Utrecht University for Applied Sciences

Digital Communication & Media

Faculty for Communication & Journalism

Viability of WebGL Product Configurators

Understanding Technical and Marketing Limitations

Author:
R.W.J. Peelen

Supervisor:
Dhr. K. Winkel

Jan. 2016

Abstract

Peppr is a company that is specialized in building photo-realistic visualizations. Late 2014, Peppr built a product configurator for SlimFitted, a company that builds tailored shirts. They wanted their customers to be able to design their own shirts.

Current product configurators for the web are built by splitting up the product into different layers. Every layer consists of a pack of images. Which, in Peppr's case, were 25 colors, 2 perspectives, 7 collars, 6 sleeves and 3 base shirts. This left Peppr with a sum of 6300 different layers (and thus, images) and when the client wants to add another color, they would have to build another set of 252 images. This is a timely and costly venture.

Peppr concluded the usual way of doing these types of projects is suboptimal and started looking for an alternative. That is where this thesis comes to play. March 2011 was the first release of WebGL (https://en.wikipedia.org/wiki/WebGL), an implementation of OpenGL technology for the web. Because WebGL renders directly from the video processor, it opens up the web to a whole new way of using 3d. The actual adoption rate has always been low because only the latest browsers would integrate the technology. Anno 2015 though, the playing field has changed. With more-and-more browsers supporting this new type of technology, the timing might be perfect to bring it to the masses.

In this thesis I will try to find if a WebGL based product configurator, is both viable in terms of technical and marketing aspects.

Contents

Chapter 1

Theoretical Framework

1-1 Introduction

Two things need to be touched before we can continue. Firstly, OpenGL (and it's derivate for the web, WebGL) to assess why it might be suitable for use in a project like this. Secondly, actual product configurators.

1-2 OpenGL & WebGL

" WebGL is an application programming interface (API) for advanced 3d graphics on the web. It is based on OpenGL ES 2.0, and provides similar rendering functionality, but in an HTML and JavaScript context. The rendering surface that is used for WebGL is the HTML5 canvas element, which was originally introduced by Apple in the WebKit open-source browser engine. The reason for introducing the HTML5 canvas was to be able to render 2D graphics in applications such as Dashboard widgets and in the Safari browser on the Apple Mac OS X operating system. Based on the canvas element, Vladimir Vukicevic at Mozilla started experimenting with 3d graphics for the canvas element. He called the initial prototype canvas 3d. In 2009 the Khronos Group started a new WebGL working group, which now consists of several major browser vendors, including Apple, Google, Mozilla, and Opera. The Khronos Group is a non-profit industry consortium that creates open standards and royalty-free APIs. It was founded in January 200 and is behind a number of other APIs and technologies such as OpenGL ES for 3d graphics for embedded devices, OpenCL for por parallel programming, OpenVG for low-level acceleration of vector graphics and OpenMAX for accelerated multimedia components. Since 2006 the Khronos Group has aso controlled and promoted OpenGL, which is a 3d graphics API for desktops. The final WebGL 1.0 specification was frozen in March 2011, and WebGL support is implemented in several browsers, including Google Chrome, Mozilla Firefox, and (at the time of this writing) in the development releases of Safari and Opera. " [1]

At the time of Andreas' writing, WebGL was very much in its early development stage. Anno 2015 though, WebGL has found its way into mainstream browsers with Internet Explorer and Android support lacking for versions earlier than its absolute latest [2]. It currently has a total adoption rate of nearly 81%. The most interesting part of using WebGL is the "low-level acceleration". Which means that on the stack of available hardware and software to render graphics, WebGL and OpenGL are closer to hardware level. This makes using WebGL and OpenGL insanely fast when compared to software / browser acceleration, which doesn't operate anywhere near hardware level. The main bottleneck for using said low-level acceleration, is that when there is no low-level graphics processing unit (GPU) available, the computing needs to be done with high-level software, which is ultimately directed (through several levels of architecture) to the CPU of the device and while some tasks are great for CPU calculations, some are not.

" Both CPU and GPU possess distinct architectural features. Modern multicore CPUs use up to a few tens of cores, which are typically out-of-order, multi-instruction issue cores. Also, CPU cores run at high-frequency and use large sized caches to minimize the latency of a single-thread. Clearly, CPUs are suited for latency-critical applications. In contrast, GPUs use much larger number of cores, which are in-order cores that share their con-trol unit. Also, GPU cores use lower frequency,and smaller sized caches[Mittal 2014a].Thus, GPUs are suited for throughput-critical applications. " [4]

As stated by Mittal and Vetter [4], the CPU is designed to handle a variety of calculations thrown at it, while the GPU is designed to be more special purpose. The importance of the CPU vs GPU is that WebGL makes use of the low-level acceleration of the GPU and more importantly, most modern smartphones these days have a GPU, making a potential adoption rate for the configuration platform much higher as end-users might well be able to use them on the go as well.

The focus of the thesis regarding the issues stated above will mainly concern the speed of the GPU on devices varying from desktop to mobile. Or in other words, if the computational power of mobile and desktop GPU's can handle high fidelity 3d models without compromising the user experience.

1-3 Current State of Product Configurators

Instead of diving into literature here, shown below is a dissection of a great product configurator built for Bugaboo.

