Pictorial Query Trees for Query Specification in Image Databases

Aya Soffer [*], Hanan Samet [†], Dmitry Zotkin
Computer Science Department and
Center for Automation Research and
Institute for Advanced Computer Science
University of Maryland at College Park
College Park, Maryland 20742
E-mail: {aya,hjs,dz}@umiacs.umd.edu

Abstract

A technique that enables specifying complex queries in image databases using pictorial query trees *is presented. The leaves of a pictorial query tree correspond to individual pictorial queries that specify which objects should appear in the target images as well as how many occurrences of each object are required. In addition, the minimum required certainty of matching between query-image objects and database-image objects, as well as spatial constraints that specify bounds on the distance between objects and the relative direction between them are also specified. Internal nodes in the query tree represent logical operations (AND, OR, XOR) and their negations on the set of pictorial queries (or subtrees) represented by its children. The syntax of query trees is described. Algorithms for processing individual pictorial queries and for parsing and computing the overall result of a pictorial query tree are outlined.*

1. Introduction

A basic requirement of an image database is the ability to query the database pictorially. The most common method of doing this is querying via an example image. The problem with this method is that in an image database we are usually not looking for an exact match. Instead, the goal is to find images that are similar to a given query image. The main issue is how to determine if two images are similar and whether the similarity criteria that are used by the database system match the user's notion of similarity.

In our previous work [8], we devised a pictorial query specification technique that enables the formulation of queries that specify which particular objects should appear in a target image as well as how many occurrences of each object are required and the desired spatial configuration among these objects. Although this method allowed combining pictorial queries via logical operators, these were basically limited to binary combinations and to the operators AND and OR. In this paper, we present an extension of our pictorial query specification that enables the formulation of complex pictorial queries via *pictorial query trees*. The leaves of a pictorial query tree correspond to individual pictorial queries. Internal nodes in the tree represent logical operations on the set of pictorial queries (or subtrees) represented by its children. Currently, three logical operations (AND, OR, XOR) and their negations are supported.

Most of the existing image database research has dealt either with global image matching based on color and texture features [6, 4, 7] or with the ambiguity associated with matching one query-image object to another [5]. There has also been some work on the specification of topological and directional relations among query objects [1, 2, 3].

2. Specifying Individual Pictorial Queries

We briefly review how individual pictorial queries are specified using our method. For more details and examples, see [8]. The matching similarity level *msl* is a number between 0 and 1 that specifies a lower bound on the certainty that two symbols are from the same class and thus considered a match. *Contextual similarity* specifies how well the content of database image *DI* matches that of query image *QI* (e.g., do all of the symbols in *QI* appear in *DI*?). We make use of four levels of contextual similarity. Figure 1 summarizes these levels. *Spatial similarity* specifies how good a match is required in terms of the relative locations and orientation of the matching symbols between the query and database image. We make use of five levels of spa-

[*] The support of the National Science Foundation under Grant CDA-950-3994 is gratefully acknowledged.

[†] The support of the National Science Foundation under Grant IRI-97-12715 is gratefully acknowledged.

tial similarity. Figure 2 summarizes the 5 levels of spatial similarity.

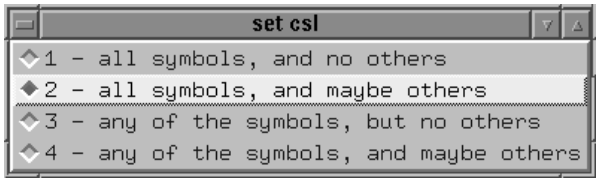

Figure 1. Contextual similarity levels (csl).

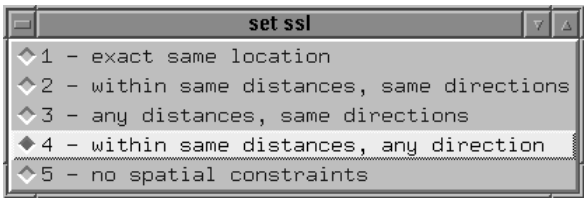

Figure 2. Spatial similarity levels (ssl).

