

AUTOMATIC ANNOTATION AND RETRIEVAL OF IMAGES

Yuqing Song

Department of Computer Science and Engineering

State University of New York at Buffalo

Buffalo, NY 14260 USA

ys2@cse.buffalo.edu

Wei Wang

Department of Computer Science and Engineering

State University of New York at Buffalo

Buffalo, NY 14260 USA

wwang3@cse.buffalo.edu

Aidong Zhang

Department of Computer Science and Engineering

State University of New York at Buffalo

Buffalo, NY 14260 USA

azhang@cse.buffalo.edu

Abstract We propose a novel approach for semantics-based image annotation and retrieval. Our approach is based on monotonic tree, a derivation of contour tree for discrete data. Monotonic tree provides a way to bridge the gap between the high-level semantics and low-level features. Each branch (subtree) of the monotonic tree is termed as a *structural element* if its area is within a given scale. The structural elements are classified and clustered based on their low level features such as color, spatial location, harshness, and shape. Each cluster corresponds to some semantic feature. The category keywords indicating the semantic features are automatically annotated to the images. Based on the semantic features extracted from images, high-level (semantics-based) querying and browsing of images can be achieved. The experimental results demonstrate the effectiveness of our approach.

Keywords: Content-based image retrieval, semantics, monotonic tree

1. Introduction

Although tremendous work has been done on content-based image retrieval, efficient and effective image retrieval still remains an open problem. Content-based image retrieval using low-level features such as color [Swain and Ballard, 1991; Smith and Chang, 1996b; Pass et al., 1996], texture [Manjunath and Ma, 1996; Smith and Chang, 1994; Sheikholeslami and Zhang, 1997], shape [Syeda-Mahmood, 1996; Mehrotra and Gary, 1995; Hirata and Kato, 1993] and others [Picard, 1996; Smith and Chang, 1996a; Ahuja and Rosenfeld, 1981] has been well studied.

However, retrieving images based on low-level features has been proven unsatisfactory. With the enormous growth of image databases, it is an urgent need to build image retrieval systems which support high-level (semantics-based) querying and browsing of images. Keyword indexing is a common scheme used by many picture libraries. For example, Getty Images [Bjarnestam, 1998] used over 10,000 keywords to index their collection of contemporary stock photographs. Current image indexing by keywords can only be done manually. According to [Eakins and Graham, 1999], the process of manual indexing suffers from two significant drawbacks. Firstly, it is inherently very labour-intensive. Secondly, manual indexing does not appear to be particularly reliable as a means of subject retrieval of images.

In recognizing the existing problems in the CBIR field, we believe that research efforts are needed to bridge the gap between the high-level semantics users are interested in and the low-level features that can be extracted. We propose a novel approach to extracting high-level semantics from low-level features. Our approach is based on monotonic tree [Song and Zhang, 2002]. Branches (subtrees) of the monotonic tree are termed as a *structural elements* if their areas are within a given scale. The structural elements are classified and clustered based on their low level features such as color, spatial location, harshness, and shape. Each cluster corresponds to some semantic feature. The category keywords indicating the semantic features are automatically annotated to the images. Based on the semantic features extracted from images, high-level (semantics-based) querying and browsing of images can be achieved.

We focus our attention on scenery images, which provide a popular testbed for semantics extraction. Technically, scenery images are relatively easier to analyze than other images. The reasons are following. Firstly, types of objects are limited in scenery images. Main scenery object types include sky, tree, building, mountain, lawn, water, and snow. Secondly, as compared with color, texture, or the spatial location of

image elements, shape features are less important in analyzing scenery images than in other images. Thus we can avoid our weakness in shape matching when extracting semantic features.

The remainder of this paper is organized as follows. Section 2 introduces the idea of the monotonic tree. In Section 3, techniques for extracting semantic features are described. Section 4 presents case study of scenery features, while Section 5 describes our system design. Section 6 offers a performance evaluation of the proposed approach. A summary and concluding remarks appear in Section 7.

