Representation and Management Issues for Large Spatial Knowledge Bases

Thodoros Topaloglou

e-mail: thodoros@cs.toronto.edu

Department of Computer Science
University of Toronto
Toronto, Ontario
M5S 1A4, Canada

September, 1991

Contents

1 Introduction

The subject of this study is the representation of spatial entities and relationships, and consequent data management issues. This is a very challenging research topic because (a) support for *spatial data* is demanded by many research areas such as robotics, engineering databases, geographic information systems etc. and (b) management of *large spatial representations* is a difficult and not a well-understood problem.

In robotics [Kui78], [MD84], [Dav86], [KB88] the need to locate, place or navigate an intelligent agent in a 2-, 3-, or 4-dimensional space is essential. In common-sense reasoning [For80], [NC91], [Kau91] the modeling of situations of the physical world also requires primitives for the representation of spatial knowledge. Although the requirements for spatial information management support in these research areas are substantially different, there is definitely a need for a mathematical formalism which characterizes the situation and helps to interpret properties of physical objects or infer properties with respect to their position in space.

In engineering applications, such as CAD/CAM [SH88] and VLSI design [GS82], large designs need to be stored and accessed efficiently. Similarly, in geographic/cartographic applications [CF81], [OSDM89], [JC88], [RFS88], massive volumes of spatial data which are characterized by extreme irregularity need to be stored and accessed effieciently. In the last ten years a lot of effort have been spent to extend databases with new data types and operators for handling spatial, temporal and structural data, and upgrade the data management techniques in order to make them more efficient. In addition to the above applications, spatial reasoning is of interest in computer graphics [Sam89], computer vision [LLCN87], medical imaging, urban planning [LM90], natural resource planning, etc.

Problem solving with spatial data consists of three fundamental components, namely:

Representation: We need a formalism which is able to represent objects, their local geometry and their position in space, and spatial relationships between objects.

Reasoning: Given a configuration of objects in space and (some of the) spatial relationships between them, we must be able to infer the spatial relationship between any pair of objects.

Management: Organize the spatial and non-spatial information in a way that searching with respect to spatial criteria becomes possible.

Each of these components must satisfy respective requirements.

The spatial representation system is good to support a maximum range of applications and be insensible to restrictions such as the dimensionality of the modeling domain, the presense or absence of a global reference frame, the orthogonality of the domain etc. However, the most desirable feature of a spatial representation system is its ability to represent imprecise and partially known information. Therefore, it must allow relationships to be stated at varying degrees of granularity, multiple (and possibly conflicting) descriptions of objects, ambiguous descriptions, qualitative relations and range values to metric relations. A more AI-minded requirement is to express the absence of an object. Also, objects in space may be seen either as points or as extended objects. Extended objects have shape (usually irregular) and non-spatial properties such as color. The representation must gracefully integrate spatial and non-spatial properties. Some sample spatial descriptions, in accordance to the above stated requirements, are (a) relative positions between objects (e.g., "University Avenue is roughly parallel to St George Street", "The medical school is between the computer science department and the library" ,"Hamilton is about eighty km from Toronto".) (b) relations between parts of objects (e.g., "The arm of the overhead projector is vertical to its base".) (c) information about quantities which are not discrete (e.g., "Grove Street slopes downhill going west".) (d) information about absence of objects (e.g., "There is no stop sign on College Street"), etc.

The reasoning models that we are interested in, must be capable of coping with qualitative and quantitative constraints, partial knowledge and uncertainty. The general reasoning problem is formalized as follows:

Given: A set of objects
A set of constraints
To find: The consequent spatial relations

Some very frequent computations include finding the relative position between objects [Dut89], [MJ90], [Gue91], finding the consistency of a set of spatial constraints [DMP89], [Gue91], [Dav86], the path finding problem (e.g., "How can I go from UofT to the Art Gallery of Ontario?") [Dav86], [McD82], [Kui78] and searching with respect to spatial criteria (e.g., "Find all restaurants on Danforth Avenue.") [EL82], [RFS88].

The data management component serves as the back-end system which provides a large spatial data repository and fast access to it. It needs to be as application independent as possible and provide as much access methods as possible. However, it can be seen not just as a serving sub-system but as a tightly integrated component. The last requirement is essential because it avoids type and impedance mismatch[1]. However, tight integration also has some drawbacks, such as the necessity for highly expressive data/knowledge models, query languages and more sophisticated query optimization.

This study attempts a literature review of methods and proposals on representational, reasoning and management issues for spatial data. Especially, it focuses on cases where spatial knowledge is incomplete or imprecise. The capabilities of the various examined methods are studied and demonstrated over the spatial configuration shown in figure 1. This is a 2-dimensional projection of the University of Toronto campus map. It depicts the buildings and the streets around the Computer Science Department. Buildings are described by their boundary (spatial description) and their name (non-spatial description). Buildings are grouped into campus blocks forming a containment hierarchy. The map contains relative scales of the buildings but not metric information. In addition, some areas of the map are partially known, thus they were drawn in vague terms. For instance, I do not know what is behind the ME building, I do not know the name and the morphology of the front building

[1] The impendance mismatch is a problem which is faced when a system has to cope with two representations or typing systems and it must very often transform data from the one to the other.

Figure 1: UofT Campus Map

in the ME block and the building between the MB and the L. I also do not care about what does exist in the top left corner of the map, thus it is left undrawn. Therefore, this part of the UofT campus map reflects exactly my personal knowledge about the campus layout which is subjective, incomplete and approximate.

The rest of this report is organized as follows. Section 2 presents ontologies for space which are used in AI. Section 3 reviews representations of spatial configurations, solids and regions. Section 4 presents various spatial reasoning methods. Section 5 is a thorough review of work done in spatial databases including data modeling, query processing and data organization issues. Section 6 concludes my study with a summary of open research issues in the examined areas.

2 Ontologies of Space

The purpose of this section is to identify the ontologies which are appropriate for the representation of spatial domains. Ontologies which are used to describe a particular domain contain two basic kinds of elements, a set of categories and a set of relations that can hold between instances of these categories. In general, ontologies of some domain must be epistemologically and pragmatically adequate. An epistemologically adequate ontology must be able to state whatever we want to state in a domain. Ontologies with pragmatic adequacy must be able to state commonly-stated things in a relatively easy fashion and must be able to infer commonly-inferred conclusions in a relatively easy way. In practice, it is very hard to find an ontology that satisfies both of the properties. Usually, ontologies do not consist of primitive elements of a domain but from abstractions which are build up out of these primitives. Therefore, in the rest of this section we will use the ontology and abstraction as same meaning words.

Historically, the first known abstraction of space is the point-based. Not surprisingly, the first treatises on the study of space and the establishment of geometry date back to Ancient Greece. The Euclidean space (Euclides' "Elements of Geometry" [Euc]) is still the most favourite abstraction of space in spatial databases. The most primitive notions in Euclidean geometry is the point and the distance between points.

The point-based abstraction of space seems to be the predominant one. Hegel [Heg59], also accepts the *point* as the infinitesimal origin of any spatial configuration. Lines and more complex spatial configurations can be constructed by continuous movements of simpler spatial entities. Contrary to the point ontologies, solid body geometries have been investigated only in this century.

The selection between points vs. extended entities is not the only criterion in choosing an appropriate ontology for space. Other criteria are boundedness vs. infiniteness, discreteness vs. continuity of space, etc. Philosophical opinions differ but for computer scientists the situation is more concrete. Points are taken as the predominant ontology for space, but a body ontology is also referenced, especially in mechanics and solid modeling. It is not surprising that both primary ontologies for time, namely points and intervals, have their counterparts in the case of space. As for infinity, there is always a point in space which is farther than the farthest known point but this is not extremely useful in any application. Terrains, geometric configurations and land maps are always bounded. The discreteness or continuity depends on the universe where we want to interpret the spatial properties. Discreteness implies isomorphism to the cartesian product of integers and continuity to that of reals or rationals.

As for the relations of space, van Benthem [vB83] states that the primary spatial operators are comparisons of relative distance.

More practical ontologies of space were investigated in knowledge representation. Lenat and Guha [LG89] in the CYC project consider different ontologies motivated by the pragmatic adequacy i.e., they select abstractions of space which serve better the task that they are interested in performing. The tasks that are performed in a CYC knowledge base environment are prediction of the behaviour of some given device, diagnosis, manufacturing and design.

Set of points: The basic entities of the domain include points, set of points (lines), forces, masses, velocities, etc. The relations that hold between sets of points may be *spatial* relations, namely, above, below, etc., and *spatio-temporal relations*, such as, connectedTo, looselyConnectedTo, etc. The basic computations used in this abstraction are related to continuum mechanics.

Equations and diagrams: A large class of problems in mechanics uses an equation-level description of systems. Very often, equations are complemented by diagrams. The basic entities here are objects, forces, velocities, etc. The relationships between objects are geometric constraints between specific points on the objects. This abstraction emphasizes the behavioural aspects of a system.

Solids: This abstraction emphasizes the geometric properties of a system. The primitive elements are cubes, cylinders, etc. The relations between instances of these primitives are divided in *spatial* and *spatio-temporal.*

Devices: In a device level abstraction entities correspond to functional primitives; for example wheel, lever, etc. Relations at device level are very specialized.

3 Spatial Representations

Spatial data representations can be studied at two levels of abstraction, the physical and the logical level. At the physical level of a spatial representation one is interested in spatial data structures and access paths. At the logical level one is interested in logical models for spatial information and their inferential capabilities. This section focuses on logical representation issues. Spatial data structures and access paths are discussed in section 5.2.

At the logical level, the goal of a spatial representation is to establish a mapping between objects in a *modeling space*, M, and their representation in a symbolic structure which is called *representation space*, R. The mapping function s from a subset of the modeling space to a subset of the representation space ($s : D \to V,\ D \subseteq M, V \subseteq R$) is called *representation scheme* [Req80]. D is the domain of s and contains all the representable objects, and V is the range of s and it contains all the valid representations. The inverse function of s, $r = s^{-1}$, relates representations to objects. If $r(v)$ is a single element set for any $v \in V$, then the representation scheme s is called *unambiguous.* If $s(r(v)) = \{v\}$ for any $v \in V$ then the representation scheme is called *unique.*

Following this definition we can characterize the spatial representations with respect to their modeling space, their representation space, the mapping conventions, and, in addition to those, their inferential capabilities and the associated computational cost.

The modeling space is very much related to the application area for which a spatial representation is targeted. For instance, the modeling space for CAD/CAM applications is 3-dimensional solids. Geographical information systems basically deal with 2-dimensional regions of arbitrary shape. Robot navigation and path planning applications consider 2- or 3-dimensional spaces.

The representation space is characterized by the primitives and the abstractions that the representation structure uses, and its operations. In path planning more emphasis is given to the empty space and the objects' boundaries than the objects' internal structure. Conversely, in CAD modeling more attention is paid to the objects' structure. In mechanics and kinematics one is interested in motion aspects. Lastly, in geographic applications, region overlapping, adjacency and relative positions of landmarks are important. Another important aspect of a spatial representation, is its ability to represent a partially known piece of space.

The mapping function consists of a notation according to which objects are represented (syntax), and a methodology for associating meaning to the representations (semantics). The mapping function sets a formal framework for describing and evaluating spatial representation schemes.

The rest of this section presents a review and an analysis of spatial representations based not only on formal properties such as uniqueness and ambiguity but also on informal ones such as easiness of implementation, length of representation and understandability. Additional attention is paid to representation schemes which are able to represent spatial uncertainty.

3.1 Spatial Representations in Artificial Intelligence

Spatial representations in AI are studied in robot navigation problems [Kui78], [McD80], [MD84], [KB87] and lately in qualitative mechanics [For80], [FNF87], [Fal87].

The desired geometric support in the robot navigation problem amounts to a convenient representation of a large scale space, usually called *working memory*, or *cognitive map*, or simply *map* [Kui78], [McD80] and the ability to upgrade the map from observations. The latter is call *assimilation* or *map learning* problem [Dav86], [Dut89], [KL88]. Given a map, the problem of navigation consists of creating and successfully executing a plan to travel from one place to another.

The modeling space is a 2-dimensional floor, called *terrain*. The terrain is either known or partially known (unexplored). The representation considers topological and geometrical properties of the objects in the terrain such as shape and position of an object in a room, position of the walls, etc. Non-spatial properties such as the color of the walls are not considered. Solid objects are represented by a 2-dimensional projection on the terrain. Most of the time, objects in a terrain are viewed as obstacles, therefore the representation of their outline is more useful than their internal structure.

Kuipers and Levitt [KL88] defined a four-level semantic hierarchy of descriptions of large scale space that supports map learning and navigation. These are: sensorimotor, procedural, topological and metric levels. Only the last two are considered interesting in this discussion.

The topological description is a description of the environment in terms of fixed entities, such as places, paths, landmarks and regions linked by topological relations such as connectivity, containment and order. The metric description is a description of the environment in terms of fixed entities, such as places, paths, landmarks and regions linked by metric relations such as relative distance, relative angle, and absolute angle and distance with respect to a frame of reference.

Traditional spatial representation methods, which were used in the robot exploration and map learning problem, are among others the Configuration Space [LP81], the Generalized Cones [Bro82], the Voronoi Diagram [Mil85], [IJRW85], the Convex Polygon Model [Lau83], the Graph Model [IJRW85], and the Polygonal Region Model [Mil85]. All of the above models assume exact information. Therefore, computation methods such as computational geometry and non-linear algebra are applicable.

