Virtual Reality Application in Data Visualization and Analysis

Master's Thesis

Written by

Xinzhou Zhang

Submitted at

TH Köln

Campus Gummersbach

Faculty of Computer Science and Engineering Science

under the Study Program

Information Systems

Matriculation number: 11096715

First Examiner: **Prof. Dr. Heide Faeskorn-Woyke**

TH Köln

Second Examiner: **Thies Smeding-Terveer**

PricewaterhouseCoopers GmbH WPG

Gummersbach, June 30th, 2017

Restriction Note

Table of Contents

List of Figures **7**

Acronyms **12**

Abstract **13**

Introduction **14**

Part 1 VR General Background **15**

1.1 What Is Virtual Reality **16**

1.2 Current Technology Offerings and Market Profile **19**

1.2.1 Technologies 21

1.2.1.1 Hardware and Platforms 21

1.2.1.2 Software 24

1.2.1.2.1 Contents 24

1.2.1.2.2 Development 28

1.2.1.3 Other Technologies and Platforms 30

Eye-tracking 30

Hand as an Input Device 31

Nosulus Rift 32

WebVR 33

1.2.2 Market Profiling 35

1.2.2.1 Use Cases 38

Part 2 VR Data Visualization Design **41**

2.1 VR User Experience Design **43**

2.1.1 Adverse Health Effects 44

2.1.1.1 Motion Sickness and Factors 44

Latency 46
2.1.1.2 Countermeasures 48
2.1.1.3 Summary 49
2.1.2 Content Design 50
2.1.2.1 Environment Elements 51
Scenes 51
Color and Lighting 53
Audio 53
Environmental Wayfinding 54
Real-world Contents 55
2.1.2.2 User Elements 56
Personal Wayfinding 56
Action Areas 57
2.1.2.3 Summary 61
2.1.3 Interaction Design 62
2.1.3.1 VR Interaction Concepts 63
Interaction Fidelity 63
Reference Frame 64
Speech and Gestures 66
Multimodal Interaction 66
2.1.3.2 VR Interaction Patterns & Techniques 68
Selection Pattern 68
Manipulation Pattern 69
Viewpoint Control Pattern 71
Indirect Control Pattern 73
Compound Pattern 74
2.1.3.3 Summary 76
2.2 3D Data Visualization 77

2.2.1 Data Visualization Principle 78
2.2.2 Data Visualization in 3D World 80
2.2.3 Summary 86
2.3 VR Social and Collaboration 87
Part 3 Project Implementation 91
3.1 Introduction 92
3.2 Hardware Solution 93
3.3 Development Environment Setup 95
3.3.1 OpenVR and SteamVR 98
3.3.2 VRTK and NewtonVR 100
3.4 Application Walkthrough 103
3.4.1 Version Poweruser 104
3.4.2 Version Showcase 113
3.5 Prototype Architecture 121
3.5.1 Application Architecture 121
3.5.1.1 Client 121
Unity Basics 122
Controller 130
Manager 140
3.5.1.2 Server 148
Controller 149
Data Model 152
Database 153
Back to the Controller 162
3.6 Summary of Project Implementation 163
Part 4 Conclusion and Outlook 165

References **168**

Declaration **176**

List of Figures

Figure 1 Current market offerings 21

Figure 2 Play areas of different VR systems 23

Figure 3 One could have a view on the current "scene" where the objects are placed in a way accordingly and other assets well organized. 29

Figure 4 Bare hand input from Leapmotion 31

Figure 5 Nosulus Rift simulates the smell 33

Figure 6 Base case forecast from Goldman Sachs 35

Figure 7 Software revenue estimates by VR and AR 36

Figure 8 World-wide VR revenue prediction in from year 2014 to 2018 37

Figure 9 VR/AR use case estimates by 2025 38

Figure 10 VR use case survey 39

Figure 11 VR system end-to-end delay 46

Figure 12 Elements like routes, markers on maps helps with spatial comprehension and providing constrains and boundaries 55

