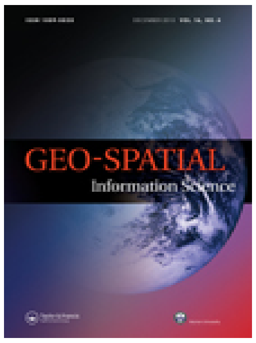


Geo-spatial Information Science

ISSN: 1009-5020 (Print) 1993-5153 (Online) Journal homepage: http://www.tandfonline.com/loi/tgsi20

Integration of CityGML and Oracle Spatial for implementing 3D network analysis solutions and routing simulation within 3D-GIS environment

Umit Atila, Ismail Rakip Karas & Alias Abdul-Rahman

To cite this article: Umit Atila, Ismail Rakip Karas & Alias Abdul-Rahman (2013) Integration of CityGML and Oracle Spatial for implementing 3D network analysis solutions and routing simulation within 3D-GIS environment, Geo-spatial Information Science, 16:4, 221-237, DOI: 10.1080/10095020.2013.867102

To link to this article: https://doi.org/10.1080/10095020.2013.867102

Geo-spatial Information Science, 2013
Vol. 16, No. 4, 221–237, http://dx.doi.org/10.1080/10095020.2013.867102

Integration of CityGML and Oracle Spatial for implementing 3D network analysis solutions and routing simulation within 3D-GIS environment

Umit ATILA[a], Ismail Rakip KARAS[a]* and Alias ABDUL-RAHMAN[b]

[a]*Department of Computer Engineering, Karabuk University, Karabuk, Turkey;* [b]*Department of Geoinformatics, Universiti Teknologi Malaysia, Johor, Malaysia*

(*Received 22 March 2013; final version received 23 September 2013*)

3D navigation within a 3D-GIS environment is increasingly getting more popular and spreading to various fields. In the last decade, especially after the 9/11 disaster, evacuating the complex and tall buildings of today in case of emergency has been an important research area for scientists. Most of the current navigation systems are still in the 2D environment and that is insufficient to visualize 3D objects and to obtain satisfactory solutions for the 3D environment. Therefore, there is currently still a lack of implementation of 3D network analysis and navigation for indoor spaces in respect to evacuation. The objective of this paper is to investigate and implement 3D visualization and navigation techniques and solutions for indoor spaces within 3D-GIS. For realizing this, we have proposed a GIS implementation that is capable of carrying out 3D visualization of a building model stored in the CityGML format and perform analysis on a network model stored in Oracle Spatial. The proposed GUI also provides routing simulation on the calculated shortest paths with voice commands and visual instructions.

Keywords: 3D-GIS; 3D network analysis; indoor navigation; evacuation

1. Introduction

3D navigation within the 3D-GIS environment is increasingly getting more popular and spreading to various fields. Especially after the 9/11 disaster, evacuating buildings safely by the shortest path in extraordinary circumstances (i.e. disastrous accidents, massive terrorist attacks etc.) happening in complex and tall buildings has been one of the most important research areas, which is the subject of 3D network analysis applications for indoor spaces.

Most of the navigation systems use 2D or 2.5D data (e.g. road layer) to find and simulate the shortest path which is lacking in the building environment (*1*). Therefore, there is a need for different approaches based on 3D which realize the 3D objects and eliminate the network analysis limitations on multilevel structures (*2–5*).

In one important study (*4*), Kwan and Lee measured the relative accessibility of the emergency response between a disaster site and an emergency station in a building. Their results showed that extending 2D GIS to 3D GIS representations of the interiors of high rise buildings can improve the overall speed of rescue. Their findings have motivated other GIS researchers to develop emergency evacuation systems of complex buildings using 3D GIS.

Passing from 2D GIS toward 3D GIS, a great number of 3D data-sets (e.g. city models) have become necessary. This situation requires a number of specific issues to be researched, e.g. 3D routing accuracy, appropriate means to visualize 3D spatial analysis, tools to effortlessly explore, and navigate through large models in real time, with the correct texture and geometry (*6*).

In GIS research, evacuation and routing is generally based on graph networks (*7*, *8*), while 3D visualization problems are achieved by CityGML (*9*). Researchers following the network approach generally modify the existing 2D routing algorithms to 3D ones (*1*). Initial requirements of 3D GIS and navigation (*6*): concepts, frameworks, and applications from a wide viewpoint were represented (*3*), but there is currently still a lack of implementation of 3D network analysis and navigation for evacuation purposes.

The objective of this paper is to investigate and implement 3D visualization and navigation techniques and solutions for indoor spaces within 3D GIS. We explain how to perform 3D network analysis using Oracle Spatial within a Java-based 3D-GIS implementation. As an initial step and for implementation, a GUI provides a 3D visualization of the building, and 3D network models based on CityGML data store spatial data in Oracle and then perform network analysis under different constraints, such as avoiding nodes or links in the network model. All experiments highlighted in this paper are performed on the 3D model of the Corporation Complex in Putrajaya, Malaysia. Section 2 presents an automatic data generation process of 3D network models from floor plans. Section 3 gives some examples of visualization of 3D building and network models from

*Corresponding author. Email: ismail.karas@karabuk.edu.tr the CityGML format. Section 4 gives some information on storing spatial data and explains how to create Network Models in Oracle Spatial. Section 5 introduces a 3D network analysis tool and gives visualized results of 3D network analysis performed by our proposed 3D-GIS implementation. Section 6 elaborates the routing engine integrated in the simulation module of the 3D-GIS implementation, and gives some visualization samples. Finally, we conclude the work with some future tasks that need to be addressed.

2. Data generation process for 3D network model

When we consider 3D navigation systems, we may need to solve complex topologies, 3D modeling, 3D network analysis, etc. For realizing all these processes, we need 3D spatial data. Data collection used to be the major task which consumed over 60% of the available resources, since geographic data were very scarce in the early days of GIS technology. In most recent GIS projects, data collection is still very time consuming and expensive; however, it currently consumes about 15–50% of the available resources (*10*).

Data generation is also still a problem for the researchers who work on GIS-based 3D navigation systems. It consumes more time than achieving their applications or doing their research. Pu and Zlatanova (*3*) point out that automatically extracting geometry and logic models of a building is difficult. In their study, they explain the advantages and disadvantages of the methods used for constructing geometry models of buildings and state that there is no automatic approach for 3D reconstruction of the interiors of buildings. They also indicate that it is very difficult to generate a logical model of a building automatically from its geometry model as the nodes and links have to be created manually or half-manually with computer-aided applications.

To overcome this deficiency, the 3D geometric and logical data of a building is obtained in CityGML format using 3D model generation software which is based on a novel method called Multidirectional Scanning for Line Extraction (MUSCLE) (*11*). This model is a conversion method which was developed to vectorize the straight lines through raster images, including township plans, maps for GIS, architectural drawings, and machine plans. Unlike the traditional vectorization process, this model generates straight lines based on a line thinning algorithm, without performing line following, chain coding, and vector reduction stages. By using this model, it is also possible to generate 3D building models based on the floor plan of the buildings (*7*). This model can be described in four main steps:

- Threshold processing.
- Horizontal and vertical scanning of the binary image.
- Detecting wrongly vectorized lines.
- Correcting wrongly vectorized lines by using diagonal scanning.

Detailed information about the MUSCLE model is beyond the scope of this paper but the process of the model can be summarized as shown in Figure 1.

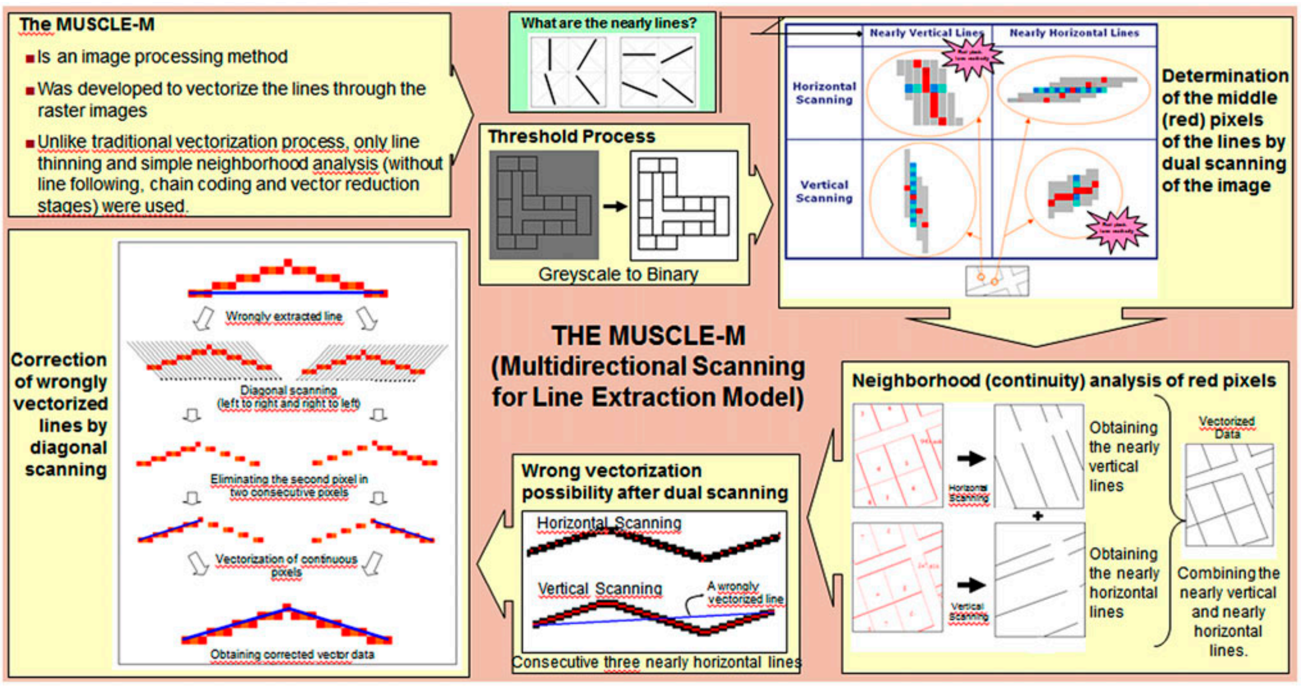

The MUSCLE-M
Is an image processing method
Was developed to vectorize the lines through the raster images
Unlike traditional vectorization process, only line thinning and simple neighborhood analysis (without line following, chain coding and vector reduction stages) were used.
What are the nearly lines?
Threshold Process
Greyscale to Binary
Nearly Vertical Lines
Nearly Horizontal Lines
Horizontal Scanning
Vertical Scanning
Determination of the middle (red) pixels of the lines by dual scanning of the image
Correction of wrongly vectorized lines by diagonal scanning
Wrongly extracted line
Diagonal scanning (left to right and right to left)
Eliminating the second pixel in two consecutive pixels
Vectorization of continuous pixels
Obtaining corrected vector data
THE MUSCLE-M (Multidirectional Scanning for Line Extraction Model)
Wrong vectorization possibility after dual scanning
Horizontal Scanning
Vertical Scanning
A wrongly vectorized line
Consecutive three nearly horizontal lines
Neighborhood (continuity) analysis of red pixels
Horizontal Scanning
Vertical Scanning
Obtaining the nearly vertical lines
Obtaining the nearly horizontal lines
Vectorized Data
Combining the nearly vertical and nearly horizontal lines.

Figure 1. MUSCLE model process.

By using the MUSCLE Model, the 3D Building and Topological Network model of a building can be generated automatically from raster floor plans. The user interface of the 3D Model Generation Software is shown in Figure 2.