More recent proposals in this area suggest qualitative models. Such approaches are the NX robot [KB87], [KB88] and the QUALVAN model and simulator [KL88]. The qualitative approach is more appropriate in describing environments with metrical inaccuracy. Both these system follow the four level semantic hierarchy of Kuipers and Levitt. The TOUR model [Kui78] it is an earlier work that can be classified in the same category. The spatial representation model in TOUR is a pioneering work in this field and its topological level has served as the basis for subsequent work by Kuipers including the NX robot.

The TOUR model makes the distinction of the topological and metrical layer too. At the topological layer, it represents a cognitive map as a network of places and paths which are related through containment and boundary relationships. The metrical layer adds quantitative information into the map. The metric information can be either local geometry at places along paths or local orientation frames with respect to a global frame of reference. Non-geometrical knowledge about a particular environment is expressed in attribute/value pairs format.

A combination of the qualitative and the quantitative approach in the cognitive maps representation is followed by McDermott and Davis [McD80], [MD84]. They divide a cognitive map into two components, a propositional component and a fuzzy map. The first corresponds to the topological level and the second is a repository of metric knowledge. Their approach is motivated by imprecision in spatial descriptions and the representational strength of supporting multiple frames of reference. Another intricate issue which is handled by their system is the shape of objects; shapes are represented by a prototype and a modification. Their program, SPAM, implements a data structure along these lines. SPAM has a quite non-trivial query processor which is capable of three retrieval modes. Retrieval of assertional knowledge is handled by a special purpose theorem prover. Retrievals form the fuzzy map, term values and truth tests, are transformed into numerical optimization problems. Lastly, retrieval of objects with specified properties is handled by a structure which is called discrimination tree and uses both qualitative and quantitative information. Performance and representational problems, observed in SPAM, are studied further by Davis [Dav85], [Dav86] in the MERCATOR program. This work is reviewed in a later section of this report.

The qualitative approach in spatial representation is taken by Forbus and his colleagues in models of reasoning about mechanics and kinematics [For80], [Fal87], [FNF87].

3.2 Representations for Solid Objects

Many engineering applications such as mechanical CAD/CAM, architectural design, computer graphics involve spatial representations which focus on solid objects [RV82], [RV83], [Wil88b]. The solids which are appropriate for a computer-based representation must have shape invariant of location and orientation, occupy a finite portion of space, be finitely describable, must have unambiguously defined boundaries, and lastly must produce other solids when motion and boolean operations are applied to them.

Representation schemes for solid objects are distinguished into elementary and hierarchical [Gue88]. In a hierarchical representation scheme, the objects are represented by some combination of simpler objects of the same dimension. Elementary representation schemes cannot represent objects as compositions of simpler objects.

3.2.1 Elementary Representation Schemes

The most primitive elementary representation scheme for solids is the *drafting scheme*. In the drafting system, solids are represented by their planar projections. This scheme is ambiguous in the sense that the same representation corresponds to several objects and not-unique in the sense that and object can admit more than one representations. Other elementary representation schemes are the *primitive instancing* scheme, the *sweep representation* scheme and the *boundary representation* scheme.

The primitive instancing scheme is based on the notion of object families (generic instances e.g., cubes, spheres, etc.) and individual objects within a family (primitive instances). Each member of a family is being distinguished by a few parameters. The main disadvantage is the lack of means of combining instances to create structures which represent new and more complex objects. Among its advantages, it is an unambiguous, unique, concise and easy to use scheme. In our example modeling space of figure 1 only few buildings (e.g., building "`49`") can be represented using primitive instancing; most of them are not formalizable because of the complexity of their shapes.

The sweep representation is based on the concept of sweeping an object in the space. The moving object plus its trajectory characterize a volume. Both are unambiguous but not-unique. In our example, a 3-dimensional representation of the {"`GB`", "`SF`"} complex can be generated by its translation with respect to the z-axis (see figure 2(a)).

Boundary representations (BR) received attention in the database modeling field [KW87], [AOG+88]. A solid is represented by segmenting its boundary into a finite number of bounded subsets usually called faces and describing each face by its bounding edges and vertices. Additional boundary representations are the vertex lists for general polygons and the wireframe representations for 3-dimensional objects [Gue88]. Boundary representations of the first type are unambiguous if their faces are represented unambiguously. In general, they are not unique and cannot be used to represent objects with holes. Figure 2(b) illustrates the boundary representation of the solid of figure 2(a). Note that there is a difficulty to represent the concave face. However, they are easy to understand.

3.2.2 Hierarchical Representation Schemes

The most common hierarchical representation schemes are the *spatial occupancy* scheme, the *cell decomposition* and the *constructive solid geometry (CSG)* [Gue88].

In the spatial occupancy scheme a solid is represented by a list of all spatial cells (voxels) that it occupies. Cells (voxels) may be cubes of fixed size and lie in a fixed spatial grid called spatial array. This scheme is unambiguous, unique, but quite verbose.

In cell decomposition scheme the solid is decomposed into cells which must be either disjoint or meet precisely at a common face. Quadtrees and octrees [Sam84] are examples of representation schemes which follow the hierarchical decomposition principle. The schemes are discussed in detail in section 3.3.3. Cell decompositions are unambiguous and not-unique.

The constructive solid geometry (CSG) scheme represents solids in terms of a set of 3-dimensional volumetric primitives (blocks, cylinders, cones and spheres are typical examples) and a set of operators (set operators such as union, intersection, difference, and similarity operators such as rotation, translation and scaling). The object is represented by a binary tree (the CSG-tree) whose interior nodes correspond to operators and leaves to primitives or to numerical arguments for the operators.

```
<CSG-tree> ::= <primitive leaf> |
```

Figure 2: Representations of solid objects

```
<CSG-tree> <set-op node> <CSG-tree> |
<CSG-tree> <motion node> <CSG-tree>
```

CSG-trees provide an unambiguous scheme to represent solid objects. The drawbacks of CSG is that it not unique and it very hard to implement search operators. However, it is widely used in current CAD/CAM systems [Wil88b]. GSC trees with uninstantiated parameters can be used to represent generic objects (also called macro- or parametric-objects or object schemata). Figure 2(d) shows the CSG tree for the solid of figure 2(c).

Combinations of the previous schemes are very common in many applications [Wil88a], [SH88]. These are called *hybrid schemes* or *multimodels* [Tak88]. The most often hybrid schemes are: CSG/boundary, CSG/solid-sweeps, CSG/general-sweep.

3.3 Representations for Regions

Symbolic representations of regions are mainly used in geographic information systems and pictorial databases. Some of the characteristics of region representations are that the modeling domain is 2-dimensional, the regions might have arbitrary complex shapes, and their dimensions are known with limited or full precisions. In most of the cases the representation makes the provision of a database component. There is also a vast number of spatial operators that need to be supported: geographic operators e.g., north-of, east-of, same-position-as, etc., local operators e.g., overlap, adjacent, includes, etc., set operators e.g., spatial union, intersection, difference and lastly similarity operators e.g., rotation translation and scaling.

There are two approaches to region representation: those that specify the boundaries of a region and those that organize the interior of the region.

In this section we survey four different region representation schemes. These are the *polygonal representations*, *symbolic projections*, *hierarchical partitioning* of space and the *space filling curves*. The first two are boundary representations whereas the last two organize the interior of a region.

3.3.1 Representations of Polygons

Several representations for polygons have been proposed. Among them, we distinguish the polygonal (polyhedral) chains of Guenther [Gue88] and the polygonal approximation method of Davis [Dav85]. More simplified polygonal representation schemes have also been proposed. The minimum bounding rectangle (MBR) model is one of them where the basic idea is to approximate complex shaped objects by the rectangle that encloses it. Minimum bounding rectangles are extensively used in the development of access methods for spatial objects (for a detailed discussion see section 5).

Polyhedral Chains

Guenther's approach is capable of representing polyhedral objects in arbitrary dimensions. The 2-dimensional counterpart of his method deals with polygons. The presentation will follow the general case although the illustrations will be two dimensional.

Any general polyhedron is represented as a convex polyhedral chain, i.e., algebraic sum of simple convex polyhedra, called *cells*. A d-dimensional polyhedral chain, x_p, in Euclidean space E^d is an expression of the form:

$$x_p = \sum_{i=1}^{m} p_i$$

where p_i are d-dimensional regular polyhedra (closed and convex point sets) in E^d. Unlike simple polyhedra, convex polyhedra are closed under all set operations (e.g., intersection, union, difference). The general polygon of the "MB" building (figure 1) is written as the sum of three convex polygons, $\mathtt{MB} = \mathtt{P}_1 + \mathtt{P}_2 + \mathtt{P}_3$ (see figure 3).

Convex chains can be viewed as a special case of constructive solid geometry (CSG). Like

Figure 3: Polygonal chain

CSG, the convex chains approach is a hierarchical representation scheme for polyhedra which is unambiguous but not necessarily unique.

Convex cells are represented by the means of the *h-vector*. Any convex polyhedron in E^d can be represented as the intersection of halfspaces in E^d. Each halfspace is viewed as a product $h \cdot H$ where H is an oriented (d-1)-dimensional hyperplane and h is an integer. In particular, $-1.H$ defines the halfspace which stands left of H, $1.H$ the right halfspace and $0.H$ the entire E^d. If $\overline{H} = H_1 H_2 \dots H_{|\overline{H}|}$ is the list of (d-1)-dimensional hyperplanes such that for each face f of any data object in the database there is a hyperplane in $\overline{H}$ that embeds f, then each cell p is represented as a ternary vector $h_p = \{0, 1, -1\}^{|\overline{H}|}$ such that $p = \bigcap_{i=1}^{|\overline{H}|}(h_p)H_i$.

Insertion of data objects is performed by adding new hyperplanes to $\overline{H}$, if necessary and similarly deletion is performed by causing some hyperplanes in $\overline{H}$ to become redundant.

This approach in the representation of polyhedral data objects, abandons the notion of vertex completely. The representation of cells by *h-vectors* has both conceptual and computational advantages. All set operators can be computed in two steps. At the first step an operation on the h-vector without any reference to geometric coordinates of the hyperplanes is performed. At the second step a generic operation, which removes the null vectors from the set of the resulting h-vectors, is applied.

The MERCATOR Representation

The MERCATOR system's representation model [Dav86] is capable of representing objects of different scales and various shapes in 2-dimensional domains. Among its novelties is the representation of any layout of objects, the representation of multiple shape descriptions for objects, and the description of partially known objects. In addition, it is able of stating measurements at varying degree of precision. Objects in a MERCATOR map are organized hierarchically for efficiency purposes. The most valuable feature of MERCATOR is that it can represent incomplete and imprecise geographic information and it is capable of approximate reasoning.

The basic representational primitive in MERCATOR is the line segment. Two-dimensional geometry is represented by straight line segments. Objects are represented by their boundaries and the description of their interior. An object's boundary is a set of edges connecting vertices. The interior of objects is represented by a set of convex polygons. There is no notion of coordinate system because this presupposes precision; only relative positions are specified.

Figure 4: Multiple region representation

Local dimensions are recorded in terms of lengths and orientation of edges. Instead of precise values MERCATOR uses range values (*fuzzy ranges*). Objects without straight line borders are approximated by polygons. The inaccuracy in a region's approximation is expressed in terms of the *grain-size*. A given object may have several region descriptions (which vary in the grain size) as shown in figure 4. Additionally, objects in a map are hierarchically organized by containment. The non-geographic properties of objects are not part of the MERCATOR representation; these are recorded in a slot-filler pair representation scheme.

The meaning of MERCATOR maps is defined in terms of truth conditional semantics which gives necessary and sufficient conditions that a map is a valid description of the world. There are facts which are true in the real world but not known to the MERCATOR map, e.g. scenes past a wall.

The topological and geometric layer distinction, following the Kuiper's and Levitt's hierarchy, is applied in MERCATOR as well. For instance, vertices, edges and polygons are considered as geometric layer entities and regions and shapes as topological entities. MERCATOR algorithms reduce the problems from the topological layer to the geometric layer. In addition, there are two notions of imprecision. Both fuzziness and grain-size are used to measure imprecision. In fact, they belong to different layers. Fuzzy ranges restrict possible position of the vertices (geometrical layer); grain-size describes the fit of the geometry to the object (topological layer).

The MERCATOR representation suffers from three basic inadequacies. First, it expresses only the presence of an object, not its absence. This can be fixed by using completeness statements (explicitly or using default rules). A reasonable completeness statement may be *all objects with property* p*, larger than grain-size* g*, inside region* R *have a corresponding clump in the map.* Second, MERCATOR cannot express natural combinations of precise shape and imprecise dimensions. Third, only limited range of worlds, namely 2-dimensional, are representable.

The two types of queries studied in the context of MERCATOR, are: (a) Where is object U with respect to object V? and (b) What objects are close to U? Two very important computations in answering these queries are first to determine relative position of two vertices and the second to assign consistent coordinates to vertices.

3.3.2 Space Filling Curves

The space filling curves [Lau85], [LM87] allow a one-dimension representation of any kind of d-dimensional data. In particular, there exists a bijection between a k-dimensional space to 1-dimensional space by the means of curve which passes only once through any point in the space. Any point on a curve is assigned a number, called *order number.* [2] Neighbouring points on the curve are generally close in space. Continuous regions of space are mapped into a sequence of curve segments or continuous a segments and therefore are characterized by a set of order numbers. The string length of the order number determines the resolution of the represented space.

