Towards a Framework for Immersive Analytics in Web-based Virtual Reality

Peter W. S. Butcher, Nigel W. John, Panagiotis D. Ritsos, *Member, IEEE*

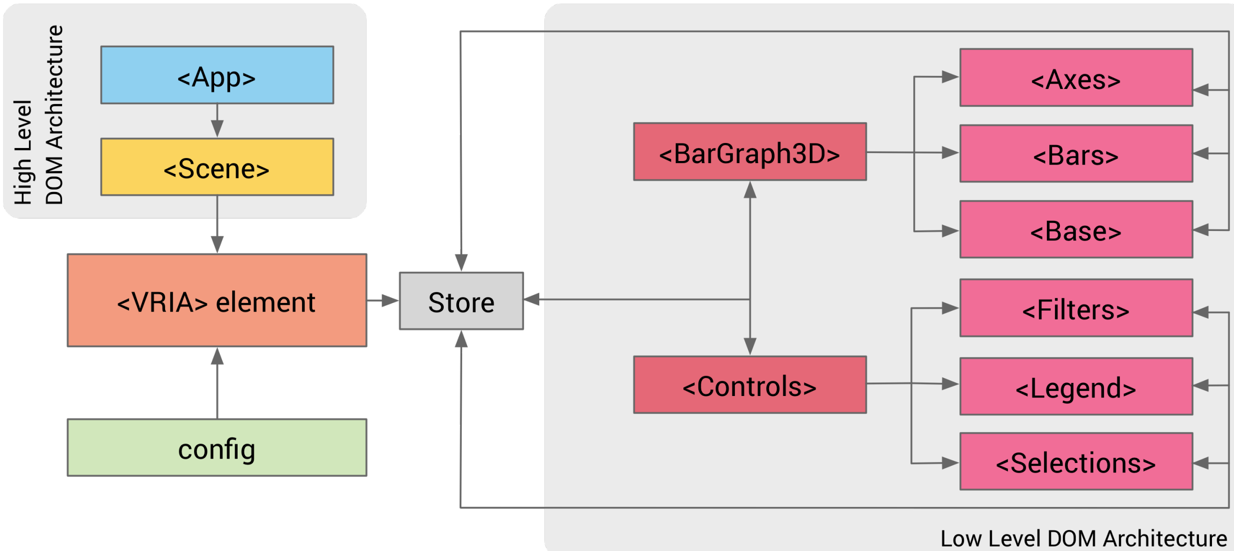


Fig. 1: Architectural overview of an example application built with <VRIA> and used in our evaluation (see Sect. 4). This diagram also gives a high-level depiction of how data flows through an application.

Abstract—
In this work, we report on the design, implementation and evaluation of a framework for building immersive analytics (IA) solutions in Virtual Reality (VR). The recent emergence of affordable VR interfaces have reignited the interest of researchers and developers in exploring new, immersive ways to visualize data. In particular, the use of open-standards web-based technologies for implementing VR in a browser aims to enable their ubiquitous and platform-independent adoption. In addition, such technologies work in synergy with established visualization libraries, through the HTML document object model (DOM). We present our prototype framework, <VRIA>, which facilitates the development of immersive analytics solutions for the web, and is built upon WebVR, A-Frame and React. We elaborate on our motivation for focusing on open-standards web technologies and discuss challenges we faced in the design and implementation of this framework. We also report on techniques and optimizations necessary for implementing immersive analytics systems on the web, and present a user experience evaluation of our framework through a use-case scenario focusing on non-expert users. Finally, we discuss the lessons learned from the development and evaluation of our framework and outline further investigations and extensions.

Index Terms—Immersive Analytics, Virtual Reality, Web technologies

1 Introduction

Immersive Analytics (IA) is an emerging research theme that builds on the recent evolution of computer interfaces, visualization and data science. IA seeks to investigate the use of novel display and interface technologies in analytical reasoning and decision making [12], with an ultimate goal to develop multi-sensory, collaborative, interactive systems that allow users to be immersed in their data.

- *Peter W. S. Butcher is with the University of Chester. E-mail: p.butcher@chester.ac.uk.*
- *Nigel W. John is with the University of Chester. E-mail: n.john@chester.ac.uk.*
- *Panagiotis D. Ritsos is with Bangor University. E-mail: p.ritsos@bangor.ac.uk.*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

In this paper we explore the paradigm of Virtual Reality (VR) as a medium for creating IA experiences for the visualization of abstract data. Despite the fact that VR has been around for decades, the recent emergence of affordable, commercially available head-mounted-displays (HMDs), such as the HTC Vive and Oculus Rift [42], has reinvigorated the interest in all things VR. We present our IA framework, <VRIA>, discussing the challenges we faced due to the use of web technologies, and presenting our approach in overcoming them. We also detail implemented performance optimizations that influence rendering, data mapping and interaction. Although the development of our framework is ongoing, at the current stage it enabled us to produce a simple, yet complete IA demonstration that we have subsequently evaluated to gain early performance insights in terms of user experience. The evaluation essentially establishes a performance and user-experience base-line for our current and future systems.

Our motivation for the investigation we present in this paper is twofold. Primarily, we are interested in building the necessary tools that will enable the visualization community to explore whether, and how, data visualization practices can be enhanced by the use of VR. Secondly, we investigate whether open-standard web technologies can enable the creation of such tools and the next generation of visualization systems.

The Web is the most ubiquitous, collaborative and platform-independent way to build and share information [48]. A visualization application with all its dependencies can be accessed via a URL from anywhere in the world with the only requirements for accessing content being the presence of a correctly configured browser and an Internet connection. It is no surprise that the most popular technologies for data visualization nowadays are from the web technologies ecosystem and are essentially based on open-standards. Libraries and tools such as D3.js [5], Processing.js, Protovis [4] and Vega [51, 52] have been shaping profoundly the landscape of contemporary data visualization tools.

Building VR demonstrations on the web requires a suitable immersive interface such as a HMD, and the mechanisms to allow a browser to render interactive 3D content. The recent interest in VR applications, along with standardization efforts have resulted in the emergence of powerful libraries and frameworks for implementing VR systems on the web, such as WebVR and A-Frame, the center-pieces of our solution. WebVR provides the necessary mechanisms for browsers to interface with HMDs such as the Oculus Rift, HTC's Vive, and Google's Daydream. A-Frame is a web framework for building VR content. As such tools (discussed further in Sect. 2) make use of the HTML DOM, they have the potential to work seamlessly with the established libraries used in Information Visualization and Visual Analytics.

We believe that the Web provides a formidable platform for exploring the synergy between data visualization and VR. Libraries such as D3.js provide powerful mechanisms for data binding, node manipulation etc., and are familiar to a large number of visualization developers. In addition, the current intensification of VR development on the web increases the opportunity for engagement with developers, researchers and data-consumers. This has the potential to democratize the development of IA systems that employ VR and thus enable more in-depth and varied investigations of immersive data visualization. Extending preliminary investigations presented in [10] and [9], our contributions are:

a) A review of prior work related to data visualization through VR.
b) A description of our framework's design and implementation, emphasizing architectural choices and optimizations that address challenges common in data visualization on the web.
c) A user-based evaluation of our framework through a simple analytical scenario, focusing on aspects of user experience and simulator sickness.
d) A discussion of the lessons learned from this investigation, which aims to set the groundwork for future work.

The paper is structured as follows: Sect. 2 covers previous efforts in immersive data visualization, and the current state-of-the-art in web technologies for data visualization using VR. Sect. 3 presents <VRIA>, discussing declarative visualization components, interaction and performance optimizations. Sect. 4 presents a user-based study of <VRIA>, looking into the aspects of usability, user experience and simulator sickness. Sect. 5 discusses the lessons learned from this research effort and outlines future work, reflecting upon the challenges to be addressed for VR-based data visualization using web technologies. Finally in Sect. 6 we summarize our findings and make our conclusions.

2 Background

Although the term 'Immersive Analytics' was only recently coined by Chandler et al. [12], fundamental aspects of beyond-the-desktop visualizations [48] have been proposed in a number of publications. For example, Lee et al. [34] discuss post-WIMP (Window-Icon-Mouse-Pointer) interactions; Elmqvist et al. [21] explore fluid, natural interactions for information visualization; Elmqvist and Pourang [20] describe the notion of ubiquitous analytics; Jansen and Dragicevic [30] propose an interaction model for beyond-the desktop; and Willet et al. [61] discuss situated and embedded data representations. Amongst this multi-flavored research thrust, VR has a prominent place as one of the oldest paradigms for beyond-the-desktop interaction. The interest to fuse VR with visualization dates back to the beginning of the century (e.g, [15, 25, 55]). Acknowledging this longevity and to highlight the evolution of VR technologies, we organize prior work on the synergy of VR with visualization as follows: a) early research that employs older technologies and standards; and b) research conducted using contemporary technologies. We also look into commercial systems that shape a popular yet not necessarily research-led approach to data visualization in VR. We further briefly acknowledge related work in Mixed Reality (MR) that uses similar technologies and is therefore valuable to our work. We finally look into the current state of the art of web technologies and open standards for VR, as these form the technical foundation of our work.