2. Introduction to Monotonic tree

Monotonic tree [Song and Zhang, 2002] is a derivation of contour tree for discrete data. Monotonic tree is used as a hierarchical representation of image structures. Contour trees [Morse, 1969; van Kreveld et al., 1997] have been used in geographic information systems (GIS) and medical imaging to display scalar data. Contours are only defined for continuous functions. For an image represented by discrete data, a continuous function is first defined as an interpolation of the data. Then the contour tree is defined on this continuous function.

We introduce a new concept termed monotonic line, which is directly defined on discrete data. An *outward-falling/climbing monotonic line* of an gray image is a boundary where the image assumes higher/lower values in the pixels adjacent to the boundary from inside than those from outside. All monotonic lines in an image form a rooted tree, called *monotonic tree*. A maximal sequence of uniquely enclosing monotonic lines is called a *monotonic slope*. All monotonic slopes in an image form the *topological monotonic tree*. A monotonic slope is called *outward-falling/climbing* if all monotonic lines in it are outward-falling/climbing. See Figure 1. For a color image, the monotonic tree and topological monotonic tree are constructed on its gray copy.

3. Extracting Semantic Features

Our feature extraction scheme is based on the topological monotonic tree. We use the branches of the topological monotonic tree to model the basic structures in an image, which are termed as *structural elements*. Structural elements have low-level features such as color, shape, harshness, and spatial location. They are clustered to form high-level features.

Feature extraction consists of three consecutive steps: (a) classifying structural elements; (b) clustering structural elements; and (c) rendering semantic regions. See Figure 2.

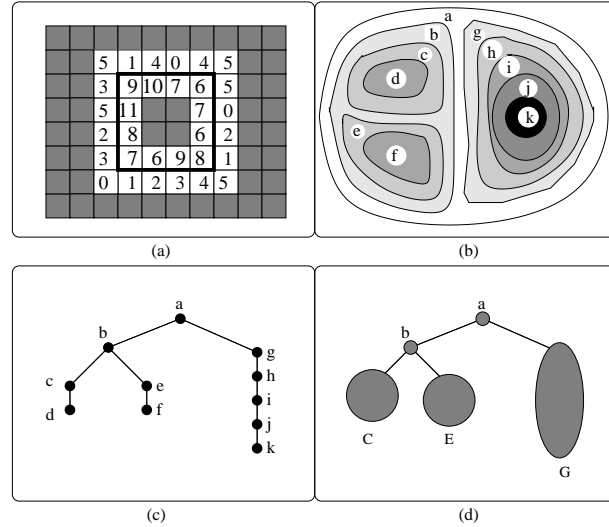


Figure 1. (a) An outward-falling monotonic line (the solid line in the figure), (b) a set of monotonic lines, (c) the monotonic tree, (d) the topological monotonic tree.

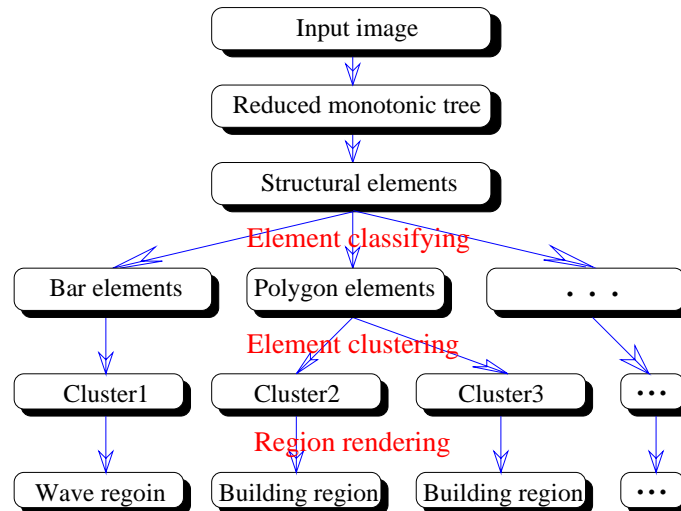


Figure 2. System design for feature extraction.