Assuming a two-dimensional square area, the order number which characterizes any square partition in it, is determined by bit interleaving (this ordering scheme is calles *z-ordering* [OM84]). Increasing the number of bits on each side to infinity any point in the square can be identified. If for example we have x=10, y=11 then the order number p=1101 addresses a quadrant of the square.

Space filling curves can support proximity searches and point-in-polygon, point-in-polyhedron and region queries. So far, the Peano curves were used by many researchers in various contexts. In particular, Gargantini [Gar82] used them in the context of quadrees, Orenstein and Merrett [OM84], [Ore86] in the development of spatial access methods (see section 5.2), and Orenstein and Manola [OM88] for the representation of spatial objects. Other space filling curves such as Hilbert and Gray-code curves, were aslo used as spatial access methods [Fal88], [FR89b] and [Jag90a].

3.3.3 Hierarchical Partitioning

The hierarchical partitioning of the space is the basis for a class of methods for representing and organizing spatial data. The *quadtree* [Sam84], [Sam89] is the mostly used represenative of this class. Many quadtree variants have been proposed which are basically differ on the type of data they are used to represent and their space partitioning method.

The quadtree approach to region representation is called *region quadtree* (see figure 5). The region quadtree performs successive subdivision of a bounded (binary) image array into four equal sized quadrants. The quadrants that do not entirely contain 0 or 1 are further partitioned, in the same way. The region quadtree is a variable resolution representation method. The quadtree can also be extended to represent 3-dimensional binary region data and the resulting representation is called *octree.*

The quadtree is a tree structure which admits a straight forward pointer-based implementation (i.e, non-leaf nodes are represented as records with four pointers to their children).

[2] The most known space filling curve, other known curves are the Gray-code curves [Fal88], and the Hilbert curves, see [FR89b], [FR89a] for a survey.

Figure 5: Region quadtree

However, pointer based representations of quadtrees require considerable amount of space, and thus large images cannot fit into core memory. Consequently, there has been a vast interest in pointless representations [Gar82], [ABJN85].

Quadtrees are useful for performing set operations such as union (overlay) and intersection of several descriptions of the same region. The required time for these algorithms is proportional to the minimum of the number of nodes at corresponding levels of the two quadtrees. It also supports operations, desired in computer graphics, such as scaling by a power of 2 and rotations by multiples of 90 degrees. Area calculations are extremely easy, too. Lastly, it can be used as an image approximation device.

Many variants of the quadtree have been proposed for region and point presentations. A complete list and description of all of them appears in [Sam84]. Here, we will name only those that are referred to in the following discussion. These are the the point quadtree, the BSP-tree and the k-d tree.

3.3.4 Symbolic Projections

The symbolic projection scheme was introduced by Chang et al [CSY87] in the context of pictorial databases. According to this any two-dimensional shape is projected to two strings, called *2D-string*, along the vertical and the horizontal direction. It is an approximate representation scheme because dimensions of picture objects are known within a precision threshold.

Figure 6: 2G-string representation

Also, it is ambiguous since given a 2D-string many possible pictures can be constructed [3]. The original 2D-string supported only two spatial operators "<" (for right/left, bottom/up characterizations) and "=" (for denoting same-position). In [CL88], they extended 2D-strings by adding more operators. A later version, the 2D-H-strings [CL88] combines the advantages of 2D-strings and quadtrees. Jungert [Jun88] also extended 2D-strings with local operators. A more flexible 2D-string representation is the generalized 2D-Gstring. Figure 6 illustrates a 2D G-string representation of a part of the UofT map.

The 2D-Gstrings provide support for spatial inferencing. For instance, the geographic relationship **north-of** can be formulated by the rule:

$$\text{if } X : o_1 = o_2 \text{ and } Y : o_1 < o_2 \vee o_1 | o_2 \text{ then } (north, o_1, o_2)$$

Queries such as "find the relationship between objects E and C" (?, E, C) or "which object is north of B" (north, ?, B), can be answered by looking up the 2D-Gstring.

3.4 Summary

We presented several approaches to the representation of spatial data and a framework of characterizing spatial representations. A classification of different representations according to their application domain is shown on figure 7. The table 1 summarizes the results of our presentation. The abbreviated column headers UM, UQ, PK and IM stand for unambiguous, unique, partial knowledge and imprecise measurements respectively. The table also shows the modeling and the representation space for each representation scheme as well as the most important operations that it supports.

4 Spatial Reasoning

This section reviews formalizations of spatial knowledge which are suitable for spatial reasoning. Unquestionably, relative position computation is the most desirable task that a spatial reasoner must perform. Additional interesting tasks are to find the consistency of a given spatial configuration, the path finding problem, the intersection between objects with spatial extension and volume operations such as encountering objects within a given spatial range. All the above tasks are independent of application domains according to the initial goals of

[3] This claim is not true for the extended 2D strings [Jun88]

spatial representation	modeling space	representation space	UM	UQ	PK	IM	coord. frame	operations
TOUR	terrain	network of places & paths	no	no	yes	qualitative variables	N local 1 global	
SPAM	terrain	frob tree	no	no	yes	fuzzy values	N local	
MERCATOR	terrain, objects of any shape	polygonal approximation	no	no	yes	fuzzy values	no	rel. position neigboring
FORB	bouncing ball world	space graph, motion diag.			yes			
primitive instancing	3-d polyhedra	set of prim. solids	yes	yes	no	no	no	- set ops union intersect. difference - translation - rotation - scaling
sweep rep.	3-d solids	axis, sweep rule	yes	no	no	no	no	
BR	3-d polyhedra	set of polygons	yes	no	no	no	no	
spatial occupancy	3-d polyhedra	octree, quadtree	no	yes	no	no		
CSG	3-d polyhedra	CSG tree	yes	no	no	no		
h-vectors	k-d polyhedra	polyhedral chains	yes	no	no	no		
space filling curves	k-d regions	1-d curves	yes	yes	no	coarser grid	orthog.	overlap containment
quadtrees	2-d regions	quadtree	no	yes	no	coarser grid	orthog.	overlap rotation x90 scaling x2
symbolic projection	2-d regions	2-d strings	no	yes	no	no	orthog.	spatial order

Table 1: A characterization of spatial representations

Figure 7: Application domains and spatial representations

this study. Application specific operators can be defined on top of them. In the following, some requirements which are essential for a spatial reasoning system are listed.

1. It must be capable of representing and reasoning about a variety of spatial constraints both qualitative and quantitative.

2. It must be capable of inferring spatial consequences given some specification of spatial and motional relationships.

3. It must be capable of reasoning in the presence of uncertain information.

4. It must be capable of reasoning about objects of various shapes and varying degree of granularity.

5. It must be independent of coordinate frames.

For us, the support of uncertainty of various forms is the main requirement. In this document, we distinguish two forms of uncertain spatial knowledge, the one is uncertainty of a spatial representation which is due to either total or partial ignorance of a spatial configuration (incomplete knowledge), and the other is imprecision of the spatial knowledge. In the first case we have to cope with incomplete knowledge in a similar way as in symbolic knowledge representation i.e., doing completion assumptions (e.g., nothing exists behind non-transparent obstacles) and applying persistency rules (e.g., "if an object A is behind an object B and A's dimensions are unknown, assume that they are the same as A's until something new is learned"). Imprecision emanates from limited accuracy (e.g., "the height of the Pratt building is between 14 and 16 meters"). Imprecision requires totally different treatment; it can be expressed through constraints in qualitative or approximate terms.

Several approaches in spatial reasoning have been proposed. Some of them formulate the problem as a constraint satisfaction problem over a network of spatial relations in the same way that reasoning with respect to time is formulated. Other approaches involve qualitative reasoning, approximate calculations based on fuzzy numbers, and numerical methods. In some cases, spatial reasoning can be thought of as a generalization of temporal reasoning in multi-dimensions.

4.1 Temporal reasoning and constraint networks

Temporal reasoning systems are broadly divided into those which deal with temporal intervals and those which deal with time points. Points or intervals are related with binary temporal constraints. The constraints are either disjunctions of simple constraints or simple constraints. They can also be qualitative or metric. In both cases, the goal of temporal reasoning is to see if this information is consistent, or to find all feasible relationships between all pairs of points or intervals. These reasoning tasks are formalized as a constraint satisfaction problem [DMP89], [vB90] or a label inferencing problem [Dav87] over a temporal constraint network, respectively. A temporal constraint network consists of a set of temporal variables $X_1, X_2, ..., X_n$, where D_i is the domain of each variable, and a set of unary and binary constraints over these variables. A solution of the network is a tuple $\{x_1, x_2, ..., x_n\}$ such that the assignment $X_i = x_i$ satisfies all the constraints. A network is consistent if at least one solution exists. Once we find a consistent network we are interested in a minimal network. A minimal network is a network with the same solution and the tightest possible constraints.

Allen proposed an interval based representation with thirteen mutually exclusive basic relations between intervals and their disjunctions. The set of the thirteen basic relations is I={eq, b, bi, m, mi, o, oi, d, di, s, si, f, fi}. If we consider all possible disjunctions of them, then there are $2^{|I|}$ different relationships which are possible between any two intervals. Allen's interval algebra (IA) is defined as the set of the $2^{|I|}$ relations, the unary inverse operator and the binary composition and intersection operators. Allen [All83] proposed a constraint propagation algorithm, for computing the transitive closure in his algebra. His algorithm runs in time O(n^3), where n is the number of intervals. Unfortunately, this algorithm is incomplete. The basic idea of this algorithm is based on previous algorithms for constraint satisfaction problems [Mon74], [Mac77], [MF85]. Vilain and Kautz [VK86] showed that computing the transitive closure of relations in Allen's interval Algebra is NP-complete problem. Valdes-Perez in [VP86] shows that Allen's constraint propagation algorithm is sound but not complete and he developed a dependency-directed backtracking algorithm [VP87] with exponential asymptotic complexity which is complete. In practice this algorithm terminates early because of quick pruning and clever backtracking.

Vilain and Kautz [VK86] defined a point algebra (PA) which allows the representation of a restricted set of Allen's relationships between intervals, and in particular they eliminated relationships which cannot be expressed as disjunctions involving a single endpoint. They use only three basic relations, $<, =$ and $>$, to relate time points and their disjunctions (i.e., 2^3=8). Their point algebra (PA) consists of eight relations $\{\ \emptyset\ <, \leq, =, >, \geq, \neq, ?\ \}$, the unary inverse operator and the binary operators composition and intersection. Then, they gave an O(n^3) algorithm for computing the transitive closure of relations on a point graph. VanBeek showed [vB89] that this algorithm is not complete either, and specifically the $\neq$ is the relation which causes the incompleteness. In fact, he came up with a O(n^4) algorithm which is complete for the point algebra. All the above methods deal with abstract time arguments. Better performed algorithms are possible when absolute time arguments are considered. Ghallab and Alaoui [GA89] propose a modification of Vilain and Kautz algorithm providing an indexing method which works in linear time and quadratic space; note that this is an empirical result. Miller and Schubert [MS88], support only the $<$, $\leq$, $=$ relations and organize time points in chains ordered by $<$, $\leq$. Their algorithm computes the relative position between two points in time $O(m)$, where m is the number of the cross chain links. Moreover, Koomen [Koo89] proposes an improvement on Allen's algorithm, by introducing an automatic referencing hierarchy based on interval containment. This reduces considerably the connectivity of the network and therefore the cost of constraint propagation. Empirical results show the superiority of the automatic referencing method over the no-referencing approach.

Constraint networks were also used for temporal reasoning for cases where absolute (e.g., "the SIGMOD-91 conference started on May 28, 1991 ..") and metric (" ..and it lasted for 4 days") temporal information is known. Metric constraint networks (MC) are studied by Dechter, Meiri and Pearl in [DMP89] and Schmiedel in [Sch88]. The problem is again formulated as network consistency problem with the nodes to represent continuous variables and the edges to express constraints between them. If the constraints are simple linear inequalities then the problem is solved in polynomial time. If constraints are disjunctions of linear inequalities the same problem becomes exponential and then they proposed a relaxation method. Dean [Dea89], [DM87] provides a linear time solution for a special case of the [DMP89] network, which is called distance graph. Dean's linear algorithm is based on indexing and caching of time intervals. Davis [Dav87] studies the same problem with [DMP89] but in its more general case. In particular, he studies the soundness and the termination of the constraint propagation algorithm for various types of constraints. According to Davis' results the threshold form polynomial to exponential complexity are the linear inequality constraints.

4.2 Spatial Reasoning and Constraint Networks

Extensions of temporal representations for **k** dimensions is one possibility for spatial representation and reasoning [Lig90]. The main advantage of this approach is that the whole body of work developed for time applies to space as well. However, the generalization of temporal representations for space is not unrestricted. This approach restricts reasoning to orthogonal domains. The key properties that time and space share, and make methods from temporal reasoning valid for spatial reasoning, are order and strictly increasing continuity (in the calculus sense) [VP86]. Both these properties are possessed by the orthogonal cartesian axes. Space, unlike time, is static in our world-view, therefore the provision for persistency that is made in many temporal representations is not applied to space unless we reason about motion. However other forms of persistency may be explored.