2.1 Immersive Data Visualization

Early explorations of the synergy of VR with visualization focused on scientific data [7, 8]. At the beginning of the millennium Van Dam et al. [55] presented a research agenda that included a call to action on how VR could be an effective medium for scientific visualization. Amongst the ever-present challenges of display technologies, rendering performance, collaboration facilitation and interaction, they highlighted a major obstacle being the lack of standardization that enables interoperability for the development of VR systems. Their observation, which applied and we believe still applies to all genres of visualization is the root of our aforementioned motivations.

Around the same time, Germans et al. [25] presented VIRPI, a platform independent and easy to use framework for building VR systems for interactive scientific visualization aimed at non-experts in VR. De Haan et al. [15] also explored different interaction scenarios with data visualizations, arguing that VR tools should relate to physical tools and interactions as these are used in physical-world scenarios. Zhang et al. [63] present a CAVE-based VR environment for visualizing tensor-valued volumetric datasets, indicating the suitability of VR in depicting complex geometrical structures.

Beyond merely displaying 3D information on a 2D display, the usefulness of stereoscopic representations for some graph visualization tasks has also been investigated. Ware and Mitchel [59] revisit prior work from Ware and Franck [58] using higher resolution displays, and demonstrate that node-link diagrams of hundreds of nodes can be successfully perceived by users in VR, assisted by stereo-viewing and motion cues. Alper et al. [1] explore stereoscopic highlighting for focus+context views of node-link diagrams by juxtaposing a detailed image of a region of interest. They also indicated the benefit in user performance when stereoscopic highlighting is combined with static visual highlighting.

3D graph layout approaches have been investigated in previous work, e.g., [29, 41, 49]. A recent approach, by Kwon et al. [33] investigates the use of immersive 3D visualizations for information visualization, looking into methods for spherical graph layout, edge bundling and techniques for interactive highlighting. Their visualization environment uses a HMD and demonstrates the effectiveness of immersive graph visualizations when using these methods.

Aided by the recent availability of new affordable VR interfaces, a large number of efforts have emerged over the last five years. Donalek et al. [18] write about immersive and collaborative data visualizations, pinpointing the potential that VR can have in data analysis tasks. They present iViz, a data visualizer based on the Unity3D game engine that attempts to address optimization limitations encountered in immersive worlds (such as OpenSimulator [43]) when dealing with complex data. iViz has full support for contemporary interfaces such as the Oculus Rift headset and allows multi-user collaborative data exploration [9]. Drouhard et al. [19] combine VR and data analytics in materials science. They discuss a series of challenges associated with immersive visualization such as safety and cybersickness concerns, interaction, data computation, feedback and user behavior acquisition. Importantly, they highlight the requirement for rigorous research evaluations so that this incarnation of VR is not 'wasted' as previous ones have been. Cordeil et al. [14] compare merits and shortfalls of CAVE and head-mounted displays (HMDs) for immersive network visualization. Their experiments show that HMDs enabled users to be faster and just as accurate as CAVE environments. Another effort from Cordeil et al. [13] introduces an immersive system built in Unity called ImAxes, for exploring multivariate data using virtual axes that can be arranged and combined in virtual space.

Of particular interest to our work is a web-based GPU-based visualization library from Stardust [45]. This library shares a number of similarities with our work (declarative grammar, web-technologies, WebVR support) but its main goal is to achieve significant performance improvements for rendering either 2D or 3D content. Our work, however, focuses on providing a flexible and extensible framework for building IA solutions on the Web that support a variety of interface devices. Nonetheless, the compatibility of the two approaches demonstrate the potential and opportunity for synergies that can be realized by the use of open-standards and Web technologies.

Mixed Reality approaches (including Augmented Reality (AR)) are also of interest. Lu et al. [35] review the current state of mobile immersive environments relevant to data visualization using both VR and AR. ElSayed et al. [22] investigate how to link visual information and objects in physical space using MR, based on a model of *"details first, analysis, then context-on-demand"*. Similarly, Sadana et al. [50] discuss lessons learned from designing for multi-touch interfaces, extrapolating them to VR and AR. Luboschik et al. [36] investigate the effect of spatial cues for data visualization. They argue for the need of adaptation of existing, non-data related VR and AR approaches. Bach et al. [2] evaluate the use of AR for visualization, comparing hand-held and desktop-based set-ups. Finally, Butscher et al. [11] explore the use of AR for the collaborative analysis of multidimensional data. They present ART, a tool that uses interactive, 3D parallel coordinate visualizations displayed on a tabletop.

In addition to academic-led research, immersive data visualization has been explored in various projects by developers and enthusiasts. Masters of Pie [37] explore data visualization in VR with V-arc, rendering data points onto a spiral arch over the user. The Virtualitics Immersive Platform [56] claims to use VR and MR, along with machine learning and natural language processing, to devise immersive data exploration tools. VR Nasdaq [38] presents a VR-based guided tour of 21 years of the Nasdaq built using Three.js and D3.js. Projects such as MathworldVR [39] also use WebVR to create simple data visualizations.

2.2 The Web as a VR Platform

In the last decade there has been significant interest in implementing VR solutions in a web browser. Various open-standards have emerged, enabling developers to create VR applications that utilize the graphics acceleration capabilities of WebGL. WebGL is the enabling technology of most (if not all modern) 3D tools for the web. A comprehensive review of available 3D technologies for the Web is beyond the scope of this paper; for that we refer the reader to the survey of Evans et al. [23]. However, their survey does not include some of the latest frameworks that have emerged since 2014. In particular WebVR and A-Frame (see Fig. 2), which have been exploited in this work.

Three.js [17, 44] is an open-source Javascript library that enables programming of 3D scenes in a Web browser. It uses the HTML5 canvas element, SVG or WebGL as rendering engines and features a scene-graph, several cameras, navigation modes, shaders (including custom) and material support. A-Frame [40] is built on Three.js and provides a higher-level API, which exposes all functionality of Three.js. A-Frame is an entity component framework that provides declarative, extensible, and composable structure to Three.js. A-Frame provides components for all major HMDs and their controllers, including the 6 degrees of freedom (DOF) HTC Vive, Oculus Touch and Windows Mixed Reality controllers, as well as 3DOF Daydream and Gear VR controllers. A-Frame also provides gaze-based controls for 3DOF device interactions such as those used with smartphones. These gaze-based controls also work with interfaces such as touch screens. A-Frame is currently the most versatile way to create WebVR content.

The WebVR API [57] is an open specification API that allows developers to create VR solutions in a web browser. It can detect changes to available display and interface hardware meaning we can tailor our VR environments and interactions to the user's current hardware configuration. For example, if a PC or mobile user does not have access to a HMD, then we can display a mono-3D scene instead and users will still be able to look around in 360 degrees and interact with a gaze cursor. The specification is currently under active development and will be replaced by the WebXR Device API. Until 2016, building VR content required a WebVR polyfill [26], providing basic stereo rendering and barrel distortion with a choice of lens configurations for use with smartphones and Google Cardboard [27] viewers [9]. The need for the polyfill reduces as support for WebVR/XR becomes standard in Web browsers. WebVR has been gaining support with browser vendors and an number of tools and demonstrations have emerged [9]. Its successor, WebXR, will also incorporate AR/MR capabilities [54].

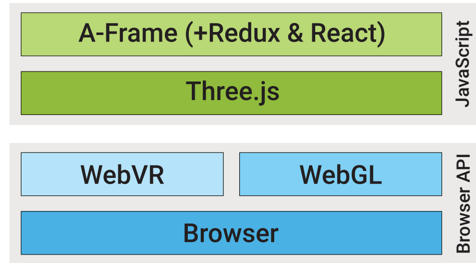


Fig. 2: Our framework is built upon the WebVR stack. At the lower level, the browser acts as a rendering engine. WebVR manages the interface with the HMD and its controllers, or the necessary fall-back to a non-immersive display (e.g., a desktop display in 2D). WebGL handles hardware acceleration for the 3D graphics. All 3D graphics are managed by Three.js and, at a higher level, A-Frame. Redux and React are are covered in the framework section (Sect. 3)

3 The <VRIA> Framework

Many visualization grammars and frameworks have been developed to facilitate the creation of information visualizations. Examples include ggplot [60], Vega [53], and Vega-Lite [51], which focus on declarative representations that separate a visualization's specification from its implementation. Other tools including Prefuse [28] provide building blocks for creating custom visualization designs. Finally, D3.js [5] provides DOM manipulation mechanisms based on data attributes. Our framework provides support for all of these approaches.