3.1. Classifying Structural Elements

Each branch (a subtree) of the topological monotonic tree is called a *structural element* if its covered area is no more than a threshold, which gives the scale in which we are interested. A structural element

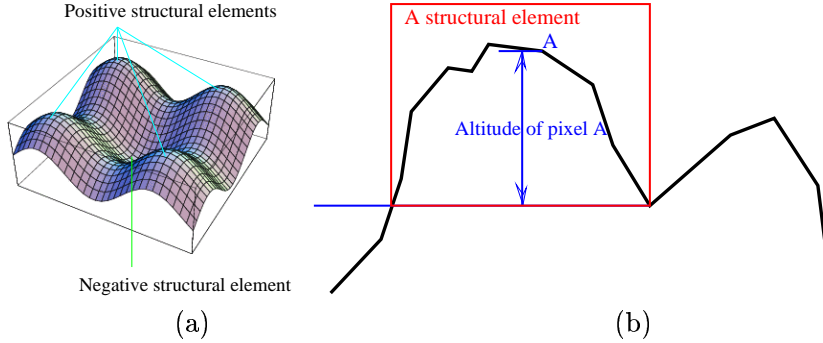


Figure 3. (a) Positive/negative structural elements; (b) the altitude of a pixel in a structural element.

is called *positive/negative* if its root (root of the subtree) is outward-falling/climbing. See Figure 3(a). Positive/negative elements are like peaks/valleys. For a positive/negative element, we define its *altitude* to be the absolute value of the average altitude of all its pixels above/below the highest/lowest pixels adjacent to the structural element. See Figure 3(b). The *harshness* of a structural element is determined by the number, area and altitude of its sub-elements.¹ We define the harshness of an element t by

$$Harshness(t) = \frac{\sum_{b \in SubElementSet(t)} Altitude(b) * Area(b)}{Area(t)},$$

where $SubElementSet(t)$ is the set of sub-elements of t , $Altitude(b)$ is the altitude of b and $Area(b)$ is the area of the region covered by b .

A structural element can be classified by its (1) color (the average color of all pixels in this element), (2) altitude, (3) harshness, and (4) shape (the shape of its covered region). By shape, we can classify elements as: (a) bars, (b) polygons, (c) irregular elements, (d) boundary-smooth elements, and (e) others. For a bar element, the ratio of its length to its width is high. A polygon element is a structural element whose boundary mainly consists of line segments. For a smooth-boundary element, its boundary is a smooth curve. The irregular elements are those whose boundaries are irregular. Figure 4 shows different cases of elements.

The semantic features of scenery images are characterized by the categories of structural elements. Three examples of the categories are polygon elements (for building), horizontal bar elements (for wave), and green harsh irregular elements (for tree). See Figures 5 and 6.

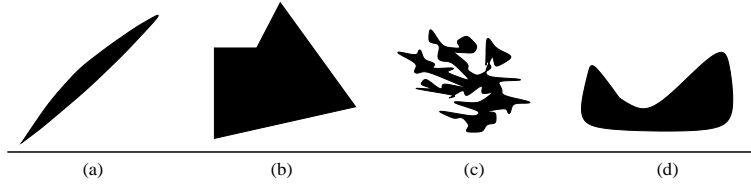


Figure 4. (a) A bar element, (b) a polygon element, (c) an irregular element, and (d) a smooth-boundary element.

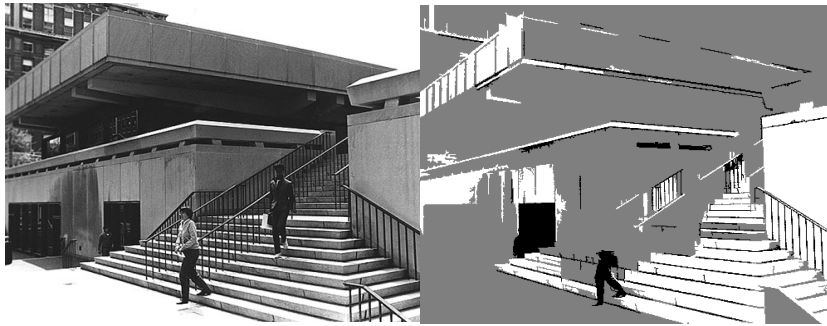


Figure 5. An image of building (in the left) and the polygon elements (in the right) in it. The polygon elements are shown in black and white.



