TU WIEN Informatics

Projectverse: A web-based Virtual Reality Platform for Visualizing Research Projects

MASTERARBEIT

zur Erlangung des akademischen Grades

Master of Science

im Rahmen des Studiums

Visual Computing

eingereicht von

Tom Lautenbach, BSc.
Matrikelnummer 12044805

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Univ.Prof. Mag.rer.nat. Dr.techn. Hannes Kaufmann
Mitwirkung: Dr.in. techn. Iana Podkosova, BSc. MSc.

Wien, 1. Dezember 2023

Tom Lautenbach

Hannes Kaufmann

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

Informatics

Projectverse: A web-based Virtual Reality Platform for Visualizing Research Projects

MASTER'S THESIS

submitted in partial fulfillment of the requirements for the degree of

Master of Science

in

Visual Computing

by

Tom Lautenbach, BSc.
Registration Number 12044805

to the Faculty of Informatics

at the TU Wien

Advisor: Univ.Prof. Mag.rer.nat. Dr.techn. Hannes Kaufmann
Assistance: Dr.in. techn. Iana Podkosova, BSc. MSc.

Vienna, 1st December, 2023

Tom Lautenbach	Hannes Kaufmann

Erklärung zur Verfassung der Arbeit

Tom Lautenbach, BSc.

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 1. Dezember 2023

Tom Lautenbach

Danksagung

Ich möchte meinen aufrichtigen Dank an alle aussprechen, die mich während des Entstehungsprozesses dieser Arbeit und auf dem Weg zu meinem Masterabschluss unterstützt haben. Insbesondere schätze ich die fortwährende Hilfe und Unterstützung meiner Familie und Freunde. Mein besonderer Dank gilt meiner Freundin, die mich stets ermutigt hat und fest an meine Arbeit glaubt. Ohne meine herausragende Betreuerin, Iana Podkosova, wäre diese Arbeit nicht annähernd so qualitätsvoll geworden. Sie hat nicht nur bei Projekten während meines Studiums, sondern auch speziell bei dieser Masterarbeit stets das Beste aus mir herausgeholt. Ihre regelmäßige Nachfrage nach meinen Fortschritten und ihre Ermutigungen, mich nur mit dem Besten zufrieden zu geben haben maßgeblich zu meinem Erfolg beigetragen. Abschließend möchte ich meine Dankbarkeit für die Möglichkeit, Projectverse auf der Website der VR-Gruppe zu präsentieren, zum Ausdruck bringen.

Acknowledgements

Thank you to all those supporting me in the creation and writing of this thesis and achieving my master's degree. I am especially thankful for the continuous help and support provided by my family and friends. I am also grateful for my girlfriends persistent encouragement and belief in my work and her providing comfort in the hard times. Furthermore, this work would not have been half as good without my wonderful supervisor, Iana Podkosova, who in projects as part of my studies as well as this master thesis always brought out the best in me by regularly checking on my progress, encouraging me to go the extra mile and not settling for any shortcuts. Lastly, I am grateful for the opportunity to run Projectverse on the VR group website.

Kurzfassung

Diese Arbeit befasst sich mit dem Entwurf, der Implementierung und der Evaluierung von Projectverse, einer webbasierten Virtual Reality (VR)-Plattform. Projectverse wurde mit der Absicht geschaffen, ein besseres Informationsmedium über die Arbeit der VR Group der TU Wien anzubieten. Projectverse bietet eine Multi-User-Umgebung, in der die Forschungsprojekte der VR Group als Sterne in Sternenkonstellationen am Nachthimmel visualisiert werden. Die BenutzerInnen können mit diesen interagieren, indem sie sie mit einer Laserpistole abschießen. Wurde auf einen Stern geschossen, so werden dem Benutzer weitere Informationen über das Projekt, das dieser Stern repräsentiert, angezeigt. Schließlich können die NutzerInnen in Projectverse in prototypische Anwendungen von Forschungsprojekten wechseln und diese ausprobieren.

Projectverse nutzt die WebXR Device API für die Kommunikation mit VR-Geräten. Die Plattform wurde in der Unity-Spiele-Engine entwickelt, daher wurde außerdem das Unity WebXR-Plugin verwendet, mit dem WebXR-Daten in Unity nutzbar gemacht werden. Außerdem wird der WebXR-Exporter, ein weiteres Unity-Plugin, verwendet, um einen Software-Build zu erstellen, der in Webbrowsern ausgeführt werden kann. Multiuser-Sitzungen und Netzwerkfunktionen wurden mit Photon PUN 2 und Photon Voice 2 implementiert. Diese Frameworks wurden gewählt, da Photon einer der größten Anbieter von Netzwerklösungen für Unity ist.

Projectverse erweiterbar in dem Sinne, dass Forscher ihre eigenen Forschungsanwendungen an die Anforderungen der Plattform - vor allem an die Browsertauglichkeit - anpassen und sie anschließend auf die Plattform hochladen können. Solche Erweiterungen der auf Projectverse verfügbaren Projekte werden dadurch berücksichtigt, dass die Plattform automatisch eine neue Visualisierung generiert, sobald sich die Projekte ändern.

Um die Anpassung von Projektanwendungen an die Anforderungen von Projectverse zu vereinfachen und zu beschleunigen, wird ein Entwickler-Toolkit für Unity bereitgestellt. Das Toolkit enthält grundlegende VR-Funktionen wie Grabbing, Fortbewegung und UI-Interaktion sowie Prefabs, die für die Verwendung von VR-Geräten mit WebXR notwendig sind.

Die Auswertung einer User Study zu Projectverse zeigt, dass die Plattform zwar nicht unbedingt dazu beiträgt, dass man sich an spezifische Forschungsprojekte wie Forschernamen oder Forschungsbereiche einzelner Projekte besser erinnert, aber eine ansprechende,

motivierende und unterhaltsame Ergänzung der Website der VR-Gruppe für diejenigen darstellt, die sich über die Arbeit der Gruppe informieren wollen.

Abstract

This thesis addresses the design, implementation and evaluation of Projectverse, a web-based Virtual Reality (VR) platform created with the intent to provide a better medium for those who wish to inform themselves about the work of the TU Wien VR group. Projectverse offers a multi-user environment in which research projects of the VR group are visualized as stars in star constellations on a night sky. Users can interact with these by shooting them using a laser gun. Upon shooting a star, further information about the project it represents is shown to the user. Finally, users can transition into prototype applications of research projects and gain first-hand experience with these applications in Projectverse.

Projectverse makes use of the WebXR Device API for communicating with VR devices. The platform was developed in the Unity game engine, therefore we also used the Unity WebXR plugin, which allows us to use WebXR data in Unity. Furthermore, we use WebXR exporter, another Unity plugin, to build Projectverse in a state that allows it to be executed in web browsers. Multiuser sessions and networking features were implemented using Photon PUN 2 and Photon Voice 2. We chose these frameworks due to Photon being one of the largest providers of networking solutions for Unity.

Projectverse is extensible in the sense that researchers can adapt their own research applications to the platform's demands - most importantly: browser-readiness - and subsequently deploy them to it. Such extensions to the projects available on Projectverse are included by the platform automatically generating a new visualization once the projects change.

To simplify and accelerate the adaptation of project applications to the demands of Projectverse, we provide a developer toolkit for Unity that includes fundamental VR functionality such as grabbing, locomotion and UI interaction as well as prefabs crucial to using VR devices with WebXR.

Our evaluation shows that Projectverse provides an engaging, motivating and fun addition to the VR group website to those who wish to gain a deeper impression of the research conducted by the VR group.

Contents

CHAPTER 1

Introduction

With Virtual Reality (VR) devices becoming more affordable, and thus, more accessible to the general audience, VR has become a novel method of presenting information traditionally conveyed in the two dimensions of a screen. With VR devices, we can create immersive experiences in virtual worlds. Interactions with these environments can be made using the mouse and keyboard, classic gamepads, or specialized, tracked VR controllers. Hand-tracking is a new approach to interacting that does not require any devices to be used with the hands. To introduce the user into the virtual environment, normally we make use of a head-mounted display (HMD). These devices present rendered images from the 3D environment and either track their own position in space (inside-out tracking) or are tracked using e.g. base stations (outside-in tracking).

VR is used largely in entertainment and gaming, as games profit from stronger immersion. Furthermore, VR is used in non-gaming applications which include information platforms and exergames: Exergames are video games that require human body movements for interaction [14]. Information platforms, on the other hand, benefit from immersion and presence by providing longer-lasting experiences that induce better memory of the content conveyed. Finally, producing industries [4] as well as the health sector, make use of VR, e.g. for the purpose of rehabilitation programs [7]. A lot of research is available on VR and its applications. However, this research is published in papers, which are presented online or in journals. Getting an actual impression of the applications created for the purpose of research is difficult, whereas VR is a highly perception-focused medium - descriptive reports and papers are hard to understand unless the readers can actually experience the applications created themselves.

In this thesis, we present a novel approach for using VR in a web browser to provide an information platform that allows the general public to gain first-hand experience in applications that were developed in the course of research projects of the TU Wien Virtual Reality Group. We develop the platform alongside a process aimed at adapting and deploying research project applications to this platform. In addition, we develop a toolkit that simplifies this adaptation process. Finally, we evaluate our platform and compare it to the traditional website that the VR group offers to provide information on its work.

1.1 Motivation

Browser-based VR is an emerging field that promises an easier use of VR applications as well as a broader audience to the topic, since even modern smart phones are capable of providing a virtual reality experience with the help of e.g. cardboard "VR adapters". This trend offers the possibility of creating websites with immersive VR and AR content. It resolves the requirement for installation before using an application and hereby lowers entry barriers blocking access to this immersion.

The research of the Virtual Reality Group at TU Wien, which is conducted in all fields of the extended reality (XR) spectrum, results in research prototype applications that require a form of immersion. Many of these applications are developed using Unity game engine, with the outcomes usually being standalone software packages. The target display devices of most of these applications are VR systems such as Oculus Quest and Rift, products from the HTC Vive family, or Valve Index. Interested parties such as the general population, research personnel or students currently have no simple way of accessing these applications, since there is no provision of software in any way, neither web-based nor standalone.

Therefore, we need a platform that provides a space to manage, provide, deploy, and experience the software originating from the VR Group's research projects. Low barriers of entry are a key requirement towards this platform, as is the demand for ease-of-use in terms of deploying new applications to it as well as managing which of the deployed applications should actually be accessible. As a result, the platform must automatically recognize the software deployed to it and forward this information to the user. Furthermore, the platform's user interface toward the end user needs to mitigate potential user errors in exploring applications, contributing to a seamless user experience.

In addition, the platform should provide a social space to those exploring the research projects deployed to it, in which the users can communicate with each other and exchange about the projects. It should also provide information on each application deployed to the platform such as who participated in the project the application showcases, and what this project is about. This social space will therefore serve as the entry point to the platform and provide access to all research project software deployed to it.

1.2 Aim of the Work

This work aims to present research work in the field of Virtual Reality, produced by the TU Wien Virtual Reality Group. The presentation of this work should be web based, i.e. within the web browser. We identify three core functionalities this presentation of research work needs to provide:

Presenting VR research prototypes in an engaging way. The presentation of VR research prototypes demands for ease of use of the platform. This principle requires simple visual metaphors and simplistic design. While experienced users might be part of the application's audience, we must assume users with little to no VR experience, since the platform is intended to be hosted on the internet. The platform therefore must offer either guidance for unexperienced users, or be constructed in a way that leaves very little room for error. This includes the use of self-explanatory tools for interacting with the platform. Visualizations should provide all information needed to understand the contents of a research project.

Adaptation of applications to be deployed to the platform. To allow for a simple adaptation of applications to our platform, we propose a toolkit that simplifies this process. This toolkit should cover or at least simplify the basic steps to enable rendering to a VR headset, and building a project in such a way that it can be deployed to the web browser, and furthermore, the aforementioned platform. Additionally, the toolkit should provide basic VR interactions such as locomotion, rotating the player character in the scene (i.e., looking around), grabbing and releasing objects, and performing simple user interface interactions, e.g. clicking a button.

Flexible deployment of applications to the platform. This functionality demands easy access and ease-of-use in interaction with the platform's administration side. In order to achieve a simple, streamlined process in application deployment to the platform, it needs to automatically recognize new applications being deployed to it as well as automatically extend itself as new software is deployed to it, in order to avoid as many manual steps as possible. Thus, the platform should also automatically include newly added projects in its presentation to the user.

Resulting from the demands of these three functionalities, as well as the requirements discussed in Chapter 1, we propose the development of two main components in the course of this thesis, namely:

1. A website landscape, the main page of which should serve as the social space and entry point to the platform's applications. The subpages of the main page, on the other hand, should serve the research project applications deployed to the platform.
2. A developer toolkit, which enables researchers to adapt their applications to the platform's environment with as few manual steps as possible. This includes functionality as provided by Unity's XR Interaction Toolkit, as well as premade structures in the sense of prefabs, that allow for quick adaption of premade scenes to the web environment.

1.3 Structure of the Thesis

Chapter 2 gives an overview of technologies available for web-based VR as well as research related to this thesis. We discuss WebGL, A-Frame, and WebXR in combination with the Unity game engine. In related research, we cover applications like exergames and learning platforms that make use of web-based VR.

In Chapter 3, we establish the requirements of our platform, and how we plan to achieve them. An outline of the design of our platform is drawn, including its components and their interactions with each other.

Chapter 4 discusses how we implement our design. We explain how the lobby, star visualizations and jumping into projects work. Finally, we explain the implementation of our developer toolkit.

The evaluation of our platform is discussed in Chapter 5. We explain our methodology as well the design of our evaluation and finally discuss the results.

CHAPTER 2

Related Work

This chapter provides an overview of the state of the art in technologies that allow web browsers to execute VR experiences. Furthermore, we give a survey of current research and applications that make use of these technologies. We first explain the technology used in this thesis, as well as alternatives to it, then we provide an overview of applications accessible online and research papers. Technologies explained include WebGL, A-Frame, and the WebXR Device API in combination with Unity and the WebXR Integration plugin. Research papers on applications utilizing WebXR are not too numerous, yet we attempt to provide a good overview of the research landscape.

2.1 Technology for web-based VR

A range of tools and frameworks are available for bringing VR applications to the web and executing them within the browser. An overview of how the presented technologies interact in the toolstack is given in Figure 2.1. While the WebGL graphics API handles communicating with the GPU to render images, the WebXR Device API is used to retrieve tracking and input signals from VR devices when they are used in web applications. To use these signals in Unity, the WebXR Unity Plugin is required, whereas A-Frame can directly process information from WebXR. Finally, the game engine is responsible for the gameplay logic, and eventually, sending draw calls to the graphics API.

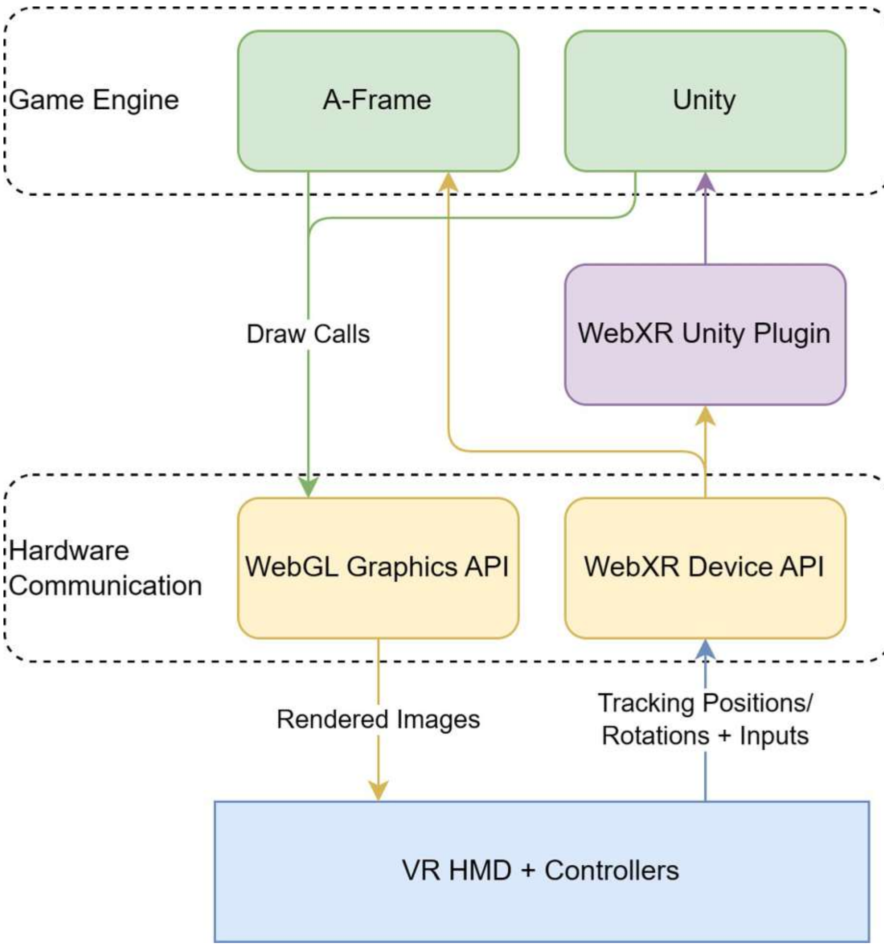


Figure 2.1: Technologies used for web-based VR.

2.1.1 WebGL

One of the lowest-level approaches to rendering 3D scenes in web browsers is WebGL, a 3D graphics API developed by the KHRONOS group [1], based on OpenGL ES. The technology extends the HTML5 canvas element and allows programming at the shader level. Srivastava [16] points out frameworks that simplify WebGL development, namely:

- **Three.js**, a JavaScript library used to create 3D content. Low in complexity and lightweight, Three.js is commonly used in web-based graphics programming.
- **Scene.js**, too, is based on JavaScript, but, in contrast to Three.js, focuses on high performance when rendering a large number of objects. It therefore is especially useful in the engineering and data visualization fields.

[1] `https://www.khronos.org/api/webgl`

- **GLGE** wraps WebGL's low-level instructions into more useful instructions and also intends to simplify 3D development.

WebGL is widely supported by most web browsers, since Apple, Google, Microsoft, and Mozilla are all members of the WebGL Working Group that develops the API[2].

One of the newest additions to web-based graphics APIs is WebGPU. Currently in the experimental state, the API supersedes WebGL, and allows for better compatibility with modern GPUs, as well as "general-purpose GPU computations, faster operations and access to more advanced GPU features"[9]. It is developed by the W3C group [3]. However, there are no notable approaches to implementing VR applications using the WebGPU software yet; thus, the community around this topic is still very small.

2.1.2 A-Frame

A-Frame[4] is a framework that allows the creation of VR applications. Based on HTML, A-Frame presents an approach in which developers can implement applications purely within an HTML file. An example for a very simple A-Frame scene can be observed in listing 2.1.

```html
<html>
  <head>
    <script src="https://aframe.io/releases/1.4.0/aframe.min.js
      "></script>
  </head>
  <body>
    <a-scene>
      <a-box position="-1 0.5 -3" rotation="0 45 0" color="#4
        CC3D9"></a-box>
      <a-sphere position="0 1.25 -5" radius="1.25" color="#
        EF2D5E"></a-sphere>
      <a-cylinder position="1 0.75 -3" radius="0.5" height="1.5
        " color="#FFC65D"></a-cylinder>
      <a-plane position="0 0 -4" rotation="-90 0 0" width="4"
        height="4" color="#7BC8A4"></a-plane>
      <a-sky color="#ECECEC"></a-sky>
    </a-scene>
  </body>
</html>
```

Listing 2.1: A-Frame example.

[2]https://www.khronos.org/webgl/
[3]https://www.w3.org/community/gpu/
[4]https://aframe.io/

This example produces the scene shown in Figure 2.2. This scene is already VR-enabled and interactable in 2D, i.e. the camera can be rotated using the mouse.

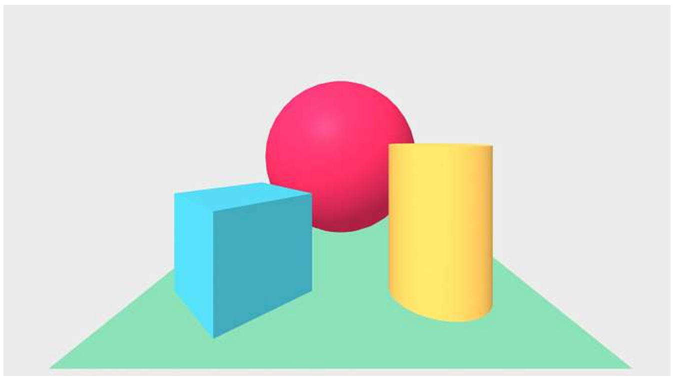


Figure 2.2: A-Frame example scene.

A-Frame, like WebGL, is compatible with most modern browsers. The developers specifically name Chrome, Microsoft Edge, and Opera to be supported.

2.1.3 WebXR Device API

Enabling developers to easily create simple scenes, A-Frame, however, requires more work when creating complex environments and application logic. Especially the handling of XR devices needs to be covered by either custom developments or the use of the WebXR device API, the most frequently used technology for bringing VR to the web browser. Being developed by the W3C group[5], WebXR is a device API that provides methods to receive input data from VR and AR devices such as pose (position and rotation) information from a headset and controllers. The API also handles output to the respective devices in the form of rendering to the devices' displays. A wide range of frameworks are supported, such as the aforementioned A-Frame, Needle Engine [6], or Unity [7]. However, the rendering part of an application using WebXR can also be developed in WebGL, thus also enabling lower-level approaches. Major benefits developers gain from working with WebXR include simplified access to tracking device positions, a standardized interface between application logic and the XR hardware, as well as the aforementioned wide variety of frameworks compatible with WebXR. In addition, WebXR's standard is widely implemented in web browsers. A (shortened) compatibility matrix can be observed in Table 2.1. WebXR is the successor to WebVR[8].

[5]`https://immersiveweb.dev/`
[6]`https://needle.tools/`
[7]`https://unity.com/`
[8]`https://webvr.info/`

Feature Name	**Chrome**	**Safari on VisionOS**	**Samsung Internet**	**Meta Quest Browser**	**Microsoft Edge**
Core	Chrome 79	Behind a feature flag	Samsung Internet 12.0	7.0, December 2019	Edge 87 on Windows Desktop, Edge 91 on Hololens 2
AR Module	Chrome for Android, 81		Samsung Internet 12.1	24.0, October 2022	Edge 91, Hololens 2 only
Gamepads Module	Chrome 79		Samsung Internet 12.0	7.1, December 2019	Edge 87 on Windows Desktop, Edge 91 on Hololens 2
Hand Input		Behind a feature flag		15.1, April 2021	Edge 93, Hololens 2 only

Table 2.1: Compatibility matrix for WebXR.

2.1.4 Unity and WebXR Integration Plugin

High-level approaches to developing web-based VR applications include the use of game engines. Major benefits from using such technologies are ease-of-use and thus accelerated development, a development environment that is fitted to the requirements of interactive applications, and the render pipeline being abstracted away from the developer.

Unity[9] is a popular choice in the field of developing VR applications. To use it for web-based VR, Unity can be set up to produce a WebGL software build, which can be exported to already include and make use of the WebXR device API. For this, the Unity WebXR Export Package [10] can be used. Initially developed by Mozilla, the package is now maintained by GitHub user "De-Panther", and provides templates for the Unity build process that already include the WebXR code required to query available XR devices as well as start, maintain, and stop a WebXR session. WebXR sessions generally cover the input and output from and to XR devices. These templates are written in HTML, JavaScript and CSS. Additionally, the WebXR Export Package contains samples as well as an interactions package, in which the use of the C# abstraction of the natively JavaScript WebXR API is showcased. This abstraction allows developers to make use of WebXR functions in Unity's native programming language, C#.

[9]https://unity.com/
[10]https://github.com/De-Panther/unity-webxr-export

Since the majority of research prototype applications we want to provide in our platform were developed in Unity, we choose this engine for the platform, together with the WebXR Integration plugin. Although A-Frame would allow for a simpler structuring of scenes, it cannot be done in a visual editor, but just in the form of code. Furthermore, scenes with higher complexity are easier to implement in Unity.

2.2 Related Resesarch

Research works using WebXR, or web-based VR in general, are not too numerous. In general, the approach is relatively new, with the current trend of making VR hardware accessible to the general audience being fairly new as well. The oldest work on web-based VR we could find dates from 1998 [5]. However, this technique has been worked on more with the recent VR trend, with older works being very rare. Traditionally, VR applications are developed in such a way that they require installation and local execution, whereas with web-based VR, the aim is to deliver an experience that can be executed in the web browser, without the need for installation. While being beneficial in terms of accessibility, this approach however has its own drawbacks such as limitations on the complexity of 3D models in terms of polygon counts and renderable materials. Standalone and desktop VR have been developed to a higher level in these fields.

Learning casting in VR. Sun et al. [17] propose the use of WebGL in their Web3D system ”VR-Casting“, a software solution that aims to educate users about the metal casting industry. The authors present a system that builds upon a HTML5 canvas, onto which the WebGL API renders images. An overview of the proposed architecture can be observed in Figure 2.3.

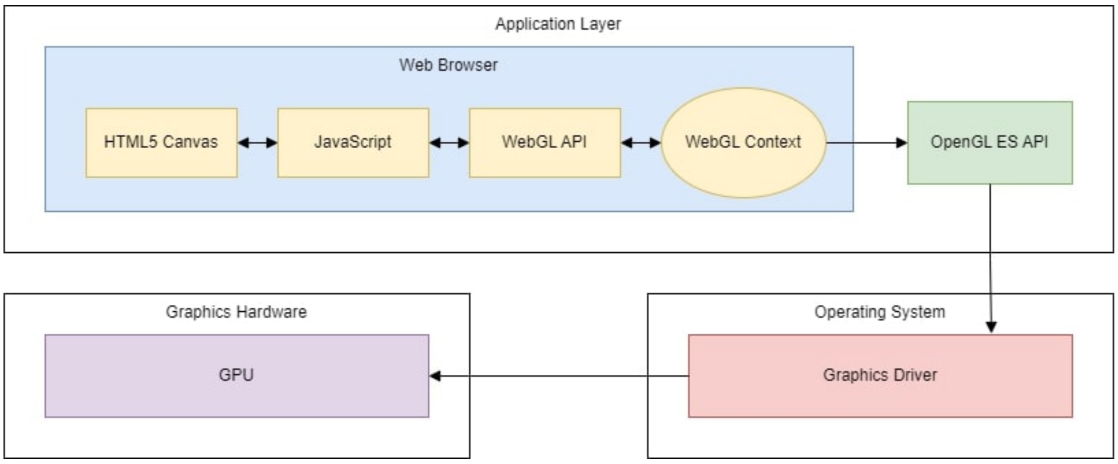


Figure 2.3: Architecture as proposed by Sun et al. [17].

