TOWARDS A 3D SPATIAL QUERY LANGUAGE FOR BUILDING INFORMATION MODELS

André Borrmann[1], Christoph van Treeck[1], and Ernst Rank[1]

ABSTRACT

The paper introduces the concept of a spatial query language for building information models (BIMs) and motivates its development. It provides formal definitions using point set theory and point set topology for 3D spatial data types as well as the directional, topological, metric and Boolean operators employed with these types. It also serves to outline the implementation of 3D spatial query processing based on an object-relational database management system.

KEY WORDS

3D Spatial Query Language, Building Information Model, Spatial Relations, Topology

INTRODUCTION

Formal languages have proven to be an efficient, precise interface for human-machine interaction, especially with respect to information retrieval and navigation in complex digital models. A spatial query language provides abstractions for spatial relationships and thereby facilitates the spatial analysis of geometric models. In existing query languages for building models, such as the *Product Model Query Language*, the utilization of spatial relations within a query is limited to simple containment relationships. This is mainly due to the structure of the underlying building information model which usually does not incorporate the explicit geometry of its components.

There is considerable demand for a spatial query language in engineering practice, especially with regard to the filtering or subdivision of building information models. The resulting partial models can serve as input for numerical simulation and analysis tools, or be made exclusively accessible to certain participants in a collaborative scenario.

2D spatial query languages are already well established in the field of Geographic Information Systems (GIS). Egenhofer, one of its pioneers describes the purpose of a spatial query language as follows (Egenhofer, 1991): “The principle demand of Spatial SQL is to provide a higher abstraction of spatial data by incorporating concepts closer to our perception of space.”. This statement is also valid for the engineering domain, where geometry plays a crucial role in the design process. But up to now, only a small number of research projects has dealt with spatial query languages for 3D models.

The paper presents the results of the first phase of the ongoing project, the formal definition of 3D spatial data types and the operators interacting with them, and provides an outlook on implementing 3D spatial query functionality on top of an object-relational database management system.

[1] Computational Civil and Environmental Engineering. Technische Universität München, Arcisstrasse 21, 80290 Munich, Germany. Phone +49 89 / 289-25117, FAX +49 89 / 289-25051, borrmann@bv.tum.de, treeck@bv.tum.de, rank@bv.tum.de

DEFINITIONS OF SPATIAL DATA TYPES

OVERVIEW

Starting point for the development of a spatial query language is the formal definition of the semantics of the available spatial data types and the operators working on them. Such a system of spatial data types is also referred to as "spatial algebra". It captures the fundamental abstractions of spatial entities and their relationships.

Our spatial type system consists of the four types *Point*, *Line*, *Surface* and *Body*. The incorporation of types with lower dimensionality also allows for the utilization of the language in the context of dimensionally reduced models which are widely used in civil engineering. The definition of the spatial types is not limited to non-curved (plane) entities.

The data types are formally defined using point set theory and point set topology (Gaal, 1963). This methodology is well established thanks to the efforts of the GIS research community.

SIMPLE VS. COMPLEX TYPES

The GIS research community distinguishes between simple and complex spatial types. Complex spatial data types can be understood as multi-component data types, i.e. a *Complex Point* can consist of an arbitrary number of points, a *Complex Line* can consist of several curves and a *Complex Body* may have a number of unconnected parts. Though simple data types reflect intuitive understanding, they do not incorporate the properties of a closed type system, where the geometric set operations *union*, *intersection*, and *difference* never result in an object outside of the type system. We therefore decided to use complex spatial objects.

In the main, our definitions for spatial data types in 3D space are aligned to the model proposed by Schneider and Weinrich (2004). An exception is the denomination of the data types: instead of *point3D*, *line3D* and *volume* we use the terms *Point*, *Line*, *Surface* and *Body*, as proposed in (Zlatanova, 2000). In addition, the type *relief* is omitted, because it is not needed for the application domain considered here.

The topological notions of *boundary (∂A), interior(A°), exterior (A^-)* and *closure ($\overline{A}$)* are given for each spatial data type. They are required for the formal specification of topological relationships.

