

Research Article

An object-oriented data model for complex objects in three-dimensional geographical information systems

WENZHONG SHI¹, BISHENG YANG^{1,2} and QINGQUAN LI²

¹Advanced Research Centre for Spatial Information Technology, Department of Land Surveying and Geo-Informatics, The Hong Kong Polytechnic University, Hong Kong

e-mail: lswzshi@polyu.edu.hk

²National Laboratory for Information Engineering in Surveying, Mapping and Remote Sensing, Wuhan University, Wuhan, Hubei, China 430079

(Received 22 April 2002; accepted 2 September 2002)

Abstract. Developing a three-dimensional (3D) data model for Geographic Information Systems (GIS) is an essential and complex issue. 3D modelling in GIS is becoming ever more important for the development of cyber cities and digital earth, which have recently become feasible. A competent 3D model forms an efficient foundation for 3D visualization, query and spatial analysis. As a development of the existing 3D models, this study proposes particular improvements in handling complex 3D objects. We present an object-oriented data model for handling complex 3D objects in GIS. First, the conceptual data model is developed based on the principle of object-oriented (OO) data modelling. This model is designed based on the following three basic geometric elements: node, segment and triangle. Accordingly, the abstract geometric objects are defined: including points, lines, surfaces and volumes. Second, the corresponding 3D logical model is designed based on the defined abstract objects and the relationships between them. Third, a formal representation of the 3D spatial objects is described in detail. Fourth, a prototype 3D GIS is developed based on the proposed 3D data model. Finally, we describe the results of an experimental study to reconstruct 3D objects using this 3D GIS and a comparison with the performance of other 3D data models. The proposed model is able to handle complex objects, such as complex buildings and TV towers, which is an essential functionality for building large-scale cyber cities, such as for Hong Kong. The proposed data model proves to be very efficient, particularly in visualization and rendering. The experimental results show which the data volume of the proposed model is compacted and the visualization speed for 3D objects is improved, compared with the existing models.

1. Introduction

GIS were originally developed by transferring and representing 3D real world entities as two-dimensional (2D) objects, as points, lines and polygons in either a vector or raster data structure. To represent natural terrain on the Earth's surface efficiently, several types of terrain model or digital elevation models were developed and adopted. Examples of these are the triangulated irregular network (TIN) and Grid. These are considered to be two and a half-dimensional (2.5D) models in GIS.

2D and 2.5D models cannot fully represent the 3D real world, particularly when we are interested in the detailed description of the internal structure of a 3D spatial entity. 3D GIS have therefore been studied during the last decade (Molenaar 1992, 1998, Raper and Kelk 1991, 1993, Pilouk *et al.* 1994, Pilouk 1996, Gruen and Wong 1998, Tempfli 1998, Zlatanova 2000), and GIS is being developed towards dynamic and multi-resolution (Bergougnoux 2000). The demands placed on 3D GIS have increased even further, especially in the recent developments of the digital Earth and cyber city, such as cyber Hong Kong.

There has been some progress in the development of 3D GIS models, such as 3D visualization (Kofler *et al.* 1996, Koninger and Bartel 1998, Yoshida, 1998; Yang and Li 2000, Huang *et al.* 2001), a 3D topology (Arnaud and Bernard 1999, Losa and Cervelle 1999, Sun *et al.* 1999, Pfund 2001), 3D analysis and query (ESRI 1997), 3D GIS user interface (Verbree *et al.* 1999), Web 3D GIS (Zlatanova and Gruber 1998, Coors 2001) and others (Raper *et al.* 1993, Kraak *et al.* 1998). However, a number of issues remain open and require further study of 3D GIS models. These may include, for example, handling complex 3D objects, improving the efficacy of 3D models in terms of speed and data volume, modelling 3D topology, visualization and others. However, there are still many barriers to be overcome in visualization (Gahegan, 1999). The specifications of the OpenGIS Consortium (OpenGIS 2002) have been proposed for the development. In this study, we will focus on improving the performance of 3D models with regard to the first two aspects: handling complex 3D objects and improving the efficiency of 3D models.

Here, we classify the 2.5D and 3D models in GIS and CAD to the following four categories: volume-based models, surfaced-based models, hybrid models and vector-based models.

The smallest unit of a volume-based model for 3D objects is a voxel. A 3D object can be represented as a composition of a set of voxels. The volume-based models may include, for example, Octree, 3D array, the needle model and Construction Solid Geometry (CSG) (Samet 1990). In a surface-based model, objects are represented by the composition of facets. Examples of surface-based models include boundary representation (BR), the shape model, Grid, and others (Li 1994). Surface-based models are mainly used to depict the geometric characteristics of objects by macro surfaces or surface primitives (Li 1994).

In computer graphics (CG) and CAD, both CSG and BR models are also widely used. In the CSG model, operations between objects include Boolean operation and transformation. Data storage for this type of model is usually smart. However, the CSG model may be insufficient for representing the topological relationships and attribute information required in 3D GIS.

The hybrid model is an integration of a volume-based and a surface-based model. Examples of hybrid models include the integrated CSG and Octree model (Li 1994), the integrated TIN and Octree model (Shi 1996), and the integrated TEN (Tetrahedron) and Octree model (Li and Li 1996).

The vector-based 3D models that have been developed in the area of GIS. This type of model may include, for example, the formal data structure (FDS) model (Molenaar 1992) the TEN (Tetrahedron-based) model (Pilouk 1996), and the simplified spatial model (SSM) (Zlatanova 2000).

Our research is along the lines of vector-based models. We first give a further detailed review of vector-based models, including 3D FDS, TEN and SSM, and then elaborate our development on the basis of the characteristics of these models.

Molennar (1992) proposed the FDS model for the 3D objects in GIS and Tempfli (1998), further extended the 3D FDS model to integrate raster data for photo-texture, Rijkers *et al.* (1993) implemented the 3D FDS model within a relational data model. Gruen and Wang (1998) further modified the 3D FDS model and developed a system (CC-GIS) which they implemented as the V3D model. There are four basic geometric elements in the 3D FDS model: node, edge, arc and face. Spatial objects are divided into four types, namely point, line, surface and volume. Figure 1 demonstrates the concept of the 3D FDS model. The model is able to describe the spatial topological relationships among elements. Here, an edge is composed of two nodes, arcs constitute line objects and boundaries of faces. Moreover, in order to avoid the ambiguity in 3D FDS, arcs that constitute the boundary of a face have to be oriented. Furthermore, a face can be part of two bodies at most. This model can be easily used to re-construct 3D objects, such as buildings. Further, the topological relationships among the spatial objects can be retained. One of the limitations of the model is that one face has to be linked with two bodies. This might limit the use of the model for efficiently constructing complex objects, such as complex buildings like the one in figure 2, which is a building with a floor in the middle. For representation of this single object (the building) in the 3D FDS model, it has to be subdivided into two objects—the upper and lower parts of the building. This

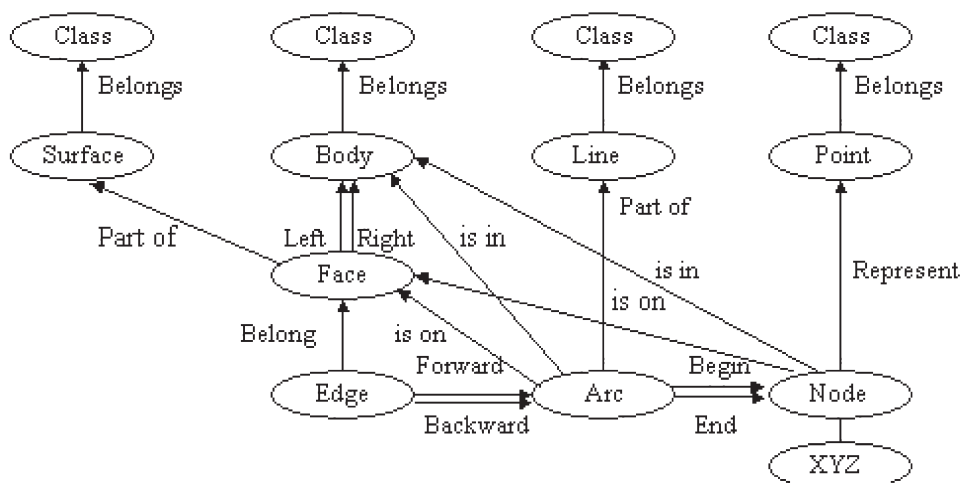


Figure 1. The concept model of 3D FDS (from Molennar 1992).

