МАТЕМАТИКА И МАТЕМАТИЧЕСКО ОБРАЗОВАНИЕ, 2013
MATHEMATICS AND EDUCATION IN MATHEMATICS, 2013
Proceedings of the Forty Second Spring Conference
of the Union of Bulgarian Mathematicians
Borovetz, April 2–6, 2013

3D RENDERING IN THE WEB BROWSER*

Vladimir Georgiev

Modern web browsers have recently increased their support for HTML5, which introduces the new canvas element used for rendering graphics and visual images on the fly. However, the interface to the canvas element is so complex, that trivial tasks like displaying a 3D model and allowing viewers to rotate it and zoom in/out are hard to achieve. Using the jQuery plugin architecture together with the 3CDL graphics library, we implemented a plugin for visualizing existing three-dimensional models in web pages, in such interactive manner, with just a few additions to the HTML markup.

Introduction. Three-dimensional technologies are crucial in fields like digital libraries and e-Learning. The presentation of 3D models of digital objects is highly appreciated and an important alternative of contemporary 2D image and video presentation of three-dimensional objects such as ancient statues, pottery or miniatures. Furthermore, using those technologies we can combine images in an interactive virtual environment, which we can use for reproducing visual and spatial characteristics of artifacts, visually compare the properties of two or more artifacts, design models of interaction between them, and perform other real-life operations in three-dimensional space.

Recent developments in the field of web browsers have improved their support for the HTML5 markup language, adding many new features for easier web development, more sophisticated form handling, semantic elements for marking up page content and support for complex graphics rendering. Even though the specification [1] is still "a work in progress", vendors are trying to keep their web browser software up to the latest standard, and now the canvas element is implemented in most major versions. This new element allows the addition of one or more three dimensional graphic scenes to any web page, varying from single model rendering to complex scenes containing multiple models, lights and other objects.

Due to the increasing demand for interactive web pages, HTML5 introduces a number of APIs (Application Programming Interfaces) for accessing its functionality and standardizing the creation of web applications. There is an interface for rendering two and three-dimensional graphics using the canvas element, which allows web developers to programmatically design and update 3D scenes in response to user events. To do that, one can choose to implement all in native JavaScript code or use third-party libraries and frameworks that make the construction and updating of the user interface, 3D ren-

*2000 **Mathematics Subject Classification:** 68N01, 68U05, 68U07, 68U35.
Key words: 3D, web browser, digital object presentation.

dering or building other components of the web page easier. Because JavaScript is an object-oriented language and most third-party frameworks allow anyone to implement their own extensions of the library, adding new custom functionality that is not already implemented is a straightforward task. The newly added functionality can be packaged and distributed to the rest of the community, so that the code can be used by other people that need it.

Currently, one can use the canvas element's rendering API for drawing 3D graphics in web pages, but because the interface is too low-level, this task would be complex and require great effort. Another difficult assignment would be linking mouse and keyboard events to the functionality required to rotate and zoom the model being viewed. This paper presents a jQuery plugin solution for showing 3D models in a modern web browser with minimum effort, realized with the C3DL graphics library. In the next section of the paper we discuss the motivation for building and the demands for such a solution. Part three describes the standard functionality a contemporary 3D JavaScript library exposes, as well as a core library used for multi-purpose development of web pages. Then we present some details and usage of the solution we built using the technologies described in the previous section. In the final section we talk about future work, plans for extending the research and improving the solution.

Requirements for 3D rendering in the Web browser. Presentation and visualization of digital objects in the web browser is an integral part of fields like digital libraries and e-Learning. In paper [2], the authors point out that traditional presentation materials such as text and images are not enough for describing the objects' properties, purpose and relations to other objects. In most cases, especially where physical access to the real artifact is not possible, an additional three-dimensional view of the digital object is needed. Digital content authors can prepare 3D models of the objects in their collections by using methods like scanning real material objects, reconstructing lost cultural heritage using authoring tools to develop the model [3], or building three-dimensional models from a set of images.

