Nat. Hazards Earth Syst. Sci. Discuss., 1, 6093–6131, 2013
www.nat-hazards-earth-syst-sci-discuss.net/1/6093/2013/
doi:10.5194/nhessd-1-6093-2013

Natural Hazards
and Earth System
Sciences
Discussions
Open Access

# Effective 3-D surface modeling for geographic information systems

**K. Yüksek**[1], **M. Alparslan**[2], **and E. Mendi**[3]

[1]Turkish Aviation Academy, Turkish Airlines, Istanbul, Turkey
[2]Computer Engineering Department, Istanbul Kultur University, Istanbul, Turkey
[3]Computer Engineering Department, KTO Karatay University, Konya, Turkey

Received: 18 August 2013 – Accepted: 8 October 2013 – Published: 5 November 2013

Correspondence to: E. Mendi (esmendi@ualr.edu)

Published by Copernicus Publications on behalf of the European Geosciences Union.

## Abstract

In this work, we propose a dynamic, flexible and interactive urban digital terrain platform (DTP) with spatial data and query processing capabilities of Geographic Information Systems (GIS), multimedia database functionality and graphical modeling infrastructure. A new data element, called Geo-Node, which stores image, spatial data and 3-D CAD objects is developed using an efficient data structure. The system effectively handles data transfer of Geo-Nodes between main memory and secondary storage with an optimized Directional Replacement Policy (DRP) based buffer management scheme. Polyhedron structures are used in Digital Surface Modeling (DSM) and smoothing process is performed by interpolation. The experimental results show that our framework achieves high performance and works effectively with urban scenes independent from the amount of spatial data and image size. The proposed platform may contribute to the development of various applications such as Web GIS systems based on 3-D graphics standards (e.g. X3-D and VRML) and services which integrate multi-dimensional spatial information and satellite/aerial imagery.

## 1 Introduction

Due to the increasing importance of visualization, GIS applications are getting widespread use in different forms of development. Initially, environmental planninig, natural resource management, publick work services, education, research, military, urban planning, land management and telecommunications were common application araes in 2-D solutions. During the past decade, there has been a broad use and need of 3-D GIS applications. The previous approaches for GIS process two dimensional data (Zlatanova et al. 2002). Nowadays, 3-D spatial data hiding frameworks have been proposed over database management systems (DBMS). Most of those applications have been developed on already existing sytems. Since GIS-based applications require processing huge size of data, these systems have heavy demands on processing and storage resources. Consequently, modelling and resource management have become an important aspect for an effective GIS application system. The use of GIS in municipal services such as infrastructure and lighting based on city modeling and related layering approach has increased dramatically (Glander and Döllner, 2009).

Given the growing use of Light Detection and Ranging (LIDAR) technology, the 3-D city modeling and analysis processing have triggered off a new era. In recent years, the research on methods using the LIDAR technologies has exponentially increased (Tao and Hu, 2001). The main purpose of such approaches is to correctly identify the terrain and the objects on the terrain. Several techniquees for precisely differentiating the terrain and the objects on the terrain have been recently proposed (Yoon and Shan, 2002). Such methods have also contributed evolving new software and hardware architectures. For instance in Dollner and Hinrichs (2000), Saha et al. (2011), and McKinney and Cai (2002), the objects and terrain are represented as object-oriented data structure concept and each building is assigned to an object. Among the properties of the building include roof type, roof polygons, height, roof surface and LIDAR point sequence. Another approach used interpolation for graphics library and smoothing (Zhou et al., 2004). Many topological models for 3-D spatial objects have been already reported in the literature to describe relationships between spatial objects. As compared to 2-D approaches, such relationships have become more complex in 3-D methods (Zlatanova et al., 2004).

In 3-D GIS modeling, type and size of data have led databases to include special add-ons in order to store and process them. Besides the classic text-based queries, geometry-based queries have become a necessity, which leads emergence of structures known as spatial databases. This feature is then added by many vendor database systems. In recent years, the structures for the storage as well as query of GIS data have reached a promising level (Kothuri et al., 2007). The increase of precision and quality in satellite and aerial imagery has brought about the tendency to use GIS applications. The need for saving and processing huge image information in databases has enabled not only spatial database extensions but also multimedia database extensions to be used in current database systems (Ramakrishnan and Gehrke, 2003). More widespread use in the field of telemedicine and the media, this feature, in fact, has potential initiative for GIS field.

There is a significant trend in re-modeling of systems in 3-D space. Much work has been done on 3-D processing of terrain and objects on terrain in GIS applications (Arens et al., 2005) based on computer graphics methods (Arens et al., 2005). Delaunay Triangulation is widely used in spatial object and terrain representations (Goias and Dutton, 1997). A recent study proposed using polyhedrons as base structures in spatial object representation (Chandra and Govardhan, 2008). On the other hand, recent advances in game engines utilize realistic representations and models such as light, and sound effects (Noh et al., 2006).

Initially, two-tier architectures were used in GIS applications. In these systems, clients operate applications in order to recieve services through the network from the database server (Coors, 2003). Storing the data on the server and sections processed on the client are important in the effectiveness of the architecture. Resource needs and the client/server layout in 3-D structures have gained increasing importance. Software complexity and user expectations are crucial in interactive GIS applications. Production of high-quality satellite and aerial images, getting ground elevation and location information accurately with Global Postioning System (GPS) have contributed to the development of realistic 3-D GIS applications.

The development of GIS applications that require high-speed data and image transmission does not grow parallel with the computer technology. This has led to a search for other means to overcome the bottleneck in terms of computing environments. In other words, a search for new opportunities has emerged to employ high-quality 3-D GIS applications in computers with limited memory resources and processing power. In Guttman (1984), the modeled terrain is entirely processed, but details of only a specific terrain area is presented during representation instead of providing all the details.

With the spread of Internet use, research of web-based GIS solutions has initially started on 2-D (Huang and Lin, 1999). Recently, 2-D GIS research replaced with 3-D approaches. The majority of existing systems are based on three-tier or *n*-tier models. Several platforms have been introduced on presentation of terrain and spatial objects to the users using VRML (Virtual Reality Modeling Language) via web browsers (Ming, 2008; Zhang et al., 2005). However, major drawback in large bandwidth requirement of processing huge size of image and information still ocuurs.

As discussed above, current state of research lacks of quick and affordable GIS framework with satisfactory level of visual quality and high interaction performance for both two-tier and *n*-tier design models. In this work, a novel digital terrain platform is proposed by creating an architecture with the appropiate components integrated from different disciplines. The system can be used for both client-server applications and web-based solutions. The remainder of this paper is organized as follows: Sect. 2 outlines an overview of the generation of spatial and multimedia databases. Section 3 describes the proposed data structure. Section 4 provides an evaluation of the framework. Finally, Sect. 5 presents the conclusions.

