A Mathematical Tool to Extend 2D Spatial Operations to Higher Dimensions

Farid Karimipour[1,2], Mahmoud R. Delavar[1], and Andrew U. Frank[2]

[1] Department of Surveying and Geomatics Engineering, College of Engineering, University of Tehran, Tehran, Iran
Tel.: (+98 21) 88008841; Fax: (+98 21) 88008839
[2] Institute for Geoinformation and Cartography, Vienna University of Technology
Gusshausstr. 27-29, A-1040 Vienna, Austria
Tel.: (+43 1) 58801-12711; Fax: (+43 1) 58801-12799
karimipour@geoinfo.tuwien.ac.at,
mdelavar@ut.ac.ir,
frank@geoinfo.tuwien.ac.at

Abstract. 3D and temporal objects must be included in GIS to handle real world phenomena. Many have studied extension of spatial operations to these multi-dimensional spaces and suggested technical solutions to extend a spatial operation to a new multi-dimensional space. These technical approaches have led to developments which can not be generalized. One technique used to extend a spatial operation from 2D to a multi-dimensional space is not likely usable for another spatial operation, nor to extend the same spatial operation to another multi-dimensional space. This paper suggested studying spatial operations via their dimension-independent properties. It intends to construct a mathematical framework to integrate spatial operations of different multi-dimensional spaces (3D and time) a GIS should support. The framework will be independent of the space in which the operations are applied using algebraic structures - and more specifically category theory - that ignore those properties of operations which depend on the objects they are applied to. Implementations for some case studies for spatial operations of moving points are presented.

Keywords: Spatial operations, Multi-dimensional GIS, Algebraic Structures, Category Theory and Functor, Functional Programming Languages.

1 Introduction

Early geospatial information systems (GIS) dealt with position in a static 2D Euclidean space. To handle real world phenomena, however, applications need 3D and temporal objects. Extension of the realm of GIS to these multi-dimensional spaces has a wide range of requirements from data storage and data structure considerations to visualization strategies [1, 4, 5, 6, 7, 17, 19, 23, 24, 26].

Extension of spatial operations to higher dimensional spaces has been the subject of many studies [3, 19, 22, 29] each has developed a technical solution to extend a spatial operation to a new multi-dimensional space with least increase in complexity

O. Gervasi et al. (Eds.): ICCSA 2008, Part I, LNCS 5072, pp. 153–164, 2008.

and speed. A common shortcoming is that the extension techniques are dependent on the specific case studied. It has resulted in developments which can not be generalized [8, 28]. For example while Mostafavi had developed a technique to construct Voronoi Diagrams and Delaunay Triangulation for 2D moving points (2D+T) [22], Ledoux extended a different technique for 3D moving points (3D+T) [19]. If we are to extend these operations for another type of points, then new research is necessary to figure out how to accomplish this task. The same is true to extend other operations, say point in polygon. Any changes in the operation or destination space of an already implemented spatial operation require new research to find a solution for this extension. It is not likely to achieve multi-dimensional counterparts of all of the already implemented 2D spatial operations in this haphazard way in the near future.

This paper proposes to study spatial operations via their space-invariant properties. This is similar to the approach proposed by Felix Klein in 1872 to study geometries via their invariants properties which are independent of dimension [16]. Having implemented a spatial operation for a simple 2D space using combination of the elements of this underlying integrated framework, it can be extended to other multi-dimensional spaces. These extensions will be done through mappings which are known relationships between 2D and the desired multi-dimensional spaces. The result is a principled and generalizable method to extend spatial operations.

Section 2 explains the motivation of the research in more details. Section 3 is devoted to the underlying mathematical concepts. In Section 4, steps of extending spatial operations for moving points based on the proposed approach are explained using written code in the functional programming language Haskell. The results of this implementation for some case studies are presented in Section 5. Finally, Section 6 contains conclusions and remarks for further steps of the research.

