A Tetrahedronized Irregular Network Based DBMS approach for 3D Topographic Data Modeling

Friso Penninga[(1)] , Peter van Oosterom[(1)] , and Baris M. Kazar[(2)]

[(1)] Delft University of Technology, OTB, section GIS technology, Jaffalaan 9, 2628 BX Delft, The Netherlands
[(2)] Oracle USA, One Oracle Drive, Nashua, NH 03062, USA

F.Penninga@otb.tudelft.nl, oosterom@geo.tudelft.nl, baris.kazar@oracle.com

Abstract

Topographic features such as physical objects become more complex due to increasing multiple land use. Increasing awareness of the importance of sustainable (urban) development leads to the need for 3D planning and analysis. As a result, topographic products need to be extended into the third dimension. In this paper, we developed a new topological 3D data model that relies on Poincaré algebra. The internal structure is based on a network of simplexes, which are well defined, and very suitable for keeping the 3D data set consistent. More complex 3D features are based on this simple structure and computed when needed. We describe an implementation of this 3D model on a commercial DBMS. We also show how a 2D visualizer can be extended to visualize these 3D objects.

1 Introduction

The 3D data models should enable 3D analysis, whereas early 3D GIS developments often focused on visualization, often in Virtual Reality-like environments. Another important characteristic of topographic data sets is the wide variety of applications, thus disabling optimization of the data model for a specific task. Due to current developments in sensor techniques (Vosselman 2005) more and more 3D data becomes available. Furthermore, the point density and thus data volume is increasing. An example of the capabilities of terrestrial laser scanning is illustrated in Figure 1.

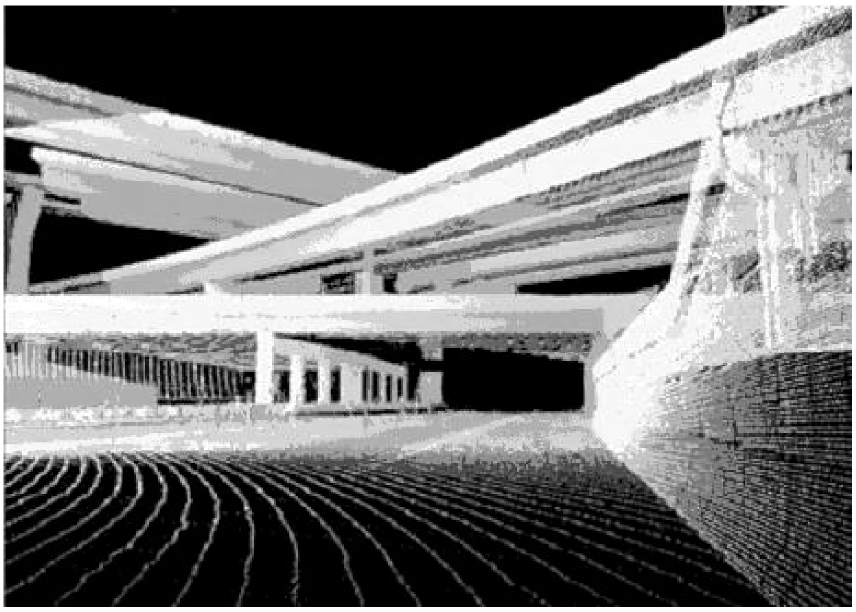


Fig. 1. Terrestrial laser scanning provides insight in complex 3D objects

There are a vast number of studies (amongst others Arens et. al. 2005, Guibas et. al. 1985, van der Most 2004, van Oosterom 1994, 1997, 2002, Penninga 2005, Verbree et. al. 2005, Zlatanova 2000a, 2000b, 2002a) on 3D data modeling. Most of these studies are summarized and neatly compared in (Zlatanova 2002b). Extending topographic data models into 3D is most relevant at large scale topography. However, this will lead to a substantial increase of data volume. With this increase, ensuring data integrity and maintaining performance become important requirements. As a result, implementing the 3D data structure in a spatial database is a sensible thing to do. In this research, the Tetrahedronized Irregular Network (TEN) is selected as an internal data structure. The selection of this structure, motivated by computational advantages, the well-definedness of the triangles (always flat), the presence of well-known topological relationships (Guibas and Stolfi, 1985), easy maintenance, visualization of triangles (Zlatanova 2002a, 2002b), and flexibility of forming more complex objects, is described by (Penninga 2005). Our model is based on the Poincaré algebra (and has therefore a solid foundation) and does pay attention to DBMS issues, such as indexing, updating and locking mechanisms.

In Section 2 an introduction to the theory behind our conceptual TEN-model is given with ingredients such as n-dimensional simplexes, boundary and coboundary, Poincaré algebra, network of simplexes (in 3D called the TEN). DBMS issues related to the 3D TEN-based modeling are introduced in Section 3, while Section 4 describes an (partial) implementation of the in the DBMS with a ‘toy’ example. The paper concludes with summarizing the most important results and indication on future research in Section 5.

2 3D Topographic data modeling in a TEN data structure

As we see topography as the collection of physical objects, two observations can be made regarding 3D topographic data modeling:

1. Physical world objects have by definition a volumetric shape. There are no such things as point, line or area features; only point, line and area representations at a certain level of generalization. Which representation to use should be stated in the DCM (Digital Cartographic Model) but not in the DLM (Digital Landscape Model), which contains our 3D topography.
2. The real world can be considered as a volume partition: a set of non-overlapping volumes that form a closed modeled space. As a consequence, objects like 'earth' or 'air' are explicitly part of the real world and thus have to be modeled.

As a result, the topographic data set consists of volume features. However, in some cases area features might be useful. Area features can be modeled in our approach to mark important boundaries between two volume features (and can have their own properties, such as surface material and color). Therefore, they cannot exist without the presence of these volume features; an area feature is the first derivative of a volume feature (and this is repeated for line features and point features). In the UML class diagrams in Figure 3 and Appendix A, these area features are modeled as association classes.