Visualization of three-dimensional models in the web browser, a process mainly known as *rendering*, has been a challenging task until recently. Because old browsers were intended to be used mainly for viewing web pages on the Internet (containing text and images), complex operations like drawing were performed by the Operating System through Add-ons, which had to be installed in users' browsers. One such component is Flash, which is still broadly used for reproducing video and sound, as well as drawing. Unfortunately, Flash is not supported by all browsers, and some people choose not to install it, ending up unable to view such web content. With the development of the Internet needs and technologies, HTML5 was proposed by members of the W3C committee, for providing the functionality needed for the next-generation web applications. For drawing purposes HTML5 provides the canvas element, which can be used for two and three-dimensional graphics rendering and displaying visual images on the fly [1].

Since the canvas element's functionality is exposed through a complex low-level API, called WebGL, several other libraries for facilitating the development of three-dimensional web applications exist. These libraries provide math, scene, material, camera, lightning and 3D model support, which makes it easier to write canvas applications without having to worry about the specific logic and calculations needed to render the scenes. Motivated by the increasing demands for including 3D content in web pages in a simple and

straightforward manner, we implemented a component (plugin) to be used for visualizing pre-built 3D models in any web page with minimum effort, targeted not only for web developers, but also for designers with little experience with technologies like that. Even though the solution is still being tested, we are considering its application in two multimedia digital libraries – Bulgarian Iconographical Digital Library [4] and Encyclopaedia Slavica Sanctorum [5].

Functionality of 3D and other JavaScript libraries. There are several JavaScript 3D frameworks wrapping the complex WebGL interface, like Canvas 3D Library (C3DL) [6] and Three.js. All of those libraries depend on the web browser's support for WebGL – they will run on browsers like Mozilla Firefox and Google Chrome, but not on Internet Explorer (at the moment of writing this article, IE does not support WebGL). However, it is expected that all browsers will conform to the HTML5 specification in the near future and have built-in support for WebGL as part of that support.

The C3DL JavaScript library mentioned above is one of the libraries providing an easy-to-use interface for three-dimensional graphics. If we have one or more 3D models prepared in advance (by a 3D authoring software such as 3DS Max or Maya), we could use that library to display them in a web page with minimum effort. We start by setting up a *scene* – a layout of our environment containing the objects, which defines the spatial relationship between them, including location and size. We then place some 3D *models* in the scene, which are produced by a 3D modelling tool or scanned from real-world objects, and exported in a common format. The C3DL library uses Collada as the source format for models it is manipulating. It is comprised of an XML file containing information about the object – like polygons, material, lightning and physics, and image files containing the *textures*. After a model is added to a scene, we could change its textures, add different *lightning* effects, and change the *material* it is built from. We should also define a *camera*, which determines the perspective the scene is viewed from, and then rotate and move the camera to any direction during the rendering process. The library also supports *particle systems* – a collection of multiple smaller objects, rendered together, like rain drops or snow. After the scene has been setup, the *rendering* process begins, which is converting the scene into the two-dimensional image the user sees, using complex calculations involving all objects in the scene like where the models, camera and lightning sources are positioned and directed to. Any object in the scene can be changed at any time – for example in response to a keyboard or mouse event, which will cause the renderer to update the scene so that the user sees the change.

jQuery is a multi-browser JavaScript library, designed to simplify the development of HTML pages, used by 53% of the most visited websites around the world [7]. It is free, open-source software, licensed under the MIT License. Because of its architecture, other jQuery developers can use its constructs to create plug-in code that extends its functionality. Currently there are thousands of plugins that cover a wide range of functionality such as web services, data grids, charts, drag and drop, events, cookie helpers, dialogs, and others.

