Demo: X3DOM – Declarative (X)3D in HTML5

Yvonne Jung
Fraunhofer IGD
Fraunhoferstrasse 5
64283 Darmstadt, Germany
yjung@igd.fhg.de

Jens Keil
Fraunhofer IGD
Fraunhoferstrasse 5
64283 Darmstadt, Germany
jkeil@igd.fhg.de

Johannes Behr
Fraunhofer IGD
Fraunhoferstrasse 5
64283 Darmstadt, Germany
jbehr@igd.fhg.de

ABSTRACT

In this demo description we present X3DOM, which is an open source framework and runtime system to support the ongoing discussion in the Web3D and W3C communities how an integration of HTML5 and declarative 3D graphics can look like by including X3D elements as part of the HTML5 DOM tree. The goal here is to have a live X3D scene-graph integrated into the HTML DOM, which allows manipulating the 3D content by only adding, removing, or changing the corresponding DOM elements. No specific plugin or plugin interface, such as the X3D-specific SAI, is required. X3DOM also supports CSS integration as well as standard HTML events like "onclick" on 3D objects.

Categories and Subject Descriptors

I.3.6 [**Methodology and Techniques**]: Standards—*Languages*; I.3.7 [**Computer Graphics**]: Three-Dimensional Graphics and Realism—*Virtual Reality*

Keywords

HTML5, CSS3, WebGL, DOM, X3D, Web Integration

1. MOTIVATION AND CONCEPT

Within this demo we like to showcase X3DOM, an open source framework and runtime system that supports the ongoing discussion in the Web3D and W3C communities how an integration of HTML5 and declarative 3D graphics can look like. The idea is to have a live X3D scene-graph integrated into the HTML DOM by including X3D elements as part of the DOM tree similar to the SVG HTML integration. This allows manipulating the 3D content by only adding, removing, or changing the corresponding DOM elements without the need for specific plugins or plugin interfaces like the X3D-specific SAI. X3DOM also supports CSS integration as well as standard HTML events on 3D objects.

The X3DOM website including online examples and links to external showcases can be found on `http://www.x3dom.org/` and the complete source code is officially available at github: `https://github.com/x3dom/x3dom`

One important design question here is, why we build upon X3D [4]. The open ISO standard X3D provides interactive 3D graphics for the web and is the only standardized 3D deployment format. It differs from other 3D formats like the interchange format Collada [1] in that it also includes the scene's runtime behavior. However, X3D is still bound to a plugin-based integration model and, like all plugin-based systems, has two major drawbacks [2, 3].

First, they plugins are not installed by default on most systems. Therefore, the user has to deal with installation, security, and browser or OS incompatibility issues, not to mention the lack of accessibility. Second, plugin-based systems define an application and event model inside the plugin, which is decoupled from the DOM content. Developers, who try to develop integrated web applications that use both, the DOM and the plugin-model, have to deal with small plugin-specific interfaces and synchronization problems.

WebGL [5], Flash 11 with Stage3D [7], and Silverlight 5 [6] now all provide access to the native GPU layer in the browser without a complicated plugin installation process, but the issue of the missing DOM integration still exists. Thus, what is still missing is a better integration with open architectures, which integrate well with existing web technologies like CSS3, HTML5, JavaScript, DOM scripting and Ajax. In this regard, the X3DOM project presents a DOM-based integration model for declarative (X)3D in HTML5 [2, 3], that allows a seamless integration of 3D contents into the HTML document model by utilizing standard web APIs for integrating content and interactions.

Besides DOM integration, we additionally propose a new HTML Profile based on the current X3D [4] standard, which extends the X3D Interchange profile. Therefore, this profile does not include X3D-specific concepts like internal Script nodes, pointing device sensors or support for PROTO types. Application developers are supposed to script and partition the content from the DOM/HTML side.

First of all we increased the support level for the networking component to 2 to include the Inline node, which is needed to portion the 3D data. Second, we increased the support level from the grouping component also to 2 to get the Switch and StaticGroup node. The latter will be used for optimiza-

Dec3D2012 workshop at WWW2012, Lyon, France

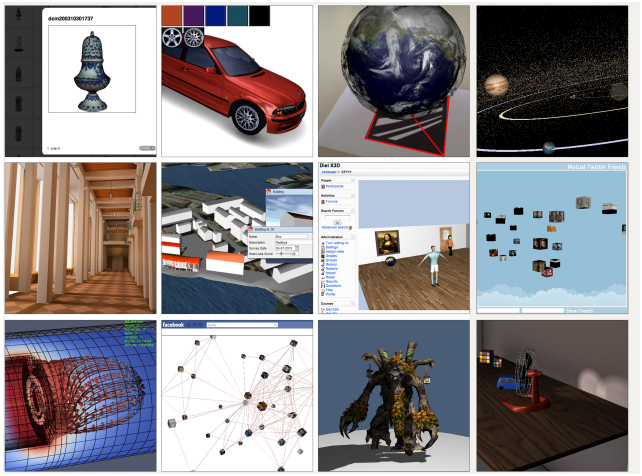

Figure 1: Some early examples of X3DOM-based Web applications (e.g., the top left image shows a line-up of laser scanned Cultural Heritage artifacts.

