ISSN: 0976-9102 (ONLINE)
DOI: 10.21917/ijivp.2013.0104

CODEBOOK ENHANCEMENT FOR VECTOR QUANTIZATION USING MINMAX VALUES FOR IMAGE COMPRESSION

S. Vimala
Department of Computer Science, Mother Teresa Women's University, India
E-mail: vimalaharini@gmail.com

Abstract
Vector Quantization (VQ) is one of the Lossy Image Compression Techniques. Vector Quantization comprises of three different phases: Codebook Generation, Image Encoding and Image Decoding. The efficiency of VQ depends on the quality of the codebook. In this paper, we have proposed a novel idea for improving the quality of codebook. The codebook is populated with high detail blocks, with which the edges of the objects in an image can be preserved. The high detail blocks are identified based on the maximum and minimum pixel values of the blocks. K-Means clustering method is also used to improve the quality of the codebook. The quality (PSNR) of the reconstructed images by the proposed method is better when compared to that of few existing codebook generation techniques. Standard images are used to test the performance of the proposed method.

Keywords:
Image Compression; Codebook; Training Set; High Detail Block; PSNR

1. INTRODUCTION

Image compression is essential for applications such as TV transmission, video conferencing, facsimile transmission, of printed material, graphics images, or transmission of sensing images obtained from satellites and reconnaissance aircraft [1]. Image compression techniques deal with the reduction of data required to represent images. This image can be economically transmitted as achieved. The image compression techniques are generally classified into two major types namely, the lossless compression techniques and lossy compression techniques [2]. In lossless compression techniques, there is no information loss, and the image can be reconstructed exactly the same as the original. Some of the lossless compression techniques are, 1). Run length encoding 2). Huffman coding 3). LZW coding 4). Area coding [2] and [3]. The lossy schemes provide much higher compression ratios compared to lossless scheme. Lossy schemes are widely used since the quality of the reconstructed images is adequate for most applications. By this scheme, the reconstructed image contains degradations with respect to the original image. Some of the lossy compression techniques are, 1). Transformation coding 2). Vector quantization 3). Fractal coding 4). Block Truncation coding and 5). Subband coding [2] and [3]. Vector Quantization (VQ) is an efficient image coding technique achieving low bit rates.

Several VQ algorithms have recently been extensively investigated/developed for speech and image compression. VQ is a lossy image compression technique and has applications in different areas: Protein Classification, Secondary Structure Computation [4], Speech Recognition, Pattern Recognition, Real-time Video based Event Detection and Anomaly Intrusion Detection system, etc [5]. VQ techniques have been used for a number of years for data compression. With relatively simple structure and computational complexity, many types of VQ, such as Classified VQ [6] and [7], Address VQ [6] and [8], Finite State VQ [6] and [9], Side Match VQ [6] and [10], Mean-Removed Classified VQ [9] and [11], and Predictive Classified VQ [9] and [12], have been used for various purposes. VQ has been applied to some other applications, such as Index Compression [9] and [13], and Inverse Halftoning [9], [14] and [15].

The compressed images using any lossy compression technique, when reconstructed, will not be exactly equal to the original image. The difference between the original image and the compressed image is called Mean Square Error (MSE) and is computed using the Eq.(1).

$$MSE = \sum_{y=1}^{M}\sum_{x=1}^{N}[I(x,Y) - I^{'}(x,y)]^2 \tag{1}$$

where, $I'(x, y)$ is the reconstructed image and $I(x, y)$ is the original image.

The quality of the reconstructed image called the Peak Signal to Noise Ratio (PSNR) is computed using the Eq.(2),

$$PSNR = 10\log_{10}\left(255^2/MSE\right) \tag{2}$$

In this paper, we have proposed a novel idea of generating a codebook by taking the first M high detail blocks. The proposed method High Detail Codeboook Generation technique (HDCBG) gives better performance in terms of PSNR values when compared to existing methods.

In section 2, some existing codebook generation algorithms are explained. In section 3, the proposed method is explained. In section 4, the results are discussed and the conclusion is given in section 5.

