Pictorial Query Specification for Browsing Through Image Databases

Aya Soffer [*]
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@umiacs.umd.edu

Hanan Samet [†]
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: hjs@umiacs.umd.edu

Abstract

A pictorial query specification technique that enables the formulation of complex pictorial queries for browsing through an image database is presented. Using our technique, it is possible to specify which particular objects should appear in the target images as well as how many occurrences of each object are required. Moreover, it is possible to specify the minimum required certainty of matching between query-image objects and database-image objects, as well as to impose spatial constraints that specify bounds on the distance between objects and the relative direction between objects. Each pictorial query is composed of one or more query images. Each query image is constructed by selecting the required query objects and positioning them according to the desired spatial configuration. Boolean combinations of two or more query images are also possible by use of AND and OR operators. A query image may be negated in order to specify conditions that should not be satisfied by the database images that are retrieved successfully. Several example queries are given that demonstrate the expressive power of this query specification method.

1 Introduction

Consider a database of images each composed of several objects (or symbols). Suppose that we wish to find all of the images in the database that contain two particular objects in a specific spatial configuration with respect to each other. One method to achieve this is by an SQL extension with additional predicates corresponding to spatial relationships. One problem with this method is that the objects in the images must be preclassified so that the user can specify them by some alphanumeric tag.

An alternative method is to specify the queries pictorially. This is a more "natural" method that facilitates the use of more complex constraints based on the implicit characteristics of the pictorial query (i.e., the particular objects in the pictorial query and their spatial arrangement). There are, however, several difficulties associated with pictorial query specifications. First of all, pictorial queries are inherently ambiguous which gives rise to several questions. In particular, what criteria should be used in order to determine that an object in a database image is the same as a particular object in the query image (termed *matching ambiguity*)? In addition, when query images are composed of several objects, are we looking for images that contain all of these objects, or would we be satisfied with any subset of these objects (termed *contextual ambiguity*)? Finally, is the spatial arrangement of the query objects of significance? For example, if one object in the query image is placed above and within 30 units of another object, what database images satisfy this query? One possibility is that only database images with exactly the same spatial configuration satisfy the query. However, the intent may be that only the distance must be the same, or maybe that any configuration may suffice (termed *spatial ambiguity*). Another difficulty with pictorial queries is that they are not always as expressive as textual queries in terms of specifying combinations of conditions and negative conditions. For example, how do we specify pictorially images that contain beaches but do not contain camping sites within 3 miles of these beaches?

There have been a number of studies of pictorial queries in recent years. These have been mainly in the domain of spatial and image databases [10]. Most of the image database research has dealt either with global image matching based on color and texture features (e.g., [5, 7]) or with the ambiguity associated with matching one query-image object to another(e.g., [1]). These methods do not address

[*]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-92-16970 is gratefully acknowledged.

the case of images that are composed of several objects and their desired spatial configuration. In other words, these methods only address the problem of matching ambiguity and do not deal with contextual and spatial ambiguity at all.

There has also been some work on the specification of topological and directional relations among query objects (e.g., [3, 4, 6, 9]). The focus of this work has been on defining spatial relations between objects and efficiently computing them when the objects are stored in a database. These studies only deal with tagged images (images in which the objects have already been recognized and tagged with their semantic meaning). Therefore, they do not address the issue of matching ambiguity. Furthermore, it is always assumed that the goal is to match as many query-image objects to database-image objects as possible, and, in most cases, it is also assumed that the relative locations of the objects must be exactly as specified by the query image or as close to that as possible (e.g., [4]). A limited form of spatial ambiguity is allowed in pictorial queries based on the *2D-string* and its variants [3]. However, the issue of the distance between objects is not addressed by any of these methods (e.g., [3, 6]). In addition, it is assumed that the database images must contain all objects in the query image. Thus, the question of contextual and spatial ambiguity in its full extent is not considered. Furthermore, none of these methods provide Boolean combinations or negations of query images.

