Can WebVR further the adoption of Virtual Reality?

Christian Dibbern[1], Manuela Uhr, Dennis Krupke, Frank Steinicke

HCI, University of Hamburg[1]

3dibbern@informatik.uni-hamburg.de, uhr@informatik.uni-hamburg.de,
krupke@informatik.uni-hamburg.de, steinicke@informatik.uni-hamburg.de

Abstract

The market for Head Mounted Displays (HMDs) as a means to experience Virtual Reality has grown tremendously over the past years. HMDs operate on their own platform (e.g. Cardboard, Vive), even though interaction techniques are similar across platforms and each platform has to be programmed individually. Manufacturers also advertise their HMDs with exclusive apps and games for their own platform. Though game engines like Unity or Unreal provide unified APIs to ease cross-platform development, the VR software market is still pretty scattered through the different platforms. The biggest challenge for VR right now is not hardware, but the slow growth in content that appeals to a mass audience, combined with the confusion associated with a lack of cross-platform support (Jiteshi Ubrani). Because of its inherent openness, the web can be a means to tackle this problem. Developers and consumers could have easier access to VR. In this article I explore how the web and VR can benefit each other.

1 Introduction

Virtual reality technology has made great improvements in recent years and solutions continue to emerge into the hands of consumers. There are many ways to construct a virtual reality setup, but the dominant force in the consumer market are Head Mounted Displays (HMD). Just like the name suggests, the user simply gets a display strapped to his head, that shows two images, one for each eye, enabling stereoscopic 3D. There are HMDs for desktop usage, like the Oculus Rift and the HTC Vive and mobile viewers, like Google Cardboard, Samsung Gear VR or Google Daydream. They work in concert with a smartphone that gets inserted into the viewer and acts as the display. Despite them all showing virtual scenes in a very similar fashion, these devices have different capabilities and use different controls. They each have their own platform, that developers need to program separately. Manufacturers advertise their headsets with exclusive apps and games for their own platforms. Though multi-platform game

Veröffentlicht durch die Gesellschaft für Informatik e. V. und die German UPA e. V. 2018 in
S. Hess, H. Fischer (Hrsg.):
Mensch und Computer 2018 – Usability Professionals, 02.–05. September 2018, Dresden.
 https://doi.org/10.18420/muc2018-up-0249

engines like Unity or Unreal provide unified APIs so its easier to program cross-platform titles, the VR software market is still pretty scattered through the different platforms.

The VR market is still very young and consumers seem to be taking a cautious approach, said Jiteshi Ubrani, senior research analyst for IDC's Mobile Device Trackers. With plenty of headset options already in the market and even more coming soon, hardware isn't the issue. The bigger challenge is the slow growth in content that appeals to a mass audience, combined with the confusion associated with a lack of cross-platform support (Ubrani et al., 2017). Additionally, every VR experience has to be downloaded through one of the respective stores and installed. This may be advantageous for large games that can be played for hours, but it can be detracting for bite-sized VR experiences like an interactive short story that only gets used once. This is where the world wide web comes in. The web is a place that is made to be accessible to everyone, so it is inherently cross-platform. The web is instant, you dont have to download, install and eventually delete an application when youre done, you just click on a link and its there. This could pave the way for smaller VR experiences, because the user can effortlessly jump from one experience to the next, maybe even without leaving VR, instead of being sucked out of the immersion for the mundane task of deleting the old app, installing a new one and finally launching it.
WebVR is a specification being worked on right now, that brings VR to the web with unified browser APIs that allow for almost native performance. When VR applications on the web become more widespread, this gives VR more exposure, which should help drive adoption among consumers. People unfamiliar with VR may be tempted to buy an HMD, because they've come across VR content on the web and they want to try it out. This aspect would only come into play if there's a sufficient amount of VR content on the web that users could stumble upon it. First of all, there needs to be more content though and while WebVR makes VR on the web technically possible, it alone doesn't do much to drive content creation.