2. EXISTING METHODS

2.1. SIMPLE CODEBOOK GENERATION (SCG)[17]

An image of size $m \times m$ pixels is first divided into small non-overlapping blocks of size 4×4 pixels. Usually m is a power of 2. The block is then converted into a one-dimensional array of 16 elements. An image of size $m \times m$ will give N training vectors, where,

$$N = (m \times m)/16 \tag{3}$$

The aim of VQ is to design an optimal codebook CB of size M, which comprises of M codevectors that will be the representative vectors of the training vectors.

$$CB = \{C_i \mid i \le i \le M\}, \quad \text{where } M < N \tag{4}$$

A codebook of size M is generated by selecting M code vectors from N training vectors., in which the training vectors at every nth position are selected to generate the initial codebook [17] using the Eq.(5).

$$n = (N/M) \tag{5}$$

For example, to generate a codebook pf size 256, the training vectors at every 16th position are selected to form the codevectors.

2.1.1 Steps to generate a codebook using SCG algorithm:

Step 1: Input the given image of size $m \times m$ pixels.

Step 2: Generate N training vectors by dividing the image into small blocks of size 4×4 pixels using the Eq.(3).

Step 3: Set the value for M.

Step 4: Compute the position n using Eq.(5).

Step 5: Select every nth training vector till the codebook of desired size M is reached.

2.2 CODEBOOK GENERATION WITH EDGE FEATURES (CBEF) [18]

The input image of size $n \times n$ pixels is divided into N sub blocks of size 4×4 pixel as,

$$N = (n * n)/16. \tag{6}$$

The N image blocks thus formed are categorized into two classes namely the edge blocks and the shade blocks. The high detail blocks are classified as edge vectors and the low detail blocks are classified as shade vectors. The set of edge vectors is treated as a separate codebook T1 and the set of shade vectors is T2. The final codebook is the concatenation of T1 and T2. The categorization of blocks is done as follows:

Mean of the components of each vector is computed as,

$$m = \frac{1}{16}\sum_{j=1}^{16} x_{ij} \tag{7}$$

Sum of the difference between the individual components and the mean value is computed as,

$$S = \sum_{j=1}^{16} \left| m - x_{ij} \right| \tag{8}$$

If the value S is greater than a threshold value, the block is categorized as an edge block that belongs to set T1, Otherwise it is categorized as shade block that belongs to T2.

2.2.1 Steps to generate a codebook using CBEF algorithm:

Step 1: Input the given image of size $n \times n$ pixels. Divide the image into blocks of size 4×4 pixels.

Step 2: Generate N training vectors using Eq.(6).

Step 3: Calculate the mean of the training vectors using the Eq.(7).

Step 4: Compute S using the Eq.(8).

Step 5: If $S >$ threshold value, then X_i belongs to T1 else X_i belongs to T2.

Step 6: Add the whole set T1 to the final codebook.

Step 7: Compute v1= sizeof(TS) - sizeof(T1)

Step 8: Compute v2= sizeof(CB) - sizeof(T1)

Step 9: Compute n = v1/v2

Step 10: Select every nth shade block form T2 and add it to the final codebook.

Step 11: Repeat the step10 v2 times to get the codebook of desired size.

This method involves more computations in identifying high detail blocks. Hence an idea to reduce are number of computations is introduced in the proposed method.

3. PROPOSED METHOD (HDCBG)

In the existing method, the input image blocks are classified into high detail and low detail blocks based on the variance using the Eq.(7) and Eq.(8). In the proposed method, the blocks are classified based on the minimum and maximum values. The difference between maximum and minimum values is computed using the Eq.(9).

$$D = \text{Max} - \text{Min} \tag{9}$$

The blocks are sorted in descending order based on D as per Eq.(10).

$$S = \{S_1, S_2, \ldots. S_m\} \quad \text{where } D_i > D_{i+1} \tag{10}$$

where $S_1, S_2, \ldots, S_m$ are vectors. From the sorted vectors, the top M vectors are selected to form the codebook. The intention behind selecting the code vectors based on the difference value is that if the difference is high, then the pixel values are different, which means they are high detail blocks (edge blocks). If edge blocks are preserved, the quality of the compressed images will be improved.

