3D GIS and spatial data

from a(n|Spatial) Information Management point of view

Shell "Subsurface Target IT Architecture" workshop
Tuesday, Nov 14, 2006 in the Carlton Beach Hotel in The Hague-Scheveningen, The Netherlands

Peter van Oosterom

March 7, 2007

Overview

- Introduction
- 3D applications
- 3D spatial data (collection)
- 3D DBMS-GIS-CAD software
- TEN/Poincaré research
- Conclusion

Introduction

- Mainstream GIS use still mainly 2D due to
  - Limited 3D awareness at the user site
  - Limited availability of 3D data sets
  - Limited 3D support in mainstream products
- However pressure on space increases
- Custom-made systems have shown the potential of 3D applications (within professional organizations)
- 3D data collection is getting less and less a problem
  - R&D to get 3D in the mainstream

TUDelft

Overview

- Introduction
- 3D applications
- 3D spatial data (collection)
- 3D DBMS-GIS-CAD software
- TEN/Poincaré research
- Conclusion

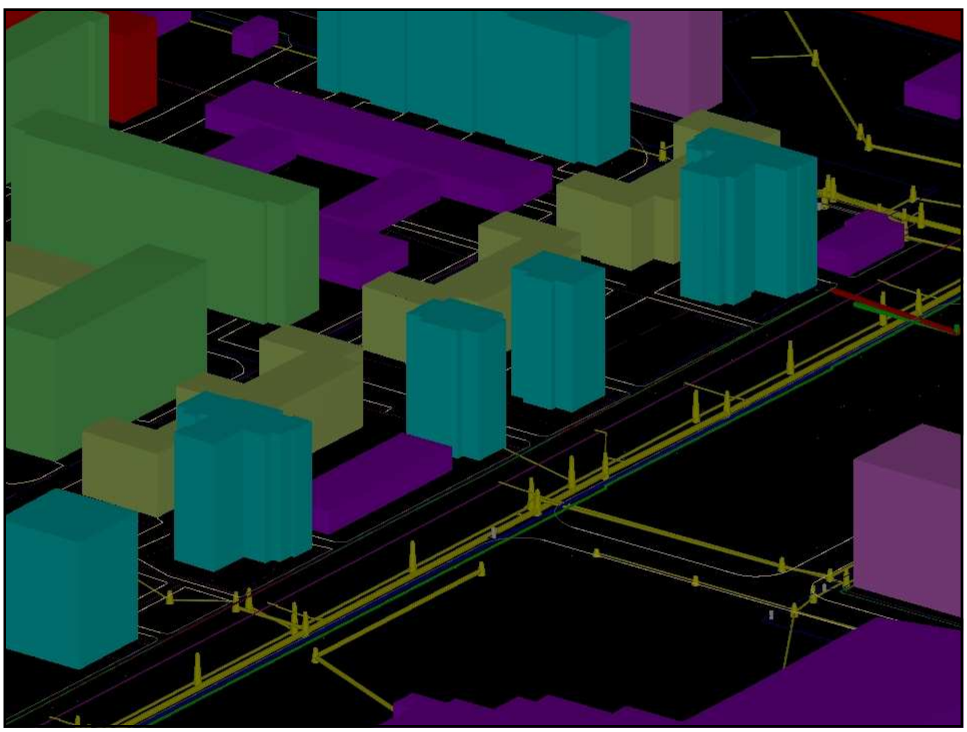
TUDelft

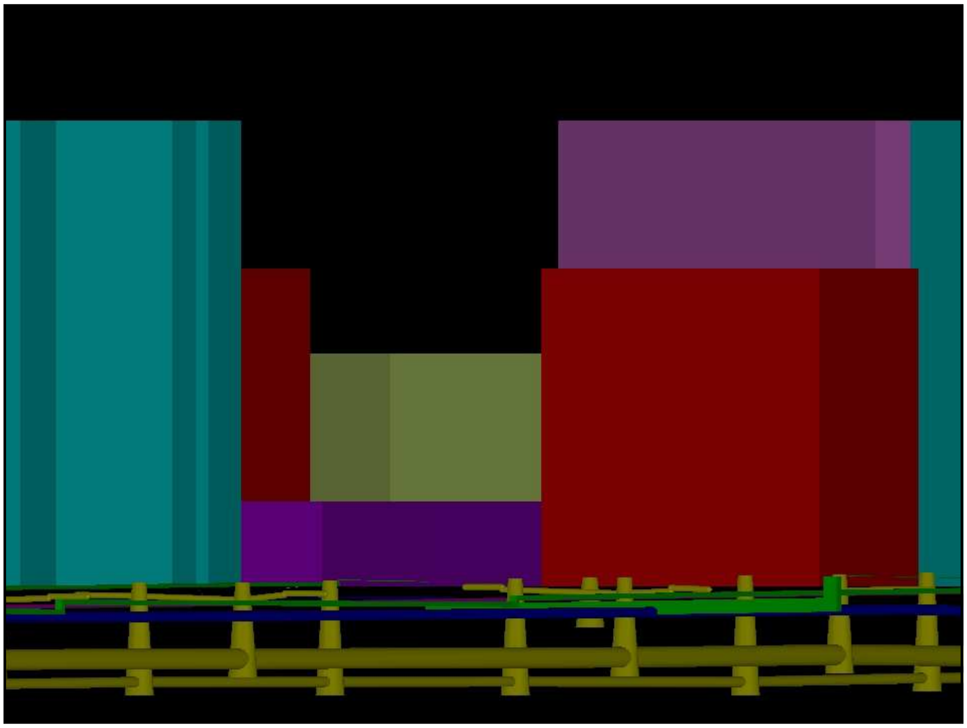
3D applications

- Taxation/valuation: volume of building
- Cadastral registration: 3D volume ‘parcel’
- Telecom: location of antennas
- Utility: subsurface networks of pipelines/cables
- Geology: (deep) subsurface model: oil, gas, minerals,…
- Aviation: 3D airspace management
- Planning: new constructions in VR environment
- Flooding: Water management: rivers, coastal zones,…

March 7, 2007 5

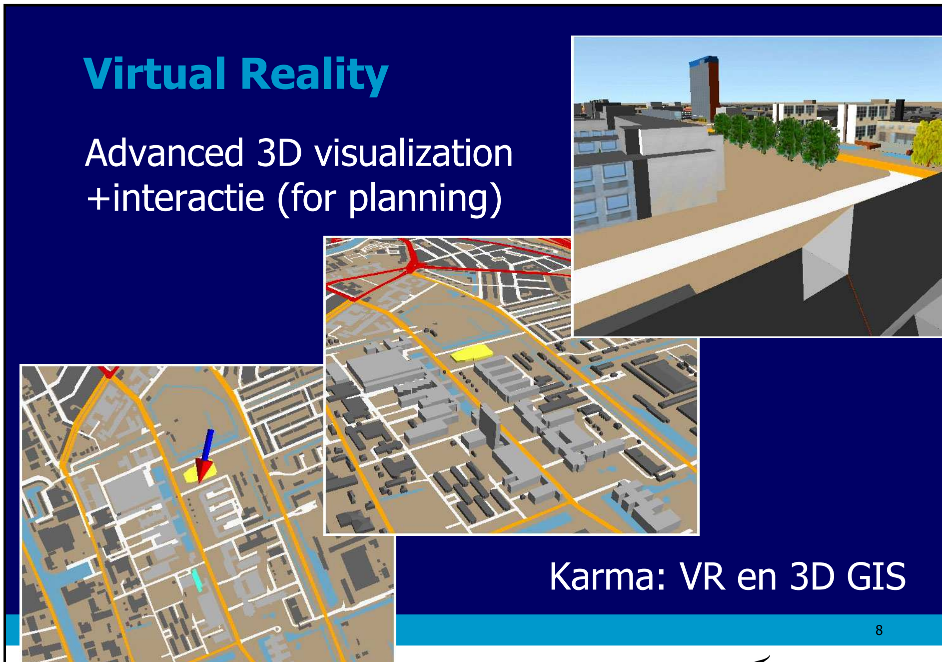
TU Delft





Virtual Reality

Advanced 3D visualization
+interactie (for planning)



Karma: VR en 3D GIS

8

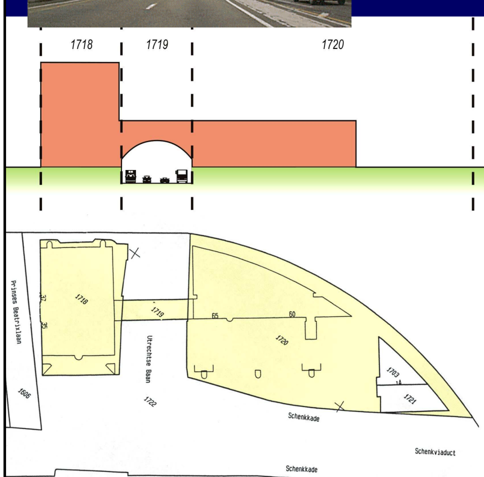
TU Delft

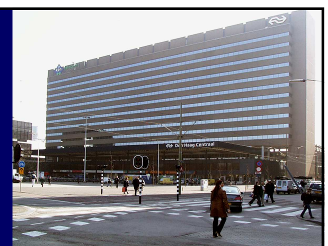
Background of the 3D Cadastre research

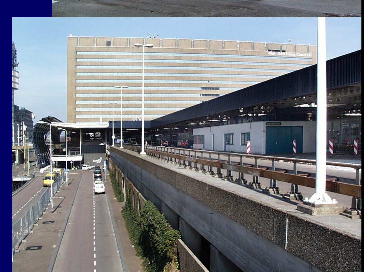
- Registration of real estate (property) at Cadastre
  - 2D parcel is basis of registration
  - Parcel is not limited in 3rd dimension
- Increasing use of space results in multi-functional land use (under and above the surface)
- It gets more and more difficult to register these situations in the current 2D system (admin tags)
- Extend current registration with 3D geo-objects within geo-DBMS (and link with legal admin tags)