## 2 Spatial and multimedia databases

For quick access and storage of the data on Geo-Nodes, the principal data elements of the system; an efficient data structure model must be employed. To do this, first, data anlaysis has been performed. The data used for creating surface include a sequential file with surface coordinates in UTM (Universal Transverse Mercator) format, satellite image of the surface and CAD objects represented on surface. One challenge is distribution and contunity of the data in Geo-Nodes. In this work, we applied Entity-Relational (ER) approach for data modeling and used relational database management system (RDBMS) structure for generating tables and relationships (Ramakrishnan and Gehrke, 2003). The first step of surface modeling is transfer of system data to the tables of corresponding spatial and multimedia database. While the data is stored in tables, methods implemented for business logic to create surface are stored in package structures of Oracle database. Packages are implemented using PL/SQL language (Kothuri et al., 2007; Pribyl, 2002). The processes of surface modeling described in this section are performed only once. If the surface is reconstructed when any data of the surface is modified, the surface modeling process must be repeated from the beginning. The duration of this process depends on the quantity and size of the data. In other words, the number of Geo-Node data elements storing the system information as discrete data is unique. Figure 1 shows the basic structure of Geo-Nodes. The information stored in Geo-Nodes include text, graphics and spatial data.

The database connectivity physically associated with Geo-Nodes is determined by data model. Relational scheme emerged from our data model including tables and their relationships is depicted in Fig. 2.

In addition to the relational model generated from data modeling, the system also has INIT table containing packages of system parameters. It eliminates the dependence of the system with the application and achieves the business logic part of the application on server side. The data retrieved from plain text file in UTM format is converted to SDO_GEOMETRY data type and stored in POINTS table. Figure 3 shows a PL/SQL example inserting spatial data to the POINTS table.

Graphic elements generating geometric representation of surface are stored in POLYHEDRONS table. Seperating points set in POINTS table into blocks, those graphic elements is closed geometries with multi-surface and referred to as polyhedron. In Fig. 4, a PL/SQL statement is given that generates closed geometries with multi-surface and insters to the POLYHEDRONS table.

Since spatial querying is time consuming process, IDs of points in every polyhedron are stored in POINT_POLYHEDRON table. Therefore, relationships of polyhedrons and points are identified by spatial queries only one time, and then stored in POINT_POLYHEDRON table. In order to perform these queries, R-Tree spatial index must be included in SDO_GEOMETRY data fields. Figure 5 presents a code block for relationships between polyhedrons and points and inserting them to the POINT_POLYHEDRON table.

Basic triangular elements that will be used in Tessellation process are converted to SDO_GEOMETRY data typa and stored in TRIANGLES table. Delaunay triangulation algorithm is used for the triangulation process. A PL/SQL statement that inserts triangles to the TRIANGELS table is given in Fig. 6.

IMAGE table contains satellite image of surface that will be modeled. The satellite images are in TIF format and can be up to 2 GB in size. The whole image is stored in IMAGE table and divided into segments corresponding to each polyhedron in POLYHEDRON table. Each image portion is stored in IMAGES table. The whole image and segments stored in IMAGE and IMAGES tables, respectively, are retained using BLOB data type field as binary.

## 3 Surface modeling

3-D surface modeling of a large urban scene requires numerous topographic points. As the surface model expands, the number of points will naturally increase. The method proposed in this study is based on the management of surface data by partitioning into specific data subsets. This section presents the generation of graphics elements, and employement as well as dynamic representation of appropiate data structure.

### 3.1 Modeling components

Using UTM coordinate system, earth coordinates of a surface region are read from a sequential file. The set of points is shifted to the origin of the cartesian coordinate system by substracting the $x$, $y$, $z$ values of the smallest point from the other points in the data set. 3-D point geometries using SDO_GEOMETRY data type are produced from the records of each row and stored in POINTS table. Then, R-Tree spatial index is generated using SDO_GEOMETRY data type in SHAPE field of the table for topological associations. Finally, a point set is obtained in 3-D space (Fig. 7). Prior to the surface modeling, the set of points is divided into specific groups. In order to partition the point set, the minimum bounding multi-surface geometric solid (polyhedron) is constructed that fully encloses the set of points (Fig. 8). The minimum bounding polyhedron containing point set in 3-dimensional space is then decomposed into surfaces according to the X_PART and Z_PART variables of INIT table in spatial database (Fig. 9). These closed surfaces are saved in POLYHEDRONS table. SHAPE field of the table stores the geometries using SDO_GEOMETRY data type. It makes use of the R-Tree spatial index which supports topological queries.

Interior points of each polyhedron, in other words intersection points, can be determined via spatial queries using R-Tree spatial index. For two geometries as input variables, spatial boolean SDO_ANYINTERACT query returns true when they have an intersection relation. The duration of this process depends on the number of points and polyhedrons. Figure 10 shows a query for determination of intersection points with polyhedron. Although R-Tree indexing is the most efficient spatial access method, it is not as fast as a standard SQL query on a database table using B-Tree index structure (Ramakrishnan and Gehrke, 2003).

Since determination of intersection points in a polyhedron via topological query is very time consuming, this will be a drawback in real-time surface representation. One possible way of saving computational effort is to store intersection points in a table, namely POINT_POLYHEDRON. B-Tree index is stored in POLYHEDRON_ID field of the table. As a result, significant performance differences can be achieved in accessing to the intersection points. Figure 11 shows an example of topological query using a B-Tree index.

Using Delaunay triangulation method, a surface can be generated from the points within polyhedrons that are stored in the POLYHEDRONS table. Thus, a triangular network is formed from each polyhedron. Triangles of the network are stored in the TRIANGLES table. All polyhedrons together comprise a triangular surface. Neighbors of the polyhedrons are produced using doubly linked list and the final surface representation is obtained by combining surfaces of polyhedrons with its neighbors.

Problems arise when the surface model is formed by partitioning the polyhedrons into triangular elements. Merging the triangular surfaces of each polyhedron with its neighboring surfaces may result in surface overlap, shared sharp edges of surfaces or surface gap. Figure 12 shows surfaces of polyhedrons each of which four squares is projection of a polyhedron. These problems are tackled by introducing artifical points (marked by cross–x) on the vertices of the polyhedrons (Fig. 13). Artificial points may not be among spatial dataset forming the surface. Therefore, closed attention should be directed toward the surface characteristics while adding artificial points in order to preserve the original surface shape. "$Y$" height values of the artificial points added to the polyhedron vertices must be accurately computed to preserve the surface characteristics. Since polyhedrons are stored in POLYHEDRONS table of spatial database using SDO_GEOMETRY data type, in horizontal plane, the "$X$" and "$Z$" values of the points located on polyhedron vertices can be found easily. In order to determine the heights of new vertices, intersection of each new vertex with other four polyhedrons are identified. If a new vertex added to the polyhedron lies on the corner of the surface formed by other polyhedrons, then it will be in intersection with only its own geometry. If it lies on the edge of the entire suface, then it will be intersecting two polyhedrons, else it will be in intersection with four polyhedrons. After the characterization of intersection state, the closest point to each new vertex is determined. In calculation of height values of new points that lie on $X$–$Z$ axis, if any points already exist in $X$–$Z$ axis, the height of existing points is used. Otherwise, the height and proximity of the closest points are taken into account. Thus, the height of an artificial point can be greatly influenced by the height of its closest point. To compute the height of an artificial point, first, height ratio, of the point is calculated as:

$$\mathrm{pR}_i = 1 - (\mathrm{pD}_i + \mathrm{sD}) \quad i = 1..4 \tag{1}$$

where $\mathrm{pD}_i$ is distance of the point $i$ to the artificial point and sD is the sum of distances of all points to the artificial point. Applying Eq. (1) to all other points, height of the new artificial point, pH is computed as:

$$pH = \frac{((pR_1 + pH_1) + (pR_2 + pH_2) + .. + (pR_i + pH_i))}{(pR_1 + pR_2 + .. + pR_i)} \quad i = 1..4 \tag{2}$$

where $pH_i$ is the height of the $i$ th closest point to the arbitrary point. Finally, the height of the arbitrary point, pH is obtained from the values of heights, $pH_i$ and height ratios, $pR_i$. This computation is applied to all vertices of the polyhedrons stored in the POLYHEDRONS table. Thus, all artificial points introduced in the vertices of the polyhedrons are defined in 3-D space. Defined point geometries are stored in POINTS table.

