Parametric Spatiotemporal Objects

Jan Chomicki
Dept. of Computer Science and Engineering
University at Buffalo
Buffalo, NY 14260
chomicki@cse.buffalo.edu

Peter Z. Revesz
Dept. of Computer Science and Engineering
University of Nebraska–Lincoln
Lincoln, NE 68588
revesz@cse.unl.edu

Abstract

We present a framework for specifying spatiotemporal objects using spatial and temporal objects, and a parametric geometric transformation. We define a number of classes of spatiotemporal objects and summarize their closure properties under set-theoretic operations.

1 Introduction

Many natural or man-made phenomena have both a spatial and a temporal extent. Consider for example, a forest fire or property histories in a city. To store information about such phenomena in a database one needs appropriate data modeling constructs. We claim that a new concept, *spatiotemporal object*, is necessary. In this paper, we present a very general framework for specifying spatiotemporal objects. To define a spatiotemporal object we need a spatial object, a temporal object, and a continuous geometric transformation (specified using a parametric representation) that determines the image of the spatial object at different time instants belonging to the temporal object. In this framework, a number of classes of spatiotemporal objects arise quite naturally. We summarize the results about the closure of those classes under set-theoretic operations and sketch how spatiotemporal objects can be implemented using existing database technology. The framework presented here was first proposed in [4]. The closure results are from [4, 10, 17].

To appreciate the need for applying set-theoretic operators to spatiotemporal objects, consider the following scenario. Let two spatial objects represent the extents of the safe areas around two different ships. Taking into account the movement of ships, the extents of the safe areas over a period of time can be represented as two spatiotemporal objects. To avoid collisions, one needs to be able to determine the intersection of those objects. The substantial literature on spatial and temporal databases does not provide much guidance in dealing with spatiotemporal phenomena. Spatial databases [18] deal with spatial objects (e.g., rectangles or polygons) and temporal databases [16] with temporal ones (e.g., time intervals). Their combination can handle *discrete* change [17] but not *continuous* change, which is required by applications dealing with phenomena like movement, natural disasters, or the growth of urban areas. In the latter applications, the temporal and spatial aspects cannot be conveniently separated.

2 Basic notions

2.1 Objects

Definition 2.1 A *spatial object of dimension* d is a subset of R^d. A *temporal object* is a subset of R (we assume a single temporal dimension). A *spatiotemporal object of dimension* d is a subset of R^{d+1}.

These definitions are very general. We will later study restricted classes of spatial and temporal objects that are important from a practical point of view and have simple representations. Such classes have been identified in the course of spatial and temporal database research. However, it is much less clear what are the "natural" spatiotemporal objects and how to represent them. The geometric approach that we present here postulates that a spatiotemporal object be defined as a spatial object together with a continuous transformation that produces an image of this object for every time instant. In the following, let R be the field of real numbers.

Definition 2.2 An *atomic geometric object* o *of dimension* d is a quadruple (V, v, I, f) where:

- V is a spatial object, called the *reference spatial object* of o,
- v is a time instant, called the *reference time* of o,
- I is subset of R, called the *time domain* of o ($v \in I$),
- f is a function from $R^d \times R$ to R^d called the *transformation function* of o.

The semantics of o is given by the corresponding spatiotemporal object s_o defined as follows:

$$s_o = \{(\bar{y}, z) : \exists \bar{x} \in R^d.\ \bar{x} \in V \wedge z \in I \wedge \bar{y} = f(\bar{x}, z - v).$$

Notice that the transformation function is defined using the time relative to the reference time. We will use t to refer to this time. It can be negative.
A transformation function f has to satisfy at least the following *consistency* requirement for every $\bar{x} \in V$:

$$f(\bar{x}, 0) = \bar{x}.$$

This means that the snapshot of the spatiotemporal object for $t = 0$ (absolute time equal to the reference instant) is the reference spatial object. In addition, the function f can satisfy two natural *continuity* properties:

- *temporal continuity:* for every $\bar{x} \in V$, the function $f_{\bar{x}}(t) = f(\bar{x}, t)$ is continuous;
- *spatial continuity:* for every $t \in I$, the function $f_t(\bar{x}) = f(\bar{x}, t)$ is continuous.

