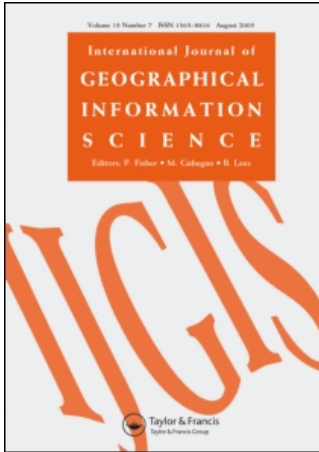


This article was downloaded by:[Tu Delft- Fac Citg]  
On: 20 May 2008  
Access Details: [subscription number 789195171]  
Publisher: Taylor & Francis  
Informa Ltd Registered in England and Wales Registered Number: 1072954  
Registered office: Mortimer House, 37-41 Mortimer Street, London W1T 3JH, UK



## International Journal of Geographical Information Science

Publication details, including instructions for authors and subscription information:  
<http://www.informaworld.com/smpp/title~content=t713599799>

### A simplicial complex-based DBMS approach to 3D topographic data modelling

F. Penninga<sup>a</sup>; P. J. M. Van Oosterom<sup>a</sup>

<sup>a</sup> Section GIS Technology, OTB, Delft University of Technology, 2628 BX Delft, the Netherlands

Online Publication Date: 01 January 2008

To cite this Article: Penninga, F. and Van Oosterom, P. J. M. (2008) 'A simplicial complex-based DBMS approach to 3D topographic data modelling', International Journal of Geographical Information Science, 22:7, 751 — 779

To link to this article: DOI: 10.1080/13658810701673535  
URL: <http://dx.doi.org/10.1080/13658810701673535>

PLEASE SCROLL DOWN FOR ARTICLE

Full terms and conditions of use: <http://www.informaworld.com/terms-and-conditions-of-access.pdf>

This article maybe used for research, teaching and private study purposes. Any substantial or systematic reproduction, re-distribution, re-selling, loan or sub-licensing, systematic supply or distribution in any form to anyone is expressly forbidden.

The publisher does not give any warranty express or implied or make any representation that the contents will be complete or accurate or up to date. The accuracy of any instructions, formulae and drug doses should be independently verified with primary sources. The publisher shall not be liable for any loss, actions, claims, proceedings, demand or costs or damages whatsoever or howsoever caused arising directly or indirectly in connection with or arising out of the use of this material.

## Research Article

# A simplicial complex-based DBMS approach to 3D topographic data modelling

F. PENNINGA\* and P. J. M. VAN OOSTEROM

Section GIS Technology, OTB, Delft University of Technology, Jaffalaan 9, 2628 BX  
Delft, the Netherlands

(Received 01 December 2006; in final form 18 August 2007)

This paper introduces a new compact topological 3D data structure. The proposed method models the real world as a complete decomposition of space and this subdivision is represented by a constrained tetrahedral network (TEN). Operators and definitions from the mathematical field of simplicial homology are used to define and handle this TEN structure. Only tetrahedrons need to be stored explicitly in a (single column) database table, while all simplexes of lower dimensions, constraints and topological relationships can be derived in views. As a result the data structure is relatively compact and easy to update, while it still offers favourable characteristics from a computational point of view as well as presence of topological relationships.

*Keywords:* 3D GIS; Spatial DBMS; Topology; 3D topography, Poincaré simplicial homology; Simplicial complexes

## 1. Introduction

### 1.1 Motivation

The real world consists of three-dimensional (3D) objects and these objects are getting more complex due to multiple land use, resulting in several buildings, roads, etc., which are built on top of each other. At the same time, increasing awareness of the importance of sustainable urban development and disaster management triggers the demand for more accurate and realistic data modelling and analysis. As a result focus within geographical information systems (GIS) is shifting from two-dimensional (2D) towards 3D modelling. Due to the intended applications (e.g. modelling noise propagation (Kluijver and Stoter 2003) or air pollution), the emphasis is also shifting from visualisation to querying and analysis.

In order to support all required operations 3D topographic (i.e. physical objects) data sets will be needed. Extending topographic data models into the third dimension is most relevant when dealing with large scale topography. As this will lead to a substantial increase in data volume, maintaining and ensuring data integrity becomes of extreme importance and choosing a topological approach is an obvious solution as topology enables better checks while updating (like detection of overlaps or gaps); see also Ellul and Haklay (2006). In this paper, a topological approach based on a tetrahedral network will be introduced.

---

\*Corresponding author. Email: F.Penninga@tudelft.nl

The current developments in the field of 3D topography are not only demand-driven. The increasing availability of high density laser scan data is most certainly a trigger in this process. Integrating 2D data with height data sets is an obvious objective when both data sets are available. Oude Elberink and Vosselman (2006) describe an integration of 2D data with height data in a topographic context (*see* figure 1 for example of input data). Besides integrating 2D data with height data (obtained by airborne laser scanning) direct 3D data acquisition by terrestrial laser scanning is emerging (*see* figure 2). A consequence of the increasing point density of laser scan data is a further increase in data volume of 3D data models. The ability to maintain data integrity becomes a crucial characteristic of a 3D data structure. This ability and the support for 3D analysis are the two main requirements for the approach introduced in this paper.

### 1.2 Research objective

This research aims at facilitating 3D topographic data modelling by designing a topological 3D data structure. The features are modelled in a constrained tetrahedral network (TEN), which offers advantages from a computational point of view as well as a set of well-known topological relationships. By applying operators and definitions from the mathematical field of simplicial homology to the constrained TEN, this paper offers a mathematically-robust 3D modelling approach. It results in a relatively compact and easy-to-update data structure, without compromising its required elements like facilitating analyses and validation operations.

### 1.3 Related research

For the last 20 years research has been performed in the field of 3D GIS. Zlatanova *et al.* (2002) give an overview of the most relevant developments in this period and

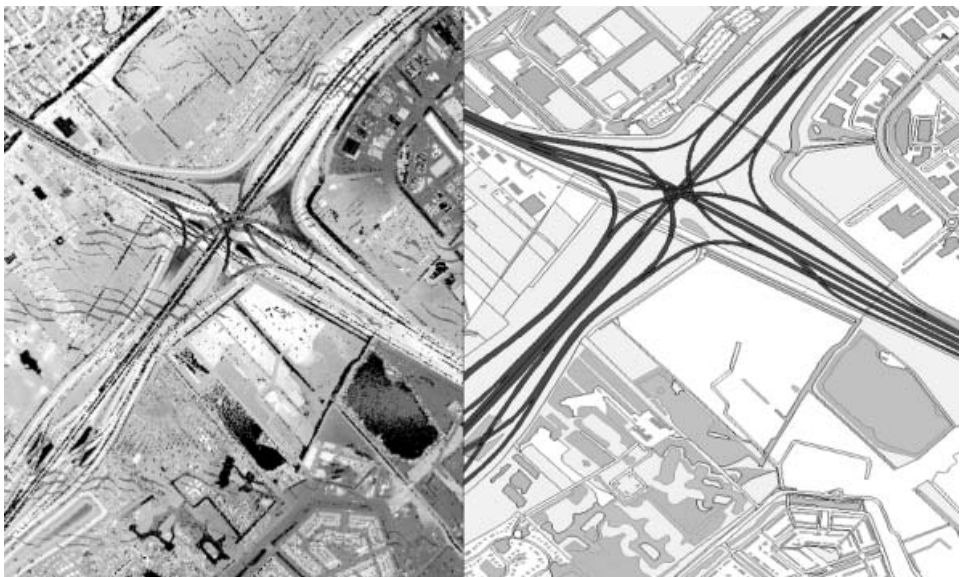


Figure 1. Height data (left) and topographic data (right) of highway interchange 'Prins Clausplein' near The Hague, the Netherlands (Courtesy of Sander Oude Elberink, ITC Enschede).

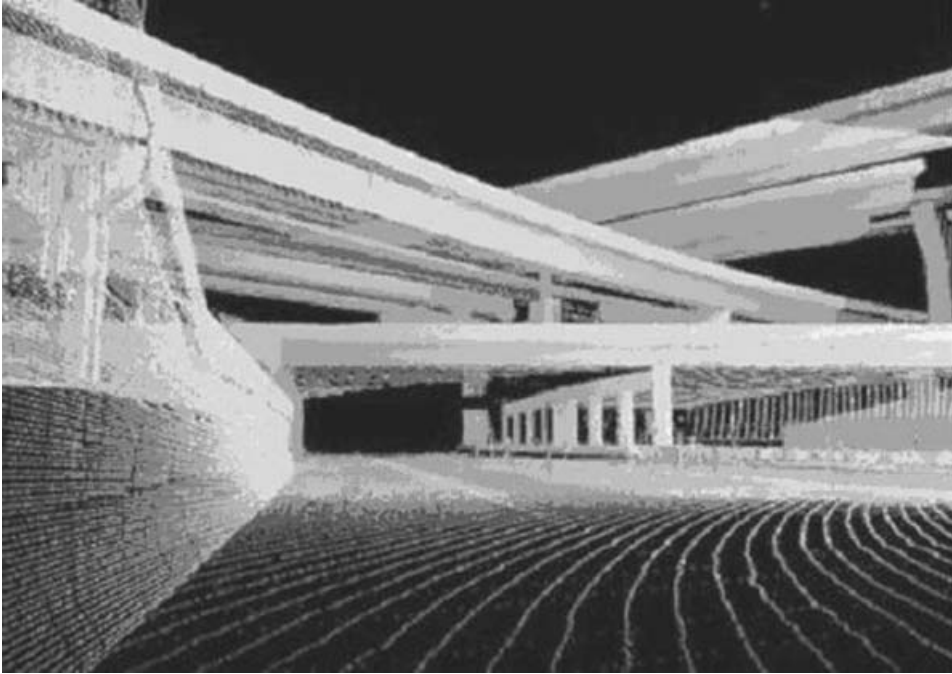


Figure 2. Terrestrial laser scanning acquires 3D data of complex objects.

Zlatanova *et al.* (2004) elaborate especially on the topological ones. With respect to the 3D approach introduced in this paper, Carlson (1987) can be seen as the starting point as he introduced a simplicial complex-based approach of 3D subsurface structures. However, he limited himself for reasons of simplicity to the use of 0-, 1- and 2-simplexes in 3D space. Nevertheless, he acknowledged the possibility of extending the simplex approach into  $n$  dimensions, as indicated by Frank and Kuhn (1986). The possibility of including 3D manifolds is explored by Pigot (1992, 1995), and Pilouk (1996) introduces the tetrahedral irregular network (TEN), in which also the 3-simplex is used as 3D manifold. In applications polyhedrons are often used as 3D primitives (Zlatanova 2000, Stoter 2004, Arens *et al.* 2005).

The concept of simplicial complexes and its mathematical description (part of the field of algebraic topology (Hatcher 2002)) is described by Giblin (1977). It is mentioned by Frank and Kuhn (1986) as one of the possible cell graph approaches. A topological data model based on 2D simplicial complexes in 2D space is introduced by Egenhofer *et al.* (1989) and implemented in the PANDA system (Egenhofer and Frank 1989), an early object-oriented database. The mathematical approach of simplexes is also used by Pigot (1992), but full applications of simplicial homology in three dimensions in a GIS context are not known to the authors.

#### 1.4 Outline

Section 2 will introduce two important aspects of the new modelling approach: first all features are by definition volumes and together these objects fill the complete 3D space and second a constrained tetrahedronised irregular network is used as basis. After that the mathematical foundation of the new approach is given in Section 3. Applying this mathematical theory is discussed in Section 4. To further illustrate the

new ideas a proof of concept implementation will be described in Section 5. This paper ends with conclusions, a discussion of the results and some ideas on further research in Section 6.

## 2. Characteristics of the new 3D modelling approach

The new modelling approach introduced in this paper has some specific characteristics. First, as described in Section 2.1, it is a full decomposition of space (a volume partition equivalent to a planar partition). Second, the features are represented internally in a tetrahedronised irregular network, which will be discussed in Section 2.2.