POINT

A value of type *Point* is defined as a finite set of isolated points in the 3D space:

$$Point = \{P \subset \mathbb{R}^3 \mid P \text{ is finite}\}$$

By definition, a *Point* $P = \{p_1,...p_n\}$ has no *boundary, i.e.* $\partial P = \varnothing$, and all points belong to the *interior*, which is equal to the *closure:* $P = P^\circ = \overline{P}$. The *exterior* of P is $P^- = \mathbb{R}^3 - P$.

LINE

A *Line* is an arbitrary collection of 3D curves. It is defined as the union of the images of a finite number of continuous mappings from 1D to 3D space:

$$Line = \{L \subset \mathbb{R}^3 \mid \bigcup_{i=1}^{n} f_i([0,1]) \text{ with } n \in \mathbb{N} \ \wedge \forall\ 1 \le i \le n : f_i : [0,1] \to \mathbb{R}^3 \text{ is a continous mapping}\}$$

A *Line* is composed of several *curves*[2]. Each of them results from a single mapping f_i. The mappings *f(0)* and *f(1)* are called the *end points* of the curve. The *boundary* of a *Line* is the set of the end points of all curves it is composed of, minus those end points that are shared by several curves. These shared points belong to the *interior* of a *Line*. The *closure* of a *Line L* is the set of all points of *L* including the end points. For the *interior* we obtain $L^\circ = \overline{L} - \partial L = L - \partial L$, and for the *exterior* we get $L^- = \mathbb{R}^3 - L$.

SURFACE

Since the definition of the *Surface* type is based on mappings of 2D regions to 3D space, the definition of the *Region2D* type as given in (Schneider, 2006) is needed first. A *Region2D* is embedded into the two-dimensional Euclidean space $\mathbb{R}^2$ and modeled as a special infinite point set. The concept of the *neighborhood* of a point is used to define the *interior*, *exterior* and *closure* of the *Region2D* type. So, assuming the existence of a Euclidean distance function in 2D:

$$d : \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R} \text{ with } d(p,q) = d((x_1, y_1),(x_2, y_2)) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

Let $q \in \mathbb{R}^2$ and $\varepsilon \in \mathbb{R}^+$. The set $N_\varepsilon(q) = \{p \in \mathbb{R}^2 \mid d(p,q) \le \varepsilon\}$ is called the *(closed) neighborhood* of radius ε and center *q*. Let $X \subseteq \mathbb{R}^2$ and $q \in \mathbb{R}^2$. *q* is an *interior point* of *X* if there is a neighborhood $N_\varepsilon(q)$ such that $N_\varepsilon(q) \subseteq X$. *q* is an *exterior point* of *X* if there is a neighborhood $N_\epsilon(q)$ such that $N_\varepsilon(q) \cap X = \emptyset$. *q* is a *boundary point* of *X* if *q* is neither an interior nor an exterior point of X. *q* is a *closure point* of *X* if *q* is either an interior or a boundary point of *X*. The set of all *interior / exterior / boundary / closure points* of X is called the *interior / exterior / boundary / closure* of *X*.

Schneider et al. use the notion of *regular closed* point sets for the definition of the *Region2D* type in order to avoid geometric anomalies: a set of points $X \subseteq \mathbb{R}^2$ is *regular closed* if, and only if, $X = \overline{X}^\circ$. The *interior* operation excludes point sets with dangling points, dangling lines and boundary parts. The *closure* operation excludes points sets containing cuts and punctures while adding the boundary that was excluded by the *interior* operation.

Specifications for *bounded* and *connected* sets are another requirement. Two point sets *X, Y* $\subseteq \mathbb{R}^2$ are *separated* if, and only if, $X \cap \overline{Y} = \emptyset = \overline{X} \cap Y$. A point set $X \subseteq \mathbb{R}^2$ is *connected* if, and only if, it is not the union of non-empty separated sets. Let $q = (x, y) \in \mathbb{R}^2$ and $\|q\| = \sqrt{x^2 + y^2}$. A set *X* $\subseteq \mathbb{R}^2$ is *bounded* if there is a number $r \in \mathbb{R}^+$ such that $\|q\| < r$ for all $q \in X$. The type *Region2D* can now be defined as:

$$Region2D = \{R \subseteq \mathbb{R}^2 \mid R \text{ is regular closed and bounded, the number of connected sets of } R \text{ is finite }\}$$

