An Abstract Model of Three-Dimensional Spatial Data Types

Markus Schneider*& Brian E Weinrich
University of Florida
Department of Computer & Information Science & Engineering
Gainesville, FL 32611, USA
{mschneid, brianew}@cise.ufl.edu

ABSTRACT

Although spatial objects of our world have an intrinsic three-dimensional (3D) nature, *3D data modeling* and *3D data management* have so far been neglected in spatial database systems and Geographical Information Systems, which map geometric data mainly to two-dimensional abstractions. But increasingly the third dimension becomes more and more relevant for application domains like pollution control, water supply, soil engineering, urban planning, and aviation. Large volumes of 3D data require a treatment in a database context for representing, querying, and manipulating them efficiently. This paper aims at investigating the complex inherent features of 3D data and presents an abstract, formal data model called *SPAL3D* (*Spatial Algebra 3D*). The data model comprises a set of *three-dimensional spatial data types* together with a collection of geometric set operations.

Categories and Subject Descriptors: H.2.8 Database Applications — Spatial Databases and GIS.

General Terms: Design.

Keywords: Data model, algebra, 3D spatial data type, spatial database, GIS.

1. INTRODUCTION

From a *data modeling* and *data management* point of view, most spatial database systems and Geographical Information Systems (GIS) are nowadays restricted to the handling of two-dimensional (2D) information. However, the third dimension plays a more and more important role in many application domains like GIS, pollution control, water supply and distribution, soil engineering, mining, urban planning and development, archaeology, aviation, to name only a few. These application areas require not only 3D visualization and spatial analysis methods, but they also need database support for storing, retrieving, querying, and manipulating the underlying 3D data. Consequently, apart from 2D (planar) and 2.5D (relief, terrain) data, spatial database systems and GIS should additionally incorporate the treatment of 3D data throughout their architecture.

In order to understand the nature and properties of 3D data and to abstract from implementation details, in a first step, the overall goal is to design a comprehensive abstract, formal data model for 3D data, called *SPAL3D* (*Spatial Algebra 3D*). This paper focuses on the rigorous definition of a set of *three-dimensional spatial data types* (like *line3D* or *volume*) together with a collection of geometric set operations. In a later step, we intend the abstract algebra to serve as a specification for its later implementation.

Section 2 presents related work. At an abstract level, Section 3 introduces the type system SPAL3D and formally defines the proposed 3D data types. Section 4 focuses on the definition of 3D geometric set operations. Finally, Section 5 draws some conclusions.

2. RELATED WORK

3D data modeling has had a long tradition in disciplines like Computer Aided Design, Computer Graphics, image processing, and robotics. In a spatial database and GIS context however, due to the complexity of 3D data, only 2.5D representations have been designed like Digital Terrain/Elevation Models (DTMs/DEMs) and Triangulated Irregular Networks (TINs). These approaches augment the 2D representation of a spatial object (like the boundary representation of a region) with a z-coordinate. In GIS, most work on 3D support has especially been carried out on visualization and spatial analysis. A simple transfer of 3D concepts from other disciplines like those mentioned above is impeded since most of their data structures are inappropriate for GIS purposes (e.g., the constructive approach in Computer Aided Design) and/or are main memory structures (e.g., in Computer Graphics) whereas spatial databases and GIS require external data and index structures.

Literature on 3D data modeling and management in a spatial database and GIS context is rare. The models in [2, 3, 10] provide several kinds of 3D data types, e.g., for 3D points, lines, surfaces, and volumes, but in a very informal manner. The topological model for 3D data in [6] rests on cell complexes. The work in [9] introduces 3D equivalents to the basic 2D data types point, line, and polygon. Additionally, a type is included for polyhedra. In [12] the so-called 3D Formal Data Structure introduces separate relations for

*This work was partially supported by the National Science Foundation under grant number NSF-CAREER-IIS-0347574.

GIS'04, November 12–13, 2004, Washington, DC, USA.

surfaces, bodies, lines, and points. But scattering complex information about several relations has enormous expressiveness and efficiency problems. Topological relationships between 3D objects have been investigated in [6, 9, 11]. The software library CGAL [4] comprises a collection of main memory implementations for 3D problems. The *GeoToolKit* [1], which is an object-oriented geo-database kernel system, currently provides the only existing implementation of a collection of 3D spatial data types and operations. The data types are modeled as *simplicial complexes* and include tetrahedra, triangular networks, and tetrahedral networks. But simplicial complexes do not allow the modeling of multi-part objects and prevent the definition of volumes with cavities.

We are not aware of any efforts to formally specify a comprehensive and general set of 3D spatial data types. This paper with its (partial) specification of an algebra for 3D data is meant to be the first step towards this goal.

3. 3D SPATIAL DATA TYPES

The first subsection explains our selection of and design considerations for 3D data types. The remaining subsections focus on a rigorous definition of each single data type.

3.1 Design Considerations

2D points, 2D lines, and 2D regions [7] result from a reduction and generalization process that maps a geometric structure in 3D space into a suitable structure in 2D space. As a starting point for a design of 3D data types, we "reset" these abstraction processes and look backwards to their origins.

