Embedded Wavelet Coder with Multistage Vector Quantization*

Kai-Chieh Liang† Jin Li † C.-C. Jay Kuo †

August 15, 1996

Abstract

An embedded image coder based on wavelet transform coding and multistage vector quantization (VQ) is proposed in this research. We have examined several critical issues to make the proposed algorithm practically applicable. They include the complexity of embedded VQ, design of the successive vector quantizer, significance evaluation of a vector symbol, and integration of wavelet transform coding and vector quantization. It is shown in experiments that the new method achieves a superior rate-distortion performance.
Keywords: image compression, vector quantization, wavelets, embedded coding.

1 Introduction

In this research, we examine the design of an embedded image coder by integrating vector quantization with the wavelet transform. An embedded coder is able to generate a bit stream in the order of importance so that it can be terminated at any point with a perceptible reconstructed image. This property is ideal for unequal error protection and rate control, and has applications in visual communication over noisy channels, progressive image transmission, browsing and retrieval. Vector quantization (VQ) is a quantization technique applied to an ordered set of symbols. The application of VQ to speech and image coding has been extensively studied by researchers. The superiority of VQ lies in the block coding gain, the flexibility in partitioning the vector space, and the ability to exploit intra-vector correlations. Subband and wavelet decompositions have also been applied to image compression for years. The superiority of wavelet-based coding over the DCT-based coding was demonstrated by the embedded zerotree wavelet coding (EZW) [8] and the layered zero coding (LZC) [10]. Several image coding ingredients were adopted in EZW. They include: zerotree prediction of insignificance across scales, significance

*This work was supported by the Integrated Media Systems Center, a National Science Foundation Engineering Research Center, the National Science Foundation Presidential Faculty Fellow Award ASC-9350309 and the AFOSR Focus Research Initiative (FRI) Award F49620-95-1-0531.

†The authors are with the Integrated Media Systems Center and the Department of Electrical Engineering-Systems, University of Southern California, Los Angeles, California 90089-2564. E-mail: kliang@sipi.usc.edu, lijin@sipi.usc.edu and cckuo@sipi.usc.edu.

evaluation of wavelet coefficients, successive approximation quantization (SAQ), and adaptive arithmetic coding. In this work, these concepts will be generalized from the scalar to the vector case, and will be combined with VQ.

Attempts have been made in combining VQ with the wavelet transform, e.g. [2], [5]. One example is to apply tree structure vector quantization (TSVQ) to wavelet coefficients, which results in a wavelet/VQ compression method. Since decoded data can be progressively refined as they proceed through the tree structure, TSVQ provides an embedded coding scheme. However, these algorithms did not propose any importance measure of the vector symbol and treated each vector in the wavelet domain equally important. The transform energy is compacted in only a few coefficients, and it is difficult to measure the significant vector as done in the scalar case. Consequently, it is challenging to encode symbols layer by layer according to their significance and to allow more refinements on important symbols. As a result, many bits are wasted in refining trivial vectors in the wavelet domain, and the performances of these algorithms are much inferior to those of EZW and LZC.

Silva, Sampson, and Ghanbari proposed a successive approximation vector quantization for wavelet coding [9], where each vector is coded progressively by reducing the error vector from the last approximation. A lattice codebook and a series of decreasing magnitudes are used for error vector refinement. The lattice codebook consists of a set of unit lattice directions in a certain vector space. The vector norm is used to measure the vector significance just as the role of the magnitude of each scalar coefficient in EZW. Thus, the developed algorithm can encode vectors in terms of their importance and more refinements can be done upon significant vectors. This algorithm works very well. Its performance is better than EZW and close to LZC. However, for the performance of this algorithm to be competitive with LZC, the dimension of the lattice has to be very large (around 16). A codebook of a 16-dimensional lattice contains more than 4000 codewords. To perform the refinement of an error vector, the encoder has to search through the huge codebook to find out one closest direction vector. This procedure introduces very high computational and storage complexities, which are the major problem of this approach.

The fundamental problems associated with the combination of embedded wavelet coding and VQ can be summarized as follows: (1) proper definition of vector significance, (2) development of successive vector refinement procedure, and (3) complexity of VQ. In this paper, we will carefully address each issue and propose a new algorithm which handles these problems satisfactorily. This paper is organized as follows. The basic building components and the detailed implementation of the proposed algorithm are presented in Sections 2 and 3, respectively. Experimental results are provided in Section 4 to demonstrate the performance of the proposed algorithm. Discussion and concluding remarks are given in Sections 5 and 6, respectively.

