2009

Three-Dimensional Spatial Information Systems: State of the Art Review

Bianca Schoen-Phelan
Dublin Institute of Technology, bianca.phelan@dit.ie

D. Laefer
University College Dublin

S. W. Morrish
University College Dublin

M. Bertolotto
University College Dublin

Recommended Citation

Schoen-Phelan, B., Laefer, D., Morrish, S. & Bertolotto, M. (2009). Three-dimensional Spatial Information Systems: State of the Art Review. Recent Patents on Computer Science, vol.2, pp.21-31. doi:10.2174/1874479610902010021

Three-dimensional Spatial Information Systems: State of the Art Review

B. Schön*, D. F. Laefer, S. W. Morrish***, M. Bertolotto***

*School of Computer Science and Informatics, University College Dublin, Ireland
**Urban Modelling Group, School of Architecture, Landscape & Civil Engineering, University College Dublin, Ireland
*** Urban Modelling Group, School of Architecture, Landscape & Civil Engineering, University College Dublin, Ireland

Abstract: A spatial information system (SIS) is critical to the hosting, querying, and analyzing of spatial data sets. The increasing availability of three-dimensional (3D) data (e.g. from aerial and terrestrial laser scanning) and the desire to use such data in large geo-spatial platforms have been dual drivers in the evolution of integrated SISs. Within this context, recent patents demonstrate efforts to handle large data sets, especially complex point clouds. While the development of feature-rich geo-systems has been well documented, the implementation of support for 3D capabilities is only now being addressed. This paper documents the underlying technologies implemented for the support for 3D features in SISs. Examples include ESRI's ArcGIS geo-database with its support for two-and-a-half dimensions (2.5D) in its Digital Elevation Model (DEM) and Triangular Irregular Network (TIN), the more recent development of the Terrain feature class, and support for 3D objects and buildings with its multi-patch feature class. Recent patents and research advances aim to extract DEMs and TINs automatically from point cloud data. In this context, various data structuring innovations are presented including both commercial and open source alternatives.

Keywords: Spatial database management system, 3D spatial data support, GIS

INTRODUCTION

A Database Management System (DBMS) controls the organization, storage, management and retrieval of all data that is kept in a database. A DBMS ensures that data inconsistencies and data redundancies are significantly reduced compared to storing information in a file system. A DBMS also facilitates data integrity, as well as multi-user control on shared data. Traditional DBMSs were not developed to support spatial data (i.e. data with a spatial component) and, as such, did not provide mechanisms for the storage and querying of such data.

However, in recent decades, the amount of spatial data has significantly increased. In fact, it has been estimated that 80% of all data presently collected have at least one spatial component (called the *extent* [1]). Therefore much attention has been dedicated recently towards developing abilities to effectively exploit and process the spatial extent of data.

In a parallel development stream, geographic information systems (GISs) have been used since the early 1960s to address the issues faced by planners and resource managers dealing with the spatial nature of systems in the real world. GISs consider both what an object is and where it is located. The mapping component is combined with the information components in planning and resource management. The earliest systems utilized a distinct file-based system of representing spatial features and the related attribute information in both vector and raster formats. ESRI's Coverage and Shape file along with Mapinfo's TAB format are examples of vector formats, while ESRI's GRID, GRASS Raster, and ERDAS IMG formats are examples of GIS raster formats [2].

ESRI's Coverage format uses an extensive set of files to support its feature topology and manage its tabular data in combined INFO files. ESRI's Shapefile format is an open file system that uses basic dBASE file (DBF) tables to support its feature attributes and is capable of linking to external databases through Open Database Connectivity (ODBC). Further developments brought about ArcSDE, permit the integration and storage of spatial data on relational databases such as Informix, IBM's DB2, Oracle's Oracle Spatial and Microsoft's SQL Server.

Later, an integrated approach to store the spatial extent (together with the attribute data) directly into the database (in the same table) was developed. This approach, which relied on the extensibility of relational DBMSs [i.e. the ability to add new types and operations to a Relational Database Management System (RDBMS)], produced the so-called Spatial Database Management System (SDBMS). This technology allows management of all data within the same engine. Additionally, retrieval and manipulation of all data are facilitated through structured query language (SQL).

An example of a commercial SDBMS is Oracle Spatial (the spatial extension to Oracle DBMS). In the early 1990s, spatial data was predominantly stored in modified RDBMSs. In the late 1990s, a new paradigm emerged called the Object-Relational Database Management System (ORDBMS). An ORDBMS allowed *geometry object types* to be added to the database.

Support for two-dimensional (2D) feature types and indexing techniques in systems dealing with spatial data sets has been well documented. As part of this, spatial indexing techniques evolved during in the middle of the 1980s with Guttman's R-tree [3] being one of the most popular and enduring indexing techniques developed.

Currently, GISs and SDBMSs aim to integrate true 3D features. This is largely driven by the increased availability of 3D data (e.g. from aerial or terrestrial laser scanning). A 3D spatial system must support 3D data types, such as point, line, surface and volume in 3D Euclidean space. Three-dimensional data types are based on a 3D geometric data model (i.e. vector and/or raster data with underlying geometry and topology). A 3D spatial system must also offer operations and functions embedded into its query language that can operate with its 3D data types [4].

This paper presents the development of 3D technologies for systematic handling of spatial data sets, with a particular emphasis on patents. First, spatial data representation is discussed, thereby introducing geometry features for vector and raster data. Commercial systems offer different levels of support for each data type, which result in varying levels of functionality.

The majority of currently available 3D data contains elevation information. Support for DEM and Digital Terrain Models (DTM) by three main vendors – Oracle, PostGIS, and ESRI – will be presented below, along with the major developments in data storage and processing.

DEM/DTMs can be represented in a raster format as index or floating point grids or in a vector format in the form of a TIN, which is a vector-based representation particularly suitable for topological queries. Whereas TINs themselves only provide two-and-a-half-dimensional (2.5D) support, a special form of TINs referred to as the Tetrahedral Irregular Network (TEN) supports true 3D volumes. A TEN is a promising new approach to increase the 3D capabilities of SDBMS. Unfortunately, current systems offer only limited TEN support. However, there is a growing body of research under development. As part of this, ESRI has developed the multipatch feature as a native surface geometry type for 3D support (as will be subsequently described). Other efforts have been made to implement freeform curves and surfaces for 3D support in SDBMSs. Both features are discussed subsequently.

In order to facilitate efficient execution of spatial queries on these new spatial types, database indexes must be able to process spatial data. Two-dimensional spatial indexing has been well researched. Yet, support for true 3D spatial indexing is still undergoing active development and research and true 3D indexing will further the uses of 3D spatial databases. Current developments and vendor solutions are herein subsequently described.

To facilitate this discussion, this paper is structured into six additional sections. Section 2 explains the difference between the main forms of spatial data (vector and raster) and specific support provided by different SIS vendors. Section 3 outlines how spatial data is processed for manipulation and visualization including a detailed illustration of DEMs. As DEMs are 2.5D presentations of 3D data (describing a surface rather than the volume of a feature), there has been significant development in the automatic generation of DEMs/TINs from 3D laser scanning point clouds.

