# Simplifying and Exchanging 3D Utility Network Objects Using City Models

Ihab HIJAZI[1] and Bryan HEMPHEN[2]
[1]An-Najah University, Nablus/Palestine · eehab@najah.edu
[2]University of Osnabrueck/Germany

This contribution was double-blind reviewed as full paper.

## Abstract

City models are extended recently to include utilities infrastructure e.g. CityGML/Utility Network ADE. The 3D geometry representation in these models is utilizing the concept of boundary representation (B-Rep), where planar faces and straight edges enclose the boundary. In case of utilities network, B-Rep would result in an extensive amount of storage space. This paper presents a new data type compatible to the swept representation, as well as the required algorithm to extract the needed information from the 3D utility network object represented as (B-Rep), i.e. centerlines as well as their profile information, and store them in the new custom data type. Another algorithm allows the generation of B-rep for visualization purposes. To show the effectiveness of the presented approach we accomplished the export of utility network into CityGML UtiltiyNetworkADE. Several 3D network object types (e.g. pipes, fitting) are exported to the new data type. The tests show the feasibility of the data type to reduce storage space, to create B-rep for visualization as well as to extract a graph for analysis. The new data types as well as the mentioned algorithms are implemented in Spatial-DBMS.

## 1 Introduction

An increasing number of cities are following the trend of building digital representations. 3D city models are being used in a growing variety of areas of urban management and planning processes such as disaster management, building development, pedestrian navigation and tourism (HIJAZI et al. 2011). City models not only represent natural landscapes but also urban objects such as buildings and streets. Most recently, they have been extended to model indoor spaces and surface features like the utilities infrastructure such as water or gas pipelines (e.g., CityGML/Utility Network ADE) (BECKER et al. 2010). Current 3D city models utilize boundary representation to represent three-dimensional city objects. Boundary representation describes solid objects as a collection of the enclosing surfaces. However, this would become complicated in the case of utility networks because every single network object would require a huge quantity of surfaces to provide a detailed representation.

3D spatial data management becomes significantly more important. Large amounts of spatial data accessed by various users are commonly stored and managed using spatial data-

Jekel, T., Car, A., Strobl, J. & Griesebner, G. (Eds.) (2013): GI_Forum 2013. Creating the GISociety.

base management systems (DBMS). Mainstream DBMS provide spatial extensions to support spatial data types for management as well as spatial functions built on these data types (BRUGMAN 2010, TET-KHUAN 2007). While current spatial DBMS usually have comprehensive support for 2D data, they still cannot be used to store and analyze 3D objects. The OGC offers the Simple Feature Geometry specification as the standard for simple 2D geometries and the functions operating on them, which most spatial DBMS implement. Currently, 3D objects are typically modeled in spatial DBMS using 2D primitives with additional z-coordinates, a method often referred to as 2.5D. Real 3D geometry support is still not available in commercial software and is available only in research settings (HIJAZI et al. 2011). Three-dimensional volumetric objects are stored in a database as collections of polygons (boundary representation) representing the surfaces that enclose the shape of the objects. While this approach is useful for visualization, the storage is complicated and takes up a lot of space in the database. ARENS (2003) and BRUGMAN (2010) also investigated the drawbacks of boundary representation for the modeling of 3D spatial objects. Both propose and implement polyhedrons as a new data type for real 3D analysis operations. On the other hand, in computer-aided design (CAD) geometric representations such as constructive solid geometry (CSG) as well as sweep representation are being used. Particularly sweep representation provides a compatible solution to represent 3D network objects (BECKER et al. 2010). It defines a solid by sweeping an object along a trajectory; current mainstream spatial DBMS do not implement this data type. DU & ZLATANOVA (2006) suggest an approach to visualize utility network data (pipes) represented as lines and points in a database on the y axis by replacing them with tiny cylinders. However, they place spheres to conceal turns or intersections, because the fact that the cylinders are all straight leads to gaps and superimposition at the joint between two segments. For performance reasons they chose spheres, which can quickly be created on the y and their orientation does not need to be adjusted by a rotation matrix in 3D space. When performance is not an issue and best possible precision needs to be obtained, this approach is not optimal (DU & ZLATANOVA 2006).