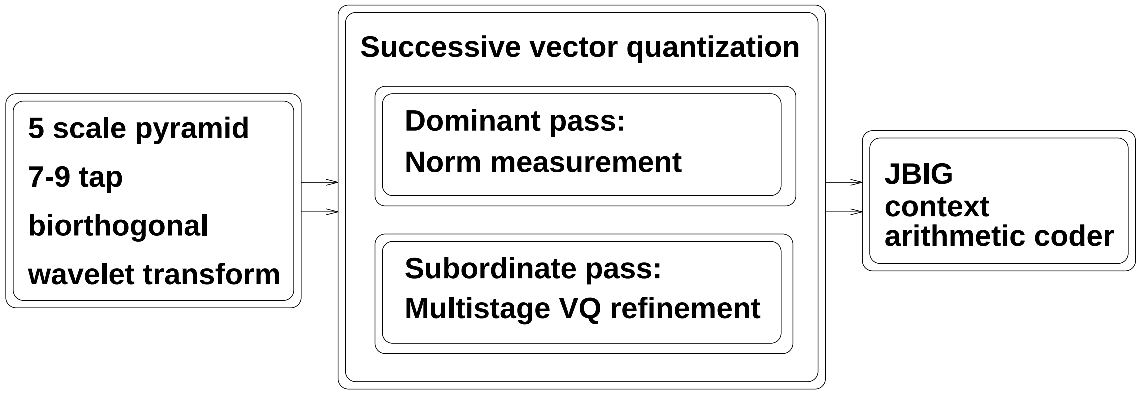


Figure 1: The framework of the proposed wavelet/VQ coding method.

2 Algorithm Overview and Successive Vector Refinement

2.1 Overview of Proposed Algorithm

The blockdiagram of the proposed algorithm is shown in Fig. 1. It contains three basic building blocks: the 7-9 tap biorthogonal wavelet transform, successive vector quantization, and the JBIG context arithmetic entropy coder. The adoption of the 7-9 tap filer is due to its demonstrated superior performance in image compression [11]. The JBIG arithmetic coder is used to perform entropy coding. The context-based arithmetic coder significantly improves entropy coding efficiency by exploiting the correlation of the current bit with its neighbors and parents in bit layers. These two building blocks have been fully discussed in the wavelet compression literature [11], [10]. The new component of this algorithm is the quantization stage in the middle. In this work, we develop a layer (precision) to layer successive vector quantization (SVQ) scheme. SVQ consists of two basic operations: vector significance measure and successive refinement. To achieve embedded coding, vectors have to be progressively refined and encoded. We propose a new method to separate vectors into several coding layers of decreasing importance, and encode one layer of symbols at one time. In particular, we select a significance measure based on the vector norm and then perform the multistage VQ for vector refinement. More details are given in Sections 2.2 and 2.3.

2.2 Vector Significance Measure

A measure of vector significance has to be chosen to separate vectors according to their importance. Possible significance measures include: the norm, the variance, and the location of a vector. When the norm is used to evaluate significance, symbols with large norms are regarded as important. The vector norm is consistent with the significance definition adopted by EZW, where the scalar magnitude is used. Another measure of significance is the variance within a vector. Vectors with higher variances usually correspond to edges or boundaries which are more sensitive to the human visual system. Also, the bit rate required to encode a vector are strongly related to the variances of the vector. The variance measure is often adopted by classified VQ for vector classification and bit allocation to improve coding efficiency. Classified VQ usually spends more bits on symbols with a larger variance and treat them as important symbols. Another measurement of significance is the location of vectors. In wavelet/subband coding, the location of vectors (e.g. subband) is not equally important. Subbands in coarser scales, which offer low frequency information and have a higher percentage in energy distribution, are more important than those in finer scales. It is worthwhile using more bits to encode symbols in these subbands. We use the norm to evaluate the importance of symbols in this work due to the following reasons. When the dc component is removed, a symbol with a larger norm is the same as that with a larger variance (or energy) value. The norm also provides a good indicator of subband locations, since most vectors with a large norm appear in coarse-scale subbands.

2.3 Successive Vector Refinement

There are three major concerns in developing the vector refinement scheme. First, it should be able to provide successive approximation, since embedded coding is our main goal. Second, its efficiency has to be better than that offered by the scalar quantization method. Otherwise, there is no need to perform vector quantization at all. Third, the complexity has to be controled under a reasonable level for the mothod to be practically useful.