3. Building Complex Pictorial Query Trees

3.1. Syntax and Semantics of Pictorial Query Trees

Complex pictorial queries that involve combinations of individual pictorial queries are specified via *pictorial query trees*. The leaves of a pictorial query tree correspond to individual pictorial queries. The result of an individual pictorial query is a set of images that satisfy the constraints imposed by the query. A leaf node may be negated (NOT). In this case, the result of the query is the set of all images that do not satisfy the pictorial query. Internal nodes in the tree represent logical operations (AND, OR, XOR) and their negations (NAND, NOR, NXOR) on the set of images that satisfy the pictorial query (or query subtree) represented by its children. The root of the tree is either a pictorial query or a logical operator, while an internal node corresponds to a logical operator and can have one or more children.

For a conjunction of query images where the same symbol appears in both query images, the user may specify whether the two query-symbols must match the same instance of the symbol in the database image, or whether two different instances are allowed, termed *object binding*.

3.2. Example Query Trees

Figures 3 and 4 demonstrate the use of pictorial query trees. Figure 3a demonstrates a simple query tree used to specify more than one acceptable spatial constraint (i.e., via use of an OR). We add the condition that there is an airport northeast of and within 7 miles of the fishing site by

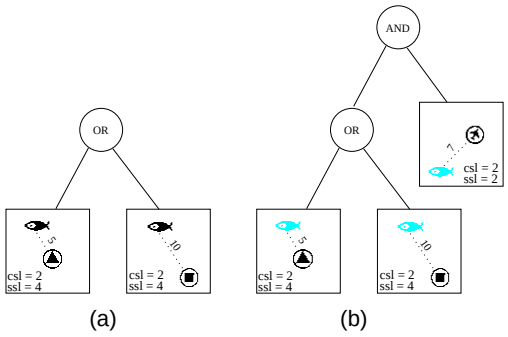

Figure 3. (a) Images with a camping site within 5 miles of a fishing site OR a hotel within 10 miles of a fishing site . (b) Images with a camping site within 5 miles of a fishing site OR a hotel within 10 miles of a fishing site AND an airport northeast of and within 7 miles of the fishing site .

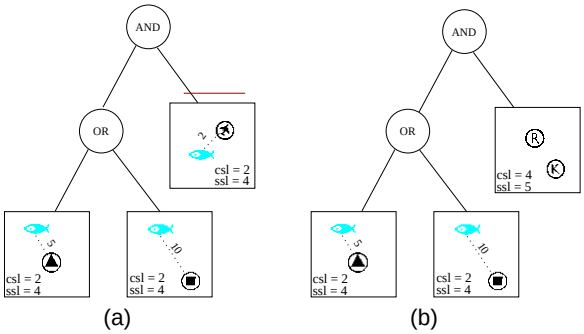

Figure 4. (a) Images with a camping site within 5 miles of a fishing site OR with a hotel within 10 miles of a fishing site AND with no airport within 2 miles of the fishing site (the line above a pictorial query represents negation). (b) Images with a camping site within 5 miles of a fishing site OR a hotel within 10 miles of a fishing site AND a restaurant or cafe .

use of an AND as shown in Figure 3b. Notice that in this case we use $ssl = 2$ since we are specifying both a distance and a direction spatial constraint (see Figure 2). In addition, we use object binding in order to specify that we want the airport to be northeast of and within 7 miles of the particular fishing site that satisfied the other part of the query. Two symbols that have the same non-black color are bound, whereas black symbols are not bound. Figure 4a demonstrates the use of negation of a pictorial query in order to specify a negative condition, namely that there should be no airport within 2 miles of our fishing site . Since the

direction is irrelevant in this case, we use $ssl = 4$. The query in Figure 4b demonstrates the use of different values of *csl* for query components. No spatial constraints are specified for the restaurant Ⓡ and cafe Ⓚ symbols, and since $csl = 4$, this component requests images containing either symbol (as opposed to both symbols in the other components).

4. Pictorial Query Processing

The first step in finding all database images that conform to a pictorial query tree specification is to process each pictorial query image (i.e., each leaf) that is part of the pictorial query tree individually.

