TOPOLOGICAL OPERATIONS ON SIMPLE THREE DIMENSIONAL OBJECTS IN GEOINFORMATION SYSTEMS

Zoran Bozickovic, Grgo Dzelalija, Antonije Ivanovic

Abstract

Key words: *GIS, geospatial databases, polyhedron, topological relationships*

Today's GIS software,and databases, enable manipulation of 2D and 2.5D spatial data which is not enough for today, and future needs. The need arises for high quality 3D GIS solutions, which currently lacks. This paper offers developed algorithm to indentify topological relationships between simple polyhedra, as the first step of solving complete 3D GIS. A new table has been created, in paper numbered as Table 2, with 9 intersection matrix with results between two simple polyhedra, which is the basis of the algorithm by which to prove some topological relationships. There have been recognized all possible results of 9-intersection matrix which can occur for each given topological relationship. The aforementioned table has been great for practical use since it enabled easy determination of conditions needed to prove certain topological relationships. Apart from developing previously mentioned algorithm, same was programmed in Python programming language and was tested with polyhedra stored using PostgreSQL database. Ovim rješenjima omogućen je napredak geoprostornih baza podataka, a time i GIS-a. We belive that this solution will help further advancement of geospatial databases, and thus the GIS.

Introduction

Almost all of today's geoinformation systems are based on two-dimensional (2D) or so called two and half dimensional (2.5D) spatial data. Term 2.5D refers to the spatial plane, eg relief, inserted in three-dimensional space (*Abdul-Rahman, Pilouk 2007*). Manipulation and presentation of three-dimensional objects in such two-dimensional GIS is not entirely possible. In fact, processing data of the real world, which contain more than two dimensions, in today's geoinformation systems lead to decrease in accuracy or results of such processing are incomplete informations. One of disadvantages of existing geoinformation systems is the fact that they are connected with relational database. The structure of objects in GIS is more complex than in standard systems, and in 3D GIS, spatial relations between objects are even more complex than they are in 2D system. It is therefore proposed that complex spatial data are more efficiently processed in object-oriented databases *(Egenhofer, Frank 1989; Worboys 2003).* GIS is one of key parts of geodesy and geoinformatics, so the aim of this paper is to point out these problems and find solutions to problem of implementation of 3D spatial data in databases.

Address Offices in Brussels : Rue du Nord 76, BE – 1000 Bruxelles. Tel +32/2/217.39.72 Fax +32/2/219.31.47
E-mail: jean-yves.pirlot@clge.eu - www.clge.eu EC-Register of interest representatives 29077535421-69

In this paper we developed an algorithm for topological queries on simple polyhedra in three dimensions. The algorithm was than tested in a way that the polyhedron represents one object from the real world. Number of different polyhedra were stored in database for testing values returned by programmed topological queries (True/False) for all relations in space between two simple polyhedra. Also, special cases, that have been discovered, are shown, and created algorithm detects them without difficulty. The main characteristic of the algorithm that was developed and presented in this paper is that it allows determination of wether are objects in a particular topological relation or not. The main precondition for this is a generalization of 3D object to a simple polyhedron. Polyhedra over whom topological queries are performed must be convex. Geometrically, the line drawn between any two points situated on the surface of a simple polyhedron, but not in the same plane of polyhedron, is always located inside body. (*URL 5*).

Algorithm

For development of the concept of algorithm, based on which toplogocial relations between simple polyhedra will be successfully indentified, with other words defining mathematically sufficient conditions for each relationship, it is needed to consider the matrix of four and nine intersection (*Egenhofer, Franzosa 1991*). Model of 4-intersection matrix is used to describe binary topological operations between two objects. With model of 4-intersection matrix, there can be distinguished 16 different binary topological relations, but only 8 of them are possible in three-dimensional space. They are: *inside*, *contains*, *covers*, *cover by*, *equal*, *disjoint*, *meet*, *overlap* (Figure 1). 9-intersection matrix is an extension of 4-intersection matrix so that for each object, in addition to interior and boundaries, takes into consideration the exterior. Thus, for two object we get a 3x3 dimension matrix with 9 sections, which makes it possible to prove all eight topological relations between any two objects in 3D space.