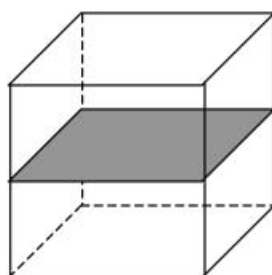


Figure 2. One building (one single object) with a floor in the middle has to be subdivided into two objects in the 3D FDS model).

inevitably leads to an increase in the data volume for storing the spatial objects and the corresponding topological relationships among them.

As illustrated in figure 3, the basic elements (the simplexes) of the TEN model (Pilouk, 1996) include node, arc, triangle and tetrahedron. A feature (a complex) in a GIS can be composed of a set of simplexes. Here, a tetrahedron, which is constructed by four triangles, is used to form spatial objects. Figure 4 shows an example of spatial object that is divided into a series of adjacent tetrahedra. The description of the spatial relationships among the points, edges, triangles and tetrahedra are provided and the geometric locations of the spatial objects are also described by their coordinates. The TEN model can precisely describe a complex spatial object.

The most important part in constructing a spatial object in the TEN model is to decompose the object into a set of the composed tetrahedra. Theoretically, it is possible to subdivide any spatial object into corresponding tetrahedra. However, the algorithms for constructing a constrained tetrahedron are still under research. Another problem is that many inner triangles will be created when we construct

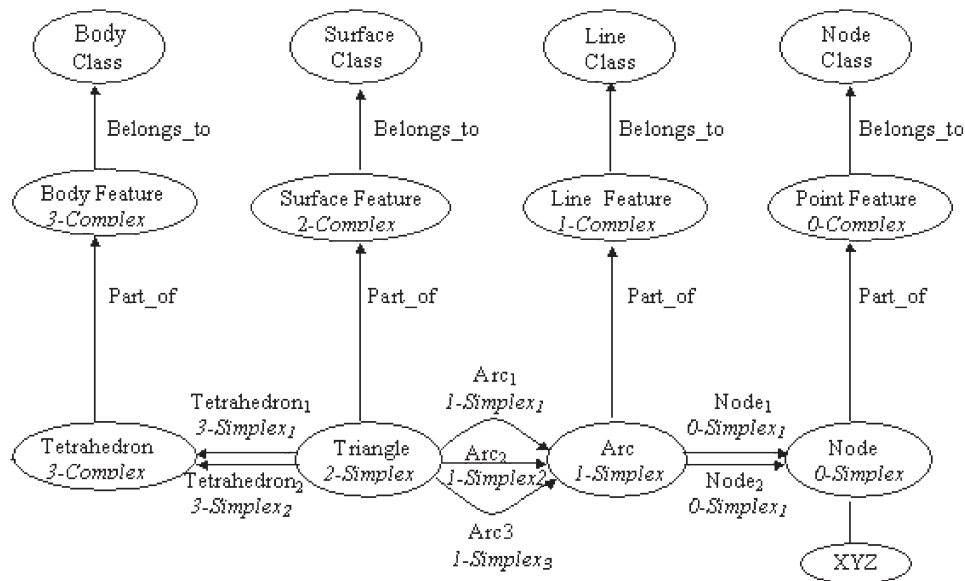


Figure 3. The concept model of TEN (after Pilouk 1996).

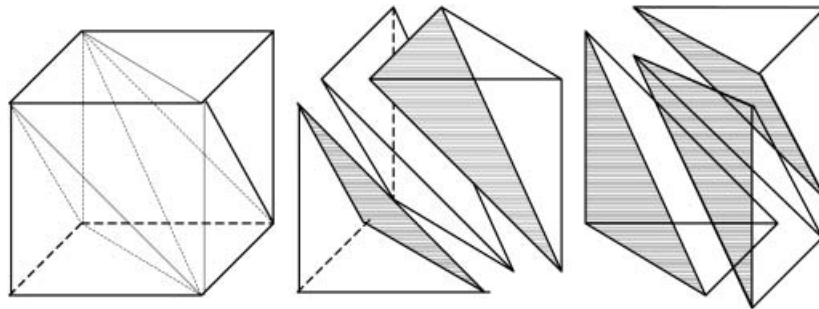


Figure 4. Subdivision of a cube by tetrahedrons.

tetrahedrons. This might be useful for the inside of irregular objects such as ores, but they are unnecessary for regular objects, such as buildings. For instance, in figure 4, the shaded triangles are invisible and have to be omitted in rendering. Otherwise, they will affect the visualization effect of the scene and also reduce the processing speed for visualization.

The SSM (Zlatanova 2000) is a further development of the 3D FDS model. The basic elements of the model are the face and node. The model defines four abstract geometric objects, namely point, line, surface and body. A simplified spatial scheme (SSS) is designed and the semantic model of the SSS is demonstrated in figure 5. Compared with the 3D FDS model, the SSM keeps the explicit relationships between body and face and eliminates the arc object. On the other hand, the SSM keeps the relationship of geometric objects and attribute data such as texture, which is very useful for the visualization of spatial objects. The SSS model defines the strict relationship of four abstract geometric objects, and only permits the existence of convex faces. If there are line(s) or point(s) on a face, the face should be divided into several sub-faces. This may add more topological relationships among faces and surfaces.

Based on the above review of the existing 3D models, it can be seen that many of the existing vector-based 3D models are very valuable in certain GIS application problems. For instance TEN model supports computation and query, which can be applied in geologic fields. Some of the models are efficient in visualization and query,

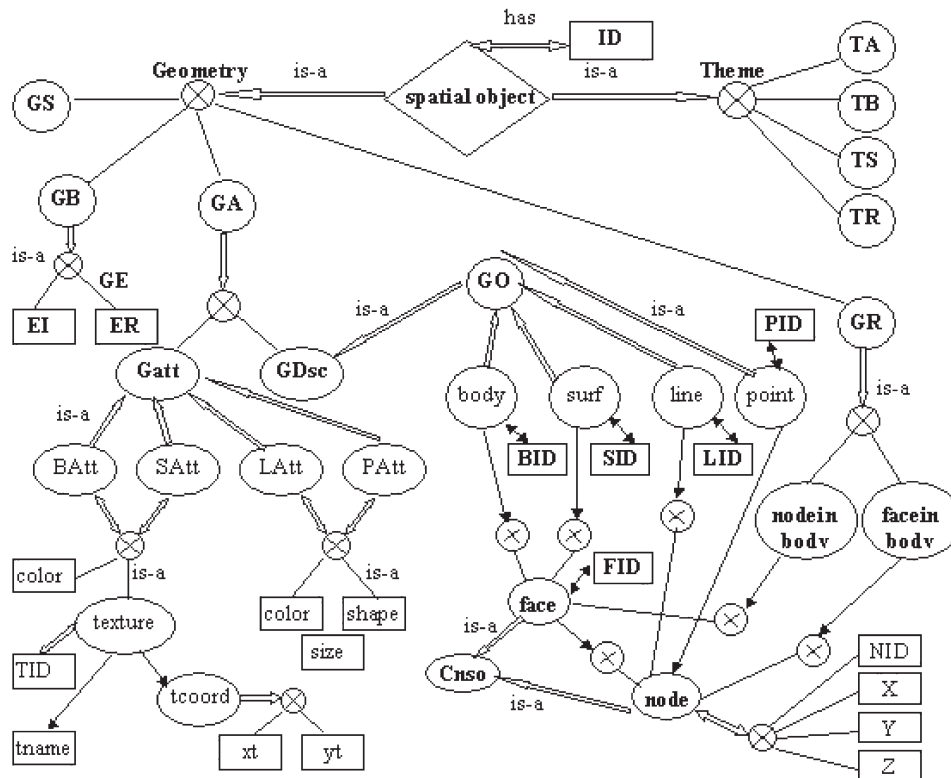


Figure 5. Semantic model of SSS (after Zlatanova 2000).

