Mátyás Gede[*]

Novel Globe Publishing Techniques Using WebGL

Keywords: globes; internet; WebGL; X3D; Cesium

Summary: Digitising old globes always brings the problem of displaying digital 3D objects. Until the recent years special software and/or plugins were required to be installed on the viewer's computer. Thanks to the emerge of WebGL, and its widespread support in virtually all recent web browsers, nowadays there are several simple ways of publishing digitised globe models on the web. This paper summarizes the traditional and new ways of globe publishing, and introduces two WebGL-based JavaScript libraries and their usage in the field of globe visualisation. The first one, X3DOM enables the embedding of X3D codes to HTML pages, while the second one, Cesium is an open-source virtual globe and map engine.

Introduction

The Virtual Globes Museum project started at the Department of Cartography and Geoinformatics, Eötvös Loránd University in 2007. One of the hardest questions was the problem of publishing interactively rotatable-zoomable 3D objects on the web. In the first few years, all the solutions required some kind of plugins on the client side: either a VRML player or the Google Earth plugin (Gede 2009). Other digital globe visualization projects of that time also used the VRML format (e. g. Krömker, Jäger 2004).
Although the base concept of various VRML globe visualizations is the same – a spherical shape draped with the globe map as a texture – there are several technical differences between them. The simplest solutions use the built-in VRML Sphere object, textured with a single Plate Carrée image of the globe map. As the size of texture images are limited in some VRML players, other solutions map several texture images to smaller quadrangles of the globe (e.g. Krömker, Jäger 2004), eventually completed by maps in azimuthal projection to the polar regions (Gede 2009).
As any kind of shapes can be defined in VRML, it is also possible to display globes together with their supporting frames. A nice example of this is the work of Zs. Ungvári, who digitally reconstructed the calendar frames of a pair of Blaeu's globes (Márton, Plihál, Ungvári 2011).
Solutions using the Google Earth plugin also drape Plate Carrée maps to the globe by defining one or more KML GroundOverlay objects. Some solutions use multiple level of details (LODs) – providing low resolution textures for overview and high resolution images for zoomed views – which can also be defined in KML. This latter technique enables the asynchronous loading of texture images (image files are only loaded when they are needed), which makes initial displaying faster.

[*] Assistant professor, Department of Cartography and Geoinformatics, Eötvös Loránd University, Budapest [saman@map.elte.hu]

X3DOM

X3DOM is an open-source JavaScript framework for embedding Extensible 3D (X3D) code to HTML pages. X3D is an ISO-standard XML-based format for declarative 3D graphics, successor of the VRML format. X3DOM is using WebGL instead of a plugin for displaying 3D objects, thus is supported by most recent browsers on desktop computers and becoming more and more accessible on various mobile platforms as well.

The X3DOM framework integrates X3D elements to the Document Object Model (DOM), which means that the 3D content can be manipulated by JavaScript code the same way as HTML elements. With only two extra lines of code (including a JavaScript file and a style sheet) X3D codes can be pasted into HTML code, and models will be displayed in the browser. *Figure 1* illustrates the source and the look of a simple HTML page displaying a 3D globe (Gede 2012).

```
<!DOCTYPE HTML>
<html>
 <head>
  <title>X3DOM globe model</title>
  <link rel="stylesheet" type="text/css"
        href="http://www.x3dom.org/x3dom/release/x3dom.css">
  </link>
  <script type="text/javascript"
         src="http://www.x3dom.org/x3dom/release/x3dom.js">
  </script>
 </head>
 <body>
  <h1>X3DOM globe model</h1>
  <x3d width="600px" height="400px">
    <scene>
      <shape>
        <appearance>
          <imageTexture url="globe.jpg"></imageTexture>
        </appearance>
        <sphere></sphere>
      </shape>
    </scene>
  </x3d>
 </body>
</html>
```

