IETE Journal of Research
Vol 48, Nos 3 & 4, May-August 2002, pp 211-215

Tree-Structured Index Grouping Algorithm for Lossless Coding of VQ Index Map

SATISH SINGH
Department of Electronics and Communication Engineering, Indian Institute of Technology, Guwahati 781 039, India.

AND

BHABATOSH CHANDA, FIETE
Electronics and Communication Science Unit, Indian Statistical Institute, Kolkata 700 035, India.

In this paper, a new algorithm is proposed for removing the spatial redundancy of the index values of the codevectors after encoding the image using Vector Quantization. This algorithm, called the Tree-Structured Index Grouping algorithm (TSIG), employs a combination of multipath search and the Index Grouping algorithm to encode the index map in a lossless fashion. After each search, based on the index values occurred in the search path, a particular bit pattern is transmitted. The performance of this algorithm is investigated on still 8-bit grayscale images and compared with the Index Grouping algorithm results. Also, a Finite State Vector Quantization (FSVQ) reordering procedure for selecting the subcodebook is investigated and the performance of the encoder is measured in terms of BPP, PSNR and encoding time. Further, to improve the performance of the algorithm, a combination of FSVQ and TSIG algorithm is proposed.

Indexing terms: Vector quantization, FSVQ, Index grouping, Tree-structured index grouping, Codebook, Subcodebook.

VECTOR Quantization (VQ) [1,2] has emerged as an efficient technique for image compression. In VQ an image is divided into blocks (usually of size 4 × 4) and the pixels in each block are arranged lexicographically to form vectors of dimension K. The VQ encoder considers each vector and generates the index of the codevector in the codebook that gives the minimum distortion. The distortion measure generally employed is the Euclidean distance. However, other distortion measures for have also been reported in literature[3]. The decoder uses the index values to pick-up the corresponding codevectors. Thus, we get a reconstructed image with some distortion. Thus the total distortion in the reconstructed image depends upon the codebook design. An excellent survey of the different VQ schemes may be found in [4]. An important class of memory VQ reported in literature is Finite State Vector Quantization (FSVQ) [5]. In this case the encoder in a particular state encodes an input vector by searching the corresponding subcodebook for the best representative vector. The state of the encoder is decided by the next-state function of the encoder [6,7]. By using a large number of states for the encoder and a small subcodebook size we can reduce the coding time and the bit rate. However, a major drawback of FSVQ is that the subcodebook may not always contain the best representative vector which pulls down the performance of FSVQ. It is observed that threre exists some correlation between the neighbouring index values in the index map after encoding the image using memoryless VQ (MVQ). By coding the index map in a lossless manner we can reduce the bit rate without introducing any extra coding distortion[8]. Recently Hsieh *et al* [9] proposed the Index Grouping (IG) algorithm which attempts to reduce the bit rate by grouping the identical index values and transmitting fewer number of bits for the group. In this paper we propose the Tree-Structured Index Grouping (TSIG) algorithm as an improvement on the IG algorithm. TSIG algorithm incorporates multipath search for grouping the index values. This results in larger groups thereby yielding a lower bit rate.

FINITE STATE VECTOR QUANTIZATION

The FSVQ [7] encoder consists of a finite set of subcodebooks, where each subcodebook belongs to a distinct state of the encoder. An input vector is encoded based on its corresponding subcode book and the corresponding index in the subcode book is transmitted. We present below the horizontal transition reordering procedure for the selection of subcodebook from the supercodebook.

In this approach the training images are coded using MVQ to obtain a score matrix S, where each element s_{ij} denotes the number of times the code-vector $\hat{x}_i$ is followed by codevector $\hat{x}_j$ in the horizontal direction. This matrix is sorted in a row-wise fashion to obtain the horizontal transition table H of size $N \times M$ ($M << N$). Each row of H contains the index of codevectors in the supercodebook arranged in decreasing order of the score obtained from S. The input image is encoded in a raster scan fashion. Based on the index (say i, in the supercodebook) of the previous image block the encoder will look for the best match among the codevectors having index values given in *ith* row of H. The codevector giving the least distortion is selected and its position in the subcodebook is transmitted. The transmission of the index value requires $\log_2 M$ bits as opposed to $\log_2 N$ bits for MVQ. Thus, this FSVQ technique results in a lower bit rate and encoding time than MVQ, while yielding acceptable image fidelty. The results of MVQ and FSVQ on some test images (256 x 256) are summarized in Tables 1 and 2.

