3D GIS for outdoor AR applications

Siyka Zlatanova and Frank van den Heuvel
Department of Geodesy, Delft University of technology
Thijsseweg 11, 2629 JA, Delft, The Netherlands
{S.Zlatanova, F.A.vandenHeuvel}@citg.tudelft.nl

1 Introduction

The architecture and the role of Geographic Information Systems (GIS) are rapidly changing. Two strong tendencies can be observed in the GIS developments: 1) Database Management Systems (DBMS) offer integrated maintenance of geometric and semantic information (Oosterom et al., 2002, Zlatanova et al., 2002) and 2) the performance of DBMS has been significantly improved (Quak et al., 2002). This makes possible an increasing number of real-time applications to make use of the real-world data organised in GIS. Outdoors mobile augmented reality (AR) systems are a typical example of such an application. An outdoor AR system (e.g. for investigation of hazardous environments, visualisation of invisible objects, shopping, tourism) may need a 3D model of size comparable to one town, i.e. thousands of houses, streets, parking lots, etc and can greatly benefit of 3DGISs.

Within the UbiCom project (http://bscw.ubicom.tudelft.nl), we have investigated the current status of 3D GIS with respect to efficient data organisation and appropriate performance that suffices real-time applications. It should be noted that besides providing user-requested information (e.g. position of underground cables, owner of a building, etc.), 3D GIS plays a critical role in the AR setup within the UbiCom project. First, the 3D GIS is to be used for the accurate positioning of the mobile unit. The pose determination in the UbiCom system is based on a vision system (Persa and Jonker, 1999). The GPS included in the mobile equipment provides an initial approximate positioning (2-10m) that is basically insufficient. The accurate positioning has to be achieved by matching lines extracted from video streams (obtained from the video camera of the mobile unit) and lines retrieved from the 3D GIS. Furthermore, the 3D GIS has to be able to provide the needed subset of data within several seconds (it is expected that the AR system will be able to track the user position for few minutes without reference to the database). Second, the rendering subsystem (for visualisation of virtual objects) needs specific data about the position and the shape of the physical objects in the field of view, i.e. those objects that can occlude the virtual objects (Pasman et al., 1999). This requires certain accuracy and consistent description of outlines of 3D objects (e.g. man-made objects such as buildings).

To our experience, the augmented reality system making use of large 3D GIS data is still lacking. Many vision systems have been currently developed but most commonly they operate only in office-like environments that do not require large 3D models. *RobiVision* (Zillich et al., 2000) is one of the projects aiming at the utilisation of rather large 3D model (a model of the indoor space of a ship) but the model is still a CAD model. *MARS* (Hollerer et al., 2001) is a project intended for outdoor application but again the positioning does not require access to real GIS. Utilisation of 2D GIS for visualisation of cables has been reported by Roberts et al., 2002. The positioning is based on a high precision kinematics GPS and an internal navigation system enabling centimetre accuracy.

Here, we present our results and observations on the reconstruction of the 3D model (with respect to the requirements mentioned above) and the different possibilities for data structuring in DBMS (i.e. Oracle). It should not be forgotten that these two aspects of 3DGIS, i.e. data collection and data structuring, are closely related and their mutual consideration can increase the efficiency and the performance of the entire AR system.

2 3D object reconstruction

The cost to produce a 3D model is usually much higher than 2D map production and benefits from a certain level of automation. Usually, different approaches and techniques are used to reconstruct terrain objects (as continuous surfaces) and 3D objects on top of it. To provide the 3D model with respect to the requirements of the UbiCom project, we selected a specific manner to represent and organise the features in the DBMS (Zlatanova and Verbree, 2000). Since the details on the facades are to be used only by the tracking system (and not for analysis), we decided to represent them as “loose” lines. For well visible physical objects, we maintain topology in order to ensure consistency.