for example 3D FDS and SSM, but more explicit topological relationships are stored for them. It is inevitable that the speed of visualization will be affected due to the cost of storing these relationships. Zlatanova (2000) compared the performance between 3D FDS and SSM, the results showed that SSM is better than 3D FDS, because SSM eliminates many topological relationships in 3D FDS. It is evident that many functions, such as 3D visualization, 3D reconstruction and graphic editing, are frequently required in 3D GIS applications. The speed of visualization and query is therefore of vital importance and has to be considered. In many 3D GIS applications, such as cyber cities and the digital Earth, there are a large number of complex spatial objects such as complex buildings and others and the topological relationships are even more complicated. Thus, the cost of storing the topological relationships for complex spatial objects is even higher. In this study we develop a new vector-based 3D model, an object-oriented 3D GIS model (OO3D), as a further development to the existing 3D models, particularly intended to improve their capability and efficiency in handling complex 3D objects. The OO modelling approach is applied and basic elements for the 3D vector model are redesigned to create the proposed 3D vector model.

The remainder of this paper is organized as follows. In Section 2, the concept and logical models of the OO3D data model are described in detail. In Section 3 the formal representation of 3D spatial objects based on the OO3D data model is given. In Section 4 a case study and the results of comparing the model with other data models are presented. Section 5 presents the conclusions and discussions.

2. Object-oriented three-dimensional GIS model

2.1. Object-Oriented modelling

OO techniques (Martin 1993) have been widely used in software engineering because of their ability to provide a simplified design of complex systems. OO techniques include OO modelling, OO database management systems and OO programming. In this study, we apply the OO modelling approach to the design of a 3D GIS model. The advantage of OO techniques is the mechanisms of reusing the software modules for different requirements without major modifications (Martin 1993). The OO approach can abstract an object based on the object type, object relationship, object attribute, operation methods and constraint sets. In OO modelling, an object has the following characteristics: classification, abstraction, inheritance, encapsulation, aggregation, association and polymorphism. An object includes both data and method. The operation on an object can only be acquired by the methods of the object. In OO modelling, an object is abstract and without actual meaning until a particular attribute is assigned to it. However an abstract object may be associated with many instances which represent objects in the real world. For example, the instances of an abstract object can be a specific building or road existing in the real world. Another important characteristics of OO modelling is the relationship between objects. This provides a mechanism which permits relationships of inheritance, instance and aggregation between different objects. The design procedure of OO can be divided into two steps: the analysis of object structure and corresponding class structure design, and the analysis of object behaviour and method design (Martin 1993). The analysis of object structure is mainly related to the kind of object and the method associated with it. The corresponding design includes design of class, which includes types of object, and the design of superclass and subclass, the defined methods. The analysis of object behaviour is concerned with what happens to the

objects, and the rules of operation. The corresponding design is method design, which is related to the design of the detailed methods.

2.2. The conceptual model of the OO3D model

Based on the principles of the OO approach, an object in the real world can be abstracted as: $\text{OBJECT}=(\Lambda, X, A, \alpha)$, where Λ is the type of object, X is the set of operations about the object, A is the set of attributes of the object and α is the set of constraints on the object.

We now define the conceptual model of the OO3D data model. Here, we define node, line segment and triangle as the three basic elements of a 3D object model. Figure 6 illustrates the basic elements of the OO3D data model.

The conceptual OO3D model is further described on the basis of the following definitions.

Definition 1: We assume that nodes V_0, V_1, \dots, V_n are the nodes in *Euclidean n Space* (R^n). According to set theory and topology, if nodes V_0, V_1, \dots, V_n are not linearly related and independent, n -Simplex (S^n) is the minimum convex set of the nodes in R^n , and the nodes V_0, V_1, \dots, V_n are the vertices of the simplex and can be denoted by

$$S^n=(V_0, V_1, \dots, V_n) \quad (1)$$

According to formula (1), when n is equal to 0, it is a 0-Simplex, which is a node. When n equals 1, it is a 1-Simplex, which is a line segment. When n equals 2, it is a 2-Simplex, which is a triangle.

In the OO3D data model, the node is considered a 0-Simplex, the segment is considered a 1-Simplex, and the triangle is considered a 2-Simplex.

In the following discussion, we suppose that all objects to be handled by the proposed data model are in a *Euclidean 3D Space*, denoted by R^3 . This is of particular importance in the aggregation of different objects and to keep the topological relationship consistent.

Definition 2: Supposing N is the set of all nodes in R^3 , nodes P_i and P_j have the following property: $P_i \cap P_j = \phi$, $\cup_{i=0}^n P_i = N$, $P_i \in N (i \neq j)$.

Definition 3: Supposing *Segment* is the set of all line segments in R^3 , the elements in *Segment* have the following property: $\cup_{i=0}^n \text{Segment}_i = \text{Segment}_j$. If Segment_i intersects with Segment_j , then $\text{Segment}_i \cap \text{Segment}_j = P_k$, otherwise the intersection result is ϕ . This means there are at least two line segments in *Segment* that share the same node.

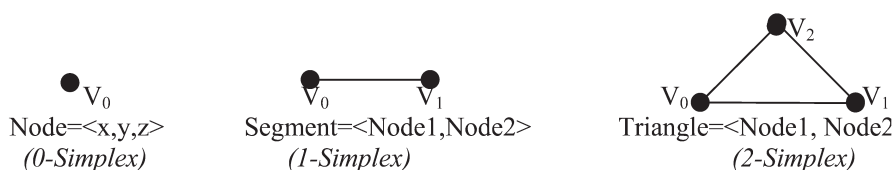


Figure 6. Basic elements of the OO3D data model.

Definition 4: Supposing $Line$ is the set of lines in R^3 and $line_i \in Line (0 < i < n)$. The elements in $Line$ have the following property: $\cup_{i=0}^n line_i = Line_j$. If $line_i$ intersects with $line_j$, then $line_i \cap line_j = Segment$, which means that there are at least two lines in $Line$ that share the same segment set.

Definition 5: Supposing F is the set of all triangles in R^3 and $f_j \in F (0 < i < n)$. The elements in F have the following property: $\cup_{i=0}^n f_i = F$. If f_i intersects with (f_j) , then $f_i \cap f_j = Segment_k$, which means that there are at least two triangles in F that share the same line segment.

Definition 6: Supposing S is the set of all surface objects in R^3 and $S_j \in S (0 < i < m)$. The elements in S have the following property: $\cup_{i=0}^n f_i = S_i$, $\cup_{i=0}^m S_i = S$. If the surface object S_i intersects with S_j , then $S_i \cap S_j = Segment (i \neq j)$. This means that there are at least two surfaces that share a common segment set.

Definition 7: Supposing $Volume$ is the set of all volumes (3D objects) in R^3 . It can be denoted as follows: $Volume = \cup_{i=0}^n Volume$, and n is the total number of the sub-objects. Vol_i is a sub-object (one of the composed objects in a complex object), and $\cup_{i=0}^k S_i = Volume_i$; k is the total number of surfaces in the $Volume_i$. If $Volume_i$ intersects with $Volume_j$, then $Volume_i \cap Volume_j = S$.

Definitions 2 to 7 are based on set theory and topology and, the definitions are illustrated as figure 7. These definitions describe the relationships between node and line segment, surface and triangle, volume and surface. These relationships are the foundation for spatial analysis and query in 3D GIS. In applying the model to real-world objects, point, line, surface and volume objects should fulfill the conditions specified in definitions 2 to 7.

