MULTIRESOLUTION SEGMENTATION-BASED IMAGE CODING WITH HIERARCHICAL DATA STRUCTURES

Hamid R. Rabiee[1], R. L. Kashyap[1] and S. R. Safavian[2]

[1]Department of Electrical Engineering
Purdue University
West Lafayette, IN 47907

[2]LCC Inc., Cellular Institute
Arlington, VA 22201

ABSTRACT

This paper presents two multiresolution segmentation-based algorithms for low bit rate image compression using hierarchical data structures. The segmentation is achieved with quadtree and BSP-tree hierarchical data structures and the encoding is performed by using the projection pursuit (matching pursuit) with a finite dictionary of spline functions with various degrees of smoothness. Comparison with JPEG at rates below 0.5 bit/pixel shows superior performance both in terms of Peak Signal-to-Noise Ratio (PSNR) and subjective image quality.

1. INTRODUCTION

Interactive multimedia communications have emerged as a major area of research in the last few years. It has opened a wide range of potential applications by combining a variety of information sources such as audio, images, graphics and video. These systems are required to deliver an explosive amount of data over the band-limited channels. Normally, most of the bandwidth is occupied by the still images and video clips. Therefore, efficient image and video compression algorithms which are capable of operating at low bit rates are needed in these systems.

In recent years, the demand for inter-operability between different systems in interconnected networks has initiated the establishment of a few international committees to devise operative image and video compression standards for different applications [1]. These standards include JPEG (*baseline*) for compression of still images, H.261 or *p*x64 for video teleconferencing over ISDN networks, and MPEG for video storage and broadcasting. All of these standards use the Block-Based Discrete Cosine Transform (BB-DCT) for still image and intra-frame coding of digital pictures. This transform coding technique which operates on square blocks of 8x8 or 16x16 elements, have gained popularity because of its relative good performance and ease of implementation. The disadvantage of dividing up an image into square blocks in a BB-DCT algorithm is that when the transmission rate is forced to be lower and lower the block structure appears in the reconstructed picture and its regularity makes the decoded image annoying to the observer. This artifact is known as the *blocking effect*. For example, JPEG suffers from blocking effects at rates below 0.25 bit/pixel as depicted in Fig. 4.1(d). Most of the other block-based image coding techniques such as vector Quantization (VQ) also suffer from this artifact [2]. The current research in very low bit rate lossy image compression includes fractal coding, subband/wavelet coding, and segmentation based coding.

Fractal coding techniques were developed based on the theory of *iterated contractive transformations* and *collage theorem* to exploit the existing self similarities of natural images [3]. Although, block-based fractal coding algorithms have not shown a considerable advantage over the BB-DCT at very low bit rates, the fractal based transform coding techniques have shown to perform better at lower bit rates [4].

Subband coding techniques have been developed based on the theory of filter banks in signal processing [5] and compactly supported wavelets in applied mathematics [6]. These multiresolution image coding techniques are rich in theory and easy to implement. They have shown promising results at very low bit-rates by utilizing their interesting localization properties in time and scale (frequency) domains [7]. They do not suffer from the blocking effects of the block-based coding techniques, however they produce some considerable artifacts near the boundaries of the objects as the bit rate decreases. This artifact is known as the *ringing effect* and is due to the Gibbs phenomenon of the filter banks.

Segmentation based or the so called *second generation* image coding techniques try to exploit *structural* properties of the image in order to achieve compression at very low bit rates [8-11]. In these techniques the image is segmented by using edge/contour maps or hierarchical data structures. The boundary information and the internal luminance or texture of these disjoint regions are then coded separately [8]. Hierarchical data structures have gained more popularity because they inherently produce an efficient multiresolution representations of the image and are relatively easy to implement [11-13]. Moreover, they can be effectively used for browsing and indexing applications in multimedia storage systems. These schemes break away from the restraint of regular block structure BB-DCT algorithms and have the additional advantage of maintaining the edge details at very low bit rates. The price to be paid however is the computational complexity and the *unnatural contouring effects* which can make the details seem artificial. Although, the contouring effects might be preferable to the blocking effects of the frequency domain processed images which are overlaid by a regular block structure.