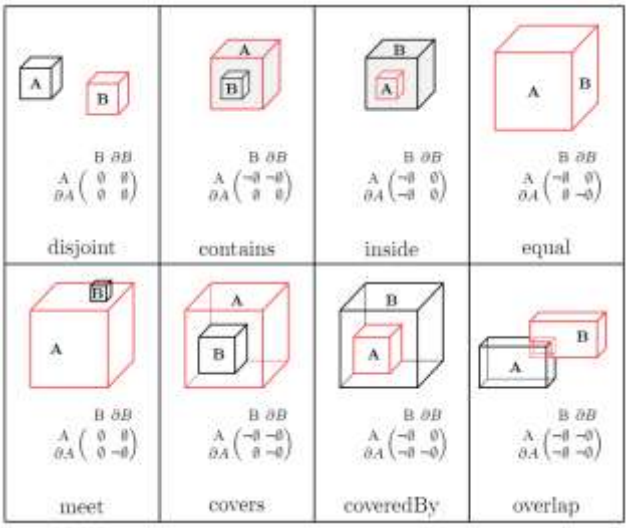

Figure 1. *Topological relations between 3D objects. (Egenhofer)*

Address Offices in Brussels : Rue du Nord 76, BE – 1000 Bruxelles. Tel +32/2/217.39.72 Fax +32/2/219.31.47
E-mail: jean-yves.pirlot@clge.eu - www.clge.eu EC-Register of interest representatives 29077535421-69

In further reviewing of topological raltionships between polyhedra let us take from 9-intersection matrix the result of intersection of boundary from base object, and interior, boundary and exterior of target object. Table 1 is based on this approach, it shows relationships between base and target object for verifying topological relationship between two polyhedra (9-intersection matrix). Result can be empty (Ø) or non-empty (¬Ø) set of points, that represent boundaries of the base object, in relation to body of the target object. Table gives same results for coveres and covered by, as well as for contains and inside. The difference between these topological relationships though is in relative relationship between two objects.

Table 1. *Results of 9-intersection matrix between base and target object*

3D TOPLOGICAL OPERATIONS	*In*	*Out*	*touch*
equal	Ø	Ø	¬Ø
meet	Ø	¬Ø	¬Ø
covers	¬Ø	Ø	¬Ø
covered by	¬Ø	Ø	¬Ø
contains	¬Ø	Ø	Ø
inside	¬Ø	Ø	Ø
disjoint	Ø	¬Ø	Ø
overlap	¬Ø	¬Ø	¬Ø

Since it is impossible to store an infinite number of points that represent boundary, we have developed a second table which is applicable in practice. Table 2 shows the relationship between vertices of one object with body of another (in, out or touch), and reverse.

Table 2. *Topological relationships of vertices of one body in relation to body of another, and reverse.*

	FIRST COLUMN **Vertices A in relation to B**			SECOND COLUMN **Vertices B in relation to A**		
	in	*Out*	*touch*	*in*	*out*	*touch*
A *inside* **B**	¬Ø	Ø	Ø	Ø	¬Ø	Ø
A *contain* **B**	Ø	¬Ø	Ø	¬Ø	Ø	Ø
A *equal* **B**	Ø	Ø	¬Ø	Ø	Ø	¬Ø
A *covers* **B**	Ø	¬Ø	¬Ø/ Ø	¬Ø/ Ø	Ø	¬Ø
A *covered by* **B**	¬Ø/ Ø	Ø	¬Ø	Ø	¬Ø	¬Ø/ Ø
A *overlap* **B**	¬Ø/ Ø	¬Ø	¬Ø/ Ø	¬Ø/ Ø	¬Ø	¬Ø/ Ø
A *meet* **B**	Ø	¬Ø	¬Ø/ Ø	Ø	¬Ø	¬Ø/ Ø
A *disjoint* **B**	Ø	¬Ø	Ø	Ø	¬Ø	Ø

