CONSTRAINT BASED MODELING IN A GIS: ROAD DESIGN AS A CASE STUDY[1]

Andrew U. Frank
Dept. for Geo-Information E127.1
Technical University Vienna
Gusshausstr. 27-29
A-1040 Vienna Austria
FRANK@GEOINFO.TUWIEN.AC.AT

Mark Wallace
Imperial College
William Penney Laboratory
London SW7 2AZ, U.K.
MGW@DOC.IC AC.UK

ABSTRACT

Modeling in GIS is limited to the standard geometric data models: vector and raster. Not all problems can be expressed in these models and extensions are requested by applications. To determine precise requirements for extensions, case studies are beneficial. In this paper the focus is on the expressive power required for design applications. A significant part of the road design task is used as a case study to explore if constraint databases can contribute to the solution. Road design is a suitable example for GIS design applications, as road design in the past used topographic maps and map analysis methods.

The general design task for the layout of a highway is presented: Find the technically feasible alternative road layouts between two points. Select the best for further assessment. Given are the design parameters of the road (design speed, minimal radius of curves, maximum slope) and information about the terrain (land cover, elevation, geology).

The problem is sufficiently complex to pose substantial research questions, but simple enough to be tractable. Three observations result from this case study:

- Constraint databases can be used to model continuos variables, and therefore space (not only discrete points in space),
- Representing space with a Delaunay triangulation leads to conceptual simplifications,
- The original constraint of the road design problem, which are of a higher degree, can be linearized to make the problem computationally tractable.

A test implementation is underway, exploring the performance aspects of the use of constraint databases for road design.

1. INTRODUCTION

Modeling in GIS is more limited then what is most often perceived. GIS users are extremely able to select only problems which they can solve with the functionality of today's GIS and other problems are not considered. In the past years, more and more functions have been added to the commercial GIS. The limitations imposed by the models employed, however, remain. These are limitations of 'expressive power' - what kind of problems can be described in the language provided by the GIS (Gallaire, Minker, and Nicolas 1984).

The major limitations arise from the standard geometric data models: the vector and raster model (Frank 1990, Goodchild 1990, Herring 1990). Efforts to combine the two

[1] Funding from Intergraph Corporation and the Austrian Fund for Scientific Research (FWF) is gratefully acknowledged. The case study is a part of an effort supported by the European Community (ESPRIT program) to develop constraint databases (project CONTESSA) Stimulation to consider the problem and additional support for the work was received from the Bundesministerium fuer wirtschaftliche Angelegenheiten (Strassenbau-Sektion)

models (Peuquet 1983) do not overcome all limitations. They combine only the power of the two models, but do not provide what is not included in one or the other. Using constraints is a novel, promising concept, which allows to model areas directly with a set of constraints: the area is the infinite point set which simultaneously fulfill all constraints.

Recently, efforts focused to overcome the requirement that all objects in a GIS must be well defined and have clearly defined boundaries (Burrough and Frank 1994). Many objects important for GIS applications have broad transition zones. Fuzzy logic provides but one model and different application areas use different concepts (an overview gives (Burrough and Frank 1995), for the use of Fuzzy Logic in a GIS system (Eastman 1993, Eastman *et al.* 1993)).

Another limitation is the requirement that locations in the GIS are expressed as coordinate in a single fixed system. Information which is not expressible as properties of determined locations cannot be entered. In emergency situations, partial information is often received. For example, an informant may report: "X was seen on the left bank of the river, between A and B". Several such pieces of intelligence constraints the solution and together they may be sufficient to direct a search action.

In design problems, information is often expressed as constraints. The solution is described in general terms and the task is to find a particular solution which fulfills the general description. For example in road design, the location and general form of the road are restricted by constraints, found in national design standards and technical guidelines of agencies. The designer's art is to find a geometric layout which respects all the restrictions and is optimal as measured on some scale (typically least cost is used). Traditionally, the designer used topographic maps and map analysis methods to determine optimal road layout. Today, GIS seems the appropriate tool.

