2D Projection Interval Relationships: A Symbolic Representation of Spatial Relationships

Mohammad Nabil*, John Shepherd and Anne H.H. Ngu

School of Computer Science and Engineering
University of New South Wales
Sydney 2052, Australia
{nabil,jas,anne}@cse.unsw.edu.au

Abstract. Spatial relationships are important ingredients for expressing constraints in retrieval systems for spatial databases. In this paper we propose a unified representation of spatial relationships, 2D Projection Interval Relationships (2D-PIR), that integrates both directional and topological relationships. We propose a graph representation for pictures that is based on 2D-PIR. This graph representation can be constructed efficiently and leads to an efficient algorithm for "picture matching".

1 Introduction

Spatial relationships lie at the heart of spatial reasoning and form a vital part of spatial query languages. Two major kinds of spatial relationships have attracted the attention of researchers: directional relationships such as *left_of, right_of, above, below,* etc., (or in terms of *east, west, north, south*, etc.) and topological relationships such as *disjoint, touch, overlap, covers*, etc. Such relationships are used as constraints in spatial query expressions. For example, the following queries explicitly use directional relationships as constraints to retrieve an object: "find a gas station which is located west of a region in a map" or "find a picture containing a computer on the top of a table to the right of a bookcase".

To the best of our knowledge, very few attempts have been made in the literature to reason about directional and topological relationships under a unified representation. However, integration of directional and topological relationships is very useful, since it gives more expressive power than each individual kind of relationship alone. For example, with only directional relationships, we can say that The Netherlands are to the west of Germany, and with only topological relationships, we can say that Germany and The Netherlands have a common border (*meet*), but until we can deal with both kinds of relationship we cannot reason about The Netherlands being a western neighbour of Germany.

This paper describes 2D Projection Interval Relationships (2D-PIR), a simple but powerful representation which provides a unified framework for spatial

* On leave of absence from the Department of Agroindustrial Technology, Bogor Agricultural University; currently at School of Computer Science and Engineering– UNSW

(directional and topological) relationships. It is inspired by Allen's and Chang's work in [1] and [2]. This representation not only integrates directional and topological relationships but also describes detailed directional relationships between spatial objects. For example, using traditional directional relationships it is difficult to express directional relationships between The Netherlands and Germany since there exist parts of Germany which are west of, east of, north, as well as south of The Netherlands. By using interval relationships 2D-PIR we can express the spatial relationship between The Netherlands and Germany more completely and more precisely. Note that 2D-PIR is a symbolic representation: each spatial object is represented by a symbol. A symbolic picture is a digraph where nodes of the graph are symbols representing spatial objects and edges of the graph are labelled by 2D-PIR relationships.

This paper only considers 2D spatial objects (regions); however it is easy to extend to higher dimensional spatial objects. The core idea is that regions are projected along the x and y axes and then, using Allen's operators, spatial relationships among objects are encoded. This method has several advantages: (a) it is simple (that is, to derive a 2D-PIR representation is straightforward), (b) it is an efficient representation in terms of storage, (c) it can be as a basis for efficient picture matching (both exact and similarity matching) and spatial reasoning can be developed on top of this representation.

The remainder of this paper is organised as follows. Section 2 presents related work. In Section 3 we discuss Allen's interval relationships and Chang's 2D-strings. We then describe our model for representing spatial relationships. Section 4 discusses how we apply this model for picture retrieval. Section 5 discusses some possible extensions and applications. Finally we present our conclusions and suggestions for future development in Section 6.

2 Related Work

2.1 Directional Relationships

Directional relationships are heavily used in everyday life. We frequently mention an object based on its directional relationship with another object. For example, a mouse is to the right of a keyboard or Munich is north of Rome. According to [15] and [16] directional relationships are fuzzy concepts since they depend on human interpretation. Note also that directional relationships are influenced by the angle of view and orientation of objects.

Peuquet and Ci-Xiang [15] have developed a model for directional relationships between polygons by incorporating size, shape and distance. They also provide an algorithm to determine directional relationships between polygons in 2D space. Some characteristics of their directional relationships are: they are binary, each direction is coupled with a semantic inverse, the area of acceptance for any given direction increases with distance. Some other characteristics are that the area of acceptance for any given direction increases with the dimension in the facing direction of the reference object in relation to the second object,

and in the selection between a pair of objects as the reference, the one which is "perceptually prominent" (the larger one) is preferred. In [16], fuzzy concepts were introduced into directional relationships, to account for the fact that such relationships are not necessarily exclusive. For example, if an object is above and to the left of another object, it may be perceived as being *above*, $left\_of$, or some degree of both of these.

2.2 Topological Relationships

Topological relationships are relationships which are invariant under topological transformation (that is, they are preserved if the objects are translated, rotated, or scaled) [4].

Egenhofer [5] uses the concepts of *boundary* and *interior* of pointsets to derive topological relationships between two-dimensional regions (pointsets) embedded on a two-dimensional plane (R^2). If A and B are two regions, A^0 and ∂A denote the interior and boundary of A respectively, then all of the possible topological relationships between A and B can be derived from the possible combinations of intersection between their boundaries and interiors; that is $\partial A \cap \partial B, A^0 \cap B^0, \partial A \cap B^0$, and $A^0 \cap \partial B$. Each of these intersections is either empty (0) or non-empty (1). This results in 16 possible combinations as shown in Table 1.