### 2.1 Full decomposition of space

With respect to modelling 3D topographic data, two fundamental observations are of great importance (Penninga 2005):

- (i) physical objects have by definition a volume. In reality, there are no point, line or polygon objects; only point, line or polygon representations exist (at a certain level of abstraction/generalisation). The ISO 19101 Geographic information – Reference model (ISO/TC211 2005) defines features as ‘abstractions of real world phenomena’. In most current modelling approaches, the abstraction (read ‘simplification’) is in the choice for a representation of lower dimension. However, as the proposed method uses a TEN (or mesh), the simplification is already in the subdivision into easy-to-handle parts (i.e. it is a finite element method);
- (ii) the real world can be considered a volume partition: a set of nonoverlapping volumes that form a closed (i.e. no gaps within the domain) modelled space. As a consequence, objects like ‘earth’ or ‘air’ are explicitly part of the real world and thus have to be modelled.

Although the model consists of volume features, some planar features might still be very useful, as they mark the boundary (or transition) between two volume features. In our modelling approach planar features can exist, but only as ‘derived features’. In terms of unified modelling language (UML) planar features would be modelled as association classes. For instance, a ‘wall’ might be the result of the association between a ‘building’ and the ‘air’. It is important to realise that planar features that mark borders between volumes might be labelled (for instance as ‘roof’ or ‘wall’ with additional attributes), but that they do not represent or describe the building. In this example the building in itself is represented by a volume, with neighbouring volumes that represent air, earth or perhaps another adjacent building.

As a result the actual 3D model will show more resemblance with the real world. Deriving visualisations from this model might result in more simplified/generalised models, as there is no one standard ‘best’ visualisation for all purposes. However, the choice of representation to use should be made in the digital cartographic model (DCM, a set of cartographic rules) and not in the digital landscape model that contains the 3D physical objects (for DCM and DLM, *see* Kraak and Ormeling (1996)).

The explicit inclusion of earth and air features is not very common, as these features are often considered as empty space in between topographic features. However, this inclusion is not only serving the abstract goal of ‘clean’ modelling, but

actually has some useful applications. First, the air and earth features do not just fill up the space between features of the other types, but are often also subject of analyses. One can think of applications like modelling noise propagation or air pollution. Second, by introducing earth and air features future extensions of the model will be enabled. Space that is currently labelled as air can be subdivided into for instance air traffic or telecommunication corridors, while earth might be subclassified in geographic layers or polluted regions. Figure 3 shows some examples of future extensions.

## 2.2 Representing features in a tetrahedronised irregular network

After initial ideas on a hybrid data model (an integrated TIN/TEN model, based on the pragmatic point of view: model in 2.5D where possible and only in exceptional cases switch to a full 3D model) the decision was made (Peninga 2005) to model all topographic features in a TEN. The preference for these simplex-based data structures is based on certain qualities of simplexes (in Section 3.1 simplexes will be discussed in a more formal way):

- (i) well defined: a  $n$ -simplex is bounded by  $n+1$   $(n-1)$ -simplexes, e.g. a 2-simplex (triangle) is bounded by 3 1-simplexes (edges);
- (ii) flatness of faces: every face can be described by three points;
- (iii) a  $n$ -simplex is convex (which simplifies among others point-in-polygon tests).

A disadvantage of simplexes is the introduction of a 1: $n$  relationship between features and their representations. A related issue is the size of these models, both in terms of disk storage and memory usage during triangulation/tetrahedronisation. However, this complexity should be tackled by the algorithms and preferably remain hidden to the average user. Impressive results in terms of speed and memory usage are described by Isenburg *et al.* (2006a) (and repeated in a specific GI-context in Isenburg *et al.* (2006b)). In our volumetric approach, the emphasis is on the volume features, although some less dimensional features can be identified as ‘derived’ features. Figure 4 illustrates the concept of a TEN-based volumetric approach. As one can see, the real world phenomena are represented by volume features. Internally these features are represented as set of tetrahedrons. Two volume features might share a boundary face that is important enough to be identified as area feature. Note that these area features are modelled as association classes in the UML class diagram and therefore are lifetime-dependent of the relationship between the two volume features.

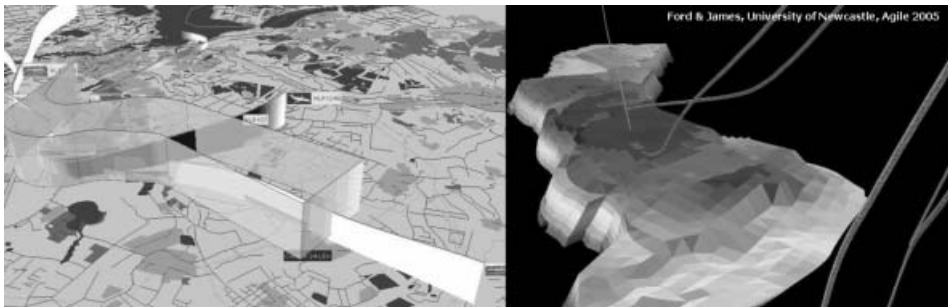


Figure 3. Air traffic corridors towards Schiphol Airport, the Netherlands (left) and an oil reservoir (right): examples of future extensions.

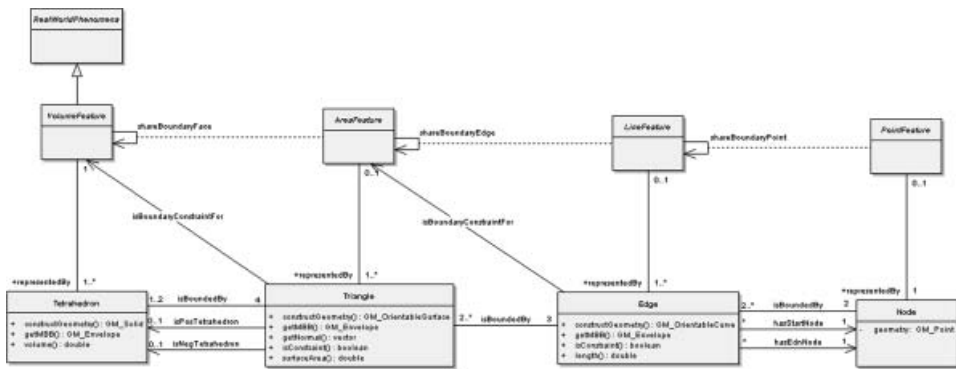


Figure 4. UML class diagram of the TEN-based volumetric modelling approach.

The most important question is how topographic features are represented within the TEN. The UML class diagram illustrates the use of constraints to ensure the presence of the feature boundaries in the TEN structure. In the 2D case (TIN: triangulated irregular network) constrained edges are used to guarantee that polygon boundaries remain present, regardless of other point insertion or edge update operations. In the current 3D case, one would like to ensure the presence of faces, as they bound the volume features, but unfortunately 3D algorithms still work with constrained edges (Shewchuk 2004). As a result using constraints requires a two-step approach: first one needs to ensure the presence of all constrained edges, then one needs to check whether the required constrained faces are present (i.e. whether they are intersected by other edges). To further illustrate the concept of representing features in a TEN, the following four steps are required to insert a volume feature in a TEN (Penninga and van Oosterom 2006):

- (i) its outer boundary needs to be triangulated and all resulting edges (and faces) should be treated as constraints;
- (ii) the interior needs to be tetrahedronised. This tetrahedronisation can be performed either directly in the TEN or separately, after which all resulting edges can be inserted into the TEN. Input in both cases is the set of constraint edges of the outer boundary;
- (iii) regardless which of the two previous options is used, local re-tetrahedronisation might be necessary in order to optimise the structure by creating better-shaped tetrahedrons;
- (iv) updating the relevant feature table(s).

The different steps in such an operation can be seen in figure 5. The input data set contains all feature boundaries, the constrained tetrahedronisation still holds these boundaries and the resulting model (where only feature boundaries are drawn) shows the same terrain and house.

### 3. Mathematical foundation

Simplexes and the relationships between simplexes of different dimensions were studied by mathematicians in the late 19th and early 20th century. This field of mathematics was known as simplicial homology and is today considered part of the field of algebraic topology (Hatcher 2002). Simplicial homology is the part of mathematics that deals with topological constructions of simplexes; the simplest

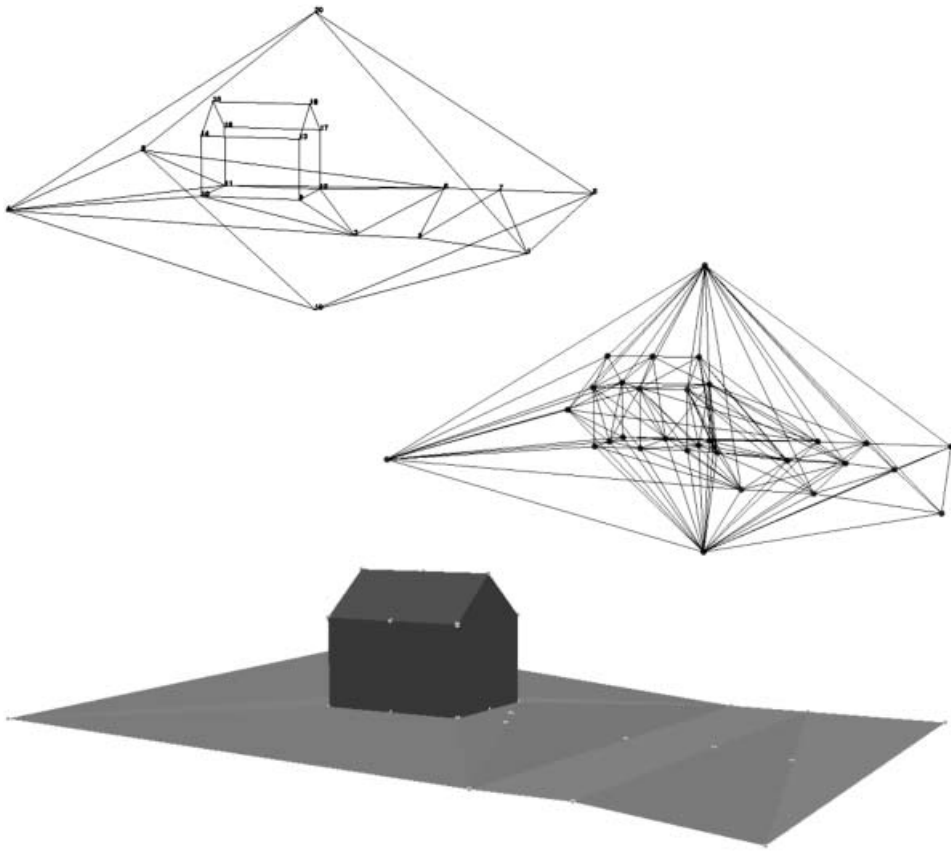


Figure 5. Input data (top), the resulting tetrahedronisation (mid) and as output the constrained triangles (i.e. the feature boundaries) (bottom).

geometries in each dimension. The foundations of simplicial homology are described by Jules Henri Poincaré (1854–1912) in Poincaré (1895). Some relevant corrections and additions can be found in Poincaré (1899). Since definitions and operators from simplicial homology are crucial in obtaining the research objectives (compact data structure, easy-to-update, provable data integrity), this section will provide relevant background information on simplicial homology. Simplicial homology will be introduced in Section 3.1, after which respectively orientation of simplexes in Section 3.2 and merging simplexes into simplicial complexes in Section 3.3 will get special attention. Some operations on simplexes and simplicial complexes will be discussed in Section 3.4. This mathematical approach will permit us to efficiently store our simplicial complex-based approach in a database, as shown in Section 4. In this section the following annotations will be used:  $S_n$  for a simplex of dimension  $n$ ,  $\partial$  for the boundary and  $v_i$  for a point. Although in mathematical language node or vertex (both indicating the topological use) would be more appropriate terms, we will use point to avoid confusion.

### 3.1 Poincaré simplicial homology

The previously introduced volumetric approach uses tetrahedrons to model the real world. These tetrahedrons in the TEN structure consist of nodes, edges and

triangles. All four data types are simplexes. A formal definition (Hatcher 2002) of a  $n$ -simplex  $S_n$  is given below.