2 Spatial Operations for Multi-dimensional Objects

Do spatial operations have space-invariant properties? This is intuitively true, because the GIS operations represent different aspects from the same real world operation. *Can we describe spatial operations based on these space-invariant properties and then generalize these abstract descriptions to any multi-dimensional space?* Frank (1999) believes that it has not yet been done for lack of efficient methods:

> "A fundamental scientific question today is how to construct complex systems from simple parts. Science is very good at analyzing individual pieces of the puzzle. The combination of these pieces to form a whole is left as "a simple exercise for the reader" – and everybody knows from experience, that these simple exercises are not easy at all… The lack of efficient methods to deal with the combination problem is likely the main reason" ([6], p. 95).

The main deficiency of current research is that it differentiates what is similar behavior of a spatial operation in different multi-dimensional spaces. It differentiates the same spatial operations in different spaces despite their unification in the real world operations. People do not think about the types of the values when doing an

operation; they do the same for adding things, independent of what is added: sheep, matches, Roman or Arabic numbers [8]. In a similar view, "A GIS must connect different conceptualizations of space and allow an integrated analysis of facts related to them" ([8], p. 86). However, the space-based and dimension differentiated views of current approaches prevent generalization and lead to patchwork.

The separation - or specialization – in how to deal with different multi-dimensional spaces is essential for exploring them, but along these separations, keeping the connections with other branches is essential. The explored space can be revised through these connections to capture more relations and establish an integrated framework. This is something which has been neglected in GIS software.

We believe there is now enough knowledge about different branches of GIScience; it is time to find the connections in order to construct an integrated framework for spatial operations in multi-dimensional spaces. While a number of methods have been developed to perform a spatial operation in different multi-dimensional spaces, they represent the same concept in the real world: there are different methods to calculate the distance between two points depending on the type of the points (2D or 3D, temporal and non-temporal, etc.), while the concept of distance in all of these multi-dimensional spaces is the same.

To prove this claim, we need a more abstract view that ignores those properties of operations which depend on the object they are applied to. It enables us to have abstract description of operations with known mappings to more specific multi-dimensional spaces so that they can be extended and combined to support a variety of multi-dimensional spaces. Frank (1999) introduces functional abstraction as a solution for such formal models:

> "… A function *square(x)* can be used in various contexts with different values for x. The same concept can be applied at a higher level of abstraction. Algebras consist of several functions that can be named and have parameters. The parameters do not stand for concrete values as in procedures, but – a step more abstract – for types. They can be combined and the type parameters duly replaced by the actual parameters, much the same way as in the application of functions" ([6], p. 96).

The required abstraction is the subject of algebra which describes an abstract class of objects and their behaviors [9, 20]. The Structure of operations in an algebra is independent of an implementation. Thus, behavior of many things can be described with the same algebra as long as their behavior is structurally equivalent [12, 14]:

> "Algebra discusses the structure of operations and defines precisely what is meant by structure. Structure of operations means properties of operations that are independent of the objects the operations are applied to. Algebra describes the 'structure' of a real world system in a precise way and independent of the representation" ([8], p. 57).

There are some advantages in an algebraic view on different spaces. Firstly, it considers the unified nature of our unique physical reality when handling the included context [13]. Secondly, if an operation in one of the spaces is developed, the implementation materials (e.g. code) can be reused for other spaces [7, 13, 14, 15].

Finally, it enables us to combine these simple extended parts to have a combined system [6].

Algebraic structures have different levels of abstraction: Set and Group are examples of algebraic structures. Somewhere at the top of the abstraction ladder, we reach category algebraic structure [21] which is an application of concepts of algebra to algebras themselves [8]. "Category theory gives a very high level abstract viewpoint: instead of discussing the properties of individual objects we directly address the properties of the operations" ([8], p. 66).

The goal is constructing a mathematical framework in which spatial operations of multi-dimensional spaces are integrated through incorporation of algebraic structures and using formalizations already used in GI science [2].

