IN TRANSITION FROM 2.5D-GIS TO 3D-GIS

Dieter Schmidt and Dieter Fritsch
Institute of Photogrammetry
Stuttgart University
P.O.B. 106037
70049 Stuttgart / Germany
dieter.schmidt@ifp.uni-stuttgart.de

Intercommission WG III/IV

KEY WORDS: GIS, Three-dimensional, Data Model, Integration, Algorithms

ABSTRACT: Some of the 2.5D geographic information systems can easily be extended for the third dimension. The extension is exercised by supplementing a 3D data model and additional 3D operations. For computing the three-dimensional operations one can refer to the alrady existing two-dimensional operations. At the implementation into Smallworld GIS the available source code can quickly be extended for the third dimension. The presentation is done by an external 3D graphic tool.

1 INTRODUCTION

Although our perception is three-dimensional, spatial objects in a geographic information system are for the time being mostly managed in a two-dimensional way. Therefore the present research activities pay special attention to the development of a 3D-GIS. There is an existing need for 3D data, but the handling is more difficult and requires more complex algorithms as it is in the two-dimensional case. Another problem is to get hold of 3D information.

The three-dimensional topology is already widely used for computer-aided drafts or computer animation and, besides that, software and hardware offer more and more 3D support. The results in the three-dimensional representation of real objects stand for the further development of geographic information systems with special emphasis on the analysis of geometry and attributes.

A first pragmatic approach of the extension of an existing commercial 2.5D-GIS is to attribute faces for additional height values. In doing so it is possible to receive a simple three-dimensional representation which allows an adequate approximation (of buildings for example) for certain applications. The representation is then achieved by an external 3D graphic program. Additional parameters like roof slopes and shapes could support the representation and help to achieve better accuracy.

Should one require, however, a more substantial representation as for instance in the field of urban planning, then the introduction of an additional three-dimensional data model cannot be omitted.

2 THE 3D DATA MODEL

As already known, two-dimensional data models suffer from its limitations to model 3D solid objects. By using digital terrain models (DTM) only three-dimensional surfaces can be described with single z-values. As to introduce an additional three-dimensional data model the definition of relations to the two-dimensional geometry and thematic attributes is necessary. An unique identifier links both the geometry model and the attributes. In most cases the data model will be stored in a relational database. As demonstrated by Molenaar (1992) an analysis can then be performed through the basic language SQL.

The definition *geo-object* used in the following, describes an object with spatial relation. It consists of thematic and spatial attributes. Spatial attributes can either be two- or three-dimensional. In addition to a 3D attribute every 3D geo-object holds as well a 2D-attribut which represents its outlines, i.e. every 3D object has a two-dimensional mark. In case of a small number of 3D objects it does not matter developing a 3D access structure, as every geo-object will be at least referenced through the two-dimensional index.

A three-dimensional data model (see fig. 1) was designed in the notation of [Rumbaugh, Plaha, Premerlani, Eddy and Lorensen, 1991] . The rectangles describe object classes (like the object class point), lines denote the association between object classes. A point at the end of an association line marks an 1:n relation. A triangle at an intersection of a line marks a generalization (for example node and intermediate points are subclasses of the class point). Generalization and specialization are the general terms to describe both directions of inheritance. A rhomb on the connecting line describes an aggregation (for example a point- shaped object consists of a node).

Within the rectangle, which describes an object class, the name is written in bold on top, below follows a line. Underneath this line attributes of the class will be numbered. An oblique stroke in front of an attribute denotes that the value can be calculated and has not to be stored (but will mostly be done because of efficiency reasons). If a face is planar it will be calculable but a calculation for every examination

