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.

Next to the render aspect, there is also the question of software, as these images need to be viewed and configured by the customer. As such, 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.

This does bring new possibilities and challenges. For one, if Peppr succeeds in its mission, WebGL configurators might be more flexible, allowing for a Content Management System (CMS) like setup. This in turn would mean that Peppr could serve this as a Software As A Service (SAAS) business model. This would however mean, that Peppr might need to rebrand itself to be a software company, or set the service up as a separate entity. Many questions arise when thinking about the possibilities, risks and challenges.

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

Contents

Chapter 1

Current state

1-1 Introduction

A couple of subjects need to be touched to set a proper perspective. Starting with the underlying technology that might make these kind of projects succeed; WebGL and OpenGL. After that there are numerous market specific issues we need to address, starting with the current state of product configurators, Gartners Hypecycle (and wether or not he has spoken about such hypes) and finally.

1-2 Service Domain

1-3 Technical Domain

Part of why this thesis is written, is the way these projects are handled currently. Peppr has done these projects in the past and states that the usual way of producing these consists of two sub projects; 3d renders, and software. In the outline below follows a small description of how the process is currently done. After that an introduction on OpenGL and WebGL.
" 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. "

1-3-1 3d Department

Image Planning

Production configurators often have semi-exponential expanding set of properties. Below a summary of the full option set split 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)

To calculate the full amount of options (x), we can use the following formula:

$$f(x) = (\alpha \cdot \beta \cdot \gamma) + (\alpha \cdot \delta \cdot \beta \cdot \epsilon)$$
$$f(x) = (25 \cdot 2 \cdot 2) + (25 \cdot 2 \cdot 2 \cdot 4)$$
$$f(x) = 500$$

Going with just 1 colour more adds 20 extra renders. So while the option set might not seem like much, adding extra options is costly. In some cases, you might be able to get around it using layers. Splitting up this case into a seat and frame layer will get us the following formula:

$$f(x) = (\alpha) + (\cdot\beta \cdot \gamma) + (\cdot\delta \cdot \beta \cdot \epsilon)$$
$$f(x) = (25) + (2 \cdot 2) + (2 \cdot 2 \cdot 4)$$
$$f(x) = 116$$

This decreases the amount of renders by around 65% - 70%. 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'), and creating new layers does add complexity, not only in the following processes but also in the software department, having to create either an API that serves those layers as one image 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 at leaste one more http-request.

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.

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.

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.

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.

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. Laying a 50mb burden upon the user when opening a site is not a good idea.

1-3-2 Software

Requirements

Starting a software project should start with 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.

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.

API Planning

If

UX Development

Front-End Development

1-4 Operational Domain

1-5 Financial Domain

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.