Utilizing Index Usage Maps for VQ Index Compression

Mohammed F. Abdel-Latif*, Tarik K. Abdel-Hamid, Magdy M. Doss, and H. Selim
Electrical Engineering Department, Faculty of Engineering, Assiut University
Assiut, Egypt.
*E-mail: farouk@aun.edu.eg

Abstract *- In practical vector quantization (VQ) of images, the used pixel block dimensions are kept small to reduce the cost of computations. This in turn results in highly correlated blocks and the corresponding VQ indices will inherit this high correlation. The compression of the basic VQ can be increased through utilising this high correlation of indices by inserting a lossless index compression stage after the VQ stage. In this paper a new index compression algorithm is introduced. In this algorithm the 2 dimensional index map is divided into non overlapping square blocks. Index usage in each of these blocks is employed to remap (renumber) the reduced number of actually used indices in this block, thus resulting in reduced bit rate expressed in bits/pixel. The proposed algorithm reduces the average bit rate by a value depending on the codebook size, namely a reduction of about 32% for codebook size of 64, and down to about 23% for codebook size of 1024. Moreover this algorithm lends itself to being cascaded by other index compression algorithms resulting in increased compression.*

Keywords *- VQ Index compression, Lossless Coding, Image compression*

1. INTRODUCTION

Vector Quantization can be used to reduce the complexity of signal processing operations such as classification, recognition, enhancement, edge detection, and many other operations, but its main usage is in compression [1]. It is used for image compression due to its high compression ratios and simple decoder implementation which could be used in low complexity end user systems. Also VQ variants can give high compression ratios with competitive peak signal-to-noise ratio (PSNR) [2].

In VQ the image is divided into non-overlapping pixel blocks (4x4 pixels blocks are common), each being represented by a k dimensional vector. VQ consists of two processes: an encoding process and a decoding process. The encoding process is a mapping from the image vectors into a reduced set of codevectors chosen from a codebook according to the nearest neighbour rule. This codebook is available at both the transmitter and receiver, and hence it is sufficient to send only the indices of the codevectors as representatives of the codevectors themselves. Since an index has much fewer bits than the corresponding codevector, compression is achieved. The encoding process results in a two dimensional (2D) index map. Each element in the index map corresponds to a block of pixels. The decoding process is just a lookup table to regain the codevectors and reconstruct the output image.

Full search algorithm can be used to find the best match codevector in the codebook based on the minimum Euclidean distance condition [2]. However, full search algorithm requires many computations to find the best match codevector. Many search algorithms are known to speed up the encoding process, such as partial distance elimination (PDE), and triangular inequality elimination (TIE) [2].

Using a codebook with a certain size, the VQ image quality mainly depends on generating a codebook matched to the image statistics. Many algorithms are available for codebook generation. Linde-Buzo-Gray (LBG) algorithm is the most cited one [3], Pairwise Nearest Neighbour (PNN), splitting method, Simulated Annealing (SA), and fuzzy clustering are known methods for codebook generation. Training vectors are used instead of statistical models in codebook generation.

VQ can be classified into memoryless VQ and memory VQ [2]. Memoryless VQ encodes each block independent from other blocks, while in memory VQ the correlation between adjacent blocks is exploited and the encoding process depends on the past history of the vector quantizer. Memory VQ results in better performance than memoryless VQ, but it is more complex [4]. Index compression also exploits the correlation between adjacent indices to enhance the VQ performance. However, index compression algorithms usually are much simpler than memory VQ algorithms.

The rest of this paper is organized as follows: in section 2 a review of some index compression algorithms is presented, in section 3 the proposed algorithm is introduced. In section 4 simulation results are reported and compared. Discussion and conclusions are given in section 5.

2. INDEX COMPRESSION ALGORITHMS

In memoryless VQ, in order to increase the quality of the reconstructed image, the codebook size is increased to cover all the image variations of grey and edge vectors. Large codebook sizes reduce the compression ratio. Index compression is applied to reduce the data needed to represent the vector quantized image. Variable length coding (VLC) techniques such as Huffman coding is the simplest lossless coding that can be applied to the VQ indices. However, this will introduce extra computations especially for large codebook sizes, and still does not give the best performance that can be reached. To overcome this problem some index compression algorithms have been proposed, such as Search Order Coding (SOC) [4], Switching Tree Coding (STC) [5], Index Grouping Algorithm (IGA) [6], and Hu and Chang method [7]. In the SOC, the previous encoded indices are searched for a matched index to the current index in a predefined search path. If a match is found, the corresponding search order is sent, else the full index is sent. Not only the correlation between adjacent blocks is exploited but also that of blocks located apart. However, SOC does not achieve better compression than the other algorithms [7]. In the IGA, the algorithm attempts to reduce the bit rate by grouping up the blocks with the same index. The block grouping is performed in the index domain. This algorithm consists of two processes, a search process and an encoding process. The search process is used to locate the indices in the index map with the same index value, and then to connect these indices so as to form a group path. The encoding process is used to encode the resulting group path in an effective and ordered manner. However, IGA is complex for hardware implementation, since the operation of group-search is not very regular. Also, it requires extra huge memory space in either encoding or decoding operations [6]. In the STC algorithm, four relations between adjacent indices are defined. Based on these relations, the current index is coded with one of four prefixes defined by three trees. Hu and Chang [7] introduced a low complexity index compression algorithm. The 2D index map is generated, and each index is checked with its adjacent entries (upper or left) in the index map.

