A 3-D city model data structure and its implementation in a relational database

Author(s):
Wang, Xinhua; Grün, Armin

Publication date:
1998

Permanent link:
https://doi.org/10.3929/ethz-a-005746845

A 3-D CITY MODEL DATA STRUCTURE AND ITS IMPLEMENTATION IN A RELATIONAL DATABASE

Xinhua Wang Armin Gruen
Institute of Geodesy and Photogrammetry
Swiss Federal Institute of Technology (ETH) Zürich
ETH Hönggerberg, CH-8093 Zürich, Switzerland
Fax: 41-1-6331101
E-mail: wang@geod.ethz.ch agruen@geod.ethz.ch

ABSTRACT

With the development of modern cities, 3-D spatial information systems (SIS) are increasingly required for spatial planning, communication systems and other applications. The geometric information to be used in a spatial information system usually includes two types: vector data (such as buildings, traffic ways, waterways, trees, DTM, etc.) and raster data (such as orthophotos, original images from aerial or still video cameras, etc.). Considering the availability and advantages of relational database technology, it is an important task to develop a data structure which integrates vector and raster data.

In this paper, a self-developed data structure (V3D) is presented. The basic geometric element is the point. Points are used to express faces and line segments. Hierarchically, a 3-D object consists of faces, a face consists of ordered polylines, a polyline consists of ordered line segments, and a line segment consists of points. Thus the geometrical and the topological information of an object are defined. Once some attributes are attached, a geometrical object becomes an entity class. A total of four classes are defined: 3-D body, surface, line and point. The DTM is considered a separate entity class because some applications require particular processing of DTM. In addition, the images taken from aerial or still video cameras are attached to the faces as special attributes. We will present the data structure together with the implementation in a relational database. Also, we will report about a prototype system, CC-SIS (CyberCity Spatial Information System).

1. INTRODUCTION

The generation and visualization of 3-D city models became an important issue in the recent past due to the increasing demand for a realistic presentation of the real world. 3-D models facilitate the processes of city planning, communication system design, control and decision making, tourism, especially in urban areas. Different applications may require different data types and manipulation functions. The geometrical information to be operated on in a 3-D city system usually includes two types of data: vector data (e.g. man-made objects) and raster images. An appropriate data model should not only represent the geometrical information, e.g. shape, length, area etc. but also implicitly or explicitly describe the topological relationship between geometrical objects, such as adjacency relations, link relations, in/out relations, positional relations. In the case of texture mapping, it also must have the ability to manipulate raster images. The complexity of spatial objects and the variety of data types, especially 3-D objects and images as two completely different data types, makes it a challenging task to develop a 3-D spatial model and data structure for the purpose at hand.

At the Institute of Geodesy and Photogrammetry, ETH Zürich we are involved in a project, "Integration of Raster Images and 3-D Objects into Geodatabases". Two important research topics treated by our group are: (a) the generation of the topology of 3-D objects by using photogrammetric tools, and (b) the investigation of the data model and the development of a system to manage the vector data and raster images based on the relational database technology. The former problem has been addressed with our CC-Modeler (CyberCity Modeler) system (Gruen, Wang, 1998). In this paper, we will present a technology for management of data, which supports the

following applications:

- Photorealistic presentation with possibilities for navigation through the 3-D city model
- Abilities to create, store, design, analyse and query city objects

The goal of this paper is to present our self-developed data structure as well as the implementation based on relational database technology for integrating 3-D vector data and raster images. In section 2 the data set for the 3-D city model will be addressed. A 3-D city model data structure will be presented in section 3. In section 4 the issue of implementation based on relational database technology is addressed. Finally, an application prototype is presented.

2. DATA SET

The implementation of a 3-D city information system depends on the application purposes. There are usually three types of data sets involved:

- Digital Terrain Model (DTM)
- Objects, e.g. buildings, roads, waterways, etc.
- Original images or orthoimages

2.1 DTM

The DTM is the most standardized data type in a 3-D city model. Usually there are three types of data structures to describe DTM, a regular grid, Triangular Irregular Network (TIN) and hybrid. Many investigations have been performed concerning the implementation of the above three types of representation (Fritsch, 1992). The regular grid structure allows an easy way of handling and storing the data, which is important for fast visualization. The advantage of a TIN structure is its fully consideration of the topographical features. It is easy to update, because each modification only requires re-triangulation in a local area. Considering the complexity of terrain structure in most city areas, the TIN structure is employed as a more suitable structure for DTM representation.