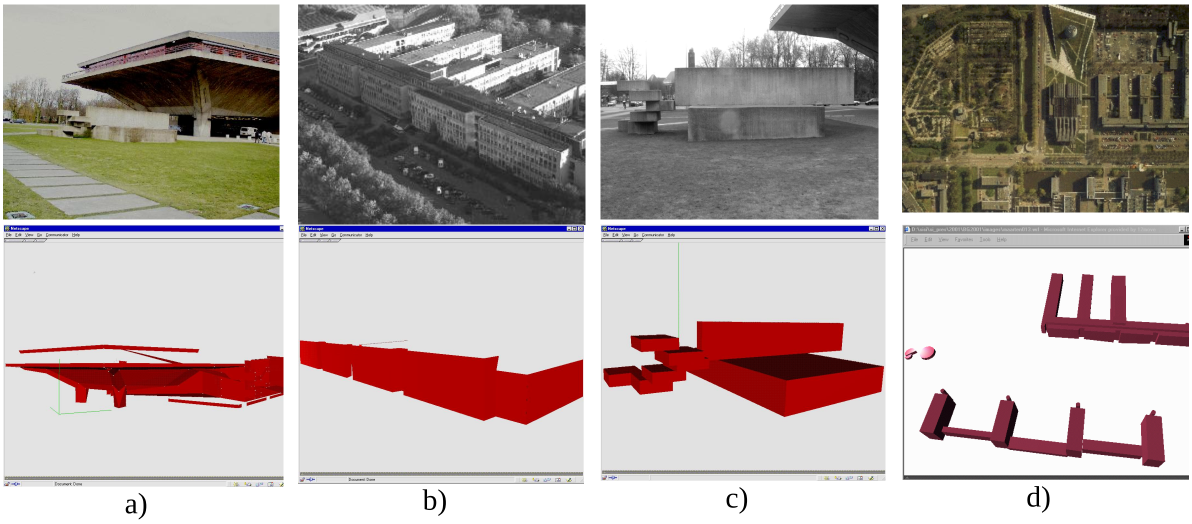


Figure 1: Images and reconstructed models of the buildings of: a) the Aula, b) the faculty of Applied Physics, c) the Art d) the Post Office and faculty of Mechanics

2.1 Phase 1: 3D reconstruction of topologically structured models

Manual reconstruction: Our experimental buildings are reconstructed in *PhotoModeler* (EOS Systems, Canada) using about 100 images taken from ground level with a digital camera Kodak DCS420 (black and white, resolution 1524x1012, focal length 20 mm). The output of the reconstruction is a 3D model in a model coordinate system. The final co-ordinates of the 3D model in the national geodetic system are obtained by an integral least-square adjustment of all the measurements (from terrestrial and aerial images, and GPS) using software package *BINGO* (GIP, Germany). The achieved average accuracy is 6 cm (standard deviation). In-house software unites coplanar triangles that share one edge in rectangular faces considering a given threshold on the angle between the neighbouring triangles. All the reconstructed facades that did not reach the ground (due to occlusion) were intersected with the terrain surface (see bellow) and new faces were created (Figure 1). Finally, all the objects were organised according to the selected 3D topological structure. As it clearly can be seen, following this approach, roof facets become hard for reconstruction. Apparently, the roofs have to be obtained either from the aerial images or from laser-altimetry data with high accuracy and further integrated with the facades. Since for the UbiCom project the roofs were practically insignificant, this operation was not performed.

Semi-automatic reconstruction of buildings: Although applicable for reconstructing buildings with complex shapes, the procedure described above appeared to be rather slow and inefficient for buildings with round shape, where no clear points can be referenced on multiple photos, or relatively simple long buildings where a lot of points are measured only to tie the images (Figure 1 b). For such man-made objects, we have adopted and extended a 3D semi-automatic procedure (Vermeij and Zlatanova, 2001). The procedure automatically reconstructs roof facets and facades from manually digitised skeletal cloud of points and a Triangulated Irregular Network (TIN) of the terrain surface, and records the data in a 3D topological structure. The accuracy, however, is lower since the measurements are on aerial photographs and not on terrestrial images that have a smaller pixel on object scale.