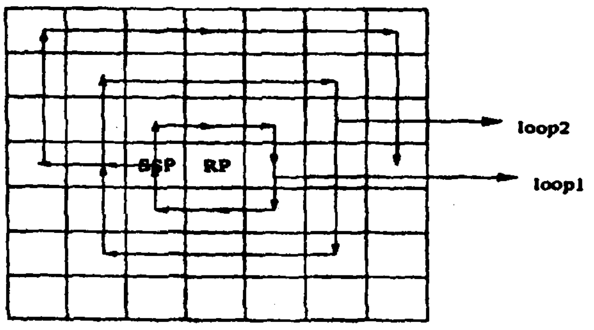
IG ALGORITHM

Index Grouping (IG) algorithm was proposed by Hsieh *et al* [9]. We begin by briefly explaining the terms used in the algorithm followed by an example for illustration. An index map is generated by using MVQ on the input image and every point in the index map is called a Search Point (SP). The starting point of group formation is called a Reference Point (RP). There exists a status map to prevent reduncant searches of SP's in the index map. Each status word in the status map is an N bit word and records the status of a SP. We also have a Search Order Code (SOC) which encodes the search order. Usually SOC is a 2-bit code for optimal performance (i.e. at most four valid SP's can be searched before a no match is declared). RP's are selected in a raster scan manner from the index map and all SP's except the ones that have already been grouped are eligible to become RP's.

The encoding of the index map is done as follows: Once the RP has been indentified, transmit a I bit (signalling bit) followed by the index value of the RP (say i) and set all the bits of the corresponding status word to 1 to indicate that this point has been grouped. Clear the search order and start searching starting from the Starting Search Point (SSP) along the path shown in Fig 1. Find a valid SP, if they do not match, set bit b_i of the status word of the SP to 1 to march as searched by i and increment search order to search the next valid SP. This is repeated till search

TABLE 1: Memoryless vector quantization

Image	Codebook size	PSNR (dB)	BPP	Time (units)
Lena	256	28.69	0.5000	3.29
	128	27.84	0.4375	1.60
Pepper	256	27.60	0.5000	3.30
	128	26.86	0.4375	1.57
F-16	256	25.06	0.5000	3.29
	128	24.46	0.4375	1.61
Boat	256	25.21	0.5000	3.31
	128	24.79	0.4375	1.64



SSP: Starting Search Point
RP: Reference Point

Fig 1 Search path for grouping the SP's

TABLE 2: FSVQ results for 256 x 256 images

Image	Subcodebook size	N = 256 BPP	N = 256 Time	N = 256 PSNR (dB)	N = 128 BPP	N = 128 Time	N = 128 PSNR (dB)
Lena	128	0.4375	1.77	28.69	-	-	-
	64	0.3750	0.91	28.69	0.3750	0.89	27.81
Pepper	128	0.4375	1.78	27.60	-	-	-
	64	0.3750	0.80	26.96	0.3750	0.90	26.37
	32	0.3125	0.50	26.01	0.3125	0.11	25.80
F-16	128	0.4375	1.77	24.48	-	-	-
	64	0.3750	0.90	23.92	0.3750	0.89	23.88
	32	0.3125	0.50	23.09	0.3125	0.11	23.10
Boat	128	0.4375	1.77	24.63	-	-	-
	64	0.3750	0.89	24.42	0.3750	0.89	24.50

order exceeds $2^2 - 1$, then a new RP is taken up for forming a new group. If a match is found, transmit a 0 bit (signalling bit) followed by the SOC and set all the bits of the corresponding status word to 1. This matched point is then taken as a RP, search order is cleared and a new search is performed again until no match is found. The encoding is schematically shown in Fig 2 using an example adapted from Hsieh *et al.*