Figure 6. A image of river (in the left); and (a) green harsh irregular elements (shown in green or dark green), (b) horizontal bar elements (shown in black and white).

3.2. Clustering Structural Elements

Given an image, for each category of structural elements we are interested in, we apply clustering algorithms to find the clusters.

For a given category, we first find the set of qualified elements (i.e., the elements belong to this category). For two qualified elements, if they overlap, then the one with lower qualifying score ² is removed. This process is called *element sifting*. After sifting, we reduce multi-

level elements of the image into one level elements, which all belong to the given category. The elements after sifting form some element pattern in the 2D plane.

For the element pattern, we construct its Delaunay graph, which is the neighboring graph of the element pattern. We then apply clustering algorithms on the neighboring graph to find the clusters in the element pattern. In our implementation, the clustering algorithm is based on the minimal spanning tree of the neighboring graph. Reference on pattern processing by neighboring graph can be found in [Ahuja, 1982; Ahuja and Tuceryan, 1989]. Reference on clustering by minimal spanning tree can be found in [Zahn, 1971; Dugad and Ahuja, 1998].

3.3. Rendering Semantic Regions

Given a cluster of structural elements, the region rendering process consists of three steps: (1) element connecting; (2) hole filling; and (3) boundary smoothing. At the first step, we connect all elements in the cluster by line segments whose lengths are within a threshold. At the second step, we fill the holes whose areas are less than an area threshold. At the last step, we smooth the boundary of the region by removing those irregular angles and branches. Figure 7 shows an example of these steps.

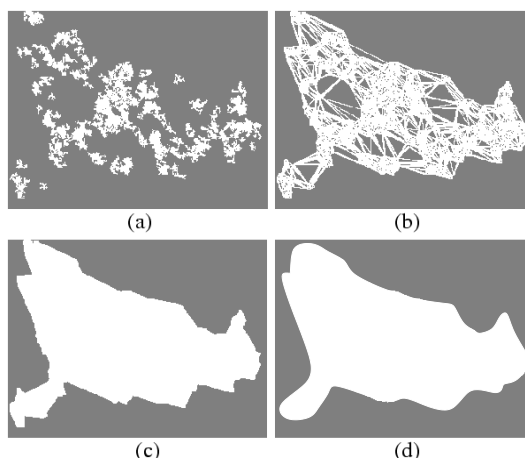


Figure 7. (a) A cluster of structural elements, (b) element connecting (c)hole filling, and (d)boundary smoothing.

4. Case Study of Semantic Features

In this section, we discuss the identification of high level scenery features: sky, building, tree, water wave, placid water, and ground. Water

wave and placid water have different structures in images, thus they are treated with different schemes. The ground feature in images can be further split into snow, lawn and other kinds of ground.

4.1. Sky

Without clouds, a sky region is a homogeneous region consisting of blue pixels. In natural images, clouds tend to change smoothly at pixel level, due to their physical properties. In location, there is usually no other object above a sky region. To retrieve sky regions, We make three simple assumptions:

- (sky.a1) A sky region is smooth;
- (sky.a2) A sky region occupies an upper part of the image; and
- (sky.a3) The color of sky regions is either blue or the color of clouds.

For our current implementation, we assume that the color of clouds is black-white.³

To find the sky regions, we first find the smooth regions in the image. The smooth regions are the complement of the harsh regions, which is characterized by intensity peaks and valleys. Under monotonic tree, the intensity peaks and valleys are modeled as small structural elements whose altitudes are high. Thus we detect the harsh regions of the image by clustering the small elements whose altitudes are high, as we discussed in last section. When we get the harsh regions, we also get the smooth regions. Then we check the location and color of the smooth regions to find the sky regions.

