12

INCORPORATING 3D ENGINE FOR BUILDING 3D NETWORK

Muhamad Uznir Ujang
Alias Abdul Rahman

Department of Geoinformatics,
Faculty of Geoinformation Science and Engineering,
Universiti Teknologi Malaysia, 81310 UTM Skudai, Johor, Malaysia.

ABSTRACT

Navigation is an important aspect in some areas such as industrial transportation, emergency navigation and navigation planning. Network analysis in navigation can minimize the transportation cost, the disaster or accident impact and also able to provide right decision for the best route in certain situations. Dijkstra's algorithm is one of the algorithms that could be used in network analysis especially for 2D shortest route. However, network analysis for three-dimensional (3D) network is required when it involves navigation inside of a multi-level building. The complexity of buildings and their surroundings requires a system that can analyze the phenomenon in 3D environment. There are several attempts to utilize Dijkstra's algorithm on 3D network. It is the aim of this study to investigate on the Dijkstra's algorithm in 3D environment. The results from the network analysis will be shown by incorporating with 3D computer game engine (3D State). Real textures of 3D objects are incorporated into the navigation system.

Keywords: Navigation, Network Analysis, and 3D Game Engine

1.0 INTRODUCTION

Disaster or emergency-based applications especially within high-rise building in urban areas could best be served by shortest path routine. A system that could provide a route network within a building certainly helps to locate safety routes for example to people such as fire and rescue missions and other emergency operations. Other navigation applications in this domain like pedestrian navigation, and even for car navigation on roads or in racing circuit using computer game engine is also worth to mention. Some of the applications that need this kind of routine for example searching the nearest restaurant, a petrol station, jungle trekking on the mountain/hills (see Balstrom, 2001) and locating the emergency exit doors during fire (emergency) in a building (see Meijers et al., 2005). It has been recognized that Dijsktra's algorithm could be used for those applications. By extending the algorithm to three-dimensional (3D) environment, more applications could be developed for high-rise buildings. Shortest path network for multi-level buildings and connectivity between levels or floors are certainly be very useful and interesting to be investigated. Several works on Dijkstra's algorithm for 3D environment have been investigated such as Karas et al. (2006). Other related works such as Meijers et al. (2005) and Pu et al. (2005) were focused on 3D indoor navigation routing. However, it is interesting to note that works on inexpensive 3D game engine couple with navigation routine is getting ground compares to fully operational commercial game engine like Quake. Fritsch (2003) investigated the Quake game engine for indoor navigation system. In this chapter, we focus on 3D shortest route together with inexpensive game engine for the navigation system.

This chapter describes the experiment on Dijkstra's algorithm for 3D building environment and incorporates with 3D game engine (3D State). 3D game engine is part of the components and the navigation development strategies are explained in Section 3. The detail experiment of the system interface development is described in

Section 4 and finally the conclusion and outlook of the navigation system with respect the development of the new idea of navigation for 3D GIS are highlighted in Section 5.

2.0 MOTIVATION

3D Geographical Information System (3D-GIS) development is at early stage and various groups try to address some aspects of 3D GIS e.g. based on Virtual Reality (VR) and terrain visualization. Africawala et al. (2004) view 3D-GIS as component of 3D Data Capture, 3D Visualization and 3D Data Modeling and Management. 3D Data Capture is component of automatic generation of terrain models from aerial imagery or laser scanning. 3D Visualization concentrates on how to visualize 3D objects and performance of 3D visualization on mobile devices. The 3D Data Modeling and Management components are on the model and management of complex 3D objects in a relational DBMS environment.

3D data provides more advantages compare to 2D data. 3D data gives more reliable or useful information than the 2D data. As we know, 3D spatial objects give more information to us compare with 2D spatial object and even more useful in emergency navigation cases, vehicle simulation, urban planning process, and even town development applications.