Allen's framework is directly applied to spatial reasoning when dealing with spatial relations on one dimension. Relations like front-of, back-of, and inside correspond directly to after, before and during. Of course, there are very few domains where we can restrict spatial reasoning to one dimension. For instance, one can think that the way to the library building is 1-dimensional and that someone who is on the way to the library is (spatially) ahead of him/her. One-dimensional spatial relations based on Allen's model are mentioned in [MJ90] and [Gue91]. In particular, Makerjee and Joe simplify Allen's model in a fashion similar to that of Vilain and Kautz. They identified five tangency cases between the endpoint of one interval and another interval; these are: ahead (+), front (f), interior (i), back (b) and posterior (-). The possible relations between two objects still remain 13 but they reduce the size of the transitivity table from 13x13 to 13x5, exploiting the fact that the rear endpoint is back from the head endpoint. Guesgen, uses eight interval relations to describe spatial relationships, namely left-of, attached-to, overlaps, inside and their inverses. Exactly the same relations are also proposed by Malik and Binford [MB83]. All [MJ90], [Gue91], [MB83] and in addition [LJ88] think that in k-dimensional orthogonal space, the spatial operators can be defined as a vector of lenght k of interval operators. This approach, let us call it *partitionable*, reduces the reasoning to k independent dimensions, any answer must be the conjunction of k answers. The opposite approach is to form one constraint network for all dimensions. This approach is called *unified*.

4.3 Qualitative Methods

Qualitative reasoning methods are well suited to spatial problems because of their expressive power for states of incomplete knowledge. Representations of space are in general incomplete. In a qualitative reasoning system, the values of parameters and functions are represented in terms of *quantity spaces* [Kui89]. Each quantity space is defined by an ordered set of *landmark values*. Systems are described by the means of qualitative constraints. Many of the methods reviewed above fall in the definition of qualitative reasoning systems.

4.3.1 Qualitative Constraint Networks

Extensions of Allen's and Vilain's-Kautz's frameworks in higher dimensions are typical instances of spatial qualitative constraint networks. Makerjee and Joe [MJ90] presented one such system. In their method, the relative position of two objects is expressed by the position of the one with respect to the other and their relative direction. Relative direction's quantity space for the 2-dimensional space, can be thought as a quantity space with four landmark values, one for each quadrant of a plane.Therefore, they derive qualitative constraints which characterize the position of objects in space. Their computational model is constraint network based.

4.3.2 Directional Relations

In everyday life, it is very common to describe relative position between objects using geographic directions, { E(ast), W(est), N(orth), S(outh), NE, NW, SE, SW } [Fra91]. Obviously, this representation is suitable only for two-dimensions. Besides, it can be seen as a 2-dimensional version of Allen's representation. The directional relationship A EastOf B is translated into the disjunctive 2-dimensional interval relationship $\mathsf{A} < \mathsf{b}, \mathsf{o} \vee \mathsf{d} \vee \mathsf{oi} \vee \mathsf{di} \vee \mathsf{e} \vee \mathsf{s} \vee \mathsf{si} \vee \mathsf{f} \vee \mathsf{fi} > \mathsf{B}$ [MMT89]. Respectively, A NorthEastOf B is equivalent to $\mathsf{A} < \mathsf{b}, \mathsf{bi} > \mathsf{B}$.

Spatial inferences based on directional relations are drawn either by propagating constraints over a network of directional constraints or reducing the reasoning to the underlying 2-dimensional Allen-type constraint network. Both of the approaches perform approximate reasoning. The first one is taken in [Fra91]. In fact, this algorithm returns results with confidence 50%.

Directional relations (DR) are not expressive enough to characterize a spatial configuration since they only deal with directions. An addition to this model is to use distance based qualitative relations such as close, far and tagent. Then, the relative position characterization makes use of two two qualitative quantities with quantity spaces {0(tagent), 1(close), 2(far)} and {N, E, W, S, NE, NW, SE, SW} respectively.

4.3.3 Qualitative Mechanics

Qualitative spatial reasoning is discussed extensively in Kinematics and Naive Physics [For80], [FNF87], [Fal87], [Sho85], [NC91]. Typical reasoning tasks in kinematics are to find potential connectivity relationships, kinematic states and state transitions. As one can observe, spatial reasoning in qualitative mechanics has a dynamic flavor (we could call it *spatio-temporal*) as opposed to spatial reasoning in geographic domains. It is also proper to say that the definition and the developement of qualitative reasoning is motivated by this kind of problems.

4.4 Quantitative Methods

Quantitative methods in spatial reasoning make use of metrics, absolute values, range values and comparisons. Suppose that the spatial configuration of figure 1 is supplemented with the following descriptions: "The distance from SF to the library is twice as long as the distance from MD to the library" (metric); "The height of GB is precisely 20 meters" (absolute); "The distance from LP to MD is between 20 to 30 meters" (range); "The height of SF is the same as LP building" (comparison). These *metric* constraints can be formalized as (in)equalities between the endpoints of the x, y and z coordinates of the objects.

The computational methods which have been proposed for spatial reasoning in the context of metric information are the metric constraint networks [DM87], [Dea89] [DMP89] and the linear programming [MB83], [McD82], [Dav86].

4.5 Semi-Qualitative Approaches

The combination of qualitative and quantitative representation is attempted in cognitive maps [McD80], [MD84], [KB88] and in constraint networks [Lad89]. In fact, any cognitive map which follows the layered model of [KL88] consists of a quantitative component (metric layer) and a qualitative component (topological layer). Reasoning in cognitive maps (spatial learning, path finding) exploits both representations. Kuipers [KB87], [KB88] derives distinctive places of the map using hill-climbing search on the metric layer and then applies qualitative simulation [Kui89]. McDermott and Davis [McD80], [MD84], [Dav86] transform the problem to a network of fuzzy constraints and apply numerical techniques (Monte Carlo algorithm). The semi-qualitative approach gains support in qualitative kinematics (poverty conjecture) [FNF87]. Lastly, it seems more convincing from the cognitive perspective too.

A graceful integration of the qualitative and the quantitative approach in spatial reasoning is proposed by Dutta [Dut89]. In this method, the relative position of objects is expressed in qualitative terms, e.g., "SF is near LP". Imprecision is expressed using range data, e.g. "the distance between the SF and the LP is between 4 and 6 meters". The characterization of near above, depends very much on the context [Rob89]. The general problem is, given a set of objects and a set of constraints (generally incomplete and sometimes conflicting), to find all the induced spatial constraints. The constraints are represented as *possibility assignment equations* of the form:

$$p \rightarrow \Pi_{(x_1,..,x_n)} = F \qquad e.g., \qquad \text{“Bob is short”} \rightarrow \Pi_{\mathrm{Height(Bob)}} = \mathrm{SHORT}$$

where p is a natural language proposition, Π is the possibility assignment and F (e.g., $SHORT$) is a fuzzy subset of the universe of discourse U and $\mu_{SHORT}(Height(Bob))$ is the membership function of $Height(Bob)$ in set $SHORT$. An approximation in the fuzziness expression is to use fuzzy numbers *(c,l,r)* (i.e., triangular distribution) instead of $\mu(u)$, e.g., about-five-miles=(5,1,1), north-eastern-direction=(45,10,10). For metric constraints the membership function is defined: $\mu_A(u) = 1$ if $u \in A$ and $\mu_A(u) = 0$ if $u \notin A$, therefore the fuzzy set has only one value ($\mu(u) = 1$) and the fuzzy number is *(c,0,0)*, e.g., "Object A is 5 miles NE of object B" $\rightarrow \Pi_{loc(A),loc(B)}=$ (5miles, NE), where, 5miles=(5,0,0) and NE=(45,0,0). The mathematical basis for his spatial reasoning algorithm is the composition of possibility assignment equations and the fuzzy set theory.

Dutta's algorithm works in two steps. Firstly, it computes the spatial constraints on relative positions of any two objects using all possible paths between them, and secondly, check all the conflicting answers for consistency. The algorithm works for k dimensions following the partitionable approach. For each dimension it forms a graph, where nodes represent objects and arcs represent known spatial constraints; the graph is encoded as a *spatial constraint matrix*. The transitive closure of any of those matrices is computed by a generalization of Warshall's algorithm in $O(n^3)$ time.

4.6 Axiomatic Approaches

An axiomatic approach in spatial reasoning is taken by Shoham [Sho85], Davis [Dav89], Roman [Rom90] and Kaufmann [Kau91]. Kaufmann's work is a unique proposal in spatial reasoning that adopts an ontology for time and space based on tolerance spaces. An interesting aspect of such spaces is that at sufficiently small scales, positions or quantities are indistinguishable. In other words physical properties are interpreted over a grid structure and theories like $\{A = B, B = C, A < C\}$ have a true interpretation. Direction, distance, orthogonality and angle are defined in tolerance spaces as well but they have substantial difference from the corresponding notions in the Euclidean space. Reasoning in this system is approximate and is proposed as the main competitor to the qualitative approach in reasoning about physical systems.

method	rel-pos(A,B)	constraints	modeling domain	algorithm
[MB83]	A_x left-of B_x A_y above B_y	linear, metric	1-,2-,3-dim, orthogonal	simplex
[MJ90]	pos(A,B)=ii pos(B,A)=- - dir(A,B)=2	qualitative	k-dim, orthogonal/ angular	constraint propagation
(2-dim) IA	A <b,{b,o}> B	qualitative	k-dim, orthog.	const. propag.
(2-dim) PA	$A_x < B_x$ $A_y = B_y$	qualitative	k-dim, orthogonal	constraint propagation
(2-dim) MC	$B_x - A_x$=[2,4] $B_y - A_y$=[1,5]	quantitative, disjunct./simple	k-dim, orthogonal	
MERCATOR [Dav86]	dist(A,B)=[6,7] rot(A,B)=[20,30]		2-dim	monte carlo & const. propag.
SPAM [MD84]	x(A,B)∈[2,4] y(A,B)∈[5,9] scale(A,B)∈[1.2, 1.8] rot(A,B)∈[30,35]	quantitative	2-dim, multiple coord. frames	numerical optimization
[Dut89]	$\Pi_{A,B}$=([5,0,0], [45,5,5])	semi-qualitative	k-dim	Warshall
DR	A NorthEast B	qualitative	2-dim	const. propag.
[Kau91]	tolerance-preserving function	–	k-dim tolerance space	methematical induction

Table 2: Spatial reasoning

4.7 Summary

Table 2 summarizes the presented approaches on spatial reasoning. For any method, the table shows its machinery for relative position specification, its constraint language, its modeling domain and the basic algorithmic approach that it takes.

5 Spatial Databases

This section reviews the research and the open issues in spatial databases. Spatial databases store k-dimensional data representing explicit knowledge about objects, their extent and position in space [GB90]. Spatial databases are widely used in applications such as geosciences, mechanical CAD, robotics, environmental protection, medical imaging, etc. Work in this area is broadly divided in spatial data modeling, data structures and access methods, and query processing and optimization.

The requirements for spatial data management differ widely between applications. In mechanical CAD the representations consist of line drawings of 3-dimensional solids. As we have already seen, cell decompositions, constructive solid geometry and boundary representations are suitable representation schemes for such applications. In practice, a single representation scheme does not satisfy all the application requirements and therefore data models which support multiple representation schemes are needed. Furthermore, the data model must support the association of a spatial representation with descriptive properties such as weight, material, etc. VLSI CAD uses simpler and tractable geometric representations. Geographic applications deal with 2-dimensional spatial data (e.g., points, lines and polygons) which are characterized by extreme irregularity. A database system for such applications must gracefully integrate non-spatial and spatial data and its query processor must be extended to support operators such as overlay, containment, distance, and various spatial searches. In vision systems the key problem is to identify the position of a 3-D object based on a 2-D image and sometimes this is required in real time.

5.1 Spatial Data Models

The fundamental question in spatial database modeling is how to embed spatial aspects in a data model and the underlying DBMS such that acceptable interfaces (optimizable query languages and pictorial interfaces) can be defined. A spatial data model must support modeling of spatial data (e.g. points, lines, polygons, solids and combinations of them), modeling of the non-spatial aspects of objects and lastly must support basic spatial operators. Kemper and Wallrath [KW87] studied the database geometric modeling problem for a specific domain. Their work concentrates on solid object modeling and considers the BR and CSG representations and operations such as translation and scaling. Others [OSD89], [SW86] concentrate on geographic data and consider intersection, adjacency, join and enclosure operators.

Most of the proposed data models for spatial data are associated with one of the dominating database modeling approaches, specifically, the relational model, the extended relational systems, the object oriented principles and lately the knowledge representation approach to

MAP

OID	TYPE
B1	block
B2	block
SF	building
LP	building
S1	street

OBJECT

OID	EID
B1	e1
B1	e2
B1	e3
B1	e4
...	...

EDGES

EID	VID
e1	v1
e1	v2
e2	v2
e2	v3
...	...

VERTICES

VID	X	Y
v1	0	0
v2	0	20
v3	40	20
v4	40	0
...	...	...

Figure 8: A relational implementation of the BR scheme

database modeling.