Successive approximation can be achieved from two aspects: iteratively approaching to the true value, or progressively reducing approximation errors. The first viewpoint concentrates on approximating vectors while the second viewpoint focuses on error vectors. Even though they are equivalent conceptually, they can lead to different detailed implementations. In the context of VQ, they result in tree structure VQ (TSVQ) and multistage VQ, respectively. TSVQ provides an efficient vector quantization method with a natural progressive property, and has been extensively studied in the VQ research community. It significantly reduces the computational complexity of VQ encoding by replacing the exhaustive full search with the binary tree search as shown in Fig. 2 (a). Multistage VQ is an error refinement scheme. At each stage, it receives the input from the previous stage, determines the best codeword, calculates the residual vector between the input and the codeword, and then sends the residual vector to the next stage. The whole procedure iterates until the desired precision is reached. Fig. 2 (b) shows the codebook structure of multistage VQ and residual codewords at each stage. As shown in the figure, the

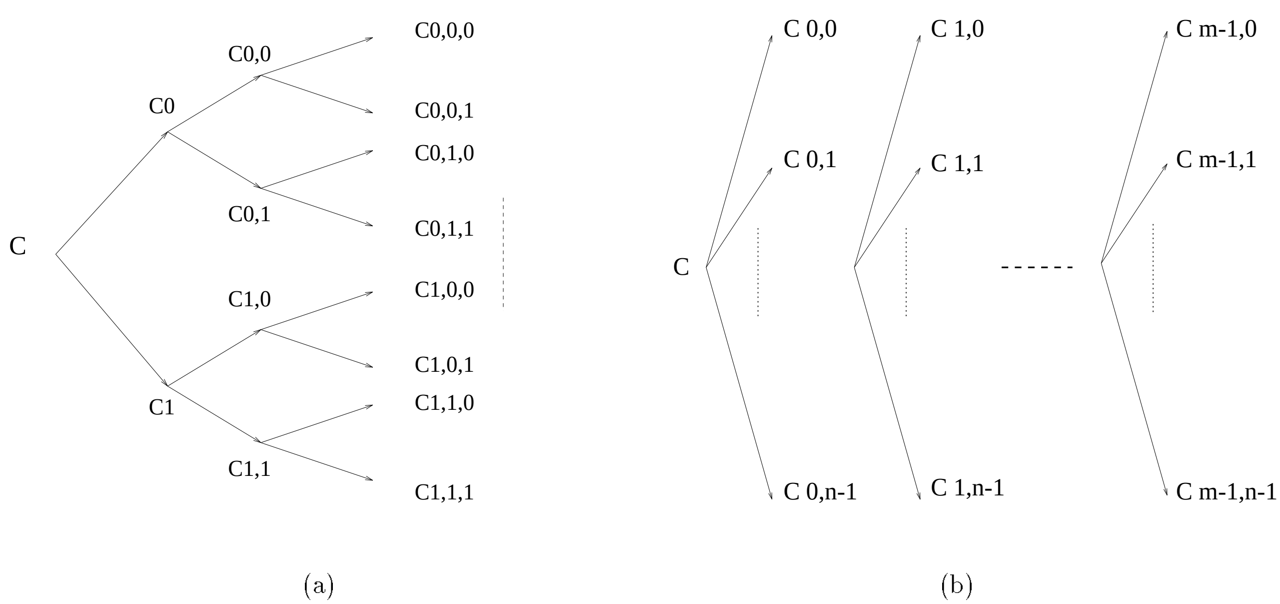


Figure 2: (a) Tree structure VQ codebook, and (b) multistage VQ codebook.

codebook size of multistage VQ grows linearly rather than exponentially as that of TSVQ. Both methods have the embedded coding property.

In successive scalar quantization, the refinement of a scalar symbol is a binary decision in a binary tree, i.e. one pixel symbol receives one bit refinement. For a successive vector quantizer to have quantization precision equal to that of a scalar quantizer, VQ must spend no more than one bit to refine each pixel component in the vector for a better rate-distortion tradeoff. For the case of two dimensional vectors, similar bit rates can be achieved by the refinement using two one-bit one-layer scalar quantizers, one two-layer binary TSVQ or one multistage VQ with four codewords. Similarly, for a four dimensional vector, we can achieve the same efficiency by using either the four-layer binary TSVQ or the multistage VQ with 16 codewords.