3 Approach and Scientific Background

We first concentrate on a representation which provides the minimum information about spatial operations in a general context to construct a mathematical framework to integrate spatial operations in different multi-dimensional spaces through concentration on space-invariant structural properties. In particular, the representation must be independent of the dimension of the space. Algebraic structures and especially category theory provide the tools to work formally at this abstract level. This section defines a "category" and the related concept "functor". Then it explains the application to extend spatial operations to multi-dimensional spaces.

3.1 Categories and Their Mappings

A *category* is a collection of primitive element types, a set of operations upon those types, and an operator algebra which is capable of expressing the interaction between operators and elements [10, 15]. In the mathematical language, a category C is consists of a class of objects and a class of morphisms, which are functions between objects, with composition and identity properties as follows [18]:

$$\begin{aligned}
&\forall A \in C \quad \exists e_A : A \to A \quad \ni [\forall f : A \to B, g : B \to C \Rightarrow e_A.f = f, g.e_A = g] \\
&\forall f : A \to B, g : B \to C \quad \exists h : A \to C \quad \ni h = f.g \\
&\forall f : A \to B, g : B \to C, h : A \to C \Rightarrow (f.g).h = f.(g.h) = f.g.h
\end{aligned} \tag{1}$$

Figure 1 shows an example for the category of sets with objects A, B, C, D and morphisms *f1* to *f5*.

A *functor* between two categories associates elements (objects) and operations (morphisms) from one category to another that preserves the structure and operator algebra [10]:

$$\begin{aligned}
&F(e_A) = e_{F(A)} \\
&\forall f : A \to B \in P, g : B \to C \in P \Rightarrow \\
&[F(f) : F(A) \to F(B) \in Q, F(g) : F(B) \to F(C) \in Q \ \ni F(f.g) = F(f).F(g)
\end{aligned} \tag{2}$$

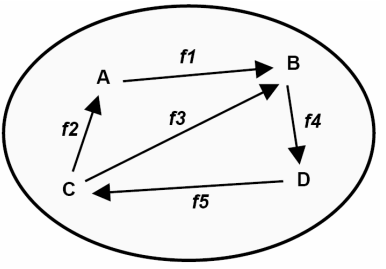

Fig. 1. A category with its objects and morphisms [14, 15]

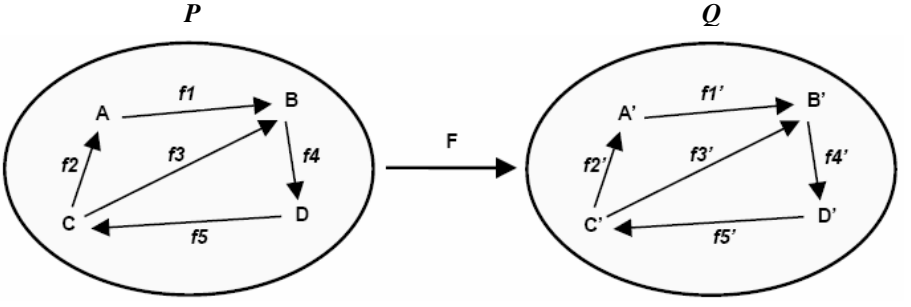

Fig. 2. Functor F transforms the first category P to the second category Q [14, 15]

The two rules guarantee that each identity morphism is mapped to its associated identity morphism and composition is preserved. Figure 2 shows functor ***F*** which transforms the category ***P*** to the category ***Q***.

3.2 Categorical Approach to Extend Spatial Operations for Multi-dimensional GIS

Representations of spatial operations in different multi-dimensional spaces have equivalent structure categories. These multi-dimensional spaces occurring in a GIS can be seen as categories with data types and primitive operations whose combination constructs more complex spatial operations. A spatial operation in the category of static 2D space based on the formal description and using the data types and primitive operations is extendable to the categories of other multi-dimensional spaces with a functor. The functor lifts the used data types and primitive operations through a defined mapping between static 2D and the desired multi-dimensional space. Then the complex spatial operations are mapped automatically, because they are defined as a combination of data types and primitive operations independent of their implementations [12, 13, 14, 15]. Figure 3 shows the described approach to extend static 2D spatial operations to their temporal 2D counterparts.