Intuitively, temporal continuity is violated if for some t_0 there is a jump or a gap at $f_{\bar{x}}(t_0)$. Spatial continuity is violated if there are holes in a snapshot of a spatiotemporal object.

Definition 2.3 A *molecular geometric object o of dimension d* is a finite set of atomic geometric objects of dimension d whose time domains are disjoint.
Discrete change is modeled using molecular geometric objects consisting of atomic objects whose transformation functions are identities. Thus discrete change is a special case of continuous change.

2.2 Classes of geometric objects

Special classes of geometric objects are defined using restrictions on their reference spatial objects, time domains, or transformation functions. We use the notation $A^{\mathcal{S},\mathcal{T},\mathcal{F}}$ to refer to the class of atomic geometric objects whose reference spatial objects belong to the class $\mathcal{S}$ of spatial objects, time domains to the class $\mathcal{T}$ of subsets of R, and transformation functions to the class $\mathcal{F}$ of functions. Similarly, we'll denote $B^{\mathcal{S},\mathcal{T},\mathcal{F}}$ to refer to the class of molecular geometric objects consisting entirely of $A^{\mathcal{S},\mathcal{T},\mathcal{F}}$ atomic objects and $(B^{\mathcal{S},\mathcal{T},\mathcal{F}})^*$ –to the class of finite unions of $B^{\mathcal{S},\mathcal{T},\mathcal{F}}$ molecular objects.
For the purpose of this paper we fix the number of spatial dimensions $d = 2$. We consider now classes of concrete temporal objects, spatial objects and functions. In this way we obtain classes of concrete spatiotemporal objects. For $\mathcal{T}$, we consider only *intervals*. For $\mathcal{S}$, we consider *Rect* (rectangles with all sides parallel to the axes) and *Polygons* (convex polygons).
There are many more choices for $\mathcal{F}$. In this paper we consider the following classes of parametric transformations:

1. Aff: affinities defined by a pair (A, B) where A is a $d \times d$-matrix (whose elements are functions of t) and B is a d-vector of functions of t (called the *displacement vector*). Then

$$f(\bar{x}, t_0) = A_{t_0}\bar{x} + B_{t_0}$$

   where A_{t_0} (resp. B_{t_0}) is obtained by substituting t in A (resp. B) by t_0. We specialize Aff to subclasses obtained by fixing the class of functions of t allowed in A and B. We have: Aff^{Rat} (rational functions which are quotients of polynomials), Aff^{Poly} (polynomials), and Aff^{Lin} (polynomials of degree 1).

2. Sc: a subclass of Aff where the matrix A is of the form

$$\begin{bmatrix} f_1(t) & 0 \\ 0 & f_2(t) \end{bmatrix}$$

   (this corresponds to (x, y)-scaling and translation). Similarly to Aff, we define the subclasses Sc^{Rat}, Sc^{Poly}, and Sc^{Lin}.

3. $Trans$: translations – a subclass of Sc where the defining pair (A, B) is such that A is the diagonal matrix (this corresponds to translations only). Again, we also have $Trans^{Rat}$, $Trans^{Poly}$, and $Trans^{Lin}$.

4. Id: a subclass of $Trans$ where B is the zero vector.

Using our framework, one can represent various kinds of continuous change: movement, growth, or shrinking. Also, discrete change can be modeled adequately. For example appearance/disappearance can be modeled by having a molecular spatiotemporal object with several separate atomic spatiotemporal objects, each representing a different incarnation.
Notation: to simplify the notation we will write $(\mathcal{S}, \mathcal{F})$ for $(B^{\mathcal{S},\mathcal{T},\mathcal{F}})^*$, as we consider only temporal intervals and molecular objects.

