# Representation and Querying of Interpolation Data in Constraint Databases *

Peter Revesz and Lixin Li
Department of Computer Science and Engineering
University of Nebraska-Lincoln
Lincoln, NE 68588, U.S.A.
E–mail: {lli, revesz}@cse.unl.edu

**Abstract**

We propose using a simple linear constraint relational representation for spatial data derived using 2D shape function interpolations. We show that many queries that could not be done in traditional GIS systems can be efficiently expressed and evaluated using linear constraint database systems.

## 1. Introduction

There are many applications in the area of Geographic Information Systems where we need to use *interpolation* of given data. For example, suppose that we have the following two sets of sensory data in our database:

1. $Incoming(y, t, u)$ records the amount of incoming ultraviolet radiation $u$ for each pair of latitude degree $y$ and time $t$, where time is measured in days.

2. $Filter(x, y, r)$ records the ratio $r$ of ultraviolet radiation that is usually filtered out by the atmosphere above location $(x, y)$ before reaching the earth.

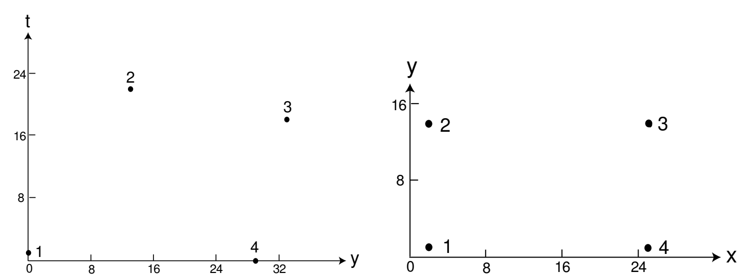


Figure 1: The spatial sample points for $Incoming$ (left) and $Filter$ (right).

Suppose that Figure 1 shows the locations of the $(y, t)$ and $(x, y)$ pairs where the measurements for $u$ and $r$, respectively, are recorded. Then Tables 1 and 2 could be instances of these two relations. Now suppose that we have the following query:

**Query 1.1** *Find the amount of ultraviolet radiation for each ground location $(x, y)$ at time $t$.*

*This research was supported in part by NSF grant EIA-0091530 and by a Gallup Research Professorship.

Since the input relations only record the incoming ultraviolet radiation $u$ and filter ratio $r$ on a few sample points, these cannot be used directly to answer the query. Therefore, to answer this query, we need first to *interpolate* $u$ and $r$ for all the points in the domain. There are many types of interpolations, but we will describe a simple one in the next section. This interpolation yields a set of linear constraints.

| ID | Y | T | U |
|---|---|---|---|
| 1 | 0 | 1 | 60 |
| 2 | 13 | 22 | 20 |
| 3 | 33 | 18 | 70 |
| 4 | 29 | 0 | 40 |

Table 1: $Incoming(y, t, u)$.

| ID | X | Y | R |
|---|---|---|---|
| 1 | 2 | 1 | 0.9 |
| 2 | 2 | 14 | 0.5 |
| 3 | 25 | 14 | 0.3 |
| 4 | 25 | 1 | 0.8 |

Table 2: $Filter(x, y, r)$.

The linear constraints express the relationship among the attributes in the relations. Therefore, as we will see, they can be represented using *constraint relations*. We give examples of constraint relations in next section. Let $INCOMING(y, t, u)$ be the constraint relation that represents the interpolation of the *Incoming* relation. Similarly, let $FILTER(x, y, r)$ be the constraint relation that represents the interpolation of the *Filter* relation. Both constraint relations can be thought of as an infinite relation of triples of rational numbers. To write queries, we do not need to know the precisely the constraints that are used in the representation of the infinite relations. Indeed, the above query can be already expressed in Datalog as follows:

$$\begin{aligned} GROUND(x, y, t, i) \quad :- \quad & INCOMING(y, t, u), \\ & FILTER(x, y, r), \\ & i = u(1 - r). \end{aligned}$$

The above query could be also expressed in SQL style or relational algebra. Whatever language is used, it is clear that the evaluation of the above query requires a join of the *INCOMING* and *FILTER* relations. Unfortunately, join operations are difficult to express in most GIS systems, including the ARCINFO/ARCVIEW system. However, join processing is very natural in constraint database systems (Revesz 2002). Moreover, only two linear constraint relations need to be joined in the query. That can be easily done in several linear constraint database systems, for example, CCUBE (Brodsky, Segal, Chen, and Exarkhopoulo 1997), DEDALE (Grumbach, Rigaux, and Segoufin 1998), and MLPQ/PReSTO (Revesz 2002).