Following our prior investigations [9, 10], we were motivated to create a framework allowing anyone to create interactive, immersive WebVR data visualizations. We include methods for rapid prototyping with simple configuration files, as well as an API facilitating the creation of bespoke visualizations and interactions. By accommodating both of these approaches, <VRIA> is suitable for both seasoned programmers and users with no prior programming experience.

3.1 Architectural Overview

<VRIA> is written in React, a JavaScript library for building user interfaces; Redux, a state container for JavaScript applications; and A-Frame, a framework for building WebVR experiences. A-Frame exposes an HTML DOM scene graph that React can use to efficiently update only the parts of our scene that require re-rendering. While operations such as searching, modifying, adding and removing nodes from the HTML DOM are very fast, re-painting large parts of the DOM tree can be computationally expensive. React uses a virtual DOM to calculate the differences in the HTML DOM before and after a change, so that only the affected DOM nodes get re-painted. When we are re-painting a DOM tree for a data visualization with potentially thousands of data points, reducing unnecessary re-paints is essential for performance. Fig. 1 gives an architectural overview of an example application built with <VRIA>.

```
import React from 'react';
import ReactDOM from 'react-dom';
import { Scene } from 'aframe-react';
import VRIA from './vria';
import config from './myExampleVisConfig.json';

class App extends React.Component {
 render() {
  return(
   <Scene>
    <VRIA config={config} />
   </Scene>
  )
 }
}

ReactDOM.render(<App />, document.getElementById('root'));
```

Listing 1: Example boilerplate code for a <VRIA> application using React and aframe-react.

The exact structure and implementation of the high-level DOM architecture of an application that makes use of <VRIA> is up to the user, and there is no requirement for the whole application to be written in React. <VRIA> can be integrated into existing applications, with the only requirement being that the overarching application makes use of A-Frame scenes.

The top level node in the example application depicted in Fig. 1 is a React component (■) that contains an A-Frame scene component (■). The scene component can contain any other A-Frame components that a user desires. <VRIA> provides a React component called *VRIA* (■) which must be placed within the scene element and should be passed a visualization configuration file (■). Everything that this component generates is mandated by the configuration file that is passed to it. The store (■), is implemented with Redux and contains the application state. It is connected to visualization and control components (■). These components map data to React components (■), that depict the visualization and all user interfaces with A-Frame and ThreeJS. They also respond to user interaction and report these actions back to the store, which then updates the visualization. An example implementation of the high level DOM architecture can be seen in Listing 1.

The use of a state container like Redux was necessary due to the complications associated with deeply-nested component hierarchies. A state container removes the need to pass data down through many intermediaries to reach deeply nested components, and provides ways for those components and their state to be easily accessible from anywhere. For example, for visualization and control components to communicate in <VRIA>, they communicate via the store rather than by executing callbacks passed down from a common ancestor. This model keeps the source of the application state in one place, makes the flow of data more predictable and lets us write cleaner, more readable code as a result.

3.2 Visualization Configuration and API

<VRIA> provides a simple method of converting a dataset into WebVR-ready 3D visualizations, coupled with a set of appropriate interfaces (controls), through a declarative format described with JSON. This forms a visualization configuration that our framework can interpret. Our declarative format may not yet be a fully-fledged 3D visualization grammar like Vega [52] or Vega-Lite [51]; but it allows users to create different visualizations through setting parameters, properties and constraints tailored to each visualization type.

In its most basic form, a visualization configuration consists of a dataset and the type of visualization we wish to use. This can either be packaged up as a JSON file (see Listing 2) or as a JavaScript module (see Listing 3). In this case no extra properties are specified in the configuration file, and as a result the user is presented with a visualization that uses the 3D scatter plot component with all of its default settings. With no configuration overrides, the visualization component will make assumptions on how to encode the data and so it is recommended to set up data encoding in the configuration file. Should a user wish to specify overrides for certain default properties of a visualization component, such as data encoding then they may do so in the configuration, as demonstrated in Listing 3.

<VRIA> currently supports tabular data formatted as JSON or CSV.

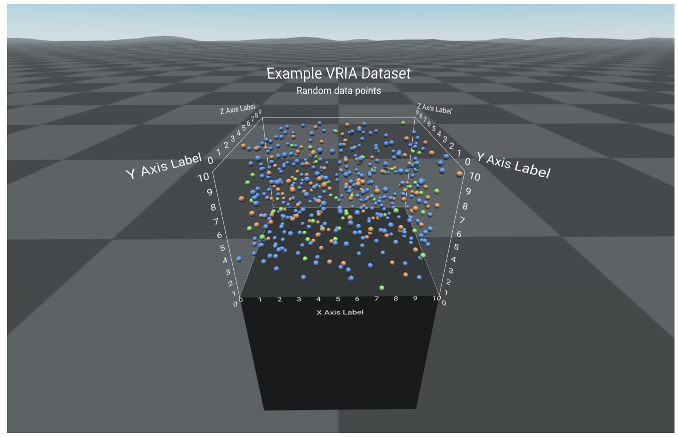


Fig. 3: Example of a scatter plot created with <VRIA> depicting a randomly generated multivariate dataset with 500 data points.

```
{
 "data": "./data/exampleDataset.csv",
 "visComponents": ["ScatterPlot3D"]
}
```

Listing 2: Basic visualization configuration as JSON for a 3D scatter plot visualization

Data sets may contain any number of fields and records. Supported visualization types currently include 3D bar charts (as demonstrated in our user-study, Sect. 4) and multivariate scatter plots (Fig. 3) with more abstract data visualization types planned. Each visualization has its own set of possible interactions that can be selectively enabled or disabled in the visualization configuration.

Visualization and control components are written using A-Frame. The framework's API is used to map controls to data transformations, which are then reflected in the visualization. The exact nature of the data transformations and desired interactions are left to the user; the framework then connects everything together. Data transformations for new visualizations can be achieved with existing libraries such as D3.js, or can be written from scratch.

<VRIA> only requires a visualization configuration for basic functionality. However, if extra functionality is required such as a custom set of interactions or a bespoke visualization type, then these can be created with A-Frame and <VRIA>'s API.

3.3 Interaction Mechanisms

The ability to provide appropriate interaction mechanisms based on the platform and hardware available to users is critical for Web-based IA applications. The Web's ubiquitousness is very much based on the notion of adaptability and responsiveness.

As with any web application, we should aim to create experiences that work for as many people as possible, even those without access to VR hardware. It is therefore recommended to treat the availability of WebVR as progressive enhancement. Some interfaces are inherently incapable of making complex interactions, for instance a gaze cursor on a smartphone compared with a hand-held controller with a HMD. WebVR employs the Gamepad API, which is experimentally supported in all major browsers bar Internet Explorer, to access tracking and interaction data for various HMDs and their navigation controllers. As A-Frame is able to provide suitable interaction mechanisms regardless of available hardware, it is possible for us to provide familiar interactions across every platform within the same application. <VRIA> utilizes this functionality by providing appropriate visualization interactions based on the hardware and platform which is currently being used. An example of this might be offering a drag interaction in the form of two separate gaze clicks at the extremities of the desired selection. The application we built with <VRIA> for our evaluation was not intended for use on smartphones, but when viewed on a smartphone the user is still able to fully interact

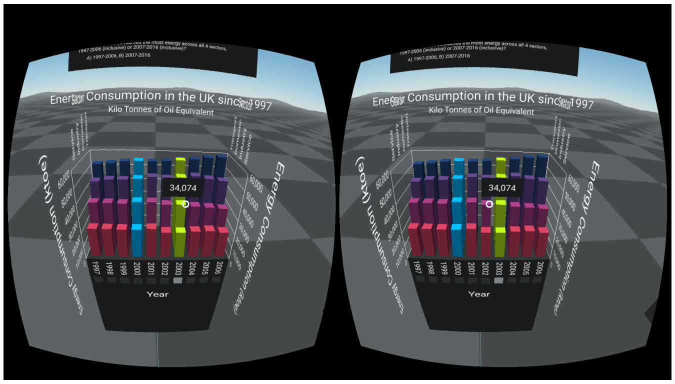


Fig. 4: The <VRIA> application used in our evaluation (Sect. 4) in use on a mid-range smartphone with Chrome for Android and Google Cardboard. The user is able to fully interact with all controls via a gaze cursor.

with every control available through touch-based interaction when not in VR mode and gaze-based interactions when using Google Cardboard (see Fig. 4 and Sect. 4).