This paper presents a pictorial query specification technique that we have developed that addresses the issue of matching, contextual, and spatial ambiguity inherent in pictorial queries. It is organized as follows. Section 2 describes the user interface for specifying pictorial queries in an example application as well as the process that we use for matching query-image objects to database-image objects in this application. In Section 3 we show how to resolve the matching, contextual, and spatial ambiguities inherent in pictorial queries. Section 4 shows how individual pictorial queries are combined to form compound queries along with examples of their use. Section 5 contains concluding remarks.

2 Example of a Pictorial Query Specification

In our approach, a pictorial query is composed of one or more query images. Each query image is constructed by selecting the symbols that should appear in the database images from a menu of symbols and by positioning these symbols so that the desired spatial constraints hold. In addition, the user must specify the image similarity level required to satisfy the matching, contextual, and spatial constraints between the query image and the required database images. The individual query images may be composed via AND and OR operators. In addition, a query image can be negated with the NOT operator in order to specify conditions that should not be satisfied by the database images that are retrieved successfully. In the case of the conjunction of query images where the same symbol appears in both query images, the user may specify whether the two query-symbols must match (i.e., be bound to) the same instance of the symbol in the database image, or whether two different instances are allowed (i.e., the instances may be different but need not be so)[1]. In order to bind two query-symbols to the same database symbol, the user selects the symbol for the second query image from the first query image, rather than selecting it from the menu of symbols.

We have implemented this pictorial query specification as a query interface for a map image database system that we have developed [8] named MARCO (denoting MAp Retrieval by COntent). The input to MARCO are raster images of separate map layers (*map layer images*) and raster images of *map composites* (the maps that result from composing the separate map layers). Map layer images are processed in order to extract contextual cues from the map layer that can be used to index the composite images. This process utilizes the symbolic knowledge found in the legend of the map to drive geographic symbol recognition. In particular, we focused on symbol layers which contain geographic symbols that represent campsites, hotels, etc.

This input process requires some user intervention in order to build an initial training set. Once this is done, the current training set library is used to assign candidate classifications to each symbol using a *weighted bounded several-nearest neighbor classifier* [2]. A certainty value (between 0 and 1) is attached to each classification indicating how certain this classification is. In cases where there is more than one possible classification, all candidate classifications are returned by the classifier with their associated certainty value and stored in the database. Classification is based on a set of *features* that describe the symbol's shape. thereby enabling us to resolve the matching ambiguity (i.e., to match query-image symbols to database-image symbols). However, for other applications we could use different features for this purpose.

Figure 1 shows the pictorial query builder used by MARCO. The user wants to retrieve all database images that contain a hotel ⊡ within 6 miles of a beach ⊜ and do not have an airport ⊛ within 1 mile of the beach ⊜ (see Figure 2 for a description of the symbols used in this query and in the rest of this paper). Furthermore, the certainty that the database-image symbols are in fact a hotel ⊡, beach ⊜, and airport ⊛ is ≥ 0.5. The symbols are "dragged and dropped" from the menu of symbols displayed in the bottom of the window. The query builder constructs this menu of symbols directly from the database which stores one example of each symbol relevant for the application at hand.

[1]Binding is irrelevant in the case of disjunction of query images, since only one part of the clause needs to hold for the query to be satisfied.

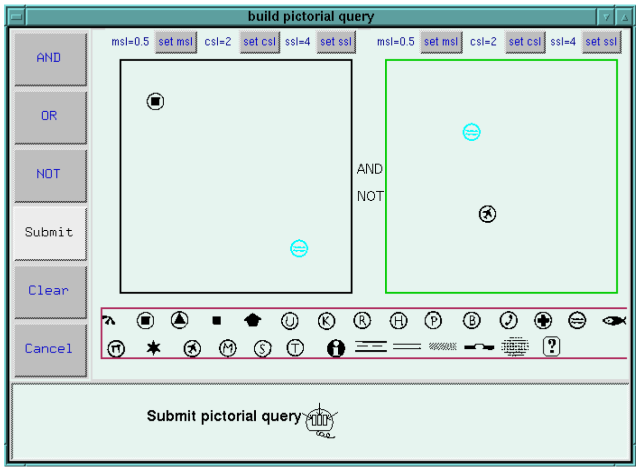


Figure 1. Pictorial query construction tool: The user has constructed a query to "retrieve images that contain a hotel ▣ within 6 miles of a beach ⊜ and do not have an airport ⊛ within 1 mile of the beach ⊜."