Key technologies used by Sun et al. include “accessing to system software through the internet” while visiting the web site resulting from the “VR-Casting” software, and WebGL as the chosen Web3D technology for rendering 3D scenes in the web browser.

The authors choose the web browser environment, and with it a client-server architecture, to make use of a centralized software core and advantages such as easier maintenance and deployment.

To help with a high amount of high-resolution models, “VR-Casting” employs the level-of-detail approach to limit the bandwidth use of the application and the rendering performance required.

Sun et al. achieve the stereoscopic rendering required for immersive VR by using anaglyph 3D as well as the polarization method. Both these effects are supported by the ThreeJS JavaScript library and used by the authors. Additionally, they offer screen 2D on mobile and desktop devices. However, head or hand tracking is not provided by the application. Instead, tapping and swiping gestures are used on mobile devices with touch screens.

Concluding their work, Sun et al. remark on the accessibility of their application, realized through a web-based implementation. Furthermore, they point out the upgradeability and maintainability, lightweight, cross-platform and portable properties of “VR-Casting”, facilitated by its architecture.

Cognitive Training in VR. Chessa et al. [3] present an exergame approach for “training of cognitive capabilities in elderly population”. Benzing and Schmidt [1] define exergames as “digital games that require bodily movements to play, stimulating an active gaming experience to function as a form of physical activity”, that is, applications that unify gaming and exercise. The exergame by Chessa et al. is in light of the COVID-19 pandemic, which made applications necessary that “can be used remotely at the users’ homes”. Their key assessments are as follows:

1. “[T]he digital autonomy of a group of volunteers who tested the web-based exergame, and”
2. “the potentiality of Unity WebGL build to create non-immersive Virtual Reality environments.”

Chessa et al. propose a virtual supermarket environment in which users should take the necessary steps to buy items. Users are presented with a shopping list which they need to find in three shelves. Clicking on the correct item adds it to the shopping basket. In the final step, users have “to pay the total due amount, by choosing the right banknotes and coins”.

Importantly, this research project builds upon the Unity 3D game engine, using a WebGL build. However, the authors do not further specify to what extent VR technologies are used. Most likely, the application runs in screen 2D.

In their pilot study, Chessa et al. conclude that about 20% of the participants “encountered problems, which did not allow them to complete the task”. They partially found this in the chosen platform, WebGL not fully supporting mobile devices at the time. The authors do not specify the devices the participants used in the experiment. Chessa et al.

further remark that the Unity WebGL build allowed for an easy distribution to the user and good upgradeability. In some cases, technical problems were observed, mostly due to display resolution settings or old web browser versions. Finally, the graphics quality of the WebGL build was assessed with the result that especially the quality of textures was poor, possibly caused by WebGL's rendering limits.

VR for students' mental health. The WebXR device API is often used in web-based VR projects, both commercially and in research. Hossain et al. [6] use WebXR in an application aimed at aiding students' well-being in an attempt to avoid anxiety. This project is again in light of the COVID-19 pandemic. The authors seek to develop a system that offers a self-assessment of anxiety symptoms and then provides support "in a personalized and emotionally engaging manner". In their experiments, Hossain et al. compare three approaches, namely a conventional web site, "a VR immersive environment with a single virtual human playing the role of a student life advisor; and an immersive environment with more than one virtual humans interacting with the user aiming to study which system engages and assist vulnerable students more effectively". Their three prototypes are developed as follows:

1. **The conventional web site** extends the university web page by a questionnaire. 2D video showing "a student life advisor taking students through the questionnaire" accompanies this questionnaire.

2. **The simple virtual environment** uses WebXR and is built in Unity. The user here enters their name and student ID. Then, a dialogue with the "virtual life advisor" begins, in which the self-assessment questionnaire is used. Based on the results, the participant is either provided with information and help offered by the university, or "asks permission from the user to be contacted by the student counselling team", and, if granted, sends the participant's details to that institution.

3. **The multiple virtual humans environment**, too, uses WebXR and is built in Unity as well. However, upon entering the experience, participants receive an explanation of the activity from an NPC. The participant can then freely go and meet another NPC, the "virtual life advisor", with whom they process the questionnaire in a dialogue, again. After this process, two more life advisors are introduced, "[t]he first virtual life advisor offers information about the university's services, while the other gives further knowledge about anxiety.".

Since the study is a work in progress, there are no results included in the paper yet.

Educating about pit mining in VR. Pomykakala et al. [10] present another approach that uses WebXR: They identify the pit mining industry as a field that could benefit from training in a virtual environment, rather than on site. Therefore, they propose an educational application that aims to inform users about the processes that contribute to open-pit mining. The authors begin with a 3D-annotated 360 degree video that shows "the complete transport cycle in the mining plant realized by rigid haul truck". The "characteristic points of the cycle", e.g. maneuvering and loading of the truck, are shown to the user and pointed out by the use of text annotations (see Figure 2.4).

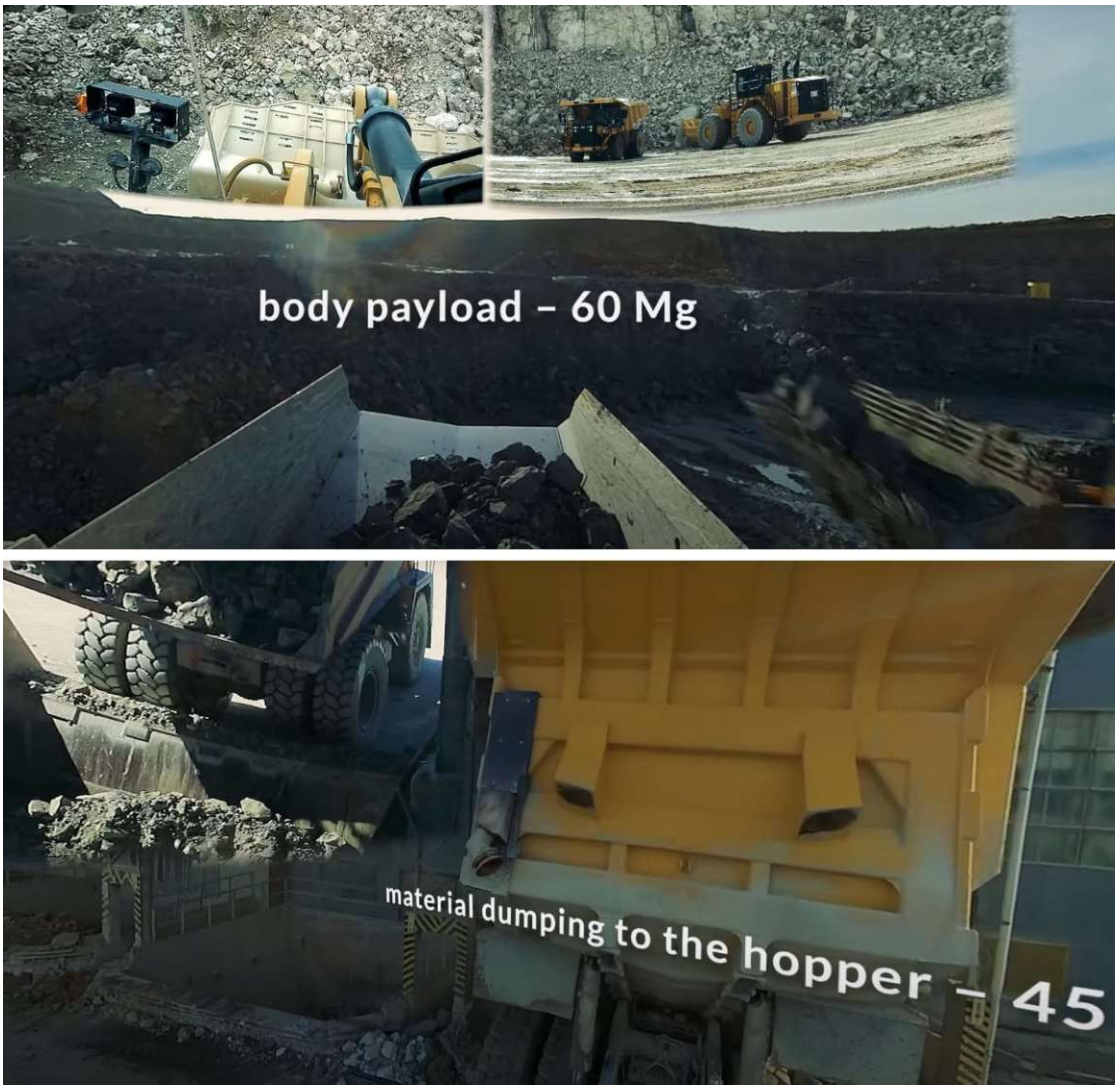


Figure 2.4: Excerpts from videos used by Pomykakala et al. [10].

The Unity game engine is then used to build a terrain model based on geodata of a pit mine and to implement an immersive experience that offers "first-person interactive mockup view and 1:1 scale first-person navigation through the map", as well as quizes and "360 orbs" in which the user can watch the aforementioned documentative videos as well as 360 degree imagery. These orbs are placed at the respective positions within the pit mine terrain their contents correspond to.

The WebXR interface the authors use for their application is "WebXR JavaScript API integration to Unity WebGL", as well as WebXR Export[11] and WebXR Interactions[12]. Since some functionalities are not included in this toolstack by default, Pomykakala et al. also use GyroscopeAccelerometerWebGL [13], Video Player WebGL [14], and ScreenOrientationWebGL [15].

The authors conclude that the use of WebXR, allows for a "wide variety of end-users devices" as well as an easy to use application and a responsive application design, that adapts to client devices seamlessly. Especially in terms of graphics quality and performance, automatic adjustments lead to a wide variety of possible end-user devices. In addition, they point out that due to the use of WebXR, the application automatically adapts to 3 DOF tracking of smartphones and 6 DOF tracking of VR headsets.

Overview of web-based VR applications. A general overview of the state of the art in the field of immersive web experiences is presented by Sathe et al. [13]. According to the authors, VR is "among one of the hottest topics in Computer Science", yet "still in its infancy". Sathe et al. describe the life cycle of a web-based VR application utilizing the WebVR API (WebXR's predecessor) as follows:

1. *"Request for VR Device"*

   Initially, a VR web page should request available VR hardware. On desktop devices, this includes VR headsets, whereas with mobile devices, the device itself, paired with additional equipment (e.g. cardboard VR), is the VR hardware.

2. *"Detecting VR mode"*

   If VR-capable devices are detected, the user should be presented with a button that toggles the VR mode. After enabling VR mode, the application needs to "check whether the device can create a session or not", that is, all sensors work and the device is in functional state.

3. *"Beginning VR session"*

   A VR session is then started. This session manages both the input and output to the display of the VR device. It also provides information received from the device, such as position and orientation.

4. *"Listening to VR instruction"*

   This step includes providing tracking information, as well as matching the application's frame rate with that of the VR device.

[11]https://github.com/De-Panther/unity-webxr-export
[12]https://openupm.com/packages/com.de-panther.webxr-interactions/
[13]https://assetstore.unity.com/packages/tools/integration/gyroscopeaccelerometerwebgl-176394
[14]https://assetstore.unity.com/packages/tools/video/video-player-webgl-192420
[15]https://assetstore.unity.com/packages/tools/camera/screenorientationwebgl-180981

5. *"End VR session"*

   Should the user terminate the application, the VR session also has to end.

Sathe et al. also propose a possible architecture and user-interaction model for an immersive VR web site. For this, they use the example of a web shop, called "SHOP360". Such a web site should at least contain the following elements:

1. *"Introduction page"*

   A standard, 2D web page should serve as the entry point to such an immersive experience. It should provide information and guidelines to the user on how to use the included VR mode. Users should be able to select whether they want to browse items in VR or standard 2D.

2. *"Entering in Virtual Reality mode"*

   The process of entering VR mode, according to Sathe et al., should start with the - in this case - mobile device being inserted into a head mounted display contraption. After this, the user should be presented with a selection of items. The user interface here consists of 'next' and 'previous' buttons that allow switching pages to show more items in the shop.

3. *"Cursor"*

   Interaction with the virtual environment should be made possible by the use of a cursor in the shape of a small ring (see Figure 2.5. All actions in the environment, i.e. "pointing, hovering and clicking", should be performed "using this cursor". The cursor follows the headset's orientation, i.e. it is always in the center of the user's field of view. Moving it over an object is considered hovering, whereas the application registers "[a] long, steady look over the object" as a click.

4. *"Interaction with Objects"*

   Items in the web shop are represented by 3D objects, and each item "has a title above it, which specifies" its name in 3D text. Hovering an item using the aforementioned mechanic leads to it rotating, whereas each item is also accompanied by 'Description' and 'Add to cart' buttons that serve the respective functionalities and can be clicked.

5. *"Description of product"*

   The aforementioned 'Description' buttons leads users to an item's description page. This information is in the form of text and non-interactable.

Sathe et al. conclude that VR "is one of the most up and coming research areas in the field of Computer Science". They assume VR web sites to "revolutionize how interactive we user space can be". In addition, the authors plan the creation of templates, "much like those created by Squarespace", in the form of a Software as a Service model.

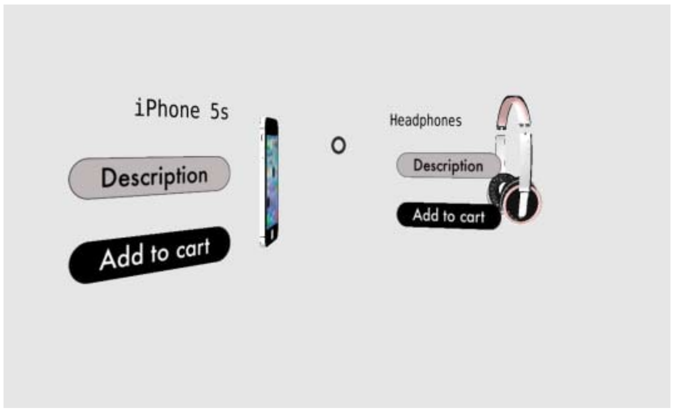


Figure 2.5: Proposed user interaction cursor by Sathe et al. [13].

2.3 Summary

All works presented in the previous section share the use of the WebXR or WebVR Device APIs, or directly rendering using WebGL, to implement VR in the web browser. The authors justify their decision for choosing web-based VR over standalone or desktop VR by a number of reasons:

- **Platform independence:** Sun et al. [17] find platform independence and smartphone compatibility to be important factors when developing an educational platform.
- **Accessibility:** Both Chessa et al. [3] and Hossain et al. [6] argue that, by using a web-based approach, applications become more accessible. This was especially important during lockdowns due to the COVID pandemic, which required remote access to applications developed in these authors' works.
- **Democratization:** Pomykakala et al. [10] reason their choice of the web environment by a democratization of the access and distribution of applications developed in this manner.
- **Simplification:** Finally, multiple works argue that, using a web-based approach, the process of accessing applications becomes simplified and unified, rather than requiring installation and configuration.

All applications discussed in Section 2.2 have their use-cases in similar domains, namely education, training or support. While web-based VR has its drawbacks, e.g. restrictions on model sizes and polygon counts, and the absence of ready tools for developing common techniques for selection, navigation and manipulation, this approach provides many attributes beneficial to our platform. The related research reassures us in our choice of a web-based approach. We aim to address and counteract some of the issues of web-based VR specifically in this thesis.

CHAPTER 3

Design

In this chapter, we explain the main components of the application we want to create in our goal to provide the general public access to demo prototypes created as part of research projects by the TU Wien VR group. For this, we first establish a list of requirements that our web-based VR platform should fulfill. Then, we discuss the means of fulfilling these requirements in the components of the application. In summary, we propose an automatically-expanding, web-based VR platform, which we named Projectverse, that hosts demo VR applications brought forward by research projects. Accompanying this platform, we propose the creation of a developer toolkit aimed at simplifying the adaptation of applications to the demands of Projectverse.

3.1 Requirements

Our objective is to create an application landscape that explains, showcases and provides information about research projects of the TU Wien VR Group. We choose to develop Projectverse in the form of a website, allowing the general public easy access to the information we provide. To achieve our goal of an easy to use and extend web platform, Projectverse has to fulfill the following requirements:

Provide better information about the TU Wien VR Group (R1): The current means of providing information about past and present research projects is in the form of a 2D website. On this website, one cannot get a full impression of immersive projects, since the website itself is not an immersive medium, as most of the project prototype applications are developed in the XR domain, i.e., aimed at VR/AR devices. Projectverse should improve the way users can access information about research projects.

Provide easy access to information (R2): The current website satisfies the requirement of being available to a wide range of users. Projectverse should therefore stick to the principle of information being publicly available on the internet. It should provide a starting point from which users can then decide to learn about research projects and transition into the applications that demonstrate them.

Provide access to research project demo applications (R3): Projectverse should allow users to experience the results of the VR group's research work first hand. This means providing applications to the user that are executed in the medium that they were developed for - VR.

Keep maintenance efforts at a minimum (R4): This requirement mainly focuses on the work required to deploy applications to Projectverse, rather than keeping Projectverse running itself. The introduction of new demo applications to Projectverse should require as few steps as possible, such that it can be populated with little effort, serving as a motivator to researchers to deploy their work on it. As a result of this requirement, Projectverse must automatically extend itself as new applications are introduced.

Visualize deployed applications in a proper way (R5): The underlying data of research projects includes text data in the form of project descriptions, team members working on a research project, and a categorization of projects, for example "XR" or "VR", together forming a network or graph. This data needs to be presented to the user in a way that supports information-gathering as well as being visually attractive. The presentation should also follow general data visualization principles.

Provide assistance to developers wanting to adapt their applications to Projectverse (R6): Since we choose a web-based approach, not all projects will inherently be executable and deployable to Projectverse in their initial state. For example, older applications, that were created without a web-based approach in mind might be built as standalone, executable, applications. Therefore, we aim to provide developers with a toolkit that makes it easy to adapt existing applications to the requirements of Projectverse.

Allow users to interact with each other and exchange about Projectverse as well as projects deployed to it (R7): We aim to create a social experience in which users find themselves in a lobby environment, the Projectverse VR environment, which they can use to talk to each other and discuss the VR group's work. This includes the implementation of multi-player and voice chat.

3.2 Overview

Projectverse Projectverse consists of two main components: The first component is the Projectverse VR environment, which unites multiple components itself. This includes an automatically expanding lobby environment (the Projectverse VR environment), the multiplayer functionality with voice chat and the functionality behind it that provides the user interface with information about research projects. In addition, Projectverse provides means of selecting, and receiving information about research projects, as well as transitioning into the applications resulting from them. We present an overview of our architecture in Figure 3.1: The web server hosts the VR group website together with

Projectverse deployed to it, as well as the JSON file, which specifies the structuring of applications deployed to Projectverse. The front-end, once loaded from the web server, is executed in the browser, and first reads the JSON file and retrieves information about research projects from the website. Next, we calculate the visualization of these projects in the Projectverse VR environment and allow interaction with it. If a user decides to transition into a research project application, it is loaded from the server and presented to the user. Project prototype applications are managed on the server in a folder structure, where each application is in a subfolder of the Projectverse root directory, whereas this root directory contains the Projectverse application itself as well as the configuration file, in which research projects are specified with their contents (main researcher, team, summary) and locations in the root directory.

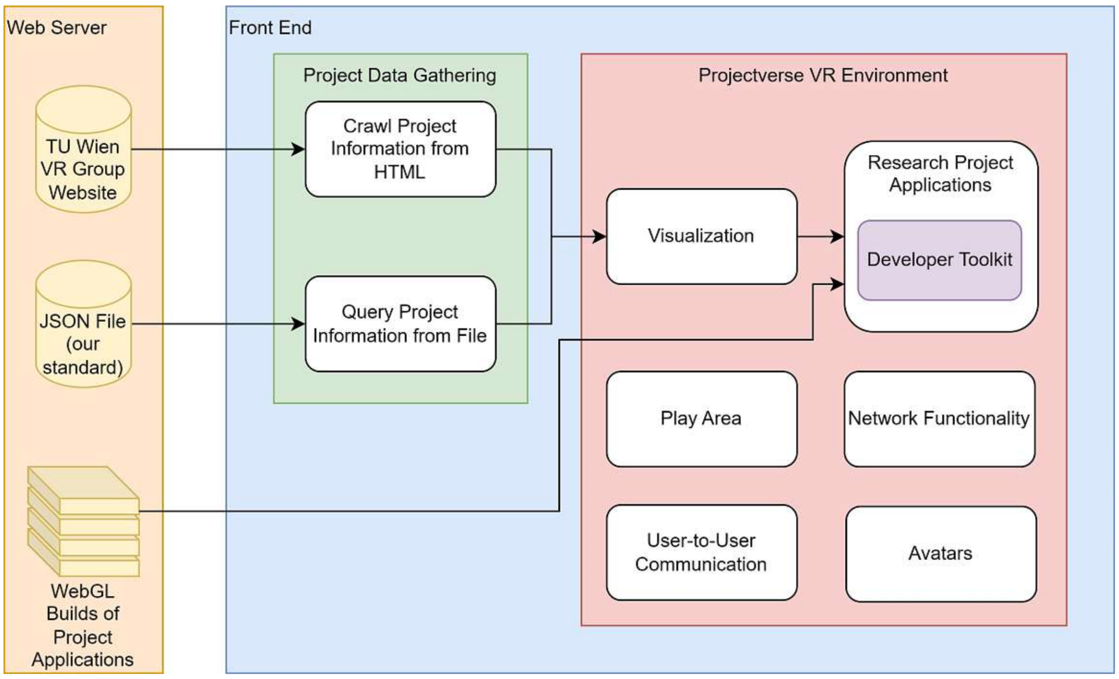

Web Server
TU Wien VR Group Website
JSON File (our standard)
WebGL Builds of Project Applications
Front End
Project Data Gathering
Crawl Project Information from HTML
Query Project Information from File
Projectverse VR Environment
Visualization
Research Project Applications
Developer Toolkit
Play Area
Network Functionality
User-to-User Communication
Avatars

Figure 3.1: Schematic outline of the components of Projectverse.

Developer Toolkit The toolkit as our second main component aims to simplify the adaptation of applications to Projectverse. Its scope includes basic VR functionality such as hardware representation in the development software, standard interactions like grabbing and locomotion techniques, and user interface interaction. The toolkit should provide developers with all the tools required to make existing applications deployable to Projectverse in - at least - a minimally executable state that allows users to explore the environment. Specific applications that introduce e.g. new locomotion methods or highly specific logic will, however, require more adaptation work, also outside of the toolkit's functionality, since the toolkit should cover only basic adaptation tools. An outline of the contents of this toolkit can be seen in Figure 3.2

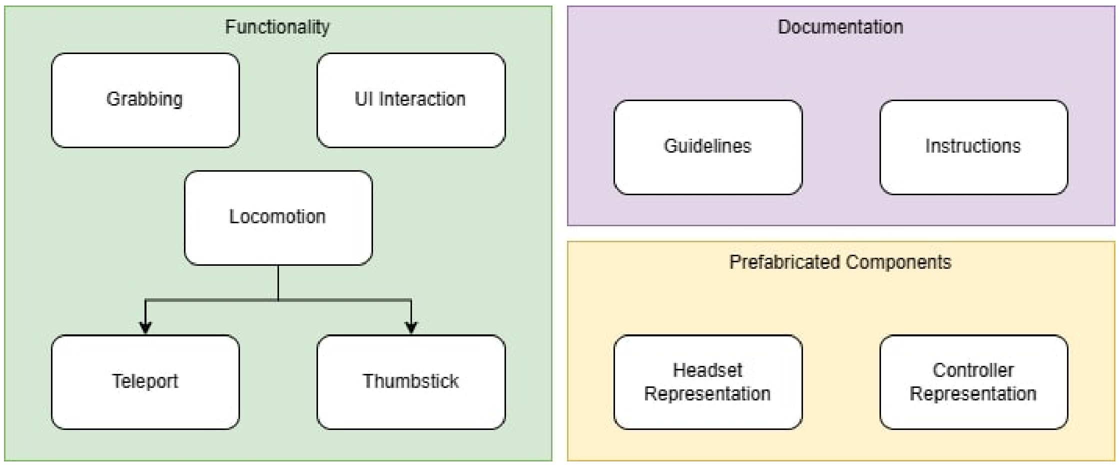


Figure 3.2: Schematic outline of the developer toolkit's contents.

3.3 Projectverse

The Projectverse VR environment serves as the starting point for users who wish to find out about the VR group's research projects. Therefore, it needs to be VR-capable and either self-explanatory or guide users through the experience. A major challenge in the creation of this page is the requirement of an automatically extending environment that can visualize an arbitrary amount of research projects, with a visualization technique allowing up to 50 projects to be visualized through metaphors at the same time.

3.3.1 Server Structure

Projectverse follows a distributed architecture, in which the server provides all information required to construct the VR environment. This information is provided in the form of a directory structure and a file. The file specifies project information, among which are paths to directories on the server in which research project prototypes are stored. The VR environment reads this file and interprets it in the form of a graph, constructing a visualization based on the data in the file. The research project prototypes are part of the server structure so that when transitioning between them, we can load them from the server and present them to the user.

Project Information

We use a JSON file that holds information on all research projects deployed to Projectverse, such as the title, category, summary, and team members. An outline of the data structure used to represent research projects in Projectverse is shown in Figure 3.3. In addition to the data that each project brings, we introduce the application location data item. This specifies under which path the application of a project is located and allows Projectverse to link between the Projectverse VR environment and project applications, that is, let the player transition between them. Finally, we extend our data model by an info URL attribute. This holds another link that specifies where the respective project's web page on the 2D website of the VR group is located. Projectverse then uses that link, and the information stored under it, to replace or fill in the project summary and team members data points should a person deploying an application not provide these. Projectverse accesses this configuration and constructs the visualization based on it.