**Definition 3.1** A  $n$ -simplex  $S_n$  is the smallest convex set in Euclidian space (denoted  $\mathbb{R}^m$ ) containing  $n+1$  points  $v_0, \dots, v_n$  that do not lie in a hyperplane of dimension less than  $n$ . As the  $n$ -dimensional simplex is defined by  $n+1$  nodes, it has the following notation:  $S_n = \langle v_0, \dots, v_n \rangle$ .

Equivalent conditions to the hyperplane condition would be that the difference vectors  $v_1 - v_0, \dots, v_n - v_0$  are linearly independent or, if one considers  $v_0, \dots, v_n$  as set of vectors, that these vectors are affinely independent. In a less formal way one could describe a  $n$ -simplex  $S_n$  as the simplest geometry of dimension  $n$ , where simplest refers to minimising the number of points required to define such a simplex. For instance, one needs at least three points to define a 2D shape (a triangle) and the hyperplane condition states that these three points should not lie on the same line (since that would result in a 1D edge). This triangle is denoted as  $S_2 = \langle v_0, v_1, v_2 \rangle$ .

Some observations on simplexes:

- (i) it is assumed that all simplexes are ordered. With any  $n$ -simplex,  $(n+1)!$  distinct ordered simplexes are associated. All even numbers of permutations of an ordered simplex  $S_n = \langle v_0, \dots, v_n \rangle$  have similar orientation, all odd numbers of permutations an opposite one. So for instance the following two statements are true:

$$\begin{aligned} S_1 &= \langle v_0, v_1 \rangle = -\langle v_1, v_0 \rangle \\ S_2 &= \langle v_0, v_1, v_2 \rangle = -\langle v_0, v_2, v_1 \rangle = \langle v_1, v_2, v_0 \rangle = \\ &= -\langle v_1, v_0, v_2 \rangle = \langle v_2, v_0, v_1 \rangle = -\langle v_2, v_1, v_0 \rangle \end{aligned}$$

The first line can be read as ‘an edge directed from point  $v_0$  to point  $v_1$  has opposite orientation to the edge directed from point  $v_1$  to point  $v_0$ ’. This characteristic can be used to change simplex orientation by performing a single permutation, thus eliminating the need of using signed simplex descriptions;

- (ii) a face of  $S_n$  is a simplex of lower dimension whose vertices form a non-empty subset of  $\{v_0, \dots, v_n\}$ . In other words, a simplex is constructed of simplexes of lower dimension and these simplexes are defined by some of the points that define the original simplex. For instance, a tetrahedron  $S_3 = \langle v_0, v_1, v_2, v_3 \rangle$  consists of four triangles  $\langle v_1, v_2, v_3 \rangle$ ,  $\langle v_0, v_2, v_3 \rangle$ ,  $\langle v_0, v_1, v_3 \rangle$  and  $\langle v_0, v_1, v_2 \rangle$ . The formula to derive these less dimensional boundaries will be given in Definition 3.2;
- (iii) if the subset is proper (i.e. not the whole of  $\{v_0, \dots, v_n\}$ ) than the face is a proper face (Giblin 1977);
- (iv) a  $n$ -simplex has in total  $2^{(n+1)} - 2$  proper faces;
- (v) for the number of faces of a specific dimension, the following holds: a  $n$ -simplex has  $\binom{n+1}{p+1}$  faces of dimension  $p$  with  $(0 \leq p < n)$ . For instance: a tetrahedron consists of four triangles, six edges and four points;
- (vi) the 0- and 1-dimensional faces of a  $n$ -simplex form a complete graph on  $n+1$  vertices;
- (vii) the boundary of a  $n$ -simplex is defined by the following sum of  $n-1$  dimensional simplexes (Poincaré 1899) (the *hat* indicates omitting the specific node):

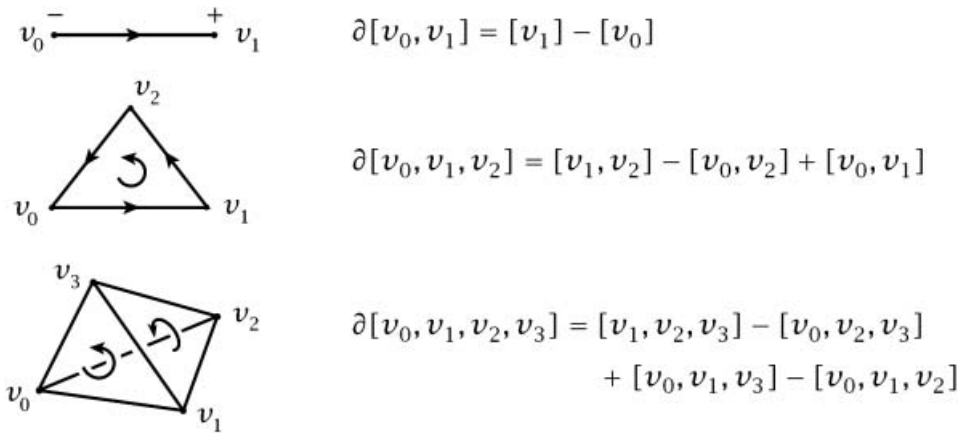


Figure 6. Simplexes and their boundaries (Hatcher 2002).

**Definition 3.2**

$$\partial S_n = \sum_{i=0}^n (-1)^i \langle v_0, \dots, \hat{v}_i, \dots, v_n \rangle$$

This results in (see figure 6) the following boundaries (note that only one permutation is showed in each dimension):

$$\begin{aligned} S_1 = \langle v_0, v_1 \rangle & \quad \partial S_1 = \langle v_1 \rangle - \langle v_0 \rangle \\ S_2 = \langle v_0, v_1, v_2 \rangle & \quad \partial S_2 = \langle v_1, v_2 \rangle - \langle v_0, v_2 \rangle + \langle v_0, v_1 \rangle \\ S_3 = \langle v_0, v_1, v_2, v_3 \rangle & \quad \partial S_3 = \langle v_1, v_2, v_3 \rangle - \langle v_0, v_2, v_3 \rangle \\ & \quad + \langle v_0, v_1, v_3 \rangle - \langle v_0, v_1, v_2 \rangle \end{aligned}$$

It is important to realise that all these observations are true in any dimension. As a result, simplicial homology definitions and operations are not only applicable to 2D and 3D modelling, but a simplicial homology-based modelling approach can also easily be extended into 4D, thus offering a solid mathematical foundation for spatio-temporal modelling. However, this paper will focus only on the 3D modelling approach.

**3.2 Orientation of simplexes**

Observation (i) introduced the concept of orientation of simplexes. In the 1D-case orientation is specified in terms of direction: an edge from A to B has opposite orientation to an edge from B to A. In 2D orientation is specified by order, i.e. edges are travelled clockwise or counter clockwise. By convention counter clockwise orientation is denoted positive (+) and clockwise orientation negative (-). In 3D orientation is specified by the direction of the normal vectors of the boundary faces. Normal vectors pointing outwards are denoted positive and normal vectors pointing inwards negative.

As a simplex  $S_n$  is defined by  $n+1$  vertices,  $(n+1)!$  permutations exist. All even numbers of permutations of an ordered simplex  $S_n = \langle v_0, \dots, v_n \rangle$  have the same orientation, all odd numbers of permutations have opposite orientation. So edge  $S_1 = \langle v_0, v_1 \rangle$  has boundary  $\partial S_1 = \langle v_1 \rangle - \langle v_0 \rangle$ . The other permutation  $S_1 = -\langle v_0,$


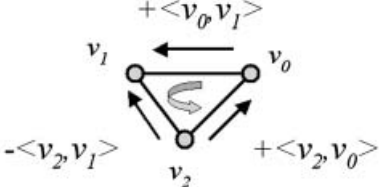





Six permutations of $S_2$	Orientation	Example: $\langle v_2, v_0, v_1 \rangle$
$\langle v_0, v_1, v_2 \rangle$	+ 	 $\partial \langle v_2, v_0, v_1 \rangle =$ $+ \langle v_0, v_1 \rangle - \langle v_2, v_1 \rangle + \langle v_2, v_0 \rangle$
$\langle v_0, v_2, v_1 \rangle$	- 	
$\langle v_2, v_0, v_1 \rangle$	+ 	
$\langle v_2, v_1, v_0 \rangle$	- 	
$\langle v_1, v_2, v_0 \rangle$	+ 	
$\langle v_1, v_0, v_2 \rangle$	- 	

Figure 7. The six permutations of simplex  $S_2$  and their orientation. Permutation  $\langle v_2, v_0, v_1 \rangle$  is illustrated in more detail.

$v_1 \rangle = \langle v_1, v_0 \rangle$  has boundary  $\partial S_1 = \langle v_0 \rangle - \langle v_1 \rangle$ , which is the opposite direction. In a similar way the boundaries of the other five permutations of  $S_2$  and the other 23 permutations of  $S_3$  can be given. The six permutations of  $S_2$  are illustrated in figure 7. It shows that two groups of three permutations are equivalent to each other, i.e. the three (1st, 3rd, 5th) with positive and the three (2nd, 4th, 6th) with negative orientation. As a consequence operators like the dual of a simplex (the dual of a simplex is the simplex with identical geometry and opposite orientation) become very simple: it only requires a single permutation.

This flexibility in handling orientations is a convenient characteristic of simplicial homology, but it brings another favourable characteristic in 3D.  $S_3$  has 24 permutations, 12 with positive and 12 with negative orientation. It holds for all 24 permutations that the four bounding triangles have identical orientation; either all normal vectors of the triangles point inwards or all normal vectors of the triangles point outwards. This is of course a desired characteristic and it requires no effort at all; as it is a direct result from Definition 3.2. Lemma 3.3 will prove this consistent orientation by applying the boundary operator twice:

Lemma 3.3 since  $\partial^2 S_n = 0$  ('the boundary of the boundary equals zero'), all boundary triangles of  $S_3$  have the same orientation.

*Proof* First the so-called zero homomorphism ( $\partial^2 = 0$ ) needs to be proven when applied to any oriented  $n$ -simplex. Now:

$$\begin{aligned}
 \partial^2 S_n &= \partial \sum_{i=0}^n (-1)^i \langle v_0, \dots, \hat{v}_i, \dots, v_n \rangle \\
 &= \sum_{i=0}^n (-1)^i \sum_{j=i+1}^n (-1)^{j-1} \langle v_0, \dots, \hat{v}_i, \dots, \hat{v}_j, \dots, v_n \rangle \\
 &\quad + \sum_{i=0}^n (-1)^i \sum_{j=0}^{i-1} (-1)^j \langle v_0, \dots, \hat{v}_j, \dots, \hat{v}_i, \dots, v_n \rangle
 \end{aligned}$$

All terms in this expression cancel in pairs, since each oriented  $(n-1)$ -simplex  $\langle v_0, \dots, \hat{v}_i, \dots, \hat{v}_j, \dots, v_n \rangle$  appears twice, the first time with sign  $(-1)^{i+j-1}$  and the second time with the opposite sign  $(-1)^{i+j}$ .

Now consider  $\partial^2 S_3$ . The boundary of a tetrahedron consists of four triangles, and the boundaries of these triangles consist of edges. Each of the six edges of  $S_3$  appears twice, as each edge bounds two triangles. The zero homomorphism states that the sum of these edges equals zero. This is the case if and only if the edges in these six pairs have opposite signs. The edges of two neighbouring triangles have opposite signs if and only if the triangles have the same orientation, i.e. either both are oriented outwards or both are oriented inwards.

### 3.3 Combining simplexes: simplicial complexes

As most volume features will be represented by more than one tetrahedron, operations on sets of simplexes will be useful. A simplicial complex is such a combinatorial object of a number of simplexes. A formal definition is given below:

**Definition 3.4** A simplicial complex  $C$  (Giblin 1977) is a finite set of connected simplexes that satisfies the following two conditions:

- (i) any face of a simplex from  $C$  is also in  $C$ ;
- (ii) the intersection of any two simplexes from  $C$  is either empty or is a face for both of them.

