A NEW ADAPTIVE SUBBAND THRESHOLDING ALGORITHM FOR MULTISTAGE LATTICE VQ IMAGE CODING

M. F. M. Salleh and J. Soraghan

Signal Processing Division, Department of Electronic and Electrical Engineering,
University of Strathclyde, Glasgow G1 1XW Scotland, UK
Email: fadzli.salleh@eee.strath.ac.uk; j.soraghan@eee.strath.ac.uk

ABSTRACT

In this paper a new adaptive subband thresholding algorithm is introduced for finding the optimize threshold values of the subbands. The search technique is exhaustive, where the distortion level and bit allocation are used as the limiting factors. The algorithm is applied in multistage lattice VQ (MLVQ) to enhance the PSNR performance. The MLVQ image coding technique reduces the quantization error of the quantized vectors by "blowing out" the residual quantization errors with a LVQ scale factor. Experimental results for images using the adaptive subband thresholding technique and the MLVQ are shown to be superior to those of JPEG2000 for a set of test images.

1. INTRODUCTION

In recent years there have been significant efforts in producing efficient image compression algorithms based on the wavelet transform and vector quantization [1-3]. A typical image compression scheme searches for the significant subband coefficients by comparing them to a threshold value as the initial compression stage. This is followed by quadtree modeling of the significant data locations [1-2]. The threshold setting is an important entity in searching for the significant coefficients in the subbands. Image subbands at different levels of decomposition carry different weight of information. For general images lower frequency subbands carry more significant data than higher frequency subbands [4]. Also the subband coefficients at the same wavelet transform level have different statistical distributions. Therefore there is a need to optimize the threshold values for each subband.

The second level of compression is achieved via quantizing the significant coefficients (vectors). The LBG algorithm [5] has been widely used for vector quantization. However, it suffers from a high computational demand. Lattice vector quantization (LVQ) is an alternative technique to reduce computation due to its regular structure [6]. Multistage quantization is an effective way of reducing quantization errors. It is achieved by having a few quantizers in series to quantize the vectors as described in [7]. In [2] a multistage residual vector quantization based on [7] is used along with LVQ that produced results that are comparable to JPEG2000 [8] at low bit rates.

In this paper, we introduce a new algorithm for finding the optimized threshold of the subbands. The optimized threshold values are used to search the significant coefficients in the subbands. This technique is then incorporated with the multistage LVQ (MLVQ) technique and applies to image compression. The rest of the paper is organized as follows. Section 2 gives a review of lattice vector quantization. Section 3 presents the new image compression scheme. The results are presented in Section 4. Section 5 concludes the paper.

2. LATTICE VECTOR QUANTIZATION

Lattice is a regular arrangement of points in *k*-space that includes the origin or the zero-vector. A lattice is defined as a set of linearly independent vectors [6]:

$$\Lambda = \{X : X = a_1 u_1 + a_2 u_2 + \dots\dots + a_n u_n\} \qquad (1)$$

where $\Lambda \in \Re^k$, $n \le k$ a_i are integers, and u_i for $i = 1,2,\dots.,n$ are the basis vectors. The most common lattice type for vector quantization are the root lattices such as A_n, D_n, E_n, Z^n and their dual as described in [6].

Conway and Slone have developed an algorithm for finding the closest point of the n–dimensional integer lattice Z^n as reported in [9]. The Z^n or cubic lattice is the simplest form of a lattice structure. This suggests that finding the closest point in the Z^n lattice to the arbitrary point or input vectors in space $x \in \Re^n$ is simple. In [9] the fast quantizing algorithm for other root lattices such as A_n, D_n, E_n, and their dual were also developed.

The lattice quantizer codebook is designed by truncating the lattice into spherical or pyramidal shapes as described in [6]. In spherical codebook the following theta function is used [6];

$$\Theta_{\Lambda}(z) = \sum_{\Lambda} q^{x.x} = \sum_{m=0}^{\infty} N_m q^m \qquad (2)$$

where N_m is the number of lattice points lying on the surface of energy m (L_2 norm-squared or Euclidean distance squared) of the codebook from the origin. Table of N_m versus m can be found in [3] for Z^n and D_n spherical codebooks.

