Information Processing & Management Vol. 28, No. 6, pp. 707-721, 1992
Printed in Great Britain.

0306-4573/92 $5.00 + .00

QUAD TREE STRUCTURES FOR IMAGE COMPRESSION APPLICATIONS*

TASSOS MARKAS†
Department of Electrical Engineering, Duke University, Durham, NC 27706

and

JOHN REIF
Department of Computer Science, Duke University, Durham, NC 27706

Abstract—Traditionally, lossy compression schemes have focused on compressing data at fixed bit rates to either communicate information over limited bandwidth communication channels, or to store information in a fixed-size storage media. In this paper we describe a class of lossy algorithms that is capable of compressing image data over a wide range of rates so that quick browsing of large amounts of information as well as detailed examination of high resolution areas can be achieved by the same compression system. To accomplish this we use a quad tree structure to decompose an image into variable size blocks which are subsequently quantized using a Tree-Structured Vector Quantizer (TSVQ). The developed algorithms utilize variable-size image blocks encoded within quad tree data structures to efficiently encode image areas with different information content. These algorithms are also capable of compressing images so that the loss of information complies with user defined distortion requirements. In this paper we describe the use of quad tree structures in image compression type applications, and we analyze their advantages over the classic vector quantization schemes. Finally, we describe their progressive compression capabilities and we demonstrate that they achieve higher compression/distortion performance compared to the classic TSVQ algorithm.

1. INTRODUCTION

A typical model of a data compression system capable of achieving high compression ratios consists of the transform coder, the quantizer, and an optional lossless compressor (Fig. 1). Although such systems were originally designed to accomplish lossy compression, it has been demonstrated that they are also capable of implementing lossless compression with higher performance than existing stand-alone lossless techniques.

The transform coder block is responsible for transforming the incoming image to a different domain of representation so that a large fraction of its information is packed in relatively few coefficients. Some examples of block transform methods include the Discrete Cosine Transform (DCT) [1], the Predictive Coding [2,3], the wavelets [4], and so on. Some coefficients can be discarded from this transformed image if they do not convey critical information with respect to the observing instrument. For example, the low-amplitude, high-frequency components of a small, DCT transformed image block can be either discarded, or encoded with a coarse quantizer since they do not convey important information with respect to the human visual system. The quantizer is a mapping of the transformed image to a finite number of reconstruction levels. Clearly, the quantization processes introduces some loss of information, and is primarily responsible for the high compression ratios achieved in lossy algorithms. Careful selection of an appropriate quantization function can minimize the error effects so that they are not visible to the human eye. Finally, the quantization error can be compressed using a lossless algorithm such as Huffman coding [5],

*This work has been supported by the Center of Excellence in Space Data and Information Sciences under NASA/CESDIS NAS USRA subcontract 550-63 of prime contract NAS5-30428. Additional support has been provided by DARPA/ISTO Contracts N00014-88-K-0458, N00014-91-C-0114, N00014-91-J-1985, and NSF Grant 88-00282/2. A preliminary version of this paper was presented in the 1991 Data Compression Conference.
†Also with Center for Systems Engineering, Research Triangle Institute, RTP, NC 27709.

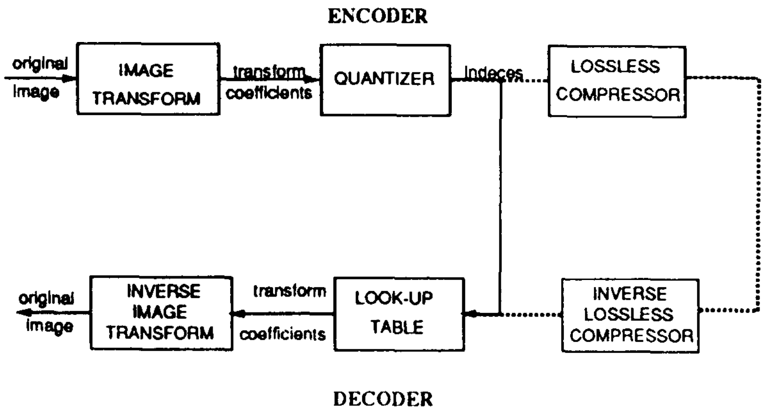

Fig. 1. A typical data compression system.

arithmetic coding [6], or Lempel-Ziv type methods [7,8] to attain a lossless technique that exceeds the performance of stand-alone lossless methods.

1.1 *Vector quantization*

Vector quantization is a lossy block-coding technique that is used extensively to compress data at low bit rates (high compression ratios). Vector quantizers compress image data by replacing an image block, represented by a discrete vector, with the index of the best-match entry found in a reconstruction codebook based on a given distortion measure [9]. A reconstruction codebook consisting of vectors of size N divides the entire search space into a fixed number of N-dimensional regions. The encoding process is responsible for identifying the corresponding region of each input block and for replacing the block with the index of the codebook vector that represents that region (Fig. 2). This is accomplished by measuring the distortion between the input block and each codebook vector, and by identifying the vector with the minimum distortion. The decoding process is a simple look-up operation where each index of the compressed data string is used to rebuild the original image. Both the encoder and the decoder maintain the same vocabulary that has been constructed off-line using training algorithms on a set of data that is similar to the one that will be compressed. Some of the training algorithms that can be used to construct the codebook include the widely used k-means algorithm [10], and neural network type algorithms.

A straightforward implementation of the vector quantization algorithm is to perform a linear search over all vocabulary entries. This full search method is a computationally expensive task since linear time is required to compare each input vector against all codebook entries using a given distortion measure. A more efficient approach that reduces the computational requirements from linear to logarithmic time is the tree-structured vector quan-

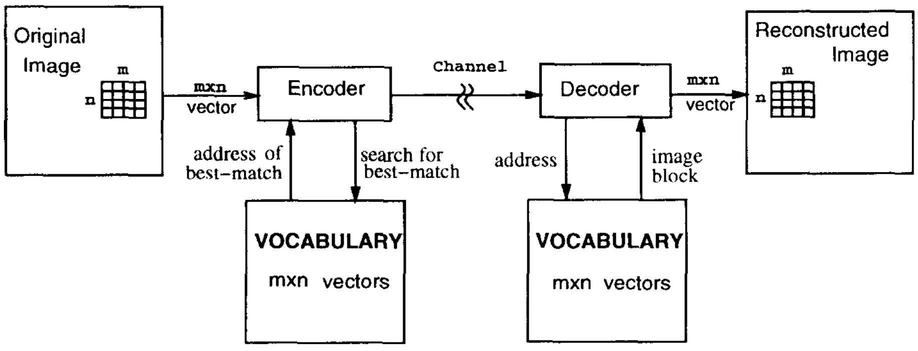

Fig. 2. A $N = m \times n$ vector quantizer.

