# WEB-BASED DYNAMIC VISUALIZATION OF 3D SPATIAL DATA

**Alias Abdul Rahman** and **Adnan Younas**

Institute of Geospatial Science and Technology,
and
Department of Geoinformatics,
Faculty of Geoinformation Science and Engineering,
Universiti Teknologi Malaysia
81310 UTM Skudai, Johor, Malaysia
{alias,adnany}@fksg.utm.my

## Abstract

True 3D GIS requires lot of efforts from GIS research community both in academia as well as from software vendors. Visualizing 3D spatial data from DBMS for such 3D system is one of the issues and problems in 3D GIS domain. This paper discusses the dynamic visualization of 3D spatial data such as buildings and other large objects using geo-DBMS couples with Web that works with VRML and X3D modeling languages. The modeling of the 3D objects are based on the Simplified Data Model (SDM). The experiment will be presented by using digital 3D building datasets and the approach is applicable for tasks like 3D strata title and buildings management. Finally, it discusses the limitations and shortcomings of the current approach, and identifies future research and development tasks.

*Key words*: 3D spatial data, VRML, X3D, visualization, and GeoDBMS.

## 1. Introduction

Managing and manipulating 3D spatial data and eventually 3D spatial objects from database require specialized tools. One of the tools is visualization. Visualization via Web technology will be a plus point for data provider as well as for end users. Efficient and appealing presentation of 3D geospatial data on the Internet presents a challenging task for ongoing research on geo-visualization (Madden and Schiewe, 2003). The growing number of complexly structured data collections accessible over the global information network increasingly requires the support of appropriate visualization methods. Visualization is needed to support information retrieval by browsing and searching (Hemmje, Kunkel and Willett, 1994; Massari et. al., 1998), as well as data mining tasks for naive end users.

Visualization of geographical information has been termed as ‘geo-visualization’. In comparison, ‘information visualization’ applications involve, for example, data mining applications, and whereas ‘scientific visualization’ involves the rendering of scientific results and research. Those involved with the perception of visualizations often speak about ‘cognitive visualization’ where, those factors related to human perception and ability to identify objects visually - are studied and considered.

Apart from the need for an integrated DBMS, there is also a trend of modeling the world in three dimensions. However, GIS applications, which use true 3D functionalities other than 3D visualizations are still rare. GIS and DBMS vendors seem to hesitate with full support for 3D. It seems that users are waiting for vendors to make the step to 3D, while vendors are waiting for an extensive request for 3D tools from their customers. In this respect, research into new 3D solutions can play a critical role (Stoter and Zlatanova 2003a).

Three-dimensional visualization systems are applicable to various application domains. One example is the visualization of large data collections within information systems (Rorvig and Hemmje, 1998). Another important application area is electronic commerce. Three dimensional visualizations are a powerful means to increase the cognitive effectiveness of virtual shopping malls, virtual libraries, or virtual galleries for naive end users. Information systems to support large events, like trade fairs or congresses, can benefit from three-dimensional visualizations (Risse and Hemmje, 1997). A virtual reality overview of a building, e.g., display information about all apartments in a building also includes the detailed information about the apartments. Visitors can find the specific apartment with its ID and with the person name to which it belongs.

To explore current possibilities, we built a prototype using Web technology. The used tools and techniques are from outside the GIS domain. In this research the potentials are studied of using these tools and techniques in GIS and geoinformation science. The prototype was built on a Oracle 10g, with PHP and VRML/X3D. It uses VRML for visualization. The aim of the prototypes is to show the possibilities and constraints of accessing 3D geo-information organized in a geo-DBMS, in order to contribute to 3D GIS developments. Related research can be found in (Nebiker, 2003; Belfore, 2002 and Luttermann and Grauer, 1999).

This paper begins with a short description on how 3D geo-objects can be modeled in a DBMS. These data models (a topological model and a geometrical model) were used in our prototype. Section 3 contains a short description on VRML and X3D, the 3D formats used in the prototypes. In section 4 the prototypes are described, together with their functionalities. Finally, the paper provides some remarks on the findings and highlights the outlook of the prototype and some future works.