The dimension of  $C$  is the largest dimension of any simplex in  $C$  (Giblin 1977). A simplicial complex is said to be of homogeneous dimension  $n$  if all simplexes of lower dimension than  $n$  in  $C$  are proper faces (refer to Observation (iii) in Section 3.1) of  $n$ -simplexes in  $C$ .

An interesting application of the boundary formula is joining or merging two simplexes of equal dimension into a simplicial complex. The boundary of a simplicial complex can be derived by adding the boundaries of the separate simplexes. As most volume features will be represented by simplicial complexes, this operation will result in the volume feature boundary.

**Definition 3.5** The boundary of the simplicial complex  $C_n$  consisting of  $m+1$  simplexes of dimension  $n$  is defined as:

$$\text{Simplicial complex } C_n = \langle S_{n0}, \dots, S_{nm} \rangle \text{ has boundary } \partial C_n = \sum_{i=0}^m \partial S_{ni}$$

For example, if we join the two neighbouring triangles  $S_{20} = \langle v_0, v_1, v_2 \rangle$  and  $S_{21} = \langle v_0, v_2, v_3 \rangle$  into a 2D complex  $C_2$  (see figure 8), adding the boundaries results in:

$$\begin{aligned} \partial C_2 &= \partial S_{21} + \partial S_{22} = (\langle v_1, v_2 \rangle - \langle v_0, v_2 \rangle + \langle v_0, v_1 \rangle) + (\langle v_2, v_3 \rangle - \langle v_0, v_3 \rangle + \langle v_0, v_2 \rangle) \\ &= \langle v_1, v_2 \rangle + \langle v_0, v_1 \rangle + \langle v_2, v_3 \rangle - \langle v_0, v_3 \rangle \end{aligned}$$

Note that the shared boundary  $\langle v_0, v_2 \rangle$  is removed as it appeared once with positive and once with negative direction. This appearance with opposite signs relies on the assumption of similar orientation of the simplexes in the simplicial complexes. As long as this similar orientation is ensured, the zero homomorphism (see Lemma 3.3) will also apply to simplicial complexes:  $\partial^2 C_n = \sum_{i=0}^m \partial^2 S_{ni} = 0$ .

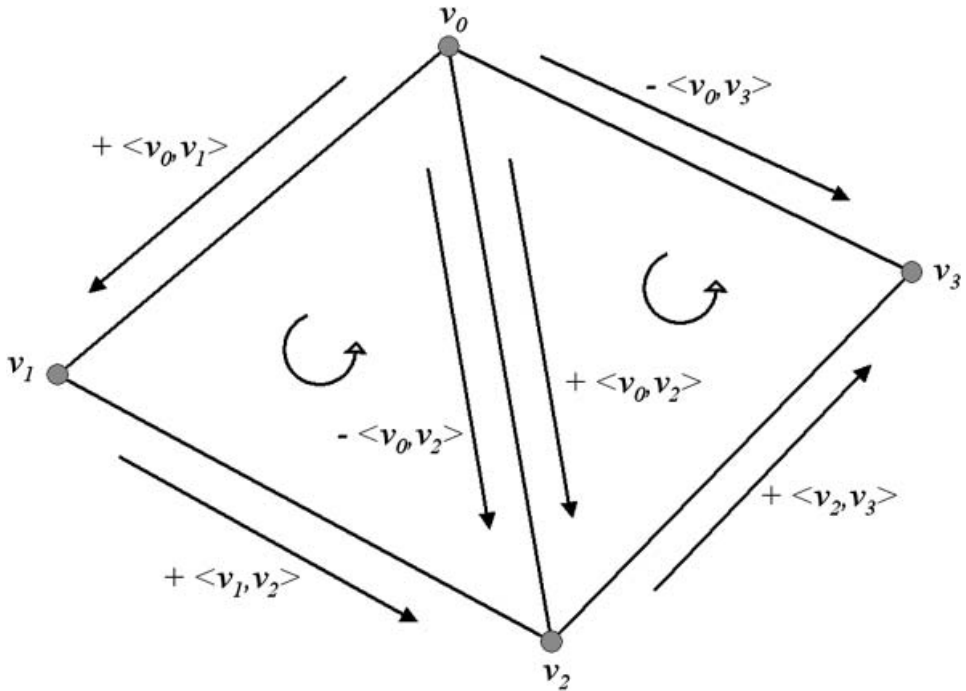


Figure 8. Merging two simplices into one simplicial complex.

As stated earlier, joining simplices into simplicial complexes and deriving its outer boundary can be very useful in our modelling approach. If for instance a building is modelled as a set of eight tetrahedrons (*see* figure 9), the building boundary representation can be obtained by merging the boundaries of all eight tetrahedrons. The triangles of  $C_3$  are the boundary triangulation of this building. This boundary triangulation might be used in the visualisation process. It is already a polyhedron, but if one is interested in a polyhedron with a minimal number of faces, merging boundary triangles with identical (within a tolerance) normal vector direction into flat polygons will result in seven flat boundary faces for this building.

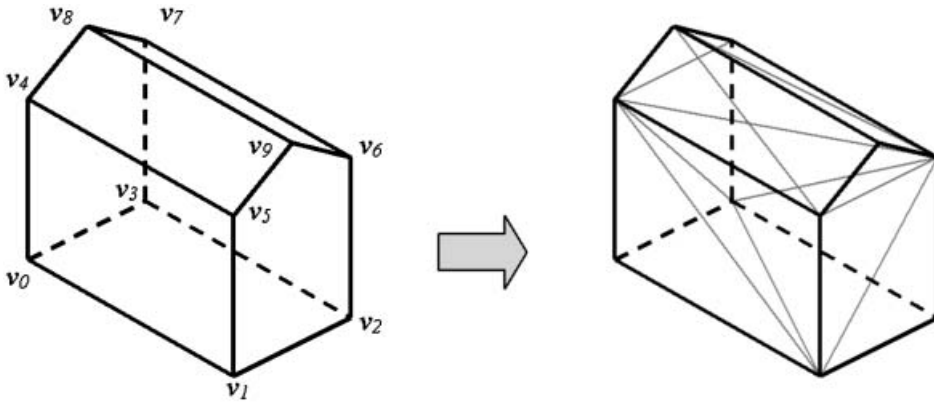
Up to this point only simplices and simplicial complexes are discussed. A special case of simplicial complexes in 3D is the TEN.

**Definition 3.6** A TEN is a simplicial complex of homogeneous dimension of three. This means that a TEN is a simplicial complex consisting only of face-connected 3-simplices that model the full 3D domain.

Such a structure contains several topological relationships, thus enabling both topological querying and, more important, validation tools in order to maintain data integrity. Another important concept with respect to topological relationships in a TEN structure is the coboundary. A coboundary is more or less the opposite of a boundary.

**Definition 3.7** The coboundary of a  $n$ -dimensional simplex  $S_n$  is the set of all  $(n+1)$ -dimensional simplices  $S_{n+1}$  of which the simplex  $S_n$  is part of their boundaries  $\partial S_{n+1}$ .

For example, a triangle has three boundary segments (its edges) and two coboundary segments (the adjacent tetrahedrons). An edge has two boundary segments (its nodes) and (in 3D!) an unknown number of coboundary segments (adjacent triangles).



$$\begin{aligned}
 S_{31} &= \langle v_0, v_1, v_3, v_4 \rangle & \partial S_{31} &= \langle v_1, v_3, v_4 \rangle - \langle v_0, v_3, v_4 \rangle + \langle v_0, v_1, v_4 \rangle - \langle v_0, v_1, v_3 \rangle \\
 S_{32} &= \langle v_1, v_2, v_3, v_6 \rangle & \partial S_{32} &= \langle v_2, v_3, v_6 \rangle - \langle v_1, v_3, v_6 \rangle + \langle v_1, v_2, v_6 \rangle - \langle v_1, v_2, v_3 \rangle \\
 S_{33} &= \langle v_1, v_3, v_4, v_6 \rangle & \partial S_{33} &= \langle v_3, v_4, v_6 \rangle - \langle v_1, v_4, v_6 \rangle + \langle v_1, v_3, v_6 \rangle - \langle v_1, v_3, v_4 \rangle \\
 S_{34} &= \langle v_1, v_4, v_5, v_6 \rangle & \partial S_{34} &= \langle v_4, v_5, v_6 \rangle - \langle v_1, v_5, v_6 \rangle + \langle v_1, v_4, v_6 \rangle - \langle v_1, v_4, v_5 \rangle \\
 S_{35} &= \langle v_3, v_4, v_6, v_7 \rangle & \partial S_{35} &= \langle v_4, v_6, v_7 \rangle - \langle v_3, v_6, v_7 \rangle + \langle v_3, v_4, v_7 \rangle - \langle v_3, v_4, v_6 \rangle \\
 S_{36} &= \langle v_4, v_6, v_7, v_8 \rangle & \partial S_{36} &= \langle v_6, v_7, v_8 \rangle - \langle v_4, v_7, v_8 \rangle + \langle v_4, v_6, v_8 \rangle - \langle v_4, v_6, v_7 \rangle \\
 S_{37} &= \langle v_4, v_5, v_6, v_8 \rangle & \partial S_{37} &= \langle v_5, v_6, v_8 \rangle - \langle v_4, v_6, v_8 \rangle + \langle v_4, v_5, v_8 \rangle - \langle v_4, v_5, v_6 \rangle \\
 S_{38} &= \langle v_5, v_6, v_8, v_9 \rangle & \partial S_{38} &= \langle v_6, v_8, v_9 \rangle - \langle v_5, v_8, v_9 \rangle + \langle v_5, v_6, v_9 \rangle - \langle v_5, v_6, v_8 \rangle +
 \end{aligned}$$


---


$$\begin{aligned}
 C_3 = & - \langle v_0, v_3, v_4 \rangle + \langle v_0, v_1, v_4 \rangle - \langle v_0, v_1, v_3 \rangle + \langle v_2, v_3, v_6 \rangle + \\
 & \langle v_1, v_2, v_6 \rangle - \langle v_1, v_2, v_3 \rangle - \langle v_1, v_5, v_6 \rangle - \langle v_1, v_4, v_5 \rangle - \langle v_3, v_6, v_7 \rangle \\
 & + \langle v_3, v_4, v_7 \rangle + \langle v_6, v_7, v_8 \rangle - \langle v_4, v_7, v_8 \rangle + \langle v_4, v_5, v_8 \rangle + \\
 & \langle v_6, v_8, v_9 \rangle - \langle v_5, v_8, v_9 \rangle + \langle v_5, v_6, v_9 \rangle
 \end{aligned}$$

Figure 9. Deriving the boundary triangulation from the TEN.

### 3.4 Operations on simplexes

In a simplicial complex-based approach features will be represented by a set of simplexes. As a result certain operations on features will translate into operations on simplexes. For instance a point-in-polyhedron test at feature level will be performed in the simplicial complex, as all simplexes are convex. Guaranteed convexity will enable the use of more efficient point-in-polyhedron algorithms.

Another example is the operation to obtain the volume of a building, which will be performed as the sum of the volume calculations of the individual tetrahedrons. The Cayley-Menger determinant (Colins 2003) is a determinant that gives the volume of a  $n$ -simplex in  $m$ -dimensional space. With simplex  $S_n = \langle v_0, \dots, v_n \rangle$  the  $(n+1) \times (n+1)$  matrix  $B = (\beta_{ij})$  is given by  $\beta_{ij} = |v_i - v_j|^2$ . Matrix  $\hat{B}$  is the  $(n+2) \times (n+2)$  matrix obtained by bordering matrix  $B$  with a top row  $(0, 1, \dots, 1)$  and a left column  $(0, 1, \dots, 1)^T$ .