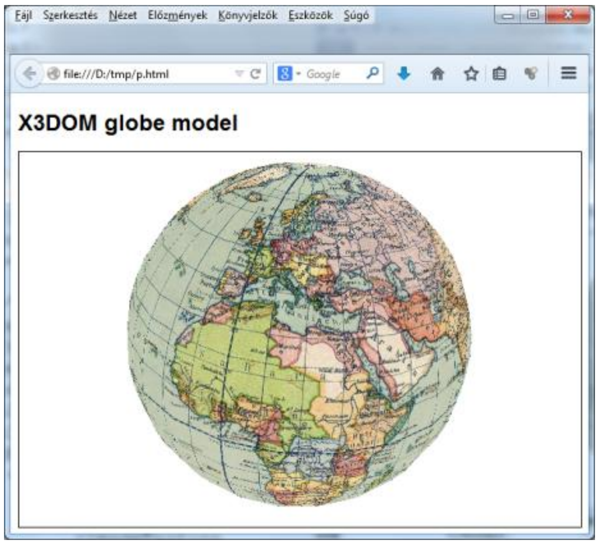

Figure 1: Sample HTML+X3D code using X3DOM and a browser viewing this code.

This example uses the Sphere node, which has the same rendering issues as its VRML version (Gede 2009): the sphere is estimated with a polyhedron with edges not matching the geographic grid lines of the globe map, which results in zig-zagging lines when zooming in.

More complex solutions usually build up the globe using self-defined shapes (usually using the IndexedFaceSet node of X3D) which enables custom texture mapping rules (e.g. using more than one texture image on one shape), and any special shapes other than a simple sphere. The Virtual Globes Museum website contains several examples of the latter case, including a two-part detachable structural-morphological Earth model shown on *Figure 2*.
Building up the sphere using several sub-surfaces should always be considered when higher resolution is desired: X3DOM fails to render even one single texture if it is larger than 4096 pixels, while it works well with several smaller images.

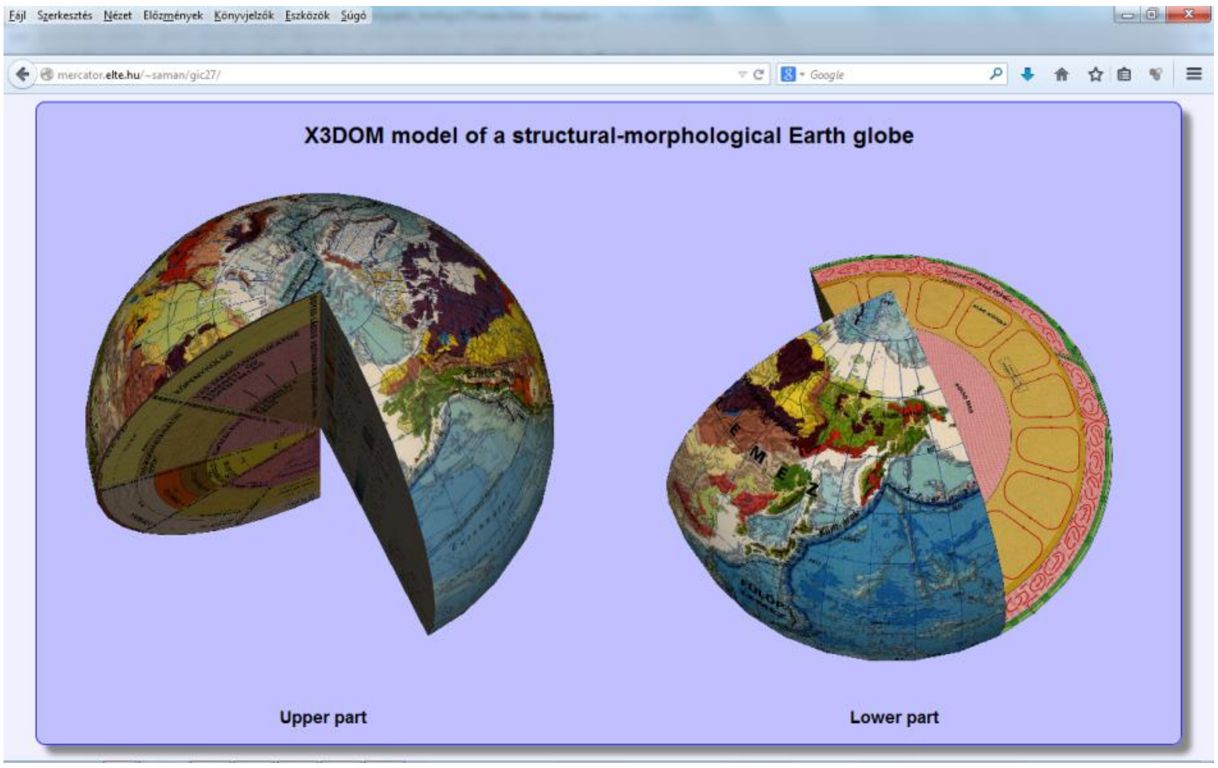
Fájl Szerkesztés Nézet Előzmények Könyvjelzők Eszközök Súgó
mercator.elte.hu/~saman/gic27/
Google
X3DOM model of a structural-morphological Earth globe
Upper part
Lower part

Figure 2: Structural-morphological Earth globe visualized using X3DOM.