Applying Delaunay Triangulation with generated points followed by combining surfaces in the polyhedrons, piecewise surfaces (Fig. 14) can be merged into a whole surface model. Although the new points on the polyhedron vertices do not disrupt the surface characteristics, the vertices of new triangles can distort the surface shape. This is case when two of the points on the polyhedron vertices are two of the points producing same triangle. If the surface is too smooth, the distortion may not be distinguished. However, if the surface is sloped and composed of a variety of shapes, base edge of the outermost triangle containing two vertices of polyhedron lies improperly on a straight line from one polyhedron vertice to another. This situation results in visibility of connection points (junctions), and hence decrease in realistic appearance. In order to avoid this, we create new artificial points reflecting surface characteristics between the new points added on the polyhedron vertices at regular intervals. The number of artificial points added along the vertices of polyhedron is the value stored in EDGE_POINT column of INIT table of the spatial database. The heights of the artificial points are calculated using Eq. (1). New added points allow smooth transition that preserves surface features between inner surfaces. Figure 15 depicts a smooth view of whole surface model.

The next step after surface modeling is texturing. In this study, satellite imagery is used as a texture overlay to generate terrain. The satellite image is cropped to the values in X_PART and Z_PART fields in width and height, respectively. Each cropped segment is stored in IMAGE field of IMAGES multimedia database table in binary format. The uniqe identity number in POLYHEDRON_ID field determines which image belongs to which polyhedron. Figure 16 shows the effect of seamless texturing between surfaces.

Proposed surface modeling technique provides 3-D representation of topographic ground surface and many types of ground and non-ground objects such as buildings, houses and infrastructure (e.g. roads and bridges including noise barriers, lamp posts and sign boards as well as buried objects such as drums and pipes). The system allows representation of any object on terrain. An object is associated with a 3-D geometry model together with a texture image and integrated to the model with its world coordinates. In order to facilitate the storage and indexing of 3-D objects represented on the surface model, two database tables have been created: (i) OBJECT_TYPES, and (ii) OBJECTS. OBJECT_TYPES table is used to incorporate different types of objects with the terrain surface. OBJECTS table contains location information and terrain conditions for placing an object referenced in OBJECT_TYPES table. After model construction, the object is inserted into the OBJECT field of OBJECT_TYPES table in binary format. Then, the object model is scaled to the desired size, rotated to the desired orientation and ultimately translated to the final destination on surface. This allows to use multiple references (instancing). That is, once the object model is abstracted, re-instancing it into several representations could be possible. Once the position of the object model is identified, the polyhedron that bounds the object is determined from spatial database using topological queries. An R-Tree index structure is built during the process of unfolding polyhedrons using topological queries on LOCATION column of OBJECTS spatial table. The polyhedron is then assigned a unique identification number in POLYHEDRON_ID field of OBJECTS table.

### 3.2 Data structure

A Geo-Node, as a stand-alone element, has no meaning itself. Its relations with other Geo-Nodes must be known. Figure 17 depicts proposed eight-connected double-linked list, in which N denotes the objects stored in linked lists.

Each Geo-node in eight-connected double-linked list stores geometry entities including triangles that form multi-surface geometry, satellite image segments that drape over surface and 3-D objects represented on terrain.

The main use of the proposed dynamic data structure is that it manages the movement of persistent Geo-Node data on secondary storage and buffer during interactive display of surface model by the user. In this context, linked list corresponds to a grid structure that accessing one element provides traversing through all elements in the linked list. Figure 18 illustrates the eight-connected linked list structure composed of Geo-Nodes. The right side of the Fig. 18 shows all Geo-Nodes allocated in the memory at a specific time, and red frame indicates subset of Geo-Nodes available to the user at that time.

RECTANGLES table of the spatial database consists of TOP, LEFT, RIGHT, BOTTOM, RIGHT_TOP, RIGHT_BOTTOM, LEFT_BOTTOM, LEFT_TOP, LEFT_BOTTOM columns of numeric data with rows. The data contains numeric values defining 3-D multi-surface geometry stored in the same row with all neighboring geometries in eight-connected double-linked list. The motion at the client side via any I/O device triggers the buffer manager based on Directional Replacement Policy (DRP). This module provides smooth and continuous retrieval of data by utilizing effective connectivity periods when the user's view is moving. A simple mechanism of the DRP algorithm is given in Fig. 19. *P* denotes the Geo-Node to be fetched using pre-fetching and, *X* denotes the Geo-Node to be replaced using the replacement policy in the movement of the user's view.

### 3.3 Dynamic representation of the surface model

Proposed multi-layer system offers friendly user interface easy to understand and provides interactive visualization. The position and orientation of the user's view on the surface are continuously updated. When current view of the user is within a polyhedron that is on the corner and edge vertices of the surface, new surface segments (plane surfaces) are identified to be visualized. These are the planes in neighboring polyhedrons of the current polyhedron. Polyhedrons together with data of corresponding triangular planes, satellite images and CAD objects in the database are prepared for visualization. The surface of each polyhedron can then be reconstructed from this data. Figure 20 shows the reconstruction of the surface model when reaching the boundary of the display.

The future viewing direction of the user on the surface can be estimated based on the path of the navigation, and hence new surface segments to be visualized can be incorporated into the buffer pool in the memory before reaching a boundary region. Whenever the position of the user within a polyhedron is placed upon another polyhedron, neighboring polyhedrons of the current polyhedron is identified using the eight-connected double-linked list, and corresponding planes and objects of this polyhedron are sent to the buffer. In query processing, when the database needs to access a data item of interest, it first requests on the buffer manager. If the desired surface in the polyhedron is already in the buffer, the buffer manager simply returns it. If it does not, the buffer manager brings desired polyhedron and corresponding information (triangles, objects and satellite image) from the database to the memory. This process is repeated for any displacement on the surface until the view of the user reaches a boundary. Before displaying an edge surface on the current polyhedron, surface segment to be rendered is reconstructed. In dynamic representation mechanism, buffer manager is first called to fetch the data needed for reconstruction. If the buffer manager has not already cached a requested block, the surface to be visualized is reconstructed reading the corresponding block from the database to the buffer pool. After surface reconstruction, buffer manager allocates space in the buffer by throwing out objects that are no longer required. Due to the huge size of data, accessing a sector on database is much slower than accessing a buffer space in memory. Therefore, buffer management module reduces the number of accesses to the database and I/Os needed, and improves the completion time of queries.