3. THE IMAGE CODING SCHEME

Figure 1 illustrates the encoder part of the MLVQ image coding scheme. A wavelet transform is used to transform the image into a number of levels. The significant coefficients of all subbands are identified, by comparing the vector energy to a certain threshold. The location information defined as a MAP sequence is represented in a quadtree structure. The search for significant coefficients starts from the lower frequency, which effectively follows the hierarchy order from parent to child through the subbands. Thus, this allows us to exploit the inter-subband MAP data redundancy. The significant coefficients or units are saved and passed to the multistage LVQ. The MLVQ produces two outputs i.e. the scale list and index sequence, which are then entropy coded, using the variable run-length coding with Golomb codes [10]. The lowest frequency subband is coded using the JPEG2000 lossless coding.

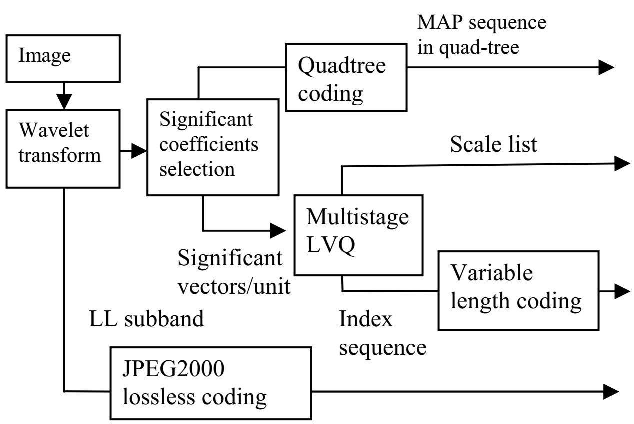

Figure 1: MLVQ image encoder

3.1 Adaptive Subband Threshoding

The threshold setting plays an important role in searching for the significant coefficients in the subband. A vector or unit which consists of the subband coefficients is considered significant if its normalize energy E defined as:

$$E = \frac{w(k)}{N_k x N_k} \sum_{i=1}^{N_k} \sum_{j=1}^{N_k} \left(X_k(i,j) \right)^2 \qquad (3)$$

is greater than a threshold T defined as

$$T = \left(\frac{ave}{100} \times \textit{threshold parameter} \right)^2 \qquad (4)$$

where X_k is a matrix in a particular subband k with dimension N_k, $w(k)$ is the perceptual weight factor and *ave* is the average pixel value of input image. The 'threshold parameter' has a valid value of 1 to 1000 and is chosen by taking into account the target bit rate.

The adaptive subband thresholding algorithm optimizes the threshold values by minimizing the distortion of the reconstructed image. The process is also restricted by the bit allocation constraint. The bit allocation is bounded using the amount of vectors available defined by (Eq 5). We define R as the target bit rate per pixel (bpp), r and c are the number of row and column of the image, D is a chosen wavelet transform level, and $LLsb\_bits$ is the amount of bits required to code the low-low subband and $othersb\_bits$ is the amount of bits required to code the remaining subbands, bit_{budget} is the total bit budget and V is the total vectors. The following relationships are defined

$$bit_{budget} = R \times (r \times c) = LLsb\_bits + othersb\_bits$$

$$V = \sum_{i=1}^{D} \frac{\left(othersb\_bits - 0.2 \times othersb\_bits - 3 \times 8 \right)}{\rho} \qquad (5)$$

$$\textit{where} \quad \rho = \begin{cases} 6, & n = 4, \\ 3, & n = 1, \end{cases}$$

The variable i represents the iteration index at every wavelet level. In this work we used 3 levels wavelet transform (D=3), and the Z^n lattice quantizer with codebook radius ($m = 3$). Hence, the denominator ρ is 6 (index representation as 6-bit) for $n = 4$ or 3 (index representation as 3-bit) for $n = 1$. Here (Eq 5) we are approximating 20% of the high frequency subband bits to be used to code the MAP data. The last term in (Eq 5) accounts for the LVQ scale factors, where there are 3 high frequency subbands available at every wavelet transform level and each of the scale factors is represented by 8 bits.

The adaptive threshold algorithm has three stages namely; *Initialization*, *Optimal Setup*, and *Threshold Optimization*. The first stage (*Initialization*) calculates the initial threshold using (Eq 4), and this value is used to search the significant coefficients in the subbands. Then the sifted subbands are used to reconstruct the image and the initial distortion is calculated. In the second stage (*Optimal Setup*) the algorithm optimizes the threshold between the wavelet transform levels. Thus in the case of a 3-level system there will be three threshold values for the three different wavelet levels.

An iterative process is carried out to search for the optimal threshold setup between the wavelet transform levels. The following relationship between threshold values at different levels is used:

$$level_1 \ threshold = \frac{initial \ threshold}{1}$$

$$level_2 \ threshold = \frac{initial \ threshold}{\beta}$$

$$level_3 \ threshold = \frac{initial \ threshold}{2\beta}$$

The parameter β is the threshold lowering factor. In the search process every time the value of β is incremented (starting from one) the distortion value and the number of vectors used are calculated. The process will stop and the optimized threshold values are saved once the current distortion is higher than the previous one.

