# INCORPORATING 3D GEO-OBJECTS INTO AN EXISTING 2D GEO-DATABASE: AN EFFICIENT USE OF GEO-DATA

**Jantien STOTER**
Tel:+31-15-2788136, Fax:+31-15-2782745, E-mail:j.e.stoter@geo.tudelft.nl

**Peter VAN OOSTEROM**
Tel:+31-15-2788136, Fax:+31-15-2782745, E-mail:oosterom@geo.tudelft.nl

Section GIS Technology, Department of Geodesy
Faculty of Civil Engineering and Geosciences
Thijsseweg 11, P.O. Box 5030, 2600 GA Delft
The Netherlands

Keywords: 3D data models, 3D GIS, 3D spatial features, geo-DBMS, SQL for spatial data types

**ABSTRACT**
GISs are changing from static modelling in 2D to dynamic modelling in multi-dimensions. Yet, it will take an infinite time until the world as been defined within a full three- or four-dimensional approach. Besides that, a 2D approach still suffices in many cases. A relevant question is what kind of detail and resolution are required to satisfy the actual needs.

2D geo-information is available at large amounts, at different scales and covering many application domains. An approach to combine 2D and 3D geo-data in one DBMS, including storage and query can support an efficient use of data. Therefore at the Department of Geodesy, Delft University of Technology a research is started on this topic in order to look for possibilities and accompanying complications.

The background of this research is the need for more than just a 2D approach by the Dutch Cadastre (the Netherlands' Kadaster). In countries with an intensive use of land, like in the Netherlands, there is a growing interest to use space under and above the surface. With this interest, there is also a need to develop and maintain insight in the situation under and above the surface. Since this detailed information is not needed for rural land and furthermore there is a lot of information available in the second dimension, a system is required which is able to add 3D geo-objects in the current 2D geo-DBMS.

In this research, concepts are developed to integrate 3D objects in the current geo-database defined in 2D. The integration consists of data definition, data manipulation and data querying of 2D and 3D data in one environment. The concepts are being translated into prototype implementations. These implementations cover the following aspects:
- modelling and storage of 3D geo-objects into a geo-DBMS based on 2D parcels (data definitions)
- collecting, preparing and inserting (generalising/converting) 3D data
- querying and visualising the data (administrative, 2D, 3D and in combination with each other)

The research will finally lead to the implementation of all these aspects in one system: a geo-DBMS supporting 2D geo-objects and 3D geo-objects as well as in combination with each other.

This article describes both the background of the research as well as the progresses made so far. The basic concepts of a geo-DBMS are discussed, including the recent standardisation efforts by ISO and the OpenGIS Consortium. The current 2D model of the Netherlands' Kadaster is explained, which also includes a discussion of some possible 3D model extensions. Some of the issues involved in 3D geo-data collection and preparation are described, either based on CAD models created by designers or on surveying. Important in this is the insertion of the 3D data into the 2D geo-DBMS and the definition of the 'horizontal zero level'. Some examples of integrated querying and viewing of 2D and 3D data are given. Finally, conclusions and future work are described.

## 1. INTRODUCTION

In science as well as in applications, there is a growing interest in modelling the world in multi-dimensions. In this scope, GISs are changing from static modelling in two dimensions to dynamic modelling in three dimensions. However, a full and complete three (or four) dimensional model of the world is a too heavy solution in many applications. Thereby, not only the development of such models and representations should be considered, but also the availability and the effort that it takes to collect the needed data to make the developed models operational. Finally, incorporating the developed concepts into present systems and working processes is an important factor for reconsideration.

A relevant question is what kind of detail, or map-scale is required to satisfy the actual needs. Geo-information in two dimensions is available at large amounts, at different scales and covering many themes. In addition, an approach to combine two-dimensional and three-dimensional temporal data in one database, including storage and query, can support an efficient use of data. Therefore at the Department of Geodesy, Delft University of Technology a research is started on this topic in order to look for possibilities and accompanying complications.

The background of this research is the need for more than just a two-dimensional approach by the Dutch Cadastre (the Netherlands' Kadaster). In the Netherlands, there is a growing interest to use space under and above the surface because there is a high pressure on the available land. With this interest, there is also a need to develop a spatial Database Management System (DBMS) to maintain insight in the situation under and above the surface.

This detailed information is often not needed for rural land and, what is more important, there is a lot of valuable information available in the second dimension. Therefore, a system is needed which is able to integrate 3D information with 2D information.

This article describes the model that has been developed so far to integrate 3D objects in the current geo-database defined in 2D. This conceptual model consists of data definition (data models), data manipulation (inserting, deleting, and updating) and data querying of 2D and 3D data in one environment. This research is started as a practical request of the Netherlands' Kadaster. However, the research question covers the fundamental problem of combining newly required 3D data with conventional 2D data, which is available in a wide range and does still suffice in most cases. To understand the developed concepts, first the practical case used in the research is explained in Section 2 of this paper. Section 3 discusses the basic concepts of a geo-DBMS, including the recent standardisation efforts by ISO and the OpenGIS Consortium. The current 2D model of the Netherlands' Kadaster is explained in Section 4, which also includes a discussion of some possible 3D model extensions. Section 5 describes some of the issues involved in the 3D data collection and preparation, either based on CAD models created by designers or on surveying. Complex in this is the insertion of the 3D data into the current (2D) geo-DBMS. That is where are the 3D data located in respect to the 2D data Some examples of integrated querying and viewing of 2D and 3D data are given in Section 6. Finally, conclusions and future work are described in Section 7.