## 4 Experimental results

The performance of the proposed surface modeling technique is evaluated using frame per second (FPS). FPS is influenced by several factors such as the number of rendered points, size of the satellite image draping over the surface, the number of rendered polyhedrons at a specific time as well as graphics card and refresh rate. Performance results are depicted in Fig. 21. FPS given in the fifth column is the maximum and minimum interval values according to various parameters. The first column (the number of points) represents the number of points to create surface model. These points are processed dividing into the subsets by the number of polyhedrons given in second column. If the same number of points is used for more polyhedrons, the number of points required for each polyhedron will decrease. Neighborhood in third column denotes the number of polyhedron in which a surface is formed during rendering. Thereby, the more neighbor relationships occur, the more number of polyhedrons that contain data of rendered surfaces. As a result, the number of points, polyhedrons and neighbor relationships determine the amount of data required for surface modeling at a specific time. The surface is overlayed by a satellite image after data processing procedure. The fourth column of the table shows the size of the satellite images. Higher resolution images mean higher size of satellite image files. Furthermore, the larger size of a satellite image is, the larger size of an image segment draping over surface of a polyhedron. Hence, large size of these image files will require higher performing hardware, i.e. faster graphics cards, more powerful processors, etc., in order to render the scene.

We also measured the memory consumption of the proposed surface modeling method. Figure 22 presents memory usage during rendering. When the user moves along from one polyhedron to the other during dynamic navigation over the surface; the terrain data, which is not in main memory, according to the current position of the user is brought to the buffer space by reading from the spatial database. Since DRP-based buffer manager loads data from database to the main memory buffer pool, overall memory consumption increases over time as long as user moves over the surface, as shown in Fig. 22. When the user's view is at the boundary of the terrain, the surface is reconstructed using the data in the buffer pool, and finally data is released from the buffer. The downward regions of memory requirement in Fig. 22 indicate the reconstruction of the model. In practice, fitting huge spatial data entirely inside the main memory is infeasible. Therefore, such buffer management will provide significant performance savings.

Experiments were conducted on an AMD Athlon 64 X2 4200+ dual-core processor having a speed of 2.20 GHz and 3 GB of main memory under Windows XP 32 bit operating system. The graphics card used was an nVIDIA GeForce 9500 GT with 128 MB of on-board memory and the monitor was set to a refresh rate of 75 Hz.

## 5 Conclusions

In this paper, we presented an interactive multi-purpose platform for web-based or client/server-based 3-D GIS applications with superior graphics support. The system utilizes main data elements, called Geo-Node, constituted from a flexible data structure and supported by an effective buffer management. The data transfer between secondary storage and main memory is optimized with "most recently used" strategy. Thus, fast access is provided to the anticipate uses of Geo-Nodes that store spatial data and corresponding images. The capabilities of data and query processing in spatial database and handling multimedia components in multimedia database are integrated with an efficient data structure. Using interpolation method, surface anomalies

Title Page
Abstract | Introduction
Conclusions | References
Tables | Figures
◀I | ▶I
◀ | ▶
Back | Close
Full Screen / Esc
Printer-friendly Version
Interactive Discussion

are eliminated. Furthermore, a game engine was used in rendering pipeline in order to improve the realism of visualization. The experimental results show that visualization of urban scene is successful with high performance. The proposed platform can offer new capabilities to build 3-D GIS applications in two tier or *n*-tier (client/server) architectures.

## References

Arens, C., Stoter, J., and Oosterom, P.: Modelling 3-D spatial objects in a geo-DBMS using a 3-D primitive, Comput. Geosci., 31, 165–177, 2005.

Chandra, N. S. and Govardhan, A.: Design and implementation of Polyhedron as a primitive to represent 3-D spatial object, Journal of Theoretical and Applied Information Technology, 4, 212–218, 2008.

Coors, V.: 3-D-GIS in networking environments, Comput. Environ. Urban, 27, 345–357, 2003.

Dollner, J. and Hinrichs, K.: An object-oriented approach for integrating 3-D visualization systems and GIS, Comput. Geosci., 25, 67–76, 2000.

Glander, T. and Döllner, J.: Abstract representations for interactive visualization of virtual 3-D city models, Comput. Environ. Urban, 33, 375–387, 2009.

Goias, N. A. and Dutton, R. W.: Delaunay Triangulation and 3-D adaptive mesh generation, Finite Elem. Anal. Des., 25, 331–341, 1997.

Guttman, A.: R-Trees: A Dynamic Index Structure for Spatial Searching, ACM SIGMOD International Conference on Management of Data, 47–57, 1984.

Huang, B. and Lin, H.: GeoVR: a web-based tool for virtual reality presentation from 2-D GIS data, Comput. Geosci., 25, 1167–1175, 1999.

Kothuri, R. V., Godfrind, A., and Beinat, E.: Pro Oracle Spatial for Oracle Database 11g, Springer, New York, ISBN: 1-59059-899-7, 2007.

McKinney, D. C. and Cai, X.: Linking GIS and water resources managment models: an object-oriented method, Environ. Modell. Softw., 17, 413–425, 2002.

Ming, W.: A. 3-D Web GIS System Based On VRML And X3-D", Second International Conference on Genetic and Evolutionary Computing, 25–26 September 2008, 197–200, 2008.

Noh, S. S., Hong, S. D., and Park, J. W.: Using a game engine technique to produce 3-D Entertainment contents, 16th International Conference on Artificial Reality and Telexistence-Workshops (ICAT'06), 29 November–1 December 2006, Hangzhou, China, 246–251, 2006.

Pribyl, B.: Learning Oracle PL/SQL, O'Reilly, California, ISBN: 0-596-00180-0, 2002.

Ramakrishnan, R. and Gehrke, J.: Database Management Systems (Third Edition), McGrawHill, New York, ISBN: 0-07-246563-8, 2003.

Saha, K., Wells, N. A., and Munro-Stasiuk, M.: An object-oriented approach to automated landform mapping: a case study of drumlins, Comput. Geosci., 37, 1324–1336, 2011.

Tao, V. and Hu, Y.: A review of post-processing algorithms for airbone LiDAR data, ASPRS Annual Conference, 23–27 April 2001, St. Louis, Missouri, USA, 2001.

Yoon, J. S. and Shan, J.: Urban DEM generation from raw airbone LiDAR data, CD-ROM Proceedings of the annual ASPRS (American Society for Photogrammetry and Remote Sensing) Conference, 19–25 April 2002, Washington, DC, USA, 2002.

Zhang, L., Yang, C., Liu, D., Ren, Y., and Rui, X.: A web-mapping system for real-time visualization of the global terrain, Comput. Geosci., 31, 343–352, 2005.

Zhou, G., Song, C., Simmers, J., and Cheng, P.: Urban 3-D GIS from LiDAR and digital aerial images, Comput. Geosci., 30, 345–353, 2004.

Zlatanova, S., Rahman, A. A., and Pilouk, M.: Trends in 3-D GIS Development, Journal of Geospatial Engineering, 4, 1–10, 2002.

Zlatanovaa, S., Rahmanb, A. A., and Shi, W.: Topological models and frameworks for 3-D spatial objects, Comput. Geosci., 30, 419–428, 2004.

