DATABASE ARCHITECTURE FOR MULTI-SCALE GIS

Christopher B. Jones
Department of Computer Studies
The Polytechnic Of Wales
Pontypridd
Mid Glamorgan, CF37 1DL, UK

ABSTRACT

Many applications of GIS, such as planning, exploration and monitoring of natural resources, require the mapping and analysis of spatial data at widely differing scales. Ideally, a single large scale representation of spatial data might be stored, from which smaller scale versions were derived. Currently however, automation of the necessary generalisation processes is not sufficiently well advanced for this to be a possibility. Consequently, multiple representations must be maintained, though proven generalisation techniques can be used to reduce data duplication, provided that processing overheads are not prohibitive. Maintenance of a multiple representation database requires a flexible approach to the use of both single-scale and multiresolution data structures. Furthermore, rule-based software is required for a) deciding whether new datasets should be merged with existing ones, or stored as separate representations, and b) selecting appropriate representations and applying generalisation procedures to satisfy user queries. This paper presents an overview of a database design, based on a deductive knowledge-based systems architecture, which attempts to meet these requirements.

INTRODUCTION

Increasing interest in the use of geographical information systems (GIS) brings with it requirements for the analysis and display of geographical information at various scales, relating to different locations and to different themes. The accumulation of this information introduces a need for sophisticated databases that are flexible with respect to the variety of stored data and to the scale and locational accuracy of the output. Such requirements arise in organisations concerned with monitoring or exploiting the natural and man-made environment.

Maintenance of data derived from a variety of source scales raises a major issue of whether the individual real-world objects should be represented once, at their highest resolution, or whether multiple versions at different scales should be stored. Ideally perhaps the former option appears most desirable, since it avoids data redundancy and the

possibility of inconsistency between versions. The approach depends however upon the assumption that smaller scale versions can be derived automatically. With the current, relatively limited, capabilities of automatic generalisation software, this is not a valid assumption (Brassel and Weibel, 1988). In a multi-scale database servicing a wide range of output requirements there is therefore good reason to store multiple representations of the same objects (Brassel, 1985). Even when small scale versions can be derived automatically, there will be situations, involving large degrees of generalisation, in which the delays due to computation could not be tolerated in an interactive GIS. In such circumstances it could be desirable to store the results of automated generalisation.

The presence of multiple representations of spatial objects, and the need for retrieval at a range of scales, places considerable demands upon a database management system. If it is to maintain and retrieve data with the minimum of user-intervention, it must incorporate software capable of making decisions about updates and retrievals. When new datasets are loaded, decisions must be taken about whether to replace existing data, merge with existing data, or store as a separate representation. On querying the database there may be several candidate representations. One of these may be selected for output or it may be used to derive an appropriate representation using automatic generalisation procedures. In addition to the inclusion of 'intelligent' software, the need arises for data structures which are efficient for access in terms of ground resolution, spatial location, topology and aspatial attributes.

A research project has been initiated with the aim of building an experimental multi-scale spatial information system. In the remainder of the paper the components of the proposed experimental system are outlined, before discussing specific issues which arise in designing and implementing multi-scale GIS. Attention is focused in particular on multiresolution data structures, indexing mechanisms and the maintenance and query of multiple scale representations.

COMPONENTS OF A MULTI-SCALE DATABASE ARCHITECTURE

An overview of the main components of a proposed multi-scale database is illustrated in Figure 1. All updates and queries are channelled through a deductive subsystem, the rule-base of which controls changes to the contents of the database and retrievals from it. The contents of the database are summarised within an object directory which, though it may be spatially segmented, serves primarily to record the presence of stored objects in terms of their application-specific classes and the nature of their representations with regard to dimension, locational accuracy and spatial data model. The rule base of the deductive subsystem refers to the

current contents of this object directory in order to make decisions about appropriate strategies for update and retrieval. It also controls the execution of spatial processors required for certain update operations and for performing, where necessary, generalisation operations on retrieved objects.