This paper aims to present the development and implementation of a new data type compatible to the swept solid representation. The research includes developing functions that work as an input tool used to extract the required information about network object and insert them in the developed data type. Also another method will be presented that re-extracts the original boundary representation from the simplified data to be used for visualization. Finally, a function is developed that outputs the simplified utility network in conformity with the recently published CityGML/UtiltiyNetworkADE, utilizing PostgreSQL's XML capabilities. This allows exchanging network utilities within current 3D CityGML standard. The data type and the functions are implemented in PostgreSQL/PostGIS.

The paper is organized in five sections. After the introduction (Section 1), Section 2 offers overview for swept representation. Section 3 starts by describing the algorithm for simplifying the network object including straight, curved and fittings objects and extract the information that are required to populate the developed data type. Also it describes the algorithm to re-extract the b-rep. Section 4 describes in detail the implementation of the methods in PostgreSQL/PostGIS and PL/pgSQL. The conclusion is given in Section 5.

# 2 Sweep Representation

Sweep representation defines a solid by sweeping an object along a trajectory. The simplest way of sweeping is to take a 2D area and sweeping it along a line to extrude a solid. Sweep representation is a schema used in CAD for the design of solids of uniform thickness in a given direction as well axisymmetric shapes like cables or pipes. These characteristics allow reducing the essential information required to describe network object to the center-line along with the radius or diameter of profile. Advantages of the sweep representation are that it is easy to understand and has high precision, because curved surfaces can be generated by curved trajectories. It requires only a minimal amount of stored space (ARENS 2003). However, sweep representation is only able to model objects with translational or rotational symmetry. Volume calculations can be difficult as swept objects can intersect themselves (ARENS 2003). The geometry representation is suggested to be ideally suited for describing man-made objects.

# 3 Proposed Solution

## 3.1 Simplifying 3D utility network objects

The approach to extract the centerline of a network object is based on simple idea illustrated in figure 1. The centerline lies between the central points of the network objects ports (the face that the network object connects through to other network objects). Based on this simple idea we extract the centerline of all network objects including objects with turns. The whole methodology is described in the following steps and illustrated in figure 2.

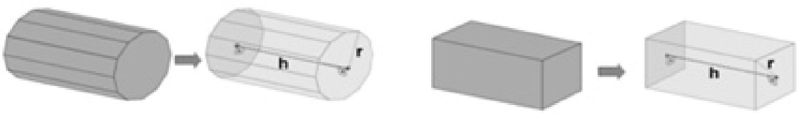


**Fig 1:** The position of the centerline within simple straight network objects

1. Determine the number of ports for each network object; the algorithm has two branches: one handles network objects with two ports and the other handles network objects with three or more ports.
   a. In case the number of ports is two, then the centroids of the ports are calculated, a line between centroids are constructed. Also information about the profile is determined. Figure 1, provides two examples: one for cylindrical pipe and the other for a pipe with rectangular profile. The network object with turns is divided into simple straight network pieces and the centerline and profile information are collected following the method similar to the straight network object as illustrated in figure 2.
   b. In case of three or more ports, the centroid of the ports is calculated. A line perpendicular to the ports starting from the centroid is built. The lines move

toward the inside of the network object and end at the central point, where all of the centerlines intersect and also end.