Having specified the above definitions, we now define the basic elements of this model: node, segment and triangle. Complex spatial objects can be constructed based on the aggregation of these basic elements. Aggregation not only applies to basic

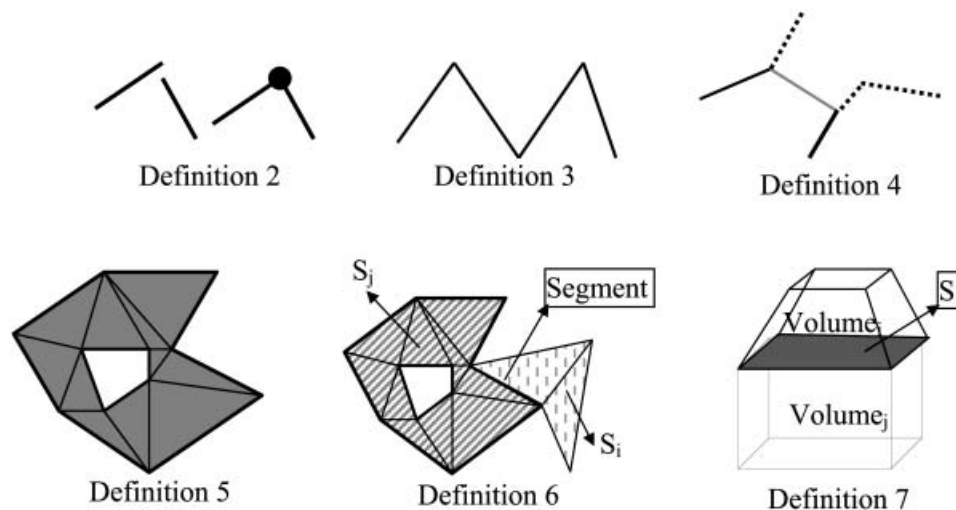


Figure 7. Illustration of the definitions.

elements themselves, but also extends to sub-objects, objects and complex objects. Figure 8 demonstrates the conceptual model of the OO3D data model. Since the lowest level element for constructing a spatial object (such as a surface or volume) is the triangle, the triangle is thus defined as one of the basic elements for constructing complex spatial objects. Furthermore, the relationship between the spatial objects can be described by the corresponding relationships between the triangles. The topological relationships between 3D objects are represented by the relationships of the composed triangles. The part relationships between two triangles can be of intersection, disjunction and adjacency, as illustrated in figure 9. Here we only illustrate several examples of the overall topological relationships, 'intersection', 'disjoint' and 'adjacent' relationships between two triangles. A completed set of the topological relationships between two objects in space is described in Zlatanova (2000).

2.3. The logical model of the OO3D object

The logical OO3D data model is designed based on the proposed conceptual model and the hierarchy relationship of the objects, considering the requirements in 3D visualization, query, data management and reconstruction.

According to the proposed conceptual model, a complex object is an aggregation of several simple objects; a 3D object (volume) is aggregation of several surface objects; a surface object is an aggregation of triangles, and a triangle is the minimum element, composed of three node objects, for constructing objects. Each node has its own position determined by X , Y and Z coordinates. The hierarchical relationships between volume object, surface object, triangle object, segment object and node

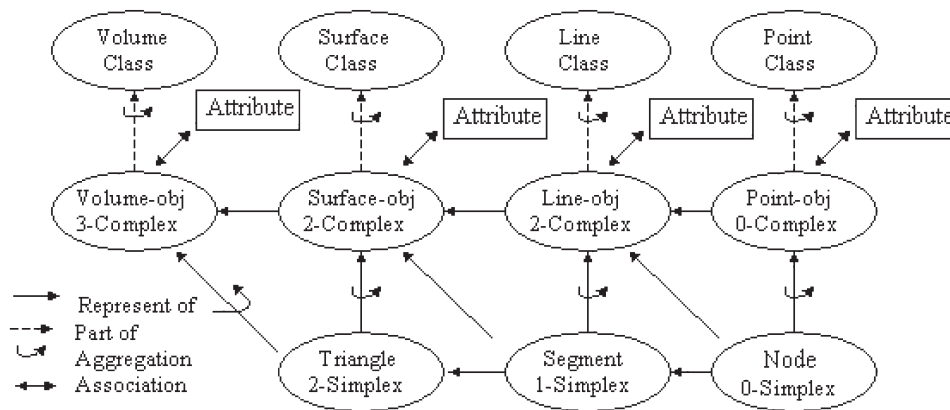


Figure 8. The conceptual model of the OO3D data model.

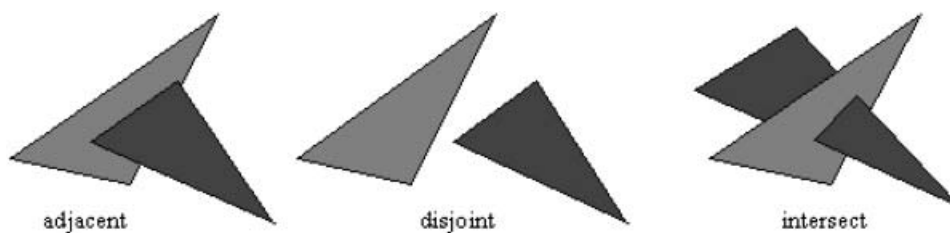


Figure 9. The relationship of two triangles in 3D space.

object are defined in the model. Figure 10 illustrates the dependency diagram of the OO3D model, but it just illustrates the basic set of dependent relationships in the OO3D model. Other relationships can be deduced from this basic set of relationships. Geometric aspect information about the spatial object is described by the ellipses.

The dependency diagram of the OO3D model can be easily converted as the relation tables or OO databases. The OO approach is adopted, considering the following advantages of the OO approach: (a) it is easier to manipulate objects when they are managed as individual objects, (b) data is not easily destroyed when objects are encapsulated together. As a result, the abstract representation of the OO3D data model is implemented based on the OO approach, as illustrated in figure 11. Each object can be associated with different attribute data, such as building surface material, texture and others.

In the OO approach, CASE Tools are often used to analyse and design the OO systems and models (Martin 1993). They are recommended as the standard for analysing objects in the OO approach.

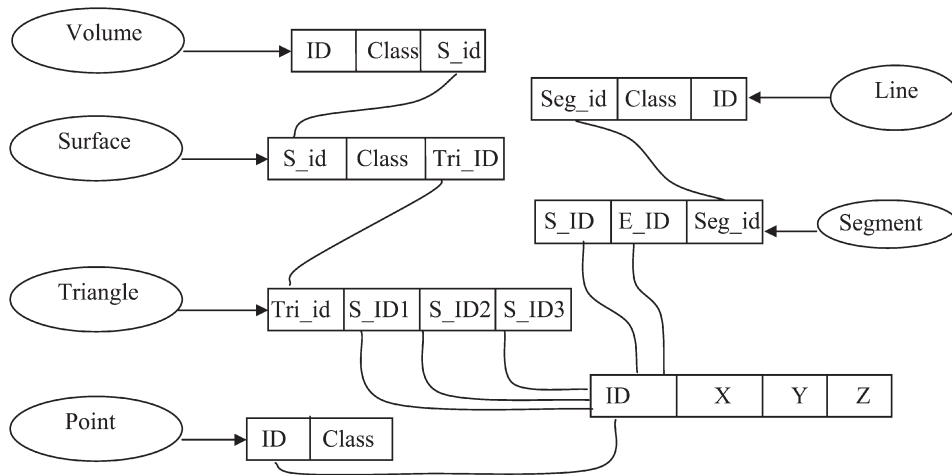


Figure 10. The dependency diagram of the OO3D data model.