As the X3D format supports all the functionality of VRML, transforming an existing website displaying VRML models to X3DOM is quite straightforward: VRML codes can be translated to X3D by various software or on-line services like the one created by InstantLabs (Fraunhofer 2014). There are two limitations: X3DOM doesn't support self-closing nodes (e.g. <Sphere />, use <Sphere></Sphere> instead) and Script nodes. Animations and other features using Script nodes in X3D can be implemented from HTML/DOM side using JavaScript as all nodes of the 3D scene became a part of the DOM.
This latter feature gives great power to X3DOM. It is possible to add new objects to the scene; to change the texture, the position or other properties of a shape. *Figure 3* displays the model of a "Combo-globe" – an openable Earth globe with a smaller sky globe in it. The outer sphere can be opened by pushing a button in the browser. The inner globe can be rotated independently from the outer one. These special effects were realized with only a few lines of additional JavaScript code.

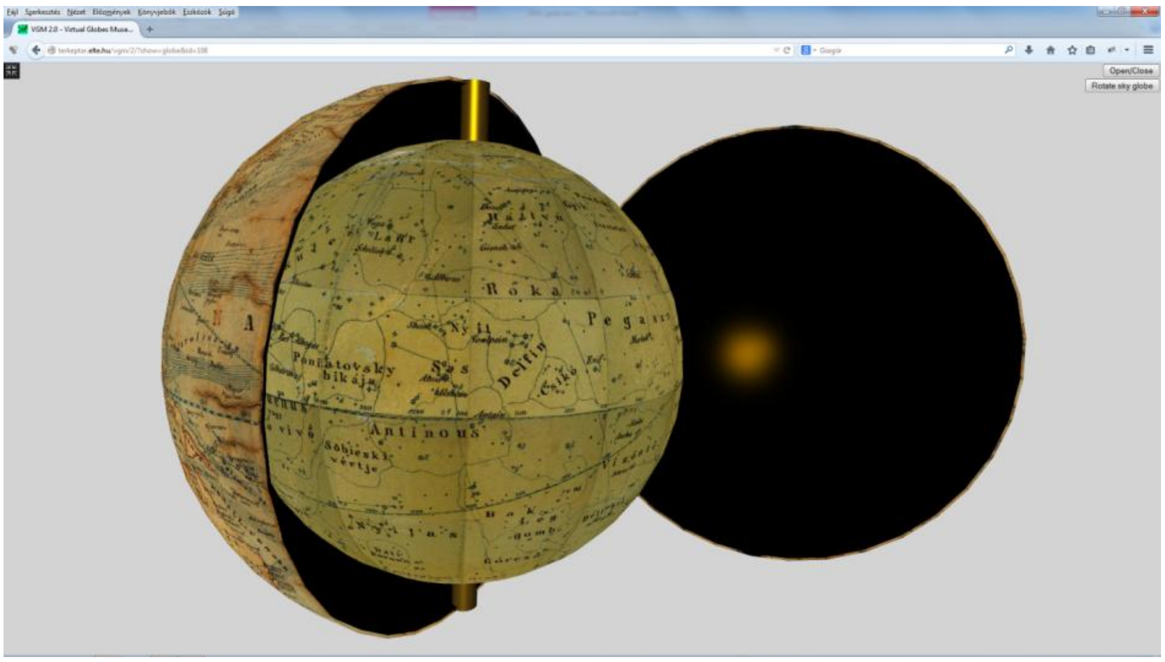

Figure 3: Model of an openable "combo-globe".