**Definition 3.8** The volume  $V$  of a simplex  $S_n$  is given by:

$$V^2(S_n) = \frac{(-1)^{n+1}}{2^n(n!)^2} \det(\hat{B})$$

Since matrix  $\hat{B}$  consists of distances between vertices instead of vertex coordinates, the formula is dimension independent, meaning that it will produce the volume of a

$n$ -simplex, irrespective of the dimension of the space in which  $S_n$  is located. For  $n=2$  (a triangle) and  $n=3$  (a tetrahedron) this results in (with  $d_{ij}$  as length of edge  $\langle v_i, v_j \rangle$ ):

$$n=2 : -16V^2 = \begin{vmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & d_{01}^2 & d_{02}^2 \\ 1 & d_{10}^2 & 0 & d_{12}^2 \\ 1 & d_{20}^2 & d_{21}^2 & 0 \end{vmatrix} \quad n=3 : 288V^2 = \begin{vmatrix} 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & d_{01}^2 & d_{02}^2 & d_{03}^2 \\ 1 & d_{10}^2 & 0 & d_{12}^2 & d_{13}^2 \\ 1 & d_{20}^2 & d_{21}^2 & 0 & d_{23}^2 \\ 1 & d_{30}^2 & d_{31}^2 & d_{32}^2 & 0 \end{vmatrix}$$

Simplexes offer not only support for calculations, but also for more complex spatial operations like buffer and overlay. Verbree *et al.* (2005) describe the possibilities of executing these basic GIS operators on tetrahedrons. Validation is another operation that can be performed on both simplexes and features. At simplicial complex level the validation of a TEN can be performed by applying the Euler-Poincaré formula:  $N-E+F-V=0$  with  $N$  the number of nodes,  $E$  the number of edges,  $F$  the number of faces and  $V$  the number of volumes (including the exterior). Section 5.4 will describe validation in more detail, *see also* figure 10 for a 2D and 3D example of the Euler-Poincaré formula. Combining simplex validation results leads to validation on feature level. If one is interested in validating a volume feature, three checks need to be performed. First of all a valid TEN is required. Second, the boundary of the volume feature (represented by a set of constraints) should be watertight. The third and last criterion is that the interior of the volume feature is face connected, thus preventing the creation of two separate volumes that touch only on edge or node level.

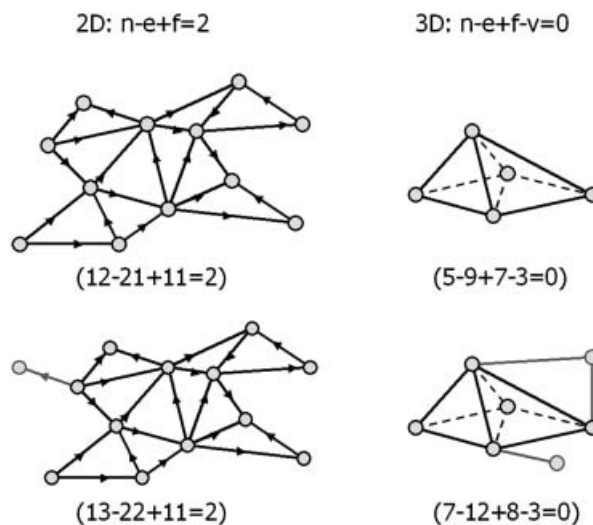


Figure 10. Using Euler-Poincaré in 2D and 3D for validation: dangling edges and faces remain undetected.

#### 4. Applying simplicial homology to the TEN: the simplicial complex-based DBMS approach

As shown in the previous section, a solid theoretical foundation exists for working with simplexes. Applying this mathematical theory on the TEN approach leads to a new conceptual model. After the introduction of this conceptual model in Section 4.1, the loosely related concept of vertex encoding will be described in Section 4.2. Based on the conceptual model and the concept of encoding vertices the DBMS implementation will be discussed in Section 4.3. Although out of scope of this paper, the section will end in Section 4.4 with some remarks on the required algorithms.

##### 4.1 Conceptual model

Compared to the initial ideas on the volumetric approach (*see* figure 4), the TEN structure itself is modified. Originally tetrahedrons were defined by four triangles, triangles by three edges and edges by two nodes. Geometry is stored at node level. As a result reconstructing geometry of for instance a tetrahedron is a relatively laborious operation. In simplicial homology, as described in the previous section, simplexes are defined by their vertices. Relationships between other simplexes, for instance between tetrahedrons and triangles, can be derived by applying the boundary operator (from Definition 3.2). As a result (Penninga *et al.* 2006), there is no need for explicit storage of these relationships. The simplicial complex-based TEN structure is illustrated in the UML class diagram in figure 11. The associations between the tetrahedron, triangle and edge class and the node class show that these simplexes are specified by an ordered list of nodes. The interrelationships between tetrahedrons, triangles and nodes (the boundary-coboundary relationships) are derived and signed (i.e. oriented). Note that for instance triangle and edge data

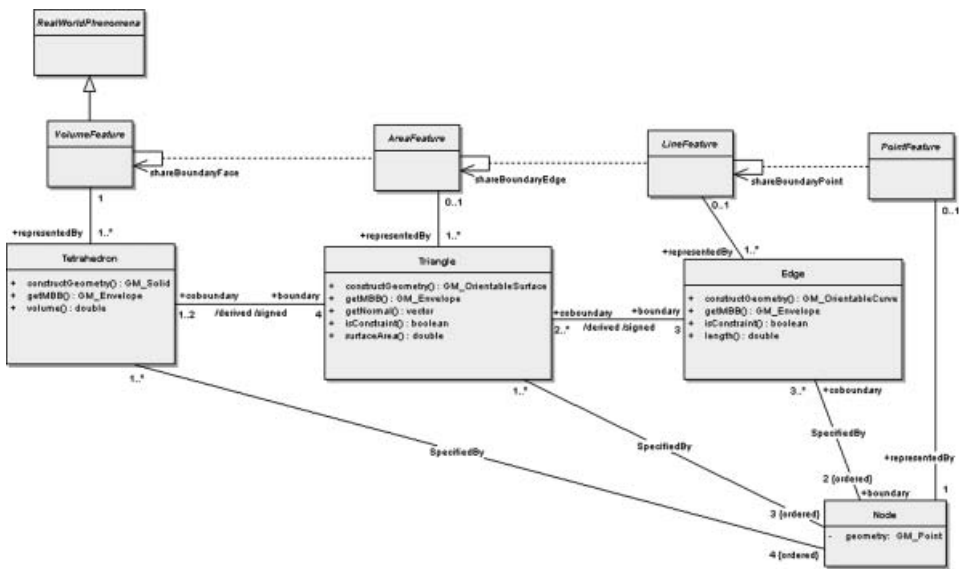


Figure 11. Conceptual model of the simplicial complex-based approach in an UML class diagram (compare to figure 4). Neither vertex encoding (Section 4.2) nor spatial DBMS implementation (Section 4.3) are taken into account in this conceptual model.

might not be stored at all, but only derived when needed. Section 4.3 will give more details on the actual implementation.

## 4.2 Vertex encoding

In the simplicial complex-based approach, simplexes will be defined by their vertices, resulting in a lot of references to the vertices. As the geometry is the only attribute of a vertex, adding a unique identifier to each point and building an index on top of this table will cause a substantial increase in data storage, which we try to avoid in our approach. To deal with this an alternative approach is used. It is based on the idea that adding a unique identifier is a bit redundant, as the geometry in itself will be a unique identifier as well. To achieve this the coordinate triple is concatenated into one long identifier code. Sorting this list will result in a very basic spatial index. In a way this approach can be seen as building and storing an index, while the original table is deleted. The possibilities of applying techniques like bitwise interleaving, which results in 3D Morton or Peano-Hilbert coding are recognised, but for reasons of insightfulness the concatenated version will be used in this paper. A small example of vertex encoding in the simplicial complex-based approach can be found in figure 12. The building from figure 9 is tetrahedronised and its tetrahedrons are described as concatenation of their four encoded vertices. Each row in the encoding should be interpreted as  $x_1y_1z_1x_2y_2z_2x_3y_3z_3x_4y_4z_4$ . In this example two positions are used for each coordinate element. For instance the last row (100000000600100600100608) should be interpreted as the tetrahedron defined by the vertices (10,00,00), (00,06,00), (10,06,00) and (10,06,08), i.e. the tetrahedron at the bottom right.

The current vertex encoding implementation is intended as a proof of concept and might be altered in more elaborate implementation tests. Current work by Penninga and van Oosterom (2007) also focuses on an alternative approach, in which node identifiers instead of actual node coordinates will be concatenated in simplex codes. Reasons to compare the two approaches lie both in the string size in the case of many decimal places as in avoiding redundant coordinate storage (as each point will be used in multiple tetrahedrons), although this will result in a second table (a node table) and more references.

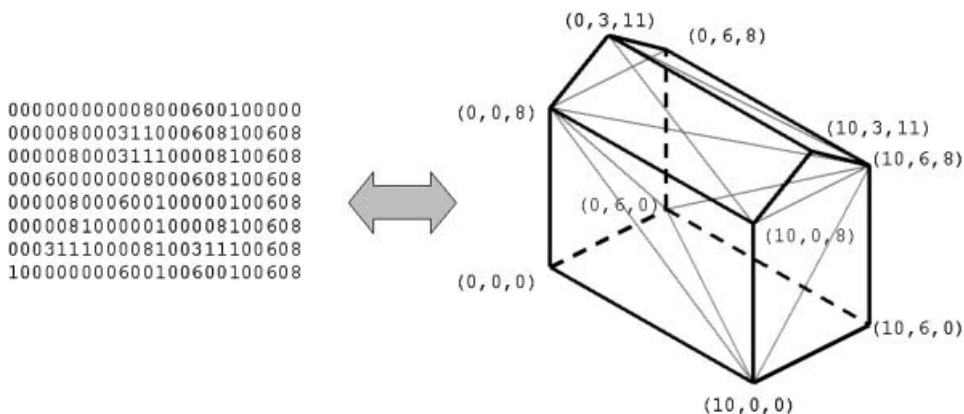


Figure 12. Describing tetrahedrons by their encoded vertices.

### 4.3 Incorporating the TEN structure in a spatial DBMS

Due to the use of the Poincaré simplicial homology in the TEN structure substantial parts of the structure can be derived. Explicit storage of tetrahedrons is required, all other information (both less dimensional simplexes and topological relationships) can be derived. Applying this in a DBMS environment will result in a table with tetrahedrons and functions to derive all other simplexes. These functions are used to create views with triangles, edges and nodes. As stated in Section 2.2, constraints are used to ensure the presence of the feature boundaries in the TEN structure. These constraints on certain triangles and edges indicate that these simplexes are required and therefore can not be deleted in retriangulation or update processes. Section 5 will show that these constrained triangles and edges can also be derived and therefore require no explicit storage either. As a result, the DBMS structure consists of a single (tetrahedron) table and several views (*see* figure 13).

In terms of explicit storage this means that the presented TEN approach is relatively compact. Consider the case of the building in figure 9. In a polyhedron approach, it would be described by its seven faces. Implicitly these seven faces also define the enclosed volume and the edges and nodes. In a classic TEN approach, the same building would require a lot more components, as both tetrahedrons, triangles, edges and nodes would be stored. However, in the TEN approach presented in this paper, only eight tetrahedrons are stored explicitly. Although tests with large data sets still need to be performed, one could expect that storage requirements are more or less in the same order of magnitude as the polyhedron approach (*see* table 1). Notwithstanding the fact that compactness is one of our goals, our approach still contains some redundancy as the coordinates of a vertex will be encoded in several tetrahedron codes. However, the delicate balance between compactness and manageability needs fine-tuning.

Compactness is not the only advantage of deriving huge parts of the structure. Another favourable characteristic is that updates are relatively easy, as only the tetrahedron table needs to be updated. Any changes in less dimensional simplexes or topological relationships are derived automatically. Especially these automatic updates in the topological relationships are pleasant, as it implies that one can benefit from the presence of topology without the need to maintain topology.