3D points are the origin of the reduction and generalization process to 2D points. Besides their positions, 3D points also model their heights. Examples are lighthouses along the U.S. coasts, landmarks within a country, and the position and heights of military bases. For this kind of 3D objects, we will define a spatial data type *point3D* in Section 3.2.

3D lines are the origin of the reduction and generalization process to 2D lines. If we consider a road in the mountains modeled as a 3D curve, this curve has rather different properties than its 2D counterpart. For example, the length computation will lead to different values. For these 3D objects, we will define a spatial data type *line3D* in Section 3.3.

3D regions, surfaces, and volumes are possible origins of the reduction and generalization process to 2D regions. All these different kinds of 3D objects (examples are landscapes, mountains, skylines of cities, and urban buildings) have an extent and can be mapped to 2D regions by simply ignoring the z-coordinates. But there are differences.

For 3D regions this means that each (x, y)-coordinate can be mapped to a single z-coordinate. All (x, y, z)-coordinates together are required to form a (predominantly) smooth and continuous surface. We permit disjoint surface parts and even holes in them. Users usually associate the feature of flatness with 2D regions. As a generalization, 3D regions may also be curved. We will give this data type the name *relief* (but not *region3D*) and define it in Section 3.5.

The data type *surface* defined in Section 3.4 generalizes the data type *relief* in the sense that it describes areal features in 3D space that may have several, different (x, y, z) coordinates with equal values for x and y. This type is especially interesting for representing the boundary, contour, or silhouette of a volume object.

The data type *volume* defined in Section 3.6 represents 3D spatial objects containing or filling an amount of space with a mass (e.g., buildings, bridges, water reservoirs). That is, it is able to model solids with a possibly very high complex structure including multiple parts and cavities.

We aim at a very general and abstract definition of these data types in the Euclidean space $\mathbb{R}^3$. The task is to determine those point sets that are admissible for the complex objects of these types. For each data type we give both an "unstructured" and a "structured" definition. The unstructured definition purely determines the point set of a 3D object. The structured definition gives a unique representation and emphasizes the component view of a 3D object. In general, objects of all 3D spatial data types may include several disjoint components. In addition, reliefs and surfaces may have holes, and volumes may incorporate cavities. The formal framework leverages point set theory and point set topology [5]. We assume that the reader is familiar with their basic concepts. For each 3D object A we specify the topological notions of *boundary* (∂A), *interior* (A°), *exterior* (A^-), and *closure* ($\overline{A}$). All data types satisfy closure properties. That is, if we calculate the geometric intersection, union, or difference between two objects of the same type, the result object is assured to have the same type.

3.2 The Data Type *point3D*

A value of type *point3D* is defined as a finite set of isolated points in 3D space. Unstructured and structured definitions coincide for this type. The spatial data type *point3D* is defined as

$$point3D = \{P \subset \mathbb{R}^3 \mid P \text{ is finite}\}$$

We call a value of this type *complex point*. If $P \in point$ is a singleton set, i.e., $|P| = 1$, P is denoted as a *simple point*. In particular, the empty set, which is the identity of geometric union, is admitted since it can be the result of a geometric operation, e.g., if a point object has nothing in common with another point object in a geometric intersection operation.

The topological notions of boundary, interior, and exterior of a complex point are defined as follows. For a simple point p we specify $\partial p = \varnothing$ and $p^\circ = p$, which is the commonly accepted definition. For a complex point $P = \{p_1, \ldots, p_n\}$ we then obviously obtain $\partial P = \varnothing$, $P^\circ = \bigcup_{i=1}^{n} p_i^\circ$, $P^- = \mathbb{R}^3 - (\partial P \cup P^\circ) = \mathbb{R}^3 - P^\circ$, and $\overline{P} = \partial P \cup P^\circ = P^\circ$.

3.3 The Data Type *line3D*

First, we give an unstructured definition for 3D lines as arbitrary collections of 3D curves. For a function $f : X \to Y$ and a set $A \subseteq X$ we introduce the notation $f(A) = \{f(x) \mid x \in A\}$. Formally, we specify a 3D line as the union of the images of a finite number of continuous mappings from 1D space to 3D space. The spatial data type *line3D* is defined as

$$line3D = \{\textstyle\bigcup_{i=1}^{n} l_i([0,1]) \mid n \in \mathbb{N}, \forall\, 1 \le i \le n : l_i : [0,1] \to \mathbb{R}^3 \text{ is a continuous mapping}\}$$

We call a value of this type *complex line*. Images of different mappings may intersect each other. The same line object can be represented by different collections of mappings. Since the interval $[0, 1]$ is bounded, a *line3D* object is bounded too.

For a structured definition, we separate the point set of a *line3D* object into appropriate components called *curves*. A

curve is considered a *non-self-intersecting 3D line* (and thus a restricted space curve), and the set of curves is defined as:

$$\begin{aligned} \mathit{curve} = \{ & f([0,1]) \mid \\ & f : [0,1] \to \mathbb{R}^3 \text{ is a continuous mapping} \wedge \\ & \forall\, a, b \in]0,1[: a \neq b \Rightarrow f(a) \neq f(b) \wedge \\ & \forall\, a \in \{0,1\}\ \forall\, b \in]0,1[: f(a) \neq f(b)\} \end{aligned}$$

The mappings $f(0)$ and $f(1)$ are called the *end points* of a curve. This definition allows loops ($f(0) = f(1)$) but forbids equality of different interior points and equality of an interior point with an end point.