The third stage (*Thresholds Optimization*), illustrated in figure 2 optimizes threshold value for each subband at every wavelet transform level. Thus there will be nine different optimized threshold values. The three threshold values found in stage 2 above are used for the "threshold parameter" expression derived from (Eq 4) as follows:

$$threshold \ \ parameter = \left(\frac{100}{ave}\right)\sqrt{level_i \ threshold}$$

$$where \ \ i = 1, 2, 3$$

In this stage the algorithm optimizes the threshold by increasing or lowering the "threshold parameter" as shown in figure 2.

The first process (*Find Direction*) is to identify the direction of the "threshold parameter" whether up or down. Then the (*Th_Param Up*) algorithm processes the subbands that have the "threshold parameter" going up. In this process, every time the "threshold parameter" value increases, a new threshold for that particular subband is computed. Then it searches the significant coefficients and the sifted subbands are used to reconstruct the image. The number of vector gain as well as the distortion may then be computed. The optimization process will stop, and the optimized values are saved when the current distortion is higher than the previous one or the number of vectors has exceeded the maximum allowed.

Finally, the (*Th_Param Down*) algorithm processes the subbands which have the "threshold parameter" going down. It involves the same steps as above before calculating the distortion. The vector gain obtained in the above step is used as the lower bound. The optimized values are saved after the current distortion is higher that the previous one or the number of vector has exceeded the maximum allowed.

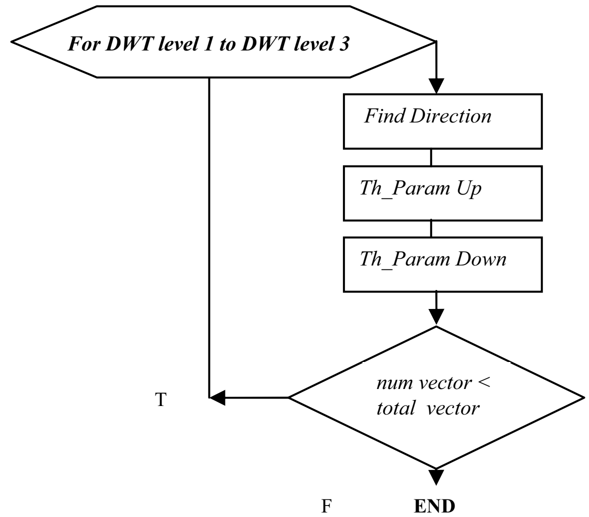

Figure 2: Thresholds Optimization (stage 3)

3.2 Multistage Lattice Vector Quantization (MLVQ)

The scaling and quantization process of the significant coefficients in each LVQ stage uses the modified technique presented in [3]. At stage one, we use the optimum setup (obtained from experiment) of codebook truncation where the significant or input vectors reside in both granular and overlap regions. At the subsequent stages the input vectors are forced to reside only in granular regions. At each LVQ stage, a spherical Z^n quantizer with codebook radius (m=3) is used. Hence, there are 64 lattice points (codewords) available with 3 layers codebook for $n = 4$ with 6 bits index representation [3]. If the origin is included, the outer lattice point will be removed to accommodate the origin. In one dimensional vector ($n = 1$), there are 7 codewords with 3 bits index representation.

The MLVQ first scale the vectors, and the scaled vectors are quantized. After quantization the vectors are checked to make sure that they are confined in the chosen spherical codebook radius. The vectors that exceed the codebook radius are rescaled and remapped to the nearest valid codeword. At every LVQ stage the quantization error vectors are obtained and "blown out" by multiplying the quantized vectors with the current scale factor. They are then used as the input vectors for the subsequent LVQ stage, and this process repeats up to stage M. The MLVQ algorithm is described as follows;

1. Determine the multistage level M
2. For i =1:M do the following;
   - Scale the significant vectors (i=1) or input vectors, and save into a scale record
   - Vector quantize the scaled vectors, and save into a quantized vectors record

- Quantization error vectors = (scaled vectors – quantized vectors) x significant vectors scale (i=1) or input vectors scale.
- Input vector = quantization error vectors

3. Repeat steps 1 and 2 for every subband.
4. If allocated bits are still available, continue refinement of the quantized vectors from the lower frequency subband towards the higher frequency.