TSIG ALGORITHM

The Tree-Structured Index Grouping (TSIG) alogrithm uses a combination of IG algorithm and a tree search for grouping the index values in the index map. The IG algorithm fails to group all the index values if they are distributed in star fashion. It tends to loose sight of the other nearby index values which may have resulted in much larger groups. TSIG algorithm tries to alleviate this drawback by looking into various directions for matching index values. In this algorithm, the terms SP, RP, index map, status map and search order have the same meaning as in the IG algorithm. In addition, we have used a simple data structure 'stack' characterized by its Top Of Stack (TOS) and having primitive operations of push and pop. The maximum search order in the TSIG algorithm is 2, which means that only 3 valid SP's will be searched in the search path. We also have a Search Indicator Word (SIW), which contains max {search order} + 1 bits. For example, if max {search order} = 2, then SIW 101 indicates that a match was found at SP's that occured at search orders 0 and 2.

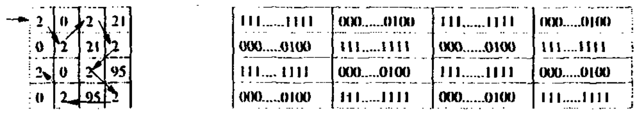
Encoding

TSIG algorithm encodes the index map as follows:

(1) Clear the status map.

(2) Identify the RP (let i = index value of RP) and the SSP, transmit the index value i of RP and set all the bits of the corresponding status word to 1.

(3) Clear search order and SIW and start searching from the SPP along the specified search path for valid SP's. At each valid SP compare i with the index value of the SP.



(*a*) Group formation using IG algorithm

$$\underbrace{1\ 00000010}_{\text{Index value}}\quad \underbrace{0\ 01\ 0\ 01\ 0\ 01\ 0\ 10\ 0\ 01\ 0\ 01\ 0\ 01}_{\text{Search Order Code}}$$

1,0 : signalling bits to mark index and search order code

(*b*) Bits transmitted for the group

Fig 2 IG algorithm

(4) If the index values do not match, set the ith bit of the corresponding satus word to 1 and the bit corresponding to search order in the SIW to 0 (to indicate a mismatch). Goto step 6.

(5) If a match is found, set all the bits of the corresponding status word to 1, set the search order bit of the SIW to 1 and push the block number of the matched SP into the stack.

(6) Increment the search order, keep checking the valid SP's according to steps 3 and 4 till the maximum search order is reached.

(7) Check the SIW. If all the bits are 0 then transmit a 0 bit indicating no match for the current RP in the search path. Goto step 9.

(8) If all bits are not 0, then transmit a 1 bit (signalling bit) followed by the SIW.

(9) Check the TOS. If stack is empty, then the grouping has finished and the encoder will form a new group. Goto step 2.

(10) If stack is not empty, pop the stack to get the block number of the matched SP, set this as the RP and goto step 3.

We now present a schematic representation of the TSIG algorithm on a test index map in Fig 3.

Decoding

Using the received bits and stack operations, the decoder is able to uniqely reconstruct the index map and the status map. The results of the IG algorithm and TSIG algorithm using different codebook sizes (N = 64, 128 and 256) are presented in Table 3.

From Table 3 we see that the TSIG algorithm gives improvement in bit rates for relatively uniform images and also gives almost similar performance with index grouping algorithm for N = 128. In order to reduce the bit rate further, we first encode the image using FSVQ to obtain the index map, this index map is then grouped using TSIG algorithm. The motivation behind FSVQ is to reduce the range of index values by using subcodebooks of smaller size. This increases the probability of finding identical index values in nearby positions. The first value of the index map is the index of the best representative vector of the first image block in the supercodebook. This is tranmitted using $\log_2 N$ bits and the status word of the first block which is only M bits long is set to 11...1. This is done because of the fact that the first index value may be greater than M whereas the status words are only M-bits long. Then for the remaining index map the TSIG algorithm is applied as described earlier in this section.