The best way to drive content creation is to get more creators invested. Integrating VR into the web gives us the opportunity to tap into the vast pool of web developers to design VR content. If you look at the use of JavaScript, which is at the core of web development, the language is atop the most in-demand languages. On a list ranking languages by the number of job openings, compiled by Ben Putano (Putano, 2017), JavaScript ranks second, directly behind Java, way ahead of the next contender C#. The list was compiled by looking at the number of hits for search queries of "<Language Name> Developer" on Indeed's website for the 50 most popular languages according to the TIOBE index. The TIOBE index is a more complicated metric that measures popularity by counting hits to the search query "<Language Name> Programming" in different search engines, the procedure is detailed in (TIOBE, 2018). JavaScript ranks sixth on the TIOBE index, directly behind C# at the end of 2017. The amount of JavaScript pull requests on GitHub for 2017, which is an indicator for the amount of code written, has JavaScript in first place, more than doubling Python and Java in second and third place (Putano, 2017). As these 3 metrics show, JavaScript is a popular language with lots of developers using it. This means it's a worthwhile cause to try to get JavaScript/web developers to create WebVR content.
More creators will not only create more content, which should drive adoption in and of itself, but probably more diverse content. The web can be a chance to expose developers from many different fields that have not had anything to do with virtual reality to VR. Developers will

probably create VR applications that no one has even thought of in their wildest dreams right now. Before we get there, though, the big question is: "How can we get web developers invested?". My answer to this question is the answer to another question: "How can we make WebVR development easy and fun for web developers?".

2 Making WebVR easy and fun for web developers

The best way to make WebVR development easy for web developers is to make it familiar. WebVR in and of itself is very low-level, and you need a lot of knowledge about 3D graphics to use it directly. Fortunately there are frameworks on top of WebVR that abstract away the complex rendering details. There are many different frameworks that can be used, but only a few support WebVR out of the box. In most cases, this functionality can be easily added by a developer, though. To make getting started as easy as possible, there is one framework that stands out in particular, A-Frame.
A-Frame is a declarative entity-component-system (ECS) framework on top of the imperative Three.js framework. It not only supports WebVR out of the box, it is centered around VR development for the web. Therefore a lot of functionality surround VR, like a 360ř videosphere, is already implemented. Furthermore A-Frame is not only a JavaScript framework, it integrates deeply into the HTML DOM, enabling a declarative approach to defining an interactive scene. The developer can define a scene in the HTML file by using the custom A-Frame tag <a-scene>. Because A-Frame uses an ECS, objects in the scene are defined as entities in the scene with the <a-entity> tag. The contents of an object are defined by adding components to the <a-entity> tag as attributes. To define a cube at the position 0,1,1 with an edge length of 1 the developer would add <a-entity geometry="primitive: box; depth: 1" position="0 1 1"></a-entity> inside the scene tag. Developers can define custom components using JavaScript, which means they can also use any of their favorite frameworks, like jQuery, in concert with A-Frame. A-Frame began as a web framework for building VR experiences at Mozilla. It's now an independent free Open Source project, with Mozilla's support. A-Frame prides itself on being one of the largest and most welcoming VR communities (*A-Frame Hompage* 2018). There are lots of tools made by the A-Frame community that make development easier in various ways, which is a big advantage. A big community also means that it is easier to get help and there are more tutorials for beginners out there.

A feature that sets A-Frame apart from many other frameworks is its Visual Inspector. It is designed to inspect and tweak A-Frame scenes at runtime. While running any A-Frame app, you can press a keyboard shortcut to inspect the running scene (unless the developer disabled this). The Visual Inspector is similar to Unity's and Unreal's interface, it shows a scene view, where you can select and manipulate objects transform properties by dragging them around the scene. There's a hierarchy panel showing all entities currently in the scene and a detail panel that shows the attached components and their attributes for the currently selected entity. Any changes made here are not saved to the HTML document, but you can download a new HTML file that contains the changes to your hard drive or copy the new HTML code to the clipboard. You can also copy the HTML code for selected entities or even just on a per component basis. This is great for debugging, because you can easily identify problems with objects that are not