The decision to explicitly include 'air' and 'earth' features – thus modeling 'empty' space in between physical objects - is influenced by the fact that this empty space is subject of many analyses. In case of modeling air pollution or flooding, the user is interested in what happens in this empty space. The remainder of this section will discuss the Poincaré algebra (2.1) and the resulting conceptual TEN model in UML class diagram (2.2).

2.1 Poincaré algebra

The Tetrahedronized Irregular Network (TEN) is the three-dimensional variant of the well-known Triangulated Irregular Network (TIN). Besides nodes, edges and triangles, a TEN also consists of tetrahedrons for representing volumetric shapes. Nodes, edges, triangles and tetrahedrons are all simplexes, i.e., the simplest possible geometry in every dimension. Modeling 3D features by the use of simplexes is described by Carlson (1987). Using simplexes has three advantages:

1. Simplexes are well-defined: a kD simplex is bounded by k+1 (k-1)D simplexes (Egenhofer et. al. 1989a). For instance: a 2D simplex (triangle) is bounded by 3 1D simplexes (edges).
2. Flatness of the faces: every face can be described by three points.
3. Every simplex is convex, regardless of its dimension.

A direct result of the well-defined character of simplexes and thus of a TEN is the availability of 3D topological relationships. Whereas in the two-dimensional case, (the TIN) the important relationships are on edge level (i.e. an edge has a face on the left and one on the right, thus defining adjacency of faces), in three dimensions the important relationships are on face level. Each face (triangle) bounds two tetrahedrons. Left and right are meaningless in 3D, but due to the ordering of the edges in the triangle one can determine the direction of the normal vector and thus relate to tetrahedrons in the positive and negative direction. The n-dimensional simplex is defined by n+1 nodes and has the following notation $S_n = <x_0,\ldots,x_n>$.

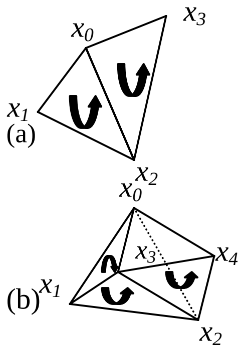
So, the first four simplexes are $S_0 = <x_0>$, $S_1 = <x_0 ,x_1>$, $S_2 = <x_0 ,x_1,x_2>$, and $S_3 = <x_0 ,x_1 ,x_2 ,x_3>$. With (n+1) nodes, there are (n+1)! combinations of these nodes, that is for the four simplexes, there are respectively 1, 2, 6 and 24 options. For S_1 the two combinations are $<x_0 ,x_1>$ and $<x_1 ,x_0>$, of which the first one (from start to end) is called positive (+) and the other one negative (-), indicated as: $<x_0 ,x_1> = - <x_1 ,x_0>$. The two-dimensional simplex has six combinations S_2: $<x_0,x_1,x_2>$, $<x_1,x_2,x_0>$, $<x_2,x_0,x_1>$, $<x_2,x_1,x_0>$, $<x_0,x_2,x_1>$, and $<x_1,x_0,x_2>$. The first three have the opposite orientation from the last the three combinations, so one can state $<x_0,x_1,x_2> = - <x_2,x_1,x_0>$. The positive orientation is counter clockwise (+) and the negative orientation is clockwise (-). For the three-dimensional simplex $S_3 = <x_0 ,x_1 ,x_2 ,x_3>$ there are 24 different combinations of which 12 are related to positive oriented tetrahedrons (+, all normal vectors outside) and the other 12 are negative oriented tetrahedrons -, all normal vectors inside). As there are several equivalent notations (combinations), it is possible to agree on a preferred notation; e.g., the combination related to a positive orientation with the nodes with lowest id's (indices) first. According to the Poincaré algebra (Geoghegan 2005), the boundary of a simplex is defined by the following sum of (n-1) dimensional simplexes (omitting the ith node and with alternating + or – sign):

$$\partial S_n = \sum_{i=0}^{n} (-1)^i < x_0, ..., \hat{x}_i, ..., x_n >$$

So, the boundary of $\partial S_1 = <x_0,x_1>$ is $<x_1> - <x_0>$ and the boundary of $\partial S_1^{neg} = <x_1,x_0>$ would be $<x_0> - <x_1>$. The boundary of $\partial S_2 = <x_0,x_1,x_2>$ is $<x_1,x_2> - <x_0,x_2> + <x_0,x_1>$. In a similar way, the boundaries related to the 5 other combinations of S_2 can be given. Finally, the boundary of $\partial S_3 = <x_0,x_1,x_2,x_3>$ is $<x_1,x_2,x_3> - <x_0,x_2,x_3> + <x_0,x_1,x_3> - <x_0,x_1,x_2>$ (and the same for the other 23 combinations). Going to the boundaries of the boundaries of a tetrahedron, that is the boundary of the triangles (edges), it can be observed that every edge is exactly used once in the positive direction and once in the negative direction (within the tetrahedron). Another interesting result from the Poincaré algebra is the number of lower dimensional simplexes used as (in)direct boundary (face) of a given simplex:

$$S_n \text{ has} \begin{pmatrix} n+1 \\ p+1 \end{pmatrix} \text{faces of dimension } p \text{ with } (0 \leq p < n)$$

