the frame. Once text is detected, the tracking process is applied to find the temporal correspondence in consecutive frames. Fig. 11 shows a tracking result for the movie *Star Wars*. There are 2600 frames in the sequence, which includes static, zooming, and scrolling text. Fig. 12 shows tracking results for a transverse text line. We detect it as horizontal scrolling by analyzing the motion status and we thus divide the line into words and track them.

The processing time changes considerably with the number of text lines per frame. Tracking movie credits takes more time than other video types since there are more text lines per frame in movie credits. For example, for the movie *Star Wars*, it takes about 1 s to track one frame (the average number of text lines in a frame is five), while it takes only 0.17 s to track text in a football game (only one text line is moving in all of the frames). But as we indicated above, we can detect the text as well as the temporal correspondences of the text blocks.

There are several limitations to our system. First, text tracking is started only when text is detected. If the text detection module fails, the system will miss the text. Second, our tracker uses SSD-based image matching to approximate the position and then uses the text contour to refine the position. Therefore, the system can only be used to track text. In addition, since we use speed prediction to predict the position of the text, the text's acceleration is limited. The tracker has difficulties when text moves too abruptly or keeps moving on a complex background. This happens especially in sports video. For example, when tracking the name of an athlete on a jersey, the text may occlude quickly because of the athlete's jumping and rotating.

V. Conclusion

We have presented a system for detecting and tracking text in digital video automatically. A hybrid wavelet/neural network based method is used to detect text regions. The tracking module uses SSD-based image matching to find an initial position, followed by contour-based stabilization to refine the matched position. The system can detect graphical text and scene text with different font sizes and can track text that undergoes complex motions. Our next focus will be on making use of detected and tracked text to build a text-based video indexing and retrieval system.

References

[1] M. Davis, "Media streams: Representing video for retrieval and repurposing," in *Proc. ACM Multimedia'94*, pp. 478–479.

[2] W. Li, S. Gauch, J. Gauch, and K. M. Pua, "VISION: A digital video library," in *DL'96: Proc.1st ACM Int. Conf. Digital Libraries: Multimedia Digital Libraries*, 1996, pp. 19–27.

[3] J. Hernando, "Voice signal processing and representation techniques for speech recognition in noisy environments," *Signal Process.*, vol. 36, p. 393, 1994.

[4] A. K. Jain and B. Yu, "Automatic text location in images and video frames," *Proc. ICPR*, pp. 1497–1499, 1998.

[5] S. K. Kim, D. W. Kim, and H. J. Kim, "A recognition of vehicle license plate using a genetic algorithm based segmentation," *Proc. ICIP: Nature*, pp. 661–664, 1996.

[6] R. Lienhart and F. Stuber, "Automatic text recognition in digital videos," in *Proc. ACM Multimedia Conf.* , 1996, pp. 11–20.

[7] J. Shim, C. Dorai, and R. Bolle, "Automatic text extraction from video for content-based annotation and retrieval," in *Proc. ICPR*, 1998, pp. 618–620.

[8] J. Zhou and D. Lopresti, "Extracting text from WWW images," in *Proc. ICDAR*, 1997, pp. 248–252.

[9] V. Wu, R. Manmatha, and E. M. Riseman, "Finding text in images," in *DL'97: Proc.2nd ACM Int. Conf. Digital Libraries, Images and Multimedia*, 1997, pp. 3–12.

[10] A. K. Jain and S. Bhattacharjee, "Text segmentation using Gabor filters for automatic document processing," *Mach. Vis. Appl.*, vol. 5, pp. 169–184, 1992.

[11] K. Etemad, D. S. Doermann, and R. Chellappa, "Multiscale document page segmentation using soft decision integration," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 19, pp. 92–96, 1997.

[12] R. Chellappa, B. S. Manjunath, and T. Simchony, "Texture segmentation with neural networks," in *Neural Networks and Signal Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1992, pp. 37–61.

[13] S. K. Rogers, J. M. Colombi, C. E. Martin, and J. C. Gainey, "Neural networks for automatic target recognition," *Neural Networks*, vol. 8, pp. 1153–1184, 1995.

[14] H. Li, D. S. Doermann, and O. Kia, "Automatic text extraction and tracking in digital video," Univ. Maryland, College Park, Tech. Reps. LAMP-TR-028, CAR-TR-900, 1998.

[15] K. Sung and T. Poggio, "Example-based learning for view-based human face detection," Mass. Inst. Technol., Cambridge, MA, A.I. Memo 1521, CBCL Paper 112 , 1994.

[16] G. D. Hager and P. N. Belhumeur, "Efficient region tracking with parametric models of geometry and illumination," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 20, pp. 1025–1039, 1998.

[17] J. Shi and C. Tomasi, "Good features to track," in *Proc. CVPR*, 1994, pp. 593–600.

[18] V. Kobla, D. S. Doermann, and K. I. Lin, "Archiving, indexing, and retrieval of video in the compressed domain," in *Proc. SPIE Conf. Multimedia Storage and Archiving Systems*, 1996, pp. 78–89.