Another common concern of VQ is its computational and storage complexities. The codebook size provides a good indication for both quantities. In Table 1, we show the number of scalar refinement layers, and four dimensional codebook sizes of TSVQ and multistage VQ. For the coding of the Lena image with LZC, it takes six layers of scalar refinements to reach image quality of 0.25 bit per pixel (bpp). Thus, to achieve a comparable performance with a four dimensional VQ, TSVQ requires a binary tree of 24 layers and multistage VQ requires 6 stages with 16 codewords at each stage. The codebook size of TSVQ is more than 2^{24} while that of multistage VQ is only 96. It is clear that the complexity of TSVQ is too high to be practical. In

Bit rate (bpp)	0.1 bpp	0.25 bpp	1.0 bpp
No. of scalar refinement layer	5	6	8
No. of TSVQ layer	20	24	32
Codebook size of TSVQ	2^{20}	2^{24}	2^{32}
No. of multistage VQ layer	5	6	8
Codebook size of multistage VQ	80	96	128

Table 1: Comparison of the scalar quantizer, TSVQ and multistage VQ.

contrast, multistage VQ has a much lower complexity. Thus, it is chosen as the vector refinement scheme in our algorithm.

3 Implementation Details

3.1 Codebook Design

There are nine images in our training image set: facial, Vegas, wool, grass, F16, tank, mall, creek and peppers. Before the training procedure, images are transformed by a five-scale pyramid 7-9 tap biorthogonal filter as shown in Fig. 3. There are totally 16 subbands in the wavelet domain. The first issue we face is whether to assign different codebooks to different subbands. After consideration, we do not incorporate subband classification in our algorithm. Complexity is one reason. Another more important reason is that subband classification does not help much in the context of multistage VQ. Since multistage VQ is an error refinement scheme, inputs to a stage are residual vectors from the previous stage and they tend to be less and less correlated as the process proceeds. Input vectors gradually become more random and white regardless of the location of original subbands.

As discussed in Section 2, vectors of wavelet transformed images are separated according to their norms into several quantization layers. A sequence of thresholds are chosen, where the first threshold T_1 is one half of the maximum norm of all vectors, the second threshold T_2 one half of the first one, and so on. Then, the first layer contains vectors whose norms are above the first threshold, the second layer those between the first and second thresholds, etc. Clearly, norms (and energy) of vectors in each layer are totally different from those in other layers. The first layer has a multistage codebook with codewords of larger norms than those of the second layer. Thus, different layers correspond to different subspaces, which are not overlapping with each other.

One simple method to reduce the number of codebooks is to use the scaled-down versions of the first codebook for the rest of layers. That is, the second codebook is the first codebook scaled down by 2, the third one scaled by 4, and etc. In such a way, one codebook can be used

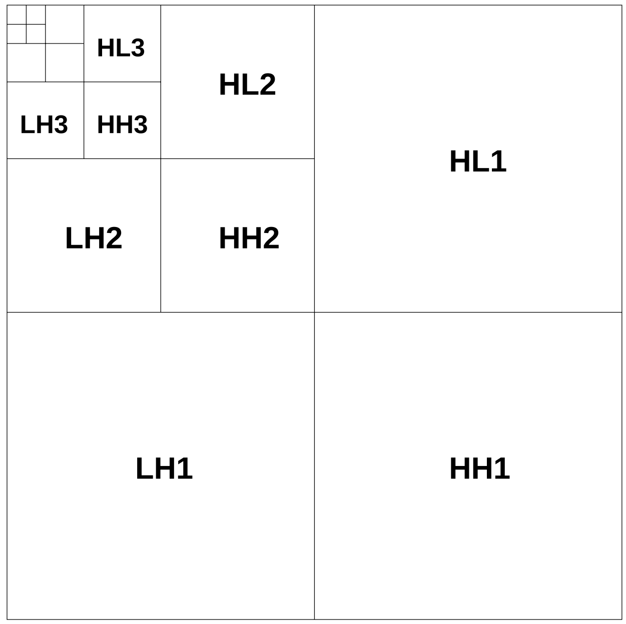


Figure 3: Five-scale QMF pyramid.

for all subbands and layers. The training procedure for the first codebook is detailed below.

1. Transform all test images by using the 7-9 tap biorthogonal filter.
2. Group neighboring wavelet coefficients into one vector.

