Medical Image Retrieval by Color Content

Vincent Ng[1], David Cheung[2], Ada Fu[3]
[1]Department of Computing, Hong Kong Polytechnic University, Kowloon, Hong Kong
[2]Department of Computing Science, Hong Kong University, Hong Kong
[3]Department of Computing Science, Hong Kong Chinese University, New Territories, Hong Kong

ABSTRACT

In this paper, we develop an indexing scheme for medical images. In general, for a given medical image, there is one object which is clinically important amongst the rest. We name this object the *dominant* object. Our proposed index is composed of three parts: (1) dominant objects in images are standardized; (2) each image will have its own quadtree constructed which depends on the color composition and the color variation in the image; and (3) an R-tree that supports the retrieval of color images. To demonstrate the effectiveness of the index developed, we use images of skin lesions as the image data. Our initial experiments gives promising results for image retrieval.

1 Introduction

Traditional database management systems are unable to provide efficient retrieval for image data. In most systems, images are either unstructured or structured by using keys such as title, author or date. Recently, content-based or feature-based methods have been proposed to provide efficient retrieval of images. Image access is done through a high-level abstraction of the image properties. Example queries are "Retrieve all red circular lesions on a tanned skin" or "Find all mammograms which has a bright cluster".

In the past, many indexing methods are either based on the images directly or the features of objects within images. In 1991, Kato has developed a system to retrieve images based on sample images or sketches [1]; while Swain uses histograms to identify and retrieve images [2]. For color indexing, Lu in 1994 [3] and Niblack in 1993 [4] proposed to use the R-tree and the R*-tree correspondingly. Recently, Gong [5] suggests to use B^+-tree in which each key is made up of different image features.

In our work, we are interested to develop an indexing scheme for medical images. In general, for a given medical image, there is one object which is clinically important amongst the rest. We call this object the *dominant* object. For example, in an image of a pigmented lesion, the dominant object is a mole; while in an X-ray of a kidney, the dominant object is the largest kidney stone. Hence, to facilitate medical image queries, a global description of color composition of an image is not appropriated. It is more interesting and important to describe the color composition of a dominant object. In addition, one would also need to consider the effects of size and orientation of the dominant object during image queries. In the work of Lu and Swain, their indices are mostly based on color histograms and lacks geometric information. When images have similar dominant objects but of different sizes or orientations, the matching between them is hard to discover. However, if we use multiple features to develop an index while each index is represented by a single scaler, then it would not have sufficient discrimination power.

To remedy the above problem, we suggest a color index which is composed of:

- Dominant objects in images are standardized;
- Each image will have its own quadtree constructed which depends on the color composition and the color variation in the image;
- An R-tree that supports the retrieval of color images.

The rest of the paper is organized according to these three major parts. We will discuss the operations on dominant objects in Section 2, in which we will also describe briefly about the segmentation method

0-7803-2559-1/95 $4.00 © 1995 IEEE

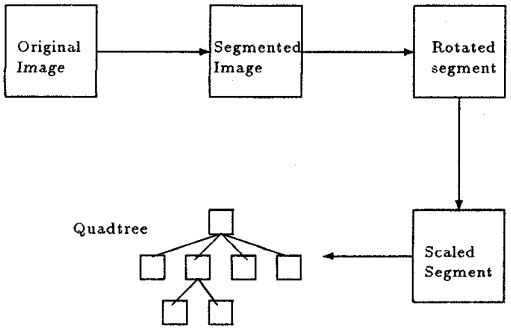

Figure 1: Index Preparation of an Image

used in this study. In Section 3, we describe the development of a quadtree associated with a given image. The similarity of two images can then be obtained by using their corresponding quadtrees. Section 4 discusses the construction of the R-tree. Section 5 presents the experimental evaluations and future works.

2 Dominant Objects

In this study, we used the images collected from a clinical study in the Pigmented Lesion Clinic in University of British Columbia, Vancouver, Canada. The clinical study is currently digitizing melanocytic lesions under a standard and controlled environment. A multistage algorithm is developed to segment these images successfully and no image failed in the segmentation process [6]. In the algorithm, weighted median filters are first used to suppress noise as well as to preserve fine details of the images. In the second part, a multichannel segmentation technique with a rule-based system is then used to find out the lesions of interest.