These example symbols are taken from the legend of the map in our example application. Alternatively, provisions exist for the user to import examples of symbols directly. Thus, the interface can automatically adjust to a different set of symbols. We use a color coding scheme to denote that two query-image symbols are bound to the same instance in the database image. That is, two symbols that have the same non-black color are bound, whereas black symbols are not bound. For example, in the query in Figure 1, the two beach ⊜ symbols are blue, and thus they are bound. That is, the same instance of the database-image beach ⊜ symbol must be matched to the query-image beach ⊜ symbols in both clauses of the pictorial query. Matching, contextual, and spatial similarity levels are set via menu buttons "set msl", "set csl", and "set ssl", respectively (see Figures 3– 4 for the corresponding menus).

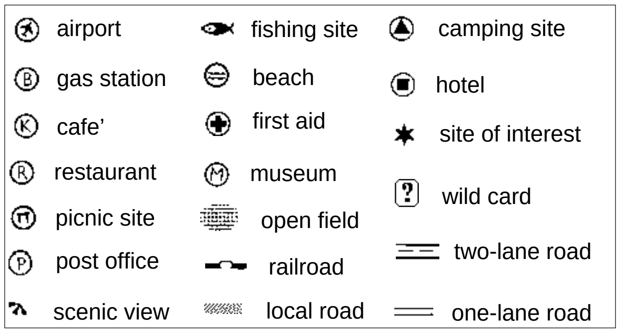

airport
fishing site
camping site
gas station
beach
hotel
cafe'
first aid
site of interest
restaurant
museum
wild card
picnic site
open field
post office
railroad
two-lane road
scenic view
local road
one-lane road

Figure 2. Semantic meanings of symbols.

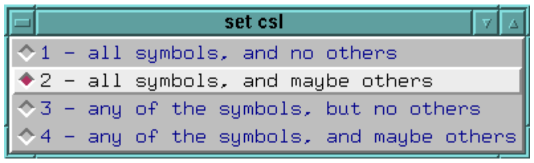

set csl
1 - all symbols, and no others
2 - all symbols, and maybe others
3 - any of the symbols, but no others
4 - any of the symbols, and maybe others

Figure 3. Menu for setting contextual similarity level (csl).

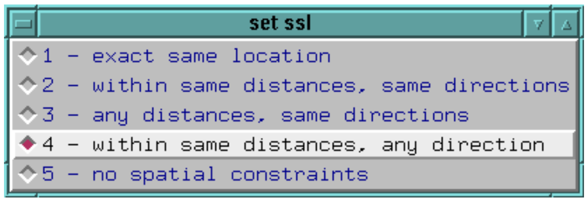

set ssl
1 - exact same location
2 - within same distances, same directions
3 - any distances, same directions
4 - within same distances, any direction
5 - no spatial constraints

Figure 4. Menu for setting spatial similarity level (ssl).

3 Resolving Matching, Contextual, and Spatial Ambiguity

In this section we describe how the matching, contextual, and spatial ambiguity inherent in pictorial queries is resolved by explicitly specifying the required level of similarity between query image *QI* and database image *DI* in these three domains. We define similarity using the following definitions. A *symbol* s is a group of connected pixels that together have a common semantic meaning. A *class* C is a group of symbols that all have the same semantic meaning.

3.1 Matching Similarity

Matching similarity specifies how close a match between a symbol s_1 in the query image *QI* and a symbol s_2 in the database image *DI* is required in order to consider them to be the same. 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. In other words, if the certainty that s_1 and s_2 are from the same class $\geq$ *msl*, then s_1 and s_2 will be considered a match.