   For the four dimensional case, 2 by 2 wavelet coefficients are grouped into a vector.
3. Take absolute values of all vector components.

   Since signs and absolute values of vector components are encoded separately in our algorithm, we consider only the magnitude of each vector component in the refinement process.
4. Find the training vectors for the first layer codebook.

   This can be done by two different ways. One is to include all training vectors of the first layer, i.e., symbols with norms larger than the first threshold T_1. Another is to manipulate the components of vectors, e.g. multiplied by 2 or 4, so that all the vectors fall in the subspace of the first layer. The latter approach contains a much larger training set and richer patterns than the former one. We choose the second method in our coding scheme.
5. Perform multistage codebook training.

   The codebook training includes: find the centroid of the training set, and the residual codewords of the first stage, second stage, and etc. The training method is Lloyd-Max iteration, which is often referred to as GLM or LBG [6].

In this work, we do not design a different codebook for each individual layer. The same codebook is applied to all subbands and layers.

3.2 Image Coding

In the following discussion, we use a four dimensional vector and a test image of size 512×512 as an example. The first-order norm (i.e. summation of the absolute values of all vector components) is chosen as the norm measurement for simplicity. The same algorithm can be easily extended to any vector size and any kind of norm measurement. The image encoding procedure can be stated as follows.

1. Transform the original spatial image to the wavelet domain by using a 5-scale 7-9 tap biorthogonal wavelet filter.
2. Group 2 by 2 neighboring wavelet coefficients into 1 four dimensional vector so that the wavelet image contain 256 by 256 vectors.
3. Record the signs and norms of each vector in the wavelet domain.
   Signs can be represented by binary digits, where 1 stands for positive and 0 for non-positive.
4. Take absolute values of all vector components.
5. Decide the initial threshold of successive vector quantization.
   Use one half of the maximum norm as the initial threshold T_1 and the second threshold T_2 be $0.5 * T_1$, and so on.
6. Perform successive vector quantization (SVQ).
   SVQ encodes symbols layer by layer according to their importance. SVQ refines the symbols with norms larger than the initial threshold in the first layer and those larger than the second threshold in the second layer, and so on. There are two data maps involved in SVQ. One is the binary significance map of size 256×256, which records the significance of each wavelet vector. Another is a vector reconstruction map, which contains 256×256 vectors of dimension 4. The reconstruction map is used to record the current reconstruction contents of vector symbols during the progressive quantization process. There are two operations in SVQ: the dominant pass and the subordinate pass. The implementation details of SVQ are described below.
   (a) The first dominant pass.
       Compare the norms of all vectors with the initial threshold T_1. If a symbol has a norm larger than the threshold, it is recorded as 1 in the significance map. This significant bit together with the signs of the vector are sent to the entropy coder. Meanwhile, those significant vectors are reconstructed to the centroid of the codebook, and the first residual vectors are formed between the centroid and significant vectors. If a

symbol has a norm smaller than the threshold, it is recorded as 0 in the significance map. Only 0 (i.e. the insignificance digit) is sent to the entropy coder. There is no need to send the signs of the insignificant vector.

(b) The first subordinate pass.

The significant symbols that have been found in the first dominant pass will be refined by the first stage of multistage VQ. In the four dimensional case, the first stage of multistage VQ contains 16 codewords for the first residual vectors, and the same 16 codewords for the second residual vectors, and so forth. The first residual vectors formed in the first dominant pass are quantized to one of the codewords of the first stage, and a new set of second residual vectors is formed between the first residual vectors and their corresponding codewords. The codeword of each vector is saved as its content in the reconstruction map and the index of codeword is sent to the entropy coder. All the information, including significance bits, signs, and indices of codewords, are coded by the entropy coder, i.e. the JBIG context-based arithmetic coder.

(c) The second dominant pass.

All vectors that have not yet been found to be significant are compared to the new threshold, $T_2 = 0.5 * T_1$ to determine their significance. If a vector is significant, it is reconstructed to the centroid of the codebook, and its first residual vector is formed between the centroid and the vector. As before, the significant bit, 1, and the signs of the vector are sent to the entropy coder. It is worthwhile to point out that only new significant bits are sent to the entropy coder. For symbols that have been identified as "significant" before, there is no need to send significance bits again and the entropy encoder just skips those bits in the significance map. If the symbol has norm smaller than the threshold, its insignificance bit (i.e. 0) is sent to the entropy coder.