Let S be the set of all curves over $\mathbb{R}^3$. Two lines $l_1, l_2 \in S$ with their continuous mappings f_1 and f_2 are called *quasi-disjoint* if, and only if, $(\forall\, a, b \in]0,1[: f_1(a) \neq f_2(b)) \wedge \neg((f_1(0) = f_2(0) \wedge f_1(1) = f_2(1)) \vee (f_1(0) = f_2(1) \wedge f_1(1) = f_2(0)))$. They *meet* in an end point p if, and only if, they are quasi-disjoint and $\exists\, a, b \in \{0,1\} : f_1(a) = p = f_2(b)$. Note that due to the uniqueness constraint the definition of *quasi-disjoint* forbids that two non-self-intersecting lines form a loop.

Next, we introduce the concept of a *block*, which is used below to specify a connected component of a complex line. In such a connected component, again due to the uniqueness constraint, an end point is either the end point of only a single curve or shared by more than two curves. The set B of *blocks* over S is defined as

$$\begin{aligned} B = \{ & \textstyle\bigcup_{i=1}^{m} l_i \mid \\ & \text{(i)} \quad m \in \mathbb{N}, \forall\, 1 \leq i \leq m : l_i \in S \\ & \text{(ii)} \quad \forall\, 1 \leq i < j \leq m : l_i \text{ and } l_j \text{ are quasi-disjoint} \\ & \text{(iii)} \quad \forall\, p \in \textstyle\bigcup_{i=1}^{m} \{f_i(0), f_i(1)\} : \\ & \qquad \mathit{card}(\{f_i \mid 1 \leq i \leq m \wedge \\ & \qquad\qquad (f_i(0) = p \ \vee \ f_i(1) = p)\}) \neq 2\} \end{aligned}$$

Two blocks $b_1, b_2 \in B$ are *disjoint* if, and only if, $b_1 \cap b_2 = \varnothing$. We are now able to give the structured definition for complex lines. The spatial data type *line3D* is defined as

$$\begin{aligned} \mathit{line3D} = \{ & \textstyle\bigcup_{i=1}^{n} b_i \mid \text{(i)} \ \ n \in \mathbb{N}, \forall\, 1 \leq i \leq n : b_i \in B \\ & \qquad\qquad \text{(ii)} \ \forall\, 1 \leq i < j \leq n : b_i \text{ and } b_j \text{ are disjoint}\} \end{aligned}$$

We call a value of this type *complex line*. No curve in one block may intersect with a curve in any other block.

The boundary of a 3D line L is the set of its end points minus those end points that are shared by several curves. The shared points belong to the interior of a 3D line. Let $E(L) = \bigcup_{i=1}^{n} \{f_i(0), f_i(1)\}$ be the set of end points of all n curves of L. We obtain

$$\begin{aligned} \partial L = E(L) - \{ & p \in E(L) \mid \mathit{card}(\{f_i \mid 1 \leq i \leq m \wedge \\ & (f_i(0) = p \ \vee \ f_i(1) = p)\}) \neq 1\} \end{aligned}$$

The closure $\overline{L}$ of a 3D line is the set of all points of L including the end points. Therefore $\overline{L} = L$ holds. For the interior of L we obtain $L^\circ = \overline{L} - \partial L = L - \partial L$, and for the exterior we get $L^- = \mathbb{R}^3 - L$, since $\mathbb{R}^3$ is the embedding space.

3.4 The Data Type *surface*

Surfaces in 3D space are especially interesting due to their role as boundaries (envelopes) of volumes (see Section 3.6). Formally, we specify a surface as the union of the images of a finite number of continuous mappings from 2D space to 3D space. The spatial data type *surface* is defined as

$$\begin{aligned} \mathit{surface} = \{ & \textstyle\bigcup_{i=1}^{n} s_i(R) \mid n \in \mathbb{N}, R \in \mathit{region2D}, \\ & \qquad \forall\, 1 \leq i \leq n : s_i : R \to \mathbb{R}^3 \text{ is a continuous mapping}\} \end{aligned}$$

We call a value of this type *complex surface*. Images of different mappings may intersect each other. The same surface object can be represented by different collections of mappings. Since the region R is bounded, a *surface* object is bounded too.