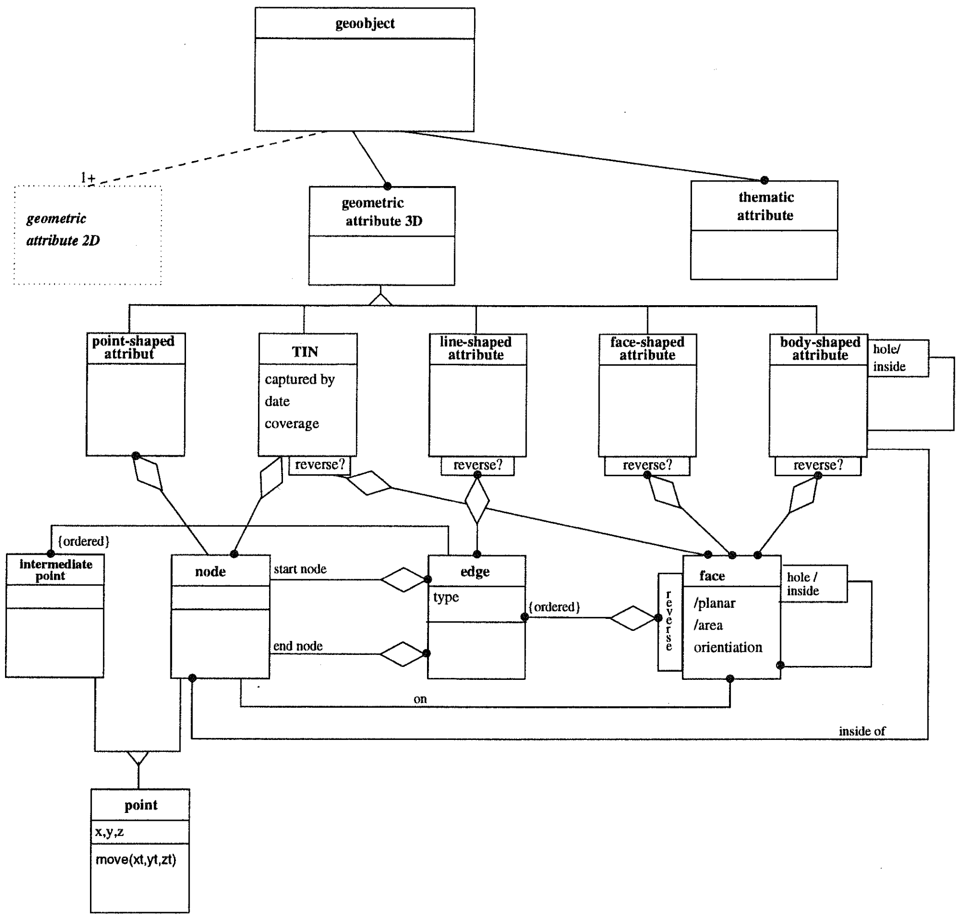

Figure 1: 3D data model

is not advisable. On the lowest level of the model there exist the geo-primitives point, edge and face. The points contain the position as an attribute (either of a special 3D coordinate type or as three attributes of the real or integer type for x, y, z-coordinates). Point is the category for nodes, e.g. edge-limiting points, and intermediate points, points between nodes. An edge is represented by initial or end nodes as well as by intermediate points. A face again consists of one or more clockwise ordered closed line segments. In this way the inside and outside of faces and bodies is determined.

On a higher level there exist five geometrical attributes which can be matched to a spatial object. A point-shaped attribute is just composed of a node. A line-shaped attribute is formed by one or more connected edges through joint nodes. Face and body-shaped attributes consist of one or more faces. In order to form a body the (planar) face have to be tied over joint edges. An triangulated irregular network (TIN) can also exist as an attribute of a geo-object for representing the height model. Triangles are formed by nodes and polygons.

As it appears from the data model a node can be referenced by point-shaped, an edge by line-shaped and a face by face and body-shaped attributes.

