COLOR IMAGE CODING, INDEXING AND RETRIEVAL USING BINARY SPACE PARTITIONING TREE

G Qiu and S Sudirman

School of Computer Science, University of Nottingham, United Kingdom

ABSTRACT

This paper presents a unified approach to colour image coding, content-based indexing, and retrieval for database applications. The binary space partitioning (BSP) tree, traditionally used in gray scale image coding [1, 2] is extended to represent colour images. The BSP tree, hence the structure, of the image is explicitly coded. A method is developed to compute the similarities of images based on their BSP tree representations. In image database applications, the images in the database are coded by BSP tree to achieve a good balance between storage efficiency and easy manipulation of image data. Content-based image querying is performed in the compressed bit streams by comparing the BSP tree of the query image with those of the images in the database.

1. INTRODUCTION

Image coding is one of the most successful fields compared to other relative fields in the sense that many mature technologies have been developed and widely used in many real life applications. Traditionally, the goal of image coding is to strive for high compression ratio and low distortion. However, with the rapid advancement in processor speed, storage device technology and faster network connection, low bit rate coding is no longer a critical factor in many practical applications. Based on a certain trade-off, higher bit rate and complexity can be acceptable. On the other hand, with very large image collections becoming more and more common, effectively managing large image database, making images easily accessible have become a challenge. Modern imaging systems not only require efficient coding, but also easy manipulation, indexing, and retrieval, the so-called "4th criterion" in image coding [3].

Recently, image database indexing/management has been actively researched by researchers from a wide range of disciplines including those from computer vision, image processing and traditional database areas. One particularly promising approach to image database indexing and retrieval is the query by image content (QBIC) method [4]. QBIC uses the visual contents of the images, such as colour distribution (colour histogram), texture attributes and other image features as indexing keys. In an image database, these visual keys are stored along with the actual imagery data, and image retrieval from the database is based on the matching of the models visual keys with those of the query images. Because extra information has to be stored with the image, traditional approach to QBIC is not efficient in terms of data storage. Not only is it inefficient, it is also inflexible in the sense that image matching/retrieval can only based on the pre-computed set of image features.

It is therefore desirable to build up image databases that are accessible "midstream" [5], i.e. using a compression model which allows image query, retrieval and modification to proceed on the compressed representation [4]. Although it is possible to index images in the compressed domain of transform-based image coding, such as JPEG (DCT) [6] and Wavelet [7], the compressed image streams of these models are not directly usable and quite complicated processing is required to compute image features for image indexing and recognition purposes [8]. Furthermore, indexing in the compressed domains of these transform-based models is based on histogram technique [6, 7], which lacks information about the spatial distribution and relation of image features and provides no visual semantic for linking high level contextual information with low level bits. To meet the new requirements, segmentation-based approach [9], or the "second generation" image coding [10], which divides image into meaningful regions, may provide a better solution than transform-based models.

However, current state of the art computer vision/image processing techniques are still not mature enough yet to ensure accurate segmentation of the images into meaningful regions (objects). Even if accurate segmentation can be performed, there is the added problem of representing the images into constituent regions effectively and efficiently to meet the multiple requirements of compression, indexing and retrieval. The binary space partitioning tree method [1, 2] is a second generation image coding technique based on constraint image segmentation providing a binary tree data structure which not only has a compact representation but also preserve the scene semantics. In this work, we extend the BSP tree coding method to colour image and present a

method for computing the similarities of images based on their BSP tree representation for image database applications.

2. COLOUR IMAGE CODING USING BSP TREE

We have extended the BSP tree image coding method to colour image, a brief description is given here and more details can be found in [11]. The method is proceeded as follow:

A binary quaternion moment thresholding [12] is used to binaries the whole colour image.

A partitioning (straight) line is then chosen to divide the image into two regions such that at least one of the regions is relatively homogenous, i.e., for a binary image it is either mostly black or mostly white. Since there are infinite number of possible lines which will partition the image, the line parameters are quantized [2]. A line selection criterion is used to allow the pixel impurity on either sides of the line to be controlled to enable more meaningful partitioning [11]. The mean colours of the two partitioned regions are then computed.

The process of thresholding, finding an optimal line and the calculation of the mean colours, is repeated for each region. The data used for the thresholding is the data from the original image. However, this time instead of thresholding the whole image, we threshold the regions resulted from the previous partition independently of each other. A region will not be partitioned if it becomes too homogenous or if it becomes too small. The process is repeated until no more regions can be partitioned or until it has reached a certain number of iterations.

