A Comparison of Two Spatio-Temporal Interpolation Methods

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

1 Introduction

All phenomena in the real world depend on time, such as physical phenomena (vibrations, dynamics, acoustics), geographic and geological phenomena (temperature, precipitation, ocean tide movements), astronomical phenomena (evolution of celestial bodies), biology and chemical phenomena (cell growth & chain reactions, chemical reactions). Many daily life and industrial processes vary with time, such as accounting, banking, inventory control, and multimedia [8]. Many systems which collect, process and analyze specific information have to deal with time, such as Geographic Information Systems/Land Information Systems (GIS/LIS) [6], environmental information systems [7], medical information systems [16], natural resource information systems, scientific data analysis [15], and natural language processing systems [2]. The ability to represent and retrieve temporal information is essential.

However, the traditional *relational data model* has the following problems and limitations in handling information dependent on time.

1. The relational data model cannot answer queries on past states.

   The relational data model represents the dynamic real world as a snapshot at a particular point in time. In many applications, queries on the past states are necessary. For examples, in the area of medical information systems, retrieving a patient's medical history is particularly important. An instance of a relational database is its current contents. Updating the state of a database is performed using data-manipulation operations such as insertion, deletion, or replacement, taking effect as soon as it is committed. Therefore, in this process, past states of the databases and those of the real world are discarded and forgotten completely. It is evident that queries on past states are not possible to answer in the relational data model.

2. The relational data model is very ineffective in representing temporal intervals and implementing temporal queries over intervals.

   For example, in order to consider queries such as "Which Ph.D. students studies in the computer science department at UNL last year", we need to represent the time intervals and implement the temporal *overlap* query. However, the relational data model neither provide the way to represent time intervals, nor the basic temporal queries such as overlap. Besides overlap, there are many other possible relationships between time intervals. In reference [1], Allen introduced thirteen possible ways in which an ordered pair of intervals can be related, such as *before*, *equal*, *meets*, *overlaps*, *during*, *starts*, and *finishes*. But none of them can be evaluated efficiently and conveniently by the relational data model.

3. The relational data model is inconvenient for querying discretely recorded continuous data [10], and has limited storage for them.

   In reality, continuous temporal data are always recorded at discrete times, for instance, time series and geographic time series data. In reference [10], such data are called discretely recorded continuous data. If we choose relational data model to manage these data, every single record will be stored in the database. Sometimes, the data sets can be huge, so sooner or later the storage of the relational databases might be used up. For querying, the relational data model will have problems to retrieve the information that is not recorded explicitly in the database.

Fortunately, these problems can be solved by using *constraint databases* [11] and related techniques. More specifically, the first problem is not a problem in constraint databases since all the temporal activities are recorded as functions and the queries on past states (even future states) can be easily computed by substituting the right time value into the functions; the second problem can be encountered by applying some linear inequality or equality constraints on time in the query languages such as *Datalog*; the third problem can be solved by applying piecewise linear approximation method in constraint databases, as proposed in references [10, 4].

But piecewise linear approximation method itself can only interpolate time series data. It cannot deal with the interpolation problem of spatial data that are dependent on time. For example, it cannot solve the following interpolation problem. Suppose that we have the following set of house price data in our input database:

> *House_known(x,y,t,p)* records the price per square foot p of the house at location (x, y) which was sold at time t, where the unit of time is $month/year$.