The detailed spatial structure of objects listed in the object directory is recorded in the topology and metric geometry components. The metric geometry component stores data referenced directly to locational coordinates and could include both vector and raster data employing specialised multiresolution data structures. In the case of vector structured objects a close relationship could be expected with corresponding elements in the topology component of the database. The distinction between topology and metric geometry is intended to facilitate efficient search based on topological information at various levels of detail. Range searches for all objects in a given rectangular window may be served directly by the metric geometry data structures.

The rule-based component of the experimental system is envisaged initially as a deductive, or logic database which may be implemented in a logic programming language with extensions for calling external procedures and for access to permanent storage. Execution of rules for update of single resolution and multiresolution spatial data structures and for generalisation will then be achieved by calling the various spatial

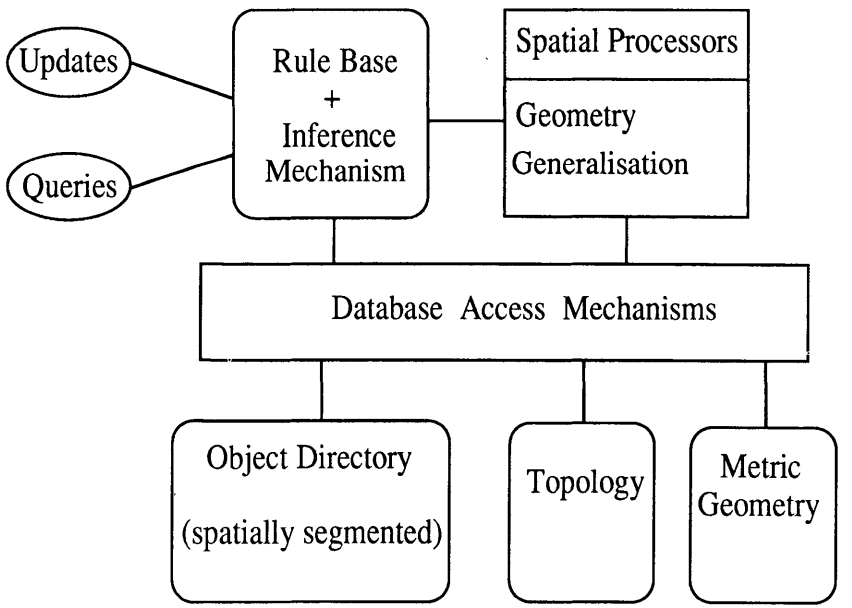

Figure 1

processors. Implementation of the spatial data structures requires the use of complex data types, while the processors which operate on them could, in some cases, consist of knowledge-based subsystems in their own right. These latter components of the database may appear therefore to be suited to implementation using object-oriented programming techniques.

Recognition of the importance of combining rule processing with object-oriented databases is reflected in the design of systems such as POSTGRES (Stonebraker, 1986). The potential of this type of database system for implementing multi-scale GIS has already been identified by Guptill (1989, 1990). From the more purely deductive database standpoint, new versions of the logic programming language Prolog are being developed to provide efficient integration with a permanent database (Bocca et al, 1989; Vieille et al, 1990). By adding facilities for handling complex data types and for calling external procedures, the deductive database architecture may also then provide a suitable basis for building multi-scale geographical databases.

MULTIRESOLUTION DATA STRUCTURES

Whether there are single or multiple representations of individual objects and classes of object, each representation may be regarded as a candidate for retrieval over a range of scales. The largest scale limit will be constrained by the locational accuracy of the geometric data. The smallest scale limit will be determined by the capability of automated generalisation functions which can operate on the object. Widely used line generalisation procedures such as the Douglas algorithm (Douglas and Peucker, 1973) have been used, in combination with smoothing operators over scale changes in excess of a factor of 100 (Abraham, 1988). When the linear features form part of areal objects, automated procedures are generally very much more restrictive, since issues such as object amalgamation and displacement must be taken into account. Automated areal generalisation was used in the ASTRA system (Leberl and Olson, 1986) but scale changes were only of a factor of about two. A variety of techniques is available for generalisation of digital terrain models (Weibel, 1987). Limits on the possible degree of generalisation of these models depends on the error tolerance of the application. However, when structure lines (ridges, valleys and form lines) are added to the model, the limits may be expected to be similar to those of the generalisation of the individual linear features.