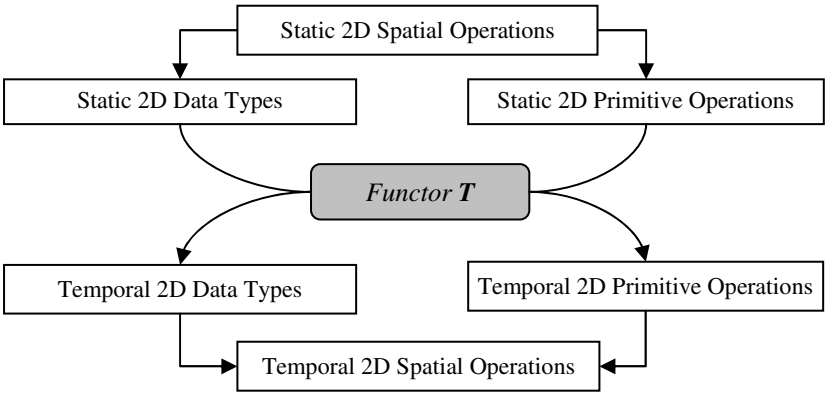

Fig. 3. Using Functor ***T*** to extend 2D spatial operations to their 3D counterparts

4 Using the Proposed Approach to Extend Spatial Operations for Moving Points

This section explains how to use the proposed approach to extend spatial operations for 2D moving points which has been implemented so far. We are working on extensions to 3D static and moving points which will be published in our future publications [11]. The code has been written in the functional programming language Haskell [25, 27].

4.1 Data Types and Their Extension

The first step of implementing spatial operations is definition of the required data types. "*Float*", "*Point*", "*Line*" and "*Polygon*" are the data types required for the spatial operations. To present moving points, we need a "*Time*" data type as well [15]:

```
data Point = Point Float Float
data Line = Line Point Point
data Polygon = Polygon [Point]
type Time = Float
```

A 2D point is composed of *x* and *y* as floating point numbers. A line is presented by its start and end points. A polygon is defined by a list of points. Finally time is defined as a discrete parameter which is presented by a floating point number.

Defining required data types, the functor *changing*, which adds time parameter "*t*" to its input value, is defined as follow [11, 15]:

```
Type Changing v = Time -> v
```

For example `Changing Float` indicates a changing floating number (i.e. a function of time).

4.2 Primitive Operations and Their Extension

To implement spatial operations, they are decomposed into a set of primitive elements. Having developed these primitive elements for 2D static points, their extension to 2D moving points is based on a functor. The mappings for time lifting, which add time parameter "*t*" to their input functions, are defined as *lift0*, *lift1* and *lift2* to lift operations with zero, one and two parameters functions, respectively [11, 15]. Lifting for operations with more arguments can be done in a similar way:

```
lift0 a = \t -> a
lift1 f a = \t -> f (a t)
lift2 f a b = \t -> f (a t) (b t)
```

For example `lift2 (+)` is the *plus* operation to add two changing number.

4.3 Spatial Operations for Moving Points

Having developed data types and primitive elements, spatial operations are defined based on their combinations. Lifting data types and primitive elements, as it was explained in subsections 4.1 and 4.2, provides lifted spatial operations, automatically.

5 Case Studies

The material prepared in section 4 was used to extend some spatial operations for 2D moving points. To verify the idea, this section first explains how the proposed approach extends distance between two 2D static points, as a primary spatial operation, for 2D moving points. Then results for three selected spatial operations "Convex Hull", "Voronoi Diagrams" and "Point in Polygon" are presented.