The relational data model has been used in engineering databases [GS82] and in geographic information systems [CF81], [CK81], [AS86]. This approach has many drawbacks. It cannot support hierarchical construction of spatial objects. Operations are very hard to compute and are left totally out of the data model. For instance, the retrieval of the bounding edges of all blocks in figure 1 requires an expensive join operation between `BLOCKS` and `EDGES` and `VERTICES` relations, assuming the following BR relational schema:

The drawbacks of the relational approach guided many researchers to extensions of the relational model. The above example also made an intimate suggestion to an extension of the relational model by using tuple identifiers. Other suggestions to extend the relational model include the "QUEL as a Datatype", the ADT-INGRES [SRG83], the nested relations in NF^2 data model[PSS$^+$87], the introduction of surrogates and long fields in system R [LP83], the entity relationship approach taken in GEM [Zan83], the DAPLEX functional data model [CDF$^+$82], the R^2D^2 model [DKA$^+$86] and many others.

Kemper and Wallrath characterize the above methods with respect to their ability to represent the structure and the manipulations over geometric entities. In other words, they talk about structural and behavioral object orientation. A structurally object-oriented data model provides facilities for mapping complex objects onto database structures and for retrieving these objects as entities but it usually lacks constructs for defining manipulations over these objects. The behavioral approach has its origins in programming languages and provides an abstract view in objects' manipulations, particularly by promoting the notion of abstract data types (ADTs). "QUEL as a Datatype" and GEM are able to model structurally complex entities via their powerful referencing mechanisms. DAPLEX supports structure through the generalization and the aggregation abstraction principles. The extensions of System R and the NF^2 data model provide constructs for modeling complex structure. The figure 9 illustrates the NF^2 relation which corresponds to the relational schema of figure 8. As shown, object composition is supported directly in nested relations. However, there is a redundancy overhead. Another drawback of nested relations is that they cannot support specialization. ADT-INGRES provides some level of behavioral object orientation by allowing the definition of ADTs with the corresponding operations. The R^2D^2 model goes a step further since it integrates the structural capabilities of the NF^2 model with the ADT concept. Apart from

MAP					
OID	TYPE	EDGES			
		EID	VERTICES		
			VID	X	Y
B1	block	e1	v1	0	0
			v2	0	20
			v3	40	20
			v4	40	0
		e2	v2	0	20
		...	...	...	...
B2	block	...	...	...	...
		...	...	...	...
...	...	...	...	...	...

Figure 9: An NF^2 implementation of the BR representation

specific drawbacks that any of these models have, the common drawback is the lack of generality and Kemper's study is a very good example. Some of them are able to model CSG easier than BR or the other way around. Some operations are very hard to be computed and impossible to be extended. These drawbacks guide to a third approach in spatial data modeling. Instead of defining extensionsm it is preferable to define extensibility. This approach is taken in the PROBE [OM88] object-oriented data model and is also feasible in a knowledge model like Telos [MBJK90].

The **extensibility** of the OODBs allows the inclusion of specific object classes that support the required data types (spatial data) and the required manipulations are done by specialized processors which are encapsulated in object classes. More specialized object classes are added as needed and customized optimizers are built along with them.

PROBE's data model (PDM) [MO86] represents spatial characteristics of entities and integrates them with their non-spatial characteristics described in an ordinary data model, DAPLEX, by introducing an entity type (class) called PTSET. Entities of type PTSET are geometric entities (e.g., lines, areas, volumes). Their purpose in the data model is to serve as values of spatial attributes such as `boundary` or `shape` of ordinary database entities. The PTSET type can be specialized to obtain entity subtypes with more specific spatial features (e.g., specific shapes). Complex shapes can be built by combining primitive ones. Specialized types can also have specialized functions that are applied to them. A PTSET can contain other PTSETs and the container is called *space*. The operations that are associated with generic PTSET entities are point set (e.g., spatial) operations and structural operations. The point set operations include set operations on PTSET entities (point set intersection, union and difference), spatial selection, overlay and geometric transformations. The structural operations are concerned with the hierarchical structure of spaces and are defined in terms of the non-spatial operators of the PDM algebra. Next example illustrates the structural facilities of PDM.

```
type MAP is ENTITY
   title(MAP) --> STRING
   scale(MAP) --> INTEGER
   area(MAP) --> PTSET
   feature(MAP) --> set of FEATURE

type FEATURE is ENTITY
   entity-type(FEATURE) --> FEATURE-TYPE
   feature-id(FEATURE) --> INTEGER
   shape(FEATURE) --> PTSET

type ROAD is FEATURE
   name(ROAD) --> STRING
   crosses(ROAD) --> set of ROAD
   length(ROAD) --> REAL

create new R in ROAD (
   name ==> "College Str",
   crosses ==> {"Univ. Ave","St George"},
   length ==> 296.89 )
```

Point sets are implemented by the *Approximate Geometry* (AG) component of the DBMS. The AG component consists of a query processor and a storage facility for spatial objects. Moreover, it implements optimizations, and make use of an *object-at-a-time* interface to spatial object classes [Ore86]. The storage structure of AG is based on a grid representation of spatial objects. Each PTSET entity is decomposed into box-shaped elements. The representation of PTSET entities by a union of box-shaped elements is approximate because it depends on the granularity of the grid decomposition. Each grid element is assigned a *z-ordering* number [Ore86] (recall the discussion on space filling curves in section 3.3.2). The use of the z-ordering numbers reduces the precedence and containment operations to simple bit-string operations. Therefore, approximate spatial operations are computed by very simple algorithms [OM88]. Once more, the operations computed by the AG are approximate. Further refinements on the output of the AG are done in a second phase of the query processing.

The bject-oriented approach in spatial data modeling have been tried also in [SH88], [MK88], [vOvdB89] and [AOG+88].

The suitability of knowledge representation languages in spatial modeling is acknowledged in [MMT89], [MK88]. Milios et al. [MMT89] introduced GeoTelos, an extension of the language Telos [MBJK90], to deal with spatial domains. GeoTelos allows one to organize objects with spatial features (`GeoConcepts`) using aggregation, generalization and classification abstraction principles, and express integrity constraints and deductive rules. In particular, deductive rules can be seen as an extensibility for user defined spatial operators. Another possible extensibility in Telos through the meta-attributes facility and the way which offers to customize the data model. The early design of GeoTelos proposed a set of spatial operators which are a generalization of Allen's temporal relations for two dimensions. Qualitative and quantitative queries are supported through the ASK and the RETRIEVE facilities of Telos.

```
TELL GeoConcept
   IN M1_Class

TELL Map
   IN GeoConcept, S_CLASS
```

```
   WITH
      attribute
         spatialDesc:SpatialClass
         surfaceValue:ValueClass
END GeoConcept
```

```
   WITH
      spatialDesc
        area:SpatialIndividual
      surfaceValue
        title: String
        scale: Real
        feature: Feature
END Map
```

```
TELL UTmap
   IN Map, TOKEN
   WITH
      area
          :"~thodoros/maps/uoft.ps"
      title
          :"UofT Campus Map / St George Campus"
      scale
          :.0004
      feature
        bl1:block1
        bl2:block2
        rd1:road1
        rd2:road2
        ....
END UTmap

ASK x/Block: x in UTmap.feature and x.area [over,over] SF.area
```

This first design does not explore all language features towards the spatial modeling direction. The extensions of Telos towards a knowledge base management system [KT91] with efficient storage structures, query processing and concurrency control make it a very promising tool for spatial data management.

5.2 Spatial Access Methods

This section discusses representation of spatial data at the physical level i.e., spatial data structures and access methods.

Almost all known spatial data structures are based on the principle of partitioning the space into cells. A two way mapping relates regions of the space to the cells that they fill up. Cells or sets of cells are assigned to disk blocks. Access of spatial objects, given a region of space, consists of finding the cells that intersect the region (*cell addressing*) and then finding the objects that occupy those cells (*data access*) [Nie89]. The general concern during accessing of spatial objects is *minimal number of disk accesses* and therefore good access methods are necessary.

General requirements for spatial data structures are (a) efficient support of insertions and deletions (should be dynamic), (b) support of a wide range of spatial queries, (c) equal treatment of all dimensions in a k-dimensional space, and (d) performance independent from data and query distribution [See90].

In the following we assume a d-dimensional data space with d $\geq$ 1. Spatial data objects in d-dimensional space are approximated by their minimum bounding rectangle (MBR). Methods for storage and accessing of (high dimensional) rectangles fall in three categories [SRF87]: (a) Methods that transform the rectangles into points in a space of higher dimensionality; hereafter we will refer to them as *point access methods.* (b) Methods that use space filling curves to map a k-dimensional space into a 1-dimensional space. (c) Methods that decompose the space into sub-regions and distribute appropriately the data objects. Seeger and Kriegel [SK88] present the latter type of methods as a combination of a point access method with one of the techniques of clipping, overlapping and transformation and they call them spatial access methods.

Assume a d-dimensional data space, S, where data in each dimension, i, range over an ordered domain, D_i, $0 \leq i \leq d-1$. A point in S is an element of $D_0 \times D_1 \times \ldots \times D_{d-1}$. Respectivelly, a region R consists of all points $(x_0, x_1, \ldots, x_{d-1})$ satisfying $lo_i \leq x_i \leq hi_i$, $0 \leq i \leq d-1$. Indexed records in S are of the form $(k_0, k_1, \ldots, k_{d-1}, id)$, [4] where k_i is an element of the D_i, $0 \leq i \leq d-1$, and id is a pointer to the real data object.

The spatial queries that are examined in the bibliography include: (a) point search, (b) range search, (c) partial match and partial range queries, (d) nearest neighbour queries, (e) spatial join and (f) zoom-in queries.

For point data, given a certain point in the space, the point search finds the corresponding data object. Respectively, the range search returns all the data objects that fall in the specified region. Partial match refers to the case where one or more components of the key are unspecified. Similarly, the partial range search refers to the case where some of the upper or lower bounds of the region are left unspecified.

For extended data objects, the point query returns all data objects that contain the given data point in space. The region query retrieves all objects that intersect with the user specified window. Spatial join between two regions R and S returns all pairs of data objects r and s such that they are in R and S respectively and they overlap each other.

As stated, the basic principle in all spatial access method is the partitioning of space. Kriegel et al. [KSSS89], and Seeger [See90] propose a classification based on the kind of partitioning which in fact can be rectangular, complete and disjoint. The combinations of these parameters designate nine areas that are used to categorize the spatial access methods. Sellis et al. [SRF87] identify three other attributes of partitioning, namely position, dimensionality and locality. The position of the split can be fixed or adaptable. In terms of dimensionality, the partition can perform an 1-dimension cut or k-dimensions cut. Lastly, as far as locality concerns, the splitting hyperplane splits only the affected regions (brickwall method) or all regions and results in a grid structure. Table 3 (based on [See90]) lists the most commonly

[4] The tuple $k_0, k_1, \ldots, k_{d-1}, id$ is called *key*

rectangular	complete	disjoint	multidimensional access method	
*	*	*	Excell	[Tam82]
*	*	*	grid file	[NHS84]
*	*	*	PLOP-hashing	[SK88]
*	*	*	skd-tree	[OSDM89]
*	*		twin grid file	[HSW88]
*			R-tree	[Gut84]
*			SR-tree	[KS90]
*			R*-tree	[BKSS90]
*		*	R+-tree	[SRF87]
*		*	Buddy-tree	[SK90]
	*	*	ZB+-tree	[OM84]
	*	*	GBD-tree	[OS90]
	*	*	hB-tree	[LS90]
	*	*	BANG-file	[Fre87]

Table 3: Classification of spatial access methods (based on [Seeg90])

known multidimensional access methods. In the rest of this section we review some of the methods of table 3. For clarity of the presentation we use the example data spaces of figure 10. Figure 10(a) represents buildings from the UofT map by their centroids (as points) and 10(b) represents the same portion of UofT map using the buildings' enclosing rectangles.

5.2.1 Multidimensional Trees

The most commonly known multi-dimensional data structures are the kd-tree [BF79] and the kd-trie [Ore82]. Both of them are variants of binary trees.

The kd-tree [BF79] organizes the data space as a binary tree by dividing it into segments using a cyclical order, that is, it chooses split axis following a cyclic order of dimensions. A split is performed any time that a tree node overflows. After the split old records remain in the same node while the new one goes in a new node. This results in an unbalanced tree where the record distribution depends heavily on the order of the insertions. The kd tree for the above example is shown on figure 11(a).

The kd-trie [Ore82] follows a similar idea as the kd-tree. Their major difference is in the position of the partition within the divided region. The split in a kd-trie always divides a region into two sub-regions of equal size. Therefore, a kd-trie is less balanced than a kd-tree but on the other hand the partition is independent of the sequence of data and the sequence of insertions. Subregions in a kd-trie are represented by a bit pattern, that results in a more compact secondary storage implementation than the kd-tree and fewer disk accesses in data retrieval. Figure 11(b) illustrates the kd-trie that corresponds to the kd-tree of figure 11(a).

When multidimensional data reside on secondary storage, efficient data access requires use

Figure 10: Examples of 2-dimensional data spaces

of access methods. Access methods for multi-dimensional point data consist of a directory and a set of data pages. The directory is a grid or a tree organized grid or a table or a hash table and maintains a mapping data space regions and data pages.

5.2.2 Point Access Methods

The point access methods presented here are the grid file [NHS84] and its successors BYOP file [OO85] and BANG [Fre87] file, the k-d-B-tree [Rob81], ZB+-tree [OM84], Buddy-tree [SK90] and PLOP-hashing [SK88].