### 4.4 Modelling 3D objects: algorithms for incremental updates

Although this paper focuses emphatically on a new data structure, some remarks on the required algorithms will be made here. In 2D triangulation algorithms are

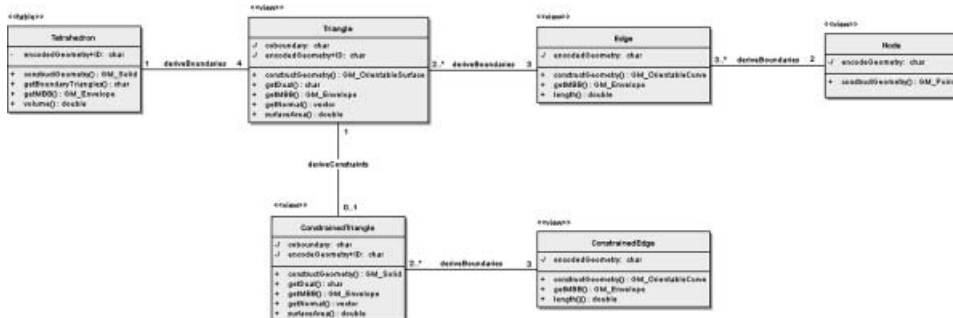


Figure 13. Schematic overview of DBMS storage structure.

Table 1. Intuitive comparison of storage requirements of the polyhedron, classic TEN and new TEN approach for the building in figure 9. The brackets indicate implicit presence (as opposite of explicit storage). It appears that the polyhedron and new TEN approach do not differ much.

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

well-known and features can be represented by constrained (Delaunay) triangulations. In 3D however, tetrahedronisation is more complex—first of all because not every shape can be tetrahedronised (Schönhardt 1928) without the insertion of additional points (i.e. Steiner points). How to derive a constrained tetrahedronisation is still an open problem (especially the question how to model multiple features in one constrained tetrahedronisation). Cavalcanti and Mello (1999) opt for a minimalist approach, as they derive a tetrahedronisation first and try to recover all constrained edges and faces afterwards. Shewchuk (2002a) describes an approach that creates a constrained tetrahedronisation incrementally. Liu and Baida (2000) discuss flipping as a tool for constrained tetrahedronisation. In Shewchuk (2003) the focus is on the importance of flips for updating. Flipping is further described by George and Borouchaki (2003) and Ledoux *et al.* (2005) use flipping to delete vertices from a tetrahedronisation. Penninga and van Oosterom (2006) describe an idea for the insertion of constrained edges as a connected set of two nodes, in which they distinguish nine unique cases. Another important aspect of tetrahedronisation algorithms is the quality of the tetrahedronisation, as it will affect the robustness of the operations. Refinement (i.e. the operation to acquire well-shaped triangles and tetrahedrons) is described by (among others) Shewchuk (1997, 2002b) and Si (2006).

Based on this very short overview, one can conclude that algorithms for constrained tetrahedronisation are still being developed. Flipping is an important technique, as it offers a robust approach to tetrahedronisation. Robustness is an important criterion as the TEN will become very large and complicated. In the TEN multiple features will be represented with constraints, whereas techniques from the more general field of meshing often focus on tetrahedronisation of a single object. Due to the expected size of the TEN, algorithms that act locally are favourable, maybe even mandatory.

## 5. Implementation: proof of concept

In order to provide more insight in the proposed new approach, this section will outline the current DBMS implementation. It is developed and tested with a small toy dataset, consisting of 56 tetrahedrons, 120 triangles, 83 edges and 20 nodes. This dataset was also used in a previous implementation (a classical TEN approach). Based on this implementation a 2D viewer (Oracle MapViewer) was adapted for 3D data by the use of a function `rotateGeom`. Both the implementation and the viewer are described in Penninga *et al.* (2006). In figure 14 the small dataset can be seen in the MapViewer. The dataset basically represents a small piece of the earth surface with a house and a road on top of it.

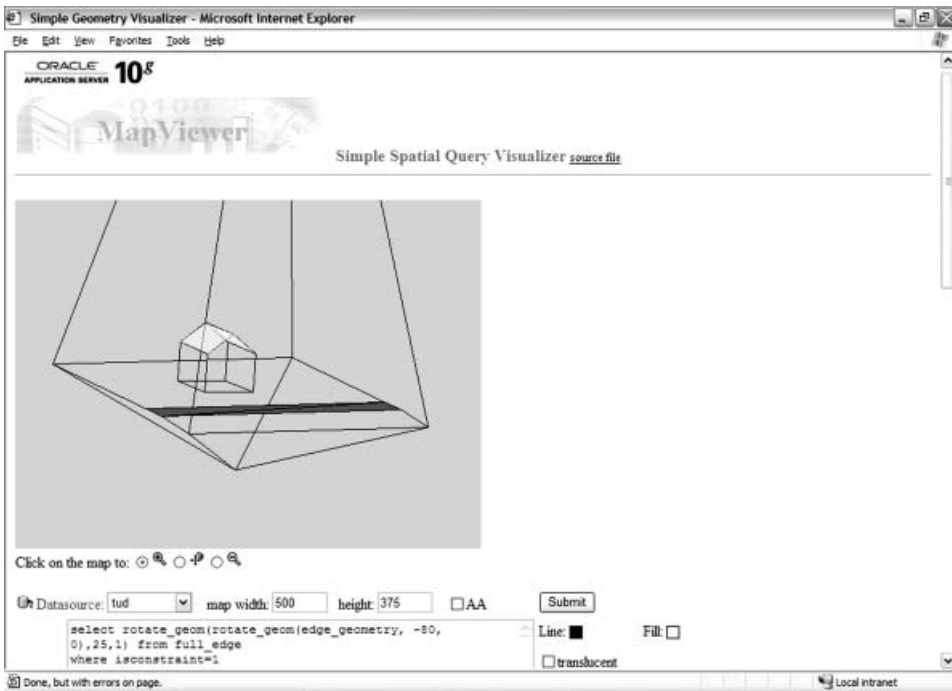


Figure 14. Adapting the 2D MapViewer for 3D data by a function rotateGeom.

This section will start in Section 5.1 with creating the data structure, i.e. to define the table and views to store tetrahedrons, triangles, edges and nodes. After that it will be shown in Section 5.2 that also the constraints can be derived, so no additional explicit storage is required. The topic of Section 5.3 is deriving topological relationships. The section continues with some remarks on validation in Section 5.4, followed by some examples on querying and analysis in Section 5.5 and ends with initial remarks on performance in Section 5.6.

### 5.1 Creating the data structure

As illustrated in figure 13, the tetrahedron table is the base table. It consists of a single column in which the encoded tetrahedrons are described in the form  $x_1y_1z_1x_2y_2z_2x_3y_3z_3x_4y_4z_4oid$ :

```
create table tetrahedron (tetcode NVARCHAR2(100));
```

Note that besides the concatenation of the four encoded vertices a unique object identifier is also added, which describes which volume object is (partly) represented by the tetrahedron. Since a tetrahedron is described by its four vertices, there are  $(4!)$  24 permutations of this single tetrahedron. By convention each tetrahedron  $\langle a, b, c, d \rangle$  is rewritten such that  $a < b < c < d$  holds, which is an arbitrary criterion. For each tetrahedron in the tetrahedron table, it is ensured that they are oriented correctly. The followed convention is that each tetrahedron has positive orientation, i.e. all normal vectors of the bounding triangles are oriented outwards. This consistent orientation is required to ensure that each boundary triangle appears two times: one with positive and the other with negative orientation. In a procedure, each tetrahedron's orientation is checked. All tetrahedrons with inward orientation are

replaced by tetrahedrons with outward orientation, which is achieved by performing a single permutation on the vertices:

```

create or replace procedure tettebleoutwards
(...)
ordertetrahedron (codelength, tetcode, currenttetcode);
checkorientation (codelength, currenttetcode, bool);
if (bool=0) then
permutation34 (codelength, currenttetcode, newtetcode);
update tetrahedron set tetcode=newtetcode where current of
tetcur;
(...)

```

The `checkorientation` procedure compares the direction of the normal vector of one of the boundary triangles with a vector from this triangle to the fourth (opposite) point of the tetrahedron. In the case of an inward orientation a single permutation is carried out by the procedure `permutation34`, which permutes the third and fourth vertex: `permutation34(< $v_0, v_1, v_2, v_3$ >)` results in `< $v_0, v_1, v_3, v_2$ >`. The ordering now meets the criterion  $a < b < d < c$ . The total result of the ordering, orientation check and in some cases permutation is (consider the tetrahedron code as a number) that every tetrahedron is described with the smallest positive tetrahedron code. This is a unique identifier of which of the 24 permutations will be used.

Based on the encoded tetrahedrons, the boundary triangles can be derived by applying the boundary operator from Definition 3.2. The procedure to derive the four boundary triangles of a tetrahedron looks like:

```

create or replace procedure deriveboundarytriangles
(...)
a:=(SUBSTR (tetcode, 1, 3*codelength));
b:=(SUBSTR (tetcode, 1+3*codelength, 3*codelength));
c:=(SUBSTR (tetcode, 1+6*codelength, 3*codelength));
d:=(SUBSTR (tetcode, 1+9*codelength, 3*codelength));
oid:=(SUBSTR (tetcode, 1+12*codelength));
ordertriangle (codelength, '+'||b||c||d||oid, tricode1);
ordertriangle (codelength, '-'||a||c||d||oid, tricode2);
ordertriangle (codelength, '+'||a||b||d||oid, tricode3);
ordertriangle (codelength, '-'||a||b||c||oid, tricode4);
(...)

```

Note that the triangles inherit the object id from the tetrahedron, i.e. each triangle has a reference to the object which is represented by the tetrahedron of which the triangle is a boundary. The reason for this will be introduced later in this section. It can also be seen that each boundary triangle is ordered by the `ordertriangle` procedure. The objective of this procedure is to gain control over which permutation is used. A triangle has six (3!) permutations, but it is important that both in positive and negative orientation the same permutation is used, as they will not cancel out in pairs otherwise (as described in section 3.3 on simplicial complexes). The procedure `ordertriangle` rewrites a triangle `<a, b, c>` such that  $a < b < c$  holds, which is again an arbitrary criterion. For example: `ordertriangle (+014035012022035012014035018003)` results in `-014035012014035018022035012003`, as its input `(+014035012 022035012`

014035018 003) does not satisfy the criterion  $a < b < c$ , so the second and third term are permuted and the odd permutation causes a sign change. One might expect that this ordering of triangles is not necessary due to the tetrahedron ordering, as performed earlier. However, this is not the case, as the need for consequent orientation (all tetrahedrons have positive orientation) causes tetrahedron codes that are not strictly ordered any more due to the permutation.

Slightly altered versions of the `deriveboundarytriangles` procedure are used to create the triangle view. The modified procedures derive respectively the first, second, third and fourth boundary triangle of a tetrahedron. The resulting view contains all triangles and their coboundaries (*see* Definition 3.7). In this case the coboundary is the tetrahedron of which the triangle is part of the boundary. This coboundary will prove useful in deriving topological relationships later in this section. The view is created as:

```
create or replace view triangle as
select deriveboundarytriangle1 (tetcode) tricode, tetcode
fromtetcode from tetrahedron
UNION ALL
select deriveboundarytriangle2 (tetcode) tricode, tetcode
fromtetcode from tetrahedron
UNION ALL
select deriveboundarytriangle3 (tetcode) tricode, tetcode
fromtetcode from tetrahedron
UNION ALL
select deriveboundarytriangle4 (tetcode) tricode, tetcode
fromtetcode from tetrahedron
```

The resulting view will contain four times the number of tetrahedrons, and every triangle (except for triangles on the outer boundary of the tetrahedronisation) appears two times: one with positive and the other with negative sign (and not in a permuted form, due to the `ordertriangle` procedure).

In a similar way the views with edges and nodes can be constructed. In current implementation edges are undirected and do not inherit object ids, as no application for this is identified at the moment. However, strict application of the boundary operator would result in directed triangles. The views are created as:

```
create or replace view edge as
select distinct deriveabsboundaryedge1 (tricode) edcode
from triangle
UNION
select distinct deriveabsboundaryedge2 (tricode) edcode
from triangle
UNION
select distinct deriveabsboundaryedge3 (tricode) edcode
from triangle;

create or replace view node as
select distinct deriveboundarynode1 (edcode) nodecode
FROM edge
UNION
select distinct deriveboundarynode2 (edcode) nodecode
FROM edge
```

With the tetrahedron table and triangle, edge and node view the data structure is accessible at different levels. Another characteristic of this approach is that both geometry and topology are present at every level, so one can determine for each operation whether a geometrical (i.e. using the simplex coordinates) or a topological (i.e. using references to boundaries, see Section 5.3 for more details) approach is the most appropriate.