3.2 Contextual Similarity

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 (e.g., Figure 5):

1. Each symbol in *QI* has a distinct matching symbol in *DI*, and each symbol in *DI* has a matching symbol in

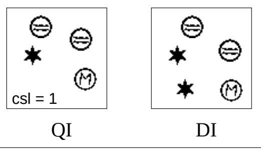
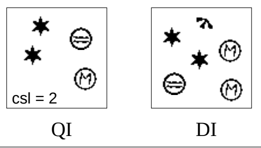
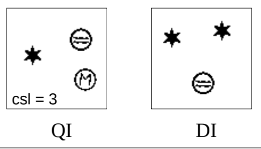
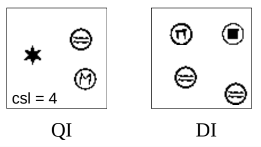
1. Every symbol in QI has a distinct matching symbol in DI, and every symbol in DI has a matching symbol in QI	
2. Every symbol in QI has a distinct matching symbol in DI (DI may contain additional symbols from any class)	
3. Every symbol in DI has a matching symbol in QI	
4. At least one symbol in QI has a matching symbol in DI (DI may contain additional symbols from any class)	

Figure 5. Contextual similarity levels (csl).

QI.

2. Each symbol in *QI* has a distinct matching symbol in *DI* (*DI* may contain additional symbols from any class).
3. Each symbol in *DI* has a matching symbol in *QI*.
4. At least one symbol in *QI* has a matching symbol in *DI* (*DI* may contain additional symbols from any class).

3.3 Spatial Similarity

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. To define spatial similarity levels, we need to distinguish between various spatial symbol types. In our application, a symbol may correspond to a point (e.g., a museum), a line (e.g., a local road), or a polygon (e.g., an open field). The location of a symbol $loc(s)$ is defined as follows:

$$\begin{cases} \text{the } (x,y) \text{ coordinates of the center of gravity of } s, & \text{when } s \text{ is a point} \\ \text{the } (x,y) \text{ coordinates of the end points of } s, & \text{when } s \text{ is a line} \\ \text{the } (x,y) \text{ coordinates of the upper-left and bottom-right corners of the minimum bounding rectangle of } s \text{ whose sides are parallel to the axes}, & \text{when } s \text{ is a polygon} \end{cases}$$

The distance between two symbols $dist(s_1, s_2)$ is defined as the Euclidean distance between s_1 and s_2. $dist(s_1, s_2) = 0$ when the two symbols intersect (e.g., a line symbol intersects a polygon symbol). $dist(s_1, s_2) = -\infty$ if one symbol is totally enclosed in the other (e.g., a line symbol is inside a polygon symbol). For example, the distance between a line l_1 and a polygon p_1 represented by its minimum bounding rectangle r_1 is defined as follows:

$$\begin{cases} 0, & \text{when } l_1 \text{ intersects } r_1 \\ -\infty, & \text{when } l_1 \text{ is inside } r_1 \\ dist(l_1, l_2), \text{where } l_2 \text{ is the edge of } r_1 \text{closest to } l_1 & \text{otherwise} \end{cases}$$

Let $rel(s_1, s_2)$ denote the relative position of symbol s_2

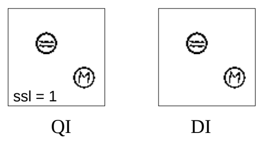
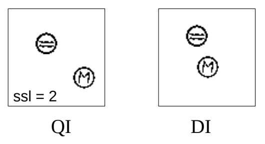
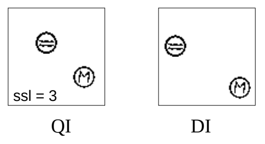
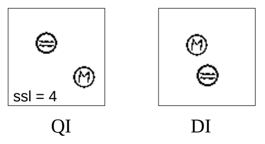
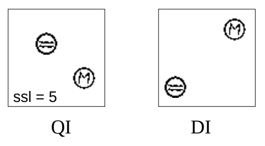
1. Matching symbols are in the same location	
2. Same relative position of matching symbols and distance between symbols in DI > L and <= distance between matching symbols in QI (L=0 by default)	
3. Same relative position of matching symbols but distance between matching symbols may vary	
4. Relative position of matching symbols varies but distance between symbols in DI > L and <= distance between matching symbols in QI (L=0 by default)	
5. Distance and relative position between matching symbols may vary (no spatial constraints)	

Figure 6. Spatial similarity levels (ssl).