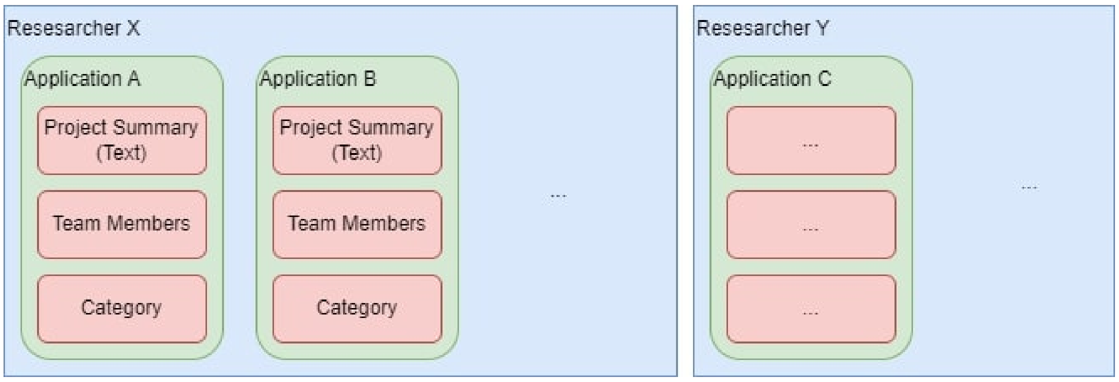


Figure 3.3: Schematic outline of the data scheme used for Projectverse.

3.3.2 VR Environment

We visualize the project graph in the shape of star constellations (R2, R5). This type of visualization allows displaying large graphs on an imaginary sphere around the user, thus reducing cluttering that a “real 3D” visualization technique would produce. We consider our technique to not be “real 3D”, because all star constellations are arranged on this imaginary sphere, rather than freely in space. Each research project is represented by a star, whereas the edges connecting stars form groups, or constellations, where each constellation forms a graph and represents a grouping of research projects by a shared property, e.g. the aforementioned project leader. Each star constellation is labeled with this shared property, whereas each star should also receive a label, denoting the title of the project it corresponds to. A schematic example of our visualization technique is shown in Figure 3.4. By employing this star constellation technique, we introduce a clear layout that is well suitable for VR, since we use the full field of view that HMDs provide us with. In addition, we completely avoid clutter and having to rearrange the visualized graph based on user movements, as is the case e.g. in the work of Sorger et al. [15], where the authors continuously recalculate node positions such that a “bubble” of nodes is formed around the head of the user. Our visualization technique provides an aesthetically pleasing, well-known from nature, view of a graph that makes it easy for users new to VR to comprehend connections between projects and the idea behind their arrangement. It is especially useful for our scenario by being highly scalable to a larger amount of projects, since there is plenty of sky for more stars to be placed.

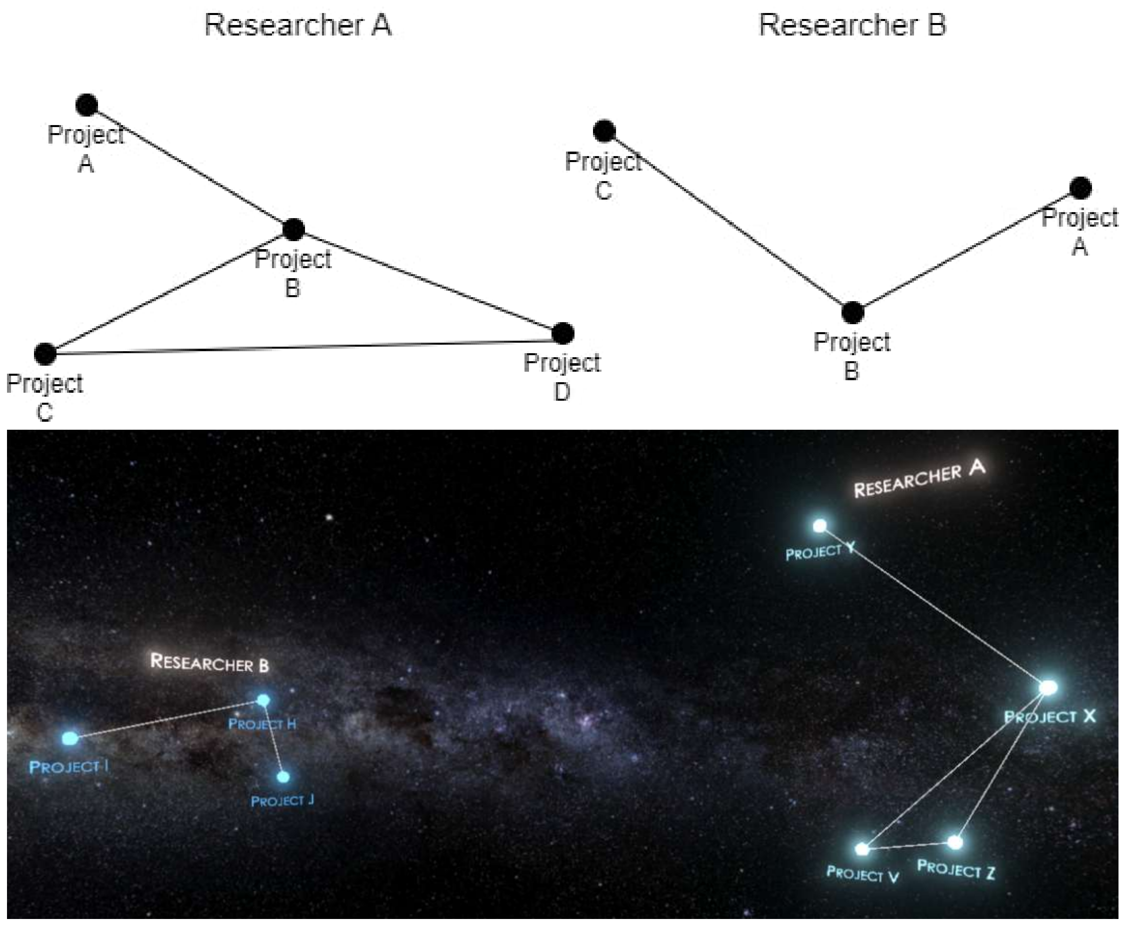


Figure 3.4: Schematic example and realization (1st person game view) of our visualization technique for project graphs.

Project Graph

We interpret project data in the form of a project graphs. At the top level, we have main researchers, who each unite a group of research projects as their work, with each research project having a prototype application and metadata about it. To bring this into the shape of a graph, we assume each project to be represented by a node, and all projects from a common main researcher to be connected by edges, i.e., for each researcher's projects, we construct an individual connected graph. These connected graphs are not connected with each other and altogether form the project graph. This graph construction can also be conducted for other attributes that projects share. Therefore, we also construct a graph that is arranged by categories, instead of main researchers.

Calculating Project Star Constellations

To create a visualization in the shape of star constellations based on the project graph, we need to calculate layouts, because the project data we receive from the JSON file does not contain a graph embedding, just meta data. Our data is annotated with a category per entry (see Figure 3.1) as well as the main researcher. Since our aim is to incorporate both these attributes into the visualization, we calculate two graph layouts: one for groupings by responsible researcher and one for groupings by project category. We label star constellations by their shared property. In the case of groupings of categories, the constellation label represents the category.

The user should be able to switch between grouping options during runtime, therefore we introduce a push-button with lighting. Pressing this button will change the arrangement of the visualization, i.e. the grouping. The transition between two grouping states should be fluid rather than instantaneous, therefore we propose continuous movement of all project stars from their old positions to the new ones upon a push of the button. This will help users understand the change they have introduced into the environment.

The calculation of both drawings is performed in coordinates, and we place all stars on an imaginary sphere around the center of the scene, which lies at the middle of our balcony play area. In short, we want to calculate a regular grid in the shape of degrees of latitude and longitude on this sphere, and then place the stars that represent our projects on this grid. A schematic of the grid is shown in Figure 3.5, where the areas valid for placing stars are in green, and the exclusion zones are in red.

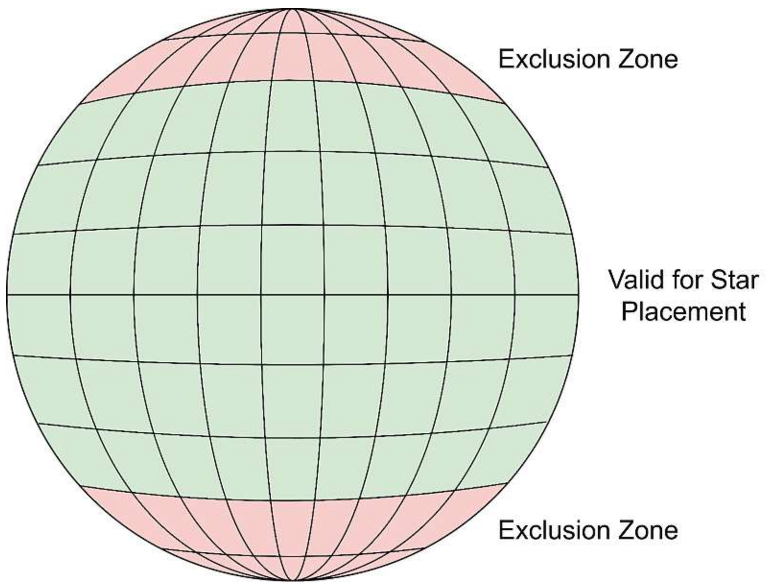


Figure 3.5: Schematic of our coordinate system for star placement calculation.

For the purpose of placing the stars, we first create a dictionary which contains all projects and either the main researcher or the category serves as the key for each list of projects. We proceed by finding the longest list of projects in this dictionary to determine the number of rows of our grid. The number of columns, on the other hand, is determined by the number of keys in the dictionary, i.e., the number of categories of main researchers. We proceed by determining the horizontal spacing, i.e., the distance between longitudinal grid lines, which we define as

$$C_{external} = \frac{360}{n_{c_{external}}}.$$

Here, $n_{C_{external}}$ is the number of "columns" in the grid, i.e., number of categories or main researchers. To properly separate our star constellations in the space, we increase the number of longitudinal grid segments by three, thus

$$c_{external} = \frac{360}{3n_{c_{external}}},$$

such that a star constellation is only created on the first of each three grid segments, whereas the other two remain empty to ensure proper spacing.

We then iterate over all keys in our dictionary of projects and determine the amount of rows, or latitudinal grid segments for each star constellation. This, we define as

$$n_{rows} = \lceil \sqrt{n_{apps}} \rceil.$$

n_{apps} denotes the number of one category's, or main researcher's, projects. We use this formula because we aim for a square-shaped arrangement of the stars, hence the side length of this square is given by the square root. We do not calculate an "exterior" number of rows, because each column segment of our sphere should hold just one star constellation. Furthermore, we define exclusion zones, in which no star should ever be placed, because they are far outside the normal field of view. These exclusion zones range from 45 to 90 degrees northern and southern latitude and are shown in Figure 3.5. Integrating these zones into our calculation of the constellation-internal vertical spacing, we get

$$r_{internal} = \frac{180 - 90}{n_{rows} + 1}.$$

Furthermore, we define the number of grid columns within a star constellation as

$$n_{c_{internal}} = n_r.$$

We then calculate the first grid location x_{start} and the internal column spacing $c_{internal}$ for the current star constellation as

$$x_{start} = \frac{n_{c_{internal}} c_{external}}{2}$$
$$c_{internal} = c_{external} / n_{c_{internal}}$$

Finally, within the iteration of our dictionary, we iterate the list of projects, and determine the position in space each one receives as a star. Here, the horizontal angle is given as

$$x_{app} = JC_{external} - x_{start} + j * c_{internal}$$

where J denotes the external column we are in, i.e., the positioning of the entire star constellation on the grid. j, on the other hand, denotes the internal column we want to occupy, i.e., the position of the star within the constellation's grid. The vertical position of a star is calculated similarly as

$$y_{app} = 45 + (i+1)r_{internal},$$

where i serves as the index for the row we are currently placing stars on.

Since star constellations in nature never come arranged in regular grids, we proceed by shuffling the positions of our stars:

$$x_{shuffled} = x_{app} + tr_{internal}, t \in [-0.5, 0.5]$$
$$y_{shuffled} = y_{app} + tc_{internal}, t \in [-0.5, 0.5]$$

Here, t is a random real number generated newly for each individual shuffling. The star positioning algorithm is displayed in Algorithm 3.1.

With our star positions now calculated, we transform them from spherical to cartesian coordinates and determine the connections between them.

For this, we arrange the project objects in a list that resembles their hierarchical order in the JSON file, i.e., we generate a list of lists of projects, where each bottom-level list holds one researcher's project, and the top-level list holds all these lists. For each bottom-level list, we employ Prim's Algorithm [11] to find a minimum spanning tree.

The goal of this algorithm is to find a minimum spanning tree, and to then add another edge to that spanning tree, over each star constellation. For this, we initially assume each star constellation to be a fully connected graph, i.e., there is an edge between each two stars in a star constellation, but no edges exist between any two stars of two different star constellations. We begin by iterating over all star constellations. Within each iteration, we then execute Prim's algorithm, so we initiate three lists of projects, one holding all visited vertices (or stars), one holding all unvisited vertices, and one holding the vertices of the minimum spanning tree in order, i.e., we arrange vertices in pairs in this list, such

Algorithm 3.1: Algorithm for the star placement.

```
Input  : Dictionary $SortedApps$ of strings to lists of projects
Output : List of lists of projects $allApps$
$n_{c_{external}} \leftarrow$ Count of keys in $SortedApps$;
$rows \leftarrow 0$;
for each key-value pair in SortedApps do
    if Count of values > rows then
        $rows \leftarrow$ Count of values;
    end
end
$C_{ext} \leftarrow \frac{360}{columns}$;
$c_{ext} \leftarrow \frac{n_{c_{external}}}{3}$;
$J \leftarrow 0$;
$allApps \leftarrow$ Empty list of lists of projects;
for each key-value pair in SortedApps do
    $group \leftarrow$ Values in key-value pair;
    $n_{rows} \leftarrow \lceil \sqrt{Countofgroup} \rceil$;
    $r_{internal} \leftarrow \frac{180-45-45}{n_{rows}+1}$;
    $n_{c_{internal}} \leftarrow n_r$;
    $x_start \leftarrow \frac{n_{c_{external}} \cdot c_{internal}}{2}$;
    $c_{internal} \leftarrow \frac{c_{external}}{n_{c_{internal}}}$;
    $appsList \leftarrow$ Empty list of projects;
    $i \leftarrow 0$;
    $j \leftarrow 0$;
    for each app in group do
        $y_{app} \leftarrow 45 + (i+1) \cdot r_{internal}$;
        $x_{app} \leftarrow J \cdot C_{external} - x_{start} + j \cdot c_internal$;
        $y_{shuffled} \leftarrow verticalAngle + Random.value \cdot r_{internal}$;
        $x_{shuffled} \leftarrow horizontalAngle + Random.value \cdot c_{internal}$;
        $pos_{star} \leftarrow \text{transformCartesian}(y_{shuffled}, x_{shuffled})$;
        $app.Position \leftarrow star$;
        $appsList.\text{Add}(app)$;
        $j \leftarrow j + 1$;
        if $i$ mod $J$ == 0 then
            $i \leftarrow i + 1$;
            $j \leftarrow 0$;
        end
    end
    $J \leftarrow J + 1$;
    $allApps.\text{Add}(appsList)$;
end
return $allApps$;
```

that if $t_i, t_{i+1} \in T$, where T is the spanning tree list, $e(t_i, t_{i+1}$ is an edge of the minimum spanning tree. We start at the first star, i.e.

$$s = S_0,$$

where S is the list of unvisited vertices. We add this star to the list of visited vertices, C and remove it from S. Next, we iterate over all vertices in C, and within each iteration, over all vertices in S. This iteration finishes when the minimum distance between two stars, or vertices, $s_i \in S$ and $c_j \in C$ is found. We now add both vertices we found to T and remove s_i from S, the list of vertices remaining to be added to the spanning tree.

Next, we iterate over T and for each vertex $t_k \in T$ retrieve the position and write it into a new list E of vectors to later serve a position for our edges. The important thing here is that we don't iterate over pairs of vertices, i.e., t_k and t_{k+1}, and increment k by 2 after each iteration, but rather just make increments of 1. Since the order of vertices in T is of such nature that every other edge $e(t_k, t_{k+1})$ is a loop, i.e., we have a list that looks like

$$E_1 = \{0, 1, 1, 3, 3, 4, 4, 2\},$$

we mostly introduce self-loops in our graph through this method. However, in some occasions, Prim's algorithm generates us a list that might look like

$$E_2 = \{0, 3, 3, 4, 0, 2, 2, 1\},$$

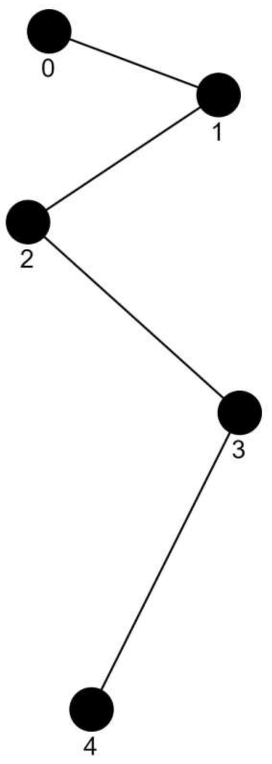
which, by our method of displaying edges, will result in a loop that, when rendered with straight lines as edges, will very rarely cause intersections between these lines. This is also explained in the way we position the stars in order. We therefore utilize this property of our method to introduce loops into our star constellations to make them look more natural compared to just tree-like visualizations. The drawings of both examples for E are shown in Figure 3.6. The entire constellation generation functionality can be observed in Algorithm 3.2.

To provide annotations for our visualization, we want to label each star and each star constellation. In the case of a constellation label, we first find the centroid $c(x, z)$ in the x-z-plane of all stars in this constellation. Next, we calculate the vertical extent of the constellation, i.e., the maximal y-value y_{max} among its stars. We then construct a new point $p(x, y_{max}, z)$ from these coordinates. Finally, we add minor vertical offset to prevent the label from being placed exactly where the constellation's highest star is, and thus shift it a bit towards positive y. This calculation is performed for all star constellations.

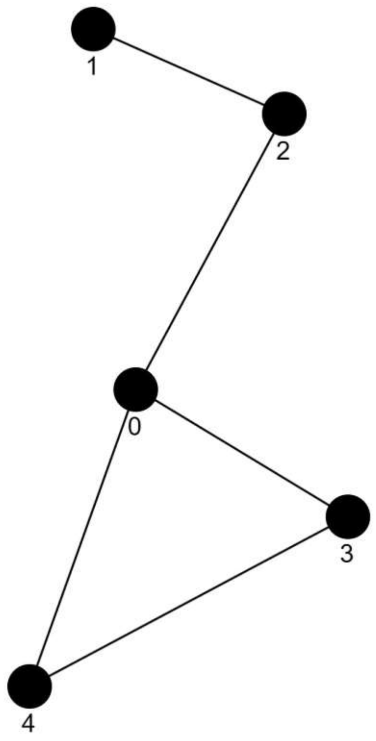
The label positions for the individual stars are calculated by simply adding a small vertical offset to each star's position, thus shifting the label towards negative y, and preventing overlap.

Algorithm 3.2: Algorithm for creating star constellations.

```
Data: StarConstellations, buildType
Result: E
E ← empty list of Vector3 arrays;
for starList in StarConstellations do
    C ← empty list of projects;
    S ← copy of starList;
    s ← $S_0$;
    C.add(s);
    S.remove(s);
    T ← empty list of projects;
    while S is not empty do
        $d_{min} \leftarrow +\infty$;
        for c in C do
            for s in S do
                $d_{sc} \leftarrow \|position_c - position_s\|$;
                if $d_{sc} < d_{min}$ then
                    $a_{min} \leftarrow$ c;
                    $b_{min} \leftarrow$ s;
                    $d_{min} \leftarrow d_{sc}$;
                end
            end
        end
        if $a_min \neq null$ and $b_min \neq null$ then
            T.append($a_{min}$);
            T.append($b_{min}$);
            C.append($b_{min}$);
            S.remove($b_{min}$);
        end
        else
            break;
        end
    end
    for $k \leftarrow 0$ to T.length - 1 do
        a ← $position_{T[i]}$;
        b ← $position_{T[i+1]}$;
        E.append(a, b);
    end
end
```



(a) E_1.



(b) E_2.

Figure 3.6: Examples for edge drawings. Self-loops are not shown.

Play Area

With the star constellations arranged on a spherical sky, we place the users in the center of this sphere. By providing a room scale, balcony-type platform with transparent handrails and floor, we ensure both that users do not feel "lost" in the large environment and have good visibility in the star constellations. In the center of the platform, there is a table-like structure with a button, from which the visualization can be controlled, i.e. changes in the arrangement of star constellations can be made. Also, this table displays written information about how to use Projectverse.

Networking and User Avatars

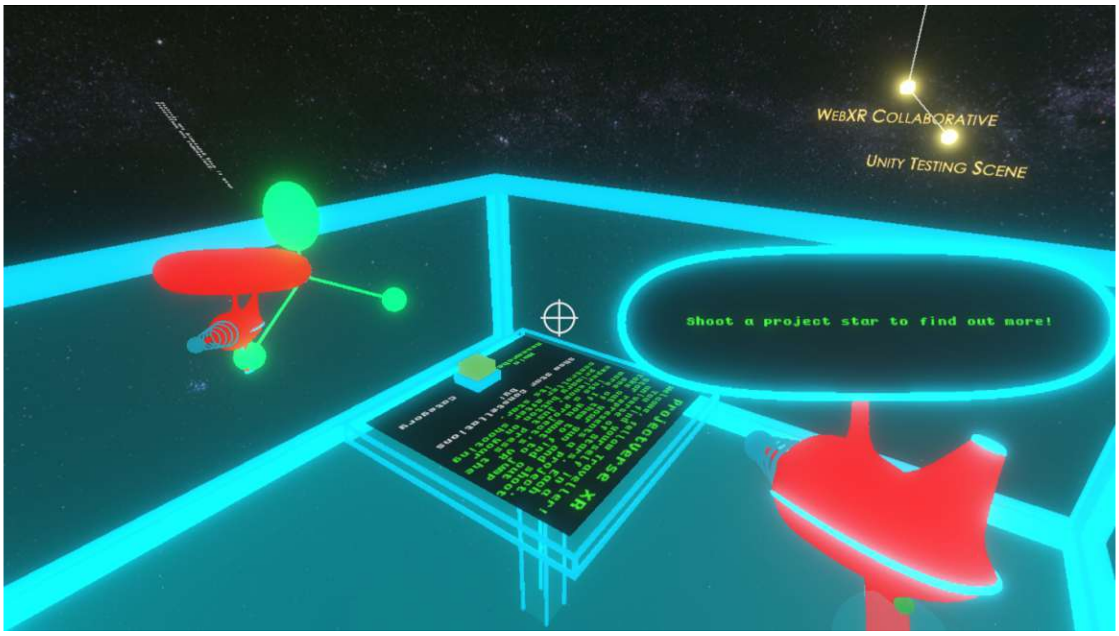


Figure 3.7: Another user's avatar (in green) as seen from a user's point of view.

Since the VR environment of Projectverse should also serve as a social interaction and exchange platform, we provide user avatars. According to the science fiction art style of Projectverse, these avatars should also resemble star constellations. We enable this by representing each user's hands with two small spheres (stars) and the head with another larger sphere. These are then connected to form a constellation. Additionally, we implement proximity voice chat, enabling users standing close together to communicate their thoughts about projects, etc. (R7). Our avatar model is shown in Figure 3.7.

Whenever the player count in our VR environment increases from 0 to 1, i.e., a new game session begins, Projectverse reads the configuration files and constructs the project graph. Furthermore, at this point, Projectverse calculates the graph embedding, i.e., the star constellations. We refer to the first player to join the VR environment as master player. Whenever a new player joins the master player, the project graph is not recalculated but automatically loaded such that the star constellations are already present without any calculations. This is because we instantiate all objects part of the graph visualization in such a way that they are automatically synchronized for the entire session. The flow of actions in networking is shown in Figure 3.8.

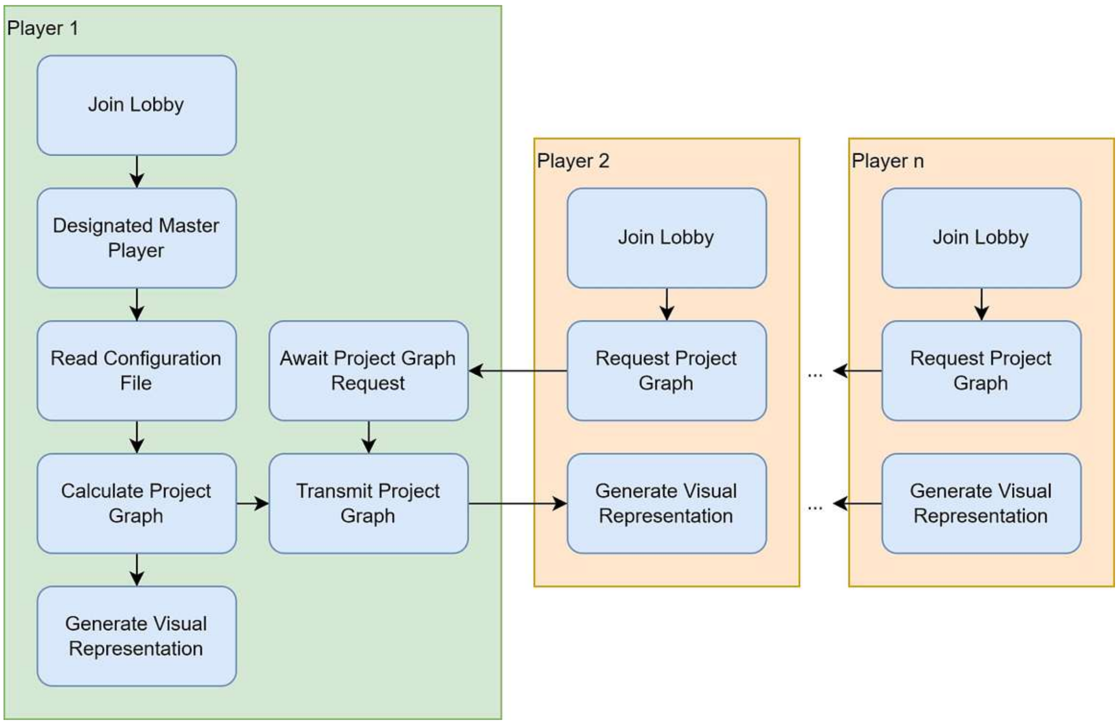
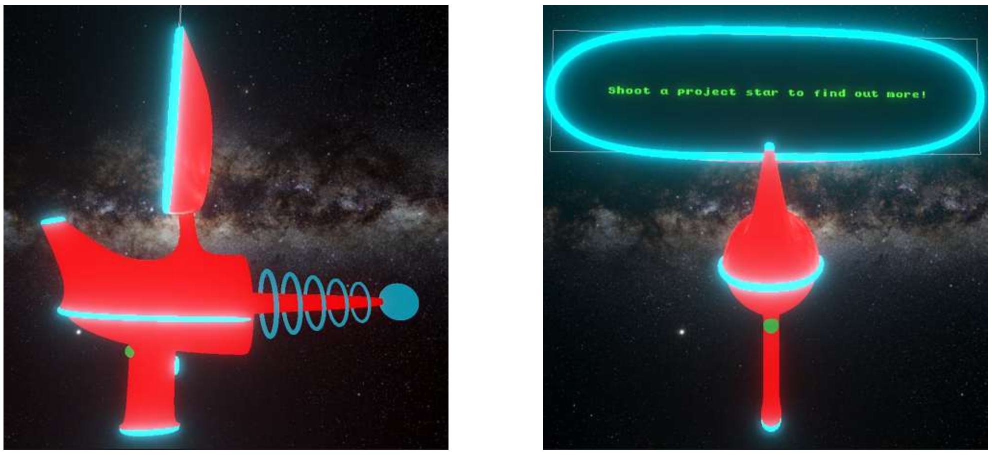


Figure 3.8: Flowchart of the graph calculation and distribution process.