First, for each symbol in the query image we find all database images, DI, that contain this symbol with the required matching similarity, msl. Next, if csl is set to 1 or 2 (which means that we want to obtain images that contain all of the symbols in QI), then the set of result images from the first step are intersected. On the other hand, if csl is 3 or 4 (any one symbol from QI is enough), then the union of the result images is taken. If the contextual similarity level is set to 1 or 3, then we need to avoid including images containing symbols that are not present in QI .

The next step is to check whether the spatial constraints are satisfied for each candidate image I that satisfied the contextual constraint. Since we allow multiple instances of symbols in the query image QI and in I, this step needs to check many possible matchings. Therefore, for each QI symbol we create a set of possible matches in I, which contains all of the symbols from the same class. If none of the possible matchings pass the spatial constraints test, then we remove the image from the candidate result set.

The spatial similarity between any two matchings is calculated using an algorithm which determines whether the spatial constraints dictated by a query image and spatial similarity level *ssl* hold in a logical image. Finally, images that passed all of the tests (matching, contextual, and spatial) are ordered by the average matching certainty of all matching symbols and returned as the result of the query.

The pictorial query tree is parsed and evaluated using the following recursive process. It first checks if N (the current node) is a leaf or an internal node. If N is a leaf node, then the algorithm for processing individual pictorial queries is invoked with the logical image QI and the values of matching, contextual, and spatial similarity levels msl, csl, and ssl that were specified by the user while creating or editing the corresponding leaf node. If the leaf node is negated in the tree, then the complement of the result images set returned by this function is taken. If N is an internal node in the query tree, then we recursively call this process on each child of N, followed by applying the appropriate logical operation on the results of these calls. The whole query tree is evaluated in this recursive manner by invoking algorithm *ProcessQueryTree* with the root of the query tree as input. Our algorithms check for multiple instances of symbols in the query and database images as well as for object binding. This is not described here for lack of space.

5. Concluding Remarks

The algorithm outlined here is a relatively naive solution for processing pictorial query trees. Many optimizations are possible. These include changing the order of processing of the individual query images in order to execute the parts that are more selective first, and combining individual query images and processing them together. These and other query optimization issues are the subject of future research.

Using our method, we cannot specify conditions involving the location of certain events between objects (e.g., the point where two one-lane roads ═ intersect). Furthermore, we do not consider the size or direction of the object itself. For example, we cannot specify "an open field ▤ whose area is at least 1 square mile" or "a local road ▨ that goes from north to south". Finally, we cannot qualify objects in terms of non-spatial conditions. For example, we would like to specify "hotels whose price is less than $80 per night". Incorporating these features into our pictorial query specification method is also a subject for future research.

References

[1] A. D. Bimbo, E. Vicario, and D. Zingoni. A spatial logic for symbolic description of image contents. *Journal of Visual Languages and Computing*, 5(3):267–286, Sept. 1994.

[2] S. K. Chang, Q. Y. Shi, and C. Y. Yan. Iconic indexing by 2-D strings. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(3):413–428, May 1987.

[3] M. J. Egenhofer. Query processing in spatial-query-by-sketch. *Journal of Visual Languages and Computing*, 8(4):403–424, Aug. 1997.

[4] C. Faloutsos, R. Barber, W. Equitz, M. Flickner, W. Niblack, and D. Petkovic. Efficient and effective querying by image content. *Journal of Intell. Info. Systems*, pages 231–62, 1994.

[5] W. I. Grosky, P. Neo, and R. Mehrotra. A pictorial index mechanism for model-based matching. *Data & Knowledge Engineering*, 8(4):309–327, Sept. 1992.

[6] C. E. Jacobs, A. Finkelstein, and D. H. Salesin. Fast multiresolution image querying. In *Proc. of SIGGRAPH'95 Conference*, pages 277–286, Los Angeles, CA, Aug. 1995.

[7] A. Pentland, R. W. Picard, and S. Sclaroff. Photobook: Content-based manipulation of image databases. In *Proc. of the SPIE, Storage and Retrieval of Image Databases II*, volume 2185, pages 34–47, San Jose, CA, Feb. 1994.

[8] A. Soffer and H. Samet. Pictorial query specification for browsing through image databasess. In *Proceedings of the Second International Conference on Visual Information Systems*, pages 117–124, San Diego, California, Dec. 1997.