So, S_2 (triangle) has 3 0D simplexes (nodes) and 3 1D simplexes (edges) as boundary ‘faces’. The simplex S_3 has respectively 4, 6 and 4 0D, 1D and 2D simplexes as boundary ‘faces’. When neighbor simplexes of the same dimension are joined or merged, then their shared boundary is removed as shown in Figure 2. For example, take neighbor triangles $<x_0,x_1,x_2>$ and $<x_0,x_2,x_3>$ then adding the boundaries results in: $(<x_1,x_2> - <x_0,x_2> + <x_0,x_1>) + (<x_2,x_3> - <x_0,x_3> + <x_0,x_2>) = <x_1,x_2> + <x_0,x_1> + <x_2,x_3> - <x_0,x_3> = <x_1,x_2> + <x_0,x_1> + <x_2,x_3> + <x_3,x_0>$. Note that the shared boundary $<x_0,x_2>$ is removed. Similarly, when merging the two neighbor tetrahedrons $<x_0,x_1,x_2,x_3>$ and $<x_0,x_2,x_4,x_3>$, then adding the boundaries (triangles) results in $<x_1,x_2,x_3> + <x_0,x_1,x_3> + <x_2,x_1,x_0> + <x_2,x_4,x_3> + <x_3,x_4,x_0> + <x_4,x_2,x_0>$. When looking at the edges again, then it can be observed that every edge is used once in the positive direction and once in the negative direction. A set of merged (joined) neighbor n-simplexes is called a simplicial complex (or n-cell). It is also possible to create a topological structure consisting of connected n-simplexes (with all their lower level boundaries: 0,...,n-1 simplexes) partitioning the whole n-dimensional domain. In 3D, this is then called the tetrahedronized network (TEN). In such a network, it is not only interesting to give the boundary of a simplex, but also to give the coboundary. For example, the boundary of a triangle is formed by the 3 edges and the coboundary is formed by the 2 tetrahedrons. Similarly, the boundary of an edge is formed by 2 nodes and the coboundary is formed by 2 or more triangles.



$$S_{21} =< x_0, x_1, x_2 > \text{and } S_{22} =< x_0, x_2, x_3 >$$
$$C_2 =< x_1,x_2 > \ - < x_0,x_2 > + \ < x_0,x_1 >$$
$$+ \ < x_2,x_3 > \ + < x_0,x_2 > + \ < x_3,x_0 >$$
$$=< x_1,x_2 > \ + \ < x_0,x_1 > \ + \ < x_2,x_3 > \ + \ < x_3,x_0 >$$

$$S_{31} =< x_0, x_1, x_2, x_3 > \text{and } S_{32} =< x_0, x_2, x_4, x_3 >$$
$$C_3 =< x_1, x_2, x_3 > - < x_0, x_2, x_3 > + < x_0, x_1, x_3 >$$
$$- < x_0, x_1, x_2 > + < x_2, x_4, x_3 > - < x_0, x_4, x_3 >$$
$$+ < x_0, x_2, x_3 > + < x_4, x_2, x_0 >$$
$$=< x_1, x_2, x_3 > + < x_0, x_1, x_3 > - < x_0, x_1, x_2 >$$
$$+ < x_2, x_4, x_3 > - < x_0, x_4, x_3 > + < x_4, x_2, x_0 >$$

Fig. 2. Adding simplex with a neighbor to form complex of simplex in (a) 2D and (b) 3D.

2.2 3D Topography: TEN based conceptual model

The closest data models to our model are implemented in the Panda system (Egenhofer et. al. 1989b) and Oracle Spatial (Kothuri et. al. 2004, Oracle 2005), both of which consider up to two-dimensional spaces. Panda is based on complexes, which are unions of neighbor simplexes also called n-cells. There is no attention called for the feature modeling as opposed to our system.

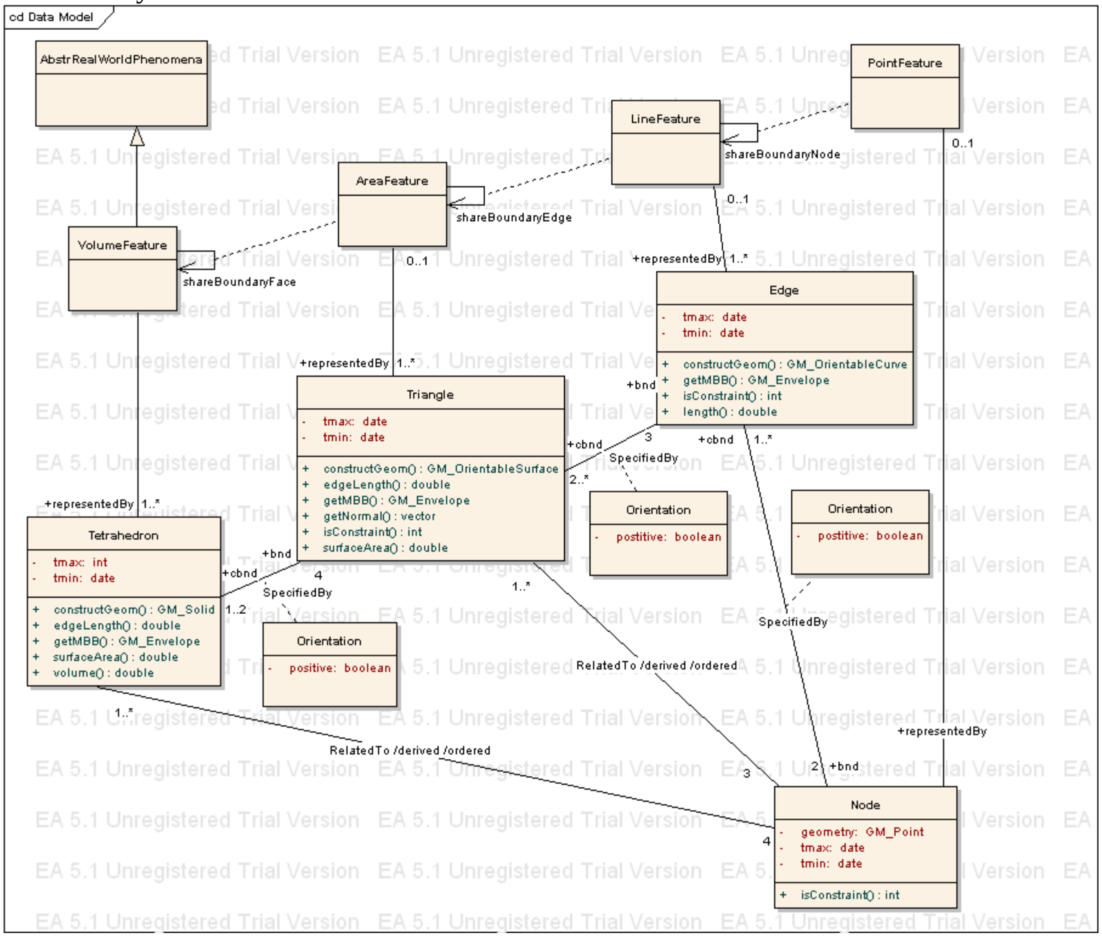


Fig. 3. UML class diagram of our first TEN model (i.e., model 1)