## 2. THE CASE

In the major urban centers (and especially their business districts) land use is becoming so intense, that very different types of 'land' use are being positioned under and above each other. Examples are:

- constructions on top of each other (see Figure 1);
- infrastructure above and under the ground;
- the increasing number of cables and pipes;
- the increasing number of owners of cables and pipes due to privatisation processes.

**Figure 1: various land use on top of each other**

With this tendency to use land more efficiently, the Kadaster, responsible for maintaining juridical boundaries and the legal status of real estates, is confronted with spatially complex property rights. Until now, the right of property of real estate is linked to the division of land in 2D parcels. A juridical horizontal division in land use can only be established by the registration of limited rights on the 2D surface parcel. Consequently, the use of land above and under the surface needs to be projected on the surface (see Figure 2). In some cases, it is even necessary to create new parcels on the surface to be able to define the legal status in 3D satisfactorily.

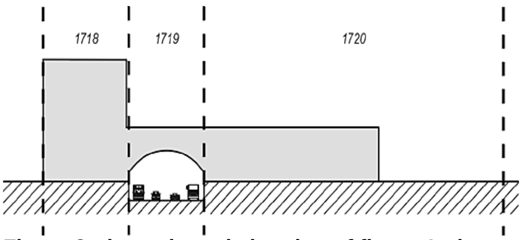



**Figure 2: the cadastral situation of figure 1: three parcels are needed to register the legal status of one building**

Some of the complexities met with the registration of 3D situations in a land information system that was originally developed to register parcels, are the following (Stoter, 2000):

- registration is only an administrative one;
- the lack of a (digital) 3D representation of the situation makes it difficult to get insight in the factual situation;
- current registration is used in situations for which it was not originally meant;
- the current solutions lead to divers registration practices for similar situations;
- the parcel is used for registration and not the object on, above or under the surface (building, tunnel etc.). This means that these real-world objects are not registered as such. This leads to the following complications:
    - the real-world object itself or characteristics of the object are not maintained and can therefore not be found at the cadastral registration system: the real world does not (only) consist of 2D parcels;
    - constructions are illogically divided into parts that match with the surface parcels;
    - it is needed to store the information of objects with every parcel that intersects with the object (e.g. a cable for which restrictions need to be registered under several parcels): this leads to redundant information which is a potential source of inconsistency.

Management of space under and above the surface will be improved when relevant information on real-world objects (location, function, geometry) is maintained and can be visualised. Besides, a (digital) geographical 3D representation of the situation when needed would facilitate the exchange and supply of valuable information on the factual situation and a better maintainable archive.

It is important to look at the actual needs of the Kadaster concerning 3D situations. These needs depend on the tasks and responsibilities of the Kadaster which consist of:

- the registration of the legal status of real estate objects;
- providing information on the legal status of real estate objects.

3D information is not needed in all cases (e.g. rural land). In this research a '3D cadastral system' has been defined as a system, which gives more insight in the juridical and factual situation above and under the surface in case this is relevant with respect to the legal security. This insight is obtained through spatially defining constructions on, above and under the surface as 3D geo-objects in the currently used cadastral system. Whether this definition of real-world objects should and will lead to the juridical registration of these objects remains beyond the scope of this research.

The approach to combine 2D and 3D data in one geo-DBMS, including storage and query, offers the facility to take the juridical relevant spatial information in the vertical dimension into account when the situation requires this (Stoter and Van Oosterom, 2000).

Besides the spatial representation of 3D geo-objects in the current geo-DBMS, attributes of these objects will also be maintained in the DBMS. The choice of having either implicit or explicit relationships (which have to be maintained) between the objects above and under the surface with the parcels on the surface will complete the incorporation of 3D objects in the current cadastral information system.

The concept of a 3D cadastral registration system is translated into prototype implementations. These implementations cover the following aspects:

- modelling and storage of 3D geo-objects into a geo-DBMS based on 2D parcels (data definitions);
- collecting and preparing 3D data and inserting (generalising/converting) the 3D data in the (2D) geo-DBMS;
- querying and visualising the data (administrative, 2D, 3D and in combination with each other).

The current database that is used by the Netherlands' Kadaster consists of (Lemmen et al., 1998):

- a geo-database for geometric properties of parcels (and buildings for reference purpose) in LKI: *Landmeetkundig Kartografisch Informatiesysteem* ('Information system for Surveying and Mapping')
- an administrative database for legal and other administrative data related to parcels (AKR: *Automatisering Kadastrale Registratie*: Automated Cadastral Registration)

Since this research focuses on technical, spatial solutions to map the 3D world when the situation requires this, the geo-DBMS will be used as starting point.

## 3. THE CONCEPT OF A GEO-DBMS

### 3. 1 An integrated architecture: the geo-DBMS

Due to the complexity of spatial features, topology and geometry spatial features used to be handled outside standard DBMSs within middleware and frontend Geographical Information Systems. This is not the optimal approach since this causes:

- the re-implementation of the same functionality many times;
- that other direct DBMS users might corrupt the data structure;
- non-optimal query plans (DBMS knows only 'half' of the characteristics of the data);
- overhead and data transfer between DBMS and middleware during query execution;
- non optimal management of the data, since GISs do not have the extended data-management functionalities DBMSs have.

The need for using a DBMS to store and manage spatial data is further enforced with the growing amount of (digital) spatial data, the growing number of users and disciplines and the growing complexity of geo-data. A geo-DBMS for managing geo-data (geometry as well as attributes) enhances the accessibility, security, consistency and integrity of spatial data and spatially linked data.

In conclusion, general and reusable tasks should be executed within the DBMS, other distinct tasks has to remain to the specific application (Van Oosterom et al., 2000).