## 2. 3D Geo-Objects in DBMS

GIS have evolved to spatial DBMS where spatial as well as non-spatial data are maintained in one integrated environment. To achieve this, mainstream DBMS like (Oracle, 2001; IBM, 2000; Informix ,2000; Ingress, 1994) have implemented spatial data types and spatial operators more or less similar to the OpenGIS Consortium Simple Features Specification for SQL (OGC,1999). The implementation consists of an SQL extension that supports storage, retrieval, query and update of simple spatial features (points, lines and polygons). OGC specifications (OGC, 2003) are until now 2D, as are the implementations of spatial data types in the mainstream DBMS. The only 3D functionalities in 3D that are available in DBMS are length and perimeter in 3D (PostGIS, 2003; MapInfo, 2003) and spatial indexing in 3D, e.g. 3D R-tree in Oracle. Although current DBMS do not support 3D objects, however z-coordinates can be used to store 0D, 1D and 2D objects in 3D.

Stoter and Zlatanova (2003b) describe how 3D geo-objects can geometrically be maintained in DBMS within current techniques. Arens *et al* (2003) have implemented support for a 3D primitive (polyhedron) in a DBMS. This implementation is based on the proposal of (Stoter and Oosterom, 2002). In (PostGIS, 2003) it is also described a topological implementation of 3D geo-objects in DBMS based on the Simplified Spatial Model (Zlatanova, 2000). The prototype presented in this paper is based on these DBMS model, which is implemented in Oracle Spatial 10g (Adnan *et. al.*, 2005).

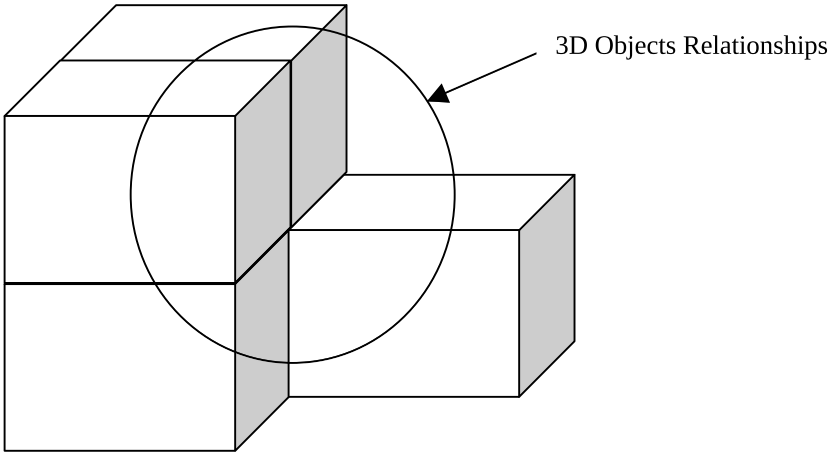


**Figure 1:** Complex 3D geo-object and its relationships

In Figure 1, a complex 3D object with different 3D sub-objects is shown. Also relationships among the different complex 3D sub-objects have been mentioned. When we need to store this 3D object into a spatial database management system there is a spatial object inside the database management system, we have to map its data with that. The detail of the Geometry object and the storage structure and information of a 3D object inside the database management is shown in the next section.

### 2.1 Schema of a spatial object

Conventional relational database systems do not provide the functionality to directly support VRML/X3D scene data. Object-relational and object-oriented database management systems have been developed to support the extension by new data types, as they are required for new application domains, like multimedia or document handling. They provide the necessary mechanisms to support the modeling of VRML/X3D data types. Since most business data including GIS data are stored in relational DBMS we have chosen an object-relational DBMS as been widely acknowledged as the new generation relational DBMS (Stonebraker, 1996) for our development platform.

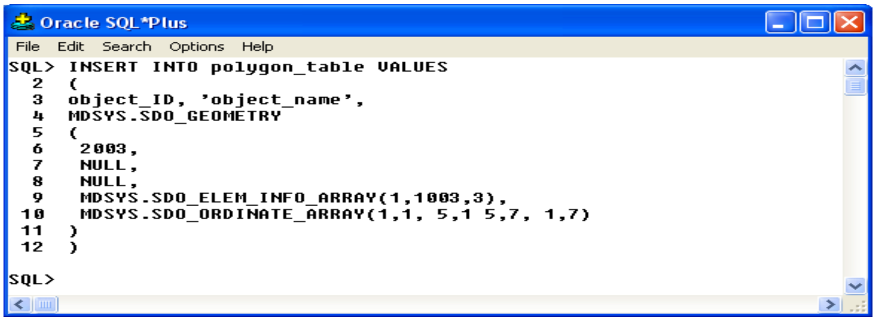
In order to visualize geometrical objects, geometry column for each of the spatial primitives, i.e. node, line, face, and solid are created. The purpose of preparing these geometry columns is to integrate the spatial database into Map 3D for the purpose of displaying objects graphically.