Given that individual representations apply over a range of scales, the question arises as to how best to store the objects to achieve efficient access at different scales. The options are single storage of the object with generalisation to smaller scales at the time of the query; storage of several pre-generalised versions of the object, with the possibility of data

duplication (as in Guptill, 1990); and storage of a non-duplicating hierarchical representation of the single object (see below). The first of these options could, in the case of linear features and terrain models, require initial retrieval of orders of magnitude excess data before simplification by a generalisation function. The second option could give efficient access to a representation which may closely approximate the retrieval specification, but at the expense of a storage overhead due to data duplication. The third option is a compromise in which a generalisation function is used to segregate the geometric component of the objects according to their contribution to shape and accuracy. By organising the component data in a hierarchical manner it is then possible to access only the geometric data required to build a representation at, or an approximation to, the required level of generalisation.

Multiresolution data structures which avoid or minimise data duplication are available for both linear features and surfaces. For linear features, the strip tree (Ballard, 1981) provides a means of accessing successively higher resolution approximations to a curve represented by rectangular strips. In its original form it is not very space efficient, as individual points may be stored several times if they bound successively narrower strips. Each rectangle must also be explicitly defined. The original strip tree consists essentially of a binary tree. The root node stores a rectangular strip which encloses the entire feature, along with pointers to two offspring. A point where the curve touches the side of the initial strip is used to subdivide the curve into two parts, each of which is represented by enclosing strips which are stored in the offspring nodes. The curve is divided recursively in this manner until individual strips coincide with straight line, zero width, segments between successive vertices of the feature.

The multi-scale line tree (or line generalisation tree) is related to the strip tree and may be regarded as a tree of variable branching ratio rather than a binary tree (Jones 1984, Jones and Abraham 1986,1987). Each level of the structure corresponds to a maximum implicit strip width. Furthermore, it is vertex rather than strip oriented, and each level stores vertices which are intermediate to those at the next higher level. The result is that it is significantly more space-efficient than the strip tree. It has been implemented in a network database in which each level of a hierarchy is stored independently of the other levels of the same line object, but in association with the equivalent generalisation levels of other objects (Abraham, 1988). Thus rapid access to all features of a particular resolution is facilitated by only retrieving, for each object, those hierarchical levels which are relevant to a specified output scale (or spatial resolution).

Use of a multi-scale line tree introduces the problem of maintaining

aspatial and topological attributes of the line features. If the hierarchy extends across a wide range of scales, the line itself may be geographically extensive such that there are distinct internal subdivisions relating to different feature codes and to topological nodes. By attaching sequence numbers to the component vertices of a line, aspatial classification and topological structure can be defined in terms of ranges of sequence numbers and individual sequence numbers which have been designated as nodes (Jones and Abraham, 1987; Abraham, 1988). To retain access efficiency, the node vertices should be stored at the highest hierarchical level (lowest resolution) at which they are likely to be required. Thus vertices which the generalisation procedure may classify as low level would, if they were logical nodes, be raised to the appropriate higher level.

Multiresolution representations of surfaces may be categorised into those based on mathematical functional models of the surface and those based on original, or derived, sample points. If the coefficients of a surface function are orthogonal, in the sense that they represent independent components of the surface shape, then a multiresolution data structure could be created by separating the storage of the components into distinct records. Each record would correspond to a 'level', characterised by the extent to which the stored coefficients contributed to the surface shape. The most important components could be stored at the highest levels, while less significant ones were stored at progressively lower levels. A problem which occurs when using global functions, such as Fourier Series, is that the reconstructed, simplified surface, may be subject locally to relatively large errors. A mathematical function approach which controls errors can be obtained by partitioning the surface into rectangular regions each of which is represented by its own function (Pfaltz, 1975). By partitioning the surface in the manner of a quadtree, the size of the quadrants can be reduced locally until the chosen function fits the surface to within a pre-specified tolerance (Chen and Tobler, 1986; Leifer and Mark, 1987). Although the method has been applied primarily to the representation of surfaces at a specified error tolerance, it could be extended into a multiresolution quadtree in which intermediate (subdivided) nodes stored a function accompanied by a measure of the associated error.