Address Offices in Brussels : Rue du Nord 76, BE – 1000 Bruxelles. Tel +32/2/217.39.72 Fax +32/2/219.31.47
E-mail: jean-yves.pirlot@clge.eu - www.clge.eu EC-Register of interest representatives 29077535421-69

In table 2, ø is showing that there is no vertice of base object which is in particular relation to target object, while ¬ø indicates that there is at least one vertice of base object that is in particular relation to target object. The sign ¬ø/ ø means that there may, or may not be one or more vertices of base object that are in particular relation with body of target object. That points out the existence of special cases. First five topological relations are easily proven by checking conditions from Table 2. If any two polyhedra fulfill conditions from table 2 for one of first five relations it can be definitely concluded that those two polyhedra are in particular topological relationship, while that is not case for remaining three relationships (disjoint, meet, overlap). For disjoint, meet and overlap there are special cases, that give same results, therefore they need additional conditions in order to discern these special cases and to prove topological relationships.

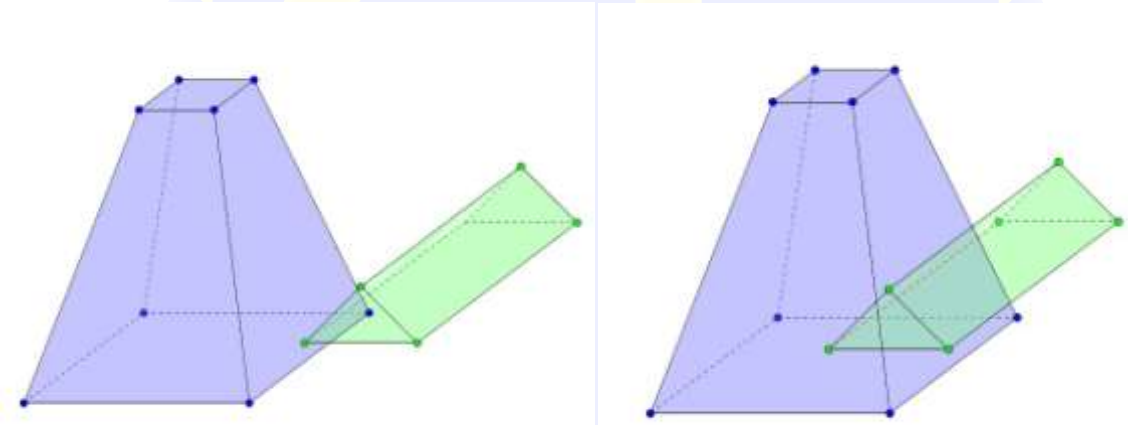

Figure 2. *For example these two situations (overlap on right, and disjoint on left) will give same results in Table 2, because all vertices from green are outside of blue one, and all blue vertices are outside green object. Only difference is that on right situation we can see red edge that clearly goes trough blue object.*

These special cases are differed by searching for additional intersections between edges of one object and polygons of another. As illustrated, topological relationship *disjoint* has no intersections, while other two (*overlap, meet*) have. Also, in relation *meet* there is at least one polygon of two objects on which all intersections are lying, while such polygon does not exist in *overlap* relationship. Algorithm using these additional conditions, together with conditions from Table 2 proves remaining three topological relationships.