User Interactions



(a) Side. (b) Back.

Figure 3.9: The laser gun interaction tool as seen in the editor view.

Interaction with the environment should be enabled using a well-known tool that suits the art style of Projectverse. We therefore choose to employ a laser gun, fitted with a screen, with which users can shoot research project stars. Information about the project that was shot is then displayed on the gun's screen. This should include both a short summary of the project's contents (as given in the data as Project Summary, Figure 3.3) and information about the researchers who participated in the project (as given in the data as Team Members, Figure 3.3. The information panel can be scrolled using the controller. Finally, by pressing a specific button while holding this laser gun, users can transition into the research project they just shot.The laser gun can be observed in Figure 3.9. The green flashing light on the gun should indicate that Projectverse is ready to transition to a project application. This flashing occurs after selecting, i.e. shooting, a project star. The transition process to the demo application of a research project should be transparent and noticeable to the user. Therefore, we use both sound and visual effects to convey this to the user. The visual effects include an "interstellar jump-like" animation, as shown in Figure 3.10.

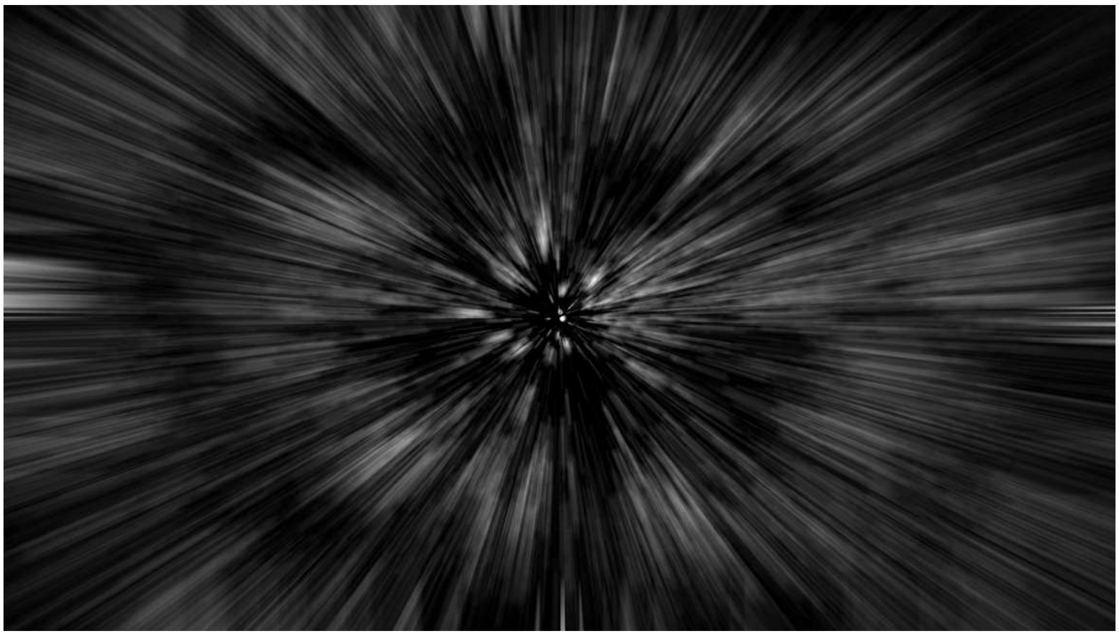


Figure 3.10: Still image of the transition visual effect.

Transitioning into Project Applications

Transitioning into Projects After the push of a button on the controller and, subsequently, the transition animation, users are brought "into" the application whose star they shot with the laser gun. They are then able to at least use a simplified, if not complete, version of the application. For applications that were developed for WebXR in full, complete functionality is available without any adaptations to gameplay, whereas applications that were built initially for Windows require adaptations to make them suitable for the web-environment. Therefore, these applications might not cover the full functionality as intended, depending on the amount of work put into their adaptation.

A minimal subset of functionality for all project prototype applications should include moving around in them in order to explore what the respective projects are about. Aside from applications developed in Unity, Projectverse can host and connect to all other VR-ready types of applications that can be executed in the browser, e.g. applications that were developed in A-Frame.

While in a project application, users leave the networked Projectverse VR environment. Therefore, their avatar gets deleted from the scene and they are no longer able to communicate with other players in the main application.

Coming back Having experienced a project for long enough, users are given a way of navigating back to the entry page, using a combination of controller buttons that is equal among all project applications they experience (R3). Upon pressing these buttons, the user leaves the project application and transitions back to the main VR environment of Projectverse, where they again join a networked sessions with others, if one is present. Otherwise, a new session is created again.

Web Interface

Since we want Projectverse to be available to a broad audience (R2), as well as stick to the current means of communicating the VR group's work (R1), which is via the Internet, we propose the creation of a web page that serves as the interface to Projectverse for the user. This web page should provide the following functionalities:

- **User Manual:** In order to accomplish good user experience, the web page should provide information about how to use Projectverse, its purpose, and ways of interaction. Furthermore, the page should contain schematics of commonly used VR controllers labeled with the actions triggered by pressing the controls on them.
- **Entry Point to Projectverse:** The VR environment of Projectverse should be a central part of the web page. Along with information about it, Projectverse should be playable on the page.
- **Means of returning to the Entry Page:** Once a users transitions into a project application, they are able to return to the entry page via a combination of VR controller buttons. Alongside this method, we propose to also provide a means of returning to the classical 2D interface of the web page. This should be realised in the form of a button on the page.
- **Provide legal information:** Since some of the assets used in the creation of Projectverse are protected by copyright, we present licensing information here.

The page itself should be part of the VR group's website, such that users who wish to inform themselves easily find it alongside the two-dimensional web site.

3.4 Developer Toolkit

As described in R6, we want to provide assistance to developers in adapting their research project applications to Projectverse. This requirement aims to increase attractiveness of Projectverse for researchers to deploy projects on it, i.e., increase the amount of content hosted. Besides that, since the steps required to adapt an application to the web environment are highly similar among many projects available for the VR Group - as they are developed in Unity and built for the Windows platform - this work can be partially automated relatively easy.

Therefore, we propose the development of a developer toolkit that comes with Projectverse and provides functionality common to the XR spectrum as well as documentation that discusses the steps necessary to make an application executable and compatible with Projectverse. Here, we distinguish between two groups of developers, i.e., two different types of application states from which we start the adaptation. Both starting points assume a Unity application, for which not only the built version but also the project files are available.

- **Application developed for the Windows platform:** In this case, the application was previously targeted to run on Windows machines and needs significant changes so it can be executed in the web browser. This includes changing the XR device API used as well as the possibility of library or plugins incompatibility.
- **Application developed for the Web environment:** In case of an application that can already be executed in the web browser, only minor adjustments are necessary. We assume that WebXR is already in use for communication with XR devices, and that no incompatibilities are present. Therefore, only the functionality for users to navigate back out of the application and into the Projectverse VR environment needs to be added here in order to implement full compatibility with Projectverse.

As discussed and depicted in Section 3.2 and Figure 3.2, we aim to provide a range of tools, functionality and assets with the developer toolkit. Among these are the following:

3.4.1 Assets

Assets we provide should include prefabricated components (prefabs) for representing the VR headset and controllers in an application. In order to work with WebXR, these components need to communicate using this driver, i.e., WebXR should propagate the user's movement of the respective devices to the prefab components. Additionally, inputs, e.g. from buttons, should also be mapped and propagated. Finally, the prefabs also need to visually represent the devices. This means, that the controllers should be visualized e.g. through spheres or controller models, whereas the headset should represented by a further sphere or a VR headset model.

3.4.2 Functionality

The functionality provided by the toolkit should enable developers to easily recreate the gameplay logic of their application. This is especially important with applications developed for the Windows platform: Due to incompatibilities, the entire XR device representation and functionality might have to be removed and replaced as part of the adaptation process. We therefore aim to provide standard XR user interactions, which include logic for grabbing and releasing objects, interacting with UI elements by pressing UI buttons and scrolling UI elements as well as providing means of transportation, or locomotion.

The standard locomotion types we want to support are thumbstick movement and teleportation. Here, thumbstick movement means moving the player continuously as commanded by the push of e.g. the Oculus Touch thumbstick or inputs on the HTC Vive Controller touchpad, whereas teleportation involves the selection of a teleportation target by a user input, and subsequently, moving the player to the selected location in a discrete step. All functionality we want to provide should be as configurable as possible. This requirement implies configurability of button mappings and parameters inherent to the respective functionality, e.g. whether grabbed objects should be dropped or thrown on release etc.. Finally, our functionality must include a way of bringing the user back from a project application to the Projectverse VR environment.

3.4.3 Documentation

Our documentation should include comprehensive instructions on all steps required to adapt both application types discussed above. This includes information on additional software installations required for the web environment as well as the removal of components that might be incompatible with the web browser. In addition, we want to cover how the functionality and assets described above are intended to be used and how they could be adapted and extended. The documentation also needs to cover how to eventually build for the web environment. Aside from this, we want to provide development guidelines that allow for applications to follow the controls layout of the Projectverse VR environment, so that users do not have to learn a special control layout for each application, resulting in better usability of Projectverse. Finally, the documentation needs to include information on how to deploy an adapted application to Projectverse as well as troubleshooting hints.

CHAPTER 4

Software Implementation

This chapter explains how Projectverse is implemented. We begin by covering the server structure and the configuration file, then we discuss the implementation of the VR platform of Projectverse. Finally, we explain how the components of our developer toolkit are created.

We implement Projectverse in Unity. Furthermore, we use the WebXR device API to communicate with VR hardware together with the WebXR Unity Plugin which allows us to use data from the WebXR Device API within the game engine. For the networking implementation, we use functionality provided by Photon for Unity. To make Projectverse executable in the browser, we build it to WebGL.

The application is part of the VR Group website, which is a wordpress site hosted on a Linux Server that runs Apache. For the Configuration file, we make use of the JSON standard. The skybox we use in Projectverse is a free asset from the Unity Asset Store[1], whereas the sounds we use when the star constellations are rearranged[2] and when a transition takes place [3] come from the freesound website.

4.1 Projectverse Implementation

This section covers the implementation of the entire Projectverse platform. From the web server and projects hosted on it together with the configuration file, to how the VR environment generates star constellation and establishes networked sessions, we explain in detail how the platform is realized.

[1]https://assetstore.unity.com/packages/2d/textures-materials/milky-way-skybox-94001
[2]https://freesound.org/people/komit.wav/sounds/402295/
[3]https://freesound.org/people/kaboose102/sounds/340256/

4.1.1 Server Structure

The server structure of Projectverse consists of a system of directories and files, all placed on a web server. In the root directory, we host the VR environment application as well as our configuration file. The subdirectories contain the builds of project prototype applications.

Configuration File

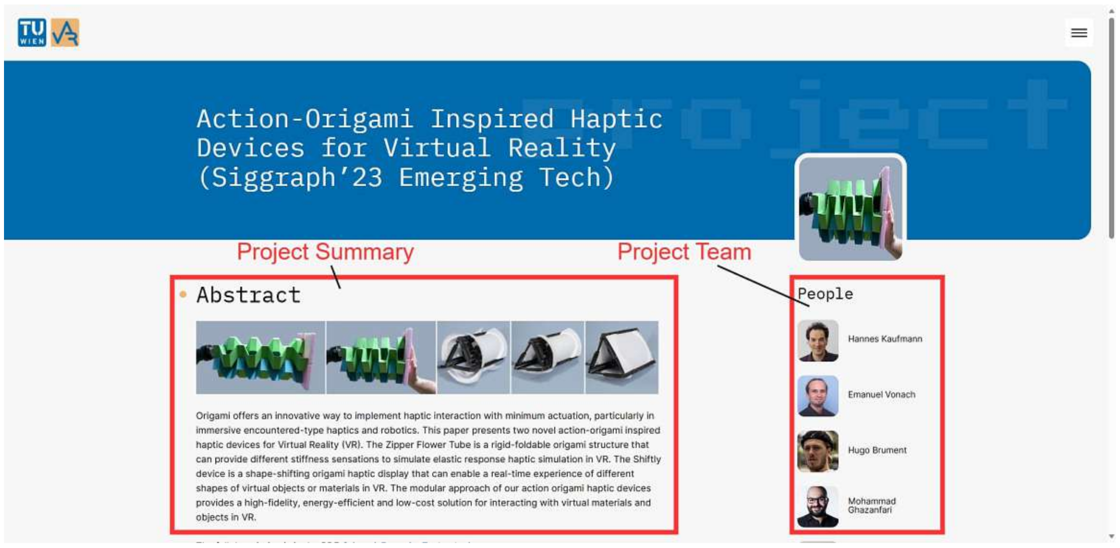
We formulate research project data in a JSON file that holds all information displayed in Figure 4.13. An overview of our data standard for this file can be observed in Listing 4.1. With this standard, we map all data relevant to a research project to the respective parameters in Projectverse's main JSON configuration file. Here, we introduce project groups. Each group is "owned" by a researcher responsible for it. This should be the main researcher, or leader, of the respective research project. One researcher can have multiple projects they are "responsible" for. Further attributes we include in this file are the project's name, its category and a short summary of its contents as well as an array of team members. Furthermore, we introduce the attribute `Location`: This is read by scripts of Projectverse and interpreted as a URL. The project's application build should be uploaded at the web page it represents. Projectverse will then be able to allow players to transition and experience that application. Another parameter relevant to the functionality or Projectverse is `GetDataFromJson`: It determines whether the project summary and team members data should be read from the JSON file or retrieved from the VR group's website. If it is set to `true`, the respective attributes also need to be present in the file. In case of `false`, the `InfoUrl` parameter must be present. This is again interpreted as a URL and should link to the web page that contains information about the respective research project. Projectverse then automatically retrieves team members and project summary information from there.

Retrieving Project Information from the VR Group Website

In the case of an application that has the `GetDataFromJson` attribute set to `false` in our configuration file, we retrieve the project summary and the name of the research team members from the website of the VR group. This must happen before we write the information to the star annotations. Therefore, we introduce the `LoadHtml` function, which requests the HTML page from the VR group's web server that is located at the URL contained in the application's `InfoUrl` field in the JSON configuration file. We use a simple `UnityWebRequest` for this process and proceed by parsing the retrieved HTML file using functionality provided by the `HtmlAgilityPack` library.

```
{
    "Groups": [
        {
            "Responsible": "Researcher A",
            "Applications": [
                {
                    "Name": "Research Project X",
                    "Location": "./research-project-x",
                    "Category": "Student Projects",
                    "GetDataFromJson": true,
                    "ProjectSummary": "Project Summary",
                    "TeamMembers": [
                        "Jane Doe",
             "Jon Doe"
                    ]
                }
            ]
        },
        {
            "Responsible": "Researcher B",
            "Applications": [
                {
                    "Name": "Research Project Y",
                    "Location": "./research-project-y",
                    "Category": "VR Projects",
                    "InfoUrl": "../projects/project-y",
                    "GetDataFromJson": false
                }
            ]
        }
    ]
}
```

Listing 4.1: Data standard for our JSON configuration file.



Action-Origami Inspired Haptic
Devices for Virtual Reality
(Siggraph'23 Emerging Tech)

Abstract

Origami offers an innovative way to implement haptic interaction with minimum actuation, particularly in immersive encountered-type haptics and robotics. This paper presents two novel action-origami inspired haptic devices for Virtual Reality (VR). The Zipper Flower Tube is a rigid-foldable origami structure that can provide different stiffness sensations to simulate elastic response haptic simulation in VR. The Shiftly device is a shape-shifting origami haptic display that can enable a real-time experience of different shapes of virtual objects or materials in VR. The modular approach of our action origami haptic devices provides a high-fidelity, energy-efficient and low-cost solution for interacting with virtual materials and objects in VR.

People

Hannes Kaufmann

Emanuel Vonach

Hugo Brument

Mohammad Ghazanfari

Figure 4.1: A research project info web page.

Since all project web pages on the VR group website have a very similar structure (see an example in Figure 4.1), we can simply query for the text we are looking for - project summary and team members - using the `SelectSingleNode` function of `HtmlAgilityPack` to first find the header of the project summary, which is always a `<h2>` element containing "Abstract":

```
HtmlNode aboutHeader = htmlDoc.DocumentNode.SelectSingleNode("
    //h2[contains(text(), 'Abstract')]");
```

We then select the HTML nodes, in this case, `<p>` elements, following our header, in which the project summary is written:

```
HtmlNodeCollection paragraphElements = aboutHeader.SelectNodes(
    ".//following-sibling::node()");
```

Finally, we iterate over all found `<p>` elements and join the text they hold together. Since Unity's text renderer cannot display non-breaking spaces correctly, we also filter these out.

Similarly, to retrieve information about the project team members, we query a `<section>` element that holds the `id` "people":

```
HtmlNode peopleSection = htmlDoc.DocumentNode.SelectSingleNode(
    "//section[@id='people']");
```

Having found this element, we select all nodes within the section that have the `class` "name". These elements contain the names of the project team's members that we need:

```
HtmlNodeCollection nameElements = peopleSection.SelectNodes("
    .//p[@class='name']");
```

Again, we finish by joining all found text together.

Having retrieved all the information relevant to a project, we save it to use once the star constellations have been generated.

4.1.2 VR Platform

The VR environment serves as the starting point of Projectverse. It provides the star constellation visualization of research project applications, information on research projects, and means of interacting with the environment. The components of the VR environment are divided into static and dynamic elements. The static elements include a simple, balcony-like structure that serves as the play area, and a small table, on which we display information about how to use Projectverse, as well as a button that provides functionality to configure the visualization as discussed below. Furthermore, we include a skybox that shows the stars of the milky way to match the art style dictated by our visualization technique, the star constellations. Together, these elements form the play area. The balcony and table structures can be observed in Figures 4.2 and 4.3. All player movements are restricted to this platform.

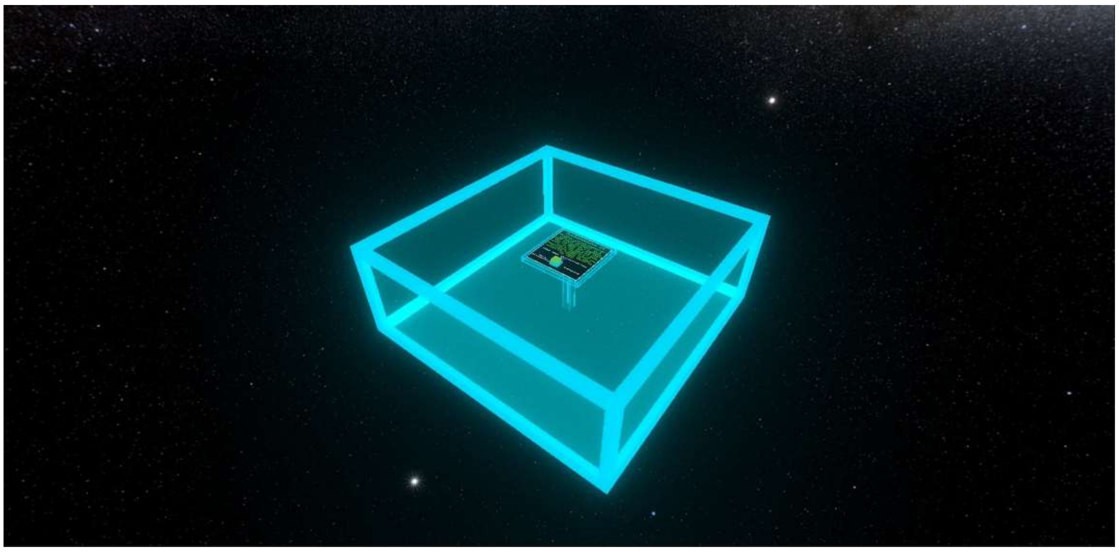
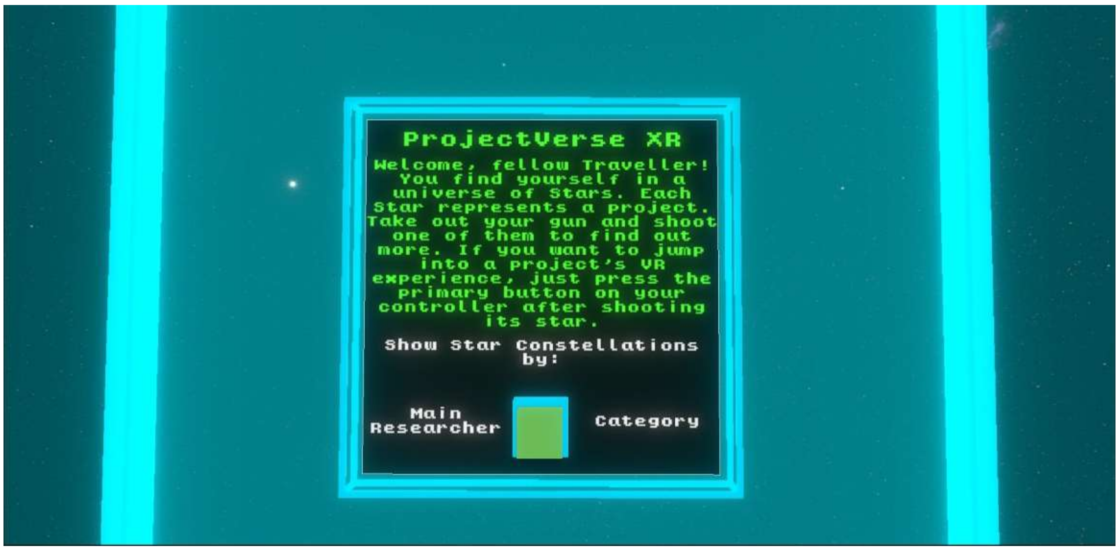


Figure 4.2: The balcony play area. Editor view.


ProjectVerse XR

Welcome, fellow Traveller! You find yourself in a universe of Stars. Each Star represents a project. Take out your gun and shoot one of them to find out more. If you want to jump into a project's VR experience, just press the primary button on your controller after shooting its star.

Show Star Constellations by:

Figure 4.3: The information panel on the table with the button to configure the visualization. Editor view.

The dynamic elements of the Projectverse VR environment include the VR device representation and all means of interacting with the environment, i.e., the laser gun that each player receives and the button to rearrange the visualization. The device representation consists of a prefab that holds tracked pose drivers to communicate WebXR device data to the Unity application. Furthermore, we use spheres to represent their own hands to the player. Finally, a small, wrist watch-like screen with information about the control layout of Projectverse is added to the sphere representing the left hand, at about wrist height. This can be observed in Figure 4.4. On the functionality side, we include scripts for grabbing and teleportation with this prefab. These are discussed in Section 4.3.1. The laser gun and push button are explained later in this section.

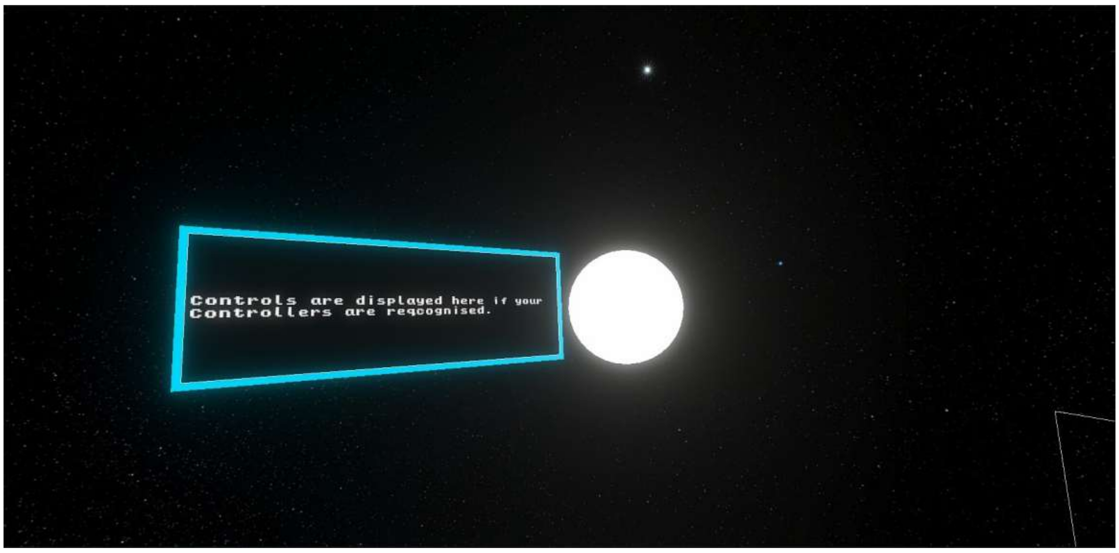

Controls are displayed here if your
Controllers are reqoognised.