Based on this definition, the *Surface* data type can accordingly be defined as the union of the images of a finite number of continuous mappings from a 2D region to 3D space:

$$Surface = \{S \subset \mathbb{R}^3 \mid \bigcup_{i=1}^{n} s_i(R) \mid n \in \mathbb{N} \wedge R \in Region2D \wedge \forall\, 1 \le i \le n : s_i : R \to \mathbb{R}^3 \text{ is a continous mapping }\}$$

The components of a *Surface* that are created by a single mapping s_i are called *superficies*[2]. All boundary points of the *Region2D* object are mapped to boundary points of the superficies. The *boundary* of a *Surface* is the set of all boundary points of the superficies it is composed of, minus those boundary points that are shared by several superficies. The shared points belong to the *interior*

[2] The definitions for curves and superficies provided in this paper introduce the basic ideas, but are too brief and not restrictive enough. Amongst others it is necessary to prevent self-intersection of curves and superficies. For a complete and sufficient definition, please see Schneider and Weinrich (2004) or follow-up publications by the authors of this paper.

of the *Surface*. The *closure* of a *Surface* S is the set of all points of S including the boundary points. For the *interior* we obtain $S^\circ = \overline{S} - \partial S = S - \partial S$, and for the *exterior* we get $S^- = \mathbb{R}^3 - S$.

BODY

Bodies are embedded into the three-dimensional Euclidean space $\mathbb{R}^3$ and modeled as special infinite point sets. It is possible for a *Body* to consist of several components and it may possess cavities. The notion of the *neighborhood* of a point is employed to define the *interior*, *exterior* and *closure* of the *Body* type, the definition of which corresponds to that given above for 2D space.

Let $X \in \mathbb{R}^3$ and $q \in \mathbb{R}^3$. q is an *interior point* of X if there is a neighborhood $N_\varepsilon(q)$ such that $N_\varepsilon(q) \subseteq X$. q is an *exterior point* of X if there is a neighborhood $N_\varepsilon(q)$ such that $N_\varepsilon(q) \cap X = \emptyset$. q is a *boundary point* of X if q is neither an interior nor an exterior point of X. q is a closure point of X if q is either an interior or a boundary point of X. The set of all *interior / exterior / boundary / closure* points of X is called the *interior / exterior / boundary / closure* of X.

For similar reasons as for the definition of *Region2D*, the definition of the type *Body* is based on the notion of *regular closed point sets*:

A set of points $X \subseteq \mathbb{R}^3$ is *regular closed* if, and only if, $X = \overline{X}^\circ$.

Here, the *interior* operation excludes point sets containing isolated or dangling point, line, and surface features. The *closure* operation excludes point sets with punctures, cuts or stripes and reestablishes the boundary that was removed by the *interior* operation. By definition, closed neighborhoods are regular closed sets.

Another essential is a specification for *bounded* and *connected* sets in 3D. Two point sets $X, Y \subseteq \mathbb{R}^3$ are *separated* if, and only if, $X \cap \overline{Y} = \emptyset = \overline{X} \cap Y$. A point set $X \subseteq \mathbb{R}^3$ is *connected* if, and only if, it is not the union of non-empty separated sets. Let $q=(x, y, z) \in \mathbb{R}^3$, and $\|q\| = \sqrt{x^2 + y^2 + z^2}$. A set $X \subseteq \mathbb{R}^3$ is *bounded* if there is a number $r \in \mathbb{R}^+$ such that $\|q\| < r$ for all $q \in X$. The spatial data type *Body* can now be defined as

$$Body = \{B \subset \mathbb{R}^3 \mid B \text{ is regular closed and bounded, the number of connected sets of B is finite }\}$$

DEFINTION OF SPATIAL OPERATORS

Spatial operators operate on spatial data types and possess spatial semantics. They can be classified into

- directional operators (such as *above, below, northOf, southOf*)
- topological operators (such as *touch, contain, equal, inside*)
- metric operators (such as *distance*)
- Boolean operators (such as *union, intersection*)

The following sections provide formal definitions for the spatial operators available within the proposed spatial algebra.

DIRECTIONAL OPERATORS

Directional operators can be used to query directional relationships of two spatial objects, i.e. their relative position. A directional operator always has two operands, which can be any combination of the spatial data types defined above. The result is a Boolean value. Before directional operators can be used, it is necessary to choose a directional reference system. For practical reasons, it will in most cases be aligned to the main axes of the building model.

