Development of a Virtual Reality Application based on Libgdx Game Engine in Mobile Environment

Nasser Tahani*, Ali Asghar Alesheikh*, Mahdi Farnaghi*

* Faculty of Geomatics Engineering, K. N. Toosi University of Technology, Tehran, Iran

Extended Abstract

Presentation of real-world objects plays an important role in Geographic Information System (GIS) applications to provide different analysis capabilities and support users' perception about the environment. Mobile devices in particular, present the user's surrounding environment as 2D maps like Google Maps and OpenStreetMap (OSM) which provide capabilities like user navigation, exploring Point of Interests (POIs) and route finding. These types of maps are limited to a top-down view, and they do not truly support interaction with real-world objects by ignoring the vertical aspect of the environment. In contrast, 3D city models that contain common elements like buildings, vegetation areas, streets and surface terrain, improve users' visual perception of environment and provide more analytical capabilities based on the 3D nature of objects. In a 3D city model users can identify nearby objects according to their shapes and appearance attributes. Furthermore, these models can be utilized in variety of applications such as visibility analysis, utility management, 3D cadaster, indoor positioning and energy demand estimation (Biljecki et al. 2015). Recent improvements in the hardware and software specifications of smartphones, along with utilization of different embedded sensors and powerful CPUs and GPUs, encourage developers to apply them in novel mobile applications like Virtual Reality (VR) and Augment Reality (AR). VR is a technology that simulates objects of real world in a virtual environment and enables users to virtually interact with those objects.

In this research we have developed a mobile application to represent real-world objects in two approaches: bird's-eye view and first-person view. The bird's-eye view is similar to the view of applications like Google Earth and ArcScene, while in the first-person view, the VR approach is implemented by means of location awareness of the mobile device. The flowchart of *fig-*

LBS 2016
13th Conference on Location-Based Services
Vienna, 14–16 November 2016

Published in "Proceedings of the 13th International Conference on Location-Based Services", edited by Georg Gartner and Haosheng Huang, LBS 2016, 14-16 November 2016, Vienna, Austria.

ure 1, presents the steps of our research and the explanation of steps will be described in detail.