A geometry column consists of geometry object that represent a spatial feature, modeled as an ordered set of primitive elements, i.e. node, line, and polygon. In Oracle Spatial, the geometry column consists of a schema (MDSYS) that prescribes the storage, syntax, and semantics of supported geometric data types. The pre-defined object type

SDO_GEOMETRY is:

```
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_STARTING_OFFSET, SDO_ETYPE, SDO_INTERPRETATION),
SDO_ORDINATES MDSYS.SDO_ORDINATE_ARRAY)
   (list of ordinates);
```

For example, see Figure 2:



```
SQL> INSERT INTO polygon_table VALUES
     (
     object_ID, 'object_name',
     MDSYS.SDO_GEOMETRY
     (
      2003,
      NULL,
      NULL,
      MDSYS.SDO_ELEM_INFO_ARRAY(1,1003,3),
      MDSYS.SDO_ORDINATE_ARRAY(1,1, 5,1 5,7, 1,7)
     )
     )

SQL>
```

**Figure 2:** Data set insertion into Polygon table

The core component to support VRML/X3D in this implementation of a VRML/X3D "Geometry" data type of Oracle spatial is used. A "Geometry" data type has been used to store VRML/X3D data as a complete and valid VRML/X3D scenes. Spatial "geometry" data type is also used for import and export of predefined scenes as well as transformations of the complete scene in to the database.

## 3. VRML and X3D

In 1994 the Web3D Consortium launched VRML (Virtual Modeling Language), which became an international ISO standard in 1997. The basis for the development of VRML was to have a simple exchange format for 3D information. This format is based on the most used semantics of modern 3D applications - hierarchical transformations, illumination models, viewpoints, geometry, fog, animation, material characteristics and texture.

The development of VRML has stopped since the Web3D Consortium started to work on a XML version of VRML, in order to integrate with other Web technologies and tools i.e., X3D (eXtensible 3D). The specifications of X3D have only recently become available on May, 2003. In our experiment we have used both X3D and VRML to visualize 3D geo-objects. The data structure of an X3D document is very much comparable to the data structure of a VRML file. So as far as the underlying data model is concerned, X3D must be seen as a subset of VRML (Web3D, 2003). The difference lies in the notation (the syntax) used. While VRML is text, with accolades for structuring, X3D is coded in XML, with 'tags' for structuring. This is a major advantage for on-the-fly retrieval, because of the ease of use of XML in Internet applications.

### 3.1 PHP and VRML/X3D Integration

PHP is becoming very popular language for creating dynamic websites, particularly for generating them from databases. However, 2D GIS are outdated. So what we want is a way to harness the power of PHP to create database-driven VRML worlds for new 3D GIS system.

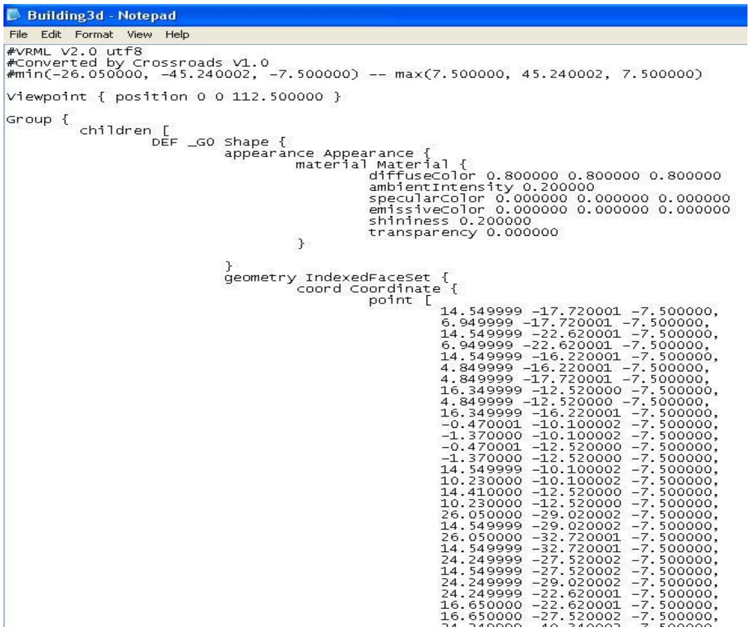
PHP can output any text information; the only thing that identifies the file type is the MIME type that gets sent by the web server before the file itself. Fortunately, in PHP, one has to take control of this very information. This requires sending the VRML MIME type (“model/vrml”), and then writing the appropriate VRML nodes.