Nowadays, the architecture of GISs is changing: systems are increasingly based on the integrated architecture, that is also storing geometric data (metric as well as topology) in the DBMS together with administrative data. The first step was to have data types and operators for the geometric primitives: point, line and polygon. This has reached the level of standardisation and is now implemented in several commercial DBMSs (see the next subsection). In case of large data sets, spatial indexing and clustering are also required, but outside the scope of standardisation. A spatial index is a structure supporting the spatial range queries by efficiently providing the addresses of the requested features. Spatial clustering makes sure that these addresses are physically close on disk and in this way too many inefficient 'jumping' on the disk is avoided. The subsequent step is also having support for the topologically structured features in the DBMS, that is complex features. With this the DBMS can check and guarantee consistency and complex operations can be executed within the DBMS. The support of spatial data types in DBMSs include:

- spatial operators (or geometry functions);
- spatial indexing;
- spatial clustering;
- topology management.

Based on these ingredients spatial queries can be stated (and answered efficiently). In case a spatial query involves two tables, each having at least one spatial attribute, which are used in a spatial operator in the where-clause, then this is called a spatial join.

**3.2 DBMS, SQL and spatial features**

Conventional DBMSs (Oracle, IBM DB2, Informix, Ingres) have implemented spatial data types and spatial functions more or less similar to the OpenGIS Consortium (OGC) Simple Features Specification for SQL (OGC, 1999).

The purpose of this specification is to define a standard SQL that supports storage, retrieval, query and update of simple spatial features. A simple feature is defined by the OpenGIS Implementation Specification to have both spatial and non-spatial attributes. Simple features are based on 2D geometry with linear interpolation between vertices.

Simple spatial feature collections are conceptually stored as tables (layers) with geometry valued columns in a relational DBMS: each feature is stored as a row in a table. The spatial attributes of features are columns whose SQL data types are based on the underlying concept of additional geometric data types for SQL. The specification describes a standard set of SQL Geometry Types based on the OpenGIS Geometry Model (OGC, 2001), together with the SQL functions of those types.

The base Geometry class has subclasses for Point, Curve, Surface and Geometry Collection. The defined geometric collections classes in 0, 1 and 2D are: MultiPoint, MultiLineString and MultiPolygon for modelling geometries corresponding to collections of Points, LineStrings and Polygons respectively. MultiCurve and MultiSurface are introduced as abstract superclasses that generalise the collection interfaces to handle Curves and Surfaces. The attributes, methods and assertions for each geometry class are described in the specification (OGC, 1999).

The object model for geometry with the supported spatial data types is shown in Figure 3.

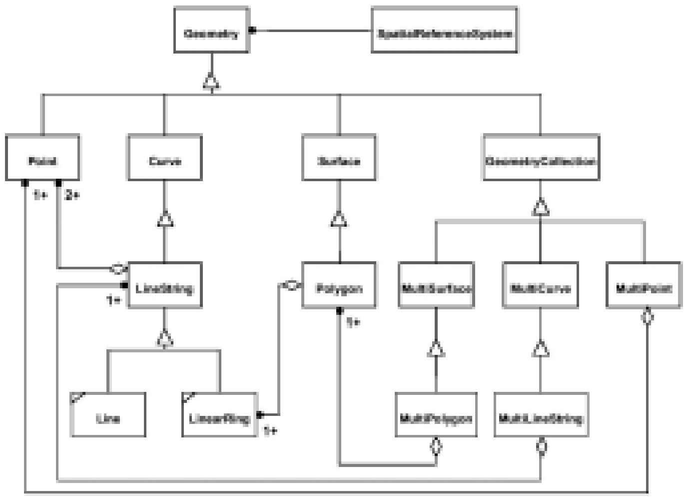



**Figure 3:the geometry object model defined by OGC (1999)**

An example, not strictly following the geometry model of the OGC, of the implementation of spatial functionality is the object-relational model in Oracle 8*i* spatial (Oracle, 1999). A geometry is stored as an object, in a single row, in a column of type MDSYS.SDO_GEOMETRY. Supported data types, besides point, line string and polygon are shown in Figure 4.

Oracle 8i offers several types of spatial indices: fixed grids, quadtree-like tiling and r-trees. Besides the overlap query functionality, that is using the operator sdo_relate, Oracle has three other spatial operators:

- sdo_filter (find overlap only based on bounding boxes);
- sdo_within_distance (find objects within a given distance from the query geometry);
- sdo_nn (find the nearest neighbours from the query geometry).

Note that spatial operators need a spatial index in order to work. Other spatial functionality in Oracle is available through several interesting geometry functions (not needing a spatial index) such as sdo_area and sdo_length (to obtain characteristics from the query geometries), sdo_buffer (to compute a buffer around a query geometry) and sdo_intersection (to clip spatial data from the source tables with the query geometries).

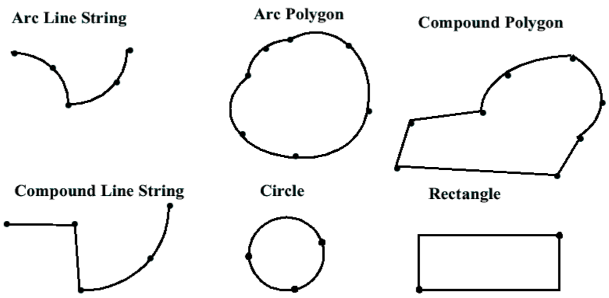



**Figure 4: Geometry types supported by the object-relation model in Oracle**

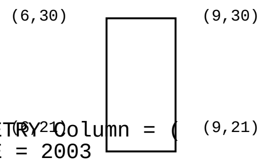
The object-relational model in Oracle defines the object type SDO_GEOMETRY as:

```
CREATE TYPE sdo_geometry AS OBJECT (
SDO_GTYPE NUMBER,
SDO_SRID NUMBER,
SDO_POINT SDO_POINT_TYPE,
SDO_ELEM_INFO MDSYS.SDO_ELEM_INFO_ARRAY,
SDO_ORDINATES MDSYS.SDO_ORDINATE_ARRAY);
```

SDO_GTYPE indicates the type of the geometry (point, linestring, polygon, multipoint, multilinestring, multipolygon). SDO_SRID is a reference to the spatial reference system used for the ordinates. SDO_ELEM_INFO is defined using a varying length array of numbers. These attributes show how to interpret the ordinates stored in the SDO_ORDINATES attribute. They include type of element (point, linestring consisting of straight lines, linestring consisting of circular arcs, polygon). A geometry object is composed of elements.