2.1. *Generating corridors from raster floor plans*

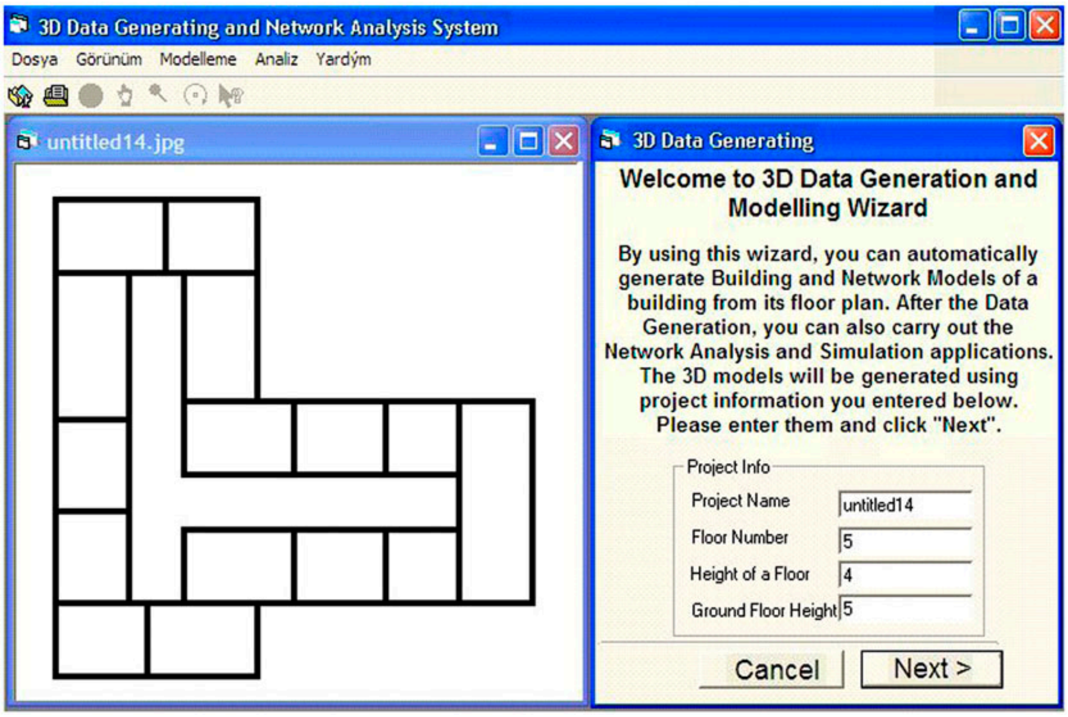
In network models, corridors are the main backbone in the floor plan since they connect the rooms with all the other entities in the building. Therefore, determining and modeling the corridors is very important. Once a corridor is provided by the user, the algorithm leaves only the corridor in the image, and then determines the middle lines based on the MUSCLE model. After a number of processes on selected middle lines, the topological model and coordinates of the corridor are found as shown in Figure 3.

2.2. *Generating rooms from raster floor plans*

In determining the rooms, the corridor is excluded from the image and only the rooms are left. Then, by applying the MUSCLE model, the center points of the rooms are determined and defined as the nodes which represent the rooms (Figure 4).

2.3. *Integrating corridors with rooms*

After locating the nodes that indicate corridors and rooms, the user interactively points out which room nodes connect with which corridor nodes, and the


3D Data Generating and Network Analysis System
Dosya Görünüm Modelleme Analiz Yardým
untitled14.jpg
3D Data Generating

Welcome to 3D Data Generation and Modelling Wizard

By using this wizard, you can automatically generate Building and Network Models of a building from its floor plan. After the Data Generation, you can also carry out the Network Analysis and Simulation applications. The 3D models will be generated using project information you entered below. Please enter them and click "Next".

Project Info

Project Name	untitled14
Floor Number	5
Height of a Floor	4
Ground Floor Height	5

Figure 2. 3D model generation user interface.

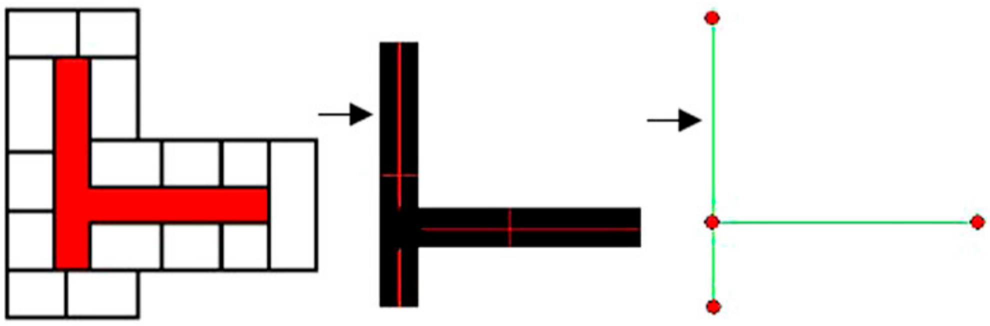


Figure 3. Generating corridors of a building from raster floor plans.

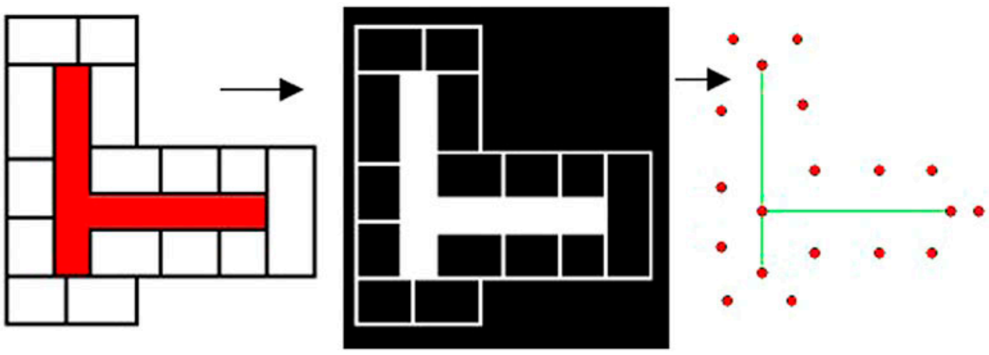


Figure 4. Generating rooms of a building from raster floor plans.

geometric network for a 2D floor plan is generated (Figure 5(a)).

After stairs (or elevator) nodes are indicated by the user (points with letter “M” in Figure 5(a)), the network is automatically designed by assigning different elevation values for each floor based on various data such as floor number and floor height, and then the 3D network model is generated as seen in Figure 5(b).

Once the network generation process is completed, the network data is converted into CityGML format. CityGML is designed as an open data model and XML-based format for the storage and exchange of virtual 3D city models (*12*). CityGML supports different Levels of Detail (LOD). LODs are required to reflect independent data collection processes with differing application requirements (Figure 6). The coarsest level LOD0 is essentially a 2.5D Digital Terrain Model, over which an aerial image or a map may be draped. LOD1 is the well-known blocks model comprising prismatic buildings with flat roofs. In contrast, a building in LOD2 has differentiated roof structures and thematically differentiated surfaces. Vegetation objects may also be represented. LOD3 denotes architectural models with detailed wall and roof structures, balconies, bays, and projections. High-resolution textures can be mapped onto these structures. In addition, detailed vegetation and transportation objects are the components of an LOD3 model. LOD4 completes a LOD3 model by adding interior structures for 3D objects. For example, buildings are composed of rooms, interior doors, stairs, and furniture (*12*).

We use CityGML’s Transportation Module to represent automatically generated 3D network models. The transportation model of CityGML is a multifunctional, multiscale model focusing on thematic and functional as well as on geometrical/topological aspects. Transportation features are represented as a linear network in LOD0. Starting from LOD1, all transportation features are geometrically described by 3D surfaces.

Starting from LOD1, a TransportationComplex provides an explicit surface geometry, reflecting the actual shape of the object, not just its centerline. The different representations of a TransportationComplex for each LOD are shown in Figure 7 (*12*).

In this study, 3D network data of a building are represented as a linear network in LOD0. The whole data generation process is described in a flow chart (Figure 8).

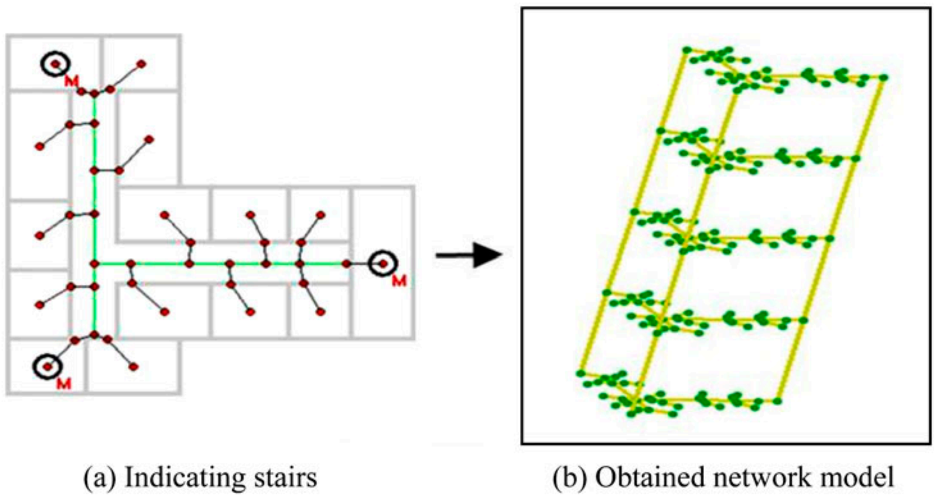


Figure 5. Generating the network model of a building from floor plans.

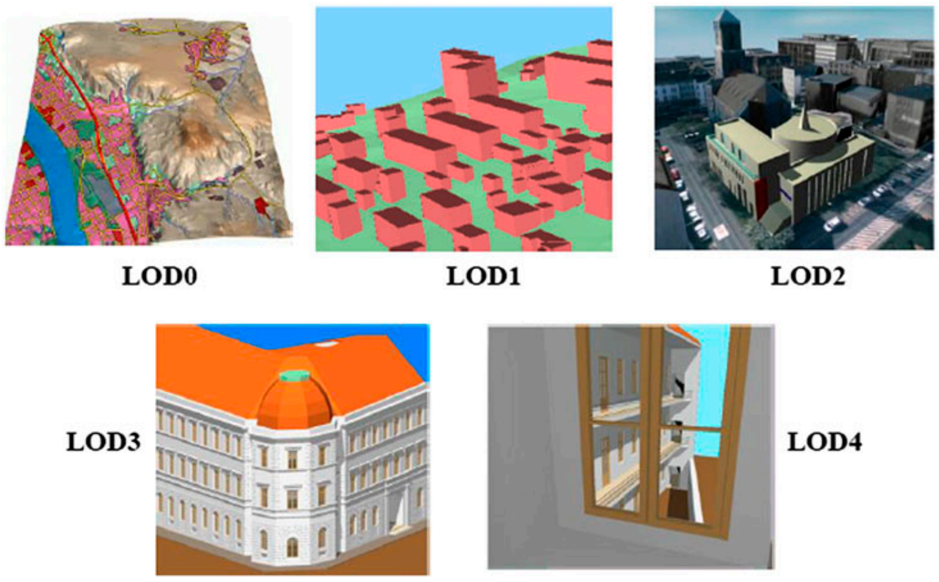


Figure 6. The five LOD defined by CityGML (*12*).

