(x_1^B - \overline{x_1})^2 + \dots + (x_d^B - \overline{x_d})^2} \\ & = \frac{1}{N}\sum_{i=1}^{N}\sqrt{(x_1^i - \overline{x_1})^2 + \dots + (x_d^i - \overline{x_d})^2} = \frac{1}{N}\sum_{i=1}^{N}\Delta r_i = \overline{\Delta r} \end{aligned} \tag{7}$$

where the mean $(\overline{x_1},\ldots,\overline{x_d})$, variance $(\overline{\Delta x_1^2},\ldots,\overline{\Delta x_d^2})$, and average radius $\overline{\Delta r}$ are all of known values. Note that Equation (4) is a natural requirement. After certain derivation, we can obtain the formulas to calculate $p_A$ and $p_B$, namely,

$$p_A = \frac{1}{2}+\frac{1}{2}\sqrt{1-\frac{\overline{\Delta r}^2}{\overline{\Delta r^2}}} = \frac{1}{2}+\frac{1}{2}\sqrt{1-\frac{\overline{\Delta r}^2}{\overline{\Delta x_1^2}+\ldots+\overline{\Delta x_d^2}}} \tag{8}$$

$$p_B = 1-p_A \tag{9}$$

Note that Equations (8) and (9) will always yield real value (by Schwartz's inequality). The formulas to calculate $C_A=\{x_1^A,\ldots,x_d^A\}$ and $C_B=\{x_1^B,\ldots,x_d^B\}$ are

$$\begin{aligned} x_1^A &= \overline{x_1} \pm \sqrt{\frac{p_B}{p_A}\overline{\Delta x_1^2}}, \\ &\vdots \\ x_d^A &= \overline{x_d} \pm \sqrt{\frac{p_B}{p_A}\overline{\Delta x_d^2}}, \end{aligned} \tag{10}$$

and

$$\begin{aligned} x_1^B &= \frac{\overline{x_1}-p_A x_1^A}{p_B}, \\ &\vdots \\ x_d^B &= \frac{\overline{x_d}-p_A x_d^A}{p_B}. \end{aligned} \tag{11}$$

The ± signs in Equation (10) can be determined as follows:

Step 1. Choose from $\{x_1,\ldots,x_d\}$ a data dimension which yields the maximal variance. Without the loss of generality, let $x_p$ be the chosen dimension, i.e. $\overline{\Delta x_p^2}=\max\{\overline{\Delta x_1^2},\ldots,\overline{\Delta x_d^2}\}$. Check whether most of the data points $\{(x_1^i,\ldots,x_d^i)\}_{i=1}^N$ in $T$ have positive $x_p^i-\overline{x_p}$. If this is the case, use the plus sign in Equation (10) to evaluate $x_p^A$; otherwise, use the minus sign for $x_p^A$.

Step 2. As for the signs of the remaining dimensions of $C_A$, $x_j^A(j=1,\ldots,d,j\neq p)$, check if most of the points $\{(x_1^i,\ldots,x_d^i)\}_{i=1}^N$ in $T$ yield positive product $(x_p^i-\overline{x_p})(x_j^i-\overline{x_j})$. If yes, set the sign of $x_j^A$ in Equation (10) identical to the sign of $x_p^A$; otherwise, set the sign of $x_j^A$ opposite to the sign of $x_p^A$.

To bisect *T*, we use the hyperplane perpendicular to the line segment connecting $C_A$ and $C_B$, and passing through the mean vector $m=(\overline{x_1},\ldots,\overline{x_d})$ of *T*.

Applying the above bisection method for *k-1* times, we obtain *k* clusters. The *k* representative vectors of the *k* clusters form the codebook. When the codebook is generated, we quantize every block of the residual image with the nearest codeword in the codebook and encode that block with the index of the codeword. The quantized residual image can be reconstructed on the other end of the network with the vector indices and the codebook.

# 4. Experiment Results

Four experiments were done for the proposed Feature-Preserving interpolative vector quantization. The images that we used are Lena, Jet, Peppers, and Tiffany. The image size is $512\times512$. Figure 3 and Figure 4 show the images obtained by the JPEG [12] algorithm and the proposed method (i.e., MVP approximation followed by the FPVQ). The block sizes that we used in our MVP approximation were $8\times8$ for Y and $32\times32$ for I and Q images. The block sizes that we used for FPVQ are $8\times8$ for Y and $16\times16$ for I and Q images. The codebook sizes were 4096 for Y, and 2048 for both I and Q. The codebook is generated by applying FPVQ on ten training images. The bit-rate of our method is 0.414 ($=\frac{8}{8\times8}+\frac{8}{32\times32}+\frac{8}{32\times32}+\frac{12}{8\times8}+\frac{11}{16\times16}+\frac{11}{16\times16}$). Table 1 shows the APSNR of the images obtained by the JPEG method and our method. The APSNR measurement used here were calculated by

$$\mathrm{APSNR}=10\times\log\frac{3\times255^2}{\mathrm{MSE}},$$

where MSE denotes the Mean Square Errors (MSE) of the images. Figure 3, Figure 4, and Table 1 show that our method gives satisfactory results at low bit-rate and is competitive to the JPEG method.