A z-value can be used to define the geometry objects. However, spatial index creation and geometry functions ignore the z-values in the current version of Oracle (8.1.7).

An example of using the object-relational model to represent a rectangle:




```
SDO_GEOMETRY Column = (
SDO_GTYPE = 2003
SDO_SRID = NULL
SDO_POINT = NULL
SDO_ELEM_INFO = (1,1003,3)
SDO_ORDINATES = (6,21,9,30) )
```

In SDO_GTYPE, the *2* indicates two-dimensional and the *3* indicates a polygon. In SDO_ELEM_INFO, the final *3* in 1,1003,3 indicates that this is a rectangle. Because it is a rectangle, only two co-ordinates are specified (lower-left 12,15 and upper-right 15,24).

Besides, the tables representing the geometry objects, metadata is maintained, describing the dimension, lower and upper bounds and tolerance in each dimension. This metadata is stored in a global table.

Curves and surfaces are not yet implemented in the commercial DBMSs. Specifications for complex features as well as for 3D geometry objects are currently being developed by OGC (OGC, 1998) in co-operation with ISO (ISO TC211, 2001). On the OGC meeting in April 2001, Liège, a request for proposal on this topic was issued (OGC (2), 2001). The request aims to extend the interfaces in the OpenGIS Simple Features Implementation Specification. The new interfaces will build on the OpenGIS Simple Features Specification to address feature collections and more complex objects and concepts including curves and surfaces in 2D and 3D, arcs and circle interpolations, conics, polynomial splines, topology and solids. The interfaces will cover creation, querying, modifying, translating, accessing, fusing, and transferring geospatial information.

## 4. CAPTURING THE WORLD IN A GEO-DBMS: DATA DEFINITIONS OF 2D AND 3D FEATURES

One of the aims of the use of geo-data is to capture the real world into computer models to get a better understanding of the world. The main part of this modelling process is to develop a model based on (relevant) objects in the real world. The Kadaster works with a data model for features in 2D (see Figure 5). This data model is maintained in a geo-DBMS.

The model that is currently used for 2D features will be extended with 3D features in case the situation requires this. For this, a data model is defined for 3D features. In the next subsections the currently used geometric model is described to represent 2D features (parcels), followed by considerations for maintaining 3D features in the DBMS.

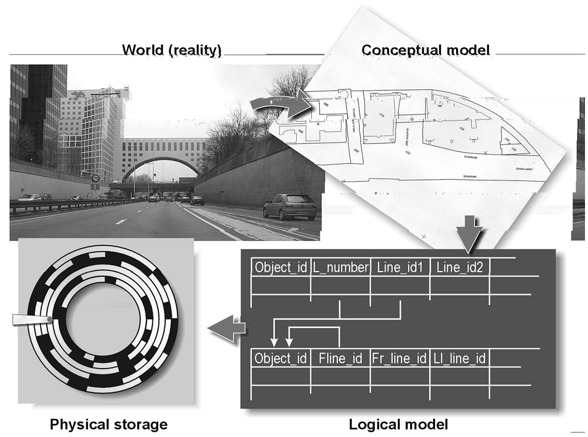



**Figure 5: capturing reality: from world to physical storage**

### 4.1 Spatial model for 2D features

The currently used data model for the cadastral system (for maintaining data on parcels as well as on the legal status of parcels) is described in Oosterom and Lemmen (2001). The spatial part of this data model will be described in the following. The spatial data are represented in the DBMS using geometric data types such as 'point', 'line' and 'box'. In addition to the use of these data types, explicit topology and historic information are stored.

The most important tables are boundary (cadastral boundaries) and parcel (parcel identifiers). The spatial extension of the objects in the tables boundary and parcel is indicated with a minimal bounding rectangle of type 'box'. The box covers a boundary or a complete parcel. The box can be spatially indexed and is useful for efficient retrieval purposes based on rectangle selections. There is no need for the geometric data type 'polygon', because the area features are stored topologically in the parcel table and boundary table using the so-called 'CHAIN-method'. The edges in the boundary table contain references to other edges according to the winged edge structure (Baumgart, 1975), which are used to form the complete boundary chains.

This approach allows calculations on correctness of topology after adjustment of the surveyed new boundaries to the cadastral data. Furthermore, it opens the possibility to relate attributes to the boundaries between parcels, e.g. relation to the source documents of surveying, date of survey, names of persons locating the boundary, etc. If each parcel would be represented in the DBMS by a closed polygon, it would be complicated to represent the basic object of cadastral surveying:

one boundary between two neighbour parcels. Closed polygon representation would lead to double (or triple or even more) storage of all co-ordinates (except the territorial boundary), which complicates the data management in a substantial way. Closed polygon representation can result in the introduction of gaps and overlaps between parcels, which is not related to reality. One more reason for the boundary based approach is in the classification of boundaries: the administrative cadastral and political subdivision in sections (cadastral zones), municipalities and provinces is possible by classifying boundaries as 'section-boundary', 'municipal-boundary', 'province-boundary' or 'national boundary'. A 'national boundary' is by definition a 'province boundary' and a 'municipal-boundary' and a 'section-boundary' and a 'parcel-boundary' etc.