As shown in Figure 1, each image is processed in several steps and to be set into a standardized format. It is intended to support the comparisons of different objects even when they are of different sizes and orientations. A single object may be digitized to different images because of external factors such as lighting conditions, focal distances of cameras, and digitizing orientations. By using the segmentation algorithm in [6], the image is initially segmented into two regions: the background region and a dominant object. We then find out the major and minor axes of the object. To standardize the orientation, we rotate the image until the major axis of the dominant object is horizontal. A minimum bounding rectangle of the object is then calculated. In order to achieve

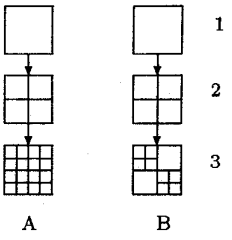

Figure 2: Quadtree Decomposition

standardized image sizes, we scale the image region enclosed by the bounding rectangle. The same factor is used both for the horizontal and the vertical directions so as to avoid any distortion of the original image. The final image is called the *object image*.

For any given color image, if the RGB color space is used and in which each primary color has N intensity levels, there will be N^3 possible distinct colors. To reduce the number of colors into a manageable set, a smaller value of N should be used. This can be done by grouping adjacent intensity levels together. In our study, each color image has a 24-bit color information. That is, there are 256 intensity levels ($N = 256$) for each primary color in an original image. We therefore divide the levels into 16 ranges evenly. Hence, $N = 16$ and the number of distinct colors becomes 4096. However, in most medical images, the color distribution is sparse and many colors will not appear. For the images of skin lesions, a color is included in the index when it occupies at least 1% of an object image and there are 365 of them.

3 Quadtree Construction

Once we have identified the dominant objects of the images, the second part is to develop a similarity measurement between them. We adopt the use color histograms which are effective in characterizing the color of an image [2, 4]. A color histogram of an image represents the percentages of different discrete colors appeared in the image. In [2], it has been reported that a single color histogram does not have sufficient discrimination power. This is improved by using a multi-level color histogram [3]. In the paper, an image is divided recursively to form a quadtree and each quadrant in the tree is associated with a color histogram. However, this approach divides the image uniformly without considering the different

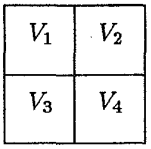

Figure 3: Merging of Color Variations

color distributions in different regions. For example, in Figure 2.A, regions are divided evenly with 1 quadrant at level 1, 4 quadrants at level 2, and 16 quadrants at level 3. We propose to divide a region according to its level of color variation, and to merge smaller regions when there is not sufficient color variation. In Figure 2.B, level 1 has one quadrant, level 2 has 4 quadrants and level 3 has 8 quadrants. The objective is to maintain a high discrimination power but a better storage utilization.

Intuitively, there are two factors which contribute to the color variation of an image. The first factor is simply the **number of colors**. If an image has a single color, there is no variation. If an image of size N has N discrete colors, there is a large color variation. The second factor is related to the **uniformity** of the pixels of the same color. If pixels of color C are adjacent to each other, the image would have a better uniformity with respect to C; otherwise when the pixels are sparsely located, the image is less uniform with respect to C.

Let N be the maximum number of possible distinct colors and the colors are represented as $C_1, C_2, \ldots, C_N$. For a given quadrant Q of an object image, we can approximate the first factor by the *color variation*, vc_Q, which is defined as:

$$vc_Q = n/min(S, N) \tag{1}$$

where n is the number of distinct colors in Q and S is the number of pixels in the quadrant. vc_Q will be close to 1 when there are many colors and will be close to 0 when there are only a few colors in Q. Associated with Q, there is a color vector, V_Q, of length N. It is used to support the calculations of vc's of larger quadrants. The i^{th}-bit in V_Q represents the existence of color C_i in Q. In Figure 3, we have calculated vc_{Q1}, vc_{Q2}, vc_{Q3}, vc_{Q4} and their corresponding V's of the four smaller quadrants. With these color vectors, we can then find vc_Q as:

$$v_Q = \| V_{Q1} \mid V_{Q2} \mid V_{Q3} \mid V_{Q4} \| /min(S, N) \tag{2}$$

Co-occurrence matrices are often used to compute the second-order statistics or the texture information

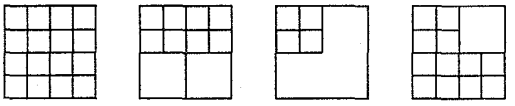

Figure 4: Uniformity of a Grid.

of a gray-level image [7]. The uniformity, u, of a region is obtained by:

$$u = \sum_{i=1}^{I} \sum_{j=1}^{I} P(i, j, d, \theta)^2 \tag{3}$$

where $P(i, j, d, \theta)$ is the probability of a pair of pixels having intensity values of i and j and they are separated by a distance d and in direction θ, and I is the number of intensity levels. As an example, in Figure 4, there are four bi-level images with their uniformities shown[1]. When comparing two regions, a more homogeneous region would have a higher value of u.