Section 4 presents true 3D representations of features through a 3D type called Multipatch, developed by ESRI. As described in section 5, NURBS (Oracle Spatial 11g) offers a more flexible approach than Multipatch by employing freeform curves and surfaces, for the representation of 3D volumes.

Section 6 discusses the indexing of multi-dimensional data in SISs. This is particularly necessary in large data sets to facilitate the execution of queries in a timely manner. Contemporary approaches that aim to provide 3D indexing are represented, as well as current vendor solutions. The paper concludes with Section 7 on current and future developments

In summary, this paper presents current research on empowering SISs with true 3D capabilities. The state-of-the-art, as reflected by research, patents, and vendor solutions, is presented with a view to providing a thorough understanding of spatial databases and their capabilities.

SPATIAL DATA REPRESENTATION

Spatial data sets and geo-spatial data sets (i.e. spatial data that use the Earth as a reference system) contain collections of spatial features, which are represented in terms of spatial primitives, such as points, lines, polygons and surfaces. For example, a point might indicate a tree or a specific address. A line could represent a road or a river. Polygons can be used in order to represent a building footprint or administrative boundary. Surfaces are embedded in the 3D space (i.e. they

possess a z-coordinate in order to describe the depth of an object). Spatial data sets may contain not only information about spatial characteristics, such as location and geometry of the objects represented but the spatial relationships between such objects, such as connectivity, distance and orientation. In particular, topology models describe topological relationships (i.e. connectivity and overlapping relationships). Topology models rely on nodes, chains, and polygons to represent relationships between objects [5].

Since spatial features and their relationships can be modeled in several different ways, facilitating interoperability is useful. As a critical part of this, the Open Geospatial Consortium, Inc. ® (OGC) is an international, voluntary consensus standards organization that develops standards for geospatial and location based services. They have defined a conceptual model for a geometry object model, which is independent of the computing platform. The OGC's conceptual model defines Geometry as an abstract class with several subclasses, such as Points, Lines, Linestrings, Linear Rings and Polygons [6]. The conceptual model is now standardized as ISO 19107 [7], however, these are defined only for a 2D space.

Such spatial data sets are primarily available in either vector or raster format. Each data format offers specific advantages to certain tasks. The raster data model for instance is particularly efficient for 3D display and the integration of image data, whereas the vector data model is more adept in applications that require fast retrieval and topological queries. An SIS, thus, must be able to support both data models, in order to facilitate a broad spectrum of applications. Figure 1 illustrates the different representations for raster and vector data.

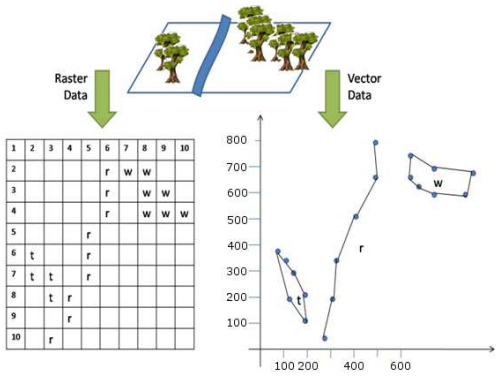


Fig. (1) Raster and vector data

Support for both formats, raster and vector data, are presented in the following two sections. Examples of their representation in an SIS with regard to their geometry and topology are used to illustrate the concept of modeling spatial data.

Vector Data Model

In this section, current technologies for storing, querying and analyzing vector data are presented, including an analysis of the level of support offered by various products with regard to geometry and topology.

Spatial objects are geometrically represented by points, lines, and areas and within a vector format. These primitives are identified through discrete Cartesian x-, y-, and z-coordinates.

The commercial product Oracle Spatial for Oracle 11g stores geometric vector data in the SDO_GEOMETRY data type. SDO_GEOMETRY supports the data types Point, Line String, Polygon (Area), Polygon with a hole, and Collection in both 2D and 3D. Those limited to 2D are Compound Line String and Compound Polygon. The exclusively 3D forms (shown in figure 2) are Composite Surface, Simple Solid, Composite Solid, and Collection [5]. SDO_GEOMETRY is further disaggregated into elements as shown in figure 3.

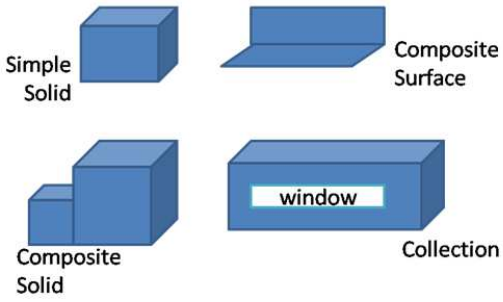


Fig. (2) Oracle 3D geometry types [8]

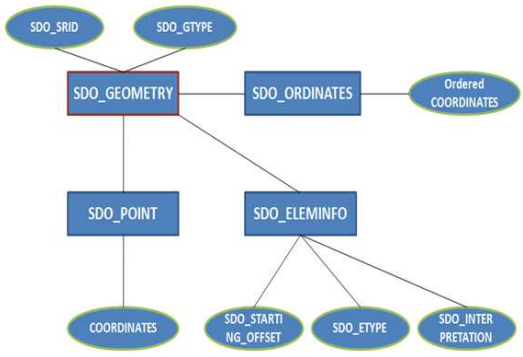


Fig. (3) SDO_Geometry [9]

As shown in figure 3, the SDO_GTYPE specifies the dimension and shape/type of the geometry. SDO_SRID specifies the spatial reference system which can be Geographic3D, Geocentric and Compound or a local coordinate system in case of 3D data.

Geographic 3D specifies latitude and longitude and ellipsoidal height, based on a geodetic datum. Oracle Spatial 11g also offers a topology model as an alternative to the vector data geometry model stored in SDO_GEOMETRY.

A topology model defines relationships between objects. The geometry of an object can be derived from a topological data model, and the topology of objects can be derived from a geometry data model. Consequently, storing only one model and deriving the other from it, if needed, seems at first glance an efficient approach. However, while storing a topological data model results in rapidly executable topological queries, it is hampered by inefficient determination of object geometry. Conversely, storing a geometry model results in efficient computation of geometry queries but results in complex topological queries. Furthermore, spatial features may share boundaries, and a topological data model is more effective for storing shared geometric features. As such, how SDBMS vendors have attempted to solve this issue is of interest. In Oracle Spatial 11g the type SDO_TOPO_GEOMETRY is used to store shared geometric features. Topological features are generally stored as nodes, edges and faces. A node is a point geometry that is shared by one or more features. A node can be unconnected to any other node or connected to one or more edges. An edge is a line-string geometry that connects to nodes. However, this line string may contain other vertices that are not considered as individual nodes and may, thus, contain several line segments. In contrast, a face is a polygonal area that is surrounded by a closed set of edges (ring). A face may contain only one outer ring or that along with a number of inner rings.