Complete plugin solution for rendering 3D models in web pages. Using the jQuery plugin architecture together with the 3CDL library, we implemented a plugin [8] for visualizing three-dimensional models in web pages. It allows a person familiar with HTML (no JavaScript knowledge is required to use the plugin) to display a 3D view of an object. An example of how this is configured is shown below:

```
<!-- add the model_path attriute to an HTML element -->
<a class="c3dl" model_path=‘‘models/duck/duck.dae">View in 3D</a>
...
<!--initialize the c3dl_3d_viewer plugin when the web page is loaded-->
<script type="text/javascript">
$(function() {
          $('.c3dl').each(function(){ $(this).c3dl_3d_viewer(); });
});
</script>
```

The code above will open a popup window when the “View in 3D” link is clicked, which will show a 3D scene containing the model from the file entered in the *model_path* attribute. The model file is expected to be in the Collada format, which defines an open standard XML schema for exchanging digital assets among various graphics software applications that would otherwise have been incompatible with each other. In addition to the XML file there could also be texture images, where applicable. When our plugin renders a 3D model, it automatically loads all textures accompanying it. It also renders any light sources and model materials described in the file, leaving the web page designers/developers not having to worry about these details.

Once rendered, the viewer can rotate the 3D model in any desired direction, making it possible to view it from all different angles and positions. The model can also be zoomed in and out, giving the opportunity to see the tiniest detail in its shape and texture. Currently, only one model at a time can be displayed, and in order to view another one, the user will have to close the rendering popup window and click on the new model display link.

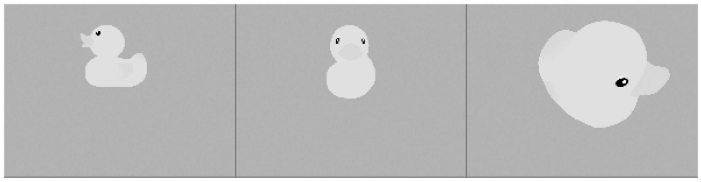

Fig. 1. Rotating and zooming 3D models in the web browser

The main functionality of the plugin we built is implemented using the jQuery and C3DL libraries. We employ the jQuery plugin architecture to develop a component allowing that functionality to be easily applied to one or more elements of a web page. After being attached to an element, the plugin inspects its attributes and if there is one named *model_path*, the functionality to render the particular object referenced in the value of that attribute is initialized. In case the element does not have that attribute, nothing happens and the initialization is skipped. The setup phase involves adding an event handler for the mouse click event on the element, which will perform the actual rendering job, and adding the model to the C3DL library, because it requires each model file to be parsed in advance, before the rendering process starts.

After being initialized, the particular HTML element is setup to respond to the mouse-click event, and when it occurs, the rendering logic begins. Rendering of the 3D model is completely done by the C3DL library. It starts by opening a modal dialog containing a

canvas element and a *Close* button for hiding the dialog. Then a scene is initialized with the id of the canvas element to use for rendering. A WebGL renderer is then prepared and assigned to the scene to render it. After that, the model is loaded in the scene and positioned in the center. An *orbit camera* is also added to the scene, which always points to the model, can be moved to any position around it and the distance to the model changed. Next, callback handlers are assigned for the mouse button press, release and move events, which are used for controlling the position of the orbit camera, causing the object to be visually rotated from the user's point of view. Finally, handlers for the mouse scroll events are added and used to control the distance between the camera and the model, and the scene rendering starts, drawing everything into the canvas element.

All of the technical details and complex logic involved in the 3D rendering process remain completely hidden from the web designers and developers who use the plugin we presented. With just a few additions, they can add support for this powerful feature to their new or existing web pages, having only to prepare the specific three-dimensional models in Collada format in advance.

Several other 3D model rendering and authoring jQuery plugins have been developed by third parties to support this functionality in the web browser. A 3D model viewer plugin [9], called 3dmodelviewer.jquery and utilizing the Three.js framework was released to be used for displaying a single three-dimensional model onto a webpage. The main difference between that component and the one discussed in this paper is that 3dmodelviewer renders the models directly into the page, while ours does that in a popup dialog, enabling the addition of more than one 3D object on the same page. Another difference is that 3dmodelviewer is more mature (ours is still in its alpha version) and offers initialization options like background color, initial camera position and autorotation. For authoring purposes, designers could use the 3DTin jQuery plugin [10] – a 3D model editor, also entirely executed in a web browser. It supports various types of geometries, lights and building camera animations. Having a different purpose than our plugin, the 3DTin editor can still be used for building models that our rendering component will display.