(d) The second subordinate pass.

For the first residual vectors generated in the recent dominant pass, they are refined by the first stage of multistage VQ. For the second residual vectors in the last dominant pass, they are refined by the second stage of multistage VQ. Again, the corresponding codewords are saved as contents of vectors in the reconstruction map, and new residual vectors are generated. The indices of codewords are sent to the entropy coder.

(e) Repeat the same procedure for dominant and subordinate passes, until the JBIG coder reaches the target bit rate.

In decoding, the decoder basically performs the reverse process of the above algorithm.

Test Images	Fingerprint	Bark	Leather	Barbara	Baboon	Boat	Lena
Proposed Method (dB)	28.52	15.55	13.27	24.83	21.30	26.49	29.89
LZC (dB)	28.09	15.44	13.17	24.69	21.24	26.47	29.98

Table 2: The PSNR performance for the coding of several test images with the proposed method and LZC at 0.1 bpp.

Bit Rates (bpp)	0.25	0.2	0.15	0.1
Proposed Method (dB)	28.34	27.40	26.55	24.83
LZC (dB)	28.12	27.31	25.54	24.69

Table 3: Comparison of rate-distortion performances for the coding of the Barbara image with the proposed method and LZC.

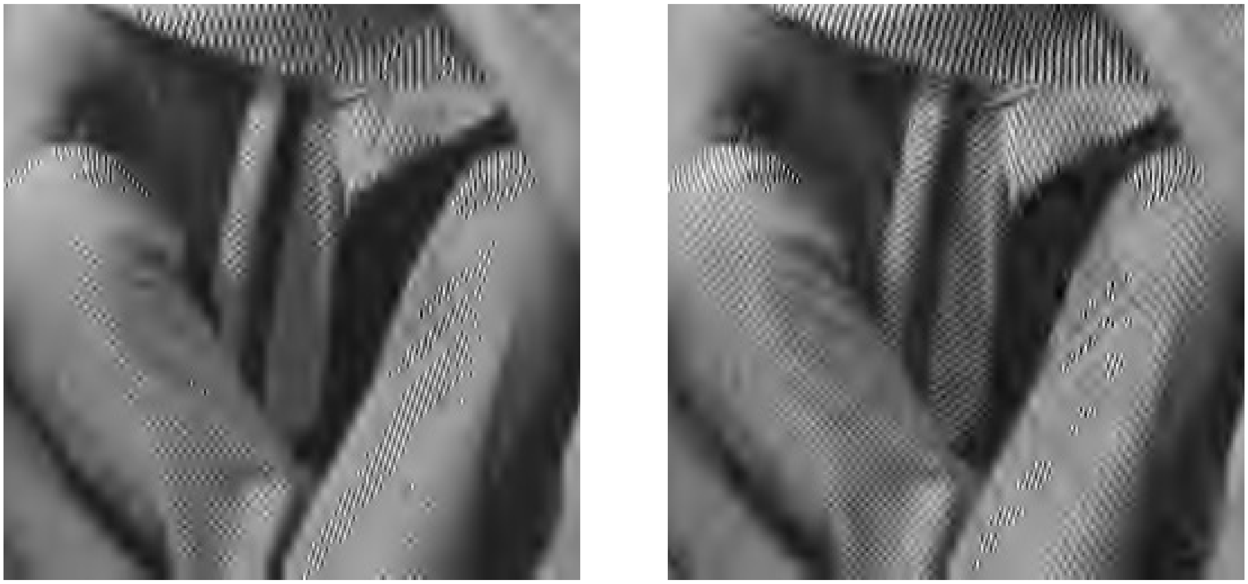
4 Experimental Results

Our new method was applied to several 8-bit gray level test images, including fingerprint, bark, leather, Lena, boat, Barbara, and baboon. The first three are textured images, and others are standard test images often used in the literature. The fingerprint image is of size 768×768 and the other images are all of size 512×512. The vector dimension is 2×2. The wavelet transform is a five-scale pyramid. The performances of the new method were compared to those of LZC, which is one of the state-of-the-art wavelet image coder and has around 1 dB gain over EZW in PSNR for typical test images.