Question that remains to be answered is how to mathematically prove whether is a point inside, outside or inside polyhedron, and to determine intersection of edge and polygon of two polyhedron. To determine whether point is inside, outside or on simple polyhedron, we used general equation of plane. After substituting coordinates and calculating there are three types of result:

Address Offices in Brussels : Rue du Nord 76, BE – 1000 Bruxelles. Tel +32/2/217.39.72 Fax +32/2/219.31.47
E-mail: jean-yves.pirlot@clge.eu - www.clge.eu EC-Register of interest representatives 29077535421-69

$$A \cdot x + B \cdot y + C \cdot z + D = 0,$$
$$A \cdot x + B \cdot y + C \cdot z + D > 0,$$
$$A \cdot x + B \cdot y + C \cdot z + D < 0.$$

When point lies in the plane, equation is equal to zero. Depending on the direction of normal vector of plane, when point is on the side of plane towards wich normal vector is oriented, the result is greater than zero. The result is less than zero when point is on the other side. Since the target object is viewed as set of polyons that enclose one volume, general equation of the plane is determined for each polygon. By inclusion of coordinates of vertices (of base object, or any other point) in plane equation of polygon it is determined wether point is inside, outside or touches the target object. For points that lie inside target object must always give value less than zero, when included in general equation of plane. So, normal vector of plane has to be oriented outwards of object. Normal vector is calculated by vector product of two vectors that lie in that plane.

$$\underset{n}{\rightarrow} = \begin{vmatrix} i & j & k \\ a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \end{vmatrix}$$

Direction of vector is defined by right hand rule. Vector is perpendicular to the vectors a and b, and viewed from the top of the normal vector, rotation from a to b is counter-clockwise. Since polygons that form a simple polyhedron are recorded in database as a set {x,y,z} of points, in which first and last points are same, it is necessary to store points (vertices) of polygon counter-clockwise when seen outside of the object.

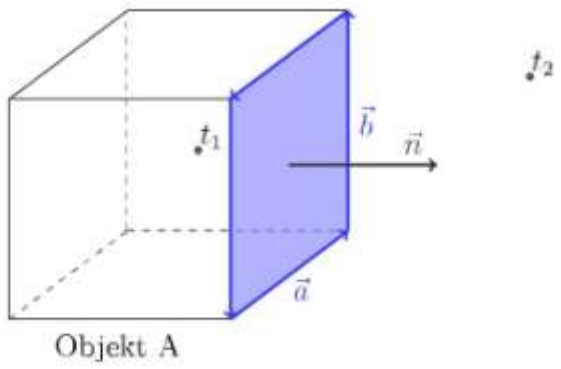

Figure 3. *Orientation of normal vector*

Enough proof that point is inside of polyhedron is that result of plane equation for every polygon is less than zero. The point is outside of object if there is only one result greater than zero. Finally, if we want to check if the point touches object, it is enough to determine if point is not outside or inside of object. When point touches body of object, some of results of plane equations will be less than zero, and some will be zero, but also if point is not either outside or inside of object it certainly touches it. This is reason why these topological operations have to

Address Offices in Brussels : Rue du Nord 76, BE – 1000 Bruxelles. Tel +32/2/217.39.72 Fax +32/2/219.31.47
E-mail: jean-yves.pirlot@clge.eu - www.clge.eu EC-Register of interest representatives 29077535421-69

work with simple polyhedra. If more complex object is used, we can easily imagine a case in which point is inside target body, and yet result obtained by inserting its coordinates in equation of plane is not less than zero, for at least one plane (Figure). For usage of more complex objects, they have to be devided with plane which gives wrong result, red plane in Figure.