Hierarchical Color Image Region Segmentation for Content-Based Image Retrieval System

Chiou-Shann Fuh, Shun-Wen Cho, and Kai Essig

***Abstract*—In this work, we propose a model of a content-based image retrieval system by using the new idea of combining a color segmentation with relationship trees and a corresponding tree-matching method. We retain the hierarchical relationship of the regions in an image during segmentation. Using the information of the relationships and features of the regions, we can represent the desired objects in images more accurately. In retrieval, we compare not only region features but also region relationships.**

***Index Terms*—Color, feature extraction, hierarchical relationships, region extraction, region merging.**

I. Introduction

A content-based image/video retrieval system is a querying system that uses content as a key for the retrieval process [1]. It is a difficult task to design an automatic retrieval system, because real-world images usually contain very complex objects and color information. One problem that occurs is how to segment a real world image perfectly.

Manuscript received November 28, 1998; revised July 27, 1999. This research was supported by the National Science Council of Taiwan, R.O.C., under Grants NSC 88-2213-E-002-031 and NSC 86-2212-E-002-025, by Mechanical Industry Research Laboratories, Industrial Technology Research Institute, under Grant MIRL 873K67BN3, by the EeRise Corporation, Tekom Technologies, and by Ulead Systems. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Thomas S. Huang.

The authors are with the Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan, R.O.C. (e-mail: fuh@csie.ntu.edu.tw; gavin@robot.csie.ntu.edu.tw; kessig@robot.csie.ntu.edu.tw).

Publisher Item Identifier S 1057-7149(00)00179-2.

Various research has been done in extracting color and spatial information from images. In recent work [7] the image is segmented into regions of constant (but unknown) reflectance to avoid unreliable results in the vicinity of an edge. The computed ratio of the reflectance of a region to that of its background is used for object recognition. Another approach [4] derives illumination invariants of color distributions in spatially filtered color images. The combination of this information can be used to recognize a wide range of spatial patterns in color images under unknown illumination conditions. Syeda-Mahmood [9] presents a method of color specification in terms of perceptual color categories that allow a reliable extraction of color regions and their subsequent use in selection. An efficient color segmentation algorithm that combines the advantages of a global region-based approach with the advantages of a local edge-based approach is presented in [3].

This work attempts to propose the new idea of combining a color segmentation method that can retain the hierarchical relationships of the regions in an image with a suitable tree-matching method that uses the relationships and the features of the regions to build an efficient content-based image retrieval system for real-world images. We can describe complex real-world images by decomposing the objects into some regions. These regions may have some relationships among them, such as overlap, relative position, and so on. By these region relationships, we can identify surrounding or surrounded regions; hence, we also know which regions consist of a simple object. Therefore, we created a color segmentation method to segment images and retain region relations at the same time in a relationship tree. In the retrieval process, these relationships and the extracted shape and color features of the regions themselves can help us to retrieve the more desired image by matching the representations of the objects in the database to the objects in the query image. For demonstration, we build a system that can retrieve some simple objects.

II. Region Extraction

The region extraction process consists of three phases: segmenting an image into regions, merging regions, and extracting features from regions.

The existing segmentation techniques are not suitable for our purpose because of two reasons. First, they are based only on color information alone. They usually produce disconnected segments, which we do not want. Second, in complex images, selecting thresholds is almost impossible.

Region-oriented segmentation techniques use not only color information but also the pixel relationships to partition an image into some regions, which are usually continuous [6], [10]. Hence our hierarchical region segmentation bases on region growing segmentation.

A. Region and Subregion Definitions

Before stating the process of our segmentation, we must define the region first.

Definition 1—Region: Let R represent the entire image. We may consider the segmentation as a process that partitions R into n regions, $R_1, R_2, \cdots, R_n$, such that the segmentation is complete, the pixels in a region are connected and that the regions must be disjoint. Additionally two regions must be different in the sense of a predicate P.

Definition 2—Subregion: We say that a region R' is a subregion of region R if there exists a close pixel sequence $S = \langle (r_0, c_0), (r_1, c_1), (r_2, c_2), \cdots, (r_{k-1}, c_{k-1}), (r_k, c_k) \rangle$, where $(r_i, c_i) \in R$ and each pair of successive pixels in sequence are neighbors, including (r_0, c_0) and (r_k, c_k), such that every pixel $(r', c') \in R'$ is surrounded by the sequence S.

This subregion definition identifies the relationship of two regions. Every region in an image must be a subregion of another region. In order to comply with this, we need a pseudoregion that represents the entire image. Hence, the first level regions that are not subregions of any real regions are collected to be subregions of the pseudoregion. The subregions of a region are siblings. These relationships and the features of every region are sufficient to represent an image.

