Web-Based 3D Visualization for COMSOL Multiphysics® Software

S. Grabmaier[1], M. Jüttner[1], W. M. Rucker[1]

[1]University of Stuttgart – Institute for Theory of Electrical Engineering, Stuttgart, Germany

Abstract

Introduction
An important point for interpreting results of numerical simulations is the visualization. Especially in the field of product or development presentations and teaching issues an available and lightweight visualization solution is expected today. Here a web based visualization for common web browsers is described. Furthermore all recently introduced mobile devices are supported and easy and intuitive interaction is provided by supporting multi touch gestures.

Web based visualization
Hypertext Markup Language 5 (HTML5) and Web graphics library (WebGL) represent a new technology standard in the internet. They do allow complex 3D application running in the web browser with high performance. To display results of a COMSOL Multiphysics® simulation in a web browser the concept of the presented system is shown in figure 1. Starting with a solved COMSOL Multiphysics® model including available 3D plots at a COMSOL server, plot data is exported by an external Java application to the webserver. This Java application uses the COMSOL Multiphysics® Java Application Programming Interface (API) for accessing COMSOL server and a binary format, similar to the COMSOL Multiphysics® data structure, to store data at the webserver. The webserver is based on Node.js and hosts a web application (web app) as modern single page website. The data transfer of the binary plot data from the webserver to the browser is handled by Web Sockets [1] to achieve an efficient and smooth data exchange. Node.js is used to provide a server with little logic and high input/output performance and scalability [2]. The transmitted volume of three COMSOL Multiphysics® example models are shown in figure 2. The web app at the client contains the logic and interactivity. It uses standardized HTML5 and JavaScript functionality. Therefore no rendering is required on the webserver. The 3D visualization is realized in WebGL [3], a 3D API for JavaScript based on the OpenGL Interface (OpenGL ES 2.0). The advantage of using WebGL is the broad availability on different devices (i.e. smartphone, tablet, PC). It covers all necessary functionality to realize a render engine comparable to the renderer used in COMSOL Multiphysics®. Using the same plot data structure for COMSOL Multiphysics® plots, almost every 3D plot (isosurface, slices, arrow volume) gets available by using a minimalistic render engine. When it comes to pure 3D performance, the WebGL solution can handle complex and data intensive plots with constant frame rates. Figure 3 shows the achieved frame rate for the visualization of a power transistor model on a common desktop computer graphic processing unit (GPU) and on a mobile GPU.

Conclusion

A web based visualization for 3D COMSOL Multiphysics® plots was implemented by using Java API and web technologies. The performance of the system was shown for common models. The data transfer for the web based 3D visualization is minimal, so visualization is possible even with low bandwidth.

Reference

[1] Web Sockets and its benefits: http://www.websocket.org/quantum.html (May, 2014)
[2] Official webpage of the Node.js project: http://nodejs.org/ (May, 2014)
[4] WebGL, overview and official specification: http://www.khronos.org/webgl/ (May, 2014)
[5] COMSOL example model: power transistor, Plot: Arrow Volume with 5000 single arrows; maximal frame rate limited to 60 FPS.

Figures used in the abstract

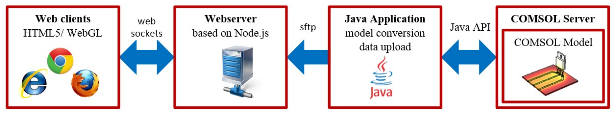
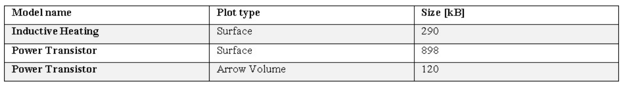
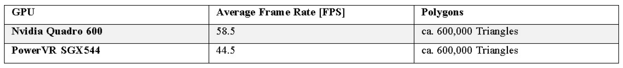


Figure 1: System Architecture.



Model name	Plot type	Size [kB]
Inductive Heating	Surface	290
Power Transistor	Surface	898
Power Transistor	Arrow Volume	120

Figure 2: Data Size of 3D Plots.



GPU	Average Frame Rate [FPS]	Polygons
Nvidia Quadro 600	58.5	ca. 600,000 Triangles
PowerVR SGX544	44.5	ca. 600,000 Triangles

Figure 3: Benchmark WebGL Renderer. [5]