Figure 4.4: Sphere for hand representation with controls information tablet. Editor view.

Networking and Avatars

Since we want the VR environment to be multiplayer-capable, we use the Photon Unity Networking 2 (PUN) package[4] available in the Unity Asset store. In order to use the functionality it includes, we create an empty "Network Manager" GameObject and attach to it a network manager script that handles the creation of the multiplayer server and its rooms when players join the VR environment. An outline of the server join functionality can be observed in Figure 4.5, excerpts from the network manager script are shown in Listing 4.2. When Projectverse is launched, this script automatically attempts to connect to the Photon multiplayer server. If this connection is successful, we request information on whether a multiplayer room is already present, or whether one needs to be created. Rooms are virtual spaces that PUN uses to separate player groups among the same application, i.e. only players in the same room can interact with each other in an application. Projectverse uses just one room, since we do not expect player counts to exceed five at a time. We provide enough space in the play area for this number of players.

[4]https://www.photonengine.com/pun

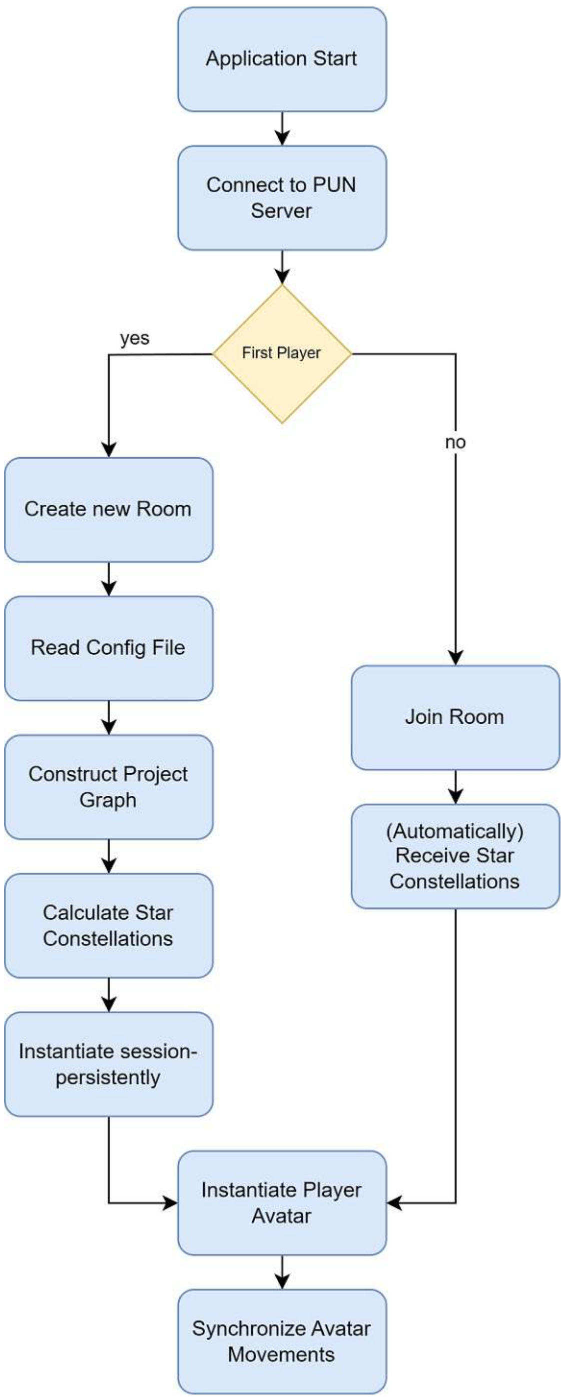

Application Start
Connect to PUN Server
First Player
yes
no
Create new Room
Read Config File
Construct Project Graph
Calculate Star Constellations
Instantiate session-persistently
Join Room
(Automatically) Receive Star Constellations
Instantiate Player Avatar
Synchronize Avatar Movements

Figure 4.5: Flowchart of actions performed when launching Projectverse.

We distinguish between the master player and all other players. The absence of a room upon server connection implies the absence of other players at the moment a player joins

```
public class ServerManager : MonoBehaviourPunCallbacks
{
    public bool AutoConnect = true;

    void Start()
    {
        if (AutoConnect)
        {
            ConnectNow();
        }
    }

    public void ConnectNow()
    {
        PhotonNetwork.ConnectUsingSettings();
    }

    public override void OnConnectedToMaster()
    {
        RoomOptions roomOptions = new RoomOptions();
        roomOptions.MaxPlayers = 5;

        PhotonNetwork.JoinOrCreateRoom("mainRoom", roomOptions,
            TypedLobby.Default);
    }

    public override void OnCreatedRoom()
    {
        CreateScene()
    }

    public override void OnJoinedRoom()
    {
        GameObject avatar = PhotonNetwork.Instantiate("
           avatarPrefab", new Vector3(0f, 0f, 0f), Quaternion.
           identity, 0, new object[] {
            PhotonNetwork.LocalPlayer.ActorNumber
        });
    }
}
```

Listing 4.2: The network manager script.

```
void Update()
    {
        Vector3 headPosition = Head.position;
        Vector3 handLeftPosition = HandLeft.position;
        Vector3 handRightPosition = HandRight.position;

        BodyLine.SetPosition(0, headPosition);

        Vector3 centerPosition = new Vector3(headPosition.x,
           headPosition.y - 0.20f, headPosition.z);
        BodyLine.SetPosition(1, centerPosition);

        HandLeftLine.SetPosition(0, handLeftPosition);
        HandLeftLine.SetPosition(1, centerPosition);

        HandRightLine.SetPosition(0, handRightPosition);
        HandRightLine.SetPosition(1, centerPosition);
    }
```

Listing 4.3: Controlling the lines of the avatars.

the server. PUN therefore dedicates this player the master player. Since no room is yet present, we create one through the master player, and automatically join it. At this point, we also make use of the master player property - because it is unique among a room - and let the master player's client initiate the calculation of the project graph. In case a room is already present and thus other players are already on the server, the script joins this room.

Finally, after all room join functionality has been completed, an avatar model is instantiated for each player who joins the room. This model represents the hands and head of each player to all others. Therefore, the client-local movements of the controllers and the VR headset are propagated through these assets to the server, and thus, all other players. Since our avatars should match the overall art style of Projectverse and, therefore, resemble star constellations, we choose spheres for the controllers and head representations, and add a light-emissive material to these. In addition, we connect these spheres to form a connected graph, where the lines represent the arms and the neck, respectively. We move these lines based on the movements of the hands and head, respectively. The intersection where the arms and head lines meet is fixed to a position 20 centimeters below the head. The function that controls these positions is shown in Listing 4.3.

The coloring of the avatars is picked from a range of four colors by the order of server joins, i.e. the master player is always player 1, and receives blue, whereas the second player to join receives green as their avatar color etc.. An overview of the multiplayer avatar in different colorings is given in Figure 4.6. The laser gun used for interacting with the environment is instantiated along with the avatar.

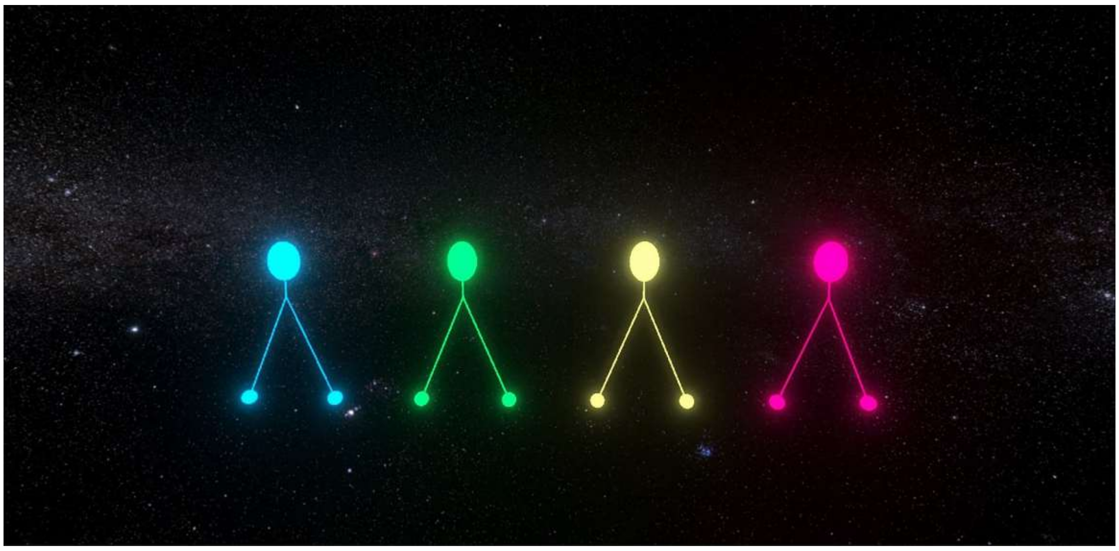


Figure 4.6: The multiplayer avatar used in Projectverse. From left to right: Players 1-4 etc.. Editor view.

Parsing JSON into the Project Graph

The JSON configuration file and the Projectverse application build are eventually placed on a web server. Therefore, we use a `UnityWebRequest` to request the transfer of the configuration file to Projectverse. Since we want to use the contents of the file in C#, in an object-oriented context, we also utilize the `FromJson` function provided by Unity's `JSONSerializeModule`. To use this function, we create a hierarchy of classes that resembles the data structure of our JSON file. The code of these classes is shown in Listing 4.4.

We construct the project graph from the objects created while parsing the configuration file. In total, we generate two drawings of the data we parsed, one where each star constellation holds the stars one researcher is responsible for, i.e., a sorting by researcher, and one, where the constellations represent the categories given in the JSON file.

Project Graph to Visualization

Utilizing the algorithms described in Section 3.3.2, we calculate the embeddings of the project graph in a C# script that iterates over lists of objects from the class structure in Listing 4.4.

With all the required positions calculated and the constellation drawings finished for both layout types, that is, by category and by main researcher, we begin instantiating the visual representations of our visualization. The default layout of the visualization shall

```csharp
[System.Serializable]
public class WebXRAppsData
{
    public WebXRAppGroup[] Groups;
}

[System.Serializable]
public class WebXRAppGroup
{
    public string Responsible;
    public WebXRApp[] Applications;
}

[System.Serializable]
public class WebXRApp
{
    public string Name;
    public string Location;
    public string InfoUrl;
    public string Category;
    public bool GetDataFromJson;
    public string ProjectSummary;
    public string[] TeamMembers;
    public Vector3 ByNamePosition;
    public Vector3 ByCategoryPosition;
}
```

Listing 4.4: C# Classes mapped to the attributes of the JSON configuration file.

be the arrangement by main researcher. First, we instantiate all stars. We use a simple spherical model for these and use a light-emitting material to properly represent a star's shine. Here, each constellation receives one of six possible colors, whereas the coloring choice works similarly to coloring the player avatars. This material is also applied to the labels of the individual stars and the constellation label, which we instantiate next. An overview of the color choices for the stars, applied to the star model, is shown in Figure 4.7.

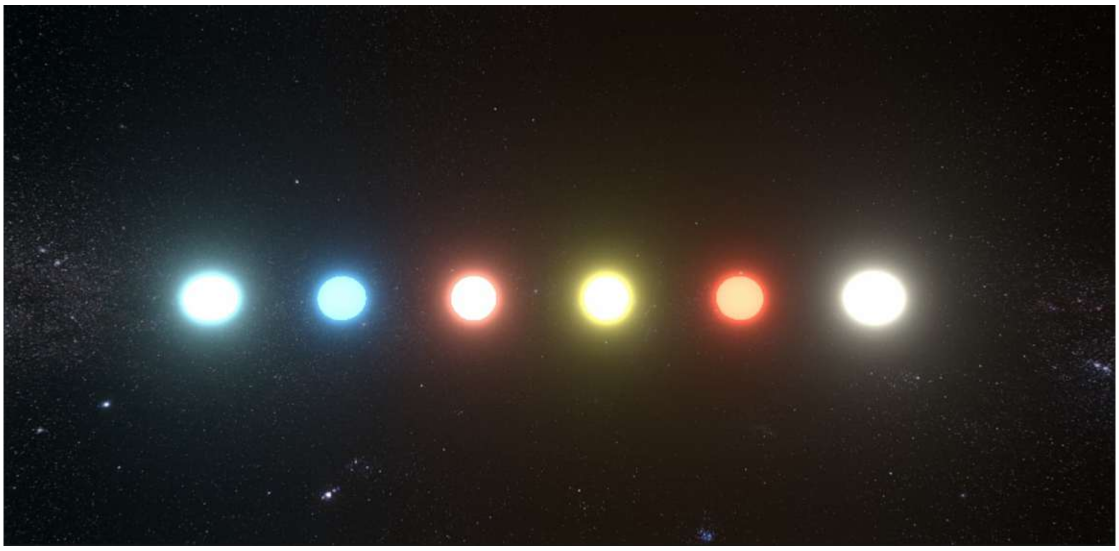


Figure 4.7: Color palette for the star constellations. Editor view.

For all instantiations, we use PUN's `PhotonNetwork.InstantiateRoomObject` function to ensure consistency along the multiplayer session. The final step of building the scene is the instantiation of Unity `LineRenderers` that connect the stars into constellations. These lines are also colored accordingly, and their `Positions` attributes are continuously synchronized with the multiplayer session using the functionality provided by `IPunObservable`. An overview of the scene with complete star constellations is shown in Figure 4.8.

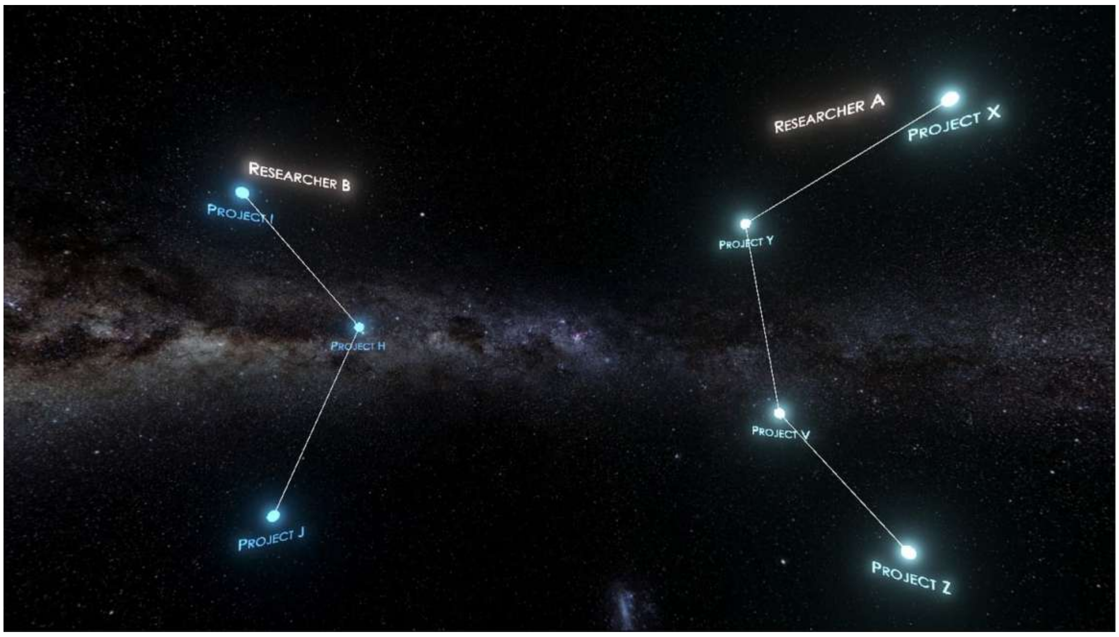


Figure 4.8: A scene of star constellations arranged by main researcher. First person game view.

Annotating Stars in Constellations

With our star constellations complete, we need to annotate the labels in it with the proper information. The star labels themselves are prefabs that consist of a canvas and a Unity `TextMeshPro` Text component placed on it. Furthermore, we use a script on the label prefab that controls the display of information on the label, that is, the text written on the `TextMeshPro` component. Photon's `OnPhotonInstantiate` callback function provides the point in time at which we want to write text to the label. We therefore pass the respective text for each label to the `PhotonNetwork.InstantiateRoomObject` function in its `data` property and read it from the `info` parameter of the `OnPhotonInstantiate` function. We then write this text to the `TextMeshPro` component.

Aside from the correct text being written in this function, we also apply the correct material and rotate the label, such that it faces the scene's origin point. The code for this function is provided in Listing 4.5.

Similarly, on each star prefab, we use a script to hold information on all the data of the project it represents, i.e., the project title, summary, location of the build on the server, and the team members. All these fields are also set using `OnPhotonInstantiate`, which ensures that they have the same values for all players in the multiplayer session.

```
public void OnPhotonInstantiate(PhotonMessageInfo info)
{
    materialIndex = (int)info.photonView.InstantiationData[0];
    appName = JsonUtility.FromJson<WebXRApp>((string)info.
       photonView.InstantiationData[1]).Name;

    LabelText.text = appName;

    transform.LookAt(Vector3.zero);

    LabelText.fontMaterial = LabelMaterials[materialIndex %
       LabelMaterials.Length];
}
```

Listing 4.5: The `OnPhotonInstantiate` function used on the star label prefabs.

User Interactions

Laser Gun We use a tool in the shape of a laser gun to let users interact with the environment. The functionality of this gun includes shooting stars in order to select them for information display, displaying said information to the user, and initiating the transition into a project application. We again use light-emitting materials for the laser gun to match the art style of Projectverse. The laser gun has a screen on top, and otherwise resembles laser blasters seen in science fiction movies like Star Trek. It is shown in Figure 3.9

The laser gun is a grabbable object in the scene, i.e., it can be grabbed using the grab button of the right hand VR controller and will then follow the controller's position. The implementation of this grabbing feature is discussed in Section 4.3.1.

To provide the functionality of the laser gun, we use a script that controls the shooting, writing of text to the gun's screen, and the transition to a project application. For shooting the gun, we use a line renderer that represents the laser emitted from the gun's tip, as well as particle systems that resemble sparks coming from the tip of the gun as well as the impact point of the laser. If the laser hits a star, the shooting stops and we then query the star's script for information on the project it represents. This information is then written to the screen on the gun. The shooting laser gun can be observed in Figure 4.9.

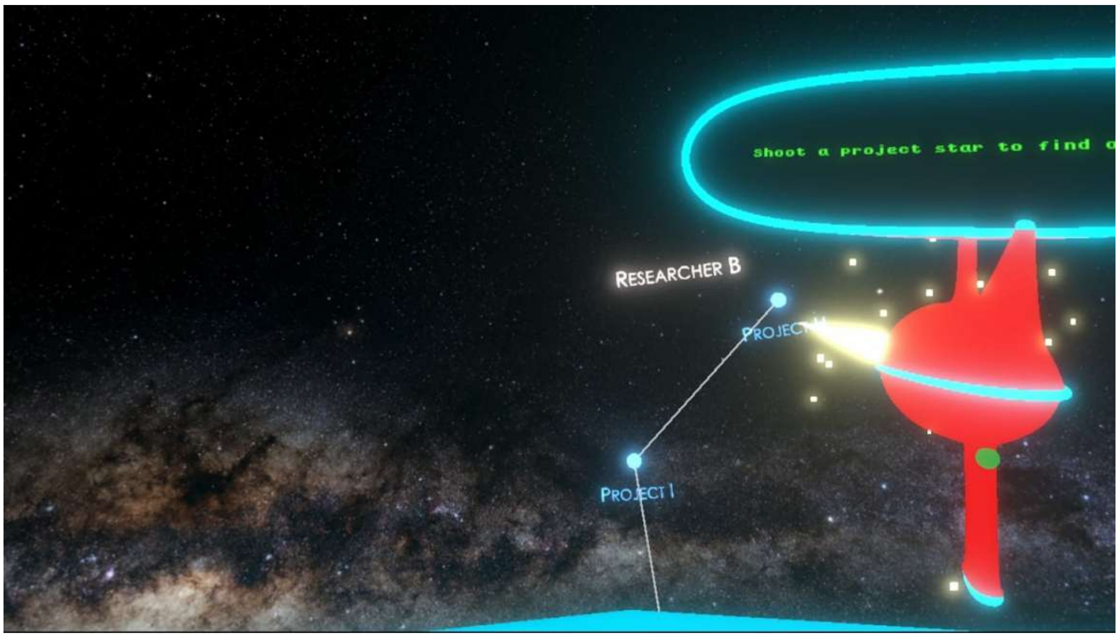


Figure 4.9: Shooting the laser gun. First person game view.

For the screen, we again use Unity `TextMeshPro` text components to which we write the respective information. This information includes the project title, summary and team members. We write all this information to the screen in a typewriter animation as used e.g. for displaying subtitles in video game dialogues. If the information is too long to fit into the display entirely, the screen becomes scrollable by pushing the thumbstick or touchpad, depending on the controller type used. Finally, we also enable the small green light on the back side of the gun's grip and show an info text on how to transition to the selected project application. This indicates to the user that the gun is ready to execute the transition to this application. The laser gun in the transition-ready state is shown in Figure 4.10.

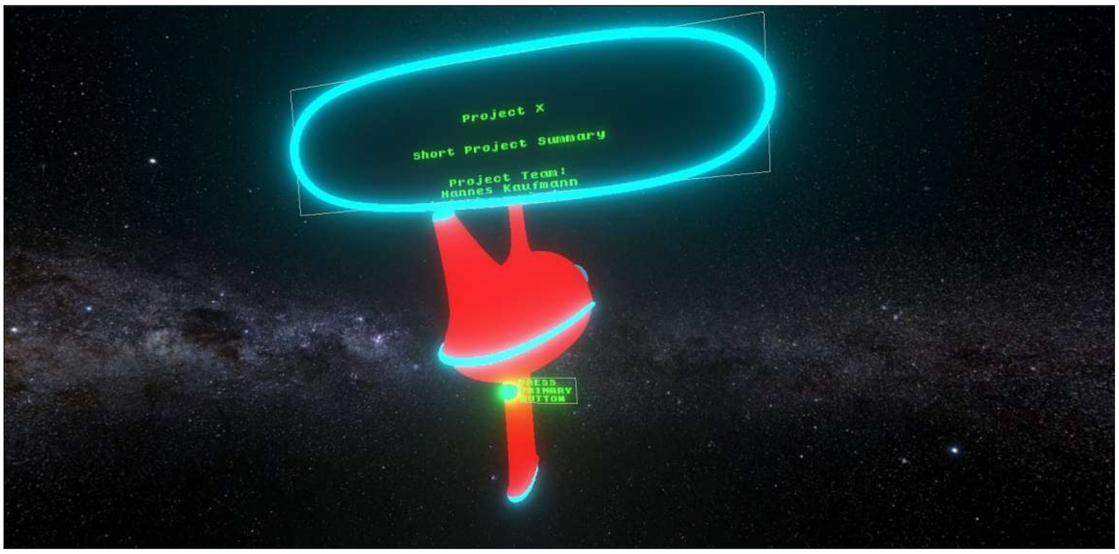


Figure 4.10: Laser gun with project information displayed, ready for transition. First person game view.

Each player receives one laser gun tool, which only they can grab and use. It is spawned at the time the player joins the multiplayer session and removed together with the player's avatar once they leave the session. To provide ease-of-use with the laser gun, we introduce a tool belt, which moves together with the player's head and is situated at about hip height, resembling the position of a holster. The belt is represented by a semi-transparent sphere object, which utilizes a snap-on mechanic to receive the gun when the player places it at the sphere's position. By grabbing the gun when it is stored in this belt, the player can remove it from the belt slot.

Changing the Layout of Constellations We enable users to change the layout of the star constellations visualization such that they can use a push button located on the table in the center of the play area to switch between two layouts, one where each star constellation represents the projects of one main researcher, and one where the star constellations represent the categories of the research projects, i.e. all projects of one category are visualized using the stars of one constellation.

The button we use for this is of mechanical nature, i.e. no further inputs than just moving the controller on top of the button, and then downwards, are required to push it.

We register pushes of the button using a script. Here, we initially disable the button for the duration of the star shifting animation. Next, we invoke a remote procedure call using functionality provided by PUN. This procedure is executed only on the master player's client since this is the client that has the right to move all objects part of the visualization within the multiplayer session, i.e., the owner of these objects' transforms.

The remote procedure first cleans up all line renderers of the present star constellations to “dissolve” them:

```
foreach (GameObject line in GameObject.FindGameObjectsWithTag("
    StarLine"))
{
    StartCoroutine(WaitForOwnershipAndDestroy(line));
}
```

In the next step, we also remove all star constellation labels, i.e., the labels that name the main researcher or category per star constellation, for them to be later replaced by the new labels of the changed visualization layout:

```
foreach (GameObject label in GameObject.FindGameObjectsWithTag(
    "StarLabel"))
{
    StartCoroutine(WaitForOwnershipAndDestroy(label));
}
```

We then move each star separately to its new position in the chosen visualization layout. Both possible layouts, and positions for all stars within these, were already calculated at the start of Projectverse (see Section 4.1.1) and only need to be retrieved at this point. This rearrangement is done in a coroutine that animates each star’s movement. The speed of a star during the movement should be slow at first, then faster, and finally slow down again before arriving at the destination. Therefore, this speed is given as

$$v = \frac{1}{2}(1 - cos(t\pi)),$$

where t denotes the normalized time change since start of the animation and is defined as

$$t = \frac{t_{now} - t_{start}}{T}.$$

Here, t_{now} denotes the current time, whereas t_{start} denotes the time of the start of the animation. T represents the chosen duration the animation should take overall. The animation uses the calculated speed value to move each star in discrete steps - once each frame - between its old and new positions.

All stars are moved at the same time and for the same duration. Once this movement finishes, we relabel each constellation by the aspect dictated by the layout, i.e., category or main researcher, and recolor the stars: Since stars of previously different constellations may now be part of the same constellation, this step is necessary. We also instantiate new line renderers as defined in our calculation of the constellations at application launch. Finally, the push button is reactivated.