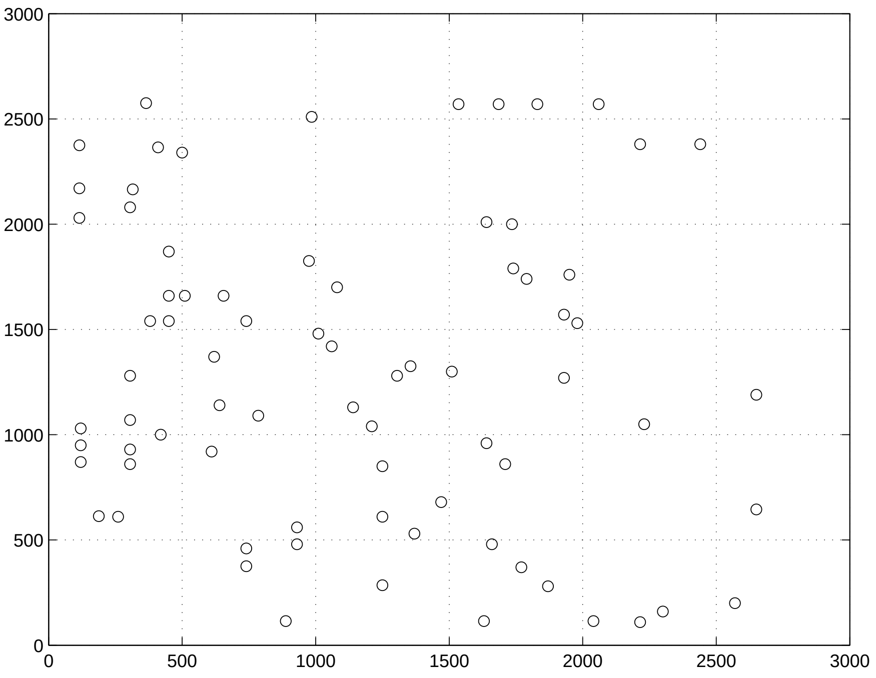

Figure 1: The sample points in House_known.

Figures 1 illustrates the locations of the 76 houses being sold when the prices per square foot p were recorded. Table 1 shows an instance of this *House_known* relation. Based on the fact that the earliest selling transaction of the sampled houses is in 1990, we encode the time in such a way that 1 represents January 1990, 2 represents February 1990, $\ldots$, 148 represents April 2002. Note that some houses are sold more than once in the past, so they have more than one tuples in the relation. For example, the house at the location $(115, 2030)$ has been sold three time in history at time 28, 112, and 137 (which represent 4/1992, 4/1999, and 5/2001). Assume we have another set of 50 houses which we have no history price record. Figures 2 illustrates the locations of these 50 houses. Table 2 shows an instance of this *House_unknown* relation. If we want to estimate their prices at a certain time, some spatio-temporal interpolation method should be applied.

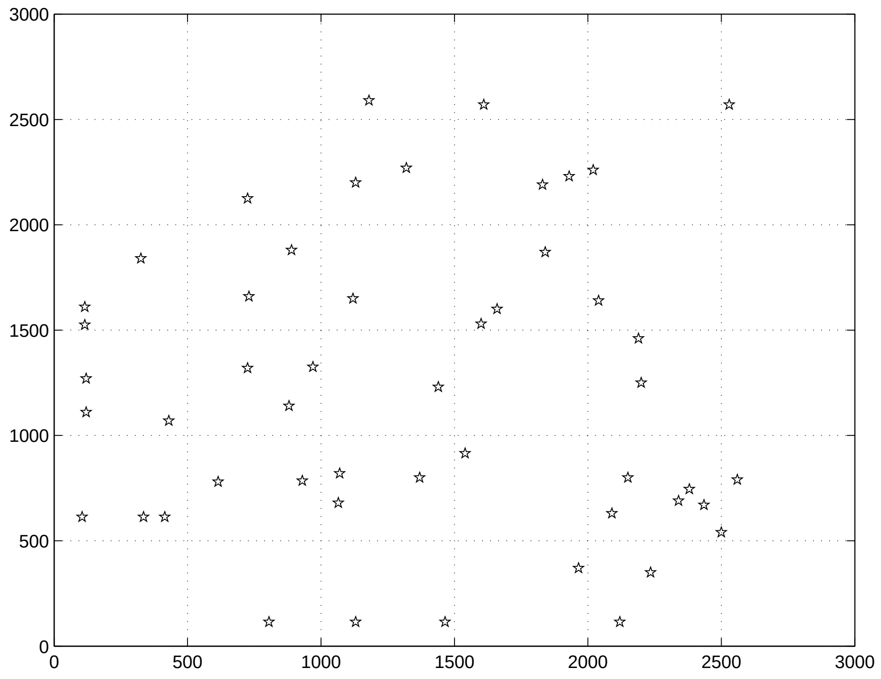

Figure 2: The sample points in House_unknown.