Conclusion and future work. This paper presented a solution for displaying existing 3D models in the web browser, allowing the user to rotate them in all directions and zoom in/out. Even though the HTML5 canvas element and WebGL are not fully supported in all browsers, most of them implement that powerful feature. The jQuery plugin we developed wraps the 3D rendering functionality and allows people with no technical knowledge to utilize it and visualize existing models.

There are many different scenarios where 3D rendering is required, varying from presentational web pages, online catalogs and other repositories to physics demos and computer games. One potential area to be researched and improved is combining more than one 3D model in the same scene, which can be used for building virtual expositions, comprised of several objects, allowing the viewer to move the camera freely around the scene. Another application of building complex scenes would be providing the ability to compare two objects by pointing out the differences and similarities in their characteristics such as dimension, shape, color, material and so on. Furthermore, a component for dynamically changing models' textures, material and lightning sources in the scene could be developed. Alternation of these resources in a scene would allow one to experiment with a digital object's look and observe how it changes in different custom scenarios.

REFERENCES

[1] HTML5 Working Draft – A vocabulary and associated APIs for HTML and XHTML. http://www.w3.org/TR/html5/
[2] O. Konstantinov, E. Kovatcheva, V. Fol, R. Nikolov. Discover the Thracians – An Approach for Use of 2D and 3D Technologies for Digitization of Cultural Heritage in the Field of E-learning. International Conference Digital Presentation and Preservation of Cultural and Scientific Heritage, September 18–21, 2012.
[3] P. Sabev. 3D Reconstruction of Cultural Values at the Regional History Museum – Veliko Tarnovo. International Conference Digital Presentation and Preservation of Cultural and Scientific Heritage, September 18–21, 2012.
[4] L. Pavlova-Draganova, D. Paneva-Marinova, R. Pavlov, M. Goynov. On the Wider Accessibility of the Valuable Phenomena of Orthodox Iconography through Digital Library. Proc. of the 3rd International Conference dedicated on Digital Heritage EuroMed 2010), November 8–13, 2010, Lymassol, Cyprus, 173–178 (Published by ARCHAEOLINGUA).
[5] M. Goynov, D. Paneva-Marinova, M. Dimitrova. Online Access to the Encyclopaedia Slavica Sanctorum. Proc. of the First International Conference "Digital Preservation and Presentation of Cultural and Scientific Heritage", September 11–14, 2011, Veliko Tarnovo, Bulgaria, 99–110.
[6] Canvas 3D Library Project Website. http://www.c3dl.org/
[7] JavaScript Libraries Usage Statistics. http://w3techs.com/technologies/overview/javascript_library/all
[8] Project Demo. http://3dweb.picforge.com/demo1.html
[9] 3dmodelviewer.jquery Plugin. https://github.com/rckt/3dmodelviewer.jquery
[10] 3DTin. http://www.3dtin.com

Vladimir Georgiev
Mathematical Linguistics Department
Institute of Mathematics and Informatics
Bulgarian Academy of Sciences
Acad. G. Bonchev Str., Bl. 8
1113 Sofia, Bulgaria
e-mail: vlado80@gmail.com

РЕНДЕРИРАНЕ НА 3D В УЕБ БРАУЗЪР

Владимир Георгиев

Напоследък новите уеб браузъри разшириха подръжката на HTML5, която включва новия canvas елемент, използван за рендериране на графика и изображения в движение. Поради това, че интерфейсът за програмиране на canvas елемента е много сложен, прости задачи като показване на 3D модели, тяхното завъртане, приближаване и отдалечаване са трудни за изпълнение. Използвайки архитектурата на jQuery, ние разработихме плъгин за визуализиране на триизмерни обекти в уеб страници, по интерактивен начин, изискващ само няколко дребни промени в HTML кода.