A detailed overview of the parcel and boundary tables is given in Tables 1 and 2.

The following attributes are included in the data model for all spatial features:

- object_id, a nation-wide unique feature identifier for all objects represented in the DBMS;
- classif, classification code of the object, e.g. parcel boundary, parcel identifier, etc.;
- location (of data type point) or shape is of data type line(50), a polyline up to 50 points, representing the cadastral boundaries, stored in a variable length way in the (object)/relational database;
- sel_code, a selection code which indicates to which map type(s) a geographic object belongs, e.g. cadastral data and/or large scale topographic data;
- source of data, which is a reference to the field documents and files from total stations, or to the id of the photogrammetric project for large scale topographic mapping, etc;
- quality, which is the mode of data collection, e.g. terrestrial, photogrammetric and includes an accuracy code which denotes the deviation from the 'true' position;
- vis_code, visibility code to classify less visible objects during photogrammetric data collection, e.g. because one road lies on top of another road;
- l_area, official legal area, which is included in the official legal documents or deeds describing the transaction, in general this area is not equal to calculated area from the spatial cadastral boundary data; this attribute is introduced only for the parcel table.

| Attribute Name | Attribute description |
|---|---|
| Ogroup | Group Id (KEY.1) |
| Object_id | Text/polygon object id (KEY.2) |
| Slc | Spatial location code, index |
| Classif | Object classification |
| Location | x,y co-ordinate pair of centroïde (parcel id) |
| D_location | Delta x,y displacement of centroïde (parcel id) |
| Rotangle | Rotation angle of centroïde (parcel id) |
| O_area | Cadastral boundary based calculated parcel-area |
| Bbox | Bounding box covering the complete parcel |
| Object_dt | Date of object creation; equal to t_min in this special case |
| T_min | Date/time of parcel creation in the database |
| T_max | Date/time of parcel deletion in the database (KEY.3) |
| Sel_code | Selection code: object to be represented on cadastral and/or topographic data sets |
| Quality | Quality label: collection mode & accuracy |
| Vis_code | Visibility code; photogrammetric data collection |
| L_area | Official (legal) area of parcel |
| Municip | Municipality code, part of parcel identifier (ALT-KEY.1) |
| Section | Cadastral section code, part of parcel identifier (ALT-KEY.2) |
| Sheet | Sheet number of the original paper map, reference to paper archive |
| Parcel | Parcel id within cadastral section, part of parcel identifier (ALT-KEY.3) |
| Pp_i_ltr | Part-parcel letter, in use to identify part of parcel before surveying |
| Pp_i_nr | Part-parcel number, in use to identify part of parcel before surveying |
| L_num | Number of boundary references |
| Line_id1 | Reference to object_id of one surrounding boundary |
| Line_id2 | Reference to object_id of one boundary of the 1st enclave (if existing) |

**Table 1: Definition of the parcel table**

| Attribute Name | Attribute description |
|---|---|
| Ogroup | Group Id (KEY.1) |
| Object_id | Line object Id (KEY.2) |
| Slc | Spatial location code, index |
| Classif | Object classification |
| Interp_cd | Line interpolation code, line/arc |
| Shape | Line co-ordinate pairs, compressed |
| Fl_line_id | Object_id of first line to the left side in begin point -> BOUNDARY |
| Fr_line_id | Object_id of first line to the right side in begin point -> BOUNDARY |
| Ll_line_id | Object_id of first line to the left side in end point -> BOUNDARY |
| Lr_line_id | Object_id of first line to the right side in end point -> BOUNDARY |
| L_obj_id | Object_id of parcel on right side -> PARCEL |
| R_obj_id | Object_id of parcel on left side -> PARCEL |
| Bbox | Bounding box covering boundary between 2 parcels |
| Line_len | Length of boundary |
| Object_dt | Date of survey of the boundary in the terrain |
| T_min | Date and time of boundary creation in the database |
| T_max | Date and time of boundary deletion in the database (KEY.3) |
| Sel_code | Selection code: object to be represented on cadastral and/or topographic data sets |
| Source_cd | Source of data, reference to field sketch, photogrammetric project |
| Quality | Quality label: collection mode & accuracy |
| Vis_code | Visibility code; photogrammetric data collection |
| L_municip | Municipality code, part of the parcel id to the left side-> ALT-PARCEL.1 |
| L_section | Cadastral section code, part of the parcel id to the left side -> ALT-PARCEL.2 |
| L_sheet | Sheet number of the original paper map, part of the parcel id to the left side |
| L_parcel | Parcel id within cadastral section, part of the parcel id to the left side -> ALT-PARCEL.3 |
| R_municip | Municipality code, part of the parcel id to the right side -> ALT-PARCEL.1 |
| R_section | Cadastral section code, part of the parcel id to the right side -> ALT-PARCEL.2 |
| R_sheet | Sheet number of the original paper map, part of the parcel id to the right side |
| R_parcel | Parcel id within cadastral section, part of the parcel id to the right side -> ALT-PARCEL.3 |

**Table 2: Definition of the boundary table**

The text/label location-attribute in the parcel table is represented with the data type 'point'. A parcel has at least one reference to one of the surrounding boundaries and one reference to a boundary of each enclave. If a parcel has more than one enclave, then these additional references from parcel to boundary are stored in another table, called parcel_over, which can store up to 10 references. In case more references are needed, more parcel_over records are used. The structure of the topological references and the relationship between parcels and boundaries is visualised in Figure 6. The spatial database contains the geometric representation of the boundaries of about 7 million parcels (Oosterom and Lemmen, 2001).