We provide two mutual exclusive directional operators for each Cartesian direction: For the *x* direction *eastOf* and *westOf*, for the *y* direction *northOf* and *southOf,* and for the z direction *above* and *below*.

In literature, usually only point-to-point relationships are addressed. But in the context of 3D building models, directional body-to-body relationships are much more interesting. Fig. 1 shows different approaches for defining directional relationships, especially relevant when considering partly concave objects. In case a, the bounding planes of the operands form the base for the decision. Another widespread approach is to define directional relationships on the basis of the centroids of the bodies. In our opinion, these methods are not appropriate because their results are imprecise in many cases. A more suitable approach is to take into account the extrusion of the second operand in the respective direction, as shown in Fig.1, case b. If the resulting body intersects with the first operand, the result of the directional operator is true. In case b, the objects B, C and D can be classified as being above or below A. Note that according to this definition, object E is neither above A nor below it.

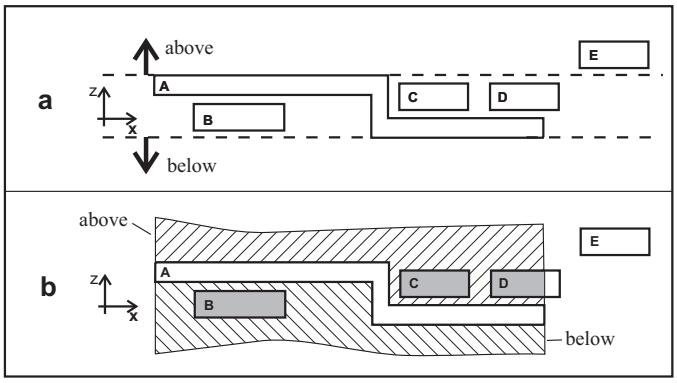


Fig. 1: Different methods for defining directional relationships

Formally, this definition can be expressed as follows: Let *A* and *B* be two 3D spatial objects, i.e. $A, B \subseteq \mathbb{R}^3$ with $a = (a_x, a_y, a_z) \in A$ and $b = (b_x, b_y, b_z) \in B$.

$$A \text{ eastOf } B \Leftrightarrow \forall a,b \text{ with } a_y = b_y \wedge a_z = b_z : a_x \geq b_x$$
$$A \text{ westOf } B \Leftrightarrow \forall a,b \text{ with } a_y = b_y \wedge a_z = b_z : a_x \leq b_x$$
$$A \text{ northOf } B \Leftrightarrow \forall a,b \text{ with } a_x = b_x \wedge a_z = b_z : a_y \geq b_y$$
$$A \text{ southOf } B \Leftrightarrow \forall a,b \text{ with } a_x = b_x \wedge a_z = b_z : a_y \leq b_y$$
$$A \text{ above } B \Leftrightarrow \forall a,b \text{ with } a_x = b_x \wedge a_y = b_y : a_z \geq b_z$$
$$A \text{ below } B \Leftrightarrow \forall a,b \text{ with } a_x = b_x \wedge a_y = b_y : a_z \leq b_z$$

TOPOLOGICAL OPERATORS

Topological operators can be used to query topological relationships of two spatial objects. Clementini & di Felice (1995) define topological relationships as follows:

"A topological space is generally described as a set of arbitrary elements (points) in which a concept of continuity is specified. Let X and Y be topological spaces. A mapping f: X→Y is said to be continuous if for each open subset V of Y, the set $f^{-1}(V)$ is an open subset of X. If a mapping f is a bijection and if both f and the inverse f^{-1}:Y→ X are continuous, then f is called a topological isomorphism. Topological isomorphisms preserve the neighborhood relations between mapped points

and include translation, rotation and scaling. Topological relations are those remaining invariant under a topological isomorphism.“

The method we use to formally describe the semantics of the topological operators is based on the work of Egenhofer & Franzosa (1991) and Clementini & di Felice (1996). Egenhofer uses a 3x3 matrix to express the nine intersections of the point sets that make up the *interior*, the *boundary* and the *exterior* of two geometric objects A and B:

$$I_9(A,B) = \begin{bmatrix} A^\circ \cap B^\circ & A^\circ \cap \delta B & A^\circ \cap B^- \\ \delta A \cap B^\circ & \delta A \cap \delta B & \delta A \cap B^- \\ A^- \cap B^\circ & A^- \cap \delta B & A^- \cap B^- \end{bmatrix}$$