Each visualization component built with <VRIA> has a set of possible interaction types, known as controls. One <VRIA> control component is described in Listing 3. These controls are bound to the depicted data and dictate the users' interactions with the visualizations. In this instance, data is encoded to all three axes as well as onto the mark itself. Data encoded on the mark is customized in the mark configuration object and in this example, the color property has been set. The mark tooltip property has been left in its default state as no overriding tooltip properties have been specified. A control component is also configured. The legend component displays a key for each variable in the dataset and optionally lets a user toggle each variable. Other control components are available and are demonstrated in our evaluation application (see Sect. 4). A-Frame allows the creation of physics-based controls, such as pushable buttons and movable sliders. Platform-specific controls do not need to be defined in the framework's visualization specification, as the interactions available to the user depend on the platform and hardware they currently use.

3.4 Optimizations

Maintaining adequate frame rates is one of the most challenging aspects of building a VR experience. Typical desktop and mobile displays have a 60Hz refresh rate which allows 16.6ms for a new frame to be produced. This is known as a frame or time budget [62]. A web browser's resultant frame budget is determined by the refresh rate of the display to which it is rendering. For a VR scene using a HMD such as the Oculus Rift or HTC Vive, the frame budget will be 11ms as these devices have a 90Hz refresh rate. Desktop displays usually only require a scene to be rendered in mono whereas a VR headset requires a stereo image, leaving just 5.5ms to create an image for each eye.

With such a small frame budget in VR scenes, it is more difficult to make all of the necessary computations required to render a new frame in time. When the frame budget is exceeded, the browser is unable to render the new frame, which results in (at least) a lost frame. On a desktop display, frame loss can result in a jerky experience if a user is viewing an animation or scrolling. In VR, the effects of frame loss can be more severe as any interruptions to immersion can lead to uncomfortable VR experiences and increased risk of cyber-sickness. As a result, certain performance optimizations are necessary.

Preventing dropped frames due to IO locking, by making excessive CPU computations every few frames or re-rendering complex geometry, also often requires compromise. There are several performance optimizations that can be made at the React level to get fine-grained control over when a component should update, further preventing unnecessary re-renders. There are also performance optimizations that can be made at the ThreeJS level including the re-use and merging

```
import dataset from './exampleDataset.csv';

export default {
 data: dataset,
 meta: {
  title: "Example VRIA Dataset",
  description: "Random data points"
 },
 visComponents: [{
  type: "ScatterPlot3D",
  config: {
   encoding: {
    x: {field: "a", type: "quantitative"},
    y: {field: "b", type: "quantitative"},
    z: {field: "c", type: "quantitative"},
    mark: [
     {field: "d", type: "nominal", property: "color"},
    ]
   },
   mark: {
    shape: "sphere",
    properties: {
     color: {
      range: {
       type: "random",
       scheme: "midPastel"
      }
... //lines omitted for brevity
 }],
 controlComponents: [{
  type: "Legend",
  for: ["ScatterPlot3D"],
  config: {
   toggle: true
  }
... //lines omitted for brevity
```

Listing 3: Example of a visualization configuration, written as a JavaScript module for a 3D scatter plot (see Fig. 3) depicting a multivariate dataset including a single control component (some lines ommited for brevity).

of object geometries to save on memory and the number of draw calls per frame. Existing <VRIA> visualizations and control components make use of these optimizations but because they occur at such low levels, our framework is currently unable to apply them to user created components that do not make use of <VRIA> primitives. Eventually, if the scene is very complex and is undergoing extensive CPU heavy tasks, users will notice stuttering or choppiness when there is motion on the display. This is known as JavaScript "Jank". This is a disadvantage of the thin client model and the single-threaded nature of JavaScript. One solution is to "debounce" interaction operations. Debouncing is a technique for limiting the rate at which a function can be triggered, thus in our case reducing the workload from complex interactions to ensure we do not overspend our frame budget. Similar techniques are also encountered in future intrinsic optimizations to be integrated in React (see Sect. 5).

By waiting for a user's interaction to be completed before attempting a re-render, we can reduce the jank associated with stacking multiple expensive computations together in quick succession. Furthermore, we can batch computations over multiple frames and only render the result once all computations have been made. We can achieve this by utilizing a callback function to performs computations on the main event loop, during periods in which the CPU would normally be idle. <VRIA> utilizes both of these techniques to keep interactions feeling as real-time as possible without impacting user experience.

4 Evaluation

We conducted a user study to evaluate <VRIA> in terms of usability, user experience and the presence of any simulator sickness symptoms. By evaluating our framework at this intermediate stage we aim to inform design decisions going forward. This will allow us to enhance and improve overall performance, data depiction and interaction mechanisms, as we target different types of immersive data visualization scenarios on the Web.

Thirty non-expert study participants were recruited to objectively evaluate the performance of a simple data visualization scenario built using <VRIA>. We collected data using the System Usability Scale (SUS) survey [6], the Simulator Sickness Questionnaire (SSQ) methodology [32] and open-ended questions. Participants were present in our laboratory during the testing.

On a secondary level, we wanted to determine to what extent our system enables efficient analysis of 3D chart data compared to traditional methods such as 2D screen-based visualizations. We believe that in order to test future IA systems, and to genuinely assess the impact they can have in analytical tasks we have to make them accessible to these non-expert users. This of course does not exclude expert users from future investigations. Instead, we wish to exploit the ubiquitousness of the Web and democratize access to Web-based IA systems.

To reduce the effect that the type of visualization has in the overall user experience we selected an analytical task that uses 3D bar charts, inspired by [31], and building upon our previous work [9, 10, 46, 47]. Although the bar charts may not be the best medium for analytical decision-making in 3D, for non-experts they are simple to understand, known to them and therefore simpler to use compared to, say, a node-link diagram. Our intent was to reduce the effect of the complexity of the visualization in the overall user experience. We therefore formulated three tentative expectations at the start of our experiment in terms of the analysis of simple chart data, that: a) immersive 3D would not be more effective than 2D, b) immersive 3D would be more compelling for the users, and c) Desktop-based 3D (non-HMD) would be the worst of all three, in terms of user performance and user experience.

4.1 Methodology

The test applications involved participants answering questions based on UK energy consumption data between 1997 and 2016 [16]. We used three test conditions:

- *Task 1:* WebVR with a <VRIA> application, interacting with the data via an Oculus Rift HMD and Oculus Touch controllers.
- *Task 2:* WebVR with a <VRIA> application but instead viewed on a 2D PC monitor with mouse interactions.
- *Task 3:* A non-immersive 2D visualization rendered on a 2D PC monitor with mouse interactions.

Each participant completed the experiment using all three test conditions with the order being randomized to distribute the effects of any learning that might skew task completion times even after a warm up period and a training question.

The SSQ was administered before and after the VR task as required, whereas the SUS and open-ended questions were completed after the experiment. Participants were given a brief introduction to the interface, the application, and using the HMD controllers. They were also given a practice question to answer before beginning the experiment. Moreover, they were required not to ask any questions during the experiment. The answers given to the task questions and completion times were also recorded. The evaluation was conducted using a Windows 10 PC (Intel i7, 32GB RAM, Nvidia GTX980 Ti). The web applications were displayed using the Mozilla Firefox Quantum browser.

4.1.1 Participants

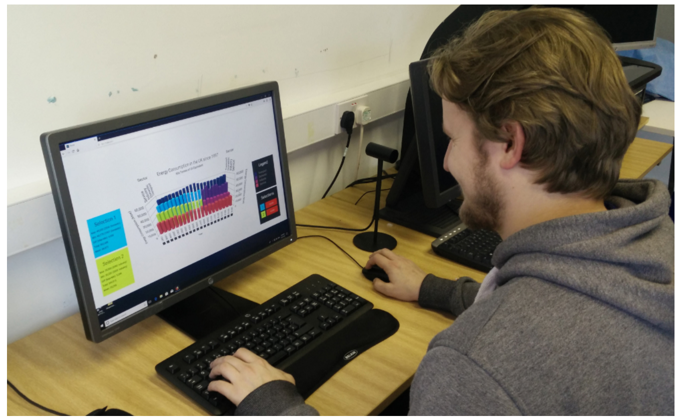
We recruited 30 participants, between 18 and 65 years of age, 24 male and 6 female. 24 had prior experience with VR, whereas 6 (3 male and 3 female) had not used any VR technology before. All participants reported they had good or corrected eyesight and were given the opportunity to determine which vision setup (e.g. glasses or contact lenses, or no optics at all) gave them the most comfortable viewing experience inside the headset.