Within Oracle Spatial, the topology model also supports hierarchical features in a bottom-up manner (i.e. a new feature layer can be derived from a previous feature layer constructed from the primitive elements, such as nodes, edges and faces). Oracle facilitates this by setting a feature ID within the SDO_TOPO_GEOMETRY constructor. The first feature layer is called a Level-0 feature. The feature layer derived from it is called a Level-1 feature. In general terms, a Level-n feature is derived from a Level-(n-1) feature. Overall, Oracle Spatial is a powerful product for managing spatial content and offers in-depth support and documentation. Figure 4 illustrates how SDO_TOPO_GEOMETRY is structured.

ESRI's ArcGIS geo-database is another useful tool for handling spatial data. The ESRI geo-database ArcGIS offers the geometry types TriangleStrip, TriangleFan, and Multipatch specifically for 3D storing and representation of vector data. Lower dimensional data can be represented through a myriad of geometry types, such as Point, Multipoint, GeometryBag, Line, Ring, Polygon and others; the Multipatch data type is discussed in Section 4.

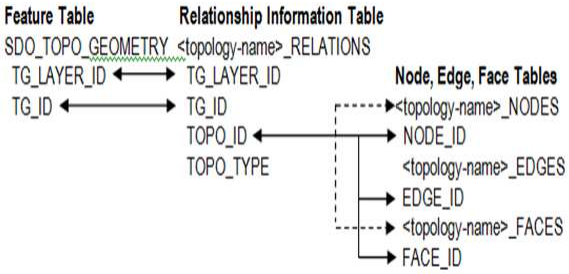


Fig. (4) SDO_TOPO_GEOMETRY [8]

Both Oracle Spatial and ESRI ArcGIS are commercial products. Alternatively, the open source community offers PostGIS, a substantial tool for handling spatial data. PostGIS is an implementation of the OGC Simple Features for the SQL specification [6]. PostGIS follows the same approach as Oracle, which is to extend a RDBMS with functionalities to manage spatial data. PostGIS is a spatial extender for the open source RDBMS PostgreSQL. PostGIS stores vector data in compliance with the OGC simple features specification [10]. Seven different geographic data types are implemented: POINT, LINESTRING, POLYGON, MULTIPOINT, MULTISTRING, and MULTIPOLYGON (a collection of different polygon objects), and GEOMETRYCOLLECTION (a collection of elements, such as points, lines and polygons). Each can be 3D, and users can mix data from different sources, as each record has its own Spatial Reference ID (SRID) [7]. Additionally, PostgreSQL offers the opportunity to implement custom data types, as an extension to a native data type. The same mechanism theoretically works within PostGIS. However, little has been published on this issue. The challenge would be to register the new data type with the geometry_columns table that is used within PostGIS, in order to locate tables that contain geometry types.

The SIS vendors discussed within this section generally offer good support for vector data with regards to geometry and topology of spatial data. The following section discusses vendors' support for the raster data model.

Raster Data Model

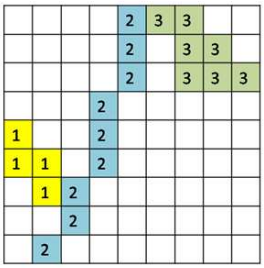
A raster data model associates collections of cells to spatial entities by making a discrete approximation of spatial features into grid cells. In a geo-referenced raster, every cell represents a specific area on the ground. Common examples of raster data objects are satellite images. Within a raster data model, point primitives are represented through single cells within a grid. A line

primitive is a string of cells with common values. A polygon primitive is represented by groups of cells with common values, and surface primitives are cells that represent an elevation. Figure 1 illustrates the basic concept of raster data in comparison with vector data.

The ESRI ArcGIS geo-database uses a native raster format for storing spatial data in a grid [11]. In ArcGIS a grid is constructed from a set of square cells. Each cell is called a tile. A tile is further subdivided into a grid of rectangular cells, known as blocks. The block contains the actual information in columns and rows and is stored in a file system. ArcGIS offers two types of grids: integer grids for representing discrete data and floating-point grids for representing continuous data. Discrete data is sometimes also called categorical or thematic. A discrete object has known and definable boundaries. Discrete objects include buildings and roads. Continuous data is also known as field or surface data. Continuous data represents a location as a measure of the concentration level (e.g. density of noise or pollution in a certain area or the location's relationship from a fixed point). Fix points include elevation, such as sea level and aspect (e.g. north, south, east and west).

ESRI has further developed their geospatial structure by moving away from the simple shapefile and file based ArcInfo Coverages and Grids to the Geodatabase structure. The Geodatabase structure leverages the power of relational databases and the tools available to create linkages between spatial and non-spatial data, such as property records or environmental data. The implementation of ArcGIS Geodatabase systems is a layered system with the personal geodatabase leveraging Microsoft's Access database, which is limited to a single user editing a database of no more than 2GB. The file-based, geo-database developed by ESRI supports TINs in its terrain dataset. File size is also limited but in this case to 1TB. The database size on the other hand has no size restrictions. There is also support for the personal SDE database in ArcEditor and ArcInfo, which uses Microsoft's SQL Express database for workgroup spatial database support. Enterprise implementation of the Geodatabase is achieved by employing ArcGIS Spatial Data Engine (SDE), which supports spatial databases on Oracle Spatial, IBM DB2, Informix, and Microsoft's SQL Server. Support for 3D features in the geo-database has evolved to presently support 3D points, lines, polygons, and multipatches (see Section 4). There is now also support for 3D grid/raster features, and TINs have been incorporated into the geodatabase as terrain features.

As discussed earlier, the ArcGIS grid is stored in tables and files, which include the unique value for each cell in the grid that stores its attributes in a value attribute table (VAT). A VAT is comprised of one record for each unique value in the grid. The VAT contains three default columns at creation that are immutable: object ID (OID), VALUE and COUNT. OID is a unique object identifier number for each row in the table. VALUE is a list of each unique cell value in the raster data sets, as an integer value. Finally, COUNT is the amount of VALUE fields contained within the grid. For example, if there are ten cells that represent a lake, and the value for lake was 1, then the VAT would represent this by setting VALUE=1 and COUNT=10 for each of the ten cells. Figure 5 illustrates how a grid translates into a VAT structure reconsidering the example from figure 1.



OID	VALUE	COUNT	TYPE	AREA	CODE
0	1	4	Single Trees	4500	ST010
1	2	9	River	9900	RV001
2	3	7	Forest Land	8100	FL301

Fig. (5) VAT table for ESRI grid

In addition to the VAT, a separate table contains information about the grid boundaries, named the BND table. A header file (extension .hdr) contains information about the grid cells themselves, such as size and type. While the STA table contains statistical information about the grid, such as mean and standard deviation. Two tile files store the data and the index of the first tile in a grid. Tiles are variable-length binary files. The log file, on the other hand, is an ASCII file that contains information about alterations performed on the grid.