3. THE PROPOSED ALGORITHM

Usually the natural image contents are varying between different textures and edges, and this in turn results in codevectors that are locally present in some areas of the image and not found in other areas, which is obvious particularly for the case of large codebook sizes. This property can be used to reduce the bit rate.

In the proposed algorithm ordinary VQ is applied to the image and the resulting indices are used to construct the 2D index map, which is divided into non overlapping square index blocks of size LxL. For each index block an index usage map is constructed which is an N bit vector (N being the size of the codebook). It indicates which indices (or codevectors) are being used in this index block. Usually only a fraction of the total number of indices is used in each index block. With the aid of the index usage map the used indices in the index block can be remapped (renumbered) into a new set of indices, which are fewer in number than the original ones (see Fig. 1). Thus the remapping process reduces the range of indices and correspondingly the number of bits needed to represent the indices themselves.

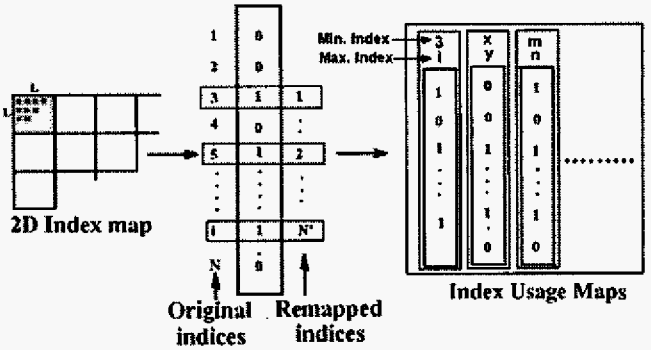

Fig. 1. The remapping process.

Since the index usage map is not the same for different index blocks, the index usage map for each index block is transmitted to the receiver as an overhead. As stated above each index usage map will require N bits. However, if the codevectors in the codebook are arranged according to their mean values, then for each index block, the value of the minimum index, the maximum index and the index usage map between them can be sent resulting in reduced overhead.

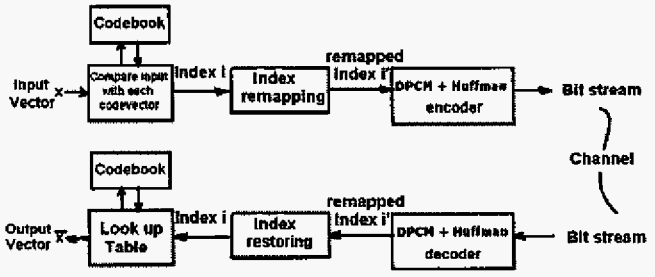

Fig. 2. The proposed algorithm block diagram

Fig. 2 depicts the proposed system where the remapping process is cascaded by a DPCM stage and a Huffman coder. At the decoder the remapped indices are used as an offset in the index usage map to retrieve the original indices. It is to be noted that since the proposed remapping process affects only the absolute values of the indices, meantime it reduces the average difference between adjacent indices; higher compression can be reached by the DPCM stage.

4. SIMULATION RESULTS

Four standard test images, namely Lena, Gold Hill (GH), F-16 and Elaine were analysed for the minimum, mean and maximum number of used indices in the different index blocks of these images. Average percentage values of these quantities are depicted against codebook size (log scale) in Fig.3. The results clearly indicate true reduction of the size of the index space. Also Fig.4 shows a 128x128 pixel Lena image divided into 32x32 index blocks. In this fig. each block of 4x4 pixels is replaced by a dot representing its remapped index. Thereby a remapped index of value '0' is represented by a black dot and with the increase of the remapped index value we move along the gray scale to reach white dot by a remapped index of value N (only possible if all indices are used in the concerned block). Darker index blocks thus indicate that fewer numbers of indices are used, while the bright index blocks are an indication that relatively more indices are used. It is clear that high detail blocks use more indices than low detail blocks, and their index blocks are thus seen to be brighter (see Fig.4).

To evaluate the proposed algorithm, it is compared with other index compression algorithms. The performance is measured either by the required bit rate (in bits/pixel) or by the percentage improvement in bit rate over VQ index bit rate.