The server strips all PHP code when sending a response. So, on lines where only JSP code is present, the server simply sends blank lines back to the browser.

It is necessary to include both PHP and VRML headers, and the content type must be changed before we set the VRML header, so the final result looks like this:

```
<?php
  header ("Content-type: model/vrml");
  echo "#VRML V2.0 utf8\n";
?>
```

In this experiment, one of the objectives is to display a 3D building in VRML code using database. But in the start, it has been tested with the same type of data as used in Oracle Spatial to display a VRML file. Figure 3 shows the VRML file contains data in itself.



```
#VRML V2.0 utf8
#Converted by Crossroads v1.0
#min(-26.050000, -45.240002, -7.500000) -- max(7.500000, 45.240002, 7.500000)

Viewpoint { position 0 0 112.500000 }

Group {
        children [
                DEF _GO Shape {
                        appearance Appearance {
                                material Material {
                                        diffuseColor 0.800000 0.800000 0.800000
                                        ambientIntensity 0.200000
                                        specularColor 0.000000 0.000000 0.000000
                                        emissiveColor 0.000000 0.000000 0.000000
                                        shininess 0.200000
                                        transparency 0.000000
                                }

                        }
                        geometry IndexedFaceSet {
                                coord Coordinate {
                                        point [
                                                14.549999 -17.720001 -7.500000,
                                                6.949999 -17.720001 -7.500000,
                                                14.549999 -22.620001 -7.500000,
                                                6.949999 -22.620001 -7.500000,
                                                14.549999 -16.220001 -7.500000,
                                                4.849999 -16.220001 -7.500000,
                                                4.849999 -17.720001 -7.500000,
                                                16.349999 -12.520000 -7.500000,
                                                4.849999 -12.520000 -7.500000,
                                                16.349999 -16.220001 -7.500000,
                                                -0.470001 -10.100002 -7.500000,
                                                -1.370000 -10.100002 -7.500000,
                                                -0.470001 -12.520000 -7.500000,
                                                -1.370000 -12.520000 -7.500000,
                                                14.549999 -10.100002 -7.500000,
                                                10.230000 -10.100002 -7.500000,
                                                14.410000 -12.520000 -7.500000,
                                                10.230000 -12.520000 -7.500000,
                                                26.050000 -29.020002 -7.500000,
                                                14.549999 -29.020002 -7.500000,
                                                26.050000 -32.720001 -7.500000,
                                                14.549999 -32.720001 -7.500000,
                                                24.249999 -27.520002 -7.500000,
                                                14.549999 -27.520002 -7.500000,
                                                24.249999 -29.020002 -7.500000,
                                                24.249999 -22.620001 -7.500000,
                                                16.650000 -22.620001 -7.500000,
                                                16.650000 -27.520002 -7.500000,
```

**Figure 3:** 3D Building’s VRML File

PHP scripting is added to this shape, which lets us use dynamic data to change the sphere’s position in space (translation X Y Z), its color (*diffuseColor R G B*), and its radius.

## 4. Prototype and Process Details

The basic idea of the prototype is to organize 3D geo-objects in a DBMS and to query them via an Internet browser. Geo-objects contain both spatial and non-spatial (administrative) information. The spatial information can be visualized after conversion into VRML or X3D and the non-spatial attribute information can be presented in (dynamic) HTML pages.

Figure 4 shows a standard request process—the page is requested via a browser. The request calls the designated PHP, which interacts with a database. The model given below explains the whole prototype process.