3 Examples

Example 1: Suppose we are given an object o_1 which is a moving rectangle. At the reference time the rectangle has left-lower corner $(9, 10)$ and right-upper corner $(19, 20)$. Suppose that during the next five units of time the rectangle is moving left with a speed of one unit decrease in x for each unit of time.
The object o_1 can be represented by a quadruple (V_1, v_1, I_1, f_1) where V_1 is the set of points $\{(x, y) : 9 \leq x \leq 19, 10 \leq y \leq 20\}$, v_1 is 0, I_1 is $0 \leq t \leq 5$, and f_1 is a transformation function that is composed of a matrix A_1:

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

and displacement vector B_1:

$$\begin{bmatrix} -t \\ 0 \end{bmatrix}$$

Example 2: Suppose we are given an object o_2 which is a moving rectangle. At the reference time the rectangle has left-lower corner $(15, 8)$ and right-upper corner $(25, 17)$. Suppose that during the next five units of time the rectangle is moving left with a speed of two units decrease in x for each unit of time.

The object o_2 can be represented by a quadruple (V_2, v_2, I_2, f_2) where V_2 is the set of points $\{(x, y) : 15 \leq x \leq 25, 8 \leq y \leq 17\}$, v_2 is 0, I_2 is $0 \leq t \leq 5$, and f_2 is a transformation function that is composed of a matrix A_2:

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

and displacement vector B_2:

$$\begin{bmatrix} -2t \\ 0 \end{bmatrix}$$

Figure 1 shows the snapshots of o_1 and o_2 at $t = 0$ (thick black lines) and $t = 5$ (thin gray lines). The filled areas represent the intersection of o_1 and o_2 at $t = 0$ (black) and $t = 5$ (gray).

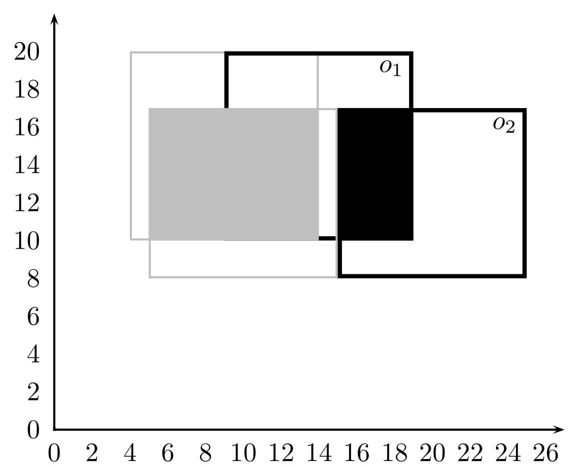

Figure 1: Rectangle intersection

The intersection of o_1 and o_2 can be represented by a new object o_3 which is a quadruple (V_3, v_3, I_3, f_3) where V_3 is the set of points $\{(x, y) : 15 \leq x \leq 19, 10 \leq y \leq 17\}$, v_3 is 0, I_3 is $0 \leq t \leq 5$, and f_3 is a transformation function that is composed of a matrix A_3:

$$\begin{bmatrix} \frac{1}{4}t + 1 & 0 \\ 0 & 1 \end{bmatrix}$$

and displacement vector B_3:

$$\begin{bmatrix} -\frac{23}{4}t \\ 0 \end{bmatrix}$$

The function f_3 is a Sc^{Lin} transformation function and the intersection itself is a $(Rect, Sc^{Lin})$ object. So although the objects o_1 and o_2 are represented using translations, a more general geometric transformation – scaling – is necessary to represent their intersection.

4 Closure

Closure under set-theoretic operations (intersection, union, set difference) is essential for the spatiotemporal objects to be usable in the context of query languages. For example, intersection is required by the spatiotemporal equijoin. Notice that the spatial and temporal objects considered separately are closed under such operations, so the challenge is to consider space and time together in a single spatiotemporal object. For spatiotemporal objects representing discrete change spatiotemporal intersection reduces to spatial and temporal intersection. However, for more general spatiotemporal objects this is not the case.