B. Color Space and Color Distance

We use the original RGB model of the images, and the color distance is the Euclidean distance in RGB color space. The RGB color distance function is described as the following:

$$d_C(\boldsymbol{v}, \boldsymbol{v}') = \sqrt{(r - r')^2 + (g - g')^2 + (b - b')^2} \tag{1}$$

where

$$\boldsymbol{v} = (r, g, b), \qquad \boldsymbol{v}' = (r', g', b')$$

where d_C denotes the color distance function. The $\boldsymbol{v}$ and $\boldsymbol{v}'$ denotes the two color value vectors in RGB color space. The color distance between two pixels can be represented as

$$d_C(C\boldsymbol{p}, C\boldsymbol{p}') = \sqrt{(r_{\boldsymbol{p}} - r_{\boldsymbol{p}'})^2 + (g_{\boldsymbol{p}} - g_{\boldsymbol{p}'})^2 + (b_{\boldsymbol{p}} - b_{\boldsymbol{p}'})^2} \tag{2}$$

where

$$C(\boldsymbol{p}) = (r_{\boldsymbol{p}}, g_{\boldsymbol{p}}, b_{\boldsymbol{p}}), \qquad \boldsymbol{p} = (i, j)$$

where C denotes a function mapping from a pixel $\boldsymbol{p} = (i, j)$ in image plane to its color value $(r_{\boldsymbol{p}}, g_{\boldsymbol{p}}, b_{\boldsymbol{p}})$ in RGB color space. Hence, the $C(\boldsymbol{p})$ and $C(\boldsymbol{p}')$ represent the RGB color values at pixel $\boldsymbol{p} = (i, j)$ and $\boldsymbol{p}' = (i', j')$, respectively.

C. Region Growing

We focus to segment the regions of the obvious objects from an image and treat the remainder as regions of the background or other obscure objects. We want the segmented regions to grow as large as possible. Hence, we use the following two criteria to limit the growth.

Local criterion:

$$d_C(C\boldsymbol{p}, C\boldsymbol{p}') < T_L \tag{3}$$

and

Global criterion:

$$d_C(C_R(R\boldsymbol{p}), C\boldsymbol{p}') < T_G \tag{4}$$

where

$$\boldsymbol{p} = (i, j), \quad \boldsymbol{p}' = (i + l, j + k), \quad l = -1, 0, 1, \quad k = -1, 0, 1$$

$$C_R(R) = \left(\frac{\sum_{\boldsymbol{i} \in R} r_{\boldsymbol{i}}}{N}, \frac{\sum_{\boldsymbol{i} \in R} g_{\boldsymbol{i}}}{N}, \frac{\sum_{\boldsymbol{i} \in R} b_{\boldsymbol{i}}}{N} \right),$$

N is the number of pixels in region R

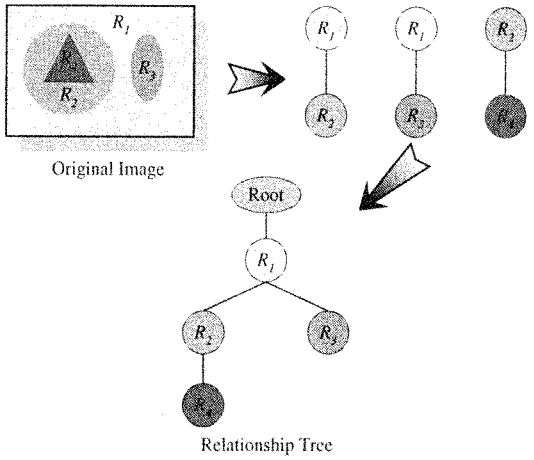


Fig. 1. Creating relationship tree.

where $R(\boldsymbol{p})$ denotes the region which pixel $\boldsymbol{p}$ belongs to and C_R denotes the average color value function for a region which maps from region R to its average color value vector. Besides, the pixel $\boldsymbol{p}'$ is a neighbor of the pixel $\boldsymbol{p}$. Equation (3), the *local criterion,* means that the color distance between pixels $\boldsymbol{p}$ and $\boldsymbol{p}'$ must be less than the *local criterion threshold* T_L. Equation (4), the *global criterion,* means that the color distance between the pixel $\boldsymbol{p}'$ and the average color value of the region which pixel $\boldsymbol{p}$ belongs to must be less than the *global criterion threshold* T_G.

The segmented regions grow one by one in region growing segmentation. For convenience, the pixels in region and neighboring (eight-neighbors) with unclassified pixels are called *growing pixels*. The growth process must run once for each of the growing pixels, and the region expands from these growing pixels according to the local and global criteria. Besides, there must exist a *seed growing pixel* for every region, otherwise the growth process can not proceed. We define the order of the pixels in entire image as an increasing order from left to right and top to bottom. The seed growing pixel of the next growing region is the unclassified pixel with the smallest order. The unclassified pixels [$\boldsymbol{p}'$ in (3) and (4)] neighboring the growing pixels [$\boldsymbol{p}$ in (3) and (4)] in a growing region are classified into the growing region when the local and global criteria are both satisfied. The region grows until no unclassified pixels satisfy the two criteria. After all pixels in the entire image are classified into the proper regions, the growth process terminates.