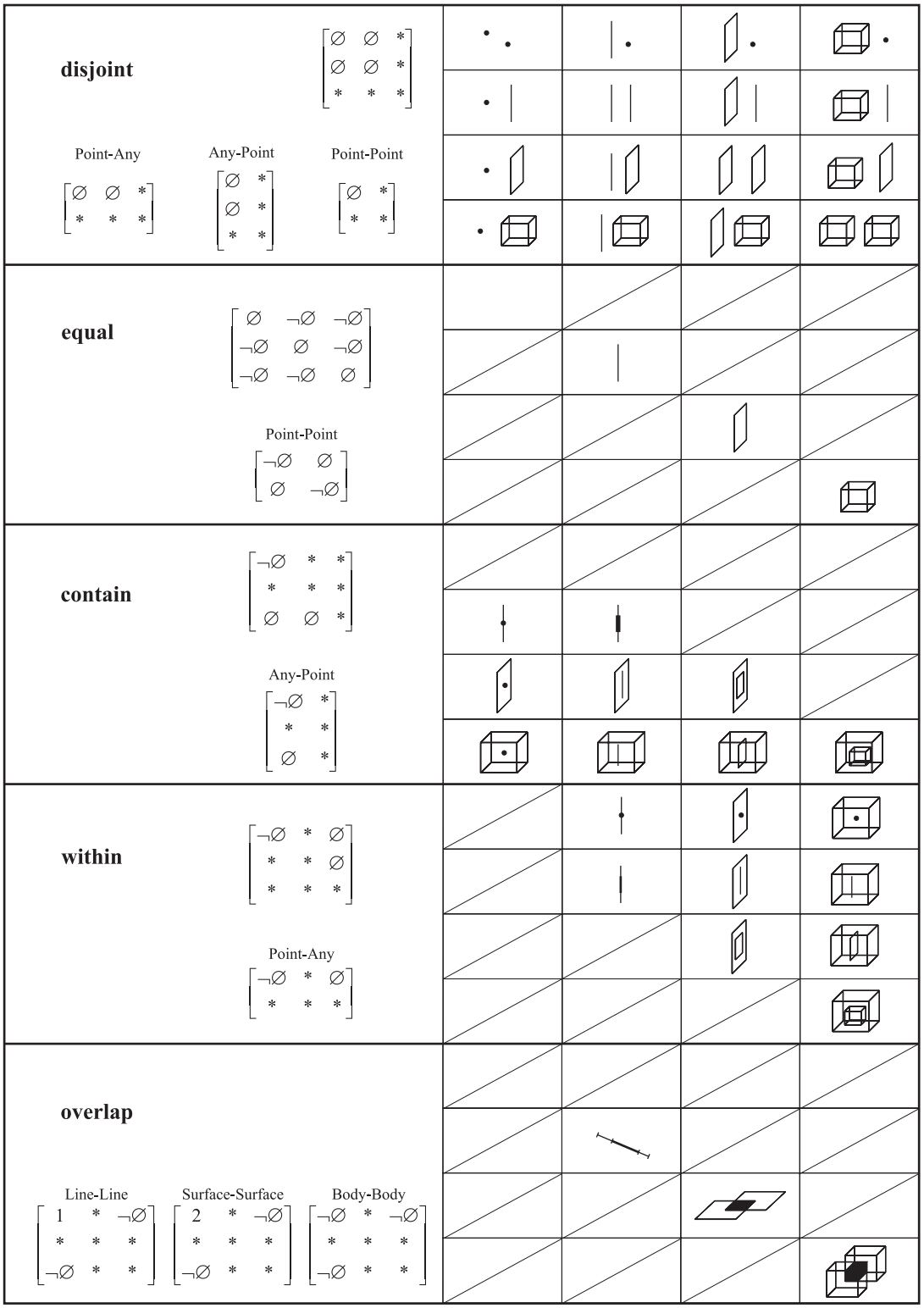
The resulting matrix can be used to precisely define topological relationships by means of empty (Ø) and non-empty (¬Ø) sets. This method is also referred to as the "9 intersection method" (9IM). Egenhofer & Franzosa found eight possible relationships using the 9IM for region-to-region relationships in 2D. Zlatanova (2000) used the 9IM to systematically investigate the number of possible relationships for *simple* Points, Lines, Surfaces and Bodies in 3D and found 69 relationships. The number is likely to be much higher for non-simple objects. Because such a differentiation is much too finely grained to be of use in engineering practice, a set of relationships has to be chosen that is of considerable significance for engineers and has a corresponding natural language equivalent.