We adopt a similar approach to define the uniformity of a color in a given quadrant Q of an object image. A bi-level image, B_i, is created for an existing color C_i in Q. A pixel, p_B, in B_i equals to 1 if the corresponding pixel in Q has color C_i; otherwise p_B is 0. The uniformity of C_i in Q, u_i, is then obtained by using Equation 3. The calculations are repeated for all colors and the overall uniformity, vu, of Q is

$$vu_Q = (\sum_{1}^{n} 1 - u_i)/n. \tag{4}$$

vu_Q will be close to 1 when Q is highly non-uniform and will be 0 when Q has only 1 color.

We use a bottom-up approach to construct the quadtree. An object image is divided into quadrants recursively. The smallest quadrants will become the leaf nodes of the quadtree. The two variations, vc and vu, and a color histogram are calculated for each node. Each internal node of the quadtree is created by merging four adjacent nodes at a lower level. During the merge, the vc's and vu's of the four nodes are compared with two thresholds, t_c and t_u. If all the variations are smaller than their corresponding thresholds, the nodes are removed from the tree and

[1] $d = 1, \theta = 0$.

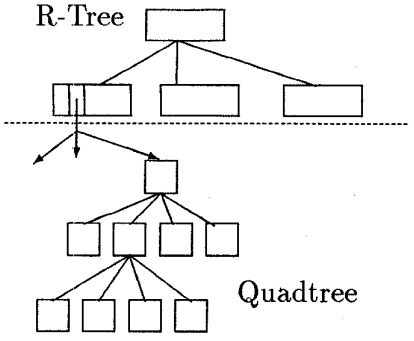

Figure 5: A Two Level Index

the complete region is represented by the new internal node. The vc, vu and the color histogram of the new internal node are then calculated. The procedures are repeated for all nodes in the quadtree until the root of the tree is created. At the end, each object image will have a quadtree to represent its color composition. After all the quadtrees of the images have been created, images can be compared by using similarity values which are obtained by using the intersections of the color histograms as described in [2].

4 R-Tree Construction

We use a two-level index as shown in Figure 5 for the image database. Layered on the top of the quadtrees, we use an R-tree [8] to index the color histograms. During the construction of the R-tree, we realize a dimensionality problem. Even if not all discrete colors in the RGB color space are appeared in an object image, a color histogram has a high dimensionality. In our image database of lesions, there are 365 bins in a color histogram. Indexing on such a high-dimensionality is computational intensive and even infeasible. To solve the problem, one approach is to use a few dominant colors instead of using all colors [3]. For many images, this may not be possible since there may not be any single color that occupies a sufficient portion of an image. A second approach is to use average color indices [9]. However, its formation does not depend on the data and can result with poor matching performance.

As updating of the image database is infrequent, we decide to build an R-tree by considering the colors used in the histograms. The idea is to merge colors into color zones such that each color zone will have similar usage frequency. Suppose there are n discrete

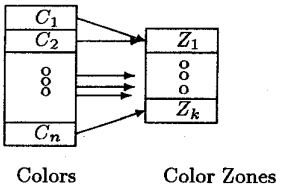

Figure 6: Mapping of colors into color zones

colors, $C_1, C_2, \ldots, C_n$ amongst all the histograms. A color C_i can be defined by its red, green and blue components as (R_i, G_i, B_i). The usage of C_i, f_i, can be approximated by the number of non-empty bins of C_i in the histograms. A large value of f_i suggests that C_i would have better discrimination power. After obtaining all the f_i's, we sort the n colors with respect to their (R, G, B) values. Accumulative values of f_i's are then computed along with the sorted list and are represented as a_i's. If k-dimensional points are to be used in the R-tree, the a_i's are used to divide the colors into k zones and a conversion table is built as showed in Figure 6. A color histogram with n colors can then be mapped to a k-dimensional point and the value of a color zone is obtained by summing up the percentages of the corresponding colors. One extra dimension, r_s, which is the ratio between the major and minor axes is included as another dimension for each entry in the R-tree. This simple shape attribute allows us to separate circular objects from elliptical objects when their quadtrees have similar color histograms.