	$\partial A \cap \partial B$	$A^0 \cap B^0$	$\partial A \cap B^0$	$A^0 \cap \partial B$	Relationships' Name
r_0	0	0	0	0	A and B are *disjoint*
r_1	0	0	0	1	*
r_2	0	0	1	0	*
r_3	0	0	1	1	*
r_4	0	1	0	0	*
r_5	0	1	0	1	A *contains* B or B *insides* A
r_6	0	1	1	0	A *insides* B or B *contains* A
r_7	0	1	1	1	A *overlaps* B (disjoint boundary)
r_8	1	0	0	0	A *meets* B
r_9	1	0	0	1	*
r_{10}	1	0	1	0	*
r_{11}	1	0	1	1	*
r_{12}	1	1	0	0	A *equals* B
r_{13}	1	1	0	1	A *covers* B
r_{14}	1	1	1	0	A *covered_by* B
r_{15}	1	1	1	1	A *overlaps* B (intersecting boundary)

*) indicates there is no topological relationship (proved in [5]).

Table 1. Topological relationships created from four intersections

There are nine valid topological relationships between two regions. But r_5 and r_6, r_{13} and r_{14} are inverses of each other. There are two *overlaps* relation-

ships (r_7 and r_{15}) since disconnected boundaries are allowed. If boundaries are assumed to be connected, then r_7 is excluded. Hence, there are eight topological relationships: *disjoint, contains, insides, overlaps, meets, equal, covers* and *covered_by*. Note that these relationships are mutually exclusive.

3 2D Projection Interval Relationships

A 2D Projection Interval Relationship (2D-PIR) is a symbolic representation of directional as well as topological relationships among spatial objects. It adapts two existing representation formalisms (Allen's temporal intervals and 2D-strings) and combines them in a novel way to produce the unified representation. Note that in this paper we only deal with 2-dimensional spatial objects; higher dimensional spatial objects are a subject for future research, but appear to be a straightforward extension of the ideas presented here. The basic idea is that each spatial object is projected along the x and y axes forming an x-interval and y-interval for the object. Using the intervals and Allen's 13 interval relationships, spatial relationships are established. Before we discuss our method, we review Allen's interval relationships and 2D-strings.

Allen [1] proposed an interval-based temporal representation and introduced a method to derive relationships between intervals. Intervals can be represented by their endpoints assuming that every interval consists of a fully ordered set of points of time. Thus an interval is represented as an ordered pair of points with the first point less then the second, and so on. Based on this representation, thirteen relationships (seven relationships have inverses, one relationship has no inverse) are derived as shown in Table 2.

Relationship	Symbol	Symbol for Inverse	Pictorial Example
X before Y	<	>	XXXXX YYYYY
X equal Y	=	=	XXXXX YYYYY
X meet Y	m	mi	XXXXXYYYYY
X overlaps Y	o	oi	XXXXX YYYYY
X during Y	d	di	XXXXX YYYYYYYYYY
X starts Y	s	si	XXXXX YYYYYYYYYY
X finishes Y	f	fi	XXXXX YYYYYYYYYY

Note: X and Y are intervals

Table 2. Allen's interval relationships

A 2D-string is a symbolic representation of a picture originally suggested by Chang et al. [2]. The representation is constructed from the original picture by stating directional relationships relative to a grid superimposed on the picture. An example is shown in Figure 1, where the upper-case letters represent spatial objects lying in the region of the plane bounded by each grid cell. Note that object A is large enough that it spans two cells.

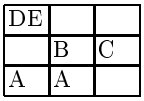

Fig. 1. A symbolic picture

In Figure 1, we can straightforwardly determine directional relationships among objects. For instance, object B is on the *left_of* object C, object D is in the same set as (i.e. "near to") object E, and object C is at the *upper_right* of object A.

A 2D-string is a pair made up of the "symbolic projections" of a picture along the x-axis and y-axis. Construction of a 2D-string for a picture commences by laying a grid over the picture; the idea is that objects are partitioned over the cells of the grid (as in Figure 1). To generate a 2D-string from a picture, we scan along the x-axis, looking at the projections of objects on this axis, and describing the relationships between the objects using the operators "=", "<" and ":" ("=" represents the relationship "at the same projected location as", "<" represents the relationship "to the left of" and ":" represents "in the same set as" relation (note that ":"⇒"=")). For example, in Figure 1, the x-axis relationships are: $A = D : E < A = B < C$. We then repeat this scan along the y-axis, to generate another string of relationships between objects (or, more precisely, their projections on the y-axis). In the example, this corresponds to: $A = A < B = C < D : E$. The 2D-string of the picture is the pair of object relationship strings.

More formally, a 2D-string is defined as follows: Let V be a set of symbols representing spatial objects. Let A be the set {"=", "<", ":"} of spatial relationships where $A \cap V = \emptyset$. A 1D-string over V is any string $x_1 x_2 ... x_n, n \geq 0$, where the x_i are in V. A 2D-string over V, written as (u, v) is defined to be $x_1 y_1 x_2 y_2 ... y_{n-1} x_n$, $x_{p(1)} z_1 x_{p(2)} z_2 ... z_{n-1} x_{p(n)}$, where $x_1 ... x_n$ is a 1D-string over V, $p : \{1, ..., n\} \rightarrow \{1, ...n\}$ is a permutation over $\{1, ...n\}$, $y_1, ... y_{n-1}$ is a 1D-string over A and $z_1, ... z_{n-1}$ is a 1D-string over A. For Figure 1, V = {A,B,C,D,E}. The 2D-string representing it is $(A = D : E < A = B < C, A = A < B = C < D : E)$ which is called an *absolute* 2D-string. If the symbol "=" is omitted then the 2D representation becomes $(AD : E < AB < C, AA < BC < D : E)$ which is called a *normal* 2D-string. If the symbol ":" is also omitted then the 2D-string becomes $(ADE < AB < C, AA < BC < DE)$ which is called

