RESEARCH PAPER | Computer Science | Volume : 5 | Issue : 4 | April 2015 | ISSN - 2249-555X

Classified Codebook with Indexmap Compression for Vector Quantization to Compress still Images

KEYWORDS: image compression, vector quantization, codebook, index map, bpp.

Dr.S.Vimala
Associate Professor, Dept. Of Comp. Science, Mother Teresa Women's University, Kodaikanal – 624 102, Tamil Nadu, India

Ms.S.Ezilarasi
Research Scholar, Dept. Of Comp. Science, Mother Teresa Women's University, Kodaikanal

ABSTRACT *Vector Quantization (VQ) is one of the Lossy Image Compression techniques. VQ comprises of three steps in compressing the images: 1. Codebook Generation, 2. Image Encoding and 3. Image Decoding. The phase of codebook generation plays a vital role in compressing the image. The performance of VQ depends on the quality of the codebook. The quality of the reconstructed image is improved at the cost of compression rate and vice versa. The feature of inter-pixel redundancy is exploited in this method. Codebook is a collection of blocks (vectors) selected at random from the input image blocks. In the proposed method, the vectors in the codebook are classified into shade (high-detail) and edge (low-detail) blocks. For shade blocks, only the average of the components of the vector is stored, leading to a significant reduction in bpp (bits per pixel). Further compression is achieved by compressing the indices of the Index Map that is generated during the Encoding phase. The proposed method is tested with some bench mark images such as Lena, Cameraman, Baboon, Boats and Bridge. Results are generated for codebooks of sizes 128, 256, 512 and 1024. In all the cases, the proposed method outperforms the existing technique in terms of bpp (bits per pixel).*

I. INTRODUCTION

Image compression techniques have become inevitable now-a-days as there is a heavy demand for images to be transmitted and stored. Image compression techniques are classified into Lossy and Lossless techniques. The compressed images of Lossless techniques are exact replica of the original images. But lossy compression techniques lead to acceptable loss in image data whose reconstructed images are suitable for Human Visual System (HVS). Vector Quantization is one of the lossy image compression techniques and is an attractive technique for compressing images [1] & [2]. Due to its simplicity and improved coding efficiency, it is applied in the field of image compression [3], [4] & [5]. VQ comprises of three phases: Codebook Generation, Encoding Phase and Decoding Phase. The performance of VQ depends on a good codebook. The method used most often for developing a codebook is the Linde-Buzo-Gray (L.G.B.) algorithm [6]. VQ finds its applications in many areas such as Speech Recognition, Protein Classification, Pattern Recognition, Secondary Structure Computation, Real Time Video based Event Detection and Anomaly Intrusion Detection System [7] & [8]. VQ is a process [9] in which the image to be compressed is broken into small blocks called vectors. These vectors are called training vectors and the collection of such vectors is called Training Set (TS). There are several methods existing for generating a codebook [10]. In image encoding phase, for each input block called training vector, the codebook index of the closest codevector is identified using the Euclidean Distance measure. The distance between the training vector and the codevector is computed using the equation (3).

$$d(x, y) = \sum_{j=1}^{16} abs[X_i - Y_{ij}] \quad (3)$$

Where X_i is the Training Vector and Y_{ij} is the i^{th} Codevector. The collection of such indices is called Index Map (IM). The compressed image is then stored / transmitted in the form of Index Map and Codebook. In the receiving end, the image is decoded by replacing every codebook index with the respective codevector to reconstruct the compressed image.

The difference between the original and the compressed images is measured using the unit called MSE given in the equation 1 [11].

$$MSE = \frac{1}{N*N} \sum_{i=1}^{N} \sum_{j=1}^{N} (I - I')^2 \quad (1)$$

Where I is the original image and I' is the reconstructed image. The quality of the reconstructed image is measured using Peak Signal to Noise Ratio (PSNR), an inverse of MSE and is computed using the equation (2).

$$PSNR = 10 \, Log_{10}\left[\frac{(255)^2}{MSE}\right] \quad (2)$$

The inverse of MSE is PSNR which is used to measure the quality of the reconstructed image and is computed using the equation 2.