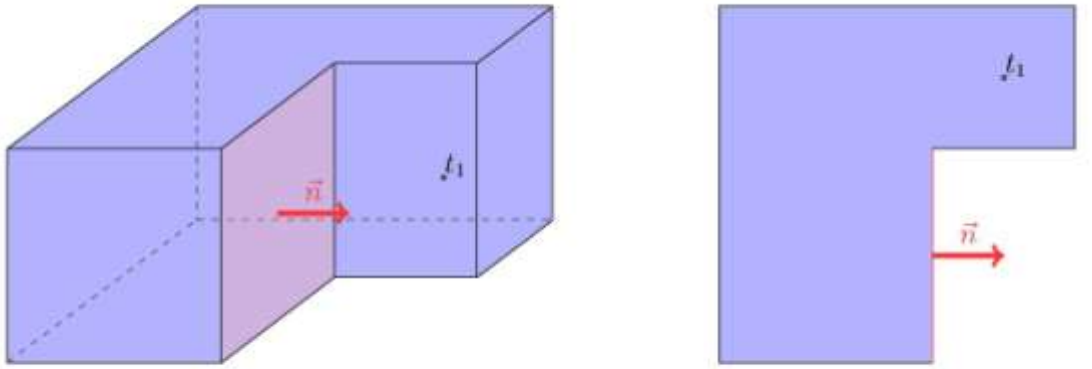

Figure 4. *Problem of complex polyhedra*

To find coordinates of intersection between odge and polygon, we use general form of plane equation and parametric equations of line.

$$A \cdot x + B \cdot y + C \cdot z + D = 0,$$

$$p = \begin{cases} x = x_1 + t \cdot a \\ y = y_1 + t \cdot b. \\ z = z_1 + t \cdot c \end{cases}$$

The solution is obtained by inserting parameters of line, in the general plane equation, in order to calculate factor t. Having calculated t, it is inserted in parametric equation of line so we can get coordinates of intersection. Algorithm performs that so that base object, which was earlier set of points, now becomes a set of lines, on which his edges lie. For each line of base object, vector elements {a, b, c} and start and end point are stored, while target object remains set of planes. Before algorithm starts to seek intersection of line and plane, it checks are the start and end point of a given edge from different sides of the plane. With that, problem of dividing by zero is avoided. After calculating intersection of line and plane, it is checked whether calculated intersection lies on polygon of object, since it is possible that intersection lies in the plane, but not in the polygon.

Address Offices in Brussels : Rue du Nord 76, BE – 1000 Bruxelles. Tel +32/2/217.39.72 Fax +32/2/219.31.47
E-mail: jean-yves.pirlot@clge.eu - www.clge.eu EC-Register of interest representatives 29077535421-69

Implementation and code verification

Developed algorithm is implemented in Python programming language, with object programming. Simple polyhedra over whom the algorithm is executed are stored in PostgreSQL 8.4 database, with spatial extension PostGIS 1.5. It is used existing PostGIS geometry data type POLYGON 3D, , and in fact for each polyhedron its polygons were stored under same id.

Since algorithm is programmed in Python, and data, over whom algorithm is executed, are stored in PostgreSQL database, we had to use Pycopg 2. This is a Python module that serves to connect and download spatial data from database, so that they could be manipulated in the Python interface. Only after the connection it is possible to perform spatial queries against data stored in database. In Python class called Sql is programmed. All topological operations are located within the class as methods. Class has one attribute, which is the SQL SELECT command that is sent from Python to PostgreSQL using module for connecting, and is needed to enter when instantiating an object. That is how in Python we get object that has all necessary spatial data and methods for determing of topological relations between polyhedra.

To test programmed topological operations, we stored 20 polyhedra in database.

Figure 5. *Polyhedra used to test disjoint*

Figure 6. *Polyhedra used to test overlap*

Figure 7. *Polyhedra used to test overlap*

Figure 8. *Polyhedra used to test meet*

Address Offices in Brussels : Rue du Nord 76, BE – 1000 Bruxelles. Tel +32/2/217.39.72 Fax +32/2/219.31.47
E-mail: jean-yves.pirlot@clge.eu - www.clge.eu EC-Register of interest representatives 29077535421-69

Conclusion

In this paper we developed algorithm for checking topological relations between simple polyhedra. We designed table 2 with results of 9-intersection matrix, which are the backbone of entire algorithm and represent original scientific contribution of this paper. It proved to be suitable for direct use and to easily discover all mathematical conditions for proving each topological relationship. To program the algorithm, programming language Python was used, while all the data were stored in PostgreSql database, using PostGIS spatial extension. Complete code has been empirically tested on simple polyhedra stored in database, either with random coordinates, or actual coordinates collected with GPS. It should be noted that often use of dividing was avoided, that in code occurs only once, while calculating coordinates of intersection between line and polygon.