Road design is a complex, multi phase effort. Here, only one phase of a general layout is considered, namely the production of a small number of major alternative road layouts. These layouts must fulfill the technical constraints. They should be ordered using some simple cost estimates to direct further refinement and assessment to the most promising ones. No potential solution must be left out, as the following process is only selecting among the candidates and does not produce new alternatives. The following assessment and decision process are not considered here.

The case study is undertaken to investigate a typical spatial engineering problem, which is well understood, but not immediately solvable with current GIS. A method sometimes proposed uses raster data model and assesses the cost of building a road across each raster cell as a function of current land use. This technique does not capture the geometric restriction of the road, i.e. minimal radius of curves, slopes etc., and yields results which are very sensitive to the selection of the cost parameters.

The problem can be formalized and expressed with a combination of tools from current spatial data models and constraints programming. Constraint programming is a set of techniques and languages especially suited to deal with design problems (Borning, Freeman-Benson, and Willson 1994). Constraint programming has been combined with database techniques to construct *Constraint Databases* (Kanellakis and Goldin 1994). It builds on the current research in the database community, which extends databases with constraints to provide generic tools to address such problems.

Constraints can express areas as a continuum and, therefore, have been used extensively to solve spatial problems (Borning 1981, Borning 1986, Borning *et al.* 1987) and for a general review (Borning, Freeman-Benson, and Willson 1994). The constraints on the road layout from technical restrictions as stated in design standards are used directly as constraints. They include minimal curve radius, maximal slope, and the regular constraints for smoothness (tangent, transition curvess between straight segments and curves). Terrain (land use, topography and geology) is modeled as a continuos function, interpolated from e.g. raster data. Constraints keep the road outside of prohibited areas, while all acceptable areas have an estimate unit cost for constructing a road there.

The structure of the paper is as follows: The next section details the problem and gives examples for typical constraints. Section 3 describes constraint programming and constraint databases. Section 4 develops the solution and gives the reformulated

constraints. The paper concludes with an assessment of the method and the limitations of today's systems.

2. DESCRIPTION OF PROBLEM

Assume that a road should be designed to run from A-town to B-city avoiding a number of restricted areas (figure 1). In this section, the constraints for the road layout are explained and the data describing the particular geographic situation are detailed. The goal is to identify all relevant information for the task and reformulate the problem in these terms.

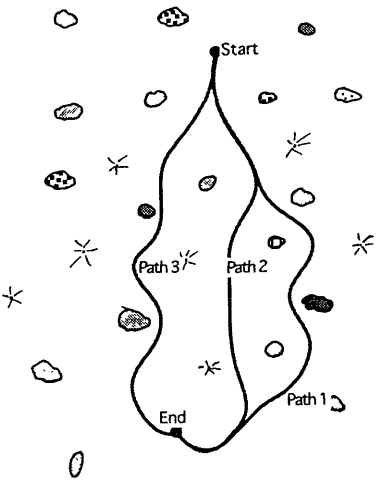

Figure 1 - The situation with three alternative paths

THE CONSTRAINTS FOR THE ROAD

The road designer is given the task to construct a road of a specific road type, e.g. a highway, between two points. This translates to a typical road cross section, with a set of descriptive attributes, namely number and width of lanes, number of breakdown lanes, etc. and the total width of the roadway.

The designer is also given a design speed. Design speed controls most geometric layout parameters, in particular minimal curve radius, maximal slope. Specific values for a highway designed for 100 km/h are listed in table 1 (following Swiss practice (Dietrich, Graf, and Rotach 1975)).

Table 1 Typical Geometric Constraints

Design speed	*100 km/h*
horizontal layout	
minimal length of elements	150 m
maximal length of straight elements	1700 m
minimal radius	425 m
minimal radius for curve connected to straight segment	4000 m
curved and straight segments: ratio of length	0.5 .. 2
sequence of two curves: ratio of radii	0.5 .. 2
clothoid parameter A	180 m
vertical layout	
minimal slope	1%
maximal slope	7%
minimal radius for slope change (for slope differences from 2% to 8%)	
valley:	5 500 .. 30 000 m
hill :	11 000 .. 60 000 m

Rules vary from country to country and the list just demonstrates the types of constraints found in design standards and provide a collection of realistic rules for the case study.