5.1 Distance between Two Moving Points

Calculating Euclidian distance of two 2D points requires some operations for "Float" data type [11, 15]:

```
class Number a where
   (+), (-), (*)  :: a -> a -> a
   sqr, sqrt :: a -> a
   sqr a = a * a
```

Then the class *point* which support vector plus and minus as well as distance operation is defined as follow [11, 15]:

```
class Number s => Points p s where

   x, y :: p s -> s
   x (Point x1 y1) = x1
   y (Point x1 y1) = y1
   (+), (-) :: p s -> p s -> p s
   (+)   a b = Point (x a + x b) (y a + y b)
   (-)   a b = Point (x a - x b) (y a - y b)
   dist :: p s -> p s -> s
   dist a b = sqrt(sqr((x a)-(x b))+sqr((y a)-(y b)))
```

Finally lifting the operations for numbers will provide us with a distance function which can be used for both static and moving points:

```
instance Number v => Number (Changing v) where
   (+) = lift2 (+)
   (-) = lift2 (-)
   (*) = lift2 (*)
   sqrt = lift1 (sqrt)
```

For example, if *p1* and *p2* are two 2D static points, their distance "*d*" is calculated as follows [11, 15]:

```
p1, p2 :: Point Float       --Static 2D points
p1 = Point 3.4 5.5
p2 = Point 4.5 4.5
d = dist p1 p2 --> 1.55  --distance between p1 and p2
```

And for 2D moving points *mp1* and *mp2*, their distance "*md*", which is a function of time, is calculated as follows:

```
mp1, mp2 :: Point (Changing Float)  --Moving 2D points
mp1 = Point (\t -> 4.0 + 0.5 * t)(\t -> 4.0 - 0.5 * t)
mp2 = Point (\t -> 0.0 + 1.0 * t)(\t -> 0.0 - 1.0 * t)
md = dist mp1 mp2   -- distance between mp1 and mp2
md 2  ----> 5.83    -- distance "md" for time 2
```

5.2 Results for Further Spatial Operations for Moving Points

The explained algorithm was developed for extending more complex spatial operations namely convex hull, point in polygon and voronoi diagrams to support 2D moving points [13, 14, 15], which needs more primitive operations in the class *Points*, e.g. ccw (counter-clock wise test for three points), sorting a list of points, etc. Details of coding these geometric algorithms are not relevant here. While the concept of the approach was verified, there were some details (i.e. conditional expressions) that needed attention and more research (for more details see [11]).

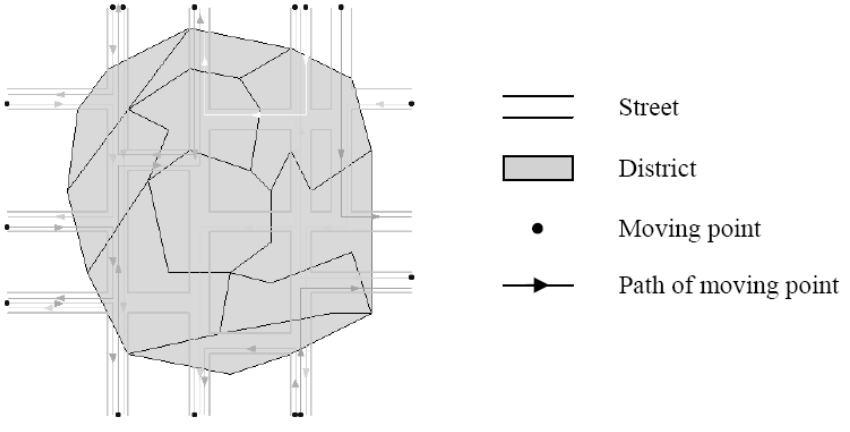

Fig. 4. Simulated environment with its streets, regions and paths of the moving points [12, 15]