Reconstruction of terrain objects: Terrain surface objects (streets, parking lots, etc.) are reconstructed combining laser altimetry data and a 2D digital map. The available laser altimetry data have an average density of 5 points per square metre. We "combine" the generated TIN with terrain objects from the topographic map of the Netherlands (i.e. GBKN) in order to incorporate the terrain objects (e.g. streets, gardens, parking lots) into TIN. Further details on the reconstruction can be found in (Zlatanova and van den Heuvel, 2001).

2.2 Phase 2: 3D reconstruction of details

The procedure for 3D reconstruction of details is fully automatic and works on two images with overlapping areas. The major steps are: edge detection, projection of edges on the rough 3D model and back projection on the next image, edge matching, and computation of the end points of the 3D edges. The algorithms are tested on the same images taken for the reconstruction of the 3D topologically structured model. Detailed description of the procedure is given in (Zlatanova and van den Heuvel, 2002). Here we will mention only our approach to match edges between images.

The matching procedure utilises an intermediate projection on a façade that is already reconstructed in Phase 1. After back projection on the second image, we obtain two sets of edges, i.e. detected and projected. To match the projected and detected edges, we apply four criteria. The first one is related to the distance between projected and detected edges. A search algorithm looks for matching candidates within an area of interest (buffer) defined as a rectangle around the projected edge. The second one takes into account the number of endpoints (one or two) of a detected edge that are located within the buffer. The detected edges from the second image that have at least one endpoint falling in the buffer are considered as candidates. The third criterion filters the candidates with respect to the angle between detected and projected edges. The fourth and last criterion refers to the length of the two matched edges, i.e. the difference between the two lengths should not be greater than a reasonable threshold. Among all the candidates, the edge that matches best is selected. The best matched edges are intersected in the 3D space and recorded in the database with a link "belong to" (see next section) a particular face namely the façade used for the projection.

2.3 Discussion on the 3D reconstruction

The work on the reconstruction of the 3D model of the campus proved that one optimal procedure for reconstruction of all the 3D objects and details does not exist and combination of different methods and data sources need to be utilised. As it was mentioned above, we succeeded in building 3D topology during the reconstruction. The direct recording in DBMS is not completed yet (i.e. the reconstruction procedures create intermediate files) but the structuring is according to the selected 3D topological model as described in the next section.

3 3D data structuring

All the tests with respect to 3D data structuring, query and visualisation are conducted in the object/relational DBMS Oracle (http://www.oracle.com). We have experimented with two representations, i.e. topological and geometric. Since Oracle does not offer 3D topology maintenance, we have developed our own 3D topological model.

3.1 Topological implementations

The proposed 3D model is a typical implicit boundary model (Zlatanova and Verbree 2000). Each physical object is associated with four abstractions namely *point*, *linestring*, *surface* and *body*, that are built of simpler elements, i.e. *node* and *face*. Nodes describe faces, linestrings (e.g. pipe lines) and points (e.g. trees, lampposts). The order of the nodes in the face is maintained as wheel. The orientation of the faces is anticlockwise looking at the objects (e.g. buildings) from outside. Faces represent surfaces (e.g. streets, parking lots) and polyhedrons (e.g. buildings). The 3D coordinates are stored with the nodes. All other references are to the ID of the low-level elements. The line features on the facades are encapsulation with their co-ordinates and stored as a separate data set, i.e. *lines*. Each line is considered as a straight line represented by two sets of co-ordinates. The relationships "belong to a face" is explicitly stored in the database.