4.1.2 Test Conditions

The <VRIA> application tasks (test conditions 1 and 2), used a 3D bar chart and the non-immersive task (test condition 3), used a layered bar chart to display the data from four different consumer sectors on a year by year basis. Users were able to filter the data and make selections, presenting users with basic statistics about their chosen data range. These statistics readouts meant it was possible to answer every question without needing to memorize values or make any calculations by hand. The user interface design was consistent across all three test conditions.

In each test condition, the participants were provided with six multiple-choice questions on the dataset. For the participants in VR, the questions were displayed in the application above the visualization (see Fig. 6a). Each set of questions required the same number of interactions to be completed with the simpler, single selection questions appearing first and the more time consuming multi-selection questions appearing later. Table 1 lists the questions used in the VR task. The other two test conditions employed similar types of questions provided on paper.

(a) WebVR and HMD evaluation (task 1)



(b) Desktop-based 2D evaluation (task 3)

Fig. 5: A study participant evaluating the WebVR IA application, using our <VRIA> environment with an Oculus Rift and Touch controller (a), and the 2D evaluation task, with mouse controls (b).

Table 1: Questions for VR Test Condition

1) What was the total energy consumption in 2000 across all 4 sectors?
159,365 ktoe 160,925 ktoe 159,102 ktoe 158,710 ktoe

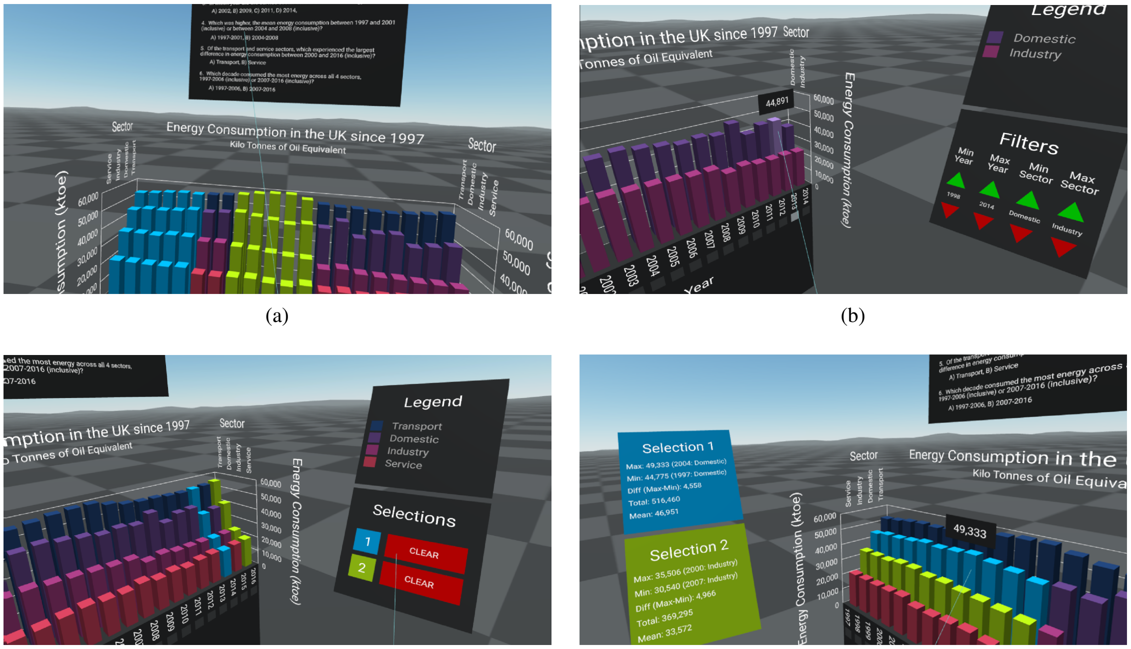
2) Which sector consumed the most energy in 2015?
Transport Domestic Industry Service

3) In which year did the service sector consume the least energy?
2002 2009 2011 2014

4) Which was higher, the mean energy consumption between 1997 and 2001 (inclusive) or between 2004 and 2008 (inclusive)?
1997 and 2001 2004 and 2008

5) Of the transport and service sectors, which experienced the largest difference in energy consumption between 2000 and 2016 (inclusive)?
Transport Service

6) Which decade consumed the most energy across all 4 sectors, 1997-2006 (inclusive) or 2007-2016 (inclusive)?
1997-2006 2007-2016



(a) (b) (c) (d)

Fig. 6: Different views of our evaluation application, built with <VRIA>, depicting a 3D bar chart of the UK energy consumption data between 1997 and 2016. The application presented the evaluation questions in VR (a), provided menus for filtering, brushing and selecting, (b) and (c), and used text-based information overlays (d). All GUI elements were accessible using the Oculus Rift's Touch controllers in test condition 1 and via mouse in test condition 2.

4.2 Results

In this section we present and discuss the results of the study.

4.2.1 Simulator Sickness

All participants completed the SSQ questionnaire [32] before and after using the HMD set-up. The results for the three symptom clusters, Nausea (N), Oculomotor (O) and Disorientation (D), and the Total Severity (TS) are presented in Table 2. On average the participants reported none to slight levels of nausea, oculomotor issues and disorientation, with only one participant reporting slight levels of oculomotor issues after the experiment.

Table 2: Simulator Sickness Questionnaire Findings

		Nausea	*Oculomotor*	*Disorientation*	*TS*
Before	**Mean**	4.13	10.36	6.50	3.55
	SD	6.94	11.69	10.17	4.47
	Min	0.00	0.00	0.00	0.00
	Max	28.62	37.90	27.84	14.48
After	**Mean**	10.18	17.94	19.02	8.54
	SD	13.93	16.35	22.38	8.70
	Min	0.00	0.00	0.00	0.00
	Max	47.70	60.64	83.52	32.11

☐ *None to slight*
☐ *slight to moderate*

4.2.2 System Usability

All participants completed a System Usability Scale [3] questionnaire after each task to rate their user experience. In terms of the usability assessment, the average SUS score for the 3D HMD task was 77.75 (SD=12.7), indicating 'good' usability in the adjective scale presented by Bangor et al. [3] (see Fig. 7). Scores ranged from a minimum of 45 to a maximum of 95. Likewise, we collected SUS scores for the other two set-ups, for comparison and completions. The 2D desktop set-up was deemed good at 82.08 (SD=12.72), and the 3D desktop set up was deemed poor at 63.08 (SD=17.82).

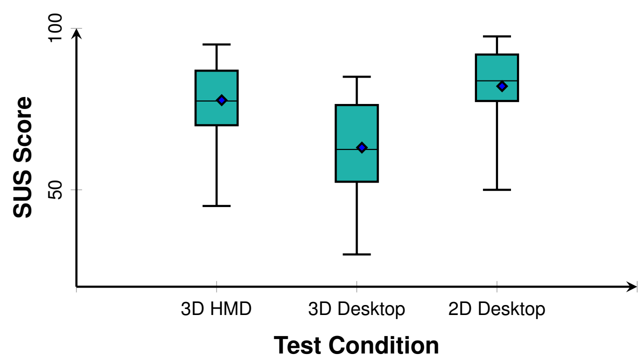


Fig. 7: Box plot of SUS scores. The usability of the 2D desktop set-up is high, compared to the pseudo-3D variant. The HMD-based system is close in perceived usability to the 2D set-up, but perceived as being marginally less usable.

4.2.3 Open-ended Questions

Upon completion of each task, participants were given the chance to write down any feedback they had, in terms of what they liked and disliked about that task and why. They were also asked whether there was anything they would like to see implemented, improved or changed to make their experience better. We collected just under 300 comments in total, with a little over 200 constructive comments related to the two tasks involving <VRIA>. We will only focus on comments relevant to the HMD-based task, and those that specifically relate to aspects of the <VRIA> application used in the two 3D set-ups.

Positive sentiments that participants used to describe their experience included: satisfying, fun, novel, entertaining, intuitive, enjoyable, amazing, engaging, unique and responsive. A number of participants stated that they found the HMD-based task *"more enjoyable"* and *"more engaging"* than the other tasks. Prominent positive comments included a number of participants stating that they were *"able to better understand the data when it looked physical"* and were *"able to spot trends more easily and didn't default to using filters and selections"*. One user remarked that the HMD-based VR gave them a *"unique perspective of the data"*. Comments relating to immersion included one user who remarked that they were able to *"focus on the task without stopping"* as they weren't being affected by *"peripheral distractions present in the real world"*. More common comments related to immersion, included users stating that they *"felt immersed in the data"* and *"liked that it felt like another world"*.

Around half of all positive comments related specifically to the available interactions with a large number of those comments stating that they felt the *"controls were well tailored to the task"* or that they *"enjoyed using the laser pointer"*, and in one case *"preferred controller to mouse"*. Three participants felt that *"physical interactions made the experience more interesting"*. A handful of users stated that *"interactions felt very natural even though it was the first time"* that they had used them. A couple of comments eluded to the bars and control elements themselves: *"Bars and controls were large enough for the sensitivity of the controller"*, *"Very responsive, liked feedback of interactive elements"*.