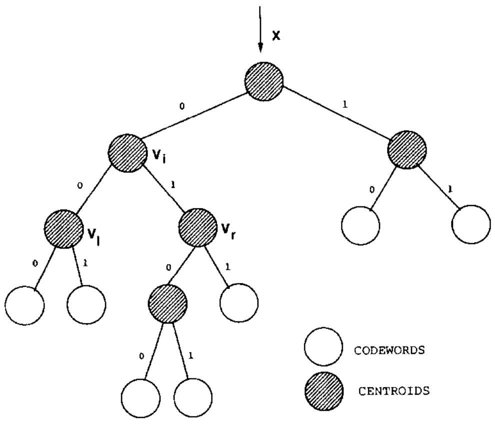

Fig. 3. A tree-structured vector quantizer.

tization algorithm [11]. This is accomplished by storing the reconstruction vocabulary in a binary tree form, so that the leaf nodes of the tree represent the codebook entries, and the intermediate nodes constitute the centroids of their children nodes (Fig. 3).

$$\mathbf{V}_i = \frac{\mathbf{V}_l + \mathbf{V}_r}{2} \tag{1}$$

Each codebook entry is represented by a unique index which is the binary path from the root node to the corresponding leaf node. Each node of the tree executes the following calculations to identify the children node that results in minimum distortion:

$$\text{if } D(\mathbf{X},\mathbf{V}_l) < D(\mathbf{X},\mathbf{V}_r) \text{ left node} \tag{2a}$$

$$\text{if } D(\mathbf{X},\mathbf{V}_l) \geq D(\mathbf{X},\mathbf{V}_r) \text{ right node} \tag{2b}$$

where D is the distortion measure.

Each centroid in the TSVQ algorithm defines a hyperplane that divides the search space in two regions which are represented by the children nodes. An L entry vocabulary divides the entire search space in L Vonoroi regions that are uniquely defined by the vectors of the leaf nodes. Figure 4 shows the Vonoroi regions of a uniform TSVQ in the two-dimensional search space, utilizing an eight-entry vocabulary.

1.2 *Progressive transmission*

The majority of the modern data compression techniques offer the capability of transmitting a set of data in a progressive fashion. Initially, the image is transmitted at low bit rates (high compression rates) where the reconstructed information is highly distorted. Additional information that restores the distorted area is only transmitted upon request. The tree-structure of the TSVQ algorithm is very suitable for progressive transmission because of the hierarchical structure of the vocabulary. Since the centroid of each tree-node represents a unique region of the search space, the k most significant bits of the index of the best-match vector can be used to obtain the kth approximation of the original data block. This property allows the TSVQ to trade off between compression rates and distortion. The potential compression rate of a TSVQ system is a function of the vector and the vocabulary sizes. Assuming a b-bit vocabulary (2^b-entries) and N-size input blocks, the lowest possible distortion is achieved at b/N bits per pixel (bpp), while the lowest rate is $1/N$ bpp with the original image degraded in a binary form. In TSVQ the possible compression rates can vary $(b - 1)/N$ bpp, whereas the quality of the reconstructed image is upper bounded by the distortion achieved at the b/N bpp rate.

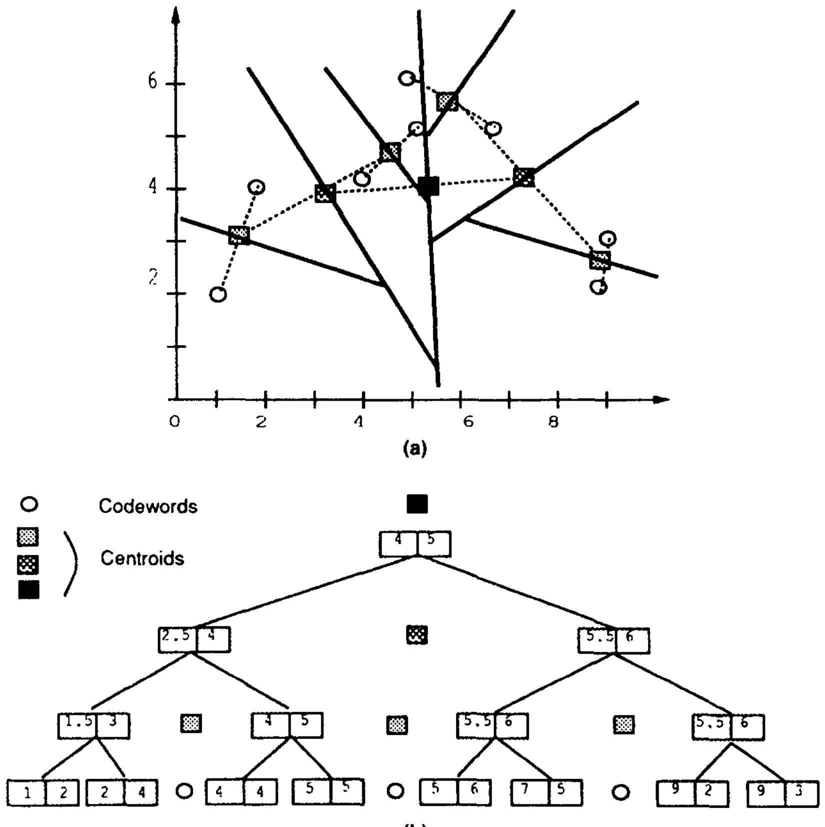

Fig. 4. Vonori regions of the TSVQ algorithm: (a) Vonori regions with eight codewords in a two-dimensional space; (b) binary-tree representation of the eight-word vocabulary.

1.3 *Distortion measures*

A number of distortion measures have been proposed for evaluating the loss of information in images. Some of these measures take into consideration the physical properties of the human eye, while others are more targeted towards quantitative analysis of image data. One of the most widely used measures is the Mean Squared Error (MSE), which unfortunately does not take into account the characteristics of the human eye. The MSE is used widely because of its ability to formulate easily the theoretical performance of the compression algorithms, so that theoretical results can be compared against the experimental ones. The measure that will be used throughout this paper to quantify the distortion of the reconstructed images is the Peak Signal-to-Noise Ratio (PSNR). The signal is the peak energy of the image (255 for Gray scale images), and noise is the MSE between the original and the reconstructed images. Clearly, this measure falls in the category of the MSE distortions, and for this reason is not very representative of the loss of information that the eye perceives.

$$PSNR = 10 * \log_{10}\left(\frac{(2^p - 1)^2}{\sum_{i=1}^{XY} (x_i - x_i')^2}\right) \tag{3}$$

where XY is the image size, p is the number of bits per pixel, and x_i, x_i' are the pixel values of the original and the reconstructed images, respectively.

The performance of the developed algorithms was evaluated using images obtained from the USC (University of Southern California) database, as well as Thermatic Mapper data obtained from the LANDSAT satellite. The required vocabularies were created using the tree-structured version of the *k*-means algorithm.