For a structured definition, we separate the point set of a *surface* object into appropriate components called *superficies*. A superficies is considered a *non-self-intersecting* surface component (and thus a restricted parametric surface) possibly with holes. Let $\mathit{sregionh2D} \subset \mathit{region2D}$ [7] be the type of single-component, simple 2D regions possibly with holes. The set of superficies is defined as:

$$\begin{aligned} \mathit{superficies} = \{ & s(R) \mid R \in \mathit{sregionh2D}, \\ & s : R \to \mathbb{R}^3 \text{ is a continuous mapping} \wedge \\ & \forall\, (x_1, y_1), (x_2, y_2) \in R^\circ : (x_1, y_1) \neq \\ & \qquad (x_2, y_2) \Rightarrow s((x_1, y_1)) \neq s((x_2, y_2)) \wedge \\ & \forall\, (x_1, y_1) \in \partial R\ \forall\, (x_2, y_2) \in R^\circ : \\ & \qquad s((x_1, y_1)) \neq s((x_2, y_2))\} \end{aligned}$$

This definition includes holes in superficies if R contains holes. The definition forbids equality of different interior points and equality of an interior point with a boundary point but allows closed superficies. A superficies S is *closed* (*partially closed*) if sets B_1 and B_2 exist with $\partial R = B_1 \cup B_2$ ($\partial R \supset B_1 \cup B_2$) and $B_1 \cap B_2 = \varnothing$ so that $s(B_1) = s(B_2)$. In this case, the set $I(S) = s(B_1)$ is part of the interior of S; otherwise $I(S) = \varnothing$. Given a superficies S, its *boundary* is $\partial S = s(\partial R) - I(S)$ and its *interior* is $S^\circ = s(R^\circ) \cup I(S)$. Hence, its closure is $\overline{S} = \partial S \cup S^\circ$, and its exterior is $S^- = \mathbb{R}^3 - \overline{S}$.

Let T be the set of all superficies over $\mathbb{R}^3$. Two superficies $S_1, S_2 \in T$ are called *quasi-disjoint* if $\overline{S_1} \cap S_2^\circ = S_1^\circ \cap \overline{S_2} = \varnothing$. We introduce the concept of a *surface component*, which is used below to specify a connected component of a complex surface. The set sc of *surface components* over T is defined as

$$\begin{aligned} sc = \{ & \textstyle\bigcup_{i=1}^{m} S_i \mid \\ & \text{(i)} \quad m \in \mathbb{N}, \forall\, 1 \leq i \leq m : S_i \in T \\ & \text{(ii)} \quad \forall\, 1 \leq i < j \leq m : S_i \text{ and } S_j \text{ are quasi-disjoint} \\ & \text{(iii)} \quad \text{Let } l \in \mathit{curve}.\ \forall\, l \subseteq \textstyle\bigcup_{i=1}^{m} \partial S_i : \\ & \qquad \mathit{card}(\{S_i \mid 1 \leq i \leq m \ \wedge \ l \subseteq \partial S_i\}) \neq 2 \\ & \text{(iv)} \quad \text{Let } l \in \mathit{curve}.\ \forall\, l \subseteq \textstyle\bigcup_{i=1}^{m} I(S_i) : \\ & \qquad \mathit{card}(\{S_i \mid 1 \leq i \leq m \ \wedge \ l \subseteq \partial S_i\}) \geq 1\} \end{aligned}$$

Condition (iii) expresses that due to the uniqueness constraint a boundary part is either the boundary of a single superficies or shared by more than two superficies. Condition (iv) deals with the special case of (partially) closed superficies. We allow that at the "seams" of a closed superficies, where boundary lines coincide and then belong to the interior, zero, one, or more superficies may meet. Two surface components $c_1, c_2 \in sc$ are *disjoint* if, and only if, $c_1 \cap c_2 = \varnothing$. We are now able to give the structured definition for complex surfaces. The spatial data type *surface* is defined as

$$\begin{aligned} \mathit{surface} = \{ & \textstyle\bigcup_{i=1}^{n} c_i \mid \text{(i)} \ \ n \in \mathbb{N}, \forall\, 1 \leq i \leq n : c_i \in sc \\ & \qquad\qquad \text{(ii)} \ \forall\, 1 \leq i < j \leq n : c_i \text{ and } c_j \text{ are disjoint}\} \end{aligned}$$

We call a value of this type *complex surface*. This definition implies that no superficies in a surface component may intersect with a superficies in any other surface component.

Let S be a *surface* object with superficies $S_1, \ldots, S_m$, and let $L(S) = \{l \in curve \,|\, l \subseteq \bigcup_{i=1}^{m} I(S_i) \ \wedge \ card(\{S_i \,|\, 1 \leq i \leq m \wedge l \subseteq \partial S_i\}) \geq 2\}$. Then the boundary of S is $\partial S = L(S) \cup \bigcup_{i=1}^{m} \partial S_i$, and the interior of S is $S^\circ = \bigcup_{i=1}^{m} \partial S_i - L(S)$. Hence, the closure of S is $\overline{S} = \partial S \cup S^\circ$, and the exterior of S is $S^- = \mathbb{R}^3 - \overline{S}$.

3.5 The Data Type *relief*

Reliefs (Figure 1) are a special and important subtype of surfaces. They describe the visible part of a surface or volume when someone looks from top onto a surface or volume and collects all (x, y)-positions with the maximum altitude. That is, each (x, y)-position is assigned at most one z-position and can be intersected at most once by a straight vertical line. The spatial data type *relief* is defined as

$$relief = \{S \in surface \,|\, (x, y, z_1) \in S \wedge (x, y, z_2) \in S \Rightarrow z_1 = z_2\}$$

Unstructured and structured definitions for this type as well as for the topological notions of boundary, interior, exterior, and closure can be taken from the type *surface*.