In general, horizontal and vertical layout are independent and can be designed separately. However, certain combinations of situations must be avoided, because they surprise drivers and can lead to accidents:

- no large changes of slopes on short distance (up-down-up-down roads)
- no curve starts behind a hill.

DESCRIPTION OF THE SITUATION

The start and end points are given as fixed locations (coordinate values) and the direction of the road in these points specified. Additional way points may be given.

Certain areas of the terrain are restricted ("forbidden areas"), for example built-up areas, lakes, natural reservation areas etc.. Cost of constructing the road over terrain varies: swamps, for example, cause foundation problems and add much cost. For all areas, cost estimates for building a road there is available ("unit cost").

In order to fit the road to the terrain and to calculate slopes, a digital elevation model gives the terrain height for any location. This can be interpolated from a regular raster or a TIN based Digital Elevation Model.

FORMALIZATION OF THE PROBLEM

The problem can be restated as follows:

Given:

- a digital terrain model, which permits determination of terrain height for any point, $h(x,y)$
- a road unit cost model, which gives the unit cost per road length for any point, $c(x, y)$
- a set of 'forbidden areas', which the road must not traverse,
- a set of way points, through which the road must pass, possibly with the direction the road must have (this set includes at least the start and end point)
- a set of design restrictions for the horizontal and vertical layout, which assure the smoothness of the road and the minimal radii.

Determine:

- substantially different alternative road layouts fulfilling the technical constraints,
- order the alternatives by cost estimates,
- list the best alternatives for further assessment.

The rank ordering must assure that the 'overall best' design is included in the alternatives provided. It is not acceptable that a promising alternative is excluded early, but it does not add much cost, if too many candidates are selected and are later discarded.

CURRENT ROAD DESIGN SOFTWARE FOR THE REFINEMENT

Road design software models the road as a sequence of road design elements. The horizontal and vertical layout is treated separately. For the horizontal layout the elements are straight lines, segments of circles and transition curves between different curves. For the vertical layout, only curves and straight lines are used.

The solution is found using a least squares adjustment technique to determine the unknown elements. Equations in the design parameters and the locations are set up and constraints added. These equations are linearized and solved under an additional optimality constraint using Lagrange multipliers (the sum of the squares of the deviations is made minimal).

This method is very effective to vary a large number of unknown by small amounts to find an optimal solution. The method fails for problems, where non-continuos jumps must be made to find an optimal situation. For example, these least squares based methods will not find an alternate path circumventing an obstacle (as shown as alternate path in figure 1)(Kuhn 1989)

APPROACH

An optimal sequence of actions is

1. determine possible substantially different alternatives as polygon of way points; select the most promising ones based on cost estimates.

2. Refine the candidates to smooth curves and refine cost estimates.

The first step must assess a large number of possible paths, most of which are clearly not respecting the geometric constraints or not close to optimal. Constraint databases are a promising tool to solve this task.

The second step refines the candidates identified in the first step using a least squares approach. Constraints databases could in principle solve this task, but the performance implications are not yet clear (see next section).

3. CONSTRAINT DATABASES

Standard databases allow the storage and retrieval of facts, e.g. the occurence of some species at certain locations, the form and location of an area. Query languages add deductive power, balancing the gain in deductive power against the additional complexity of the retrieval and inference process (Gallaire, Minker, and Nicolas 1984). A relational database can search quicker than a Prolog interpreter, but the expensive power of a relational query language is much less than the expressive power of Prolog. This is the standard trade-off between expressive power and performance. The current standard query languages lack the expressive power to adequately deal with spatial queries of the complexity typically found in a design process (Egenhofer 1992, Frank 1982) (Egenhofer 1991).