Although the topographic model is very strict in its way of handling only volume (and implied area, line and point) features, the actual TEN implementation will be more generic and also support true point, line and area features (which might be good representations at smaller scales for features that are in reality also volumes). For other cases, line and point features might be useful, too. Three different conceptual models (UML class diagrams) have been created. These are all more or less identical as they all capture the tehraheronized network structure and the features embedded. However, there are already remarkable differences in these conceptual models. As these conceptual models form the start for the logical and technical models (implementation), this is an important issue. We will therefore present three different conceptual models (in UML class diagrams). The first one is given in Figure 3 as model 1. The primitives are directed (positive) and the boundary/coboundary associations between node and edge, edge and triangle, and triangle and tetrahedron are signed (indicated by the association class Orientation). An efficient implementation of this model will not explicitly include the association class Orientation, but will use signed (+/-) references to encode the orientation. The association between tetrahedron (or triangle) and node can be derived (and gives a correct ordering of the nodes within the primitive).

The second model (presented in Appendix A) is based on the first model, but instead of the association classes with explicit Orientation indicating the sign/direction (+/-), new undirected classes are created, which are the counterparts of their directed origins. The model may suggest that both positive and negative versions of the directed primitives are stored (as the undirected counterpart is a composition of a positive and negative directed primitive), but this is not the case: only the positive oriented primitives are stored. Similar to the first model, the association between tetrahedron (or triangle) and node can be derived and is again ordered.

Finally, the third model (again depicted in Appendix A): this is the conceptual model based on Poincaré formulas: direct associations from node to all three other primitive classes (edge, triangle, tetrahedron). The ordering of the nodes to define the primitives is important as it implies the orientation. Based on these associations, now the other boundary/coboundary associations can be derived (tetrahedron-triangle and triangle-edge), and it should be noted that these are again signed. The main differences between all these models are: (1) which associations are 'explicit' and which are derived and (2) in case of signed association (references), is this modeled with an association class or with an additional undirected primitive? Somehow, the third model seems to be the least redundant with respect to references. However, it is common use to explicitly model (and store) the references between a primitive and its boundary. Therefore we will continue in this paper with the first (or second) model, but the orientation is implemented via +/- sign in the references (and not via explicit classes).

3 Incorporating the TEN structure in a spatial DBMS

In this section the following issues will be further discussed: incremental update within the TEN feature model (3.1), primitive update functions (3.2), feature level updates (3.3), and storage requirements (3.4).

3.1 Implementing an incremental algorithm

All data are stored in a spatial database. Initially the database is empty, and via incremental updates is should be brought from one consistent state into the next consistent state. The most straightforward implementation of the TEN structure consists of four tables with nodes, edges, triangles and tetrahedrons and a table with volume features. If one wants to add a feature (for instance, a building), one needs to ensure correct representation in the TEN model by enforcing the boundary faces of this building to be present. As tetrahedronization algorithms can only handle constrained edges, the building's surface first needs to be triangulated. The resulting edges are the input for the building tetrahedronization, which is performed separately from the TEN network. The complete set of edges is then inserted as constraints into the TEN model by an incremental tetrahedronization algorithm. Note that this is one specific procedure and more efficient/direct procedures and associated algorithms could be imagined (though not so easy to realize). As a last step, the volume feature table needs to be updated. A new record is created which links the building to the representing tetrahedrons and the previous 'air' tetrahedrons on the specific location are removed.

If one wants to remove this building from the data set, for instance because it is demolished, the record from the volume feature table can be deleted. At the same time, the TEN needs to be updated. The constraints on the edges of the surface triangulation can be removed only if this building is the only feature that is bounded by this constrained edge. In the case of the demolished building, constrained edges on the building's floor also bound the earth surface and therefore needs to remain present in the TEN model. The tetrahedrons that were previously representing the building now need to be re-classified, in this case, most likely just as 'air'. This reclassification is necessary to maintain the volume partition. At this moment, the building is entirely removed from the model, both on TEN and on feature level, but the deletion process is not finished. As a last step, it is necessary to check whether the TEN can be simplified by creating larger tetrahedrons or can be optimized by creating better-shaped tetrahedrons (by flipping; see 3.2). As an alternative, one might delete directly all edges that were part of the building, except for (constrained) edges that also contribute to the shape of other features. The resulting hole in the TEN needs to be re-triangulated and the created tetrahedrons will be linked to the 'air' feature.

New (volume) features that are inserted take over the space of the existing features. This will be not a problem in case of the air and ground tetrahedrons. However, in case of tehrahedrons belonging to other types of features, the correctness of this occupation has to be checked (by the user) before committing. Further, it should be noted that most features are also (indirectly) connected to the earth surface, and also this has to be translated into constraints, which can be used to validate changes. Now that the update process is described, the algorithm requirements can be extracted. For creating and maintaining the TEN, an incremental algorithm is required. Due to the potential enormous amount of data, this incremental algorithm has to work in the database and should preferably impact the TEN structure as locally as possible. In the TEN, all simplexes should be available. As the tetrahedrons represent volume features, the triangles contain most topological relationships, the edges contain the constraints and the nodes contain the geometry. Another requirement is the need for numerical stability through detection and repair of ill-shaped triangles and tetrahedrons. Shewchuk has performed a lot of research (Shewchuk 1997, Shewchuk 2004) in the field of Delaunay mesh refinement in both 2D and 3D.