a *reduced* 2D-string. The *augmented* 2D-string is the *reduced* 2D-string where a permutation function is included in 2D string representation. The *augmented* 2D-string of the above picture is $(ADE < AB < C, AA < BC < DE, 145623)$. The integer 145623 is a permutation function which represents positions of symbols in the second string in respect to the first string. For example the first A of the second string occupies the first position in the first string and the second A of second string occupies the fourth position in the first string. The *absolute* 2D-String provides a precise encoding of a picture, but is not a particularly efficient encoding scheme. The *normal* 2D-string provides more efficient encoding and the *reduced* 2D-string provides the most efficient encoding. However, in a *reduced* 2D-string the absolute positioning information is lost. The *augmented* 2D-string preserves positioning information so that picture reconstruction is not ambiguous.

There are several extensions of 2D-strings, but the main idea remains that a picture is partitioned over a grid before the 2D-string is generated. For example, in [11] a picture is partitioned both parallel to the x and y coordinate axes, where the partitions are performed at all the extreme points of all objects in the picture.

2D-PIR also uses the projection concept from the 2D-string representation. It differs from the 2D-string representation in using Allen's interval relationships over the projections of the entire objects along the x and y axes, rather than $\{<, :, =\}$. There is thus no need to partition the image initially. A consequence of this approach, however, is that we must use a *graph*, rather than a *string*, to represent the relationships among the objects in a picture.

Using Allen's operators (interval relationships), directional relationships between intervals, which are the projections of objects in both the x and y axes (interval projection relationships), can be generated. Furthermore they can also be used to capture topological relationships, especially for rectangular objects. The main advantage of this method is that it offers more information about spatial relationships between objects in a picture as compared to traditional method. In other words, it has more expressive power than traditional methods. We will illustrate this power later.

A 2D-PIR between two spatial objects A and B is (χ, ψ). The domain of χ and ψ are $\{<, =, m, o, d, s, f, >, mi, oi, di, si, fi\}$ which are symbols for interval relationships. Note that χ represents the interval relationship between objects A and B projected along the x-axis, and ψ represents the interval relationship between objects A and B projected along the y-axis. Figure 2 illustrates this.

In Figure 2 2D-PIR between A and B is $(<, oi)$, between A and C is $(<, m)$ and between B and C is (fi, si). Let us consider these relationships. What kind of information do we get when the 2D-PIR for A and B is $(<, oi)$? First, it tells us that A and B are *disjoint* and A is *left of* B since $x_A < x_B$. Secondly, it informs us that *part of* A is *above* B since y_A *oi* y_B or y_B *o* y_A. Now, we turn to B and C. The 2D-PIR between B and C is (fi, si). The information we can get from this relationship is: 1) B *covers* C (or C *covered by* B), 2) C is located on the bottom-left of B. Thus, we can infer that C is covered by B on the bottom-left.

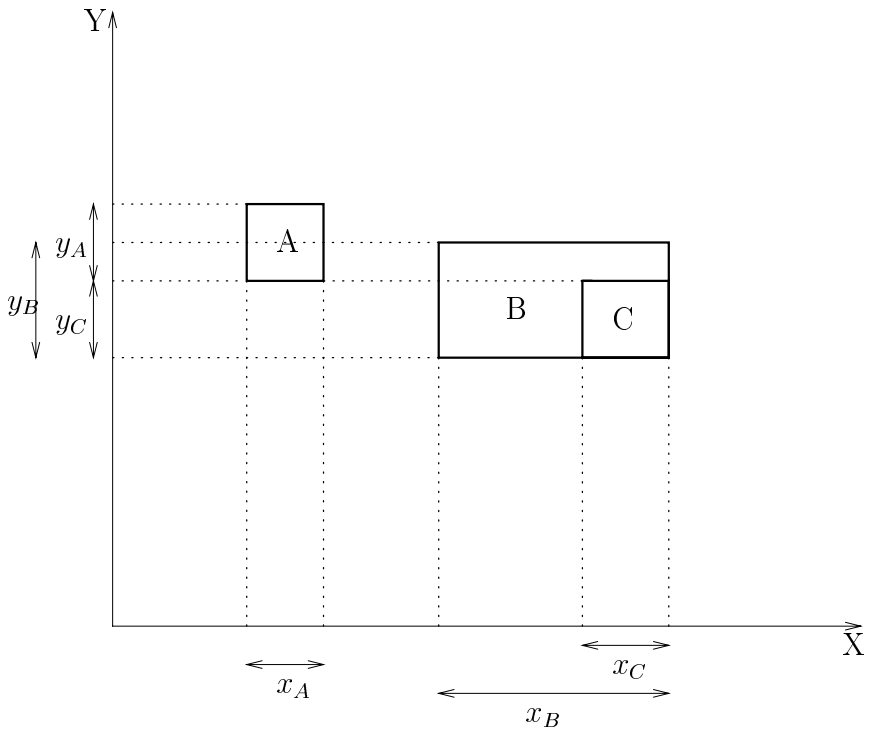

Fig. 2. A 2D-projection picture

The above example shows only rectangular objects whose boundaries are parallel to the x and y axes. The shape of real world objects is rarely so simple or regular. To solve this problem, the 2D-string representation uses a minimum bounding rectangle (MBR) on each object, with rectangle boundaries parallel to horizontal (x) and vertical (y) axes. Adopting this approach effectively means ignoring the orientation of objects. In the case of 2D-PIR, this causes problems, as depicted in Figure 3. In this figure, 2D-PIR between A and B is (di, di). The relationship that would be inferred is B *inside* A (A *contains* B) which is incorrect. There are two alternatives to solve this problem.