with respect to symbol s_1. In our implementation, the function $rel(s_1, s_2)$ can take on one of the following values: N,NW,W,SW,S,SE,E,NE,C where N,W,S,E are the four cardinal directions, NW,NE,SW,SE are the diagonal directions, and C denotes coincidence. The definition of $rel(s_1, s_2)$ is in terms of $loc(s_1)$ and $loc(s_2)$, and varies depending on the types of the argument symbols. For example, assuming an origin in the upper-left corner, for point symbols: $(loc_x(s_j) < loc_x(s_i) \wedge loc_y(s_j) < loc_y(s_i)) \Rightarrow NW(s_j, s_i)$. Whenever the two symbols coincide, $rel(s_1, s_2) = C$.

We make use of five levels of spatial similarity (e.g., Figure 6). They differ in the degree of freedom in the relative location of matching symbols in the two images. The spatial similarity level enables specification of the required database images in terms of minimum and maximum distance between symbols, and their relative directions.

1. The matching symbols of *QI* and *DI* are in the exact same locations in both images.

2. The relative position of the matching symbols of *QI* and *DI* is the same, and the distance between them is bounded from below by some given value L and bounded from above by the distance between the symbols in *QI*. By default $L = 0$. If $L = 0$, then $0 \leq dist(s_i, s_j) \leq dist(s_k, s_l)$ (i.e., it is a range search). If $L = dist(s_k, s_l)$, then $dist(s_i, s_j) = dist(s_k, s_l)$ (i.e., it is an exact distance search).

3. The relative position of the matching symbols of *QI* and *DI* is the same, but the distance between them may vary.

4. The relative position of the matching symbols of *QI* and *DI* may vary, but the distance between them is bounded from below by some given value L and bounded from above by the distance between the symbols in *QI*. By default $L = 0$.

5. The location of the matching symbols, the distance between them, and the relative position of these symbols may vary (i.e., no spatial constraints).

3.4 Total Image Similarity

The total similarity between *QI* and *DI* is defined by combining the three similarity factors. For example, $DI \equiv_{0.5,2,3} QI$ specifies that the matching, contextual, and spatial similarity of the two images is at levels 0.5, 2, and 3, respectively. That is, for each symbol in *QI* there is a matching symbol with a certainty ≥ 0.5 from the same class in *DI*, the location of the symbols and the distance between them may vary, but the inter-symbol spatial relationship between them is the same. In general, if $DI \equiv_{msl,csl,ssl} QI$ and if S' is the set of all the symbols of DI that match some symbol in QI with a certainty $\geq msl$, then the set of classes of the symbols of S' is a subset of the set of classes of the symbols of QI. Also, for every pair of symbols s_1 and $s_2 \in S'$, the spatial constraints dictated by ssl and the positions of the matching symbols in QI hold. In other words, the spatial constraints must simultaneously hold between all of the matching symbols that appear in both query and database images.

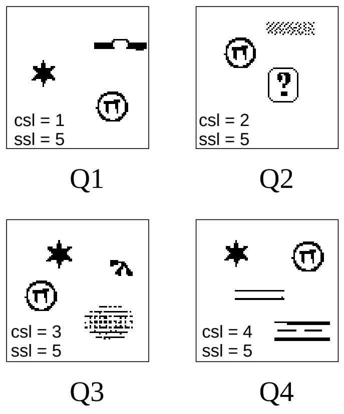


Figure 7. Examples of different contextual similarity levels. The question mark $\boxed{?}$ symbol denotes a wild card (i.e., any symbol matches it).

3.5 Example Queries varying csl

Figure 7 demonstrates the use of different contextual similarity levels for query specification. In all of these queries we assume that $ssl = 5$ (i.e., no spatial constraints are imposed). We do not specify *msl* in these or any other example queries since its use is straightforward and does not require further illustration. Query Q1 requests all images that contain a site of interest, a picnic site, a railroad, and no other symbols. Query Q2 requests all images that contain a picnic site, a local road and at least one other arbitrary symbol (wild card $\boxed{?}$, there may be more). Query Q3 requests all images that contain a site of interest, or a scenic view, or a picnic site, or an open field (an image may contain all four) but no other symbols. Query Q4 requests all images that contain a site of interest, or a picnic site, or a one-lane road, or a two-lane road (an image may contain one or all of them as well as other symbols).