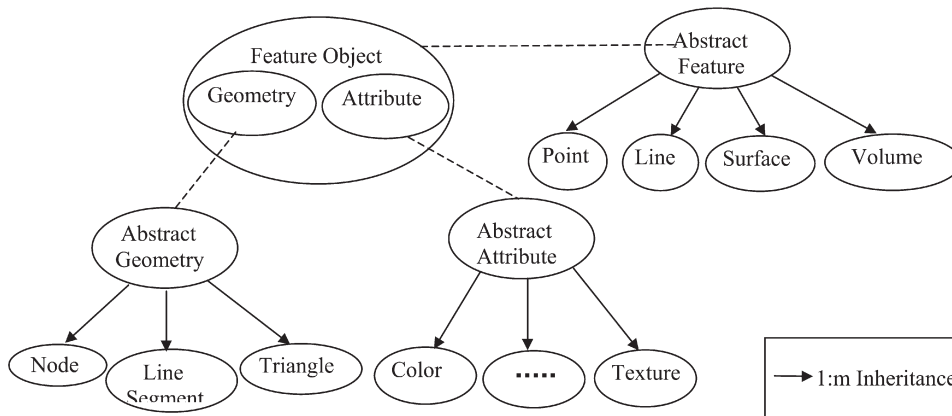


Figure 11. Abstract representation of the OO3D object model.

Based on the concept model and dependency diagram of the OO3D data model and the types and methods of abstract objects, the relationships and constraints among abstract objects can be analysed and designed by using CASE Tools. Once these conditions are designed, the classes of objects can be designed and coded, and the development of the corresponding logical model can also be finished. Figure 12 illustrates the logical model of the OO3D model, which is designed by using CASE Tools.

As illustrated in figure 11, a feature object in GIS has two aspects: attribute and geometry. In the OO3D model, three elements (node, line segment and triangle) are defined as the basis-abstract geometry. Each basic element has its own behaviour and attribute, while at the same time also possessing the behaviour and attribute of its superclass. The instances of abstract attribute may include, for example, the colour and texture of a building surface. Abstract features are abstractions of features in the real world, such as a manhole, roads, terrain surface and buildings. These can be summarized as point, line, surface and volume objects.

Figure 12 further illustrates the logic OO3D model in detail. In this logic model, we define the feature objects, methods and instances.

A feature object can be one of the following four features: point, line, surface and volume. A point feature can be described by a node (0-simplex). The abstract geometry of a node is described by its coordinate (x, y, z). The line feature is composed

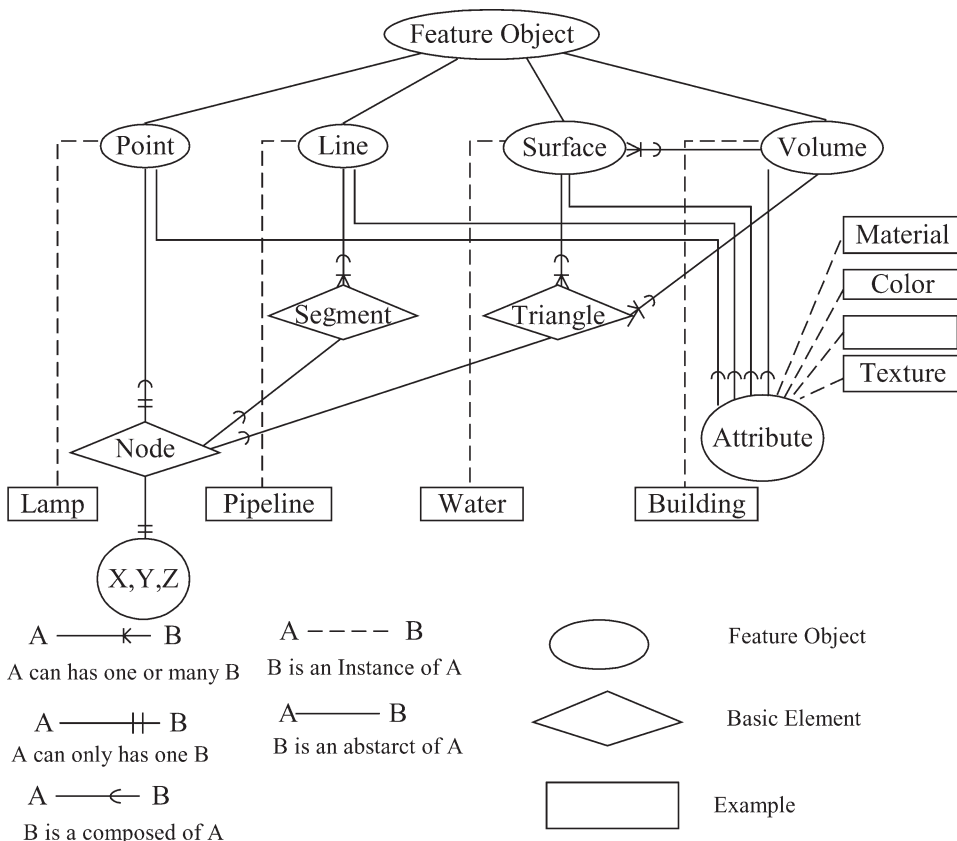


Figure 12. Logical OO3D model.

of many segments (1-simplexes), and a line segment is composed of two nodes. The surface feature is composed of many triangles (2-simplexes), and a triangle is composed of three nodes. The volume feature is composed of many triangles (2-simplexes) or surfaces.

For each of the basic elements (node, line segment and triangle), the method and constraint about them are defined. These methods may include, for example, move, copy and rendering. The feature objects are composed of the basic elements; they are the superclasses of the feature objects. Thus, feature objects inherit the methods of basic elements. Therefore, these methods also apply to feature objects. That is to say, we can apply move, copy, and render operations to the feature objects.

An attribute is a description of the behaviour of spatial objects. An attribute can be associated with each type of feature object: point, line, surface and volume. For example, the instance of attribute can be colour, texture and material of a building.

3. The formal representation of the OO3D model

A formal representation is a representation of spatial object based on the OO3D data model. It forms a theoretical foundation of spatial object reconstruction and system development. Based on the logical model of the OO3D data model, complex volume objects in the real world can be described. An example is given as follows:

$$\begin{aligned} \text{Complex Volume Object} &= \langle \text{Volume}_1, \dots, \text{Volume}_n \rangle \\ \text{Volume}_i &= \langle \text{Surface}_1, \dots, \text{Surface}_n \rangle \\ \text{Surface}_i &= \langle \text{Triangle}_1, \dots, \text{Triangle}_n \rangle \\ \text{Triangle}_i &= \langle \text{Node}_1, \text{Node}_2, \text{Node}_3 \rangle \\ \text{Texture_set} &= \langle \text{Texture}_1, \dots, \text{Texture}_n \rangle \\ \text{Texture}_i &= \langle \text{Texture}_i, \text{Surface}_j_ID \rangle \end{aligned}$$

Texture_set is the set with all textures. A particular texture can be, for example, the data format in which the texture is stored as either bmp or jpg. Each texture can be mapped to many different surfaces. Table 1 lists the formal representation of point, line, surface, simple volume and complex volume objects.






4. Examples and analysis

Based on the OO3D model, a 3D GIS, SpaceInfo, is developed in the c++ language with the OpenGL graphics engine. The main functions of SpaceInfo include reconstruction of 3D objects based on captured 3D data, such as for buildings and terrain; management of reconstructed 3D models (environment), 3D model import and export (dxf, vrml, 3ds), visualization and analysis of the reconstructed 3D models, and spatial query. In SpaceInfo, many different queries can be implemented, such as point query, region, buffer query, attribute query, updates, and others. Figure 13 illustrates a snapshot of the reconstructed 3D models with complex objects by SpaceInfo.

4.1. Construction of a building by the OO3D model

In order to demonstrate how the formal representation can be applied to presenting 3D objects, a volume object is given in the following example (figure 14).

Table 1. The formal representation of 3D objects.