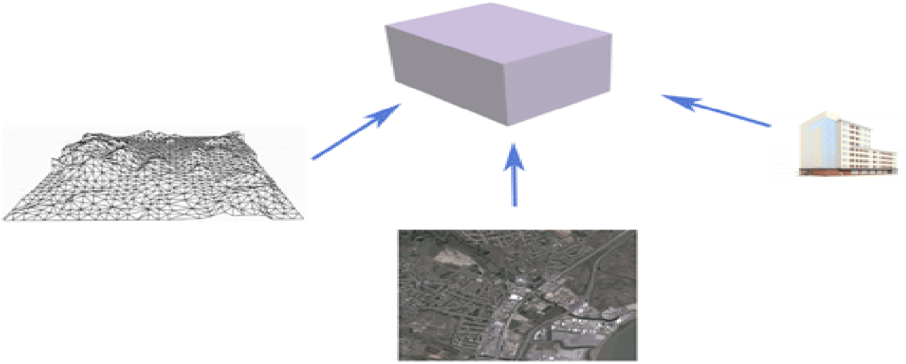


**Fig. 1.** Geo-Node Structure.

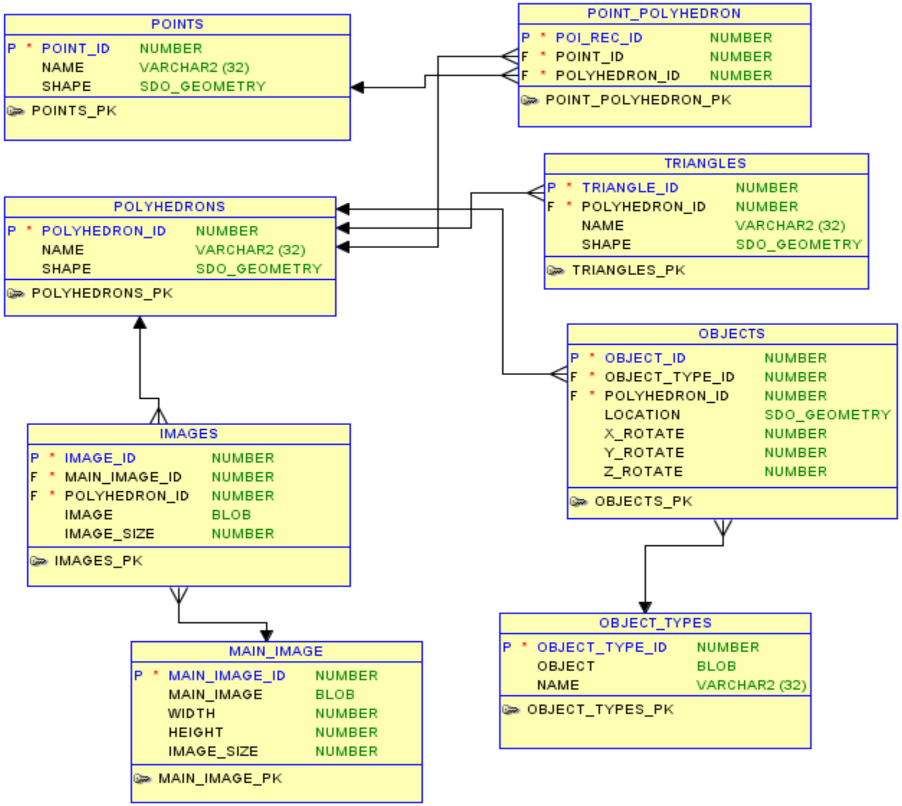
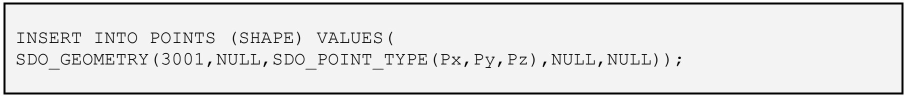
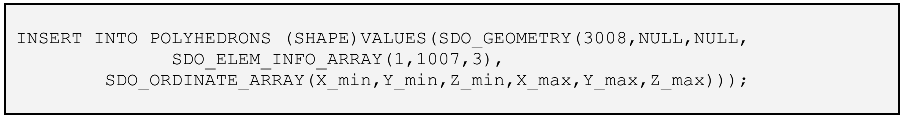


**Fig. 2.** ER diagram of the data stored in tables.



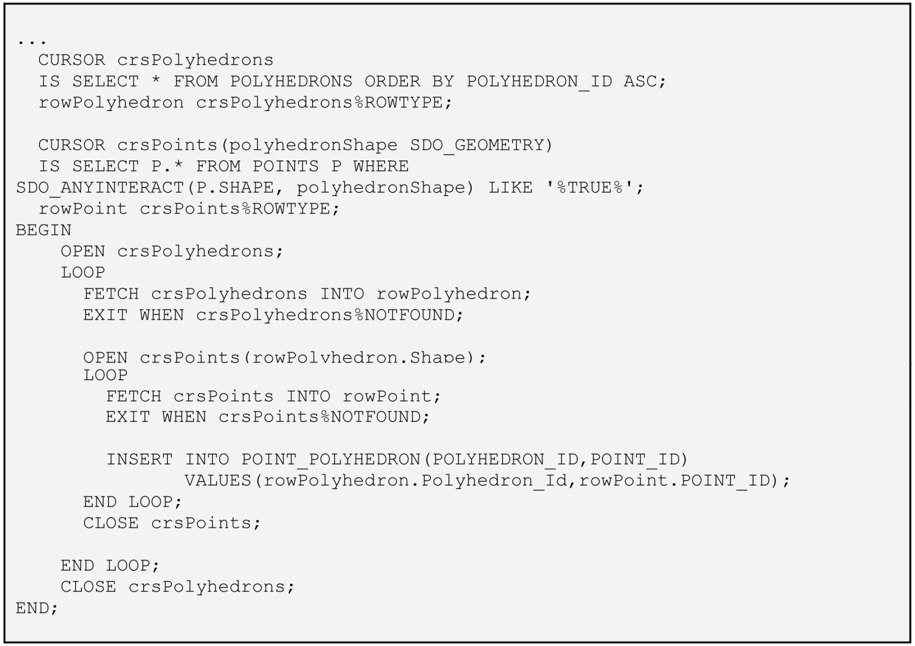
```
INSERT INTO POINTS (SHAPE) VALUES(
SDO_GEOMETRY(3001,NULL,SDO_POINT_TYPE(Px,Py,Pz),NULL,NULL));
```

**Fig. 3.** PL/SQL statement inserting point geometry to the POINTS table.



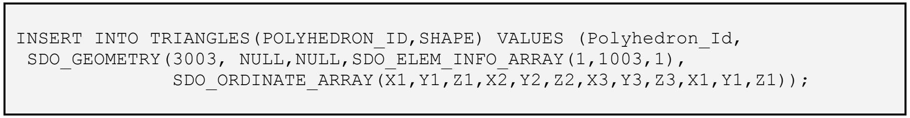
```
INSERT  INTO  POLYHEDRONS  (SHAPE)VALUES(SDO_GEOMETRY(3008,NULL,NULL,
              SDO_ELEM_INFO_ARRAY(1,1007,3),
        SDO_ORDINATE_ARRAY(X_min,Y_min,Z_min,X_max,Y_max,Z_max)));
```