In a method called 'Improved Coding Efficiency by Intensity Transformation for VQ (ICEIT-VQ)' [12], the pixels in the original image are transformed by dividing them by 8 to reduce the bpp by 3. VQ is then applied to the transformed image thereby the bpp is reduced by a significant level. In the proposed method, the coding efficiency is improved in two levels and is discussed in Section 2. In Section 3, Results are discussed and the Conclusion is given in Section 4.

II. PROPOSED METHOD

In the simple codebook generation technique, the vectors at random from training set are selected to form the codebook. If the codebook size M is 256, then the size of the codebook is computed as 256 x 16 x 8. In the proposed method, the codevectors in the codebook are classified into edge and shade blocks. This classification is done by computing the magnitude using the equation 3.

$$m = \sum_{i=1}^{16} abs(x - x') \quad (3)$$

Volume : 5 | Issue : 4 | April 2015 | ISSN - 2249-555X

Where x' is the mean of the block. If the magnitude value is less than 20, then the block is classified into shade block. The value 20 is a threshold value chosen at random to classify the block. If it is a greater value, there will be further degradation in the quality of the image because even if the pixel values in a block are highly different, the block may be classified into a shade block. In the codebook, for the shade blocks, only the average value of the components is stored. For the edge blocks, the entire codevector is stored. For a shade block, it requires only 8 bits to store the average value and it requires *16 x 8* bits for an edge block. Thus if there are *n* shade blocks, the size of the codebook is reduced by *n * 15 *8* which would lead to a significant reduction in bitrate.

In the second level, the size of the IndexMap is reduced. The indices in the IndexMap are read in a Rasterscan order. Each index value in the (x, y) coordinate is compared against its neighbours in the coordinates (x,y-1), (x-1,y-1), (x-1,y) and (x-1,y+1). If it is equal to any of its specified neighbours, the index is replaced with 00/01/10/11. Generally the index value requires 8 bits. But with the proposed idea, it requires only 2 bits. Hence if *n* indices are equal to one of their neighbours, the size of the IndexMap IM is reduced by *n x 6* bits. In the first level is a lossy method which leads to slight degradation in the quality of the reconstructed image but a significant improvement in the compression rate.

The second level of compression is a lossless technique where the quality of the image remains unchanged. On the whole, the proposed method yields a good compression rate.

Algorithm

- Step1: Split the input image into non-overlapping blocks of size 4 x 4 pixels.
- Step2: Convert the blocks into vectors to form a Training Set of size N.
- Step3: Select every n^{th} training vector from the training set TS to form the codebook CB of size M where n = N/M.
- Step4: Classify each codevector either as a shade block or edge block by computing the magnitude using the equation 3.
- Step5: If shade block, store the mean of the codevector else retain the entire codevector.
- Step6: Encode the image by generating the IndexMap IM.
- Step7: Check if each index in the IndexMap IM is equal to one of its neighbours. If equal, replace the existing index with one of the values 00/01/10/11.
- Step8: Store or Transmit the original image in the form of Compressed Codebook and Compressed Index Map.

The compression rate is calculated as follows:
With the normal codebook:

- Size of the codebook : 256 x 16 x 8 = 32768 (if M=256)
- Size of the IndexMap : 64 x 64 x 8 = 32768
- Size of the Compressed Image : 32768+32768 = 65536
- Compression Rate : 65536/(256*256) = *1 bpp* (bits per pixel)

With the Compressed Codebook

- Size of the Codebook : (n x 8) + ((256-n)*16*8) where n is the number of shade blocks.

(for the cameraman image n= 119)

- Hence size of the codebook = 18488
- Size of the IndexMap = 32768
- Size of the Compressed Image = 18488+32768 = 51256
- Compression Rate = 51256/(256*256) = *0.78 bpp*

With the compressed Codebook and compressed IndexMap

- Size of the compressed Codebook = 18488
- Size of the IndexMap: (1901 * 2) + (4096-1901) * 8 = 21362
- where 1901 is the no. of indices equal to one of their neighboring indices in the case of cameraman image.
- Compression Rate: (18488+21362) / (256*256) = *0.61 bpp*

III. RESULTS AND DISCUSSION

The proposed algorithm is tested with standard images such as Cameraman, Lena, Baboon, Boats and Bridge which are taken for the study and are given in the Figure-1.