Cesium

Cesium is a JavaScript virtual globe and map library written in JavaScript using WebGL. It was developed by Analytical Graphics, Inc. (Cozzi 2013). The library offers three different views: a globe, a 2D map and a 2.5D view called "Columbus view" (– obviously the first one should be used for globes). In addition to the visualization engine, worldwide satellite imagery and digital elevation dataset are also available. Using these data with Cesium, embedding a "google-earth-like" widget into a webpage requires only a few lines of code (*Fig. 4*).

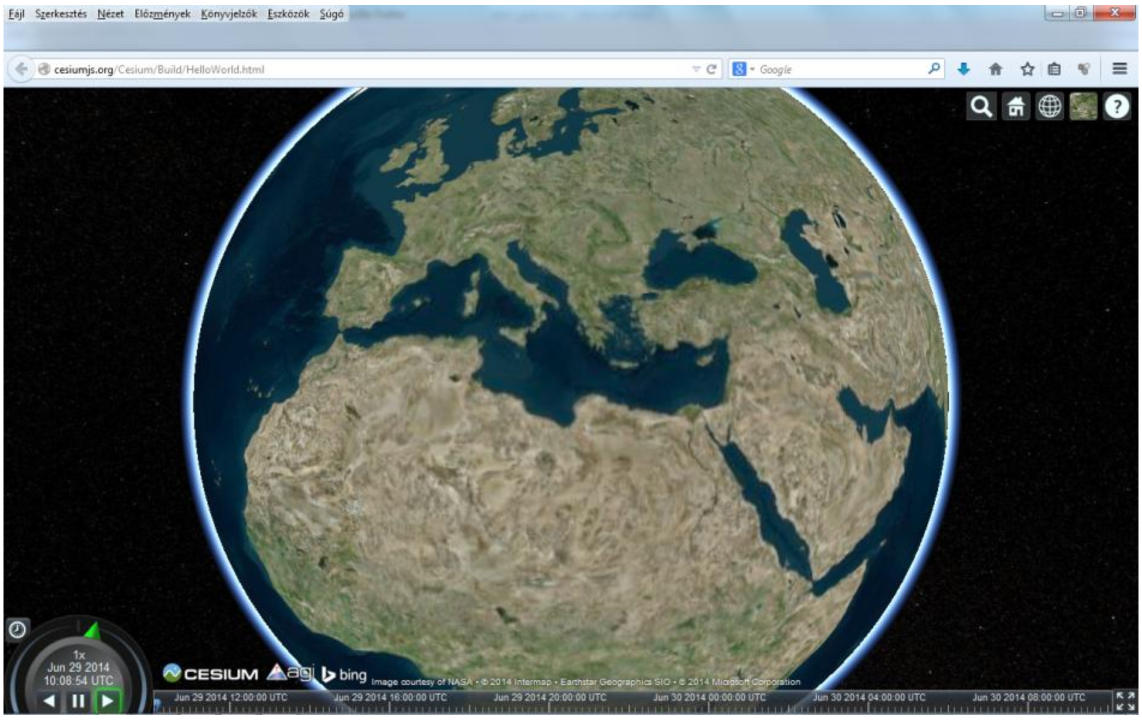

Figure 4: Virtual Earth using Cesium.

In order to display old globes using Cesium, the satellite imagery has to be replaced with the globe map. There are two approaches to achieve this. The first one is to define an overlay using the Cesium.SingleTileImageryProvider class. This class drapes a single Plate Carrée map image to the geographic quadrangle (or to the whole globe if no bounding box is given). The sample code of this solution looks as follows:

```
<div id="cesiumContainer"></div>
<script>
 var cw = new Cesium.CesiumWidget('cesiumContainer',
      { imageryProvider: new Cesium.SingleTileImageryProvider({
                        { url: 'globe.jpg' })
      });
</script>
```

The default Cesium view is completed by the Sun, the Moon, stars and atmosphere. When displaying old globes, these items are probably undesirable. The following lines of code hide them:

```
   cw.scene.moon.show = false;
   cw.scene.sun.show = false;
   cw.scene.skyAtmosphere.show = false;
   cw.scene.skyBox.show = false;
```

Figure 5 shows the resulting screen.