The grid file [NHS84] is a representative of a whole class of grid directory based access methods. The grid file performs a grid partition of the data space in a complete, disjoint and rectangular fashion, and assigns grid regions to data buckets. The data structure used for that is called grid directory. Due to its size the grid directory is stored in two storage levels. It consists of a dynamic k-dimensional array whose elements (data buckets) are in one-to-one correspondence with the grid blocks of the partitioning, and k one-dimensional arrays, each one of them defining a partition of a domain D_i. In grid file, it is very likely that neighboring objects are stored close to each other in neighboring pages. In terms of performance, the grid file guarantees two disk accesses for point queries and efficient range queries with respect to all attributes. The basic drawback of the grid file is its bad behaviour with non-uniform data. Specifically, when a bucket which corresponds to a grid region overflows, the bucket splits into two; this introduces a new partition which splits all regions across its plane (although only the overflowed bucket is physically divided).

Relatively better behaviour with non-uniform data distributions is achieved by the interpolation-based grid file (or BYOP file) [OO85]. The BYOP file combines the grid partition with a hierarchical division of the data space. Essentially, its difference from the ordinary grid files is, first, that the partition of the overflown regions is local and second that it assigns a number as an address to any grid directory region. The BANG (Balanced And Nested Grid) file

[Fre87] is another variation of the grid file (it is based on the interpolation grid file) which has the best behaviour for non-uniform data distributions. The directory of the BANG file is organized as a tree structure on the same principle as the other grid files. All levels above the first level are partitions of partitions (nested partitions). Any region which corresponds to a disk block, is assigned a block region identifier which consists of a tree level number and a region number (as BYOP file). This organization permits the balancing of data distribution between regions by redistribution of nested regions. Finally, the partition covers the whole space (complete), is disjoint and not rectangular. The form of the grid directory for the ordinary grid file, the BYOP file and the BANG file are shown on figures 11(e)-(g). A very similar idea to the grid file's is used in EXCELL [Tam82].

The ZB+-tree maps a k-dimensional space into 1-dimension following the convention of the z-ordering. Any data record is associated with a z-value, i.e., a bit string, which can be represented very easily by an integer; therefore, a one dimensional indexing method such as a B-tree can be used. Orenstein and Merrett show how index sequential access data structures can be generalized to multi dimensions by applying the z-ordering. The advantage of ZB+-trees is that they inherit all properties of B+-trees, i.e. high storage utilization and dynamic behaviour. The performed partitioning is complete non-rectangular and disjoint (figure 11(c)). Orenstein and Merrett demonstrated cases where the grid locality is not preserved during splitting.

The k-d-B-tree [Rob81] is a combination of a kd-tree and a B-tree. Its partition scheme is complete, rectangular and disjoint and it depends on the distribution and the correlations of the data as well as on the sequence of insertions (figure 11(d)). Experimental results show that its storage utilization is about 60%. Its performance for range queries is in average proportional to the height of the tree but in the worst case the tree grows linearly with the number of data records.

All access methods discussed so far partition the entire data space, no matter if not all of it is occupied by data. Contrary to that, buddy-tree partition does not deal with empty space (see figure 11(g)). Buddy-tree also offers high dynamic behaviour and performance independent of data distribution.

A totally different method which is neither directory-based nor tree-based is the PLOP-hashing [SK88]. This is a multidimensional dynamic hashing scheme without directory. Instead, it uses an order preserving address function to map key values to storage buckets. Actually, PLOP-hashing can be viewed as a grid file without directory and it is suitable for non-uniform data distributions. It assumes rectangular, complete and disjoint partition of the data space.

5.2.3 Multidimensional Spatial Access Methods

The R[+|*] Trees

The **R** family trees are very popular access methods for rectangular objects. The most known variants of R family are the Guttman's R-tree [Gut84], R+-tree [SRF87], Greene's R-tree [Gre89] and R*-tree [BKSS90]. R- and R*-trees are derived from B+-trees with the technique

Figure 11: Points Access Methods

of overlapping regions. Instead, R+-trees are derived from k-d-B-trees with clipping.

Seeger and Kriegel [SK88] showed that spatial access methods (SAMs) are combinations of a point access method (PAM) and one of the techniques of clipping, overlapping and transformation. Clipping is related to the partition of a rectangle into a set of disjoint rectangles such that each of them is uniquely covered by the page regions. The overlapping region scheme allows directory regions that correspond to different data buckets to overlap; enclosed rectangles are stored to exactly one data bucket; therefore, sometimes the point query has to explore more than one alternative branches of the tree. Transformation schemes represent MBRs of multidimensional objects by points of higher dimensionality.

R-trees store multidimensional rectangles. Non-leaf nodes contain entries of the form (cp, Rectangle), where cp is the address of the child node and Rectangle its MBR. Leaf nodes contain entries of the form (oid, Rectangle) where oid refers to the data record in the database and Rectangle its enclosing rectangle. If M is the maximum number of entries for each node and m is a parameter such that $2 \leq m \leq M/2$, then R-trees require m to M entries for each node, unless it is the root. Tree nodes are stored in separate disk pages. The main issue in R-trees is to dynamically build up bounding rectangles for a set of arbitrary rectangles such that insertion and retrieval operations are efficient and dynamic. When a leaf page fills, it splits. If this split causes an overflow to the parent page, then the split is propagated upwards. The splitting algorithm is crucial for the performance of the R-tree.

Figure 12(a) shows the R-tree organization of our data space. Roussopoulos and Leifker [RL85] suggested an improvement to the performance of the R-tree, minimizing the coverage and overlap by applying a packing algorithm. The coverage at a level l of an R-tree is a measure of total area covered by rectangles at that level. Overlap is the total area covered by more than one rectangle. Later, Sellis et al. [SRF87] came up with R+-tree which avoids overlap and still cares about minimal coverage. An R+-tree organization of our example is shown on figure 12(b). R+-tree, when compared to R-tree, has improved performance, especially for point queries. However, it pays for storage utilization and the insertion and deletion algorithms. The R+-tree is possible that it will not work for very dense distribution of data, because of the excessive clipping [Gre89]. On the other hand, the R-tree suffers when data objects have big MBRs.

R*-tree seems to be the best performing variant of the R-trees family. The space partitioning algorithm in R*-tree considers minimization criteria such as minimal coverage and overlap, minimal perimeter of the directory rectangles and high storage utilization. An improvement to the performance in R*-tree is achieved by the technique of the forced reinsert. The forced reinsert reorganizes the directory of rectangles by deleting half of its data and insert them again. Especially, the deleted rectangles are those whose centers are the most distant from the centers of their bounding box. This results in more quadratic directory rectangles, decreased overlap, less splits and improved localities. Performance experiments prove that R*-tree outperforms all other R-tree in all queries [BKSS90]; in these experiments there was not any comparison with the R+-tree.

The only analytic performance results in the area are by Faloutsos et al. [FSR87] for the limited cases of R- and R+-trees.

Figure 12: (a) R-tree, (b) R+-tree

Polyhedral Access Methods

The bounding rectangles approach appears to be inadequate in many situations [Jag90b], [Gue89]. Some motivating examples are the multidimensional search where some of the keys are not independent, the concave shapes in geographic applications, curved lines, etc. In such cases, better results are obtained if we approximate the objects with more tight polyhedral bounds. The first proposed method is the BSP-tree which is a variation of kd-tree with its partition to be non-orthogonal. Jagadish [Jag90b] proposed a mapping from the k-dimensional attribute space to an m-dimensional orientation space. Then, an R-tree type indexing is possible for hyper-rectangles of the orientation space which corresponds to convex polyhedra of the attribute space. He calls this access method P-tree (polyhedral-tree). Guenther [Gue88] defined the cell-tree. Cells are simple convex polyhedra which are represented by the means of the h-vector (see section 3.3.1). The cell-tree, as the R-tree, is a height-balanced tree, fully dynamic with the enclosing bounding k-dimensional polyhedra to be disjoint.

5.3 Spatial Query Languages

The goal of a spatial database language is to allow the easy formulation of queries that involve both spatial and non-spatial predicates without loss of spatial semantics. A desired requirement is that the queries should be optimizable.

Existing spatial query languages are roughly divided in (a) extensions of the relational languages, (b) object-oriented languages and (c) pictorial languages.

Kemper and Wallrath [KW87] in their analysis of database geometric modeling demonstrate several extensions of QUEL and SQL. Chang and Fu [CF81] developed the QPE (query by pictorial example) language as an extension of the QBE (query by example) and the predicate calculus; in the same paper, they presented an overview of the state-of-the-art pictorial query languages as of 1980. PSQL [RFS88], [RL85] and GEOQL [OSDM89] are more recent proposals based on SQL. [LM88] used Peano tuple algebra, an extension of relational algebra, and computational geometry for processing an important subclass of spatial queries. Guting also [Gut88] extended relational algebra in a many-sorted relational algebra with geometric types and geometric operators.

As for object-oriented languages, object logic [AOG+88], deductive inferences over an object-oriented framework [MK88] and GeoTelos [MMT89] are some examples.

Pictorial query languages emphasize in direct manipulations of pictorial information integrated with ordinary querying facilities. Some known pictorial languages are the PSQL and the PICQUERY [JC88].

5.4 Query Optimization

Optimization problems in spatial databases are significantly different than business oriented-databases. PROBE [OM88], [Ore89] and GEOQL [OSDM89], [OSD89] are the only known systems which showed progress in this direction. In particular, PROBE offers a way of extending the query operations and the optimizer (see section 5.1). In GEOQL, Ooi et al. define an optimizer which separates the spatial and the non-spatial predicates of the query which are handled by different processors and a high level optimization procedure selects the query execution plan with the lowest cost. So far, the only source for optimization of a spatial subquery is the use of a spatial index.

5.5 Summary

This section presented an overview of the research in spatial databases. Work in this area is broadly divided into spatial data modeling, data structures and access methods, and query processing and optimization. The object orientation and the knowledge representation are suitable modeling approaches for spatial databases because they can support the structural aspects and a wide range of spatial operations in spatial representation schemes. The spatial data structures and the access methods are well studied fields. There is virtually no work in query processing and optimization at the query language level although there is a huge amount of work at the data access level. The motive for this is that the corresponding problems have not yet received satisfactory answers even for object oriented databases or knowledge bases. Lastly, the issue that have not received enough attention is the treatment of uncertainty and partial information in spatial databases; this among other is considered as a topic for future research.

6 Future Research

We have reviewed research in the fields of spatial representations, spatial reasoning and spatial databases. We also motivated the importance of spatial knowledge management by presenting several applications which demand support for spatial data. One first observation is that efforts in spatial representation and reasoning are driven by the requirements of specific applications. For instance, in robotics the emphasis is placed in the representation of the free space and the path finding problem. In qualitative mechanics, the representation and reasoning is mainly based on tangency relations. Geographic applications require relative position computations and representation of extremely arbitrary shapes. As a result, there is not any system that provides adequate functionality for a wide range of spatial applications. The second observation is that many applications have to cope with large amounts of spatial data such as CAD/CAM and geographic information systems. For this purpose, the functionality of a database system is necessary. Besides, the extension of databases with some form of intelligence is acknowledged in both of the above areas. In this study, we tried to collect all the different requirements and the drawbacks of existing solutions, in order to form a proposal for further research in this direction. In the sequel, some topics for further research are presented. The first one is the proposal of a general framework for spatial knowledge management. The rest are a number of specific open problems that are identified throughout this study.

1. **Spatial Knowledge Base Management Systems.**

   A Spatial Knowledge Base Management Systems (**SKBMS**) is supposed to be a system which provides spatial data support for a wide range of applications (first observation), and make efficient use of large amounts of spatial data (second observation). A SKBMS consists of three components: a spatial representation component, a reasoning component and a data management component. The specific research questions that are targeted to any of this components are listed below:

   (a) **Representation.** Spatial representations, up to now, deal with regularized object shapes. More emphasis must be put to arbitrary shapes. Relative position is a distinguished spatial relationship which is more naturally expressed and easily computed if qualitative and quantitative (topological and metrical) terms are combined. Formal characterization of spatial relationships and spatial implication is also desirable.

   (b) **Reasoning.** The requirements for spatial reasoning were set in section 4. The two most important of them are the treatment of spatial uncertainty and the independence of application domain. Consequently, the envisioned reasoning component can be thought of as a general purpose reasoning kernel, where application specific operators can be implemented on top.

   Spatial reasoning inherits many of the difficulties of temporal reasoning. The basic question is if we can do any better in space taking into account properties of the spatial ontologies. Hierarchies, i.e. structure, offer one possibility. The static world view of space (contrary to time) is another possibility. What else?

(c) **Management.** The database component seems to be the heart of a large spatial knowledge base management system. It must cooperate with a reasoner and must be capable of storing different types of spatial information such as metric, approximate and incomplete. The selection of the data models primitives and operations is a critical issue. Of course, object orientation and knowledge representation are the predominant technologies. Telos, as a structural knowledge representation language, is seriously considered as a candidate. Within the Telos framework, we can extend the methods of knowledge base query processing, at least those few, towards spatial knowledge bases.

2. **Incomplete Spatial Databases.**

   One of the most important concerns in this study is the representation of incomplete spatial information (see sections 3, 4). In fact, we distinguished two types of incompleteness (a) imprecise measurements and (b) lack of knowledge. The second type of incompleteness is usually handled by the high level reasoning procedures by the means of default rules, completion axioms and other methods known in Artificial Intelligence. The approximate and imprecise data can be handled in the database level. The existing access methods presuppose complete metric information. Some unexplored possibilities are (a) to store large qualitative networks in the database providing the approximate access methods and (b) to extend existing access methods to work with range data rather than exact values, for instance, instead of points store hyper-rectangles and instead of rectangles rectangular frames.