In this paper, by borrowing the idea of *shape functions* from Finite Element Analysis, we discuss and compare two alternative methods in constraint databases to interpolate spatial data that are changing with time, which are similar to the above house price data.

The rest of this paper is organized as follows. Section 2 discusses the ST (space-time) product interpolation method that treats time as a special dimension. Section 3 discusses an alternative tetrahedral method that deals time as a regular third dimension. Section 4 describes the application of the two interpolation methods in constraint databases to the example in this section. Finally, in Section 5, we present a brief discussion of the advantages and limitations of our two approaches to spatio-temporal interpolation representation and querying.

X	Y	T	P (the price per square foot)
115	2030	28	71.37
115	2030	112	100.31
115	2030	137	98.765
115	2170	65	84.88
115	2170	99	92.59
500	2340	36	68.58
500	2340	55	69.03
⋮	⋮	⋮	⋮
260	610	36	62.64

Table 1: $House\_known(x, y, t, p)$.

X	Y	T
115	1525	16
115	1525	58
115	1525	81
115	1610	63
115	1610	119
890	1880	36
890	1880	75
⋮	⋮	⋮
615	780	59

Table 2: $House\_unknown(x, y, t)$.

2 ST Product Method

This method treats *time* as a special dimension. This approach can be described by the following four steps: triangular meshing, linear approximation in space, linear approximation in time, and multiplication space and time.

2.1 Triangular Meshes

When dealing with complex geometric domains, it is convenient to divide the total domain into a finite number of simple sub-domains which can have triangular or quadrilateral shapes in the case of 2-D problems. For complicated domains, irregular unstructured meshes with triangular or quadrilateral sub-domains are needed. Mesh generation using triangular or quadrilateral domains is important in Finite Element discretization of engineering problems. For the generation of triangular meshes, quite successful algorithms have been developed. A popular method for the generation of triangular meshes is the "Delaunay Triangulation" [13]. Delaunay triangulation is related to the construction of the so called "Voronoi diagram", which is related to "Convex Hull" problems [9].

2.2 Linear 2-D Approximation in Space

A linear 2-D approximation function for a triangular area can be written in terms of three shape functions N_1, N_2, N_3, and the corner values w_1, w_2, w_3. Shape functions are popular in engineering applications, for example, in Finite Element algorithms [18, 3].

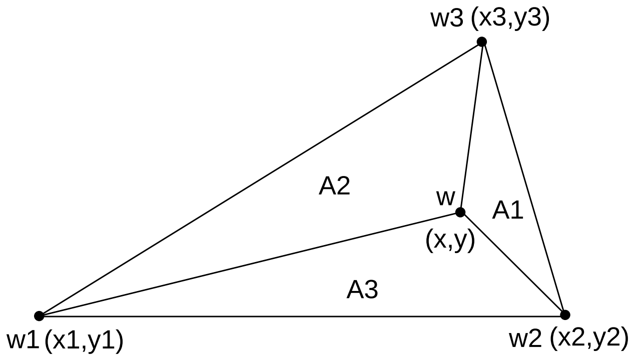

Figure 3: Computing shape functions by area divisions.