3.0 3D GAME ENGINE

3.1 The Basic

3D engine often comes in packages called System Development Kit (SDK). The SDK usually contains several core functionalities for the

developers to develop their applications such as rendering (so-called "renderer") engine, physics engine, media engine, scripting, Artificial Intelligence (AI) and networking capability. The renderer is used to display 2D and 3D scene to the users so that they can make their decisions based on the displays. This function is the most important part in the engine because this is where the engine will be judged by users. For most users (especially gamers), this make-or-break function depicts the quality of the engine and ultimately, the 3D games or applications that are built upon it. Most of the current game engines are optimized for both indoor and outdoor visualizations (e.g. *Unreal engine 2* and *Unreal engine 3* developed by *Epic Games*), however some of the engines are dedicated only for indoor visualization (e.g. *Doom* and *Quake* engine developed by *id Software*) in order to keep the engine's performance at a higher rate. The physics engine is used to simulate most of the physics applied in the application such as the gravity and the ballistics law meanwhile the media engine handles the sounds, videos and animations that are use in the application. Many game engines are available in the market and the cost varies. 3D State game engine is inexpensive and downloadable from the Internet (www.3dstate.com).

The 3D State game comes with several tools, namely Programmer's Software Developers Kit (SDK), World Builder, World Text Editor and World Viewer, as they are very useful for system developers.

The available components in the game allow us to navigate within the scene and in this case the indoor navigation of building elements. A game engine is some sort of machine to visualize 3D worlds in real time using basic elements of a game, which are not to be programmed repeatedly. Today, a game engine is a very complex software system containing many elements and which is also capable of distributed processing (Fritsch, 2003).

3.2 The 3D State Engine

In this project, we utilize the 3D State game engine. It is an open source engine where anyone can use it for non-commercial usage. For commercial use, users must purchase a license from the engine's developer. It is one of the fastest engines available today because its capabilities of delivering high quality graphics in high frame rate per second (FPS) by using its speed technology. The engine proprietary, Virtual-Reality engine is based on the state-of-the-art algorithms and techniques, including the proprietary PIRR technology (Photo realistic Interactive Real-Time Rendering). This give the engine an advantage for more advanced performance both in speed (frames per second) and image quality than other personal computer (PC) based engines. 3D State engine is also a general purpose engine where enables users to develop any kind of 3D applications for both indoor and outdoor environment using different kind of view mode used in computer games today such as first person shooter view, chase view or the third-person shooter view. 3D State provides users with the Software Development Kit (SDK) to make it easier to add any kind of 3D content to any applications. The SDK contains wide range of tools that is developed and tuned to work smoothly with the 3D State engine. The tools included are such as the 3DS2WLD converter uses to convert 3D models in 3D Studio model (*.3DS) format to the world (*.WLD) format and the World Builder for building 3D worlds or scenes. Additionally, the 3D State engine is a DLL and this means that if the engine's new versions are released, users will only have to replace the previous version with the new version instead of changing their codes. Moreover, users do not even have to recompile their codes again. From the literature, we can see that most of the new type of GIS such as 3D GIS and web-based GIS are moving towards a system with 3D visualization as an end component.

4.0 THE EXPERIMENT AND GAME INTERFACE DEVELOPMENT

4.1 Dijkstra's Algorithm for 3D Network

In this project, Dijkstra's algorithm been used in the network analysis to calculate the shortest path between two locations in the 3D environment building. In order to utilize Dijkstra's algorithm for 3D environment, first we have to establish a 3D network for the area, like 3D building. Important information like IDs, distances, and potential candidates of nodes in the network are fundamental in this problem domain. Normally, all these information are available in the 3D network and we have to extract and assign them with proper node IDs. By putting ID to each node, then the network is now applicable for Dijkstra's algorithm.

The next step is to calculate the distance between connected nodes (i.e. successor and predecessor nodes). It is not necessary to calculate the distance between all the nodes in the network as this will consume a lot of time. If node #1 is connected to node #2 and node #3, then the calculation will be based on those three nodes. To accomplish that, information on predecessor and successor nodes is important. To make computers understand which nodes are connected or not, therefore the network data file must be examined thoroughly. From here the data file will be examined in order to find which part of the data is useful or not, and to determine which node is connected/reachable or not.

The distance between nodes for 3D network can easily calculated by using simple mathematical formula, see Figure 2.