Transitioning into a Project

With a research project selected using the laser gun, the player can press the respective button to initiate the transition to this project's application. For Oculus Touch controllers, this is the "A"-Button on the right hand controller, on HTC Vive controllers, we use the left controller's grab button, with the gun being held in the right hand. The transition to an application works as follows:

1. **Countdown on the Laser Gun:** A countdown is shown on the gun's display, starting at five seconds and counting down to zero, indicating that the transition is about to take place.
2. **Information on the Laser Gun:** Since with WebXR's current state, we cannot persist an ongoing VR session among multiple web pages, we show a short information text to the user, requiring them to restart the VR session once the transition is complete. Since every project application is hosted on a different subpage of the web page of the Projectverse VR environment, we need to move to a different web page for the transition.
3. **Particle System:** As proposed in Section 3.3 and shown in Figure 3.10, we show the user a particle system that resembles "interstellar jump" animations known from science fiction movies like Star Wars or Star Trek. This particle system originates from the star and its particles move quickly towards the player, increasing their count over the duration of the countdown. In addition, we play a teleport-like sound to accompany this animation.
4. **Transition:** Once the countdown ends, we move to the selected application's page, where the respective program is automatically launched. The user now needs to start the VR session on the screen or the virtual desktop, in order to play the application on their VR headset.

A still image of the transition animation is shown in Figure 4.11.

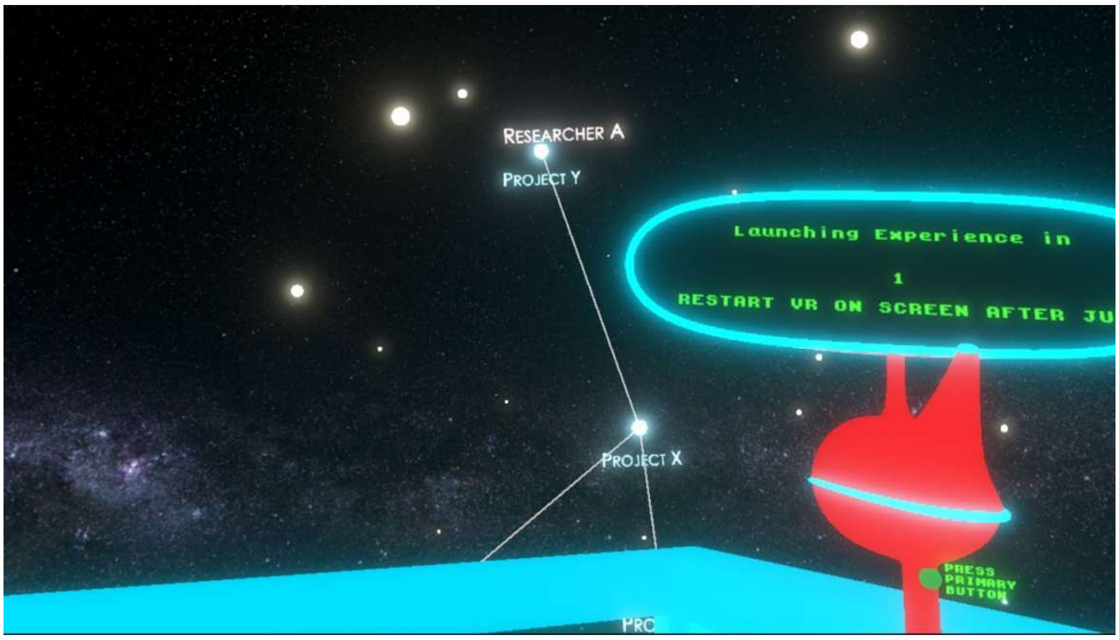


Figure 4.11: Laser gun and selected star during the transition animation. First person game view.

To initiate the countdown, we once again write to the `TextMeshPro` text component on the gun's screen and then launch two coroutines, which overwrite the countdown number and animate the particle system for the visual effect, respectively. We also start playing the particle system and sound effect at this point:

```
ProjectNameTMP.fontSize = 2;
ProjectNameTMP.text =
    "\n\nLaunching Experience in\n\n\n\n" +
    CountDownDuration +
    "\n\nRESTART VR ON SCREEN AFTER JUMP";

StartCoroutine(Countdown(CountDownDuration));
StartCoroutine(PositionAndUpdateJumpParticles());
JumpParticleSystem.GetComponent<ParticleSystem>().Play();
JumpParticleSystem.GetComponent<AudioSource>().Play();
```

The coroutine for animating the particle system is executed in a terminal state of the Projectverse VR environment, i.e., shortly before it quits due to the transition. We therefore use an endless loop here which is executed once per frame. In this loop, we find the direction from the VR headset to the selected star and place the particle system's origin five meters away from the headset in that direction. In addition, depending on the time change since the start of the coroutine, we increase the particle count using its `EmissionModule` property:

```
private IEnumerator PositionAndUpdateJumpParticles()
{
    float t = 0;

    while (true)
    {
        t += Time.deltaTime;

        Vector3 cameraPosition = MainCamera.transform.position;
        Vector3 DirectionToStar = lastHitStar.transform.
            position - cameraPosition;
        DirectionToStar.Normalize();

        JumpParticleSystem.transform.position = cameraPosition
            + DirectionToStar * 5;
        JumpParticleSystem.transform.LookAt(cameraPosition);

        ParticleSystem.EmissionModule emissionModule =
            JumpParticleSystem.GetComponent<ParticleSystem>().
            emission;
        ParticleSystem.MinMaxCurve minMaxCurve = emissionModule
            .rateOverTime;
        minMaxCurve.constant = 10 * t + 50;
        emissionModule.rateOverTime = minMaxCurve;

        yield return null;
    }
}
```

The coroutine that animates the countdown on the gun's screen, on the other hand, we replace the previous countdown number on the screen in a loop that is executed once per second and terminates as we reach zero. After the termination, the transition is executed from this coroutine, utilizing the `PageSwitcher.CallOpenURL` function:

```
private IEnumerator Countdown(int startValue)
{
    int i = startValue + 1;
    while (i > 0)
    {
        i--;

        ProjectNameTMP.text = ProjectNameTMP.text.Replace((i +
           1).ToString(), i.ToString());
        yield return new WaitForSeconds(1);
    }

    if (Application.platform == RuntimePlatform.WebGLPlayer)
        PageSwitcher.CallOpenURL(lastHitStar.GetComponent<
           InfoGrabber>().Location);
}
```

The `PageSwitcher.CallOpenURL` function's only purpose is to call an external function named `PageSwitcher.OpenUrl`. This external function utilizes Unity's framework for interacting with browser scripting [5] and is defined in JavaScript. It is passed a URL as a parameter, which it then redirects the browser window to:

```
mergeInto(LibraryManager.library, {
  OpenURL: function (url) {
    window.open(Pointer_stringify(url), '_self');
  },
});
```

When transitioning to an application, we call this function with the URL saved in the `Location` attribute of the JSON configuration file (see Listing label=lst:config-json). Since the WebXR Device API does not yet support sessions that persist when switching pages, the VR session quits at this point and needs to be reactivated using a button on the screen. Users can circumvent having to take of the HMD by using virtual desktop within VR to click said button.

4.1.3 Enabling Projectverse for the Web Environment

Since Projectverse should be hosted on a web page, we use WebXR exporter [6] in Unity's build process. We build the program using the Unity WebGL build target and subsequently deploy it to the VR group's web server by placing it in the respective directory. This directory serves as Projectverse's root, and all research project applications that are

[5] https://docs.unity3d.com/Manual/webgl-interactingwithbrowserscripting.html

[6] https://github.com/De-Panther/unity-webxr-export

deployed to the it are placed in its subfolders, which allows us to use relative URLs in the JSON configuration file. An overview of Projectverse's web server folder structure is shown in Figure 4.12.

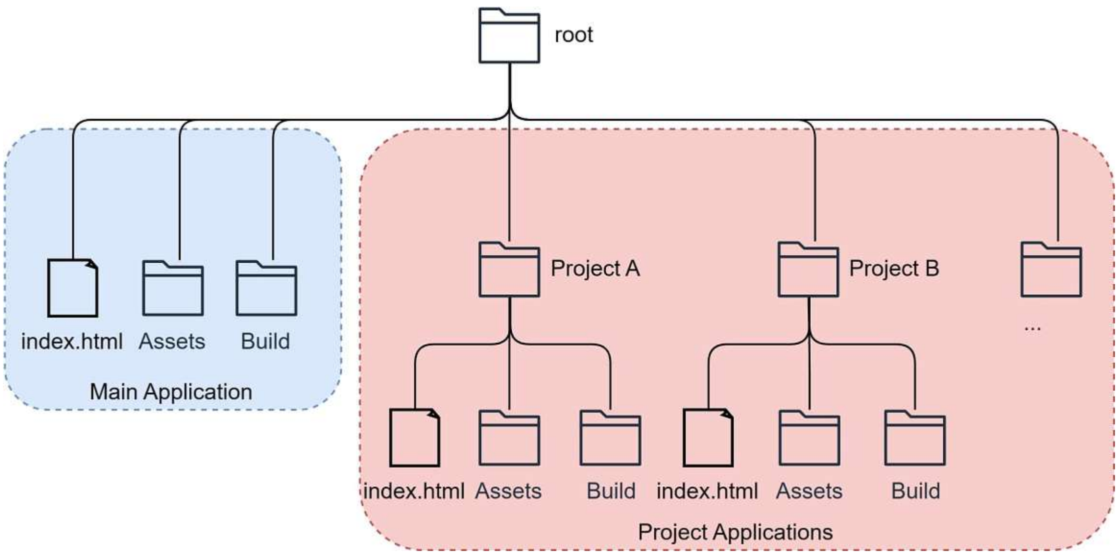


Figure 4.12: Folder structure on the web server.

4.2 Deployment to the Web Server and adding new Projects

The process of deploying project applications to Projectverse should be as follows:

1. **Adapting applications to the web environment.** Some applications are already in a state that enables them to be executed in the web browser, e.g. projects created with the A-Frame framework. However, most of the VR group's applications were developed in Unity and built for the Windows platform. This means that at least re-building these applications to the WebGL target is necessary. In most cases, additional steps are required, such as importing and enabling WebXR to communicate with and use VR devices within the browser.
2. **Building the Project.** After all necessary adaptations are made, the application needs to be built. Since we want Projectverse to run in the web, applications hosted, or deployed, on it, need to also be web-capable. In the case of Unity applications, this means building to WebGL.
3. **Uploading to the web server.** With the application build complete, the application needs to be uploaded to the web server on which Projectverse is hosted.

4. **Adding configuration entry.** To allow the VR environment to generate a project star for the newly added project, it must be added to the previously mentioned configuration file (see Figure 4.13). Here, a data entry must be made that contains at least the title of the application, the location of its build on the web server and its category. In addition, information needs to be provided on whether the project summary and team members should be scraped from the project's web page. If not, these data have to be entered as well.

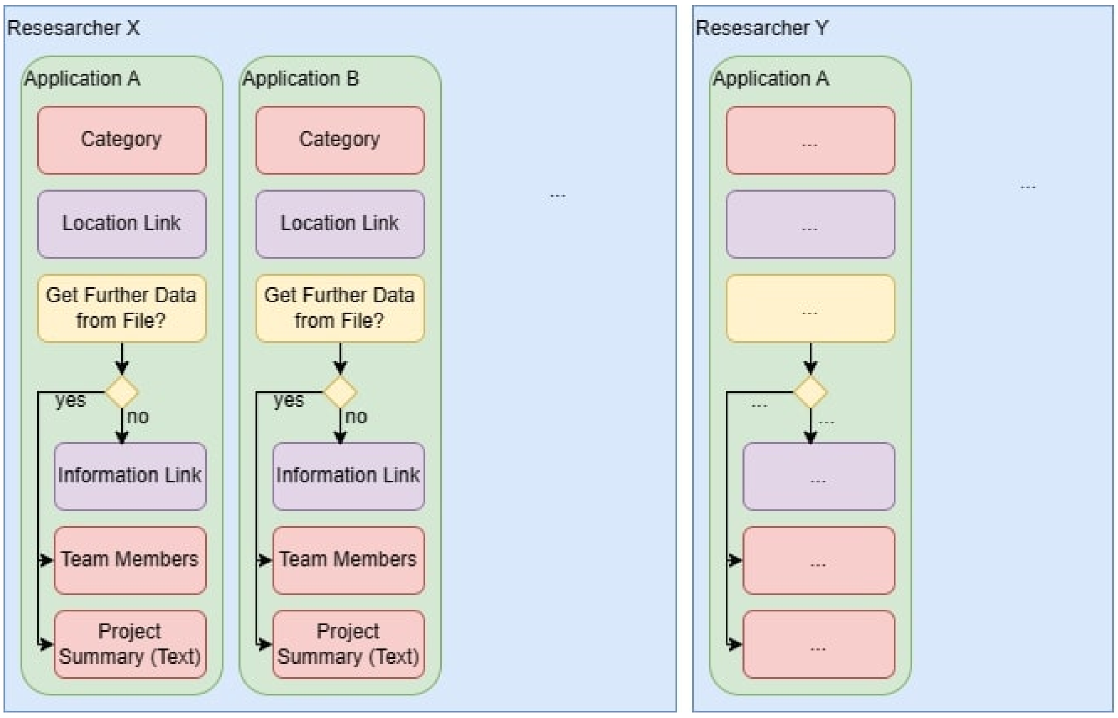


Figure 4.13: Schematic outline of the configuration data specification used by Projectverse.

4.3 Developer Toolkit

We require applications resulting from research projects that should be deployed on Projectverse to be adapted to the web environment. To fulfill this requirement, we provide a toolkit to the developers of these applications that helps with the adaptation. Since we develop the toolkit for Unity, the applications profiting from it also need to be implemented in this game engine. Applications originating from other frameworks can still be deployed to Projectverse, given that they run in the web browser.

Our toolkit relies heavily on functionality provided by WebXR Exporter[7] and WebXR Interactions[8]. These packages allow us to read data from the VR headset and controllers

[7]https://openupm.com/packages/com.de-panther.webxr/
[8]https://openupm.com/packages/com.de-panther.webxr-interactions/

when used within a WebXR session, as is the case in the VR environment of Projectverse. Project applications that were developed for the Windows environment, on the other hand, do not natively support WebXR, but use OpenXR for Unity[9] or the Unity Oculus Integration[10] to communicate with VR devices. Therefore, developers first need to exchange these plugins for those that use WebXR when adapting their applications.

Since this exchange implicates the requirement for replacing the VR device representation in Unity, as well as parts of the gameplay logic that use this representation and functionality provided by e.g. OpenXR, like grabbing objects or locomotion methods, we provide the replacement functionality in our toolkit alongside a VR device representation prefab.

4.3.1 Functionality

The functionality contained in our toolkit includes, as proposed in Section 3.4 and Figure 3.2, scripts for grabbing objects, interacting with UI elements, as well as teleporting and moving with the controller's thumbstick or touchpad as locomotion methods.

Grabbing We implement the grab mechanic in three different ways:

- **Grab by Tag:** This method allows developers to set one or multiple tags in Unity which designate grabbable objects. These tags then need to be assigned to the objects that should be grabbable.
- **Grab by Script:** With the grab logic being controlled from a script attached to the VR controller, another, simple script is placed on the objects that should be grabbable.
- **Grab Interactable:** For this method, we provide a separate script that controls the grab logic from the grabbable objects' side. We do not need a grab script on the VR controller in this case, but one on each object that should be grabbable.

With all of these methods, we allow the user to choose the button that should be used to grab an object. For the logic of the grab itself, we draw inspiration from Robinett and Holloway[12], who require that for grabbing an object in a virtual world, this object's position relative to the hand must be constant during the grab, i.e., while being grabbed, the object follows the controller. We implement this mechanic in two different ways, namely:

[9]https://docs.unity3d.com/Packages/com.unity.xr.openxr@1.8/manual/index.html

[10]https://assetstore.unity.com/packages/tools/integration/oculus-integration-82022

- **Transform Parenting:** Transforms allow for parent-child relationships in Unity, in which a child moves alongside its parent when the parent's position changes. We utilize this property for grabbing by making the grabbed object a child of the controller's transform.
- **Joint:** In cases where parent-child relationships in the scene should not be changed for the use of grabbing objects, we provide the joint method, which uses a `FixedJoint` component on the controller which connects to the grabbed object for the duration of the grab, thus moving it together with the controller.

In addition to these features, we also let the developer decide the following options:

- **Haptic Feedback:** In case this option is used, we trigger vibrations when a grabbable object is within grabbing range of the controller, moves out of this range, or is grabbed or released.
- **Controller Visibility:** Some applications of grabbing require that the controller model should be invisible while an object is grabbed, e.g. when very small objects are grabbed, they might be occluded by the controller model. This option allows developers to make the controller model invisible for the duration of a grab.

Locomotion For locomotion methods, we provide thumbstick/touchpad movement and teleportation. Both are realized as scripts to be attached to the controller object in Unity. With the thumbstick movement, we let the developer decide between three approaches:

- **Freeflight:** The input from the thumbstick or touchpad is applied directly to the player's position. Depending on the movement speed factor selected by the developer, we move the player through three-dimensional space.
- **Floor locked:** Before changing the player's position, we project the movement vector onto the x-z-plane, thus moving the player parallel to the ground and avoiding movements in the y-axis.

Furthermore, we provide an option to allow only forward and backward movements, i.e., moving the player only along their local z-axis rather than both the x- and z-axes.

The teleportation method we provide is oriented on that proposed by Bozgeyikli et al.[2] and its workflow can be described as follows:

1. **Initiation:** By making a controller input, e.g. pushing and holding a button or the touchpad or joystick, the player initiates a teleport. A ray is emitted from the controller in the direction the player's hand points.

2. **Target Selection:** By rotating the controller, the player can change the ray's direction and thus, the targeted teleport location. This location is verified and then displayed as a marker in the shape of a circle at the point where the ray meets the surface the player targets.

3. **Verification:** Depending on the developer's choice of verification factors, the ray is either drawn as a solid line for targets that the player is allowed to teleport to, or a dashed line for targets to which teleportation is not allowed.

4. **Execution:** Upon release of the teleport input, and in case of possible target verification, the player is transported to the target location instantly by modifying their position. The line and the marker disappear.

For target validation methods, we offer the following options:

- **Tag-based:** All objects whose surfaces are valid teleportation targets receive the same tag in Unity.
- **Layer-based:** All objects whose surfaces are valid teleportation targets receive one of the layers the developer allows.
- **Gradient-based:** All surfaces whose gradients are less than a developer-defined angle are valid teleportation targets.

Additionally, we provide functionality to ignore certain surfaces when targeting a teleport, i.e., the targeting ray can penetrate these surfaces without verifying them at all. This can be useful e.g. when teleporting through windows or other transparent surfaces should be allowed. We implement this functionality on the basis of layers, i.e., the developer can select a range of layers that should be ignored by the targeting ray.

Finally, our teleport locomotion implementation also provides what we refer to as directional teleport: With this method, we use the thumbstick or touchpad as teleport input and let the user hereby select their orientation after teleport execution. We determine this orientation by the input position of the thumbstick or touchpad at the moment of teleport execution. The future position is visualized to the player using a tipped marker for the target selection, whose tip points in the selected direction. Both marker types are shown in Figure 4.14.

UI Interaction For UI interaction, we provide functionality to press UI buttons and scroll Unity `ScrollRect` elements. In both cases, the user interface at hand must be in world scale mode. For interacting with it, we provide the user with a laser pointer, which is implemented using a `LineRenderer` Component. The resulting line works similar to a laser pointer and is enabled when the user points at a valid UI element. Valid UI elements are determined by a selection of layers by the developer. In order to work with

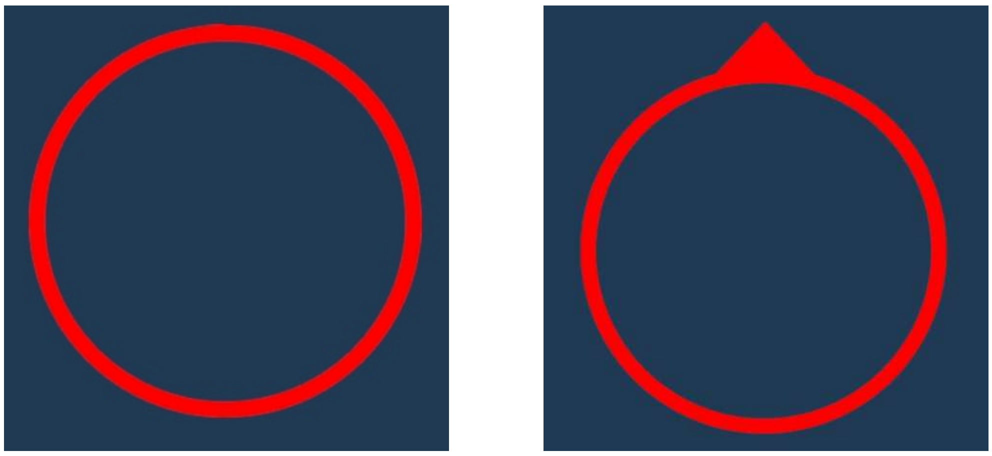

(a) Round marker. (b) Tipped marker.

Figure 4.14: Target markers used in our teleportation method.

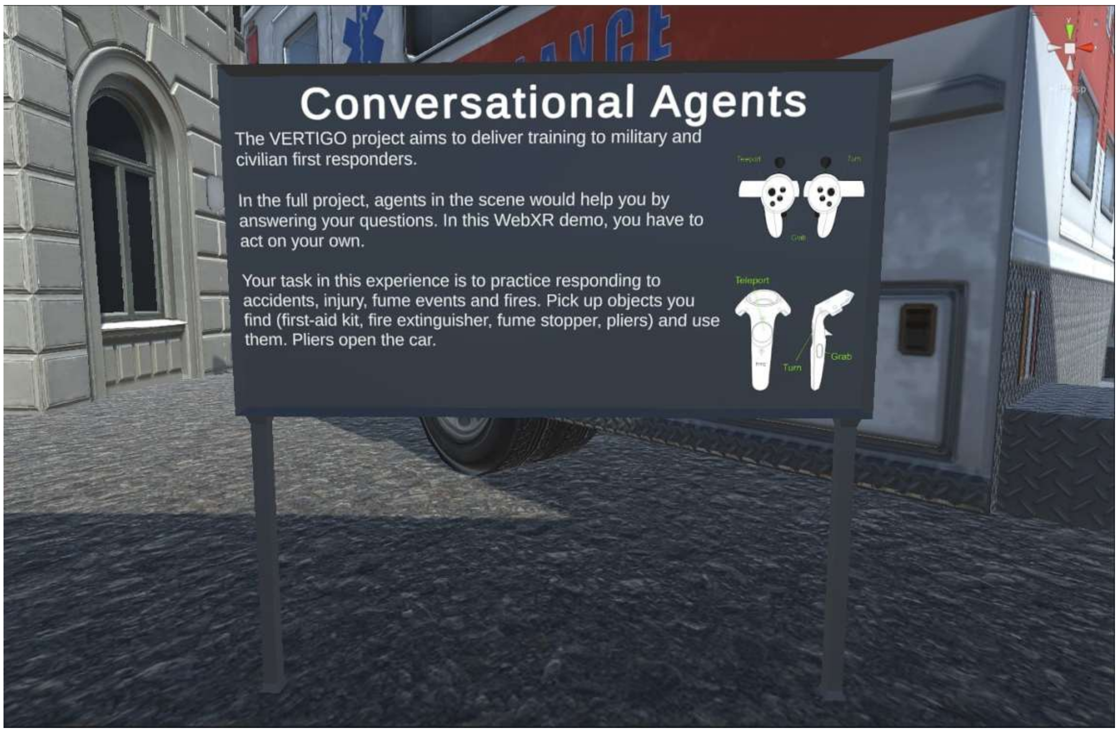
our method, all UI elements need to have `Collider` components on them in order to register a hit with the laser.

In case the user points the laser at a button, we then execute a push of this element when the user pushes the input determined by the developer. Here, we offer all buttons on the VR controllers, but not the joystick or touchpad. For scrolling a `ScrollRect`, we monitor inputs from the touchpad or joystick when such an element is hovered, and then scroll it according to the input.

4.3.2 Assets

To further simplify the adaptation process of project applications, we also provide assets in the form of prefabs. The following prefabs are included in our toolkit:

- **VR Device Representation:** Since the removal of OpenXR or the Oculus package also removes the VR device representation from an application, this prefab's purpose is to replace it. It contains sphere models for the controllers as well as a `GameObject` with not mesh for the VR headset, such that all three devices can be tracked and represented in the application. Here, the tracking source is WebXR. We also provide functionality with this prefab that allows users to transition back from to the Projectverse VR environment from a project application. The script for this purpose is already attached to the prefab, so that developers can easily include this functionality in their applications.
- **Info Board:** In order to provide information on control layouts as well as a short summary of a project application to users, we provide an info board prefab. Along with it, we provide images of controllers to be labeled with the control layout of the application. The info board is shown in Figure 4.15.


Conversational Agents

The VERTIGO project aims to deliver training to military and civilian first responders.

In the full project, agents in the scene would help you by answering your questions. In this WebXR demo, you have to act on your own.

Your task in this experience is to practice responding to accidents, injury, fume events and fires. Pick up objects you find (first-aid kit, fire extinguisher, fume stopper, pliers) and use them. Pliers open the car.

Figure 4.15: The info board prefab in use in a project application.

As mentioned above, to bring a player back from a project application, we use a script. To keep the input required for this action consistend among applications, we hard-code it and thus, require both grab buttons as well as both triggers to be pressed. These buttons are used because they are present in all consumer VR headsets. To bring the user back to the Projectverse VR environment when the required input is made, we again use the `PageSwitcher.CallOpenURL` function as discussed in Section 4.1.2, however, we use `"../"` as the target URL, since all project applications are situated on subpages of the Projectverse VR Environment.

4.3.3 Documentation

Our toolkit's final component is documentation we provide along with it. In it, we describe in detail the steps required to adapt both applications that were initially developed for the Windows platform as well as those already browser-ready, to Projectverse.

In the case of Windows applications, we require the following steps:

- **Package Installation:** To work with Projectverse, the WebXR Exporter and WebXR Interactions packages need to be importet into the project. Besides that, our developer toolkit also needs to be included in the application.