In this algorithm, the quantization errors are produced for an extra set of input vectors to be quantized. The advantage of "blowing out" the quantization error vectors is that they can be mapped to many more lattice points during the subsequent LVQ stages. Thus the MLVQ can reduce quantization errors and produce better image quality**.**

4. RESULTS

The grey "goldhill" and "camera" images of size 256x256 are used for simulation. The test images are decomposed into three wavelet transform levels. The unit size is set to 2×2 producing four dimensional vectors in the first level, and 1×1 producing one dimensional vectors for all other subbands. The performance results of the codec with and without adaptive threshold are compared with JPEG2000 [8] as shown in table 1 and 2 respectively. It is clear that using the adaptive subband thresholding algorithm improves the performance over the constant subband thresholding of MLVQ image coding scheme. The overall performance is superior to JPEG2000. At 0.15bpp the new scheme is 1.45dB better PSNR than JPEG2000 for "goldhill" and 2.42dB better PSNR than JPEG2000 for "camera". Figure 3 shows the visual comparison at 0.15bpp.

5. CONCLUSIONS

The new adaptive subband thresholding algorithm which is restricted by the bit allocation constraint increases the performance of the MLVQ image codec. The MLVQ technique presented in this paper refines the quantized vectors, and reduces the quantization errors. Thus the new image compression outperforms JPEG2000 at very low bit rates for the tested images.

6. REFERENCES

[1] S. P. Voukelatos and J. J. Soraghan, "Very low bit-rate color video coding using adaptive subband vector quantization with dynamic bit allocation," IEEE Trans. Circuits and Systems for Video Technology, vol. 7, no,2, pp.424-428, April 1997.

[2] H. Man, F. Kossentini, and M. J. T. Smith, "A Family of Efficient and Channel Error Resilient Wavelet/Suband Image Coders," IEEE Trans. on Circuit and System for Video Tech., vol.9, no.1, Feb. 1999.

[3] M. Barlaud, P. Sole, T. Goidon, M. Antonini, and P. Mathieu, "Pyramidal Lattice Vector Quantization for Multiscale Image Coding," IEEE Trans. Image Processing, vol. 3, no. 4, pp. 367-381, July 1994.

[4] A. O. Zaid, C. Olivier, and F. Marmoiton, "Wavelet image coding with adaptive dead-zone selection: application to JPEG2000," IEEE International Conference on Image Processing, vol. 3, pp. 253-256, June 2002

[5] Y. Linde, A. Buzo, and R. M. Gray, "An Algorithm for Vector Quantizer Design," IEEE Trans. Communication, vol. COM-28, pp. 84-95, Jan. 1980.

[6] J. H. Conway and N. J. A. Sloane. Sphere Packings, Lattices, and Groups. New York: Springer-Verlag, 1988.

[7] F. Kossentini, M. Smith, and C. Barnes, "Necessary conditions for the optimality of variable rate residual vector quantizers," IEEE Trans. Inf. Theory, vol. 41, pp. 1903-1914, Nov. 1995.

[8] M. Rabbani and R. Joshi, "An overview of the JPEG2000 Still Image Compression Standard," Signal Processing: Image communication, vol. 17, no. 1, October 2001.

[9] J. H. Conway and N. J. A. Slone, "Fast Quantizing and Decoding Algorithms for Lattice Quantizers and Codes," *IEEE Trans. Information Theory*, vol. IT-28, pp. 227-232, March 1982.

[10] S.W. Golomb, "Run-length Encodings", IEEE Trans. Inf. Theory, vol. IT-12, pp. 399-401, 1966.

TABLE I
PSNR (DB) FOR IMAGE GOLDHILL (256X256)

Bit rate (bpp)	JPEG2000	MLVQ Th Constant	MLVQ Th Adaptive
0.05	22.80	23.08	23.20
0.10	24.34	24.52	25.37
0.15	25.03	25.33	26.48
0.20	25.72	25.89	26.80
0.30	27.06	27.20	28.30
0.50	29.03	29.20	29.87

TABLE II
PSNR (DB) FOR IMAGE CAMERA (256X256)

Bit rate (bpp)	JPEG2000	MLVQ Th Constant	MLVQ Th Adaptive
0.05	20.90	20.96	21.67
0.10	23.49	23.18	25.46
0.15	24.28	24.78	26.70
0.20	25.41	25.51	27.03
0.30	27.61	27.84	29.21
0.50	30.59	30.76	31.68

Figure 4: JPEG2000 (24.28dB), MLVQ (26.70dB) at 0.15bpp