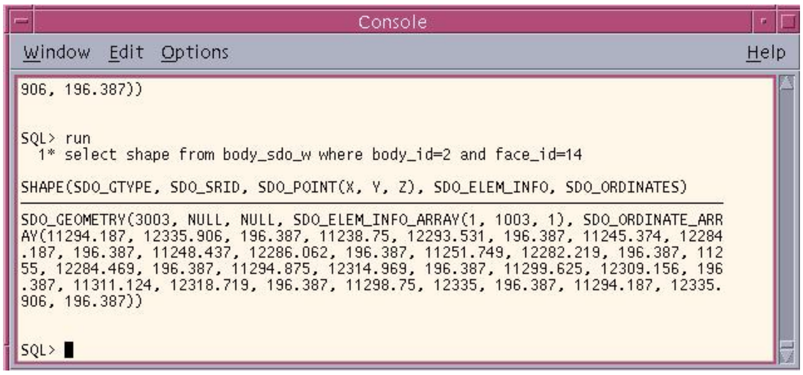


Figure 2: Representation of one polygon in Oracle Spatial

The conceptual schema mentioned above can be implemented following different approaches. The first straightforward approach is the relational implementation. For each object a separate relational table is created. The implementation of the NODE table is trivial: one column for the identifier of the node and the three columns for the (geodetic) co-ordinates of the points. Other tables have similar structure. For example, the FACE table consists of three columns, i.e. a column with the ID of the face, a column giving indication about the order and the number of the nodes in a face, and a column ID of the nodes. Next possibility is creating *object-oriented* views from the relational tables. Views are especially appropriate for retrieval of standard data sets, e.g. the geometry needed for composing a VRML file. The last possibility is object-oriented implementation. Practically, this is a two-step procedure, i.e. creating objects and creating tables. We use two extended Oracle data types, i.e. *varrays* and *nested tables.* While *varrays* are recommended for objects which elements are always retrieved in their completeness, nested tables are said to be suitable for accessing and retrieving individual elements of an object. We have implemented and tested both representations.

All the reconstructed 3D objects are recorded in these representations, but for performance test another relatively large data set (1600 buildings, Figure 5, a) is used. The basic query used for the test is "extract objects needed by the AR system for the accurate positioning" for given position and direction of view. The performance has shown advantages of relational representation and object-oriented views compare to nested tables and variable arrays. Further appropriate spatial indexing and tuning of the database are recommendable for the object-oriented implementations. The best timing for 600 buildings extracted from 1600 buildings is 10 seconds.

3.2 Geometric implementations

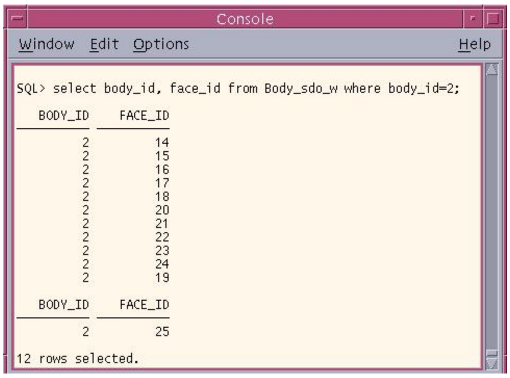
Although promising, the topological implementation may appear inefficient (in terms of response time) for very large data sets (since the 3D model easily can approach a size of several Gb of data). Therefore, a large share of our research was devoted to possibilities to organise the 3D data in Oracle Spatial and use the operations already provided by the vendor.

Oracle Spatial offers *geometric types* to describe spatial objects. These types are defined following object-relational approach and contain information about type, dimension, coordinate system, holes of objects, and provide the list with the coordinates. Currently, the supported geometric types are 2D (point, line, polygon) but 3D coordinates are accepted. It is possible to decode 7 geometric types. Figure 2 shows the representation (a sequence of X,Y,Z coordinates) of one 3D polygon (face 23 from object 2). In contrast to the topological model, 3D coordinates of objects are maintained for each object. To indicate that the polygon is closed, the first coordinate is repeated at the end.