D. Creating Region Relationships

With the subregion definition in Section II-A the relationships of all regions in an image can be created. For example, if region R_2 and R_3 are identified as subregions of region R_1 by subregion definition, and region R_4 is a subregion of region R_2, then we can create a relationship tree as shown in Fig. 1. Each node in the relationship tree represents a region in the original image. The regions which are not surrounded by any other regions, respectively, are considered as subregions of the entire image itself. For generality, we design a pseudoregion that represents the entire image itself, the node of this pseudoregion is the root node of the relationship tree. All nodes of regions that are not subregions of any other regions are children nodes of the root node.

The relationship tree is created during region segmentation. The root node is created before segmentation. When a region growth completes, the node of this region is added into the relationship tree. The position the added node is placed is determined by the subregion relationship identification. We must mention that it is not necessary to identify each pair of regions in an image for creating the relationship tree. The subregion relationship identification just proceeds at the time when a node is added. The added node is placed under the lowest level node of the region which can surround the region of the added node. Because the seed growth pixel of each region is the pixel with the smallest order at the time, the subregions of a region must grow later than the surrounding region and the relationship tree created by subregion identification is correct.

In order to create a unique relationship tree structure for the same object in different images, we sort the subregions under each region by their areas in descending order.

III. Merging and Eliminating Regions

A. The Merging Process

Regions with an area less than the area threshold are considered as nonsignificant regions. We eliminate a nonsignificant small region by merging it into a region adjacent to this small region which is most similar in color. Besides, in order to retain the correctness of the relationship tree, the tree must be corrected when two regions are merged. The merging process proceeds during region segmentation. After a region completes its growth and is placed in its proper position in the relationship tree, merging process calculates all color distances between this added region and the regions of its parent or sibling region node, respectively. If the added region is placed in the first level of the relationship tree, merging process needs only to calculate the color distances between the added region and the region of its sibling region node. We proceed the merging process after every region is generated to reduce the number of regions. Therefore, our result slightly differs from the result produced by merging after all regions are generated. This is not important because no segmentation result is absolutely correct, and our matching algorithm can tolerate this.

The color distance function between two regions are written as

$$d_C(C_R(R),\, C_R(R')). \tag{5}$$

The *similarity criterion* to judge whether two regions are similar in color is

$$d_C(C_R(R),\, C_R(R')) < T_S. \tag{6}$$

This criterion means that two regions are similar in color if their color distance is less than the threshold T_S. The *nonsignificant criterion* to judge whether a region is nonsignificant is

$$A(R) < T_N \tag{7}$$

where A denotes the area function for region. The criterion means that the region R is a nonsignificant region if its area is less than the area threshold T_N. If the nonsignificant criterion is satisfied, this region is merged to the region with the closest color distance; but the merging process does not consider whether the similarity criterion satisfies in this situation. It must be noticed that the relation of the similar and nonsignificant criteria is an "or" relation. If both two criteria are not satisfied, the added region is simply located into the relationship tree in the usual way.

All region features of the merged region are recalculated from the features of the two original regions. Each region's contribution to these features is weighted according to its relative size. After two regions are merged together, it may result in that some regions are surrounded

by the merged region. Hence, the relationship tree must be corrected in response to merging. It is obvious that the affected part of the tree structure after merging is just the subtree under the node of the region surrounding both merging regions. The root of the subtree is the parent node of the added region node. The other parts of the relationship tree are not affected. If the region being merged to is the root node of the subtree, the subtree remains correct. If the region being merged to is the sibling node of the added region node, merging process must find the region nodes whose regions are surrounded by the resulted merged region from the other sibling nodes. Subsequently, all subtrees led by the found nodes are moved to the position under the merged region node. Moreover, the children nodes of the merged region node must also be resorted. A correction of the tree structure after merging of two regions is shown in Fig. 2. It is noticed that the first generated region in the first level nodes of the relationship tree may not be merged with any other adjacent regions. The reason is that the merging proceeds when every region completes its growth and the other adjacent regions are not generated at the time when the first region is generated. This problem is solved by proceeding the merging for the first level nodes at the end of the segmentation algorithm.

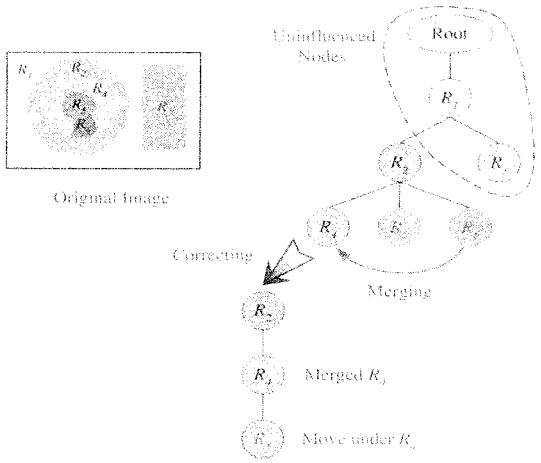


Fig. 2. Correcting the tree after merging.Region R_6 is added into the tree, and merged into region R_4. This results in that region R_5 is surrounded by merged region R_4. The tree must be corrected accordingly.