In general, the result of applying a set-theoretic operator to two spatiotemporal objects of a given class may fail to be an object of this class. It would then be essential to determine the smallest possible class containing such a result. Thus, in the study of closure, we have two kinds of results: positive and negative. For positive results, one shows that applying a set-theoretic operator to any objects of class C_1 always results in an object of class C_2 (if $C_1 = C_2$ in this case, then C_1 is *closed* under the operator). For negative results, one shows that there are objects of class C_1 that the result of a set-theoretic operator applied to those objects is not an object of class C_2 (if $C_1 = C_2$ in this case, then C_1 is *not closed* under the operator).

It is easy to see that the closure of the classes of spatiotemporal objects under union is immediately obtained. Also, the behavior of those classes under set difference is identical to that under intersection, so in the following we only present the results about intersection. The papers [4, 10, 17] provide a complete characterization of the closure under intersection. We summarize those results here, elaborating on some special cases. The general picture is that in most cases we obtain closure for spatiotemporal objects based on rectangles but not for those that are based on arbitrary polygons.

We start by formulating in our framework a result of Worboys. Worboys' framework [17] is capable of representing only discrete change.

Theorem 4.1 *[17]* $(Polygons, Id)$ *is closed under intersection.*

The following results deal with classes of spatiotemporal objects that can represent continuous change.

Theorem 4.2 *[4]* $(Rect, Sc^{Lin})$ *is closed under intersection .*

Proof: It is obvious that two sets of rectangular objects are closed under any of the above operators if any two arbitrary rectangular objects are closed under the above operators.

Let us assume that o_1 is a $(Rect, Sc^{Lin})$ object represented by a quadruple (V_1, v_1, I_1, f_1) where V_1 is the set of points $\{(x,y) : e_1^x \theta x \theta k_1^x, e_1^y \theta y \theta k_1^y\}$, v_1 is t_0, I_1 is $g_1 \theta t \theta h_1$, where θ is either $<$ or $\leq$, and f_1 is a transformation function that is composed of a matrix A_1:

$$\begin{bmatrix} \alpha_1^x t + \gamma_1^x & 0 \\ 0 & \alpha_1^y t + \gamma_1^y \end{bmatrix}$$

and displacement vector B_1:

$$\begin{bmatrix} \beta_1^x t + \delta_1^x \\ \beta_1^y t + \delta_1^y \end{bmatrix}$$

Similarly, let us assume that o_2 is another $(Rect, Sc^{Lin})$ object similar to o_1 except that each parameter is represented with superscript 2 in o_2 and $v_2 = v_1$.

Then the intersection of o_1 and o_2 can be represented by a new object o_3 which is a quadruple (V_3, v_3, I_3, f_3) where V_3 is the set of points in the intersection of V_1 and V_2, $v_3 = v_2 = v_1$, I_3 is the intersection of I_1 and I_2, and f_3 is a transformation function that is composed of a matrix A_3:

$$\begin{bmatrix} \alpha_3^x t + \gamma_3^x & 0 \\ 0 & \alpha_3^y t + \gamma_3^y \end{bmatrix}$$

and displacement vector B_3:

$$\begin{bmatrix} \beta_3^x t + \delta_3^x \\ \beta_3^y t + \delta_3^y \end{bmatrix}$$

To define the necessary parameters, we first for any t and z either x or y,
$P^z(t)^- = \max((\alpha_1^z t + \gamma_1^z)e_1^z + (\beta_1^z t + \delta_1^z), (\alpha_2^z t + \gamma_2^z)e_2^z + (\beta_2^z t + \delta_2^z))$
$P^z(t)^+ = \min((\alpha_1^z t + \gamma_1^z)k_1^z + (\beta_1^z t + \delta_1^z), (\alpha_2^z t + \gamma_2^z)k_2^z + (\beta_2^z t + \delta_2^z))$
$\Delta^z(t) = P^z(t)^+ - P^z(t)^-$

In the above $P^z(t)$ is the interval that is the projection of the spatiotemporal object o_3 onto the z axis at time t. The superscript $-$ and $+$ after $P^z(t)$ means the left or right end point of this interval and $\Delta^z(t)$ means the length of the interval.

We assume that $\Delta^z(0) \neq 0$. Since the reference time of any object can be changed to another value, it is possible to find a reference time where this condition holds for any pair of non-degenerate objects with a non-empty intersection. This we do as follows.