Figure – 1: Input Images taken for the study.

Table-1: Results in terms of PSNR and bpp with respect to Normal Codebook Generation and the proposed method.

CB Size	Image	Normal Codebook		Compressed Codebook (with edge and shade blocks)		Compressed IndexMap	
		PSNR	Bpp	PSNR	bpp	PSNR	bpp

Volume : 5 | Issue : 4 | April 2015 | ISSN - 2249-555X

128	Lena	29.52	0.69	29.42	0.62	29.42	0.49
	Cameraman	26.42	0.69	26.51	0.55	26.51	0.35
	Baboon	32.29	0.69	32.16	0.56	32.16	0.49
	Boats	25.74	0.69	25.86	0.49	25.86	0.32
	Bridge	25.74	0.69	25.83	0.39	25.83	0.39
	Average	27.94	0.69	27.96	0.52	27.96	0.41
256	Lena	31.24	1	31.08	0.80	31.08	0.68
	Cameraman	27.41	1	27.44	0.78	27.44	0.61
	Baboon	34.23	1	34.13	0.69	34.13	0.62
	Boats	27.95	1	27.97	0.67	27.97	0.54
	Bridge	26.76	1	26.79	0.57	26.79	0.50
	Average	29.52	1.00	29.48	0.70	29.48	0.59
512	Lena	32.45	1.56	32.28	1.13	32.28	1.01
	Cameraman	29.01	1.56	29.02	1.14	29.02	0.96
	Baboon	35.57	1.56	35.36	0.95	35.36	0.91
	Boats	29.93	1.56	29.30	0.98	29.30	0.86
	Bridge	27.91	1.56	27.94	0.69	27.94	0.63
	Average	30.97	1.56	30.78	0.98	30.78	0.87
1024	Lena	33.97	2.63	33.77	1.74	33.77	1.63
	Cameraman	30.98	2.63	30.93	1.84	30.93	1.65
	Baboon	37.43	2.63	37.00	1.39	37.00	1.35
	Boats	31.32	2.63	31.23	1.56	31.23	1.44
	Bridge	29.30	2.63	29.27	0.88	29.27	0.84
Average		32.60	2.63	32.44	1.48	32.44	1.38

The results generated for different codebooks of sizes 128, 256, 512 and 1024 are given in Table-1. For a codebook of size 128, the PSNR obtained is 29.52 and the bpp achieved is 0.69 for the image Lena. When the codebook is compressed by grouping the codevectors into *Edge* and *Shade* blocks, the bpp is reduced to 0.62 with the PSNR of 29.42. By compressing the IndexMap, the bpp has been reduced to 0.49 which is a significant improvement in Compression rate. For the Cameraman and Bridge images, the PSNR is raised along with the improvement in the compression rate for the codebooks of sizes 128, 256 and 512.

TABLE 2: Performance of ICEIT-VQ and the proposed methods in terms of PSNR and bpp.

CB Size	Image	ICEIT-VQ		Proposed Method	
		PSNR	Bpp	PSNR	bpp
128	Lena	29.22	0.59	29.42	0.49
	Cameraman	26.01	0.59	26.51	0.35
	Baboon	32.41	0.59	32.16	0.49
	Boats	26.23	0.59	25.86	0.32
	Bridge	24.44	0.59	25.83	0.39
	Average	27.66	0.59	27.96	0.41
256	Lena	30.34	0.81	31.08	0.68
	Cameraman	27.04	0.81	27.44	0.61
	Baboon	33.64	0.81	34.13	0.62
	Boats	27.43	0.81	27.97	0.54
	Bridge	25.40	0.81	26.79	0.50
	Average	28.77	0.81	29.48	0.59
512	Lena	31.58	1.19	32.28	1.01
	Cameraman	27.89	1.19	29.02	0.96
	Baboon	34.81	1.19	35.36	0.91
	Boats	28.57	1.19	29.30	0.86
	Bridge	26.44	1.19	27.94	0.63
	Average	29.86	1.19	30.78	0.87
1024	Lena	32.90	1.88	33.77	1.63
	Cameraman	29.62	1.88	30.93	1.65
	Baboon	31.30	1.88	37.00	1.35
	Boats	29.97	1.88	31.23	1.44
	Bridge	27.85	1.88	29.27	0.84
Average		31.30	1.88	32.44	1.38