B. Threshold Selection

We select the local, global, and similarity thresholds, T_L, T_G, and T_S, in growing process adequately to prevent an iteration of the time-consuming merging process until all regions in an image cannot be merged further. This method does not solve the problem perfectly, but it is a reasonable compromise between the result and the performance.

The *local criterion threshold* T_L cannot be selected too large to segment two regions. However, too small T_L produces too detailed segmentation. The proper T_L value is the color distance between two color values differing about twenty color levels in all R, G, and B axes.

The *global criterion threshold* T_G can be selected larger to make the segmented regions as large as possible, but it also can not be too large to segment two obvious adjacent regions. The proper T_G value is the distance between two color values differing about fifty color levels in all R, G, and B axes.

The *similarity criterion threshold* T_S is selected similar to T_L, because we hope the adjacent regions can be discriminated as pixels. Besides, in order to solve the iterative merging problem mentioned above, the value of the T_G must be larger than T_S.

The last threshold is the *nonsignificant criterion threshold* T_N. If the application concerns more details in an image, we select smaller T_N. Otherwise, the T_N can be larger.

All threshold selection also depends on the application.

C. Extracting Features from Regions

After representing the region structures of the image as a relationship tree, we must extract the features of regions from the original image to improve the quality of the description of the final relationship tree. The selected features must be insensitive to the three transformations, even invariant to them.

The *means* and *standard deviations* of the R, G, and B values of the pixels in a region are sufficient to represent the color attributes of a region.

Table I shows the shape features we select: the *thinness ratio* T, the *density ratio* D, and the *invariant moment* ϕ, derived from the *normalized central moments* [2]. These features are sufficient to represent the regions of a simple object.

TABLE I
SHAPE FEATURES OF REGION R

Equations for the selected features: 1 2 3 4
$T = 4\pi \frac{N}{\|P\|^2}$
$D = \frac{number\ of\ pixels}{number\ of\ pixels\ including\ sub\text{-}regions}$
$\phi = \frac{1}{N^2}\left[\sum_{\mathbf{p}=(\mathbf{r},\mathbf{c})\in\mathbf{R}} (r-\bar{r})^2 + \sum_{\mathbf{p}=(\mathbf{r},\mathbf{c})\in\mathbf{R}} (c-\bar{c})^2\right]$

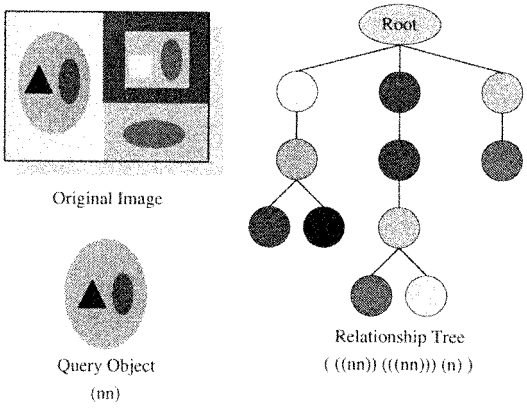


Fig. 3. Relationship tree matching.

IV. IMAGE RETRIEVAL

The image retrieval process itself is a matching process that matches the query data with the data in database. Because of the hierarchical region segmentation, the retrieval process must match not only the features, but also the relationship structure of the regions in query object when querying a simple object. When querying a simple object by our approach, the relationship matching process needs only to match the subtree led by the region node of the outermost region of the query object with each subtree in all database images.

A. Matching Region Relationships

Every tree can be represented as a string. We represent a leaf node as an "n" character, and a branch node as a "(" character. Besides, in

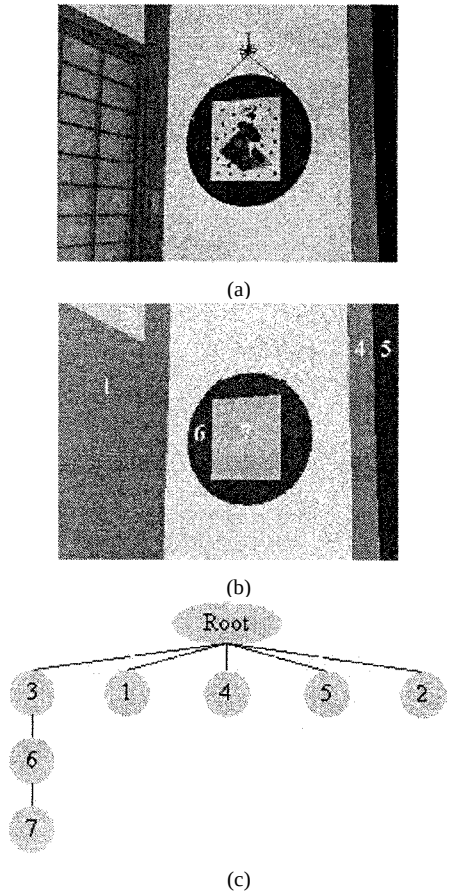
TABLE II
THRESHOLD VALUES

Threshold	Value
Local criterion threshold T_L	20
Global criterion threshold T_G	50
Similarity criterion threshold T_S	20
Non-significant criterion threshold T_N	1000