2. MOTIVATION

Vector quantization is a lossy technique whose distortion is lower bounded by the size of the existing vocabulary. For certain applications, where only minimal loss of information can be tolerated, the quality of the decoded image may not be adequate. Such applications include target recognition, object measurement, and quantitative analysis or observation of multi-spectral satellite images. To eliminate the lower distortion bound posed by the TSVQ we will present a class of distortion-controlled methods that are capable of compressing data in a wide range of operating conditions (distortion, compression ratios). The compression rates of the distortion-controlled vector quantizers (DCVQ) vary from low rates, suitable for quick browsing of large amounts of image data, to high rates for accurately reconstructing images with high information content. The first technique that will be presented is the Multi-Resolution Vector Quantization (MRVQ), an algorithm that compresses data by dividing an image block into a number of variable-size subblocks and encodes them using a quad tree representation. Two similar approaches have been proposed in [12,13]. Our approach uses a variable-size and variable-depth quad tree representation to utilize large size blocks without reducing the distortion/compression performance at high bit rates. The MRVQ method requires multiple vocabularies consisting of different size vectors that will be used for compressing image blocks at different levels of the quad tree. The training of these vocabularies is a computationally expensive task, and for that reason we will present methods for creating these product codebooks from a single vocabulary on-the-fly. The second class of DCVQ algorithms that will be presented is based on recursive quantization of the error vector, which is defined as the difference between the current approximation and the original vector. We will refer to these algorithms as Error Coding Vector Quantizers (ECVQ), and Error Coding Multi-Resolution (ECMR) methods. The latter combines the advantages of both MRVQ, and ECMQ schemes to maximize the compression/distortion performance.

It has also been shown that pruning methods [14,15] experience better performance compared to the basic TSVQ scheme. These schemes can be incorporated in the design of the DCVQ methods to take full advantage of the state-of-the-art data compression algorithms.

3. QUAD-TREE IMAGE COMPRESSION

The classic TSVQ algorithm uses a fixed number of bits to encode all image blocks independently of their information content. However, the amount of information contained in typical images is not uniformly distributed among the different regions. This introduces some inefficiency in the way the TSVQ algorithm allocates the coding bits in each image block. A solution to this problem is to use a hierarchical structure where the number of bits required to encode an image region is proportional to the information content of the region. The quad tree data structure [16] is one of the most suitable structures for this purpose. Quad trees are very useful structures in image processing type applications since each block can be decomposed in four equal quadrants, thus preserving the spatial characteristics of adjacent blocks.

In the next section we will present a base image-compression algorithm that utilizes quad trees, and in the subsequent sections we will present some improvements of this base algorithm, and we will examine their compression/distortion performance.

4. MULTI-RESOLUTION VECTOR QUANTIZATION

The MRVQ algorithm utilizes the quad tree structure to decompose the image in variable size blocks, so that large size blocks can be used to compress image areas with low information content at low bit rates, and small size blocks can be used to compress areas with high information content at high bit rates. A parameter that specifies the acceptable distortion of the entire image is given to the system in the form of average error per pixel. This parameter is responsible for determining the size of image blocks during the quad tree decomposition. Initially, the image is divided into large size blocks ($N = n \times m$) which are quantized by applying TSVQ in a vocabulary (V_0^N) that contains $n \times m$-size vectors. The best-match vector obtained from this operation represents the current approximation of the original vector. If this approximation satisfies the specified distortion criteria, the index of the codebook vector replaces the N-size image block. Otherwise, the image block is divided into four equal size subblocks (quadrants) which are quantized separately using vectors from a $(n/2) \times (m/2)$-vector vocabulary ($V_1^{N/4}$). The same procedure is repeated recursively until the distortion requirements are satisfied in all subblocks, or until a leaf node is encountered during the recursive process. The highest possible resolution that can be obtained in the MRVQ algorithm is a single pixel (image area is left uncompressed). A V_{d-1}^1 (d is the depth of the quad tree) vocabulary containing all the possible intensity values is assumed in this case. The advantage of this recursive decomposition is that by reducing the vector size in each quad tree level, the search space is reduced by several orders of magnitude. A transition between two consecutive tree levels reduces the input space from $(2^p)^N$ to $(2^p)^{N/4}$ possible vector combinations (p is the number of bits per pixel in the uncompressed image). This results in a more efficient reconstruction of the original image block.

The compressed file consists of the indices of the best-match vectors which are embedded in the quad tree structure. The encoding of the quad tree representation itself introduces an additional overhead that needs to be minimized in order to maintain high compression ratios. Each tree node requires four additional bits to indicate its active children nodes (i.e., subblocks where distortion criteria are not met). A logic zero denotes that the best match vector satisfies the local distortion requirements of the corresponding subblock, whereas a logic one denotes that the subblock will be quantized in a lower level utilizing smaller size vectors. The quad tree product VQ method, presented in [12,13] requires a fixed-size, variable-depth quad tree which is used throughout the compressing process. To reduce the coding overhead of the quad tree in image blocks with high information content, we have introduced a variable size structure where a tree level is eliminated if no best-match vector has been used to approximate a block in the current or any of the upper levels [17]. This scheme increases the compression performance by eliminating the upper levels of the tree structure in blocks with high information content. Another advantage of this scheme is that large size blocks can be utilized to compress blocks at low rates without affecting the compression performance at high bit rates. With this feature, the MRVQ algorithm can compress image areas with minimal information (i.e., black background of a picture, sea or clouds in an aerial photo) at very low bit rates. Figure 5 shows the quad tree decomposition of an image block and the compressed code that is associated with this particular block. It should be also noted that a minimum overhead (3 bits per block) is required to encode the maximum size of the quad tree structure. Entropy coders can also be used to encode the tree structure. In this case, the entire tree should remain intact since the redundancy in the upper levels can be removed by the coding algorithm. It should be also noted that in this case the maximum size of the tree remains constant. For this reason it is not included in the compressed file, thus resulting in an additional savings of three bits per block.