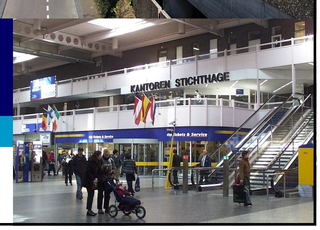
TU Delft

3D factual situations



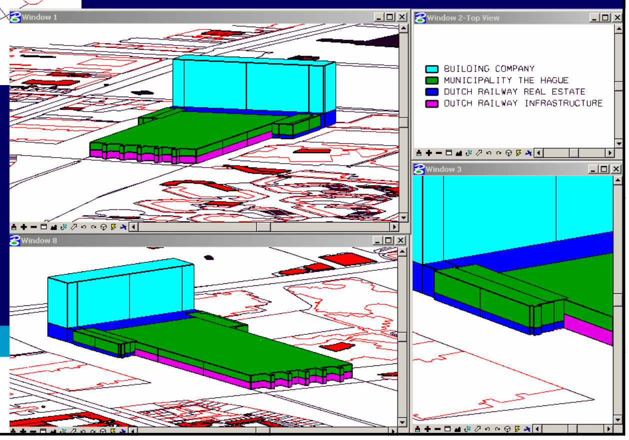




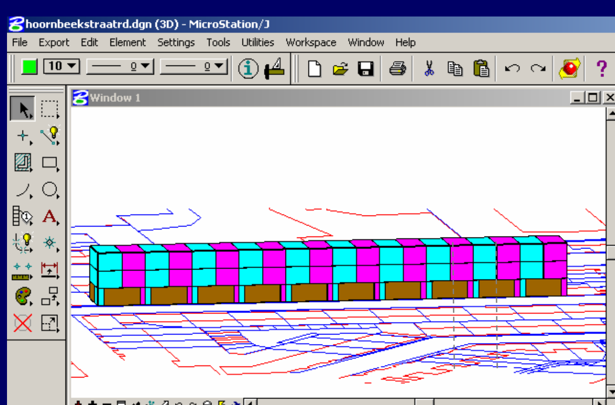


Current representation

Possible 3D registration

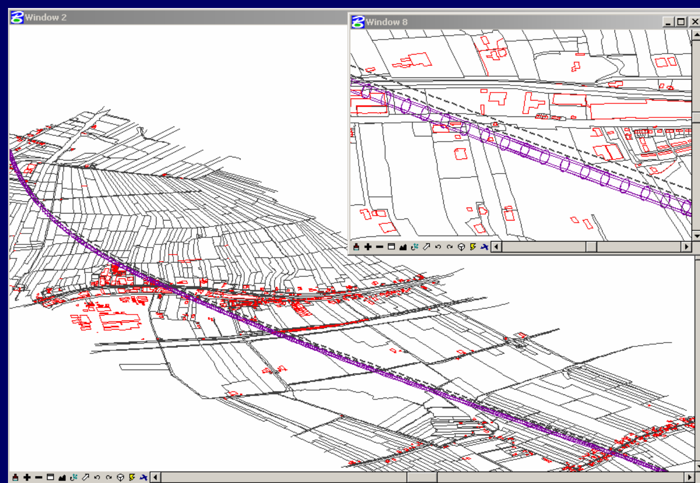


3D apartement complex on 2D parcels

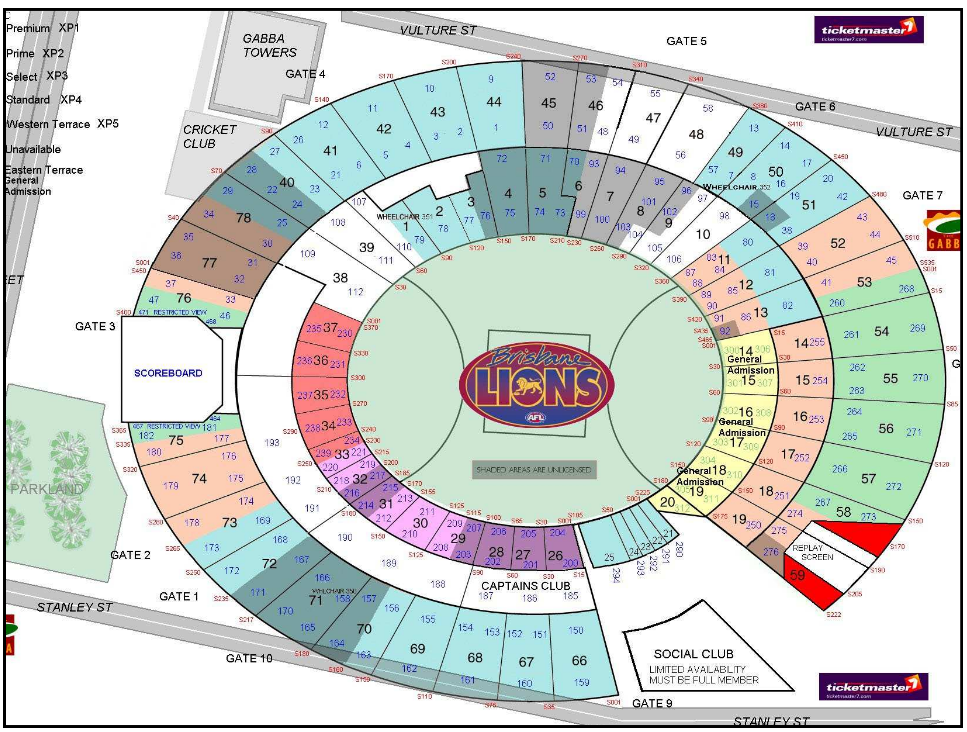


TU Delft

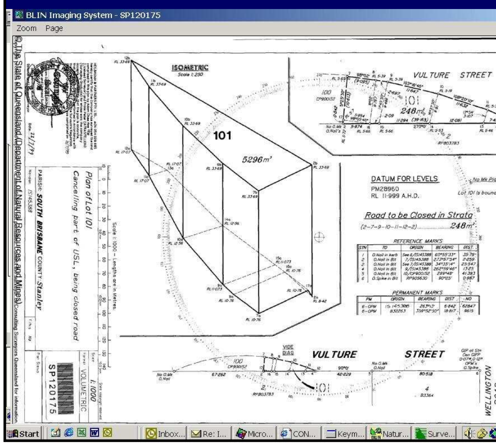
Registration of 3D infra object (HSL)

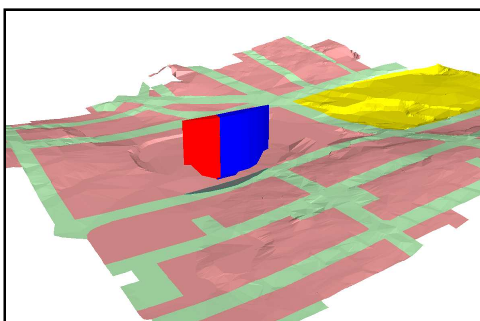


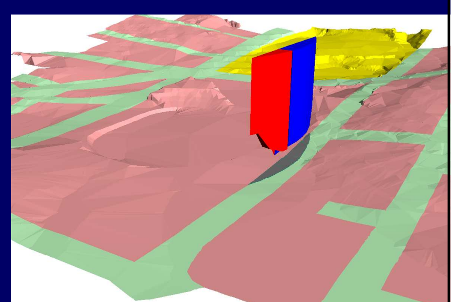
TUDelft

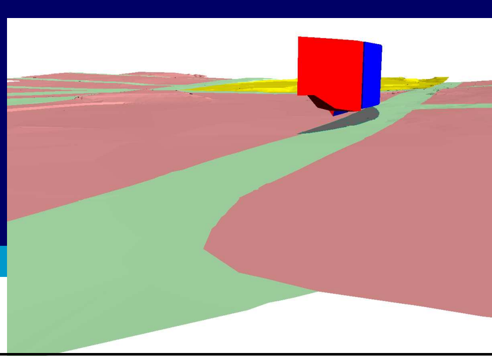


The Gabba 3D space as volumetric property









TU Delft

Overview

- Introduction
- 3D applications
- 3D spatial data (collection)
- 3D DBMS-GIS-CAD software
- TEN/Poincaré research
- Conclusion

3D data collection

- 3D data collection does not seam to be the problem: no longer separation between horizontal and vertical component in modern surveying (WGS84, ETRS)
- GPS (Galileo) based positioning delivers 3D coord's
- Laser altimetry (Lidar) very well suited for obtaining terrain elevation models (surfaces)
- Terrestrial rotating laser scanners are suited to create 3D models of objects (in addition to close range photogrammetry)
- Multi-beam sonar for river, lake or sea floor mapping

3D (2.5D) data
available data sets

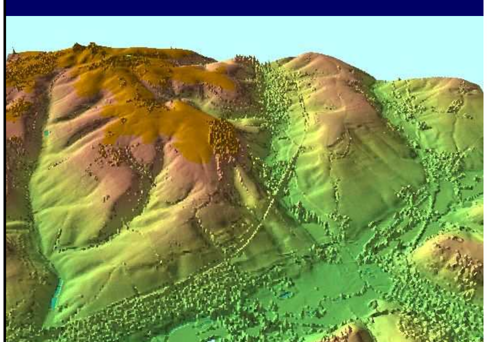
- Traditional elevation models (contour/breaklines, grid)
- Modern laser altimetry based nation wide data: e.g. AHN in the Netherlands (at least 1 point per 16 m^2)
- For routes/trace's even 16 point per 1 m^2 (Flymap, helicopter)
- Large scale topographic data of infrastrucure (terrestrial surveys of roads, waterways)
- 3D deep subsurface models (geology)
- Work in progress on: 3D Cadastre and 3D Top10NL

March 7, 2007 19

TUDelft

AHN

laserscan data
1 point/ 16 m^2



3D data modeling aspects
z-value: relative or absolute

- Z-value 3D object absolute in (inter)national height reference system, drawback:
  - The spatial relationship between traditional 2D data sets (on the surface) and the new 3D objects, requires explicit analysis (takes time)
- Z-value 3D object relative w.r.t. 2D surface, drawbacks:
  - absolute coordinates of 3D objects must be converted to relative coordinates
  - Data-consistency can be a problem (surface change)

Overview

- Introduction
- 3D applications
- 3D spatial data (collection)
- 3D DBMS-GIS-CAD software
- TEN/Poincaré research
- Conclusion