Clementini & di Felice (1996) extend the 9IM by (1) only specifying whether the point set resulting from the nine intersection operations is empty, or not, where it is necessary and (2) by specifying the dimensionality of the point sets where it is important. The former essentially means that certain 9IM configurations are subsumed under a single common name and others are disregarded completely. For the latter, a *dim* operation has been defined that returns the maximum dimension of the point set. This method is also referred to as the “dimensionally extended 9 intersection method” (DE-9IM). DE-9IM matrices use the following symbols besides Ø and ¬Ø: The star (*) means there is no specification for the resulting set at this position, and numbers (0, 1, 2, 3) refer to the dimensionality of the resulting set.

Fig. 2 shows all topological relationships defined within our framework together with the DE-9IM matrices that formally describe them. It also contains a matrix of pictograms that illustrates the semantics of each relationship for any of the possible combinations of spatial data types. In the pictogram matrix, rows correspond to the type of the first operand and columns to that of the second one. Most of these definitions are equal to that of Clementini & di Felice (1996) and the OpenGIS standard (OGC, 2005) respectively, with the following exceptions:

1. The underlying data types are defined for the 3D space. Accordingly we included the data type *Body* within our definitions.
2. We restrict the validity of each operator to a subset of all possible combinations of operands.
3. Since the boundary of a *Point* is by definition an empty set, we do not include the definitions for the *Point* type in the matrices valid for all other types. Instead, we specify additional matrices for relationships where *Points* are involved with the following occupation: If *A* is a *Point* and *B* is of any other spatial data type, we define:

$$I(A,B) = \begin{bmatrix} A^\circ \cap B^\circ & A^\circ \cap \delta B & A^\circ \cap B^- \\ A^- \cap B^\circ & A^- \cap \delta B & A^- \cap B^- \end{bmatrix}$$


disjoint
Point-Any
Any-Point
Point-Point
equal
Point-Point
contain
Any-Point
within
Point-Any
overlap
Line-Line
Surface-Surface
Body-Body