currently in view of the camera. You can theoretically use the Visual Inspector as a graphical tool to define a scene, but it's not an elegant way to do so, because anytime you want to save your changes, you need to download a new HTML file. Problems can also arise from that fact that you're not looking at the static scene, you are looking at the app at run-time in a paused state. Any entities that dynamically change will not reflect their static properties in the inspector and dynamically spawned objects may clutter the Hierarchy panel, making it harder to find relevant objects.
Because of the lively community around A-Frame a few separate Editors are in active development. Ottifox is a WebVR creation tool for the Mac that works similarly to Unity's editor but exports A-Frame code. Ottifox licenses are available from $69. The Hologram project provides a similar editor, but also defines a Hologram framework on top of A-Frame. Instead of specifying the scene with A-Frame HTML tags directly, Hologram creates a JavaScript program that sets up the A-Frame Scene at run-time. The editor allows the developer to specify animations and interactions in CoffeScript, which gets compiled to JavaScript. This makes programming easier for beginners, but by obscuring A-Frame markup, Hologram also makes it difficult to interface with A-Frame directly in a Hologram project, which may be important if you want to use functionality that is not provided in Hologram. Accessing Three.js from A-Frame is easy in comparison. A big plus for Hologram is that it is free open source software, so anyone can take a look at the source code and potentially contribute.
Supercraft provides a much different approach to WebVR content creation. It is a WebVR application that you can use to build A-Frame scenes in VR. Building scenes for a VR application inside a VR application makes sense, because it is inherently difficult to create 3D content on a 2D monitor. By building a scene in VR the user gets a better sense of depth, making scene creation easier and more fun. In addition, there are also great VR modelling apps, like Google Blocks (only available as a standalone desktop app), so not only scene assembly, but modelling can be done in VR. Using A-Frame, developers have options to build a webVR app the way they like. In many simple cases defining a scene in HTML by hand is sufficient. If not, they are free to use other tools to help them design scenes, either in a classical way using a desktop editor or in immersive VR. And that's what I think makes development fun.

3 Possible applications of VR on the web

Many people tend to think of games first when they think of Virtual Reality. This is not surprising considering the fact that VR headsets have been heavily marketed towards the gaming sector. And while games in VR are great, there are other applications of VR that are not only worthwhile, but can be enhanced by being on the web. I want to describe some examples below

3.1 Data Visualization

Data visualizations can be employed in many different fields. They have been shown shown to greatly improve understanding of numerical data (Marr, 1982), (Biederman, 1987), (Spence, 1990). Visualization makes data more graspable and therefore easier to understand (Diehl,

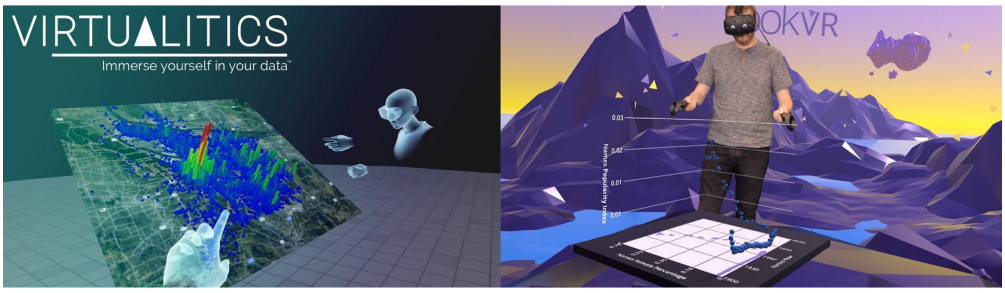

Figure 1: Virtualitics collaborative map diagram (left) and LookVR 3D plot (right)

2007). With the advent of big data especially, visualization of complex high-dimensional data has become important. Because there is only so much information you can convey in a two-dimensional plot, 3D visualizations have become more popular. VR can take visualization even further, because through the use of stereoscopic 3D the user gets a better sense of scale and presence. Ambivalences in perception, compared to 3D renderings viewed on a monitor, can be resolved (Maletic et al., 2001).
In recent years solutions for 3D data visualizations in VR have begun to emerge, like Virtualitcs or LookVR. Virtualitics is a U.S. company dedicated to providing a big data analytics software under the same name that features smart visualization tools in VR, aided by machine learning (Virtualitics, 2018). Virtualitics is also optimized for collaboration. Two or more users can look at a dataset together to analyze, build VR dashboards, present and discuss insights. Looker provides a similar software with LookVR. LookVR is similar to Virtualitics, but it works on top of the existing data analysis framework Looker. Compared to Virtualitcs, VR is more of an afterthought than an integral part, which doesn't necessarily make it worse, though. Impressions from both visualization tools can be seen in 1. Although both approaches are meant to be used by data scientists for analyzing data, their visualizations could also be used to present relationships in the data to an audience. And while I don't think these whole tools necessarily need to work on the web, an export option for finished plots would be great. If you could export these visualizations to webVR, this would open the door to make data more easily understandable for everybody. Having an interactive plot to inspect data, will also make insights the analyst gained from the data more transparent, because they're easier to verify. For example, this would be very helpful for research in all kinds of fields.