3D DBMS-GIS-CAD software

- Spatial DBMS foundation for both GIS (presentation, analysis) and CAD (creating models)
- 2D data types available in DBMSs similar to OGC Simple Feature Specification (SFS) for SQL
- Required 3D extensions of the geo-DBMS:
  - Support of 3D geometry (spatial data types including functions)
  - Support of 3D topological structures
  - 3D Spatial indexing and clustering

State of the Art: example Oracle Spatial (10g)

- Supported spatial data types: point, lines, polygons (arcs, boxes, sets), that is, the 0D-2D primitives
- Topology structure management is not supported (start of 2D topology structure management in 10g)
- Z-values can be used to store 3D features; e.g. 3D polygons, 3D lines, 3D points (and 'multi' versions)
- 3D Spatial indexing is available
- But: z-values are not recognized in spatial functions
- True 3D volume objects can not be represented

3D modeling example, creating 2D and 3D tables

```
create table geom2d (
 shape mdsys.sdo_geometry not null,
 TAG number(11) not null);

create table geom3d (
 shape mdsys.sdo_geometry not null,
 TAG number(11) not null);
```

Note same data type used in 2D and 3D case

TU Delft

3D modeling example, filling 2D and 3D tables

```
/* a 2D box */
insert into geom2d (TAG,shape) values (8,
  mdsys.SDO_GEOMETRY(2003,…,
  mdsys.SDO_ORDINATE_ARRAY(0,0, 100,100)));

/* a 3D box */
insert into geom3d (TAG,shape) values (9,
  mdsys.SDO_GEOMETRY(3003,…,
  mdsys.SDO_ORDINATE_ARRAY(0,0,50, 100,100,50)));
```

Note: different GTYPE and additional z-values, which are not used in the functions (distance, area,…)

TU Delft

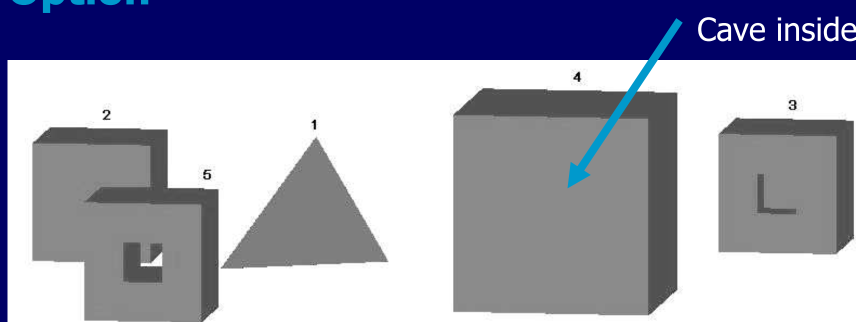
Extension with 3D data types: possible 3D primitives

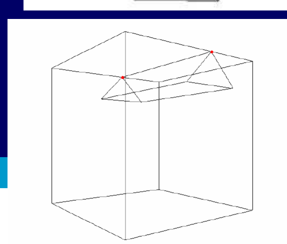
- Tetrahedron primitive: simplest 3D primitive:
  - easy algorithms (volume, area, distance, buffer),
  - but many primitives needed for real 3D object
- Polyhedron: ‘equivalent’ of 2D polygon, boundary defined by flat faces:
  - both boundary and interior may contain contain concavities (quite complex),
  - one polyhedron for one real 3D object
- Polyhedron with non flat faces (sphere/cylinder):
  - Close to 2D situation with circular arcs Complex to define, image, compute,…

TUDelft

‘Normal’ Polyhedron Most Realistic Option

Cave inside

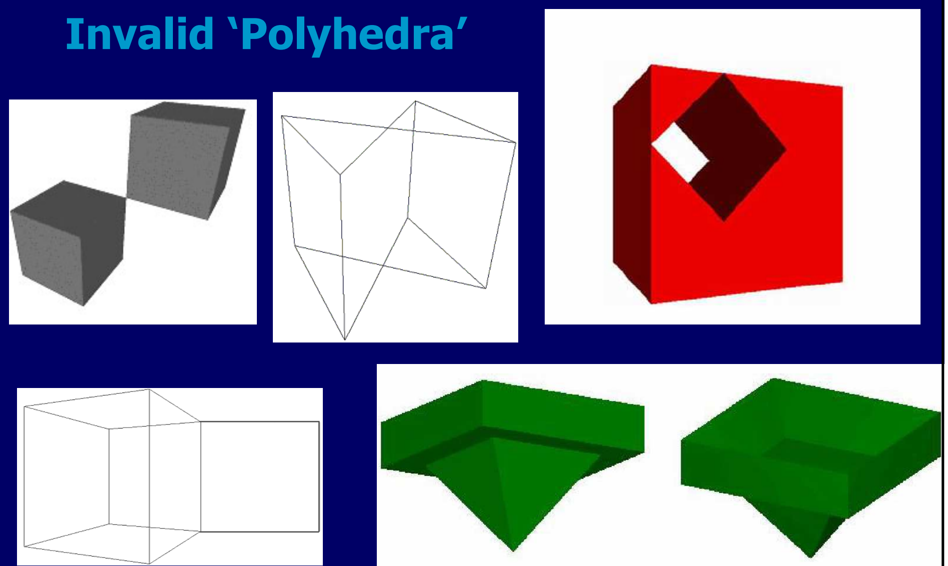




More than 3 vertices in a face.
How flat is this face?

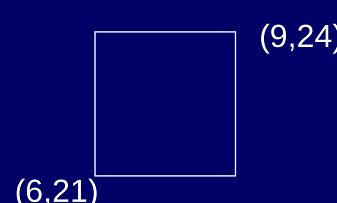
TUDelft

Invalid 'Polyhedra'



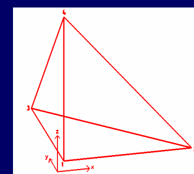
Normal encoding in Oracle sdo_geometry

```
SDO_GEOMETRY Column = (
SDO_GTYPE = 2003
SDO_SRID = NULL
SDO_POINT = NULL
SDO_ELEM_INFO = (1,1003,3)
SDO_ORDINATES = (6,21, 9,24))
```



GTYPE=2003 indicates: 2D (2xxx) and polygon (xxx3)
ELEM_INFO=(1,1003,3) indicates a rectangle

3D data type: encoding proposal polyhedron in Oracle (1)



Based on internal topology:
first nodes, then faces

```
insert into geom3d (shape, TAG) values (
 mdsys.SDO_GEOMETRY(3008,NULL,NULL,-- polyhedron, no ref point, no srid
  mdsys.SDO_ELEM_INFO_ARRAY(13,1006,1 -- first flat face at offset 13,
     -- because the first twelve positions are used for the coordinates
   16,1006,1, 19,1006,1, 22,1006,1),  -- others faces at 16, 19 and 22
  mdsys.SDO_ORDINATE_ARRAY(0,0,0,      -- coordinate triplet of point 1,
   1,0,0, 0,1,0, 0,0,1,                -- and of points 2, 3 and 4
   1,2,3, 1,2,4, 1,3,4, 2,3,4)),      -- the 4 faces by refs to the points
 1));                                  -- the TAG of example 1
```

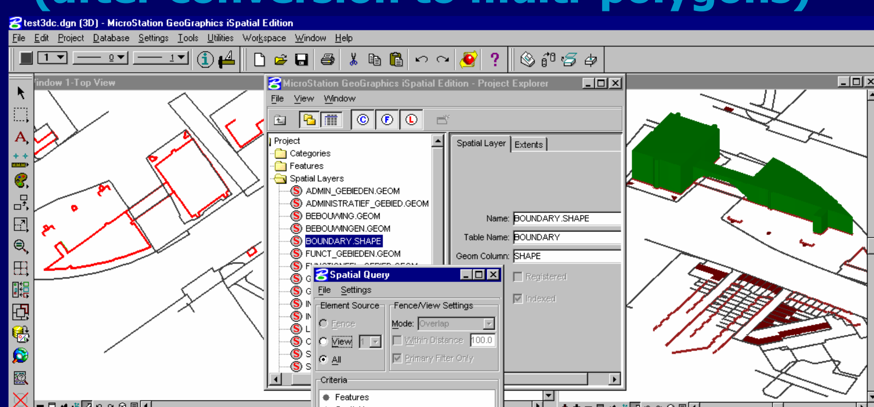
New GTYPE=3008

Faces of outer boundary
start at offset 13

Faces refer to the nodes

TU Delft

Oracle storage/ MicroStation editing (after conversion to multi-polygons)



- Prototype developed in Oracle by Calin Arens (MSc-thesis project at TU Delft)

Note, the mix of 2D and 3D data

TU Delft

Further extension for 3D space: curved lines and surfaces (NURBS)

- Not a volume primitive (3D), but line (1D) and surface (2D) primitives in 3D space Non Uniform Rational B-Splines (NURBS)
- Used for freeform curves and surfaces in CAD
- Part of industry standards IGES, STEP, PHIGS
- Specified via degree, control points, knots and weights
- Types added: GM_NURBSCurve and GM_NURBSSurface

- Prototype developed in Oracle by Pu Shi (MSc-thesis project at TU Delft)

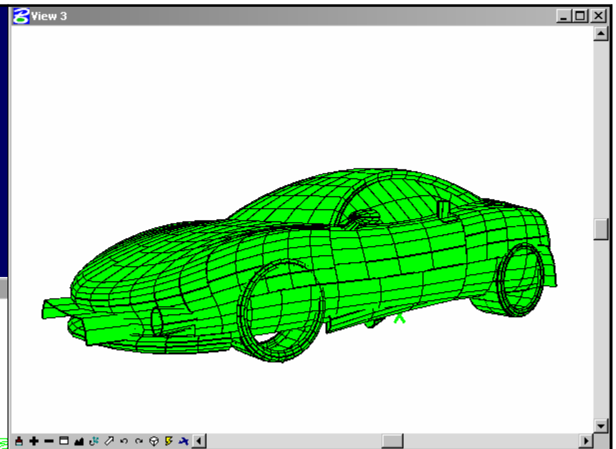
TUDelft

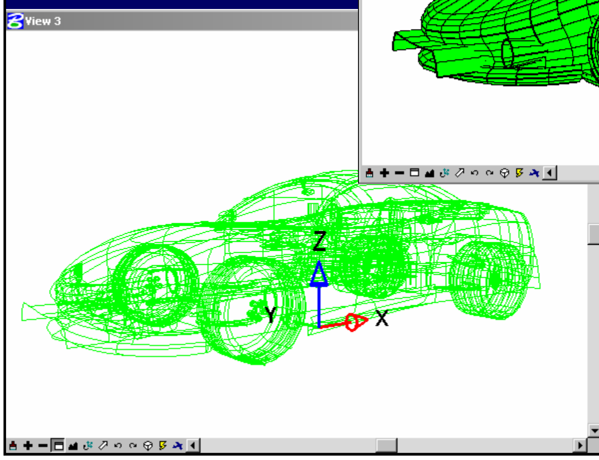
SQL Example with NURBS

```
// create a table with spatial column
create table test(id number,col GM_NURBSCurve);

// insert a NURBS curve
insert into test values(2,GM_NURBSCurve(2,GM_PointArray
 (135,225,346,127,256,336,945,20,30,504,70,698,434,40,4),
 GM_KnotVector(Vector(-0.5,0,0.5,1,2,3,4,5),NULL),NULL));

//select the convex hull geometry from
select a.col.convexHull() from test a;
```