We list the coding results in Tables 2 and 3. As shown from the tables, our new method is superior to LZC for almost every test images at 0.1 bpp. It is particularly effective in the coding of texture dominant images such as fingerprint, bark or Barbara. Decoded results are shown in Figs. 4, 5, 6 for visual comparison, where the textured regions were demonstrated to show the differences between the two coding methods. It is clear that our method provides a better texture information than LZC. Barbara is often considered as a very difficult test image by researchers. Additional results for Barbara are provided in Table 3. where our new method outperforms LZC at every bit rate. The PSNR difference is even up to 1 dB at 0.15 bpp.

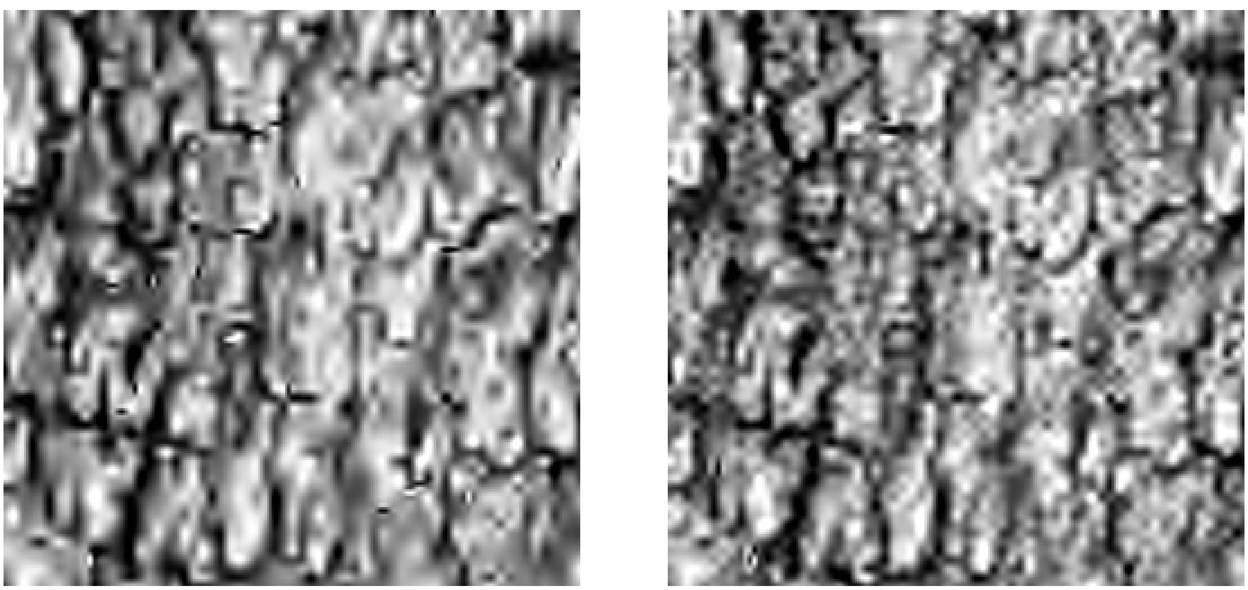
5 Discussion

The embedded wavelet coding with multistage VQ has successfully demonstrated its ability to achieve an excellent rate distortion performance, especially at very low bit rates. The superior performance of the proposed method can be explained by the following reasons. First, VQ is a block length coding method, which can approach the entropy rate more effectively than the scalar



(a) (b)

Figure 4: Barbara image at 0.1 bpp with (a) LZC and (b) the proposed method.



(a) (b)

Figure 5: Bark image at 0.1 bpp with (a) LZC and (b) the proposed method.

(a) (b)

Figure 6: Fingerprint image at 0.1 bpp with (a) LZC and (b) the proposed method.

coding. Second, vector quantization uses the correlation within vector to improve quantization efficiency and reduce quantization errors. Third, the training set consists of wavelet coefficients from one LL subband and HH, LH and HL subbands of five-scale QMF-pyramid as shown in Fig. 3. Thus, the training set basically consists of high frequency and high variant patterns, and the resulting codebook after Lloyd Max iterations will be more representative of patterns like textures or edges.

