MULTIRESOLUTION IMAGE COMPRESSION WITH BSP TREES AND MULTILEVEL BTC

Hamid R. Rabiee[1,2], R. L. Kashyap[1] and H. Radha[2]

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

[2]AT&T Bell Laboratories
Indianapolis, IN 46250

ABSTRACT

This paper presents a new multiresolution segmentation-based algorithm for image compression. High quality low bit rate image compression is achieved by recursively coding the Binary Space Partitioning (BSP) tree representation of images with multilevel Block Truncation Coding (BTC). Comparison with JPEG at rates below 0.25 bit/pixel shows superior performance both in terms of Power Signal-to-Noise Ratio (PSNR) and subjective image quality.

1. INTRODUCTION

The storage and transmission of digital images in their original or raw format is usually very expensive or impractical. In order to make the widespread use of visual information in modern communication systems practical, data compression at very low bit rates must be employed. In general, an acceptable image compression technique for interactive multimedia systems should at least be able to produce good quality reconstructed images at very low bit rates (below 0.25 bit/pixel) and to provide progressive transmission.

One of the main objectives in any lossy compression technique is to optimize the trade off between the amount of compression measured in bits per pixel (bits/pixel), and the reconstructed image quality measured by the Peak Signal-to-Noise Ratio (PSNR) or subjective evaluation [1]. In block-based or subband lossy image compression techniques, visual distortions in the form of *blocking effects* or *ringing effects* are noticed as the bit rate decreases. This results in the degradation of the reconstructed image and the reduction of its PSNR. For example, the still image compression standard JPEG [2], suffers from blocking effects at rates below 0.25 bit/pixel.

In progressive transmission, image information is transmitted in multiple stages. At each stage, an approximation to the original image at a lower resolution is reconstructed at the receiver and the reconstructed image progressively improves as more information is received. Progressive transmission is required in many multimedia applications where a user may only have access to low bandwidth communication channels. For example, if progressive transmission in a telebrowsing application is used, the user can stop the transmission of an intermediate version of an image if it is of no interest to him and consequently reduce the required search time and bandwidth.

The above mentioned requirements provoke the use of *multi-resolution segmentation-based* image coding techniques for the future modern communication systems. The current research in very low bit rate lossy image compression includes fractal coding [3], subband/wavelet coding [4], and segmentation based coding [5].

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. Despite their appealing mathematical foundations, fractal coding techniques have not yet shown satisfactory results at very low bit rate applications. Subband coding techniques have been developed based on the theory of filter banks in signal processing [6] and compactly supported wavelets in applied mathematics [7]. These multiresolution image coding techniques are rich in theory and easy to implement. They have shown promising results at very low to moderate bit-rates by utilizing their interesting localization properties in both time and scale (frequency) domains. 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. In these techniques the image is segmented using edge/contour maps [8] or hierarchical data structures [9], [10]. Hierarchical data structures have gained more popularity because they inherently produce a compact multiresolution representations of the data and are relatively easy to implement.

In this paper we introduce a new lossy multiresolution algorithm for very low bit rate image compression based on Binary Space Partitioning (BSP) [10] representation and multilevel Block Truncation Coding (BTC) [11], [12], [15] of images. 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 an overview of the BTC algorithm. Section 4 describes the new multiresolution BTC-BSP image coding algorithm. Finally, the preliminary experimental results and the concluding remarks are provided in section 5.

2. HIERARCHICAL DATA STRUCTURES

Natural gray-level images can usually be divided into regions of different sizes with variable amounts of detail and information. Such segmentation of the image is useful for efficient coding of images. There are a variety of hierarchical data structures for representing spatial data at multiple resolutions[9].

These models have been developed based on the principle of recursive decomposition and have found many applications in computer graphics, computer vision, pattern recognition, solid modeling, image processing, and geographic information systems. The hierarchical data structures are attractive for the following reasons. They are relatively simple to implement, they adaptively decompose the image into subregions, and the decomposition actually results in image segmentation.

The most popular hierarchical data structures for image processing and computer vision applications are Quadtree [9], Horizontal Vertical (HV) binary tree [9], and Binary Space Partitioning (BSP) binary tree [9], [10].

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. Quadtree segmentation of a $2^n x 2^n$ 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 nodes. 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* [9]. 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 generation of HV tree is similar to that of Quadtree with the exception that each node has only two children and the decomposition of the nodes alternates between horizontal and vertical bisections of the regions.