4.2. Ground and Placid Water

We make three assumptions about ground regions.

- (ground.a1) A ground region is smooth;
- (ground.a2) A ground region is in the lower part of the image; and
- (ground.a3) In a ground region, micro-structures are more horizontal than vertical.

When scenery pictures are taken, the direction of projection is usually horizontal or nearly horizontal. Thus, as the natural scene is projected to the image plane, the structures on the ground appear more horizontal than vertical, which is the reason why we make the third assumption.

Similar to detecting background regions, we first find the smooth regions in the image. Then for each smooth region, we check if the last two assumptions hold or not. For the last assumption, we count the horizontal and vertical elements in the smooth region. The last assumption

holds if the horizontal elements are more than the vertical elements in the region.

A ground region found this way could be lawn, snow, or other kinds of ground. We distinguish these kinds of regions by their colors. We assume that lawn regions are green, snow regions white.

For placid water, we make four assumptions. The first three are the same as the three assumptions of ground. Besides, we assume that the color of placid water is blue.

4.3. Wave

Small water waves have very regular patterns. When the small waves projected to a picture, they appear to be horizontal bars if the projecting direction is nearly horizontal. See the image in Figure 6. Wild waves have complicated structures. In the images with wild waves, we usually can detect a piece of surface consisting of parallel bar structures. See the image in Figure 8. We assume that a wave region is a region consisting of horizontal bar elements. We detect wave regions by clustering horizontal bar elements in the image.



Figure 8. An example of image with big wave.

4.4. Green Tree

If we look at the tree region in the Figure 6 carefully, we can find that the micro structures in the region are very irregular. Based on this observation, we assume that a tree region is a region consisting of green⁴ harsh, irregular elements. The tree regions in an image are found by clustering the green, harsh irregular elements in the image.

4.5. Building

The shapes of most buildings are characterized by the line segments inside them. We assume that a building region in an image is a region consisting of polygon elements. To check whether a structural element

is a polygon element, we first partition its boundary into line segments and other kinds of segments. A structural element is a polygon element if its boundary mainly consists of line segments. Reference on curve partitioning can be found in [Robl and Farber, 1998; Rosin and West, 1989; Rosin and West, 1992].

All of the above cases can be viewed in our demo system located at <http://monet.cse.buffalo.edu:8888/>.

5. Image querying

Our demo system, named SceneryAnalyzer, has three main components: (1) image database, (2) feature extraction, and (3) image querying. The feature extraction is done off-line. During feature extraction, each image is processed to generate semantic features, which are stored in a feature vector. The semantic features are automatically annotated in the gray copy of the original image. All feature vectors are stored in a vector base. To retrieve images, users can select any combination of the listed features. After the query is submitted, the icons of images with the selected features are output in pages. If users click an icon, a window is pop up with the original and annotated images.

6. Experiments

We conducted experiments to compare the performance between our approach and traditional CBIR techniques including keyblock model [L. Zhu, A. Rao and A. Zhang, 2000], traditional color histogram [Swain and Ballard, 1991], color coherent vector [Pass et al., 1996], and wavelet (Haar and Daubechies) texture techniques [Smith and Chang, 1994; Strang and Nguyen, 1996]. The comparison is made by the precisions and recalls of each method on six scenery features: sky, building, tree, wave, ground, and placid water.

We used 6776 COREL images in the experiments. They are selected from CD7 and CD8 of COREL Gallery 1,300,000. The COREL images can be split into two parts: scenery and nonscenery parts. There are 4125 scenery images, which are pictures taken at countries all around the world. The nonscenery part has 2651 images, which cover a large variety, including different kinds of textures, marine fishes, molecules, space scenes, and insects. Table 1 shows the statistics of the scenery features in the COREL images (each entry shows the number of images with some feature).

