THREE-DIMENSIONAL REPRESENTATION OF GEOGRAPHIC DATA IN A WEB-BASED GIS

MARCIN KULAWIAK

Gdansk University of Technology
Narutowicza Street 11/12, 80-233 Gdansk, Poland
Marcin.Kulawiak@eti.pg.gda.pl

Presenting geographic data in a web environment has been a long standing problem. For many years the low performance of web browsers has limited the visualization of spatial data to only two dimensions. More recently, the introduction of open standards for 3D acceleration of web applications sparked the emergence of new methods of presenting three-dimensional data in a web browser without using third-party extensions. However, these solutions are relatively young and not yet fit for use in a production environment. This work presents a method of three-dimensional representation of geospatial data in the context of two-dimensional map. The presented system uses the well-proven OpenLayers library as well as HTML5 and WebGL technologies to display three-dimensional geospatial data in the web without the need for additional web browser plug-ins. The presented application may thus constitute a viable solution until the more robust solutions mature.

INTRODUCTION

Geographic data is innately multidimensional, and as such should be analysed in three or more dimensions. However, due to millenia of technological constraints, its visualizations has been mostly confined to two dimensional maps. Even the introduction of computer engineering and Geographic Information Systems (GIS) has not caused an immediate change in this aspect. Once three-dimensional desktop GIS software has become widely available, regular GIS applications have already evolved and moved to the Web. Over the years, limited performance of Web browsers has held back the ability of Web GIS to display anything more than two-dimensional data. Different attempts at presenting three or four dimensions have either required the installation of specialized and platform-dependant browser plugins [1] or displayed spatio-temporal data in the form of animations embedded in the context of a two-dimensional map [2]. More recently, with the wide adoption of HTML5 standards such as WebGL, new projects started to emerge. These applications promise the full functionality of three-dimensional desktop GIS available to the users of most modern Web browsers.

However, these applications are relatively young and it will take years before they may be used in a production environment. The presented work proposes a solution which provides a middle ground between well-proven two-dimensional Web mapping and fully-fledged three-dimensional Web GIS.

1. THE APPLIED SOFTWARE

WebGL

WebGL (Web Graphics Library) is a JavaScript API for rendering interactive 3D graphics and 2D graphics within any compatible web browser without the use of plug-ins. It is an open standard specified by the Khronos Group consortium with the participation of major Web browser vendors such as Apple, Google, Mozilla and Opera [3]. Version 1.0 of the WebGL specification was released in March 2011, and it states that WebGL programs consist of control code written in JavaScript and shader code that is executed on a computer's Graphics Processing Unit (GPU).

Because of the early involvement of key browser developers, WebGL has been well connected with other emerging Web standards such as HTML5. Thus, WebGL spatial elements may be presented inside a HTML5 canvas element and may be accessed using Document Object Model interfaces. This allows, for example, GPU acceleration of physics and image processing effects. Features such as dynamic scene modification or automatic memory management are provided as part of the JavaScript language. Moreover, the implementation of WebGL in current versions of Mozilla Firefox of Google Chrome is nearing completion and allows end users to execute complex 3D applications.

JavaScript

JavaScript is an interpreted computer programming language originally implemented as part of web browsers. It was developed at Netscape in 1994 and introduced as part of version 2.0 of the company's Navigator browser. Since then it has seen wide adoption and extension by developers of many different web browsers. Because this situation threatened the language's integrity, in an attempt to create a common standard of a dynamic scripting language, the ECMAScript specification has been published in 1997 by the ECMA General Assembly. Since then the document has seen several revisions, with the latest version of ECMAScript 5.1 being published in June 2011 [4]. JavaScript allows developers to implement scripts that operate on the client side, which enables them to interact with the user, control the behavior of the web browser, alter the content of displayed documents and communicate asynchronously with remote servers [5]. JavaScript is object-oriented, prototype-based, dynamic and weakly typed. JavaScript also exhibits several similarities with the Java programming language, including syntax and naming conventions. JavaScript's standard library offers many functions may be familiar to Java programmers, and JavaScript's Math and Date objects are directly based on classes from Java 1.0 [6].

Recently, JavaScript is becoming widely adopted in applications outside of web browsers. These include mobile applications [7], PDF documents, desktop widgets and even server-side web applications [8].

2. THE PROPOSED SYSTEM

Three-dimensional geographical data is most often collected via specialised hardware. In particular this may include satellite stereo pair images, LIDAR scanners or multibeam echosounders which produce huge point clouds. This type of data is usually stored in sub-optimal formats and as such is not fit to be directly visualized in a 3D form. This applies in

particular to the web browser environment, which exhibits additional constraints introduced by limited available memory and processing power. Because of this, any type of data that is to be visualized in the web context should be appropriately processed and optimized for presentation in this particular environment.