General comments related to the usability of the application included a number of participants calling it *"clean"*, *"simple"* and *"easy to use"*. Two participants commented on the use of color, reporting a *"good use of clear color differentiation"*. Two users noted that that the UI elements were *"nicely separated, giving each space its own focus"*.

The single most common comment came from eleven people expressing that they "had to turn [their] head a lot between selections, questions and visualizations". Other negative comments included mention of a "lack of a hover prompt on filters" and difficulty trying to read "vertically aligned text". Usability issues raised included a mention that the pointer controller was *"harder to use on distant objects"*. One participant also shared that they found the task *"entertaining"* but that it *"didn't feel as efficient as other [tasks]"*.

The 3D desktop set-up employed inverted mouse drag to move the camera around in the environment which eight participants *"found unfamiliar and counterintuitive"*. Users also found it difficult to *"select years at the edge of the chart because of the perspective"*

Most participants left improvement suggestions, the most popular (made by seven participants), was the ability to *"allow objects in the scene to be moved [as well as scaled] by the user"*. Six comments suggested that we *"move scene objects closer together to reduce the amount of head turning"*. Suggestions to improve the user interface and interactions included the ability to *"place the study questionnaire [and/or other UI elements] on to the hand controllers"* so that a user may glance at them at wherever they're facing in the scene. There were also requests for the addition of *"options to clear all selections and reset filters"*, rather than having to reset filters and clear selections individually.

4.2.4 Task Efficiency

Completion times and accuracy data (a mark out of six per set of questions) for each test condition were run through a repeated measures ANOVA within-between interactions using a Bonferroni post-hoc statistical test. The difference in accuracy between the different test conditions showed no statistical significance, largely due to there only being a total of eight mistakes made across the entire sample, by seven participants. Three mistakes were made in the HMD-based test condition, three in the 3D desktop test condition, and two in the 2D desktop test condition. There was however statistical significance in completion times between the HMD-based and 2D test conditions, and between the 3D and 2D tasks on desktop, as shown in Table 3 and Fig. 8. Once outliers are omitted, the mean task completion times were:

- *Test Condition 1:* 231.8s

Table 3: Pairwise Comparisons

(I) Task	(J) Task	Mean Diff (I-J)	Std. Error	Sig.^{b}	95% Confidence Interval for Difference^b. Lower Bound	Upper Bound
1	2	-9.063	10.325	1.000	-35.636	17.510
	3	45.743*	7.653	.000	26.048	65.439
2	1	9.063	10.325	1.000	-17.510	35.636
	3	54.806*	13.976	.002	18.837	90.775
3	1	-45.743*	7.653	.000	-65.439	-26.048
	2	-54.806*	13.976	.002	-90.775	-18.837

Based on estimated marginal means

⋆. The mean difference in significant at the 0.5 level.

b. Adjustment for multiple comparisons: Bonferroni.

- *Test Condition 2:* 232.4s
- *Test Condition 3:* 160.5s

Although the HMD-based and 3D desktop-based task means ended up being practically identical, our initial expectations for task efficiency held true because of the lower SUS score of the 2D desktop condition.

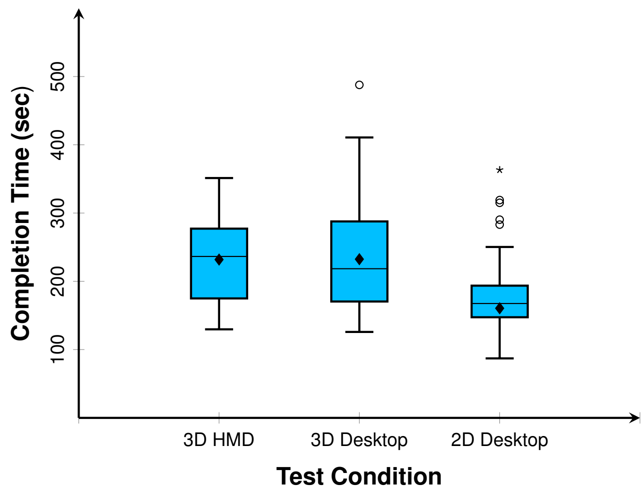


Fig. 8: Boxplots showing the difference in completion times across the three test conditions.

4.3 Discussion

As indicated by the low occurrences of simulator sickness symptoms, the rendering performance of the experience produced by <VRIA> was satisfactory for this task. <VRIA> shows the potential to create good quality solutions that render satisfactorily on WebVR-supporting browsers, and can be experienced without side effects using contemporary HMDs.

Overall, although users took longer on average to complete each of the tasks that used <VRIA> applications (test condition 1 and 2), compared with the traditional 2D application (test condition 3), user feedback suggests that their experience with the HMD was more compelling than that of the other tasks. This was demonstrated by those that stated that it was the immersive nature of the virtual environment that enabled them to gain enough understanding of the data to answer some of the questions without relying on or defaulting to using the available interactions. In that regard, VR has the potential to create experiences that engage the user's more effectively on a preliminary stage despite the limitations, in terms of analytical versatility. Moreover, our observations highlight the need to investigate more 'what works in 3D', that could be successfully coupled with what appears to be perceived as a more compelling experience.

Completion times in the <VRIA> tasks were affected largely by the amount of time participants were having to move their head between the various parts of the visualization. Feedback from the open-ended questions and the task completion times for the 2D test condition suggest that moving user interfaces further into view may help speed up completion time and increase efficiency. When designing a user interface for a VR environment, our temptation is to utilize the space around users to encourage them to move around and build up their sense of immersion. Our study has demonstrated that whilst this may lead to further enjoyment or appreciation of the virtual world, it does not improve task efficiency.

5 Lessons Learned and Future Work

During the production of <VRIA> and our evaluation application, we faced several technical challenges as outlined in Sect. 3. These challenges required us to implement performance optimizations to ensure <VRIA> applications could maintain adequate frame rates when rendering large datasets with real-time interactions. Whilst we opted for a Desktop HMD for our evaluation set-up, <VRIA> is still able to provide experiences on less powerful devices such as smartphones. WebVR and <VRIA> advocate for progressive enhancement, a fundamental web strategy that adds extra presentational or interactive features to a web page or application when supported hardware is available.

Another optimization involved the inclusion of a state management system. As our component trees grew more complex, and the need to pass data down to deeply nested components increased, we integrated Redux, one of the many state management solutions available for use in JavaScript applications. Since its inclusion, Redux has simplified the flow of data through <VRIA> applications, resulting in a cleaner code base. The Web ecosystem has numerous tools that offer optimizations and abstractions, towards improving performance. Integrating such tools in Web-based immersive data visualization systems can result in simpler and more elegant implementations, much like with <VRIA>.

Moreover, the synergistic use of established visualization libraries with <VRIA> can be an enabler for the visualization community, allowing more opportunities for investigations and application development. For example, although we do not employ D3.js for our evaluation set-up, we have been using it seamlessly with <VRIA>, such as in our scatterplot component. In that regard, our experience from developing <VRIA> strengthens our belief that this synthesis of open-standards tools, over the HTML DOM is a strong foundation for immersive data visualizations.

However, this approach faces challenges, as web browsers are only now beginning to have the performance capabilities and features necessary to run VR solutions. Moreover, many of the open standards that facilitate VR development, such as WebVR/XR, are under early stages of development. As the tools and frameworks we use in framework evolve, new functionality, such as React's Time-Slicing [24] that prioritizes I/O and user interactions over DOM updates, thus improving rendering speed, will enhance the capabilities of <VRIA>.

While <VRIA> currently does all of its computations on the client, a more scalable solution, for more complex investigations, is to move interaction computations to a server and handle interactions asynchronously. This will ensure the user gets a smooth VR experience, but potentially at the cost of increased interaction latency. It is also possible to use Web Workers (background scripts) to break large computations out of the main thread but this still occurs on the client and may not increase performance on less powerful devices.

Finally, eliciting the thoughts of non-expert users has provided us with objective feedback on practically every aspect of an application, built with <VRIA>. The results of our evaluation will inform future design decisions, and have set performance base-lines in terms of usability and user experience. From our investigation, it is evident there is a need for experimental investigations, about 'what works in 3D' for data visualization. Secondary aspects that facilitate analytical tasks, such as allowing users to position control objects within the space to their liking and not too far out of their peripheral vision, should also be investigated. Moreover, as the Web is a natural setting for collaborative investigations, we plan to extend our framework with features that enable such functionality.