3 IMPLEMENTATION OF 3D OPERATIONS BY 2D OPERATIONS AND TRANSFORMATIONS

A 2D-GIS is provided with comprehensive methods for 2D calculations. In order to perform 3D calculations with a 2D-GIS a macro language or better (like in Smallworld GIS) programming language is necessary, in which additions and multiplications and if possible trigonometrical functions on coordinate values can be carried out. For the calculations of 3D predicates the already existing 2D operations can be used after transformations and the subsequent projections in the two-dimensional space. By that planar faces are based in one of the planes, which are defined by two of the three coordinate axes and all objects (or their projections) in this plane will be treated as ordinary in the planimetry of the 2D-GIS.

One possibility of transforming a face in one of the three coordinate levels is the translation to the origin and 3 rotations around the x-, y-, z- axis. One can consider this as a change of the coordinate system which mathematically is the more perfect case. This will be exemplified by faces for which intersections should be calculated.

In order to transform any face defined in the 3D Euclidean space into the 2D xy-plane we take advantage of the well-known characteristics of orthogonal matrices. The coordinates of an object are transformed contragredient to the base vectors. By applying orthogonal base vectors the matrix for the base vector transform is identical with the coordinate transform matrix. The base vectors to be orthogonalized are vectors defining two faces with the given three points P_1, P_2, P_3 (see fig. 2) of the (planar) face as well as the standard vector for the face. For the orthogonalization the Schmidtsche orthonormalization procedure is used. The normalization part can be omited.

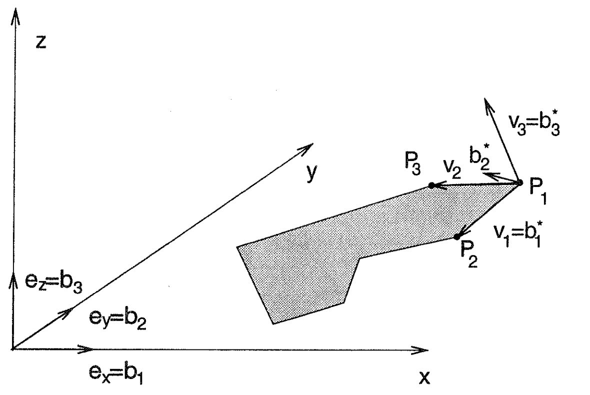

Figure 2: One of the faces and the resulting base vectors

By stepwise orthogonalization the orthogonal base (b_1, b_2, b_3) is achieved:

$$\begin{aligned} \mathbf{v_1} &= P_1P_2 \\ \mathbf{v_2} &= P_1P_3 \\ \mathbf{v_3} &= \mathbf{v_1} \times \mathbf{v_2} \end{aligned}$$

In the last equation the calculation of the last positions does not apply as the numerator turns zero, this means the orthogonal base vectors b_1, b_2 are perpendicular to the origin vector v_3.

$$\begin{aligned} \mathbf{b_1^*} &= \mathbf{v_1} \\ \mathbf{b_2^*} &= \mathbf{v_2} - \frac{(\mathbf{v_2},\mathbf{b_1^*})}{(\mathbf{b_1^*},\mathbf{b_1^*})}\mathbf{b_1^*} \\ \mathbf{b_3^*} &= \mathbf{v_3} - \frac{(\mathbf{v_3},\mathbf{b_1^*})}{(\mathbf{b_1^*},\mathbf{b_1^*})}\mathbf{b_1^*} - \frac{(\mathbf{v_3},\mathbf{b_2^*})}{(\mathbf{b_2^*},\mathbf{b_2^*})}\mathbf{b_2^*} \end{aligned}$$

For transformation between base $B = \{\mathbf{e_x}, \mathbf{e_y}, \mathbf{e_z}\}$ and $B^* = \{\mathbf{b_1^*}, \mathbf{b_2^*}, \mathbf{b_3^*}\}$ the following equation is applicable

$$\begin{pmatrix} \mathbf{b_1^*} \\ \mathbf{b_2^*} \\ \mathbf{b_3^*} \end{pmatrix} = \mathbf{A} \begin{pmatrix} \mathbf{e_x} \\ \mathbf{e_y} \\ \mathbf{e_z} \end{pmatrix}$$