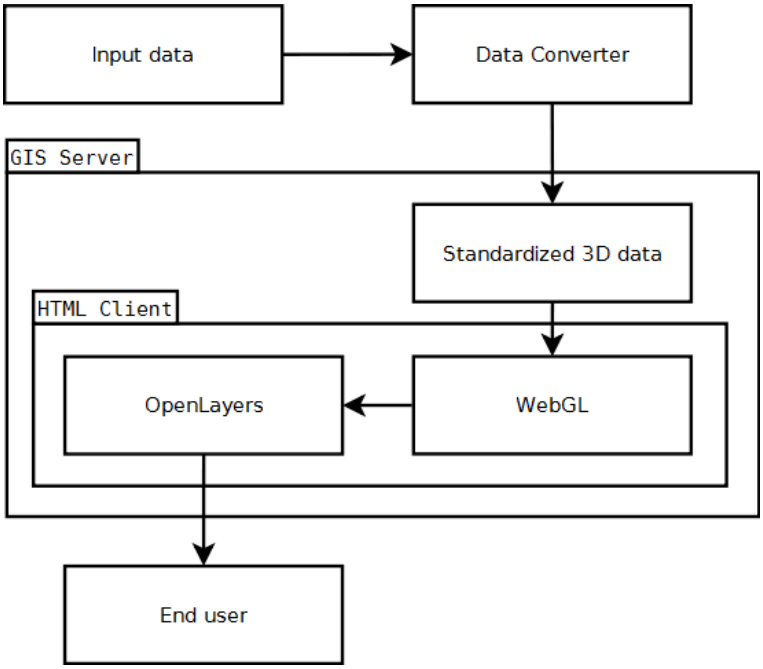

Fig.1. Diagram of the system components.

In this context, the proposed system consists of several modules, as shown in Fig. 1. Because the system is Web based, its main module is realized in the form of a GIS Server. The GIS Server acts as a container for spatial data and GIS libraries necessary to construct a visualization for remote clients. Because the input data may be acquired from several different sources, it must be converted to a common format before it may be displayed online in the HTML client. To this end, the system provides a Data Converter which recognizes several types of three-dimensional data and allows the user to save their contents in the form of a unified file containing optimized indexed vertices [9]. Aside from simple conversion, the application offers the ability to normalize, rotate and scale the converted models.

The resulting files are stored in a dedicated database on the GIS Server, from where they may be accessed by the HTML Client. The HTML Client is realized with the use of the OpenLayers JavaScript library. OpenLayers implements a constantly developing JavaScript Application Programming Interface (API) for building rich web-based geographic applications with no server-side dependencies. In some ways it behaves similarly to API's such as Google Maps and MS Bing Maps, however OpenLayers is Open Source, which makes it free for modification and redistribution. In addition, it supports open protocols such as OGC WMS and WFS, which makes it an excellent platform for data exchange [10]. The client provides basic functionality such as navigating the map and switching visible layers, while also acting as a platform for integrating 3D visualizations. The OpenLayers map contains a dedicated layer of markers, which pinpoint the locations of collected three-dimensional data. Every marker is associated with a certain three-dimensional model as well as a popup which contains a HTML5 canvas element. When the element is initiated, it opens

up the appropriate model file via an Ajax request to the server, loads the indexed vertices and displays the model in three dimensions via WebGL. The end user may then view, rotate and zoom the three-dimensional model using the mouse. This way the system achieves three-dimensional presentation of spatial data in the geographical context of a two-dimensional map.

3. RESULTS

This section contains sample results of the application of the proposed system to visualization of three-dimensional data in the context of a two-dimensional map.

When opened, the Web GIS client shows the user a standard OpenLayers map with several selectable background layers. In the following examples the background layers were imported from the Google Maps service via its JavaScript API. The map also consists of an overlay layer which contains a selection of markers showing the locations which contain three-dimensional data. The user may navigate the map by zooming and panning it using the mouse as well as dedicated control buttons located in the map's upper left corner. A sample view of the main content of the HTML client is shown in Fig. 2.

Fig.2. A sample map with a markers layer overlaid on a Google Maps background.

When the user decides to view the contents of a marker, he may click the marker icon, which will trigger the presentation of the associated popup cloud. When the cloud is triggered, the HTML client makes an Ajax request to the Web GIS server and retrieves the appropriate data file. When the file containing indexed vertices for the 3D model has been downloaded, the popup cloud is drawn, and a WebGL canvas is established within it. The three-dimensional model is then drawn on the WebGL canvas. The user may then interact with the model inside the WebGL canvas by moving the mouse inside the popup cloud window. The three-dimensional model within the popup may be freely rotated or moved and the user perspective may be changed by zooming in or out, as shown in Fig. 3. The user may close the popup cloud by clicking the button located in the popup's upper right corner (encircled in red on Fig. 3).