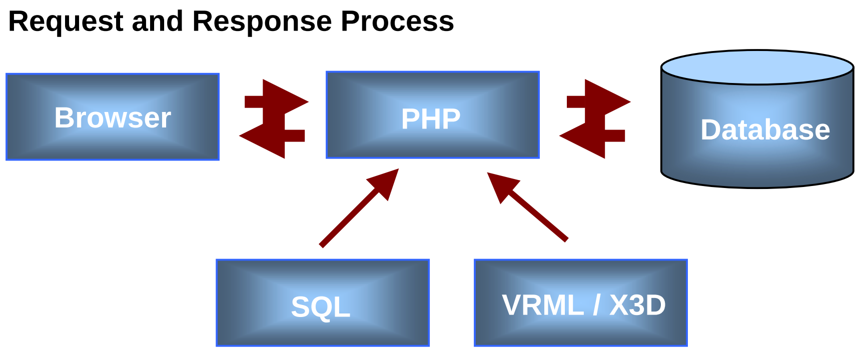


**Figure 4:** The Request and Response

After receiving the response, the system follows the flow as shown in Figure 3. The database sends the requested data to the PHP, which formats the data and sends the response to the requesting browser. In our case study, the data is returned to the PHP, which then generates a VRML scene using the data from the database.

On a client request a connection is made to the DBMS and the spatial information of interest is selected from the DBMS and converted into X3D/VRML. A browser plug-in at the client side makes it possible to view the VRML or X3D output. VRML and X3D provide the possibility to start a script when a user clicks on an object. This functionality is used to retrieve the non-spatial information that is linked to a 3D geo-object. Via the VRML/X3D plug-in a request is sent to a (application) server. The server receives and interprets the incoming information and sends a HTML with the required information back to the browser.

For retrieving the spatial and the non-spatial information from the DBMS a technique is needed to communicate between a client and a database on a server. For this communication, several techniques are available such as ColdFusion, ASP.NET, ASP, JSP or PHP. The choice of the used technique is dependent on the used Web server. The detailed architecture of publishing a 3D dataset on the Web is shown in Figure 5 below:

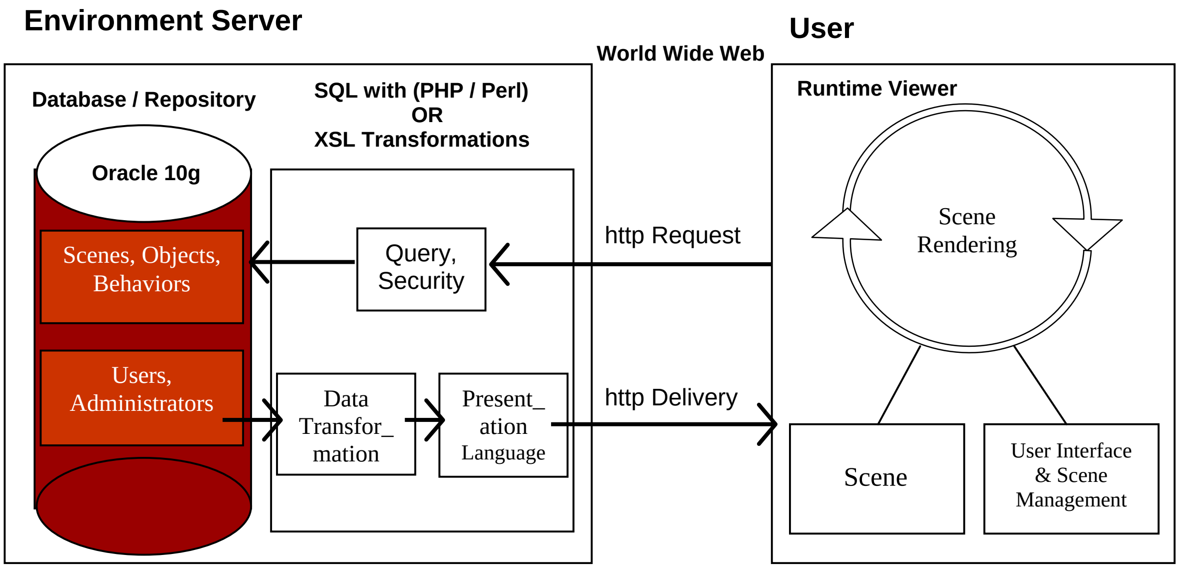


**Figure 5:** Web publishing architecture of 3D data sets using 3D-Geo DBMS

To show the possibilities to query 3D geo-objects via an Internet client, first a simple prototype was built, and it is based on Microsoft technology.