**Fig. 4.** PL/SQL statement generating closed geometries with multi-surface and inserting to the POLYHEDRONS table.



```
...
  CURSOR crsPolyhedrons
  IS SELECT * FROM POLYHEDRONS ORDER BY POLYHEDRON_ID ASC;
  rowPolyhedron crsPolyhedrons%ROWTYPE;

  CURSOR crsPoints(polyhedronShape SDO_GEOMETRY)
  IS SELECT P.* FROM POINTS P WHERE
SDO_ANYINTERACT(P.SHAPE, polyhedronShape) LIKE '%TRUE%';
  rowPoint crsPoints%ROWTYPE;
BEGIN
    OPEN crsPolyhedrons;
    LOOP
      FETCH crsPolyhedrons INTO rowPolyhedron;
      EXIT WHEN crsPolyhedrons%NOTFOUND;

      OPEN crsPoints(rowPolyhedron.Shape);
      LOOP
        FETCH crsPoints INTO rowPoint;
        EXIT WHEN crsPoints%NOTFOUND;

        INSERT INTO POINT_POLYHEDRON(POLYHEDRON_ID,POINT_ID)
               VALUES(rowPolyhedron.Polyhedron_Id,rowPoint.POINT_ID);
      END LOOP;
      CLOSE crsPoints;

    END LOOP;
    CLOSE crsPolyhedrons;
END;
```

**Fig. 5.** PL/SQL code block identifying the relationships between polyhedrons and points, then inserting to the POINT_POLYHEDRON table.



```
INSERT INTO TRIANGLES(POLYHEDRON_ID,SHAPE) VALUES (Polyhedron_Id,
 SDO_GEOMETRY(3003, NULL,NULL,SDO_ELEM_INFO_ARRAY(1,1003,1),
              SDO_ORDINATE_ARRAY(X1,Y1,Z1,X2,Y2,Z2,X3,Y3,Z3,X1,Y1,Z1));
```

**Fig. 6.** PL/SQL statement inserting triangle to the TRIANGELS table.

**Fig. 7.** A sample of point set.

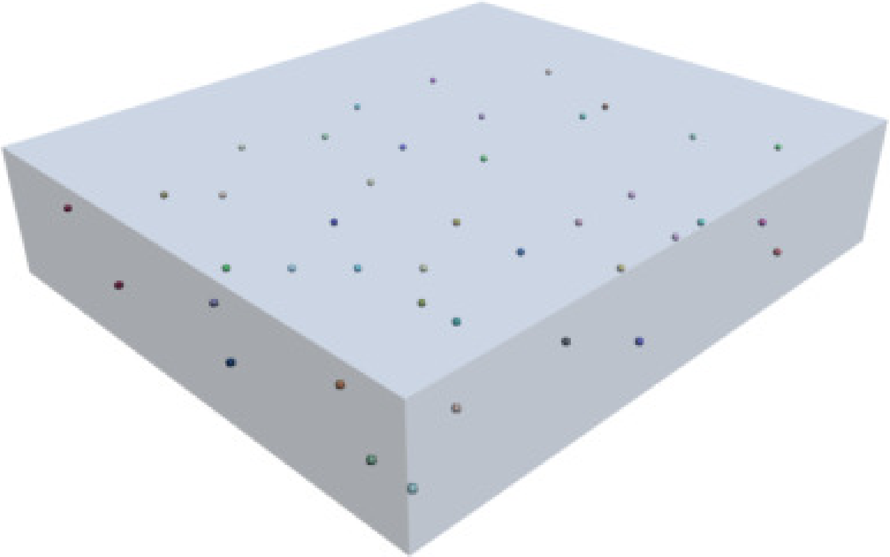


**Fig. 8.** Multi-surface closed geometry containing point set.

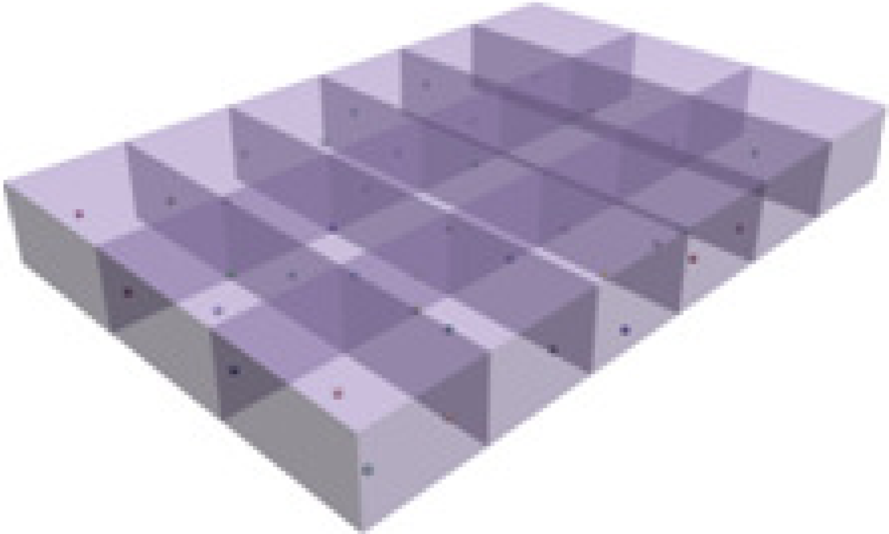
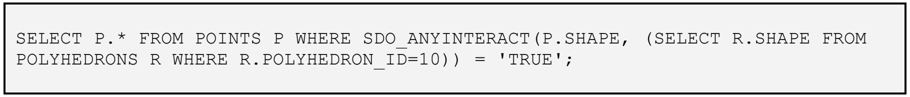


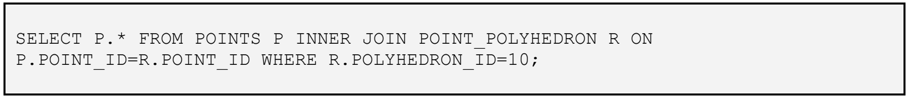
**Fig. 9.** Decomposed geometry containing point set.



```
SELECT P.* FROM POINTS P WHERE SDO_ANYINTERACT(P.SHAPE, (SELECT R.SHAPE FROM
POLYHEDRONS R WHERE R.POLYHEDRON_ID=10)) = 'TRUE';
```

**Fig. 10.** Query determining intersection points with a polyhedron.

Title Page
Abstract Introduction
Conclusions References
Tables Figures
◀I ▶I
◀ ▶
Back Close
Full Screen / Esc
Printer-friendly Version
Interactive Discussion



```
SELECT P.* FROM POINTS P INNER JOIN POINT_POLYHEDRON R ON
P.POINT_ID=R.POINT_ID WHERE R.POLYHEDRON_ID=10;
```