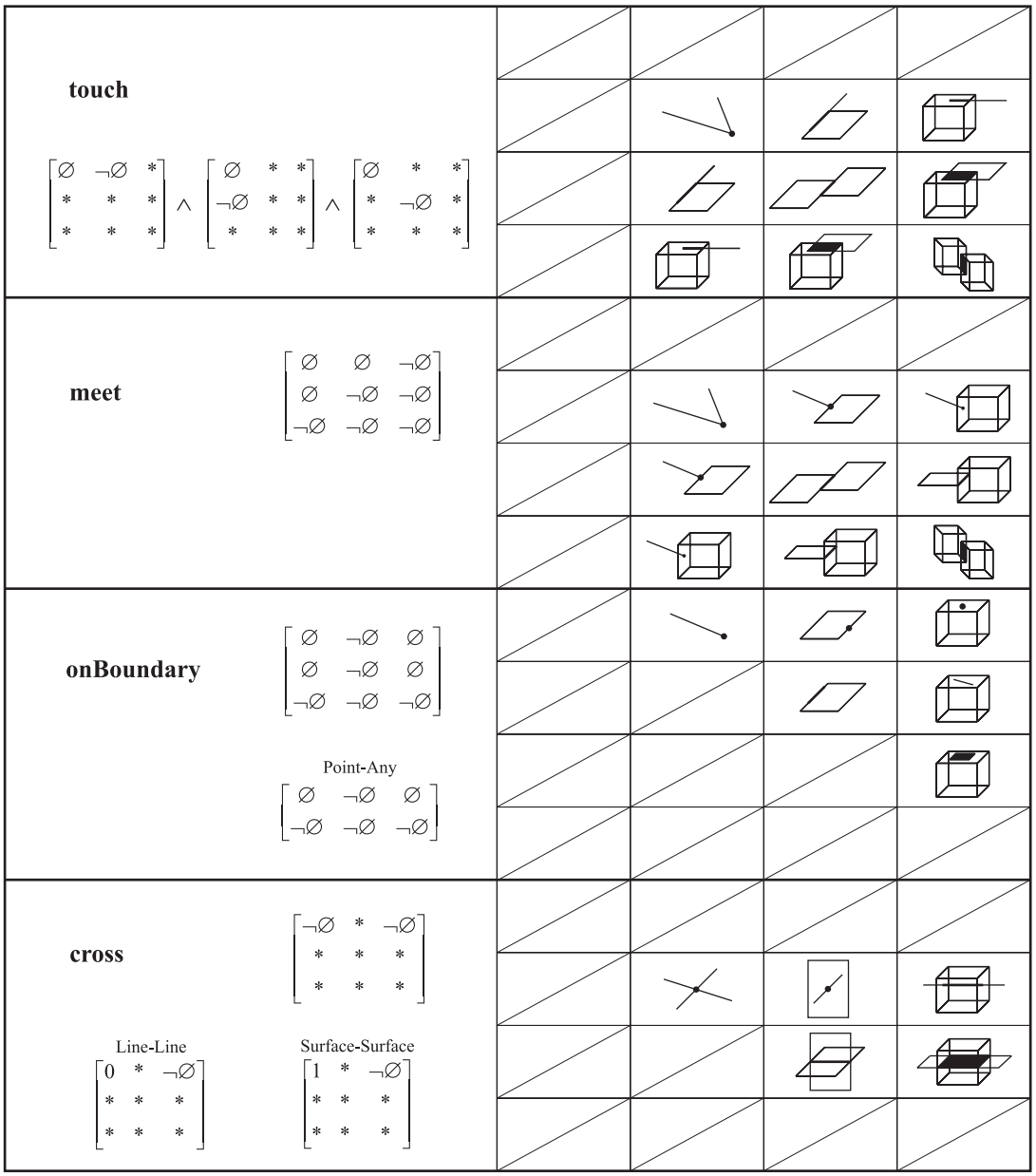


Fig. 2: The topological relationships defined within the spatial algebra. The DE-9IM matrices that formally define the semantics of the relationship is shown on the left hand-side. In DE-9IM matrices the following symbols are used besides Ø and ¬Ø: The star (*) means there is no specification for the resulting set at this position, and numbers (0, 1, 2, 3) refer to the dimensionality of the resulting set. In the pictogram matrix on the right-hand side, rows correspond to the first operand and columns to the second one.

If B is a *Point* and A is of any other spatial data type, we obtain:

$$I(A,B) = \begin{bmatrix} A^\circ \cap B^\circ & A^\circ \cap B^- \\ \delta A \cap B^\circ & \delta A \cap B^- \\ A^- \cap B^\circ & A^- \cap B^- \end{bmatrix}$$

In the case that A and B are points, we write:

$$I(A,B) = \begin{bmatrix} A^\circ \cap B^\circ & A^\circ \cap B^- \\ A^- \cap B^\circ & A^- \cap B^- \end{bmatrix}$$

4. For the purpose of an extended differentiability, we define two additional relationships *meet* and *onBoundary* as refinements of *touch*. For user convenience, we also define the relationships *within* (inverse to *contain*) and *equal*.
5. We define two special relationships *surround* and *encompass* for (partially) concave spatial objects, as depicted in Fig. 3. Here, the convex hull of the second operand is taken into account. A formal definition for the corresponding convex object of an spatial object will be given in an extended publication.

METRIC OPERATOR

The only metric operator defined for the spatial data types so far is the *distance* operator. It has two operands which can be of any combination of the spatial data types. The *distance* between two spatial objects of arbitrary type is defined as the shortest distance between any of the points belonging to the closure of these objects.

$$A, B \subset \mathbb{R}^3 : distance(A,B) = \min_{a,b}(d(a,b),\ a \in \bar{A},\ b \in \bar{B})$$