2.2 3-D Objects

In the past, several methods for 3-D object description have been investigated. Those models can be grossly subdivided into Wire-Frame, B-Rep, VBR, Cell-Decomposition, FBR, CSG. Breunig, 1996, has discussed their suitability for 3-D GIS by comparing the different 3-D representations according to the following criteria: domain, validity (geometric and technical), non-ambiguity and unambiguity, close operation, efficiency of geometric algorithm, accuracy, need for storage. He presented a new model, the e-complex model. However, this model seems not to be suitable for our application.

Another efficient model is Molenaar's 3-D topological vector structure, the Formal Data Structure (FDS) (Molenaar 1992, Rikkers, Molenaar, 1994). In the topological model, four geometric elements are used as the elementary data types of the geometrical part, and the relationships between geometric elements and object types are determined by five rules. FDS is a complete vector structure and shows powerful specifications for the presentation of topology, position and shape. The topology representation is beneficial for topology queries among the geometric elements of an object, but to generate a FDS description of 3-D objects is difficult, because FDS requires topological definitions between point and arc, arc and edge, edge and surface, edge and edge etc.

This situation lead us to develop a modified facet model based on B-Rep, V3D. This data model can be generated more easily and is designed to adapt images as well.

3-D objects usually can be classified into two types: surface objects (such as roads, waterways etc.) and body objects (such as building). Both types of objects can be completely represented with a facet model. On the other hand, a facet model is easily reconstructed with data from a photogrammetric system. CC-Modeler, developed in ETH Zürich, is a special photogrammetric tool to generate the facet model of 3-D objects from point cloud data obtained with a photogrammetric system (Gruen, Wang, 1998). It delivers the facet model data structure, V3D. V3D can conveniently be translated into other data files, such as DXF, IV, and AutoLisp, which are readable by

AutoCAD, MicroStation, Polytrim, ArcView and Inventor. The detailed structure of V3D will be presented in section 3.

2.3 Images

Images are another important data set in 3-D city models. In most applications, images are used as realistic texture data. Photorealistic texturing applied to 3-D objects gives the most natural presentation of the real world. Texture presents details which are not modelled in the vector data set and material properties.

Generally, there are two types of data sources: aerial images and terrestrial images taken from street level. The former usually are used for mapping on terrain surfaces and roofs of buildings, the latter are for building facades and other vertical faces. From a data structure point of view, both kinds of images are expressed as 2-D raster data, which can be stored or manipulated as a special layer in a 3-D system.

3. DATA STRUCTURE

V3D is a hybrid data structure. It not only models 3-D objects, but also combines raster images and attribute information for each object. The terrain objects are grouped into four different geometric object types:

- Point Objects: zero-dimensional objects which have a position but no spatial extension.
- Line Objects: one-dimensional or two and half dimensional objects with length as the only measurable spatial extension, which means that Line Objects are built up of connect line segments.
- Surface Objects: two-dimensional objects with area and perimeter as measurable spatial extension, which are composed of facet patches.
- Body Objects: three-dimensional objects with volume and surface area as measurable spatial extension, which are bordered by facets.