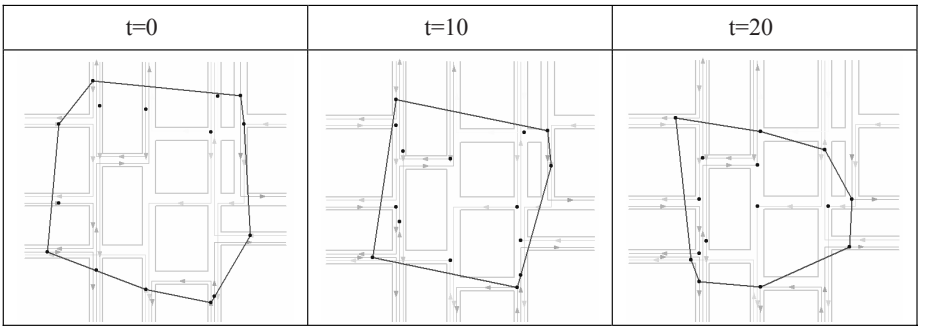

Fig. 5. Convex hull for the moving points for times 0, 10 and 20 [12, 15]

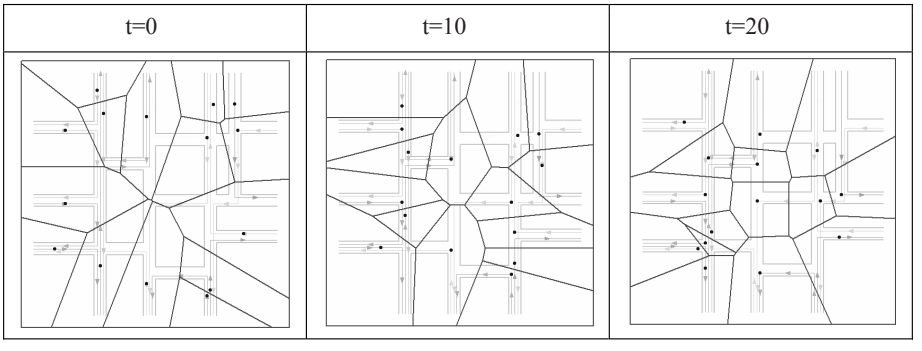

Fig. 6. Voronoi diagrams for the moving points for times 0, 10 and 20 [12, 15]

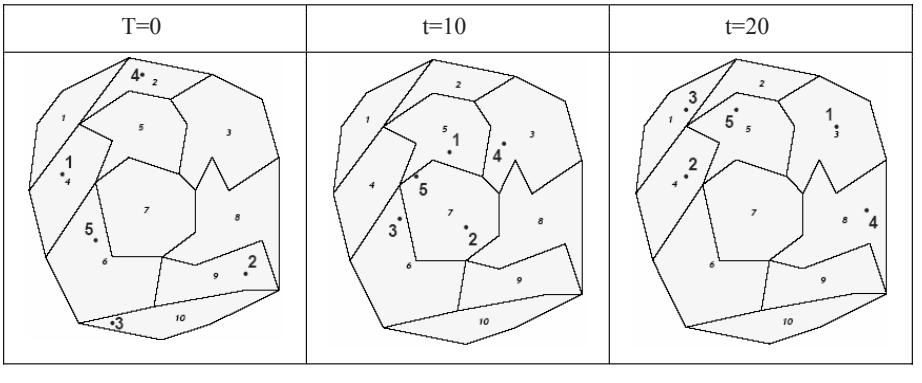

Fig. 7. Point in polygon for the moving points for times 0, 10 and 20 [12, 13, 15]

A simulated transportation system, which was made of fifteen moving points and eight regions, was selected as an example (Figure 4). The results of the convex hull and voronoi diagrams of these moving points are represented in Figures 5 and 6,

Table 1. Related regions of the moving points for times 0, 10 and 20 [12, 13, 15]

Point ID	Region		
	t=0	t=10	t=20
1	4	5	3
2	9	7	4
3	10	6	1
4	2	3	8
5	6	7	5

respectively for times 0, 10 and 20. Results of point in polygon were tested using interaction of five of the moving points with the eight regions (Figure 7 and Table 1).