3.2 Basic updating of the topological elements

The basic update procedures to modify an existing topology complex are described here, which are low level editing operations and are usually done in three steps: First, the user decides a window to be updated in a user session. A lock operator is executed after specifying the area of interest. The database will lock all of the features and topological elements overlapping the specified area of interest to prevent other users from updating the same area (Oracle 2005). Second, the user selects all of the required pieces of topology and features into memory and operates only on them. At this stage, new topological elements can be added or old ones can be removed as well. Third, all of the changes done in the session are committed back to the database. Next, we describe some of the basic functions available in the interface. The operations on the primitives consist of the following:

1. Move node (only allowed without destroying topology structure)
2. Insert node and incident edges/triangles/tetrahedrons (or the reverse operation 'remove node'), where 3 cases can be distinguished depending on where the node is inserted (see Figure 4):
   - Middle of tetrahedron (one tetrahedron involved) and added are +1 node, +4 edges, +6 triangles, and +3 tetrahedrons (respectively the 0/1/2/3-simplexes).
   - Middle of triangle (2 tetrahedrons involved) and added are +1 node, +5 edges, +7 triangles, and +4 tetrahedrons.
   - Middle of edge (*n* tetrahedrons involved) and added are +1 node, *+(n+1)* edges, *+2n* triangles, *+n* tetrahedrons.
3. Flipping of tetrahedrons, two cases, depending on configuration (see Figure 5):
   - 2-3 bistellar flip
   - 4-4 bistellar flip

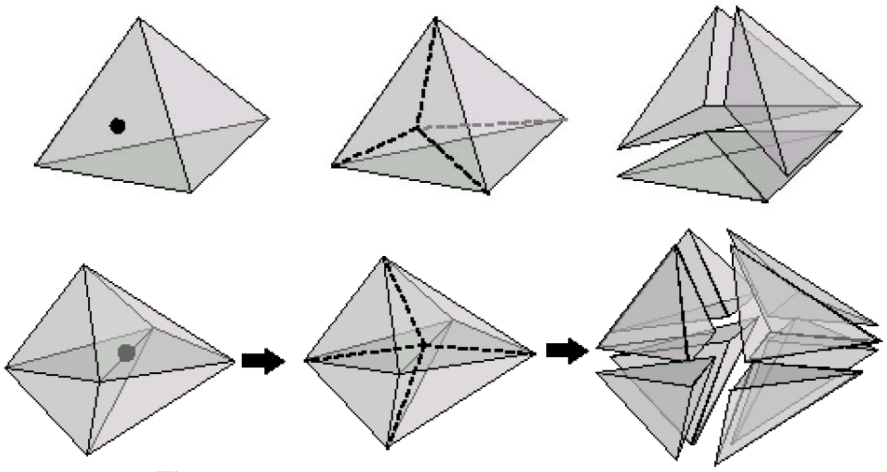


Fig. 4. Inserting a node: in triangle, neighbor tetrahedron not displayed) (above) and in edge with four incident tetrahedrons (below), both taken form (van der Most 2004)

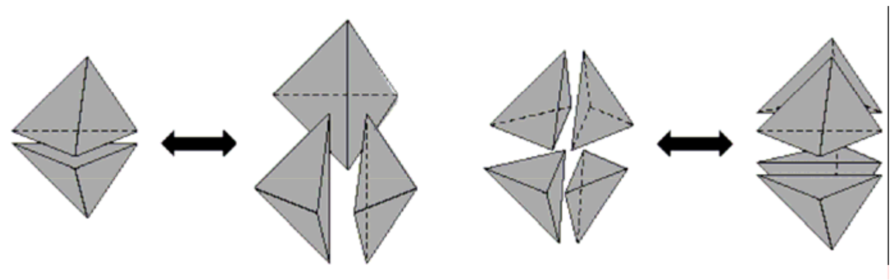


Fig. 5. Flipping in 3D 2-3 bistellar flip (left) and 4-4 bistellar flip (and right), again taken from (van der Most 2004, Verbree et. al. 2005).

The feature mapping when topology is updated can be described as follows: Since spatial features are defined on topological elements, it is very important to keep the integrity of spatial features even when updates are allowed on the underlying topological elements. The features define the constraint (nodes,) edges and triangles and care must be taken that these constraint simplexes are included within the TEN, in order to be able to represent the features. The other type of edges and triangles are introduced to make the TEN structure a ‘complete’ tetrahedronization (and during manipulation there is more freedom for these simplexes: flip, remove, etc.)

3.3 Feature level updates

It is also desirable to have an interface where feature geometry can be directly inserted into the topology complex returning a list of corresponding topological elements. This is more natural to users only dealing with the feature geometry and not necessarily caring about the actual storage model used for the geometry. In such cases, the interface can take the geometry as the input and insert into the topology complex. This operation will translate into a series of lower level topology update operations. At the end of this step, a list of primitives mapping to the input geometry are returned.

Feature creation from the primitives: In some cases, an existing feature needs a minor modification for some reason. For example, a segment of the road needs a small adjustment, which results in a new shape for the road. In simple feature model, this would result in updating the whole geometry for the road, even though one needs to change only a piece of the road. In the topological model, the edge (or edges) corresponding to the new shape is updated and the road automatically derives the new shape from its corresponding edges. In case of a 3D building, this could be a small extension to the back of a building.

3.4 Storage requirements

If one considers the tetrahedronization of the building in Figure 6, it will be clear that storing the building in a TEN requires a lot of storage. In Table 1 the required number of tetrahedrons, triangles, edges and nodes is compared to the number of volumes, faces, edges and points in a polyhedron approach.