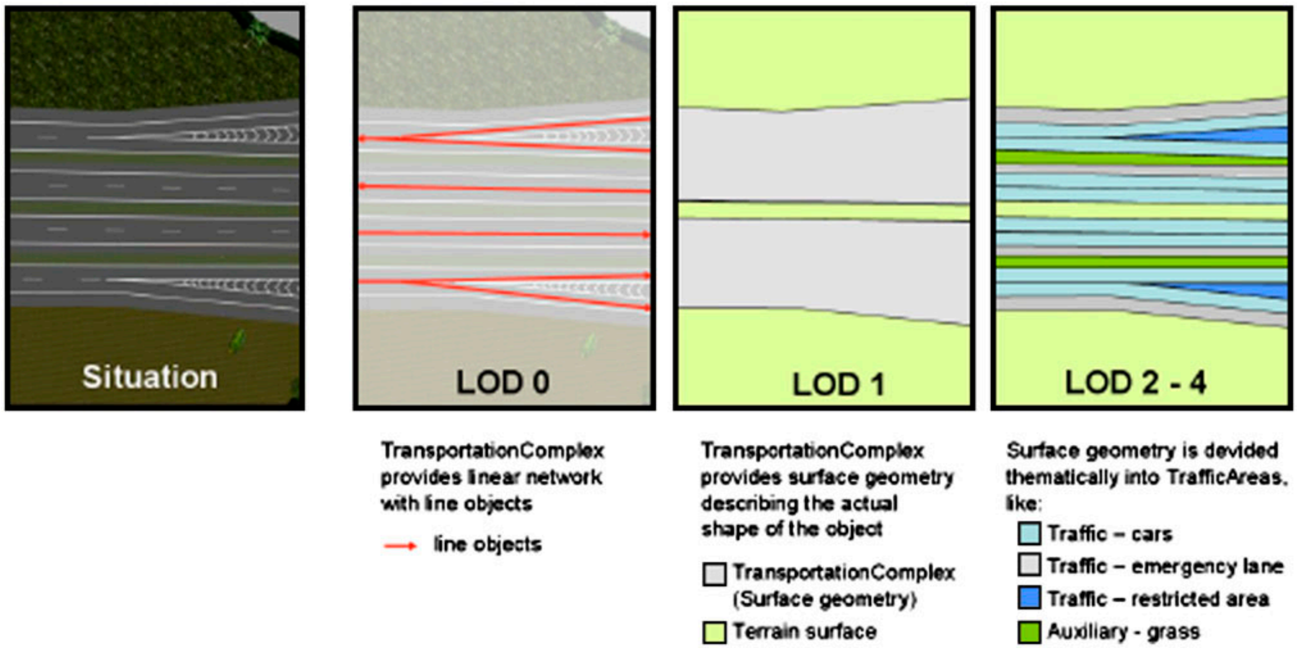

Situation
LOD 0
LOD 1
LOD 2 - 4
TransportationComplex provides linear network with line objects
line objects
TransportationComplex provides surface geometry describing the actual shape of the object
TransportationComplex (Surface geometry)
Terrain surface
Surface geometry is devided thematically into TrafficAreas, like:
Traffic – cars
Traffic – emergency lane
Traffic – restricted area
Auxiliary - grass

Figure 7. TransportationComplex in LOD 0, 1, and 2–4 (example shows part of a motorway) (*12*).

The outline of a sample CityGML code representing a 3D network model that uses a Transportation Module is given in Table 1.

3. Visualization of a 3D building and a network model

Visualization of a 3D building model is performed by our own proposed Java-based 3D-GIS implementation. The implementation uses *citygml4j* Java class library and API for facilitating work with the CityGML and JOGL Java bindings for the OpenGL graphic library to carry out visualization of 3D spatial objects.

The application supports four different types of a view mode for spatial objects. These modes are Wire-Frame (Figure 9(a)), HiddenLine (Figure 9(b)), Shaded (Figure 9(c)), and Shaded with Texture (Figure 9(d)).

The prepared implementation reads CityGML data-sets from LOD0 to LOD2. 3D building models are represented in LOD2 described by polygons using the Building Module of CityGML (Figure 10). Network models are represented as linear networks in LOD0 using CityGML's Transportation Module (Figure 11).

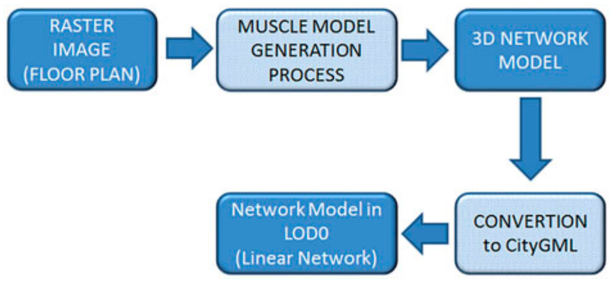

RASTER IMAGE (FLOOR PLAN)
MUSCLE MODEL GENERATION PROCESS
3D NETWORK MODEL
CONVERTION to CityGML
Network Model in LOD0 (Linear Network)

Figure 8. Data generation process of a building's network model in LOD0 using 3D model generation software.

Table 1. A 3D network model in City GML, which uses a Transportation Module.

```
<?xml version="1.0" encoding="UTF-8"?>
<CityModel …….>
<cityObjectMember>
  <tran:Track gml:id="Atila_Network">
    <gml:name>Atila Network</gml:name>
      <tran:lod0Network>
        <gml:GeometricComplex>
          <gml:element >
  <gml:LineString gml:id="Link-1-2-Corridor"><gml:
  posList srsDimension="3">58.36 29.51 4.37 69.23 29.51
  4.37</gml:posList></gml:LineString>
          </gml:element>
        </gml:GeometricComplex>
      </tran:lod0Network>
                .
                .
  </tran:Track>
</cityObjectMember>
</CityModel>
```

4. Managing 3D network in geo-database management systems

Geo-database management systems (Geo-DBMS) are developed for facilitating and organizing works with spatial data. Using geo-DBMS in 3D modeling and spatial analysis has a lot of advantages. Besides the standard advantages of DBMS with respect to centralized control, data independence, data redundancy, data consistency, sharing data, data integrity, and improved security, geo-DBMS brings efficient management of large spatial datasets. The management of a 3D network requires usage of a graph model in DBMS. While CityGML is used to store and visualize 3D spatial objects, the graph model is used to perform network analysis. Oracle Spatial is one of the most powerful geo-DBMS which offers a combination of geometry models and graph models.

A network is a type of mathematical graph that captures relationships between objects using connectivity. A network consists of nodes and links. Oracle Spatial maintains a combination of geometry and graph models within the Network Data Model. Network elements (links and nodes) may have geometric information associated with them. A logical network contains connectivity information but no geometric information. A spatial network contains both connectivity information and geometric information. In a spatial network, the nodes and links are SDO_GEOMETRY objects representing points and lines, respectively. A spatial network can also use other kinds of geometry representations. One variant lets you use linear referenced geometries. Another lets you use topology objects (*13*).

Network support in the Oracle database is composed of the following elements (*13*):

- A data model to store networks inside the database as a set of network tables. This is the persistent copy of a network.
- SQL functions to define and maintain networks (the SDO_NET package).
- Network analysis functions in Java. The Java API works on a copy of the network loaded from the database. This is the volatile copy of the network.
- Network analysis functions in PL/SQL (the SDO_-NET_MEM package).

Figure 12 illustrates the relationship between the elements of the Oracle Network Model (*13*).

To define a network in Oracle Spatial, at least two tables should be created, i.e. a node table and a link one. These tables should be provided with the proper structure and content to model the network. A network can also have a path table and a path link table. These tables are optional and are filled with the results of analyses, such as the shortest path between two nodes (*13*). Figure 13 shows the relationship between the tables that describe a network (*13*).

A node table (Table 2) describes all nodes in the network. Each node has a unique numeric identifier (the NODE_ID column). Other optional columns are geometry, cost, hierarchy_level, parent_node_id, node_name, node_type, and active. A link table (Table 3) describes all links in the network. Each link has a unique numeric identifier (the LINK_ID column) and contains the identifiers of the two nodes it connects. Other optional columns are geometry, cost, bi-directed, parent_link_id, active, link_level, link_name, and link_type (*13*). In this study, as we define a spatial network containing both connectivity and geometric information, we use SDO_GEOMETRY for representing points and lines.

There are two ways to define data structures for a network. One is to create the network automatically by calling the CREATE_SDO_NETWORK procedure defined in the SDO_NET package in Oracle Spatial. This procedure creates all the tables and populates the metadata. If the CREATE_SDO_NETWORK procedure fails to complete network creation, this may produce a half-created network. In other words, some tables are created but network creation is not completed yet. Therefore, before we create the network again, we must first manually drop the existing network using the DROP_NETWORK procedure. In addition, the automatic network creation method gives very little control

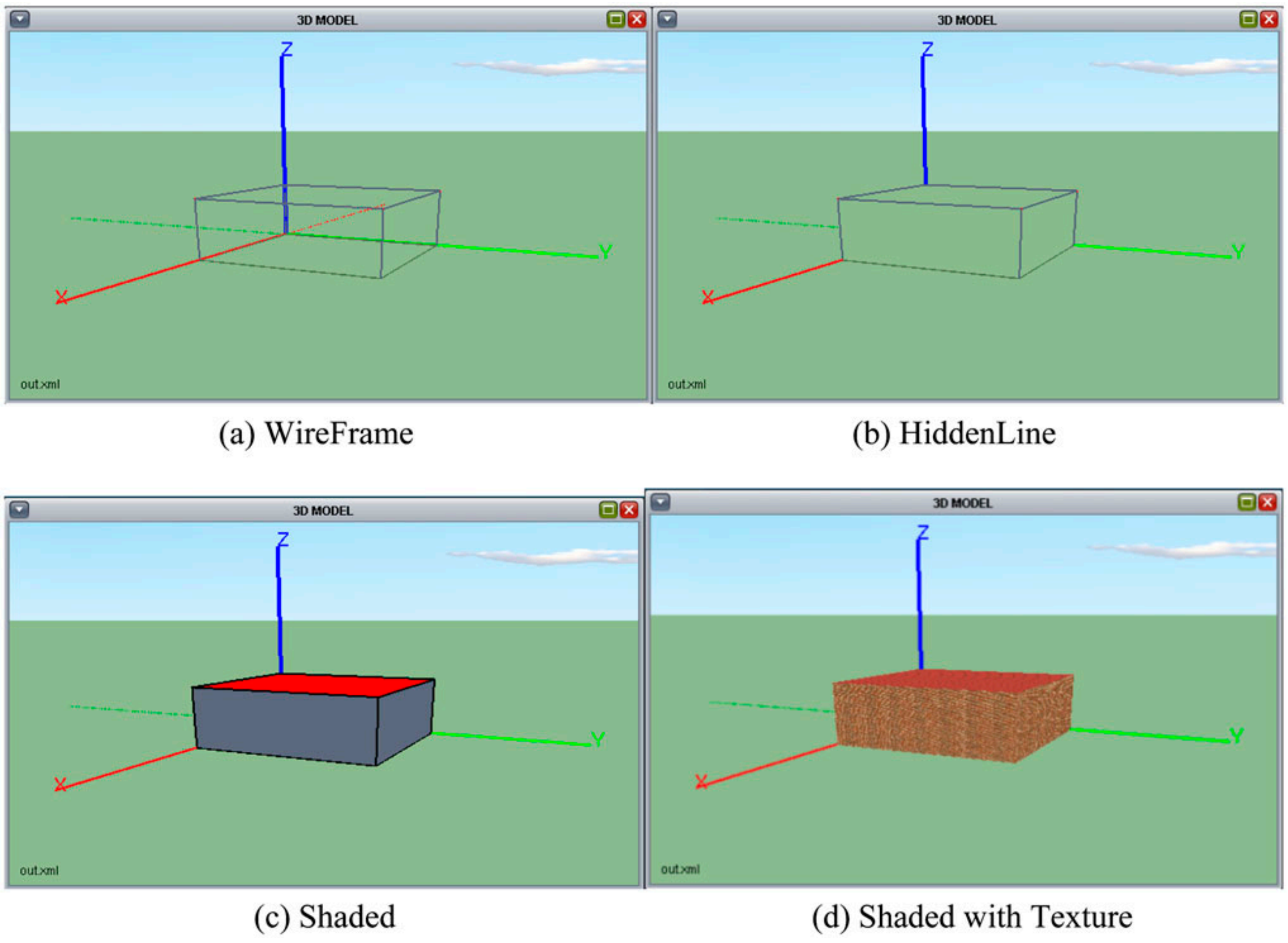


Figure 9. Viewing modes of 3D spatial objects in our proposed 3D-GIS implementation.

over the actual structuring of the tables and gives no control at all over their physical storage (table spaces, space management, partitioning, and so on). But this procedure is easy to use and makes sure the table structures are consistent with the metadata (*13*).