Both Virtualitics and LookVR are meant to provide general purpose visualizations to use on any kind of data. Another approach is to create special visualizations to present a specific data set. This way, the whole presentation is in your hands. A visualization doesn't just have to be informative it can also be fun. This can be especially helpful to engage non-professionals in your content. A great example of this is the *21 Years of NASDAQ VR visualization* by the Wall Street journal. It shows the rise of the U.S. stock market, its eventual fall leading to the financial crisis of 2008 and its recuperation up to 2015. To make this more interesting the user takes a ride on the data, driving up to the peaks and falling down to the valleys. This is supposed to convey the magnitude of the financial crisis by showing the steepness of the fall in VR, which is a great medium to provide a sense of scale. The *21 Years of NASDAQ VR visualization* is

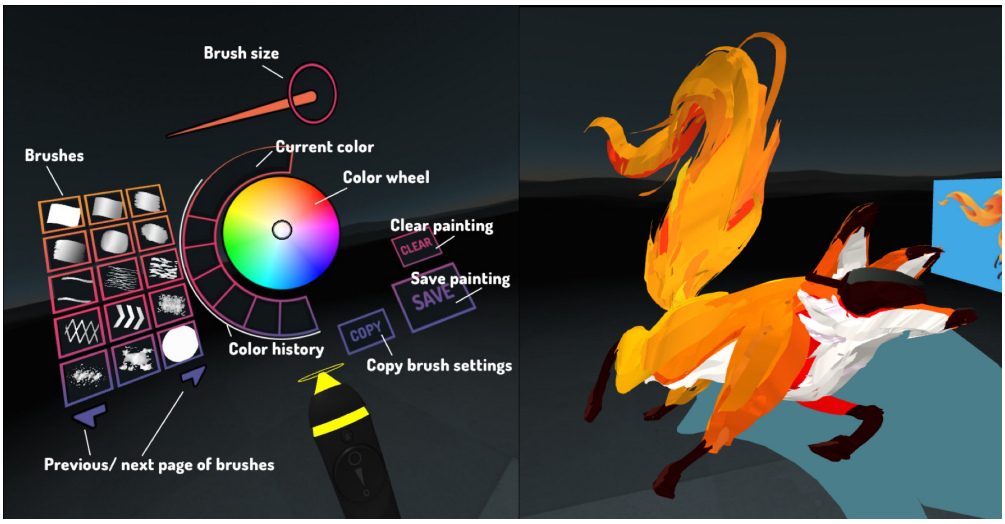

Figure 2: A-Painter

accessible through the Wall Street Journal website and works using WebVR (Journal, 2015).

3.2 Art

Art is a field that has continually embraced new media without abandoning the old. When photography was invented people asked themselves if paintings would become obsolete, which we now know was not the case. In my mind Virtual Reality is strikingly similar to photography, but in the 3D realm. VR technology could potentially replace entire museums, but just like photography, I think VR will become another interesting medium that artists can make use of without replacing old ones. One way artists can make use of VR technology today, is by creating 3D paintings in VR. Google's Tilt Brush has made this possible in a desktop app for the Oculus Rift and HTC Vive. Because the application simply outputs a 3D model, the finished artwork can be viewed outside the app as well. It can easily be shared to Google's Poly platform. Members from A-Frame's development team have created A-Painter, a de-facto Tilt Brush clone, that runs on WebVR. Because A-Painter already runs in the browser, an artist can share his work by just copying a link. Anyone who clicks the link will then be able to view the artwork right inside the A-Painter application.

3.3 Architecture, Interior Design and more

There are many more applications of VR that would benefit from being on the web. To not go beyond the scope of this article I will only briefly describe a few more interesting possibilities for VR on the web. There many opportunities to employ VR in the context of Architecture or Interior Design. Being able to not only view images from a house you might want to buy, but doing a virtual tour through the house gives a better impression of the space. This works with a

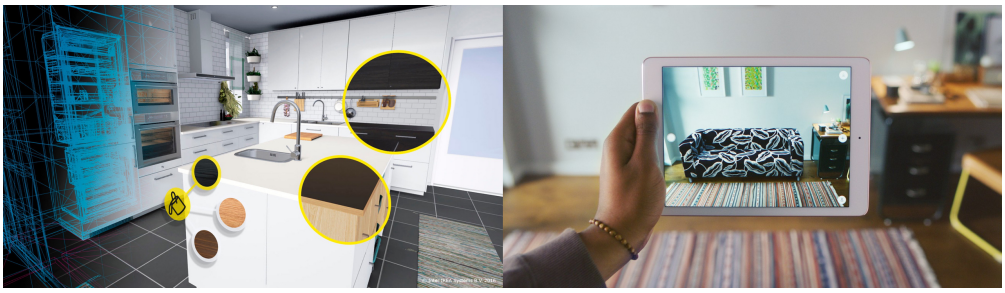

Figure 3: Ikea VR Kitchen (left) and Ikea AR Place (right)