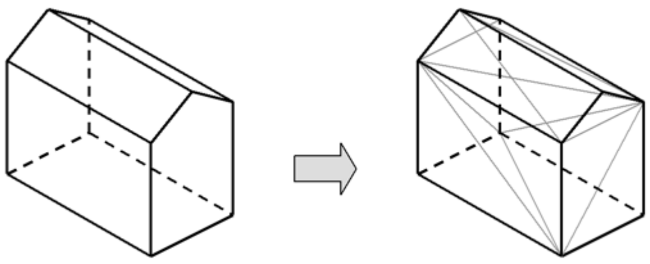


Fig. 6. Tetrahedronized building

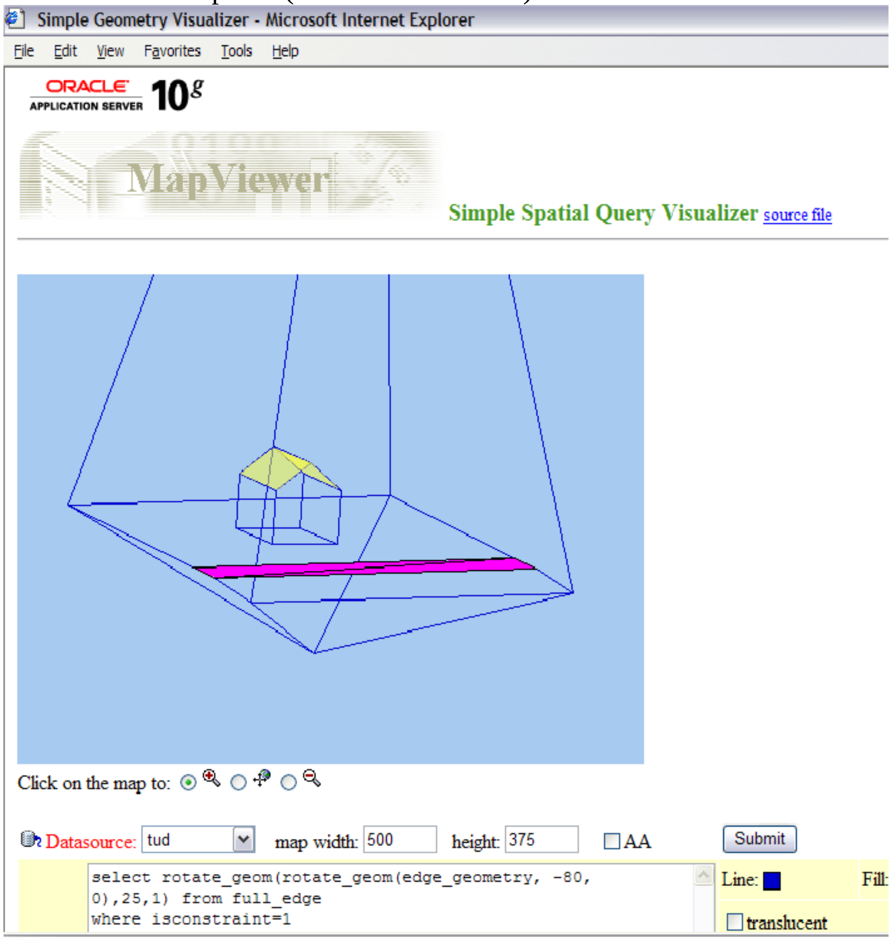
Table 1. Comparison between polyhedron and TEN model of the building

Building as polyhedron	Building as TEN
(1 volume)	8 tetrahedrons
7 faces	24 triangles
(15 edges)	25 edges
(10 points)	10 nodes

In order to reach acceptable performance, it has to be decided which relationships (as modeled in the class diagrams in Figure 3 and Appendix A) will be stored explicitly. The performance requirements do not tolerate full storage of all possible relationships. Several approaches exist in 2D to reduce storage requirements of TINs by either working with an edge or a triangle based approach, in which not both triangles, edges and nodes are stored explicitly. However, in the 3D situation and in the case of constraints in the TEN this is very difficult.

4 Implementation: first experiences

A small ‘toy’ data set is created by hand. It consists of an earth surface with a road on top and a single building with a saddle roof. This dataset was tetrahedronized by hand. In order to get 'air' and 'earth' tetrahedrons two extreme points were added, one on top and one at the fat bottom. Figure 7 shows the small data set, with the building and the road in front of it. This small data set, consisting of three volume features (building, air, earth), is composed by 56 tetrahedrons, 120 triangles, 83 edges and 20 nodes in Oracle Spatial (Kothuri et. al. 2004).


Simple Geometry Visualizer - Microsoft Internet Explorer
File Edit View Favorites Tools Help
ORACLE APPLICATION SERVER 10g
MapViewer
Simple Spatial Query Visualizer source file
Click on the map to:
Submit

Datasource: tud map width: 500 height: 375 ☐ AA

```
select rotate_geom(rotate_geom(edge_geometry, -80,
0),25,1) from full_edge
where isconstraint=1
```

Line: 🟦 Fill:

☐ translucent

Fig. 7. Small test data set rotated ‘3D view’

The database tables (node, edge, triangle, tetrahedron) contain mainly references, geometry is only stored in the node table (note this is the physical model corresponding to conceptual model 1 in Figure 3).

Functions are defined to obtain the geometry associated with the edges (get_edge_geometry) and the triangles (and test). These are then the counterparts of the 'constructGeom' methods in the conceptual UML models. One example will be given to obtain the geometry for the edge (including the definition of a view with geometry 'full_edge' and a functional spatial index on the 3D geometry of the edge):

```
create view full_edge as
select a.*, get_edge_geometry(eid) edge_geometry from edge a;

insert into user_sdo_geom_metadata values('EDGE',
   'TUD.GET_EDGE_GEOMETRY(EID)',
   sdo_dim_array(sdo_dim_element('X', -100000, 100000, 0.05),
   sdo_dim_element('Y', -100000, 100000, 0.05),
   sdo_dim_element('Z', -100000, 100000, 0.05)), null);

drop index edge_Sidx;
create index edge_sidx on edge(get_edge_geometry(eid))
indextype is mdsys.spatial_index
parameters('sdo_indx_dims=3')
```

The realization of a simple 3D viewer was based on an available 2D viewer i.e., Oracle AS MapViewer (Kothuri et. al. 2004) and the implementation of one 3D rotation function 'rotate_geom' (suitable for any type of Oracle spatial geometry). Further by depth sorting (after rotation) also hidden line hidden surface may be obtained (painter algorithm). Further nice features are semi-transparency. Further simple improvements could be the development of a GUI that defines the rotation angles for a set of views.