Figure 3 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, $\mathcal{A}_1$, $\mathcal{A}_2$ and $\mathcal{A}_3$ are the three sub-triangle areas, and $\mathcal{A}$ is the value of the big outside triangle area. Suppose also that we need to interpolate the value of a point (x, y). Then we have:

$$w(x,y) \quad = \quad N_1(x,y)w_1 + N_2(x,y)w_2 + N_3(x,y)w_3 \tag{1}$$

where N_1, N_2 and N_3 are the following shape functions:

$$N_1(x,y) = \frac{\mathcal{A}_1}{\mathcal{A}}, \quad N_2(x,y) = \frac{\mathcal{A}_2}{\mathcal{A}}, \quad N_3(x,y) = \frac{\mathcal{A}_3}{\mathcal{A}} \ . \tag{2}$$

Clearly, the area of the Delaunay triangle in Figure 3 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 [12]:

$$\begin{aligned} R(x,y,w) \ :- \ w = \ & [((y_2 - y_3)w_1 + (y_3 - y_1)w_2 + (y_1 - y_2)w_3)/(2\mathcal{A})] \ \ x \ \ + \\ & [((x_3 - x_2)w_1 + (x_1 - x_3)w_2 + (x_2 - x_1)w_3)/(2\mathcal{A})] \ y \ \ + \end{aligned} \tag{3}$$

$$[((x_2y_3 - x_3y_2)w_1 + (x_3y_1 - x_1y_3)w_2 + (x_1y_2 - x_2y_1)w_3)/(2\mathcal{A})],$$
$$C.$$

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 2-D spatial interpolation.

2.3 Linear Approximation in Time

Assume the value at the node i at time t_1 is w_{i1}, and at time t_2 the value is w_{i2}. The value at the node i at any time between t_1 and t_2 can be approximated using time shape functions in the following way:

$$w_i(t) = \frac{t_2 - t}{t_2 - t_1} w_{i1} + \frac{t - t_1}{t_2 - t_1} w_{i2} \ . \tag{4}$$

2.4 Multiplication of Space and Time

By multiplying formulas 1 and 4, the linear approximation function for any point constraint to the sub-domain I at any time between t_1 and t_2 can be expressed as follows:

$$\begin{aligned} w(x,y,t) &= N_1(x,y)\left[\frac{t_2 - t}{t_2 - t_1} w_{11} + \frac{t - t_1}{t_2 - t_1} w_{12}\right] \\ &\quad + N_2(x,y)\left[\frac{t_2 - t}{t_2 - t_1} w_{21} + \frac{t - t_1}{t_2 - t_1} w_{22}\right] \\ &\quad + N_3(x,y)\left[\frac{t_2 - t}{t_2 - t_1} w_{31} + \frac{t - t_1}{t_2 - t_1} w_{32}\right] \\ &= \frac{t_2 - t}{t_2 - t_1}\left[N_1(x,y)w_{11} + N_2(x,y)w_{21} + N_3(x,y)w_{31}\right] \\ &\quad + \frac{t - t_1}{t_2 - t_1}\left[N_1(x,y)w_{12} + N_2(x,y)w_{22} + N_3(x,y)w_{32}\right] \ . \end{aligned} \tag{5}$$

Note that since the interpolation is obtained by multiplying two linear interpolation functions in space and time, the resulting constraint relation is no longer linear.

This ST product method is based on the assumption that the sample data are measured at the same time instance at all the locations. For the cases that the data are not measured at different times, we can easily extend formulas 4 and 5 by piecewise linear approximation [10].

3 Tetrahedral Method

This method treats *time* as a regular third dimension. Therefore, the spatio-temporal interpolation problem is transferred to a three-dimensional (two dimensions in space and one dimension in time) interpolation problem.

3.1 Tetrahedral Meshes

Three-dimensional domains can be divided into finite number of simple sub-domains. For example, we can use tetrahedral or hexahedral sub-domains. Tetrahedral meshing is of particular interest. With a large number of tetrahedral elements, we can also approximate complicated 3-D objects. Figure 4 shows a tetrahedral mesh of a 3-D object. This object has a cutout (one quarter of a cylinder) behind the boundary defined by the points $ABCD$.