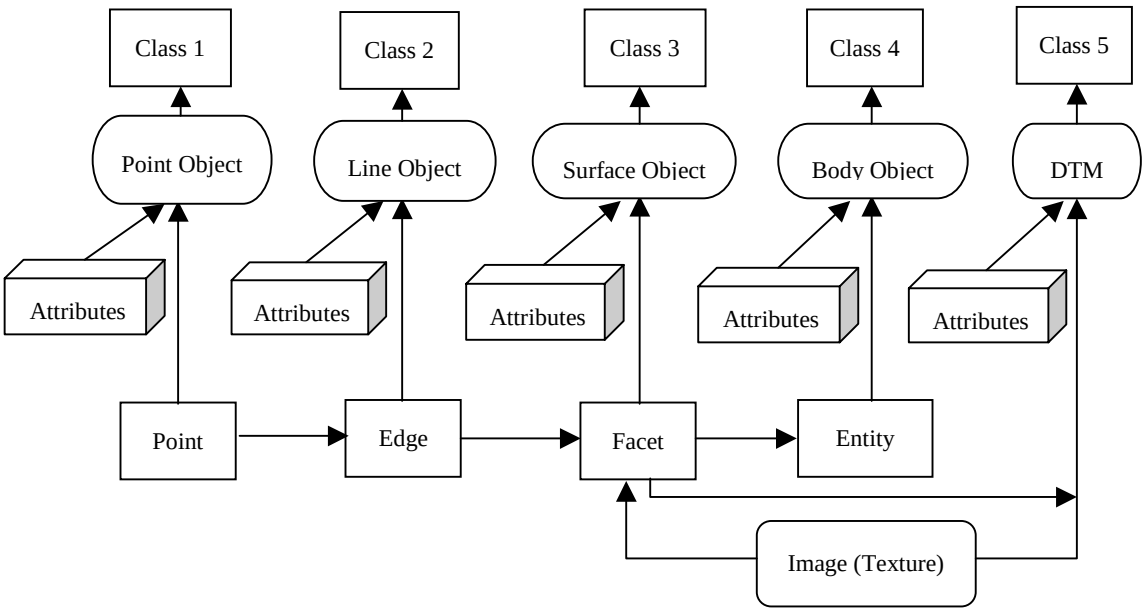

Figure 1: The logical data structure of V3D

In V3D, each special object is identified by Type Identifier Code (TIC), referred to as PIC, LIC, SIC and BIC, respectively. Two data sets are attached to each object type: thematic data and geometric data. The image data

can be attached to the surface object, body object and DTM object. In fact, the thematic data attributes are built up in a separate data table. It is linked to the object type with a related class label. The definition of the thematic data is user-dependent.

The geometric data set contains the geometric information of 3-D objects, i.e. the information of position, shape, size, structure definition, and image index. The diagram in Fig. 1 shows the logical data structure.

For the four object types, four geometrical elements are designed, i.e. *Point*, *Edge*, *Facet* and *Entity*. *Point* is the basic geometric element in this diagram. The *Point* can present a point object. It also can be the begin or end point of an *Edge*. The *Edge* is a line segment, which is an ordered connection between two points: begin point and end point. Further, it can be a straight part of a line object or lie on a facet. The *Facet* is the intermediate geometrical element. It is completely described by the ordered edges that define the border of the facet. One or more facets can be related to a surface object or *Entity* geometrical element. Moreover, *Facet* is related to an image patch. *Entity* is the highest level geometrical element, and it can carry shape information. An entity is completely defined by its bordering facets. Image data and thematic data are two special data sets, which are built up in two separated data tables. Each facet is always related to an image patch through a corresponding link.

Once the attribute table is attached and the TIC is labelled, a geometrical element becomes an object type. Especially, DTM is treated as a special data type, which is described by a series of facets.

Obviously, the topological relationships between geometrical elements are implicitly defined by the data structure. A point object is presented by a distinct *Point* element. The line object is described by ordered *Edges*. The surface object is described by the *Facet* with the information of image patches. Similarly, the body object is described by *Entity* that are defined by the facets. Thus, the topological relationships between *Point* and *Edge*, *Edge* and *Facet*, *Facet* and *Entity* are registered by the links between the geometrical elements.

4. IMPLEMENTATION IN A RELATIONAL DATABASE

In a relational database the most common object to be manipulated is the relation table. Other objects such as index, views, sequence, synonyms and data dictionary are usually used for query and data access. "Table" is the basic storage structure, which is a two-dimensional matrix consisting of columns and rows of data elements. Each row in a table contains the information needed to describe one instance of the entity, each column represents an attribute of the entity.

The data model shown in Figure 1 is a relational model, which can be implemented by relational data base technology. Figure 2 shows the relational model of the V3D data structure. Each object type is defined as a table, shown as the upper row. A point type table includes three terms. The Point Identification Code (PIC) is an identification code for a point type object. The Attribute Identification (AID) is coded to relate an attribute table. Different types of objects may have different attribute tables. For example, the attribute tables of "tree" may have different thematic definitions than "pole". The Name of Point (NP) is the identification of a geometric point, which is used to relate it with a distinct element in a point geometric element table. The *Point* table is the most basic geometrical element table, which defines the coordinate position of the geometrical points.