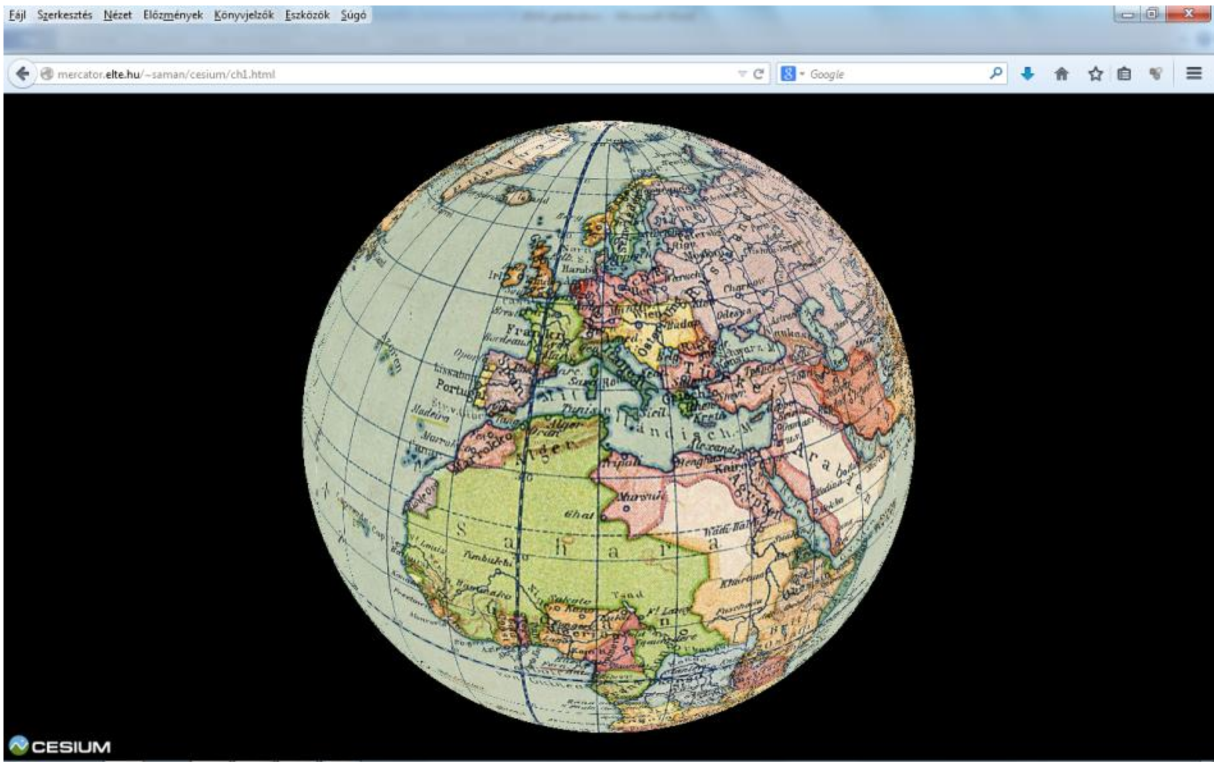
Fájl Szerkesztés Nézet Előzmények Könyvjelzők Eszközök Súgó
mercator.elte.hu/~saman/cesium/ch1.html
Google
CESIUM

Figure 5: Displaying an old globe using Cesium.

In spite of the simplicity of this solution, the rendering of the globe map is perfect, even around the poles - the usual weak points of any 3D representations where a rectangular image is draped to a sphere. The only shortage is the image size: the maximum dimensions of a single texture file are 4096 by 4096 pixels.

A slightly more difficult solution is to create a Tile Map Service (TMS) from the globe map. A TMS consists of an appropriate directory structure storing map tiles, completed by a few XML files describing the service. Note that the row numbering of standard TMS starts from the bottom, unlike Google Maps or OpenStreetMap row numbering (which starts from the top). A complete TMS specification can be found at the OSGeo website (OSGeo 2012).

When using TMS in Cesium, the more complex task is to create the tile set. Although there are a handful of software for creating TMS tiles, they usually only support the "google-style" row numbering which results misplaced tiles in Cesium. The easier task is to use the TMS: the only difference in the code is that Cesium.TileMapServiceImageryProvider is used instead of the Cesium.SingleTileImageryProvider class.

There are two major advantages of using a TMS instead of a single tile: initial rendering can be much faster, and there is no limit of resolution.

Other WebGL geobrowsers

There are several other WebGL based geobrowsers similar to Cesium. However, due to various minor issues, their usability for displaying old globes is limited.