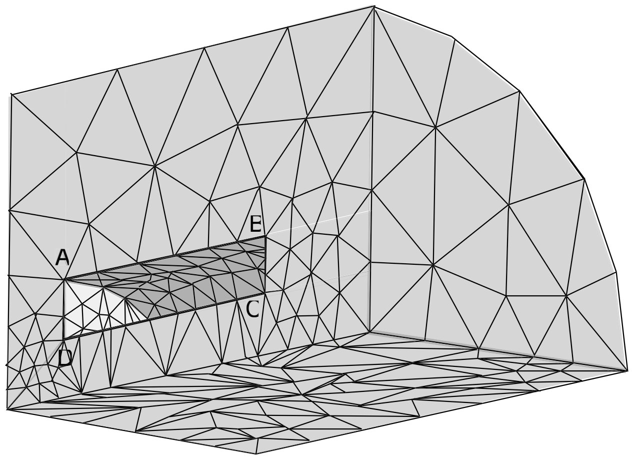

Figure 4: A Tetrahedral Mesh.

It is a difficult topic to automatically generate tetrahedral meshes. Fortunately, there exist several algorithms to generate automatic tetrahedral meshes, such as tetrahedral mesh generation by Delaunay refinement [14] and tetrahedral mesh improvement using swapping and smoothing [5]. In this paper, the software package *Matlab* is used to generate tetrahedra meshes based on 3-D Delaunay tessellation.

3.2 Linear 3-D Approximation

Similarly to the linear approximation function for the 2-D problem solved in Section 2.2, a linear approximation function for a 3-D tetrahedral element can be written in terms of four shape functions N_1, N_2, N_3, N_4 and the corner values w_1, w_2, w_3, w_4.

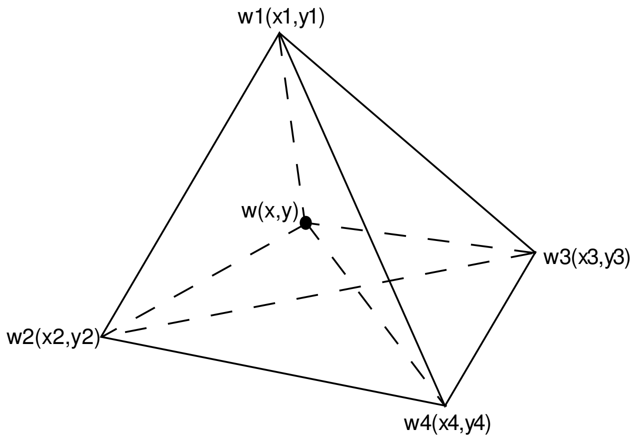

Figure 5: Computing shape functions by volume divisions.

Figure 5 shows a tetrahedron with corner vertices (x_1, y_1, t_1), (x_2, y_2, t_2), (x_3, y_3, t_3), and (x_4, y_4, t_4). Let us assume that w_1, w_2, w_3, and w_4 are the known values of these corner vertices, $\mathcal{V}_1$, $\mathcal{V}_2$, $\mathcal{V}_3$ and $\mathcal{V}_4$ are the four volume values of sub-tetrahedra $ww_2w_3w_4$, $w_1ww_3w_4$, $w_1w_2ww_4$, and $w_1w_2w_3w$, respectively. $\mathcal{V}$ is the volume value of the big outside tetrahedron $w_1w_2w_3w_4$. Suppose also that we need to interpolate the value of a point (x, y, t). Then we have:

$$w(x,y,t) = N_1(x,y,t)w_1 + N_2(x,y,t)w_2 + N_3(x,y,t)w_3 + N_4(x,y,t)w_4 \tag{6}$$

where N_1, N_2, N_3 and N_4 are the following shape functions:

$$N_1(x,y,t) = \frac{\mathcal{V}_1}{\mathcal{V}}, \quad N_2(x,y,t) = \frac{\mathcal{V}_2}{\mathcal{V}}, \quad N_3(x,y,t) = \frac{\mathcal{V}_3}{\mathcal{V}}, \quad N_4(x,y,t) = \frac{\mathcal{V}_4}{\mathcal{V}} \ . \tag{7}$$