TUDelft

From MicroStation to Oracle and back (without data loss)





35

TU Delft

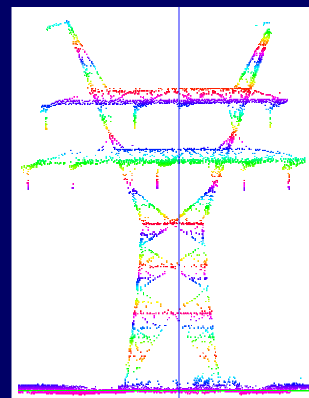
Further extension for 3D space: point clouds

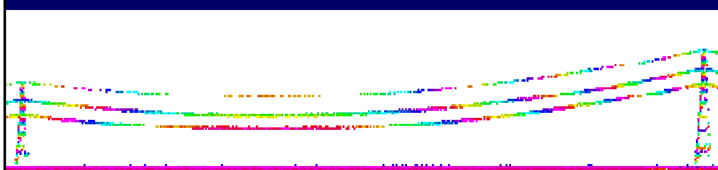
- Require also data management, but too 'expensive' to be stored using standards types (point and multipoint)
- TU Delft MSc thesis project of Martine Hoefsloot (at Fugro) investigates maximizing the efficiency of the standard SDO_GEOMETRY (point or multipoint)
- Efficient data storage (including meta data) for later altimetry, terrestrial laser scanning, multi-beam echo soundings, etc.
- New data type needed…

March 7, 2007 36

TU Delft

Fli-Map





37

TU Delft

Expected new 3D functionality in Beta of Oracle Spatial (11g)

- Z-values can be used in spatial 3D features; e.g. 3D polygons, 3D lines, 3D points (and 'multi' versions)
- True 3D volume objects can be represented via polyhedron (supported by set of functions)
- Specific data type for point clouds (again supported by set of functions)
- Support of TINs (NURBS not sure)

- Beta available on Red Hat Linux, not yet on Solaris, Windows (TU Delft beta tester, not yet started)

March 7, 2007 38

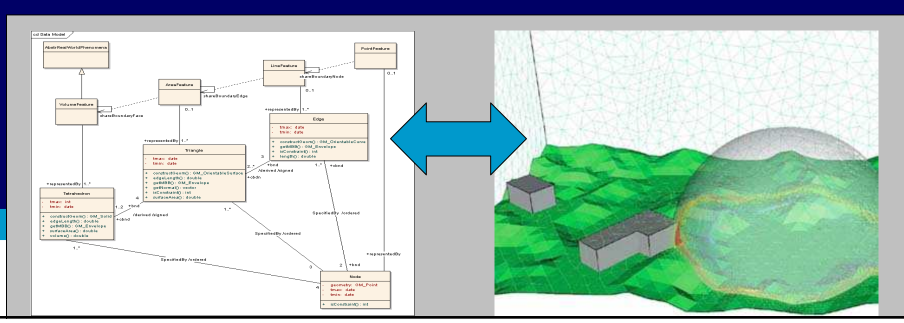
TU Delft

Overview

- Introduction
- 3D applications
- 3D spatial data (collection)
- 3D DBMS-GIS-CAD software
- TEN/Poincaré research
- Conclusion

GIS mathematics: Poincaré simplicial homology for 3D volume modeling

A. TEN Data Structure
B. Poincaré simplicial homology
C. Poincaré boundary properties
D. TEN based analysis
E. Conceptual Model
F. DBMS aspects
G. Implementation

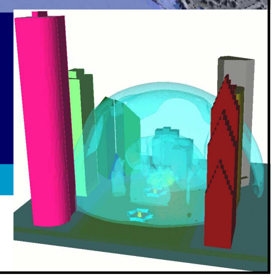


Need for 3D Topography

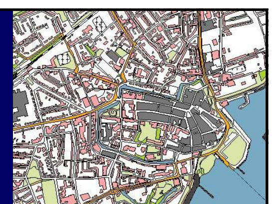
Real world is based on 3D objects
Objects + object representations get more complex due to multiple use of space

Applications in:
Sustainable development (planning)
Support disaster management

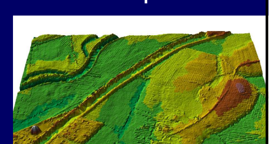
3D Topography: more than visualization!



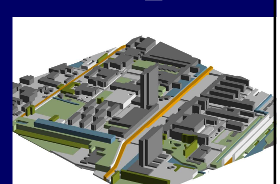
Research Goal



+



=



Develop a new topographic model to be realized within a robust data structure and filled with existing 2D, 2.5D and 3D data

Data structure:
design / develop / implement a data structure that supports 3D analyses and maintains data-integrity

TUDelft

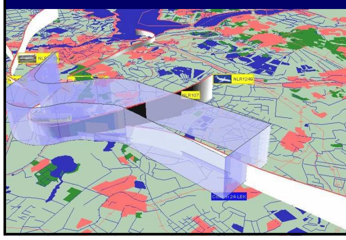
Full 3D Model

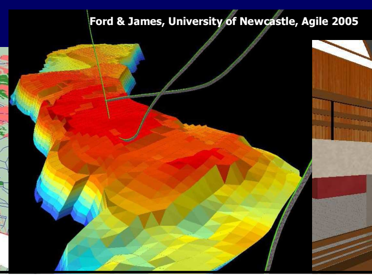
Observation: the real world is a volume partition
Result: ‘air’ and ‘earth’ (connecting the physical objects) are modeled

Advantages of volume partitioning:

- Air/earth often subject of analysis (noise, smell, pollution,..)
- Model can be refined to include:

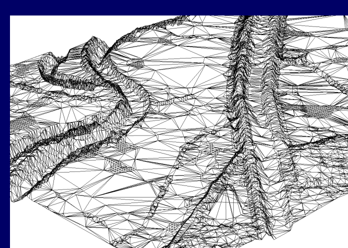
air traffic routes | geology layers (oil) | indoor topography





A. TEN Structure

- Support 3D (volume) analysis
- Irregular data with varying density
- Topological relationships
  enable consistency control
  (and analysis)



After considering alternatives the Tetrahedronized irregular Network (TEN) was selected (3D ‘brother’ of a TIN)

TU Delft

A. TEN Structure, Advantages

Based on simplexes (TEN):

- Well defined: a nD-simplex is bounded by n+1 (n-1)D-simplexes
  2D-simplex (triangle) bounded by 3 1D-simplexes (line segments)
- Flat: every plane is defined by 3 points (triangle in 3D)
- nD-simplex is convex (simple point-in-polygon tests)

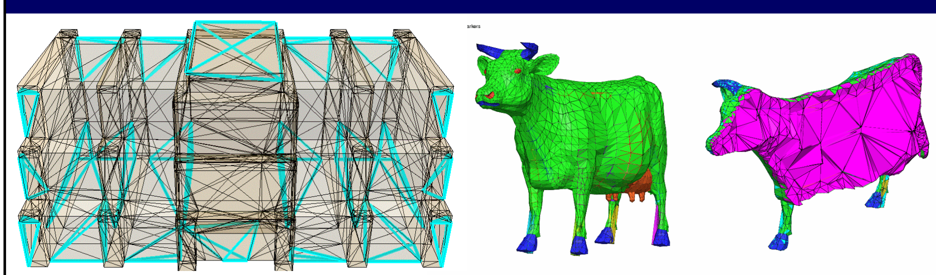
The TEN structure is very suitable for 3D analyses!

TU Delft

A. TEN Structure, Many Primitives Needed, a Disadvantage?

In 3D (complex) shapes → subdivide in (many) tetrahedrons:

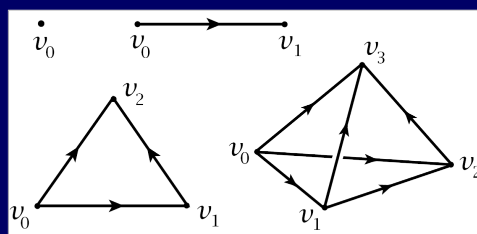
TEN is based on points, line segments, triangles and tetrahedrons: simplexes ('simplest shape in a given dimension')



B. Poincaré simplicial homology (1)

Solid mathematical foundation:

A n-simplex S_n is defined as smallest convex set in Euclidian space R^m of $n+1$ points $v_0, \ldots, v_n$
(which do not lie in a hyper plane of dimension less than n)



B. Poincaré simplicial homology (2)

The boundary of simplex S_n is defined as sum of *(n-1)* dimensional simplexes (note that 'hat' means skip the node):

$$\partial S_n = \sum_{i=0}^{n} (-1)^i < v_0, \ldots, \hat{v}_i, \ldots, v_n >$$

remark: sum has n+1 terms

$S_1 = <v_0, v_1>$	$\partial S_1 = <v_1> - <v_0>$
$S_2 = <v_0, v_1, v_2>$	$\partial S_2 = <v_1, v_2> - <v_0, v_2> + <v_0, v_1>$
$S_3 = <v_0, v_1, v_2, v_3>$	$\partial S_3 = <v_1, v_2, v_3> - <v_0, v_2, v_3> + <v_0, v_1, v_3> - <v_0, v_1, v_2>$

