To be presented at the*Third Canadian Conference on Computing in Civil and Building Engineering*, Montreal, 26-28 August 1996

# On a Query Language for Weighted Geometries

Rudi Stouffs
Architecture and CAAD, Swiss Federal Institute of Technology, Zurich
Ramesh Krishnamurti
Dept. of Architecture, Carnegie Mellon University, Pittsburgh, Pa.

## Abstract

Current CAD systems tend to focus on the representation of design artifacts, and on the tools and operations for their creation and manipulation. Techniques for querying are added, mostly, as afterthoughts, constrained by the data representation system and methods. Nevertheless, querying a design is as much an intricate aspect of the design process as is creation and manipulation. In this paper, we explore the foundation for a query language that allows for a rich body of queries, using both geometric and non-geometric data, and incorporating spatial rules as a syntactical expression for pattern matching on geometries. We rely on an algebraic model that specifies arithmetic operations and relations on weighted geometries. We augment these with operations derived from the conceptual techniques of counting, pattern matching and rules.

Keywords: geometric modeling, query language, algebraic model, spatial grammars.

## Introduction

Computational design relies on effective models of geometry, for the creation of design artifacts and for the querying of the characteristics of such artifacts. According to Mäntylä (1988) these representations must adequately answer “arbitrary geometric questions algorithmically.” However, this task becomes increasingly more difficult as designers pose new questions that go beyond geometry and require other information to be included. In computational design environments, particularly, emphasis has shifted from representation or visualization to manipulation and design; extensible data models become necessary for users to add new forms of information for support. Independent from whether queries are geometrical, or require other kinds of data, it is important that data be accessible in a uniform and consistent fashion, so that new queries can be constructed and posed without having to alter either the representational model or the access mechanisms in any way. A viable query language has to be based on a model for representing different types of information; moreover, this model must adhere to a consistent logic through simple and straightforward operations.

We present an algebraic model that is based on a part relationship for *weighted geometries* – those with non-geometric attributes (Krishnamurti, 1992; Stouffs, 1994). The algebraic model applies to mixed-dimensional geometries with spatial and non-spatial elements. It specifies basic arithmetic operations and supports basic geometrical relations. It also facilitates the representation and application of spatial rules. We explore the basis for a query language developed from an adaptation of this algebraic model as an extensible representational model for weighted geometries. The arithmetic operations and geometric relations form the basic components of the query language. These are augmented with operations derived from techniques of counting, pattern matching and rules. Counting provides a direct method for answering basic geometric queries. Pattern matching and spatial rules allow for the composition of more complex and versatile geometric and non-geometric queries.

## A Uniform Model for Weighted Geometries

Although it originated in shape grammar research (Stiny, 1980a, 1986, 1991), the algebraic model offers insight into geometric modeling and design exploration. It presents a uniform and consistent approach for dealing with shapes, includes non-geometric elements and attributes, handles mixed-dimensional geometries, and provides for new and extended ways for manipulating geometries through the use of spatial rules (Stouffs and Krishnamurti, 1994; Krishnamurti and Stouffs, forthcoming). Thus, it provides for an extensible and standardized data representation model (Stouffs and Krishnamurti, 1996).

## The Algebraic Model

The algebraic model supports a straightforward approach for dealing with different data *sorts* (Stouffs and Krishnamurti, 1996). The common arithmetic operations of sum, difference and product extend nicely to weighted geometries. For a consistent approach, these operations are defined in terms of a part relation for each sort. This is intuitive as the following example for line thicknesses illustrates (Stiny, 1992): a single line drawn multiple times, every time with different thickness, appears as if it were drawn once with the largest thickness, even though it assumes the same line with other thicknesses. Thus, the sum on line thicknesses is the least upper bound of the set of thickness values, corresponding to the part relation.

The same reasoning holds for other attribute sorts. Consider the situation in which points are each associated a set of labels, termed labeled points. The algebraic operations on labeled points can be considered in similar fashion to the operations on points. For example, the sum of two labeled points equals the set of both labeled points if the points are not coincident. Otherwise, the sum equals the single point associated with the set union of the two sets of labels. Likewise, the operations of product and difference apply to labeled points, using the set operations of intersection and difference on sets of labels. Thus, a labeled point is a part of another labeled point, if the points are coincident and the labels to the first point form a subset of the labels to the second point. Similar definitions hold for weighted geometries of higher dimensionalities, except that when combining overlapping geometries with different attributes, the result is depended upon three parts: the difference of the first geometry with the second, the difference of the second geometry with the first, and the common part of both geometries. The attribute value of the common part is derived in the same way as for points. In the case of sum, the other two parts receive their attribute value from the original shape each is a part of. Thus, labeled geometries is a sort which defines an algebra that is ordered by a part relation and closed under the arithmetic operations of sum, product and difference.