In Section 2, we describe a simple area-based interpolation approach, which is a 2D case of shape function interpolations. By this approach interpolations can be represented using simple linear constraint relations. Other approaches, such as TIN-based interpolations (Revesz 2002, Chen and Revesz 2000, Grumbach, Rigaux, and Segoufin 2000) are mathematically equivalent but more complex in derivation, or, such as inverse distance weighting and Kriging, require polynomial constraint relations.

In Section 3, we describe the application of the area-based interpolation approach to the example in this section. We also present a brief discussion of the advantages and limitations of our approach to interpolation representation and querying.

## 2. Area-Based Interpolation in Constraint Databases

The approach for linear interpolation in constraint databases is based on Delaunay triangulation (Shewchuk 1996, Goodman and O'Rourke 1997) of the sample points. After that, the value of any point $(x, y)$ can be interpolated by considering only the triangle in which $(x, y)$ is located. Such triangulations are important in Finite Element discretizations of engineering problems, where interpolations with *shape functions* are also popular (Zienkiewics and Taylor 2000, Buchanan 1995). The area-based interpolation described below is a special case of two dimensional shape functions with three nodes.

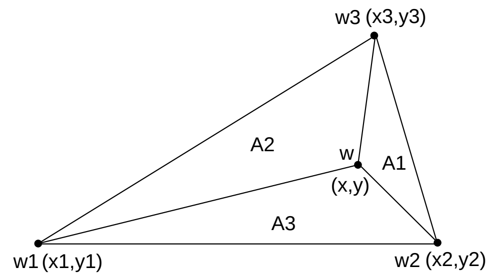


Figure 2: Computing shape functions by area divisions.

Figure 2 shows a triangle with corner vertices $(x_1, y_1)$, $(x_2, y_2)$, and $(x_3, y_3)$. Let us assume that $w_1, w_2$ and $w_3$ are the known values of these corner vertices and all the coordinate values are also known. Suppose also that we need to interpolate the value of a point $(x, y)$, where $x$ and $y$ are also given constants. Then we have:

$$w = \frac{\mathcal{A}_1}{\mathcal{A}} A w_1 + \frac{\mathcal{A}_2}{\mathcal{A}} w_2 + \frac{\mathcal{A}_3}{\mathcal{A}} w_3 \tag{1}$$

Clearly, the area of the Delaunay triangle in Figure 2 can be represented by a conjunction $C$ of three linear inequalities corresponding to the three sides of the triangle. Then, by Equation (1) the value $w$ of any point $x, y$ inside the triangle can be represented by the linear constraint tuple:

$$\begin{aligned} R(x, y, w) \quad :- C, \quad w = \quad & [((y_2 - y_3)w_1 + (y_3 - y_1)w_2 + (y_1 - y_2)w_3)/(2\mathcal{A})]x + \\ & [((x_3 - x_2)w_1 + (x_1 - x_3)w_2 + (x_2 - x_1)w_3)/(2\mathcal{A})]y + \\ & [((x_2 y_3 - x_3 y_2)w_1 + (x_3 y_1 - x_1 y_3)w_2 + (x_1 y_2 - x_2 y_1)w_3)/(2\mathcal{A})]. \end{aligned}$$

where $\mathcal{A}$ is the constant value of the triangle area. By representing the interpolation in each triangle by a separate constraint tuple, we can find in linear time a constraint relation to represent the whole interpolation.