6 Conclusion

Extending spatial operations to multi-dimensional objects is an essential advancement toward multi-dimensional 3D and temporal GIS. Current approaches recommend particular technical solutions to extend a spatial operation to a new multi-dimensional space. What is reported here is the extension of spatial operations via their dimension-independent properties. This approach leads to a consistent solution toward a multi-dimensional GIS.

The achieved results to extend three selected spatial operations to 2D moving points demonstrate the viability of the approach. Using the formalization of functions from category theory and the high level of abstraction of functional programming languages enabled us to implement the desired algorithm effectively. A fully general dimension-independent and automatic solution to lift all operations in the same way needs some more research which is reported elsewhere [11].

Performance (e.g. complexity and speed) is showed in most of the current research in computer science and computational geometry. However, performance is one of four areas (the others are “ontology and semantics”, “user interface” and “error and uncertain data”) that link the formal treatment of geospatial data to its use and must come after the theory for geospatial data processing [8]. “Without this clear separation, we taint the description of the things we presently understand with our ignorance in other areas” ([8], pp. 24-25). The main concern of this study is on mathematical validation of the conceptual framework first and investigation of implementation issues. Performance is investigated later and can likely be delegated to computer scientists building compilers.

References

1. Abdul-Rahman, A., Zlatanova, S.V., Coors, V.(eds.): Innovations in 3D Geo Information Systems. In: Proc. the 1st International Workshop on 3D Geoinformation, Lecture Notes in Geoinformation and Cartography, December 7-14, 2006, Kuala, Springer, Heidelberg (2006)
2. Bittner, T., Frank, A.U.: An Introduction to the Application of Formal Theories to GIS. In: Dollinger, F., Strobl, J. (eds.) Proc. Angewandte Geographische Information sverarbeitung IX (AGIT), Salzburg, Austria, pp. 11–22 (1997)