5 Conclusions and further research

Extending topographic data models into 3D is most relevant at large scale topography. However, this will lead to a substantial increase of data volume. With this increase, ensuring data integrity and maintaining performance become important requirements. As a result, implementing the 3D data structure in a spatial database is a sensible thing to do. In this paper, we developed a new topological 3D data model that relies on

Poincaré algebra. We described an implementation of this 3D model on a commercial DBMS. We also showed how a 2D visualizer can be extended to visualize these 3D objects. In this research, the Tetrahedronized Irregular Network (TEN) is selected as an internal data structure. The selection of this structure, motivated by computational advantages, the well-definedness of the triangles (always flat), the presence of well-known topological relationships, easy maintenance, visualization of triangles, and flexibility of forming more complex objects. As future work, we will work on temporal topology where there are no holes and overlap in time.

Alternative physical models than the one presented in Section 4 (based on conceptual model 1) are possible; for example the direct specification of the nodes of a tetrahedron and views for edges and triangles. Issues to be kept in mind are: 1. What to do with the 'isconstraint' attributes? (these attributes might also be part of the view and could be computed). 2. Would these views be efficient enough for manipulation (updating and querying)? This is very difficult to estimate upfront. Future work will consist of experiments needed to discover these aspects. The well-definedness of the TEN model comes with a prize of a large number of (conceptual) simplexes, therefore it is important to investigate in detail the actual size of tables and indices and evaluate what to store explicitly and what to derive (and present as a view).

Acknowledgements

This publication is the result of the Bsik Space for Geo-information '3D Topography' research project. Thanks to John Herring, Siva Ravada, Ravi Kothuri and Han Wammes for their constructive discussions on 3D TEN modeling, which formed the basis of this paper. Their ideas and comments were of great help, but all (remaining) errors are the sole fault of the authors. Further, Friso Penninga and Peter van Oosterom participate in the research program 'Sustainable Urban Areas' (SUA) carried out by Delft University of Technology.

References

Arens, C., Stoter, J., and van Oosterom, P. (2005). Modeling 3D spatial objects in a geo-DBMS using a 3D primitive, In: Computers & Geosciences, Volume 31, 2, pp. 165-177

Carlson, E. (1987). Three-dimensional (3D) modeling in a geographical database. In: Auto-Carto 8, pp.336-345

Egenhofer, M.J., Frank, A.U. and J.P. Jackson (1989a). A topological data model for spatial databases. In: Design and Implementation of Large Spatial, Lecture Notes in Computer Science, Vol. 409, pp. 271-286

Egenhofer, M. and Andrew, F. (1989b). PANDA: An Extensible DBMS Supporting Object-Oriented Software Techniques Database Systems in Office, Engineering, and Science, Zurich, Switzerland, T. Harder (ed.), Informatik Fachberichte, Vol. 204, Springer-Verlag, pp. 74-79

Geoghegan, R. (2005). Topological Methods in Group Theory, to appear in Springer.

Guibas, L. and Stolfi, J. (1985). Primitives for the manipulation of general subdivisions and the computation of voronoi diagrams. ACM Transactions on Graphics, Vol.4, No.2, pp.74-123

Kothuri, R., Godfrind, A., Beinat, E. (2004). Pro Oracle Spatial: The essential guide to developing spatially enabled business applications, Apress.

van der Most, A. (2004). An algorithm for overlaying 3D features using a tetrahedral network Master's Thesis TU Delft, 2004, 96 p.

van Oosterom, P. , Vertegaal, W., van Hekken, M., and Vijlbrief, T. (1994). Integrated 3D Modeling within a GIS. Int. Workshop on Advanced Geographic Data Modeling, pp. 80–95

van Oosterom, P. (1997). Maintaining Consistent Topology including Historical Data in a Large Spatial Database. In Proceedings Auto-Carto 13, Seattle WA, 8-10 April 1997, pages 327-336.

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

Oracle Spatial Topology and Network Data Models (2005). http://www.oracle.com/technology/documentation/spatial.html

Penninga, F. (2005). 3D Topographic data modeling: Why rigidity is preferable to pragmatism. In: Spatial Information Theory, Cosit 2005, Lecture Notes in Computer Science, Vol. 3693, pp.409-425

Shewchuk, J.R. (1997). Delaunay refinement mesh generation, PhD thesis, Carnegie Mellon University

Shewchuk, J. (2004). General-Dimensional Constrained Delaunay and Constrained Regular Triangulations I: Combinatorial Properties. To appear in: Discrete & computational Geometry. Available at: http://www-2.cs.cmu.edu/jrs.

Verbree, E., van der Most, A., Quak, W., van Oosterom, P. (2005). Towards a 3D Feature Overlay through a Tetrahedral Mesh Data Structure. In: Cartography and Geographic Information Science, Volume 32, Number 4, October 2005, pp. 303-314(12)