In general, sorts combine to form new sorts: a sort of weighted geometries is composed of sorts of geometries and weights. As another example, consider a wall composed of a number of material layers. Each layer is specified as a geometry with weights for the representation of such properties as material, color, and thermal-performance. The resulting sort specifies algebraic operations that preserve this layering structure while operating simultaneously on all layers.  In solid modeling, similar geometric operations of union, intersection and difference often obliterate the internal boundaries between different regions (Rossignac and Requicha, 1991).  The uniform and consistent way of handling attribute sorts permits the integration of user-defined sorts.  Sorts are defined semantically by the information to be presented, and syntactically by the data type adopted and the part relation defined over this type.

### **Mixed-dimensional Geometries**

An important aspect of computational design representation is the ability to represent geometries of different dimensionalities.  Consider the example of building design, where the final product is a composition of purely solid elements or building components.  Yet, the dimensionality of each individual component is not essential for all purposes, through the course of the design and evaluation process.  An abstraction is often more valuable.  In the structural evaluation of a design, walls can be represented as planes with simple integral attributes – a true three-dimensional model would be too complicated, unless approximated.  In general, even a single component may be represented as different elements of mixed-dimensionality, each element projecting information for a specific application.  Mixed-dimensional models have recently found support in solid modeling (Rossignac and Requicha, 1991; Gursoz et al., 1991).

Under the algebraic model, we can consider separate part relations for geometries of different dimensionalities, thereby specifying different sorts. This leads to the notion of a composite geometry as a single conceptual entity, consisting of distinct components of different dimensionality that are operated upon in parallel by the algebraic operations.  Composite geometries can also be regarded as consisting of multiple components that are coordinated or related.  For example, consider a set of drawings from amongst plans, elevations and sections of a same building.  Each drawing may be considered a shape, or, the set of drawings as a whole can be regarded as a composite shape.  Its components may be visualized in the same or in different spaces – these coexist without spatial interference.

## Spatial Matching and Rules

The algebraic model allows for a powerful geometric pattern matching based on the concept of *emergent* geometries: geometries that are not a priori defined, but emerge under the part relation (Stiny, 1986, 1993).  The specification of an emergent geometry is an expression of pattern matching, spatial rules are a formalism for this specification.

Fundamental to the algebraic model is that every element specifies an indefinite set of elements that are each a part of the original element.  Applied to geometries, this means that, *any* part of a geometry is a geometry and can be manipulated as such; these geometries emerge under the part relation – albeit, they were not originally envisioned as such.  Thus, users can deal with geometries in indeterminate ways.  This is quite distinct from the selection process in current CAD approaches where the only objects that can be selected correspond to those prescribed minimal entities that have been predefined in the data-structures.  Recent research suggests that emergence can be used to explain, on the basis of continuity and articulate consistency, why descriptions of design are post-rationalized so as to explain the precedents that justify the designs (Stiny, 1994; Krishnamurti and Stouffs, forthcoming).

Emergent geometries only become explicit when manipulated as such.  Recognizing emergent geometries requires determining a transformation under which a specified similar geometry is a part of the original geometry.  The specification of spatial transformations on emergent geometries leads naturally to design generation and search (Mitchell, 1993; Stiny, 1993).  More broadly, any transformation or mutation of an object into another, or into parts thereof, constitutes an action of search or exploration.

A spatial rule $a \rightarrow b$ constitutes a formal specification of geometric recognition and subsequent manipulation.  The lhs $a$ specifies the similar shape to be recognized, the rhs $b$ specifies the replacement leading to the resulting shape.  A rule application consists of replacing the emergent shape corresponding to $a$, under some allowable transformation, by $b$, under the same transformation.  In general, a rule may be considered a particular composition of a number of operations and/or transformations that is recognized and applicable as a new, single, operation.  The combination of a set of (semantically related) spatial rules into a formal rewriting system is termed a spatial grammar.  Spatial grammars have found a wide ranging use, for analysis and synthesis, in a number of fields (Stiny 1980b; Koning and Eizenberg, 1981; Flemming, 1981; Finger and Rinderle, 1989; Brown et al., 1994; Deng, 1994; Chiou and Krishnamurti, 1995; Meyer, 1995; Schmidt, 1995).  A grammar is more than a framework for generation; it is a tool that permits a structuring of a collection of rules and/or operations.  Grammars can be combined to form composite grammars (Carlson, 1993).