In this paper the Projection Pursuit encoding technique [14], [15] with a *dictionary* of spline functions is used to encode the quadtree [12] and Binary Space Partitioning (BSP) tree [9], [12] representation of digital still images. The performance of these techniques is then compared with JPEG at rates below 0.5 bit/pixel based on the Peak Signal-to-Noise Ratio (PSNR). The organization of this paper is as follows. Section 2 is devoted to an introduction of hierarchical data structures for image representation. Section 3 presents efficient encoding of the hierarchical data structures with projection pursuit and a dictionary of spline functions. Finally, the experimental results and the concluding remarks are provided in section 4.

2. SEGMENTATION AND HIERARCHICAL DATA STRUCTURES

Segmentation of natural gray-level images into regions of different sizes with variable amounts of detail and information can be useful for efficient coding of still images and video at low bit rates. There are a variety of hierarchical data structures for representing spatial data at multiple resolutions [12]. These models have been developed based on the principle of recursive decomposition and have found many applications in computer graphics, computer vision, pattern recognition, and image processing. The hierarchical data structures are attractive for image representation because they are relatively simple to implement, and they adaptively decompose the image into subregions which results in image segmentation. The most popular hierarchical data structures for image processing and computer vision applications are Quadtree and Binary Space Partitioning (BSP) binary tree [9-13].

Quadtree decomposition is a simple technique of representing images at multiple resolutions. In this technique, the image is recursively divided into four equal *square* regions depending on the *activities* in the blocks [13]. Quadtree segmentation of a 2^nx2^n image results in a tree whose root represents the original image at resolution level zero, and the four equally sized squares represent its children at resolution level one. Each pixel at every resolution level has its own intensity and the parent node intensity is equal to the mean value of the intensities of its children. At each node, a decision must be made as to whether to decompose the corresponding block into four equal size squares or to stop the decomposition. In order to arrive at a decision several measures of activity have been introduced in the literature, however the most widely used measure of activity is the *absolute difference* [12]. At each node, the value of the absolute difference is compared with a threshold value and if the absolute difference is smaller than the threshold, the recursive decomposition at that node is stopped. Otherwise, the node is further decomposed into four squares of equal sizes. Quadtree is the most widely used data structure in image processing applications.

The Binary Space Partitioning (BSP) is a recursive partitioning technique which has been used extensively in computer graphics since early 80's [12]. The BSP algorithm for image representation [9] takes as input an unpartitioned region R (initially the entire image) and a line l, which has been selected according to some criterion, to intersect R. Then it partitions R with l into two half-regions, R^- and R^+. The two half regions can then be similarly partitioned in a recursive manner until a termination criterion is met. This results in a hierarchy of convex regions called *cells*. A good segmentation is obtained when the pixel values within each cell are *homogeneous*. The non-leaf nodes of a BSP tree are associated with the partitioning lines, the leaves represent the cells of the image, and every node in the tree represents a convex region of the image.

Although the process of generating BSP-trees is more complex than that of quadtrees, the BSP-trees are more efficient than the quadtrees in representing images and hence more suitable for low bit rate image coding. These facts are illustrated in the following example. The quadtree and BSP-tree segmentation maps of a synthetic polygon are shown in Fig. 2.1. The quadtree was generated by using the absolute difference criterion [12] and the BSP-tree with a boundary-based Hough transform technique [9]. The quadtree and binary tree representation of Fig. 2.1(a) are shown in Fig. 2.2.