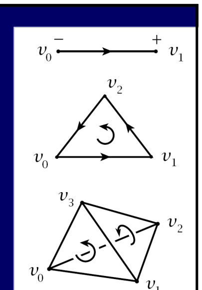
B. Poincaré simplicial homology (3)

$$S_n \text{ has} \binom{n+1}{p+1} \text{faces of dimension } p \text{ with } (0 \le p < n)$$

- 2D: this means that triangle (S_2) has 3 edges (S_1) and 3 nodes (S_0)
- 3D: this means that tetrahedron (S_3) has 4 triangles (S_2), 6 edges (S_1) and 4 nodes (S_0)

TU Delft

B. Poincaré simplicial homology (4)

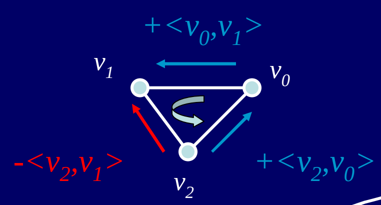


With $(n+1)$ points, there are $(n+1)!$ permutations of these points. In 3D for the 4 simplexes this means 1, 2, 6 and 24 options (S_0 obvious):

- For S_1 the two permutations are $<v_0, v_1>$ and $<v_1, v_0>$ (one positive and one negative $<v_0, v_1> = - <v_1, v_0>$)
- For S_2 there are 6: $<v_0,v_1,v_2>$, $<v_1,v_2,v_0>$, $<v_2,v_0,v_1>$, $<v_2,v_1,v_0>$, $<v_0,v_2,v_1>$, and $<v_1,v_0,v_2>$. First 3 opposite orientation from last 3, e.g. $<v_0,v_1,v_2> = - <v_2,v_1,v_0>$. counter clockwise (+) and the negative orientation is clockwise (-)
- For S_3 there are 24, of which 12 with all normal vectors outside (+) and 12 others with all normal vectors inside (-)!

TU Delft

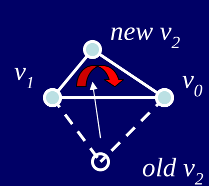
S_2 permutations and +/- orientation



$$\partial < v_2, v_0, v_1 > =$$
$$+ < v_0, v_1 > - < v_2, v_1 > + < v_2, v_0 >$$

$<v_0,v_1,v_2>$	+
$<v_0,v_2,v_1>$	-
$<v_2,v_0,v_1>$	+
$<v_2,v_1,v_0>$	-
$<v_1,v_2,v_0>$	+
$<v_1,v_0,v_2>$	-

S_2 permutations and +/- orientation



Note: +/- orientation depends on configuration (in this case on surface observed from above)

$<v_0,v_1,v_2>$	-
$<v_0,v_2,v_1>$	+
$<v_2,v_0,v_1>$	-
$<v_2,v_1,v_0>$	+
$<v_1,v_2,v_0>$	-
$<v_1,v_0,v_2>$	+

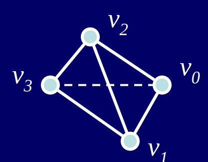
S_3 permutations and +/- orientation

$\langle v_0,v_1,v_2,v_3 \rangle$
$\langle v_0,v_1,v_3,v_2 \rangle$
$\langle v_0,v_3,v_1,v_2 \rangle$
$\langle v_0,v_3,v_2,v_1 \rangle$
$\langle v_0,v_2,v_3,v_1 \rangle$
$\langle v_0,v_2,v_1,v_3 \rangle$

$\langle v_2,v_0,v_1,v_3 \rangle$
$\langle v_2,v_0,v_3,v_1 \rangle$
$\langle v_2,v_3,v_0,v_1 \rangle$
$\langle v_2,v_3,v_1,v_0 \rangle$
$\langle v_2,v_1,v_3,v_0 \rangle$
$\langle v_2,v_1,v_0,v_3 \rangle$

$\langle v_1,v_2,v_0,v_3 \rangle$
$\langle v_1,v_2,v_3,v_0 \rangle$
$\langle v_1,v_3,v_2,v_0 \rangle$
$\langle v_1,v_3,v_0,v_2 \rangle$
$\langle v_1,v_0,v_3,v_2 \rangle$
$\langle v_1,v_0,v_2,v_3 \rangle$

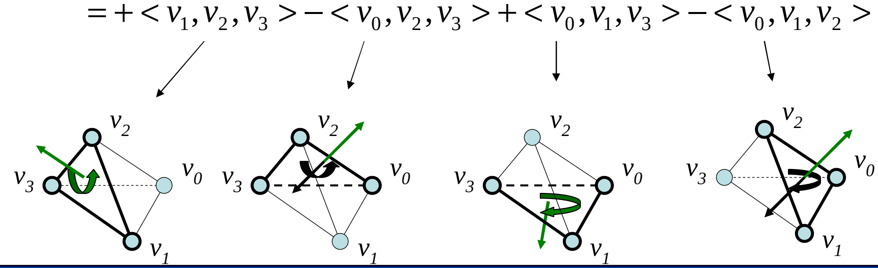
$\langle v_3,v_0,v_1,v_2 \rangle$
$\langle v_3,v_0,v_2,v_1 \rangle$
$\langle v_3,v_2,v_0,v_1 \rangle$
$\langle v_3,v_2,v_1,v_0 \rangle$
$\langle v_3,v_1,v_2,v_0 \rangle$
$\langle v_3,v_1,v_0,v_2 \rangle$



S_3 permutations and +/- orientation

S_{31}: For given configuration, first permutation of S_3 is positive (all signed normals pointing outside)

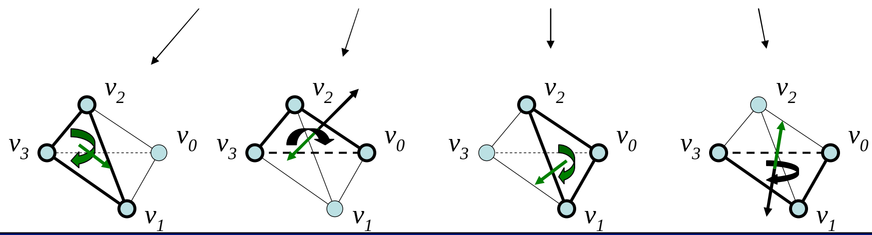
$$\partial S_{31} = \partial \langle v_0, v_1, v_2, v_3 \rangle$$
$$= +\langle v_1, v_2, v_3 \rangle - \langle v_0, v_2, v_3 \rangle + \langle v_0, v_1, v_3 \rangle - \langle v_0, v_1, v_2 \rangle$$



S_3 permutations and +/- orientation

S_{32}: For given configuration, second permutation of S_3 is negative (all signed normals pointing inside)

$$\partial S_{32} = \partial < v_0, v_1, v_3, v_2 >$$
$$= + < v_1, v_3, v_2 > - < v_0, v_3, v_2 > + < v_0, v_1, v_2 > - < v_0, v_1, v_3 >$$



TU Delft

C. Poincaré boundary properties (1)

- Within a single simplex, the sum of the signed boundaries of the boundaries of a simplex is 0 (taking into account the orientation)

$$\partial(\partial S_n) = \partial\left(\sum_{i=0}^{n} (-1)^i < v_0,\ldots,\hat{v}_i,\ldots,v_n >\right)$$
$$= \sum_{j<i} (-1)^i (-1)^j < v_0,\ldots,\hat{v}_j,\ldots,\hat{v}_i,\ldots,v_n >$$
$$+ \sum_{i<j} (-1)^i (-1)^{j-1} < v_0,\ldots,\hat{v}_i,\ldots,\hat{v}_j,\ldots,v_n > = 0$$

TU Delft

C. Poincaré boundary properties (2)

- In 3D this means that every edge is used once in positive direction and once in the negative direction (within simplex $S_3 = <v_0, v_1, v_2, v_3>$)

$$\partial S_3 = <v_1, v_2, v_3> - <v_0, v_2, v_3> + <v_0, v_1, v_3> - <v_0, v_1, v_2>$$

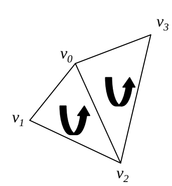
$\partial\partial S_3 =$

j\i	0	1	2	3
0		$-v_2v_3$	$+ v_1v_3$	$- v_1v_2$
1	$+ v_2v_3$		$- v_0v_3$	$+ v_0v_2$
2	$- v_1v_3$	$+ v_0v_3$		$- v_0v_1$
3	$+ v_1v_2$	$- v_1v_2$	$+ v_0v_1$	

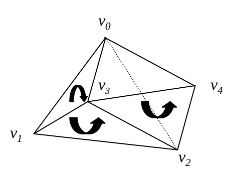
$\boxed{i < j : (-1)^i(-1)^{j+1}}$ $\boxed{j < i : (-1)^i(-1)^j}$

TUDelft

C. Poincaré boundary properties (3)
Adding Simplices to Complexes



$$S_{21} = <v_0, v_1, v_2> \text{ and } S_{22} = <v_0, v_2, v_3>$$
$$C_2 = <v_1, v_2> - <v_0, v_2> + <v_0, v_1>$$
$$+ <v_2, v_3> - <v_0, v_3> + <v_0, v_2>$$
$$= <v_1, v_2> + <v_0, v_1> + <v_2, v_3> + <v_3, v_0>$$

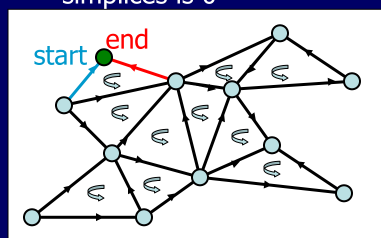


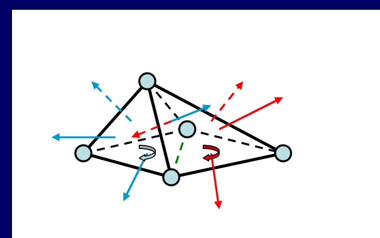
$$S_{31} = <v_0, v_1, v_2, v_3> \text{ and } S_{32} = <v_0, v_2, v_4, v_3>$$
$$C_3 = <v_1, v_2, v_3> - <v_0, v_2, v_3> + <v_0, v_1, v_3>$$
$$- <v_0, v_1, v_2> + <v_2, v_4, v_3> - <v_0, v_4, v_3>$$
$$+ <v_0, v_2, v_3> - <v_0, v_2, v_4>$$
$$= <v_1, v_2, v_3> + <v_0, v_1, v_3> - <v_0, v_1, v_2>$$
$$+ <v_2, v_4, v_3> - <v_0, v_4, v_3> - <v_0, v_2, v_4>$$