As the base vectors of the initial coordinate system correspond to the identity matrix it can be stated that **A** is identical with the new base vectors. As to transform the coordinates x_1, x_2, x_3 into the new coordinate system, the transposition of the inversion of A has to be found. The base vectors of the new coordinate system are orthogonal to each other, therefore the transformation matrix, consisting of these vectors is also orthogonal. The inverse of an orthogonal matrix is its transposition, i.e. the matrix of the transformation of the base vectors and the coordinates is identical.

$$\begin{pmatrix} \mathbf{x_1^*} \\ \mathbf{x_2^*} \\ \mathbf{x_3^*} \end{pmatrix} = \mathbf{A} \begin{pmatrix} \mathbf{x_1} \\ \mathbf{x_2} \\ \mathbf{x_3} \end{pmatrix}$$

Instead of three rotations obtained by trigonometric calculations only the basic arithmetics are required.

To put the second face perpendicular to the xz - plane, a second time the orthormalisation procedure has to be applied. The normal vector orthogonal to the face will be one of the new base vectors. Orthogonalisation and coordinate transforms are performed analogously.

If both faces are already in the xy - plane, 2D operations can be used to compute the intersection. Otherwise three projections as explained in the following will be used (see fig. 3).

Using the projection onto the yz - plane the face F_1 appears as a line, face F_2 still appears as a face. The resulting intersection of line and face are one or more intervalls along the y-axis. Inside these intervalls an intersection is possible but not necessary.

The projection onto the xz - plane results in a single x-value, the value where both lines representing faces intersect. This point is also only a prerequisite to calculate the intersecting lines.

In the last projection onto the xy - plane lines are drawn parallel to the y - axis along the previously determined intervalls. The intersection between these line segments and the second face F_2 results in the intersecting line between both faces. This calculation can be performed by 2D operations. In the case of Smallworld GIS the class `sector_rope` and its method `all_intersections(other_sector_rope)` may be used.

4 IMPLEMENTATION IN SMALLWORLD GIS

[1]

Smallworld GIS provides an outstanding programming environment not only for application development but also for system development. The extension by 3D functionality is facilitated by

- the usage of only one language for system and application development
- the creation of more specialized subclasses in the object-oriented framework
- the availability of the source code for all but the kernel classes
- the interactive development environment (e.g. class browser)

The library installed with the GIS uses the X11 windows system, a proprietary relational database and classes for collections (i.e. a collection of objects like vectors with fixed size, stretchable vectors, hash tables, queues and sorted collections).

[1]Since Version 2.1.2 it is possible to use a z - coordinate in Smallworld GIS. For its implementation some new classes have to be introduced. Genuine 3D functionality is only realized on the level of the 3D coordinates, i.e. its not possible to model body-shaped objects.