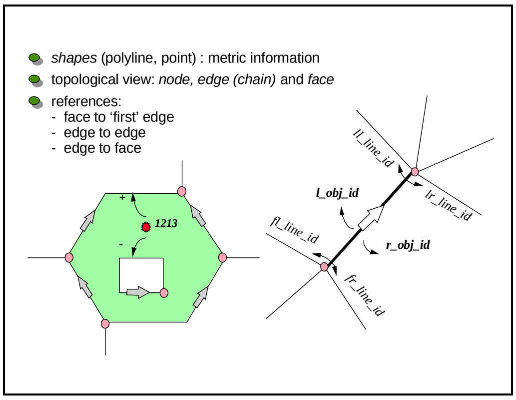



**Figure 6: topology model in the spatial DBMS of the Kadaster**

### 4.2 Spatial model for 3D Features

The data model for 2D features will be extended with a data model for 3D features. This data model will be used to represent the relevant spatial properties of real world objects.

Storing 3D co-ordinates and topology references is not a problem in current DBMSs. However, 3D models are not supported by DBMSs. They do not recognise the stored topology as such, nor can they perform a 3D spatial indexing or represent and manipulate spatial objects in 3D. Although, there are DBMSs that contain some or parts of the mentioned functionalities. However, the performance improvement using 3D spatial indexing as an extension of 2D indexing is expected to be not significant: the third dimension used by spatial data will be many times less selective than the used x,y space.

An experiment with Oracle showed that is possible to define 3D co-ordinates. However, geometry functions omit the z-value. In the following example, a spatial feature table (geom3d, with the attributes TAG and SDO_GEOMETRY) is created in which a 2D polygon together with a 3D polygon is inserted. After that, the geometrical functions AREA and LENGTH are performed on both polygons. A geometry validate is performed as well to show that the polygons are both valid.

As one sees in the results of the queries SDO_AREA and SDO_LENGTH return the same value for both polygons, although the 3D polygon actually has a greater area and length.

```
/*----------------*/
/* create table   */
/*----------------*/
create table geom3d (
 shape mdsys.sdo_geometry not null,
 TAG number(11) not null);

/*----------------*/
/* inserting data */
/*----------------*/

/* 66: a 2D polygon */
insert into geom3d (shape,TAG) values (
  mdsys.SDO_GEOMETRY(2003, NULL, NULL,
  mdsys.SDO_ELEM_INFO_ARRAY(1, 1003, 1),
  mdsys.SDO_ORDINATE_ARRAY(12,15, 15,15, 15,24,
  12,24, 12,15)), 66);

/* 88: a 3D polygon */
insert into geom3d (shape,TAG) values (
  mdsys.SDO_GEOMETRY(3003, NULL, NULL,
  mdsys.SDO_ELEM_INFO_ARRAY(1, 1003, 1),
  mdsys.SDO_ORDINATE_ARRAY(12,15,0, 15,15,0,
  15,24,999, 12,24,999, 12,15,0)), 88);

select TAG,
  SDO_GEOM.SDO_AREA(shape, 1) area,
  SDO_GEOM.SDO_LENGTH(shape, 1) length
from geom3d;

       TAG       AREA     LENGTH
---------- ---------- ----------
        66         27         24
        88         27         24

select SDO_GEOM.VALIDATE_GEOMETRY(shape, 1)
  geom_validate
from geom3d;

GEOM_VALIDATE
----------------------------------
TRUE
TRUE
```

A full functional support of 3D geographical objects within the DBMS will enable:

- to check correctness (after edit operations);
- to avoid redundancy;
- to maintain consistency;
- to facilitate complex operators (map overlay, split/merge operations);
- to use topology in spatial queries;
- to integrate (store and query) 2D data and 3D data;

The geometric primitives for the 3D cadastral data model are extrapolated from (in analogy with) the representation of geo-objects in the 2D cadastral data model: point, line and/or circular arc and polygon (described by a sequence of lines). Possible alternatives for this model are (decreasing in complexity):

- objects based on primitives available in 3D CAD models;
- polyhedrons and spheres/cylinders;
- polyhedrons;
- tetrahedrons.

Polyhedrons are solids, bounded by flat surfaces with each surface bounded by straight lines. Note that it may be difficult to describe a polyhedron correctly, as the four or more 3D points of a face do not always lie in a flat surface. A tetrahedron is a polyhedron with four triangular faces. The advantage is that the faces are always defined by three 3D points, which lie per definition in a flat surface (introducing no model error). For further explanation on tetrahedrons and tetrahedral networks see Kraak and Verbree (1992) and Pilouk (1996).

Since the Kadaster uses straight lines and circular arcs to represent spatial features in 2D the initial preference is to use these as the base in 3D. The 3D primitives will therefore firstly be composed of polyhedrons and spheres/cylinders. The bounding envelope of a 3D real world object will be defined and used to represent the geo-objects. Attributes of the geo-objects will also be stored. The data model to maintain and retrieve 3D objects will be implemented in two ways:

1. *using self containing 3D geometric data types*
   Now, z-values can be used to represent 3D features. However, as was seen, geometry functions do not recognise the z-value in Oracle. It is examined whether and how the relevant 3D information still can be recognised in spatial queries.
2. *using a topological model to maintain and retrieve 3D objects*
   A topological model for 3D objects is currently developed at our Department (Zlatanova and Verbree, 2000). A 3D geometry object is therein defined as a polyhedron consisting of nodes and faces. The edge is not used in this model, since this did not bring significant facilitation. The topological model is being implemented in Oracle.