The decoder functions in a exactly opposite fashion: when it receives the first the $\log_2 N$ bits it reconstructs the index value and sets the status word of the first RP as 11....1. Then the same decoding process as described earlier takes over. After the index map is obtained, we decode the index map using FSVQ decoder to get the reconstructed image. In this way the bit rate is reduced though some more distortion is introduced due to FSVQ. The results of this coding scheme on different 256 x 256 images for various supercodebook and subcodebook sizes are summarized in table 4. We observe from table 2 that $N = 256$, $M = 128$ and $N = 128$, $M = 64$ are the best suited for the generation of the index map.

CONCLUSION

In this paper we have explored FSVQ and TSIG algorithm as techniques for bit rate reduction. In our simulations all the codebooks used were generated using the fuzzy c-means algorithm and the images Lena and Pepper were part of the training set of images. From Table 2 we conclude that FSVQ is considerably faster than MVQ, and a subcodebook size of 64 can be employed to yield good results. We observe that the TSIG algorithm performs better than IG algorithm on images having relatively smooth regions, as this ensures that more SP's can be grouped together in a single group. Also, as the size of codebook decreases the percentage reduction in bit rate increases, because the probability of finding matched index values in the search path is higher for a smaller codebook size. The better performance of this algorithm in such cases stems from the fact that it uses .ess number of bits to group the SP's, if all 2 (or 3) search points are matched then we require only 4 bits as opposed to (3 + 3 =) 6 (or (3 + 3 + 3 =) 9) bits using IG algorithm. Finally, in order to decrease bit rate further we used the TSIG algorithm to encode the index map generated by FSVQ using different subcodebook sizes ($M = 128, 64$) and found out the bit rates can be reduced even further while maintaining acceptable image fidelty. An example of results is shown in Fig 4 for $N = 256$ and $M = 64$. Also, using FSVQ to generate the index map serves to reduce the

TABLE 3: Comparison of results of IG and TSIG algorithms

Image	Codebook size	BPP			Time (units)		
		MVQ	IG algorithm	TSIG algorithm	MVQ	IG algorithm	TSIG algorithm
Lena	64	0.3750	0.2679	0.2615	0.78	1.03	1.52
	128	0.4375	0.3193	0.3184	1.60	1.89	2.05
	256	0.5000	0.3825	0.3906	3.29	3.66	4.77
Pepper	64	0.3750	0.2894	0.2574	0.80	1.04	1.12
	128	0.4375	0.3456	0.3089	1.60	1.88	1.92
	256	0.5000	0.4194	0.3766	3.29	3.65	3.69
Boat	64	0.3750	0.2791	0.2769	0.78	0.98	2.98
	128	0.4375	0.3277	0.3340	1.61	1.85	2.81
	256	0.5000	0.3925	0.4085	3.28	3.64	4.48
F-16	64	0.3750	0.2684	0.2382	0.79	0.99	4.59
	128	0.4375	0.3131	0.2864	1.62	1.86	3.49
	256	0.5000	0.3810	0.3615	3.28	3.68	3.84
Cman	64	0.3850	0.2474	0.2113	0.79	0.99	3.92
	128	0.4375	0.2867	0.2568	1.62	1.83	3.82
	256	0.5000	0.3293	0.3034	3.28	3.60	4.44

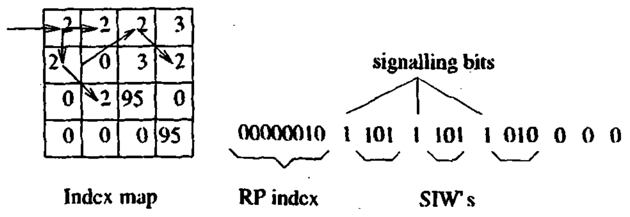


Fig 3 Schematic representation of TSIG algorithm using $N = 256$

TABLE 4 TSIG algorithm applied to index map generated by FSVQ

Image	BPP			Time (units)		
256 x 256	N=256 M=128	N=256 M=64	N=128 M=64	N=256 M=128	N=256 M=64	N=128 M=64
Lena	0.3520	0.3194	0.2891	2.22	1.65	1.74
Pepper	0.3887	0.3512	0.3171	2.10	1.12	1.59
Boat	0.3649	0.3264	0.2985	2.97	3.10	3.27
F-16	0.3235	0.2997	0.2682	3.64	4.69	4.80
Cman	0.2933	0.2683	0.2511	4.16	5.80	6.21