Surfaces represented by sample points are usually organised either as regular grids of elevation values or as an irregular set of significant points. Irregularly distributed points are typically structured by triangulation, to form a triangulated irregular network, or TIN (Peucker et al, 1978). Because the sample density of a TIN is adapted to local variation in surface detail the structure lends itself to implementation as a multiresolution structure.

The Delaunay pyramid (De Floriani, 1989) is a hierarchical multiresolution tree for storing triangulations. The top level of the tree stores a Delaunay triangulation of a subset of the original dataset of important points. The next lower level is constructed by adding vertices which are chosen to be the most distant from the previous triangulated surface. Points are added to the previous surface, which is re-triangulated to accommodate them until the error between this new surface and the remaining points is within a pre-set tolerance. The next lower level is created in a similar manner, controlled by the error tolerance for that level. Each level stores a list of the triangles and vertices of which it is composed, the differences (in terms of triangles) between the adjacent upper and lower levels, and pointers from certain triangles to those which replace them, and are hence intersected by them, at the immediate lower level. Note that only a subset of triangles at each level points to lower triangles, since some of the previous triangles will be retained in the lower level.

An advantage of a triangulated surface model is that it provides the possibility of being integrated with point, linear and polygonal features. If the vertices which define the latter features are merged with those which define a digital elevation model then, after triangulation, the linear and polygonal features can be constituted by the edges within the triangulation, while point features are represented by single nodes. To ensure that linear features are retained in this way, the triangulation process must be constrained by boundaries defined by the linear features (see De Floriani and Puppo, 1988, for the constrained triangulation of multiresolution topographic surfaces). Provided all nodes are uniquely identified, the embedded spatial objects and their topology can be referenced directly to sequences of, and individual, triangulation nodes. In a multiresolution structure, references to nodes can include their level within the hierarchy and, just as with the multi-scale line tree topology, their nodes would be stored at the highest level that they could be expected to be of use. A multiresolution triangulation data structure integrated with topology and feature specification is currently being developed (details will be published elsewhere).

INDEXING MECHANISMS

Appropriate schemes for efficient spatial access to multiresolution hierarchies may vary according to whether the objects encoded in the hierarchies are very extensive compared with potential regions of interest. This factor determines the desirability of incorporating spatial indexing within the object representation in addition to a spatial index which refers only to the entire objects. The latter indexing scheme would indicate the storage location of objects, the geometry of which was stored in, for example, a multi-scale line tree, a multiresolution triangulation or a single level representation. Methods of implementing

the primary object index include techniques such as i) a fixed grid with references to intersecting objects; ii) a bounding quadtree cell scheme (Abel and Smith, 1983); and iii) an R-tree, or one of its relatives, which works with minimum bounding rectangles (Guttman 1984, Faloutsos, 1987). Depending on the nature of the application, an additional aspatial index to objects could also be desirable.

If the geometry of objects referenced by the spatial or aspatial index was extensive compared with the search window, it would be necessary to traverse the geometric data structure, selecting those parts inside the window. If the geometry was stored as a multiresolution hierarchy (line tree or triangulation), covering a wide range of scales, then it could frequently be expected to be spatially extensive relative to query windows for large scale applications. The multi-scale line tree was implemented on this assumption and incorporated spatial indexing within each level of the hierarchy. In that experimental database, both fixed grid and quadtree schemes were applied, in which the cells of the grids and of the quadtrees stored sets of vertices in chained records. When the fixed grid size was selected to be different for each level (according to a regular pyramid) the performance of the two schemes was found to be similar (Abraham, 1988).

In De Floriani's Delaunay pyramid (De Floriani, 1989), the pointer-based implementation provides some direction to spatial search within the structure once candidate triangles have been identified at the top level. The implementation described appears to have been oriented towards point rather than window searches. An alternative approach, currently being pursued, is to impose a spatial index on each level.