## 5. COLLECTION AND PREPARING 3D DATA: INSERTING 3D DATA INTO THE 2D GEO-DBMS

3D spatial information of objects under and above the ground is available with designers, mostly as CAD models. Therefore, it is examined how this information can be used and what conversions and generalisations are needed to obtain the (spatial) relevant information, such as the outer boundary of the object. The alternative is to collect and measure 3D data, but this can be very difficult in the case of subsurface constructions. Aspects that should be looked at are:

- exchange formats;
- converting the used CAD primitives to meaningful and usable geo primitives fitting in the chosen 3D data model;
- converting the detailed file-based models to a collection of real-world objects containing relevant geo-information;
- converting the models defined in local co-ordinate systems to models defined in the national spatial reference system.

### 5.1 From local co-ordinates to national spatial reference system(s)

The transformation of a local co-ordinate system to a national reference system is object of special interest.

In the Netherlands, the horizontal location of a point can be defined in the national reference framework. The Netherlands' Kadaster maintains this framework (co-ordinate system), by means of 6,000 points for which the exact location is determined and preserved (Kadaster, 1999).

The vertical location of a point can be defined in respect to NAP: *Normaal Amsterdam Peil* (The Netherlands National Ordnance Datum). This is the tidal plane of reference used in the Netherlands to measure the altitude of land and structures. The NAP is at approximately the same height as mean sea level. It is visible in the landscape by about 55,000 marks. These marks are placed in buildings, bridges etc.. Almost everywhere in the Netherlands a mark can be found within one kilometre, which can be used for height determination. The Directorate-General of Public Works and Water Management is responsible for maintaining the height marks (2000).

### 5.2 Definition of 3D geo-data in respect to 2D geo-data

The insertion of the 3D data into the current (2D) geo-DBMS touches the fundamental issue of combining 2D and 3D data in one environment: what is the vertical relation between 2D and 3D data. In other words: what is the 'horizontal zero level' on which the 2D geo-objects are defined. With this, two possible representations of z-co-ordinates of the 3D geo-objects can be distinguished:

1. *an absolute z-co-ordinate, defined in the national reference system*
   When z-co-ordinates of the 3D geo-objects are stored in a national reference system, the height of surface parcels is needed to be able to define geometrical and topological relations between the 3D objects and the 2D parcels. The collection and input of this additional information will take considerable time. Moreover, the complexity of the 2D data increases, since 2D parcels need to be defined in 3D space. This can not be done by simply adding one z-co-ordinate per parcel, since parcels contain too much spatial variance for this approach (even in a 'flat' country like the Netherlands).
2. *a relative z-co-ordinate, defined in relation to the surface*
   When z-co-ordinates of the 3D geo-objects are stored in respect to the surface, the current geo-DBMS does not to need to be extended with additional z–information on current 2D geo-data, saving time and data complexity. On the other hand, other difficulties arise, such as the conversion of the z-co-ordinates of the 3D geo-objects, known in the national spatial reference system, to relative co-ordinates. In this case only the 3D situation in the surrounding of the 3D geo-object needs to be explored, instead of locating all 2D geo-data in 3D space.

## 6. QUERYING AND VIEWING

The incorporation of 3D geo-objects into the existing geo-DBMS based on 2D parcels makes it possible to query the data 2D, 3D and in combination with each other. A query example from the previous section which shows the additional value of the integration of 2D data and 3D data in one DBMS, is:

> find all 2D objects which 'intersect' with (the projection) of a certain 3D object

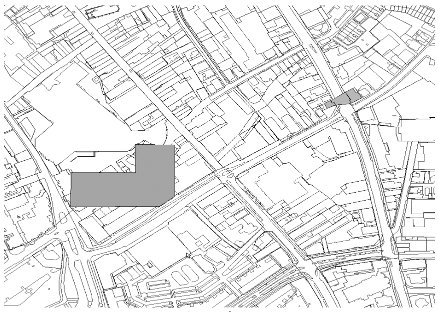
From a DBMS point of view, the relations between 2D parcels and 3D constructions should be derived by the spatial system.

As the current cadastral model is based on a topology structure, which is stored in the DBMS, but not managed by the DBMS, it is not possible to select the parcels which are overlapped by an object from our geom3d table with a SQL query. For this purpose, the parcels should be 'materialised' first to polygons and with these polygons the actual overlap computation can be performed. As the first query below shows, it is possible to select parcels based on (projected) overlap of their bounding box with the 2D and 3D objects from the geom3d table, introduced in section 4.2. It is also possible to select the parcel boundaries based on overlap with the objects from the geom3d table.

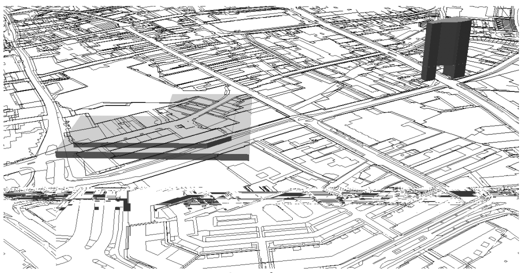
```
/* parcel selection based on bbox */
select distinct municip, section, parcel
from parcel p, geom3d g3d
where sdo_geom.relate
  (p.bbox,'anyinteract',g3d.shape,.1)='TRUE';

/* boundary selection based on true shape */
select b.shape
from b.shape, geom3d g3d
where sdo_geom.relate
  (b.shape,'anyinteract',g3d.shape,.1)='TRUE';
```

An inventory is made of all required queries, which can benefit from the existence of 2D and 3D data in one geo-DBMS. These queries will be implemented.