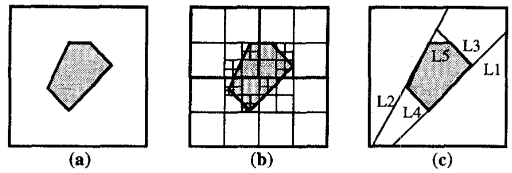

Fig. 2.1 - (a)The synthetic image of a polygon, (b)The Quadtree segmentation map of (a), (c) The boundary-based BSP segmentation map of (a).

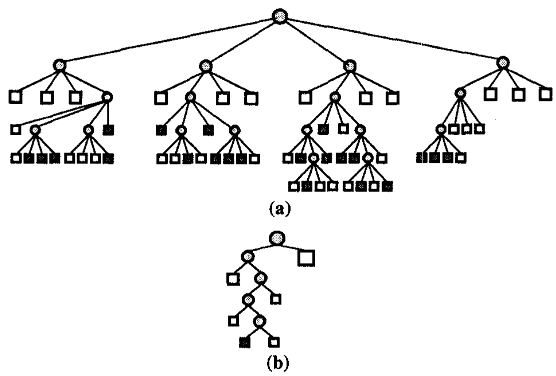

Fig. 2.2 - Representation of Fig. 2.1(a) by hierarchical data structures: (a) Quadtree, (b) Boundary-based BSP binary tree.

Note that unlike the quadtree representation which only allow square segmented regions, the segmented regions or cells of a BSP-tree could be arbitrary shaped polygons which results in a more efficient and compact representation of digital images. If we only consider the number of bits required to encode the tree (not the cells or segments), the quadtree needs 18% more bits than the BSP-tree in this example. In quadtree the partitioning lines are simply vertical and horizontal lines at dyadic intervals and no extra computation or bits are needed to encode these lines, but it is considerably more complex than a BSP-tree. For the BSP-tree, most of the bits and computational power are used for specifying and encoding of the partitioning line parameters (ρ,θ) at each node. However, the partitioning lines in a BSP-tree are chosen to align with the true edges of the objects in the image which is a desirable property in very low bit rate image coding techniques [8-11].

3. IMAGE CODING WITH HIERARCHICAL DATA STRUCTURES

In a hierarchical data structure the nodes contain the structural information of the tree and the leaves contain the intensity or texture information of the corresponding segmented regions or cells. Therefore, low bit rate image compression is possible if the tree structure and the cell informations can be coded efficiently. The complexity of the tree structure is directly related to the complexity of the image and the corresponding model for cell representation and the stopping criterion [12]. The basic theory of representation of spatial data requires that the shapes of the basis vectors used for this purpose to be as close as possible to those of the typical data vectors. However, it can be shown that over a redundant dictionary, this is an NP-complete problem. Therefore, it is reasonable to use a finite dictionary of functions with various degrees of smoothness for efficient approximation of the cells and to use the desired PSNR or bit rate as the stopping criterion at various resolution levels [11], [15].

3.1 Finite Dictionary Projection Pursuit Quadtree (PP-QT) Coding of Still Images

Projection pursuit [14-15] is an iterative technique for function approximation. In every iteration of this algorithm, from a set of fixed basis functions called a *dictionary,* a function is selected that *best* (typically based on the L_2 norm) approximates the *current* data in the given cell. In the first step of the iteration, the current image is the original image, and in the step k the current image is the *residual* image which is obtained by subtracting the linear combination of all the *(k-1)th* previous approximations from the original image. In this paper we used a finite dictionary of convolutional splines [17] of up to order 2 which constitute the space of all piecewise constants, linear functions, and polynomials of order 2. The first spline function is simply the indicator function of interval [0,1), and the higher order splines are obtained by convolving the two lower order splines. This finite dictionary of spline basis functions (multisplines) have the required regularity, symmetry, and compact support which are desired in image approximation applications[17]. Multisplines of up to order 2 which have been indexed on a web are successfully used in [18] for iterative multiresolution coding of images. This method however do not explicitly take advantage of the structural properties of the digital images.