Bearing in mind that a multi-scale database may be very large and that objects may occur at widely differing levels of class-generalisation hierarchies, the concept of a single spatial index and a single list or index of objects becomes rather monolithic. Given that the scale of the output can be expected to be correlated with the level of class generalisation, a natural development of the indexing system is to segregate it into generalisation levels allowing direct access into an appropriate level. Each level could be associated with some limiting spatial resolution and would reference only classes of object which were regarded as likely to become significant at that scale. The choice of classes could be somewhat arbitrary on the assumption that a data dictionary indicated the correlation between class and level. It would not be necessary to refer explicitly to the parents of classes in a class-generalisation hierarchy, provided the content and structure of all such hierarchies was stored separately, allowing them to be inferred (see, for example, Egenhofer and Frank, 1989).

MAINTENANCE OF MULTIPLE REPRESENTATIONS

An important issue in maintaining multiple representations is the extent to which data duplication and data redundancy are to be tolerated. Duplication will occur when one representation is a simplified version of the other if its geometry, such as the vertices of a line or triangulated surface, is a subset of that of the other version. If an automatic procedure exists for performing the simplification, then the smaller scale version may be regarded as redundant. Data redundancy in this sense can also arise in the absence of data duplication provided that there is an automatic procedure for deriving a required small scale version from the larger scale version. For the purposes of an interactive information system however, this notion of redundancy may be questioned if the processing required by the automatic procedure was too much to provide an acceptable response time.

The multiresolution data structures referred to earlier give rapid access to generalised versions which are geometric subsets, and they therefore provide a means of avoiding data duplication, at least for linear features and surfaces. When 'quantum leap' differences occur in the course of generalisation, due for example to changes in dimensionality and to merging and displacement of objects, the existing types of multiresolution data structures cannot be used. It can also be expected that where automatic procedures do exist for this degree of generalisation, there is a greater chance of being too slow for satisfactory user interaction. It is in the event of major changes in the geometric representation that the storage of multiple versions is most likely to be appropriate. This does not however preclude the use of multiresolution data structures for separately maintaining both the smaller and larger scale representations across their different ranges of scales.

Another situation in which multiple versions might be stored is that in which data duplication was very localised, due to the presence of geographically small areas of large scale, high resolution data within a region which was covered by a much more extensive, smaller scale representation. A method of maintaining a consistent representation at the small scale, while also avoiding the data redundancy, would be to generate a multiresolution data structure from the large scale data and merge it, at the top level, with the existing small scale version. This would involve cutting out the duplicated section and edge matching between the two versions (see Monmonier, 1989b, for a discussion of techniques for automatic matching of map features which differ in their original scale of representation). It may be envisaged that the processing overheads incurred in local deletions followed by merging of the new data may not be deemed justifiable for relatively small quantities of data, since the coverage at the larger scales would only be patchy. As more

extensive coverage at the larger scales accumulated in the database, a point would be reached at which the delete and merge process became justifiable.

Control over the decision on when to merge new data with stored data can be placed within a rule base which is integral to the database management system. An analogy may be made with trigger mechanisms which have been incorporated in database systems such as POSTGRES (Stonebraker, 1986). Triggers are an automatic means of maintaining integrity based on rules which dictate that once a particular data element has changed, it may propagate a sequence of changes to related records in the database. Each trigger may be expressed as a production rule which is implemented by a forward chaining mechanism in which the firing of one trigger may lead to subsequent firing of another trigger.

The possibility of a chain of triggered updates can be envisaged in a multiple representation database if the insertion of large scale representations filled gaps in an intermediate scale representation, enabling the latter to be merged with an existing, smaller scale, representation. Thus databases which include trigger mechanisms can be seen, to some extent, as dynamic, self-maintaining systems. If there was any doubt about the reliability of such systems, with regard for example to correct matching and merging of geometry and topology, these updates could be subject to user-verification before being committed to the database. All operations could be reversible if historical records were maintained in archival memory.