Constraint databases offer, within a single integrated database system, support for enhanced handling of data and knowledge. In a constraint database continuos data is directly supported. A single data item (a "constraint tuple") is a constraint on the tuple variables. A traditional data value is a special case of a constraint tuple of the form *Variable List = Value List* or*Var1=Val1 & Var2=Val2 & ...* An example is *<X,Y> = <10,5>* or equivalently *X=10 & Y=5*. A region is defined by similar constraints, except that the equality is replaced by an inequality, for example *5 =<X & X+Y =<15 & 3 =<Y*.

Such a simple modeling of continuos data makes the system very easy to extend and modify. Efficient computation of queries such as intersection is made possible by integrating constraint solvers for appropriate classes of constraints (Kanellakis, Kuper, and Revesz 1990, Kanellakis, and Goldin 1994) with novel forms of indexing for multi-dimensional data (Freestone 1993).

Current database query optimizers are designed for relatively simple queries requiring a predetermined number of "joins", "selections" and "projections". Query optimization centers around the order in which these operations should be carried out. By contrast the queries posed in the current application are much more complex. Existing query optimizers are not designed to handle such queries. Experiments with query optimizers (Bressan, Provost, and Wallace 1994) have revealed two limitations: the optimization process itself takes a very long time or fails to terminate, and the resulting query evaluation plan, if it is produced, is very inefficient to execute.

In conjunction with their very general data representation, constraint databases offer intelligent, dynamic query optimization, based on the optimizers developed for constraint programming systems. The uniformity of the data representation, as constraint tuples, simplifies the optimization problem since it is not necessary to treat each different kind of spatial object as another special case.

A standard constraint program comprises of two parts: a declarative part in which the constraints on the problem are stated; and a procedural part in which an algorithm is programmed for searching for (optimal) solutions which satisfy the constraints. As a result of making choices - or during the set-at-a-time evaluation of queries in a constraint database - new constraints are imposed on the problem variables.

Often these constraints are just value assignments $X=N$, where X is a problem variable and N a value; however, the added constraints can also be more complicated. The simplest way to apply constraints during search is as checks on precise solutions found by the search algorithm. In this case the constraints are delayed until each relevant variable has been assigned a value. In our application this means that each potential route plan for the road is generated, and then it is checked for feasibility. This treatment is, in general, too inefficient for practical applications. The key to constraint programming, and constraint databases, is that the constraints are applied actively to optimize the search for a feasible (optimal) solution.

Constraint programming systems support a number of specialized constraint solvers for handling particular classes of constraints. For handling linear rational constraints, as shown in the above example, a solver based on the Simplex method is built into the system. Thus the query

Find $<X,Y>$ such that $5 =< X \;\&\; X+Y =< 15 \;\&\; 3 =< Y$ and *region(X,Y)*

can be answered by the system directly, even if region is defined by a set of constraint tuples. The result of the query is, again, a set of constraint tuples, each representing the conjunction of the query condition and a constraint tuple from 'region'.

Constraint programming systems have also been built which incorporate a decision procedure for nonlinear constraints (Monfroy 1992), which would allow direct queries to be stated about the feasibility of roads with curves through regions involving curved boundaries. However, such constraint solvers are computationally very expensive, and are still not viable for real application of this kind.

Nevertheless complex constraints can be handled by "delaying" them, until they can be directly tested or easily solved. For example a constraint limiting the distance from a certain point can be expressed as a quadratic equation $X^2+Y^2 =< Z$. This can be handled efficiently by waiting until the values of X and/or Y and/or Z are known. As an example consider a database in which finitely many $<X,Y>$ pairs were held. The query

Find $<X,Y>$ pairs satisfying $X^2+Y^2 =< 20$

could be processed by calculating for each $<X,Y>$ the value $N= X^2+Y^2$, and returning only those pairs for which $N=<20$.