**Example 2.1** *Let* $w_1 = 1$, $w_2 = 2$, *and* $w_3 = 3$ *be the values measured at locations* $(x_1, y_1) = (0, 0)$, $(x_2, y_2) = (10, 0)$, *and* $(x_3, y_3) = (10, 5)$, *respectively. Using Equation (1) we can interpolate the unknown value* $w$ *at location* $(x, y) = (8, 2)$ *as follows:*

$$w = \frac{1}{5} \times 1 + \frac{2}{5} \times 2 + \frac{2}{5} \times 3 = 2.2$$

## 3. Application

Let us return now to the example in Section 1. Figures 3 shows the Delaunay triangulations for the sample points in $Incoming(y, t, u)$ and $Filter(x, y, r)$. Next, we find as shown in Tables 3 and 4 the constraint representation for *INCOMING* and *FILTER* using the technique in Section 2.

One of the limitations of the area-based interpolation is that it may not be as accurate as some more complex approximation methods. On the other hand this yields fast calculations. It is also eaier to scale the constraint data as required (for example to change from feet to meters), because it just requires multiplication by a constant factor. This scaling problem is much more difficult to handle in many GIS systems. Also, the constraint relation representation needs less storage space.

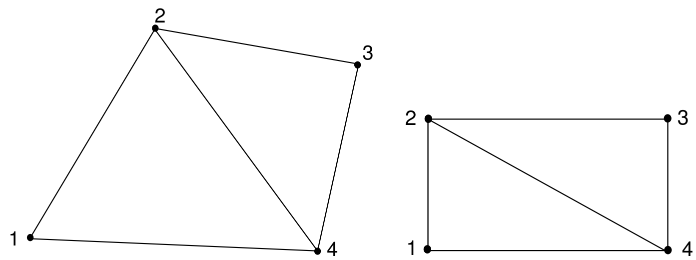


Figure 3: Delaunay triangulations for the two input sets.

| Y | T | U | |
|---|---|---|---|
| y | t | u | $21x - 13y + 13 \geq 0$, $x + 29y - 29 \leq 0$, $11x + 8y - 319 \geq 0$, $u = 0.69x - 1.45y + 62.33$ |
| y | t | u | $11x + 8y - 319 \leq 0$, $x + 5y - 97 \leq 0$, $9x - 2y - 261 \leq 0$, $u = 2.71x - 1.06y + 79.95$ |

Table 3: $INCOMING(y, t, u)$.

## References

A. Brodsky, V. Segal, J. Chen, and P. Exarkhopoulo. The CCUBE constraint object-oriented database system. *Constraints*, 2(3–4):245–77, 1997.

G. R. Buchanan. *Finite Element Analysis*. McGraw-Hill, New York, 1995.

R. Chen and P. Revesz. Geo-temporal data transformation and visualization. In *The First International Conference on Geographic Information Science*, pages 240–242, Savannah, Georgia, USA, 2000.

J. E. Goodman and J. O'Rourke, editors. *Handbook of Discrete and Computational Geometry*. CRC Press, Boca Raton, New York, 1997.

S. Grumbach, P. Rigaux, and L. Segoufin. Spatio-temporal data handling with constraints. In *ACM Symposium on Geographic Information Systems*, pages 106–11, 1998.

S. Grumbach, P. Rigaux, and L. Segoufin. Manipulating interpolated data is easier than you thought. In *Proc. IEEE International Conference on Very Large Databases*, pages 156–65, 2000.

P. Revesz. *Introduction to Constraint Databases*. Springer-Verlag, 2002.

J. R. Shewchuk. Triangle: Engineering a 2d quality mesh generator and delaunay triangulator. In *Proc. First Workshop on Applied Computational Geometry*, pages 124–133, May 1996.

O. C. Zienkiewics and R. L. Taylor. *Finite Element Method, Vol. 1, The Basis*. Butterworth Heinemann, London, 2000.

| X | Y | R | |
|---|---|---|---|
| x | y | r | $13x - 23y + 296 \geq 0$, $x \geq 2$, $y \geq 1$, $r = 0.0004x - 0.0031y + 0.1168$ |
| x | y | r | $13x - 23y + 296 \leq 0$, $x \leq 25$, $y \leq 14$, $r = 0.0013x - 0.0038y + 0.1056$ |

Table 4: $FILTER(x, y, r)$.