3. De Berg, M., Kreveld, M.V., Overmars, M., Schwarzkopf, O.: Computational Geometry: Algorithms and Applications. Springer, Heidelberg (2000)
4. Frank, A.U.: Qualitative Temporal Reasoning in GIS - Ordered Time Scales. In: Proc. 6th International Symposium on Spatial Data Handling (SDH 1994), Edinburgh, Scotland, IGU Commission on GIS, September 5-9 (1994)
5. Frank, A.U.: Different types of times in GIS. In: Egenhofer, M.J., Golledge, R.G. (eds.) Spatial and Temporal Reasoning in GIS, pp. 40–61. Oxford University Press, New York (1998)
6. Frank, A.U.: One step up the abstraction ladder: Combining algebras – From functional pieces to a whole. In: Freksa, C., Mark, D.M. (eds.) COSIT 1999. LNCS, vol. 1661, pp. 95–107. Springer, Heidelberg (1999)
7. Frank, A.U., Gruenbacher, A.: Temporal Data: 2nd order concepts lead to an algebra for spatio-temporal objects. In: Proc. Workshop on Complex Reasoning on Geographical Data, Cyprus, December 1 (2001)
8. Frank, A.U.: Practical Geometry - Mathematics for Geographic Information Systems, Script for GIS Theory course at TU Wien (2007)
9. Guttag, J.V., Horning, J.J.: The Algebraic Specification of Abstract Data Types. Acta Informatica 10, 27–52 (1978)
10. Herring, J., Egenhofer, M.J., Frank, A.U.: Using Category Theory to Model GIS Applications. In: Proc. 4th International Symposium on Spatial Data Handling, Zurich, Switzerland, pp. 820–829 (1990)
11. Karimipour, F.: How to Extend GIS Operations to 3D and Temporal Data, draft manuscript (2008)
12. Karimipour, F.: Logical Formalization of Spatial Analyses of Moving Objects Using Algebraic Structures, M.Sc. Thesis (in Persian with English abstract), College of Engineering, University of Tehran, Iran (2005)
13. Karimipour, F., Delavar, M.R., Frank, A.U., Rezayan, H.: Point in Polygon Analysis for Moving Objects. In: Gold, C. (ed.) Proc. 4th Workshop on Dynamic & Multi-dimensional GIS, Pontypridd, Wales, UK, ISPRS Working Group II/IV, September 5-8, 2005, pp. 68–72 (2005)
14. Karimipour, F., Delavar, M.R., Frank, A.U.: Applications of Category Theory for Dynamic GIS Analyses. In: Digital Proc. GIS Planet 2005 Conference, Estoril, Portugal, 30 May - 2 June (2005)
15. Karmipour, F., Delavar, M.R., Rezayan, H.: Formalization of Moving Objects' Spatial Analysis Using Algebraic Structures. In: Proc. Extended Abstracts of GIScience 2006 Conference, Munster, Germany, IfGI Prints, vol. 28, pp. 105–111 (2006)
16. Klein, F.: Elementary Mathematics from an Advanced Standpoint: Geometry. Dover Books on Mathematics. Dover (2004)
17. Langran, G.: Time in Geographic Information Systems, Ph.D. Dissertation, University of Washington (1989)
18. Lawvere, F.W., Schanuel, S.H.: Conceptual Mathematics: A First Introduction to Categories. Cambridge University Press, Cambridge (2005)
19. Ledoux, H.: The Kinetic 3D Voronoi Diagram: A Tool for Simulating Environmental Processes. In: Oosterom, P.V., Zlatanova, S., Penninga, F., Fendel, E. (eds.) Advances in 3D GeoInformation Systems, Proc. the 2nd International Workshop on 3D Geoinformation, Delft, the Netherlands, December 12-14, 2007. Lecture Notes in Geoinformation and Cartography, pp. 361–380. Springer, Heidelberg (2008)
20. Loeckx, J., Ehrich, H.D., Markus, W.: Specification of Abstract Data Types. John Wiley and B.G. Teubner, Chichester, UK and Stuttgart (1996)

21. MacLane, S., Birkhoff, G.: Algebra, 3rd edn. AMS Chelsea Publishing (1999)
22. Mostafavia, M.A., Gold, C., Dakowiczb, M.: Delete and Insert Operations in Voronoi/Delaunay Methods and Applications. Journal of Computers and Geosciences 29, 523–530 (2003)
23. Oosterom, P.V., Zlatanova, S., Penninga, F., Fendel, E.(eds.): Advances in 3D GeoInformation Systems. In: Proc. the 2nd International Workshop on 3D Geoinformation, Delft, the Netherlands, December 12-14, 2007. Lecture Notes in Geoinformation and Cartography. Springer, Heidelberg (2007)
24. Peuquet, D.J.: Time in GIS and Geographical Databases. In: Longley, P.A., Goodchild, M.F., Maguire, D.J., Rhind, D.W. (eds.) Geographical Information System: Principals and Technical Issues, 2nd edn., vol. 1, pp. 91–103. John Wiley & Sons, Chichester (1999)
25. Peyton Jones, S., Hughes, J.: Haskell 98: A Non-Strict, Purely Functional Language (1999), (accessed February 20, 2008) http://www.haskell.org/onlinereport/
26. Raper, J.: Multidimensional Geographic Information Science. Taylor and Francis, London (2000)
27. Thompson, S.: Haskell: The Craft of Functional Programming. Addison- Welsey, Reading (1999)
28. Rezayan, H., Frank, A.U., Karimipour, F., Delavar, M.R.: Temporal Topological Relationships of Convex Spaces in Space Syntax Theory. In: Proc. International Symposium on Spatio-temporal Modeling (ISSTM 2005), Bejing, China, August 27-29, 2005, pp. 81–91 (2005)
29. CGAL Website (accessed February 20, 2008), http://www.cgal.org/