3D model of the space as well as a few 360ř images that get connected together like in Google Streetview.
Showrooms for kitchens or bedrooms can be presented in a VR environment. This is something that Ikea has done, for example. If you take this thought further, Augmented Reality is an even better fit for this, though, because customers can place furniture or even a whole kitchen in their actual house to see how it would look. In fact, once again, Ikea is working on a solution for this, it's called *Ikea Place*, because the customer can place furniture anywhere. You can see this in 3. Travel websites are another example where virtual walkthroughs can be used to not only show off beautiful vistas, but to make would-be customers feel like they're there.

4 Conclusion

As I stated in the introduction, the first hurdle in making WebVR a success is to get web developers invested, by making WebVR content creation easy and fun. In section 2 I presented A-Frame, a tool that aims to do just that. A web standard for defining interactive VR scenes on the web is the ultimate goal. I strongly believe many ideas from A-Frame should be reflected in that standard, but there are other worthwhile approaches, too. Standardization is a process that takes time, though and it shouldn't be rushed, because that might result in a standard that's lacking in functionality and/or not future proof. Although A-Frame is not perfect, it's a good good way to create WebVR content now, that is friendly to newcomers. Which is why I think it's safe to say that a way to create WebVR content that is web-developer-friendly already exists.
Right now, not many people own VR headsets, which is why it's essential to provide a fallback to render on a standard monitor. Thankfully this works out of the box in A-Frame. Consumers should be enticed to get a headset because they believe it will enhance their experience, not just to be able to open a WebVR app. The best way to promote WebVR apps among consumers is to start integrating little VR experiences into frequently used websites, so they start to stumble upon them. I also presented a few applications aside from games that make sense on the web. Some of these are backed by big companies like Ikea, so interest is definitely there. Many of the applications that already exist are not on the web yet, but in many cases they should be easy to port. When big companies like Ikea start start putting their VR offerings on the web, I think

it will create a tipping point where WebVR enters the minds of the masses. It is at this point WebVR will either take off or fail. If we look at the best case scenario, there will be a VR hype because consumers understand the value of VR and that it is applicable to many scenarios. This will drive consumer adoption as well as push more developers to create WebVR experiences. In the worst case, WebVR just never takes off and fades away quietly. In conclusion I can't say for certain that WebVR will make VR succeed, but I do think that the success of VR is tied to the success of WebVR. In my mind WebVR is the way to bring VR to the masses and if WebVR doesn't succeed, the adoption of VR will stagnate and VR will continue to be a niche medium. What remains to be done to avert the failure of WebVR is to keep in mind that not only the quantity of apps is important. Because without high quality apps what is the reason for consumers to not only become invested in VR but to stay invested?

References

A-Frame Hompage. (2018). https://aframe.io. Accessed: 01.04.2018.

Biederman, I. (1987). Recognition-by-components: A theory of human image understanding. *Psychological review*, *94*(2), 115.

Diehl, S. (2007). *Software visualization: Visualizing the structure, behaviour, and evolution of software*. Springer Science & Business Media.

Journal, W. S. (2015). 21 years of nasdaq vr visualization. http://graphics.wsj.com/3d-nasdaq/.

Maletic, J. I., Leigh, J., & Marcus, A. (2001). Visualizing software in an immersive virtual reality environment. In *Proceedings of icse* (Vol. 1, pp. 12–13). Citeseer.

Marr, D. (1982). Vision: A computational investigation into the human representation and processing of visual information. new york, ny, usa: Henry holt and co. *Inc. June*.

Putano, B. (2017). Most popular and influential programming languages of 2018. https://stackify.com/popular-programming-languages-2018/.

Spence, I. (1990). Visual psychophysics of simple graphical elements. *Journal of Experimental Psychology: Human Perception and Performance*, *16*(4), 683.

TIOBE. (2018). Tiobe programming index community definition. https://www.tiobe.com/tiobe-index/programming-languages-definition/.

Ubrani, J., Shirer, M., & Mainelli, T. (2017). Worldwide shipments of ar/vr headsets gain momentum in the first quarter with strong growth forecast for the rest of 2017. https://www.idc.com/getdoc.jsp?containerId=prUS42707617. Accessed: 05.11.2017. Retrieved from https://www.idc.com/getdoc.jsp?containerId=prUS42707617

Virtualitics, I. (2018). Virtualitics hompage. https://www.virtualitics.com/. Last Accessed: 06.2018.