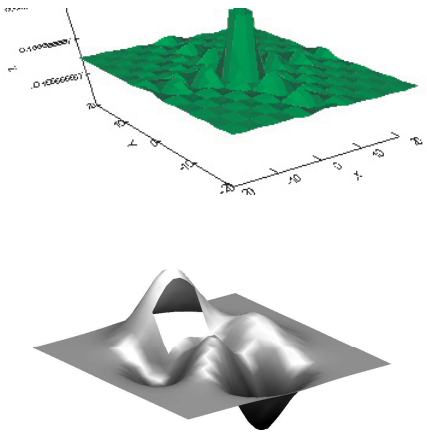

Figure 1: Example of a *relief* object with a hole.

3.6 The Data Type *volume*

Our definition of volumes is based on point set theory and point set topology [5]. Volumes are embedded into the three-dimensional Euclidean space $\mathbb{R}^3$ and modeled as special infinite point sets. We begin with some needed concepts from point set topology in $\mathbb{R}^3$.

We assume the existence of a Euclidean distance function $d : \mathbb{R}^3 \times \mathbb{R}^3 \to \mathbb{R}$ with $d(p, q) = d((x_1, y_1, z_1), (x_2, y_2, z_2)) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2}$. Let $q \in \mathbb{R}^3$ and $\epsilon \in \mathbb{R}^+$. The set $N_\epsilon(q) = \{p \in \mathbb{R}^3 \,|\, d(p, q) \leq \epsilon\}$ is called the *(closed) neighborhood of radius ϵ and center q*.

Let $X \subseteq \mathbb{R}^3$ and $q \in \mathbb{R}^3$. q is an *interior point* of X if there exists a neighborhood $N_\epsilon(q)$ such that $N_\epsilon(q) \subseteq X$. q is an *exterior point* of X if there exists a neighborhood $N_\epsilon(q)$ such that $N_\epsilon(q) \cap X = \varnothing$. q is a *boundary point* of X if q is neither an interior nor exterior point of X. q is a *closure point* of X if q is either an interior or boundary point of X. The set of all interior / exterior / boundary / closure points of X is called the *interior / exterior / boundary / closure* of X and is denoted by X° / X^- / ∂X / $\overline{X}$. A point q is a *limit point* of X if for every neighborhood $N_\epsilon(q)$ holds that $(N_\epsilon(q) - \{q\}) \cap X \neq \varnothing$. X is called an *open set* in $\mathbb{R}^3$ if $X = X^\circ$. X is called a *closed set* in $\mathbb{R}^3$ if every limit point of X is a point of X. It follows from the definition that every interior point of X is a limit point of X. Thus, limit points need not be boundary points. The converse is also true. A boundary point of X need not be a limit point; it is then called an *isolated* point of X. For the closure of X we obtain that $\overline{X} = \partial X \cup X^\circ$.

It is obvious that arbitrary 3D point sets do not necessarily form a volume. But open and closed point sets in $\mathbb{R}^3$ are also inadequate models for volumes since they can suffer from undesired geometric anomalies. A volume defined as an open point set runs into the problem that it may have missing points, lines, and surfaces in the form of punctures, cuts, and stripes. At any rate, its boundary is missing. A volume defined as a closed point set admits isolated or dangling point, line, and surface features.

Regular closed point sets [8] avoid these anomalies. Let $X \subseteq \mathbb{R}^3$. X is called *regular closed* if, and only if, $X = \overline{X^\circ}$. The effect of the *interior* operation is to eliminate dangling points, dangling lines, dangling surfaces, and boundary parts. The effect of the *closure* operation is to eliminate punctures, cuts, and stripes by appropriately supplementing points and adding the boundary. Due to their definition, closed neighborhoods are regular closed sets.

For the specification of the *volume* data type, definitions are needed for bounded and connected sets. Two sets $X, Y \subseteq \mathbb{R}^3$ are said to be *separated* if, and only if, $X \cap \overline{Y} = \varnothing = \overline{X} \cap Y$. A set $X \subseteq \mathbb{R}^3$ is *connected* if, and only if, it is not the union of two non-empty separated sets. Let $q = (x, y, z) \in \mathbb{R}^3$. Then the *length* or *norm* of q is defined as $||q|| = \sqrt{x^2 + y^2 + z^2}$. A set $X \subseteq \mathbb{R}^3$ is said to be *bounded* if there exists a number $r \in \mathbb{R}^+$ such that $||q|| < r$ for every $q \in X$.

We are now able to give an unstructured type definition for volumes. The spatial data type *volume* is defined as

$$\begin{aligned} volume = \{V \subseteq \mathbb{R}^3 \,|\, &\text{(i)} \ \ V \text{ is regular closed} \\ &\text{(ii)} \ \ V \text{ is bounded} \\ &\text{(iii) The number of connected} \\ &\quad \text{sets of } V \text{ is finite}\} \end{aligned}$$

We call a value of this type *complex volume*.