The partitioned image information is put into a binary tree data structure (Fig. 1) where each non-leaf node holds the average colour and parameters of the straight line which partitioning the region, and the leaf nodes hold the average colours of the smallest region (not further partitioned)

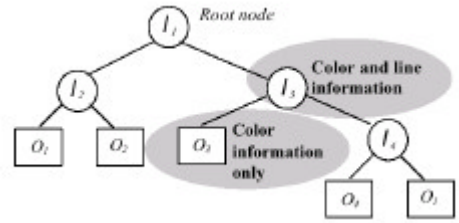

Fig.1 The image structure is encoded in a binary tree

The tree is coded in a top-down fashion. Starting from the root node, then the left child, and then the right child, until it reaches the leaves. 1 bit is used to signify whether a node is partitioned or not. It a node is partitioned, the line parameters (ρ and θ in normal representation $r = xcosq+ysinq$) are coded, if not the average colour is coded. θ is quantized to 8 possible values and ρ is quantized according to the method of [2].

Although we have not exploited all possible redundancies exist in the BSP tree representation, the compression performance are satisfactory. Fig. 2 shows an example of coding an outdoor scene at various bit rates. Compared with transform-based coding, such as JPEG, at moderate bit rates, typically, about 1.5 bpp, the rate distortion performances of the BSP tree method are comparable to those of JPEG's [11].

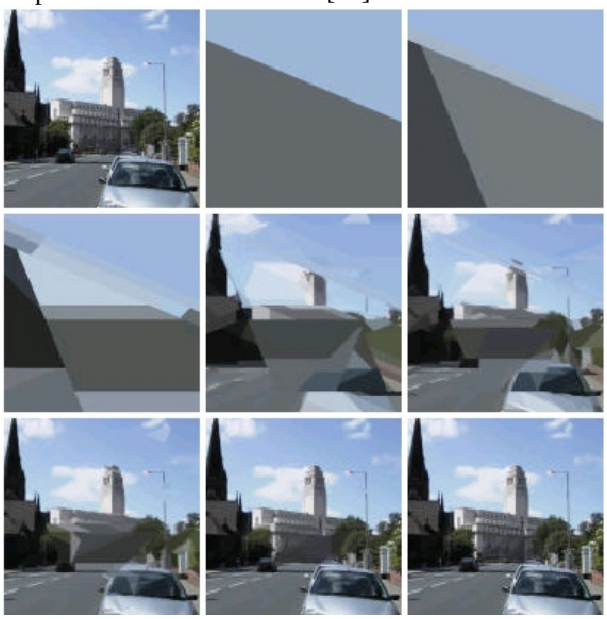

Fig.2, BSP Tree coded outdoor scene. From left to right and top to bottom: Original image (24 bpp), 1st partition (4e-4 bpp), 2nd (6e-4 bpp), 4th partition (2e-3 bpp), 8th partition (0.33 bpp), 10th partition (0.1 bpp), 12th partition (0.27 bpp) and 20th partition (1.3 bpp).

3. COMPUTING THE SIMILARITIES OF COLOUR IMAGES USING THEIR BSP TREES

A tree is a special type of connected graph, a systematic approach to image matching based on the BSP tree representations could follow a graph-matching approach [13, 14]. Alternatively, we could compute the distance between trees [16, Chapter 4]. Our ongoing work is being directed towards these directions. Graph-matching/tree distance computation are themselves very complicated tasks. The association of the visual semantics with the tree (graph) makes the tasks even harder, because it is not only the structure of the tree, but also the contents of the nodes have to be matched in an appropriate manner to reflect the visual similarities of images. In this section, we should present a somewhat empirical method for computing the similarities of images based on their BSP tree segmentation. We are hopeful to report a more formal approach (based on graph matching [14] and tree distance [16]) to the conference in 10 months time.

We define an entity in the tree called *node-family* of a tree. A node family is a sub-tree consisting of a node and its two children. Let $\boldsymbol{N}_k(i,j)$ denote the j^{th} node family at layer i of tree k, and $\boldsymbol{N}_l(m,n)$ denote the n^{th} node family at layer m of tree l. Comparison of the contents of the two images represented by trees k and l is performed as follow: For each $\boldsymbol{N}_k(i,j)$, for all i and j, calculate the node family differences $\boldsymbol{DN}$ (definition given later) between $\boldsymbol{N}_k(i,j)$ and all the node families of the tree l at levels i-1, i, and i+1 (i and i+1 if i is the root layer, and i-1 and i if i is a leaf layer), and take the minimum node family difference value and denote it as $\boldsymbol{D_k}(i,j)$. The difference from tree k to tree l $\boldsymbol{DT}_{k\text{-}l}$ is the average of $\boldsymbol{D_k}(i,j)$ over all i and j. The same process is repeated from tree l to tree k to calculate $\boldsymbol{DT}_{l\text{-}k}$ since $\boldsymbol{DT}_{k\text{-}l}$ is not necessarily equal to $\boldsymbol{DT}_{l\text{-}k}$. And the overall tree difference $\boldsymbol{DT}(k,l)$ is calculated as the average of the two,

$$\boldsymbol{DT}(k,l) = (\boldsymbol{DT}_{k\text{-}l} + \boldsymbol{DT}_{l\text{-}k})/2 \quad (1)$$