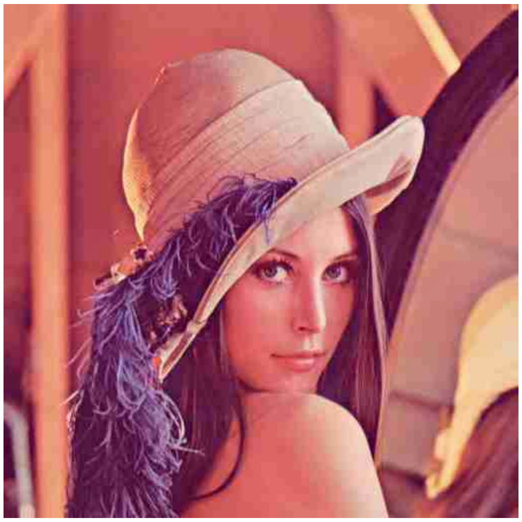

Fig. 3: The image obtained by the JPEG method. The APSNR is 29.85 dB and the bit rate is 0.426 bpp.

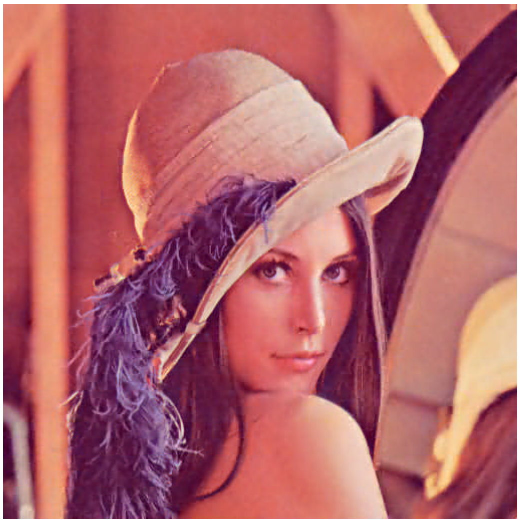

Fig. 4: The image obtained by our method. The APSNR is 31.48 dB and the bit rate is 0.414 bpp.

TABLE 1
The APSNR/bit-rate of the JPEG method and our method. The first three images Lena, Jet, Peppers were in our training set and the last image Tiffany was not.

| images | JPEG method | our method |
|---|---|---|
| Lena | 29.85 dB/0.426 bpp | 31.48 dB/0.414 bpp |
| Jet | 29.00 dB/0.464 bpp | 31.00 dB/0.414 bpp |
| Peppers | 27.59 dB/0.450 bpp | 29.48 dB/0.414 bpp |
| Tiffany | 28.92 dB/0.4433 bpp | 28.98 dB/0.414 bpp |

## 5. Conclusion

In this paper, a feature-preserving interpolative vector quantization method is proposed to compress images. On compressing an image, the proposed method decomposes the image into a low frequency image (i.e., the approximation image) and a high frequency image (i.e., the residual image which is generated by subtracting the approximation image from the original image). The proposed method applies the MVP algorithm to generate the approximation image and uses the FPVQ algorithm to quantize the residual image. The encoder sends the top left pixel's intensity value in every block of the approximation image and the index of the represented codeword for every block of the residual image to the network. The decoder reconstructs the image on the other end of the network by adding the approximation image (generated by interpolating received intensity values) and the quantized residual image (generated by replacing received indices with their corresponding codewords in a pre-stored codebook). The experimental results show that the proposed method usually yields satisfactory results.

## 6. Acknowledges

This work is supported by the National Science Council.

## 7. References

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

[2] N M Nasrabadi and R A King, "Imaging coding using vector quantization: a review," *IEEE Trans. Commun.*, Vol. 36, pp. 957-971, 1988.

[3] T Murakami, K Asai, and E Yamazaki, "Vector quantizer of video signals," *Electronic Letters*, pp. 1005-1006, 1982.

[4] A Gersho and R M Gray, "Vector quantization and signal compression," Boston Kluwer Academic Publisher, 1992.

[5] Linde Y, Buzo A, and Gray R M, "An algorithm for vector quantifier design," *IEEE Trans. Commun.*, Vol. 28, pp. 84-95, 1980.

[6] Baker R L, Vector quantization of digital images, Ph.D. dissertation, Standford Univ., Standford, CA, June 1984.

[7] R. L. Baker and R. M. Gray, ``Image compression using non-adaptive spatial vector quantization," *Proc. 16th Asilornar Conf. Circuits Syst. Comput.*, Oct. 1982.

[8] C M Huang and R W Harris, "A comparison of several vector quantization codebook generation approaches," *IEEE Trans. Image Processing*, Vol. 2, No. 1, pp. 108-112, 1993.

[9] H M Hang and B G Haskell, "Interpolative vector quantization of color images," *IEEE Trans. Communication.*, Vol. 36, No. 4, pp. 465-470, 1988.

[10] Wu-Ja Lin, Wei-Yu Han, Kuang-Shyr Wu, and Yen-Po Lee, "Image approximation by preserving color features," *accepted and to appear in the 7th international conference on computer vision, pattern recognition and image processing in conjunction with the 9th joint conference on information sciences (JCIS 2006)*, Oct. 8-11, Kaohsiung, Taiwan, R.O.C., 2006.

[11] Wu-Ja Lin and Ja-Chen Lin, "Real-time and automatic two-class clustering by analytical formulas," *Pattern Recognition*, Vol. 29, No. 11, pp. 1919-1930, 1996.

[12] Wallace G K, The JPEG still picture compression standard, Commun. *ACM*, Vol. 34, pp. 30--44, 1991.