In Oracle Spatial 11g, raster data is stored in the SDO_GEORASTER data type, which represents an n-dimensional matrix of cells. The SDO_GEORASTER consists of RASTERTYPE, SPATIALEXTENT, RASTERDATATABLE, RASTERID and METADATA. Among other things, the RASTERTYPE specifies the dimension of the data, which currently only supports up to two dimensions.

The previous sections outlined how vector and raster data can be employed to model spatial data by relating to its geometry and topology. Along with vector and raster data models, digital elevation models (DEM) are often presented as a third data model. Technically,

however, DEMs can be represented in raster and vector format and are, thus, discussed separately below.

DIGITAL ELEVATION MODEL

DEMs are one of the most commonly used data sets in the area of spatial research. They are particularly popular for visualization of 3D content. DEMs are digital representations of surfaces. A DEM forms the basis for a DTM or a Digital Surface Model (DSM). A DSM contains both location and elevation information, as well as meta data information about urban features. A DTM is generated by digitally removing all of the urban features within the DSM, in order to expose the underlying terrain. A DEM is usually a raster model (regular spaced grids) or a triangular irregular network (TIN). Each cell within the raster data model has a value that corresponds to its elevation.

DEM data sets are commonly collected using remote sensing technologies, such as interferometric synthetic aperture radar, where two passes of a radar satellite produce a DEM with a resolution of approximately ten meters. DEMs can also be generated by using digital image correlation, where two optical images that are taken from different angles are correlated [12].

Data collection for DEMs

Recently, data collection through light detection and ranging (LiDAR) technology has gained increasing popularity for serving as input data, to generate DEMs. LiDAR data is acquired by employing a pulsed laser device, to record the distance from the camera to each point in an image [13]. The quality of a DEM is significantly determined by the roughness of the terrain, the sampling density that is determined by the data collection method, the resolution of the grid, the choice of interpolation algorithm, the vertical resolution, and the choice of terrain analysis algorithm. Figure 6 presents a DEM of a part of Dublin Ireland's city center that roughly comprises Trinity College.

Technology such as LiDAR challenges SDBMSs, with respect to the sheer volume of data points referred to collectively as a *point cloud*. Point clouds raise the question of how to allocate these sets of points to feature types within a database. Additionally, point cloud data are collected by a scanner and then transferred in a format that is proprietary to the scanner manufacturer. In the following paragraphs several patents will be presented that address these issues.

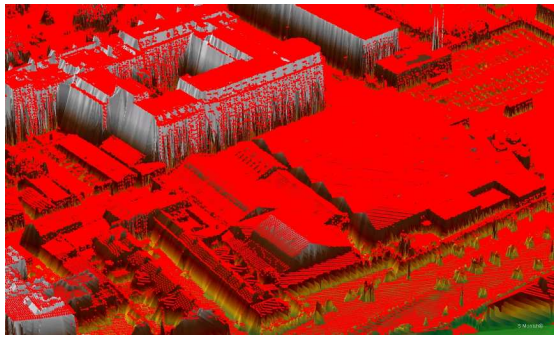


Fig. (6) DEM of a portion Dublin Ireland's city center

Patent US7065461 solves the problem of handling large data sets by providing a user interface through which the user selects a point cloud area, which is to be fitted into an "object". A circle is created around an area of point clouds and points that are to be included are determined through a statistical method, such as least squares [14].

Patent US7117116 also proposes a mechanism to automatically bound point clouds. In this case, a mesh point cloud module identifies and segregates only visible patches of a point cloud. With this approach, data at the edges of objects might be accidentally omitted. The patent, therefore, proposes a mechanism to generate additional data points, in order to compensate for this effect [15].

Patent application US20070257908, on the other hand, tackles the issue of proprietary data formats from scanner manufacturers. This system first determines, if the input file is in text or binary format. In case the source file is in a text format, it is routed directly into a parser that loads the point cloud into a data structure provided by a database. If the source file is in binary format, the innovation first transforms the data into a readable format [16].

Of the SDBMS vendors however, only Oracle Spatial offers a built in data type for point cloud storage called SDO_PC. Meta data associated with the point cloud is stored in a base table, whereas the actual point cloud data is stored in a different table. Individual points within the point cloud are divided into subsets and then loaded into multiple rows, with the points stored as a BLOB data type [17]. Oracle Spatial offers further processing of the point cloud data into a TIN as to be discussed in section 3.3.

Creation of a DEM

The other DEM format is the raster, which is a regular arrangement of pixel cells that are stored as a matrix. It can be used for a systematic analysis of the relationship between locations and their properties [18]. An example

is the calculation of minimum, maximum, and average values.

Several challenges are associated with the generation of DEMs. The main one is how to extract a DEM automatically from the point cloud. Regions that have no clear boundaries, such as coastal sections and areas adjacent to rivers appear particularly prone to errors. Additionally, automatic DEM generation appears to be problematic, when distinguishing between actual buildings and material that is only covering the building, such as vegetation. In terms of visualizing DEMs, an interesting challenge is how temporal changes within a particular area can be represented in a meaningful manner. The following inventions strive to contribute solutions for the aforementioned difficulties for DEMs.

Patent US6748121 provides an intelligent mechanism for automatic extraction of digital elevation data [19], as conventional methods have falsely created lands near coasts and rivers. This innovation works in three steps. First, the Center-of-Gravity (COG) also known as the Empty-Center-Index (ECI) is eliminated from the result of a conventional DEM generating method, such as nearest neighbor, spline, or moving window average. The COG or ECI is an artificial elevation that is generated from edges by conventional interpolation methods. In the next step, a hole-filling segmentation evaluates, whether the previous elimination should be reconstructed. The decision is based on the segment size. The last step removes noise from the initially interpolated area. The resulting DEM is particularly accurate with regards to open areas of coastal regions and around rivers.

Patent US7298891 [20], WO/2006/019595 [21] and EP1779291 [22] were all filed by the Harris Corporation and present a method for automatic extraction of a DEM from raw topographical points. The invention relies on two filtering steps of the volumetric input data. In the first step, the ground is estimated by filtering ground points from aboveground obstructions. In the next step, the ground points are filtered in order to construct a multi-dimensional shell of DEM points.

Another challenge for DEMs is to distinguish buildings from other objects, such as foliage. Patent US7191066 [23], EP185186 [24] and WO/2006/086252 [25] by Rahmes et al. propose a methodology for determining whether a certain object within the DEM is a building or foliage. This invention first establishes a "perimeter versus area parameter" for each individual object within the DEM. This value is then used to classify objects as either buildings or foliage. In the next step, the objects classified as foliage are compared with regard to their height value versus a height threshold. If the height value is greater than the threshold, the object is reclassified as a building. The following step examines all objects that were classified as a building within the first iteration. In this a "perimeter versus area parameter" evaluation is applied to this set of data. If the "perimeter versus area parameter" is greater than the threshold, the object is reclassified as foliage. The last step generates two different DEMs: one for buildings and one for foliage.