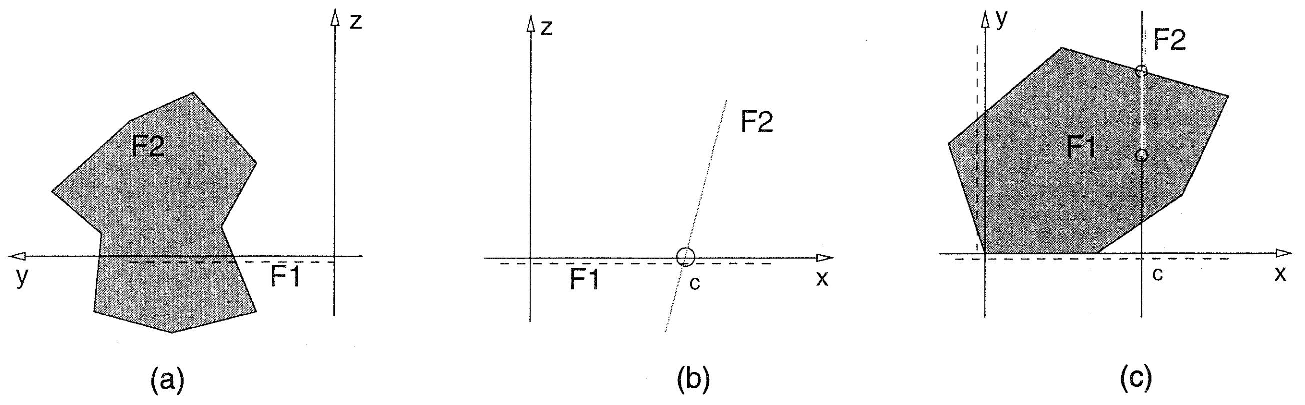

Figure 3: Three projections of the transformed areas

4.1 3D extension in Smallworld GIS

Some extensions can be performed by modification of the original source code. For example the class **bounding_box** serves as a prototype of the 3D bounding box. In many cases only minor modifications are necessary especially when classes are dealing with two-dimensional coordinates. Problems only occur if optimizations on 2D methods were necessary. The so-called primitives are implemented in C code and are therefore not accessible for modification. The solution is only a total rewriting of these methods.

4.2 3D data model

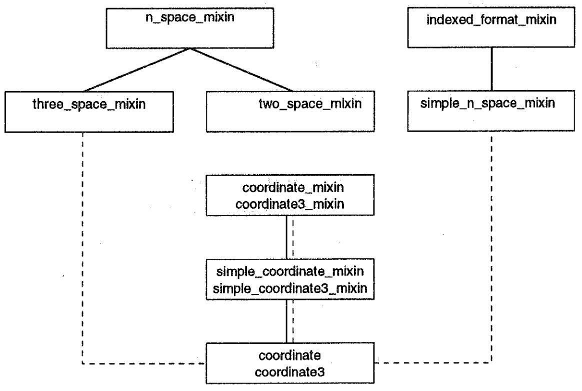

Figure 4: Hierarchy of Coordinate Classes

The class **coordinate** is ancestor of the classes **simple_coordinate_mixin**, **simple_n_space_mixin** and **three_space_mixin**. Only three classes need to be changed to cope with 3D data. In Fig. 4 each rectangle corresponds to a class in the 2D (upper half) or 3D data model (lower half, not set if equal to 2D data model). Inheritance relations of the 3D data model are drawn dotted, relations valid for both models are drawn bold.

In addition to the extension of the class **coordinate** the database type **ds_coord** can also be equipped to handle the additional z-value. As in the two-dimensional case a coercion i.e. a conversion between both types is also realized.

On a higher level lists of coordinates (**sector**, **sector_rope**) are used to deal with lines resp. **links**. These classes are the heart of the GIS. The essential geometric operations are implemented as methods of these classes. Therefore fundamental changes to the source code have to be performed before their methods can be used as new 3D methods (examples are the calculation of distance, angles, bounding box). As long as there are no two-dimensional coordinates explicitly mentioned, there is no need for a modification.

There are enough conceptual similiarities between the Smallworld 2D data model and the 3D data model to borrow some of the already existing classes as well as methods of the object-oriented implementation of this GIS-product. The geometric attributes execept for body-shaped attributes can be redesigned from their 2D counterparts. The class of the edges is called **links** in the two-dimensional model. Line-shaped attributes are **chains**, faces are **polygons**, face-shaped attributes are called **areas**.

Special methods are available to construct tables and procedures to deal with the 2D data model. They can also be used for the construction of the 3D data model.

4.3 3D operators

The transformation described in the previous section is realized by methods of the classes **sector3_rope** and **matrix_2**. The matrix is necessary to transform all involved geometric entities with the transformation matrix to the new coordinate system.

It is possible to define new predicates, spatial and others by adding new methods to the class called **predicate**. The test is performed by a procedure appended to the predicate.

The definition of the new predicates should be based on the sound formal description of [van Oosterom, Vertegaal, van Hekken and Vijlbrief, 1994]. They have proved that it is sufficient to have only five predicates that are capable of describing all relationships between point/line/area. Their approach is an extension of the point set approach of [Egenhofer and Franzosa, 1991].