Example: distance between node A (x1, y1, z1) to node B (x2, y2, z2),

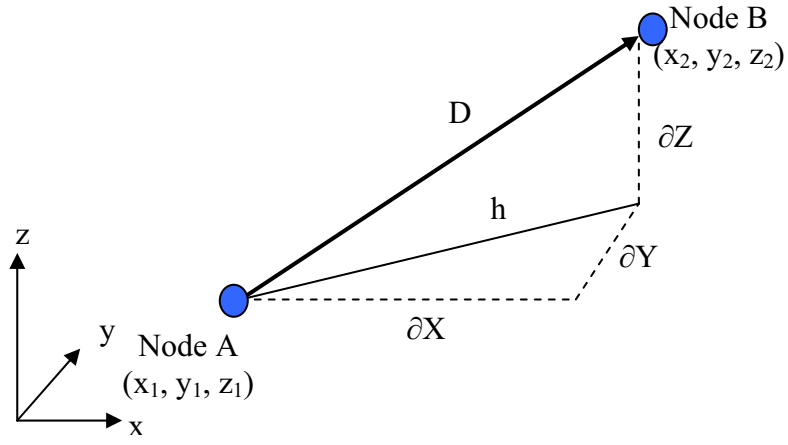

Figure 2: Distance calculation for 3D network

Delta (∂) for x, y and z can be calculate by finding the differential between nodes,

$\partial X = (x_2 - x_1)$; $\partial Y = (y_2 - y_1)$; $\partial Z = (z_2 - z_1)$

Then the distance (D) could be derived by two equations as below:

$D = \sqrt{(h^2 + \partial Z^2)}$ ……. Equation 1

h can be derived by:

$h = \sqrt{(\partial X^2 + \partial Y^2)}$ ……..Equation 2

By combining both equation 1 and 2, the final equation to find the distance (D) between the nodes is:

$D = \sqrt{[(\partial X^2 + \partial Y^2) + \partial Z^2]}$ or $D = \sqrt{[(x2 - x1)^2 + (y2 - y1)^2 + (z2 - z1)^2]}$

Then from here, the Dijkstra's algorithm can be executed successfully as in the case of 2D network.

4.2 Study Area

The study area is a building for Faculty Geoinformation Science and Engineering (C02, C03, C04, C05 and C06 building) at Universiti Teknologi Malaysia, Skudai, Johor. We converted the analogue plans to digital via digitization process and constructed to 3D. Real texture is attached to the building (to have a more realistic view) as we perceived in the real world as shown in Figure 3.

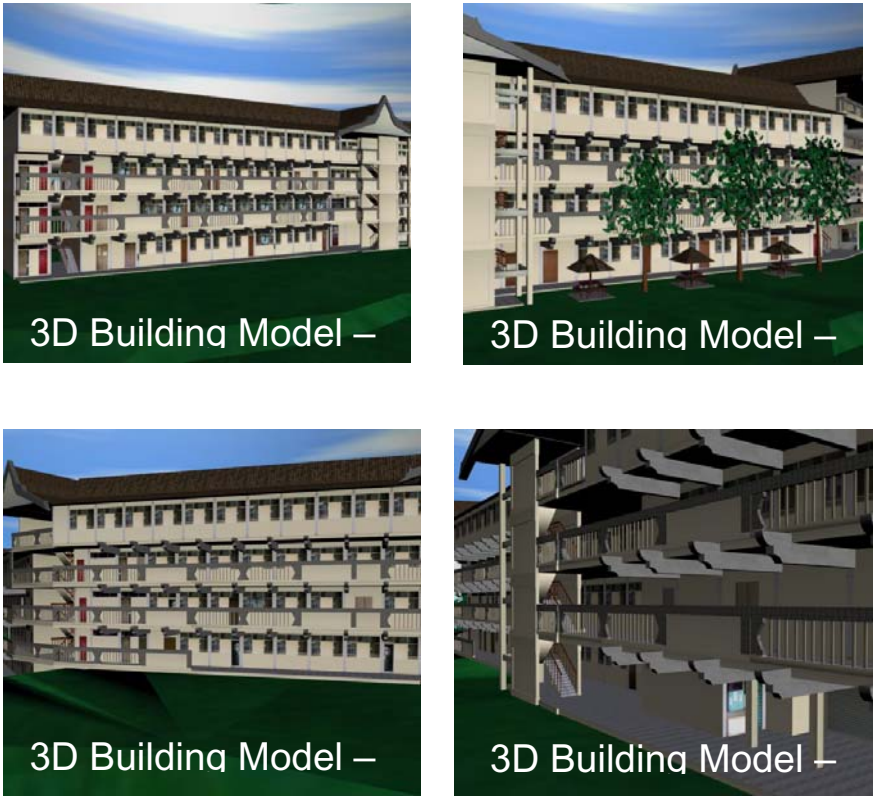

Figure 3: The 3D building models of the study area.