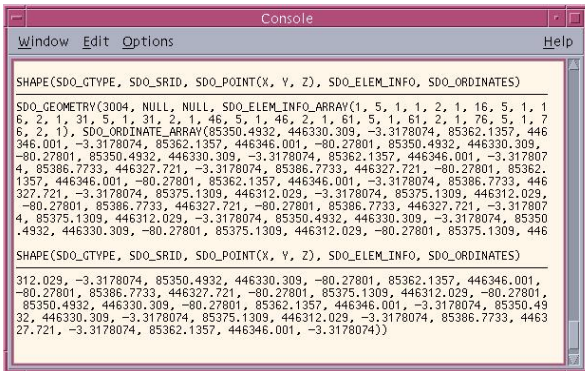
Table 1: Descriptions of BODY_SDO table by: 3D polygons and a 3D collection.

Name	Type	Name	Type
MLINK	NUMBER(10)	MSLINK	NUMBER(10)
BODY_ID	NUMBER(10)	SHAPE	MDSYS.SDO_GEOMETRY
FACE_ID	NUMBER(10)		
SHAPE	MDSYS.SDO_GEOMETRY		

Table 1 shows two possible descriptions of 3D objects within the geometric model of Oracle. In the first representation (Figure 3, a), each building has unique identifier (ID), stored in the column BODY_ID. The column FACE_ID contains the unique ID of the face. The geometry of each face is organised as 3D polygon in the column SHAPE. Apparently, several records represent every building. This representation is a bit inefficient, but a "kind" of topology (i.e. stored relationships between the faces and the 3D object) is maintained. For example, the query "find the neighbouring building" can be easily completed by only comparing the IDs of the faces composing the buildings (thus avoiding the coordinate comparison).



a)



b)

Figure 3: 3D object represented as a) a list of polygons and b) collection of polygons

In the second representation (Figure 3, b), the MSLINK is the ID of the building and the SHAPE column contains the 3D coordinates of all the polygons composing one building. Thus, every building is described as a *collection* of polygons. Although the number of records is reduced (i.e. one building is represented by only one record), the redundancy of coordinates cannot be avoided. Each triple of coordinates is repeated at least three times in the list of coordinates (Figure 3, b).

Regardless what kind of representation is used the data can be further organised in a metadata table, spatially indexed (using several different approaches) and accessed by any application for visualisation and editing. We have experimented with GeoGraphics (extension of MicroStation, Bentley) to query, edit and post the changes in the database. It is possible, for example, to query, extract and edit only one building (Figure 4).

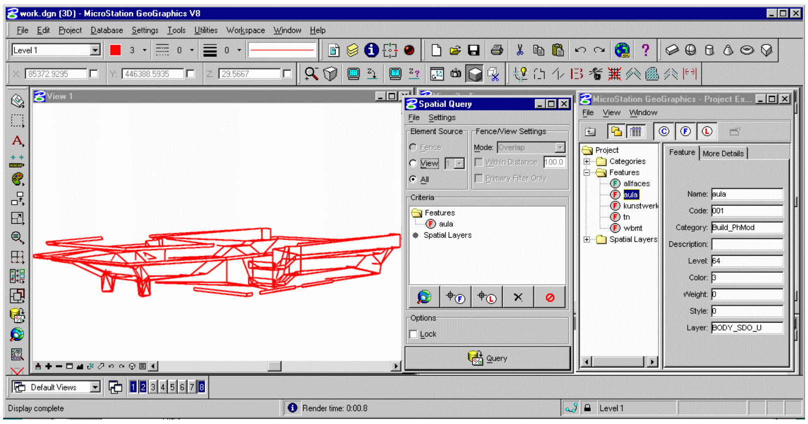
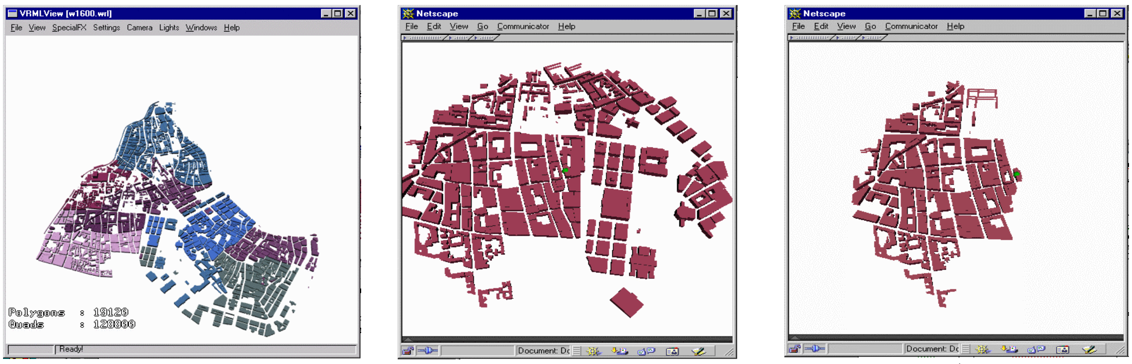