Object type	Symbol	Data	Method	Attribute	Constraint	Examples
Point		Coordinates $\langle x, y, z \rangle$	Delete, Move, Copy, ...	Colour, Texture, ...	None	Trees, lamp post, ...
Line		Segment Set $\langle Segment_1, \dots, Segment_n \rangle$	Delete, Move, Copy, ...	Colour, Length, ...	$Segment_i \cup$ $Segment_j =$ $\{Point, \phi\}$ $(i, j = 1, \dots, n)$	Road central line
Surface		Triangle Set $\langle Triangle_1, \dots, Triangle_n \rangle$	Aggregate, Delete, Move, Copy, ...	Colour, Texture, Material, Area, ...	$Triangle_i \cap$ $Triangle_j =$ $\{Segment_k, \phi\}$ $(i, j = 1, \dots, n)$	Road surface, land parcel, ...
Simple Volume		Surface Set $\langle Surface_1, \dots, Surface_n \rangle$	Aggregate, Delete, Move, Copy, ...	Colour, Texture, Material, Volume, ...	$Surface_i \cap$ $Surface_n =$ $\{Triangle, \phi\}$ $(i, j = 1, \dots, n)$	Simple buildings, ...
Complex Volume		Volume Set $\langle Volume_1, \dots, Volume_n \rangle$	Aggregate, Delete, Decompose Copy, ...	Colour, Texture, Material, Volume ...	$Volume_i \cap$ $Volume_j =$ $\{Surface, \phi\}$ $(i, j = 1, \dots, n)$	Complex buildings, towers, bridges, ...

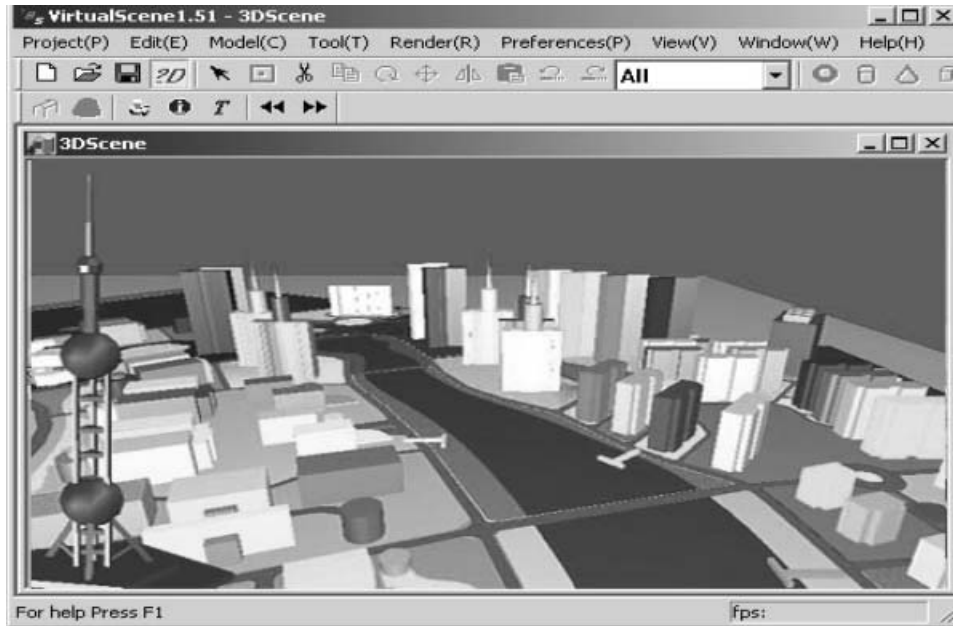


Figure 13. A snapshot of reconstructed 3D models with complex objects.

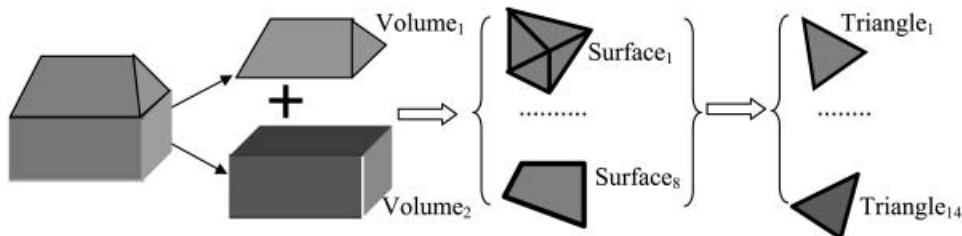


Figure 14. Representation of a complex volume object.

$$\begin{aligned}
 \text{Building Object} &= \langle \text{Volume}_1, \text{Volume}_2 \rangle \\
 \text{Volume}_1 &= \langle \text{Surface}_1, \dots, \text{Surface}_4 \rangle \\
 \text{Volume}_2 &= \langle \text{Surface}_5, \dots, \text{Surface}_{10} \rangle \\
 \text{Surface}_1 &= \langle \text{Triangle}_1, \dots, \text{Triangle}_6 \rangle \\
 \text{Surface}_2 &= \langle \text{Triangle}_7, \text{Triangle}_8 \rangle \\
 &\dots\dots\dots \\
 \text{Triangle}_i &= \langle \text{Node}_1, \text{Node}_2, \text{Node}_3 \rangle
 \end{aligned}$$

First, the object type should be determined as either a complex volume object, a simple volume object, a surface object, a line object or a point object. Once the type of object is determined, the hierarchy relationships between complex volume, simple volume, surface, segment and nodes can be established. The corresponding hierarchical relationships, including complex volume objects and simple objects, simple volume and surfaces, surface triangles, triangles and segments, triangles and nodes,

segments and nodes can be constructed. At the same time, the attribute of each surface object can be associated with the surface object, such as texture, construction material, colour and others. Having defined the hierarchical relationship of volume, surface, triangle, segment and node, the data for the object can be stored in the OO database and the object constructed.

The applicability of the proposed OO3D model and the formal representation method is demonstrated by the SpaceInfo system. Figure 15 is the example 3D model of a study area of Hong Kong reconstructed by the software based on the proposed OO3D model. Figure 16 is the VRML visualization of the same model as exported by the SpaceInfo.

In order to verify the performance of the proposed OO3D model, we used SpaceInfo to construct 3D city models. The performance of our proposed model is tested in terms of both data volume and speed. The experiments are conducted in a computer with the following configurations:

- CPU: PIII700
- RAM: 128 Mb
- Bandwidth of Local Network: 100 Mb

4.2. Data volume performance

To compare the performance between the proposed SSS with the TEN and 3D FDS models Hong Kong city is used as the test area to test the performance of our proposed OO3D model.

The 3D building models and terrain model are created using the SpaceInfo system. The experimental results, both data volume and processing time, for the OO3D model are derived from the SpaceInfo system.

The data types and size of the Hong Kong study area as represented by the OO3D models are listed in table 2.

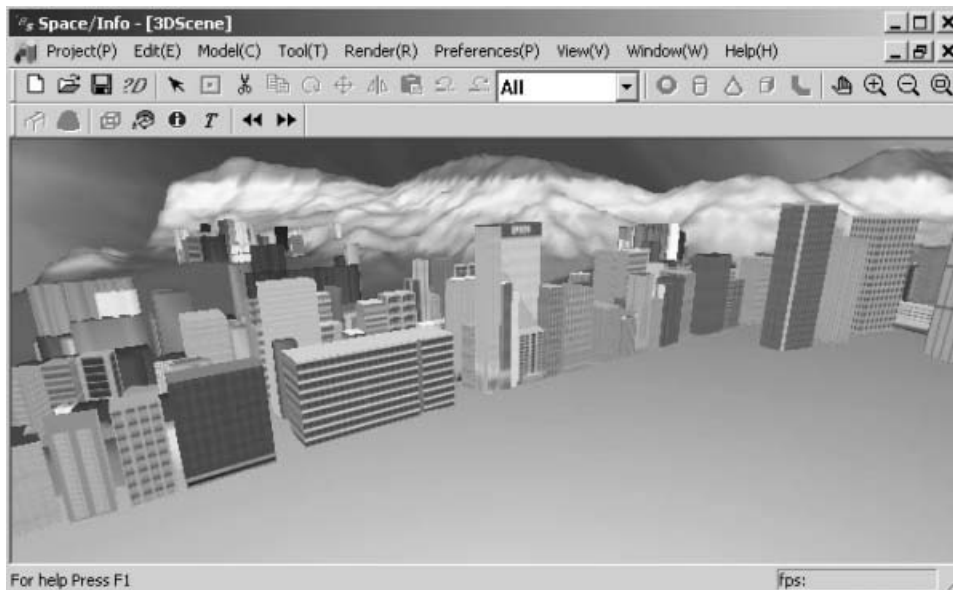


Figure 15. The reconstruction of Hong Kong.