At $t = 1$ we have for coefficients $a_1 = \alpha_3^x + \gamma_3^x$ and $b_1 = \beta_3^x + \delta_3^x$,
$P^x(1)^- = a_1 P^x(0)^- + b_1$
$P^x(1)^+ = a_1 P^x(0)^+ + b_1$
Hence $a_1 = \frac{\Delta^x(1)}{\Delta^x(0)}$ and
$b_1 = P^x(1)^- - \frac{\Delta^x(1)}{\Delta^x(0)} P^x(0)^-$
Similarly, for $t = 2$ we have for coefficients $a_2 = 2\alpha_3^x + \gamma_3^x$ and $b_2 = 2\beta_3^x + \delta_3^x$,
$P^x(2)^- = a_2 P^x(0)^- + b_2$
$P^x(2)^+ = a_2 P^x(0)^+ + b_2$
where $a_2 = \frac{\Delta^x(2)}{\Delta^x(0)}$ and
$b_2 = P^x(2)^- - \frac{\Delta^x(2)}{\Delta^x(0)} P^x(0)^-$
Solving for the coefficients, we have:
$\alpha_3^x = a_2 - a_1$
$\beta_3^x = b_2 - b_1$
$\gamma_3^x = 2a_1 - a_2$
$\delta_3^x = 2b_1 - b_2$
Substituting we get the following equations for both when z is x and similarly when z is y.
$\alpha_3^z = \frac{\Delta^z(2) - \Delta^z(1)}{\Delta^z(0)}$
$\gamma_3^z = \frac{2\Delta^z(1) - \Delta^z(2)}{\Delta^z(0)}$
$\beta_3^z = -P^z(0)^- \alpha_3^z - P^z(1)^- + P^z(2)^-$
$\delta_3^z = -P^z(0)^- \gamma_3^z + 2P^z(1)^- - P^z(2)^-$
□

Theorem 4.3 *[10] $(Rect, Sc^L)$ is closed under intersection for $L \in \{Poly, Rat\}$.*

Theorem 4.4 *[4] $(Polygons, Sc^L)$ is not closed under intersection for $L \in \{Lin, Poly\}$.*

Proof: Consider a spatiotemporal object $o_1 = (V_1, v_1, I_1, f_1)$ where V_1 is a right-angled triangle with vertices $(0,0), (0,1), (1,0)$, $v_1 = 0$, $I_1 = [0,5]$ and f_1 is given as the matrix

$$\begin{bmatrix} 1 & 0 \\ 0 & t+1 \end{bmatrix}$$

together with a zero displacement vector, and a spatiotemporal object $o_2 = (V_2, v_2, I_2, f_2)$ where V_2 is a right-angled triangle with vertices $(0,0), (1,0), (1,1)$, $v_2 = 0$, $I_2 = I_1$ and f_2 is given as the matrix

$$\begin{bmatrix} 1 & 0 \\ 0 & 2t+1 \end{bmatrix}$$

together with a zero displacement vector. The intersection of those two triangles is another triangle with the vertices $(0,0), (1,0), (x_t, y_t)$. It is easy to see that

$$x_t = \frac{y_t}{2t+1}$$

and

$$1 - x_t = \frac{y_t}{t+1}.$$

Thus

$$x_t = \frac{t+1}{3t+2}$$

and

$$y_t = \frac{(t+1)(2t+1)}{3t+2}.$$

Figure 1 shows the snapshots of o_1 and o_2 at $t = 0$ (thick black lines) and $t = 2$ (thin gray lines). The filled areas represent the intersection of o_1 and o_2 at $t = 0$ (black) and $t = 2$ (gray). The intersection of o_1 and o_2 can be represented as two spatiotemporal objects