TABLE III
PERFORMANCE OF THE SEGMENTATION

Item	Second
Maximum processing time on an image [5]	2.333
Minimum processing time on an image [6]	0.841
Average processing time on an image	1.265
Total processing time on 200 images	253.0230

[1] N: The number of the pixels in R.
[2] P: R's perimeter, $P = \{(r_k, c_k) \in R | \{\text{all 4-neighbors of } (r_k, c_k)\} - R \neq \emptyset\}$.
[3] $\mathbf{p}$: A pixel in R, its coordinate in image plane is (r, c).
[4] $(\bar{r}, \bar{c})$: Centroid of R.
[5] When processing Image086.
[6] When processing Image002.
[7] *: denotes similar images



(a)

(b)

(c)

Fig. 4. Segmentation result and relationship tree example of wall decoration image: (a) original image, (b) segmentation result, (c) relationship tree.

order to represent a subtree led by a branch node, we insert a ")" character after the substring of a subtree. It is noticed that the ")" character does not represent any region node. Hence, the relationship tree of an image in Fig. 3 can be represented as a string "(((nn))(((nn)))(n))." It is obvious that when querying by an object the matching process is only to find the substrings which are the same as the string of the object from the string of each image in database; a common function in C run-time library. The disadvantage of this method is that the tree structures must be the same in the query object image and database images. Otherwise, the strings of the query image and database images cannot be matched. However, this method is the simplest and fastest method for relationship tree matching.

B. Matching Regions

The color and shape features in our approach are represented as a nine value feature vector $\langle \mu_{\text{red}}, \mu_{\text{green}}, \mu_{\text{blue}}, \sigma_{\text{red}}, \sigma_{\text{green}}, \sigma_{\text{blue}}, T, D, \phi \rangle$. The smaller the dissimilarity score between two feature vectors, the more similar the two regions. The dissimilarity score functions of two regions R and R' are described as the following:

$$S_{\text{color}}(R, R') = \sum_{\mu = \mu_{\text{red}}, \mu_{\text{green}}, \mu_{\text{blue}}} \left[\mu(R) - \mu(R')\right]^2 + \sum_{\sigma = \sigma_{\text{red}}, \sigma_{\text{green}}, \sigma_{\text{blue}}} \left[\sigma(R) - \sigma(R')\right]^2 \quad (8)$$

$$S_{\text{shape}}(R, R') = [T(R) - T(R')]^2 + [D(R) - D(R')]^2 + [\phi(R) - \phi(R')]^2 \quad (9)$$

$$S_{\text{region}}(R, R') = \sqrt{S_{\text{color}}(R, R') + S_{\text{shape}}(R, R')} \quad (10)$$

where $\mu(R)$ is the mean and $\sigma(R)$ is the standard deviation of the region R. The dissimilarity score is the Euclidean distance between the two feature vectors. Equation (8) calculates the color and (9) the shape dissimilarity between two regions. Equation (10) combines these two measures to calculate the dissimilarity score (or distance) of the two regions.

The matching process is to find the matching substrings from the strings of all images in database and to calculate the dissimilarity score between every corresponding region pair in the found matching subtrees. The dissimilarity score between two structure matching subtrees T and T' is defined as

$$S_{\text{tree}}(T, T') = \sum_{R \in T,\, R' \in T',\, R \leftrightarrow R'} S_{\text{region}}(R, R'). \quad (11)$$

The symbol "$\leftrightarrow$" means the corresponding relation between two structural matching subtrees T and T'. Equation (11) means that the dissimilarity score between the two subtrees is simply the summation of the dissimilarity scores between each corresponding region pair in the two subtrees. The dissimilarity score for an image is the minimum dissimilarity score for a subtree in the image. Finally, the retrieval result is a name list of the images which contain the matching objects and is sorted by the smallest tree dissimilarity scores of all relationship matching images in ascending order.