The formal description of the MRVQ algorithm is the following: Assuming that X^N is an input vector of size N, and X^K is a subvector of X^N at level i with size $K = N/4^i$, then the corresponding vector of node j at level $i(X^K)$ is generated by applying a mask function M in the vector of its parent node (level $(i - 1)$, node $(j/4)$).

$$X_{ij}^K = X_{(i-1)(j/4)}^{4K} M_{j \, mod \, 4} \tag{4}$$

M_i is a function that selects one quadrant of a given vector (0: upper-left, 1: upper-right, 2: lower-left, and 3: lower-right). There is also an L-entry tree structured vocabulary (V_{iL}^K)

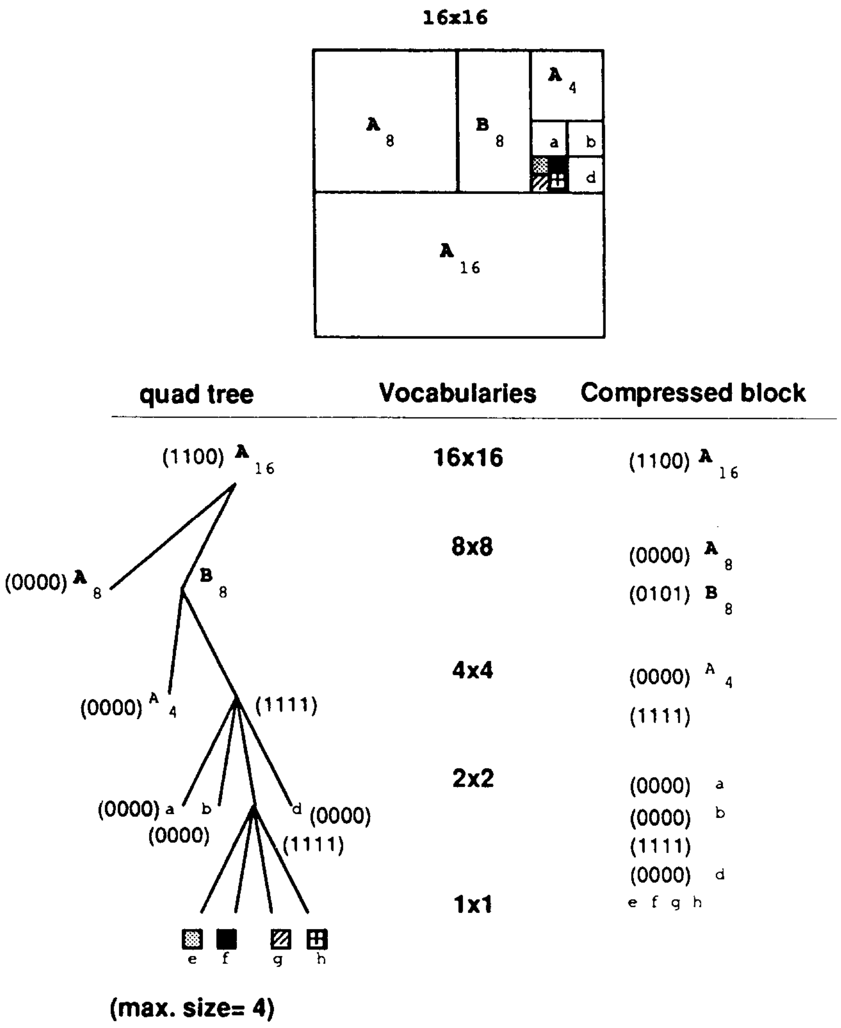

Fig. 5. Quad-tree representation of the MRVQ algorithm.

with codebook vectors of size $K(V_{il}^K, l = 0, \ldots L-1)$ for each tree level (i). It should be noted that L can be a function of i so that different size vocabularies can be assigned in different levels.

If T_{ij} is the jth tree node at the ith level, then the approximation of the input vector in the T_{ij} node is given by the recursive expression:

$$\hat{X}_{ij} = \overline{b_{ij}}\,\mathrm{BM}(T_{ij}) + b_{ij}\sum_{k=0}^{3}\left[\overline{r_{ijk}}\,\mathrm{BM}(T_{ij})M_k + r_{ijk}\hat{X}_{(i+1)(4j+k)}\right] \tag{5}$$

where BM is a function that returns the best-match entry of current approximation applied in the vocabulary V_i^k using the MSE distortion measure:

$$\mathrm{BM}(T_{ij}) = \min_{l=0,\ldots L}(\|X_{ij}^K, V_{il}^K\|) \tag{6}$$

$$\|X^K, Y^K\| = \sum_{k=0}^{K-1}(X^k - Y^k)^2 \tag{7}$$

b is a binary function that denotes if the best-match vector satisfies the distortion requirements, and v_q is a similar function applied in the qth quadrant. If we assume that the distortion requirement is $\pm E$ intensity values per pixel, the formal definition of the two functions is

$$b_{ij} = \begin{cases} 0 & \|X^K, \mathrm{BM}(T_{ij})\| \leq KE^2, \text{ or } i > D \\ 1 & \text{otherwise} \end{cases} \tag{8}$$

where D is the maximum size of the quad tree, and

$$r_{ijq} = \begin{cases} 0 & \|Y^P, Z^P\| \leq KE^2/4,\ Y^P = M_q X^K,\ Z^P = \mathrm{BM}(T_{ij}M_q) \\ 1 & \text{otherwise} \end{cases} \tag{9}$$

These two binary functions form a unique quad tree that defines the coding of an image block. The function b_{ij} determines the type of the ij node. If the node is not leaf, then the r function determines the children nodes that have a connection with the parent node. the $\hat{X}_{00}$ quantity represents the final approximation of the input vector, and $\|X^N, \hat{X}_{00}\|$ gives the distortion of the entire block.

The number of bits that is required to encode an image block can be calculated as follows: If B_{ij} is the number of bits that is required to encode the corresponding subtree, then the size of a compressed image block is given by B_{00}. The first term of the B_{ij} quantity is the coding overhead of the best-match vector, the second term is the overhead associated with the coding of the active children nodes, and the last term represents the encoding overhead of the subtrees that originate from node ij. It should be noted that this expression does not account for possibly eliminated quad tree levels.

$$\begin{aligned} B_{ij} = {} & \overline{b_{ij}} B(V_{ij}) H(\text{“0000”}) + b_{ij} \mathrm{NAND}_{k=0,\ldots,3}(r_{ijk}) B(V_{ij}) H(\text{“}r_{ij0} r_{ij1} r_{ij2} r_{ij3}\text{”}) \\ & + b_{ij} \sum_{k=0}^{3} r_{ijk} B_{(i+1)(4j+k)} \end{aligned} \tag{10}$$

where $B(V_{ij})$ is the number of bits required to represent the index of the best-match vector found in the vocabulary V_{ij} for node ij, $H(s)$ is the number of bits required to encode the string s using an entropy coding algorithm, and NAND(s) is the bit-wise NAND function of the binary string s. It should be noted that all the expressions above do not account for possibly eliminated quad tree levels (variable size MRVQ algorithm).

4.1 *Performance of the MRVQ algorithm*

The effect of the maximum size image block in terms of compression/distortion performance is depicted in Fig. 6. This figure shows the ability of the MRVQ algorithm to achieve low compression rates using large size blocks without penalizing the compression performance at high rates. The same figure shows the trade-off between compression/distortion performance and dynamic compression range. The restricted quad trees, in terms of size and resolution that can be achieved, offer a slightly better performance over their dynamic range.