$o_3 = (V_3, v_3, I_3, f_3)$ and $o_4 = (V_4, v_4, I_4, f_4)$. For the object o_3, V_3 is a right-angled triangle with the vertices $(0,0), (\frac{1}{2}, 0), (\frac{1}{2}, \frac{1}{2})$, $v_3 = 0$, $I_3 = I_1$, and f_3 is given as

$$\begin{bmatrix} \frac{2(t+1)}{3t+2} & 0 \\ 0 & \frac{2(t+1)(2t+1)}{3t+2} \end{bmatrix}$$

For the object o_4, V_4 is a right-angled triangle with the vertices $(\frac{1}{2}, 0), (1, 0), (\frac{1}{2}, \frac{1}{2})$, $v_4 = 0$, $I_4 = I_1$, and $f_4 = f_3$. It is clear that o_3 and o_4 cannot be expressed using scaling which is linear, or even polynomial, in t. $\Box$

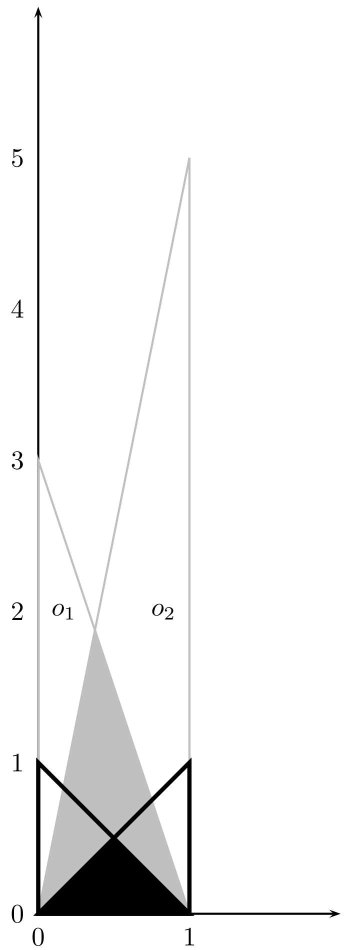

Figure 2: Triangle intersection

In [4], we conjectured that $(Polygons, Sc^{Rat})$ is closed under intersection but Haesevoets and Kuijpers [10] showed that it is not the case.

Theorem 4.5 *[10]* $(Polygons, Sc^{Rat})$ *is not closed under intersection.*

Haesevoets and Kuijpers [10] established also a number of results about affinities and translations.

Theorem 4.6 *[10]* (C, Aff^{Rat}) *is closed under intersection for* $C \in \{Rect, Polygons\}$. $(Polygons, Aff^{L})$ *is not closed under intersection for* $L \in \{Lin, Poly\}$.

Theorem 4.7 *[10]* $(C, Trans^{L})$ *is not closed under intersection for* $C \in \{Rect, Polygons\}$ *and* $L \in \{Lin, Poly, Rat\}$.

They also considered spatial objects that are triangles or triangles with the sides parallel to the coordinate axes. The closure results for the classes of spatiotemporal objects based on those two classes parallel the above results about polygon-based classes.

5 Implementation issues

To implement our approach, it is sufficient to be able to represent in a database the following:

- spatial objects (a solved problem for many classes of such objects),
- temporal objects (again a solved problem),
- function objects (lambda terms).

Although to our knowledge none of the currently available DBMS provides the last option, we believe that the object-relational (or object-oriented) technology will soon make it feasible. In fact, one of the earliest object-relational DBMS, Postgres [15], allowed storing functions as tuple components. Also, some object-oriented data models, e.g., OODAPLEX [19], permit functions as first-class objects.
Moreover, storing functions themselves is sometimes not necessary. If the transformation functions are polynomials or rational functions, they can be represented as lists of coefficients. For linear polynomials, such lists are of fixed length, opening the possibility of representing the corresponding spatiotemporal objects using the standard relational data model.

6 Related work

Spatiotemporal data models and query languages are a topic of growing interest.
The paper [17], mentioned in Section 4, presents one of the first such models. However, it is only capable of modeling discrete change.
In [6] the authors define in an abstract way *moving points and regions*. Apart from moving points, no other classes of concrete, database-representable spatiotemporal objects are defined. In that approach continuous movement (but not growth or shrinking) can be modeled using linear interpolation functions. In the subsequent paper [7], the authors discuss *moving*, *growing* and *shrinking regions*, imposing an additional requirement that the resulting spatiotemporal object be a polyhedron. This guarantees closure but eliminates the possibility of representing scaling (see Figure 2) and more general transformations.
In [8] the authors propose a formal spatiotemporal data model based on *constraints* in which, like in [17], only

discrete change can be modeled. An SQL-based query language is also presented.