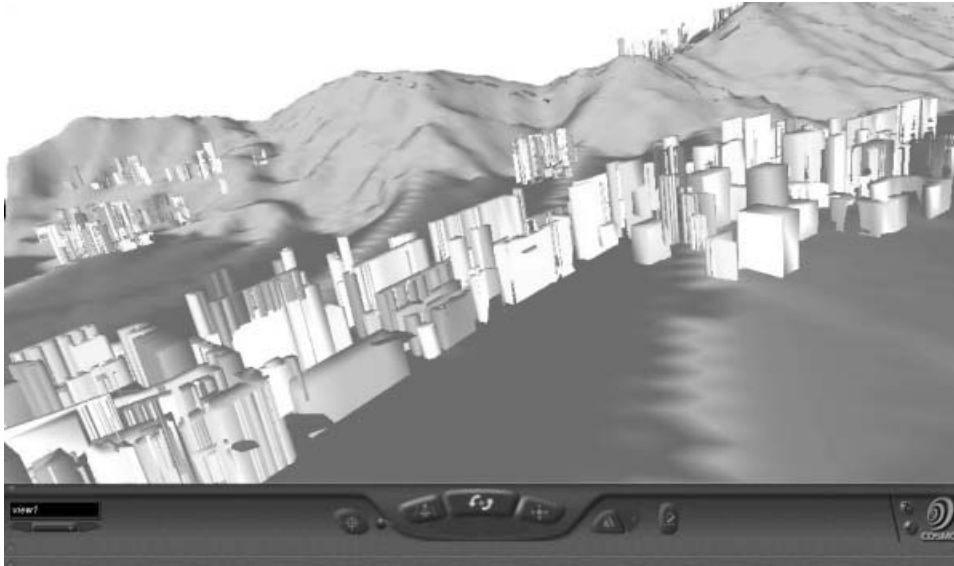


Figure 16. 3D visualization of the study area based on VRML.

Table 2. Data type and volume of the study area in Hong Kong.

Date type/model	3D FSS	SSS	OO3D
Body	1600	1600	1600
Surface	0	—	—
Line object	—	—	—
Point object	—	—	—
Face	18 578	18 578	—
Arc	25 003	—	—
Edge	92 268	—	—
Node	30 756	30 756	30 756
Triangle	—	—	54 866

In OO3D model, the node, triangle and body is stored as a binary large object (BLOB) in RDBMS, in order to depict the content of BLOB, we developed API functions. The topological relationships between them are stored in BLOB. The size of data storage is calculated by the sum of all BLOBs in RDBMS.

Table 3 lists the size of data storage based on OO3D. Based on table 3, we can see that the amount of data storage for OO3D is 1 830 258 bytes.

The reason why the OO3D model uses less storage is that the part of topological relationships are stored implicitly in the OO3D model. More computation may be involved if topological relationships between spatial objects are required for complicated spatial analysis. Second, instead of the face the triangle is used as one of the basic elements in OO3D model. Furthermore, the OO approach is used in OO3D to store spatial objects, and so large numbers of the link relationships used in the relational tables are omitted. The data volume of the OO3D model is less than that of SSM. On the other hand, the topological relationships of the OO3D model is difficult to maintain compared to the SSM, because object data is stored as BLOBs in OO3D.

Table 3. The size of data storage based on the 3D FDS model.

3D FDS	Byte/record	Record No.	Bytes
Body_Object	24	1600	38 400
Surface_Object	24	—	—
Line_Object	24	—	—
Point_Object	28	—	—
Face	20	18 578	371 560
Arc	12	25 003	300 036
Node	16	30 756	494 096
Edge	13	92 268	1 199 484
Total	161	168 205	2 401 576

4.3. Speed performance

Dynamic visualization is an indispensable function of a 3D GIS, and speed is of vital importance in 3D data query, spatial analysis and interactive operation. With the SpaceInfo system, the speed performance of the OO3D model is tested with the data of the study in Hong Kong city. The speed of the OO3D model is based on SpaceInfo in a personal computer.

The time cost for a spatial query about volume and surface objects according to spatial coordinates (x, y, z) is tested. The time interval for a query is counted from the beginning of the query to the completion of displaying the searched object(s). Table 4 lists the response time of querying various buildings.

In the implementation of the OO3D model, an index for the objects is designed based on the quad-tree method, so as to increase the speed of visualization. The external factors, such as bandwidth the network, speed of the network, and others may affect the response time of the operation of a model. We classify the response time according to internal and external circumstances. This comparison test is based on the response time under internal circumstance, to minimize the effects from external factors, so the result reflects the speed of the model more realistically.

Table 5 lists the experimental results of speed performance of the OO3D model for both indexed and non-indexed data under the internal circumstance. From table 5, we can see that the response time of the OO3D model with indexing is faster than that without. This indicates that the larger the data volume, the greater the degree to which the OO3D system with indexing is faster. That is to say, when the data volume is greater, the OO3D-based system with indexing can show its merits in speed more obviously.

The reason that the OO3D-based system (SpaceInfo) has a better performance in response speed, particularly in 3D visualization, might be because of the following

Table 4. The size of data storage based on the SSS model.

SSS	Byte/record	Record no.	Bytes
Body-G	10	18 578	38 400
Body_AB	32	1600	51 200
Body_T	24	1600	38 400
Face	10	92 268	922 680
Node	16	30 756	492 096
Total	90	143 202	1 690 156

Table 5. The size of data storage of the OO3D model.

OO3D	Object no.	Bytes
Body_Object	1600	10 760
Surface_Object	—	—
Triangle	54 866	658 632
Node	30 756	369 072
Total	87 222	1 038 464

Table 6. The response time of the OO3D model based on OpenGL in PC.

Objects	Respond time (sec)	Number of nodes
1600 buildings	4.18	30 756
One building	0.02	14
Surface of building	0.1	4

Table 7. Speed performance of the OO3D model in processing data of the Hong Kong study.

Number of buildings	Internal	
	OO3D model (sec)	Indexed OO3D (based on quad-tree (sec))
1	0.5	0.06
20	2.1	0.12
50	4.3	0.6
600	10.5	6.5
1600		33.5

two points. As the total data volume of OO3D is smaller, the time cost in the transition of data through the network is smaller. The other reason is that each object is stored as a BLOB in the database, the whole data set for an object can be acquired from a single table when we retrieve an object from a database. As a result, it is faster than models in which the system has to retrieve several tables in order to search for one object, though we have to develop API functions to decipher the contents of a BLOB.

5. Conclusions

In this paper, we presented a new object-oriented data model for handling complex 3D objects. First, the conceptual data model was developed based on the OO data modelling approach. The model was designed based on the three basic elements: node, segment and triangle. The abstract geometric objects covered by the model (including points, lines, surfaces and volumes) were also defined. Second, the corresponding logical 3D model was based on the definitions of the abstract objects and the relationships between them. Third, a formal representation of the 3D spatial objects was described in detail. Fourth, a system (SpaceInfo) was developed on the basis of the proposed model. An experiment to reconstruct 3D objects based on the SpaceInfo system was carried out. Different experimental datasets are selected to test the performance of the proposed OO3D model. The results demonstrate that

the proposed OO3D model is more efficient in terms of both data volume and speed of response (not reported in detail) for 3D visualization.