Vector quantizers with codebooks of sizes 64, 128, 256, 512, and 1024 are constructed. Five 512x512 pixel images having a resolution of 256 gray levels are used to generate the codebook needed for the simulation. The generation algorithm is the splitting method. The standard test images Lena, gold hill, F-16, and Elaine, which are not part of the training set, are used to judge the algorithm.

Index compression methods such as Huffman coder, the Hu and Chang algorithm, and the search order coding (SOC) index compression, are used here for comparison.

The Huffman coder uses local Huffman tables. The performance of the remapping method is enhanced through the use of DPCM coding and Huffman coding, which uses global tables for the sake of more data compression. For the Hu and Chang method, the used index offset threshold is chosen to be 16.

In Tables 1, 2, 3, 4 and 5 the bit rates of the index compression methods are displayed with the corresponding codebook size. For small code book size (N=64), the average bit rate achieved by the proposed algorithm are almost the same as those achieved by DPCM+Huff. and less than the corresponding average bit rates from the other algorithms. By moderate codebook size (N=128) the proposed algorithm gives bit rates slightly less than those of the Hu and Chang method, which are less than the corresponding bit rates of Huff.+ DPCM and SOC. For higher code book sizes (N=256, N=512 and N=1024) the proposed algorithm gives lower average bit rates than all the other algorithms.

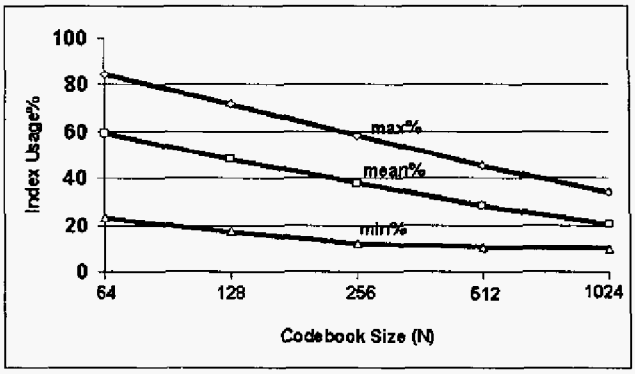

Fig. 3. The average minimum, mean, and maximum values of index usage percentage in the index blocks

Fig. 4. Illustration of index usage (N=64)

Table 1. Bit rate for code book size N=64

Image	Lena	GH	F-16	Elaine	Ave.
VQ	0.375	0.375	0.375	0.375	0.375
DPCM+Huff.	0.2509	0.2376	0.2019	0.2357	**0.2315**
Hu & Chang	0.2358	0.2565	0.2249	0.2487	**0.2415**
SOC	0.2495	0.2695	0.2510	0.2511	**0.2553**
Index Remap	0.2470	0.2381	0.2064	0.2315	**0.2307**

Table 2. Bit rate for code book size N=128

Image	Lena	GH	F-16	Elaine	Ave.
VQ	0.4375	0.4375	0.4375	0.4375	0.4375
DPCM+Huff.	0.3066	0.2968	0.2541	0.2959	**0.2884**
Hu & Chang	0.2673	0.2934	0.2577	0.2822	**0.2752**
SOC	0.2839	0.3201	0.2880	0.2916	**0.2959**
Index Remap	0.2912	0.2856	0.2644	0.2797	**0.2802**

Table 3. Bit rate for code book size N=256

Image	Lena	GH	F-16	Elaine	Ave.
VQ	0.5	0.5	0.5	0.5	0.5
DPCM+Huff.	0.3759	0.3643	0.3207	0.3673	**0.3570**
Hu & Chang	0.3250	0.3538	0.3084	0.3481	**0.3338**
SOC	0.3395	0.3883	0.3393	0.3601	**0.3568**
Index Remap	0.3420	0.3382	0.3086	0.3292	**0.3295**

Table 4. Bit rate for code book size N=512

Image	Lena	GH	F-16	Elaine	Ave.
VQ	0.5625	0.5625	0.5625	0.5625	0.5625
DPCM+Huff.	0.4553	0.4482	0.3846	0.4547	**0.4357**
Hu & Chang	0.3907	0.4376	0.3539	0.4355	**0.4044**
SOC	0.3950	0.4609	0.3870	0.4312	**0.4185**
Index Remap	0.3932	0.3948	0.3506	0.3847	**0.3808**

Table 5. Bit rate for code book size N=1024

Image	Lena	GH	F-16	Elaine	Ave.
VQ	0.625	0.625	0.625	0.625	0.625
DPCM+Huff.	0.5745	0.5643	0.4997	0.5724	**0.5527**
Hu & Chang	0.5007	0.5472	0.4381	0.5610	**0.5118**
SOC	0.4864	0.5561	0.4673	0.5380	**0.5120**
Index Remap	0.4637	0.4637	0.4210	0.4535	**0.4437**