DATABASE QUERIES ON MULTIPLE REPRESENTATIONS

A query to a multi-scale, multiple representation database can be expected to be faced with a choice of versions which are candidates for retrieval. An automatic query processor would then need to make a choice of the appropriate retrieval to meet the user's requirements. Criteria for an appropriate retrieval would differ according to whether the output was required for analytical purposes or solely cartographic purposes. In the latter case the version retrieved might be the one which most closely resembled the level of generalisation dictated by the map's theme and scale. Such a version could be obtained by a variety of means. There could be a single level stored representation of the appropriate generalisation. Alternatively there could be a multiresolution data structure which encompassed the required generalisation level and could therefore be traversed to construct the output. Failing that, there could be a large scale version which could be generalised by software. In the latter case the automated generalisation process could operate only on that large scale version or perhaps, as Monmonier (1989a) has proposed, an additional smaller scale version could be used to guide generalisation to an intermediate level. If no sufficiently large scale data were

available, a poorer quality version could be retrieved and the user warned accordingly, or a failure reported.

The above strategies would not in general be suitable for queries based on the need for data analysis problems in which locational accuracy was of prime importance. Cartographic generalisation would not then be desirable and the appropriate version would be that derived directly from, or a subset of, the largest scale representation. Particular problems could arise with this sort of query if coverage of the query window required access to representations with differing locational accuracy. In any event, data retrieved for analytical purposes would need to be labelled with their accuracy, and processes involving overlay between different objects would need to maintain a measure of the errors propagated by the combination of geometric objects.

It is apparent that implementation of a query processor capable of adapting to user requirements will require the specification of rules to control the action to be taken under the various conditions of user needs and data availablility. The query processor could operate initially on the object directory which recorded the class, location, dimension, accuracy and spatial data model of objects stored in the data base. The rules could then be applied to select the best representation given the query conditions. This would include taking the decision on whether to apply automatic generalisation procedures and choosing which procedures were most suitable. The mechanism for implementing a deductive system governing queries may differ somewhat from that governing updates, referred to in the previous section. Because a query may be regarded as a specific goal, it lends itself to a backward chaining mechanism which attempts to match the contents of the database with the search conditions.

SUMMARY

The construction of a database, capable of maintaining multiple scale representations of spatial objects, poses major problems with regard both to the development of efficient multiresolution data structures and to controlling update and answering queries. The need for explicit rules governing update, database integrity and the retrieval of generalised objects indicates the desirability of a deductive, knowledge-based architecture providing declarative rule specification. Storage of complex objects in specialised data structures, along with the need for associated processors for update and generalisation, suggests however that it may also be appropriate to use object-oriented programming techniques. A research project is currently in progress with the aim of experimenting with deductive databases for implementing a multi-scale spatial information system. In the planned system, rules of update and query processing are specified in a deductive, logic database which is interfaced to spatial processors and spatial data structures which may be

implemented, at least in part, in procedural or object-oriented languages. The operation of the spatial processors may themselves employ knowledge-based inference techniques which are encapsulated within the respective modules. The primary, deductive component of the system makes decisions about appropriate update and retrieval operations by referring to the current contents of an object directory, which summarises the nature of stored object representations in terms of their feature class, location, dimension, accuracy, and spatial data model. Details of the spatial structure of stored objects are maintained within separate topology and metric geometry components of the database, to which the object directory refers.

REFERENCES

Abel, D.J. and J.L. Smith 1983, A data structure and algorithm based on a linear key for rectangular retrieval: Computer Vision, Graphics and Image Processing, Vol. 24, pp. 1-13.

Abraham, I.M. 1988, Automated Cartographic Line Generalisation and Scale-Independent Databases, PhD Thesis, The Polytechnic of Wales.

Ballard, D.H. 1981, Strip trees: a hierarchical representation for curves: Communications of the ACM, 24, pp. 310-321.

Bocca, J., M. Dahmen, M. Freeston, G. Macartney, P.J. Pearson 1989, KB-PROLOG, a PROLOG for very large kowledge bases: Proceedings 7th British National Conference on Databases, Edinburgh, pp. 163-184.

Brassel, K.E. 1985, Strategies and data models for computer-aided generalization: International Yearbook of Cartography, Vol. 25, pp. 11-28.

Brassel, K.E. and R. Weibel 1988, A review and conceptual framework of automated map generalization: International Journal of Geographical Information Systems, Vol. 2, No. 3, pp.229-244.