## 5.2 Deriving constraints

As mentioned at the end of Section 2.2, features in the model are represented by a set of tetrahedrons. To ensure that these tetrahedrons represent the correct geometry, the outer boundary is triangulated and these triangles are used as constraints. This implies that these triangles will remain present as long as the feature is part of the model (i.e. they are not deleted in a flipping process). To achieve this, the incremental tetrahedronisation algorithm needs to keep track of these constrained triangles. In contrast with what one might expect, it is not necessary to store these constraints explicitly, as they can be derived as well, thus reducing data storage:

```
create or replace view constrainedtriangle as
select t1.tricode tricode from triangle t1
where not exists (select t2.tricode from triangle t2
where t1.tricode=t2.tricode*-1);
```

This statement uses the fact that although every triangle (in a geometric sense) appears twice (with opposite orientation) in the triangle view, not every triangle code appears twice. Boundary triangles are unpaired, since in this case the triangle code will differ due to the different inherited object ids. In the case of internal triangles (i.e. within an object) the triangle and its dual will have (apart from the sign) the exact same triangle code (geometry+object id). Deriving constrained edges from constrained triangles is easy, as all boundary edges from constrained triangles are constrained edges.

## 5.3 Deriving topological relationships

In a TEN the number of possible topological relationships is limited (note that topological relationships between features are not taken into account). As the TEN can be considered as a decomposition of space, relationships like overlap, cover or inside do not occur. Only relationships based on the interaction between tetrahedron boundaries occur. Tetrahedrons (and their boundaries) are either disjoint or touch. Three different types of the topological relationship touch can be distinguished:

- (i) (only) two nodes of neighbouring tetrahedrons touch;
- (ii) (only) two edges of neighbouring tetrahedrons touch;
- (iii) two triangles of neighbouring tetrahedrons touch.

The third case is the definition of a true neighbour relationship between two tetrahedrons. The other two cases are less important. As the neighbour relation is very important in certain operations and algorithms, two related relationships are derived in views in the implementation. The first is the relationship between a triangle and its dual. This relationship is important in the process of finding neighbours from tetrahedrons. Obviously triangles at the outer boundary of the

tetrahedronisation will not have a dual. The view is created by a select statement that uses the identical geometric part of the triangle codes:

```
create or replace view dualtriangle as
select t1.tricode tricode, t2.tricode dualtricode
from triangle t1, triangle t2
where removeobjectid (t2.tricode)=-1 *removeobjectid
(t1.tricode);
```

By combining the triangle view and the dualtriangle view, neighbouring tetrahedrons can be found:

```
create or replace function getneighbourtet1
(...)
select fromtetcode into neighbourtet from triangle
where removeobjectid(tricode)=-1 *removeobjectid
(tricode);
(...)
```

and based on functions like this one the view with tetrahedrons and their neighbours can be created. Analogue to this approach topological relationships at feature level can be derived.

#### 5.4 Validating the data structure

The Euler-Poincaré formula  $N - E + F - V = 0$  is introduced in Section 3.4. As can be seen in figure 10, this formula holds for all simplicial complexes, including simplicial complexes that consist of simplexes of different dimensions. Due to this characteristic dangling edges and faces cannot be detected, but for instance holes (i.e. missing faces) can be detected.

Besides validation at data structure level, one can also think of validation on feature level. As mentioned in Section 3.4, a valid volume feature satisfies three conditions:

- (i) the TEN in which the volume feature is modelled, should be valid;
- (ii) the boundary triangulation (a set of constraints in the TEN) should form a watertight boundary;
- (iii) the interior of the volume feature should be face-connected.

Within the simplicial complex-based approach the validation strategy is to start with a valid tetrahedronisation and to check every update for correctness before committing it to the database. As a result one will migrate from one valid state into another valid state. This strategy will also include the application of for instance flipping algorithms for the deletion of vertices (Ledoux *et al.* 2005), as such algorithms are designed to maintain a valid TEN during each step of the process.

Other correctness checks can be implemented, like for instance a check on the triangle view to ensure that every triangle appears two times (with opposite sign, ignoring the inherited object ids). Also validation on feature level can be considered; for instance one can check whether all constrained triangles form a valid polyhedron and whether these constrained triangles are intersected by edges (which is not allowed). For more details on the validation of polyhedrons, see Arens *et al.* (2005).

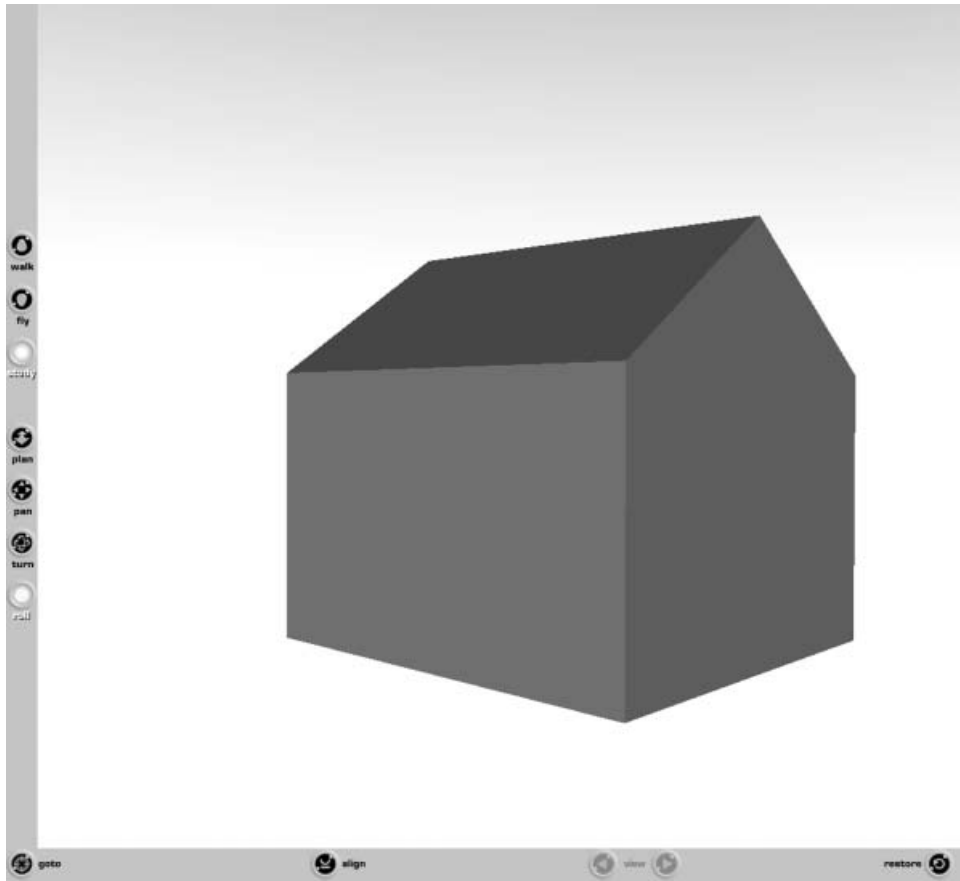


Figure 15. Output in VRML: result of select tricode from constrainedtriangle where getobjectid (3,tricode)=3. This is the same object as in figure 14.

### 5.5 Query and analysis

The presence of views helps to simplify a lot of queries as the functions on which the views are based can be omitted from the queries. The most frequently used elements and relationships are made available through these views. If one is interested in for instance a boundary representation of a feature, one could query the constrained triangle view with a specific object id. The resulting set of constrained triangles will form a valid polyhedron, *see* figure 15 for an example. One might consider simplifying this polyhedron further by merging triangles with identical (given a specific tolerance) normal vectors into polygons. However, a polyhedron may consist of triangular faces and these triangulations might be useful for visualisation purposes.

The number of analyses that can be performed on the TEN structure is virtually unlimited. One can think of basic operations like distance, line-of-sight or volume calculations, or more complex operations like tetrahedron-based buffer and overlay (Verbree *et al.* 2005). As mentioned in Section 3.4, operations can be performed both on simplex level and on feature level. Feature operations will collapse into operations on individual simplexes and from these separate results the feature result

will be composed. Also a wide variety of simulations can be performed on the tetrahedral mesh, like flooding or air flow simulations. Tetrahedral meshes can be used and optimised (into better-shaped tetrahedrons/triangles) for simulation purposes (Joe 1995, Cutler *et al.* 2004).

## **5.6 Performance**

The tetrahedron table is potentially very large, so indexing becomes an important aspect of the data structure. Sorting the table on the tetrahedron code after bitwise interleaving will function as an index, as tetrahedrons in a particular area will be stored close to each other in the table as well. However, a secondary index might still be needed. As the tetrahedron code contains all geometry, constructing the minimal bounding boxes and building a R-tree will be a logical step. To ensure performance for queries on the views, function based indexes are created for all functions that are used to create views. Updates in the data structure will also require updates in the indexes.

## **6. Conclusions and discussion**

### **6.1 Conclusions**

The objective of our research is to develop a data structure that focuses on analytical capabilities and to maintain data consistency. This paper has introduced a new topological approach, based on a tetrahedral network. It is based on the observation that all physical objects are volumetric by nature. As a result the real world can be considered as a volume partition: a full decomposition of 3D space. Space is partitioned by a large structure consisting of connected 3-simplexes (tetrahedrons). Operators and definitions from the field of simplicial homology are used to define and handle this structure of tetrahedrons. Applying simplicial homology offers full control over orientation of simplexes and enables one to derive substantial parts of the TEN structure efficiently, instead of explicitly storing all primitives. As a result only the single column tetrahedron table has to be stored explicitly. Due to the encoded vertices and inheritance of object ids, all constrained edges and faces can be derived, thus avoiding redundant data storage. Since the topological relationships are also derived, updating the structure turns out to be limited to updating the tetrahedron table (and perhaps a feature attribute table). All implicit updates in simplexes of lower dimension or topological relationships propagate from this single update action.

Besides the advantages of the solid theoretical foundation offered by simplicial homology, the tetrahedral network was also selected as a structure due to its favourable characteristics from a computational point of view. All elements of the tetrahedral network consist of flat faces, all elements are convex and they are well defined. Where for instance a polyhedron approach might result in a virtually endless variation in shapes and geometries, the TEN limits this variation to a single shape.

The described data structure is developed as a DBMS data structure. Spatial DBMS characteristics as the usage of views, function based indexes, 3D R-trees and more complex operations are extensively used and contribute to the compactness and versatility of the data structure. Furthermore, a database is capable of coping with large data volumes, which is an essential characteristic in handling large scale 3D data.

## 6.2 Discussion

One of the often raised objections to a TEN approach is its complexity, which might confuse the user as the link between tetrahedrons and the real world features can be hard to recognise. However, one can think of a setup in which the user handles only features (as polyhedrons), while the algorithms translate these polyhedrons into constrained triangles and use these to construct or update the constrained tetrahedronisation. In other words, the user interface determines the perceived complexity, not the internal representation. Considering the proposed data structure as an internal data representation could even be taken to the next level, as one might argue that a polyhedron data type in a DBMS should be represented internally as a set of tetrahedrons to determine and maintain the polyhedron's validity.

Another discussion topic is to identify the innovative aspects of the proposed method. As discussed in section 1.3, neither the idea to use a TEN data structure for 3D data nor the idea to use simplexes (in terms of simplicial homology) in a DBMS implementation is new. However, the proposed approach reduces data storage and eliminates the need for explicit updates of both topology and simplexes of lower dimension. By doing so, the approach tackles common drawbacks as TEN extensiveness and laboriousness of maintaining topology. Furthermore, applying simplicial homology offers full control over orientation of simplexes, which is a huge advantage especially in 3D. In addition to this aspect, the mathematical theory of simplicial homology offers a solid theoretical foundation for both the data structure and data operations. Integrating these concepts with database functionality results in a new innovative approach to 3D data modelling.