Visualization of surface structures is particularly interesting, if the surface is examined over a period of time and the identified changes are to be visualized. Patent WO01/26059 [26] presents a method of producing a survey animated digital model. Vertical stereoscopic photography is used as an input and is digitized. In the next step, the images are merged, and the vector data is extracted, which results in a 3D map of the area. The following step enriches the model with additional data, such as trees, hedges, buildings, and artificial boundaries. The subsequent step uses ground level and string feature data attributes, as well as the ground point and breakline data to generate the DTM. The final step enriches the model with orthophoto data and micro relief enhancement feature data, which results in an animated digital model.

DEMs often use 2.5D visualization. This designation of 2.5D is shorthand for 2.cD, with c denoting the volume filling capacity of a topographic surface. The dimension must lie between 2D (plane) and 3D (solid object) due to the fact that a 2D surface, such as the Earth, is textured with GIS data. This is generally referred to as draping [18, 27].

In general, 3D data models can be classified into three categories:

1. Surface based models, such as 3D Formal Data Structure (3D FDS) and Boundary-Representation (B-Rep).

2. Volume based models, such as Constructive Solid Geometry (CSG) and Tetrahedral Network.

3. Hybrids of the former two types, such as Octree-TEN and TIN-Octree.

The following section presents TINs and an advancement of TINs called TENs.

Triangular irregular network

SISs mainly support TINS, the vectorial representation of DEMs. A TIN is vector-based digital, geographic data created by triangulating a set of vertices [9] that are usually provided by a DEM. A TIN is also a network of vertices, the so called mass points. Mass points

each have coordinates in 3D and are connected via edges to generate a triangular tessellation. A TIN is constituted of irregularly distributed nodes and lines with 3D coordinates (x,y,z) that are arranged in a network of non-overlapping triangles. The main difference between a raster DEM and a TIN lies in the distribution of points. In a raster, DEM points are arranged regularly, whereas in a TIN, an algorithm determines the necessary points for terrain representation. Consequently, with a TIN fewer points need to be stored in a database than with a DEM [28]. Integrated TINs take this one step further and incorporate feature data into the tinning process [29]. Like the DEM, a TIN offers support for 2.5D. A TIN is typically constructed using a form of Delaunay triangulation, which generates triangles that are as equiangular as possible, in order to avoid long and thin triangles, because they are particularly unfavorable for approximation problems. Three-dimensional visualization of TIN data is readily generated by rendering its triangular facets. Figure 7 illustrates a TIN of Trinity College Dublin's square.

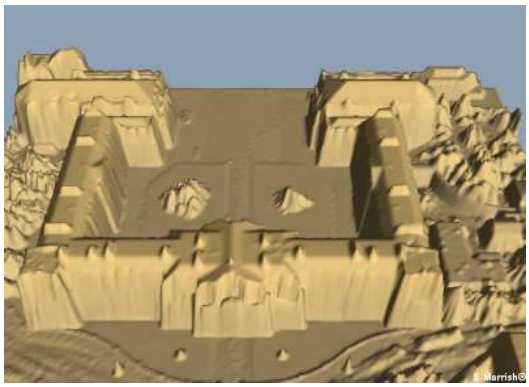


Fig.(7) TIN of Trinity College Dublin

Topological queries, such as overlap, are extensively implemented for the 2D case, where two or more planar partitions intersect. However, implementing the intersections of 3D volumetric partitions is more complex. A promising approach is TEN, which is basically a true 3D TIN. In a TEN the Delaunay triangulation is extended to another point in order to construct a 3D object. Features are represented by their boundaries through a TIN and added one after the other into the TEN [30]. Overlap queries are, thus, supported by the TEN's internal neighborhood search. TINs are typically used in order to represent 2.5D elevations of a surface, whereas TENs triangulate 3D volumetric objects through tetrahedrons.

Oracle 11g offers the SDO_TIN_PKG for creation and querying of TINs. Oracle will create a Delaunay TIN, if no constraints are specified. Oracle relies on two tables in order to store the TIN data. A "base table" that contains a column with the SDO_TIN type stores the meta data associated with the TIN. The actual points are stored in blocks within another table that is commonly referred to as the "block table". The block table stores both the point information and the triangles' information in a BLOB column. There is no upper limit in the number of points and triangles that can be stored in Oracle 11g. Oracle also offers an automatic clean-up of the block table, if a TIN object is deleted or a base table is truncated. A TIN can be queried by specifying a so called query window with the SDO_TIN_PKG.CLIP_TIN function, which takes an SDO_TIN object, an SDO_GEOMETRY as a query window, and several other optional parameters and returns a new block table as a result of the query. The query, thus, accesses only relevant blocks. Other queries include retrieving the triangles in each block as a collection of SDO_GEOMETRY objects, and retrieving the IDs of the points returned by a query [17]. Additionally, a custom built TIN can be generated, which is particularly useful in cases where a coarser resolution TIN is to be retrieved from a given TIN. Oracle does not offer in-built support for coarser TINs, however, they can be generated manually and then associated with the original TIN.

The ESRI ArcGIS geo-database also supports the creation of TINs. Moreover, the latest version ArcGIS 9.3 includes Terrain Feature Classes, which store a hierarchy of TINs for different map scales. Another option is their Z Feature Class, which stores elevation data in a z-value for each vertex in 2D polygons. ESRI's multipatch feature class stores a 3D geometry that is constructed of planar rings and triangles; this feature class is discussed further in Section 4. In ESRI's ArcGIS, however, a TIN is stored as a directory of binary files [11].

PostGIS does not natively support the creation or storage of a TIN data type. Instead a patch from X3D creates a serialized mesh [31], similar to TINs. However, this mesh is not as powerful as a single, large serialized mesh, and it cannot handle large, region-spanning TINs with millions of faces. Another approach would be to implement a relational TIN model on the primitives defined by the Simple Feature Specification [7]. However, this does not solve the issue of loading the TIN into the database. Additionally, it still leaves the task of providing useful operators on the TIN to each individual programmer.

The main advantage of TIN models over other approaches is their inherent multi-resolution capability. They are able to resolve fine regions and sudden changes on surfaces. Abrupt changes are often present in urban areas, where the interpolation of a raster DEM is not suitable to represent the abrupt changes in height caused by buildings and other urban objects. Patent WO/2004/097574 for instance uses this characteristic to generate a variable resolution model with an indexing

function that indicates the impact of data in a model [32]. The input can either be general raster data or a DEM. The output can be a TIN or a Finite Elements Method mesh. From the original data, first an index function is selected and then applied to the data. The following step sorts data into "bins" based on the indexing function. The subsequent step selects data from the bins according to a selection function. The output is a FEM mesh.

So far, terrain surfaces have been discussed, but they do not provide support for true 3D spatial objects. The following sections discuss advances within the area of true 3D feature types within SDBMS.