## Towards a Query Language

As stated in the introduction, the algebraic model for weighted geometries can be augmented with operations derived from techniques of counting, pattern matching and rules.  Below we consider each in relation to the query language.

## **Counting**

When counting geometries, it is important that we adopt a canonical representation for *shapes*, i.e., collections of disjoint (weighted) geometries.  Otherwise, different representations may yield different numbers of composing geometries.  The *maximal element* representation that underlies the algebraic model is a canonical representation for shapes.  The maximal elements of a shape are its disjoint geometries, each of which is connected.  For example, a point is maximal if it is not coincident with any other point.  Algebraically, the sum of two points is the set of both points if they are not coincident and is the single point otherwise.  We consider these properties in the following examples that illustrate the principle of counting as a method for answering a body of queries.

First, we consider trivial examples that can be efficiently implemented in any geometric modeling system.  Two geometries touch or share boundary, if the number of geometries in the summed shape is less than the sum of the number of geometries in the original shapes.  Similarly, two geometries collide or overlap if the product is non-empty, that is, the number of geometries in the product shape is greater than zero.

We next consider more difficult queries that can be solved entirely in a similar fashion, using the algebraic approach.  Assume two utility line networks in a city street, one existing, the other to be added.  It is important that the lines in the networks neither interfere nor collide.  When designing a new network, avoiding collision is straight-forward.  When merging two networks, it becomes more difficult.  Consider the following queries: determine if any two lines (each from a different network) collide, where collisions take place, and how many such colliding pairs of lines exist and which lines.

To solve this, we consider geometries with labels as weights, and sum the two sets of labeled (utility) lines.  Colliding lines will lead to geometries with more than one label.  The number of such geometries corresponds to the number of collisions, the labels of each collision specify the colliding lines and the geometry specifies the location of the collision (figure 1).  Thus, algebraically, the answers can be extracted from a single data set, derived using a sum operation.

## **Pattern Matching**

In conventional CAD systems, the objects of manipulation are a priori defined in the

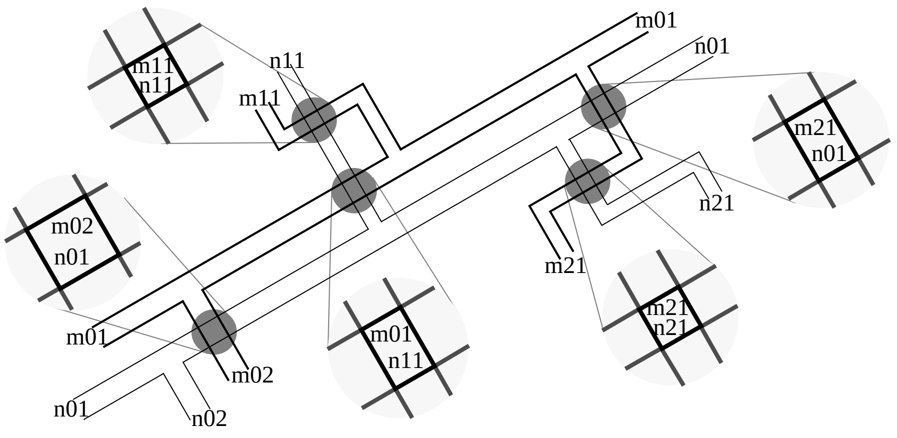

**Figure 1** Collision queries on the merging of two utility lines networks (2D representation).

data-structures. Queries can be instantiated either spatially using a lead from the user, e.g., the cursor position at query activation, or otherwise, based on object attributes. Spatially instantiated queries can be resolved using a straightforward search of the database. Attribute queries, additionally, require non-geometric pattern matching, where a pattern is specified as one or more attribute constraints. In the case of the utility lines' example, matching is performed on the number of labels assigned to a geometry, i.e., on the cardinality of the attributes' value. If a specific line is searched, or its collisions, matching constitutes a set-membership operation on labels.

In general, an answer to the query may be the queried object(s), any (geometric) aspect or attribute data of the object(s), or derived information. Consider the design of a power utility network above the ground. Using labels, we can query any pole that is on a particular circuit. We can query the length of a line, between two consecutive poles, to determine if an additional pole may be needed to support this distance. If the line is too short, on the other hand, and the poles come into a certain proximity, then, the cost may rise or a danger level may be achieved, and the circuit may have to be redesigned. Such a case may either be queried explicitly by the user or it may be specified as a trigger that automatically warns the user when the case presents itself.