For each scenery feature, to show the performance of SceneryAnalyzer, we calculate the precisions and recalls of first $\frac{n}{30}$, $\frac{2n}{30}$, $\frac{3n}{30}$, ..., , and n

Table 1. The statistics of scenery features in 6776 COREL images.

feature	sky	building	tree	wave	placid water	ground		
						lawn	snow	other
images	2369	1920	1479	161	882	298	68	659

images retrieved by SceneryAnalyzer with this feature, where n is the total number of images retrieved by SceneryAnalyzer.

Traditional CBIR techniques accept only queries by examples. Let’s take an example to show how we choose query sets and calculate the precision-recall for these methods. The example is about keyblock on sky feature. There are 2369 COREL images with sky regions. For each sky image, we use it as a query on the COREL database to select top 100 images by the keyblock approach, and count the number of sky images in this retrieved set. Then we sort the 2369 sky images descendingly by the numbers of sky images in their corresponding retrieved sets. Let the sorted list be *SKYLIST*. Then we select the first 5%, i.e., 118 images of *SKYLIST* as the query set, which is denoted as *QUERYSET*. Then for each COREL image I , we calculate its distance to *QUERYSET* - $\{I\}$ by the keyblock approach. The COREL images are sorted ascendantly by this distance. Top 2369⁵ COREL images are retrieved. Based on the retrieved images by keyblock, we calculate and plot the precision-recall of keyblock on sky, as we did for SceneryAnalyzer.

The result is in Figure 9. By comparing the graphs in Figure 9, we can see that our method outperforms all the others on each feature. To see the comparison clearly, we calculate the average precision-recall on the six scenery features, which is shown in Figure 10. From these comparisons, we can see that our method is much better than the traditional techniques on scenery features.

7. Conclusion

In this paper, we used a model termed monotonic tree to model high-level scenery features. Based on the monotonic tree representation, primitive elements of low-level features such as color, shape, harshness, and spatial location can be easily identified, clustered and combined to form semantically meaning regions (or features) for images. Thus, images can be automatically annotated with category keywords, including sky, building, tree, wave, lawn, water, snow, and ground. With this annotation, high-level (semantics-based) querying and browsing of images can be supported.