MULTIPATCH

The ESRI geo-database models support 3D objects in a feature class called multipatch, which is just another geometry type in the ESRI database. Multipatch is constructed much like the OpenGL 3D primitive triangle, in that it is constructed of strips and fans and defines an object's boundaries through triangular faces. Multipatch can be created using ESRI's ArcObjects from raw source data or from existing geometries. Raw data are often provided by ASCII text files that contain a sequence of x-, y-, and z-coordinates that can be loaded into ESRI's ArcCatalog. Within the geometry definition, the geometry type must have the attribute "esriGeometryMultipatch" set in order to properly consider the z-coordinate for the third dimension. After creating the feature class, it can be populated with data in the form of either "triangle strips" or "triangle fans". Each is, in essence, a collection of points (IPointCollection in ESRI), where each point is added to the point collection, until the "triangle strip" or "triangle fan" is complete. Figure 8 illustrates an example of a multipatch object.

Fig. (8) Multipatch [11]

For the computation of normals on faces in ArcObjects, the points in a triangle must be arranged clockwise. This way ArcObjects can determine which side is the outside of a face. Once the IPointCollection is filled with all points, the collection is added to the geometry collection object, which is basically the multipatch feature. One of the main limitations of multipatch is its size in the database. ESRI notes that 1.5km of a pipeline can be stored as a simple line feature using less than 1MB of memory, whereas the same object may require up to 100MB, when represented as a multipatch object [17].

Multipatches are designed to represent 3D volume objects. Some natural occurrences might have undefined volume boundaries. Patent US6839632 [33] proposes a method for constructing a 3D polygonal model of a 3D irregular volume using ESRI's geo-database and the multipatch feature. This was developed to represent irregular volumes, such as natural fields, where boundaries are not fully specified. The resulting representation allows a user to "visualize the geometric and attribute relationships" between the irregular 3D bodies. First, a solid 3D irregular volume is modeled within the ArcGIS. Within this step, at least one 2D polygon is identified that serves as a boundary of the 3D irregular volume. Additionally, the top and bottom faces of the volume are estimated. Then, the multipatches are constructed of a network of triangular panels for top surface, bottom surface, and the sides. A more flexible manner of describing 3D objects within an SIS are freeform curves and surfaces, which are mathematical constructs and are discussed in the following section.

FREEFORM CURVES AND SURFACES

Freeform curves and surfaces are widely used within Computer Aided Design (CAD) applications in order to visualize surfaces. For instance, patent US5237647 utilizes sensors in order to capture an object and then employs freeform lines in order to generate a representation [34].

General freeform curves and surfaces are defined through several attributes, among which are control points and certain vectors. Bézier, B-spline, and NURBS are generally used in order to model freeform curves, of which NURBS is the most general form [35]. This means that B-spline is a special case of NURBS, and Bézier is a special form of B-spline. Figure 9 illustrates a cubic Bézier curve, which requires four control points. A B-Spline also requires a knot vector. In addition to these parameters, NURBS also requires weight values. A freeform surface on the other hand only requires a knot vector, degree, and u- and v-vectors [10]. While figure 9 illustrates a Bézier curve with four control points, figure 10 illustrates a Bézier surface which is based on 16 points.

RDBMS' capability to extend data types and operations can be exploited in order to represent features via freeform curves. Pu, for instance, used freeform curves and surfaces to construct a 3D data type in Oracle Spatial 10g [9]. For each free form curve, data types are created individually. Another approach might have been to create a data type just for NURBS and then to describe B-spline and Bézier curves from the NURBS

representation, as B-spline and Bézier are special cases of NURBS. Pu [9] argues that leaving empty values for some of the parameters would decrease system efficiency. Additionally, the OGC spatial schema recommends distinct data types for different shapes. Furthermore, some geometry algorithms differ among curves. Irrespective of these issues, there are two possible ways of implementing the new type in Oracle Spatial. One option is to extend the SDO_GEOMETRY type. The attribute of its SDO_GTYPE has still a free range of IDs available for storing new data types. The new type and the geometry type are stored within the same data type and internally stored within the same table. Consequently, spatial operations like spatial data insertion, spatial querying, and spatial indexing are natively supported within Oracle Spatial. Moreover, the new type is particularly easy to implement, as the only steps required are to assign a new number to the SDO_GTYPE and to set up rules for the SDO_ELEM_INFO field. On the other hand, storing new data types in SDO_GEOMETRY results in the storage of significant amounts of redundant data. Additionally, none of the existing spatial functions support freeform data. Geometric functions for simple data types are defined differently for 2D and 3D objects and, thus, require different mathematical algorithms for operations such as insertion and length. Pu, however, followed a different approach, the one recommended by the OGC of implementing a separate data type for each identifiable geometry (a B-Spline data type for a B-spline geometry, etc.). Due to the fact that Pu's implementation is not based on the SDO_GEOMETRY type within Oracle Spatial, implementation should be exportable with little effort to create custom data types in other spatial DBMS, such as PostGIS.

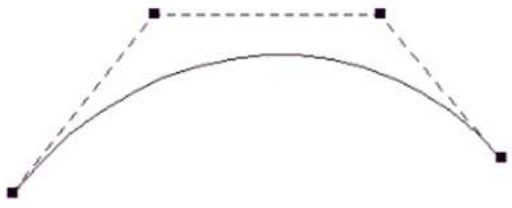


Fig. (9) Bézier curve with 4 control points [9]

Large data sets in particular need to be managed efficiently in order to allow for queries to be processed in a timely manner. In 2D, several standards have been implemented, with R-trees [3] being among the most popular approaches. Adapting these techniques to the third dimension is, however, not trivial. The following section presents current advances with regard to indexing spatial data in SISs.

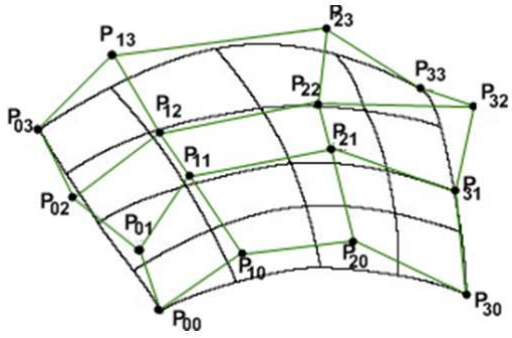


Fig. (10) Bézier surface with 16 control points [9]

INDEXING

Indexing in a database is used to speed up operations. A spatial index organizes the space and the objects within this space in a particular manner, so that a spatial query or a spatial operator does not have to traverse through the complete table to retrieve specific data. SDBMS vendors typically offer two types of spatial indexes: quadtrees [36] and R-trees [3]. There are several derivates of these, however they are not implemented presently within SDBMS. R-trees seem particularly popular with SDBMS vendors, as most indexes are either based on R-trees or use R-trees directly through a dedicated data structure. Alternatively they map spatial objects into one-dimensional space in order to use a standard index, such as a B-tree [37]. In 2D, an R-tree is constructed by enclosing an object into a minimum bounding rectangle (MBR) [3]. Theoretically, it should be trivial to extend a rectangle into 3D by enclosing an object by a box. Another approach to elevate a 2D spatial index to 3D is the development of octree, which is based on a quadtree structure [36]. In this data structure each node can have up to four child nodes and by doing so decomposes the space into 2D cells. Contrary to the quadtree, each node in an octree can have up to eight child nodes and, thus, divides the space not into 2D cells but into 3D cubes. However, this approach is not implemented presently within commercial systems.