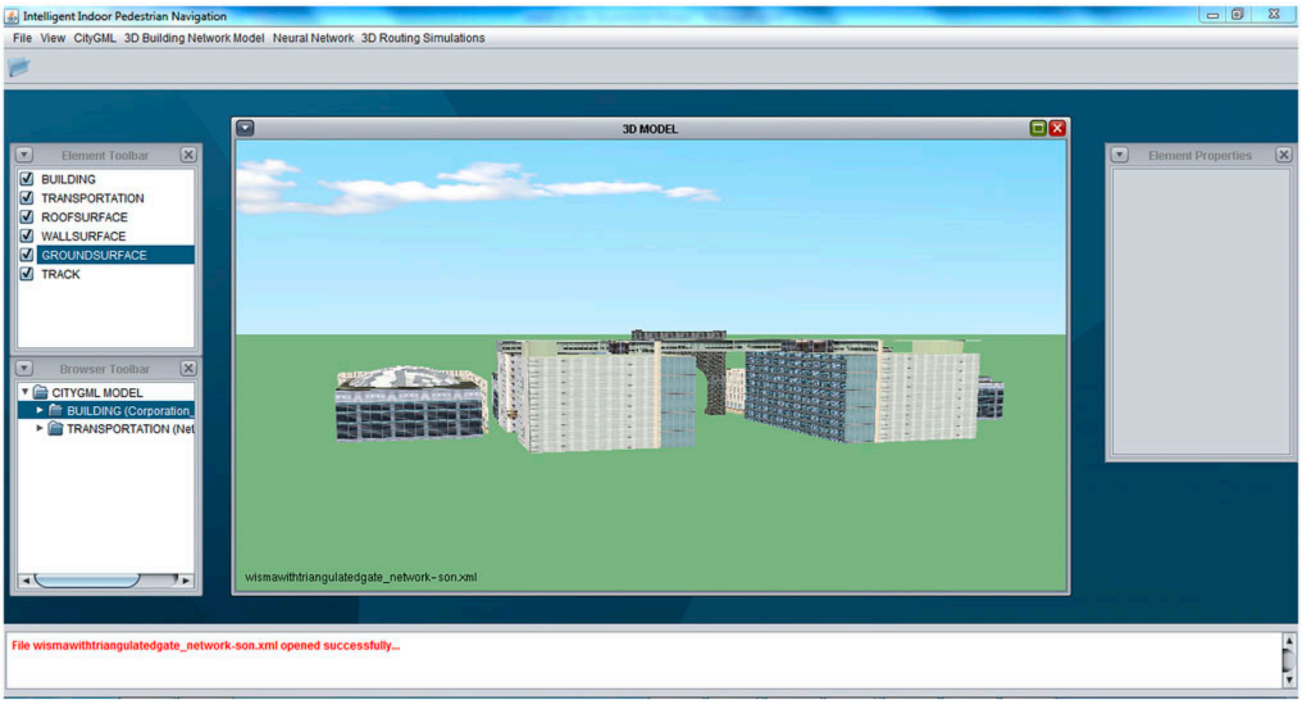
The other and more flexible way is creating tables manually, which is used in our implementation. Creating tables manually is not enough to define a network in Oracle Spatial. The actual naming of the tables that constitute a network and their structure should be defined in a metadata table called USER_SDO_NETWORK_METADATA, as shown in Table 4 by an insert statement ensuring that the table structures are consistent with the metadata.

Our implementation automates network definition in the Oracle Spatial database using the manual network creation method presented in this section. As soon as the 3D model of a building in LOD2 and its linear network model in LOD0 are opened from the CityGML format, an appropriate menu item of our implementation, indicating a network tool, is activated. Our network tool reads CityGML data, creates tables to define the network, inserts proper data into the tables, and defines the network. The Network creation tool lets you define networks using the same tables already defined for other networks or you can choose to define new tables for any network you create. The Network creation tool also lists all networks stored in the database which are based on the same CityGML data (Figure 14).

5. Performing network analysis

Our implementation performs network analysis with its network analysis tool based on a Java API provided by the Network Data Model of Oracle Spatial (*14*). The Java API provides a range of analysis functions and is provided as a package called *oracle.spatial.network* in a Java archive file called *sdnom.jar*. To use the API in a Java application, we should include it in our class path. The API is composed of three main sets of classes (*13*):

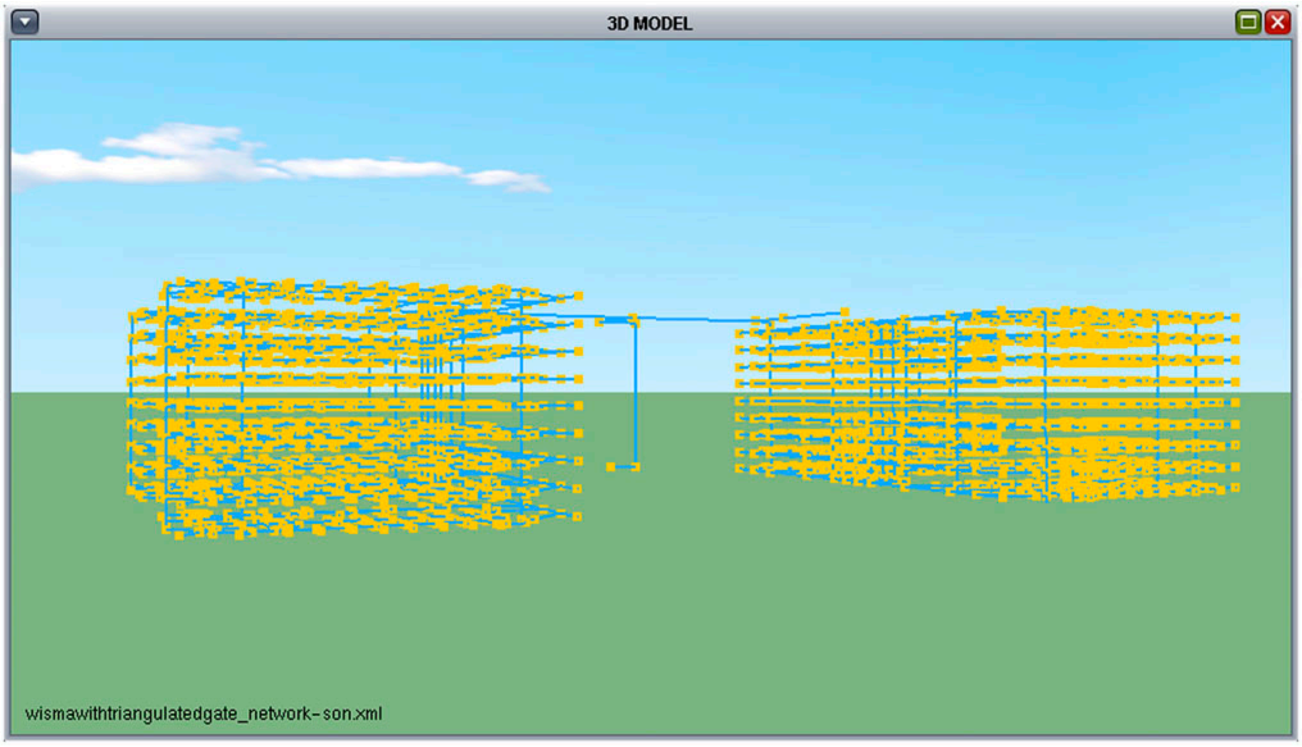
- Network, Node, Link, and Path: These classes store and maintain networks and network elements.
- NetworkManager: This class performs network analysis, and it also reads networks from the database and writes them back.
- NetworkFactory: This class creates networks and network elements.



wismawithtriangulatedgate_network-son.xml

File wismawithtriangulatedgate_network-son.xml opened successfully...

Figure 10. Building model (Textured viewing mode).



wismawithtriangulatedgate_network-son.xml

Figure 11. Network model.

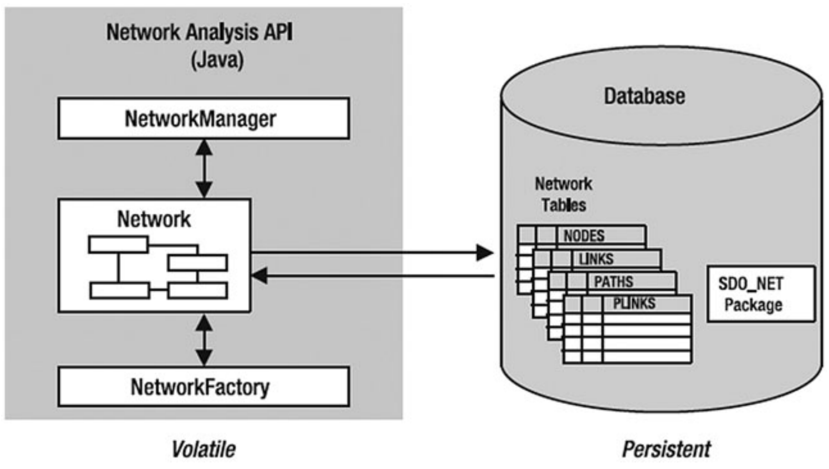


Figure 12. Oracle network data model.

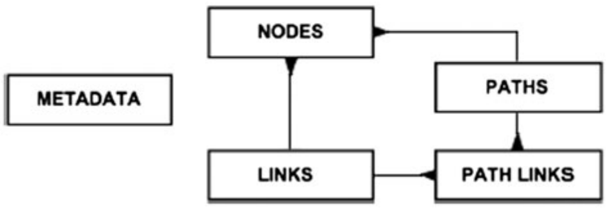


Figure 13. Main network tables.

Table 2. Columns of node table in the network model.

Parameter	Value
NODE_ID	230
NODE_NAME	NODE-230
GEOMETRY	MDSYS.SDO_GEOMETRY (3001, NULL, MDSYS.SDO_POINT_TYPE (42.2019449799705, 100.382921548946, −3.7), NULL, NULL)
ACTIVE	Y

Table 3. Columns of link table in the network model.