2. Store centerline and radius in a data type that can host this information.

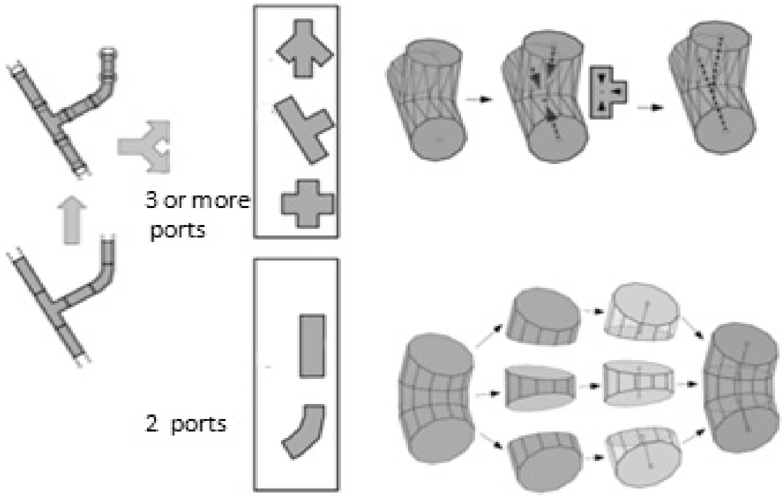


**Fig 2:** Centerline extraction

### 3.2 Re-extraction of B-Representation

The re-extraction of b-rep is based on prototype approach, where a network object can be seen as the composition of simple straight pipes organized together in a certain way. The whole procedure can be summarized in the following three steps, figure 3:

1. Create a default cylinder around the point of origin with standard parameters.
2. Use 3D transformations to fit each part of the centerline(s) to the axis of the default cylinder.
3. Use the reverse parameters for transforming both the cylinder and the centerline back to the centerline's original position.

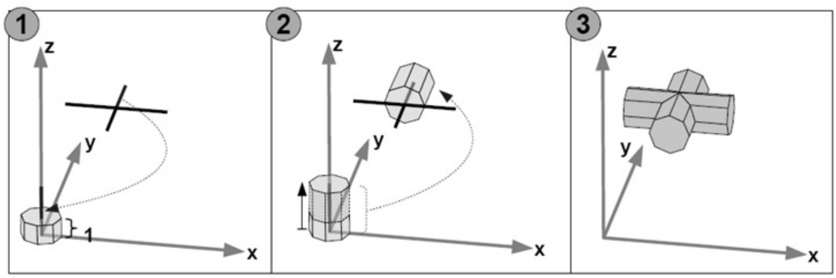


**Fig 3:** Procedure for B-rep creation

# 4 Implementation

To implement the methods proposed in the previous section, different tools are used; below is a description of these software components:

- PgAdmin III: The standard front-end administration tool delivered together with PostgreSQL is used for creation and execution of the functions (POSTGRESQL 2013).
- Pgsql2shp: a tool provided by PostGIS that changes a PostGIS database table or SQL query to the ESRI shapefile format.
- ESRI ArcScene: a component of ESRI's – ArcGIS Software and part of the 3D Analyst package, which is able to visualize 3D geodata.

## 4.1 Creation of the data types

The sweep representation is created as a composite data type consisting of geometry for the centerline and a decimal value for the radius of the profile using the following SQL query:

CREATE TYPE swept rep AS (cline geometry, radius double precision);

## 4.2 Centerline extraction function

This function contains several functions. A short overview is given in Table 1; centerline_extraction (object ID) automatically gets the whole procedure running. It directly operates on network object table and deals with the column (the_geom) containing the boundary representation of the network objects, as well as with the column ID containing unique identification numbers for the network components. The function temporarily assigns number of ports to network objects based on their id and calls the functions for the two different cases as described in the previous section (handleComplexPipe()) and handleSimplePipe()). centerline_extraction() automatically stores the sweep representation of network object in another table with its unique ID.

**Table 1:** Description for centerline function and it is sub-functions

| Name of Function | Purpose |
|---|---|
| centerline extraction() | main function for performing the centerline extraction |
| handleSimplePipe(pipeid integer, d double precision) | handles the extraction of centerlines from both simple network objects and network objects with turns; second parameter is a threshold for the maximum distance at which vertices intersect |
| handleComplexPipe(pipeid integer, d double precision) | handles the centerline extraction of fittings |
| showRadius(s sweptrep) | outputs the radius from the S-rep data type |
| showCenterline(s sweptrep) | outputs the centerline from S-rep the data type |
| makeSweptrep(c geometry, r double precision) | creates the sweep representation of an object composed of a geometry and radius as input values |