The line type table has similar content as the point type table. The difference is that a line type object is identified by the Line Identification Code (LIC) which is not directly linked with the geometric element table Edge, but linked with an intermediate relational table LIC-NIL and then indexed to the *Edge* table. The *Edge* table defines the geometrical element *Edge*, in which each edge (NIL) is described by the beginning point (BP) and the end point(EP). The surface type table and body type table have similar terms as the line type table. For each type of object a distinct identification code (SIC or BIC) is labelled. Both SIC and BIC are linked with a merging geometrical element table, *Facet-Entity-Image*, in which the topological relationships between *Facet* and Surface, *Facet* and *Entity*, *Facet* and *Image* are defined. *Facet-Entity-Image* table has two links: one is related to the *Image* table; the other is related to the NIL-SID table. *Image* table is a basic table, which describes all attributes of images, such as the image name, format, pixels, camera parameters, orientation parameters etc. The NIL-SID table is another intermediate table, which defines the corresponding relationships between *Facet* and *Edge*. Its NIL column is related to the *Edge* table. The DTM is treated as a special class, which is related to the NIL-SID table through an intermediate table DID-SID-Image.

Based on the relational structure shown on the diagram in Figure 2, the query of a geometrical description of a distinct object type is easily realized. For example, the query "Select the geometrical description of an object