The first approach introduces topological relationships into 2D-PIRs. A 2D-PIR is now defined as a triple (δ, χ, ψ) where δ is a topological relationship from the set $\{dt, to, ct, in, ov, co, eq, cb\}$[1]. By introducing these topological relationships, bounding rectangles are not needed. The relationship between A and B will be (dt, di, di) which indicates that A and B are *disjoint* objects and B is

[1] These represent the topological relationships *disjoint, meets, contains, insides, overlaps, covers, equal, covered_by*

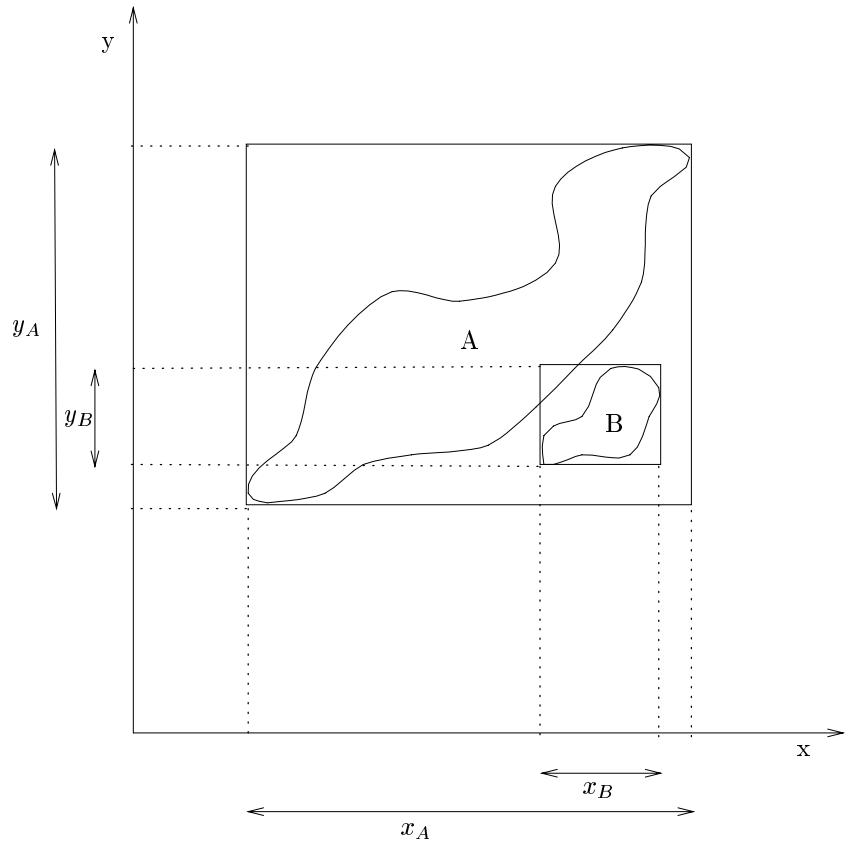

Fig. 3. Another 2D-projection picture

surrounded by A. The triple (dt, di, di) is interpreted as the "surrounded by" relationship, since $\chi = di$ and $\psi = di$ implies that there exists *parts of* A which are *right, left, above* as well as *below* B.

The second approach uses *true* MBRs (that is, MBRs whose boundaries are governed by an orientation angle φ). Each object is surrounded by an MBR, and then a local coordinate system is created whose axes are parallel to the edges of the MBR for the largest object. Next, projection is performed based on a local coordinate system (slope projection) as explained in [9, 10]. Figure 4 is an example of slope projection of objects in Figure 3.

A slope 2D-PIR between two objects is triple (φ, χ, ψ) where χ and ψ are relationships among intervals projected in local coordinate rotated at angle φ relative to the global coordinate system. So a slope 2D-PIR between B and A is $(\varphi, d, <)$ which tells that A is *above* and *disjoint* with B and part of A is *left of*

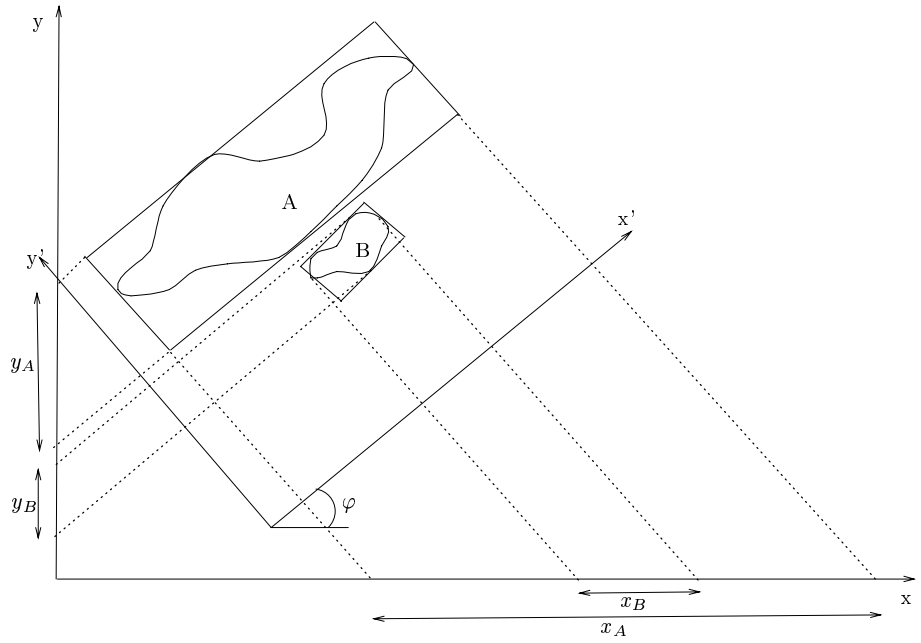

Fig. 4. Slope projection

as well as *right of* B according to the orientation angle φ. Note that the angle φ is available when the MBR of an object is computed.