Applying a decision procedure is the most powerful way to handle constraints, while delaying them until all the variable values are known is the weakest (but computationally cheapest). In between these extremes is a variety of ways of using constraints actively during the search.

To use constraints actively, which is often called "propagation", the query evaluator extracts simple information from them, which can be easily combined with information extracted from the other constraints. The extracted information is often logically weaker than the constraint itself, and so for correctness the original constraint must also be delayed until, at a later point in the search further information can be extracted.

A well-known example of constraint propagation is interval reasoning. Given initial intervals $0<X<10,\ 0<Y<10,\ 0<Z<10,$ and the constraint $X^2+Y^2=Z,$ the evaluator can immediately deduce that $0<X<3.17$ and $0<Y<3.17$. These new intervals still do not capture the precise constraint, so the constraint must be delayed until the intervals associated with X, Y or Z become reduced further (as a result of processing another constraint, which may have been added during the search for a solution).

A powerful integration of interval reasoning in constraint programming is presented in (Benhamou, McAllester, and Hentenryck 1994). One form of constraint is a bound on the maximum cost of any solution. This cost can be dynamically reduced as new, better, solutions are found, thus yielding a form of branch-and-bound algorithm. Propagation on this bound constraint corresponds to estimating lower bounds on the cost of the remaining part of the problem. Thus the branch-and-bound algorithm automatically adapts itself to take into account the specific constraints of the problem.

Earlier theoretical work on Constraint Databases focused on embedding constraint solvers in database systems, but more recently constraint propagation has been implemented and tested in a database context (Harland and Ramamohanarao 1993) (Brodsky, Jaffar, and Maher 1993) (Bressan 1994)

4. ROAD LAYOUT WITH CONSTRAINT DATABASE

The road layout problem can be subdivided in several steps, which use different types of knowledge. These steps differ in the complexity of the constraints used and the strength of the restrictions imposed. Performance of the system is affected by the order in which candidates are produced and constraints considered. The following sequence of steps appears optimal:

1. Produce all major path alternatives, using topological reasoning.
2. Select geometrically acceptable path.
3. Assess the path to find a small set of best candidates.
4. Refine approximation to a smooth curve.
5. Refine cost estimate.

The constraint database contains the data consisting of the description of the situation and the constraints and provides an efficient search method to execute the query

find optimal path from A to B.

This reflects the general trend in AI and Computer Science to separate the descriptive specification of a problem from the computational organization to solve it. It makes it easy for the user to change the data and the constraints (e.g. to adapt to a different set of standards) without inference with the generic search engine (Frank, Robinson, and Blaze 1986a, Frank, Robinson, and Blaze 1986b, Frank, Robinson, and Blaze 1986c).

In order to apply this paradigm the representation of the problem must be adapted to the power of the search process. Depending on the representation the search considers a larger or smaller search space and can exclude candidates quicker without detailed exploration. A smaller search space is achieved using a two step process, separating a discrete topological and continuos problem. In figure 1 the major alternative path is immediately recognized. The path can vary in its exact position within a corridor, but the topological relations with respect to the obstacles are fixed.

PRODUCE SUBSTANTIALLY DIFFERENT ALTERNATIVES (CORRIDORS)

Produce a ranked set of substantially different alternative paths which can fulfill the constraints. A substantially different path is represented by a corridor within which the exact path can vary (figure 2). A corridor is acceptable if there is at least one path which fulfills the constraint in it; the paths within one corridor are not 'substantially different', meaning that they can vary without changing the assessment much.