3.1 STEPS TO GENERATE A CODEBOOK USING PROPOSED ALGORITHM

Step 1: Divide the input image into small blocks of size 4×4 pixels.

Step 2: Identify the maximum and minimum values as Max and Min for each block.

Step 3: Compute the difference D using the Eq.(9).

Step 4: Sort the training vectors based on the D values to form the set S as for Eq.(10).

Step 5: Set the codebook size to M.

Step 6: Select M vectors from the sorted list S to form the codebook.

To improve the quality of the codebook further, we have incorporated K-means clustering technique as part of the proposed method.

3.2 K-MEANS CLUSTERING TECHNIQUE

The initial codebook that is generated by any of the codebook generation techniques can further be optimized. The N training vectors are grouped into M clusters with the initial codevectors as the centroids of all the clusters. Pick up any code vector C_i. Find all the image blocks X_i that are closer to C than to any other C_j, i.e. find the set of all X_i that satisfy the Eq.(11).

$$s(X_i, C_i) < s(X_i, C_j) \text{ for all } j \neq i, \tag{11}$$

where the distance between the training vector X_i and the codevector C_i is computed as,

$$s(X, C_i) = \sum_{j=1}^{16} (X_j - C_{ij}) \tag{12}$$

where, $1 \le i \le M$ and $1 = j \le 16$.

Compute the sum vector by adding all the corresponding components of the training vectors X_i that are closer to C_i. The individual components of the sum vector are calculated by adding the corresponding components of all the training vectors of the same cluster as,

$$Sum_{ij} = \sum_{j=1}^{c} X_{ij} \tag{13}$$

where c is the cluster strength.

Now every component is divided by the cluster strength to get the new centroid of the cluster.

$$Centroid = Sum_{ij} / v_i, \quad \text{where } i = 1,2,\ldots.v \tag{14}$$

The codevector X_i is replaced with the new centroid to form the revised codebook. These steps are repeated till the codebooks of consecutive iterations converge.

3.3 K-MEANS CLUSTERING ALGORITHM

Step 1: The training vectors are grouped into M clusters based on the distance between the codevectors and the training vectors, using the Eq.(12).

Step 2: Compute the sum vector for each cluster using the Eq.(13).

Step 3: Compute the centroid for each cluster using the Eq.(14).

Step 4: Replace the existing codevector with the new centroid to from the revised codebook.

Step 5: Repeat the steps1 through 4 till the codebooks of consecutive iterations converge.

4. RESULTS AND DISCUSSION

All the algorithms; SCG, CBEF, and the proposed algorithm are implemented on 256 × 256 grayscale images: Lena, Cameraman, Bridge and Boats. The input images are given in Fig.1. The algorithms are implemented using Matlab 7.0 on Windows Operating System. The hardware used is the Intel Core 2E7400@ Duo 2.8 GHZ Processor with 2 GB RAM. Codebooks of sizes 512, 1024 are created using all three discussed methods. The quality of the reconstructed images in terms of PSNR is given in Table.1.

(a) Lena (b) Cameraman (c) Bridge (d) Boats

Fig.1. Input images taken for the study

When the Codebook size is 512, the bpp obtained is 1.56 and the bpp is 2.62 with the codebook of size 1024 codevectors. From the table, it is observed that the PSNR values obtained with the proposed method is better when compared with that of the SCG and CBEF methods for a codebook of size 1024. When the compression ratio (CR) is high, the quality (PSNR) of the reconstructed image will be less and vice versa.