## 5. An Approach Towards Supporting Dynamic Scene Generation

Today, the dynamic generation of HTML pages is a standard functionality of all commercial database systems. This feature has been proven to be a very effective and practical approach to support the (two-dimensional) visualization of information stored within database systems. In this paper, we outline how a similar functionality can be realized to dynamically generate VRML scenes from a database management system (DBMS). This approach overcomes many of the limitations of static VRML scenes, by exploiting the persistence, scalability and security mechanisms of database management systems. In addition, it also provides a direct way to efficiently generate three dimensional visualizations from existing information in the database.

In the experiment as shown here, a "Geometry" data type allows, for example, to store all apartments of a building as VRML scenes together with the walls and its floor information along with the its ID. It is possible to select a subset of all apartments with certain properties and merge them in a new scene, e.g., in order to display all apartments that have already been rented to customers.

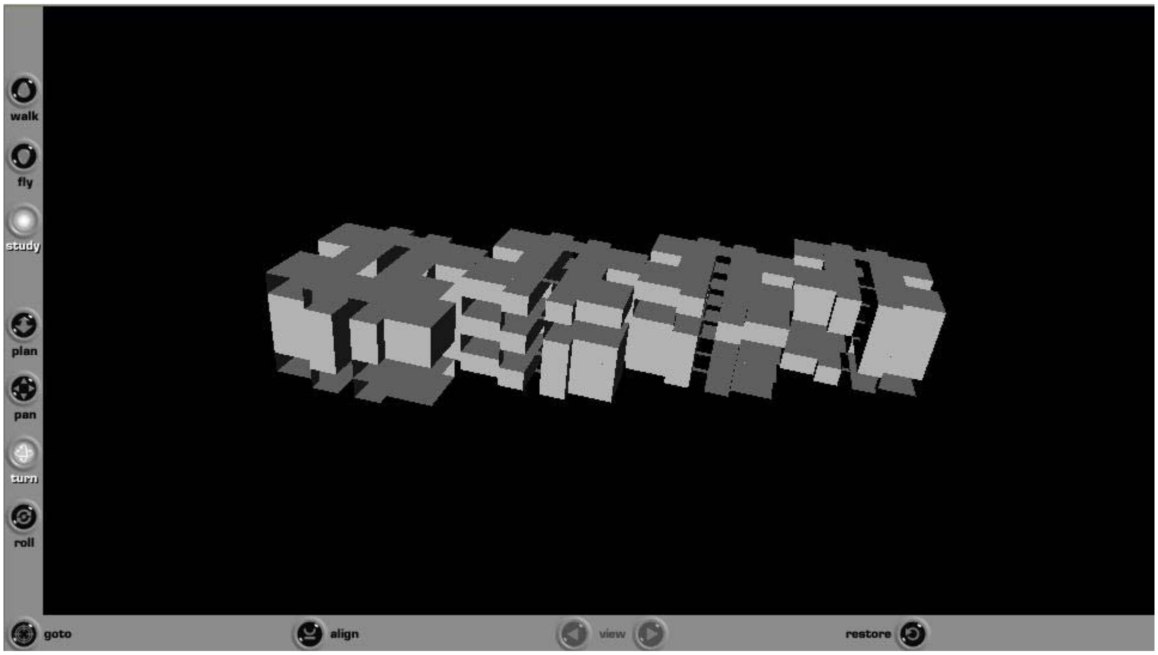


**Figure 6:** 3D dynamic data visualization of 3D building data on Web

In Figure 5, a complete building structure is shown using the VRML client application installed as a plug-in with Web browser. Figure 6 shows that how complex 3D data is and to generate the dynamic scene through database onto the Web browser was using the detailed architecture as shown above in the Figure 5.

Dynamic scene generation in VRML/X3D using the data from the Oracle Spatial 10g could be done in a way given below:

A database connection is used as a statement inside PHP coding. Here are the strings used to connect to the Oracle database:

```
<?
$connection = OCILogon ("User Name", "Password", "service name")
                     or die ("cannot connect to database");
?>
```

The SQL in this case is quite simple—return all data contained in the table after execution:

```
<?
   $stmt = OCIParse($connection, "select * from building" )
               or die ("cannot select");
   OCIExecute($stmt, OCI_DEFAULT);
?>
```

Once the result is assembled, the code loops through all of the records and displays as many Floors as there are records in the database. This is the loop:

```
<?
$count = 0;
While (OCIFetch ($stmt)) {
color = ociresult ($s, "color");
radius = ociresult ($s, "radius");
?>

DEF Floor <? $count ?> Transform {
translation <? $count*15 ?> 0 0
children [
Shape {
appearance Appearance {
material Material {
diffuseColor <? $color ?>}
}
geometry Sphere {
```

```
radius <?$radius?>
} } ] }
<?
count++;
}
```

Notice how the values from the database are inserted into the VRML. Two strings, color and radius, are set to the values from the database and then displayed in the VRML code as *<? $color ?> and <? $radius ?>*.