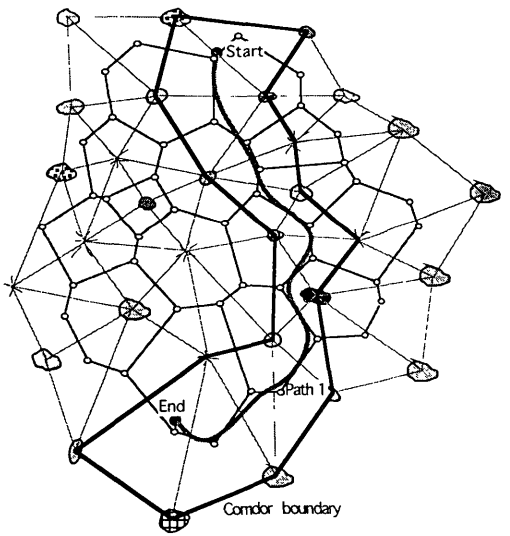

Figure 2 - A path with the corridor for variation

The substantially different alternative paths are created using Delaunay triangulation. The space with the obstacles is triangulated. Then the Voronoi diagram (the dual of the Delaunay triangulation) is created. All substantially different alternative paths consist of the edges in the Voronoi diagram. To each path belongs a corridor, which consists of all triangles from the Delaunay triangulation touched by the path (see figure 2).

The representation of the terrain is based on a triangulation, such that the cost function within the triangle is flat. Variation of the path within the triangle does not affect cost. Considering only the obstacles ("forbidden areas"), as in figure 1, creates a small

number of alternatives. If one considers the partition of the cost surface in classes of equal cost and equal height and integrates these boundaries into the Delaunay triangulation (Gold 1994), then the Voronoi diagram contains many more alternative paths. Each triangle contains one way-point and the triangles must be small enough that no more way points are necessary (figure 3).

Forbidden areas can be left out from the triangulation and barriers can be included (constrained Delaunay triangulation). The triangulation must represent the intersection of the cost function with the elevation data, such that variation within one triangle element is bounded.

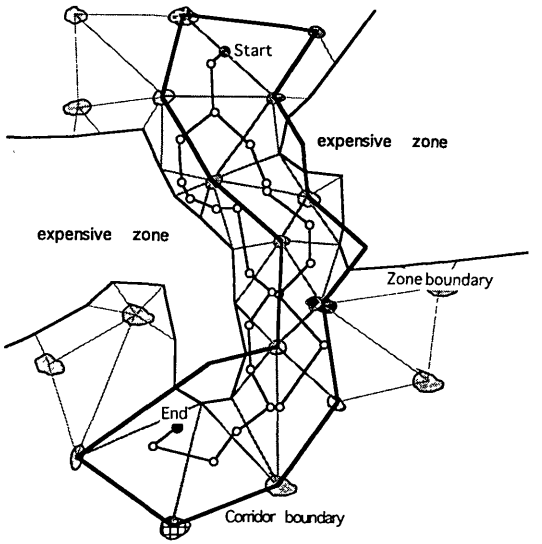

Figure 3 - A finer triangulation, including boundaries of cost zones

ASSESSMENT OF A LINEAR PATH

Roadway alternatives in the Voronoi diagram are described as a sequence of points connected by straight lines (polygon of tangents, Figure 4). It is possible to quickly assess their viability for road design without going to a curve model. Modeling curves is computationally considerably more expensive as it requires to solve problems of higher degree (bi-quadratic or of degree 4).

GEOMETRIC FORM - HORIZONTAL LAYOUT

For each change in direction a segment of a circle of at least minimal radius is necessary. This results in conditions for the length of the two adjoining segments in function of the change in direction. Additional length is necessary for a clothoide to achieve a smooth change in radius (figure 4).

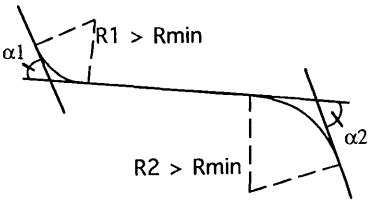

Figure 4 - Minimal length of segment

$length_{seg}(alpha_1 + alpha_2) = R_{min} * (alpha_1 + alpha_2) + (A^2 * R_{min})$

Segment length is further restricted by the minimal length. Thus the constraint is

$length'_{seg}(alpha_1 + alpha_2) = max\ (length_{seg}(alpha_1 + alpha_2),\ minimum_{seg})$

SLOPE:
Considering the height value for start and end of each tangent together with the length of the line quickly eliminates all paths with slopes much steeper than the maximum slope. For any change of slope the length of the neighboring segments must be long enough to accommodate the gradual change of slope (the calculation is the same as for horizontal layout).