The described top-down MRVQ approach is a computationally inefficient method for compressing high-detail areas since the algorithm spends a significant amount of time calculating distortions and identifying best-matches for large size blocks that will not be used in the final bit stream. A more efficient approach is to use a down-up approach, where quantization is initially applied at the lowest possible levels of the quad tree using small size vectors. At each level, the algorithm examines the distortion of the best-match vector to verify if it complies with the given criteria. If this is true, the current best-match vector is used to encode the block, and the current node becomes leaf node of the tree. If the distortion criteria are not met, the best-match vectors of the immediately lower level will be

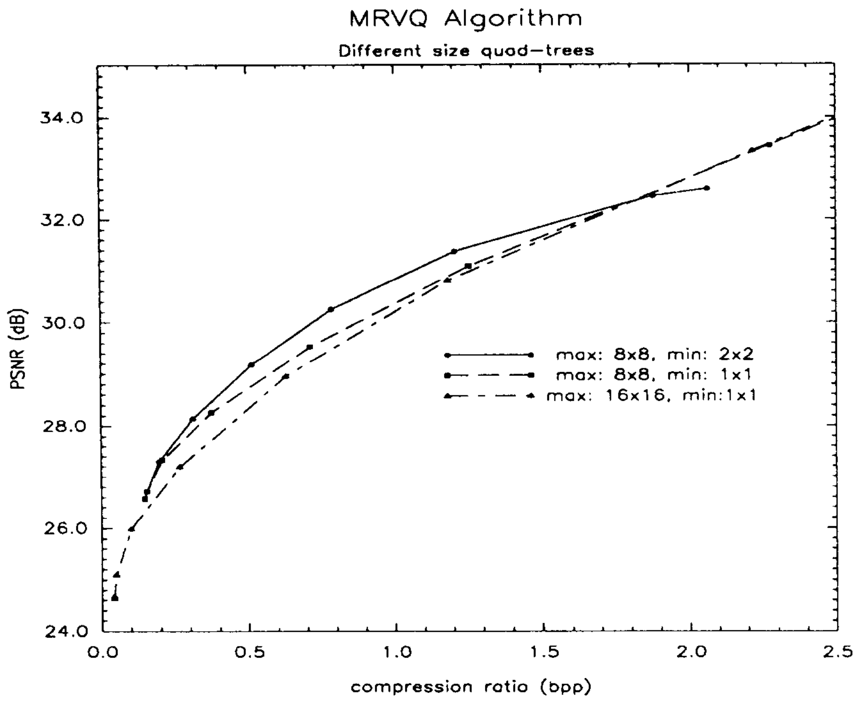

Fig. 6. Compression performance of quad tree with different size.

used to encode the quadrants of the current node. Although the down-up approach is computationally more efficient, it can not be used to implement progressive transmission.

The performance of the MRVQ method can be further improved if additional efficiency measures are applied during the decomposition process. One method is to trade distortion for compression by eliminating the subtrees of the quad tree structure that fail to significantly reduce the block distortion. A similar pruning scheme has been proposed in [15] to perform a similar operation for the TSVQ algorithms. The pruning scheme of the MRVQ algorithm is accomplished as follows: At each node of the quad tree we examine the distortion improvement and compression rate of the corresponding subtree. If the ratio of these two quantities is below a predefined threshold, the entire subtree is deleted, the current node becomes leaf node, and the best-match vector of the current node is used to approximate the examined subblock.

$$\frac{d(\mathrm{MSE})}{d(\mathrm{rate})} < \text{threshold} \tag{11}$$

The pruning algorithm improves the compression rates of medium and high bit rates but it also reduces the dynamic range of the compression curve. The performance of the pruning scheme for various threshold values is shown in Fig. 7.

4.2 *Generation of product vocabularies*

The MRVQ algorithm uses different vocabularies for each level of the quad tree that should be constructed using different sets of training data. As was mentioned earlier, this training phase is a computationally expensive process. In addition, hardware systems with limited storage resources may pose a severe restriction on the size of these vocabularies. An alternative approach is to generate these vocabularies on-the-fly using only a single vocabulary that will be referred as the "core" vocabulary.

A number of linear interpolation methods have been evaluated for generating product codebooks from the "core" vocabulary. The first method is the replication scheme [18], where a $n \times m$ size block is expanded to a $(2n) \times (2m)$ block by replicating adjacent pix-

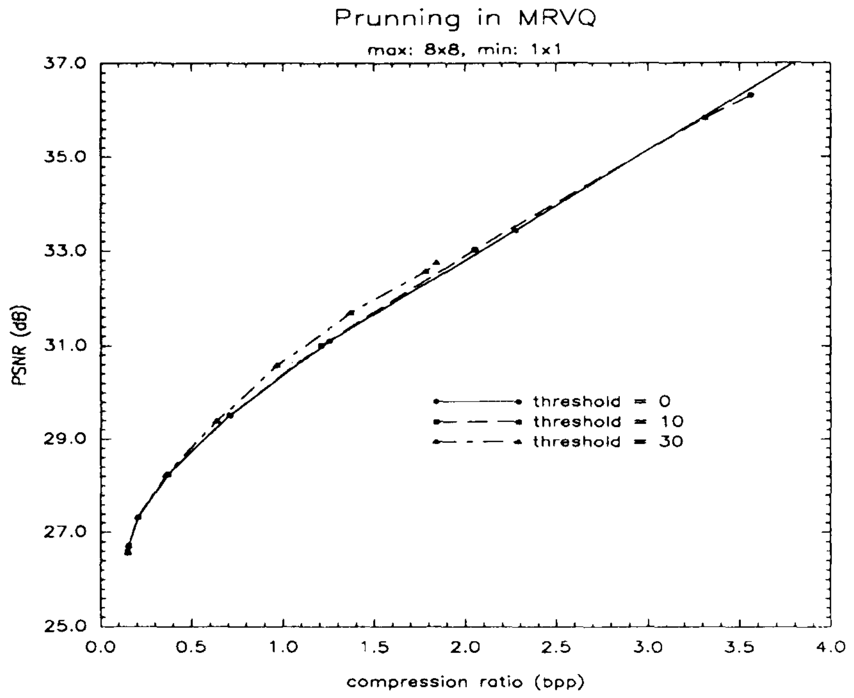

Fig. 7. Compression performance of pruning schemes.

els. Another method that offers smoother transitions among the neighboring pixels is the bilinear interpolation [19]. The disadvantage of this scheme is the inaccurate generation of pixels in the right, and lower sides of the image block. This effect causes severe degradation in the right and lower edges of the image block when generating large size vectors. To overcome this problem we modified the bilinear transformation so that the intensity of the left and lower pixels are computed using a postprocessing averaging scheme. Quantization in this case is only applied on the rest of the pixels. This interpolation scheme reduces the vector quantization effect (sharp discontinuities among adjacent blocks) when large vectors are used to compress an image block.