To perform an image query, a user provides a query image and a tolerance value. The query image is segmented, its dominant object will be found, and the object image is obtained. Together with the tolerance value and the color histograms of the resultant object image, it forms a search window for the R-tree. The search results are the root nodes of quadtrees in the image database. Similarities between the quadtrees and the quadtree of the query image are calculated. For those values within the tolerance, their corresponding images are returned to the user.

5 Experiments

We used images of skin lesions as the image data. The images are collected from a clinical study in Canada. At the time of writing, there have been 126 images digitized. Each image has a background

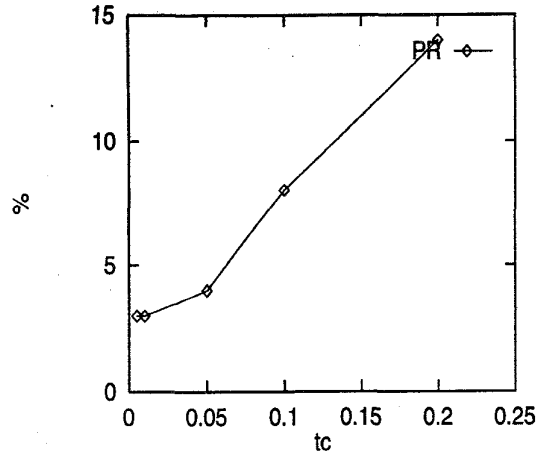

Figure 7: Average percentages of retrieved images versus t_c with a fixed t_u.

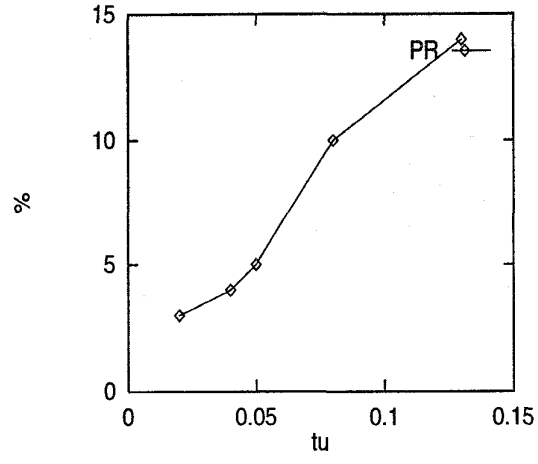

Figure 8: Average percentages of retrieved images versus t_u with a fixed t_c.

area and at least one lesion. To simplify our work, we have excluded images containing hairs or multiple lesions. This has left us with 95 images as a basic set. From the basic set, we created two additional set of images. Images in the first new set are created by randomly re-distributing 5% of the pixels in the dominant objects of the original images. The second set is done similarly by re-distributing 20% of the pixels. Therefore, the image database has a total of 285 images.

The two-level index has been implemented in C under the Unix operating system. Two performance metrics are adopted. The first one is the percentage of retrieval (PR)[2]. The second one is the ratio between the average number of nodes in the quadtrees and the maximum possible number of nodes in a quadtree tree with the same number of levels (AN). Hence, a small PR value is desirable for an accurate retrieval. In our experiments, quadtrees of at most 4 levels are created. Therefore each quadtree can have up to 85 nodes and small values of AN indicate better storage utilizations.

To perform the experiments, we randomly select 10 images as the query images from the database. Experiments are done in a dedicated Sun SparcStation in order to minimize the deviations due to the operating environment. In the experiment, quadtrees of the query images are first created and and the average query result of the 10 images is reported. In the first set of experiments, we chose 5 different threshold values of t_c's while setting t_u to 0. In Figure 7, we observed that when the threshold is getting smaller, as expected, the accuracy of retrieval improved. That is, a smaller number of images are matched with the query image. However, the tradeoff is to use more storage space as shown in Figure 9. If the matching requirement was relaxed with a few percentages, the saving of storage space for the quadtrees is substantial (25%). A second set of experiments were done similarly, but 5 different threshold values of t_u's are used while t_c is set to 0. We observed a similar result from Figure 8 and Figure 10. These two experiments have demonstrated that relatively good retrieval performance can be achieved with less storage requirement.