Both approaches have advantages and disadvantages. The first approach guarantees the correctness of derived spatial relationships. However, it requires extra power to compute the topological relationships. One may refine this approach for objects whose relationship is (dt, di, di) or (dt, d, d) (that is disjoint objects whose MBRs overlap completely). For example, in Figure 3, if B is on the other side of A then the 2D-PIR relationship is still (dt, di, di). To discriminate these situations, the bigger object (in this case A) is divided using vertical strips corresponding to the smaller object's (in this case B) projection on the x-axis (this is also applied on the y-axis). This subdivides A into at most three pieces, where each piece a has 2D-PIR relationship with B. Final relationships between A and B will be a sequence of three 2D-PIR relationships.

The second approach provides a simple solution, since the angle φ is available when the MBR of an object is computed However, it allows incorrect spatial relationships to be derived if the MBR of an object overlaps another object as depicted in Figure 5.

In Figure 5, the spatial relationships between A and B using the first and second approaches are (dt, di, di) and $(45, di, di)$ respectively. Note that $\varphi = 45^0$ means the orientation angle of the referred object is parallel to the x-axis. Note

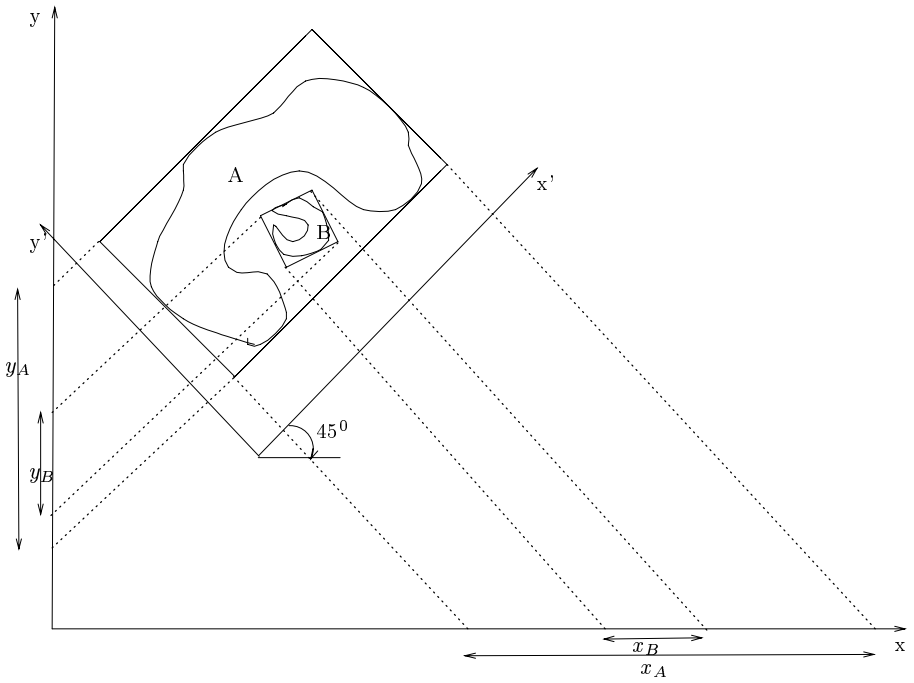

Fig. 5. Picture with MBR objects completely overlap

also that the first approach gives the correct spatial relationship between A and B. A *disjoint* with B and $\chi = di$ and $\psi = di$ can be interpreted as B is *surrounded by* A. The second approach cannot detect that A and B are *disjoint*. The relationship $(45, di, di)$ between A and B is best interpreted as B *inside* A, which is incorrect.

Both approaches are useful for different applications. For example, in multimedia databases, retrieval of images based on *similarity* is a central idea. For this application, both alternatives could be used. The first alternative will guarantee that only correct answers are returned. The second alternative will give more answers and some of them will not be valid answers to the query. In picture similarity retrieval, users generally expect to have to do some post-filtering of query results. However, we prefer to use the first alternative, since it guarantees the correctness of spatial relationships among objects in a picture. Hence, it is possible to develop reasoning mechanisms on it.

The above discussion provides the basis for our symbolic model for pictures. As mentioned previously we choose a directed graph to model a picture based on 2D-PIR relationships among objects in the picture. In other words, our symbolic model for a picture is a network of object symbols and 2D-PIR relationships.

Definition 1. A 2D-PIR graph is a connected labelled digraph $G(V, R)$ where V is a finite non-empty set of symbols representing objects in a picture and R is a set of edges labelled by 2D-PIR relationships.

Definition 2. A symbolic picture model is a A 2D-PIR graph.

We now construct an example 2D-PIR graph. Figure 6 is the 2D-PIR graph of Figure 2 using the first alternative 2D-PIR representation.