Table.1. Comparison of the quality of the reconstructed images

CB Size	512			1024		
Images	**SCG**	**CBEF**	**HDCBG**	**SCG**	**CBEF**	**HDCBG**
Lena	37.04	37.35	37.31	38.83	40.42	42.31
Cameraman	33.09	34.32	34.29	35.60	33.44	40.01
Bridge	31.15	31.13	30.57	33.61	33.36	35.66
Boats	33.54	33.29	33.72	36.22	34.91	38.95
Average	33.71	34.02	33.97	36.07	35.53	39.23

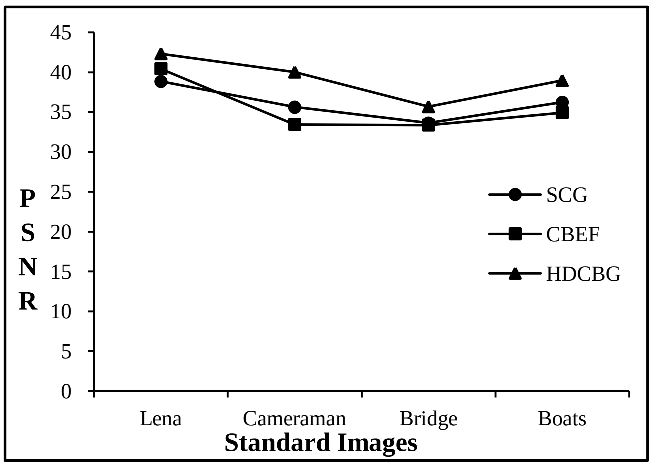

Fig.2. Quality (PSNR) of reconstructed image

On an average, the PSNR value of SCG method is 36.07, CBEF method is 35.53 and that of the proposed method is 39.23, which is a significant improvement. For a codebook of size 512, the results of the proposed method are better than that of SCG method, but is only slightly less than that of CBEF method. The performance of the existing methods SCG and CBEF and the proposed method in terms of PSNR is given in Fig.2. In this figure, the PSNR values are depicted in Y axis and the images are given in the X axis.

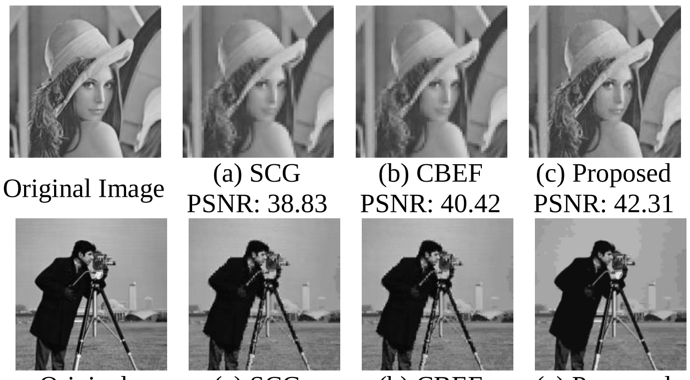

Fig.3. Visual comparison of the reconstructed images with a codebook of size 1024

The computations are heavy in CBEF method in identifying the high detailed blocks. For every vector, mean must be computed. And then the sum of difference between the mean and the individual values must be computed. This takes time. But in the proposed method, it is sufficient to find the maximum and minimum values of a vector to identify a high detail block. In Fig.2, it is clear that the results obtained with the proposed method are better than the existing methods. The reconstructed images that are compressed using the methods discussed in this paper are given in Fig.3 for visual comparison. In Fig.3, when we look at the images reconstructed with the existing methods and the proposed method, the images reconstructed with the proposed method look better with smooth edges. The proposed method is simple when compared to that of the existing method CBEF. The difference between the maximum and minimum intensity values will be high only if edge pixels are in the input block. Hence the difference values are arranged in descending order and the top *M* respective blocks are selected to populate the Codebook and thus the edge blocks are preserved.

5. CONCLUSION

In the proposed method, codebook is generated with high detail blocks that improve the quality of the codebook. This ultimately improves the performance of the Vector Quantization in terms of the quality of the compressed images. The training set includes the images Lena, Cameraman, Bridge, Boats of size 256 × 256 pixels. The results of the existing techniques SCG and CBEF are compared with that of the proposed method HDCBG with different codebook sizes 512 and 1024. For a codebook of size 1024, the proposed method yields better results when compared to the existing methods SCG and CBEF. For the codebook of size 512, it is better than SCG in terms of PSNR, but there is only slight difference in PSNR when compared to the results of CBEF. But in terms of Computational complexity, the proposed method is better than CBEF. We have implemented this method for gray level images. This can also be tried for color images. This technique is useful for handheld devices where image processing applications have become inevitable now-a-days and the little loss in image data does not make much difference between the original image and the compressed image as the images are only used for Human Visual System and not for the purpose of Analysis by machines.

