'Photogrammetric Week '95', D. Fritsch & D. Hobbie, Eds., Wichmann Verlag, Heidelberg, 1995



# The Object-oriented DTM in GIS

**DIETER FRITSCH and DIETER SCHMIDT, Stuttgart**

**ABSTRACT**

The integration of digital terrain models (DTM) in geographic information systems (GIS) implies automatically an extension of the GIS reference surface and its queryspace. It is trivial that a DTM is the natural boundary representation of theearth's surface. Man-made objects, for instance houses, bridges, dams should be considered in a second step because these objects cannot be represented well by boundary surfaces - they have to be approximated by solid modelling techniques like boundary representations or primitive instancing. Therefore, an efficient DTM integration in GIS is the first task to be solved.
In a first prototype developed in the object-oriented language Python the main features of a 3D GIS will be discussed. Based on this query space spatial operators are derived which are implemented as methods of objects or external functions.

## 1. INTRODUCTION

In the past digital terrain models (DTM) and geographic information systems (GIS) were developed isolated from each other. Both tools exist for about 30 years. The applications of today require more and more the three-dimensional reference surface for GIS, although temporary aspects demand for four dimensions including time as additional reference parameter. While the development of geographic information systems advances by using object-oriented programming (OOP) techniques, DTM research has so far been neglected. Therefore, it is obvious to merge DTM research and development with the efforts which take place within the discipline of GIS.
At present most GIS products and also authoritative GISs use a flatsurface (planimetry) as spatial reference - only few go one step ahead and consider DTM as additional (separate) data layer. Thus, it is worthwhile to merge 2D planimetric data with 1D topographic data in an efficient manner using 2D topological elements. The output is a 2.5D geometric dimension for GIS because the geometric reference is now the boundary representation of the natural relief.
In the second step of our approach we link real 3D objects, for instances houses and bridges to evolve GIS geometry to a more realistic view of the real world. Therefore, 3D geographic information systems emerge from 2D via 2.5D to 3D.
In developing a 3D GIS the most pressing problem was to come to a first proto-type in no time. Therefore a high level object-oriented language was selected for implementing it. The language Python developed at the Stichting Mathematisch Centrum in Amsterdam (The Netherlands) by Guido van Rossum turned out to be one of the most versatile and extensible languages.
Another problem in developing 3D applications is to display the emerging 3D objects. Intensive search produced the program Geomview from the Geometry Center, Minneapolis, as a very suitable display engine. The most important feature is its responsivness to mouse clicks to select objects and sending coordinates back to the program. Dynamic update of the display is also possible.
Some terms used throughout the rest of the article will be explained here. A *geoobject* (geographic object) is a object representing a real world object like a parcel, a street, a house or a bridge. Every geoobject has besides its conventional attribute types like string or number the attribute types point, line, area or body. These are called *geoattributes* or in another context *geometric objects* resp. *geometric classes*. They inturn are composed of the *geoprimitives* nodes, links i.e. line segments, and faces.

'Photogrammetric Week '95', D. Fritsch & D. Hobbie, Eds., Wichmann Verlag, Heidelberg, 1995