The defined predicates are:

- The relationship *touch* applies if two geometric elements have only some part of their boundary in common. This relationship is symetric and does not apply to the point/point situation.

- The relationship *in* applies if the former element is completely contained in the latter one. This relationship is transitiv and applies to all geometric elements.
- The relationship *cross* applies for the line/line, line/area as well as for the line/body, area/body situation.
- The relationship *overlap* applies if the result of the intersection is of the same dimension as the two participating elements.
- The relationship *disjoint* expresses that no common set exists.

In [Bähr, 1992] as well as in [Gapp, 1994] more predicates are defined based on projective relations like *above* or *behind*. Predicates of distance (e.g. *near*)or of order (e.g. *between*) are also possible to implement.

The 2D predicates are realized mainly by the methods `surrounds??()` and `segpoint_near??()`. Using coordinate transforms and projections this methods can be used for the computation of most of the 3D predicates.

Apart from predicates there are also object generating operators (e.g. the results of intersections) and measurement functions (e.g. the distance function). The distance will be calculated at the level of and as a method of coordinates (`distance_to(another)` resp. `distance_to_line(a_line)`).

4.4 Representation in 3D

To display 3D objects considerable more complex algorithms have to be used than those which are necessary to display two-dimensional data. A 3D graphic program like *Geomview* can be applied to minimize the costs of the implementation and maximize the abilities of the conceive system. Changes of viewpoints, rotation, scaling and selection of geoobjects is thereby feasible without much effort. In the case of using Geomview communication is possible in both direction.

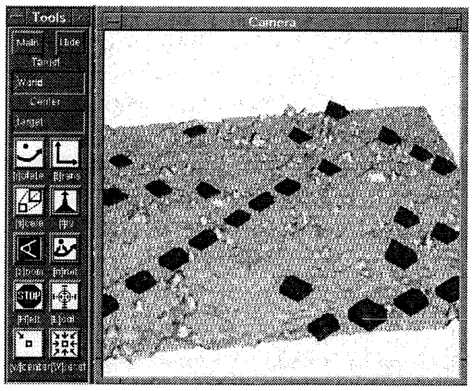

Figure 5: Display of the landscape by Geomview

5 Summary

The three steps to a successful 3D implementation in an existing 2.5D GIS are

- an additional data model for 3D data,
- methods to convert 3D data into 2D data to apply 2D operations and
- an external interactive 3D viewer for displaying 3D entities.

Most GIS products will not allow extensions of that kind. With Smallworld GIS we took advantage of the open object-oriented interactive environment to try a possible implementation. The algorithm to transform all geometric objects has already been implemented. Also methods have been implemented to control *Geomview* and receive in turn information about selected geometry.

Further work has to be done to implement the complete data model to test its fitness and implement some more operations.

References

Bähr, U. [1992], Untersuchungen zu räumlichen Abfragesprachen, Diplomarbeit, TU München.

Egenhofer, M. J. and Franzosa, R. D. [1991], 'Point-set topological spatial relations', *International Journal of Geographical Information Systems* **5**(2), 161–174.

Gapp, K.-P. [1994], A computational model of the basic meanings of graded composite spatial relations in 3d space, *in* Molenaar and de Hoop [1994], pp. 66–79.

Molenaar, M. and de Hoop, S., eds [1994], *Advanced Geographic Data Modelling*, number 40 *in* 'Publications on Geodesy', Netherlands Geodetic Commision.

Rumbaugh, J., Plaha, M., Premerlani, W., Eddy, F. and Lorensen, W. [1991], *Objektorientiertes Modellieren und Entwerfen*, Prentice-Hall International and Verlag Carl Hanser.

van Oosterom, P., Vertegaal, W., van Hekken, M. and Vijlbrief, T. [1994], Integrated 3d modelling within a gis, *in* Molenaar and de Hoop [1994], pp. 80–95.