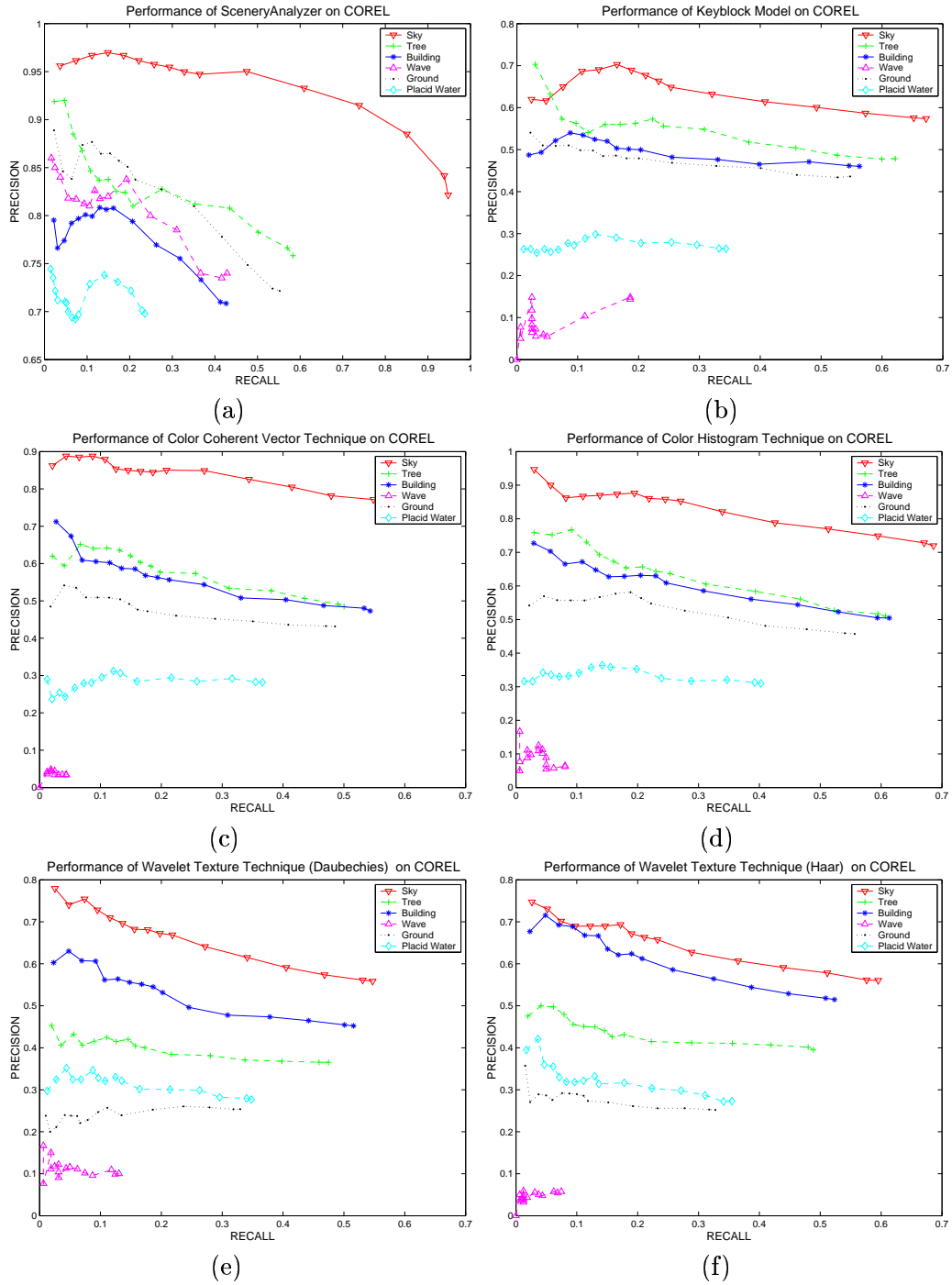


Figure 9. Performance of (a) SceneryAnalyzer, (b) keyblock model, (c) color coherent vector, (d) color histogram, (e) Daubechies wavelet, and (f) Haar wavelet on COREL images

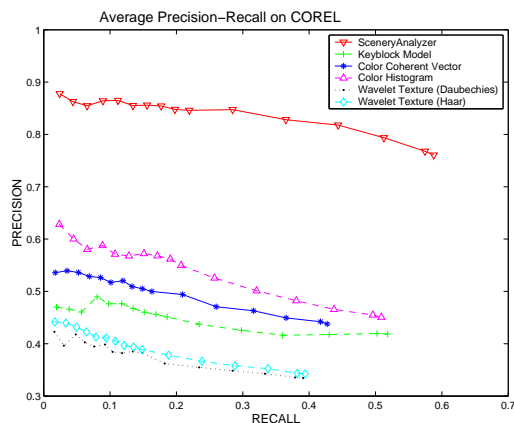


Figure 10. Average precision-recall on six features: sky, building, tree, wave, ground and placid water.

Notes

1. An element is a branch of the topological monotonic tree. All sub-branches of this branch are sub-elements.
2. For a given category, the qualifying score of an element indicates how qualified the element is to belong to this category.
3. Clouds can have more colors. In the following discussion, we also make some simple assumptions about the colors of trees, water and snow. More comprehensive color patterns of sky, trees, water, and snow can be integrated into our system, which is not the focus of this paper.
4. Again, we make a simple assumption about the color of trees.
5. This is the number of COREL images with sky.