The 3D State engine supports several types of 3D environment format such as (*.wld, *.morfit, *.3dstate, *.STATE). The 3D building model data sets must be in a useable/supported format. Various converters are freely available in the Internet to convert from another 3D model formats. The 3D State package also includes several 3D converters.

4.3 The Development Process

In general, the development process is illustrated in Figure 4. The *Spatial Data* (3D Building Model and 3D Network) phase is developed for data input. The *3D Engine Integration* phase as indicated in the figure explains the main processing stage. The output as of the interface could be demonstrated as an operational navigation system.

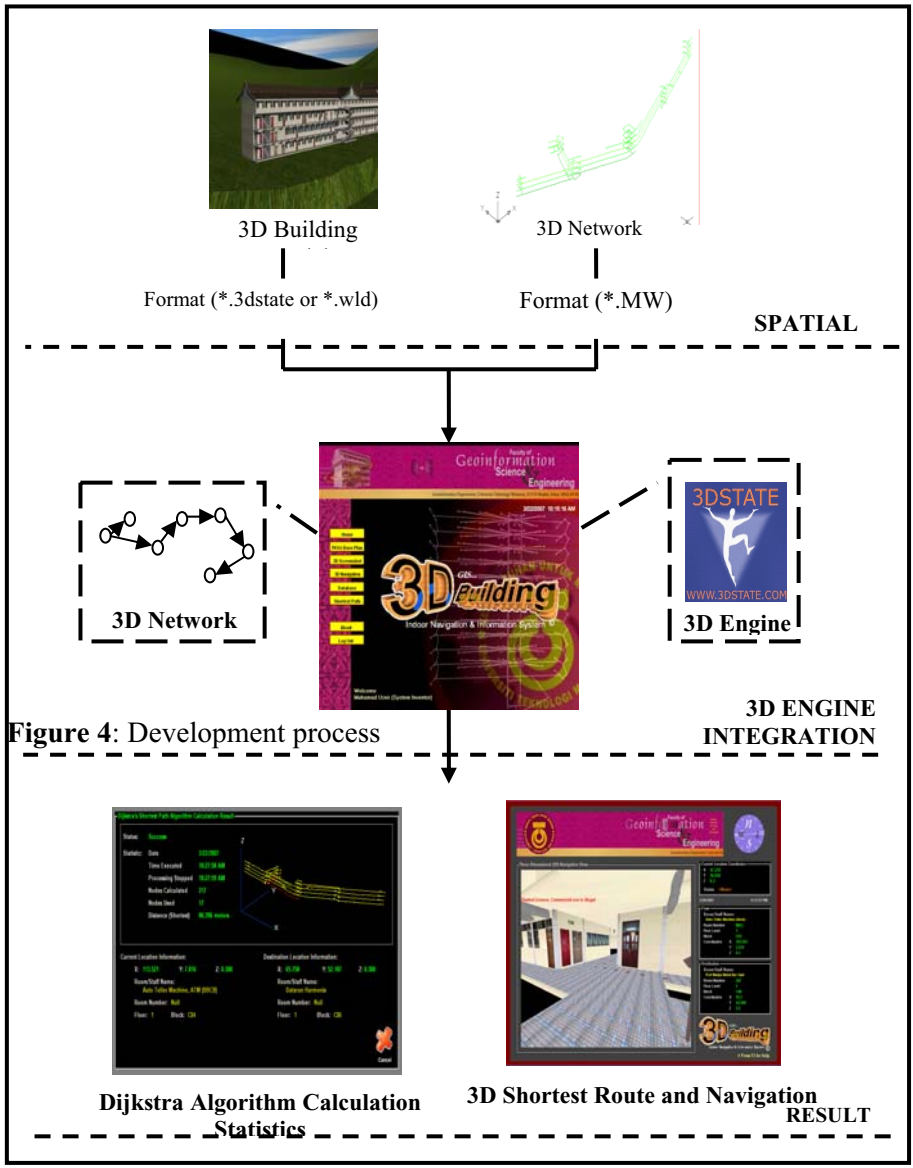

Figure 4: Development process

Figure 4: shows the database initialization by selecting the 3D network data file and also the 3D environment file.