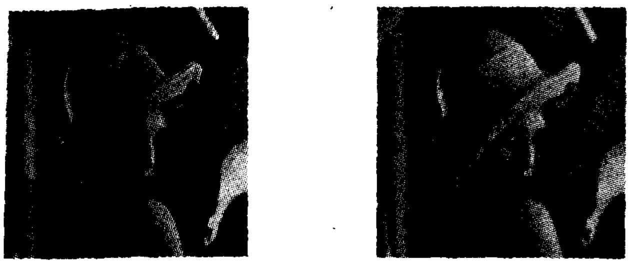


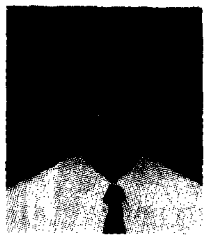
Fig 4 Compression results of Lena image. (*a*) Original, (*b*) FSVQ (with TSIG) for $N = 256$ and $M = 64$

encoding time of the algorithm. However, TSIG algorithm has its disadvantages. The performance of the algorithm deteriorates slightly for images having a lot of variation in graylevel value. Another disadvantage is that the memory requirement of the algorithm is relatively high.

REFERENCES

1. B Chanda & D Dutta Majumder, *Digital Image Processing and Analysis*, Prentice Hall of India, New Delhi, 2000.
2. Y Linde, A Buzo & R M Gray, "An algorithm for vector quantizer design, *IEEE Trans Commun*, vol COM-28, pp 84-95, Jan 1980.
3. R M Gray, A Buzo, A H Gray & Y Matsuyama, Distortion measures for speech processing, *IEEE Trans Acoust, Speech, Signal Processing*, vol ASSP-28, pp 367-376, Aug 1980.
4. N M Nasrabadi & R A King, Image coding using vector quantization: A review, *IEEE Trans Commun*, vol COM-36, pp 957-971, Aug 1988.
5. J Foster, R M Gray & M Dunham, Finite-state vector quantization for waveform coding, *IEEE Trans Inform Theory*, vol IT-31, pp 348-355, May 1985.
6. N M Nasrabadi & S A Rizvi, Next-state functions for finite-state vector quantization, *IEEE Trans Image Processing*, vol 4, pp 1592-1601, Dec 1995.
7. M Dunham, R Gray, An algorithm for the design of labelled-transition finite-state vector quantizers, *IEEE Trans Commun*, vol COMM-33, pp 83-89, Jan 1985.
8. C H Hsieh & J C Tsai, Lossless compression of VQ index with search-order coding, *IEEE Trans Imag Proc*, vol 5, pp 1579-1582, Nov 1996.
9. C H Hsieh, J C Tsai & P C Lu, Noiseless coding of VQ Index using index grouping algorithm, *IEEE Trans Commun*, vol COM-44, pp 1643-1648.

AUTHORS



Satish Singh born in 1980 and is a final year student of Electronics and Communication Engineering in the Indian Institute of Technology, Guwahati. Worked in a project under the supervision of Dr B Chanda in the Indian Statistical Institute, Calcutta in 2001. His research interests are Communication Systems and Image Processing.

* * *

Bhabatosh Chanda born in 1957. Received BE in Electronics and Telecommunication Engineering and PhD in Electrical Engineering from University of Calcutta in 1979 and 1988 respectively. Received "Young Scientist Medal" of Indian National Science Academy in 1989 and "Computer Engineering Division Medal" of the Institution of Engineers (India) in 1998. He is also recepient of UN fellowship, UNESCO-INRIA fellowship and Diamond Jubilee fellowship of National Academy of Science, India. He worked at Intelligent System lab, University of Washington, Seattle, USA as a visiting faculty from 1995 to 1996. He has published more than 70 technical articles. His research interest includes Image Processing, Pattern Recognition, Computer Vision and Mathematical Morphology. Currently working as Professor in the Indian Statistical Institute, Calcutta, India.

* * *