Vosselman, G. (2005). Sensing Geo-information, Inaugural address, ITC Enschede

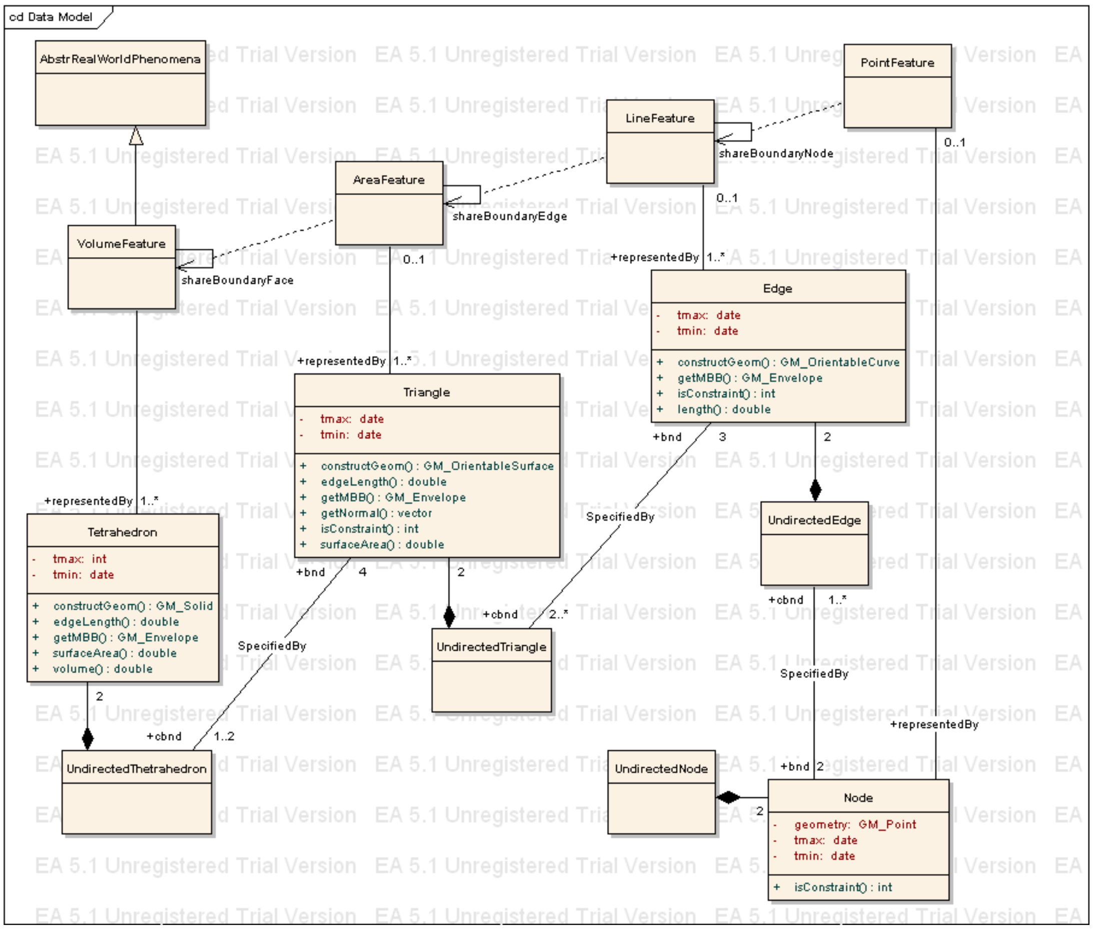
Zlatanova, S. (2000a). On 3D Topological Relationships. Int. Workshop on Database and Expert System Applications, pp. 913–919

Zlatanova, S. (2000b), 3D GIS for urban development, PhD thesis, Graz University of Technology.

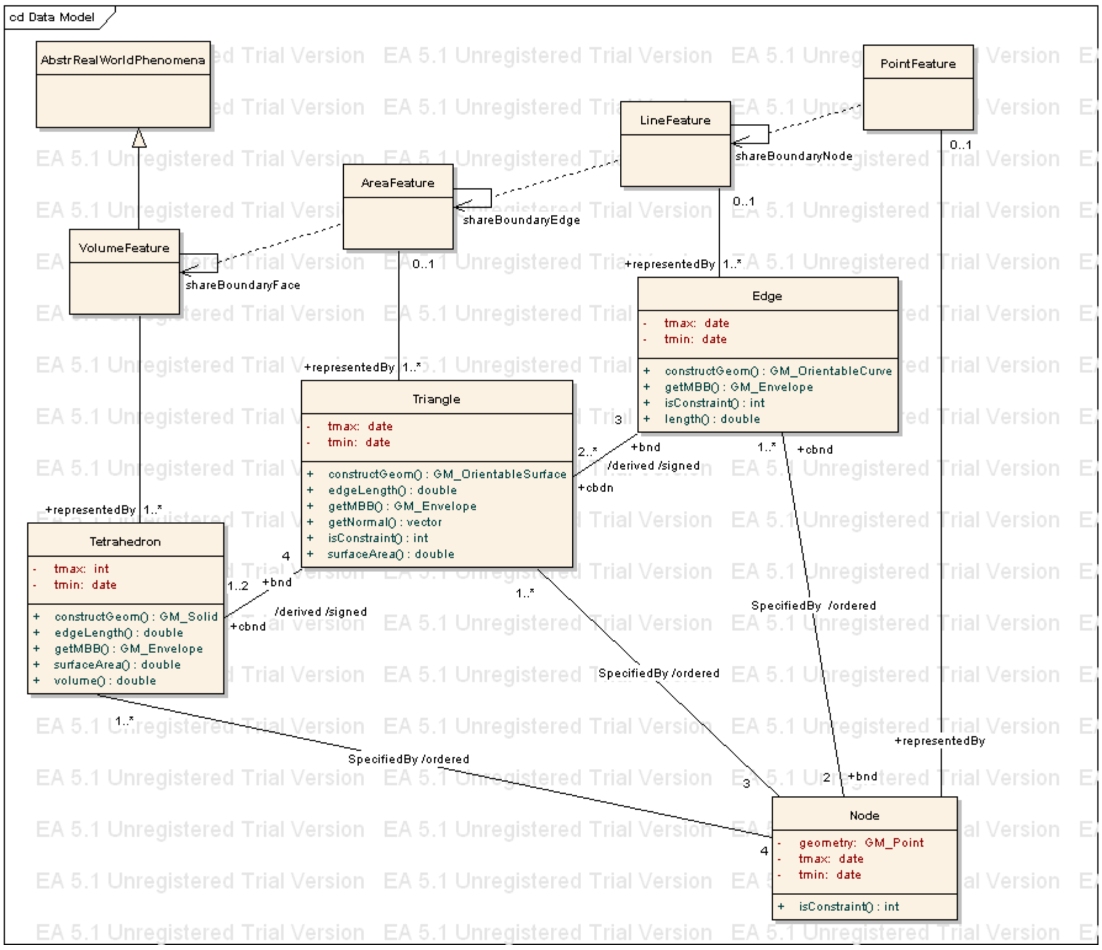
Zlatanova, S., Rahman, A. A., Pilouk, M. (2002a). 3D GIS: current status and perspectives, Proceedings of the Joint Conference on Geo-spatial theory, Processing and Applications, 8-12 July, Ottawa, Canada, 6p.

Zlatanova, S., Rahman, A. A., Shi, W. (2002b) Topology for 3D spatial objects, International Symposium and Exhibition on Geoinformation 2002, 22-24 October, Kuala Lumpur, Malaysia, 7p.

Appendix A



UML class diagram of our second model



UML class diagram of our third TEN model (based on Poincaré)