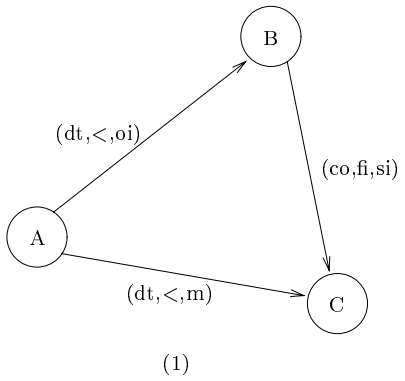

Fig. 6. A 2D-PIR graph

4 Picture Retrieval Using 2D-PIR

In the previous section, we formally defined 2D-PIR graphs. Since a picture is represented by a graph then to decide whether or not two pictures are equivalent or similar it is necessary to compare the corresponding graph representations. The fundamental equality relationship between graphs is known as *graph isomorphism*. Two graphs G_1 and G_2 are *isomorphic* if there exists a 1-1 and onto function $f : V(G_1) \rightarrow V(G_2)$ such that $xy \in E(G_1)$ if, and only if $f(x)f(y) \in E(G_2)$ [8]. Detecting graph isomorphism in the general case is known to be NP-Hard. However, if we construct our 2D-PIR graphs carefully, we can devise efficient processing algorithms for them. The basic idea is to choose a fixed topology for the graph representing a picture. Using the characteristics of directional relationships that were described in the previous section, the graph construction method is simple and straightforward.

4.1 Graph Construction

The graph that represents a picture can be constructed using the following simple algorithm.

Algorithm: Graph construction
input: A preprocessed picture (all objects are recognised and all necessary
information such as topological and projection relationships
have been computed)
output: A directed graph (adjacency list) representing the picture
begin
(1) Arrange objects from left to right and top to bottom (that is objects
in the picture are sorted based on their position left-right and then
top-bottom priority.
(2) Let P be the ordered set of objects in a picture based on the arrangement
(3) in (1) for each object O_i in P
(4) create a linked-list of 2D-PIR relationships with the rest of the objects.
(5) endfor
end.

The time complexity of this algorithm is easily analysed. Lines 1 and 2 are $O(1)$ and lines 4 to 5 are $O(e)$ where e is the number of edges in the graph which is always $(n^2 - n)/2$ (n is the number of objects in the picture). Thus, the complexity of the algorithm in term of objects is $O(n^2)$. Figure 7 is an example of a picture and its corresponding graph (adjacency list) constructed by the algorithm.

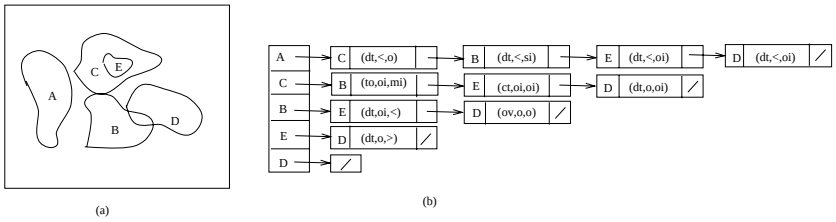

Fig. 7. A picture (a) and the corresponding graph (adjacency list) (b)

4.2 Picture Matching

There are two kinds of picture retrieval: picture retrieval based on exact matching and picture retrieval based on similarity matching. Both of these retrieval techniques can be supported by the 2D-PIR representation using a picture matching algorithm that will be presented shortly. The 2D-PIR representation can retrieve pictures based on sub-picture match (exact and similar) using the same algorithm. Note that if symbolic pictures are constructed using the previous graph construction algorithm, then the picture matching algorithm is simple and straightforward.

Algorithm: Picture matching
input: two 2D-PIR graphs (G_1 and G_2) representing pictures P_1 and P_2.
output: integer 0 : no-match
integer 1: P_1 is a sub-picture of P_2
integer 2: P_1 is equivalent of P_2

```
begin
(1)   if V(G1) = V(G2) then
(2)       for vi ∈ V(G1)
(3)           if linked-list of vi ∈ V(G1) = linked-list vi ∈ V(G2)
(4)               continue
(5)           else return 0 /*no match*/
(6)           endif
(7)           return 2 /* P1 is equivalent with P2 */
(8)       endfor
(9)   elseif V(G1) ⊂ V(G2) then
(10)      for vi ∈ V(G1)
(11)          find vj ∈ V(G2) such that vi = vj
(12)          if each element of linked-list vi is also element of linked-list vj
(13)              continue
(14)          else return 0 /*no match*/
(15)          endif
(16)          return 1 /*P1 is sub-picture of P2*/
(17)      endfor
(18)  else return 0 /*no match*/
(19)  endif
end.
```

A query picture in the above algorithm is a picture or a picture sketch which is used by a user to pose a query. The above algorithm assumes that a query picture contains at most the same number of objects as the number of objects in the intended picture stored in a database. This assumption seems reasonable since users tends to make relatively simple queries containing only a small number of objects. However, the algorithm can be modified easily if a query picture is allowed to be "super-picture" of a database picture. The algorithm can also be extended to deal with a query picture which overlaps a database picture.

This algorithm is dominated by lines 2 to 8 or line 10 to 17, which require $(n^2 - n)/2$ comparisons – that is, $O(n^2)$ in the worst case. Note that n is the number of objects in P_1. It is efficient when compared to the general graph isomorphism algorithm that needs $n^2 n!$ comparisons [8]. It is also better than the 2D-string matching algorithm which is $O(M) + O(N^2 * lp^3)$. Note that both M and N in the 2D-string matching algorithm are the number of entries in matching tables which are greater than the number of objects in a picture (since the 2D-string method uses partition) and lp is the maximum length of matching tables.