## 6.3 Further research

Some further research topics can be identified.

- (i) Test the data structure with a more extensive data set. Within the research project a 3D data set will become available of the inner city of 's-Hertogenbosch, an ancient Dutch city with a small river called Binnendieze that runs under houses and streets for about one third of its length. This data set should give further insight in the possibilities of the new data structure. Currently the model is tested with a data set with 1796 buildings, resulting in 167.598 tetrahedrons. Preliminary implementation results will be presented in the work by Penninga and van Oosterom (2007).
- (ii) Integrate the required constrained tetrahedronisation algorithms, as discussed in section 4.4. With this algorithms updating the structure will become possible. Preserving all constraints and maintaining a valid data structure will be the challenges.
- (iii) Implement more analyses and simulations as proof of the feasibility of a TEN data structure for such operations. More specific, it should also prove the feasibility of the current structure, with little explicit storage and lots of derived elements.
- (iv) The current implementation is based on the idea of a single valued vector map (Molenaar 1989). However, one could think of a simplicial-complex based approach that models for instance both 3D topography and ownership. This would require the introduction of multi valued constraints in the structure. Both at theoretical and implementation level this is an open problem. This multi valued approach should be compared to an approach in

which two single independent valued vector maps (separate topography and ownership map) have to be combined by a map overlay.

- (v) In the future simplexes should be considered as possible building blocks for 4D modelling. As simplicial homology offers control over orientation and derivation of boundaries, a 4D simplex might be suitable for temporal 3D modelling. Although a 4-simplex is hard to imagine, simplicial homology states that it is defined by five 4D vertices and bounded by five tetrahedrons.

### Acknowledgements

This publication is the result of the Bsic Space for Geo-information 3D topography (<http://www.gdmc.nl/3dtopo> 2006) research project. Furthermore, Friso Penninga and Peter van Oosterom participate in the research program 'Sustainable Urban Areas' (SUA) carried out by Delft University of Technology. Thanks to John Herring, Siva Ravada, Ravi Kothuri, Baris Kazar and Han Wammes for their constructive discussion on 3D TEN modelling in a DBMS context, and to Hugo Ledoux for proofreading this paper.

### References

- ARENS, C., STOTER, J. and VAN OOSTEROM, P., 2005, Modelling 3D spatial objects in a geo-DBMS using a 3D primitive. *Computers and Geosciences*, **31**, pp. 165–177.
- CARLSON, E., 1987, Three-dimensional conceptual modeling of subsurface structures. In *Proceedings of the Auto-Carto 8* (Falls Church, Virginia: ACSM/ASPRS), pp. 336–345.
- CAVALCANTI, P.R. and MELLO, U.T., 1999, Three-dimensional constrained delaunay triangulation: a minimalist approach. In *Proceedings of the 8th International Meshing Roundtable*, pp. 119–129 (South Lake Tahoe: Sandia National Laboratories).
- COLINS, K.D., 'Cayley-Menger determinant. From Mathworld – A Wolfram Web Resource. <http://mathworld.wolfram.com/Cayley-MengerDeterminant.html>', 2003.
- CUTLER, B., DORSEY, J. and McMILLAN, L., 2004, Simplification and improvement of tetrahedral models for simulation. In *Proceedings of the Eurographics Symposium on Geometry Processing*, R. Scopigno and D. Zorin (Eds), pp. 93–102 (New York: ACM Press).
- DE KLUIJVER, H. and STOTER, J., 2003, Noise mapping and GIS: optimising quality and efficiency of noise effect studies. *Computers, Environment and Urban Systems*, **27**, pp. 85–102.
- EGENHOFER, M. and FRANK, A., 1989, PANDA: an extensible Dbms supporting object-oriented software techniques. In *Proceedings of the Datenbanksysteme in Büro, Technik und Wissenschaft. Proceedings of GISI Fachtagung*, Zürich, 1989, Informatik Fachberichten (Springer-Verlag), pp. 74–79.
- EGENHOFER, M., FRANK, A. and JACKSON, J., 1989, A topological data model for spatial databases. In *Proceedings of First Symposium SSD'89*, Lecture Notes on Computer Science, Vol. 409 (Berlin: Springer), pp. 271–286.
- ELLUL, C. and HAKLAY, M., 2006, Requirements for topology in 3D GIS. *Transactions in GIS*, **10**, pp. 157–175.
- FRANK, A.U. and KUHN, W., 1986, Cell graphs: a provable correct method for the storage of geometry. In *Proceedings of the 2nd International Symposium on Spatial Data Handling*, Seattle, pp. 411–436.
- GEORGE, P.L. and BOROUCHAKI, H., 2003, Back to edge flips in 3 dimensions. In *Proceedings of the 12th International Meshing Roundtable*, pp. 393–402 (Sandia National Laboratories).
- GIBLIN, P., 1977, *Graphs, Surfaces and Homology, An Introduction to Algebraic Topology* (New York: Chapman and Hall).

- HATCHER, A., 2002, *Algebraic Topology* (Cambridge University Press) available at <http://www.math.cornell.edu/hatcher> <http://www.gdmc.nl/3dtopo> 2006.
- ISENBURG, M., LIU, Y., SHEWCHUK, J. and SNOEYINK, J., 2006a, Streaming computation of Delaunay triangulations. *ACM Transactions on Graphics*, 25 Special Issue on Proceedings of ACM SIGGRAPH 2006.
- ISENBURG, M., LIU, Y., SHEWCHUK, J., SNOEYINK, J. and THIRION, T., 2006b, Generating raster DEM from mass points via TIN streaming. In *Proceedings of the Geographic Information Science, 4th International Conference, GIScience 2006*, Münster, Germany, September 2006 M. Raubal, H.J. Miller, A.U. Frank, and M.F. Goodchild (Eds), 4197 of Lecture Notes on Computer Science, pp. 186–198 (Berlin: Springer).
- ISO/TC211, 2005, Geographic Information - Reference Model. Technical report ISO 19101, International Organization for Standardization.
- JOE, B., 1995, Construction of three-dimensional improved-quality triangulations using local transformations. *SIAM Journal on Scientific Computing*, **16**, pp. 1292–1307.
- KRAAK, M. and ORMELING, F., 1996, *Cartography, Visualization of Spatial Data* (Harrow: Longman).
- LEDoux, H., GOLD, C.M. and BACIU, G., 2005, Flipping to robustly delete a vertex in a Delaunay tetrahedralization. In *Proceedings of the International Conference on Computational Science and its Applications-ICCSA 2005*, 3480 of Lecture Notes on Computer Science, pp. 737–747 (Berlin/Heidelberg: Springer).
- LIU, A. and BAIDA, M., 2000, How far flipping can go towards 3D conforming/constrained triangulation. In *Proceedings of the 9th International Meshing Roundtable*, pp. 307–313 (Sandia National Laboratories).
- MOLENAAR, M., 1989, Single valued vector maps: a concept in geographic information systems. *Geoinformations-systeme*, **2**, pp. 18–27.
- OUDE ELBERINK, S. and VOSSELMAN, G., 2006, Adding the third dimension to a topographic database using airborne laser scanner data. In *Proceedings of the Photogrammetric Computer Vision 2006*, pp. 92–97 (Bonn: IAPRS).
- PENNINGA, F., 2005, 3D topographic data modelling: why rigidity is preferable to pragmatism. In *Proceedings of the Spatial Information Theory, Cosit'05*, A.G. Cohn and D.M. Mark (Eds), 3693 of Lecture Notes on Computer Science, pp. 409–425 (Berlin: Springer).
- PENNINGA, F. and VAN OOSTEROM, P., 2006, Updating features in a TEN-based DBMS approach for 3D topographic data modelling. In *Proceedings of the Geographic Information Science, Fourth International Conference, GIScience 2006*, Münster, Germany, September 2006, Extended Abstracts M. Raubal, H.J. Miller, A.U. Frank, and M.F. Goodchild (Eds), 28 of IfGI prints, pp. 147–152.
- PENNINGA, F. and VAN OOSTEROM, P., 2007, First implementation results and open issues on the Poincaré-TEN data structure. In *Proceedings of the ACCEPTED FOR: Advances in 3D Geo Information Systems, 3D GeoInfo'07, 2nd International Workshop on 3D Geo-Information: Requirements, Acquisition, Modelling, Analysis, Visualisation*, December 2007, Delft, the Netherlands P. van Oosterom, S. Zlatanova, F. Penninga, and E.M. Fendel (Eds) (Berlin: Springer), pp. 177–198.
- PENNINGA, F., VAN OOSTEROM, P. and KAZAR, B.M., 2006, A TEN-based DBMS approach for 3D topographic data modelling. In *Proceedings of the Progress in Spatial Data Handling, 12th International Symposium on Spatial Data Handling*, A. Riedl, W. Kainz and G. Elmes (Eds), pp. 581–598 (Münster: Springer).
- PIGOT, S., 1995, A topological model for a 3-dimensional spatial information system. PhD thesis, University of Tasmania, Australia.
- PIGOT, S., 1992, A topological model for a 3D spatial information system. In *Proceedings of the 5th International Symposium on Spatial Data Handling*, pp. 344–360 (Charleston, ICU).

- PILOUK, M., 1996, Integrated modelling for 3D GIS. PhD thesis, ITC Enschede, the Netherlands.
- POINCARÉ, H., 1895, Analysis situs. *Journal de l'Ecole Polytechnique*, **1**, pp. 1–123.
- POINCARÉ, H., 1899, Complément à l'Analysis situs. *Rendiconti del Circolo Matematico di Palermo*, **13**, pp. 285–343.
- SCHÖNHARDT., 1928, Über die zerlegung von dreieckspolyedern in tetraeder. *Mathematische Annalen*, **98**, pp. 309–312.
- SHEWCHUK, J.R., 1997, Delaunay refinement mesh generation. PhD thesis, Carnegie Mellon University.
- SHEWCHUK, J.R., 2002a, Constrained delaunay tetrahedralizations and provable good boundary recovery. In *Proceedings of the 11th International Meshing Roundtable*, Ithaca, pp. 193–204.
- SHEWCHUK, J.R., 2002b, Delaunay refinement algorithms for triangular mesh generation. *Computational Geometry: Theory and Applications*, **22**, pp. 21–74.
- SHEWCHUK, J.R., 2003, Updating and constructing constrained delaunay and constrained regular triangulations by flips. In *Proceedings of the 19th Annual Symposium on Computational Geometry*, pp. 181–190 (New York: ACM Press).
- SHEWCHUK, J.R., 2004, General-dimensional constrained delaunay and constrained regular triangulations I: combinatorial properties. *Discrete and Computational Geometry*, available at: <http://www-2.cs.cmu.edu/jrs>.
- SI, H., 2006, On refinement of constrained delaunay tetrahedralizations. In *Proceedings of the 15th International Meshing Roundtable*, pp. 509–528 (Berlin: Springer-Verlag).
- STOTER, J., 2004, 3D cadastre. PhD thesis, Delft University of Technology, the Netherlands.
- VERBREE, E., VAN DER MOST A., QUAK, W. and VAN OOSTEROM, P., 2005, Towards a 3D feature overlay through a tetrahedral mesh data structure. *Cartography and Geographic Information Science*, **32**, pp. 303–314.
- ZLATANOVA, S., ABDUL RAHMAN, A. and SHI, W., 2004, Topological models and frameworks for 3D spatial objects. *Computers and Geosciences*, **30**, pp. 419–428.
- ZLATANOVA, S., 2000, 3D GIS for urban development. PhD thesis, Graz University of Technology, Austria.
- ZLATANOVA, S., ABDUL RAHMAN, A. and PILOUK, M., 2002, 3D GIS: current status and perspectives. In *Proceedings of Joint Conference on Geo-Spatial Theory, Processing and Applications*, Ottawa, Proceedings on CD-ROM.