Figure 8 shows the performance of the evaluated interpolation methods. The tree-structured version of the *k*-means algorithm was used to create the 2 × 2-size vocabulary. The remaining vocabularies were generated by the 2 × 2 "core" vocabulary using the described methods. Figure 8 shows that the replication method achieves almost identical performance with the separate vocabulary MRVQ case, and it is also the simplest one to compute.

5. ERROR CODING VECTOR QUANTIZER (ECVQ)

This scheme is an iterative procedure that quantizes the error vectors, generated by subtracting the current approximation from the original vector, using a TSVQ quantizer to obtain the new approximation. The basic idea behind this scheme is to reduce the vector variance at each iteration step so that blocks can be efficiently approximated using a minimum number of iterations. At each step, the current approximation is compared against the original vector to determine if the distortion requirements are met. The recursive formula that describes the ECVQ method is

$$\mathbf{x}'_i = \mathbf{x}'_{i-1} + \mathrm{BM}(\mathbf{x} - \mathbf{x}'_{i-1}), \quad \text{and} \quad \mathbf{x}'_0 = \mathrm{BM}(\mathbf{x}) \tag{12}$$

where $\mathbf{x}$ is the input vector, $\mathbf{x}'_i$ is the ith approximation of the input vector, and BM is the function that identifies the best-match from the appropriate vocabulary.

To capture the different variance of the error vectors in each iteration step, the vocab-

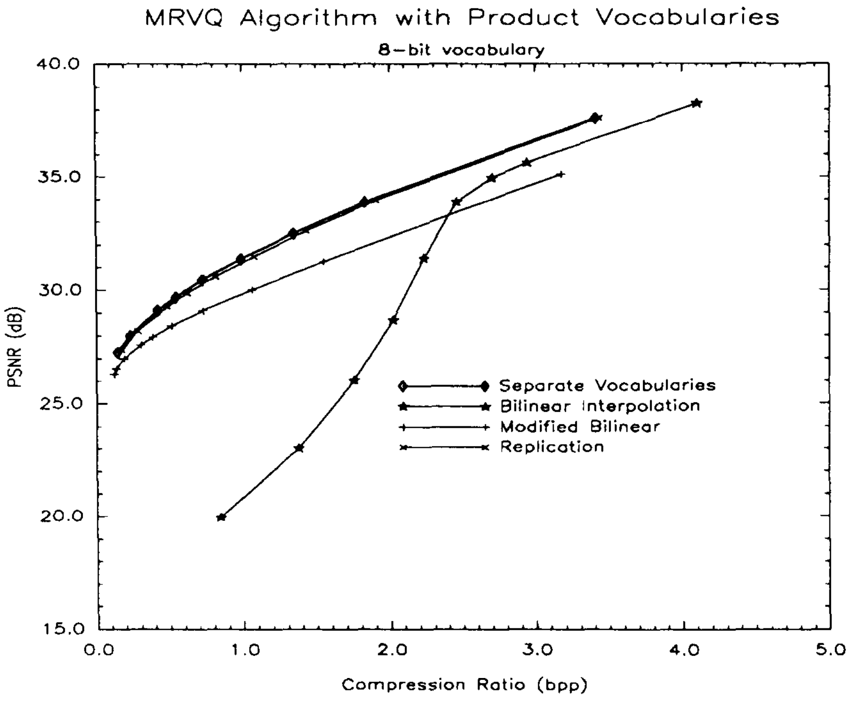

Fig. 8. Generation of product vocabularies.

ularies are trained using different sets of data. The initial vocabulary is created using vectors obtained from the intensity domain of the training set of images, while the remaining ones are created using different size vectors obtained from error images. The codebook that is used during the ith iteration is trained using a set of $n \times m$-size vectors taken from error images that were generated by compressing the original set of images with a $(n/i) \times (m/i)$ TSVQ compressor.

The performance of the ECVQ scheme is inferior to the one obtained in the MRVQ case, especially at high bit rates. The primary reasons for this behavior is that the vector size remains constant in all iteration steps. The fact that the search-space in a vector quantization algorithm grows exponentially with the vector size indicates that there is an extremely large number of possible combinations, even at the latest stages of the iterative procedure. Even if the variance has been significantly reduced in those stages, the large search space cannot be accurately covered in a few thousands of codebook vectors.

6. ERROR CODING MULTI-RESOLUTION ALGORITHM

A modified version of the previous algorithm is to use the MRVQ method so that the size of the input vector is also reduced during each iteration step. By reducing the vector variance and the vector size a fewer number of iterations is required to achieve performance comparable to the MRVQ case performance. The advantage of this approach is that the variance of the error vectors is much smaller than the variance of the intensity vectors. This implies that the number of lower levels required to efficiently represent the image block is significantly reduced. This results in a more accurate approximation of the lower levels of the quad tree. This algorithm maintains all the important features of the MRVQ method that were described in earlier sections. The error vocabularies, used during the ith level of the quad tree, was created using a set of K-size vectors ($K = N/2^i$) taken from error images that were generated by compressing the original set with a K-dimensional TSVQ compressor.

The ECVQ differs from the MRVQ case in that the corresponding input vector in each tree node is formed by subtracting the current best match approximation of the parent node from the original input vector and applying the M function to extract the corresponding quadrant:

$$X_{ij}^{K} = [X_{(i-1)(j/4)}^{4K} - \text{BM}(T_{(i-1)(j/4)})]M_{j \bmod 4} \tag{13}$$

The vocabularies in the ECMR case are generated using a set of training images obtained by subtracting the original image from a TSVQ compressed image. If X^K is a set of vectors that are used to train the V_i vocabularies, then $X^K = X^K - \text{BM}(V_i)$ is a set of vectors that can be used to train the ECMR vocabularies. The approximation of the input vector in node ij is given, as in the MRVQ case, by the following expression:

$$\hat{X}_{ij} = \overline{b_{ij}}\,\text{BM}(T_{ij}) + b_{ij}\sum_{k=0}^{3}[\overline{r_{ijk}}\,\text{BM}(T_{ij})M_k + r_{ijk}\hat{X}_{(i+1)(4j+k)}] \tag{14}$$

In the ECMR case the best-match vectors are included in all nodes even if a node has all of its children in active status. The number of bits required to encode an image block in the ECMR case is given by

$$B_{ij} = B(V_{ij})[\overline{b_{ij}}H(\text{“0000”}) + b_{ij}\text{NAND}_{k=0,\ldots,3}(r_{ijk})H(\text{“}r_{ij0}r_{ij1}r_{ij2}r_{ij3}\text{”}) + b_{ij}\sum_{k=0}^{3} r_{ijk}B_{(i+1)(4j+k)}] \tag{15}$$

7. HUFFMAN CODING OF THE QUAD-TREE STRUCTURE

The variable size quad tree approach that was described in the earlier sections exploited very good performance at high bit rates and average performance in moderate and low bit rates when compared with the classic TSVQ algorithm. In certain types of images with high information content throughout the entire region, the classic TSVQ showed better performance over a small range of the compression/distortion curve. Such behavior occurs in moderate compression rates where the entire image is primarily compressed using the same size vectors. In this case, the overhead associated with the coding of the quad tree structure exceeds the small savings achieved by utilizing large size blocks.