- **Removing old VR frameworks:** As discussed above, developers need to remove OpenXR or Oculus if they were used to communicate with VR devices.
- **Recreation of Gameplay:** In this step, we describe how to use our toolkit's components to recreate the gameplay logic of the application.
- **Build:** Due to the characteristics of the web environment, developers need to specify certain build settings and application configurations in order to build an application that can run in the browser.

Browser-ready applications, on the other hand, require only the installation of our developer toolkit as well as attaching our script for transitioning back to the Projectverse VR environment to the player `GameObject` and finally, compiling a new build.

To finally deploy an adapted and built application, it needs to be placed in a newly created subfolder of the Projectverse root folder, as shown in Figure 4.12 as well as its information entered into the JSON configuration file. Projectverse will then automatically generate a star for it and either add this to an existing star constellation or create a new one upon refreshing the web page.

CHAPTER 5

Evaluation

5.1 Methodology

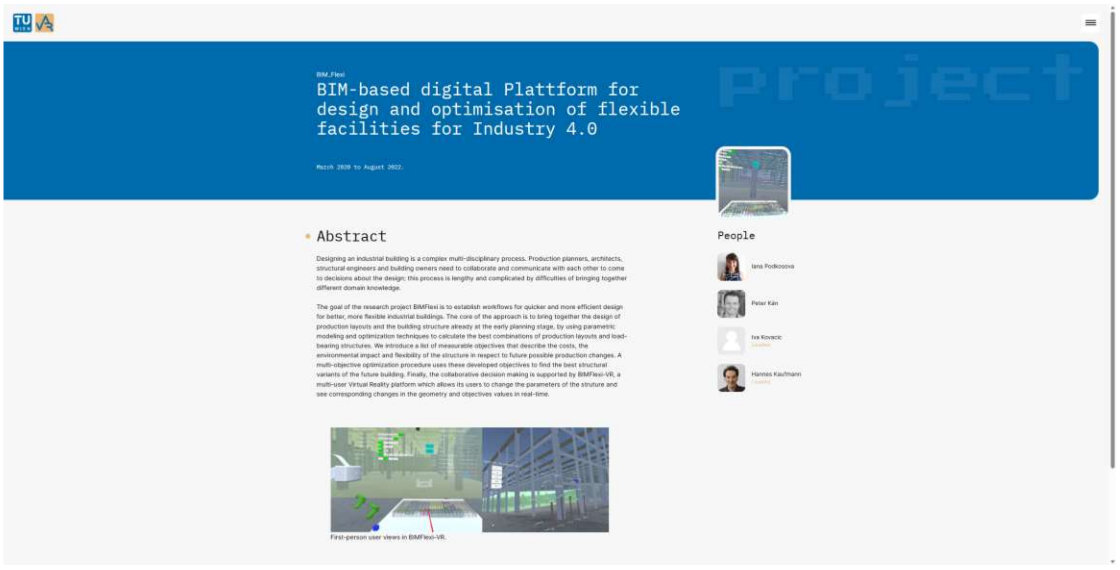
The intention behind Projectverse is to provide deeper and broader knowledge of the VR group's work in combination with making it accessible in a suitable medium. Therefore, we conducted a user study to evaluate the user friendliness, the extent of knowledge acquisition, and overall user preference of Projectverse to analyze the benefits Projectverse contributes when used alongside the VR group website. We invited single participants to our study that was conducted in a mixed form between guided and unguided trial, i.e., during some parts of the evaluation, assistance was provided when a participants asked for it.

We conducted a repeated-measures experiment. Participants first experienced the VR group website, then our Projectverse. The order of conditions was not counter-balanced since in our concepts, users would use Projectverse only after having browsed the website, in order to get a deeper understanding of projects they read about on the website. The evaluation was conducted in two parts. Participants first used the website, and answered the questionnaire section about it, then used Projectverse and again, answered the questionnaire section about it. After this, participants answered the questionnaire section about preferences.

During the second part of the evaluation, participants were always accompanied by an experimenter who would sit alongside the participant and initially explain the Projectverse VR environment and how to use it. Furthermore, questions regarding e.g. the controls layout of Projectverse were answered and help would be provided when problems with Projectverse itself occurred.

5.1.1 Task and Virtual Environment

As discussed, our study consisted of two parts, the website and Projectverse. In the website part, participants were asked to visit the VR group website and familiarize themselves with research projects. We asked participants to "visit at least 5 project pages", and specifically requested them to include "BIM_Flexi", "Conversational Agents", and "Haas VR", since we were going to to ask questions about these projects. The website content of BIM_Flexi as well as its application in Projectverse are shown in Figures 5.1 and 5.2. In the Projectverse part, participants were asked to visit Projectverse using an HMD and to again familiarize themselves with projects found on Projectverse. Again, we requested at least 5 projects, and asked for "BIM_Flexi", "Haas VR", and "Conversational Agents" to be among these. We also specifically instructed participants to not only read the information on the gun display, but also jump into the project applications to fully experience them. We gave instructions on how to use Oculus Home with virtual desktop such that they would not have to reactivate the VR session on the desktop when jumping. In both parts of the study, participants were instructed to close the website or Projectverse after finishing their experience and only then answer the questionnaire.


BIM_Flexi

BIM-based digital Plattform for design and optimisation of flexible facilities for Industry 4.0

March 2020 to August 2022.

Abstract

Designing an industrial building is a complex multi-disciplinary process. Production planners, architects, structural engineers and building owners need to collaborate and communicate with each other to come to decisions about the design; this process is lengthy and complicated by difficulties of bringing together different domain knowledge.

The goal of the research project BIMFlexi is to establish workflows for quicker and more efficient design for better, more flexible industrial buildings. The core of the approach is to bring together the design of production layouts and the building structure already at the early planning stage, by using parametric modeling and optimization techniques to calculate the best combinations of production layouts and load-bearing structures. We introduce a list of measurable objectives that describe the costs, the environmental impact and flexibility of the structure in respect to future possible production changes. A multi-objective optimization procedure uses these developed objectives to find the best structural variants of the future building. Finally, the collaborative decision making is supported by BIMFlexi-VR, a multi-user Virtual Reality platform which allows its users to change the parameters of the struture and see corresponding changes in the geometry and objectives values in real-time.

First-person user views in BIMFlexi-VR.

People

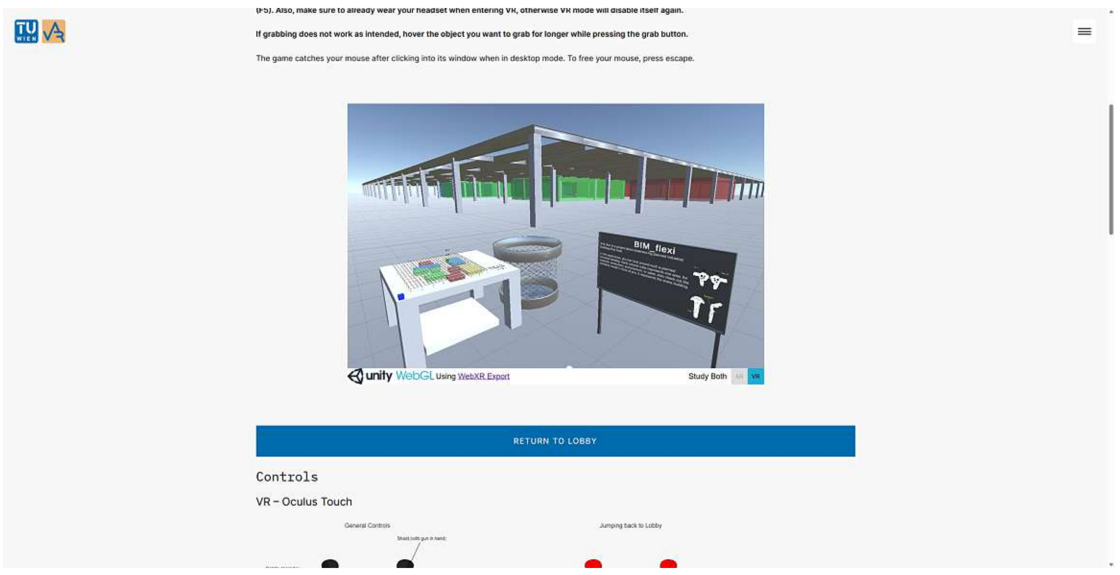
Iana Podkosova

Peter Kán

Iva Kovacic

Hannes Kaufmann

Figure 5.1: BIM_Flexi on the VR Group Website.


(F5). Also, make sure to already wear your headset when entering VR, otherwise VR mode will disable itself again.

If grabbing does not work as intended, hover the object you want to grab for longer while pressing the grab button.

The game catches your mouse after clicking into its window when in desktop mode. To free your mouse, press escape.

Figure 5.2: BIM_Flexi in Projectverse.

To provide a comparable information basis, we configured Projectverse to provide applications of the three research projects named above, which are also described in detail on the website alongside four applications resulting from student work. The sky in Projectverse was populated with seven stars, arranged in three constellations when sorted by main researcher, and five constellations when sorted by category. The multiplayer functionality was also activated, but not used since participants experienced Projectverse one person at a time.

The three projects we specifically asked participants to visit provide the following gameplay:

- **BIM_Flexi:** This project focuses on designing industrial buildings in a collaborative environment. The application we provide on Projectverse allows participants to experience a building visualized through BIM_Flexi by moving through it using teleport locomotion.
- **Haas VR:** This project aims to visualize large industrial environments, i.e., machinery and production chains. In the application we used for the user study, participants could experience a wafer-making machine and change its configuration using trigger-areas with which they could interact by moving the controllers into them. This would change certain parts of the machine. Movement was enabled through thumbstick/touchpad locomotion.
- **Conversational Agents:** This project aims to investigate the impact of conversational embodied agents in VR, specifically in urban exploration scenarios. The

application version used for the study places participants in a first-responder scenario in which they could free trapped car accident victims, fight fires or neutralize toxic substances. Since the conversational agents of this application require resources that are not available in the browser environment, we disabled them for the study version. Movement was enabled through thumbstick/touchpad locomotion.

In all applications, we provided info boards that showed the project description together with the control layout. An example is shown in Figure 4.15.

5.1.2 Metrics

Demographics Participants were asked to identify their gender and age, as well as state their levels of experience with both VR headsets and video games between 1 and 5, where 1 is the lowest and 5 the highest.

Questionnaire Our study questionnaire includes questions about user experience, knowledge gains and retaining, and workload. The full questionnaire can be found in Appendix A. Participants were asked to answer questions after experiencing each condition, the website and Projectverse. In the final part, we asked participants to choose between the website and Projectverse in multiple categories. We assessed knowledge gained from both interfaces alongside ease-of-use and personal preferences as well as comparing both mediums directly. Answers could be either on a 1 to 5 Likert scale or in free text as well as binary yes/no answers, depending on the question. For yes/no questions, we asked participants to explain their choice in a couple of sentences.

5.1.3 Materials

The evaluation was conducted with one Oculus Rift S VR headset and Oculus Touch Controllers. The HMD was connected to a desktop computer, which was used both to examine the website as well as use Projectverse. The experiments took place in a small room, where the play area measured approximately 1,5x1,8m. We set the HMD's virtual play area limit to this area. Each participant was brought to this room alone such that there would be no collisions with others.

5.1.4 Participants and Procedure

We had 11 participants, among them 7 men and 4 women. Each conducted the entire user study, thus all 11 participants' answers are included. Ages ranged between 20 and 26, with the median at 24. The distribution of participant ages is shown in Figure 5.3. The levels of experience with VR and video games are shown in Figure 5.4. The median experience level for using VR headsets was at 2. All but one participants answered experience levels 1 or 2, whereas the remaining one answered 4. For video games experience, the median was 4, with all options from the range present in the answers.

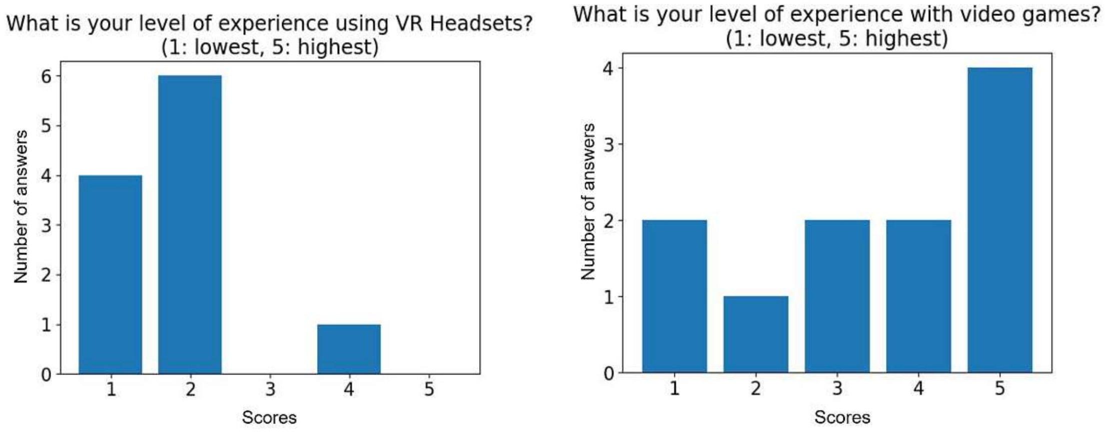


Figure 5.4: Experience levels of participants in our user study.

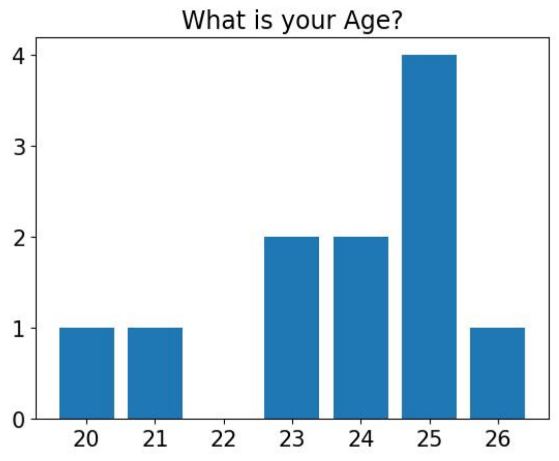


Figure 5.3: Age distribution of our user study.

Upon arrival, participants were greeted and introduced to the play area and its limitations as well as the HMD device and the controllers. The hardware and how to use it was explained as well as the virtual play area limit and how to handle it.

After this, participants tested the website by themselves and were subsequently asked to fill in the website part of the questionnaire. In the next step, with guidance from an experimenter, participants put on the HMD and could again familiarize themselves with the controls. Next, they could test Projectverse and ask questions about how to use it or the controls. After testing, participants answered the second part of the questionnaire, and finally, the third, in which they compared both media. The whole procedure lasted between 45 and 75 minutes per participant.

ID	Question
Q1	What were your initial impressions when using the application?
Q2	Did you get enough information to understand a project? If not, what did you miss? If yes, what was especially important for your understanding?
Q3	Which medium was easier in terms of navigation and interaction? Why was this medium easier?
Q4	Which medium provided you with a better understanding of project contents? Why did this medium provide you with a better understanding?
Q5	Which medium was more comfortable to use for an extended period? Why was this medium more comfortable to use?
Q6	Which medium required more work from you to find out information about the projects? Why did this medium require more work?
Q7	Which medium would you recommend to others who wish to find out about the VR Group's work? Why would you recommend this medium?
Q8	Which, if any, modifications would you like to see in the website?
Q9	Which, if any, modifications would you like to see in [the VR application]?
Q10	Do you see any added benefit to exploring projects through ProjectVerse? If so, what is this benefit?

Table 5.1: Questions used in the preference part of the qualitative evaluation.

5.2 Results

We analyzed the responses to our questionnaire in two main parts, one being the qualitative evaluation, and one the quantitative evaluation. In the qualitative evaluation, we compare the answers given in free text form from both the website and the VR parts as well as the final, comparative part. In the quantitative evaluation, we evaluate subjective scores of the questions presented on the 1-to-5 Likert scale [8] of Projectverse in comparison to the website in e.g. knowledge gains or physical and mental demand of both media.

5.2.1 Qualitative Evaluation

Preference In the qualitative evaluation, we analyzed and compared questionnaire responses to each two corresponding questions from the website and the VR part. In addition, we analyzed and summarized the responses given in the final comparative part. In this first section, we evaluated answers to questions about preference. These questions are shown in Table 5.1.

Q1 Participants responded that, in general, they were satisfied with the navigation options of the website and remarked a good, usable layout that provided good organisation and overview. However, one participant also mentioned confusion with the overview provided, whereas another commented the navigation to feel "clunky with weirdly placed arrow buttons". For initial thoughts on Projectverse, participants found the visualization and overall art style "like a cool old computer game", "quite entertaining", "really liked the galaxy design", and "[were] amazed by the visualization". On the other hand, we received feedback on the controls and VR, in general, to be initially overwhelming, but easy to get used to, and then also easy to use. It was also remarked that Projectverse provides a "much clearer [picture] of the projects by experiencing them [personally] instead of just reading about them". The need to reactivate the VR session after a jump was commented by one participant as "[introducing] a bit of friction".

Q2 Participants answered on Q2 that they would on the one hand require more pictures, whereas others remarked that the images provided on the website helped a "quicker understanding of the project". It was also mentioned that the core concepts of projects were explained on the website, however, one could not imagine "how the project would look like in the real world" from this representation of information. For Projectverse, one participant also mentioned the lack of pictures which would have helped to understand projects better. On the other hand, participants remarked at the large impact of the info boards in the project applications as well as the use of VR environments and "being able to navigate through the projects [themselves]" to understanding research projects. Two participants commented on the font size of the gun display being too small to read quickly. While participants remarked that on the website, they got an abstract understanding of projects, Projectverse provided them with quicker understanding. Furthermore, it was answered that "doing stuff [actively]" helped when using Projectverse.

Q3 Except for one participant, all found the website to be easier to use. This choice was explained by being used to navigating websites, higher intuitiveness and experience with them as well as an absence of motion sickness in this medium. Furthermore, the website was described as less demanding and as following conventions which for VR applications, "there aren't clear cut [conventions]", resulting in this participant having to re-read the controls. However, Projectverse was described as motivating due to its game-like design by one participant, whereas another found it easier to see something right in front of them and being able to grab it as compared to just read about it.

Q4 The majority of participants (7) found Projectverse to provide better understanding, while three participants answered the website here, and one did not have any preference. Participants justified their decisions for Projectverse by the fact that this medium enabled them to experience research projects first-hand, showing the "intentions and goals better". Furthermore, being "literally in the project" as well as being able to walk around in and trying out a research project application "made it [easier] to remember afterwards". It was also remarked that, while the website required imagination of how the projects looked like, Projectverse provided intuitive understanding and “conveyed things about the projects that weren’t even in the text”. Participants that found the website to provide them with a better understanding justified this by the “textual description” of projects being more accurate, the website in general being less distracting, and being more used to the medium. These participants found the website to be more detailed and to help “generate a greater knowledge of the project”.

Q5 All but one participant named the website to be more comfortable to use, whereas one chose Projectverse. This participant found Projectverse to be “more fun to use”. Among the others, VR was found to be more physically demanding, overwhelming and distracting. Furthermore, having to stand up and “wear a heavy device” along with being used to websites and their navigation were other reasons for finding the website more comfortable.

Q6 Here, six participants chose Projectverse, three the website, and two found no difference. When asked about their reasons, participants that chose Projectverse found it more challenging to use and get used to, requiring more interactions, work, concentration and time, and being more difficult to navigate. One participant that chose the website here justified this by the website leaving less of a mental imprint of project contents. Another one reasoned that, while Projectverse provided a dynamic experience, having to first read information on the website and then watch a video on it was more work-intense. Participants that had no preference did not justify their decision.

Q7 A majority of nine chose Projectverse here, compared to one participants deciding for the website and one having no preference. Reasons for choosing Projectverse were a better fit to the projects, information being clearer, a more fun experience and really getting to know the projects. Four participants specifically mentioned they had fun when trying out Projectverse. Two remarked that the domain of the information being the same as that of its conveying provided much better understanding. The participant that chose the website here remarked that to simply inform oneself, it is easier to use. Someone who wants to experience projects first hand should first read the website, then use Projectverse. The participant that had no preference states they would recommend both methods with one being purely theoretical, and the other offering a “hands-on” experience.

Q8+Q9 According to the answers, the website should be extended by pictures and references to Projectverse as well as changes to the navigation and structure. On the other hand, modifications to Projectverse should include images in the project descriptions on the gun display along with larger font sizes, sounds and voiceover, different locomotion methods (unspecified which) and a lower railing on the central play area. Furthermore, the controls should be unified among all project applications and described better, possibly by the use of a step-by-step tutorial.

ID	Question
Q11	Which research areas/categories do you remember [after exploring the projects]?
Q12	Who works on the [Conversational Agents (W) / BIM_Flexi (P)] project?
Q13	What is the [Conversational Agents (W) / BIM_Flexi (W + P)] project about?
Q14	Which new things did you learn about [BIM_Flexi] using Projectverse? (P)
Q15	Which new things did you learn about [Conversation Agents] using Projectverse? (P)
Q16	What can one see and do in the Haas VR project? (P)

Table 5.2: Questions used in the knowledge part of the qualitative evaluation.

Q10 Answers here mention higher motivation, a longer-lasting, more detailed memory of projects as well as a hands-on experience and quicker learning when participants informed themselves using Projectverse. Projectverse was also referred to as being more engaging and showing directly what was previously just readable information. One participant commented that VR projects "benefit tremendously from being able to try [them] out". The answers state that "The experience sticks with you for a longer period of time", while being more motivating than the website when it comes to reading about projects, which was reasoned in the "nice design and functions" of Projectverse. One participant remarked that Projectverse provided them with "a better imagination or gut feeling about a project", while also stating that the experience might stay longer in their memory than the text they read on the website.

Learning For the qualitative evaluation of the learning performance in the website and Projectverse, we analyzed questionnaire responses to the questions given in Table 5.2. Again, we compared answers to questions about both Projectverse and the website and summarized the responses. Some questions were asked only in regards to Projectverse, these are marked with (P), whereas the question variations we asked about the website are marked with (W).

Q11 Answers to this question were largely similar in both media. The number of categories or areas mentioned ranged between 0 and 7 for the website and 0 and 5 for Projectverse. Responses were generally short for both media, and there appears no major difference between the answers.

Q12 For both media a range between 0 and 2 names were answered. Participants answered the question similarly for both the website and Projectverse.

Q13 Participants reproduced the contents of the website well. Communication and text to speech were answered as contents of the Conversational Agents project. Furthermore, human-like conversations and urban VR visualisations were mentioned here, in addition to

ID	Question
Q17	How easy was it for you to find information about projects?
Q18	How intuitive were the menu and navigation options?
Q19	How many projects did you explore?
Q20	How many researcher names do you remember after using the [website/Projectverse]?
Q21	How mentally demanding was the task?
Q22	How physically demanding was the task?
Q23	How engaging was the task?

Table 5.3: Questions for which we performed the Wilcoxon Signed Rank Test.

the effect these improved conversations have to perception, immersion and presence. In the Projectverse question, participants answered that BIM_Flexi is about the visualization of industrial buildings and "the architecture and building process". One participant specifically named the miniature model of a building this project proposes, which can only be learnt from using Projectverse, since this information is not included in the website.

Q14 Participants answered that they learned how to use the teleport locomotion technique, how to transition between projects and how BIM_Flexi "can be applied to our (work) [lives]". Furthermore, participants stated to have learned how BIM_Flexi "is supposed to work" and that they could get "a good impression of how the actual buildings will look like as well as a sense of scale". One participant remarked that, although not having learned something new about the project, they could "explore what it looks like in VR", which, they stated, would definitely stay in their memory.

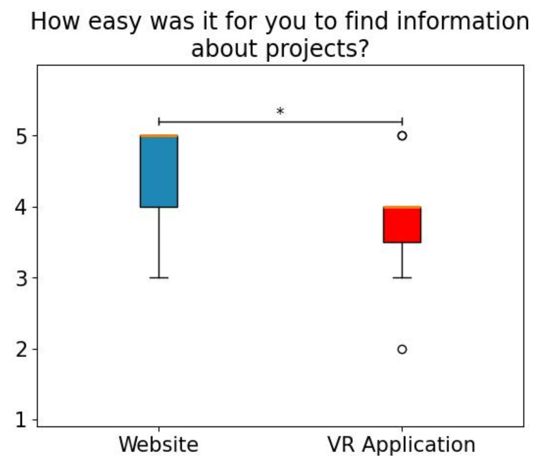
Q15 While 2 participants stated they learned nothing new and 2 did not answer at all, others answered that they learned that the project was "not only about language and conversation but also [emergency] situations" as well as aiming to recreate a first responder training scenario. One participant answered that they learned how the project can be applied in real-life situations, whereas another stated that they learned about the project being interactable and the experience providing them with "a much clearer, less vague outlook of what is expected from this project".

Q16 Answers include the visualization of an industrial projects, specifically, wafer making in a factory, with the use of a machine. Furthermore, the moving production line of this project was mentioned as well as the possibility to interact with parts of it.

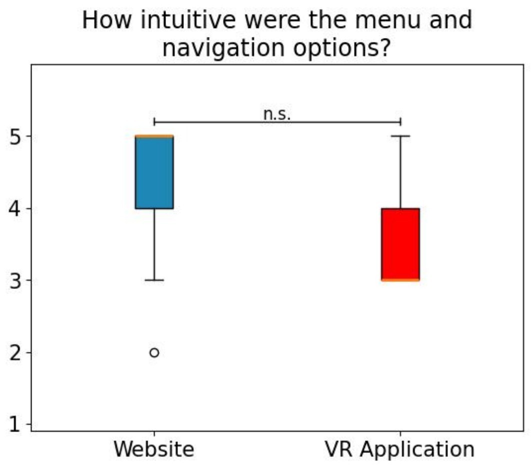
5.2.2 Quantitative Evaluation

For the quantitative evaluation of our questionnaire, we used the questions in Table 5.3. For all but Q19 and Q20, we asked participants to answer on the 1-to-5 Likert scale.