### 4.2.1 Comparison of memory requirements

To verify the result, the storage size of the network objects in B-rep is compared to the ones resulted from the implemented algorithm. Six examples of utility network objects have been simplified with the developed methods. Table 2 provides a comparison based on the number of vertices contained by the data types B-Rep and the developed swept representation. The number of vertices is checked by executing the following query:

```
SELECT SUM(nPoints(the_geom)) FROM NetworkObjectTableName
```

The result shows that the storage requirements reduced dramatically while still being able to fully describe the geometry of network objects. Although stored capacity is cheap nowadays and stored space is not a big issue, efficient storage is still desirable; especially when considering the fact that the memory requirements for several examples could be reduced by over 99%. The higher the precision of approximation, the higher the memory requirements, the higher the reduction of memory requirements after the simplification by the proposed methods.

**Table 2:** Comparison of storage space requirements using B-Rep and the develop data type

| Database | Num of Vertices B-rep | Num of Vertices Sweep Rep. | Ratio |
|---|---|---|---|
| example1 | 3960 | 16 | 0,0040 |
| example2 | 2367 | 12 | 0,0051 |
| example3 | 1955 | 12 | 0,0055 |
| example4 | 2921 | 16 | 0,0055 |
| example5 | 1470 | 8 | 0,0054 |
| example6 | 1732 | 10 | 0,0058 |

## 4.3 B-Rep extraction function

The function (brep_extraction()) constructs B-rep from the sweep representation; it shows the correctness and the feasibility of the data type. The steps from the proposed solution in Section 3.2 are translated into sub-functions as Table 3 shows. brep_extraction() is the main function looping through all centerlines of the utility network and calling the functions that create the B-rep.

Similar to the centerline extraction, the actual B-rep creation is performed by two derived functions because two types of centerlines have to be distinguished and treated individually. Nevertheless, brep_extraction() starts the whole procedure in the first place and does not have any parameters. The function reads the centerlines from the new developed data type, brep_extraction() calls createCylinder() as its first step; so the default cylinder is available for transformations in the table cylinder_polygons. Finally, the resulting polygons presenting the B-rep of the whole utility network can be found in table brep which can be used and exported for use by the scene languages for visualization.

**Table 3:** Description for B-rep function and its sub functions

| Name of Function | Purpose |
|---|---|
| brep extraction() | Main function for creating the boundary representation from centerlines |
| handleStraightcenterline(geometry,the radius double precision) | Sub-function for handling centerlines of simple network objects and fittings |
| handleCurvedcenterline(geometry, the radius double precision, n integer) | Sub-function for handling centerlines of network objects with turns |
| createCylinder() | Creates the template cylinder |

### 4.3.1 Comparison to Original Boundary Representation

The script is tested through visually comparing the network object resulted from the developed function to the original ones. Figure 5 shows re-extracted B-rep from the simplified data on the left side and the original B-rep on the right side. The first two figures show a simple network object; the next are network objects with turns; and the last two figures show fittings. The first and last case looks similar to each other and cannot really be distinguished. The re-extracted B-rep for the network objects with turns shows problems in the segments connecting it to other network objects. A gap appears on the left side, which occurs because of the inaccuracy of the original data.

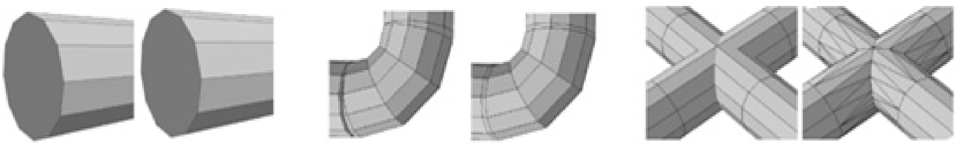

**Fig 4:** Comparison of the original B-rep (right) with the created (left) network objects

## 4.4 CityGML Output

In order to exchange utility networks using CityGML/Utility Network ADE a special function called citygml_output() is developed. The function loops through all centerlines created by the centerline extraction script and builds the XML structure; PostgreSQL provides a set of features to create XML documents such as mapping whole tables to XML or processing XML using XPath. Additionally, PostgreSQL supplies a XML datatype (POSTGRESQL 2013). For the creation of the CityGML structure only two functions have been used and these are:

- xmlconcat() creates a sequence of distinct XML values to create a single value containing an XML content fragment. Both input parameters as well as the output parameter are of the special datatype XML.
- xmlelement() creates a XML element with the specified name, attributes and content.