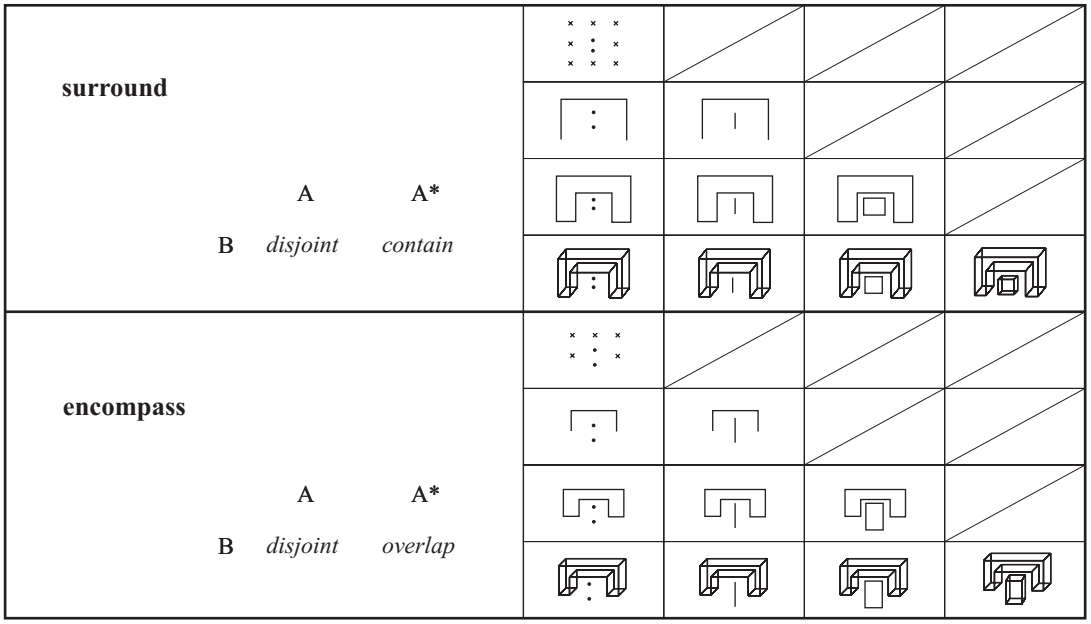


Fig. 3: The topological relationships *surround* and *encompass* for partly concave spatial objects. They are formally defined by means of the topological relationship between the first (A) and the second operand (B), and between the first operand (A) and the corresponding convex object of the second operand (B*).

BOOLEAN OPERATORS

The Boolean operators *union, intersection, difference* have two operands that are of the same spatial type. The result likewise resembles that of the operands. The operators possess the same semantics as their equivalents in pure point set theory.

IMPLEMENTATION CONCEPTS

The implementation of the abstract type system into a query language will be performed on the basis of the query language SQL, which is a widely established standard in the field of object-relational databases. The international standard SQL:1999 extends the relational model to include object-oriented aspects, such as the possibility to define complex abstract data types with integrated methods. Database management systems (DBMS) that support these features are also referred to as *object-relational* database management systems (ORDBMS). They combine the advantages of the relational concept, such as high efficiency in query processing, with that of object-oriented modeling, such as the high level of abstraction by providing encapsulation and inheritance (Melton, 2003).

Due to lack of space, a detailed description of our implementation concepts will be given in follow-up publications. This involves the choice of data structures capable of storing the geometry of building components, concepts for combining non-geometric (semantic) information with shape descriptions, and strategies for a fast processing of queries containing spatial predicates including index structures and alternative geometry representations based on uniform Cartesian grids ("voxels").

REFERENCES

Clementini, E., and Felice, P. D. (1996). "A model for representing topological relationships between complex geometric features in spatial databases." *Information Sciences*, 90(1-4), 121-136.

Egenhofer, M., and Franzosa, R. (1991). "Pointset topological spatial relations." *International Journal of Geographical Information Systems*, 5(2), 161-174.

Gaal, S. (1963). *Point Set Topology.* Academic Press, New York, USA.

Melton, J. (2003). *Advanced SQL:1999. Understanding Object-Relational and Other Advanced Features*, Morgan Kaufmann, San Francisco, USA.

Open Geospatial Consortium (OGC) (2005). "OpenGIS Implementation Specification for Geographic information". Document 06-126.

Schneider, M., and Weinrich, B. E. (2004). "An abstract model of three-dimensional spatial data types." *12th ACM Intern. Worksh. on Geographic Information Systems*. Washington, DC, USA.

Schneider, M., and Behr, T. (2006). "Topological relationships between complex spatial objects." *ACM Transactions on Database Systems*. In Press.

Zlatanova, S. (2000). "On 3D Topological Relationships." *11th Int. Workshop on Database and Expert Systems Applications*, Greenwich, London, UK.