Figure 4: Query the building of the Aula

Indeed, the elements that can be edited correspond to the geometry representation in Oracle Spatial, i.e. either a set of “loose” polygons or a *collection* of polygons. GeoGraphics interprets the two representations differently. In the first case the object is visually one thing but a click on it will highlight one polygon. In the second case the object is one group. To be able to edit it, the object has to be “ungroup” into composing polygons. In both cases, the editing operations are restricted to the defined objects (in our case polygons and their vertices).



a) b) c)

Figure 5: Oracle Spatial query: a) Vienna 3D model, b) spatial operator SDO_WITHIN_DISTANCE and c) function FOV

Among, the large number of spatial operators provided by Oracle Spatial (utilising the supported geometric types), SDO_WITHIN_DISTANCE is the most suitable for the queries needed for our AR system. Given a position and radius of interest, the function returns all the objects within this radius. We have implemented a SQL/PL function Field-of-View (FOV) that further limits the number of objects with respect to the direction and angle of view. Figure 5 shows a VRML file created on the fly as the FOV function is executed. The green sphere is the point of interest (e.g. the user of the AR system). The position, direction and the angle of the FOV are known (i.e. obtained from the GPS receiver and the inertial system). The radius of interest can be specified with respect to the 3D model that is used (in case of many objects it can be reduced to 200-300 m.). The function is executed on a database level, which ensures excellent performance. For example, an area of interest less than 700m (actually much larger that can be seen by the user) can be extracted within 3 seconds.

3.3 Discussion on the 3D structuring

Currently, all the representations in the geometric model have showed better performance compare to the topological model, which is not a surprise. First of all, the nature of the query (needed for the accurate positioning) is pure geometric, i.e. 3D coordinates of objects. In the topological model the 3D coordinates are stored in the NODE table, which means that all the tables (i.e. BODY, FACE, NODE) have to be traversed to obtain them. In contract, in the geometric model, they are organised in one table (one or more records). Second, the geometric model is integrated within the DBMS (and thus optimised), while the topological model is organised in user-defined objects and tables. Third, Oracle Spatial maintains spatial indexing for the objects, which is not applicable for the topological model. This is to say that presently, the geometric model is more appropriate for real-time applications compare to the topological model. In principle, the topological model has many advantages compare to the geometric model, e.g. avoids redundant storage, easier to maintain consistency, efficient for visualisation of large data sets due to the less data to be read from disk, efficient for certain query operations (e.g. find neighbours). We believe that once implemented in DBMS, the 3D topological model will contribute largely to the entire functionality and performance of the DBMS.

4 Conclusions

Our implementations and experiments give us the confidence that 3DGIS and more specifically geo-databases (as a general "container" of geo-information) progressively approach the level of suitability for real-time applications. Currently, it is possible to store and query points, lines and polygons with 3D coordinates. Some of the provided spatial operators (although using only 2D coordinates) can be readily utilised for restricted search in large 3D models.