Indexing is implemented differently by particular vendors, which might be a function of historical product development, where the current spatial index has evolved out of an existing technology. PostgreSQL, for instance, supports three indexing structures: B-tree for data that can be sorted along one axis, R-tree for spatial data which is then broken up into rectangles, sub-rectangles and sub-subrectangles, and the Generalized Search Tree (GiST) index, a "template data structure for abstract data types" that offers more robust support for spatial indexing than the PostgreSQL R-tree implementation [38]. GiST is a template for implementing other indexing methods, such as B-tree and R-tree, and is a balanced tree structure that contains <key, pointer> pairs. The key is a member of a user-defined class. It

represents an attribute that is valid for all items that the pointer element can reach. A key in an R-tree like GiST refers to a bounding box. For instance: *all items that the pointer reaches are in Ireland*. PostGIS consequently offers an R-tree index on top of GiST [39]. Figure 11 illustrates the concept of a GiST implementation for data access methods.

Compared to a normal R-tree index, a GiST index is "null save" (i.e. GiST can index columns that contain *null* values). In addition to this, PostgeSQL allows a page size of 8K; R-trees fail when trying to index GIS data that exceeds 8K. As a consequence of this, GiST supports "lossiness", which means that only important parts of an object (i.e. the bounding boxes) are stored in the index [39]. MS SQL also works with a limit of 8K for page sizes (they are called *blocks* in Oracle), whereas Oracle offers a variable page size of 2, 4, 8 or 16K.

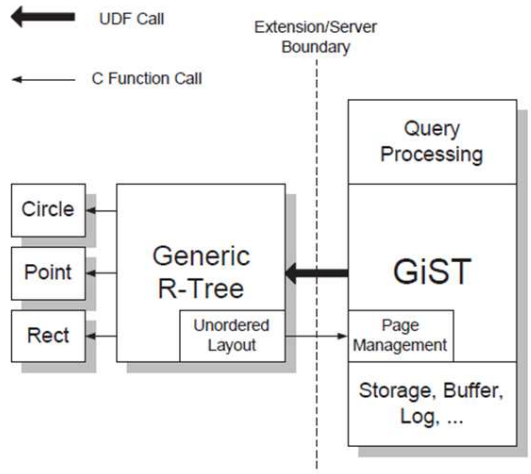


Fig. (11) Example Implementation of GiST [40]

Oracle Spatial 11g also provides a spatial index on the basis of an R-tree and a partitioning function for logical tables, which includes their spatial indexes. Partitioning delivers significant performance and manageability advantages. Additionally, the creation of a spatial index can be performed in parallel and spatial queries themselves can be performed in parallel. This is particularly useful for "nearest neighbor", "within distance", and "relate" spatial queries [8].

ESRI's ArcGIS geo-database offers a spatial index on their shapefile, as well as the ArcSDE geo-database. In contrast to the previously presented spatial databases, ArcGIS does not use a tree-structure for the storage of a spatial index, but instead uses a grid [11]. ArcGIS determines automatically which grid size is appropriate for a new feature class that has been generated and filled with data. If new features are added to an existing feature class, the index's grid size is not automatically computed, but has to be set by the user. The main advantage of a grid-based spatial index over a tree structure is that since the spatial index structure can be created first, the added data does not require any changes to the index structure. On the other hand, a tree structure might be more efficient as it is tied to the internal data storage structure [29]. Of notes is the fact that the Grid spatial index employed in ArcGIS does not support 3D aspects.

In most cases, indexes only support two dimensionality with simple 3D extensions [40]. Efficient querying of a spatial database however, requires a true 3D spatial index. A major challenge is processing the range of geometries that may need to be stored within the database, without significant efficency losses. For instance, polygons that contain a multitude of vertices and span a wide area may significantly slow down querying operations. This might be solved through clipping polygons before applying an index [39]. The efficiency of grid index techniques depends on the efficient determination of the size of each cell in the grid. Furthermore, storage becomes an issue in spatial indexes, as they generally become quite large.

Patent application US20080133469 offers an improvement of a spatial grid index by determining the optimum grid cell size [41]. In particular, this invention improves the grid indexing process that locates the minimum bounding rectangle (MBR) and the associated geometric shape. First, an index performance evaluator was developed called the "Ne". The "Ne" evaluates the grid performance and is based on statistical data of the grid and is used to evaluate the approach. The indexing mechanism works on a per level basis through the grid structure. The grid resembles a cube. Firstly, it determines whether more than four grid cells overlap any geometric shape. If yes, then the appropriate grid cell size is determined by analyzing information for each grid level by assessing the following information:

- Geometric shapes and the number of grid cells that overlap
- Average size of the geometric shape
- Number of index entries/geometric shapes
- Threshold to determine when a new indexing level should be used

The procedure then is to filter out level i-1 index entries that are within the threshold. Next, a set of consolidation entries for the number of geometric shapes that overlap with the same number of grid cells must be determined. These consolidation entries are sorted in descending order according to the number of geometric shapes. From this, a consolidation entry based on the maximum number of geometric shapes that overlap with more than the threshold grid cells can be derived. The resulting grid cell size for that level "i" is the grid cell size of the maximum consolidation entry. The Ne shows that this procedure results in better performance for spatial search queries than previous methods.

Patent application US20080133559 also aims at improving spatial grid indexing [42]. This invention reduces the actual number of indexes in a grid index. A pool storage area is established, and a threshold is determined that regulates how many grid cells a shape may overlap. If the threshold number is not exceeded, the geometric shape is stored within the grid index. However, if the geometric shape exceeds the threshold number, it is saved within a "pool storage area". This "interim area" is examined further in the next step. If a geometric shape in this step overlaps more than a predefined number of grid cells, the shape is then overlaid with a coarser level grid. If it does not exceed the number, this grid level is used for indexing. If after application of a coarser level grid, the geometric shape still overlaps a certain number of grid cells after two iterations of this procedure, the geometric shape is then stored in a general pool. If it does not overlap the maximum number of grid cells at this stage, this level of coarseness is used for indexing.

Patent US6463180 suggests two methods to reduce storage of spatial data that is represented through an R-tree structure [43]. The first encodes the relationship between parent and child nodes and stores this relationship in a file. The other method uses a pointer-less preorder traversal. With this, each node's spatial extent is encoded with respect to the parent node's extent. The encoded spatial identifier therein contains at least octant overlap.

With regard to producing efficient spatial indexes for multi-dimensional data, several strategies have been developed to increase index efficiencies. However, vendor support for these is limited, and creation of true 3D indexes is still an ongoing research problem.