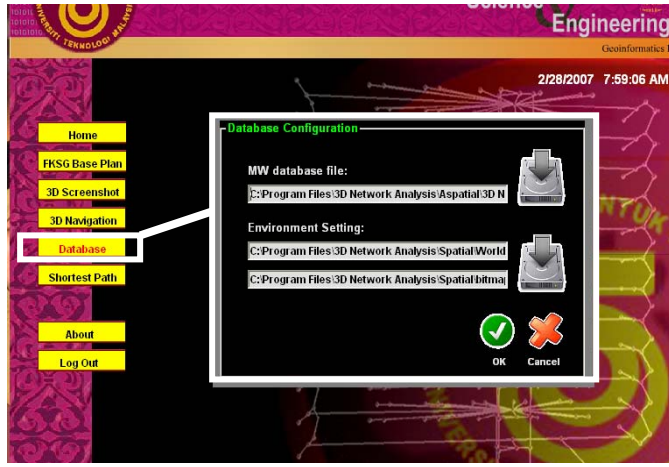

Figure 5: Database Initialization

Database initialization purpose is for 3D State engine to initialize the correct file to use in the application before executing the 3D State API command. The engine will read and open the database file by executing the following command:

STATE_engine_load_world(char *world_file_name, **char** *world_directory_path, **char** *
bitmaps_directory_path, **int** world_mode);

Before running the shortest path calculation, a user needs to select their current location and destination locations (e.g. based on Room's ID or Staff Name) prior to begin the Dijkstra's shortest route calculation (see Figure 6). The user can also acquire the required information by selecting the Building blocks, Floors, Room's ID and Room's Name of the building by using the Advance Search windows

(see Figure 7). These windows can be useful for unfamiliar users of the building and have no prior knowledge of the building as in the case of rescue operation personnel. The information at the combo box in the Shortest path windows and Advance Search windows is based on the 3D network file that set by the user during the database initialization (Figure 7). After the required information for the current and destination location has been set, the Dijkstra's algorithm can be executed/run for the 3D Network calculation as illustrated in Figure 8. It shows the Dijkstra's Shortest Path Calculation Result windows for the 3D network based on the user current location and destination location.

Figure 6: Set Current and Destination Location

Figure 7: Advance Search

The shortest path calculation windows able to display the information about the *Status*, *Date*, *Calculation* time, *Nodes Calculated*, *Nodes Used*, *Shortest Distance*, and both information about current and destination location (including the x, y and z coordinates). The Status of the calculation is the information whether the executable calculation is successful or the destination location is not reachable for the user to navigate to. *Shortest Distance* shows the distance (shortest) in meters from current location to destination location. The system also able to show the shortest route graphically in the 3D network as illustrated (see Figure 8). We highlight the shortest route of the 3D navigation in red colour.

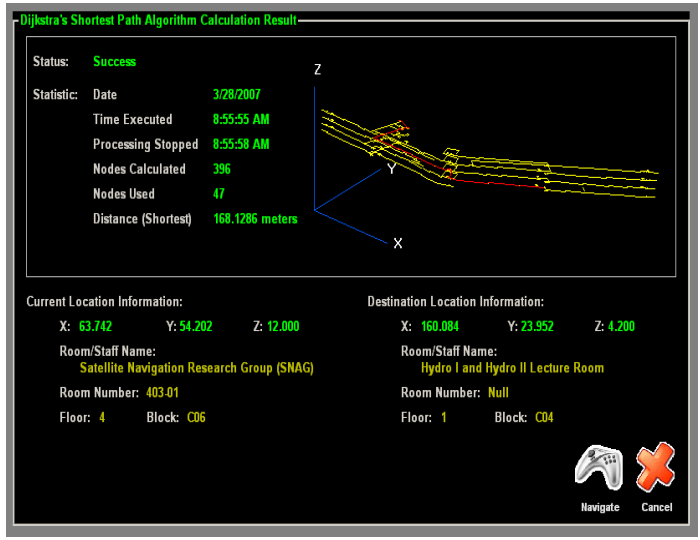

Figure 8: The Shortest Route Calculation Windows

User of this navigation system also able to control the way he or she would like to navigate. Users can move from one point to another in the *FKSG* 3D building environment without having a prior knowledge about the building and they are also able to see the real textures of the 3D building. This is the part where 3D game engine is used for simulation and visualization. To navigate inside the building, the collision detection is needed and also the possible camera movements need to be considered. 3D State game engine enable that by setting some basic parameters for example:

STATE_engine_is_movement_possible_camera_space,
STATE_object_is_movement_possible,
STATE_camera_move_with_collision_detection.