## 2. FROM 2D TO 3D VIA 2.5D

Most applications nowadays use 2D GIS for display, storage and analyses. But many applications require a model of the surface to calculate the height or the slope at certain points. Many GISs have additional terrain modellers to make use of height data in form of a regular grid or a triangulated irregular network (TIN). But many kinds of analyses are impossible or difficult to perform because the integration is only a partial one. The default user interface must be extended to take advantage of the new possible queries e.g. concerning height or slope.
To add a digital terrain model to a GIS results only in a 2.5D GIS because it is not possible to model a solid. A simple and convenient way is to add descriptions of man-made objects like buildings to planimetric objects and attaching them to the surface. In CAD applications different solid models are available. 3D GISs may thereby provide a more realistic representation of the modeled world. Some draw-backs are its more expensive data management and the problem of data acquisition Normally you can only expect to extract the (average) height of buildings and not an exact reconstruction of the building. Haala (1995) describes an automatic system to recognize houses and extract boundary representations from stereo images using range information. 3D models have to be displayed, used in analyses and stored in an efficient way. They will usually be created by CAD programs. This already limits the available representation of solids to be used in GISs. The *boundary representation* (b-rep) is probably the most widespread 3D model and has already many algorithms available to compute physical properties. Different kinds of *b-reps* have been developed to suit different object properties (e.g. curved surfaces) , see Foley, van Dam, Feiner and Hughes (1990). The simplest model concerning storage and display is called *primitive instancing*. Most houses can be described by using parameters like slope of the roof, height of the house, width and length. To display the house a specialized function uses the parameters as arguments. *Spatial occupancy enumeration* (SOE) uses spatial cells of fixed size and is therefore raster oriented. *Octrees* are hierarchical variants of SOEs derived from quadtrees. The *sweep representation* describes an object by 'sweeping' another object (normally a 2D area) along a trajectory through space. This can also be used as an approximation in describing buildings with a ground plane and a known height. *Cell decomposition* uses a parameterized set of primitive cells 'glued' together to build complex objects (e.g. think of row houses). In *constructive solid geometry* (CSG) simple primitives are combined by means of regularized Boolean set operators.
The description must be unique, i.e. two descriptions describing the same solid must be equal. An example of a query requiring that characteristic is: "Show me all houses equal to the one I'm pointing at!" Boundary representations are not unique because a face can be described by many smaller faces. Only spatial occupancy enumerations and octrees guarantee the uniqueness.
The domain of representation should be large enough to allow a useful set of physical objects to be represented. Many solid modellers can only approximate smooth curves by many short straight lines. And it also affects the compactness of the representation. This is of minor importance because man-made objects seldom have smooth curves. Primitive instancing can only represent objects for which it has a defining function. Spatial-partitioning and polygonal boundary representation produce only approximations of solids. A higher resolution requires more processing time as well as storage space and may be therefore not practical.
The creation of an invalid representation should be impossible. By using boundary representation faces do not automatically form the boundary of a solid. On the other hand, a valid representation should be easily created.
The characteristic to maintain closure under operation like translation or rotation is also important.
Why not extend an already functioning GIS? It is really not trivial to add one dimension to planimetric GISs. Existing systems like the object-oriented Small-world GIS have only additional tables to store DTMs but no real integration of the third dimension i.e. no 3D spatial queries or interactions between

the surface and the planimetry are realized. That's why an extension requires extensive changes in internals of the system. This is difficult even in object-oriented systems with the complete source code because a host of existing methods must be adapted or rewritten. This includes database tables, display routines, import modules, query processing and so on.

## 3. DATA MODEL

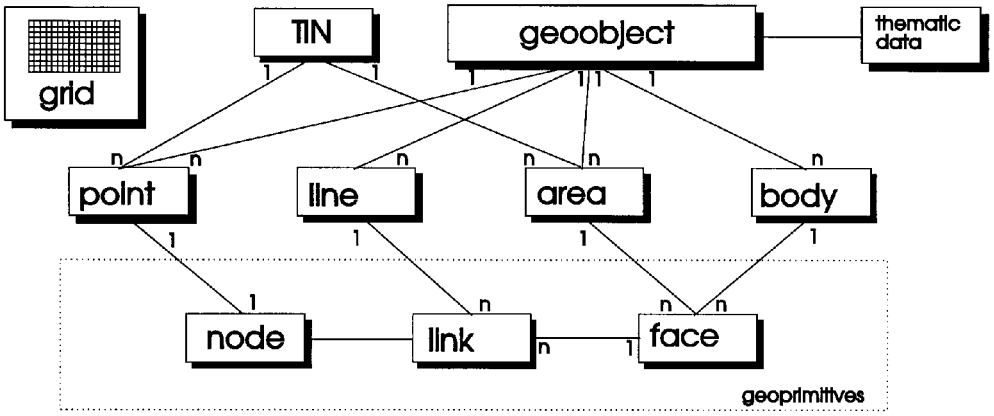


Figure 1: 3D data model.