6 Conclusions

We presented <VRIA>, a framework for building Immersive Analytics solutions in VR, using open-standards Web-technologies. <VRIA> is built using WebVR, A-Frame and React. The resulting VR solutions can be experienced through a WebVR-compliant browser in a variety of devices, ranging from smartphones to HMD-equipped desktop computers. <VRIA> uses a declarative format for specifying visualization types through simple configuration files, simplifying visualization prototyping, data binding and interaction configuration. We have evaluated our framework in terms of the overall user experience and occurrences of simulator sickness, using a typical desktop-workstation set-up, and the Oculus Rift HMD, with its Touch controllers. Our results demonstrate that <VRIA> allows the creation of very compelling data-evaluation experiences in VR. However, they also demonstrate the challenges that immersive data visualization faces in terms of how it compares to more traditional methods of interactive analysis, at least as well-known visualizations such as a bar chart is concerned. Indeed, the question of what works effectively in 3D immersive visualization, in terms of visualizations and interaction techniques is outstanding.

As the interest in immersive data visualization increases, there is a clear need to investigate and devise appropriate visualization techniques, tailor-made for immersive 3D graphical environments. We believe that <VRIA> is in a unique position to facilitate such investigations, as it allows researchers to deploy their experimental set-ups though a Web-browser in a variety of devices, and consequently increase participation and data-collection opportunities.

Acknowledgments

We would like to thank Danielle Albers Szafir, Maxime Cordeil, Benjamin Bach and Jonathan C. Roberts for discussions that inspired and influenced this paper.

References

[1] B. Alper, T. Hollerer, J. Kuchera-Morin, and A. Forbes. Stereoscopic Highlighting: 2D Graph Visualization on Stereo Displays. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2325–2333, Dec 2011. doi: 10.1109/TVCG.2011.234

[2] B. Bach, R. Sicat, J. Beyer, M. Cordeil, and H. Pfister. The Hologram in My Hand: How Effective is Interactive Exploration of 3D Visualizations in Immersive Tangible Augmented Reality? *IEEE Transactions on Visualization and Computer Graphics*, 24(1):457–467, Jan 2018. doi: 10.1109/TVCG.2017.2745941

[3] A. Bangor, P. T. Kortum, and J. T. Miller. An Empirical Evaluation of the System Usability Scale. *International Journal of Human-Computer Interaction*, 24(6):574–594, 2008. doi: 10.1080/10447310802205776

[4] M. Bostock and J. Heer. Protovis: A Graphical Toolkit for Visualization. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1121–1128, Nov 2009. doi: 10.1109/TVCG.2009.174

[5] M. Bostock, V. Ogievetsky, and J. Heer. D^3 Data-Driven Documents. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2301–2309, Dec 2011. doi: 10.1109/TVCG.2011.185

[6] J. Brooke. SUS-A quick and dirty usability scale. *Usability evaluation in industry*, 189(194):4–7, 1996.

[7] F. P. Brooks. What's Real About Virtual Reality? *IEEE Computer Graphics & Applications*, 19(6):16–27, Nov. 1999. doi: 10.1109/38.799723

[8] S. Bryson. Virtual Reality in Scientific Visualization. *Communications of the ACM*, 39(5):62–71, May 1996. doi: 10.1145/229459.229467

[9] P. W. Butcher and P. D. Ritsos. Building Immersive Data Visualizations for the Web. In *Procs. of International Conference on Cyberworlds (CW'17)*, pp. 142–145, Sept 2017. doi: 10.1109/CW.2017.11

[10] P. W. Butcher, J. C. Roberts, and P. D. Ritsos. Immersive Analytics with WebVR and Google Cardboard. In *Posters of the IEEE Conference on Visualization (IEEE VIS 2016), Baltimore, MD, USA*, 2016.

[11] S. Butscher, S. Hubenschmid, J. Müller, J. Fuchs, and H. Reiterer. Clusters, Trends, and Outliers: How Immersive Technologies Can Facilitate the Collaborative Analysis of Multidimensional Data. In *Proceedings of the Annual ACM Conference on Human Factors in Computing Systems*, CHI '18, pp. xx–xx. ACM, New York, NY, USA, 2018. doi: 10.1145/3173574.3173664

[12] T. Chandler, M. Cordeil, T. Czauderna, T. Dwyer, J. Glowacki, C. Goncu, M. Klapperstueck, K. Klein, K. Marriott, F. Schreiber, and E. Wilson. Immersive Analytics. In *Procs. of Big Data Visual Analytics*, (BDVA), pp. 1–8, Sept 2015. doi: 10.1109/BDVA.2015.7314296

[13] M. Cordeil, A. Cunningham, T. Dwyer, B. H. Thomas, and K. Marriott. ImAxes: Immersive Axes As Embodied Affordances for Interactive Multivariate Data Visualisation. In *Procs. of the ACM Symposium on User Interface Software and Technology*, (UIST), pp. 71–83. ACM, New York, NY, USA, 2017. doi: 10.1145/3126594.3126613

[14] M. Cordeil, T. Dwyer, K. Klein, B. Laha, K. Marriott, and B. H. Thomas. mmersive Collaborative Analysis of Network Connectivity: CAVE-style or Head-Mounted Display? *IEEE Transactions on Visualization and Computer Graphics*, 23(1):441–450, Jan 2017. doi: 10.1109/TVCG.2016.2599107

[15] G. de Haan, M. Koutek, and F. H. Post. Towards Intuitive Exploration Tools for Data Visualization in VR. In *Procs. of the ACM Symposium on Virtual Reality Software and Technology*, (VRST), pp. 105–112. ACM, New York, NY, USA, 2002. doi: 10.1145/585740.585758

[16] Department for Business, Energy & Industrial Strategy, UK Government. *Energy Consumption in the UK*. [Online]. Available `https://www.gov.uk/government/collections/energy-consumption-in-the-uk`, 2016. Accessed: 28/03/18.

[17] J. Dirksen. *Learning Three. js: the JavaScript 3D library for WebGL*. Packt Publishing Ltd, 2013.

[18] C. Donalek, S. G. Djorgovski, A. Cioc, A. Wang, J. Zhang, E. Lawler, S. Yeh, A. Mahabal, M. Graham, A. Drake, et al. Immersive and collaborative data visualization using virtual reality platforms. In *Procs. of IEEE International Conference on Big Data*, pp. 609–614, Oct 2014. doi: 10.1109/BigData.2014.7004282

[19] M. Drouhard, C. A. Steed, S. Hahn, T. Proffen, J. Daniel, and M. Matheson. Immersive visualization for materials science data analysis using the Oculus Rift. In *Procs. of the IEEE International Conference on Big Data (Big Data)*, pp. 2453–2461, Oct 2015. doi: 10.1109/BigData.2015.7364040

[20] N. Elmqvist and P. Irani. Ubiquitous Analytics: Interacting with Big Data Anywhere, Anytime. *Computer*, 46(4):86–89, April 2013. doi: 10.1109/MC.2013.147

[21] N. Elmqvist, A. Vande Moere, H. C. Jetter, D. l. Cernea, H. Reiterer, and T. Jankun-Kelly. Fluid interaction for information visualization. *Information Visualization*, 10(4):327–340, Oct. 2011. doi: 10.1177/1473871611413180

[22] N. A. ElSayed, B. H. Thomas, K. Marriott, J. Piantadosi, and R. T. Smith. Situated Analytics: Demonstrating immersive analytical tools with Augmented Reality. *Journal of Visual Languages & Computing*, 36:13 – 23, 2016. doi: 10.1016/j.jvlc.2016.07.006

[23] A. Evans, M. Romeo, A. Bahrehmand, J. Agenjo, and J. Blat. 3D graphics on the web: A survey. *Computers & Graphics*, 41:43 – 61, 2014. doi: 10.1016/j.cag.2014.02.002

[24] Facebook Inc. *Sneek Peak: Beyond React 16*. [Online]. Available `https://reactjs.org/blog/2018/03/01/sneak-peek-beyond-react-16.html`, 2018. Accessed: 17/03/18.

[25] D. Germans, H. J. W. Spoelder, L. Renambot, and H. E. Bal. VIRPI: A High-level Toolkit for Interactive Scientific Visualization in Virtual Reality. In *Procs. of the Eurographics Conference on Immersive Projection Technology and Virtual Environments*, (EGVE), pp. 109–120. Eurographics Association, 2001. doi: 10.1007/978-3-7091-6221-7_12

[26] Google Inc. *WebVR Polyfill*. [Online]. Available `https://github.com/googlevr/webvr-polyfill/`, 2015. Accessed: 14/05/16.

[27] Google Inc. *Google Cardboard*. [Online]. Available `https://vr.google.com/cardboard/`, 2017. Accessed: 13/05/17.