Chen, Z.-T., and W. Tobler 1986, Quadtree representations of digital terrain: Proceedings Auto Carto London, Vol. 1, pp. 475-484.

De Floriani, L. 1989, A pyramidal data structure for triangle-based surface description: IEEE computer Graphics and Applications, March 1989, pp. 67-78.

De Floriani, L. and E. Puppo 1988, Constrained Delaunay triangulation for multiresolution surface description: Proceedings Ninth IEEE International Conference on Pattern Recognition, CS Press, Los Alamitos, California, pp. 566-569.

Douglas, D.H. and T.K. Peucker 1973, Algorithms for the reduction of the number of points required to represent a digitized line or its caricature: Canadian Cartographer, Vol. 10, No. 2, pp.112-122.

Egenhofer, M.J. and A.U. Frank 1989, Object-oriented modeling in GIS: inheritance and propagation: Proceedings Auto-Carto 9, Ninth International Conference on Computer-Assisted Cartography, Baltimore, Maryland, pp.588-598.

Faloutsos, C., T. Sellis, N. Roussopoulos 1987, Analysis of object-oriented spatial access methods: Proceedings ACM SIGMOD'87, pp. 426-439.

Guptill, S.C. 1989, Speculations on seamless, scaleless cartographic data bases: Proceedings Auto Carto 9, Ninth International Conference on Computer-Assisted Cartography, Baltimore, Maryland, pp. 436-443.

Guptill, S.C. 1990, Multiple representations of geographic entities through space and time: Proceedings 4th International Symposium on Spatial Data Handling, Zurich, pp. 859-868

Guttman, A. 1984, R-trees: a dynamic index structure for spatial searching: Proceedings ACM SIGMOD'84, pp.47-57.

Jones, C.B. 1984, A tree data structure for cartographic line generalisation: Proceedings Eurocarto III, Research Center Joanneum, Institute for Image Processing and Computer Graphics, Graz.

Jones, C.B. and I.M. Abraham 1986, Design considerations for a scale-independent database: Proceedings, Second International Symposium on Spatial Data Handling, Seattle, pp.384-398.

Jones, C.B. and I.M. Abraham 1987, Line generalisation in a global cartographic database: Cartographica, Vol. 24, No. 3, pp.32-45.

Leberl, F.W. and D. Olson 1986, ASTRA - A system for automated scale transition: Photogrammetric Engineering and Remote Sensing, Vol. 52, No. 2, pp. 251-258.

Leifer, L.A. and D.M. Mark 1987, Recursive approximation of topographic data using quadtrees and orthogonal polynomials: Proceedings Auto-Carto 8, Eight International Conference on Computer-Assisted Cartography, Baltimore, Maryland, pp. 650-659.

Monmonier, M. 1989a, Interpolated generalisation: cartographic theory for expert-guided feature displacement: Cartographica, Vol. 26, No. 1, pp. 43-64.

Monmonier, M. 1989b, Regionalizing and matching features for interpolated displacement in the automated generalisation of digital cartographic databases: Cartographica, Vol. 26, No. 2, pp.21-39.

Peucker, T.K., R.F. Fowler, J.J. Little, D.M. Mark 1978, The triangulated irregular network: Proceedings Digital Terrain Models (DTM) Symposium, ASP-ACSM, St. Louis, pp. 516-540.

Pfaltz, J.L. 1975, Representation of geographic surfaces within a computer: in Display and Analysis of Spatial Data, Edited by J.C. Davis and M. J. McCullagh, Wiley, pp. 210-230.

Stonebraker, M.R. and L.A. Rowe 1986, The design of POSTGRES: Proceedings ACM SIGMOD'86, pp.340-355.

Vieille, L., P. Bayer, V. Kuchenhoff, A. Lefebvre 1990, EKS-V1, a short overview: Proceedings AAAI-90 Workshop on Knowledge Base Management Systems, Boston.

Weibel, R. 1987, An adaptive methodology for automated relief generalization, Proceedings Auto-Carto 8, Eight International Conference on Computer-Assisted Cartography, Baltimore, Maryland, pp. 42-49.