*2D view*



*3D view*

**Figure 7: visualising 2D and 3D data in one environment**

To access the data and to view the results of queries a viewer will be built on top of the geo-DBMS. It will be examined how the data can be viewed administratively, as well as two-dimensionally (traditional GIS interface) and three-dimensionally (perspective, stereo) and in combination with each other in one environment (see Figure 7). Possible techniques, which will be looked at, are 3D Java and (geo)VRML.

## 7. CONCLUSION AND FUTURE WORK

In science and in applications, there is a tendency to model the world in multi-dimensions. However, it will take considerable time until geo-data and applications are capable to model the world (fully) in 3D. Moreover, a 2D approach still suffices in many cases. It is therefore relevant to look at a combination of 2D spatial data and 3D spatial data in one DBMS-environment.

At the Department of Geodesy of the Delft University of Technology a research has been started on this topic. This article described concepts developed so far concerning the incorporation of 3D geo-objects in an existing 2D geo-DBMS and the translation of these concepts into prototype implementations. This means data definition, data manipulation and data querying of 2D and 3D data in one environment.

The starting point is to represent and maintain spatial features within a conventional DBMS instead of handling these outside the DBMS (in GISs). The support of spatial data types in a geo-DBMS include:
- spatial operators (or geometry functions);
- spatial indexing;
- spatial clustering;
- topology management.

The data model that is currently used to represent 2D features is being extended with a 3D model extension. Possible data models studied in this research to represent 3D geo-objects are:
- using self containing 3D geometric data types;
- using a topological model to maintain and retrieve 3D objects.

Future work will focus on obtaining the required geo-data to represent 3D real world objects in the DBMS. This is the relevant data needed to store the geo-objects in the DBMS and will consist of the bounding envelope of the object, defined in a national spatial reference system.

Further research is carried out to query and view 2D and 3D data in one environment. The required queries are being defined and will be implemented in the geo-DBMS.

Finally, this research will lead to the implementation of all these aspects in one system: a geo-DBMS supporting 2D geo-objects and 3D geo-objects as well as in combination with each other.

## REFERENCES

Bruce G. Baumgart (1975). *A polyhedron representation for computer vision*. In National Computer Conference, pages 589-596, 1975.

*Civil Code (Burgelijk Wetboek)* (1992): Boek 5: Zakelijke Rechten, 1992. (In Dutch.)

Directorate-General of Public Works and Water Management (2000), T*he Amsterdam Ordnance Datum (NAP),* flyer published by the Survey Department, March 2000.

FIG (1995). Fédération International des Géomètres (1995): *The FIG Statement on the Cadastre*, Publication No. 11, 1995.

ISO TC 211 (2001), ISO/DIS 19107: *Geographic information - Spatial schema. Final text of CD 19107, International Organization for Standardization*, January 2001.

Kadaster (1999), *Het kadaster en de Rijksdriehoeksmeting,* flyer published by the Netherlands' Kadaster. (In Dutch)

Kap A.P. and J.A. Zevenbergen (2000): *Valkuilen en kansen bij de opzet van landelijke registraties: een (inter)nationale vergelijking, Technical Report*, Department of Geodesy, Delft University of Technology, 2000. (In Dutch.)

Kraak, MJ, and E. Verbree (1992*), Tetrahedrons and Animated Maps in 2D and 3D Space*, In: Proceedings 5th Int. Symp. on Spatial Data Handling, Charleston, USA, 1992, pp. 63-71.

Lemmen, C.H.J., E.P. Oosterbroek and P.M.J. van Oosterom (1998): *New spatial data management developments in the Netherlands Cadastre*, proceedings of the FIG XXI International Congress, Brighton UK, Commission 3, Land Information Systems, pp. 398-409, 19-25 July 1998.

OGC (edited by Buhler, K. McKee, L.) (1998): *The OpenGIS Guide, third edition, An introduction to Interoperable Geo-processing*, The OGC Project Technical Committee of the OpenGIS Consortium; Wayland, Mass., VS.

OGC (1999), *OpenGIS Simple Features Specification for SQL*, revision 1.1, OpenGIS Project Document 99-049.

OGC (2001), Spatial Schema, author: J. Herring, OpenGIS Project Document, 00-063.

OGC (2) (2001), OGC request 12: Geometry Working Group, a request for proposals: OGC Feature Geometry.

Oosterom, P.J.M. van and C.H.J. Lemmen (2001), *Spatial data management on a very large cadastral database*, Computers, Environments and Urban Systems (CEUS). Volume 25, number 4-5, 2001.

Oosterom, P.J.M. van, E. Verbree and A. Kap (2000*), Storing and Manipulating Simple and Complex Features in Database Management Systems*. In: Proceedings of the 3rd AGILE Conference on Geographic Information Science, Helsinki/Espoo, Finland, pp. 178-182, 25-27 May 2000.

Oracle (1999) *Oracle Spatial, User's Guide and Reference,* Release 8.1.6, December 1999, Part No. A77132-01.

Pilouk, M. (1996), *Integrated Modelling for 3D GIS*, Ph.D. Dissertation, March 19, 1996, Enschede, ITC.

Stoter, J.E. and P.J.M. van Oosterom (2000): *3D registration of real estate objects*, In: Proceedings of GIScience, Savannah, USA, pp. 55-58, 28-31 October, 2000.

Stoter, J.E. (2000): *Considerations for a 3D Cadastre, Delft University of Technology, GISt No. 2,* Report to the Netherlands' Kadaster, Delft, October 2000, 30 p.

Zlatanova, S. and E. Verbree (2000), *a 3D topological model for augmented reality*, In: Proceedings Second International Symposium on mobile multimedia systems and applications, November 9-10, 2000, TU Delft, the Netherlands.