TUDelft

C. Poincaré boundary properties (4)

- Within a complex (of homogeneous highest dimension)
  - sum of signed boundaries of (highest dimensional) simplices is 0 *with exception of outer boundary of complex*
  - sum of signed boundaries of boundaries of (highest dimensional) simplices is 0





- 2D: boundary is edge
  boundary of boundary is node

3D: boundary is triangle
boundary of boundary is edge

TU Delft

C. Poincaré boundary properties (5)

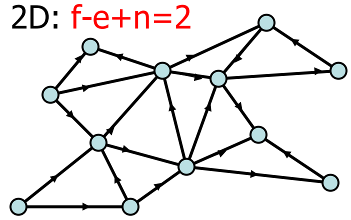
- Within a simplical complex of homegeous dimension (e.g. the TEN network in 3D), the co-boundary of S_n is defined as the set of higher dimesional simplices (set of S_{n+1}) of which S_n is boundary
- For example in a TEN network (3D):
  - the co-boundary of a triangle is formed by set of the *two* adjacent thetrahedrons
  - The co-boundary of a edge is formed by set of *k* incident triangles

TU Delft

D. Analysis Euler-Poincaré formula check if network can be ‘correct’ (1)

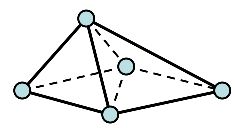
- The ‘balance’ between the number of simplices of different dimensions in a simplical complex is given by the Euler-Poincaré formula (including outside world):

2D: f-e+n=2



(11-21+12=2)

3D: t-f+e-n=0

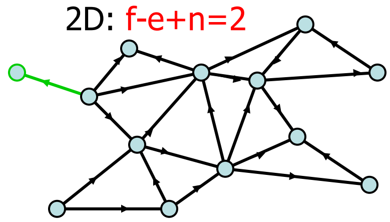


(3-7+9-5=0)

D. Analysis Euler-Poincaré formula check if network can be correct (2)

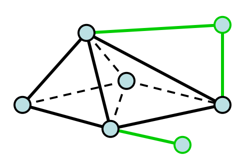
- Formula is valid for any simplical complex. Therefore can not detect dangling edges or faces in simplicial complex with homogeniuos highest dimesion such as a TEN.

2D: f-e+n=2



11-21+12=2 → 11-22+13=2

3D: t-f+e-n=0



3-7+9-5=0 → 3-8+12-7=0

D. Analysis, compute content (area in 2D, volume in 3D,…)

- 2D: compute area of triangle (base * half of height), also according to Heron's formula with semiperimeter s=1/2(a+b+c):

$$Area(S_2) = \sqrt{s(s-a)(s-b)(s-c)}$$

- Can be stated using Cayley-Menger determinant:

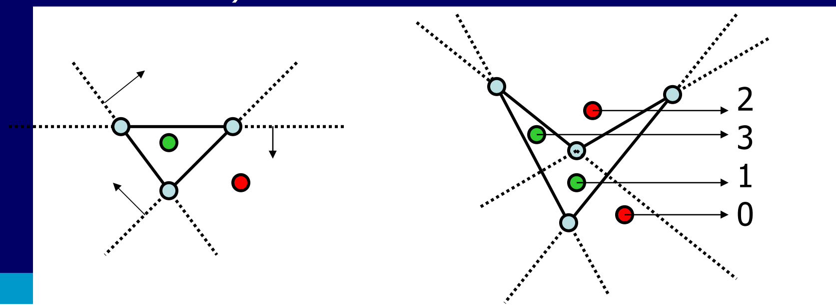
$$-16 * Area(S_2)^2 = \begin{vmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & c^2 & b^2 \\ 1 & c^2 & 0 & a^2 \\ 1 & b^2 & a^2 & 0 \end{vmatrix}$$

- Also defined for nD simplex based on (n+2)*(n+2) det

TU Delft

D. Analysis, point-in-polygon test (point- in-polyhedron test similar)

- nD-simplex is convex, much simpler point-in-polygon tests (for every boundary check is point is on correct side of boundary) compared to concave test (shoot ray from point to infinite and count number of boundary intersections)

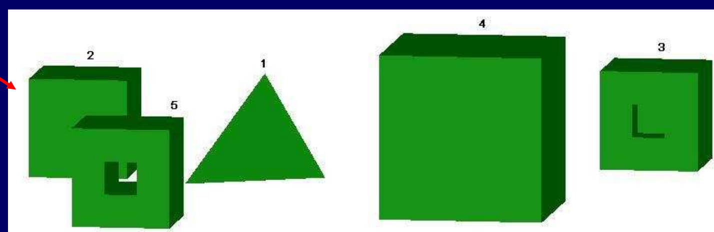


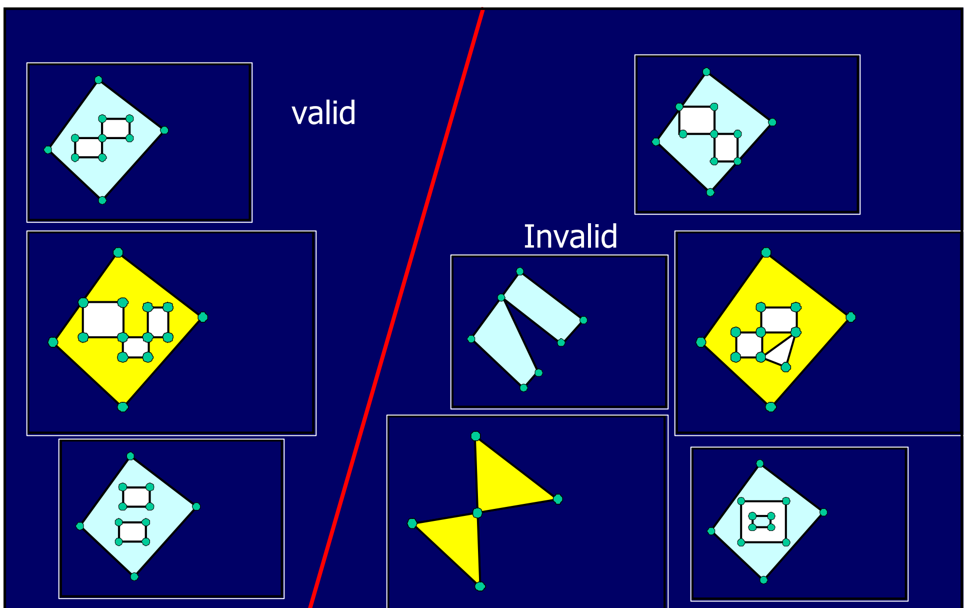
TU Delft

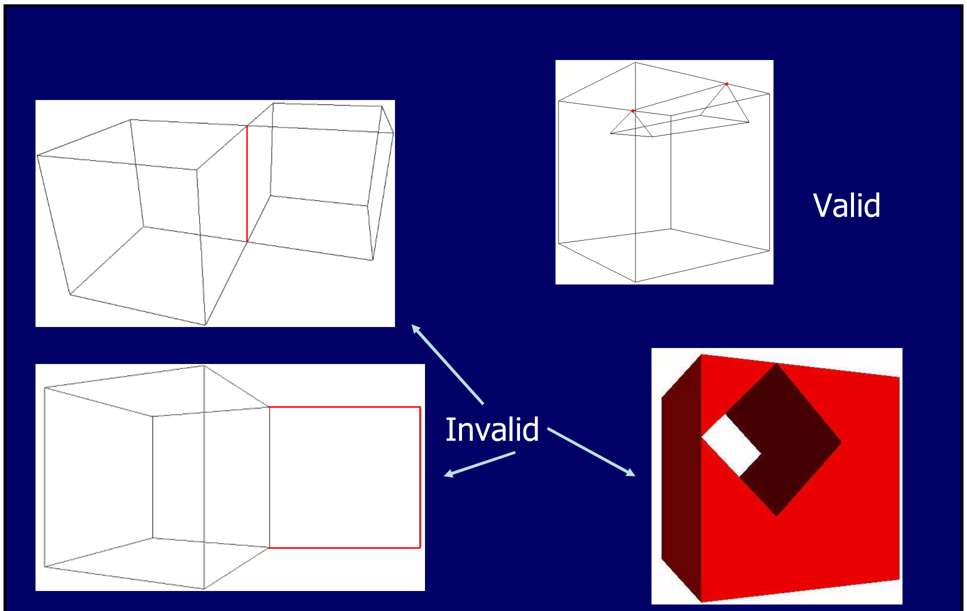
D. Analysis, check if polyhedral object is correct (connected)

- Definition of a valid polyhedral object (bounded by flat faces and possible having holes and handles):
  from every point of the polyhedron is should be able to reach every other point of the polyhedron via the interior.