Although, the geometric model suffices for an AR system, we recommend 3D topological models to be created while reconstructing real-world 3D models. At a later stage the topological model can be converted easily to some of the possible geometric representations of DBMS. Conversion functions between the geometric and the topological models then have to ensure the consistency of the two representations. In our experiments, such functions convert the relational implementation (of the topological model) into the geometric representations. The reverse function (from geometry to topology) still has to be developed.

If AR system relies on the 3D model to refine the positioning of the mobile unit, a special attention has to be paid to the 3D reconstruction procedure. Our experience has clearly showed that the 3D model (ensuring the necessary accuracy and detail for all kind of applications) does not exist. A careful survey and analysis of all the possible data sources, their processing and structuring is needed before the reconstruction is started in order to design an optimal reconstruction procedure given the requirements of the project at hand. In this respect, we considers our reconstruction approach successful: we ensure accuracy, details and corresponding appropriate data organisation ready for import in a DBMS.

References

Hollerer, T., S. Feiner, D. Hallaway, B. Bell, M. Lanzagorta, D. Brown, S. Julier, Y. Baillot and L. Rossenblum, 2001, User interface management techniques for collaborative mobile augmented reality, in: Computer&Graphics, 25 (2001), pp. 799-810

Oosterom, P. v, J. Stoter, W. Quak and S. Zlatanova, 2002, The balance between geometry and topology, in: Advances in Spatial Data Handling, 10th International Symposium on Spatial Data Handling, D.Richardson and P.van Oosterom (Eds.), Springer-Verlag, Berlin, pp. 209-224

Pasman, W., A. van der Schaaf, R.L Lagendijk and F.W. Jansen, 1999, Low latency rendering for mobile augmented reality, Proceedings of the ASCI'99, 15-17 June, Heijen, the Netherlands, pp. 372-376

Persa, S and P. Jonker, 1999, On Positioning Systems for Augmented Reality Applications, Handheld and Ubiquitous Computing 1999, Springer Lecture Notes in Computer Science 1707

Quak, W., P. Oosterom and T. Tijssen, 2002, Testing current DBMS products with realistic spatial data, in: Proceedings of the 5th AGILE conference on Geographic Information Science, ISBN 84-7632-734-X, 25-27 April, Palma de Mallorca, Spain, pp. 189-194

Roberts, G. A. Frans, A. Dodson, B. Denby, S. Cooper and R. Hollands, 2002, Look beneath the surface with augmented reality, in: GIS World, February 2002, pp. 14-17

Stoter, J. and P. van Oosterom, 2002, Incorporating 3D geo-objects into a 2D geo-DBMS, Proceedings of ASPRS/ACSM, 19-26 April, 2002, Washington, USA, CDROM, 12p.

Vermeij, M. and S. Zlatanova, 2001, Semi-automatic 3D building reconstruction using Softplotter, in: Proceedings of the International Symposium on “Geodetic, Photogrammetric and Satellite technologies: development and integrated applications”, 8-9 November, Sofia, Bulgaria, pp.305-314

Zillich, M., D. Legenstein, M. Ayromlou and M. Vincze, 2000, Robust object tracking for robot manipulation and navigation, Proceedings of the XIX congress of ISPRS, Com. V/12, pp. 951-958

Zlatanova, S. and F. van den Heuvel, 2001, 3D city modelling for augmented reality applications, in: Proceedings of the International Symposium on architectural photogrammetry, 8-21 September, Postdam, Germany, p.152-159

Zlatanova, S. and F. van den Heuvel, 2002, Knowledge-based automatic 3D line extraction from close range images in: Proceedings of ISPRS, Vol. 34, part 5, ISBN 1682-1750, 6-11 September, Corfu, Greece, pp. 233-238

Zlatanova, S., A.A. Rahman and M.Pilouk, 2002, Present status of 3D GIS, in: GIM International, Vol.16, No. 6, June 2002, pp. 41-43

Zlatanova, S. and E. Verbree, 2000, A 3D topological model for augmented reality, in: Proceedings of the Second International Symposium on Mobile Multimedia Systems & Applications, 9-10 November, Delft, The Netherlands, pp.19-26