Let $f(x,y)$ denote the intensity of the image at location (x,y), $\hat{f}_k(x,y)$ its estimated value, and $r_k(x,y)$ the residual image at iteration k. Let $\underline{\Theta}_k = \{g_k, \underline{\alpha}_k, \beta_k, \gamma_k\}$ be the set of parameters for the model at iteration k, where g_k correspond to one of the spline functions in the dictionary. Let $d(.,.)$ be the desired error metric, and $z^T = (x, y)$. Then, for every cell in the segmentation map let $r_0(\underline{z}) = f(\underline{z})$ and compute the optimal parameter vector $\underline{\Theta}_k^*$. To compute the optimal parameters at iteration k, let

$$J(\underline{z}) = d\{r_{k-1}(\underline{z}), \gamma_k g_k\{\underline{\alpha}_k^T \cdot \underline{z} + \beta_k\}\} \tag{3.1}$$

then

$$\underline{\Theta}_k^* = arg\{\min_{g_k \in \Phi}\{\min_{\underline{\alpha}_k, \beta_k, \gamma_k} \sum_{\underline{z} \in B} J(\underline{z})\}\} \tag{3.2}$$

and the reconstructed image at iteration k can be written as

$$\hat{f}_k^*(\underline{z}) = \sum_{l=1}^{k} \gamma_l^* g_l^* (\underline{\alpha}_l^{*T} \cdot \underline{z} + \beta_l^*) \tag{3.3}$$

and the residual image at iteration k is given by

$$r_k(\underline{z}) = f(\underline{z}) - \hat{f}_k^*(\underline{z}) \tag{3.4}$$

Then compute the PSNR and the bit rate at this iteration. If these values does not satisfy the desired limits, iterate through equation (3.2) to (3.4), otherwise

$$\hat{f}_{L^*}(\underline{z}) = \sum_{l=1}^{L^*} \gamma_l^* g_l^* (\underline{\alpha}_l^{*T} \cdot \underline{z} + \beta_l^*) \tag{3.5}$$

We used a maximum of 4 iterations at each resolution of the quadtree and if the desired PSNR or bit rate was not reached at a particular node, then the tree was expanded at that node. This procedure can be repeated until the stopping criterion is satisfied. The entropy coding of the quadtree representation of images is well studied in the literature [2], [13], [15]. In this paper we used Lloyd-Max quantizers along with adaptive arithmetic coding [19] to encode the tree and the cell information separately into the compressed image.

3.2 Finite Dictionary Projection Pursuit BSP-tree (PP-BSP) Coding of Still Images

The major problem here is the quantization and encoding of the partitioning line parameters ρ and θ. In [10] the lines are restricted to have four possible orientations: horizontal, vertical, 45 degrees, and 135 degrees. This results in convex polygons with at most 8 sides for the cell structure. In [9] an arbitrary orientation for the partitioning lines are permitted. Therefore the cells could be represented with arbitrary shape polygons. The price to be paid is however more computational complexity and difficulty in optimal quantization and coding of the orientation parameter θ. Here we assumed 16 possible orientations for each partitioning line based on our experimental studies. The construction of the projection pursuit BSP-tree is similar to that of quadtree, with the exception that at each node the parameters of the line are obtained by minimizing the error ε, given by

$$\varepsilon[\Re_l(\theta)] = \sum_{\underline{z} \in \Re_l(\theta)} \{f(\underline{z}) - \gamma_k g_k\{\underline{\alpha}_k^T \cdot \underline{z} + \beta_k\}\}^2 \tag{3.6}$$

where θ corresponds to the orientation of the partitioning line l, and $\Re_l(\theta)$ denotes the segmented regions on both sides of the line associated with θ. When the desired stopping criterion is met, the line parameters and cell attributes are quantized and entropy coded separately by using Lloyd-Max quantizers and adaptive arithmetic coding [19] to construct the compressed image. The encoded bit stream then can be easily decoded and recombined at the receiver up to the desired resolution. The experimental results for PP-QT and PP-BSP algorithms are provided in the following section.