In terms of realistic movements, implementation is made by simulating human movement so that users can navigate as like they were walking inside of the building (see Figure 9).

Figure 9: 3D Navigation View

The shortest path (result) is shown by the arrow pointing towards the destination location. Users can follow the arrow until it reaches the destination area. The current location coordinates section at the top right window indicates the current user coordinate in the 3D environment. It changes to a new value if the user moves. 3D State game engine also allows input from user using computer input hardware. In this project, we are using navigational arrow from the keyboard for the directions of the user movements in the 3D environments. To rotate the camera; **STATE_camera_rotate_x** (**DWORD** camera_handle, **double** degrees, **int** space_flag) command is used if the rotation is based on x axis. It also applied for the other axis.

Figure 10: 2D Map

Another capability of the engine that is usefull for the navigation is the usage of the 2D map in the main navigation view (see Figure 10). The 2D map can assist user to navigate inside of the building by indicating the user location and displaying it using the red circle symbol in the map. The destination location target is the green circle symbol.

5.0 CONCLUDING REMARKS

From the result presented in the preceding section, it appears that the Dijkstra's algorithm can be implemented in the 3D network environment. It acts mostly like Dijkstra's algorithm implementation into 2D environment. The difference for 3D network is that it needs extra calculations for distance to include the z coordinates value by using the mathematical formula.

In this project Dijkstra's algorithm are implemented for 3D navigation by incorporating inexpensive 3D game engine and capable of providing shortest route information. It is an operational navigation software and could be extended for much larger 3D world. The approach works and certainly is useful for any agencies and personnel who deal with stress or non-stress operations like building emergency rescue and missions, evacuations and training purposes. Implementation of Dijkstra's algorithm in 3D network makes the shortest path analysis for multi-levels buildings can be more accurate and efficient for navigation or simulation in a building.

Further research to integrate volume in 3D environment with 3D network could produce better GIS application in future. With some support from 3D database, many analyses in 3D environment can be addressed and hopefully this kind of research would enhance 3D GIS development and 3D network analysis specifically. Our future work is to extend the approach by utilizing more stable 3D game engine,

incorporating with more realistic navigation tool and also to be able to manipulate 3D objects for indoor and outdoor 3D navigation.

REFERENCES

Africawala, A., Castillo, J., and Schubert, J. (2004). GIS Management and Implementation Gisc/Poec 6383. 3D GIS Technology Assessment Report.

Balstrom, T. (2001). Identifying Least Cost Routes in Mountainous Terrain. http://gis.esri.com/library/userconf/proc00/professional/papers

Beck, M. (2002). Realisierung eines Geoinformations systems – Visualisierung und Analysefunctionalalitat mit einur 3D Engine. Master thesis, Stuttgart University, Institute for Photogrammetry (ifp), Germany.

Chen, X., Meaker, J. dan Zhan, F.B. (2005). Agent-Based Modeling and Analysis of Hurricane Evacuation Procedures for the Florida Keys. Texas Center for Geographic Information Science, Department of Geography, Texas State University

Cherkassy, B., Goldberg, A., dan Radzik, T. (1991). Shortest Paths Algorithms: Theory and Experimental Evaluation. Central Institue for Economics and Mathematics, Krasikova, Moscow, Russia.

Fritsch, D. (2003). 3D building visualization – Outdoor and indoor applications. Photogrammetric Week Proceedings, Stuttgart, Germany.

Karas, I.R., F. Batuk, A. E., Akay, and I. Baz (2006). Automatically extracting 3D models and network analysis for indoors. In. Innovations in 3D Geo Information Systems, Springer-Verlag, Heidelberg, pp. 395-404

Meijers, M., Zlatanova, S., dan Pfeifer, N. (2005). 3D Geo-Information Indoors: Structuring For Evacuation. Delft University of Technology, the Netherlands.

Pu S., and S. Zlatanova (2005). Evacuation route calculation of inner buildings. Geoinformation for disaster management. Springer-Verlag, Heidelberg, pp. 1143-1161.

Semanta, S., Jha, M., dan Oluokun, C. (2005). Travel Time Calculation with GIS in Rail Station Location Optimization. Department of Civil Engineering, Morgan State University, Baltimore, USA.