In these examples, the length of a line is a geometric aspect of the object and as such an integral part of the representation, that can be queried. However, before any line is strung between two poles, the distance between these poles is not yet an aspect or attribute of any object in the database. Then, we can retrieve this distance as a derived attribute or as the result of a function that operates on both objects, so that we can query or trigger two poles that become available and satisfy the need to string a line.

The algebraic model permits much more powerful geometric pattern matching, based on the concept of emergent shapes. The specification of an emergent shape is an expression of geometric pattern matching. In general, recognizing emergent shapes relates to finding one or all valid transformations under which a shape is a part of a given shape. The set of valid transformations is commonly the set of all similarity transformations (i.e., translations, rotations, reflections and scale). The recognition or matching mechanism thus determines if a specified "similar" shape is a part of a given shape under a valid transformation. Thus, the solution to the matching problem consists of finding a correspondence between the geometries in the similar shape and geometries in the database, and of determining the transformation that represents this correspondence.

Geometric recognition serves constraint based systems, e.g., if a line is within proximity then snap to the end points of an arc. It is also useful for "sketching" devices, in conjunction with a library of geometries. A sketch of a stove may be recognized and replaced by the correct library geometry (figure 2).

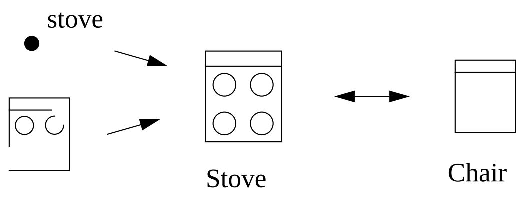

**Figure 2** Geometric recognition and replacement of library elements.

## <u>Rules</u>

A *shape rule* may be considered a particular composition of a number of operations and/or transformations that is recognized and applicable as a new, single, operation. Shape rules serve to facilitate common operations, e.g., for changing one geometry into another or for creating a new shape based on an existing shape and a rule. Examples are the recognition and subsequent replacement of library geometries or the placement of a library geometry from a labeled point (figure 2). Shape rules may also serve to define complex manipulations and relationships. For example, rules may define relationships between points for specifying connectivity, e.g., triangulation, and create collections of lines from points. Consider the example of the power utility network above the ground. Given a partial design, the path of the lines and a first pole, or a series of poles, rules can be used to specify the completion of the network (figure 3).

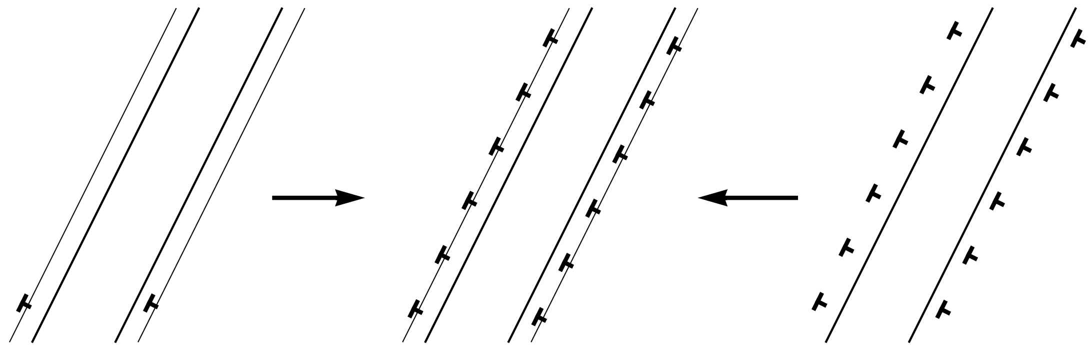

**Figure 3** Power utility network above the ground: shape rules can specify the completion of a design.

## Conclusion

The algebraic model, as an extensible representational model for weighted geometries, supports the development of a query language allowing for a rich body of queries. The arithmetic operations and the geometric relations, all defined by the part relation, augmented with operations derived from the techniques of counting, pattern matching and rules, provide for the specification of queries using both geometric and non-geometric data. Although no ideas may be radically different than any particular pieces in the industry, important is the ease in which these behaviors can be created using the algebraic model. Rules may be created via a graphical interface, fourth generation language or other rapid development mechanism. This way, behaviors can be developed on an extensive and fairly rapid basis.

## Acknowledgment

The first author benefitted from discussion with Marc Schindewolf, especially on the selection of query examples.