Let us look at some examples and demonstrate how the algorithm works. **Example 1.** shown in Figure 8 has two pictures: a query picture (QP) and a

database picture (DP).

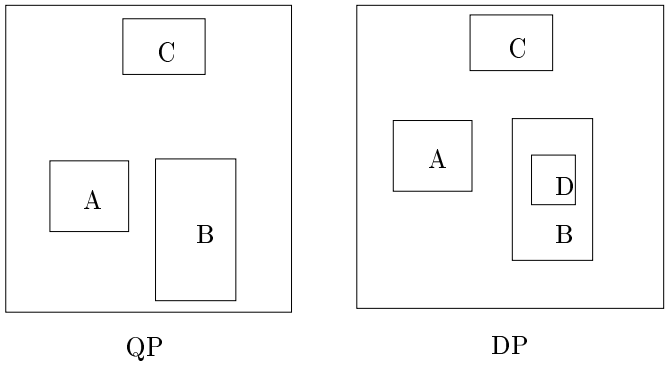

Fig. 8. Example 1: picture matching

Figure 9 contains graphs (symbolic picture models), represented as adjacency-lists, for QP and DP constructed by the graph construction algorithm.

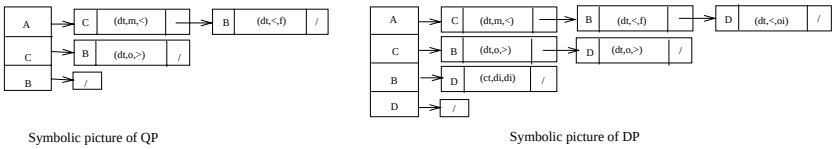

Fig. 9. Symbolic pictures of Figure 8

These graphs are inputs for the picture matching algorithm. The execution of the algorithm is as follows: since the nodes of QP are a subset of the nodes of DP ($V(QP) \subset V(DP)$), the algorithm executes lines l0 to 17. The linked-list comparison between the same element (node) of QP and DP (line 16) is always true, so the algorithm will return 1 (QP is a sub-picture of DP).

Example 2. Replace the symbol C in QP of Example 1 with another symbol, e.g. M. The picture matching algorithm will quickly detect that QP does not match with DP (line 18).

The picture matching algorithm described above performs exact matching. This type of matching is probably not very useful for certain applications. For

example, in multimedia applications similarity retrieval is preferred. Consider picture 10

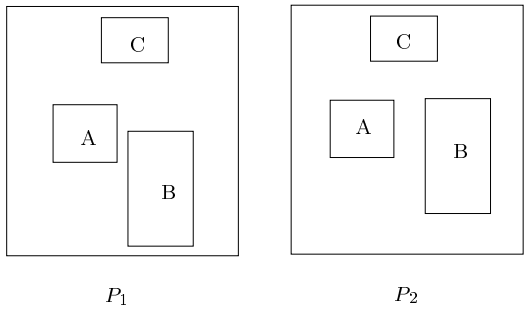

Fig. 10. Similar pictures

Using the picture matching algorithm above, pictures P_1 and P_2 are considered "not to match", since the relationships between A and B in P_1 $(dt, <, oi)$ and P_2 $(dt, <, f)$ are different. However, most users would consider P_1 and P_2 to be similar.

It is not our intention to discuss similarity matching in detail, so we simply highlight some important aspects of similarity matching using the 2D-PIR representation. The core idea is how to *measure* that two pictures are considered similar. By quantifying the concept "*degree of similarity*", we can perform useful retrieval operations such as thresholding and ranking.

The degree of similarity between two pictures represented using 2D-PIR picture is dependent on the degree of similarity between their corresponding 2D-PIR relationships. The degree of similarity between 2D-PIR relationships, in turn, is dependent on the degree of similarity between components of 2D-PIRs. For example, the degree of similarity between $(\delta_1, \chi_1, \psi_1)$ and $(\delta_2, \chi_2, \psi_2)$ is dependent on the degree of similarity between δ_1 and δ_2, χ_1 and χ_2, and ψ_1 and ψ_2. As a result, it is necessary to construct similarity metrics both for topological relationships (δ) as well as for interval relationships (χ or ψ).

The notion of *topological distance* as well as closest-topological-relationship-graph developed in [6] can be used to measure the degree of similarity between two topological relationships.

A metric for interval distance can be developed from the notion of *conceptual neighbours* introduced by Freksa [7]. According to Freksa, two intervals are *conceptual neighbours*, if they can be directly transformed into one another by continuously deforming (i.e shortening, lengthening, moving) the intervals (in a topological sense). For example, the relationship *before* (<) and *meets* (m) are conceptual neighbours, since they can be transformed into one another directly

by lengthening one of the intervals.

The algorithm for similarity matching is almost the same as the algorithm for exact matching, except that, instead of returning 0, 1 or 2, it returns a measure of the degree of similarity. For similarity retrieval, a sequence of pictures governed by the degree of similarity with the query picture will be retrieved. The final decision as to which picture is appropriate is left to a secondary filtering stage.

5 Application and Extension

A potential application of this work is in fast picture retrieval from a multimedia database. Current work in content-based retrieval for images uses colour, texture and shape [14, 3, 12, 13]. The present work can be used to enhance current retrieval methods. Picture retrieval using spatial relations, as presented in this paper, is also useful for Geographical Information System (GIS) and Computer Aided Design (CAD) applications.

Our definition of symbolic picture models using 2D-PIR graphs directly supports graphical query. Using this scheme, the user draws a query in a window using a mouse. This picture query will be translated to a 2D-PIR graph and, using our picture matching algorithm, the intended picture(s) can be retrieved.