Parameter	Value
LINK_ID	15
START_NODE_ID	452
END_NODE_ID	455
LINK_NAME	Link-452-455-Corridor
GEOMETRY	MDSYS.SDO_GEOMETRY (3002, NULL, NULL, MDSYS.SDO_ELEM_INFO_ARRAY (1, 2, 1), MDSYS.SDO_ORDINATE_ARRAY (115.306027729301, 85.9775129777152, 1.8, 115.306027729301, 82.9 483382781573, 1.8)
LINK_LENGTH	3, 029,174,699,557,899
ACTIVE	Y
LINK_TYPE	Corridor

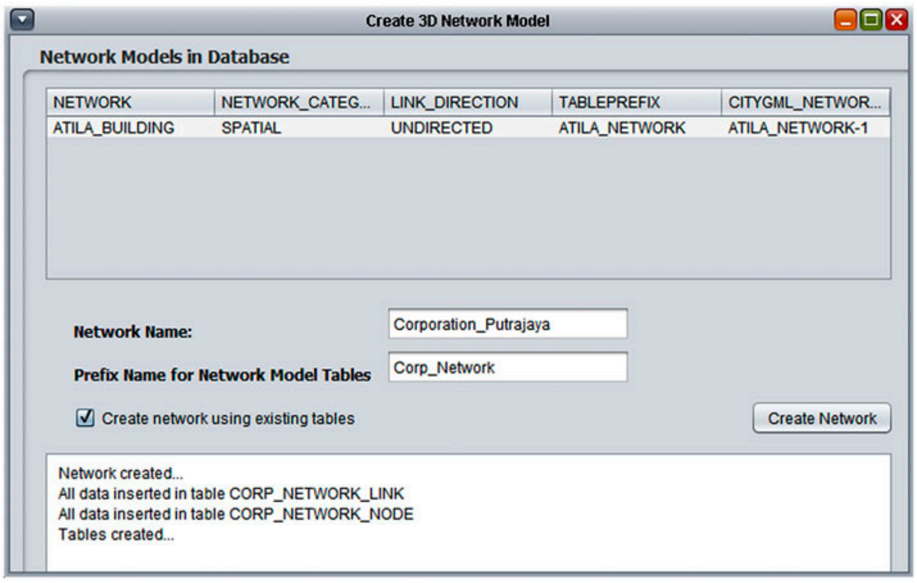
Table 4. A metadata table called USER_SDO_NETWORK_METADATA.

```
INSERT INTO USER_SDO_NETWORK_METADATA
  (NETWORK, NETWORK_CATEGORY,
  GEOMETRY_TYPE,
  NO_OF_HIERARCHY_LEVELS, NO_OF_PARTITIONS,
  LINK_DIRECTION,
  NODE_TABLE_NAME, NODE_GEOM_COLUMN,
  NODE_COST_COLUMN,
  LINK_TABLE_NAME, LINK_GEOM_COLUMN,
  LINK_COST_COLUMN,
  PATH_TABLE_NAME, PATH_GEOM_COLUMN,
  PATH_LINK_TABLE_NAME,
  NETWORK_TYPE) VALUES
  ('CORPORATION_PUTRAJAYA', 'SPATIAL',
  'SDO_GEOMETRY', '1',
  '1', 'UNDIRECTED', 'CORP_NETWORK_NODE',
  'LOCATION', NULL,
  'CORP_NETWORK_LINK', 'GEOMETRY',
  'LINK_LENGTH',
  'CORP_NETWORK_PATH', 'GEOMETRY',
  'CORP_NETWORK_PATH_LINK', 'Corp_Network')
```

Our network analysis tool allows doing most common 3D network analyses supported by Oracle Spatial (Figure 15). These analyses are:

- Shortest path.
- Traveling salesman.
- Given number of nearest neighbors.
- All possible shortest paths between given nodes.
- All nodes within given distance.
- Finding reaching nodes to a given node.
- Finding all possible paths between two nodes.
- Finding shortest paths to a node from all other nodes in the network.

With this network analysis tool it is also possible to perform common 3D network analysis with full functionality including constraints and to see the results on a 3D graphical screen. In this section, some examples for network analysis will be presented.



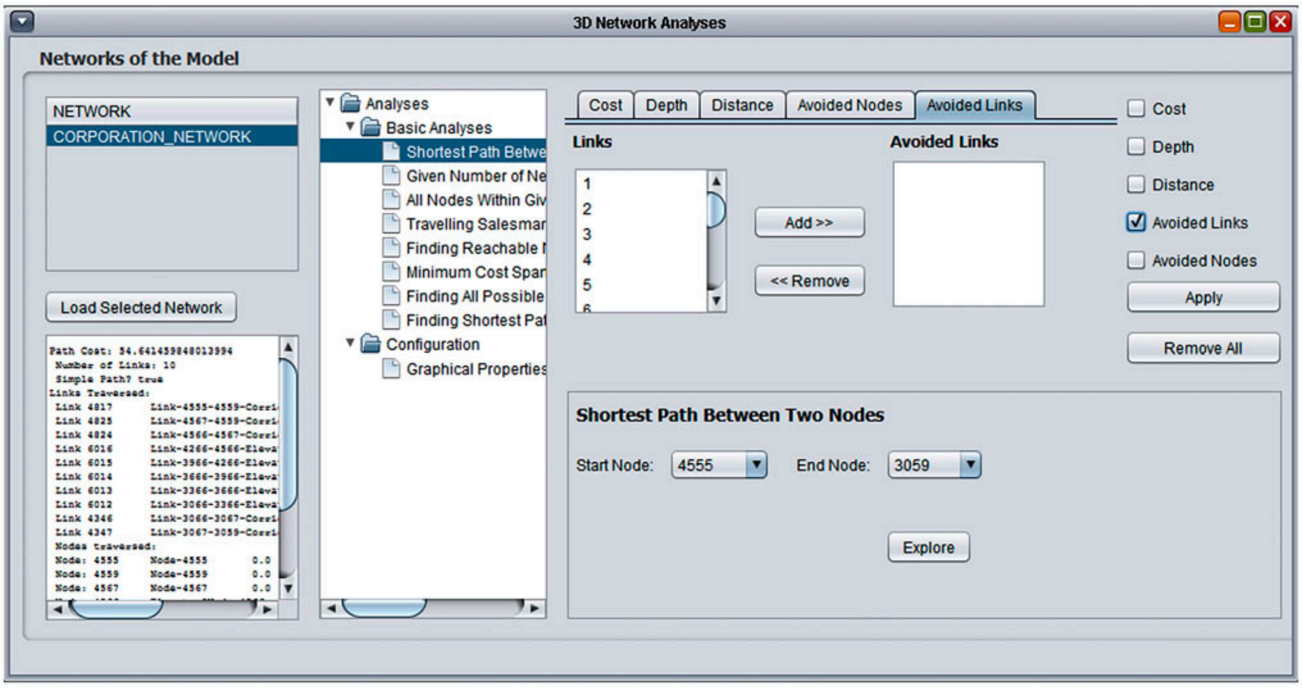
NETWORK	NETWORK_CATEG...	LINK_DIRECTION	TABLEPREFIX	CITYGML_NETWOR...
ATILA_BUILDING	SPATIAL	UNDIRECTED	ATILA_NETWORK	ATILA_NETWORK-1

Network Name:	Corporation_Putrajaya
Prefix Name for Network Model Tables	Corp_Network

[✓] Create network using existing tables

```
Network created...
All data inserted in table CORP_NETWORK_LINK
All data inserted in table CORP_NETWORK_NODE
Tables created...
```

Figure 14. Automatic network creation tool.



```
Path Cost: 54.641459848013994
 Number of Links: 10
 Simple Path? true
Links Traversed:
 Link 4817      Link-4555-4559-Corri
 Link 4825      Link-4567-4559-Corri
 Link 4824      Link-4566-4567-Corri
 Link 6016      Link-4266-4566-Eleva
 Link 6015      Link-3966-4266-Eleva
 Link 6014      Link-3666-3966-Eleva
 Link 6013      Link-3366-3666-Eleva
 Link 6012      Link-3066-3366-Eleva
 Link 4346      Link-3066-3067-Corri
 Link 4347      Link-3067-3059-Corri
 Nodes traversed:
 Node: 4555     Node-4555        0.0
 Node: 4559     Node-4559        0.0
 Node: 4567     Node-4567        0.0
```

[✓] Avoided Links

Start Node: 4555 End Node: 3059

Figure 15. 3D network analysis tool.

Figure 16 shows a shortest path analysis result without any constraints. Figure 17 shows how the shortest path is updated after links associated with elevators are avoided, shown by red lines which means the elevator is not in use any more in that part of the building. To find the shortest path between two nodes, we use the *shortestPath*() method provided by the Java API.

Another analysis is selecting nodes based on the distance that separates them from a starting node. We use *withinCost*() method provided by the Java API to perform this analysis and it calculates all paths to the nodes within a given distance. Therefore, this analysis is not based on straight-line distances and uses distances along the network. As an example, a person can find all the rooms within a 40-m walking distance from his own room. Figure 18 shows all nodes within a 40-m walking distance to node 3354, and Figure 19 shows the updated result of the same analysis after events have occurred in all associated links used to go upstairs.

Another analysis finds the nearest nodes to a starting node. This analysis finds the given number of neighbors of a node by locating the nearest neighbors following the

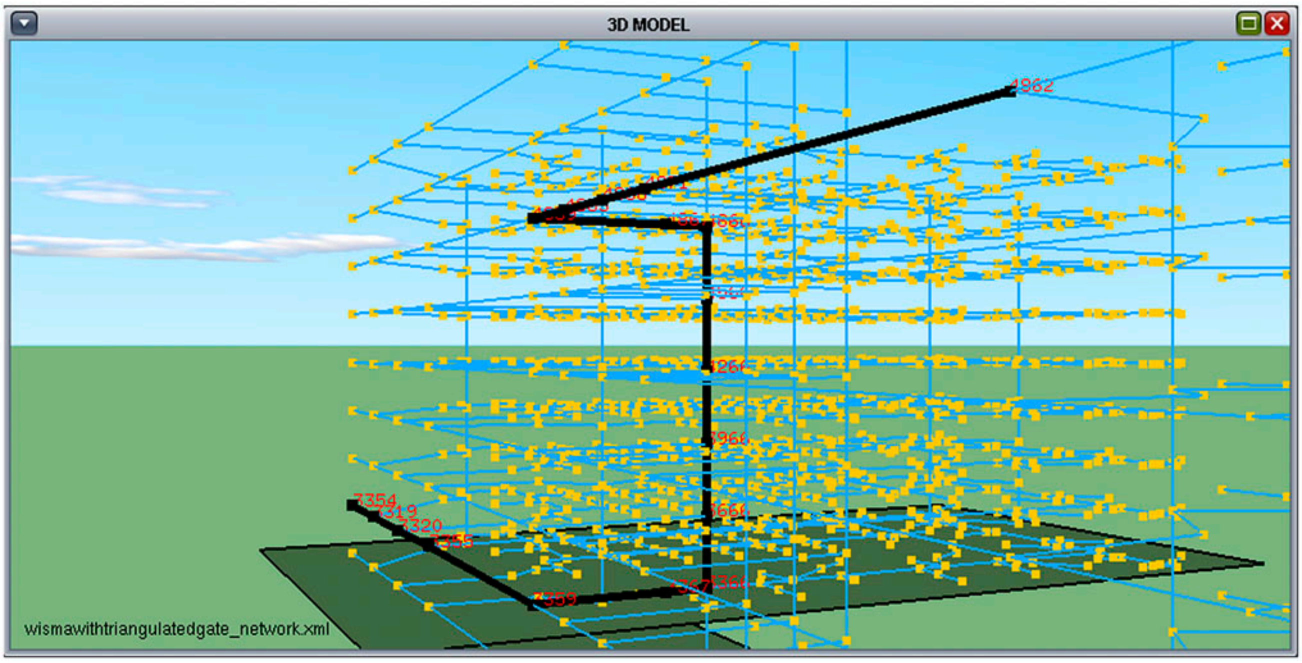


Figure 16. Shortest path between two nodes without any constraint.