We have proposed elsewhere [5] a spatiotemporal data model based on *parametric polygons*: polygons whose vertices are defined using linear functions of time. This model is also capable of modeling continuous change but is not closed under intersection. A variation of this model restricted to rectangles but extended with periodic functions is given in [2]. The latter model is closed under set theoretic operators, enabling the definition of an extended relational algebra query language, for which query evaluation can be done in PTIME in the size of the input spatiotemporal database. The closure properties for [5] and [2] seem analoguous to the closure properties of the framework presented in this paper, especially Theorems 4.4 and 4.2, respectively, but the relationships among these frameworks needs to be further explored.

Both discrete and continuous change can be represented using *constraint databases* [12]. Compared to the latter technology, our approach seems more constructive and amenable to implementation using standard database techniques as outlined in Section 5. On the other hand, constraint databases do not suffer from the lack of closure under intersection. To some degree, it is due to the fact that the intersection of two generalized tuples in constraint databases need not immediately computed but rather the tuples may be only conjoined together. In most implementations of constraint databases [1, 9, 14] the "real" computation of the intersection occurs during projection or the presentation of the query result to the user. It is unclear whether such a strategy offers any computational advantages over the approach in which the intersections are computed immediately. In fact, recent work on spatial constraint databases [13] proposes extensions to relational algebra that require immediate computations of spatial object intersections. Also, our approach is potentially more general than constraint databases. For example, by moving beyond rational functions (but keeping the same basic framework) we can represent rotations with a fixed center. Finally, in our model it is easy to obtain any snapshot of a spatiotemporal object, making tasks like animation straightforward. It is not so in constraint databases where geometric representations of snapshots have to be explicitly constructed from constraints [3].

7 Conclusions and Future Work

We have presented a formal framework for specifying a broad spectrum of spatiotemporal objects. We believe the framework is quite practical. Within this framework, we have formulated the issue of *closure* of spatiotemporal objects under set-theoretic operations. The mostly negative results about closure presented here indicate that a richer representation for spatiotemporal objects may be necessary to support queries like spatiotemporal join that require intersecting such objects. Perhaps, similarly to Constructive Solid Geometry [11], one should consider objects that are AND-OR-NOT trees with leaves corresponding to atomic objects. For such objects closure will be much easier to obtain. On the other hand, the content of such objects will be less explicit, which will make their processing during display or animation less computationally efficient.

A common limitation for each of the parametric data models considered in this and earlier papers is their inability of representing more complex motions such as rotation with a moving center. One possible approach here is to represent such rotations with an extra time-parametric function $\theta(t)$ and two extra rotation center functions $x_r(t)$ and $y_r(t)$ where $(x_r(t), y_r(t))$ is the center of rotation at any time t. That is, the object that would be normally placed at a certain position is rotated $\theta(t)$ degrees around $(x_r(t), y_r(t))$ (the center of rotation). Consider for example a triangle with corner vertices $(0, 0), (1, 0)$ and $(0, 1)$. Suppose that the center of rotation is always (t, t) and the degree of rotation is always πt degrees. Then for example at $t = 1$, the object would be another triangle with corner vertices $(2, 2), (2, 1)$ and $(1, 2)$.

There are a number of interesting issues for each parametric spatiotemporal model. First, we need to compare the expressive power of the various data models. Second, we need to study closure under set operators. For example, is the above representation closed under intersection? Third, we need to study the query expressiveness and interoperability. Fourth, what is the complexity of query evaluation assuming that a query language can be defined under which the data model has a closed-form? Fifth, we need to look at implementation issues, for example query optimization if a query language exists. Finally, as a long-term goal, we would like to look at actual implementations of several parametric data models.