TABLE IV
RETRIEVAL RESULTS OF QUERIES 1 TO 4

Rank	Query 1	Query 2	Query 3	Query 4
1	062 * [7]	029 *	168 *	157 *
2	192	028 *	164 *	159 *
3	200 *	027 *	163 *	158 *
4	009	116 *	167 *	060
5	059	197 *	162 *	136 *
6	067 *	026 *	082	160 *
7	144	121	041 *	094 *
8	151	043	095	138 *
9	196	120	039 *	161 *
10	089	182	165 *	165
11	150	100	135	050
12	173	144	040 *	049
13	141	065	140	051
14	016	064	139	132
15	153	119	112	154 *
16	142	044	114	022
17	066	080	113	178
18	169	107	049	155 *
19	088	066	068	179
20	097	158	070	176

TABLE V
RETRIEVAL EFFICIENCY OF TEN QUERIES AT $L = 5, 10, 15,$ AND 20

Query	N_T	$L = 5$	$L = 10$	$L = 15$	$L = 20$
1	5	0.40	0.60	0.60	0.60
2	6	1	1	1	1
3	6	0.60	0.50	0.66	0.66
4	11	0.80	0.60	0.81	0.81
5	10	1	0.80	0.90	0.90
6	12	0.80	0.70	0.75	0.83
7	14	1	1	1	1
8	17	0.80	0.80	0.60	0.58
9	10	0.80	0.80	1	1
10	14	0.60	0.80	0.78	0.78
Average	-	0.78	0.76	0.81	0.81

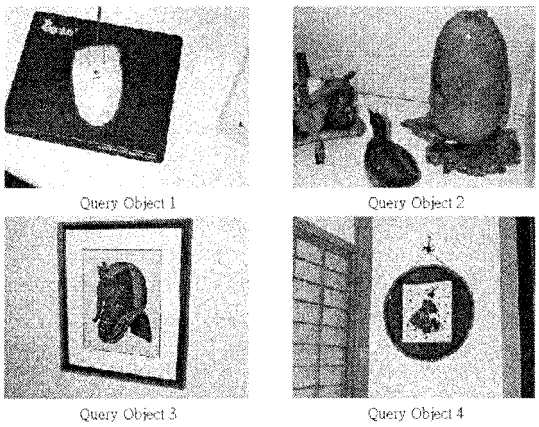


Fig. 5. Images of the query objects 1 to 4. Query objects are enclosed by red curves.

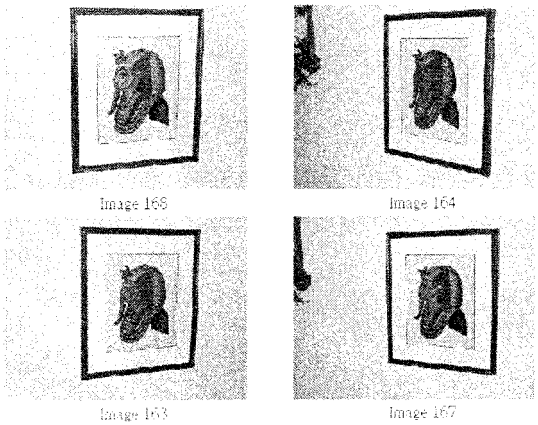


Fig. 6. Images of rank 1 to 4 for query object 3.

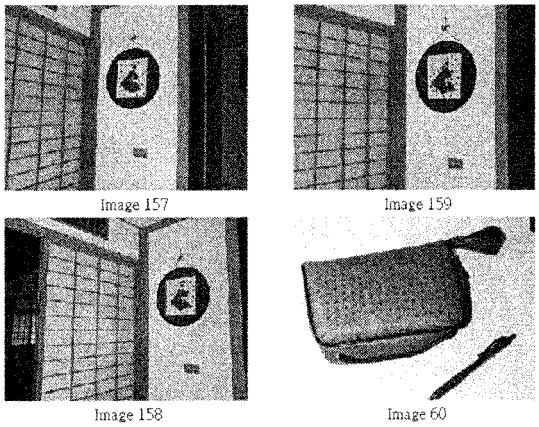


Fig. 7. Images of rank 1 to 4 for query object 4.

V. DEMONSTRATION PROGRAM

A. Implementation

We implement our approach under Microsoft Windows NT Workstation 4.0 and NT Service Pack 3.0. The machine we used is an AMD K6-200 PC with 128MB EDO-DRAM main memory and 512KB pipeline burst SRAM second level cache. The development kit is Microsoft Visual C++ 5.0.

Some parts are programmed in assembly language or by Intel's MMX (MultiMedia eXtension) techniques to accelerate the algorithm.

B. The Thresholds in Our Experiment

The four threshold values T_L, T_G, T_S, and T_N in our experiment are listed in Table II. Certainly, thresholds can be adjusted to make the segmentation better for an individual image.

C. Creation of Image Database

Our experimental image database contains 200 24-b color images. These images are all photographed by Kodak DC-210 digital camera with its standard resolution and best quality settings. The experimental image dimension is 320 pixels in width and 240 pixels in height. The performance of our segmentation when creating the database with 200 24-b color real world images is listed in Table III.

Fig. 4 shows one image of the image database with its segmentation result and relationship tree. The numbers labeled in the nodes of the

tree represent the labels which are assigned during segmentation. It is noticed that the regions in the same subtree are sorted by region size in descending order.

VI. Experimental Result

We design the experiment on our retrieval system by querying ten objects and retrieving some images. For each query object, the images containing the same object are determined by human eye as the ground truth.