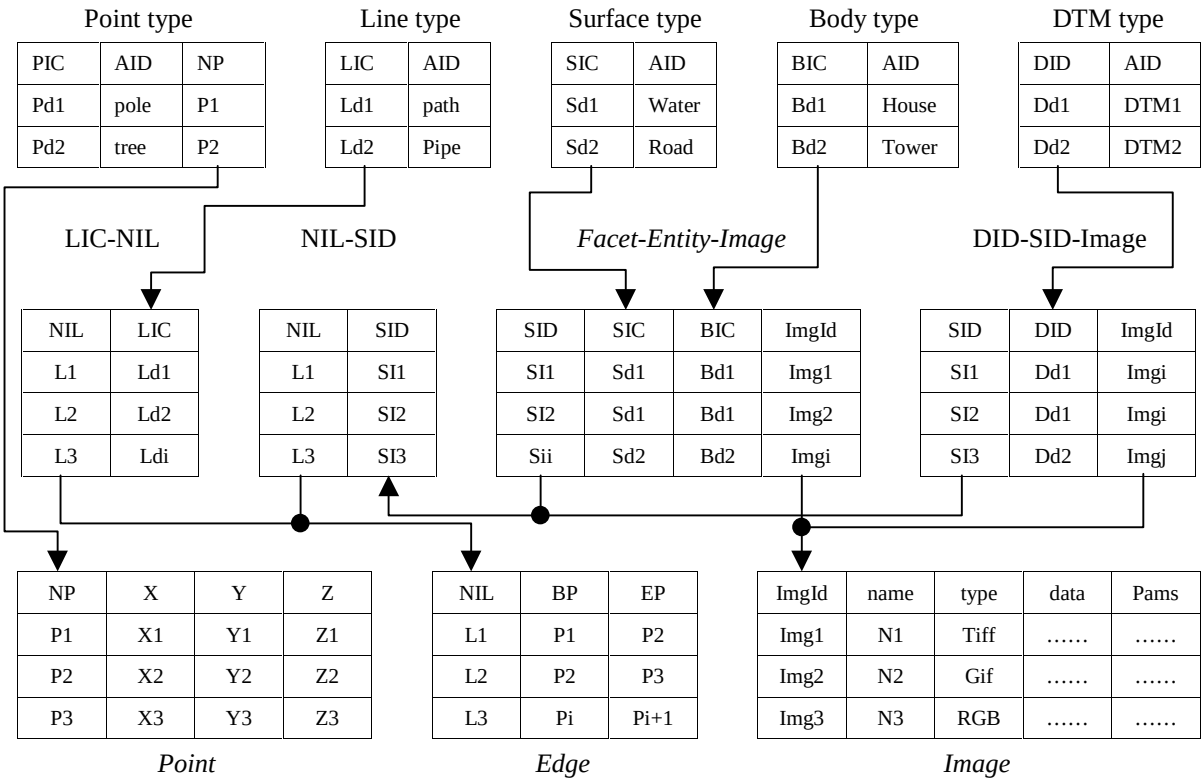

Figure 2: The relational model of the V3D data structure

with the identification code BIC = 202", will first index all Facet identification in the *Facet-Entity-Image* table by its BIC, and then get all edge name identification (NIL), finally index the position information of structure points with the help of *Edge* table and *Point* table. The geometrical information is defined in the following data structure as result of the query for a distinct body type object (such as building):

```
typedef struct _VEXPOINTS
{       double      x;
        double      y;
        double      z;
}VEXPOINTS;
typedef struct _FACE
{       int         num_elem;
        int         *Element;
        int         Id_Image;
}FACE
typedef struct _ROOF
{       int             num_face;
        FACE        *Face;
}ROOF
typedef struct _WALL
{       int             num_face;
```

```
        FACE        *Face;
}WALL
typedef struct _ENTITY
{       int         Entity_type;
        int         Id_code;
        int         num_point;
        VEXPOINTS       *vexPoints;
        ROOF        Roof;
        WALL        Wall;
} ENTITY            Entity
```

The queries of topological relationships are divided into two types: relationships between the geometrical elements of an object and those between objects themselves. The relationships between the geometrical elements are implicitly defined in the above data structure. Though the internal topology is not directly supplied, users can flexibly deduce the relationships, such as joint, adjacency, left or right, etc. The queries of topological relationships between objects are not considered in the above data structure because they are application-dependent.

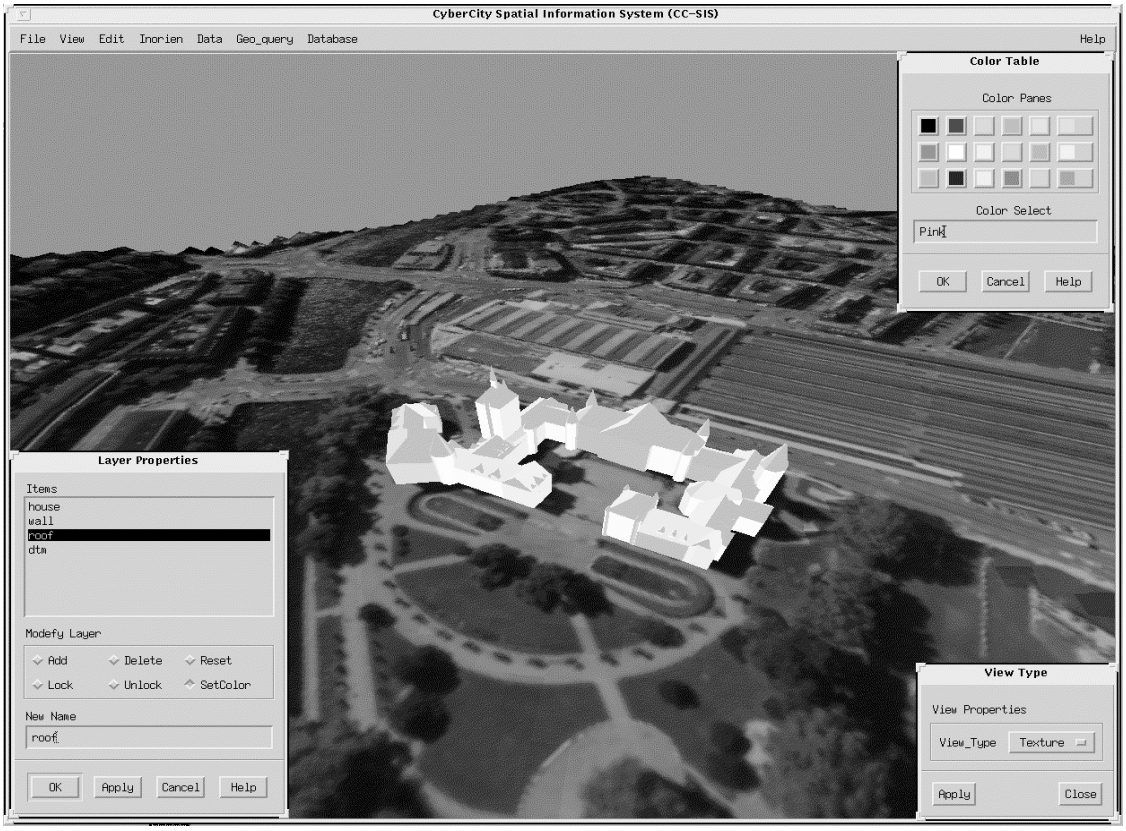

Figure 3: Overvoiew of CC-SIS interface

5. PROTOTYPE SYSTEM

Based on the above data model and structure, a special information system, CC-SIS, has been successfully developed and implemented on a workstation (Sun SPARC) under X-Windows, OSF/Motif, OpenGL as well as