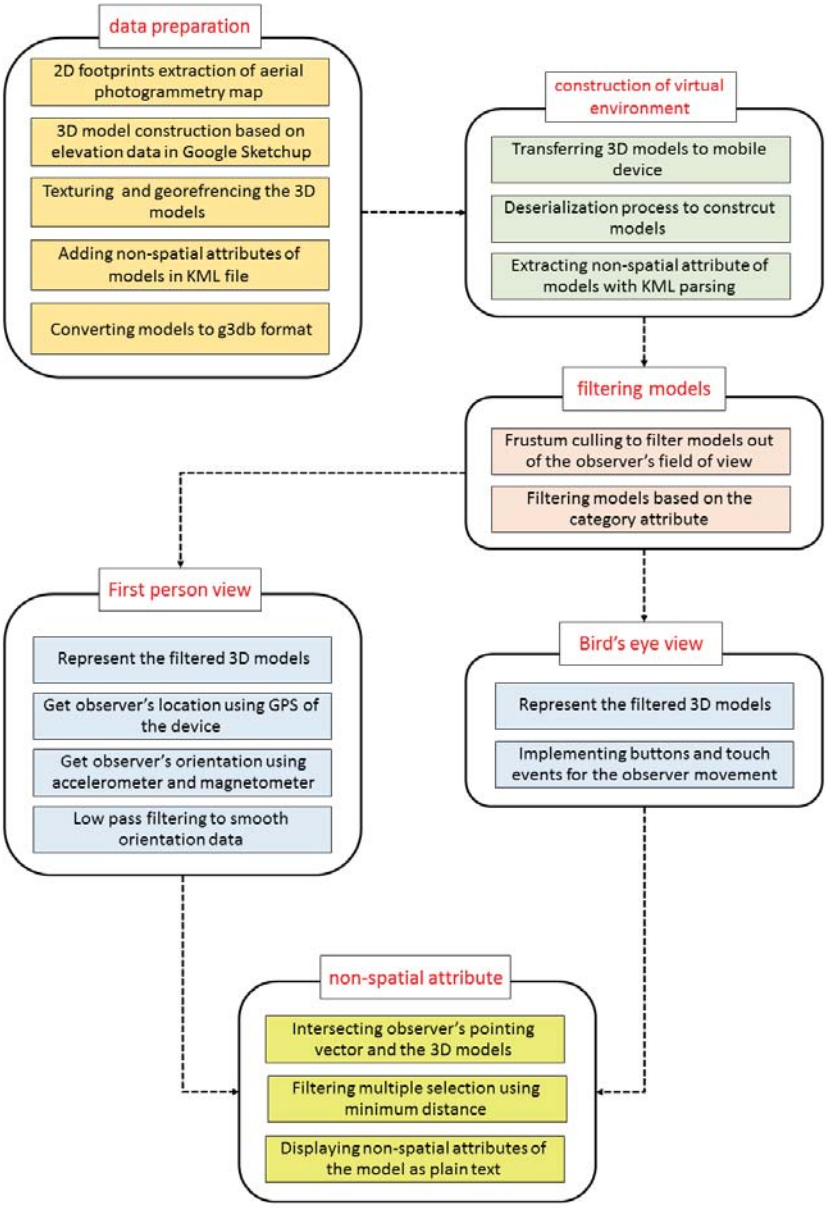

Figure 1. The flowchart of our research.

Presentaion of 3D city models on mobile devices can be devided in two categories based on rendering architecture: browser-based and client-server architecture. In the first group, 3D city models are visualized on the browser of mobile device as a web application (Prieto & Izkara 2012, Rodrigues et al. 2013). HTML5 and WebGL are widespread technologies of the first category. In the second group, data are stored in server-side and the client renders and visualizes the transferred 3D models on the mobile device (Nurminen 2008, Baldauf & Musialski 2010, Lethinen et al. 2012). The browser-based approach requires appropriate network connection; they may not have access to all smartphone sensors. On the other hand, in the client-server approach all hardware features of the mobile device can be accessed to make the application location-aware. But it needs more developing efforts due to different operating systems of mobile devices. In this research, the application have been developed accroding to the client-server approach to utilize location and orientation based sensors of the device.

OpenGL ES, a powerful API for rendering 2D and 3D features, provides the required functionalities to render graphic models in mobile devices. OpenGL ES utilizes the smartphones' GPUs to accelerate 3D graphic rendering. Additionally, various graphic libraries and game engines have been developed recently based on OpenGL ES. JmonkeyEngine3, Godot and Libgdx are instances of these libraries which provide features like shading, lighting, loading complex models, managing graphic resources and detecting collisions. In the developed application, we have used Libgdx game engine to make our virtual environment along with loading and rendering 3D models in the application.

3D city models are created with different methods like LIDAR (Light Detection And Ranging) and photogrammetric technologies. 2D city plans can be used to create 3D models in graphics softwares such as Google Sketchup and CityEngine as well. In our study, the city model contains objects in the campus of K. N. Toosi unversity of technology. The 2D city plans have been produced by aerial photogrammetry in 1:2000 scale and contains most city features like buildings, vegetation and streets. The 3D models were created in Google Sketchup software based on the 2D footprint and corresponding objects elevations. Then, the models were georeferenced and their coordinates and non-spatial attributes were exporeted as KML file. Regarding the limitation of obj format in rendering complex graphics, it is recommended to convert graphic objects to Libgdx standard format known as g3db (Bose 2014). Every objects of the 3D city model consists of three parts: the graphic part as .g3db file, the textures file and the KML file. We have utilized the deserialization process to construct graphic models in mobile device and locating them according to the coordinate attribute of KML file as shown in *figure* 2. Deserialization is the process of creating ob-

jects from sequence of bytes which resides on a storage space. The non-spatial attributes of models are extracted by parsing KML file and storing them in the set of key-value pairs.