Clearly, the volume of the tetrahedron in Figure 5 can be represented by a conjunction C of four linear inequalities corresponding to the four facets of the tetrahedron. Then, by

Equation (6) the value w of any point (x, y, t) inside the tetrahedron can be represented by the linear constraint tuple:

$$\begin{aligned} R(x,y,t,w) \quad :- \ w = \quad & [(b_1w_1 + b_2w_2 + b_3w_3 + b_4w_4)/(6\mathcal{V})] \ x \quad + \\ & [(c_1w_1 + c_2w_2 + c_3w_3 + c_4w_4)/(6\mathcal{V})] \ y \quad + \\ & [(d_1w_1 + d_2w_2 + d_3w_3 + d_4w_4)/(6\mathcal{V})] \ t \quad + \\ & [(a_1w_1 + a_2w_2 + a_3w_3 + a_4w_4)/(6\mathcal{V})] \ , \\ & C. \end{aligned} \tag{8}$$

where a_1, b_1, c_1, d_1, a_2, b_2, c_2, d_2, a_3, b_3, c_3, d_3, a_4, b_4, c_4, and d_4 are all constants. More specifically, we have

$$\begin{aligned} a_1 &= det \begin{bmatrix} x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \\ x_4 & y_4 & z_4 \end{bmatrix} \\ b_1 &= -det \begin{bmatrix} 1 & y_2 & z_2 \\ 1 & y_3 & z_3 \\ 1 & y_4 & z_4 \end{bmatrix} \\ c_1 &= -det \begin{bmatrix} x_2 & 1 & z_2 \\ x_3 & 1 & z_3 \\ x_4 & 1 & z_4 \end{bmatrix} \\ d_1 &= -det \begin{bmatrix} x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \\ x_4 & y_4 & 1 \end{bmatrix} \end{aligned} \tag{9}$$

The other constants can be obtained by cyclic interchange of the subscripts in the order 4, 1, 2, 3 [17].

By representing the interpolation in each tetrahedron by a separate constraint tuple, we can find in linear time a constraint relation to represent the whole interpolation.

Note that since the interpolation is obtained by adding time as a third dimension, the resulting constraint relation is still linear.

4 Application

Let us return now to the example in Section 1. There are 76 houses with history prices, while 50 houses with unknown prices. We want to interpolate the prices of the 50 houses at certain times. Figure 6 shows the 76 houses with circles and the 50 house with pentagons.