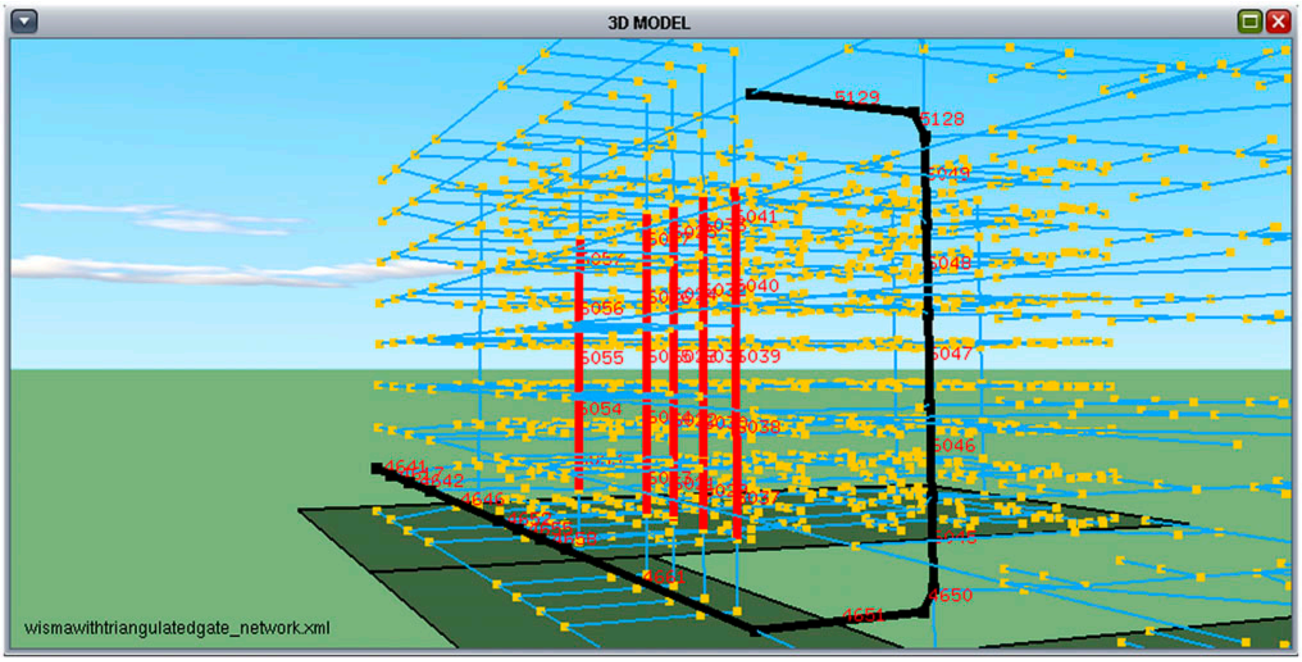


Figure 17. Recalculated shortest path considering avoided elevators in a part of building.

links. This analysis also does not use straight-line distances. We use the *nearest Neighbors*() method provided by the Java API. Figure 20 shows all paths to the 10 nearest neighbors of node 3354. To explain the analysis results better, we give details of the analysis in Table 5.

Oracle Spatial also allows multiple path searches. There are two kinds of multiple path search analyses. One finds all possible paths between two nodes and the other finds the shortest paths from one node to all the other nodes. We give an example of finding all possible paths between two nodes here. This analysis could return a very large number of responses on a large fully connected network, but Oracle Spatial lets us limit the search space by specifying one or more of the following constraints (*13*). In this analysis we use *allPaths*() method provided by the Java API.

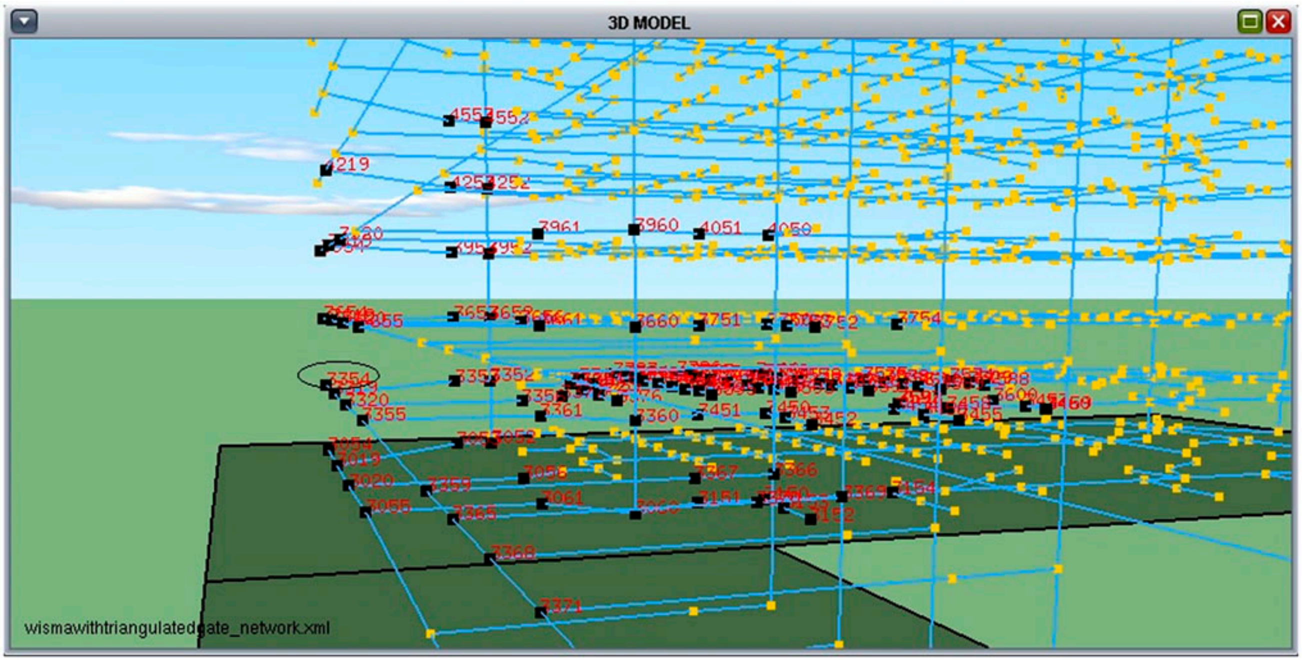


Figure 18. All nodes within a 40-m walking distance to node 3354.

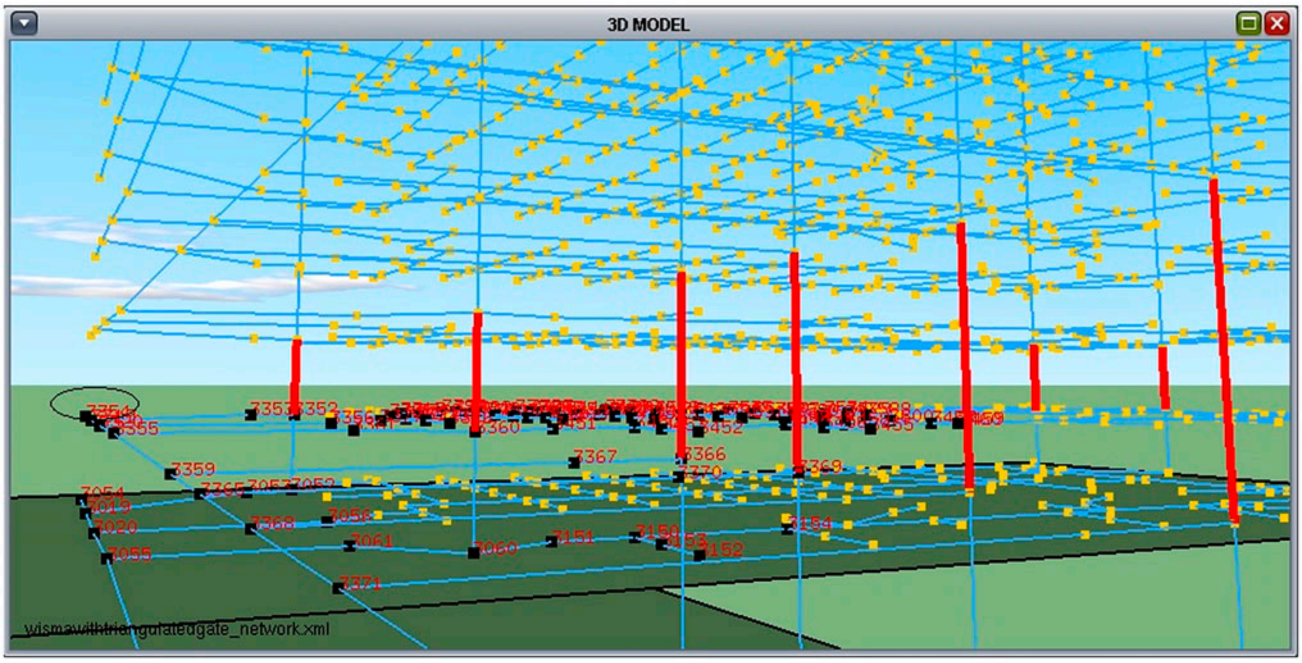


Figure 19. All nodes within a 40-m walking distance to node 3354 after constraints applied.

- Depth: Return only the solutions that have less than the specified number of links.
- Cost: Return only the solutions whose cost is less than the specified value.
- Solutions: Return only the *N* best solutions.

Notice the figure showing our 3D network analysis tool (Figure 15), which is used for performing common 3D network analysis, at the beginning of this section. You can see Depth and Cost tabs to apply these constraints on analysis. Figure 21 shows the three best solutions between nodes 3354 and 3655 with different colours.

The last analysis presented in this paper is the Traveling Salesman Problem (TSP). The TSP is one of the most popular and most studied problems in computer science. The TSP objective is to find the shortest route for a traveling salesman who, starting from his home

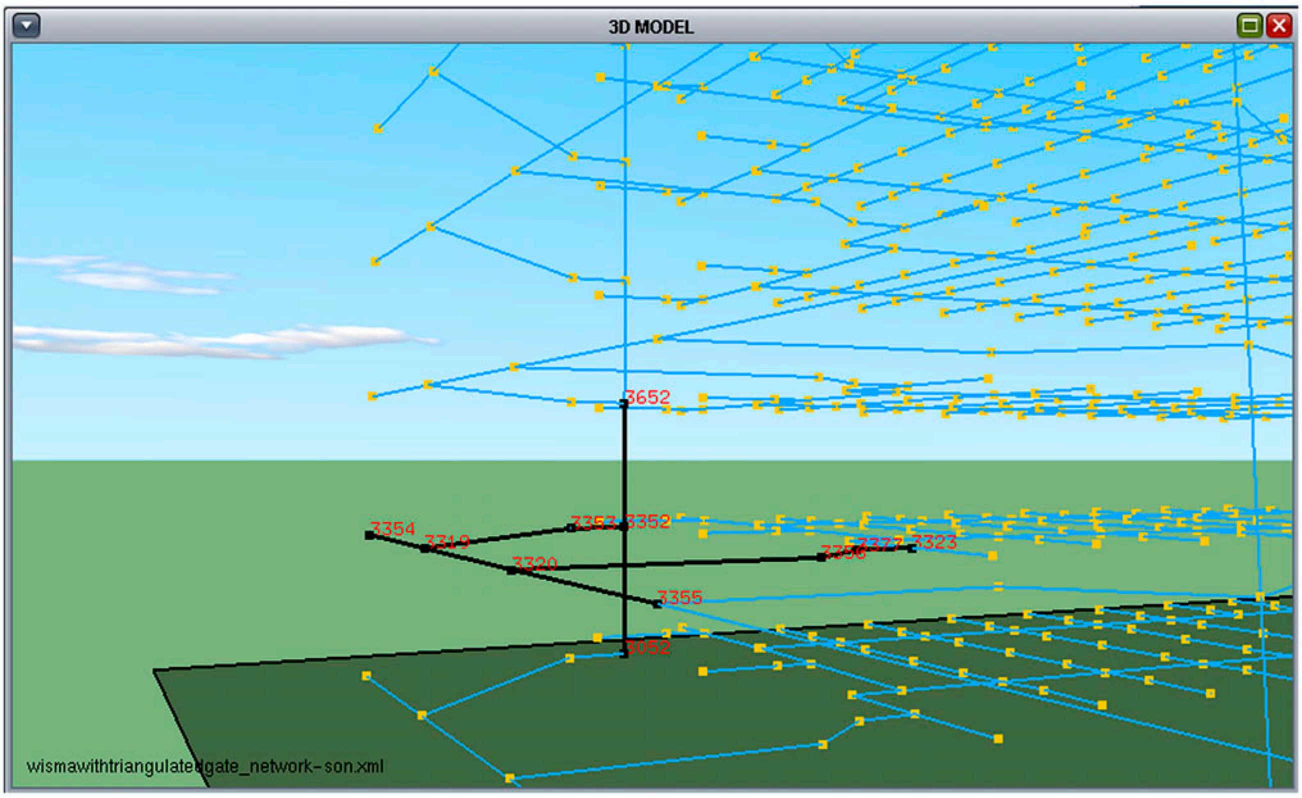


Figure 20. All paths to the 10 nearest neighbors of node 3354.