Figure 5 gives an example of a utility network compliant to the UtilityNetworkADE visualized using the CityGML viewer – Aristoteles 3D.

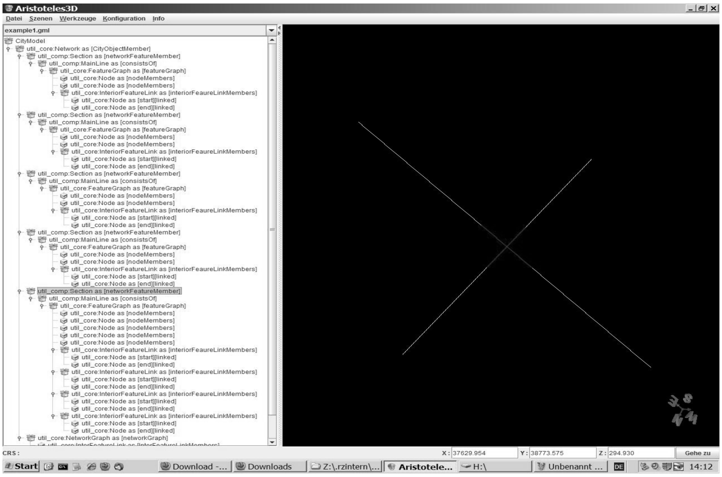


**Fig. 5:** A sample utility network in CityGML format visualized with Aristoteles 3D

# 5 Conclusion

The paper presented a simple and plain approach to the simplification of boundary representation of 3D utility network objects through sweep representation. The algorithm and the simplification of the network objects show a dramatic reduction of stored space while being able to re-extract its 3D shape. Moreover the swept presentation provides simple representation that can be used for graph analysis in current GIS software.

Our concept was tested within the PostgreSQL/PostGIS software, but the approach can be modified for any DBMS. The whole technology used to implement the concept presented in the paper is open-source. The result of the work is to facilitate the management and analysis of the new emerged 3D utilities network model within spatial DBMS, as well as to provide the required mechanism to exchange utilities networks using the CityGML.

# References

ARENS, C. A. (2003), Modelling 3D spatial objects in a GeoDBMS using a 3D primitive. Master Thesis, Netherlands.

BECKER, T., NAGEL, C. & KOLBE, T. (2010), UtilityNetworkADE, Core Model Draft version. http://www.citygmlwiki.org/index.php/CityGML_UtilityNetworkADE (January 3, 2013).

BRUGMAN, B. (2010), 3D Topological structure management within a DBMS. Master Thesis, Netherlands,

DU, Y. & ZLATANOVA, S. (2006), An approach for 3D visualization of pipelines. In: ABDUL-RAHMAN, A., ZLATANOVA, S. & COORS, V. (Eds.), Innovation in 3D-Geo Information System, Springer, Berlin/Heidelberg, 395-404.

HIJAZI I., EHLERS, M., ZLATANOVA, S., BECKER T., VAN BERLO, L. (2011), Initial investigations for modeling interior Utilities within 3D Geo Context: Transforming IFC-interior utility to CityGML/UtilityNetworkADE. In: KOLBE, T. H., KONIG, G. & NAGEL, C. (Eds.), Advances in 3D Geo-Information Science.

POSTGRESQL (2013), About. http://www.postgresql.org/about/ (17.01.2013).

TET-KHUAN, C., ABDUL-RAHMAN, A. & ZLATANOVA, S. (2007), 3D Spatial Operations in Geo DBMS Environment for 3D GIS. Proceedings International conference on Computational Science and Its Applications, 151-163.