The translation is handled by an integer (count), which keeps track of the number of records and displaces the floor by 15 on the x-axis every iteration of the loop.

The variable count has another use. As we generate objects, we give them a name: DEF Floor1, DEF Floor2, and so on. We do this by inserting the count into the VRML node's definition DEF Floor <? $count ?>

A DBMS supporting a three dimensional visualization must be able to generate new VRML scenes, both from existing operational business data represented by conventional data types and from existing multimedia data represented by specialized media data types. Within the Building, statistical data about how many persons can live in one apartment could be visualized by an arrow diagram, where the arrow size is proportional to the number of family members allowed to live in an apartment. As always, free the resources database resources at the end.

```
<?
   OCIFreeStatement ($stmt);
    OCILogoff ($connection);
?>
```

The VRML client has to be able to directly read and write the DBMS from within a VRML scene. For more advanced interaction modes in multi-user environments, this mechanism needs to be complemented by an event handling system. This allows signaling a change in a scene to all other users actively working on the same scene. The VRML event handling of the other users can then react by appropriately updating the scene, e.g. by reloading a part or the whole scene.

The most important benefit of the above-outlined approach is that by means of storing VRML scenes within a DBMS, we achieve persistence of changes to scenes. Furthermore, the multi-user access control enables the sharing of VRML data among multiple users, thus we move from isolated, static scenes to shared spaces of dynamically generated three-dimensionally visualized information. Scalability is achieved by loading and generating scenes and scene components dynamically either at loading time or at run time. The corresponding loading/generation schemes can be determined both by physical characteristics of the VRML scene and the logical structure of the application. Controlled access to scene data is supported by the security and view mechanisms of the underlying DBMS. For example, in the "Building" scenario, one might display the rent of an apartment only to authorized members or customers but not to visitors.

## 6. Conclusion and Future Works

Geo-DBMS have become a major topic in geo-science. Spatial data types for 2D are currently available in mainstream DBMS and now geo-science is focusing and supporting for 3D, 4D and topology in DBMS. The next step is to make the data organized in a geo-DBMS available within and between organizations and for individual users via Web technology.

This paper contributes to these developments, since it illustrates how 3D visualization techniques and techniques to query DBMS via a Web server can be combined. We have shown how 3D geometry stored in an Oracle database can be converted into VRML or X3D, and how the 3D objects can be presented in a 'simple' Internet browser together with their non-spatial attribute information. Future work will focus on a number of research issues.

We have discussed some of the VRML/X3D and Geo-DBMS features which we used for animation and user interaction with the rendered 3D model. With our approach, we have achieved the most of the requirements for dynamic information visualization support we have identified. Security and persistence are provided as a basic property of the DBMS.

Although experience with the prototype looks promising with respect to performance, serious tests on larger data sets will be set up. Fast rendering of 3D objects is of course critical when displaying data via the Internet. One option is the use of compression techniques. Other strategies have to do with navigation and zoom-levels. First data with low Level of Detail (LoD) can be presented, and only when the user zooms in the Level of Detail is increased (Kofler, 1998). This touches the fundamental research issue of storing multi-representations in DBMS or performing generalizations on the fly.

Another issue is that of visualization constraints. X3D and VRML environments were originally meant to visualize all kinds of objects. Therefore, constraints should be implemented in the prototypes to meet specific conditions for visualizing geo-information, e.g. one should not be able to turn the data set upside down.

The prototype described in this paper form a contribution to accessing 3D geo-objects organized in a DBMS. Our experiences show potentials for the use of common Web technology in GIS and more specific in 3D GIS applications, which should be further elaborated in our future research work.

## References

Adnan Younas, Chen Tet Khuan, Alias Abdul Rahman, and Siyka Zlatanova (2005), “3D Topological Spatial Data Modeling And Visualization in Geo-DBMS”, International Symposium & Exhibition on Geoinformation 2005, Penang, Malaysia, 27 – 29 September 2005

Arens, C., J.E. Stoter and P.J.M. van Oosterom (2003), Modelling 3D spatial objects in a GEO-DBMS using a 3D primitive, AGILE conference, April 2003, Lyon, France

Belfore, L.A. (2002), An architecture supporting live updates and dynamic content in VRML based virtual worlds. Symposium on Military, Government and Aerospace Simulation 2002 (MGA 2002), San Diego, California, April 2002, 138-143.