In Table 2, the PSNR and the bpp obtained with respect to the ICEIT-VQ and the proposed methods are given for the codebooks of sizes 128, 256, 512 and 1024. In ICEIT method, the bpp obtained for a codebook of size 128 is 0.59 for all images. Since a uniform transformation is done for all the pixels and the bpp is the same for all images. But in the proposed

method, the bpp obtained is different for different images. This is because, depending on the image, the number of edge and shade blocks differ for their respective codebook. And also during the Index Map compression in the proposed method, the inter-block redundancy is exploited and the bpp varies according to the similarity of the neighbouring blocks. Hence the bpp is different for each image unlike the ICEIT-VQ method.

For a codebook of size, in the existing method the average bpp achieved is 0.59 and in the proposed method, the bpp achieved is 0.41. The PSNR is also raised to 27.96 from 27.66. Similarly for codbook sizes, the performance of the proposed method is better than that of the existing method both in terms of PSNR and bpp. This is a significant improvement.

IV. CONCLUSION

In the proposed method, the codevectors are classified into Edge and Shade blocks based on the magnitude. For Shade blocks, only the average of the vector is stored which leads to a good compression ratio. In the next level, the IndexMap is compressed by reducing the number of bits required to store the same neighbouring indices. This leads to further reduction in bitrate. The proposed method is tested with some standard images Cameraman, Lena, Boats, Bridge and Baboon. In all the cases, the results are better in terms of bpp when compared to that of the existing method with slight reduction in PSNR. This method is suitable for hand held devices and can also be applied for color image and video compression.

REFERENCE R.Baker and R.M.Gray, "Image Compression using Non-Adaptive Spatial Vector Quantization", Proc. 16th Asilomar Conference on Circuits, Systems and Computers, pp. 55-61, Oct. 1982. | [2] R.M.Gray, "Vector Quantization", IEEE ASSP Magazine, Vol. 1, pp. 4-29, 1984. | [3] NM.Nasrabadi and Y.Feng, "Image Compression using Address Vector Quantization", IEEE Transactions and Communication, Vol. 38, No.12, pp. 2166-2173, 1990. | [4] A.Gersho, "On the Structure of Vector Quantization", IEEE Trans. Inform. Theory, Vol. IT-28, pp. 157-166, Mar. 1982. | [5] W.H.Equitz, "A New Vector Quantization Clustering Algorithm", IEEE Trans. And Acoustics Speech Signal Processing, Vol. 37, No. 3, pp. 312-321, 1992. | [7] 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. | [7] 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] Y.Linde, A.Buzo and R.M.Gray, "An Algorithm for Vector Quantizer Design", IEEE Trans. Commun., Vol. COM-28, No. 1, pp. 84-95, 1980. | [9] 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, Jul. 2010. | [10] Pasi Franti, Timo Kaukoranta, Day-Fann Shen and Kuo Shu Chang, "Fast and Memory Efficient Implementation of the Exact PNN", IEEE Transactions on Image Processing, Vol. 9, No. 5, May 2000. | [11] Chang-Qian Chen, "An Enhanced Generalized Lloyd Algorithm", IEEE Signal Processing Letters, Vol. 11, No. 2, pp. 167 – 170. | [12] S.Vimala, "Improved Coding Efficiency by Intensity Transformation for VQ to Compress Still Images", International Journal of Emerging Technology and Research (IJRTE), Vol. 1, No. 7, pp. 39-43, Dec. 2014. |