COST OF THE ROADWAY:
The alternative paths are assessed for their cost in order to sort them by increasing cost for further testing. A shortest path within the corridor is constructed by considering the nodes of the triangles and the center points and form a new triangulation, effectively subdividing each triangle in six. In this subnetwork the path with least cost is determined and this value used for the ordering of alternatives. If the road crosses a linear obstacle (river, railway) the cost of special constructions are added.

It is possible to use an A* search algorithm, if we compute a lower cost estimate for the completion of any partial path using Euclidean distance and minimal unit cost. The error of the cost assessment depends on the size of the triangles and can be estimated.

5. IMPLEMENTATION METHODS

The method to construct viable alternatives for further selection described above can be programmed in a standard programming language. The constraints given by the design standards are then coded in by the programmer, possibly allowing changes of the constants in the constraints by the users.

Using a constraint programming language and a database puts the description of the constraints in a format the transportation engineer can understand and change. Nevertheless, the most efficient methods for solving the constraints are used. Most importantly, much less programming must be done and therefore the transportation engineer depends less on specialized acquired programs, where the assumptions built in are not visible.

6. CONCLUSION

An interesting and realistic part of the road design process is naturally expressed by the experts as a set of constraints. Using a constraint database for these GIS applications to design applications leads to an expression of the tasks in a formal language understandable to the domain expert.

The road design problem studied here is a realistic example for spatial design problems as many GIS applications include, e.g., in communications (optimal placement of stations for cellular phone system, design of distribution networks for cable based communication etc.), drainage systems and many others.

Fundamental for the formalization of these design problems is the appropriate representation of the problem. In this case study

- space is represented with a Delaunay triangulation, merging the cost surface and the digital terrain model
- the constraints on the road geometry are simplified to a linear form, leading to much faster constraint resolutions.

The most promising corridors selected in the triangulation are then further refined and assessed to determine the best solution. Constraint programming produces all efficient solutions and assures that none is overlooked.

We are currently working on the implementation of the described methods in a constraints database and to determine performance. Of particular interest is the combination of the first selection based on the triangulation and the second refinement to smooth curve.

REFERENCES

Benhamou, F , McAllester, D , and Hentenryck, P Van "CLP (Intervals) Revisited." In *International Symposium on Logic Programming in* 1994.

Borning, A *The Programming Language Aspects of ThingLab, a Constraint-Oriented Simulation Laboratory* ACM Transactions on Programming Languages and Systems//vol. 3, No. 4, October 1981//pp. 353 - 387: copy, 1981.

Borning, A. *Defining Constraints Graphically* Mantei, M. and Orbeton, P. (Eds.): Human Factors in Computing Systems, CHI'86 Conference Proceedings//Boston, April 13 - 17, 1986//pp. 137 - 143: orig, 1986.

Borning, A. *et al.* . "Constraint Hierarchies." In *Object Oriented Programming Systems, Languages and Applications (OOPSLA'87) in Orlando, Florida, October 1987*, 48-60, 1987.

Borning, Alan, Freeman-Benson, Bjorn, and Willson, Molly. "Constraint Hierarchies " In *Constraint Programming*, ed. Mayoh, Brian, Tyugu, Enn, and Uustalu, Tarmo. 1 - 16. Berlin. Springer Verlag, 1994.

Bressan, S "Database query optimization and evaluation as constraint satisfaction problem solving." In *ILPS Workshop on Constraint Databases in Lincoln*, CSE, Univ. Lincoln Nebraska, 1994.

Bressan, S., Provost, T. Le, and Wallace, M. *Towards the Application of Generalized Propagation to Database Query Processing*. European Computer Research Center, 1994. CORE 94-1.