Figure 13 Wayfinding technique in Google Earth VR 57

Figure 14 The action areas with different focuses 58

Figure 15 Different interaction areas for bare-hand controller - above 58

Figure 16 Different interaction areas for bare-hand controller - 3D 59

Figure 17 A good balance between high and low interaction fidelity yields a "magical interaction" 64

Figure 18 A volume-based selection pattern helps with selecting objects in 3D world 69

Figure 19 Proxy manipulation pattern allows user to manipulate a large object easily 71

Figure 20 3D multi-touch view point controls 73

Figure 21 World-in-miniature pattern combines proxy, multi-touch and automation interaction patterns 75

Figure 22 The data visualization pipeline 78

Figure 23 Movements of a smartphone over seven days, as captured by the Backitude application for Android, sampling at 15 second intervals, and subsequently visualized with Google Latitude. Gaps and discontinuous jumps in the data are caused by traveling through 80

Figure 24 Two design options of the spatiotemporal visualization. A and B show a "moving plane": the ground plane, red reference line, and vertical position of the green icon all move downward as time advances. C and D show a "moving trajectory": the ground plane, red reference line, and vertical position of the green icon are all fixed, but the trajectory moves upward as time advances. A and C show Monday morning at 10:00, B and D show Monday evening at 23:25 81

Figure 25 A 3D visualization in VR visualizes 8 dimensions of a dataset 82

Figure 26 3D visualization of OLAP cube 83

Figure 27 3D object conveys less effective information 84

Figure 28 Perspective distorts the size of the object and the objects can block each other in the view 84

Figure 29 Cutting plane and cutting cube is helping to solve problems brought by 3D visualization 85

Figure 30 The Uncanny Valley should be avoided in the relationship between familiarity and human likeness of an object 89

Figure 31 Taking selfies with video calls in VR by Facebook 90

Figure 32 Oculus Avatar SDK that embodies the users in a projection metaphor 90

Figure 33 Unity editor provides an integrated user interface 95

Figure 34 . Base SDKs connect the hardware and Unity assets or packages provides methods for high-level reference 99

Figure 35 Interaction library provides high-level interaction implementations which are decoupled from the vendors 101

Figure 36 The application starts in a familiar office scene 104

Figure 37 User can toggle on/off the on-controller GUI and use the laser pointer to interact with it 105

Figure 38 After user selects what and how the data to be visualized and presses the confirm button, the data is visualized 106

Figure 39 User can change what dimension to be visualized and refresh the data 107

Figure 40 User can also change the type of the map object on which the data points is placed 108

Figure 41 User can use the controller to manipulate the perspective of the map object so that the data points can be observed from another angle 108

Figure 42 The legend that indicates two dimensions of the data points: height and color hue .. 109

Figure 43 When user points at the data point using the laser pointer, the detail information will be toggled on..... 110

Figure 44 The detail information of the data point is also shown on the controller so that the user can see it easily from the hand 110

Figure 45 Teleportation and HMD tracking allow user to change the viewpoint easily 111

Figure 46 A start scene with a robot drone 113

Figure 47 A simplified version of GUI..... 114

Figure 48 A 3D bar chart shows a survey result which has multiple dimensions 116

Figure 49 A 2D version of the same set of data from the survey: “Erwartungen durch technologischen Wandel” 117

Figure 50 User can easily identify the category dimension and its outstanding values 118

Figure 51 User can easily identify distribution of value dimension by changing the perspective or the point of view 119

Figure 52 User can easily identify distribution patterns by moving the point of view intuitively 120

Figure 53 A client-server architecture with MVC design pattern..... 122

Figure 54 A GameObject as light 123

Figure 55 A GameObject as a normal cube 124

Figure 56 The structure of Unity’s classes 126

Figure 57 A GameObject that has its own behavior script and integrates with the VRTK behaviors 129