Let j and n be two node families. Let L_{jp} and L_{np} be the lines associated with the parent nodes of j and n respectively. Let C_{jl} and C_{jr} be the colors associated with the left and right child node of j respectively. Let C_{nl} and C_{nr} be the colors associated with the left and right child node of n respectively. The node family difference between j and n, $\boldsymbol{DN}(j,n)$ is defined as the sum of the differences between the lines and the colors, i.e.

$$DN(j,n) = l_1 D(L_{jp}, L_{np}) + l_2 (D(C_{jl},C_{nl}) + D(C_{jr}, C_{nr}))/2 \quad (2)$$

where λ_1 and λ_2 are weighting factors giving different importance to the colour and line (structure) differences (they have to be determined empirically) and

$$D(L_{jp}, L_{np}) = (D(\theta_{jn}) + D(\rho_{jn}))/2$$
$$D(\theta_{jn}) = \min(|\theta_j - \theta_n|, \pi - |\theta_j - \theta_n|); \; D(\rho_{jn}) = |\rho_j - \rho_n|$$
$$D(C_{jl},C_{nl}) = \| C_{jl} - C_{nl} \|; \; D(C_{jr},C_{nr}) = \| C_{jr} - C_{nr} \|$$

Because the line parameters and colors represent different modalities, care must be taken in combining them together. One difficulty in combining the distances of the partitioning lines and the distance of colors is that they represent *a priori* not comparable modalities, with different dynamic ranges. In the absence of any systematic method for combining different modalities, we propose to normalize all difference entities before combining them. We believe the problem of combining different modalities is intrinsically hard, extensive experimenting work is necessary to assess the importance of different modalities after normalization. Experiments are currently on going to select more appropriate values for λ_1 and λ_2, however, the results presented here are those of $\lambda_1 = \lambda_2 = 1$.

In our co-ordinate definition, the origin is the top-left hand corner of the image, the values of q range from $-p/2$ to $p/2$. Assuming the image height is H and width W, the values of r range from $-H$ to $\ddot{O}(H^2+W^2)$, and the RGB values of the color has a range between 0 – 255. Before calculating the distances, q's are normalized to [-0.5,+0.5], r's are also normalized to (-0.5,+0.5] and RGB's are normalized to [0,1].

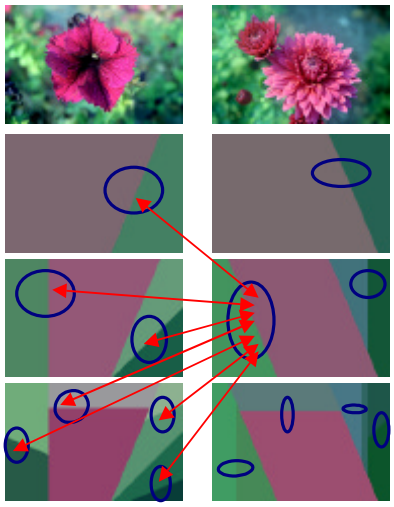

Fig. 3, Illustration of image matching based on BSP tree representation

The explanation of the matching method just described can be illustrated using Fig. 3. These are the actual BSP tree partitions of two similar images. A circle in the figure indicates a family node, i.e., it covers one straight line and the two regions separated by the line. As illustrated, a family node on the right image is compared with 7 node families on the left image (depending on the layer position of the node, the number of comparisons is different). Intuitively, each family node captures a microstructure of the scene (at different resolutions), the matching algorithm tries to match these microstructures of the two matching image at a similar resolution. The rationale is that if two images have similar microstructures (as partitioned by BSP tree) then they are likely to be similar overall. Bearing in mind that these microstructures are themselves formed by hierarchical segmentation, and therefore represent meaning scene partitions. As an example, Fig. 4 illustrates two similar scenes and their first level partition, and in fact we have observed similar meaningful structural partitioning on many scenes. This is a major advantage compared to other retrieval approaches based on colour or texture. As we will show in the next section, state of the art techniques, such as color correlogram will fail to match these two images even though they are the same building (but taken at different angle and under different weather conditions).