Acknowledgments

The work of the first author was supported by NSF grant IRI-9632870. The work of the second author was supported by NSF grants IRI-9632871 and IRI-9625055, and by a Gallup Research Professorship.

REFERENCES

[1] A. Brodsky, V. Segal, J. Chen, and P. Exarkhopoulo. The ccube constraint object-oriented database system. *Constraints*, 2:3-4:245–277, 1997.

[2] M. Cai, D. Keshwani, and P.Z Revesz. Parametric rectangles: A model for querying and animating spatiotemporal databases. In *Proc. 7th International Conference on Extending Database Technology*, LNCS 1777, pages 430–444. Springer, 2000.

[3] J. Chomicki, Y. Liu, and P.Z. Revesz. Animating spatiotemporal constraint databases. In *Proc.*

Workshop on Spatio-Temporal Database Management, LNCS 1262, pages 142–161. Springer–Verlag, 1999.

[4] J. Chomicki and P. Z. Revesz. A Geometric Framework for Specifying Spatiotemporal Objects. In *International Workshop on Time Representation and Reasoning*, pages 41–46, Orlando, Florida, May 1999. IEEE Computer Society.

[5] J. Chomicki and P. Z. Revesz. Constraint-Based Interoperability of Spatiotemporal Databases. *Geoinformatica*, 3(3):211–243, September 1999. Preliminary version in SSD'97.

[6] M. Erwig, R.H. Güting, M. M. Schneider, and M. Vazirgiannis. Spatio-Temporal Data Types: An Approach to Modeling and Querying Moving Objects in Databases. *Geoinformatica*, 3(3):269–296, 1999. Early version in ACM-GIS'98.

[7] L. Forlizzi, R. H. Güting, E. Nardelli, and M. Schneider. A Data Model and Data Structures for Moving Objects Databases. In *ACM SIGMOD International Conference on Management of Data*, pages 319–330, 2000.

[8] S. Grumbach, P. Rigaux, and L. Segoufin. Spatio-Temporal Data Handling with Constraints. In *ACM Symposium on Geographic Information Systems*, November 1998.

[9] S. Grumbach, P. Rigaux, and L. Segoufin. The DEDALE System for Complex Spatial Queries. In *ACM SIGMOD International Conference on Management of Data*, pages 213–224, June 1998.

[10] S. Haesevoets and B. Kuijpers. Closure Properties of Classes of Spatio-Temporal Objects Under Boolean Set Operations. In *International Workshop on Time Representation and Reasoning*, 2000.

[11] C. M. Hoffmann. *Geometric and Solid Modeling:An Introduction.* Morgan Kaufmann, 1989.

[12] P. C. Kanellakis, G. M. Kuper, and P. Z. Revesz. Constraint Query Languages. *Journal of Computer and System Sciences*, 51(1):26–52, August 1995.

[13] G. Kuper, S. Ramaswamy, K. Shim, and J. Su. A Constraint-based Spatial Extension to SQL. In *ACM Symposium on Geographic Information Systems*, November 1998.

[14] P. Z. Revesz and Y. Li. MLPQ: A Linear Constraint Database System with Aggregate Operators. In *International Database Engineering and Applications Symposium*, pages 132–137. IEEE Press, 1997.

[15] M. Stonebraker and G. Kemnitz. The POSTGRES Next-Generation Database Management System. *Communications of the ACM*, 34(10):78–92, October 1991.

[16] A. Tansel, J. Clifford, S. Gadia, S. Jajodia, A. Segev, and R. T. Snodgrass, editors. *Temporal Databases: Theory, Design, and Implementation.* Benjamin/Cummings, 1993.

[17] M. F. Worboys. A Unified Model for Spatial and Temporal Information. *Computer Journal*, 37(1):26–34, 1994.

[18] Michael F. Worboys. *GIS: A Computing Perspective.* Taylor&Francis, 1995.

[19] G. T. J. Wuu and U. Dayal. A Uniform Model for Temporal and Versioned Object-oriented Databases. In Tansel et al. [16], pages 230–247.