Brodsky, A., Jaffar, J., and Maher, M. *Towards Practical Constraint Databases*. IBM Yorktown Heights Research Center, 1993

Burrough, Peter, and Frank, Andrew U "Concepts and paradigms in spatial information: Are current geographic information systems truly generic?" *IJGIS* in press (1994).

Burrough, P. A , and Frank, A U. *Geographic Objects with Indetermined Boundaries*. London· Taylor & Francis, 1995.

Dietrich, K., Graf, U., and Rotach, M. *Road Design (Strassenprojektierung)*. 2. ed., Zurich. Institut fur Verkehrsplanung und Transporttechnik ETHZ, 1975.

Eastman, J Ron. *IDRISI Version 4 1 Update Manual* Version 4.0, rev. 2 ed., Worcester, Mass.: Clark University, 1993.

Eastman, J. Ron *et al. GIS and Decision Making*. Version 4 0, rev. 2 ed., Vol 4 UNITAR Explorations in GIS Technology, Geneva, Switzerland: UNITAR, 1993.

Frank, Andrew U. "Requirements for a Database Management System for a GIS." *PE & RS* 54 (11 1988): 1557-1564

Frank, A. U. "Spatial Concepts, Geometric Data Models and Data Structures." In *GIS Design Models and Functionality in Leicester, UK*, edited by Maguire, David, Midlands Regional Research Laboratory, University of Leicester, 1990

Frank, Andrew U., Robinson, V , and Blaze, M. "An assessment of expert systems applied to problems in geographic information systems." *ASCE Journal of Surveying Engineering* 112 (3 1986a):

Frank, Andrew U., Robinson, V , and Blaze, M "Expert systems for geographic information systems: Review and Prospects." *Surveying and Mapping* 112 (2 1986b): 119-130.

Frank, Andrew U , Robinson, V., and Blaze, M "An introduction to expert systems " *ASCE Journal of Surveying Engineering* 112 (3 1986c):

Freestone, M "Begriffsverzeichnis· A Concept Index " In *11th British National Database Conference in* 1 - 22, 1993.

Gallaire, H , Minker, Jack, and Nicolas, Jean-Marie. "Logic and Databases: a deductive approach." *ACM* 16 (2 1984)· 153-184

Gold, Christopher. "Three Approaches to Automated Topology, and How Computational Geometry Helps." In *Six International Symposium on Spatial Data Handling in Edinburgh*, edited by Waugh, T. C., and Healey, R G , AGI, 145 - 158, 1994

Goodchild, Michael F. "A geographical perspective on spatial data models." In *GIS Design Models and Functionality in Leicester*, edited by Maguire, David, Midlands Regional Research Laboratory, 1990

Harland, J , and Ramamohanarao, K. "Constraint Propagation for Linear Recursive Rules." In *ICLP in Budapest*, 1993.

Herring, John R. "TIGRIS. A Data Model for an Object Oriented Geographic Information System " In *GIS Design Models and Functionality in Leicester*, edited by Maguire, David, Midlands Regional Research Laboratory, 1990.

Kanellakis, P., Kuper, G , and Revesz, P. "Constraint Query Languages." In *Principles of Data Systems in* 1990

Kanellakis, P. C , and Goldin, D. Q. "Constraint Programming and Database Query Languages." In *2nd Conference on Theoretical Aspects of Computer Science in* 1994.

Kuhn, W *Interaktion mit raumbezogenen Informationssystemen - Vom Konstruieren zum Editieren geometrischer Modelle*. Dissertation Nr 8897//Mitteilung Nr. 44, Institut fur Geodäsie und Photogrammetrie//ETH Zurich. 1989.

Monfroy, E "Groebner Bases. Strategies and Applications." In Conference on Artificial Intelligence and Symbolid Mathematical Computation in Karlsruhe, BRD, 1992.

Peuquet, D J "A hybird structure for the storage and manipulation of very large spatial data sets." Computer Vision, Graphics, and Image Processing 24 (1 1983): 14-27.