4. EXPERIMENTAL RESULTS

The proposed algorithm was tested on different test images. The original test image Lenna (512x512x8), which is the most widely used test image in the image compression literature, is shown in Fig. 4.1(a). The decoded Lenna at 0.125 bit/pixel and PSNR of 27.25 dB, which has been encoded with the recursive PP-QT algorithm of section 3 is shown in Fig. 4.1(b). The decoded Lenna at 0.125 bit/pixel and PSNR of 27.65 dB, which has been encoded with the recursive PP-BSP algorithm of section 3 is shown in Fig. 4.1(c). For comparison, the JPEG reconstructed Lenna at 0.125 bit/pixel and PSNR of 26.20 dB is shown in Fig. 4.1 (d). In this set of experiments we used Laplacian and Gaussian Lloyd-Max quantizers based on the distribution of the line parameters and representative parameters of the spline functions. The resulting quantized values were then encoded by using an adaptive arithmetic encoder [19]. In general, our segmentation based algorithms performance is superior to JPEG at rates below 0.5 bit/pixel based on subjective image quality as illustrated in Fig. 4.1 and the PSNR measure of quality as shown in Fig. 4.2. Moreover, due to the multiresolution structure and segmentation properties of the hierarchical data structures, their compressed bit streams can be used for progressive image transmission and browsing applications. It is important to note that the PP-QT has a variable size square block structure which causes some blocking effects, and the PP-BSP has a variable size polygonal structure which causes some contouring effects in the decoded image, at very low bit rates. The contouring effects might be preferable to the blocking effects depending on the subjective evaluation of the human observers. The main disadvantage of segmentation-based compression techniques is their computational complexity (in particular PP-BSP) compared to the BB-DCT algorithms such as JPEG. Therefore, for storage and transmission of images at higher rates, it might be advantageous to use JPEG, because of its simplicity and good performance.

ACKNOWLEDGMENTS

The work of the first two authors was partially supported by the Innovative Science and Technology (IST) program of the BMDO monitored by the Office of Naval Research under contract ONR N00014-91-J-4126.

REFERENCES

[1] V. Bhaskaran and K. Konstantinides, *Image and Video Compression Standards*, Kluwer, 1995.

[2] R. Clarke, *Digital Compression of Still Images and Video*, Academic Press, 1995.

[3] A.E. Jacquin, *"Fractal Image Coding: A Review,"* Proceedings IEEE, vol. 81, Oct. 1993.

[4] K. Barthel, J. Schuttemeyer, T. Voye, and P. Noll, *"A new Image Coding Technique Unifying Fractal and Transform Coding,"* IEEE 1st International Conf. Image Processing, 1994.

[5] O. Rioul and M. Vetterli, *"Wavelets and Signal Processing,"* IEEE SP Magazine, pp. 14-38, Oct. 1991.

[6] I. Daubechies, *Ten Lectures on Wavelets*, SIAM Publications, 1992.

[7] J.M. Shapiro, *"Embedded Image Coding Using Zerotress of Wavelet Coefficients,"* IEEE Trans. Signal Processing, vol. 41, Dec. 1993.

[8] M. Kunt, A. Ikonomopoulos and M. Kocher, *"Second Generation Image Coding Techniques,"* Proceedings IEEE, vol. 73, April 1985.

[9] H. Radha, R. Leonardi, M. Vetterli and B. Naylor, *"Binary Space Partitioning (BSP) Tree Representation of Images,"* Journal of Visual Communication and Image Representation, Sept. 1991.

[10] X. Wu and Y. Fang, *"A Segmentation-Based Predictive Multiresolution Image Coder,"* IEEE Trans. Image Processing, vol. 4, Jan. 1995.