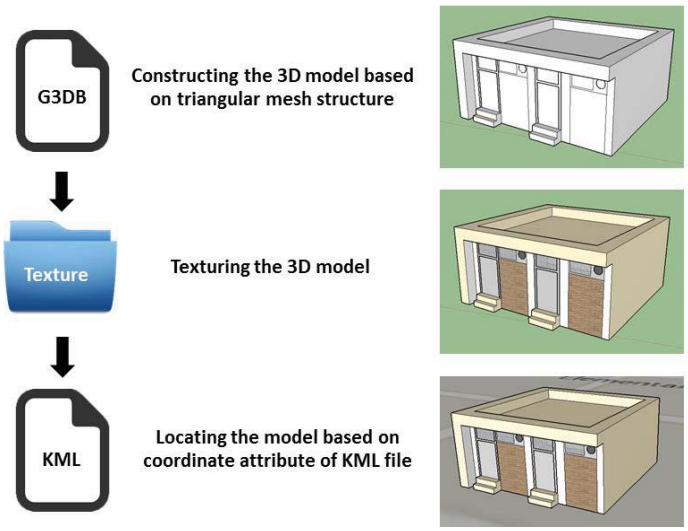

Figure 2. The deserialization of the 3D model.

Our developed application support two view approaches to represent 3D city model. In the bird's-eye view, the observer is elevated above the ground and can explore real world objects. In this view, exploring in the environment is provided by means of designed buttons and touch events on the screen as shown in *table 1*.

Event	Interaction
top-down button	observer moves up & down
left-right button	observer moves left & right
one finger touch	rotation of the observer
one finger long press	get non-spatial attribute of the object
two finger touch	zoom in & zoom out

Table 1. The task of touch events and buttons in the bird's-eye view.

Figure 3 demonstrates the campus in the bird's-eye view approach. At the top left of the screen, the attribute information of the main building is shown which includes information like its category, name, number of floors and area.

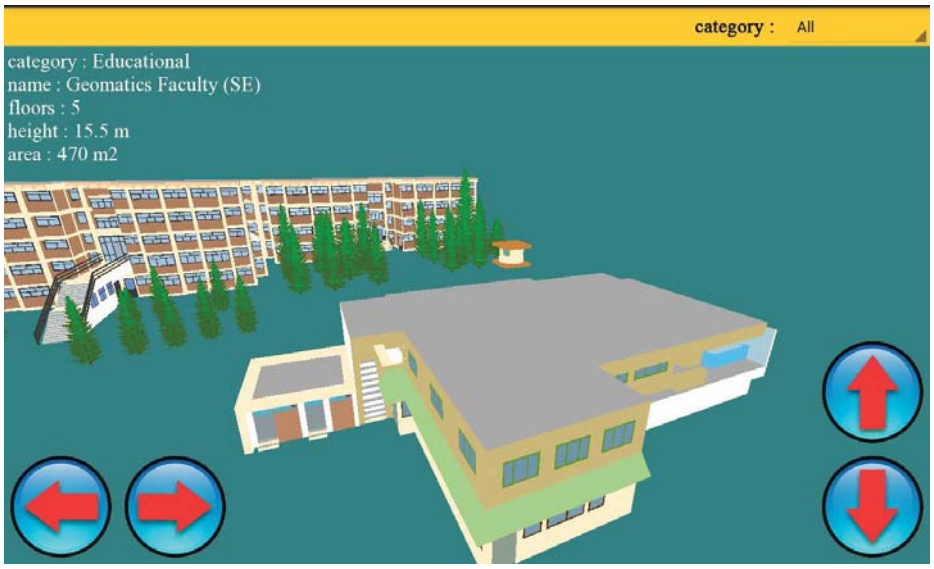

Figure 3. The bird's-eye view of the campus.