The structured definition models complex volumes as objects possibly consisting of several components and possibly having cavities. It distinguishes *simple volumes*, *simple volumes with cavities* (also called *solids*), and *complex volumes*. In order to obtain a more fine-grained and structured view of volumes, we transfer the topological predicates on 2D simple regions to simple volumes for which they are also valid [11]. The set *sv* of *simple volumes* is defined as

$$\begin{aligned} sv = \{V \subseteq \mathbb{R}^3 \,| \\ &\text{(i)} \ \ V \text{ is regular closed} \\ &\text{(ii)} \ \ V \text{ is bounded} \\ &\text{(iii)} \ \ V \text{ is homeomorphic (i.e., topologically} \\ &\quad \text{equivalent) to a neighborhood in } \mathbb{R}^3\} \end{aligned}$$

This, in particular, means that a simple volume has a connected interior, a connected boundary called *surface*, and a single connected exterior. Hence, it does not consist of several components, and it does not have cavities. The concept

of a *cavity* is topologically not directly inferable since point set topology does not distinguish between "outer" exterior and "inner" exterior of a set. This requires an explicit and constructive definition of a volume containing cavities and a use of the topological predicates *meet*, *covers*, *contains*, and *disjoint* for simple volumes. The set *svc* of *simple volumes with cavities*, also called *solids*, is defined as:

$$\begin{aligned} svc = \{V \subseteq \mathbb{R}^3 \mid \\ &\text{(i)} \quad \exists V_0, \ldots, V_n \in sv : V = V_0 - \bigcup_{i=1}^{n} V_i^\circ \\ &\text{(ii)} \quad V \text{ is connected} \\ &\text{(iii)} \quad \forall 1 \le i \le n : contains(V_0, V_i) \vee \\ &\qquad (covers(V_0, V_i) \wedge (V_0 \cap V_i \in point3D \vee \\ &\qquad V_0 \cap V_i \in line3D) \\ &\text{(iv)} \quad \forall 1 \le i < j \le n : disjoint(V_i, V_j) \vee \\ &\qquad (meet(V_i, V_j) \wedge (V_i \cap V_j \in point3D \vee \\ &\qquad V_i \cap V_j \in line3D)\} \end{aligned}$$

For $V \in svc$ with $V = V_0 - \bigcup_{i=1}^{n} V_i^\circ$, $V_1, \ldots, V_n$ are called *cavities*. Condition (i) determines the point set of a simple volume with cavities. Condition (ii) requires that despite the subtraction of point sets in (i) the remaining point set remains connected. Conditions (iii) and (iv) allow a cavity within a solid to touch the boundary of V_0 or of another cavity either in 3D point objects or 3D line objects but not in common, partial surfaces. This is necessary to ensure uniqueness of representation and to achieve closure under the geometric operations *union*, *intersection*, and *difference*. For example, subtracting a solid A from a solid B may lead to such a cavity in B. On the other hand, to allow two cavities to have a partially common surface makes no sense because then adjacent cavities could be merged to a single cavity by eliminating the common surface (similarly for surface adjacency of a cavity with the boundary of V_0).

Let $V = V_0 - \bigcup_{i=1}^{n} V_i^\circ$ be a simple volume with cavities $V_1, \ldots, V_n$. Then the boundary of V is given as $\partial V = \bigcup_{i=0}^{n} \partial V_i$ and the interior of V is given as $V^\circ = V_0^\circ - \bigcup_{i=1}^{n} V_i$. We are now able to give a structured definition for complex volumes. The spatial data type *volume* is defined as

$$\begin{aligned} volume = \{V \subseteq \mathbb{R}^3 \mid \\ &\text{(i)} \quad \exists V_1, \ldots, V_n \in svc : V = \bigcup_{i=1}^{n} V_i \\ &\text{(ii)} \quad \forall 1 \le i < j \le n : V_i^\circ \cap V_j^\circ = \varnothing \\ &\text{(iii)} \quad \forall 1 \le i < j \le n : \partial V_i \cap \partial V_j = \varnothing \vee \\ &\qquad \partial V_i \cap \partial V_j \in point3D \vee \\ &\qquad \partial V_i \cap \partial V_j \in line3D\} \end{aligned}$$

We call a value of this type *complex volume*. The definition requires of a solid to be disjoint from another solid, or to meet another solid in one or several single boundary points or curves, or to lie within a cavity of another solid and possibly share one or several single boundary points or curves with the boundary of the cavity. Solids having common surface parts with other solids or cavities are disallowed. The argumentation is similar to that for the solid definition.

Let $V = \bigcup_{i=1}^{n} V_i$ be a volume with $V_1, \ldots, V_n \in svc$. Then the boundary of V is given as $\partial V = \bigcup_{i=1}^{n} \partial V_i$, and the interior of V is given as $V^\circ = \bigcup_{i=1}^{n} V_i^\circ = V - \partial V$. The closure of V is $\overline{V} = \partial V \cup V^\circ$, and the exterior of V is $V^- = \mathbb{R}^3 - \overline{V}$.