With this algorithm it is presented a possibility od concrete usage and implementation of 3D data (bodies) in databases and GIS. These algorithms have potential application in the real world. As anexample, the problem of trucks passing beneath underpasses in the city, or ships in harbours. It is often case that drivers use GPS to navigate trough the city, and often happens that they are navigated trough the passage that is too small for their vehicle. You could also give an example of aircraft landings (or especially helicopter navigation) in the airport where height of surrounding skyscrapers are endangerig safety of aircraft. Also, at the time when robots perform more end more of our tasks, 3D GIS will greatly facilitate coping and "sense" of 3D space of an robot, as in the example of the warehouse (or dockyard, parking etc.) where all organisation and activities are performed exclusively by robots. As already mentioned, the algorithms for indentifying topological relations are applicable only to simple polyhedra. Next logical step in the development of our algorithms would be to ensure their application to objects more complex than simple polyhedra. Since developed algorithm has proven to be effective in solving the problem of simple polyhedra and had shown great perspective, we sugest focusing on finding solution for problem of splitting complex object into simple polyhedra. This would expand applicability of these algorithms to more complex objects and enable full use of 3D GIS, and so real world would be even better represented.

Address Offices in Brussels : Rue du Nord 76, BE – 1000 Bruxelles. Tel +32/2/217.39.72 Fax +32/2/219.31.47
E-mail: jean-yves.pirlot@clge.eu - www.clge.eu EC-Register of interest representatives 29077535421-69

Literature

Abdul-Rahman, A., Pilouk, M. (2007): Spatial Data Modelling for 3D GIS, Springer, Berlin Heidelberg

Bronstein, I.N., Semendjajew, K.A., Musiol, G., Mühlig, H. (2004): Matematički priručnik, Golden marketing - Tehnička knjiga, Zagreb

Egenhofer, M., Frank, A. (1989): Object-oriented Software Engineering Considerations for Future GIS, University of Maine

Egenhofer, M., Franzosa, R. (1991): Point-Set Topological Spatial Relations, International Journal for Geographical Information Systems, 5(2), 161-174

Egenhofer, M., Hering, J. (1995): Advances in spatial databases, Springer-Verlag

Elezović, N., Aglić, A. (2006): Linearna algebra-zbirka zadataka, Element, Zagreb

Tet-Khuan, C., Abdul-Rahman, A., Zlatanova, S. (2007): 3D Spatial Operations in Geo DBMS Environment for 3D GIS, ICCSA, 1, 151-163

Worboys, M. (2003): GIS A Computing Perspective, Taylor & Francis e-Library

Zlatanova, S. (2000): 3D GIS for Urban Development, International Institute for Aerospace Survey and Earth Sciences, Enschede

URL:

URL 1: Directions Magazine, http://www.directionsmag.com/articles/3d-geo-dbms/123603, (22.04.2012.)

URL 2: Geo-Database Management Center, http://www.gdmc.nl/3DCadastres/literature/3Dcad_2011_03.pdf, (22.04.2012.)

URL 3: Python, http://www.python.org/about/, (29.04.2012.)

Address Offices in Brussels : Rue du Nord 76, BE – 1000 Bruxelles. Tel +32/2/217.39.72 Fax +32/2/219.31.47
E-mail: jean-yves.pirlot@clge.eu - www.clge.eu EC-Register of interest representatives 29077535421-69

URL 4: PostGIS 1.5.3 Manual, http://postgis.refractions.net/documentation/manual-1.5, (17.03.2012.)

URL 5: Wolfram Mathworld http://mathworld.wolfram.com/ConvexPolyhedron.html, (14.05.2012.)

URL 6: LaTeX–A document preparation system, http://www.latex-project.org/, (29.04.2012.)

Address Offices in Brussels : Rue du Nord 76, BE – 1000 Bruxelles. Tel +32/2/217.39.72 Fax +32/2/219.31.47
E-mail: jean-yves.pirlot@clge.eu - www.clge.eu EC-Register of interest representatives 29077535421-69