2D-PIR can be extended to 3D-PIR to represent spatial relationships in a three dimensional space. An object in three dimensional space can be projected into x, y, and z axes to form 3D-PIR.

2D-PIR can potentially be extended to moving objects by incorporating time intervals into the relationships, forming spatiotemporal relationships. A 2D-PIR model of a picture can be extended to spatiotemporal projection relationships (ST-PIR) to represent relationships among moving objects in a scene. This extension of 2D-PIR looks promising, since ST-PIR will significantly reduce the storage requirement for representing moving objects in a scene, and hence moving object queries can be answered in a reasonable time.

6 Conclusion

This paper proposes a unified representation of spatial relationships using 2D-PIR. A 2D-PIR is a spatial relationship representation that integrates topological and directional relationships in one unified representation. This paper also proposes a method for symbolic modelling of pictures using 2D-PIR graphs. A 2D-PIR graph is a graph where the nodes represent the objects in a picture and the edges are labelled by spatial relationships represented by 2D-PIR.

We have shown that using minimum bounding rectangles (MBR) with boundaries parallel to horizontal and vertical axes will cause problems in 2D-PIR relationship interpretation. We propose two alternative solutions: slope projection and introduction of topological relationships into 2D-PIR representation. Slope projection does not solve the interpretation problem for overlapping MBR; however, it is a simple solution and may be preferred for certain applications where

correctness is not very important or applications where there are not many overlapping objects. Introducing topological relationships into 2D-PIR guarantees a correct interpretation of a 2D-PIR relationship, but extra computation is needed to derive topological relationships.

A symbolic picture model using a 2D-PIR graph is a simple, yet powerful representation of a picture. A 2D-PIR graph is constructed from a picture using a simple, efficient method. The construction also leads to a graph structure for which we develop an efficient picture-matching algorithm, which can perform both exact matching and similarity-metric matching.

2D-PIR relationships as well as 2D-PIR graphs have the potential for application to GIS, CAD, multimedia databases and image databases. Using 2D-PIR graphs, picture retrieval can be implemented relatively efficiently. In addition, it supports a direct graphical query facility (e.g. by drawing in a window).

Further possibilities for this work include: spatial reasoning using 2D-PIR, extending 2D-PIR to 3D-PIR, extending 2D-PIR and 2D-PIR graphs for moving objects (incorporating time in 2D-PIR).

References

1. J. F. Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843, November 1983.
2. S. K. Chang, Q. Y. Shi, and C. W. Yan. Iconic indexing by 2D strings. *IEEE Transactions on Pattern Recognition and Machine Intelligent*, PAMI-9(3):413–428, 1987.
3. T.-S. Chua, S.-K. Lim, and H.-K. Pung. Content-based retrieval of segmented images. In *Proceedings of The Second Annual ACM Multimedia Conference*, 1994.
4. E. Clementini, P. Felice, and P. van Oostrom. A small set of formal topological relationships suitable for end-user interaction. In *Advances in Spatial Databases: Third International Symposium–SSd'93*, volume 692 of *Lecture Notes in Computer Science*, pages 277–295. Springer-Verlag, 1993.
5. M. J. Egenhofer. Point-set topological spatial relations. *Int. J. Geographical Information Systems*, 5(2):161–174, 1991.
6. M. J. Egenhofer and K. K. Al-Taha. Reasoning about gradual changes of topological relationships. In A. Frank, I. Campari, and U. Formentini, editors, *Theories and Methods of Spatio-Temporal Reasoning in Geographic Space*, number 639 in Lecture Notes in Computer Science, pages 196–219. Springer-Verlag, 1992.
7. C. Freksa. Temporal reasoning based on semi-intervals. *Artificial Intelligence*, 54(1-2):199–227, 1992.
8. R. Gould. *Graph Theory*. The Benjamin/Cummings Publishing Company, Inc, California, 1988.
9. E. Jungert. Qualitative spatial reasoning from the observer's point of view–toward a generalisation of symbolic projection. *Pattern Recognition*, 27(6):801–813, 1994.
10. E. Jungert. Rotation invariance in symbolic slope projection as a means to support spatial reasoning. In *Workshop on Spatial and Temporal Interaction: Representation and Reasoning*, pages 166–170. 1994.
11. E. Jungert and S. K. Chang. An image algebra for pictorial data manipulation. *CVGIP: Image Understanding*, 58(2):147–160, September 1993.

12. H. Lu, B.-C. Ooi, and K.-L. Tan. Efficient image retrieval by color content. Department of Information Systems and Computer Science, National University of Singapore, 1994.
13. R. Mehrota and J. E. Gary. Feature-based retrieval of similar shapes. In *9th International Conference on Data Engineering*, pages 108–115, April 19-23 1993.
14. W. Niblack, R. Barber, W. Equitz, M. Flicker, E. Glasman, D. Petkovic, P. Yanker, C. Falautsos, and G. Taubin. The QBIC project: Querying images by content using color, texture, and shape. Research Report, Report Rj 9203(81511), IBM Almanden Research Center, San Jose, CA., 1993.
15. D. J. Peuquet and Z. Ci-Xiang. An algorithm to determine the directional relationship between arbitrary-shaped polygons in the plane. *Pattern Recognition*, 20(1):65–74, 1987.
16. T. Takahashi, N. Shima, and F. Kishino. An image retrieval method using inqueries on spatial relationships. *Journal of Information Processing*, 15(3):441–449, 1992.

This article was processed using the LaTeX macro package with LLNCS style