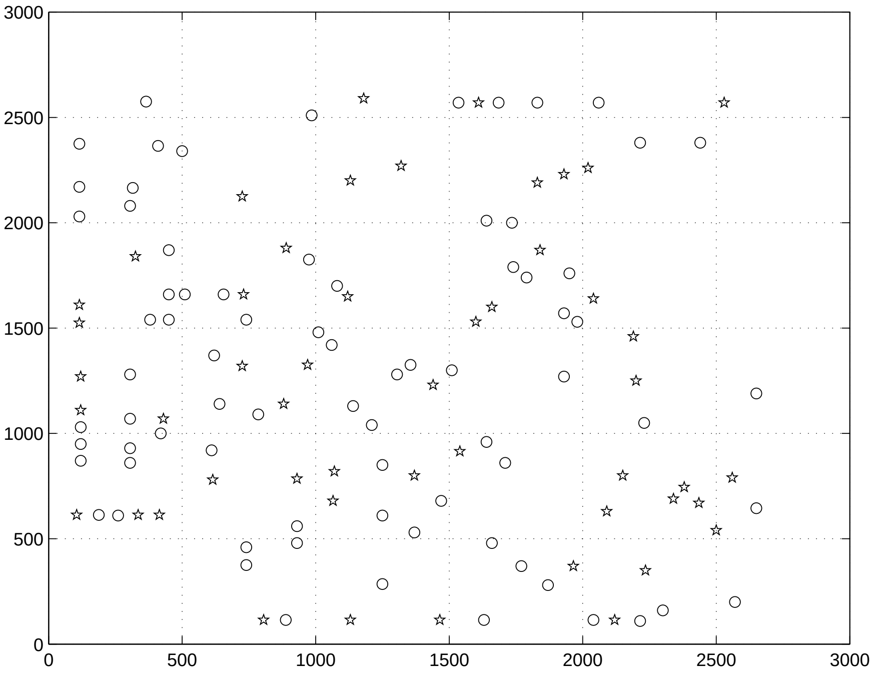

Figure 6: 76 houses with known history prices and 50 houses with unknown prices.

Experiments have been conducted to analyze and compare the qualities of ST product method and tetrahedral method according to Mean Absolute Error (MAE) and Root Mean Square Error (RMSE). Actually, the 50 unknown price houses have the true history prices recorded so that we can compare them with the experimental results.

Table 3 summarizes the quality analysis of the two methods. It shows that tetrahedral method is better for this house price testing case, since both MAE and RMSE error measures

	MAE	RMSE
ST Product Method	10.109	11.338
Tetrahedral Method	7.917	8.983

Table 3: Comparison result.

of tetrahedral method are less than ST product method.

Let *HOUSE(x,y,t,p)* constraint relations that represent the interpolation of *House_known (x,y,t,u)* input relation. *HOUSE (x,y,t,p)* can be deemed as an infinite relation of triples of rational numbers. To write queries, we do not need to know how the constraints are used in the representation of the infinite relations. Assume there is another input relation *Built (x,y,t)*, which record the time (in month) when the house at location (x, y) was first built.

Consider the following queries:

Query 4.1 *For each house, find the starting sale price when the house was built.*

The above query can be expressed in Datalog as follows:

$$\begin{aligned} Start\_price(x,y,p) \quad &:- \quad Built(x,y,t), \\ &\qquad HOUSE(x,y,t,p) \ . \end{aligned}$$

Query 4.2 *Suppose that we know house prices in general decline for some time after the first sale. For each house, find the first month when it become profitable, that is, the first month when its price exceeded its initial sale price.*

The above query can be expressed in Datalog as follows:

$$\begin{aligned} not\_Profitable(x,y,t) \quad &:- \quad Built(x,y,t) \ . \\ not\_Profitable(x,y,t_2) \quad &:- \quad not\_Profitable(x,y,t_1), \\ &\qquad HOUSE(x,y,t_2,p_2), \\ &\qquad Start\_price(x,y,p), \\ &\qquad t_2 = t_1 + 1, \ p_2 \le p. \end{aligned}$$

$$\begin{aligned} Profitable(x, y, t_2) \;\; :- \;\; & not\_Profitable(x, y, t_1), \\ & HOUSE(x, y, t_2, p_2), \\ & Start\_price(x, y, p), \\ & t_2 = t_1 + 1, \;\; p_2 > p. \end{aligned}$$

Query 4.3 *How many months did it take for each house to become profitable?*

The above query can be expressed in Datalog as follows:

$$\begin{aligned} Time\_to\_Profit(x, y, t_3) \;\; :- \;\; & Built(x, y, t_1), \\ & Profitable(x, y, t_2), \\ & t_3 = t_2 - t_1. \end{aligned}$$

5 Concluding Remarks

This paper analyzes two spatio-temporal interpolation methods (i) ST product method and (ii) tetrahedral method in constraint databases. For the house price estimation input data example, the experimental results show that the tetrahedral method is better than ST product method. It is because the house price data are selected in such a way that they are dense in space (in a close neighborhood) but sparse in time (houses are sold not frequently).