The Binary Space Partitioning (BSP) tree is a recursive partitioning technique which takes as input an unpartitioned region R initially the entire image, and a line l (selected according to some criterion) which intersect R, and produces as output two new regions formed by partitioning of R by 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. Note that unlike the quadtree and HV binary tree representations which only allow square or rectangular segmented regions, the segmented regions of a BSP tree could be arbitrary shaped polygons which results in a more efficient and compact representation of digital images. Moreover, 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 segmentation-based image coding algorithms [13]. These facts are illustrated in Figures 2.1, 2.2, and 2.3.

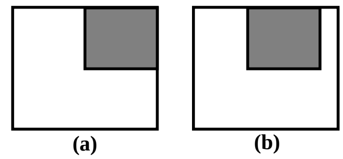

Fig. 2.1 - (a)The synthetic image of a square, (b)the same square as in (a) shifted to the left by one pixel [14].

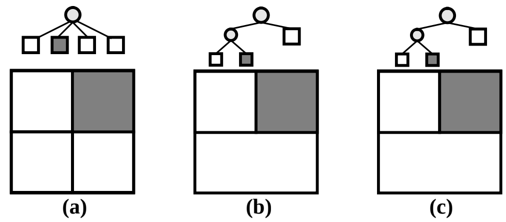

Fig. 2.2 - Representation of Fig. 2.1(a) by hierarchical data structures:(a) Quadtree, (b)HV binary tree, (c)BSP binary tree.

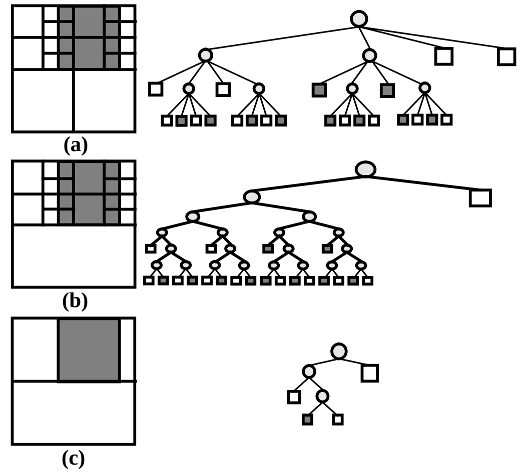

Fig. 2.3 - Representation of Fig. 2.1(b) by hierarchical data structures: (a) Quadtree, (b) HV binary tree, (c) BSP binary tree. Clearly, the BSP representation is more efficient than Quadtree and HV binary tree and can be encoded at lower rates.

3. BLOCK TRUNCATION CODING

Block Truncation Coding (BTC) [11] is an efficient and simple algorithm which has been used by many researchers for coding images at low to moderate bit rates [15]. In the basic bi-level BTC algorithm the input image is divided into non-overlapping blocks of size NxN (normally N=4 or 8) and the pixels in each block are divided into *low* and *high* intensity pixels by using a threshold t to generate the truncation map $\boldsymbol{T}(i,j)$;

$$\boldsymbol{T}(i,j) = \begin{cases} 1, & \text{if } x(i,j) \geq t \\ 0, & \text{if } x(i,j) < t \end{cases} \qquad (3.1)$$

where $x(i,j)$ is the gray level of the pixel at location (i,j). In each block two representative low and high levels a and b are calculated. At the decoder, the reconstructed image $y(i,j)$ can be computed as;

$$y(i,j) = a + (b-a)\,\boldsymbol{T}(i,j) \tag{3.2}$$

In the basic BTC, compression is achieved by only transmitting the binary truncation block and the representative parameters a and b (8 bpp). Therefore for the basic BTC algorithm the overall bit rate is equal to $1+16/N^2$.

There have been a variety of modifications [15] to the basic BTC algorithm in recent years. Most of these modifications try to improve the Mean Squared Error (MSE) performance of the basic BTC algorithm by selecting the optimal threshold t and the representative parameters a and b, or by using variable block sizes (Quadtree). In general, the quality of the bi-level BTC encoded images at moderate bit rates are good. However at lower rates, they often suffer from blocking effects by producing *ragged* high contrast edges within and across the blocks. In [12] an equi-spaced 3-level BTC was introduced to alleviate this problem. In a 3-level BTC, the gray levels $x(i,j)$ in a block are coarsely quantized into 3 equi-spaced levels $Q_k=c+sk$ ($k=-1,0,1$) by the set of thresholds t_k, and the truncation map is given by;

$$\boldsymbol{T}(i,j) = k \quad if \quad t_k \le x(i,j) < t_{k+1} \tag{3.3}$$

where $t_{-1} = -\infty$ and $t_2 = \infty$. Here the parameters t_k, c, and s must be estimated and the reconstructed image $y(i,j)$ is given by;

$$y(i,j) = c + s\boldsymbol{T}(i,j) \tag{3.4}$$

In our algorithm we alternatively used the 2-level ELM-BTC [15] and the modified 3-level LM-BTC [13] based on the *visual entropy* [16] of the segmented regions.