After the ground truth is determined, we apply our retrieval system to obtain a list of similar images. The length of this list can be determined by users. For each query, the efficiency of retrieval η_L [5] for a given list of length L is defined as the following:

$$\eta_L = \begin{cases} \dfrac{N_S}{N_T}, & \text{if } N_T \le L, \\ \dfrac{N_S}{L}, & \text{if } N_T > L \end{cases} \tag{12}$$

where N_S is the number of the similar images retrieved in the result list, and N_T is the number of the ground truth images for the query object. Each of the ten experimental queries is, respectively, made at four different lists of lengths $L = 5,\ 10,\ 15,$ and 20.

The retrieval results of four example queries are listed in Table IV. This table just lists the numbers of the first twenty retrieval images for each query, because the longest length of retrieval list is twenty. The retrieval efficiency of the ten queries at $L = 5,\ 10,\ 15,$ and 20 is shown in Table V. Besides, the original images of the four query objects are shown in Fig. 5.

In our experiment, the time spent by every query is less than one second. This time is determined by the size and structure of the database. Our experiment database does not have any special structure, and the retrieval is sequential.

The retrieval efficiency of query object 1 is not high, because it is a one-region object. Besides, the relationship tree of query object 1 is a simple one-node tree and the object region is very common. This results in matching with each region node in every database image. However the query object 2 has more special shape, and the retrieval efficiency is high. The reason for misdetection and false-alarm in queries 2 to 4 is the segmentation. The segmentation of some ground truth images is not similar to the query object images, especially when relationship trees are different. Hence, these images can not be retrieved. The reflection and light condition in images can influence the segmentation results. Additionally when the sizes of the objects are much bigger or much smaller than that of the query object, the segmentation results are also different. The results for the more complicated query images 3 and 4 are quite good. The ranks 1 to 4 for query objects 3 and 4 are shown in Figs. 6 and 7, respectively.

VII. Conclusion and Future Work

The idea of combining a color segmentation with the creation of a hierarchical relationship tree and the use of the corresponding tree-matching method leads to an image retrieval system that has a better retrieval efficiency than those systems which only use region information. From the experiment, our approach has good retrieval efficiency when the region relationships of query objects are slightly complex. An improvement for our system is to use *Color coherence vectors* (CCV) [8] which provide more information regarding the spatial relationships of the image objects. Instead of designing the database as a continuous sequence of relationship trees, it is more efficient to use a higher level tree structure. This is especially important for huge databases. A disadvantage of our algorithm is that the retrieval process relies on exact tree matching. We can calculate the distance between two different trees by counting the number of *relabel*, *delete*, or *insert* operations for a node when we transfer a tree into another tree. To enable inexact matching we can also accept trees for solutions whose distance measure to the tree of the query image is below a definite threshold T_D. If we precompute pairwise distances between trees and also consider the smallest actual distance calculated, we are also able to eliminate trees that cannot contribute to the solution [11]. We can further consider sibling relationships.

We are currently experimenting with an image database of 550 images and we also include implementation of inexact or fuzzy matching.

References

[1] P. Aigrain, H. Zhang, and D. Petkovic, "Content-based representation and retrieval of visual media: A state-of-the-art review," *Multimed. Tools Applic.*, vol. 3, pp. 179–202, 1996.

[2] R. M. Haralick and L. G. Shapiro, *Computer and Robot Vision*. Reading, MA: Addison-Wesley, 1992, vol. I, II.

[3] G. Healey, "Segmenting images using normalized color," *IEEE Trans. Syst., Man, Cybern.*, vol. 22, pp. 64–73, 1992.

[4] G. Healey and D. Slater, "Computing illumination-invariant descriptors of spatially filtered color image regions," *IEEE Trans. Image Processing*, vol. 6, pp. 1003–1013, 1997.

[5] B. M. Mehtre, M. S. Kankanhalli, A. D. Narasimhalu, and G. C. Man, "Color matching for image retrieval," *Pattern Recognit. Lett.*, vol. 16, pp. 325–331, 1995.

[6] A. Moghaddamzadeh and N. Bourbakis, "A fuzzy region growing approach for segmentation of color images," *Pattern Recognit.*, vol. 30, pp. 867–881, 1997.

[7] S. K. Nayar and R. M. Bolle, "Computing reflectance ratios from an image," *Int. J. Comput. Vis.*, vol. 17, pp. 219–240, 1996.

[8] G. Pass, R. Zabih, and J. Miller, "Comparing images using color coherence vectors," in *Proc. ACM Conf. Multimedia*, Boston, MA, 1996.

[9] T. F. Syeda-Mahmood, "Data and model-driven selection using color regions," *Int. J. Comput. Vis.*, vol. 21, pp. 9–36, 1997.

[10] A. Tremeau and N. Borel, "A region growing and merging algorithm to color segmentation," *Pattern Recognit.*, vol. 30, pp. 1191–1203, 1997.

[11] J. T. L. Wang, K. Zhang, K. Jeong, and D. Shasha, "A system for approximate tree matching," *IEEE Trans. Knowl. Data Eng.*, vol. 6, pp. 559–571, 1994.