Codd, E.F. (1970), A Relational Model for Large Shared Data Banks. In: Communications of the ACM, Vol. 13, No.6, Seite 377-387, Juni 1970.

Hemmje, M.; Kunkel, C.; Willett, A. (1994), LyberWorld -A Visualization User Interface Supporting Fulltext Retrieval. In: Proceedings of ACM SIGIR '94, July 3-6, Dublin, 1994.

IBM (2000), IBM DB2 Spatial Extender User's Guide and Reference. Special Web release edition.

Informix (2000), Informix Spatial DataBlade Module User's Guide. http://www-3.ibm.com/software/data/informix/pubs/library/dbm.html

Ingres (1994), INGRES/Object Management Extension User's Guide, Release 6.5 (1994). CAOpenIngres.

Kofler, M. (1998), R-trees for the visualisation of large 3D GIS Database, Ph.D. thesis, Technical University, Graz, Austria, 1998.

Luttermann, H. and M. Grauer (1999), Using interactive, temporal visualizations for WWW-based presentation and exploration of spatio-temporal data. Spatio-Temporal Database Management 1999, 100- 118.

Madden, M. and J. Schiewe, “Theme issue: challenges in geospatial analysis and visualization”, ISPRS Journal of Photogrammetry and Remote Sensing (special issue), Vol. 57(5-6), pp. 301-303, 2003.

MapInfo (2003), http://www.mapinfo.com

Massari, A., Saladini, L., Sisinni, F., Napolitano, W., Hemmje, M., Paradiso, A., Leissler, M. (1998), Virtual Reality Systems For Browsing Multimedia. To appear in: Furth, B. (ed.): Handbook of Multimedia Computing

Nebiker, S. (2003), Support for visualisation and animation in a scalable 3D GIS environment -motivation, concepts and implementation. ISPRS Commission V Working Group 6, Workshop on "Visualization and Animation of Reality-based 3D Models", February 2003, Engadin, Switzerland.

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

OGC (2003), OpenGIS specifications, http://www.opengis.org/techno/specs.htm

Oracle (2001), Oracle Spatial User's Guide and Reference Release 9.0.1 Part Number A88805-01, June 2001

PostGIS (2003), http://postgis.refractions.net

Risse, T., Hemmje, M., (1997), Entwurf und Implementierung einer VRML Bibliothek für ein objektrelationales Datenbanksystem auf der Basis des VRML 2.0 Standards.In: GMD-Studien, Nr. 324, September 1997, GMD – Forschungszentrum Informationstechnik GmbH, Sankt Augustin.

Rorvig, M., Hemmje, M. (1998), Foundations of Advanced Information Visualization for Visual Information (Retrieval) Systems. To appear in: Perspectives on Visual Information Retrieval Interfaces. Journal of the American Society for Information Science, 1998

Stonebraker, M. (1996), Object-relational DBMSs: the next great wave**;** 1996 Morgan Kaufmann Publishers, Inc., San Francisco.

Stoter, J.E. and P.J.M. van Oosterom (2002), Incorporating 3D geo-objects into a 2D geo-DBMS. In: XXII FIG International Congress and the ACSM-ASPRS Conference and Technology Exhibition, April 19-26 2002, Washington D.C., USA, 12 p. CD-rom.

Stoter, J.E. and S. Zlatanova (2003a), 3D GIS, where are we standing?, Paper submitted to ISPRS workshop on spatial and temporal data modelling, Quebec City, Canada, October, 2003.

Stoter, J.E. and S. Zlatanova (2003b), Visualising and editing of 3D objects organised in a DBMS, EUROSDR workshop: Rendering and visualisation, January 2003, Enschede, The Netherlands.

Thomas, M. D. (2003), Oracle XSQL : Combining SQL, Oracle Text, XSLT, and Java to Publish Dynamic Web Content, Wiley, 2003.

Web3D (2003), http://www.web3d.org/fs_specifications.htm

Zlatanova, S. (2000), 3D GIS for urban development, PhD thesis, TUGraz, Austria, ITC publication, The Netherlands