Hole
inside







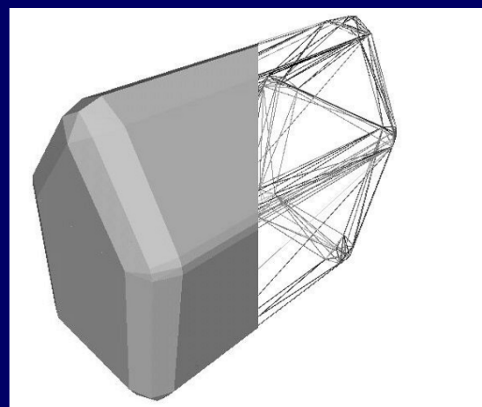
D. polyhedral object correctness test

- Traditional approach: check for 2-manifold (every edges used twice) and boundaries may not cross
- If outer and inner boundary touch: do topology analysis. Potentially invalid: if the connected intersections (edges) of inner and outer boundaries (faces) do form a ring (however, this is not a sufficient condition)

- Much easier: check if tetrahedrons (within a correct TEN) of one object are neighbors sharing a triangle (so edge or node is not sufficient)

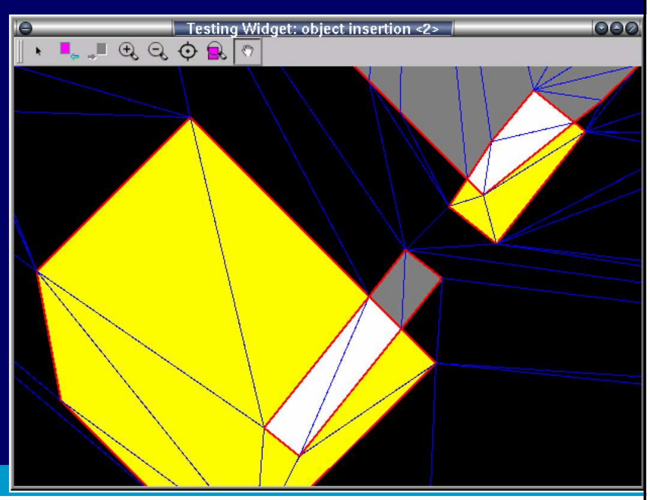
D. Analysis, TEN-based buffer approximation

MSc-thesis
Jeroen de Vries



D. Analysis, TEN-based overlay

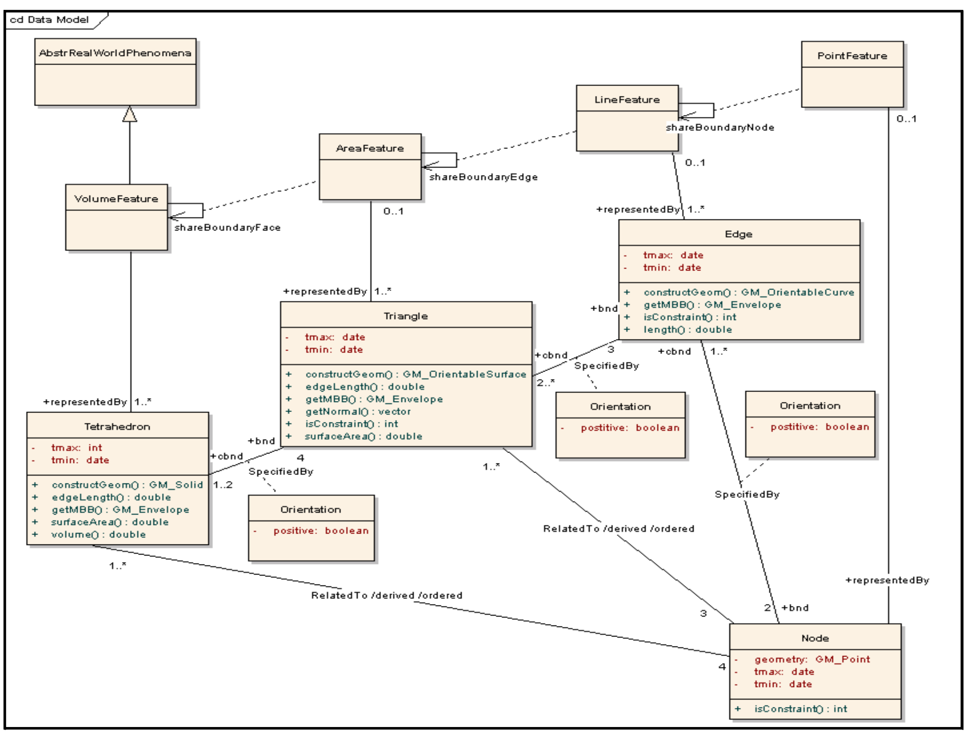
MSc-thesis
Arno van der Most
(prototype in 2D,
Concept in 3D)

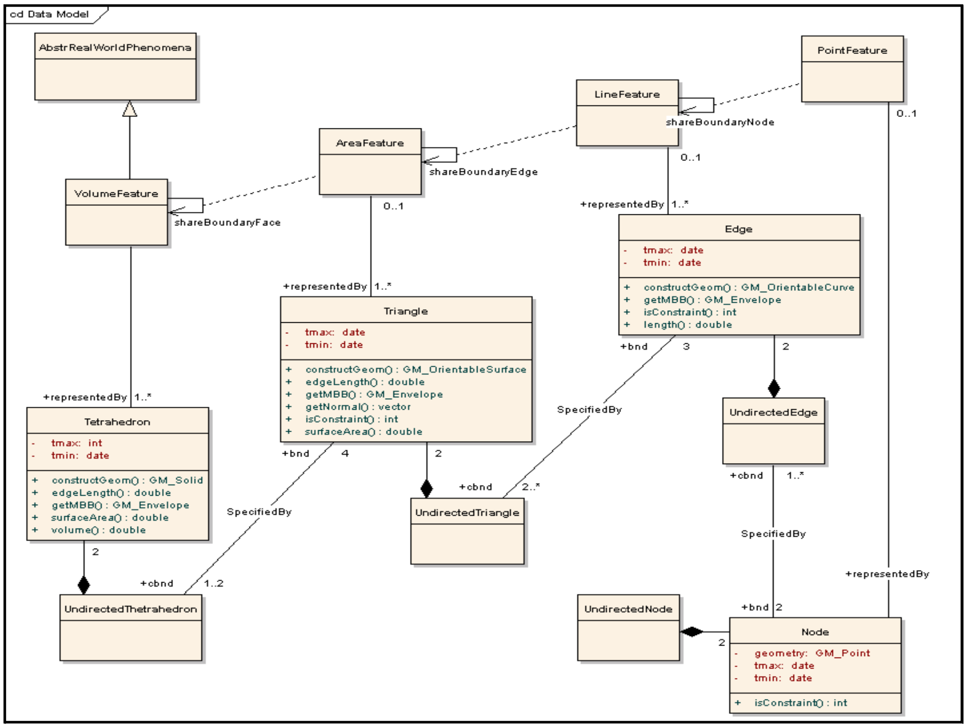


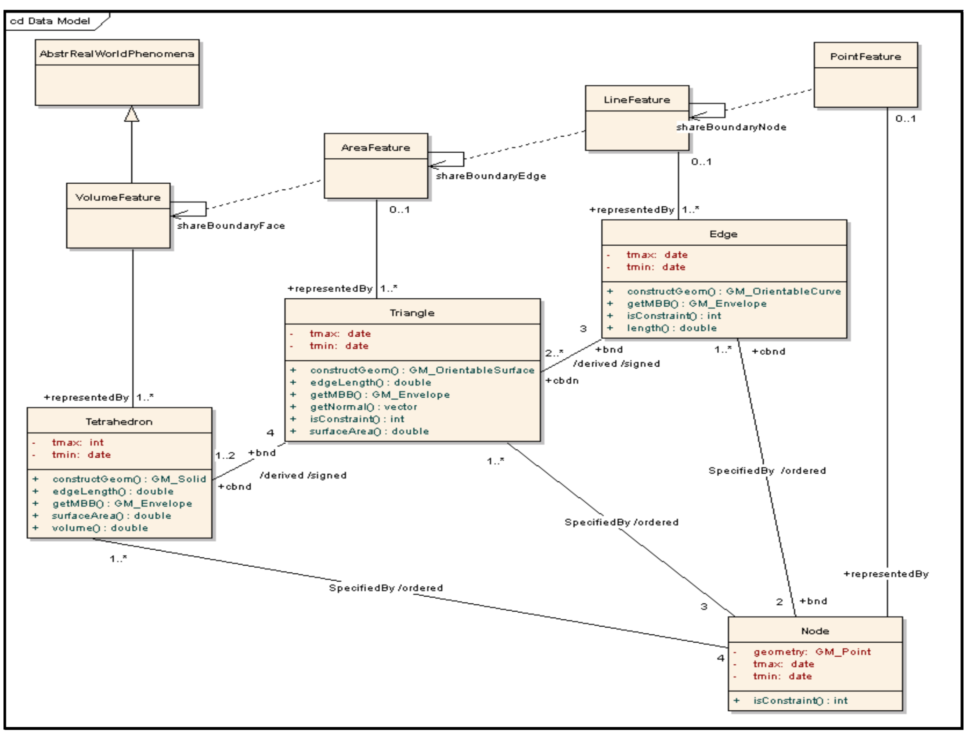
E. Conceptual Model

- Earlier work: Panda (Egenhofer et. al. 1989) and Oracle Spatial (Kothuri et. al. 2004) only 2D space. Panda has no attention for feature modeling
- Features attached to set of primitives (simplices)
- TEN model seams simple, but different perspectives results in 3 different conceptual models for same TEN:
  1. Explicit oriented relations between next higher level
  2. As above but with directed and undirected primitives
  3. Only with ordered relations to nodes (others derived)

TU Delft

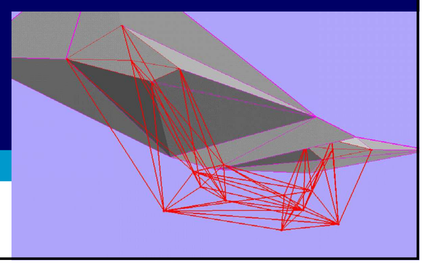



cd Data Model
AbstrRealWorldPhenomena
PointFeature
LineFeature
shareBoundaryNode
AreaFeature
shareBoundaryEdge
VolumeFeature
shareBoundaryFace
0..1
+representedBy 1..*
Edge
- tmax: date
- tmin: date
+ constructGeom() : GM_OrientableCurve
+ getMBB() : GM_Envelope
+ isConstraint() : int
+ length() : double
Triangle
- tmax: date
- tmin: date
+ constructGeom() : GM_OrientableSurface
+ edgeLength() : double
+ getMBB() : GM_Envelope
+ getNormal() : vector
+ isConstraint() : int
+ surfaceArea() : double
Tetrahedron
- tmax: int
- tmin: date
+ constructGeom() : GM_Solid
+ edgeLength() : double
+ getMBB() : GM_Envelope
+ surfaceArea() : double
+ volume() : double
+bnd
SpecifiedBy
+cbnd
UndirectedEdge
UndirectedTriangle
UndirectedThetrahedron
UndirectedNode
+representedBy
Node
- geometry: GM_Point
- tmax: date
- tmin: date
+ isConstraint() : int