   Another parameter that need further study is the granularity. Objects can satisfy some relations in one scale and not satisfy the same relations in another scale. Therefore, inconsistencies found at the finer level may be propagated at the coarser levels, or may suggest changes in the representation scheme and the data structures. There should be a lot of trade offs the study of which is a topic for further research. A formal account of varying granularity by Hobbs [Hob85] is one of the few works in the Artificial Intelligence bibliography. Scaleless maps [BSW91] is a very interesting proposal in databases. A scaleless map represents objects in the largest available scale. The scale information is taken into account in the access method. Objects are selected by the access method and then they are scaled down/up to produce the target answer.

3. **Spatial Query Optimization.**

   Query optimization in spatial databases is a quite open field for further research. As we remarked in section 5.4, most of the existing prototypes optimize with respect to a single spatial index. Therefore, query processing techniques which explore more than one index can be explored and consequently the study of the execution cost of a sequence of spatial indices becomes a valid question.

4. **Spatial Access Methods.**

   Although a lot of work have been done in spatial access methods there are still many questionable issues [GB90]. Most of the existing access methods work well only for regularized data (e.g., rectangles) and under very restricted assumptions about their distribution in space, overlapping, etc. Since it happens that "well behaved" data are very rare in nature, we need more robust access methods which perform well under varying conditions. An analytic performance characterization of them is also desired. So far, most of the access methods, with only few exceptions, are justified only by experimental or empirical results.

References

[ABJN85] D. Ayala, P. Brunet, R. Juan, and I. Navazo. Object Representation by Means of Nonminimal Division Quadtrees and Octrees. *ACM Transactions on Graphics*, 4(1):41–59, 1985.

[All83] J.F. Allen. Maintaining Knowledge about Temporal Intervals. *Communications of the ACM*, 26(11):832–843, November 1983.

[AOG+88] L. Anderson, H. Ohkawa, J. Gjovaag, D. Maier, and S. Shulman. Representing CSG Solids Using a Logic-Based Object Data Model. Technical Report CS/E 88-039, Oregon Graduate Center, 1988.

[AS86] D. Abel and J. Smith. A Relational GIS Database Accomodating Independent Partitionings of the Region. In *Proceedings of the 2nd International Symposium on Spatial Data Handling*, pages 213–224, 1986.

[BF79] J. L. Bentley and J.H. Friedman. Data Structures for Range Searching. *Computing Surveys*, 11(4):397–409, 1979.

[BKSS90] N. Beckman, H.-P. Kriegel, R. Schneider, and B. Seeger. The R* tree: An Efficient and Robust Access Method for Points and Rectangles. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 323–331, 1990.

[Bro82] R.A. Brooks. Solving the find-Edge problem by Good Representations of Free Space. In *Proceedings of AAAI-82*, pages 381–387, 1982.

[BSW91] B. Becker, H.-W. Six, and P. Widmayer. Spatial Priority Search: An Access Technique for Scaleless Maps. In *Proceedings of ACM SIGMOD International Conference on Management of Data*, pages 128–137, 1991.

[CDF+82] A. Chan, S. Danberg, S. Fox, W.-T. Lin, A. Nori, and D. Ries. Storage and Access Structures to Support a Semantic Data Model. In *Proceedings of the 8th VLDB Conference*, pages 308–316, 1982.

[CF81] N.S. Chang and K.S. Fu. Picture Query Languages for Pictorial Database Systems. *Computer*, 14(11):23–33, 1981.

[CK81] S.-K. Chang and T. Kunii. Pictorial Database Systems. *Computer*, 14(11):13–21, 1981.

[CL88] S.-K. Chang and Y. Li. Representation of Multi-Resolution Symbolic and Binary Pictures Using 2DH Strings. In *Proceedings of IEEE Workshop on Language for Automation*, pages 190–195, 1988.

[CSY87] S.-K. Chang, Q.-Y. Shi, and C.-W. Yan. Iconic Indexing by 2-D Strings. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, PAMI-9(3):413–427, 1987.

[Dav85] Ernest Davis. The MERCATOR Representation of Spatial Knowledge. In *Proceedings of IJCAI-85*, pages 295–301, 1985.

[Dav86] Ernest Davis. *Representing and Acquiring Geographic Knowledge*. Pitman and Morgan Kaufmann Publishers Inc., 1986.

[Dav87] Ernest Davis. Constraint Propagation with Interval Labels. *Artificial Intelligence*, 32:281–331, 1987.

[Dav89] Ernest Davis. A Framework for Qualitative Reasoning About Solid Objects. In D. Weld and J. DeKleer, editors, *Readings in Qualitative Reasoning about Physical Systems*, pages 603–609, 1989.

[Dea89] T. Dean. Using Temporal Hierarchies to Efficiently Maintain Large Temporal Databases. *Journal of ACM*, 36(4):687–718, 1989.

[DKA+86] P. Dadam, K. Kuspert, F. Anderson, H. Blanken, R. Erbe, J. Gunauer, V. Lum, P. Pistor, and Walch. G. A DBMS prototype to support extended NF2-relations: An integrated view on flat tables and hierarchies. In *Proceedings of ACM SIGMOD International Conference on Management of Data*, pages 376–387, 1986.

[DM87] T. Dean and D.V. McDermott. Temporal Data Base Management. *Artificial Intelligence*, 32:1–55, 1987.

[DMP89] R. Dechter, I. Meiri, and J. Pearl. Temporal Constraint Networks. In *Proceedings of 1st Knowledge Representation and Reasoning Conference*, pages 83–93, 1989.

[Dut89] Soumitra Dutta. Qualitative Spatial Reasoning: A semi-quantitative Approach Using Fuzzy Logic. In *Proceedings of the First Symposium on the Design and the Implementation of Large Spatial Databases (SSD'89)*, pages 345–364. Lecture Notes in Computer Science, Springer Verlag, 1989.

[Euc] *Ευκλειδης. Στοιχεια Γεωμετριας, 300 B.C.* original appears in "Euclid in Greek" by T. Heath, Cambridge University Press, 1920.

[EL82] R.J. Elliot and M.E Lesk. Route Finding in Street Maps by Computers and People. In *Proceedings of 2nd National Conference on Artificial Intelligence*, pages 258–261, Los Altos, CA, 1982.

[Fal87] Boi Faltings. Qualitative Kinematics in Mechanisms. In *Proceedings of IJCAI-87*, pages 436–442, 1987.

[Fal88] Christos Faloutos. Gray Codes for Partial Match and Range Queries. *IEEE Trans. on Software Engineering*, 14(10):1381–1393, 1988.

[FNF87] K. Forbus, P. Nielsen, and B. Faltings. Qualitative Kinematics: A Framework. In *Proceedings of IJCAI-87*, pages 430–435, 1987.

[For80] K.D. Forbus. Spatial and Qualitative Aspects of Reasoning. In *Proceedings of AAAI-80*, 1980.

[FR89a] Christos Faloutos and Yi Rong. Spatial Access Methods Using Fractals: Algorithms and Performance E valuation. Technical Report UMIACS-TR-89-31, CS-TR-2214, University of Maryland, 1989.

[FR89b] Christos Faloutos and Shari Roseman. Fractals for Secondary Key Retrieval. Technical Report UMIACS-TR-89-47, CS-TR-2242, University of Maryland, 1989.

[Fra91] Andrew Frank. Qualitative Spatial Reasoning with Cardinal Directions. submitted for publication, 1991.

[Fre87] M. Freeston. The BANG file: A New Kind of Grid File. In U. Dayal and I. Traiger, editors, *Proceedings of 1987 ACM-SIGMOD Inetrnational Conference on Management of Data*, San Fransisco, USA, 1987.

[FSR87] Christos Faloutsos, Timos Sellis, and Nick Roussopoulos. Analysis of Object Oriented Spatial Access Methods. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 426–439, 1987.

[GA89] M. Ghallab and M.A. Alaoui. Managing Efficiently Temporal Relations through Indexed Spanning Trees. In *Proceedings of IJCAI-89*, pages 1297–1303, 1989.

[Gar82] Irene Gargantini. An Effective Way to Represent Quadtrees. *Communications of ACM*, 25(12):905–910, 1982.

[GB90] O. Guenther and A. Buchmann. Research Issues in Spatial Databases. *SIGMOD Record*, 19(4):61–68, 1990.

[Gre89] Diane Greene. An Implementation and Performance Analysis of Spatial Data Access Methods. In *Proceedings of the International Conference on Data Engineering*, pages 606–615, 1989.

[GS82] A. Guttman and M. Stonebraker. Using a Relational Database Management System for Computer Aided Design Data. Technical Report UCB/ERL M82/37, Electronics Research Laboratory, University of California, Berkeley, 1982.

[Gue88] Oliver Guenther. *Efficient Structures for Geometric Data Management.* Lecture Notes in Computer Science, 337. Springer Verlag, 1988.

[Gue89] Oliver Guenther. The Design of the Cell Tree: An Object-Oriented Index Structure for Geometric Databases. In *Proceedings of International Conference on Data Engineering*, pages 598–605, 1989.

[Gue91] H.W. Guesgen. Feasible Reasoning about Space. submitted for publicaation, 1991.

[Gut84] Antonin Guttman. R-Trees: A Dynamic Index Structure For Spatial Searching. In *Proceedings of ACM SIGMOD International Conference on Management of Data*, pages 47–57, 1984.

[Gut88] R.-H. Guting. Geo-Relational Algebra: A Model and Query Language for Geometric Database Systems. In *Proceedings of International Conference on Extending Database Technology - EDBT'88*, pages 506–527, 1988.

[Heg59] G.W.F. Hegel. *Encyclopedia of Philosophy, (translated and annotated by G.E. Mueller).* Philosophical Library, New York, 1959.

[Hob85] J. Hobbs. Granularity. In *Proceedings of IJCAI-85*, pages 432–435, 1985.

[HSW88] A. Hutflesz, H.-W. Six, and P. Widmayer. Twin Grid Files: Space Optimizing Access Schemes. In *Proceedings of ACM SIGMOD International Conference on Management of Data*, pages 183–190, 1988.

[IJRW85] S.S. Iyengar, C.C. Jorgensen, S.V.N. Rao, and C.R. Weisbin. Learned Navigation Paths for a Robot in Unexplored Terrain. In *IEEE 2nd Conference on Artificial Intelligence Applications*, pages 148–155, 1985.

[Jag90a] H.V. Jagadish. Linear Clustering of Objects with Multiple Attributes. In *Proceedings of ACM SIGMOD International Conference on Management of Data*, pages 332–342, 1990.

[Jag90b] H.V. Jagadish. Spatial Search with Polyhedra. In *Proceedings of International Conference on Data Engineering*, pages 311–319, 1990.

[JC88] T. Joseph and A. Cardenas. PICQUERY: A High Level Query Language for Pictorial Database Management. *IEEE Transactions on Software Engineering*, 14(5):630–638, 1988.

[Jun88] E. Jungert. Extended Symbolic Projection Used in Knowledge Structure for Spatial Reasoning. In *Proceedings of 4th BPRA Conference on Pattern Recognition*, 1988.

[Kau91] Srephen Kaufman. A Formal Theory of Spatial Reasoning. In *Proceedings of AAAI Spring Symposium on Logical Foundations of Commonsense Reasoning*, pages 92–101, 1991.

[KB87] B.J. Kuipers and Y.-T. Byun. A Qualitative Approach to Robot Exploration and Map-Learning. In *Proceedings of Spatial Reasoning and Multi-Sensor Fusion Workshop*, pages 390–404, 1987.

[KB88] B.J. Kuipers and Y.-T. Byun. A Robust Qualitative Method for Robot Spatial Learning. In *Proceedings of AAAI-88*, pages 774–779, 1988.

[KL88] B.J. Kuipers and S.T. Levitt. Navigation and Mapping in Large-Scale Space. *AI Magazine*, pages 25–43, Summer 1988.

[Koo89] Johannes Koomen. Localizing Temporal Constraint Propagation. In R. Brachman, H. Levesque, and R. Reiter, editors, *Proceedings of 1st International Conference on Principles of Knowledge Representation and Reasoning*, pages 198–202, Toronto, Ontario, 1989.

[KS90] Curtis Kolovson and Michael Stonebraker. S-Trees: Database Indexing Techniques for Multi-Dimensional Interval Data. Technical Report UCB/ERL M90/35, Electronics Research Laboratory, University of California, Berkeley, 1990.

[KSSS89] H.-P. Kriegel, M. Schiwietz, R. Schneider, and B. Seeger. Performance Comparison of Point Spatial Access Methods. In *First International Symposium on Design and Implementation of Large Spatial Databases (SSD'89)*, pages 89–114. Lecture Notes in Computer Science, Springer Verlag, 1989.

[KT91] M. Koubarakis and T. Topaloglou. Physical Design for Telos Knowledge Bases. preliminary report, 1991.

[Kui78] B.J. Kuipers. Modelling Spatial Knowledge. *Cognitive Science*, 2:129–153, 1978.

[Kui89] B.J. Kuipers. Qualitative Reasoning: Modelling, Simulation with Incomplete Knowledge. *Automatica*, 1989.

[KW87] A. Kemper and M. Wallrath. An Analysis of Geometric Modeling in Database Systems. *ACM Computing Surveys*, 19(1):47–91, 1987.