CURRENT AND FUTURE DEVELOPMENTS

This paper presented developments on the support of 3D data types in SIS. DEMs and TINs provide 2.5D support, which are often employed for the representation of surfaces. Native support for TINs is provided by Oracle spatial 11g, as well as by the ESRI ArcGIS geo-database but not PostGIS. Several advances in the area of DEMs have been presented in the form of recent patents that address the issue of automatic generation of DEMs for difficult areas, such as coastal regions and urban areas.

LiDAR point cloud data are increasingly used as input for DEMs due to the ease of acquiring highly accurate and fast survey information, but they pose significant challenges for SISs due to the need to assign various portions of the vast data to feature types within spatial databases, along with their subsequent hosting and querying. Recent patents provide some initial solutions.

Support for 3D has been greatly improved by SIS vendors who now offer 3D data types, such as ESRI's multipatch and 3D volumetric data types in Oracle, such as simple solids. Due to the extensibility of SISs, it is possible to implement new data types including those based on freeform curves.

In order to query these new data types spatial indexing mechanisms supporting 3D data are necessary. True 3D indexing is still an emerging area of research within SISs. Approaches that are currently used by commercial as well as open source products have been presented. For example R-tree is the most popular indexing technology for spatial data. However, indexes can become quite large. While several patented innovations for efficient storage of indexes have been presented, little has been documented on the support for indexing for true 3D data types. Innovation in this area is likely to increase as most SDBMS vendors are in need of efficient storage for querying of the new 3D data types.

In the near future, it is highly likely that more SIS vendors will incorporate 3D functionality into their products. Additionally, it can be expected that current technology will evolve towards offering greater flexibility. For instance, it is not possible currently to update a TIN within Oracle Spatial 11g once it has been created [17]. Research within this area can be expected soon to allow more dynamic usage of 2.5D and 3D technology.

ACKNOWLEDGEMENTS

This work was generously support by Ireland's National Digital Research Centre's grant EoI/0701/008 "Hosting and analysis capabilities for 3D LiDAR point cloud data", EoI/0701/008. Data for some of this work was provided by Science Foundation Ireland's sponsored grant 05/PICA/I830 GUILD: Generating Urban Infrastructures from LIDAR Data.

REFERENCES

[1]. Sheckhar S, Chawla S. Spatial databases - a tour, Prentice Hall, 2003.
[2]. Ford A. ESRI ArcUser. January - March 2007.
[3]. Guttmann A., R-tree: A dynamic index structure for spatial searching, *SIGMOD Conference*, ACM Press, 1984.
[4]. Bruening M, Zlatanova S. 3D Geo-DBMS. Directions Magazine, 2004.
[5]. Davis B. GIS: A visual approach, One World Press, 1996.
[6]. OGC. OGC Abstract specification. OpenGIS Project Document Number 01-101. [Online] OGC, 2008. [Cited: 15 September 2008.] http://www.opengeospatial.org/standards/as.
[7]. ISO. Geographic information - spatial schema. 2003. ISO 19107.
[8]. Oracle. Oracle Spatial 11g: Advanced Spatial Data Management for Enterprise Applications. [White Paper]. July 2007.

[9]. Pu S. Managing freedom curves and surfaces in a spatial DBMS, The Netherlands : Delft University of Technology, 2005. Master Thesis.
[10]. OGC. OpenGIS® Implementation specification for geographic information - simple feature access - Part 1: Common architecture. [ed.] J.R. Herring. 2006.
[11]. ESRI. ArcGIS Desktop Help 9.2. [Online] 11 June 2008. [Cited: 18 September 2008.] http://webhelp.esri.com/arcgisdesktop/9.2/index.cfm?TopicName=welcome.
[12]. Wilson JP, Gallant JC. Terrain analysis: Principles and applications, John Wiley and Sons, 2000.
[13]. Shiode N. 3D urban models: recent developments in the digital modeling of urban environments in three dimensions. Geo J 2001; 53: 263-269.
[14]. Chang, C.K., Chen, L.H., Wu, X.Y.: US7065461 (**2007**).
[15]. Chang, C.K.: US7117116 (**2006**).
[16]. Chang, C.K., Wu, X.Y.: US20070257908 (**2007**).
[17]. Kothuri, R., Godfrind, A. and Beinat, E. Pro Oracle Spatial for Oracle Database 11g, Apress, 2007.
[18]. Cote P. Digital elevation models. [Online] Harvard University Graduate School of Design. [Cited: 15 September 2008.] http://www.gsd.harvard.edu/gis/manual/dem.
[19]. Kim, S.B., Kim, T.G.: US6748121(**2004**).
[20]. McDowall, T., et al., US7298891 (**2007**)
[21]. McDowall, T., et al., WO06019595 (**2006**)
[22]. McDowall, T., et al., EP1779291 (**2007**)
[23]. Rahmes, M., et al., US7191066 (**2007**)
[24]. Rahmes, M., et al. , EP1851686 (**2007**)
[25]. Rahmes, M., et al. WO06086252 (**2006**)
[26]. Roche, S., O'Neill, K., Shackelton, C.: WO0126059 (**2001**).
[27]. Longley, P.A., et al. Geographic Information Systems and Sciencee. Chichester : Whiley , 2001.
[28]. Franklin, W.R. Triangular irregular network to approximate digital terrain. Electrical, Computer and Systems Engineering Department, Rensselaer Polytechnic Institute. Troy, NY, US : s.n., 1994. Section 2.3 Research Interests.
[29]. Stanzione T, Johnson K. GIS enabled modeling and simulation (GEMS). ESRI. 2007. ESRI UC 2007.
[30]. Pilouk S. A topological model for a 3-dimensional spatial information system. ITC, The Netherlands. 1996. PhD Thesis.
[31]. Courtin O, Jonglez D. PostGIS and X3D. [Presentation for the Free and open source software for geospatial Conference]. 2007.
[32]. Bras, R.L., et al.: WO04097574 (**2004**).
[33]. Grace, J.D.: US6839632 (**2005**).
[34]. Roberts, A.F. et al., US5237647 (**1993**).
[35]. Farin G. Curves and surfaces for computer aided geometric design: A practical guide, Academic Press, 1990.
[36]. Samet H. Hierarchical spatial data structures. *Int Symposium on Advances in Spatial Databases.* 1989; 89: 17-18.
[37]. Bayer R. Binary B-Trees for virtual memory, *ACM SIGFIDET Workshop*. pp. 219-235 San Diego, California, US, 1971..
[38]. Geo-Consortium. Introduction to spatial data management with PostGIS. 2007.
[39]. Ramsey P. PostGIS Manual. [Online] Refractions. [Cited: 15 September 2008.] http://postgis.refractions.net/documentation/.
[40]. Arens C, Stoter J, van Oosterom P. Modeling 3D spatial objects in a geo-DBMS using a 3D primitive. J Comput Geosci 2005; 31: 165-177.
[41]. Chen, Y., Rao, F. Stolze, K.: US20080133469 (**2008**).
[42]. Adler, D.W., Stolze, K.: US20080133559 (**2008**).
[43]. Krishnaswamy, R.P.: US6463180 (**2002**)