Tetrahedral method is linear, which can be implemented in MLPQ system [11]. Spatio-temporal queries in constraint databases can be easily and efficiently evaluated in MLPQ, which are difficult or impossible to implement in conventional relational databases.

However, for the input data that do not satisfy the property that they are dense in space and sparse in time, the tetrahedral method may not be better than ST product method. For example, for the temperature data that are measured regularly in various locations in a area, the ST product method may have better result since the temperature of location A at time t may have influence to interpolate the temperature of neighboring location B only at the same time t.

References

[1] J. F. Allen: Maintaining knowledge about temporal intervals. *Communications of ACM*, vol. 26, No. 11, pp. 832–843, November 1983.

[2] I. Androutsopoulos, G. Ritchie, and P. Thanisch: Time, tense and aspect in natural language database interfaces. *Journal of Natural Language Engineering*, vol. 4, No. 3, pp. 211–228, November 1998.

[3] G. R. Buchanan: *Finite Element Analysis*, McGraw-Hill, New York, 1995.

[4] R. Chen, M. Ouyang, P. Z. Revesz: Approximating Data in Constraint Databases. *Proc. Symposium on Abstract, Reformation, and Approximation 2000*, Texas, pp. 124–143, July 2000.

[5] L. A. Freitag and C. O. Gooch: Tetrahedral mesh improvement using swapping and smoothing. *International Journal for Numerical Methods in Engineering*, vol. 40, pp. 3979–4002, 1997.

[6] G. Langran: A review of temporal datbase research and its use in GIS applications. *International Journal of Geographical Information Systems*, Vol. 3, No. 3, pp. 215–232, 1989.

[7] J. Moreira, C. Ribeiro, and J. Saglio: Representation and Manipulation of Moving Points: An Extended Data Model for Location Estimation. *Cartography and Geographic Information Systems (CaGIS) - Special Issue on Dealing with Time*, Vol. 26, No. 2, pp. 109–123, April 1999.

[8] M. Nabil, A. Ngu, and J. Shepherd: Modelling Spatio-temporal Relationships in Multimedia Databases. *Proceedings of IEEE workshop on Spatial and Temporal Interaction: Representation and Reasoning*, Singapore, 1995.

[9] F. P. Preparata and M. I. Shamos: *Computational Geometry: An Introduction.* Springer-Verlag, 1988.

[10] P. Z. Revesz, R. Chen, M. Ouyang: Approximate Query Evaluation Using Linear Constraint Databases. *Proc. Symposium on Temporal Representation and Reasoning*, Cividale del Friuli, Italy, July 2001.

[11] P. Revesz: *Introduction to Constraint Databases*, Springer-Verlag, 2002.

[12] P. Z. Revesz, L. Li: Representation and Querying of Interpolation Data in Constraint Databases. *Proc. National Conference for Digital Government Research*, accepted, Los Angeles, CA, May, 2002.

[13] J. R. Shewchuk: Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator. *Proc. First Workshop on Applied Computational Geometry*, 1996.

[14] J. R. Shewchuk: Triangle: Tetrahedral mesh generation by Delaunay refinement. *Proc. 14th Annual ACM Symposium on Computational Geometry* ACM Press, pp. 86–95, 1998.

[15] R. Snodgrass: Temporal and Real-Time Databases: A Survey. *IEEE Transactions on Knowledge and Data Engineering*, vol. 7, No. 4, pp. 513–532, August 1995.

[16] G. Wiederhold, J. F. Fries, and S. Weyl: Structured organization of clinical data bases. *Proceedings of the National Computer Conference*, vol. 44, Anaheim, CA (May19-22): AFIPS Press, Reston, VA, pp. 479–485, 1975.

[17] O. C. Zienkiewics and R. L. Taylor: *Finite Element Method, Vol. 1, The Basic Formulation and Linear Problems.* McGraw-Hill, 1989.

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