Table 5. 10 nearest neighbors of node 3354.

Node no.	Path cost (m)	Traversed nodes in the path
3319	4.98	3354, 3319
3320	10.34	3354, 3319, 3320
3353	12.66	3354, 3319, 3353
3352	14.13	3354, 3319, 3353, 3352
3355	15.62	3354, 3319, 3320, 3355
3356	16.06	3354, 3319, 3320, 3356
3377	18.25	3354, 3319, 3320, 3356, 3377
3323	19.57	3354, 3319, 3320, 3356, 3377, 3323
3052	19.63	3354, 3319, 3353, 3352, 3052
3652	19.63	3354, 3319, 3353, 3352, 3652

city, has to visit every city on a given list precisely once and then return to his home city (*15*).

Suppose that a postman delivers letters in a complex building. The postman does not want to waste his time and wants to visit customers in the optimal order so as to minimize travel time. In this analysis we need to give the list of the nodes to be visited as input to the *tsPath*() method provided by the Java API, and we get the shortest path that passes through all the nodes we specified. Figure 22 shows a sample TSP analysis. In this example, the starting point is node 3354 and nodes going to be visited are 3654, 3661, and 3656. Note that the whole tour is highlighted by arrows with changing colors at each target node. Table 6 shows the details of the TSP analysis given in this example and the visitation order of the actual nodes should be 3354, 3661, 3656, 3654, and 3354 successively.

6. Routing instruction engine

One of the most important components of an ideal evacuation system is an instruction engine which should produce real-time instructions for the users to assist them in the routing process until they arrive at their destination. Our implementation has such an instruction engine which is integrated into the simulation module to produce voice commands and visual instructions for assisting users dynamically on the way to their destination. This instruction engine is intended to be the infrastructure of a voice enabled mobile navigation system for indoor spaces in our future work (Figure 23).

The most significant job for producing routing instructions is to determine the direction that users should follow. For generating instruction commands, a method developed by Karas (*16*) has been used. According to the direction determined by this method, "Go upstairs, Go downstairs, Go on the floor, Turn left, Turn right, Keep going" commands are generated and vocalized while the user approaches each node.

When the red point (the user) passes through a node in the simulation, firstly, the difference of elevations between the first next node and the second next node that the user will visit is compared. If the second next node is at a higher elevation than the first one, the instruction engine generates a "Go Upstairs" command (Figure 24(a)). If it is lower, a "Go Downstairs" command is generated (Figure 24(b)).

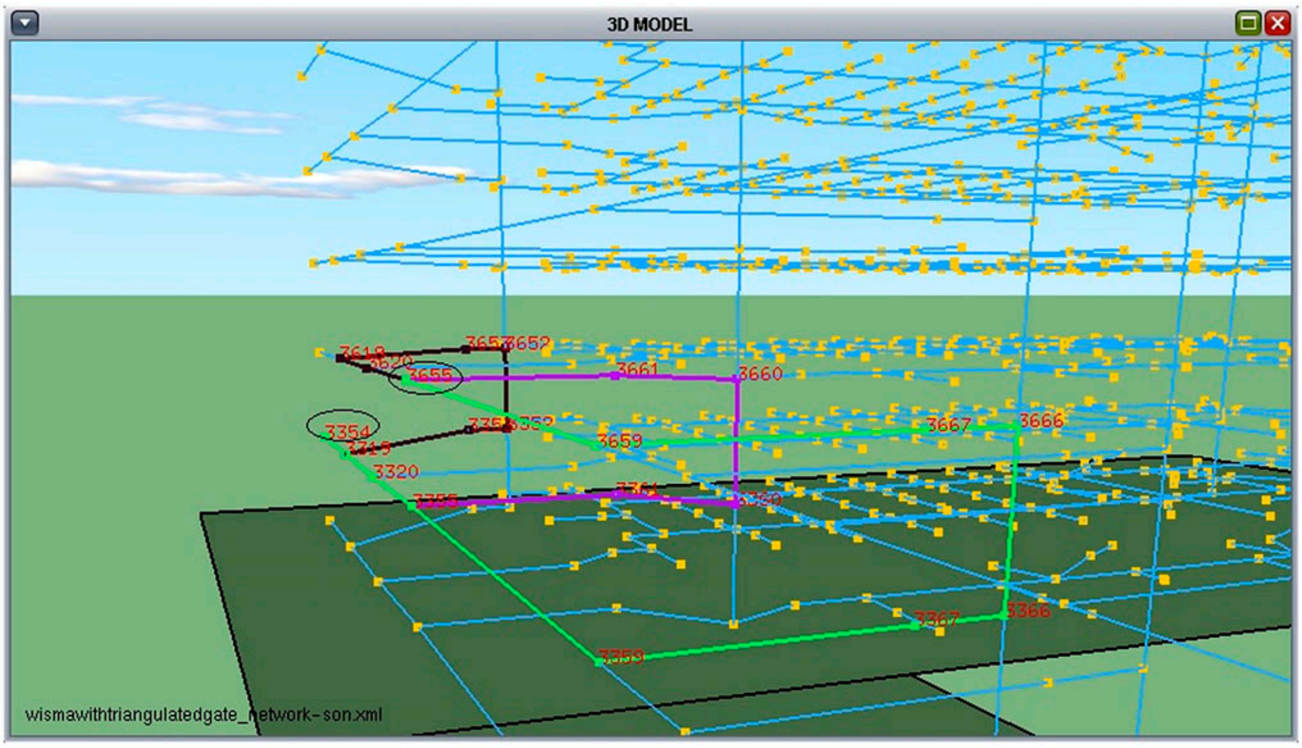


Figure 21. All three best solutions between nodes 3354 and 3655.

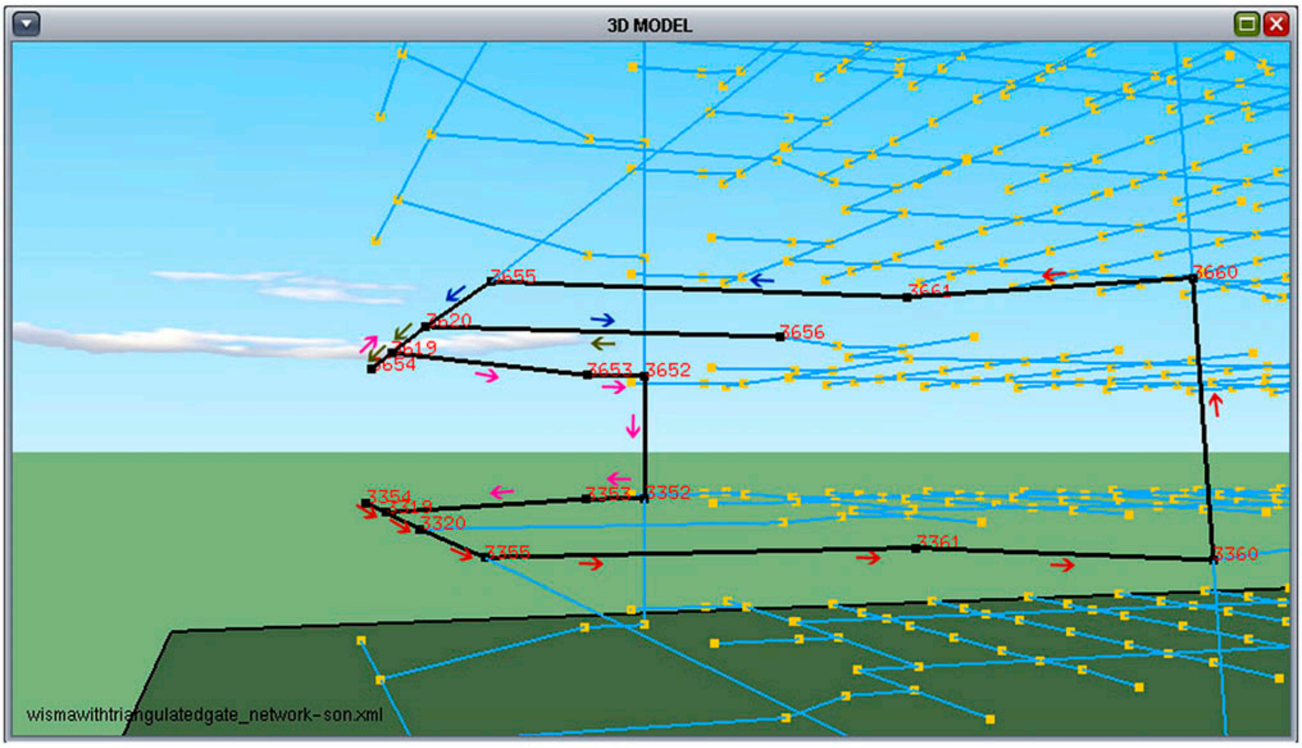


Figure 22. A sample TSP analysis.

Table 6. Details of TSP analysis.

Name of 20 links in the path	Cost of each link in the path (m)
Link-3354-3319-Corridor	4.98
Link-3319-3320-Corridor	5.35
Link-3320-3355-Corridor	5.28
Link-3361-3355-Corridor	5.06
Link-3360-3361-Corridor	3.22
Link-3360-3660-Stairs6	5.50
Link-3660-3661-Corridor	3.22
Link-3661-3655-Corridor	5.06
Link-3620-3655-Corridor	5.28
Link-3656-3620-Corridor	5.72
Link-3656-3620-Corridor	5.72
Link-3619-3620-Corridor	5.35
Link-3654-3619-Corridor	4.98
Link-3654-3619-Corridor	4.98
Link-3653-3619-Corridor	7.68
Link-3652-3653-Corridor	1.46
Link-3352-3652-Elevator14	5.50
Link-3352-3353-Corridor	1.46
Link-3353-3319-Corridor	7.68
Link-3354-3319-Corridor	4.98
Total path cost	98.52

Apart from these, the user needs to walk on the floor after descending or ascending by using an elevator or stairs. In other words, if the elevation of the first next node and the second next node are equal, but the current node is different, a “Go on the Floor” command is generated.

If the elevations of the three nodes are equal then the instruction engine decides to go straight or turn right or left. To make this decision, perpendicular distance calculations should be performed. By using perpendicular distance calculations of surveying computations, it can be determined if a node is on the right side or on the left side of a line segment. According to this calculation, for a line segment which starts with node A and ends with B, if a node C is on the right side of this line segment then the sign of the perpendicular distance of C is obtained as positive (+), otherwise negative (−) (*17*). Assuming A is the node that the user passes through, B is the first subsequent node and C is the second subsequent node that the user will visit, if the length of the perpendicular distance of node C to the line segment AB is calculated, the instruction to the user can be determined by checking the sign of the distance. If it is positive (+), the command should be “Turn Right” (Figure 24(c)), if negative the command is “Turn Left” (Figure 24(d)). If the calculated distance is zero, the instruction engine produces a “Keep going” command (Figure 24(e)).

After all these processes, the generated command is vocalized by the simulation while the red point step by step approaches to the first subsequent node. The explanations of the terms of equation in Figure 24 are as follows.

A: The node that the user is currently passing through.
B: The first next node that the user will visit.
C: The second next node that the user will visit.
D: The perpendicular distance of the C node to the AB line.
E: The elevation of the nodes.
(AB): Bearing of the AB direction.