4. BTC-BSP CODING OF STILL IMAGES

The Quadtree data structures have been successfully used in segmentation-based image coding techniques [8], [16]. More recently, there has been a growing interest in low bit rate image compression using binary trees [5], [14], [17]. In [5] a Binary Adaptive Segmentation (BAS) tree is first generated by adaptively partitioning the image with lines which are restricted to four possible orientations of 0, 45, 90, and 135 degrees. The partitioning lines are selected by using a least-squares piecewise linear approximation and the segmented regions are coded by a predictive Vector Quantization [VQ] encoding scheme. Note that the BAS trees can be considered as subsets of the more general BSP trees, because the restriction on the partitioning lines in BAS tree confines the shape of the segmented regions to polygons with a maximum of 8 vertices, whereas in a BSP tree, arbitrarily shaped polygons are possible. This flexibility of the BSP trees may lead to a more efficient representation of the data and consequently better compression performance.

The BSP tree was first used for representation and coding of binary data [9] and more recently has been used for representation and coding of gray scale images [10], [14], [17]. In [17] both a modified Hough Transform (HT) and a Least-Squares Partitioning Line (LPL) transform is used to select the partitioning lines with arbitrary orientations. The binary tree structure bits, partitioning line parameters (ρ and θ) and the cell attributes are then adaptively quantized and encoded, separately. Finally, an optimal pruning algorithm is used to reduce the bit rate.

Two important issues in BSP image compression are the selection of the partitioning lines and the selection of the termination criterion. As was explained earlier, in both [5] and [17] a least-squares approach was used for selecting the partitioning lines and first or zeroth order polynomials were used to model the segmented regions or cells. The number of possible line orientations in BAS is not adequate, and it might not always be advantageous to consider all the possible orientations for generating the BSP tree [13]. It is possible to use higher order polynomials for modeling the cells, in order to obtain smoother reconstructed images. However this may significantly increase the computational complexity of the algorithms. In summary, a simple cell model requires a more complicated tree (specially for texture regions) and a more complicated cell model requires more bits and computational power. In order to alleviate some of these inter-related problems, we used the following algorithm which uses the power and simplicity of the BTC and takes advantage of the efficiency of the BSP tree representation.

At the coarsest level of the tree (level 0), our algorithm segments the image into two half planes by using a piecewise constant approximation technique based on the least-squares optimization [18] (without any continuity constraints across the partitions). We limit the orientation of the partitioning lines to $15k$ (k=0,1, ... , 12) degrees and among all possible bipartitions, we choose the one that leads to the best approximation of the two segmented regions in least-squares sense. In other words we try to minimize the error ε given by;

$$\varepsilon\,[\Re_d(k)] = \sum_{(i,j)\,\in\,\Re_d(k)} [x(i,j) - y(i,j)]^2 \tag{4.1}$$

where k corresponds to the orientation of the partitioning line, $\Re_d(k)$ denotes the segmented regions on both sides of the line associated with k, and $y(i,j)$ is given by (3.2) or (3.4) depending on the visual entropies of the selected regions. After selecting the partitioning line, each half plane is encoded by the corresponding 2 or 3 level BTC algorithm. Before generating the next finer level of resolution by expanding the BSP tree, the PSNR or the bit rate is computed to serve as a termination criterion. If the PSNR or the bit rate meet the requirements at any level of the tree, the segmentation is terminated. Otherwise, the mean values of the segmented regions are saved as node attributes (to create a multiresolution representation of the image), and then the segmentation is recursively continued in each of the bipartitions until the termination criterion is met. In order to create the compressed BTC-BSP bit stream we split the information into three categories of tree structure attributes, partitioning line attributes, and cell attributes. The tree structure attributes are binary flags and are entropy coded with arithmetic coding [1]. The line attributes, ρ and θ, are quantized by Lloyd-Max quantizers and are adaptively entropy coded with arithmetic coding. Finally, the cell attributes a, b, c, and s were quantized by Lloyd-Max quantizers and the truncation maps were run-length coded to exploit the inter-symbol redundancies, before using the arithmetic encoder on these set of attributes.