## References

Brown, K. N., C. A. McMahon and J. H. Sims Williams. 1994. A Formal Language for the Design of Manufacturable Objects. In *Formal Design Methods for CAD*, eds. J. S. Gero and E. Tyugu, 135-155. IFIP Transactions B: Applications in Technology, B-18.

Carlson, C. 1993. Grammatical Programming: An Algebraic Approach to the Description of Design Spaces. Ph.D. diss. Pittsburgh, Pa.: Carnegie Mellon University, Dept. of Architecture.

Chiou S. and R. Krishnamurti. 1995. The Grammar of Taiwanese Traditional Vernacular Dwellings. *Environment and Planning B: Planning and Design* 22: 689-720.

Deng, Y. 1994. Feature Based Design: Synthesizing Structure From Behavior. Ph.D. diss. Pittsburgh, Pa.: University of Pittsburgh, Dept. of Industrial Engineering.

Finger, S. and J. R. Rinderle. 1989. A Transformational Approach to Mechanical Design Using a Bond Graph Grammar. In *Proceedings of the 1st ASME Design Theory and Methodology Conference*. Montreal; September 1989.

Flemming, U. 1981. The Secret of Casa Guiliani Frigerio. *Environment and Planning B: Planning and Design* 8: 87-96.

Gursoz, E. L., Y. Choi and F. B. Prinz. 1991. Boolean Set Operations on Non-Manifold Boundary Representation Objects. *Computer Aided Design* 23: 33-39.

Koning, H. and J. Eizenberg. 1981. The Language of the Prairie: Frank Lloyd Wright's Prairie Houses. *Environment and Planning B: Planning and Design* 8: 295-323.

Krishnamurti, R. 1992. The Maximal Representation of a Shape. *Environment and Planning B: Planning and Design* 19: 267-288.

Krishnamurti, R. and R. Stouffs. forthcoming. Spatial Change: Continuity, Reversibility and Emergent Shapes. *Environment and Planning B: Planning and Design*

Mäntylä, M. 1988. *An Introduction to Solid Modeling*. Rockville, Md.: Computer Science Press.

Meyer, S. 1995. A Description of the Structural Design of Tall Buildings Through the Grammar Paradigm. Ph.D. diss. Pittsburgh, Pa.: Carnegie Mellon University, Dept. of Civil Engineering.

Mitchell, W. J. 1993. A Computational View of Design Creativity. In *Modeling Creativity and Knowledge-Based Creative Design*, eds. J. S. Gero and M. L. Maher, 25-42. Hillsdale, N. J.: Lawrence Erlbaum.

Rossignac, J. R. and A. A. G. Requicha. 1991. Constructive Non-Regularized Geometry. *Computer Aided Design* 23: 21-32.

Schmidt, L. 1995. An Implementation using Grammars of an Abstraction-based Model of Mechanical design for Design Optimization and Design Space Characterization. Ph.D. diss. Pittsburgh, Pa.: Carnegie Mellon University, Dept. of Mechanical Engineering.

Stiny, G. 1980a. Introduction to Shape and Shape Grammars. *Environment and Planning B: Planning and Design* 7: 343-351.

Stiny, G. 1980b. Kindergarten Grammars: Designing with Froebel's Building Gifts. *Environment and Planning B: Planning and Design* 7: 409-462.

Stiny, G. 1986. A New Line on Drafting Systems. *Design Computing* 1: 5-19.

Stiny, G. 1991. The Algebras of Design. *Research in Engineering Design* 2: 171-181.

Stiny, G. 1992. Weights. *Environment and Planning B: Planning and Design* 19: 413-430.

Stiny, G. 1993. Emergence and Continuity in Shape Grammars. In *CAAD Futures '93*, eds. U. Flemming and S. Van Wyk, 37-54. Amsterdam: North-Holland.

Stiny, G. 1994. Shape Rules: Closure, Continuity and Emergence. *Environment and Planning B: Planning and Design* 21: s49-s78.

Stouffs, R. 1994. The Algebra of Shapes. Ph.D. diss. Pittsburgh, Pa.: Carnegie Mellon University, Dept. of Architecture.

Stouffs, R. and R. Krishnamurti. 1994. An Algebraic Approach to Shape Computation (a Position Paper). In Workshop notes *Reasoning with Shapes in Design. Artificial Intelligence in Design '94*, 50-55. Lausanne, Switzerland: Swiss Federal Institute of Technology. Lausanne, Switzerland; August 1994.

Stouffs, R. and R. Krishnamurti. 1996. The Extensibility and Applicability of Geometric Representations. In *3rd International Conference on Design and Decision Support Systems in Architecture and Urban Planning*. Spa, Belgium; August 1996.