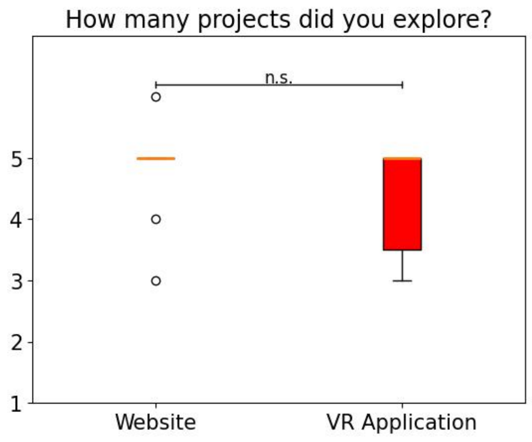
We compared questionnaire responses in respect to differences between the two media. For this, we used the Wilcoxon Signed Rank Test. The test details for those comparisons are shown in Table 5.4. The scores for these questions are shown in Figure 5.5. Due to our relatively small number of 11 participants, the results of these statistical tests should be treated as exploratory and indicating some first tendencies that might be investigated further in future research.



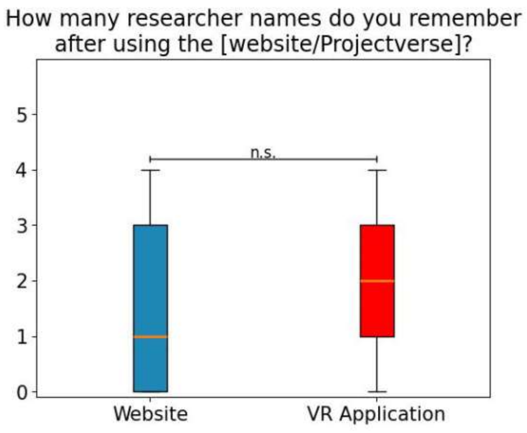
(a) Q17.



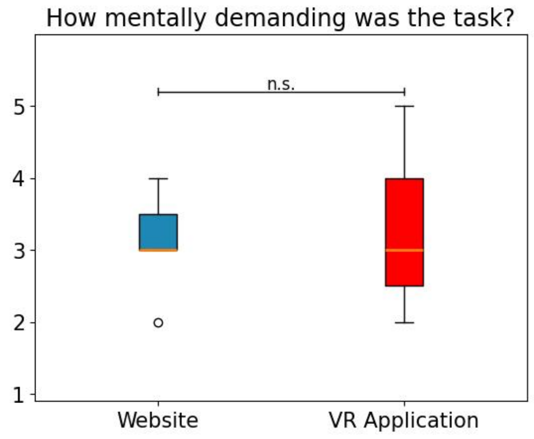
(b) Q18.



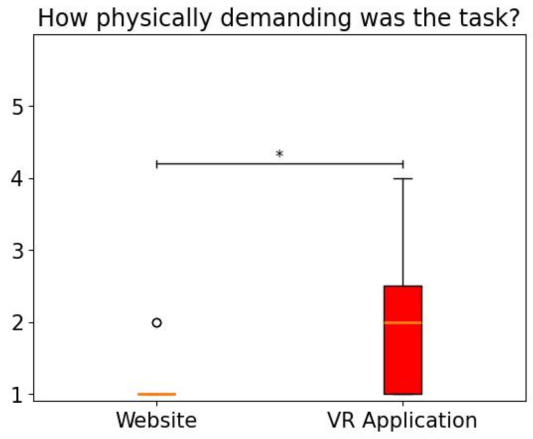
(c) Q19.



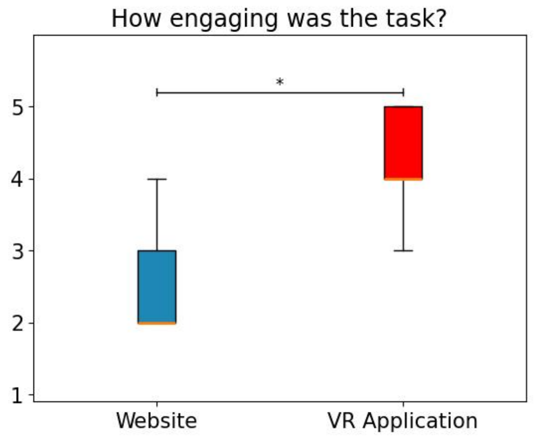
(d) Q20.



(e) Q21.



(f) Q22.



(g) Q23.

Figure 5.5: Boxplots of the question scores for Q17-Q23, with median values, interquartile ranges and outliers.

Q17+Q18 User friendliness and intuitiveness of the website was rated relatively high, compared to fairly lower ratings for Projectverse. The medians were 5 for the website and 4 and 3 for Projectverse in questions regarding these. We found a statistically significant effect for Q17, where participants rated the website easier to find information about projects.

Q20 Participants could remember between 0 and 4 researcher names after using either medium, with the median for the website at 1 and for Projectverse at 2. We found no statistically significant effect of the medium here. Participants did sometimes not reproduce names correctly, i.e., introduce spelling errors. Though, we still count misspelled

Question	Test Statistic	Standard Error	Standardized Test Statistic	p-Value
Q17	**36.000**	**6.364**	**-2.750**	**0.006**
Q18	11.000	9.552	-1.727	0.084
Q19	9.000	6.964	-1.292	0.196
Q20	10.500	3.623	0.828	0.408
Q21	15.500	5.690	0.264	0.792
Q22	**28.000**	**5.690**	**2.460**	**0.014**
Q23	**55.000**	**9.650**	**2.850**	**0.004**

Table 5.4: Results of the Wilcoxon Signed Rank Test. Statistically significant differences in bold.

names as correct answers. In any case, a slight increase in memory is expected after seeing the names for the second time.

Q21+Q22 Projectverse was rated higher than the website in terms of physical demand. The median was 3 for both media in mental demand and at 1 and 2 respectively, for physical demand in the website and Projectverse. We did not find a statistically significant effect of the medium on mental demand.

Q23 We found a statistically significant effect of the medium on how engaging the task was perceived. Projectverse was rated higher in this question, with medians at 2 and 4 for the website and Projectverse, respectively.

5.2.3 Participant Remarks and Observations

While supervising the experiment and accompanying participants, the experimenters took notes of verbal remarks as well as difficulties encountered by participants.

Questions about the control layout of the lobby and the project applications were frequent. On the other hand, having to restart the VR session after transitioning into a project demo was not an issue and carried out with ease by the participants. Also, some participants encountered difficulties with the virtual environment of one project application. Some remarked that the info panels with the controls within project applications were hard to read from the spawn point, whereas the controls layout itself was found to be irritating and causing motion sickness in one project application, where thumbstick movement was used. The grabbing mechanic in the lobby seemed unintuitive to some participants.

The process of entering and exiting project applications was quickly learned and used often and with ease. When a jump into the wrong application occurred, participants quickly recognized this and swiftly jumped back to the lobby and into the intended application. Participants seemed highly motivated to "play through" the environments they experienced.

Aside from the three projects that we specifically asked the participants to explore, most (8) also visited between one (2 participants) and two (6 participants) additional project applications out of curiosity.

5.3 Discussion

Our evaluation showed that although being more physically demanding, Projectverse provided a more engaging experience to participants when informing themselves about the work of the VR group. In addition, they personally reported better understanding of project contents due to the domain of Projectverse, which they perceived as the proper environment for the information we were trying to convey. We take from this, that Projectverse provides a better basis for learning about the work of the VR group. Despite the website being more comfortable to use, participants reported that Projectverse provided them with more information and a deeper understanding of projects, whereas the amount of work required was perceived more in case of Projectverse. Specifically, it seems that participants interpret “work” differently - as a combination of physical and mental effort. Mental effort seems to include both the demands of learning something new, e.g. how to use VR, as well as remembering information learned and encountering passive moments of learning that might be boring or require more focus because of their low-intensity nature.

To provide even better understanding of research projects, participants required pictures on the website alongside references to Projectverse on project pages. In Projectverse, improvements should be made to the UI, such that texts would be better readable. Furthermore, Projectverse could benefit from more sounds and voiceover conveying information, as well as changes to the play area resulting in less occlusion.

Most participants saw added benefit from Projectverse, reasons here ranged between directly experiencing projects, being more engaged or experiencing instead of reading. We conclude from this, that Projectverse is an adequate addition to the VR group website that “fills in the gaps” that are left when learning about the work of the VR group from just reading the website. The ability to experience research projects first hand, in an immersive environment, leads to an engaging experience that stuck with participants and - according to their answers - left a long-lasting memory of the content of research projects.

Our motivation to create Projectverse was to provide information about the VR group in an environment more suitable to the VR field than a traditional website. Judging by the results of our evaluation, we assume this purpose to be fulfilled in the platform we created. Motivation and engagement are essential factors to learning something new - by establishing an environment that provides fun experiences in which users enjoy spending time, we introduce these factors into the VR group website. As Pomykakala et al. [10] state, WebXR makes our platform accessible and “effective for a wide audience”. Furthermore, Projectverse is platform independent, which Sun et al. point out to be an important feature in learning applications [17].

Taken together, our results indicate that, while in some cases resulting in deeper knowledge and understanding, Projectverse mostly increases motivation and engagement. We consider this an important advantage over the website, since the aim of our work was to provide information to the general public, whose members wish to inform themselves about the VR group. Higher motivation leads to longer-lasting memory and - possibly - to more usage of Projectverse.

CHAPTER 6

Conclusion

6.1 Conclusion

In this thesis, we presented Projectverse, our VR platform for providing access to research project applications alongside a developer toolkit to extend it. To implement Projectverse, we used the WebXR Device API within the Unity game engine to use VR hardware in the web browser. Upon launching the application, users join a multiplayer session in the lobby. Here, they can inform themselves about research projects by using the laser gun tool to select stars that represent these projects and reading the information we provide. Furthermore, user can interact with others in the form of voice chat, and see other players represented by avatars. Having decided on a research project to explore further, users can choose to jump into it by again using the laser gun tool. Within a research project application, further information is provided via info boards. Project applications provide their own logic and experience.

In order to simplify the adaptation process of these applications, we provide the developer toolkit which includes basic VR functionality that utilizes the WebXR Device API together with prefabs making use of said functionality. Together with the documentation we provide, the developer toolkit allows researchers to suit their applications to the platform.

Our project shows that the medium we choose to present the research work of the VR group supports the information we aim to convey. The user study suggests that participants were more motivated and engaged while using our application when compared to the traditional approach of a website. We take from this that our work provides a useful addition to the VR group website specifically, and to information platforms in general. In uniting tried-and-tested technologies like Unity with new additions to the VR field - WebXR - we implemented a long-lasting platform that has future potential: New, yet to be released hardware will be usable with Projectverse, whereas the platform itself can be adapted to any field or use case that could benefit from a hands-on, engaging, experience. Future work of the VR group can easily be added to Projectverse, which ensures frequent use and a rich environment.

6.2 Future Extensions

As the WebXR Device API at the time of the development of our platform does not support VR sessions that persist over multiple web pages, we had to implement the switch between application such that a reactivation of the VR sesstion was necessary whenevery such a switch takes places, i.e., when jumping into a project and when jumping back to the lobby. The "navigation" repository[1] of immersive web, the group maintaining WebXR, suggests that with navigation, a feature is planned that persists VR sessions while navigating between pages. With this feature eventually implemented, our platform can be extended to make use of it in order to avoid the reactivation of VR sessions upon jumping.

Another future possibility of improvement is indicated by participant comments during the user study, suggesting an adaptation of some features to improve user experience. These include the placement of info boards within project applications in a standardized way, as well as increasing font sizes on the laser gun as well as on the info boards.

Finally, the adaptation process could be automated to at least some degree, e.g. by removing old VR APIs and frameworks from projects automatically as well as importing our toolkit and replacing the VR device representation.

Altogether, we envision a fully-functional VR environment that allows for uninterrupted VR sessions on the VR group website. Research project applications in this environment provide feature-complete functionality, such that the applications on Projectverse completely cover all user actions available in their - possibly Windows-built - counterparts. Since multiplayer is already part of Projectverse, this feature could be expanded to allow for multi-user sessions within project applications. With more and more research project applications being deployed to it, the future sky of Projectverse will be full of stars and provide a rich environment to learn, explore and enjoy together.

Projectverse is available at `https://www.vr.tuwien.ac.at/projectverse/`.

[1] https://github.com/immersive-web/navigation

List of Figures

List of Tables

List of Algorithms

Bibliography

[1] Valentin Benzing and Mirko Schmidt. Exergaming for children and adolescents: Strengths, weaknesses, opportunities and threats. *Journal of clinical medicine*, 7 (11), 2018. ISSN 2077-0383. doi: 10.3390/jcm7110422.

[2] Evren Bozgeyikli, Andrew Raij, Srinivas Katkoori, and Rajiv Dubey. Point & teleport locomotion technique for virtual reality. In Anna Cox, Zachary O. Toups, Regan L. Mandryk, and Paul Cairns, editors, *Proceedings of the 2016 Annual Symposium on Computer-Human Interaction in Play*, pages 205–216, New York, NY, USA, 2016. ACM. ISBN 9781450344562. doi: 10.1145/2967934.2968105.

[3] Manuela Chessa, Chiara Bassano, and Fabio Solari. A webgl virtual reality exergame for assessing the cognitive capabilities of elderly people: A study about digital autonomy for web-based applications. In Alberto Del Bimbo, Rita Cucchiara, Stan Sclaroff, Giovanni Maria Farinella, Tao Mei, Marco Bertini, Hugo Jair Escalante, and Roberto Vezzani, editors, *Pattern Recognition. ICPR International Workshops and Challenges*, volume 12662 of *Lecture Notes in Computer Science*, pages 163–170. Springer International Publishing, Cham, 2021. ISBN 978-3-030-68789-2. doi: 10.1007/978-3-030-68790-8{\textunderscore}14.

[4] Adrian Ciprian Firu, Alin Ion Tapîrdea, Anamaria Ioana Feier, and George Drăghici. Virtual reality in the automotive field in industry 4.0. *Materials Today: Proceedings*, 45:4177–4182, 2021. ISSN 22147853. doi: 10.1016/j.matpr.2020.12.037.

[5] Nuha El-Khalili and Ken Brodlie. Architectural design issues for web-based virtual reality training systems. 1998.

[6] Mhanaj Hossain, Daphne Economou, and Jeffrey Ferguson. Work-in-progress-webxr to support student wellbeing and anxiety. In *2021 7th International Conference of the Immersive Learning Research Network (iLRN)*, pages 1–3. IEEE, 2021. ISBN 978-1-7348995-2-8. doi: 10.23919/iLRN52045.2021.9459324.

[7] Kate E. Laver, Belinda Lange, Stacey George, Judith E. Deutsch, Gustavo Saposnik, and Maria Crotty. Virtual reality for stroke rehabilitation. *The Cochrane database of systematic reviews*, 11(11):CD008349, 2017. doi: 10.1002/14651858.CD008349.pub4.

[8] R. Likert. *A Technique for the Measurement of Attitudes*. A Technique for the Measurement of Attitudes. Archives of Psychology, 1932. URL `https://books.google.at/books?id=9rotAAAAYAAJ`.

[9] Mozilla Corporation. Webgpu api - web apis | mdn, 19.08.2023. URL `https://developer.mozilla.org/en-US/docs/Web/API/WebGPU_API`.

[10] Radoslaw Pomykakala, Artur Cybulski, Tadeusz Klatka, Michal Patyk, Julia Bonieckal, Maciej Kedzierski, Mateusz Sikora, Jakub Juszczak, and Magdalena Igras-Cybulska. "put your feet in open pit" - a webxr unity application for learning about the technological processes in the open pit mine. In *2022 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*, pages 493–496. IEEE, 3/12/2022 - 3/16/2022. ISBN 978-1-6654-8402-2. doi: 10.1109/VRW55335.2022.00110.

[11] R. C. Prim. Shortest connection networks and some generalizations. *The Bell System Technical Journal*, 36(6):1389–1401, 1957. doi: 10.1002/j.1538-7305.1957.tb01515.x.

[12] Warren Robinett and Richard Holloway. Implementation of flying, scaling and grabbing in virtual worlds. In Marc Levoy, Edwin E. Catmull, and David Zeltzer, editors, *Proceedings of the 1992 symposium on Interactive 3D graphics - SI3D '92*, pages 189–192, New York, New York, USA, 1992. ACM Press. ISBN 0897914678. doi: 10.1145/147156.147201.

[13] Vinit Sathe, Piyush Gupta, Karan Kaushik, Suvarna Bhat, and Sachin Deshpande. Virtual reality websites(vr web). In *2017 International conference of Electronics, Communication and Aerospace Technology (ICECA)*, pages 647–652. IEEE, 2017. ISBN 978-1-5090-5685-9. doi: 10.1109/ICECA.2017.8203619.

[14] Hayeon Song, Wei Peng, and Kwan Min Lee. Promoting exercise self-efficacy with an exergame. *Journal of health communication*, 16(2):148–162, 2011. doi: 10.1080/10810730.2010.535107.

[15] J. Sorger, A. Arleo, P. Kán, W. Knecht, and M. Waldner. Egocentric network exploration for immersive analytics. *Computer Graphics Forum*, 40(7):241–252, 2021. ISSN 0167-7055. doi: 10.1111/cgf.14417.

[16] Soumil Srivastava. Webgl: The new standard for 3d graphics on the web - ar/vr journey: Augmented & virtual reality magazine. *AR/VR Journey: Augmented & Virtual Reality Magazine*, 20.05.2020. URL `https://arvrjourney.com/webgl-the-new-standard-for-3d-graphics-on-the-web-2d8e206e7ef0`.

[17] Fei Sun, Zhaochuang Zhang, Dunming Liao, Tao Chen, and Jianxin Zhou. A lightweight and cross-platform web3d system for casting process based on virtual reality technology using webgl. *The International Journal of Advanced Manufacturing Technology*, 80(5-8):801–816, 2015. ISSN 0268-3768. doi: 10.1007/s00170-015-7050-1.

Appendix

Appendix A

ProjectVerse XR - User Study

Welcome to the ProjectVerse XR User Study! This study is divided into three parts, in which we will ask you to do the following things:

1. General Questions: You are already on the general questions page. Here, we ask you to just answer some simple questions about yourself.
2. Website Version: You will be asked to navigate to a website here, on which you will learn about the work of TU Wien's Virtual and Aurmented Reality Research Group. After having visited the website, you will be asked to answer some questions about it.
3. VR Version: You will be asked to use a VR application that runs in the web browser. Here, you will also learn about the work of TU Wien's Virtual and Augmented Reality Group by exploring immersive projects. You will answer questions about the ProjectVerse website afterwards.

This user study will take approximately 45 minutes - 1 hour. Thank you for participating!

General Questions

1. What is your Age?

____

2. What is your level of experience using Virtual Reality Headsets?

Mark only one oval.

	1	2	3	4	5	
Non	◯	◯	◯	◯	◯	Frequent Use

3. What is your level of experience with video games?

Mark only one oval.

	1	2	3	4	5	
Non	◯	◯	◯	◯	◯	Frequent Use

4. What is your gender?

Mark only one oval.

- ⬭ Female
- ⬭ Male
- ⬭ Other: ______________________________

Website Version

In this section, you will use the website of Virtual and Augmented Reality Group to find out about our current research projects, their contents and the researchers working on them.

Please visit Research Unit Virtual & Augmented Reality – TU Wien and familiarize yourself with different projects that can be found in the "Projects" section in the first third of the page.

Please visit at least 5 project pages but make sure that "BIM_Flexi", "Conversational Agents" and "Haas VR" are among them. You will be asked questions about these projects.

Once you are done browsing the website, please close it, and only then answer the following questions.

Website Version

In this section, you will use the website of Virtual and Augmented Reality Group to find out about our current research projects, their contents and the researchers working on them.

Please visit Research Unit Virtual & Augmented Reality – TU Wien and familiarize yourself with different projects that can be found in the "Projects" section in the first third of the page.

Please visit at least 5 project pages but make sure that "BIM_Flexi", "Conversational Agents" and "Haas VR" are among them. You will be asked questions about these projects.

Once you are done browsing the website, please close it, and only then answer the following questions.

5. What were your initial impressions when using the application? (Please write at least two sentences)

6. How easy was it for you to find information about projects?

Mark only one oval.

	1	2	3	4	5	
Very	◯	◯	◯	◯	◯	Very easy

7. How intuitive were the menu and navigation options?

Mark only one oval.

	1	2	3	4	5	
Very	◯	◯	◯	◯	◯	Very intuitive

8. Did you get enough information to understand a project?

Mark only one oval.

◯ Yes

◯ No

9. If not, what did you miss? If yes, what was especially important for your understanding? (Write at least one sentence)

Website - specific questions

In this section, you will be asked about some details that your learned from the projects website.

10. How many projects did you explore?

11. How many researcher names do you remember after exploring the website?

12. Which research categories do you remember after exploring the projects? (Please enter some keywords you connect to the projects)

13. Who works on the "Conversational Agents" project? Please add names of the researchers that you remember.

14. What is the "Conversational Agents" project about? (Please write at least two sentences)

15. What is the "BIM_Flexi" project about? (Please write at least two sentences)

16. What is the research area of "Haas VR"?

Website - workload questions

17. How mentally demanding was the task?

Mark only one oval.

	1	2	3	4	5	
Very	◯	◯	◯	◯	◯	Very High

18. How physically demanding was the task?

Mark only one oval.

	1	2	3	4	5	
Very	◯	◯	◯	◯	◯	Very High

19. How engaging was the task?

Mark only one oval.

	1	2	3	4	5	
Very	◯	◯	◯	◯	◯	Very Enganging

VR Version

In this section, you will use TU Wien's VR Group's ProjectVerse Browser Application to find out about our current research projects, their contents and the researchers working on them.

Please visit ProjectVerse XR – Research Unit Virtual & Augmented Reality (tuwien.ac.at) and familiarize yourself with different projects that can be found on the application. You will need to use Oculus Quest connected to your PC with the Link cable.

Please explore ("jump into") at least 5 projects but make sure to visit the "BIM_flexi", "Haas VR" and "Conversational Agents" projects. You will be asked questions about these projects. Also make sure to use the category switching button in the application.

Please also familiarize yourself with information about the usage of ProjectVerse on the application website. Controls instructions are provided as well.

Use the application in VR mode and try jumping into projects. When in the Oculus Home environment, click on the Destkop button to see your browser window in VR. This way you can activate VR for each project demo directly from Oculus Home, without taking off the VR glasses.

Once you are done using the application, please close it, and only then answer the following questions.

VR Version

In this section, you will use TU Wien's VR Group's ProjectVerse Browser Application to find out about our current research projects, their contents and the researchers working on them.

Please visit [ProjectVerse XR – Research Unit Virtual & Augmented Reality (tuwien.ac.at)]() and familiarize yourself with different projects that can be found on the application. You will need to use Oculus Quest connected to your PC with the Link cable.

Please explore ("jump into") at least 5 projects but make sure to visit the "BIM_flexi", "Haas VR" and "Conversational Agents" projects. You will be asked questions about these projects. Also make sure to use the category switching button in the application.

Please also familiarize yourself with information about the usage of ProjectVerse on the application website. Controls instructions are provided as well. **Use the application in VR mode and try jumping into projects. When in the Oculus Home environment, click on the Destkop button to see your browser window in VR. This way you can activate VR for each project demo directly from Oculus Home, without taking off the VR glasses.**

Once you are done using the application, please close it, and only then answer the following questions.

20. What were your initial impressions when using ProjectVerse? (Please write at least two sentences)

____

____

____

____

____

21. How easy was it for you to find information about projects?

Mark only one oval.

	1	2	3	4	5	
Very	◯	◯	◯	◯	◯	Very easy

22. How intuitive were the menu and navigation options?

Mark only one oval.

	1	2	3	4	5	
Very	◯	◯	◯	◯	◯	Very intuitive

23. Did you get enough information to understand a project?

Mark only one oval.

◯ Yes

◯ No

24. If not, what did you miss? If yes, what was especially important for your understanding? (Write at least one sentence)

____

VR - specific questions

In this section, you will be asked to reproduce information gathered from the VR application.

25. How many projects did you explore?

____

26. How many researcher names cany you remember after visiting ProjectVerse?

____

27. Which research areas/categories do you remember?

____

28. Who works on the "BIM_Flexi" project?

29. What is the "BIM_Flexi" project about? (Please write at least two sentences)

30. Which new things did you learn about this project using ProjectVerse? (Please write at least two sentences)

31. What is the "Conversational Agents" project about? (Please write at least two sentences)

32. Which new things did you learn about this project using ProjectVerse? (Please write at least two sentences)

33. What can one see and do in the "Haas VR" project? (Please write at least one sentence)

VR - workload questions

34. How mentally demanding was the task?

Mark only one oval.

	1	2	3	4	5	
Very	◯	◯	◯	◯	◯	Very High

35. How physically demanding was the task?

Mark only one oval.

	1	2	3	4	5	
Very	◯	◯	◯	◯	◯	Very High

36. How engaging was the task?

Mark only one oval.

	1	2	3	4	5	
Very	◯	◯	◯	◯	◯	Very Enganging

Comparison

In this section, you will be asked questions about which platform better presents research projects to the general audience.

37. Which medium was easier in terms of navigation and interaction?

Mark only one oval.

- ◯ Website
- ◯ VR Application
- ◯ No preference

38. Why was this medium easier? (Write at least two sentences)

39. Which medium provided you with a better understanding of project contents?

Mark only one oval.

- ◯ Website
- ◯ VR Application
- ◯ No preference

40. Why did this medium provide you with a better understanding? (Write at least two sentences)

41. Which medium was more comfortable to use for an extended period?

Mark only one oval.

- ⬭ Website
- ⬭ VR Application
- ⬭ No preference

42. Why was this medium more comfortable to use? (Write at least two sentences)

43. Which medium required more work from you to find out information about the projects?

Mark only one oval.

- ⬭ Website
- ⬭ VR Application
- ⬭ No preference

44. Why did this medium require more work? (Write at least two sentences)

45. Which medium would you recommend to others who wish to find out about the VR Group's work?

Mark only one oval.

- Website
- VR Application
- No preference

46. Why would you recommend this medium? (Write at least two sentences)

47. Which, if any, modifications would you like to see in the website? (Write at least one sentence)

48. Which, if any, modifications would you like to see in the ProjectVerse? (Write at least one sentence)

49. Do you see any added benefit to exploring projects through ProjectVerse? If so, what is this benefit?

This content is neither created nor endorsed by Google.

Google Forms