3.6 Example Queries varying ssl

Figure 8 demonstrates the use of different spatial similarity levels for query specification. In all of these queries we assume $csl = 2$ (i.e., every symbol in the query image has a distinct matching symbol in the database image). Query Q1 requests all images that contain a picnic site within 1 mile of an open field. Query Q2 requests images with a hotel and any arbitrary symbol (wild card $\boxed{?}$) within 1 mile and southeast of the hotel. Query Q3 requests all images that contain an airport northwest of a beach. Query Q4 requests images that contain a picnic site within 1 mile of a one-lane road and within 3 miles of a scenic view,

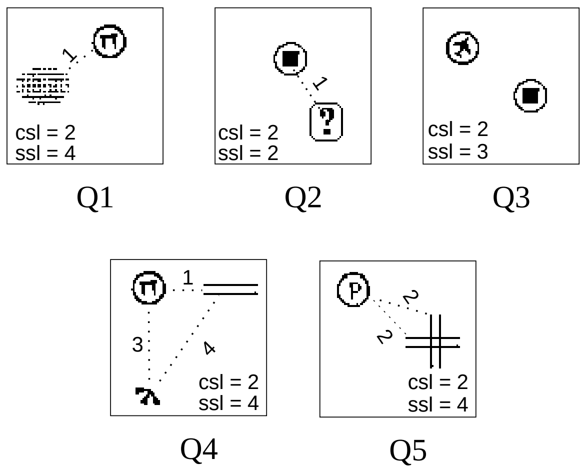


Figure 8. Examples of different spatial similarity levels. "csl" denotes contextual similarity level, "ssl" denotes spatial similarity level. The question mark symbol denotes a wild card (i.e., any symbol matches it).

and requires that the scenic view be within 4 miles of the one-lane road. Query Q5 requests images that contain a post office within 2 miles of of two one-lane roads that intersect. Note that the dotted lines with the distance label that appear in the query images in Figure 8 are only used to denote the distance between symbols in the figure; they are not actually part of the query image. The query image only contains symbols. The distance (and relative directions) between the symbols is specified implicitly in the query image *QI* by the actual distance (and relative direction) between the symbols in *QI* provided that the spatial similarity level specifies that they are to be taken into account in computing the response to the query.

4 Compound Queries and Negation

So far we have described how to construct an individual query image and resolve the matching, contextual, and spatial ambiguity by specifying *msl*, *csl*, and *ssl*. Now we show how to add expressive power to our pictorial specifications via compound queries and negation. A pictorial specification may be composed of several query images. The query images can be joined with AND and OR operators. In addition, a query image can be negated with the NOT operator. The semantic meaning of these operators is as follows:

$$[DI \equiv (QI_1 \text{ OR } QI_2)] \Longrightarrow (DI \equiv QI_1) \vee (DI \equiv QI_2)$$

$$[DI \equiv (QI_1 \text{ AND } QI_2)] \Longrightarrow (DI \equiv QI_1) \wedge (DI \equiv QI_2)$$

$$[DI \equiv \text{NOT } (QI)] \Longrightarrow \neg(DI \equiv QI)$$

In the case of the 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*). In order to bind two query-symbols to the same database symbol, the user selects the symbol for the second query image from the first query image, rather than selecting it from the menu of symbols.

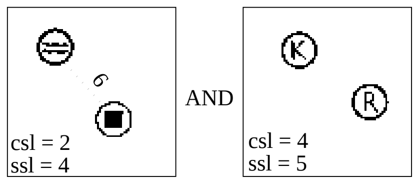


Figure 9. "Display images with a hotel within 6 miles of a beach and with a cafe or a restaurant ".