To increase the performance of both multi-resolution algorithms over their entire dynamic range it is critical to minimize the quad tree encoding overhead. Experimental results obtained from a large number of images have shown that certain codes used in the quad tree description are much more frequently used than other ones. This is true for a significant number of codes over a wide range of compression rates. For example the "0000" symbol (i.e., no active children nodes) that indicates that the best-match vector satisfies the distortion requirements is the most frequently occurring pattern independent of the compression rate. Another common symbol in low bit rates is the pattern "1111" (i.e., all children nodes active) which denotes that the entire vector will be encoded using high resolution vectors. A static Huffman coder applied in a highly nonuniform probability distribution can increase considerably the entropy of the source, thus resulting in high compression. By using this scheme and by eliminating the codes that were used to encode the maximum size of the tree, we were able to increase the compression performance of both multi-resolution algorithms over their entire dynamic range. Figure 9 shows that the Huffman scheme experiences better performance compared to variable size quad tree approach. It should also be noted that in the latter case the quad tree codes were compressed using a LZW type algorithm [8].

7.1 *Performance evaluation of the DCVQ schemes*

The required vocabularies were created using the tree-structured version of the k-means algorithm. Each vocabulary had 256 entries and was trained using a set of satellite images containing Thermatic Mapper data obtained from the LANDSAT satellite. A different im-

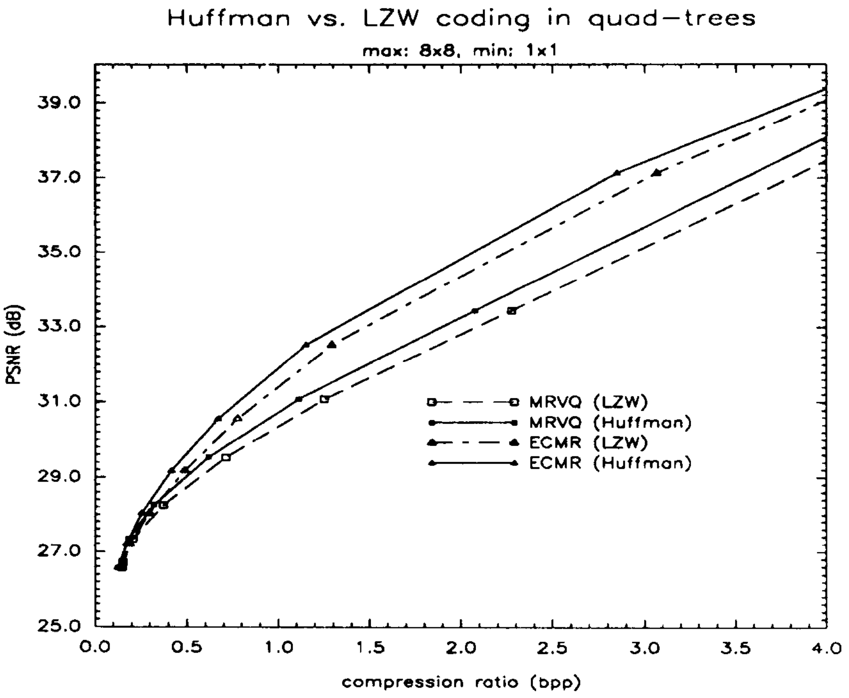

Fig. 9. Compression performance of quad-tree codes.

age that was not included in the training set was then used to evaluate the performance of the developed algorithms.

Figure 10 shows the MRVQ and the ECMR curves obtained at the lowest possible rates so that they meet the specified distortion requirements. This figure shows that both multi-

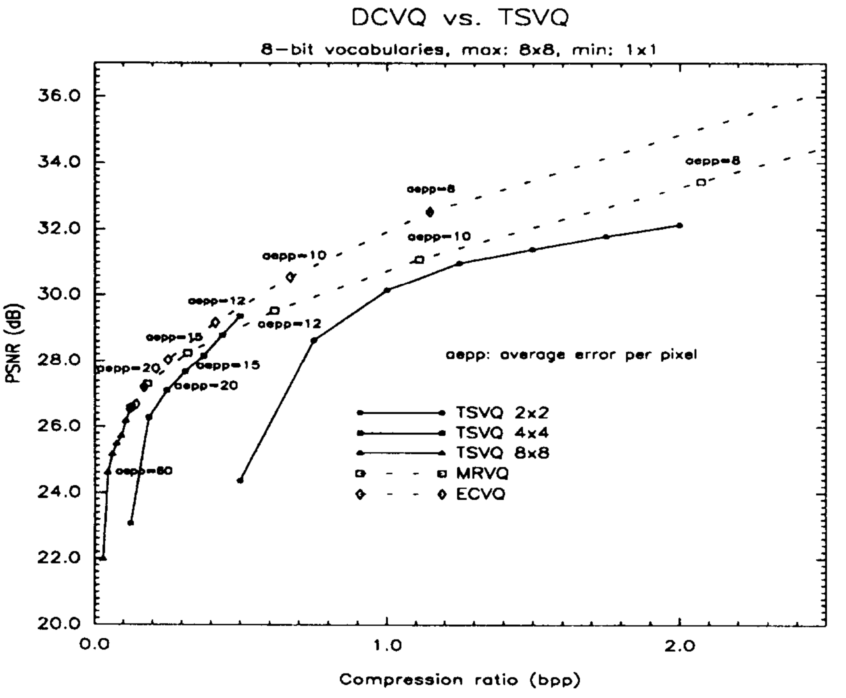

Fig. 10. Multi-resolution schemes versus TSVQ algorithms.

resolution methods exceed the performance of the TSVQ over the entire dynamic range. In addition, both algorithms offer better visual characteristics than the TSVQ scheme since they are capable of minimizing the error of the image edges which are important in the human visual system. The ECMR method outperforms the MRVQ method since it is capable of minimizing the number of small-size vectors that are used to reproduce high-detail areas. This stems from the fact that the number of nodes at a given level of the quad tree structure grows exponentially with the tree depth (4^n at the nth level). This means that compression of an image block using small size vectors is accomplished at low levels, thus introducing high overhead. The ECMR algorithm manages to minimize the lower levels of the tree by reducing the variance of the input source as well as the dimensionality of the search space at each iteration step. The DCVQ algorithms were also applied in images obtained from the USC database and they have experienced similar performance.