OpenWebGlobe, created by Swiss researchers (Christen, Nebiker, Loesch 2012) also offers the possibility of custom image overlays, but rendering fails beyond the 85th latitudes, therefore map of the polar regions is not visible.

Another example is the *WebGL Earth JavaScript API*, created as a bachelor thesis by Petr Sloup, maintained by Klokan Technologies (Sloup 2011). Although this interface also supports the use of custom imagery, the images have to be in Mercator projection and tiled according to the TMS standard. Due to the use of Mercator projection, latitudes higher than 85 degrees are not displayable in this case as well.

Conclusions

The use of WebGL based globe visualizations instead of various plug-ins has several advantages: this solution is really platform-independent (plug-ins are usually available for a few platforms/operating systems) and does not require users to install any additional software.

The drawback of these solutions is performance: plug-ins optimized for a given operating system (for example, Google Earth using DirectX on Windows) usually offer faster rendering.

When designing a new website offering 3D view of globes, the question arises: which solution should be used? In most cases both X3DOM and Cesium is appropriate. Their hardware/software requirements are the same: any device with a browser supporting WebGL is useable.

X3DOM should be chosen when the 3D scene consists not only of the globe itself, but other "accessories" as well, e.g. the support frame or, if the globe has irregular (non-spherical) shape: truncated or detachable globes. In these cases we can take advantage of the X3D format – virtually any shape can be defined.

If the aim is to display a globe in very high resolution, Cesium is a better solution, using a multi-level Tile Map Service. As the size of TMS tiles are normally 256*256 pixels, it will work well even on lower performance devices.

Further plans

Raised relief globes are quite peculiar type of earth globes. Currently these objects are processed and displayed in the VGM the same way as normal globes.
Using a custom elevation dataset in Cesium makes possible to display the real shape of raised relief globes. The hard task here is acquiring elevation data from the globe: it could be done either using a laser scanner or by photogrammetric shape reconstruction. Our plan is to develop a workflow of acquiring elevation data from relief globes and using it to display them using their real shape.
Another interesting plan is to add "tags" to specific areas of globes. Tagged areas would turn highlighted when the mouse pointer hovers above them, and clicking on them would display additional information – e.g. translation of the Latin text of a title field – or draw the observer's attention to interesting details such as imaginary creatures filling the unknown areas of the map or the first appearance of an object on maps, etc. This feature can be implemented using either X3DOM or Cesium.

Acknowledgement

This research was financed by the Hungarian National Foundation for Scientific Research (OTKA PD 111737).

References

Cozzi, P. (2013). Cesium: 3D Maps on the Web. FOSS4G NA, 05/2013.
http://cesiumjs.org/presentations/Cesium3DMapsOnTheWeb.pdf

Christen M., Nebiker, S. and Loesch B. (2012). Web-based Large-scale 3D-Geovisualisation using WebGL – The OpenWebGlobe Project. *International Journal of 3-D Information Modeling* 1/3, pp.16-25.

Fraunhofer (2014). X3D encoding converter.
http://doc.instantreality.org/tools/x3d_encoding_converter/

Gede, M. (2009). Publishing Globes on the Internet. *Acta Geodaetica et Geophysica Hungarica* 44 (1): 141-148

Gede, M. (2012). The Possibilities of Globe Publishing on the Web. *In: Michael P Peterson (ed.) Online Maps with APIs and WebServices. Heidelberg: Springer, 2012*. pp. 219-238.

Krömker, S., Jäger, W. (2004). Map Projection versus Image Processing — the Role of Mathematics in the Restoration Process. *In: Der Heidelberger Karl-Theodor-Globus von 1751 bis 2000. Schriften der Mathematisch-naturwissenschaftlichen Klasse der Heidelberg Akademie der Wissenschaften* Nr. 14 (2004) Volume 14, 2004, pp 69-100

Márton, M., Plihál, K., Ungvári, Zs (2011): Restoring Blaeu's Globes by Modern Methods. *Poster, ICC 2011, Paris.*

OSGeo (2012). Tile Map Service Specification.
http://wiki.osgeo.org/wiki/Tile_Map_Service_Specification

Sloup, P. (2011). WebGL Earth. Bachelor thesis, Masaryk University, Brno.
http://is.muni.cz/th/325196/fi_b/thesis.pdf