**Fig. 11.** A topological query with B-Tree index.

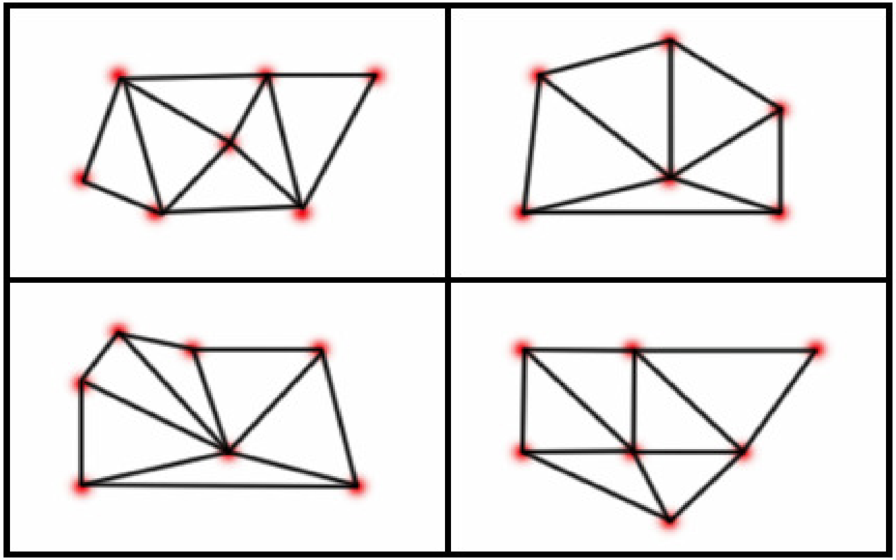


**Fig. 12.** Surfaces of polyhedrons.

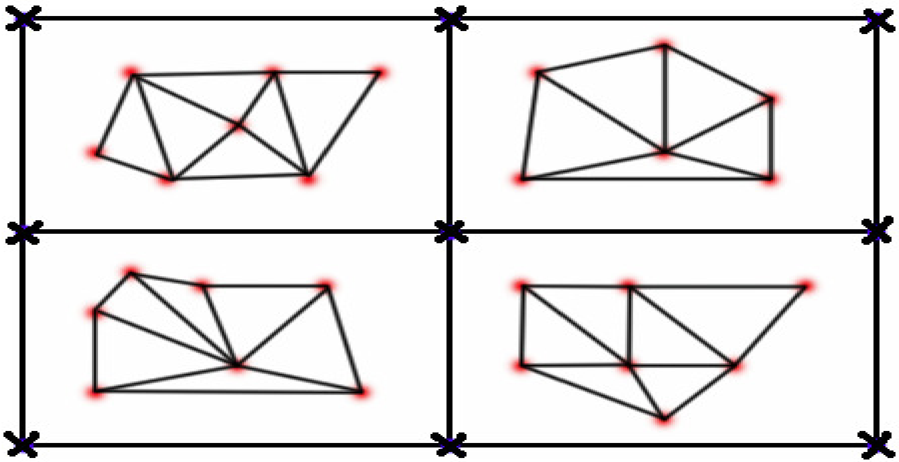


**Fig. 13.** Artifical points, shown by cross (x), added to merge the surface.

Title Page
Abstract | Introduction
Conclusions | References
Tables | Figures
◂◂ | ▸▸
◂ | ▸
Back | Close
Full Screen / Esc
Printer-friendly Version
Interactive Discussion

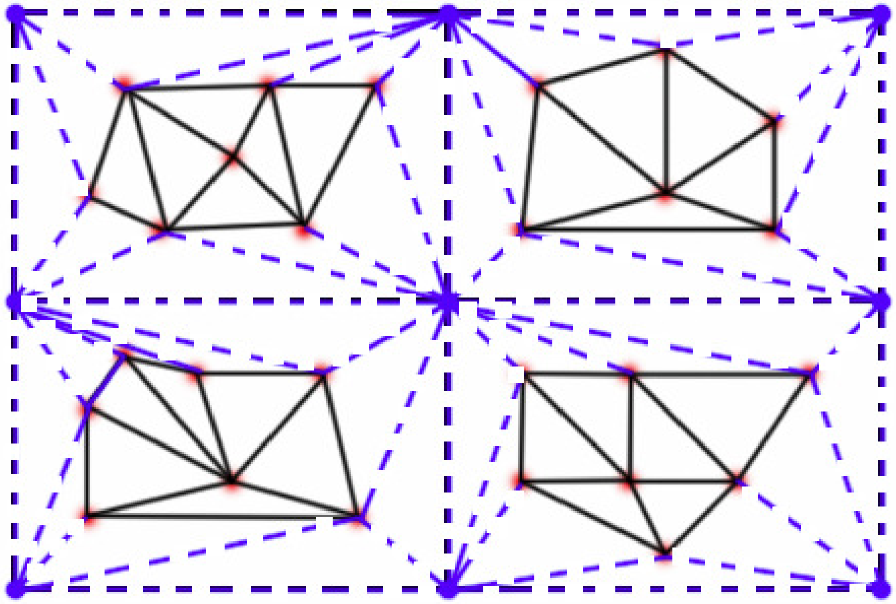


**Fig. 14.** A whole surface model merged with new dotted triangles.

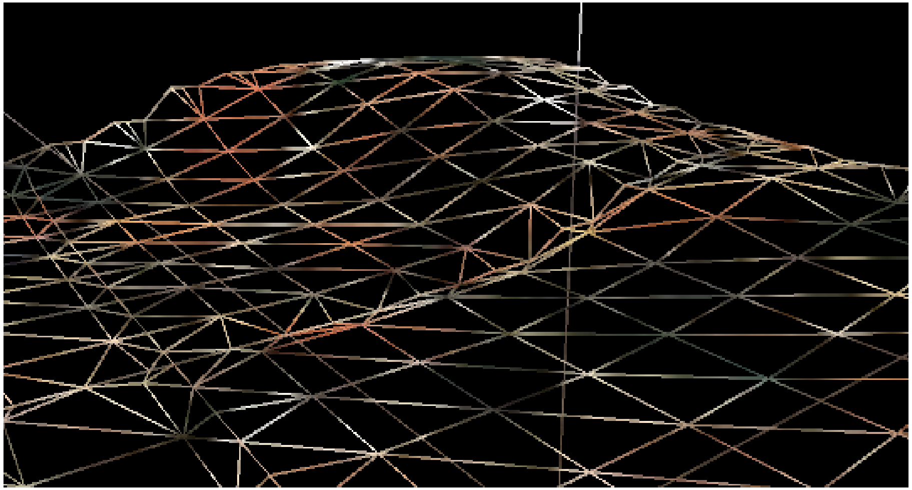


**Fig. 15.** Smooth view of whole surface model.

**Fig. 16.** Textured model with continuous smooth transitions (upper) and without continuous smooth transitions (bottom) between surfaces.