[28] J. Heer, S. K. Card, and J. A. Landay. Prefuse: A Toolkit for Interactive Information Visualization. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '05, pp. 421–430. ACM, New York, NY, USA, 2005. doi: 10.1145/1054972.1055031

[29] I. Herman, G. Melancon, and M. S. Marshall. Graph visualization and navigation in information visualization: A survey. *IEEE Transactions on Visualization and Computer Graphics*, 6(1):24–43, Jan 2000. doi: 10.1109/2945.841119

[30] Y. Jansen and P. Dragicevic. An Interaction Model for Visualizations Beyond The Desktop. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2396–2405, Dec 2013. doi: 10.1109/TVCG.2013.134

[31] Y. Jansen, P. Dragicevic, and J.-D. Fekete. Evaluating the Efficiency of Physical Visualizations. In *Procs. of SIGCHI Conference on Human Factors in Computing Systems*, CHI, pp. 2593–2602. ACM, New York, NY, USA, 2013. doi: 10.1145/2470654.2481359

[32] R. S. Kennedy, N. E. Lane, K. S. Berbaum, and M. G. Lilienthal. Simulator sickness questionnaire: An enhanced method for quantifying simulator sickness. *The International Journal of Aviation Psychology*, 3(3):203–220, 1993. doi: 10.1207/s15327108ijap0303_3

[33] O. Kwon, C. Muelder, K. Lee, and K. Ma. A Study of Layout, Rendering, and Interaction Methods for Immersive Graph Visualization. *IEEE Transactions on Visualization and Computer Graphics*, 22(7):102–1815, July 2016. doi: 10.1109/TVCG.2016.2520921

[34] B. Lee, P. Isenberg, N. H. Riche, and S. Carpendale. Beyond Mouse and Keyboard: Expanding Design Considerations for Information Visualization Interactions. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2689–2698, Dec 2012. doi: 10.1109/TVCG.2012.204

[35] A. Lu, J. Huang, S. Zhang, C. Wang, and W. Wang. Towards Mobile Immersive Analysis: A Study of Applications. In J. Chen, E. G. Marai, K. Mariott, F. Schreiber, and B. H. Thomas, eds., *Procs. of Immersive Analytics Workshop, IEEE VR*, March 2016. doi: 10.1109/IMMERSIVE.2016.7932378

[36] M. Luboschik, P. Berger, and O. Staadt. On Spatial Perception Issues In Augmented Reality Based Immersive Analytics. In *Procs. of ACM International Conference on Interactive Surfaces and Spaces*, ISS, pp. 47–53. ACM, New York, NY, USA, 2016. doi: 10.1145/3009939.3009947

[37] Masters of Pie. *Masters of Pie, Winners of the big data VR challenge 2015*. [Online]. Available `http://www.mastersofpie.com/big-data-vr-challenge-winners/`, 2017. Accessed: 13/05/17.

[38] T. Merrel. *Is the NASDAQ in Another Bubble? A virtual reality tour of the NASDAQ*. WSJ [Online]. Available: `http://graphics.wsj.com/3d-nasdaq/`, 2015. [Accessed: 6-Mar-2017].

[39] Michal Takac. *MathworldVR*. [Online]. Available `https://mathworldvr.com/`, 2017. Accessed: 24/03/18.

[40] Mozilla. *A-Frame*. [Online]. Available `https://aframe.io`, 2017. Accessed: 18/04/17.

[41] T. Munzner. Exploring large graphs in 3D hyperbolic space. *IEEE Computer Graphics & Applications*, 18(4):18–23, Jul 1998. doi: 10.1109/38.689657

[42] Oculus VR, LLC. *Oculus Rift*. [Online]. Available `https://www.oculus.com/rift/`, 2017. Accessed: 13/05/17.

[43] OpenSimulator. *OpenSimulator*. [Online]. Available `http://opensimulator.org/`, 2017. Accessed: 18/04/17.

[44] R. Cabello. *Three.js*. [Online]. Available `https://github.com/mrdoob/three.js`, 2017. Accessed: 13/05/17.

[45] D. Ren, B. Lee, and T. Höllerer. Stardust: Accessible and Transparent GPU Support for Information Visualization Rendering. *Computer Graphics Forum*, 36(3):179–188, 2017. doi: 10.1111/cgf.13178

[46] P. D. Ritsos, J. Jackson, and J. C. Roberts. Web-based Immersive Analytics in Handheld Augmented Reality. In *Posters presented at the IEEE Conference on Visualization (IEEE VIS 2017), Phoenix, Arizona, USA*, 2017.

[47] P. D. Ritsos, J. Mearman, J. R. Jackson, and J. C. Roberts. Synthetic Visualizations in Web-based Mixed Reality. In B. Bach, M. Cordeil, T. Dwyer, B. Lee, B. Saket, A. Endert, C. Collins, and S. Carpendale, eds., *Immersive Analytics: Exploring Future Visualization and Interaction Technologies for Data Analytics Workshop, IEEE Conference on Visualization (VIS), Phoenix, Arizona, USA*, Oct. 2017.

[48] J. C. Roberts, P. D. Ritsos, S. K. Badam, D. Brodbeck, J. Kennedy, and N. Elmqvist. Visualization Beyond the Desktop - the next big thing. *IEEE Computer Graphics & Applications*, 34(6):26–34, Nov. 2014. doi: 10.1109/MCG.2014.82

[49] G. G. Robertson, S. K. Card, and J. D. Mackinlay. Information Visualization Using 3D Interactive Animation. *Communications of the ACM*, 36(4):57–71, Apr. 1993. doi: 10.1145/255950.153577

[50] R. Sadana, V. Setlur, and J. Stasko. Redefining a Contribution for Immersive Visualization Research. In *Procs. of the ACM Companion on Interactive Surfaces and Spaces*, ISS, pp. 41–45. ACM, New York, NY, USA, 2016. doi: 10.1145/3009939.3009946

[51] A. Satyanarayan, D. Moritz, K. Wongsuphasawat, and J. Heer. Vega-Lite: A Grammar of Interactive Graphics. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):341–350, Jan 2017. doi: 10.1109/TVCG.2016.2599030

[52] A. Satyanarayan, R. Russell, J. Hoffswell, and J. Heer. Reactive Vega: A Streaming Dataflow Architecture for Declarative Interactive Visualization. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):659–

668, Jan 2016. doi: 10.1109/TVCG.2015.2467091

[53] A. Satyanarayan, K. Wongsuphasawat, and J. Heer. Declarative Interaction Design for Data Visualization. In *Procs of the ACM Symposium on User Interface Software and Technology*, UIST '14, pp. 669–678. ACM, New York, NY, USA, 2014. doi: 10.1145/2642918.2647360

[54] V. Vukicevic and B. Jones and K. Gilbert and C. Van Wiemeersch. *WebXR Device API Specifications*. [Online]. Available https://immersive-web.github.io/webvr/spec/1.1/, 2016. Accessed: 26/03/18.

[55] A. van Dam, A. S. Forsberg, D. H. Laidlaw, J. J. LaViola, and R. M. Simpson. Immersive VR for scientific visualization: a progress report. *IEEE Computer Graphics & Applications*, 20(6):26–52, Nov 2000. doi: 10.1109/38.888006

[56] Virtualitics. *VIP - Virtualitics Immersive Platform*. [Online]. Available https://www.virtualitics.com/, 2018. Accessed: 24/03/18.

[57] W3C. *WebVR*. [Online]. Available https://webvr.info/, 2017. Accessed: 18/04/17.

[58] C. Ware and G. Franck. Evaluating Stereo and Motion Cues for Visualizing Information Nets in Three Dimensions. *ACM Transactions on Graphics*, 15(2):121–140, Apr. 1996. doi: 10.1145/234972.234975

[59] C. Ware and P. Mitchell. Visualizing Graphs in Three Dimensions. *ACM Transactions on Applied Perception*, 5(1):2:1–2:15, Jan. 2008. doi: 10.1145/1279640.1279642

[60] H. Wickham. *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York, 2009.

[61] W. Willett, Y. Jansen, and P. Dragicevic. Embedded Data Representations. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):461–470, Jan 2017. doi: 10.1109/TVCG.2016.2598608

[62] World Wide Web Consortium. *Frame Timing*. [Online]. Available https://www.w3.org/TR/frame-timing/, 2016. Accessed: 26/03/18.

[63] S. Zhang, C. Demiralp, D. F. Keefe, M. DaSilva, D. H. Laidlaw, B. D. Greenberg, P. J. Basser, C. Pierpaoli, E. A. Chiocca, and T. S. Deisboeck. An immersive virtual environment for DT-MRI volume visualization applications: a case study. In *Procs. of Visualization*, pp. 437–584, Oct 2001. doi: 10.1109/VISUAL.2001.964545