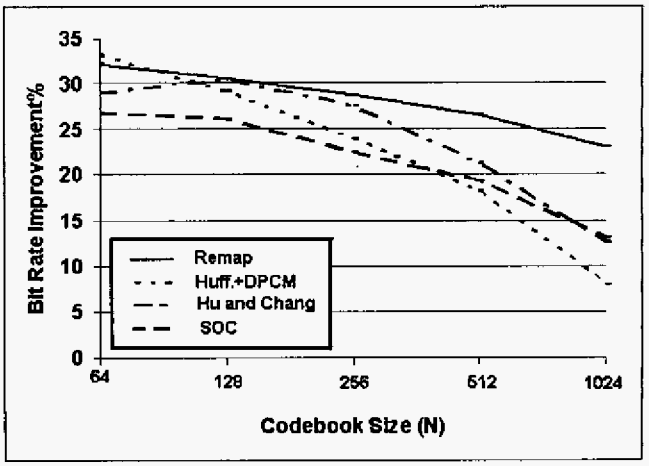

Fig. 5. Bit rate improvement (%) over VQ

Fig.5 shows the relation between the codebook size (log scale) and the percentage of bit rate improvement of different index compression algorithms over VQ. It shows that the proposed algorithm is less sensitive to the increase in codebook size compared with the other algorithms. The results represent the average of 10 standard images.

Generally, index compression algorithms cannot be cascaded, since the resultant output is stream of bits which does not contain any type of correlation. Index remapping, on the contrary, generates a new set of indices, thus any index compression algorithm can follow the remapping process to increase the compression efficiency. Fig.6 compares the bit rate when Hu and Chang method is used and when the same method is used but preceded by index remapping. Note that the overhead for sending the needed remapping information exceeds the improvement due to remapping by the small codebook size of 64.

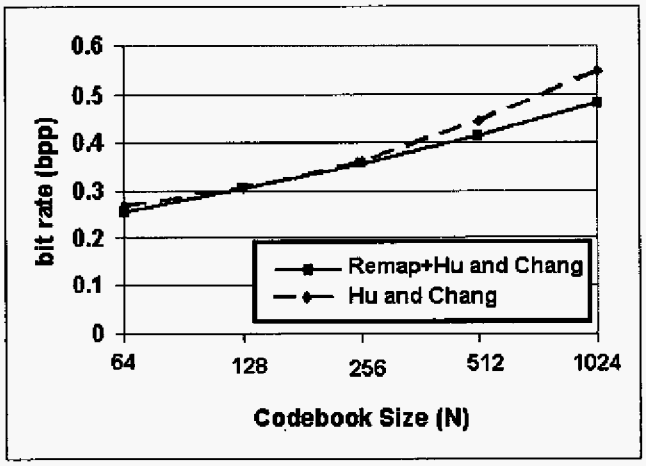

Fig. 6. The change of the bit rate for Hu and Chang method and Hu and Chang method preceded by index remapping with the Codebook size

5. CONCLUSION

This paper presents a new lossless compression algorithm for VQ indices. It uses the index usage activity in the image to construct index usage maps which define the used indices in different parts of the image. Based on the index usage maps, a remapping process is applied to the indices and the remapped indices are further compressed with DPCM and Huffman coder. The proposed algorithm is seen to improve the bit rate (bits/pixel) by a measured average value of 32.19% for codebook size of 64, down to 23.01% for codebook size of 1024.

REFERENCES

[1] P.C Cosman, K. I. Oehler, E. A. Riskin, and R. M. Gray,"Using Vector Quantization for Image Processing," Proc IEEE, Vol. 81, No. 9, pp. 1326-1341, Sept. 1993.

[2] A. Gersho and R. M. Gray, "Vector Quantization and Signal Compression". Norwell, MA: Kluwer, 1992.

[3] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," IEEE Trans. Commun., vol. COM-28, pp. 84–95, Jan. 1980.

[4] C. H. Hsieh and J. C. Tsai, "Lossless compression of VQ index with search order coding", IEEE Trans. on Image processing, vol. 5, No. 11, pp. 1579–1582, 1996.

[5] C. H. Hsieh, J. C Tsai, and P. C. Lu, "Noiseless Coding of VQ Index Using Index Grouping Algorithm," IEEE Trans. On Communications, Vol. 44, No. 12, pp. 1643–1648, Dec. 1996.

[6] M. H. Sheu, S.C. Tsai, and M. D. Shieh "A lossless index coding algorithm and VLSI design for vector quantization" AP-ASIC '99. The First IEEE Asia Pacific Conference on ASICs, 1999.

[7] Y. C. Hu and C. C. Chang, "Low complexity index-compressed vector quantization for image compression", IEEE Trans. on Consumer Electronics, vol. 45, No. 1 pp. 219–224, Feb. 1999.