[11] H.R. Rabiee, R.L. Kashyap, and H. Radha, *"Multiresolution Image Compression with BSP Trees and Multi-Level BTC,"* proceedings of IEEE 2nd International Conference on Image Processing, Washington D.C., 1995.

[12] H. Samet, *Application of Spatial Data Structures: Computer Graphics, Image Processing, and GIS*, Addison Wesley, Reading, MA, 1990.

[13] J. Vaisey and A. Gersho, "Image Compression with Variable Block Size Segmentation," IEEE Trans. Signal Processing, vol. 40, no. 8, Aug. 1992.

[14] P. Huber, "*Projection Pursuit*," Ann. Statist., vol. 13, 435-525, 1985.

[15] S.R. Safavian, H.R. Rabiee, and M. Fardanesh, *"Adaptive Multiresolution Image Coding with Projection Pursuit Neural Networks,"* IEEE 29th Asilomar Conf., Pacific Grove, CA, 1995.

[16] A. Blake and A. Zisserman, *Visual Reconstruction*, MIT Press, Cambridge, MA, 1987.

[17] A. Aldroubi and M. Unser, *"Families of Multiresolution and Wavelet Spaces with Optimal properties,"* Numer. Func. Anal. Optimiz., vol. 14, pp. 417-446, 1993.

[18] S. Moni and R.L. Kashyap, "Multiresolution using Multisplines for Image Compression," proceedings of IEEE 2nd International Conference on Image Processing, Washington D.C., 1995.

[19] N.S. Jayant and P. Noll, *Digital Coding of Waveforms*, Prentice Hall, 1984.

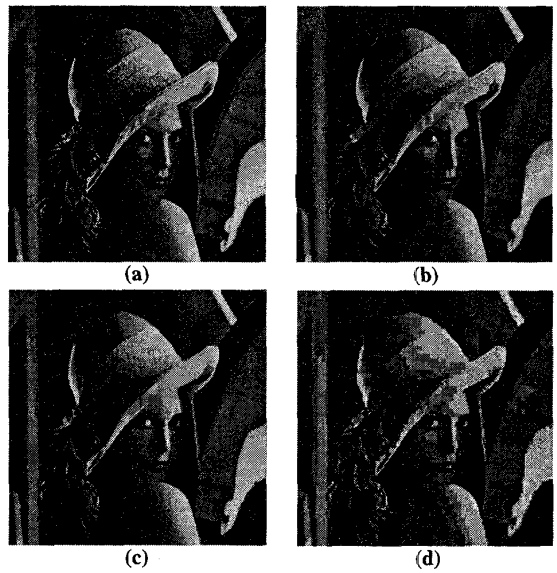

Fig. 4.1 - (a) Original Lenna (512x512x8), (b) Decoded Lena: PP-QT algorithm at 0.125 bit/pixel and PSNR=27.25 dB (c) Decoded Lenna: PP-BSP algorithm at 0.125 bit/pixel and PSNR= 27.65 dB, (d) Decoded Lenna: JPEG base-line algorithm at 0.125 bit/pixel and PSNR= 26.20 dB.

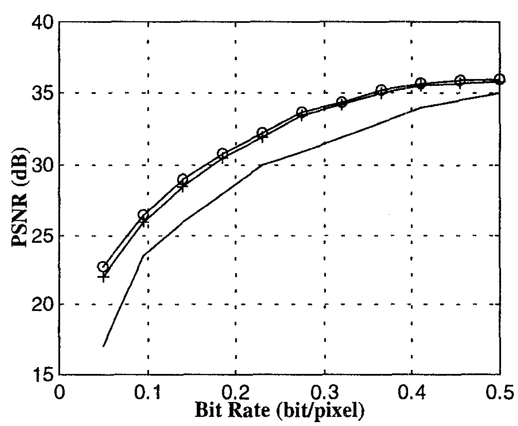

Fig. 4.2 - PSNR vs. bit rate curves for the decoded Lenna: PP-BSP (-o-), PP-QT (-+-), and JPEG (___).