4. GEOMETRIC SET OPERATIONS

Operations and predicates play an important role in the algebra SPAL3D. Due to space limitations, we will only focus on the geometric set operations *intersection*, *union*, and *difference*. Let SPATIAL3D = {*point3D* [$= \alpha_1$], *line3D* [$= \alpha_2$], *relief* [$= \alpha_3$], *surface* [$= \alpha_4$], *volume* [$= \alpha_5$]}, and let SURFACE = {*relief*, *surface*}. The role of the α_i's is to simplify the notations below. Moreover, we employ the two type variables α and β ranging over SPATIAL3D and/or SURFACE.

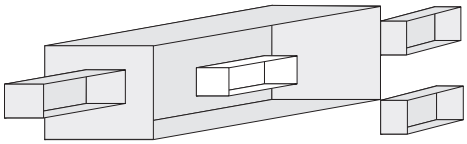

Figure 2: A volume with three (not four) components. The largest component contains a cavity.

4.1 The *union Operation*

Semantically, the *union* operation is only meaningful for operand objects of the same type. It then yields the set union of the operand objects as a result object of that type. For unions on different types, due to regularization, the result is the higher-dimensional argument; lower-dimensional objects are considered to be dangling or penetrating features. Hence, the result is uninteresting. Syntactically, however, if operations are nested, it can be meaningful to admit union on different object types, since otherwise the union operation is undefined. For equal operand types, the signatures and semantics specifications are:

$$\begin{aligned} &union : \alpha \times \alpha \to \alpha \quad \forall \alpha \in \text{SPATIAL3D} \\ &union(A, B) = A \cup B \end{aligned}$$

Since *relief* is a subtype of *surface*, the semantics above also holds for the following signatures:

$$union : \alpha \times \beta \to surface \quad \forall \alpha, \beta \in \text{SURFACE}, \alpha \neq \beta$$

Finally, for unions on different types, we obtain:

$$\begin{aligned} &union : \alpha_i \times \alpha_j \to \alpha_i \mid \alpha_j \times \alpha_i \to \alpha_i \\ &\qquad \forall \alpha_i, \alpha_j \in \text{SPATIAL3D}, \ 1 \le j < i \le 5 \\ &union(A, B) = A \mid union(A, B) = B \end{aligned}$$

4.2 The *difference Operation*

The *difference* operation returns the regularized set difference of the first object minus the second object. The operand types may be any combination of types in SPATIAL3D, even though some of them are not relevant. The result type is always the type of the first operand. Only those combinations of operand types return new results where the dimension of the second operand is equal or higher than the dimension of the first operand. If the dimension of the second operand is smaller, then the first operand is returned unchanged. Closure has to be applied to the result since otherwise anomalies can be generated. We obtain:

$$\begin{aligned} &difference : \alpha \times \alpha \to \alpha \quad && \forall \alpha \in \text{SPATIAL3D} \\ &difference : \alpha_i \times \alpha_j \to \alpha_i \quad && \forall \alpha_i, \alpha_j \in \text{SPATIAL3D}, \\ &\qquad && 1 \le i, j \le 5, \ i \neq j \\ &difference(A, B) = \overline{A - B} \end{aligned}$$

4.3 The *intersection Operation*

The *intersection* operation produces a result of a dimension smaller or equal to the dimension of the lower-dimensional

operand. For example, the intersection of a 3D line and a surface may result in 3D points and 3D lines. We define this operation for all type combinations with regularized semantics, i.e., it returns the highest-dimensional part of the result. For the lower-dimensional parts of the result, we need specialized operations which are not dealt with in this paper. The signatures of this operation are:

$$\begin{aligned} &\mathit{intersection} : \alpha \times \alpha \to \alpha \qquad \forall\, \alpha \in \text{SPATIAL3D} \\ &\mathit{intersection} : \alpha_i \times \alpha_j \to \alpha_j \mid \alpha_j \times \alpha_i \to \alpha_j \\ &\qquad\qquad \forall\, \alpha_i, \alpha_j \in \text{SPATIAL3D},\ 1 \le j < i \le 5 \end{aligned}$$

For some type combinations of the *intersection* operation, we may need to "throw away" point and / or line parts since they represent anomalies. We specify them by introducing the auxiliary operators *point_res* and *line_res*, respectively. The operator *point_res* is defined as:

$$\mathit{point\_res}(A,B) = \begin{cases} \{p \in A \cap B \mid p \text{ is isolated in } A \cap B\} \\ \quad \text{if } A, B \in \mathit{line3D} \\ \{p \in A \cap B \mid p \text{ is isolated in } A \cap B\} \\ \quad \text{if } A \in \mathit{line3D}, B \in \alpha_i, 3 \le i \le 5 \\ \{p \in \partial A \cap \partial B \mid p \text{ is isolated in } \partial A \cap \partial B\} \\ \quad \text{if } A, B \in \alpha_i, 3 \le i \le 5 \end{cases}$$