The described data model is similar to the data model used in Smallworld GIS except for the DTM classes. TINs and Grids are handled in Smallworld GIS separately.
The used data model has three different layers. The bottom layer consists of the classes *node*, *link* (a link is a line segment that can be part of many lines) and *face*.
An object of the class *node* stores its coordinates in a tuple of size three. If the z-value is unknown it is set to `None` meaning undefined. The height may be calculated from the underlying TIN or Grid. A *link* stores pointers to start and end node as attribute and also coordinates of all intermediate nodes. A *face* is composed of *links*. The orientation of some links may be reversed to have one direction (clockwise seen from the inside of the body) per face to distinguish between outer or inner surface. There is also a method checking for the planarity of *faces*.
The middle layer consists of the classes *point*, *line*, *area* and *body*. *Point* stores a pointer to a node. It is also used to describe a surface point of TINs. A *line* stores a pointer to one or more links. The *links* must be connected. *Area* and *body* have pointers to one or more *faces*. The differences are that faces composing a body must be planar (has to be checked), that each *link* composing a *face* must be shared by exactly two *faces* and that at least three *links* must meet at each *node*. Also *faces* must not interpenetrate.
The top layer consists of several user-defined object classes inheriting several methods from the class *geoobject*. Every class adds its special attributes to the class data structure. These may be the conventional attribute types like number or string as well as the geoattributes *point*, *line*, *area* or *body*. It is possible for a *geoobject* to have several geoattributes. This is e.g. useful when displaying a *geoobject* at different scales.
This data model is not designated to handle *lines* or *areas* not situated on the surface. This would require to store topological relations between these geometric objects and solids. This case is seldom required.

### 3.1 Topological classes

An interesting feature in Smallworld GIS is called 'manifold' that has been adopted for the current system. The best translation may be 'topological class'. It describes the interaction between geometric

'Photogrammetric Week '95', D. Fritsch & D. Hobbie, Eds., Wichmann Verlag, Heidelberg, 1995



objects in terms of rules. An example may be the crossing of two lines belonging to two roads. To perform network analyses both lines must be connected. This is realized by inserting a node and splitting both lines in two line segments. Other rules may involve connecting areas and line or areas of different geoobject classes with each other. The methods or rules applied to members of a topological class are only used while inserting a geoobject.
On the other side many objects are topologically independent e.g. the border of a district can be changed without moving an adjacent road. Moving a node or a link can therefore mean the creation of a new node or link without destroying the old one. A reference counter has to be kept.
This interaction is useful for network analysis or movement of objects. E.g. aquery for shortest path uses only roads and no borders of parcels. Another example is the movement of roads where the intersection is also moved.
Interaction of the terrain with geographic objects like houses or rivers can be handled in the same way. By using a TIN surface patches for building solid must be left open. The surface patches that are part of the boundary representation are also part of the TIN. I.e. links composing faces of the body are also part of triangles of the TIN.

## 4. IMPLEMENTATION

A first prototype has been designed in the language *Python*. It is an interpreted, interactive, object-oriented programming language designed for rapid prototyping.But it is also modular to develop larger programmes. This modularity includes not only Python programmes but also other C libraries (e.g. window systems). Finally Python is portable between different platforms. Programmes developed on UNIX systems can also run on MacOS or DOS.
The classes of the hierarchical layers described above have been put into separate modules. Coordinates are stored in tupels of size three, line segments are stored in variable-sized arrays.
The geometric attributes points, lines and areas - implemented as object classes- use pointer to its belonging geometric objects and bounding boxes to store an approximation of its extent.
The first implementation of solids used primitive instancing as representation. The advantage is that only numerical or boolean attributes need to be stored along with the geoobjects. Every object has its own display routine converting the parameters in a description understood by Geomview.
The next step was to model solids by an additional geometric attribute class using boundary representation providing more flexibility. Solids are modeled by the class *body*. It stores pointers to planar *faces*. The *faces* making connection to the ground can in turn be used as *areas* in a planimetric view.