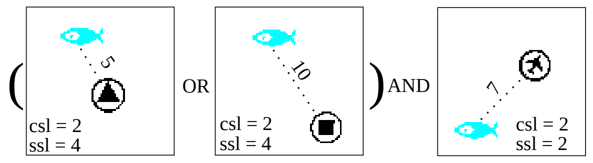


Figure 10. "Display images with a camping site within 5 miles of a fishing site OR with a hotel within 10 miles of a fishing site AND with an airport northeast of and within 7 miles of the fishing site ".

Compound queries can be used to specify more complex queries. In particular, two separate query images with different values of *csl* and *ssl* can be combined via the AND operator to specify a query with spatial constraints between some symbols, but with no spatial constraints between other symbols. For example, consider the query in Figure 9 which requests "all images with a hotel within 6 miles of a beach and with a cafe or a restaurant ". No spatial constraints are specified for the restaurant and cafe symbols; however, the hotel must be within 6 miles of a beach. Notice that each query image component has a different *csl* value associated with it. Thus, the first component requests images containing both symbols, whereas the second component requests images containing either symbol. Compound queries can also be used to specify more than one acceptable spatial constraint. For example, the query "display all images with a camping site within 5 miles of a particular fishing site OR with a hotel within 10 miles of the same fishing site AND with an airport northeast of and within 7 miles of the same fishing site " can be specified as shown in Figure 10. Recall that we use a color coding scheme to denote that two query-image symbols are bound to the same instance in the database image (i.e., the fishing site in this example). That is, two symbols that have the same non-black color are bound, whereas black symbols are not bound.

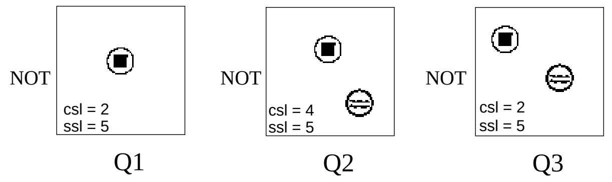


Figure 11. Examples of negation. Q1: "images with no hotel "; Q2: "images that do not have both a beach and a hotel "; Q3: "images that do not have a beach or do not have a hotel ".

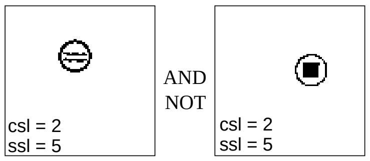


Figure 12. "Display images with a beach but with no hotel ".

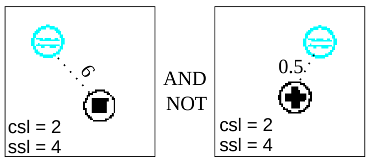


Figure 13. "Display images with a hotel within 6 miles of a beach and with no first aid station within 0.5 mile of the beach ".

Negation of queries can be used in order to specify conditions that should not be satisfied by the database images that are retrieved successfully. Figure 11 demonstrates how negation can be used to express retrieval of images that do not contain a particular symbol, a pair of symbols, or one of two symbols. Query Q1 requests "images with no hotel ", Query Q2 requests "images that do not have both a beach and a hotel ", while query Q3 requests "images that do not have a beach or do not have a hotel ". Negation in conjunction with compound queries can be used to specify

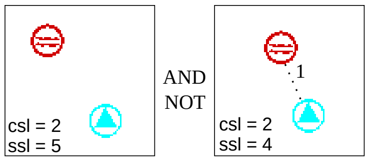


Figure 14. "Display images with a camping site further than 1 mile from a beach ".

both positive and negative conditions as is the case for the query in Figure 12 which requests "images that do have a beach , but do not have a hotel ". Compound queries with symbol binding and negation can be used to specify more than one spatial condition for the same symbol as is the case for the query in Figure 13 which requests "all images with a hotel within 6 miles of a beach and with no first aid station within 0.5 mile of the beach ". Another application of compound queries with symbol binding and negation is to to specify distance constraints in terms of an upper bound. For example, the query in Figure 14 requests "all images with a camping site further than 1 mile from a beach ".

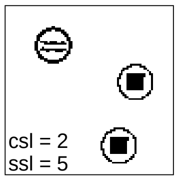


Figure 15. A pictorial query to " display images that contain a beach and at least two hotels ".