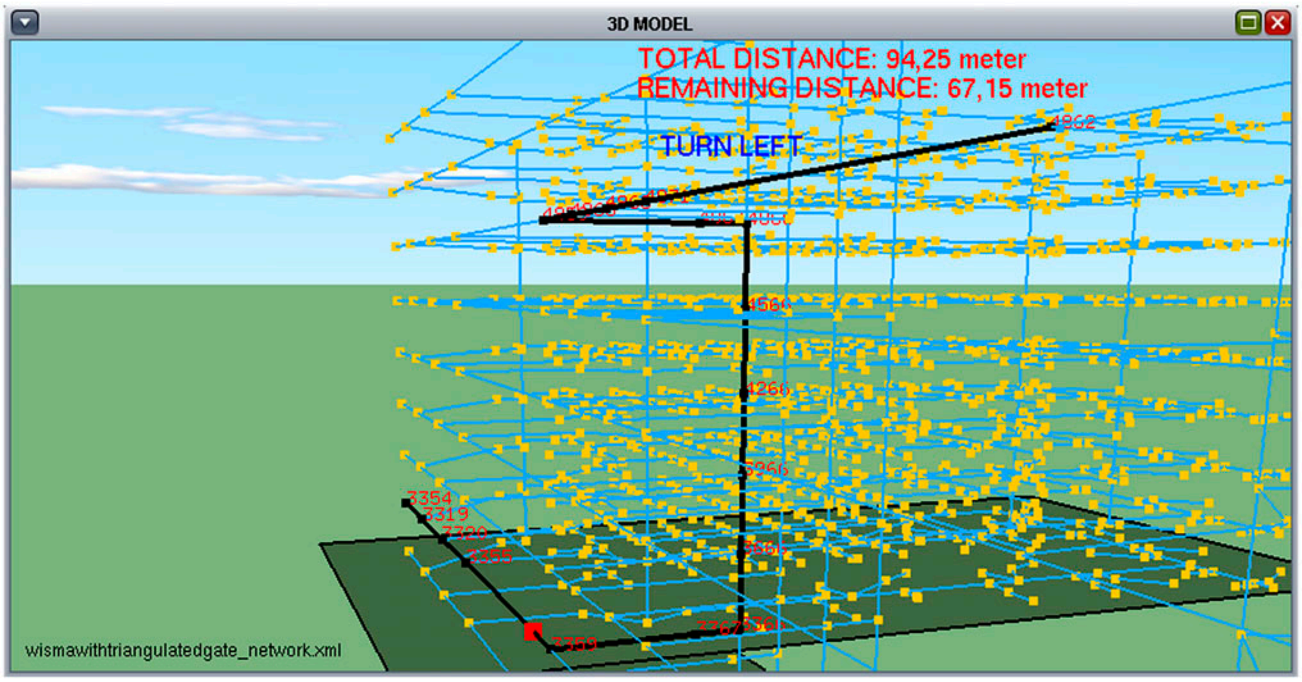

3D MODEL
TOTAL DISTANCE: 94,25 meter
REMAINING DISTANCE: 67,15 meter
TURN LEFT
wismawithtriangulatedgate_network.xml

Figure 23. Routing simulation process of the instruction engine (The red point (■) is the user).

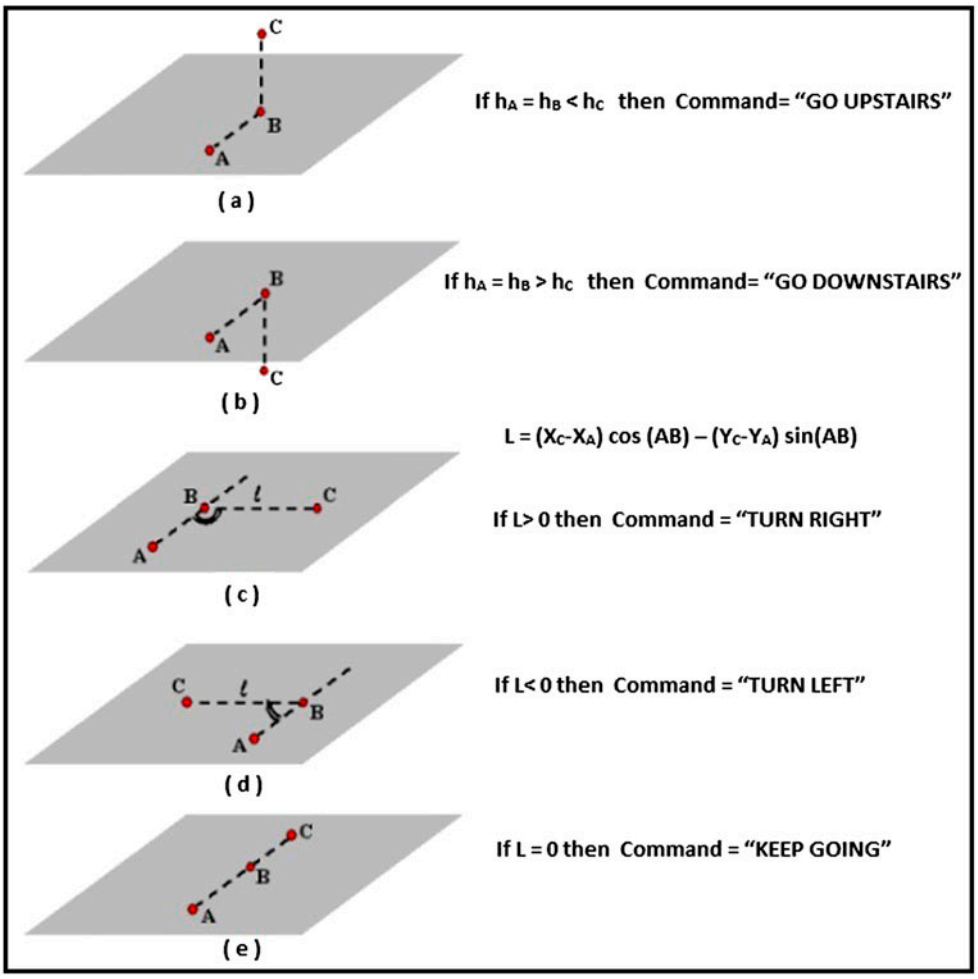

If $h_A = h_B < h_C$ then Command= "GO UPSTAIRS"
(a)
If $h_A = h_B > h_C$ then Command= "GO DOWNSTAIRS"
(b)
$L = (X_C - X_A) \cos(AB) - (Y_C - Y_A) \sin(AB)$
If L> 0 then Command = "TURN RIGHT"
(c)
If L< 0 then Command = "TURN LEFT"
(d)
If L = 0 then Command = "KEEP GOING"
(e)

Figure 24. Generating instructions based on side direction calculation for routing.

7. Conclusions

This paper presents a Java-based 3D-GIS implementation which can visualize 3D building and network models based on the CityGML format, automate a 3D network definition in Oracle Spatial's Network Data Model and perform network analysis. We showed five different examples of performing 3D network analysis with both graph-based and geometric constraints applied. We also elaborated a method for generating voice commands and visual instructions for assisting people dynamically on the way to their destination, which is intended to be the routing engine infrastructure of our intelligent evacuation system work in progress. Our experiments successfully showed that our 3D-GIS implementation could be improved to design an ideal navigation system for evacuation purposes.

Acknowledgments

This study was supported by TUBITAK – The Scientific and Technological Research Council of Turkey (Project No. 112Y050) research grant. We are indebted for its financial support.

Notes on contributors

Umit Atila is an assistant professor in Computer Engineering Department at Karabuk University, Turkey. He received BSc and MSc degree from Gazi University and PhD degree from Karabuk University in Computer Engineering Department. His research interest includes artificial intelligence, image processing, 3D graphic programing, and 3D network analysis within 3D GIS.

Ismail Rakip Karas is an associate professor in Computer Engineering Department at Karabuk University, Turkey. His research is focused on 3D GIS in particular 3D network analyses. He received BSc degree from Selcuk University, MSc degree from Gebze Institute of Technology, and PhD degree from RS and GIS program of Yildiz Technical University, all in Geomatics Engineering in Turkey.

Alias Abdul-Rahman is a professor of Universiti Teknologi Malaysia (UTM), works on 3D GIS for his professional activities, and as chair for ISPRS WG II/5 on Multi-Dimensional GIS and Mobile Data Model.

References

(1) Musliman, I.A.; Rahman, A.A. Implementing 3D Network Analysis in 3D GIS. *Proceeding of XXIst ISPRS Congress*, 37 (B2), Beijing, China, 2008; pp 913–918.

(2) Cutter, S., Richardson, D.B., Wilbanks, T.J., Eds.; *The Geographical Dimensions of Terrorism;* Routledge: New York, 2003; pp 75–117.

(3) Pu, S.; Zlatanova, S. Evacuation Route Calculation of Inner Buildings. In *Geo-information for Disaster Management*: van Oosterom, P.J.M., Zlatanova, S., Fendel, E.M., Eds.; Springer-Verlag: Heidelberg, 2005; pp 1143–1161.

(4) Kwan, M.P.; Lee, J. Emergency Response After 9/11: The Potential of Real-time 3D GIS for Quick Emergency Response in Micro-spatial Environments. *Comput. Env. Urban Sys.* 2005**,** *29,* 93–113.

(5) Zlatanova, S.; van Oosterom, P.; Verbree, E. 3D Technology for Improving Disaster Management: Geo-DBMS and Positioning. *Proceedings of the XXth ISPRS Congress*, 35 (B7), Istanbul, Turkey, 2004; pp 628–633.

(6) Musliman, I.A.; Rahman, A.A.; Coors, V. 3D Navigation for 3D-GIS – initial Requirements. In *Innovations in 3D Geo Information Systems*: Abdul-Rahman, A., Zlatanova, S., Coors, V., Eds.; Springer: Berlin, 2006; pp 259–268.

(7) Karas, I.R.; Batuk, F.; Akay, A.E.; Baz, I. Automatically Extracting 3D Models and Network Analysis for Indoors. In *Innovation in 3D-Geo Information System*: Abdul-Rahman, A., Zlatanova, S., Coors, V., Eds.; Springer: Berlin, 2006; pp 395–404.

(8) Jun, C.; Kim, H.; Kim, G. Developing an Indoor Evacuation Simulator Using a Hybrid 3D Model. In *3D Geo-information Science*: Lee, J., Zlatanova, S., Eds.; Springer-Verlag: Berlin, 2009; pp 173–178.

(9) Kolbe, T.H. Representing and Exchanging 3D City Models with CityGML. In *3D Geo-information Science*: Lee, J., Zlatanova, S., Eds.; Springer Verlag: Berlin, 2008; pp 15–31.

(10) Longley, P.A.; Goodchild, M.F.; Maguire, D.J.; Rhind, D.W. *GIS Data Collection, Geographic Information Systems and Science;* Wiley: Hoboken, NJ, 2001; pp 203–224.

(11) Karas, I.R.; Bayram, B.; Batuk, F.; Akay, A.; Baz, I. Multidirectional Scanning Model, MUSCLE, to Vectorize Raster Images with Straight Lines. *Sensors*. 2008**,** *8,* 2673–2694.

(12) Gröger, G.; Kolbe, T.H.; Czerwinski, A.; Nagel, C. *OpenGIS City Geography Markup Language (CityGML) Encoding Standard*, version 1.0.0; International OGC Standard; Open Geospatial Consortium: Wayland, MA, 2008.

(13) Kothuri, R.; Godfrind, A.; Beinat, E. *Pro Oracle Spatial for Oracle Database 11g;* Apress: New York, 2010.

(14) Oracle Corporation. http://www.oracle.com/technetwork/database-options/spatialandgraph/overview/ndm-1707438.html (accessed Oct 7, 2013).

(15) Larranaga, P.; Kuijpers, C.M.H.; Murga, R.H.; Inza, I.; Dizdarevic, S. Genetic Algorithms for the Travelling Salesman Problem: A Review of Representations and Operators. *Artif. Intell. Rev.* 1999**,** *13,* 129–170.

(16) Karas, I.R. Assessment of Topological Relationships of the Objects in the Scope of Network Analyses in 3D GIS Environment. Ph.D. Thesis, Yildiz Technical University, Geomatics Engineering, Remote Sensing and GIS Program, Istanbul, 2007.

(17) Serbetci, M.; Atasoy, V. *Jeodezik Hesap (Survey Computations);* Karadeniz Technical University Publications, Trabzon, 1990; pp 72–78.