In the first-person view, we have utilized the location and orientation information extracted from sensors of mobile devices, like GPS, accelerometer and magnetometer. Unlike the bird's-eye view approach that controls the observer using buttons and touch events, in the first-person view, user's exploration in the virtual environment has been provided by aid of context awareness of the device. In the VR approach, the position of the observer is acquired by GPS and the pointing vector is determined by combining accelerometer and magnetometer data. Considering small sizes of accelerometer and magnetometer sensors, the retrieved data are noisy and need to be filtered. We have applied a low-pass filter as shown in *equation 1* to eliminate noise of the data.

$$y_i = y_{i-1} + \alpha * (x_i - y_{i-1}) \qquad (1)$$

In this equation, the y_i is the output data, α is the threshold of the low-pass filter and the x_i is the input data retrieved from orientation sensors. *Figure 4* demonstrates the Faculty of Geomatics Engineering in the first-person view. The displacement of the models is caused by GPS positioning and the orientation errors as shown in the picture.

Figure 4. Presentation of the Faculty of Geomatics in the first-person view.

Typically, rendering process of complex graphics imposes work overload on mobile devices. Also due to the limited screen size of smartphones, it is not appropriate to load all graphic objects on the scene. We have utilized the concept of view frustum culling to overcome these problems. In this procedure, we have created the minimum bounding box (MBB) of objects and constructed the frustum of observer. The designed algorithm detects objects which are contained in the observer's frustum to render them at the graphic scene.

Considering the limited screen sizes of the mobile devices, the representation of all objects makes user confused among various 3D models. The developed application provides filtering data according to the category of the models to represent only selected layers of models. As mentioned, non-spatial attributes of models have been stored as set of key-value pairs. We have used collision detection concept to identify those objects that are selected by the user. In this approach, the object is selected if the observer's pointing vector intersects with the MMB of the object. In the case of multiple intersections, the minimum distance between observer and the models are taken into account to determine the desired object. Then, non-spatial attributes of selected objects are displayed as plain text on the screen.

The developed application helps users to explore urban environment in two views. In the bird's-eye view users can explore 3D city model using designed

buttons and touch events. While in the first-person approach, the user can explore 3D objects as a VR application. To optimize the rendering process of the application, the view frustum culling has been applied to exclude those objects that are out of observer's field of view. Moreover, users can get attribute information of objects in both approaches by selecting the desired object. The filtering capability of the application supports the user to manage the representation state of different model types.

In spite of the improvement in smartphones hardware and software, simultaneous rendering of whole contents of a city model on the mobile device is an inefficient task. As a future work, in order to tackle this problem; we will implement indexing methods in client-side to interactively search and access spatial objects.

References

Baldauf, M., & Musialski, P. (2010). A device-aware spatial 3D visualization platform for mobile urban exploration. In The fourth international conference on mobile ubiquitous computing, systems, services and technologies (UBICOMM 2010), IARIA.

Biljecki, F., Stoter, J., Ledoux, H., Zlatanova, S., & Çöltekin, A. (2015). Applications of 3D city models: state of the art review. ISPRS International Journal of Geo-Information, 4(4), 2842-2889. in 1983)

Bose, J. (2014). LibGDX Game Development Essentials. Packt Publishing Ltd.

Lehtinen, V., Nurminen, A., & Oulasvirta, A. (2012, March). Integrating spatial sensing to an interactive mobile 3D map. In 3D User Interfaces (3DUI), 2012 IEEE Symposium on (pp. 11-14). IEEE.

Nurminen, A. (2008). Mobile 3D City Maps. IEEE Computer Graphics and Applications, 28(4), 20-31.

Prieto, I., & Izkara, J. L. (2012, August). Visualization of 3D city models on mobile devices. In Proceedings of the 17th International Conference on 3D Web Technology (pp. 101-104). ACM.

Rodrigues, J. I., Figueiredo, M. J., & Costa, C. P. (2013, July). Web3DGIS for City Models with CityGML and X3D. In IV (pp. 384-388).