REFERENCES

[1] Nasser M. Nasrabadi and King R. A, “Image Coding using Vector Quantization: A Review”, *IEEE Transactions on Communications*, Vol. 36, No. 8, pp. 957-971, 1988.

[2] Sonal and Dinesh, Kumar, “A study of various image compression technique”, *National Conference on Challenges and Opportunities in Information Technology*, 2007.

[3] Rafael C. Gonzalez and Richard E. Woods, “*Digital Image Processing*”, Second Edition, Prentice Hall, 2002.

[4] Juan J. Merelo Guervos, M. A. Andrade, Carlos Urena, Alberto Prieto and Federico Moran, “Application of Vector Quantization Algorithms to Protein Classification and Secondary Structure Computation”, *Proceedings of the International Workshop on Artificial Neural Networks*, pp. 415-421, 1991.

[5] C. Garcia and G. Tzirita, “Face Detection using Quantized Skin Color Regions Merging and Wavelet Packet Analysis”, *IEEE Transactions on Multimedia*, Vol. 1, No. 3, pp. 264-277, 1999.

[6] Jim Z. C. Lai, Yi-Ching Liaw and Julie Liu, “A Fast VQ Codebook Generation Algorithm using CodeWord Displacement”, *Pattern Recognition*, Vol. 41, No. 1, pp. 315-319, 2008.

[7] Y. C. Liaw, Winston Lo and Jim Z. C. Lai, “Image Restoration of Compression Image using Classified Vector Quantization”, *Pattern Recognition*, Vol. 35, No. 2, pp. 181-192, 2002.

[8] N. M. Nasrabadi and Y. Feng, “Image Compression using Address Vector Quantization”, *IEEE Transactions on Communications*, Vol. 38, No. 12, pp. 2166-2173, 1990.

[9] J. Foster, R. M. Gray and M. O. Dunham, “Finite State Vector Quantization for Waveform Coding”, *IEEE Transactions on Information Theory*, Vol. 31, No. 3, pp. 348-359, 1985.

[10] T. Kim, “Side Match and Overlap Match Vector Quantizers for Images”, *IEEE Transactions on Image Processing*, Vol. 1, No.2, pp. 170-185, 1992.

[11] J. Z. C. Lai, Y. C. Liaw and W. Lo, “Artifact Reduction of JPEG Coded Images using Mean-Removed Classified Vector Quantization”, *Signal Processing*, Vol. 82, No. 10, pp. 1375-1388. 2002.

[12] K. N. Ngan and H. C. Koh, “Predictive Classified Vector Quantization”, *IEEE Transactions on Image Processing*, Vol. 1, No. 3, pp. 269-280, 1992.

[13] C. H. Hsieh and K. C. Tsai, “Lossless Compression of VQ Index with Search Order Coding”, *IEEE Transactions on Image Processing*, Vol. 5, No. 11, pp. 1579-1582, 1996.

[14] J. Z. C. Lai and J. Y. Yen, “Inverse Error Diffusion using classified Vector Quantization Technique”, *IEEE Transactions on Image Processing*, Vol. 7, No. 12, pp. 1753-1758, 1998.

[15] P. C. Chang, C. S. Yu and T. H. Lee, “Hyprid LMS-MMSE Inverse Halftoning Technique”, *IEEE Transactions on Image Processing*, Vol. 10, No. 1, pp. 95-103, 2001.

[16] Allan Gersho and R. M. Gray, “*Vector Quantization and Signal Compression*”, Springer, 1992.

[17] K. Somasundaram and S. Vimala, “Simple and fast Ordered Codebook Generation for Vector Quantization”, *Proceedings of the National Conference on Image Processing*, 2010.

[18] K. Somasundaram and S. Vimala, “Codebook Generation for Vector Quantization with Edge Features”, *CiiT International Journal of Digital Image Processing*, Vol. 2, No. 7, pp. 194-198, 2010.