[Lad89] Peter Ladkin. Metric Constraint Satisfaction with Intervals. Technical Report TR-89-038, International Computer Science Center, Berkeley, 1989.

[Lau83] Jean-Paul Laumond. Model Structuring and Concept Recognition: Two Aspects of Learning for a Mobile Robot. In *Proceedings of IJCAI-83*, pages 839–841, 1983.

[Lau85] R. Laurini. Graphics Databases Built on Peano Space-Filling Curves. In *Proceedings of EUROGRAPHICS'85 Conference*, pages 327–338, 1985.

[LG89] D. Lenat and R. Guha. *Building Large Knowledge-Based Systems.* Addison-Wesley, 1989.

[Lig90] G. Ligozat. Weak Representations of Interval Algebras. In *Proceedings of AAAI-90*, pages 715–710, 1990.

[LJ88] Nikos A. Lorentzos and Roger G. Johnson. Extending Relational Algebra To Manipulate Temporal Data. *Information Systems*, 13(3):289–296, 1988.

[LLCN87] T. Levitt, D. Lawton, D. Chelberg, and P. Nelson. Qualitative Navigation. In *Proceedings of DARPA Image Understanding Workshop*, pages 447–465, 1987.

[LM87] R. Laurini and F. Milleret. Peano Relations in CAD/CAM Databases. In *Proceedings of IEEE Int. Conference on Data and Knowledge Systems for Manufacturing and Engineering*, pages 115–123, 1987.

[LM88] R. Laurini and F. Milleret. Relational Algebra versus Computational Geometry. In Rafanelli, Klensin, and Svensson, editors, *Proceedings of the 4th Int. Conference on Statistical and Scientific Database Management*, pages 291–313, 1988.

[LM90] R. Laurini and F. Milleret. Towards Intelligent GIS. In N. Polydorides, editor, *Proceedings of 3rd Workshop on Urban and Regional Spatial Analysis*, pages 127–156, Patras, Greece, 1990.

[LP81] Tomas Lozano-Perez. Automatic Planning of Manipulator Transfer Movements. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-11(10):681–698, 1981.

[LP83] R. Lorie and W. Plouffe. Complex Objects and Their Use in Design Transaction. In *Proceedings of ACM-SIGMOD Conference on Engineering Design Application*, pages 115–121, 1983.

[LS90] D. Lomet and B. Salzberg. The Performance of a Multiversion Access Method. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 353–363, 1990.

[Mac77] A.K. Mackworth. Consistency in Networks of Relations. *Artificial Intelligence*, 8(1):99–118, 1977.

[MB83] J. Malik and T.O. Binford. Reasoning in Time and Space. In *Proceedings of IJCAI-83*, pages 343–345, 1983.

[MBJK90] J. Mylopoulos, A. Borgida, M. Jarke, and M. Koubarakis. Telos: A Language for Representing Knowledge About Information Systems. *ACM Transactions on Office Information Systems*, to appear 1990.

[McD80] Drew McDermott. A Theory of Metric Spatial Inference. In *Proceedings of AAAI-80*, pages 246–248, 1980.

[McD82] D.V. McDermott. A Temporal Logic for Reasoning About Processes and Plans. *Cognitive Science*, 6:101–155, 1982.

[MD84] Drew McDermott and Ernest Davis. Planning Routes through Uncertain Territory. *Artificial Intelligence*, 22:107–156, 1984.

[MF85] A.K. Mackworth and E.C. Freuder. The Complexity of Some Polynomial Network Consistency Algorithms for Constraint Satisfaction Problems. *Artificial Intelligence*, 25(1):65–74, 1985.

[Mil85] David Miller. A Spatial Representation System for Mobile Robots. In *IEEE International Conference on Robotics and Automation*, pages 122–127, 1985.

[MJ90] A. Makerjee and G. Joe. A Qualitative Model for Space. In *Proceedings of AAAI-90*, pages 721–727, 1990.

[MK88] Mohan and Kashyap. An Object-Oriented Knowledge Representation for Spatial Information. *IEEE Transactions on Software Engineering*, 14(5):675–681, 1988.

[MMT89] E. Milios, J. Mylopoulos, and J. Tsotsos. Towards Computational Geoscience: Issues of Knowledge Representation. technical report, 1989.

[MO86] Frank Manola and Jack Orenstein. Towards a General Spatial Data Model for an Object-Oriented DBMS. In *Proceedings of the 12th VLDB Conference*, pages 328–335, 1986.

[Mon74] U. Montanari. Networks of Constraints: Fundamental Properties and Applications to Picture Processing. *Information Sciences*, 7:95–132, 1974.

[MS88] S. Miller and L. Schubert. Using Specialists to Accelerate General Reasoning. In *Proc. of AAAI-88*, pages 161–165, 1988.

[NC91] N.H. Narayanan and B Chandraseckaran. Reasoning Visually about Spatial Interactions. In *Proceedings of IJCAI-91*, 1991.

[NHS84] J. Nievergelt, H. Hinterberger, and K. Sevcik. The Grid File: An Adaptable, Symmetric Multikey File Structure. *ACM Transactions on Database Systems*, 9(1):38–71, 1984.

[Nie89] Jurg Nievergelt. 7 ± 2 Criteria for Accessing and Comparing Spatial Data Structures. In *Proceedings of the First Symposium on the Design and the Implementation of Large Spatial Databases (SSD'89)*, pages 3–27. Lecture Notes in Computer Science, Springer Verlag, 1989.

[OM84] J.A. Orenstein and T.H. Merrett. A Class of Data Structures for Associative Searching. In *Proceedings of SIGMOD-84*, pages 181–190, 1984.

[OM88] J. Orenstein and F. Manola. PROBE: Spatial Data Modelling and Query Processing in an Image Database Application. *IEEE Transactions on Software Engineering*, 14(5):611–629, 1988.

[OO85] E. Ozkaharan and M. Ouksel. Dynamic and Order Preserving Data Partitioning for Database Machines. In *Proceedings of VLDB-85*, pages 358–368, 1985.

[Ore82] Jack Orenstein. Multidimensional Tries Used for Associative Searching. *Information Processing Letters*, 14(4):150–157, 1982.

[Ore86] Jack Orenstein. Spatial Query Processing in an Object-Oriented Database System. In *Proceedings of SIGMOD-86*, pages 326–333, 1986.

[Ore89] Jack Orenstein. Strategies for Optimizing the Use of Redundancy in Spatial Databases. In *Proceedings of the First Symposium on the Design and the Implementation of Large Spatial Databases (SSD'89)*, pages 115–134. Lecture Notes in Computer Science, Springer Verlag, 1989.

[OS90] Y. Ohsawa and M. Sakauchi. A New Tree Type Data Structure with Homogeneous Nodes Suitable for a Very Large Spatial Database. In *Proceedings of International Conference on Data Engineering*, pages 296–303, 1990.

[OSD89] B.C. Ooi and R. Sacks-Davis. Query Optimization in an Extended DBMS. In *Proceedings of 3rd International Conference on Foundations of Data Organizations and Algorithms*, pages 48–63, 1989.

[OSDM89] B.C. Ooi, R. Sacks-Davis, and K.J. McDonell. Extending a DBMS for Geographic Applicaations. In *Proceedings of International Conference on Data Engineering*, pages 590–597, 1989.

[PSS+87] H.-B. Paul, H.-J. Schek, M.H. Scholl, G. Weikum, and U. Deppish. Architecture and Implementation of a Darmstadt Database Kernel System. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 196–207, San Francisco, CA, 1987.

[Req80] Aristides Requicha. Representations of Rigit Solids: Theory, Methods and Systems. *ACM Computing Surveys*, 12(4):337–365, 1980.

[RFS88] Nick Roussopoulos, Christos Faloutsos, and Timos Sellis. An Efficient Pictorial Database System for PSQL. *IEEE Transactions on Software Engineering*, 14(5):639–650, 1988.

[RL85] Nick Roussopoulos and Daniel Leifker. Direct Spatial Search on Pictorial Databases Using Packed R-trees. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 17–31, 1985.

[Rob81] John Robinson. The K-D-B-Tree: A Search Structure for Large Multidimensional Dynamic Indexes. In *Proceedings of ACM SIGMOD International Conference on Management of Data*, pages 10–18, 1981.

[Rob89] Vince Robinson. Specification of Natural Language Concepts for Spatial Information Retrieval. Technical Report NFS IST-8412406, 1989.

[Rom90] G.-C. Roman. Formal Specification of Geographic Data Processing Requirements. *IEEE Transactions on Knowledge and Data Engineering*, 2(4):370–380, 1990.

[RV82] A. Requicha and H. Voelcker. Solid Modeling: A Historical Summary and Contemporary Assessment. *IEEE Computer Graphics and Applications*, pages 9–24, March 1982.

[RV83] A. Requicha and H. Voelcker. Solid Modeling: Current Status and Research Directions. *IEEE Computer Graphics and Applications*, pages 25–37, October 1983.

[Sam84] Hanan Samet. The Quadtree and Related Hierarchical Data Structures. *ACM Computing Surveys*, 16(2):187–260, 1984.

[Sam89] Hanan Samet. *The Design and Analysis of Spatial Data Structures: Computer Graphics, Image Processing and GIS*. Addison-Wesley, 1989.

[Sch88] Albrecht Schmiedel. Temporal Constraint Networks. Technical report, Technische Universitat Berlin, KIT Group, 1988.

[See90] Bernhard Seeger. Multidimensional Access Methods and their Applications. tutorial, 1990.

[SH88] D. Spooner and M. Hardwick. A Conceptual Framework for Data Management in Mechanical CAD. In M.J. Wozny, H.W. McLaughlin, and J.L. Encarnacao, editors, *Geometric Modeling for CAD Applications*, pages 317–329. North-Holland, 1988.

[Sho85] Yoan Shoham. Naive Kinematics: one aspect of shape. In *Proceedings of IJCAI-85*, pages 436–422, 1985.

[SK88] B. Seeger and H.-P. Kriegel. Techniques for Design and Implementation of Efficient Spatial Access Methods. In *Proceedings of the 14th VLDB Conference*, pages 360–371, 1988.

[SK90] B. Seeger and H.-P. Kriegel. An Efficient and Robust Access Method for Spatial Database Systems. In *Proceedings of the 16th VLDB Conference*, pages 590–601, 1990.

[SRF87] Timos Sellis, Nick Roussopoulos, and Christos Faloutsos. The R^+-Tree: A Dynamic Index for Multidimensional Objects. In *Proceedings of the 13th VLDB Conference*, pages 507–518, 1987.

[SRG83] M. Stonebraker, B. Rubenstein, and A. Guttman. Application of Abstract Data Type and AbstractIndices to CAD Databases. In *Proceedings of ACM-SIGMOD Conference on Engineering Design Application*, pages 107–113, 1983.

[SW86] H.-J. Schek and W. Waterfeld. A Database Kernel System for Geoscientific Aplications. In *Proceedings of the 2nd International Symposium on Spatial Data Handling*, pages 273–288, 1986.

[Tak88] Tapio Takala. The Role of Multidimensional Models in CAD Systems. In M.J. Wozny, H.W. McLaughin, and J.L. Encarnacao, editors, *Geometric Modeling for CAD Applications*, pages 115–121. Elsevier Sience Publishers (North-Holland), 1988.

[Tam82] Marku Tamminen. Efficient Spatial Access to a Database. In *Proceedings of ACM SIGMOD International Conference on Management of Data*, pages 200–206, 1982.

[vB83] J. van Benthem. *The Logic of Time*. D. Reidel Publishing Company, 1983.

[vB89] Peter van Beek. Approximation Algorithms for Temporal Reasoning. In *Proceedings of IJCAI-89*, pages 1291–1296, 1989.

[vB90] Peter van Beek. Exact and Approximate Reasoning About Qualitative Temporal Relations. Technical Report TR 90-29, Department of Computing Science, University of Alberta, August 1990.

[VK86] Marc Vilain and Henry Kautz. Constraint Propagation Algorithms for Temporal Reasoning. In *Proceedings of AAAI-86*, pages 377–382, 1986.

[vOvdB89] P. van Oosterom and J. van den Bos. An Object-Oriented Approach to the Design of Geographic Information Systems. *Computer & Graphics*, 13(4):409–418, 1989.

[VP86] R.E. Valdés-Pérez. Spatio-temporal reasoning and linear inequalities. Technical Report Memo 875, MIT Artificial Intelligence Laboratory, 1986.

[VP87] R.E. Valdés-Pérez. The Satisfiability of Temporal Constraint Networks. In *Proceedings of AAAI-87*, pages 256–260, 1987.

[Wil88a] Peter Wilson. Multiple Representations of Solid Models. In M.J. Wozny, H.W. McLaughin, and J.L. Encarnacao, editors, *Geometric Modeling for CAD Applications*, pages 99–113. Elsevier Sience Publishers (North-Holland), 1988.

[Wil88b] Peter Wilson. Solid Modeling Research and Applications in the U.S.A. In M.J. Wozny, H.W. McLaughin, and J.L. Encarnacao, editors, *Geometric Modeling for CAD Applications*, pages 69–89. Elsevier Sience Publishers (North-Holland), 1988.

[Zan83] Carlo Zaniolo. The Database Language GEM. In *Proceedings of ACM SIGMOD International Conference on Management of Data*, pages 207–218, 1983.