References

- Ahuja, N. (1982). Dot pattern processing using voronoi neighborhoods. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 4(3):336–343.
- Ahuja, N. and Rosenfeld, A. (1981). Mosaic models for texture. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 3(1):1–11.
- Ahuja, N. and Tuceryan, M. (1989). Extraction of early perceptual structure in dot patterns: integrating region, boundary, and component gestalt. *Computer Vision, Graphics, and Image Processing*, 48(3):304–356.
- Bjarnestam, A. (Feb.5, 1998). Description of an image retrieval system. In *The Challenge of Image Retrieval research workshop, Newcastle upon Tyne*.
- Dugad, R. and Ahuja, N. (1998). Unsupervised multidimensional hierarchical clustering. In *IEEE International Conference on Acoustics Speech and Signal Processing, Seattle*.
- Eakins, J. and Graham, M. (Jan. 10, 1999). Content-based image retrieval. In *Reports of JISC Technology Applications Programme*.

- Hirata, K. and Kato, T. (1993). Rough sketch-based image information retrieval. *NEC Research & Development*, 34(2):263–273.
- L. Zhu, A. Rao and A. Zhang (2000). Theory of keyblock-based image retrieval. *ACM Transactions on Information Systems*.
- Manjunath, B. and Ma, W. (1996). Texture Features for Browsing and Retrieval of Image Data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(8):837–842.
- Mehrotra, R. and Gary, J. E. (1995). Similar-shape retrieval in shape data management. *IEEE Computer*, 28(9):57–62.
- Morse, S. (1969). Concepts of use in computer map processing. *Communications of the ACM*, 12(3):147–152.
- Pass, G., Zabih, R., and Miller, J. (1996). Comparing images using color coherence vectors. In *Proceedings of ACM Multimedia 96*, pages 65–73, Boston MA USA.
- Picard, R. (1996). A society of models for video and image libraries. Technical Report 360, MIT Media Laboratory Perceptual Computing.
- Robl, C. and Farber, G. (1998). Contour tracer for a fast and precise edge-line extraction. In *IAPR Workshop On Machine Vision Applications (MVA98)*.
- Rosin, P. and West, G. (1989). Segmentation of edges into lines and arcs. *Image and Vision Computing*, 7(2):109–114.
- Rosin, P. and West, G. (1992). Multi-stage combined ellipse and line detection. In *British Machine Vision Conference (BMVC92)*, pages 197–206.
- Sheikholeslami, G. and Zhang, A. (1997). An Approach to Clustering Large Visual Databases Using Wavelet Transform. In *Proceedings of the SPIE Conference on Visual Data Exploration and Analysis IV*, pages 322–333, San Jose.
- Smith, J. and Chang, S. (1996a). Visualseek: A fully automated content-based image query system. In *ACM Multimedia 96*.
- Smith, J. R. and Chang, S. (1994). Transform Features For Texture Classification and Discrimination in Large Image Databases. In *Proceedings of the IEEE International Conference on Image Processing*, pages 407–411.
- Smith, J. R. and Chang, S.-F. (1996b). VisualSeek: a fully automated content-based image query system. In *Proceedings of ACM Multimedia 96*, pages 87–98, Boston MA USA.
- Song, Y. and Zhang, A. (April 3-5, 2002). Monotonic tree. In *The 10th International Conference on Discrete Geometry for Computer Imagery, Bordeaux, France*.
- Strang, G. and Nguyen, T. (1996). *Wavelets and Filter Banks*. Wellesley-Cambridge Press, Wellesley, MA.
- Swain, M. and Ballard, D. (1991). Color Indexing. *Int Journal of Computer Vision*, 7(1):11–32.
- Syeda-Mahmood, T. (1996). Finding shape similarity using a constrained non-rigid transform. In *International Conference on Pattern Recognition*.
- van Kreveland, M., van Oostrum, R., Bajaj, C., Pascucci, V., and Schikore, D. (1997). Contour trees and small seed sets for iso-surface traversal. In *Proc. 13th Ann. Sympos. Comput. Geom.*, pages 212–220.
- Zahn, C. (1971). Graph-theoretical methods for detecting and describing gestalt clusters. *IEEE Trans. on Computers*, C-20:68–86.