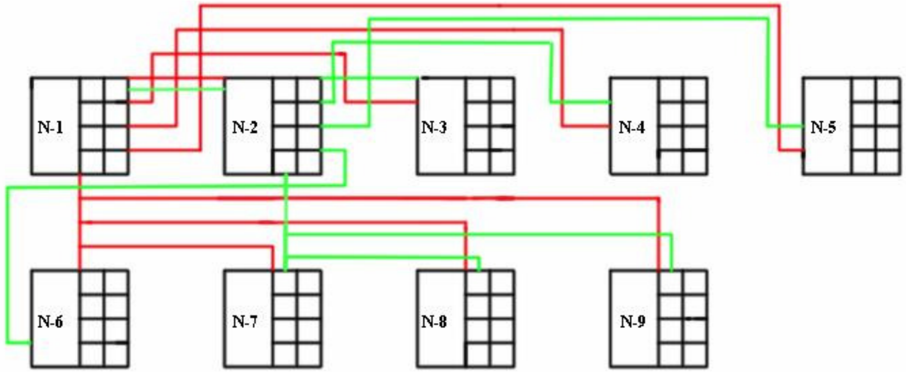


**Fig. 17.** Eight-Connected Linked List.

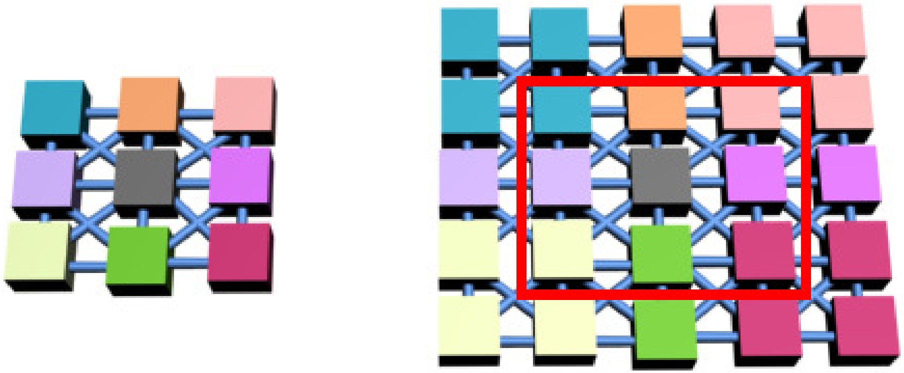


**Fig. 18.** Illustration of eight-connected linked list composed of Geo-Nodes.

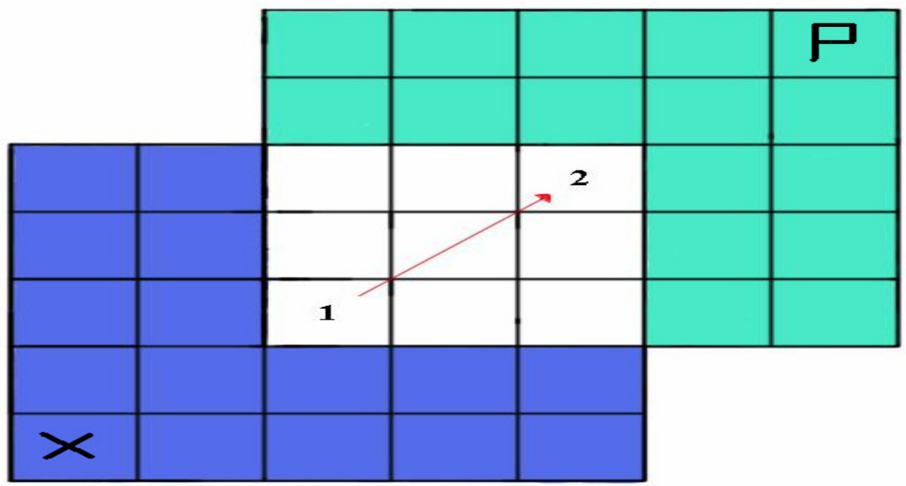
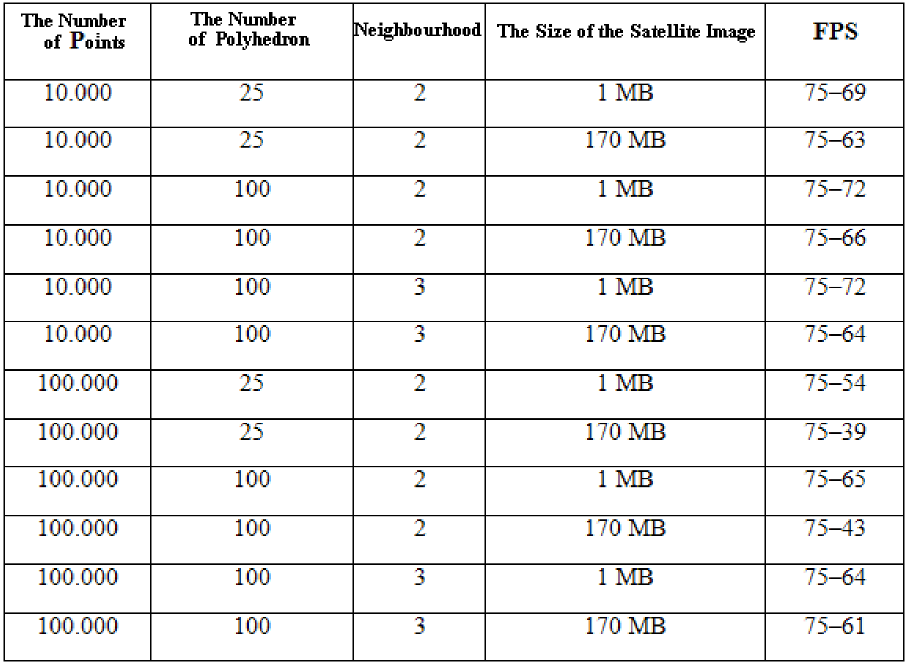


**Fig. 19.** The DRP mechanism.

**Fig. 20.** Reconstruction of the model in the direction of the movement upon reaching the boundary of the surface at a specific time.

Title Page
Abstract | Introduction
Conclusions | References
Tables | Figures
◀I | ▶I
◀ | ▶
Back | Close
Full Screen / Esc
Printer-friendly Version
Interactive Discussion



| The Number of Points | The Number of Polyhedron | Neighbourhood | The Size of the Satellite Image | FPS |
|---|---|---|---|---|
| 10.000 | 25 | 2 | 1 MB | 75–69 |
| 10.000 | 25 | 2 | 170 MB | 75–63 |
| 10.000 | 100 | 2 | 1 MB | 75–72 |
| 10.000 | 100 | 2 | 170 MB | 75–66 |
| 10.000 | 100 | 3 | 1 MB | 75–72 |
| 10.000 | 100 | 3 | 170 MB | 75–64 |
| 100.000 | 25 | 2 | 1 MB | 75–54 |
| 100.000 | 25 | 2 | 170 MB | 75–39 |
| 100.000 | 100 | 2 | 1 MB | 75–65 |
| 100.000 | 100 | 2 | 170 MB | 75–43 |
| 100.000 | 100 | 3 | 1 MB | 75–64 |
| 100.000 | 100 | 3 | 170 MB | 75–61 |

**Fig. 21.** Performance results.

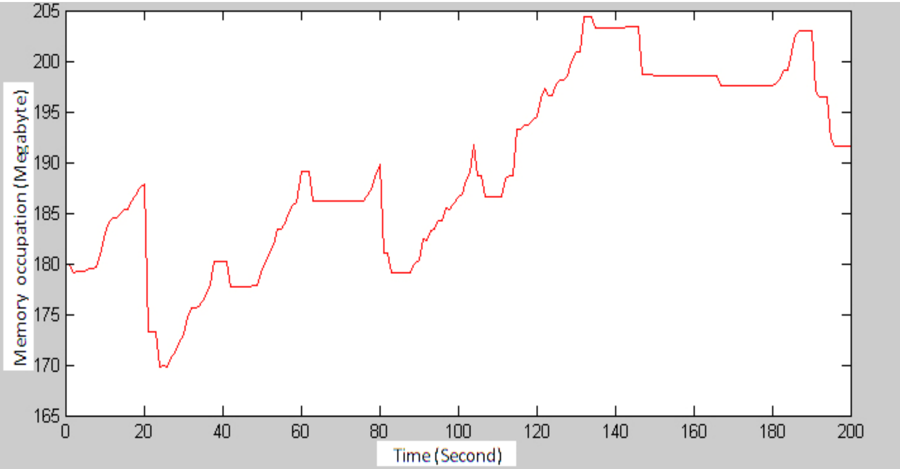


**Fig. 22.** Memory usage (MB) vs. scene time (sec).