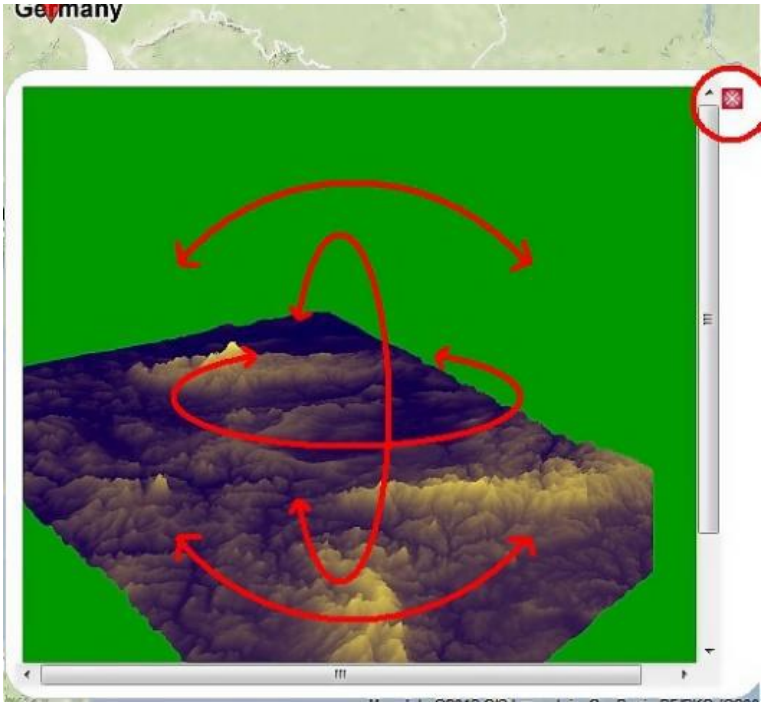

Fig.3. Orientation of 3D data in the context of a 2D map.

The system can present multiple types of three-dimensional data from a variety of sensors. The example in Fig. 3 shows a Digital Terrain Model of the city of Kassel in Germany. If the collected data is not fit to be presented in the form of a solid 3D model, it may be visualized as a cloud of three-dimensional points. This is the case of multibeam sonar bathymetry data depicting a shipwreck near the coast of Norway, which is presented in Fig. 4.

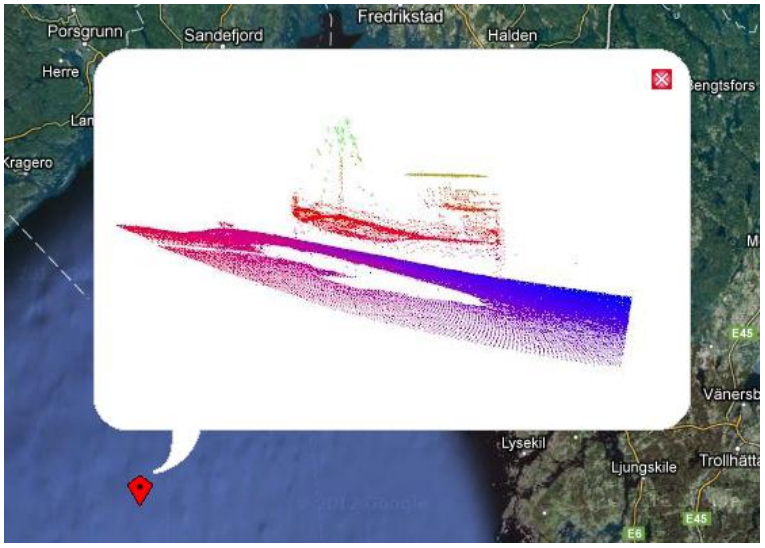

Fig.4. Sample 3D visualization of multibeam sonar data in the Web GIS.

4. SUMMARY

The presented results show that the proposed system constitutes a viable solution for presenting three-dimensional information in the context of a standard two-dimensional map. This solution will work in any Web browser which supports modern open standards such as WebGL. The solution utilizes well-proven libraries such as OpenLayers and is relatively easy to implement. The presented application may thus constitute a viable alternative to presenting three-dimensional data in geographical context in a web environment until a more complex solution matures enough to be used in a production environment.

ACKNOWLEDGEMENTS

This research was sponsored by the National Centre for Research and Development for the purposes of national defense and security.

REFERENCES

[1] C. Zhu, E. C. Tan, T. Kai, Y. Chan, "3D Terrain visualization for Web GIS", Map Asia 2003, Kuala Lumpur, 2003.

[2] M. Kulawiak, A. Chybicki, M. Moszynski, Web-based GIS as a tool for supporting marine research, Marine Geodesy Volume 33, Issue 2 & 3, p. 135-153, Taylor & Francis, 2010.

[3] C. Marrin, WebGL Specification 1.0, https://www.khronos.org/registry/webgl/specs/1.0/ [Accessed 20.03.2013].

[4] ECMA International, ECMAScript Language Specification, http://www.ecma-international.org/publications/files/ECMA-ST/Ecma-262.pdf [Accessed 20.03.2013]

[5] D. Flanagan, JavaScript: The Definitive Guide (5th ed.). O'Reilly & Associates, 2006 ISBN 0-596-10199-6.

[6] B. Eich. "Popularity", http://brendaneich.com/2008/04/popularity/ [Accessed 20.03.2013].

[7] PhoneGap, http://phonegap.com/ [Accessed 20.03.2013].

[8] Node.js, http://nodejs.org/ [Accessed 20.03.2013].

[9] P. Klamant, Three-dimensional representation of geospatial data in a Web-GIS, Master Thesis, Gdansk university of Technology, 2012.

[10] C. Schmidt, OpenLayers: Free Maps for the Web. 2008 http://www.openlayers.org, [Accessed 20.03.2013].