One of the major advantages of the proposed OO3D model is its capability in handling complex 3D objects, in comparison with the previous models that are designed for mainly handling simple objects. This further improvement is particularly crucial for developing a cyber version of large cities where many complex objects exist, such as complex buildings.

The superior performance of the OO3D model in both data volume and speed lies in its model design, where the basic elements are node, segment and triangle. This design differs from the TEN and 3D FDS models (no Arc element as in the TEN and 3D FDS); it also differs from SSM (instead of the face element, a triangle element is used). This design reduces data storage in the construction of spatial objects.

For complex 3D models, the OO3D model stores several nodes in them, each node is related to one part of the complex model. When all the nodes are displayed, the full data is displayed. In 3D visualization, it can be judged whether a node needs to be displayed based on several conditions, such as the distance between object and viewer. Thus, a Level-of-Detail model of complex spatial objects can be very easily and efficiently generated in the proposed OO3D model, compared with other models. This is very critical in improving visualization speed, particularly when the data volume is large as a 3D data for a large city.

The OO3D model can be applied effectively in urban 3D applications, such as 3D building model reconstruction; surface model reconstruction, e.g., for water streams and road networks; 3D data management and maintenance. The model can be applied to 3D visualization and queries. Based on OO3D, spatial queries (region query, point query, buffer query) and high performance visualization can be implemented. On the other hand, the spatial analysis functions need to be improved in the future.

The topological relationships between the elements of spatial objects and those among spatial objects are implicitly stored in the OO3D model. This may affect some spatial analysis-related applications where topological relationships are involved, for instance in an application where a complicated Boolean computation between two 3D spatial objects are required. This will be one of the future developments for the current OO3D model.

Acknowledgments

The work described in this paper was substantially supported by grants from the Research Grants Council of the Hong Kong Special Administrative Region (Project No: PolyU 5050, 3-ZB40) and the Hong Kong Polytechnic University (1.34.9709), and partially supported by a grant from the China National Natural Science Foundation (No. 69833010).

References

- ARNAUD, S., and BERNARD, C., 1999, 3D topological modelling and visualization for 3D GIS. *Computer & Graphics*, **23**, 469–478.
- BERGOUIGNOUX, P., 2000, A perspective on dynamic and multi-dimensional GIS in the 21st century. *Geoinformatics*, **4**, 343–348.
- COORS, V., 2001, 3D GIS in networking environments. In *Proceedings of International Workshop on 3D Cadastres, 28-30 November, Delft, The Netherlands*, (Denmark: International Federation of Surveyors), pp. 159–169.
- ESRI, 1997. *ArcView 3D Analyst* (Redlands, California: Environmental Systems Research Institute Inc).

- GAHEGAN, M., 1999, Four barriers to development of effective exploratory visualization tools for geosciences. *International Journal of Geographical Information Science*, **13**, 289–309.
- GRUEN, A., and WANG, X. H., 1998, CC-modeler: a topology generator for 3-D city models. *ISPRS Journal of Photogrammetry and Remote Sensing*, **53**, 286–295.
- HUANG, B., JIANG, B., and LIN, H., 2001, An integration of GIS, virtual reality and the internet for visualization, analysis and exploration of spatial data. *International Journal of Geographical Information Science*, **15**, 439–456.
- KOFLER, M., REHATSCHKE, H., and GRUBER, M., 1996, A database for a 3D GIS for urban environments supporting photo-realistic visualization, home page <http://www.icg.tu-graz.ac.at/isprs96>.
- KRAAK, M.-J., SMETS, G., and SIDIANIN, P., 1998, Virtual reality, the new 3D interface for geographical information systems. In *Spatial Multimedia and Virtual Reality*, edited by J. Raper and A. Camara (London: Taylor and Francis), pp. 130–136.
- KONINGER, A., and BARTEL, S., 1998, 3D-GIS for urban purposes. *Geoinformatics*, **2**, 79–103.
- LI, R. X., 1994, Data structure and application issues in 3D geographical information system. *Geomatic*, **48**, 209–224.
- LI, Q. Q., and LI, D. R., 1996, Hybrid data structure based on Octree and Tetrahedron in 3D GIS. *Internal Archives of Photogrammetry and Remote Sensing*, **Vienna**, 503–507.
- LOSA, A. DE LA, and CERVELLE, B., 1999, 3D Topological modelling and visualization for 3D GIS. *Computer and Graphics*, **23**, 469–478.
- MARTIN, J., 1993, *Principles of Object-Oriented Analysis and Design* (Englewood Cliffs: Prentice-Hall Inc).
- MOLENAAR, M., 1992, A topology for 3D vector maps. *ITC Journal*, **22**, 25–33.
- MOLENAAR, M., 1998, An Introduction to the Theory of Spatial Object Modelling for GIS (Holland: T.J. International Ltd).
- OPENGIS SPECIFICATIONS, 2002, OpenGIS Consortium, Inc, available at: <http://www.opengis.org/techno/abstract.html>
- PFUND, M., 2001, Topological data structure for a 3D GIS. In *Proceedings of The 3rd ISPRS Workshop on Dynamic and Multi-dimensional GIS*, **34**, Part 2W2, 23–25 May, Bangkok, Thailand, pp. 233–237.
- PILOUK, M., 1996, Integrated Modelling for 3D GIS. PhD dissertation, ITC Netherlands.
- PILOUK, M., TEMPFLI, K., and MOLENAAR, M., 1994, A tetrahedron-based 3D vector data model for Geoinformation. *Advanced Geographic Data Modelling*, **40**, 129–140.
- RAPER, J., and KELK, B., 1991, Three-dimensional GIS. In *Geographical Information System: Principles and Application* (London: Longman), **1**, pp.299–317.
- RAPER, J., and KELK, B., 1993, Three dimensional GIS. In GIS Volume I: Principle, edited by D. J. Maguire, M. F. Goodchild, and D. W. Rhind (London: Longman Science & Technical), pp. 299–317.
- RAPER, J., MCCARTHY, F., and LIVINGSTONE, D., 1993, Interfacing GIS with virtual reality technology. In *Proceedings Association for Geographic Information Conference, Birmingham*, **3**, pp. 1–4.
- RIKKERS, R., MOLENAAR, M., and STUIVER, J., 1993, A query oriented implementation of a 3D topologic data structure. In *EGIS'93, Vol.2, Genoa, Italy*, pp. 1411–1420.
- SAMET, H., 1990, *The Design and Analysis of Spatial Data Structures* (Reading, Massachusetts: Addison-Wesley Publishing Company Inc.).
- SHI, W. Z., 1996, A hybrid data model for three-dimensional GIS. In *Proceedings of Geoinformatics'96 Wuhan International Symposium* (Wuhan: WTUSM), pp. 400–409.
- SUN, M., CHEN, J., and ZHOU, Q., 1999, A 3-dimensional model for visualizing cloverleaf junction in a city model. In *Proceedings of UM3'99 - International Workshop on Urban 3D/Multi-Media Mapping*, edited by R. Shibasaki and Z. Shi, Tokyo, Japan, pp. 203–208.
- TEMPFLI, K., 1998, 3D topographic mapping for urban GIS. *ITC Journal*, **3/4**, 181–190.
- VERBREE, J., MAREN, G. Y., and GERMS, R., 1999, Interaction in virtual world views-linking 3D GIS with VR. *International Journal of Geographical Information Science*, **19**, 385–396.
- YANG, B. S., and LI, Q. Q., 2000, Building model creating and storing in 3D urban GIS. *International Archives of Photogrammetry and Remote Sensing*, **33**, 504–511.
- YOSHIDA, T., 1998, Three-dimensional object modelling in a three-dimensional urban map. *Proceedings of UM3'98*. Tokyo, Japan, pp. 55–62.
- ZLATANOVA, S., 2000, 3D GIS for urban development. PhD dissertation, ITC Netherlands.
- ZLATANOVA, S., and GRUBER, M., 1998, 3D GIS on the Web. In *Proceedings of ISPRS, Commission. IV, Stuttgart, Germany*, pp. 691–699.