A point p is isolated in a set A, if an $\epsilon \in \mathbb{R}^+$ exists such that $N_\epsilon(p) - \{p\} = \varnothing$. The operator *line_res* is defined as:

$$\mathit{line\_res}(A,B) = \begin{cases} \bigcup\{l \subset (\partial A \cap \partial B) \cup (\partial A \cap B^\circ) \cup \\ \quad (A^\circ \cap \partial B) \cup (A^\circ \cap B^\circ) \mid l \in \mathit{line3D}\} \\ \quad \text{if } A, B \in \text{SURFACE} \\ \bigcup\{l \subset (\partial A \cap \partial B) \cup (A^\circ \cap \partial B) \mid l \in \mathit{line3D}\} \\ \quad \text{if } A \in \text{SURFACE}, B \in \mathit{volume} \\ \bigcup\{l \subset \partial A \cap \partial B \mid l \in \mathit{line3D}\} \\ \quad \text{if } A, B \in \mathit{volume} \end{cases}$$

Two surfaces may share common linear boundary parts (meeting situation), or the boundary part of one surface intersects the interior of the other surface in a line (touching situation), or both interiors intersect in a line. A surface and a volume can share linear boundary parts, or a surface can be tangent with its interior to a volume's boundary in form of a line. Two volumes can share common linear boundary parts.

Now, we can give the semantics specification of the *intersection* operation:

$$\mathit{intersection}(A,B) = \begin{cases} A \cap B \\ \quad \text{if } A \in \mathit{point3D}, B \in \alpha_i, 2 \le i \le 5 \\ (A \cap B) - \mathit{point\_res}(A,B) \\ \quad \text{if } A, B \in \mathit{line3D} \\ (A \cap B) - \mathit{point\_res}(\partial A, \partial B) \\ \quad \text{if } A \in \mathit{line3D}, B \in \alpha_i, 3 \le i \le 5 \\ (A \cap B) - (\mathit{point\_res}(A,B) \cup \\ \qquad \mathit{line\_res}(A,B)) \\ \quad \text{if } A \in \mathit{relief}, B \in \alpha_i, 3 \le i \le 5 \text{ or} \\ \qquad A \in \mathit{surface}, B \in \alpha_i, 4 \le i \le 5 \\ \overline{(A \cap B)^\circ} \\ \quad \text{if } A, B \in \mathit{volume} \end{cases}$$

We have omitted the symmetric cases.

5. CONCLUSIONS AND FUTURE WORK

In this paper, in a first step towards an abstract model for 3D data, we have given a rigorous definition of a collection of three-dimensional data types together with geometric set operations defined on them in the context of the algebra SPAL3D. These definitions clarify the structure of 3D objects from an abstract point of view and serve as a specification for a later implementation.

The abstract model of SPAL3D is incomplete in its current form. Hence, in the future we will supplement it with other operations which, e.g., produce new spatial objects, yield numerical values, or are predicates. Later, of course, an important topic will be the efficient implementation of SPAL3D.

6. REFERENCES

[1] O. Balovnev, T. Bode, M. Breunig, A.B. Cremers, W. Müller, G. Pogodaev, S. Shumilov, J. Siebeck, A. Siehl, and A. Thomsen. The Story of the GeoToolKit – An Object-Oriented Geodatabase Kernel System. *GeoInformatica*, 8(1):5–47, 2004.

[2] B. de Cambray. Three-Dimensional (3D) Modelling in a Geographical Database. *11th Int. Symp. on Computer-Assisted Cartography*, pp. 338–347, 1993.

[3] B. de Cambray and T. S. Yeh. A Multidimensional (2D, 2.5D, and 3D) Geographical Data Model. *6th Int. Conf. on Management of Data*, 1994.

[4] A. Fabri, G.-J. Giezeman, L. Kettner, S. Schirra, and S. Schönherr. On the Design of CGAL, the Computational Geometry Algorithms Library. *Software - Practice and Experience*, 30:1167–1202, 2000.

[5] S. Gaal. *Point Set Topology*. Academic Press, 1964.

[6] S. Pigot. Topological Models for 3-Dimensional Spatial Information Systems. *10th Int. Symp. on Computer-Assisted Cartography*, pp. 368–392, 1991.

[7] M. Schneider. *Spatial Data Types for Database Systems - Finite Resolution Geometry for Geographic Information Systems*, volume LNCS 1288. Springer-Verlag, Berlin Heidelberg, 1997.

[8] R. B. Tilove. Set Membership Classification: A Unified Approach to Geometric Intersection Problems. *IEEE Trans. on Computers*, C-29:874–883, 1980.

[9] P. van Oosterom, W. Vertegaal, M. van Hekken, and T. Vijlbrief. Integrated 3D Modeling within a GIS. *Int. Workshop on Advanced Geographic Data Modeling*, pp. 80–95, 1994.

[10] K. Zeitouni, B. de Cambray, and T. Delpy. Topological Modelling for 3D GIS. *4th Int. Conf. on Computers in Urban Planning and Urban Management*, 1995.

[11] S. Zlatanova. On 3D Topological Relationships. *Int. Workshop on Database and Expert System Applications*, pp. 913–919, 2000.

[12] S. Zlatanova and K. Tempfli. Data Structuring and Visualization of 3D Urban Data. *Int. Conf. of the Association of Geographic Laboratories in Europe*, 1998.