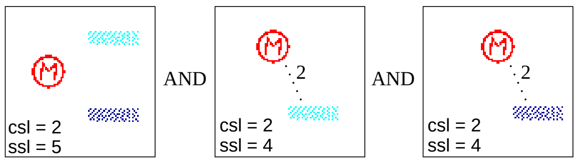


Figure 16. A pictorial query to "display images with two different local roads within 2 miles of a museum ".

In the example queries that we have seen so far there is only one instance of each symbol in each query image. Our pictorial specification method does however allow multiple instances of each symbol. This is useful in order to specify the number of occurrences of a particular symbol that are required in a target database image. According to the definition of *csl* (see Figure 5), if *csl* is set to 1 or 2, then every symbol in *QI* has a distinct matching symbol in *DI*. Thus, if there are two instances of a symbol in *QI*, and *csl* is set to 1 or 2, then there must be at least two instances of this symbol in *DI* for it to satisfy the query. For example, the query in Figure 15 requests "images that contain a beach ⊜ and at least two hotels ◙". Another use of compound queries with multiple instances of symbols is to impose the constraint that two query-image symbols from the same class must be matched to two different instances of this symbol in the database image. This is demonstrated by the query in Figure 16, which requests "all images with two different local roads ▨ within 2 miles of a museum Ⓜ".

5 Concluding Remarks

A pictorial query specification technique that enables the formulation of complex pictorial queries for image databases has been described. Using this technique, it is possible to specify which objects should appear in the target images as well as how many occurrences of each object are required. Moreover, spatial constraints can be imposed that specify bounds on the distance between objects, as well as the relative direction or orientation between objects.

While it is possible to express rather complex queries using our method, there are some conditions that cannot be specified. In particular, we cannot specify conditions involving the location of certain events between objects. For example, in Figure 8, we showed how to specify the condition "post office Ⓟ within 2 miles of two one-lane roads ═ that intersect". However, we cannot specify that we want the post office Ⓟ to be within 2 miles of the point where these two one-lane roads ═ intersect. In addition, although we take the extent of objects into account in distance and relative position computations, 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 a subject for future research.

6 Acknowledgments

We are grateful to Karttakeskus, Map Center, Helsinki, Finland for providing us the map data. We have benefited from discussions with Walid G. Aref.

References

[1] A. Del Bimbo and P. Pala. Image indexing using shape-based visual fetaures. In *Proceedings of the 13th International Conference on Pattern Recognition*, volume III, pages 351–355, Vienna, Austria, August 1996.

[2] J.L. Blue, G.T. Candela, P.J. Grother, R. Chellappa, and C.L. Wilson. Evaluation of pattern classifiers for fingerprints and OCR applications. *Pattern Recognition*, 27(4):485–501, April 1994.

[3] 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.

[4] V. Gudivada and V. Raghavan. Design and evaluation of algorithms for image retrieval by spatial similarity. *ACM Transactions on Information Systems*, 13(2):115–144, April 1995.

[5] W. Niblack, R. Barber, W. Equitz, M. Flickner, E. Glasman, D. Petkovic, and P. Yanker. The QBIC project: Querying images by content using color, texture, and shape. In *Proceeding of the SPIE, Storage and Retrieval of Image and Video Databases*, volume 1908, pages 173–187, San Jose, CA, February 1993.

[6] D. Papadias, T. Sellis, Y. Theodoridis, and M. J. Egenhofer. Topological relations in the world of minimum bounding rectangles: a study with r-trees. In *Proceedings of the ACM SIGMOD Conference*, pages 92–103, San Jose, CA, May 1995.

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

[8] H. Samet and A. Soffer. MARCO: MAp Retrieval by COntent. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(8):783–798, August 1996.

[9] A. P. Sistla, C. Yu, and R. Haddad. Reasoning about spatial relationships in picture retrieval systems. In J. Bocca, M. Jarke, and C. Zaniolo, editors, *Proceedings of the 20th International Conference on Very Large Data Bases*, pages 570–581, Santiago, Chile, September 1994.

[10] A. W. M. Smeulders and R. Jain, editors. *Proceedings of the First International Workshop on Image Databases and Multi Media Search*, Amsterdam, The Netherlands, August 1996.