8. PROGRESSIVE TRANSMISSION IN DCVQ SCHEMES

Progressive transmission in the MRVQ and ECMR algorithms can be accomplished with three different approaches. The first approach is to transmit only the k most-significant bits of the vector indices and leave the quad tree code intact. The second approach is to partially transmit the quad tree (one level at a time) along with the associated indices of the best-matches. The most efficient approach is to use a combination of both methods. The distortion requirement parameter is used in this case to determine in what level the current approximation should terminate. Once a partial quad tree has been constructed that satisfies the distortion requirement, the indices of the best-match vectors are transmitted progressively, one level at a time, until the entire structure has been fully transmitted. If the current approximation does offer the necessary quality, the user can request additional resolution by specifying tighter distortion requirements. The encoder responds by selecting the leaf nodes of the current structure that do not meet the new distortion requirement. A new subtree is then created in each of the selected nodes, and the new information is transmitted to the user.

Expansion of the quad tree at extremely high resolutions (single pixel) can also yield lossless compression. However, the size of the compressed image in this case will be larger than the original one because of the quad tree coding overhead. Once a high level of accuracy has been obtained, it is preferable to use a lossless algorithm for compressing the error image. This will yield a more efficient lossless compression algorithm.

9. EFFECTS OF TRANSMISSION ERRORS

In communication type applications images are transmitted through noisy channels. In most cases, the communication protocols guarantee fault free transmission that can be achieved using error correcting codes, collision detection, and retransmission methods, and so on. However, there is a significant number of applications (i.e., satellite communications) where such schemes impose a very high overhead and therefore they are considered non-cost-efficient. Clearly, it can be concluded that the compression and the error correction problems cannot be treated separately when dealing with noisy communication channels. To implement a data compression system capable of transmitting and receiving image data it is essential to identify a suitable error correction code that will minimize the effect of an error given a specific data compression algorithm.

Traditionally, in error correcting codes equal protection capability is provided for all samples. In image compression applications equal protection does not necessarily minimize the overall error. A simple reversal error, a single bit is reversed due to channel noise, can cause complete loss of information in certain types of lossless algorithms such as Lempel-Ziv type algorithms and Huffman coders. In Pulse Code Modulation coders (PCM) a reversal error will affect only the intensity of a single pixel. In Differential Pulse Code Modulation (DPCM) systems the effect of an error will be more severe since the reconstructed pixel intensities are calculated recursively using past values. Finally, in transform coders and in vector quantization the error will be limited to the block where the fault occurred. Several optimizations have been suggested for the predictive and the transform methods. Such treatments vary from filter optimization methods to the design of hybrid systems that offer higher protection to samples with larger variances.

The compressed data in the multi-resolution algorithms consists of quad tree information encoded with some kind of compression scheme, and vocabulary indices. The former information consists of variable-length codes that are very susceptible in reversal errors. Such errors usually result in total loss of information. In contrast, the vocabulary indices obtained from uniform binary trees are less susceptible to reversal errors since the error does not propagate outside of the block boundaries. Also, the most significant bits of the vocabulary indices convey much more information than the lower ones. This indicates that hybrid error correction schemes that can offer high protection against errors in the quad tree codes, and minimal protection in the least significant bits of the vocabulary indices can offer low error probability without decreasing the compression ratio.

10. CONCLUSIONS

In this paper we have presented a class of distortion controlled vector quantizers that are capable of compressing images so that they comply with certain distortion requirements. This class of data compression algorithms is applicable in cases where significant loss of information cannot be tolerated. These algorithms can be used in a wide range of applications, from compressing images at low rates for quick browsing of large amounts of data, to high rates for detailed examination of images with high information content. In conclusion, it has been shown that the performance of these algorithms exceeds the performance of classic TSVQ methods at low rates based on the MSE distortion measure. The ECMR method outperforms the MRVQ method since it is capable of minimizing the lower levels of the quad tree by reducing the variance of the input source as well as the dimensionality of the search space at each iteration step.

REFERENCES

1. Ahmed, N.; Natarajan, T.; Rao, K.R. Discrete cosine transform. IEEE Trans. Comput., C-23:90–93; 1974.
2. Habibi, A. Comparison on n-th order DPCM encoder with linear transformations and block quantization techniques. IEEE Trans. Commun. Technol., COM-19:948–956; 1971.
3. Hang, H.; Woods, J.W. Predictive vector quantization of images. IEEE Trans. Acous. Speech and Sig. Proc., 33:1208–1219; 1985.
4. Rioul, O; Vetteri, M. Wavelets and signal processing. IEEE SP Magazine, 14–38; October, 1991.
5. Huffman, D.A. A method for the construction of minimum redundancy codes. Proceedings of IRE, 40:1098–1101; 1952.
6. Witten, I.H.; Neal, R.M.; Cleary, J.G. Arithmetic coding for data compression. Communications of ACM, 6:520–540; 1987.
7. Lempel, A.; Ziv, J. On the complexity of finite sequences. IEEE Trans. on Information Theory, 22:75–81; 1976.
8. Welch, T.A. A technique for high-performance data compression. Communications of ACM, 8–19; June, 1984.
9. Gray, R.M. Vector quantization. IEEE ASSP Magazine, 1:4–29; 1984.
10. Linde, Y.; Buzo, A.; Gray, R.M. An algorithm for vector quantizer design. IEEE Trans. on Communications, 28:84–95; 1980.
11. Gray, R.M.; Abut, H. Full search and tree searched vector quantization of speech waveforms. Proceedings of ICASSP, 593–596; May, 1982.
12. Vaisey, D.J.; Gersho, A. Variable blocksize image coding. Proceedings of ICASSP, 25(1):1–4; 1987.
13. Chiu, C.; Baker, R.L. Quad tree product vector quantization of images. Proceedings of SPIE, 1099, Advances in Image Compression and Automatic Target Recognition:142–153; March, 1989.
14. Breiman, L.; Friedman, J.H.; Olsen, R.A.; Stone, C.J. Classification and regression trees. In: The Wadsworth statistics probability series. Belmond, CA: Wadsworth; 1984.
15. Tzou, K. Progressive image transmission: A review and comparison of techniques. Optical Engineering, 26:581–589: 1987.
16. Klinger, A.; Dyer, C.R. Experiments on picture representation using regular decomposition. Computer Graphics and Image Processing, 5:68–105; 1976.
17. Markas, T; Reif, J. Image compression methods with distortion controlled capabilities. Proceedings of Data Compression Conference; 1991:93–102.
18. Jain, A.K. Fundamentals of digital image processing. Englewood Cliffs, NJ: Prentice-Hall, 1989: p. 253.
19. Lim, J.S. Two-dimensional signal and image processing. Englewood Cliffs, NJ: Prentice-Hall, 1990: pp. 495–496.