tion later on. We also increased the support level for the navigation component to 2 to include Billboard and LOD. In addition, we included some further standard components such as Text/ level 1 (since text is essential for web applications), Shaders/ level 1 (we already run on a GLSL-based shader pipeline), and the Followers/ level 1 (to get the flexible per-frame updates on the 3D side).

2. APPLICATION EXAMPLES

In this section a few applications are exemplarily showcased to demonstrate the potential of the proposed framework. In the area of cultural heritage for instance, working with 3D scanner data for preservation is getting more and more common. Figure 1 (top left) shows an example of an interactive web application for the visualization of scanned historical 3D objects, which was implemented with the help of X3DOM and standard web technologies.

The next image shows a simple online car configurator with a minimalistic UI. The user can choose certain colors from a given color palette or change the rims in real-time. In addition, the car can be viewed from any position, since nothing is prerendered. In contrast to today's online configurators, which are using semi-3D presentation methods, this car is a real 3D model. Hence, even small or complex animations of the doors or other parts are possible. The communication between UI and 3D model is completely implemented in JavaScript while styling is done with CSS.

The AR application shown next uses X3DOM for hardware-accelerated rendering of the earth globe, which hovers above the marker. Like in every AR application, a webcam and a video image is required to achieve the augmentation effect. Because native device access is not yet possible in HTML, a Flash-based marker tracker is used instead. The top right image shows the simulation of all planets and 100,000 of the known 480,000 asteroids of the Solar System.

The two screenshots in the middle are external showcases: the left image shows a GIS application with a floodwater

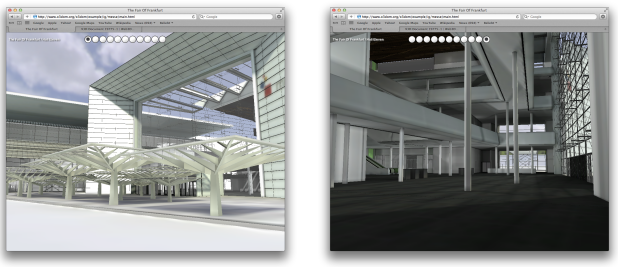

Figure 2: Architectural walk-through model of Hall 11 of the Fair of Frankfurt – represented and rendered using our new image geometry approach for fast content delivery and data compression.

scenario, where the user interactively can choose the water-level, whereas the right image shows an e-learning scenario. The rightmost image in the middle row and the second image in the bottom row are showing two similar use-cases (both also being external showcases): the visualization of the network of twitter friends and of facebook friends respectively.

The leftmost image in the bottom row of Figure 1 shows the visualization of a flow field simulation, where the respective color can be retrieved by picking a certain 3D flow region with the mouse. This color again maps to a specific temperature, which is obtained by inverting the original look-up value. This kind of application has a huge potential in the field of automotive for communicating and presenting e.g. simulation results towards decision makers or consumers, without distributing a whole application while providing the user with more information than a static image or fact sheet.

Figure 2 shows some screenshots of a walk-trough application realized with X3DOM. To ease the application development by excluding the vertex attribute data from the HTML file on the one hand and to compress this large geometric data set on the other hand we use so-called image geometries to represent and render vertex data by utilizing images as data containers. Within the course of this demo we will also show this new feature of X3DOM in action.

3. REFERENCES

[1] R. Arnaud and M. Barnes. *Collada*. AK Peters, 2006.

[2] J. Behr, Y. Jung, T. Drevensek, and A. Aderhold. Dynamic and interactive aspects of X3DOM. In *Proceedings Web3D 2011*, pages 81–87, New York, USA, 2011. ACM Press.

[3] J. Behr, Y. Jung, J. Keil, T. Drevensek, P. Eschler, M. Zöllner, and D. W. Fellner. A scalable architecture for the HTML5/ X3D integration model X3DOM. In S. Spencer, editor, *Proceedings Web3D 2010*, pages 185–193, New York, USA, 2010. ACM Press.

[4] W. Consortium. Extensible 3d (X3D), 2011. http://www.web3d.org/x3d/specifications/.

[5] C. Marrin. Webgl specification, 2011. https://www.khronos.org/registry/webgl/specs/1.0/.

[6] Microsoft. Silverlight, 2011. http://www.microsoft.com/SILVERLIGHT/.

[7] A. Systems. Flash, 2011. http://www.adobe.com/products/flashplayer.html.