These informations could be easily merged at the decoder to reconstruct the encoded image at multiple resolutions. The proposed algorithm was used to encode a variety of natural images. Some of the preliminary experimental results on the test image Lenna, are provided in the following section.

5. EXPERIMENTAL RESULTS

The proposed algorithm was tested on different natural images. The original test image Lenna (512x512x8), which is the most widely used test image in the image compression literature, is shown in Fig. 5.1 (a). The BSP segmentation map of Lenna which was generated with the modified piecewise least-squares approximation of section 4 is shown in Fig. 5.1 (b). The decoded Lenna at 0.15 bit/pixel and PSNR of 28.59 dB, which has been encoded with the recursive BTC-BSP algorithm of section 4 is shown in Fig. 5.1(c). For comparison, the JPEG reconstructed Lenna at 0.15 bit/pixel and PSNR of 27.55 is shown in Fig. 5.1 (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 BTC algorithm. The BTC algorithm showed to be more effective than low order polynomials to encode the texture regions of the image. It was also very efficient in encoding the smooth segmented regions of the image, because the relative smoothness of the cells, produced truncation maps which had high degrees of inter-symbol correlations. These inter-symbol correlations were fully exploited by run length coding of the truncation maps.

In general, the BTC-BSP algorithm performance is superior to JPEG at rates up to 0.3 bit/pixel both in terms of PSNR and subjective image quality. Therefore, at higher rates, it might be advantageous to use the block-based DCT coders such as JPEG, because of their simplicity and good performance. A comprehensive comparison study between different classes of segmentation-based image compression techniques and the JPEG standard can be found in [13].

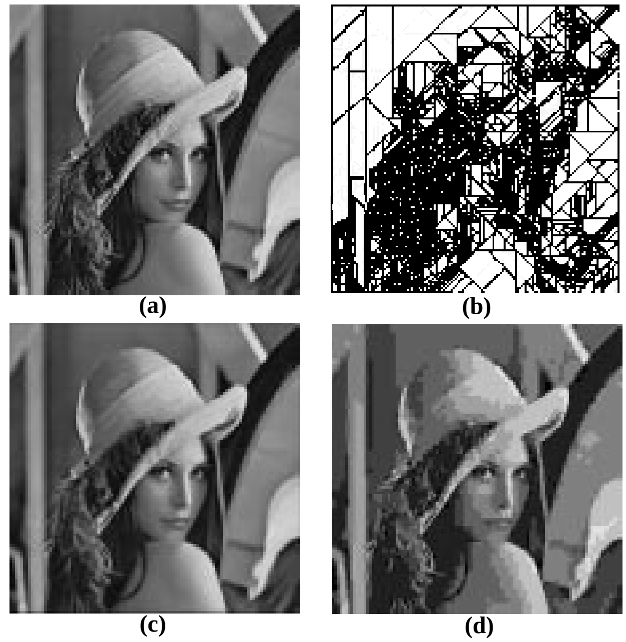

Fig. 5.1 - (a) Original Lenna (512x512x8), (b) BSP segmentation map of Lenna with the modified piecewise least-squares approximation, (c) Decoded Lenna: BTC-BSP algorithm at 0.15 bit/pixel and PSNR=28.59 dB, (d) Decoded Lenna: JPEG base-line algorithm at 0.15 bit/pixel and PSNR=27.55 dB.

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

[2] W. B. Pennebaker and J. L. Mitchell, *JPEG Still Image Data Compression Standard*, Van Nostrand Reinhold, New York, 1993.

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

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

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

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

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

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

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

[10] 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.*

[11] E.J. Delp and O.R. Mitchell, *"Image Compression Using Block Truncation Coding,"* IEEE Trans. Comm., vol. COM-27, 1979.

[12] E. Walach, D. Chevion and E. Karnin, "On Modification of BTC approach to image compression," in Signal Processing III, Elsevier, Amsterdam, 1986.

[13] H. Rabiee, R.L. Kashyap and H. Radha, "Multiresolution Segmentation-Based Techniques For Low Bit Rate Image Compression," IEEE Trans. in Image Processing, *to be submitted*.

[14] H. Radha, *PhD Thesis*, Columbia Univ., May 1993.

[15] H. Mitchell, N. Zilverberg and M. Avraham, *"A Comparison of Different Block Truncation Coding Algorithms for Image Compression,"* Signal processing nad Image Commun., no. 6, 1994.

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

[17] H. Radha, M. Vetterli, and R. Leonardi, *"Image Compression Using Binary Space Partitioning Trees," Submitted to* IEEE Trans. Image Processing, 1994.

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