### 4.1 Measurement functions and predicates

The extended query space makes three kinds of spatial results possible. The functional operators like distance or height are implemented as methods of the geometric attributes point, line, area and body. If the result is not clear a tuple consisting of minimum, maximum and average value is returned. This is marked in the following table with the number 3 meaning a tuple of three values.
Topological predicates give only a truth value. They are realised as separate functions dealing with two

| Method | point | line | area | body | description |
|---|---|---|---|---|---|
| distance | 1 | 3 | 3 | 3 | distance between objects |
| height | 1 | 3 | 3 | 3 | average z-value |
| perimeter | - | 3 | 3 | - | perimeter resp. length |
| gradient | - | 3 | 3 | - | slope |
| area | - | - | 1 | 1 | (surface) area |
| volume | - | - | - | 1 | |

geometric objects (like point is inside area). The following predicates are valid for 3D objects. **P**,**L**,**A**,**B** mean *point*, *line*, *area* and *body* respectively.

| Topol. rel. | PP | PL | PA | PB | LL | LA | LB | AA | AB | BB |
|---|---|---|---|---|---|---|---|---|---|---|
| equal | x | | | | x | | | x | | x |
| touch | | x | x | x | x | x | x | x | x | x |
| inside | x | x | x | x | x | x | x | x | x | x |
| cross | | | | | x | x | x | x | x | |
| overlap | | | | | | x | | x | | x |
| disjoint | x | x | x | x | x | x | x | x | x | x |

Several other functions have been put in a separate modul using the internal data structures of geometric objects i.e. tupel for coordinates or vectors of tupels for lines and areas. They can be used independently of the geometric classes.
The display of 3D data takes place with the program Geomview. It is a interactive program for viewing and manipulating geometric objects. It allows interactive control over the point of view, speed of movement, appearance of surfaces and lines. Displayed objects can be controlled independently.