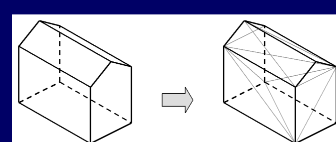

cd Data Model
AbstrRealWorldPhenomena
PointFeature
LineFeature
shareBoundaryNode
AreaFeature
shareBoundaryEdge
VolumeFeature
shareBoundaryFace
0..1
+representedBy 1..*
Edge
- tmax: date
- tmin: date
+ constructGeom() : GM_OrientableCurve
+ getMBB() : GM_Envelope
+ isConstraint() : int
+ length() : double
Triangle
- tmax: date
- tmin: date
+ constructGeom() : GM_OrientableSurface
+ edgeLength() : double
+ getMBB() : GM_Envelope
+ getNormal() : vector
+ isConstraint() : int
+ surfaceArea() : double
Tetrahedron
- tmax: int
- tmin: date
+ constructGeom() : GM_Solid
+ edgeLength() : double
+ getMBB() : GM_Envelope
+ surfaceArea() : double
+ volume() : double
+bnd
+cbdn
+cbnd
/derived /signed
SpecifiedBy /ordered
+representedBy
Node
- geometry: GM_Point
- tmax: date
- tmin: date
+ isConstraint() : int

F. DBMS Aspects

- Models are expected to be large → efficient encoding (of both base tables and indices), so what to store explicitly and what to store implicitly (derive)?
- Consistency is very important:
  1. Start with initial correct DBMS (e.g., empty)
  2. Make sure that update results in another correct state
- Specific constraints: earth surface (triangles with ground on one side and something else on other side) must form a connected surface (every earth surface triangle has 3 neighbor earth surface triangles)
- No holes allowed, but trough holes (tunnel) possible



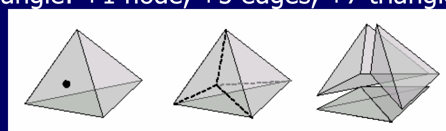
F. DBMS Aspects
Incremental Update



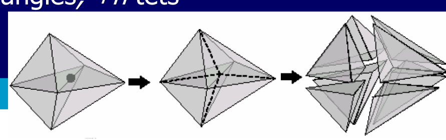
- Inserting a feature (e.g., house): means that boundary (triangle) has to be present in TEN
- TEN algorithms do support constrained edges, but not triangles → first triangulate boundary
- Resulting edges are inserted in constrained TEN (updating node/edge/triangle/tetrahedron table)
- Finally link feature (house) to set of tetrahedrons
- New features take space of old features (could be air or earth), which should agree
- Lock relevant objects during transaction

F. DBMS Aspects: Update TEN Structure, Basic Actions (1)

- Move node (without topology destruction)
- Insert node and incident edges/triangles/tetrahedrons on the middle of:
  1. tetrahedron:+1 node, +4 edges, +6 triangles, and +3 tets
  2. triangle: +1 node, +5 edges, +7 triangles, and +4 tets



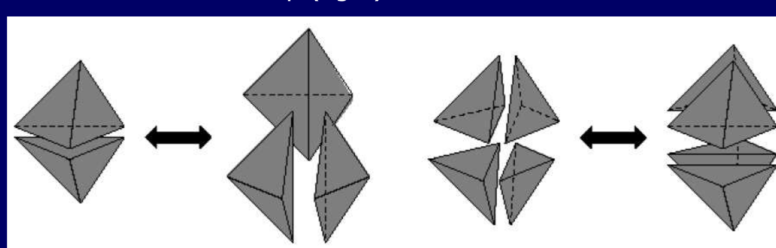
  3. Edge (n tets involved): +1 node, *+(n+1)* edges*, +2n* triangles*, +n* tets



TU Delft

F. DBMS Aspects: Update TEN Structure, Basic Actions (2)

- Flipping of tetrahedrons, two cases possible:
  1. 2-3 bistellar flip (left)
  2. 4-4 bistellar flip (right)



- Inserting constrained edges (not so easy…)

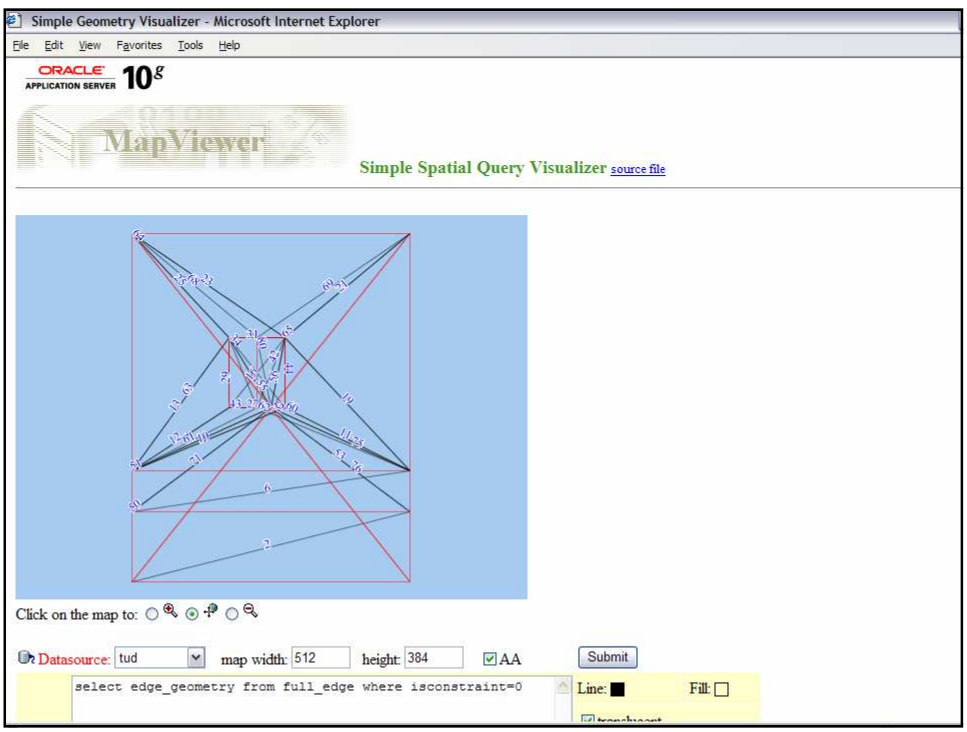
TU Delft

G. TEN topology table definitions

```
create table node(nid integer, geom sdo_geometry);

create table edge(eid integer, startnode integer,
 endnode integer, isconstraint integer);

create table triangle(trid integer, edge1 integer,
 edge2 integer, edge3 integer,
 isconstraint integer, afid integer);

create table tetrahedron(tetid integer,
 triangle1 integer, triangle2 integer,
 triangle3 integer, triangle4 integer, vfid integer);
```

March 7, 2007 79

TU Delft

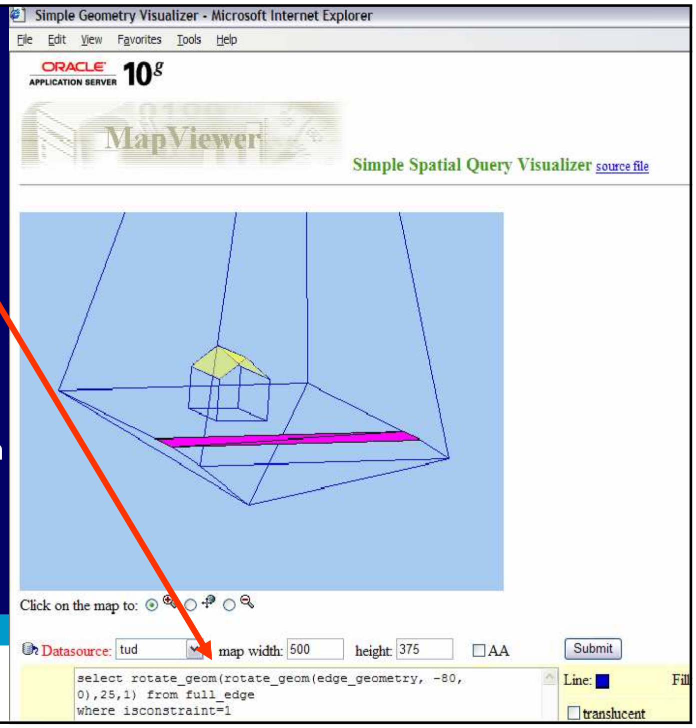


Simple 3D viewer created, based on:

1. 2D viewer
2. **`rotate_geom`** function

Note use of semi-Transparency

Hidden line/surface by painter algorithm and depth sorting



G. Next implementation

- Only geometry in node (point)
- functions (e.g., **`get_edge_geometry`**) compute geometry for other primitives

- Can be used in view

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

- After metadata registration, add index:

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

Overview

- Introduction
- 3D applications
- 3D spatial data (collection)
- 3D DBMS-GIS-CAD software
- TEN/Poincaré research
- Conclusion

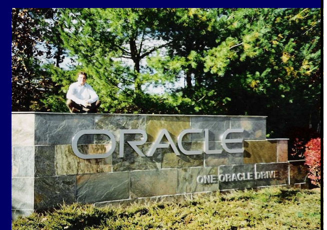
TUDelft

Conclusion

- Concepts to integrate 2D en 3D geo-objects within one environment (DBMS)
- Integration consists of: storage, query and visualization
- DBMS as basic fundament: for both "project-based" and "data-based" approaches (CAD and GIS)
- Near future: storage of 3D geo-objects as 3D volume primitives within the DBMS (with GIS/CAD connection)
- Based on mature 3D collection: standard 3D data sets will be part of a nations Geo-Information Infrastrucure
- 3D topography prototype: TEN based on Poincaré simplicial homology (full 3D partition, topology)

TUDelft

Acknowledgements



→ Bsik Space for Geo-information (RGI) '3D Topography' research project

→ NL Cadastre and the Queensland Government for the 3D cadastre cases

→ Long-term TUD-Oracle partnership, thanks to John Herring, Siva Ravada, Ravi Kothuri and Han Wammes for their constructive discussions

Friso Penninga and Peter van Oosterom participate in the research program 'Sustainable Urban Areas' (SUA) carried out by TUD

TUDelft