Fig 4 BSP Tree partitioning scenes into meaning structures

4. IMAGE RETRIEVAL RESULTS USING BSP TREES

The image comparison method was implemented in a database consists of just over 1000 photographic images. We hand picked 61 pairs of query images and embedded them in the database. Each pair consists of similar scenes or similar objects, 18 pairs of which are shown in Fig. 5.

Fig 5 A subset of query image pairs

The tree matching were done up to partition level 5. For each query image, the differences between its tree and those of the image in the database was calculated and the results were sorted in ascending order. To measure the matching performance, we calculated the matching percentile defined as

$$MP = 100 \times (N-R) / (N-1) \quad \text{if } R \leq 10 \text{ or } 0 \text{ otherwise} \qquad (3)$$

where N is the total number of images, R is the rank of the target image (paired with the query). As a comparison, we also implemented the color correlogram method [15]. Each of the 61 pairs of query images was used in turn as the query image (a total of 122 queries were performed). In the ideal case, the image paired with the query image should be returned in the first rank. The experiment showed very promising results. The average matching percentile for all 122 queries using the BSP method was 94.2 and that of color correlogram was 93.4. By analysing the results, we found that in most cases, color correlogram failed on images of very similar scenes taken at a slightly different angle; or under different lighting/weather conditions. Fig. 6 shows two more examples (plus Fig. 4) where similar scenes taken under slightly different conditions and the colour correlogram method failed to pick up the targets and the BSP tree method succeeded.

5. CONCLUDING REMARKS

In this paper, we have proposed the use of a second-generation constraint adaptive segmentation based image coding method to construct image database for efficient storage and easy access of image data. We extended the binary space partitioning tree method to colour images and developed an image matching method based the BSP tree representation. We have shown that the method achieved good coding performance at moderate bit rates and have good potential in image database applications. We believe the BSP tree provides a powerful, efficient and flexible approach to image representation. The potential in image matching is yet to be fully exploited. For example, the partition forms a series of polygon [1], which along with their spatial relations can be used to form powerful image matching methods. Our on going work includes the use of formal graph matching/tree distance approaches to fully exploit the potential of this representation scheme.

Fig. 6. Query images pairs color correlogram failed whilst the BSP tree method succeeded.

6. REFERENCES

[1] X. Wu, “Image coding by adaptive tree-structured segmentation”, IEEE Trans Information Theory, vol. 38, pp. 1755 – 1767, 1992

[2] H. Radha, et al, "Image compression using binary space partitioning tree", IEEE Trans. on Image Processing, vol.5, pp 1610-1624, 1996

[3] R.W. Picard, "Content Access for Image/Video Coding: "The Fourth Criterion"", MIT Media Lab TR No. 295. 1994

[4] W. Niblack et al, “Querying images by content using color, texture and shape”, Proc. SPIE, vol. 1908, pp. 173 – 187, 1993

[5] G. Schaefer and G. Qiu, “Midstream content access based on visual pattern coding”, Proc. SPIE, vol. 3972, pp.284 – 292, 2000

[6] E. Freg and C. S. Li, “Computing image histogram from compressed data”, Proc. SPIE, Vol. 2898, pp. 118 –124, 1996

[7] M. K. Mandal et al, “Fast Wavelet histogram techniques for image indexing”, Computer Vision and Image Understanding, vol. 75, pp. 99-110, 1999

[8] W. B. Seales et al, “Object recognition in compressed imagery”, Image and Vision Computing, 16, pp. 337-352, 1998

[9] C. Carson et al, “Blobworld: A system for region-based image indexing and retrieval”, Proc. Int. Conf. Vis. Inf. Sys. 1999

[10] M. Kunt et al, “Second generation image coding”, Proc. IEEE, vol. 73, pp. 549 –574, 1985

[11] S. Sudirman and G. Qiu, “Colour image representation using BSP Tree”, Proc. of CGIP 2000, Color in Graphics and Image Processing October 1- 4, 2000, Saint-Etienne, France

[12] S.C. Pei and C.M. Cheng, "Color image processing by using binary quaternion moment-preserving thresholding", *IEEE* Trans. on Image Processing, vol. 8, pp.614-628, 1999

[13] M. Pelillo et al "Matching hierarchical structures using association graphs", *IEEE PAMI*vol.21, pp.1105-1120, 1999

[14] H. D. Tagare et al, “Arrangement: A spatial relation between parts for evaluating similarity of tomographics sections” IEEE PAMI, vol. 17, pp. 880 – 893, 1995

[15] J. Huang, et. al., "Image indexing using color correlogram", Proc. CVPR, pp. 762-768, 1997

[16] L. Miclet, Structural Methods in Pattern Recognition, North Oxford Academic Publishers Ltd., 1986