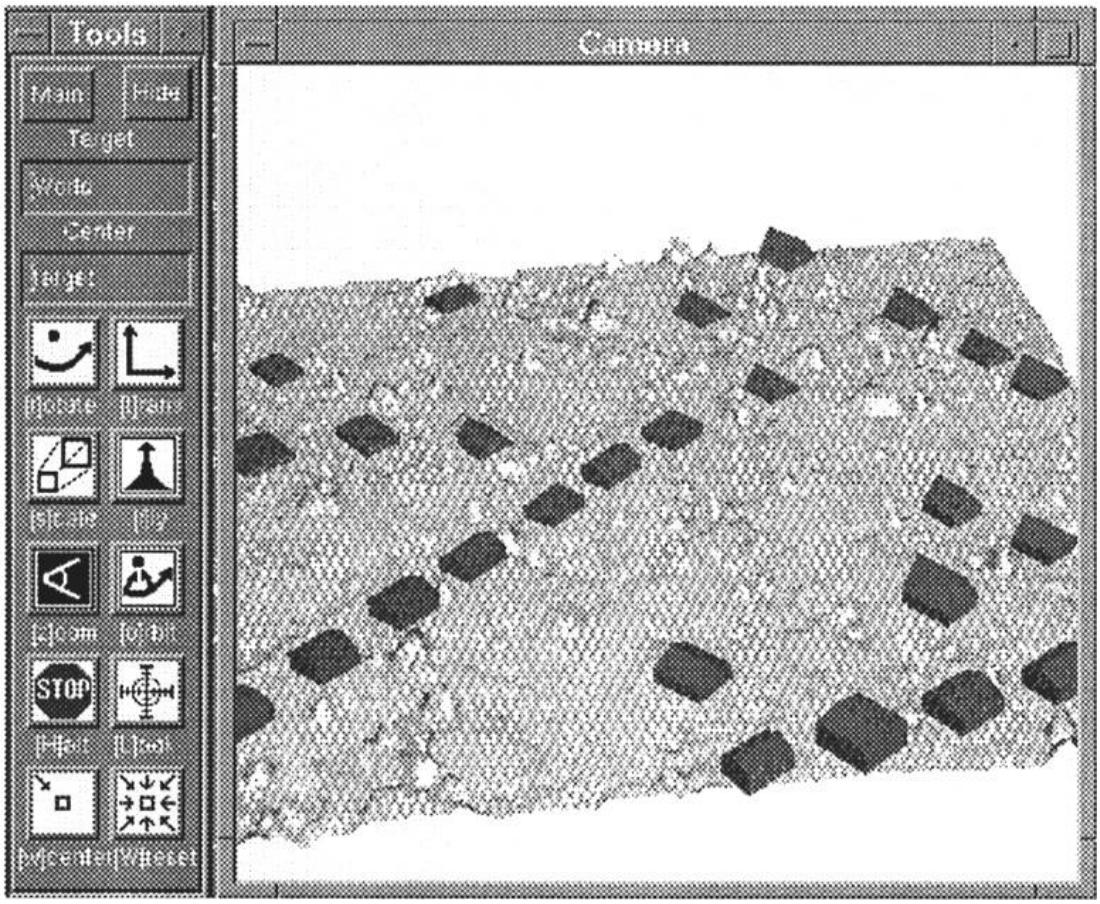


Figure 2: Geomview control panel with grid DTM and attached houses in camera view.

Geomview is used as display engine for the spatial data. Geometric description will e sent object by object with an internal name used to reference the geometry. This is useful in conjunction with the capability of picking objects or points on faces or edges. The coordinates will then be send back to the main program together with the name of the selected object. This way points or objects can be selected for processing in more complex queries. It is also possible to dynamically change the geometry, color etc. of the object or add other objects.

'Photogrammetric Week '95', D. Fritsch & D. Hobbie, Eds., Wichmann Verlag, Heidelberg, 1995



## 5. CONCLUSION AND FURTHER WORK

3D GISs are already feasible. Object-oriented techniques simplify the development and later the extension of spatial data types and spatial analyses. The prototyp built already handles almost all spatial functions and many spatial predicates. Some problems occuring in attaching different 3D models to a surface, can be overcome by other representations in conjunction with a topological model exploiting the TIN structure.
Further work must be done in testing the prototype against a specific application. This may show the fitness of the system for practical purposes. For this task special complex analyses functions dealing with the data model will be required.
Some applications that requires DTM as well as 3D solids are the spreading of radio signals, urban planning or military applications. E.g. an application examining the intensity of radio signals may yield areas of radio shadow and possibly the position of further senders (Kidner, Jones, Knight and Smith, 1990).

## 6. REFERENCES

Foley, J. D., van Dam, A., Feiner, S. K. and Hughes, J. F. (1990), Computer Graphics- Principles and Practice, Addison-Wesley.

Fritsch, D. (1990), Towards three-dimensional data structures in GIS, in EGIS '90, Amsterdam, pp. 335-345.

Fritsch, D. and Schmidt, D. (1994), DTM integration and three-dimensional query spaces in geographic information systems, in H. Ebner, C.Heipke and K.Eder, eds, Spatial Information from Digital Photogrammetry and Computer Vision, Vol. I, Int. Arch. Photogr. and Remote Sensing, pp. 235-242.

Haala, N. (1995), Building recognition by combination of image data and height data, in Proceedings of the 3rd International Workshop "High Precision Navigation", Dümmler Verlag, Bonn.

Kidner, D. B., Jones, C. B., Knight, D. G. and Smith, D. H. (1990), Digital terrain models for radio path profiles, in K. Brassel and H. Kishimoto, eds, Proceedings of the 4th International Symposium on Spatial Data Handling, Vol. I, pp. 240-249.

Molenaar, M. (1992), A topology for 3D vector maps, ITC Journal 1, 25-33. Smallworld (1994), Smallworld GIS -Customisation Course Workbook, Smallworld Systems, Cambridge.