It is worthwhile to comment on similarities and differences between our method and LZC. First, LZC (and EZW) adopts the scalar binary tree structure refinement method to approximate the input symbol successively. In the scalar domain, successive approximation to the input symbol is equivalent to successive minimization of residual errors. Thus, TSVQ and multistage VQ are identical for the scalar case. However, they behave differently when the vector size is larger than 1. We showed that the multistage VQ should be used instead of TSVQ. Second, LZC has an equal quantizer interval throughout the whole successive quantization procedure. In other words, LZC adopts a uniform scalar quantizer. In contrast, multistage VQ is a non-uniform vector quantizer, which exploits the correlation between vector components and allocates quantization centroids in accordance with the probability distribution density. Specifically, more vector centroids are put in the higher density region so that the space partitioning is not uniform. A well designed non-uniform quantizer should reduce quantization errors more than a uniform quantizer, since it uses more information (e.g. distributions and characteristics of symbols) in the quantization process. On the other hand, by examining the quantization histograms of uniform and non-uniform quantizers to determine the entropy rate, the histogram of non-uniform quantizer is smoother than the uniform quantizer, which implies a higher coding cost. This can be explained by the fact that the Lloyd Max iteration attempts to adjust the locations of centroids and, roughly speaking, to make each quantized region contain about the same number of symbols. As a result, a uniform quantizer has a larger quantization error but a lower entropy while a non-uniform quantizer has a smaller quantization error but a higher entropy. According to the information theory, the overall rate distortion performance of a uniform quantizer is superior to a non-uniform quantizer in higher bit rates and the non-uniform quantizer can be superior in the medium and low bit rate ranges. It justifies the use of the multistage VQ in the lower bit rate range.

Another merit of our proposed algorithm is the low complexity of multistage VQ. According to our experiments, we can use 96 codewords for 6 stages to achieve a coding rate up to 0.1 bpp with a single four dimensional codebook. Similarly, 112 codewords for 7 stages are able to achieve 0.25 bpp, and 256 codewords for 16 stages are enough for most high rate codings. Compared with embedded lattice VQ [9], where more than 4000 codewords of dimension 16 were used, the computational and storage complexity of the multistage VQ is much lower.

6 Conclusions

A new algorithm for image coding based on the wavelet transform and multistage VQ has been proposed in this work. The algorithm performs competitively with all known state-of-the-art compression techniques. Besides a good rate-distortion performance, the new method produces a fully embedded code which is convenient for rate control. The building components of the algorithm include: 7-9 tap biorthogonal wavelet transform, vector norm significance measure, multistage VQ and JBIG context arithmetic coder. The low complexity of VQ is achieved by the multistage structure. Only one codebook with a size less than 256 code vectors is sufficient to encode vectors from any subband or quantization layer. Multistage VQ is particularly efficient in the coding of texture dominant images, which are often more difficult to encode by traditional coding methods.

References

[1] ISO/IEC-JTC1/SC2, "Progressive bi-level image compression," ISO Standard CD 11544, ISO, Sep. 1991.

[2] N. Chaddha, P. A. Chou, and T. H. Y. Meng, "Scalable compression based on tree structured vector quantization of perceptually weighted block, lapped, and wavelet transforms", *Proc. IEEE Int. Conf. on Image Proc.*, Washington, DC, USA; Oct. 1995.

[3] T. M. Cover, and J. A. Thomas, *Elements of information theory*, pp. 86–90, John Wiley & Sons, Inc., New York.

[4] A. Gersho, and R. M. Gray, *Vector quantization and signal compression*, pp. 345–347, Kluwer Academic Publishers, 1992.

[5] H. Jafarkhani, and N. Farvardin, "Scalable wavelet image coding scheme using multi-stage pruned tree-structured vector quantization", *Proc. IEEE Int. Conf. on Image Proc.*, Washington, DC, USA; Oct. 1995.

[6] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design", *IEEE Trans. on Comm.*, Vol. COM-28, pp. 84–95, Aug. 1980.

[7] W. B. Pennebaker, *JPEG Still Image Data Compression Standard*, Van Nostrand Reinhold, New York, 1993.

[8] J. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients", *IEEE Trans. on Signal Processing*, No. 12, Dec. 1993.

[9] E. A. B. da Silva, D. G. Sampson, M. Ghanbari, "A successive approximation vector quantizer for wavelet transform image coding", *IEEE Trans. on Image Processing*, Vol. 5, No. 2, Feb. 1996.

[10] D. Taubman and A. Zakhor, "Multirate 3-D subband coding of video", *IEEE Trans. on Image Processing*, No. 5, pp. 572–588, Sep. 1994.

[11] J. D. Villasenor, B. Belzer, and J. Liao, "Wavelet filter evaluation for image compression", *IEEE Trans. on Image Processing*, Vol. 4, Issue 8, Aug. 1995.