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 makes tailored shirts. Te customer wanted to built a platform on which their customers would be able to customize their own shirts

Most web-based product configurators are built by splitting up the full productimage into different layers, (mostly) one for each different configurable component. In Peppr's case, the customer wanted 25 colors, 2 perspectives, 7 types of collars, 6 types of sleeves and 3 different plaids. This left Peppr with a sum of 6300 different combinations. Even with the smartly dividing the images, Peppr need to make a staggering amount of images.

Next to the massive amount of images, there is also a problem that needs to be solved with regards to software. The end-user needs to be able to view and edit the shirt configuration in a user-friendly way. As of yet, most configurators have vastly different option sets (with different amounts of layers), components and style. This makes building a universal software application to handle all product configurators (in current manner) extremely difficult.

That is where this thesis comes to play. March 2011 was the first release of WebGL (https://en.wikipedia.org/wiki/W an implementation of OpenGL technology for the web. WebGL technology renders directly from the video processor and it opens up the web to a whole new way of using actual 3d models. Being able to use the 3d models instead of the vast amount of images would make the configurators easier to build, and thus, less expensive to make and maintain. Unfortunately, the actual adoption rate of WebGL has always been low because only the latest browsers would integrate the technology. Anno 2016 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, we will explore the possibilities of said configurator and try to find if this technology is ripe for production.

Contents

Chapter 1

Current State

1-1 Introduction

Peppr has done these projects in the past and states that the usual way of producing configurators consists of two sub-projects; the images, and the software. In the outline below follows a brief description of how the process is currently done at Peppr. Next, some new technologies that might make another way of doing these configurators possible. For this introduction, the following example project shall be used. Please note that the following process is the way Peppr would handle a project like this.
" A chair manufacturer wants a product configurator for one of their most popular chairs. It can be delivered in 25 different colours and has 4 different subframes. Two of the subframes are fully made from steel and can be delivered in either black or plain steel, the other two have wooden elements and have four different wood colour options. "

1-2 3d Department

1-2-1 Image Planning

Product configurators possibly have to deal with an exponentially growing set of options. To properly plan and see wether or not, Peppr believes it is good practice to see if these images can be split into separate components that make the set easier to maintain. Below is a summary of the full option set, split per item so we can start calculating how many images will be necessary.

- 25 colours - **(α)**
- 2 steel frames - **(β)**
- 2 frame colours - **(γ)**
- 2 wood frames - **(δ)**

- 4 wood colours - **(ϵ)**

In this case, any seat has a pick of several frame options, and several colours. But steel frames have two colours and wood frames have four colours. To calculate the full amount of options (x), we can use the following formula:

$$f(x) = (\alpha \cdot \beta \cdot \gamma) + (\alpha \cdot \delta \cdot \epsilon)$$
$$f(x) = (25 \cdot 2 \cdot 2) + (25 \cdot 2 \cdot 4)$$
$$f(x) = 300$$

Going with just 1 colour more adds 12 extra renders. While this may not seem like much, more often than not these renders need to be build from several different files. This means an artist will have some setup time to produce another option. So while the option set in itself might not seem like much, adding extra options is costly. Fortunately, in some cases, you might be able to get around it using layers. In this case, splitting up the chair into a seat and frame layer will get us the following formula:

$$f(x) = (\alpha) + (\cdot\beta \cdot \gamma) + (\cdot\delta \cdot \epsilon)$$
$$f(x) = (25) + (2 \cdot 2) + (2 \cdot 4)$$
$$f(x) = 37$$

This decreases the amount of renders by almost tenfold. Unfortunately, this is not possible in all cases due to the fact that a layer might be both in front and to the back of something simultaneously, making it extremely difficult to 'mask'. Next to that, creating new layers does add complexity and constraints. Adding new options / layers later on might prove extremely difficult if they were to overlap in with existing layers.
Apart to difficulties in the 3d process, these layers need to be build into the software as well. Either by creating an API that serves those layers as one image (Like the Bugaboo configurator [5]) or a front-end that can layer multiple images in a correct way. In the lather case, the user will directly notice this option as it adds more http latency, this should be resolved with the adoption of HTTP 2.0 and the pipelining of http-requests ([7]).

1-2-2 Modeling

Step two in the chain is the modeling proces whereby a virtual model is being created, either from scratch, or by 3d scanning real-life objects. If the choice to go for layers in the renders have been made, the modeler must make sure that there are clear creases where the cuts of those layers go so there is no overlapping geometry.

1-2-3 Lighting & Shading

In the lighting department, the model is being put into a suitable environment (mostly studio like setups), where it is lit and shaded (process of creating a life-like material) to perfection. The lighter must make sure that when there are layers involved, there is no unwanted shadow casting because when hiding certain layers for rendering.

1-2-4 Render

This is where things get together and the calculation from 3d model to actual image start. Depending on the configuration and setup, one renders the entire sequence, if < 100 renders and only one point of view for instance, one might swap out the model at certain frames. If one needs to render 360's, rendering one file at at time is better suitable.

1-2-5 Post Production

Every 3d model needs a bit of post production to make it look better. Also, if the model was split out into different layers but these were rendered as masks (an option to overcome overlapping images), this is where they would be split out into the different layers.

1-2-6 Compression

Using images of the web, especially on @2x or @3x resolution devices, mostly phones and tablets which get their date through mobile networks, compression is extremely important. With current mobile data speeds and capped contracts, laying a 50mb burden upon the user when opening a site is not a good idea.

1-3 Software

1-3-1 Requirements

Peppr starts out a software project by working out both functional and technical requirements. These will form a base to develop an API and front-end in a way that does not (should not) surprise the makers.

1-3-2 To CMS or not to CMS

This is a question that is tricky to answer. A content management system has basic functionality built in (user management, file handling, basic front-end), but it does require to work in the way that system is meant to be worked in. The other option, going with a from the ground up written system, will be much leaner and quicker when deployed, but will not be as mature as a popular CMS, which might result in a buggy experience for the end user.

1-3-3 API Planning

If

1-3-4 UX Development

1-3-5 Front-End Development

Chapter 2

Problem Statements & Hypothesis

2-1 Problem Statements

Here I'll try to deconstruct problems that arise when looking through the domains

2-1-1 Service objectives

2-1-2 Technical objectives

2-1-3 Organizational objectives

2-1-4 Financial objectives

2-2 Hypothesis

Here I'll try to specify testable hypothesis

2-2-1 Service objectives

2-2-2 Technical objectives

2-2-3 Organizational objectives

2-2-4 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

Conclusion

Chapter 6

Further Recommendations

Chapter 7

Attachments

Chapter 8

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

[8] Jesse James Garrett *Ajax: A New Approach to Web Applications*, Feb. 2005. https://courses.cs.washington.edu/courses/cse490h/07sp/readings/ajax_adaptive_path.pdf