1-3-1 Bugaboo

The configurator for Bugaboo [5] was designed by Momkai and the 3d images were built by 3Dimerce [6]. There are three things that stand out when analyzing this particular configurator: The user interface, the images and the speed. The user interface is very clear, it sets the stroller directly in view and the options are small until you need them. This creates a lot of negative space around the stroller, making it very easy for the user to discern the most important elements, and thus, to help them understand the interface more easily. The images used aren't exactly photo-real, but are convincing enough for the untrained eye to be believable. More importantly though,

dissecting the api revealed the configurator used 50 frames for the full 360. The change in angle is so small per frame that the experience feels incredibly smooth. Finally, the speed. By default, there are three different options directly in view (one can also add accessories to the buggy, which are also shown). From a technical point-of-view, those layers are pre-rendered individually and later put together. Momkai and 3dimerce [6] did something very clever here. Merging the layers client-side will bring along extra http request [7], which until http2 comes along will be penalised with a possible, albeit temporary, blockage of the page-load due to the latency an http-request brings with it. So, they decided to serve the images from an API. Instead of merging the images client-side, they merged them on the server and served a single image to the front-end. This not only makes it less prone to loading errors, but also makes the front-end application way easier to build. Next to that it will reek the speed benefits.

Now, while this is a prime example of a modern day configurator, both in looks and technology. There are 8 fabrics, 7 frame colours, and 2 seat colours just for their Chameleon model. Those options, at 50 frames per 360, means there are 5.600 renders. At another 22 accessories that have 2 point-of views, and the total rendered frame combinations are 5.644. 3dimerce stated the rendering of these took about 3 minutes. Which means that just in computational power, there where nearly 100 hours involved. Not to mention there were 4 strollers involved in the project. Next to the rendering, 3d models had to be made, lit, shaded and textured. The head 3d'er had to write a script to make sure the proces of rendering all separate layers went smoothly. Finally, there would have been quite a bit of work building up the API to handle the dynamic image generation, and the front-end where all those options came together and they had to link it up to an inventory system. This would have been a costly endeavour.

1-4 Conclusion

Looking at both OpenGL and its subsidiary WegGL, the technology has matured enough to be used by the public. While not massively adopted, the increase in speed in GPU's on mobile devices will allow for this technology to flourish. As far as the current state of product configurators. We've looked at and directed two examples of modern product configurators. Both of which use imagery for their media elements. As we dived into the details of the configurator and split up both the 3d and web side, the only logical conclusion to be made is that building these with imagery is a very costly endeavour and might be a lot cheaper when using a WebGL product configurator.

Chapter 2

Introduction

2-1 Problem Statement

Here I'll try to state problems per domain as widely as possible.

2-1-1 Service domain

2-1-2 Technical domain

2-1-3 Organizational domain

2-1-4 Financial domain

2-2 Objectives

Here I'll try to deform those problem statements to actual researchable objectives.

2-2-1 Main Objective

Broad Research question

2-2-2 Service objectives

2-2-3 Technical objectives

2-2-4 Organizational objectives

2-2-5 Financial objectives

Chapter 3

Plan of Action

3-1 Research Methodology

Short introduction as to why this section exists

3-2 Assignment of Methodology

Certain aspects of the objectives of the research need certain types of research. I'll assign them here while going into detail in the subsections below.

3-2-1 Service objectives

3-2-2 Technical objectives

3-2-3 Organizational objectives

3-2-4 Financial objectives

3-3 Experimental

Explanation of experimental research related to loading times / browser compatibility goes here

3-4 Literature

Explanation for empirical evidence based research goes here

3-5 Interviews

Here I'll try to find ways to interview larger companies on the matter and see if they would be interested in a pilot project.

3-6 Research Validation & Critical Notes

Chapter 4

Research Findings

Here I'll state the findings on the research per domain objectives, with a research conclusion in the end.

4-1 Service objectives

4-2 Technical objectives

4-3 Organizational objectives

4-4 Financial objectives

4-5 Research Conclusion

This will contain a go / no go moment, in which shall be decided if the building section shall be finished within this research.

Chapter 5

Building – Optional

5-1 Introduction to Single Page Web-Applications

5-1-1 Single Page Concept

5-1-2 Design Patterns

5-2 MEAN Stack

Introduction to the MEAN stack and per component an explanation to the use of it

5-2-1 MongoDB

5-2-2 ExpressIO

5-2-3 Angular

5-2-4 Node JS

5-3 WebGL Engine

5-3-1 ThreeJS

5-4 Requirements

Requirements for the actual app, which components are vital to the usefulness, in other words, what's the MVP

5-4-1 Functional

5-4-2 Technical

5-4-3 Data

5-4-4 MoSCoW

5-4-5 MVP

5-5 DataModel

5-5-1 Node based vs Table based

5-5-2 Data Requirements to Nodes

5-6 Wire framing

5-6-1 User Stories

5-6-2 Prototype

Chapter 6

Conclusion

Chapter 7

Further Recommendations

Chapter 8

Attachments

Chapter 9

Bibliography

[1] Andreas Anyuru, *Professional WebGL Programming: Developing 3D Graphics for the Web*, John Wiley and Sons Ltd, West Sussex, 2012.

[2] Alexis Deveria, *Can I use" provides up-to-date browser support tables for support of front-end web technologies on desktop and mobile web browsers.*, caniuse.com

[3] *This is the official OpenGL Website, where they've stated a lot of information on the history of OpenGL*, https://www.opengl.org/about/

[4] Sparsh Mittal and Jeffrey S. Vetter *A Survey of CPU-GPU Heterogeneous Computing Techniques*, Oak Ridge National Laboratory and Georgia Tech, 2015.

[5] Bugaboo *Bugaboo configurator*, Bugaboo website, Oct. 2015. http://blog.irayrender.com/post/112595883086/bugaboo-configurator-making-of

[6] 3Dimerce *Bugaboo configurator Making of*, iRender blog, Oct. 2015. https://www.bugaboo.com/NL/nl_NL/strollers/create?sId=CAM3STD

[7] Wikipedia *HTTP/2*, Oct. 2015. https://en.wikipedia.org/wiki/HTTP/2#Differences_from_HTTP_1.1