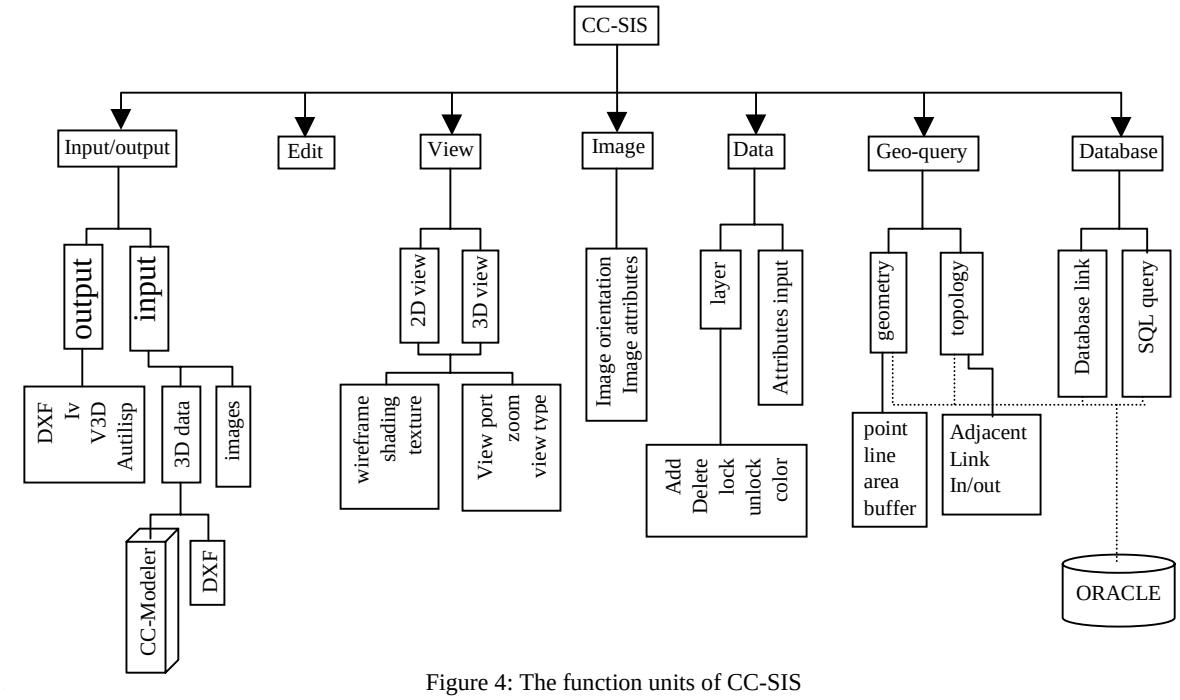

Figure 4: The function units of CC-SIS

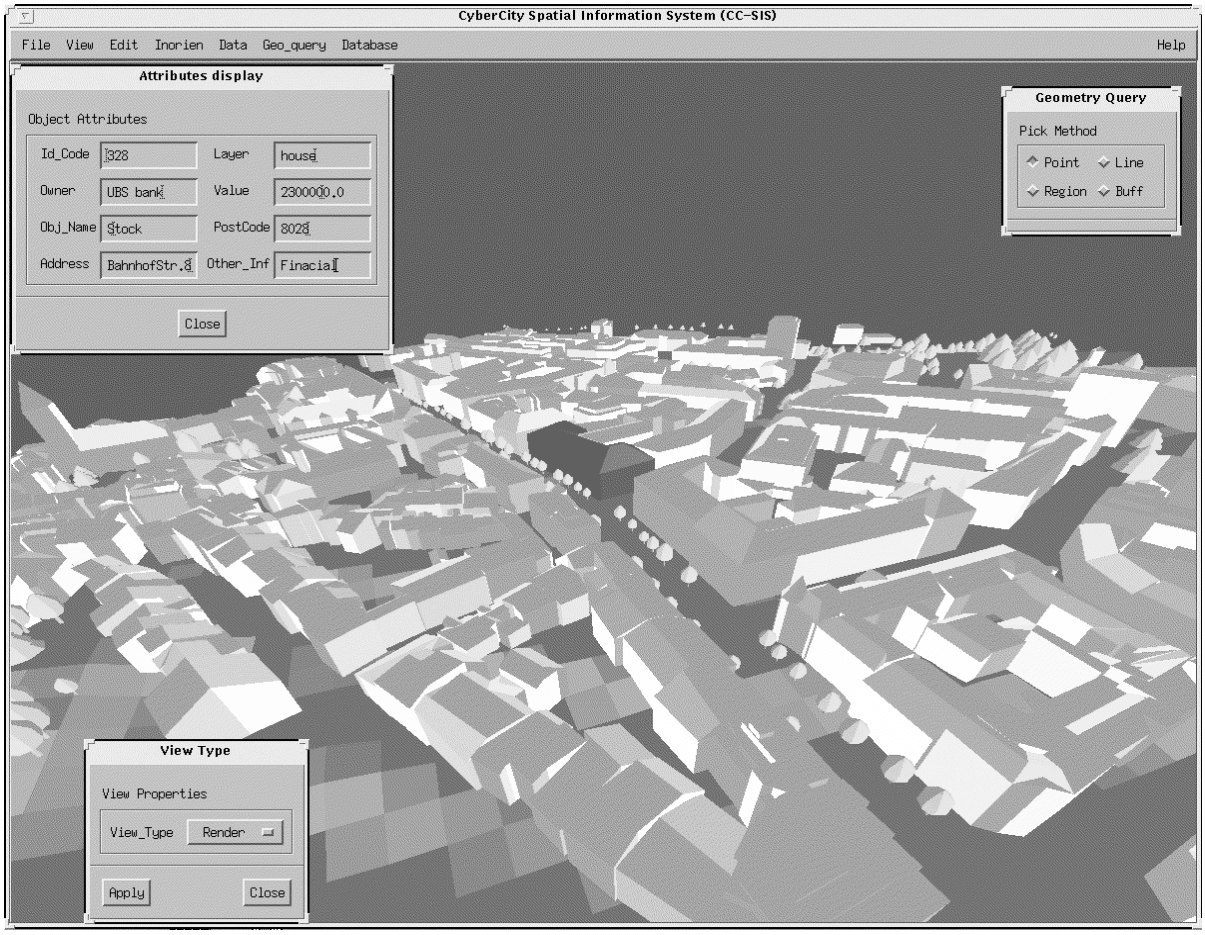

Figure 5: The geometrical query of CC-SIS

6. CONCLUSIONS

Based on the V3D vector data structure, a relational database has been created. The data to be operated on can be logically separated into geometrical parts, image and thematic. In this paper the focus is put on the geometrical part of the database. Our pilot applications show that V3D is a suitable structure for the representation of 3-D objects, images and thematic data. It is possible to answer most of the questions about topology, position and shape of objects by means of geometric or SQL queries.

Further research will be concerned with the following themes:

(1) More extensive investigation of the possibilities of the implementation of V3D

(2) Further analysis of topology and queries. Special attention will be paid to the relationships between the different geometrical elements and object types.

REFERENCES

1. Breunig, M., 1996. "Integration of spatial information for geo-information systems", Springer Cop., Berlin.

2. Fritsch, D., Pfannenstein, A., 1992. "Integration of DTM data structures into GIS data models", Int. Archives of Photogrammetry and Remote Sensing, Vol. XXIX, B3, pp. 497-503, Washington.

3. Gruen, A., Wang, X., 1998. "CC-Modeler: A topology Generator for 3-D City Models", Int. Archives of Photogrammetry and Remote Sensing, Vol.32, Part 4, pp.188-196.

4. Molenaar, M., 1992. "A topology for 3D vector maps", ITC Journal, No. 1, pp. 25-33.

5. Rikkers, R., Molenaar, M., 1994. "A query oriented implementation of a topologic data structure for 3-dimensional vector maps", International Journal of Geographical Information Systems, Vol. 8, No. 3, pp. 243-260.