The two-level index presented in this paper produces some interesting results; however, some areas need to be investigated further. It would be more efficient if the calculation of the uniformity of a region can be obtained from the uniformities of the subregions. In our image queries, color variations and uniformities are not used in the R-tree. It is believed that including them in the entries of the R-tree will improve the search performance.

[2] PR = (no. of retrieved images)/(total no. of images)*100.

References

[1] T. Kato, T. Kurita and H. Shimogaki. "Intelligent Visual Interaction with Image Database Systems- Toward the Multimedia Personal Interface". *Journal of Information Processing of Japan*, Vol. 14, No. 2, pp. 134-143, 1991.

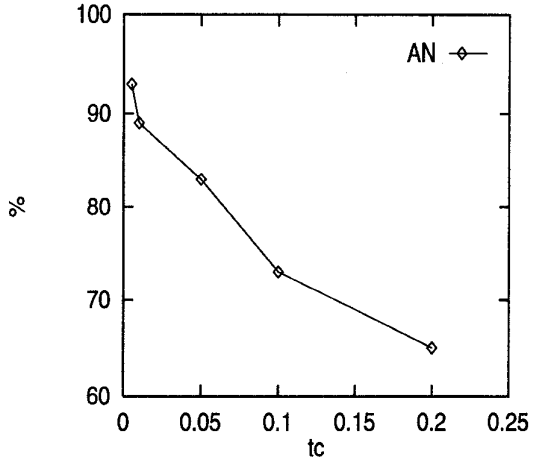

Figure 9: Average tree sizes (in percentages) versus t_c with a fixed t_u.

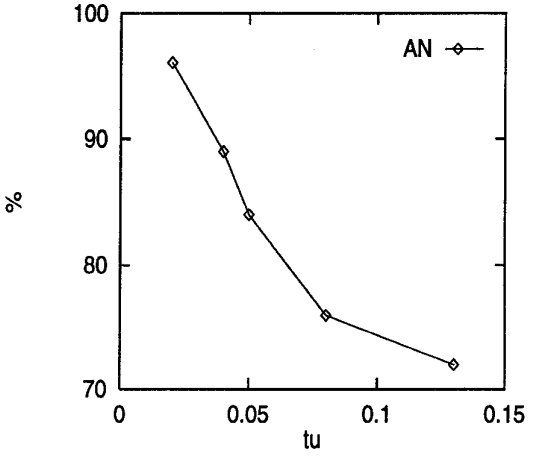

Figure 10: Average tree sizes (in percentages) versus t_u with a fixed t_c.

[2] M.J. Swain and D.H. Ballard. "Color indexing". *International Journal of Computer Vision*, Vol. 7, No. 1, pp. 11-32, 1991.

[3] H. Lu, B.C. Ooi, KL Tan. "Efficient Image retrieval By Color Contents". *Proc. of Applications of Databases, First International Conference*, June 1994, pp. 95-108.

[4] W. Niblack et al. "The QBIC project: Querying Images by content using color, texture, and shape." *In SPIE 1908, Storage and Retrieval for Image and Video Databases*, Feb. 1993.

[5] Y. Gong, H. Zhang, H.C. Chuan, M. Sakauchi. "An Image Database System with Content Capturing and Fast Image Indexing Abilities", *Proc. of International Conference on Multimedia Computing and Systems*, Los Alamitos, U.S.A., 1994, pp. 121-130.

[6] T. Lee, V. Ng, D. McLean, A. Coldman, R. Gallagher, J. Sale. "A Multi-Stage Segmentation Method for Images of Skin Lesions", *Proc. of IEEE Pacific Rim Conference on Communications, Computers, and Signal Processing*, May 17-19, 1995, Victoria, Canada.

[7] R. Nevatia. *Machine Perception*, Prentice Hall, 1982.

[8] A. Guttman. "R-Trees: A Dynamic Index Structure for Spatial Searching", *Proc. of SIGMOD*, 1984, pp. 47-57.

[9] H.S. Sawhney and J.L. Hafner. "Efficient Color Histogram Indexing", *Proc. of IEEE International Conference on Image Processing*, November 1994.