Figure 58 Overview of the operations between Controller and Manager..... 131

Figure 59 Register listeners in Awake() 132

Figure 60 Initiate variables and Manager references in Start()..... 133

Figure 61 MapObjectFactory will create a map object after the map is downloaded 135

Figure 62 OnGeoChartDataUpdated() called after data are returned from server..... 137

Figure 63 RenderGeoChartDataOnMapObject() will finally visualize the data in the scene..... 139

Figure 64 Managers are responsible of loading data from network by utilizing NetworkService 141

Figure 65 GeoChartManager utilizes NetworkService to load data that are to be visualized 143

Figure 66 RefreshButtonOnClick() triggers map object creation and data loading 145

Figure 67 After the map object is created, RefreshGeoChart() is triggered to load the data from backend by passing parameters to the server 146

Figure 68 A data model that will form the JSON format data which received from server 147

Figure 69 An overview of the server and its communication with client 148

Figure 70 BuildChart() method takes in the parameters to generate dynamic SQL and forms the result as a data transfer object 150

Figure 71 Other actions in the controller of server 151

Figure 72 GeoChartDataModelDto entity class 152

Figure 73 DataSourceProperty entity class 152

Figure 74 DataSource entity class 153

Figure 75 Database context class in .NET MVC framework 153

Figure 76 Current data tables in the database 154

Figure 77 The stored procedure that generate an intermediate @temp table by executing a dynamic SQL query 156

Figure 78 Prepare and reform the data that are to be selected 158

Figure 79 Form the result to match the data model 161

Acronyms

API	Application Program Interface
AR	Augmented Reality
ASW	Asynchronous Spacewarp
CPU	Central Processing Unit
DoF	Degree of Freedom
FoV	Field of View
FPS	Frame Per Second
GUI	Graphical User Interface
GDP	Gross Domestic Product
HCI	Human Computer Interactions
HMD	Head-Mounted Display
IDE	Integrated Development Environment
JSON	JavaScript Object Notation
MVC	Model-View-Controller
OLAP	Online Analytical Processing
SDK	Software Development Kit
SQL	Structured Query Language
TAM	Total Available Market
UI	User Interface
VR	Virtual Reality
VRTK	Virtual Reality Toolkit
XML	Extensible Markup Language

Abstract

This thesis is aimed for finding a solution for non-gaming application of Virtual Reality technology in data visualization and analysis. Starting by reconstructing the concept of Virtual Reality, the paper then describes the principles, concepts and techniques of designing a Virtual Reality application. In the last part of the thesis, a detailed description of how a prototype implemented is presented to provide a preview of how data visualization and analysis and Virtual Reality technology can be combined together in order to enable users to perceive and comprehend data in a possibly better way.

Introduction

Virtual Reality is on hype. Although this technology was imagined and invented a long time ago, it is recently that it becomes one of the most essential technologies emerging from the horizon of future computing[1]. It may not be surprising that this technology will be used in gaming, entertainment, and advertising industries, but what are the other possibilities out there for this technology to thrive? How should Virtual Reality be designed and applied to business usages? What could be a way to visualize and analyze business data using Virtual Reality? Those are the questions this thesis will try to answer. But before answering how to apply this emerging technology into those non-entertainment areas, it is essential to rethink what this technology is all about. Therefore, the discussion on how to comprehend this technology and the forms of this technology is firstly introduced. Then the thesis will try to find a way to design the application before going into discussions about how to implement virtual reality in a non-entertainment scenario. Hence the second part will focus on the design principles of virtual reality and the third part will show a functioning prototype application which incorporates the design guidelines and considerations. Finally, through the implementation of the application, the thesis will be able to give answers to the questions mentioned above and based on the implementation, it will project an outlook on the future development of the virtual reality application in data visualization and analysis.

---

[1] "2017 Global Digital IQ Survey: Virtual Reality."

Part 1 VR General Background

1.1 What Is Virtual Reality

Since the introduction from Oculus, Virtual Reality has been talked a lot recently. According to a survey from PricewaterhouseCoopers, in a sample of 1057 interviewees, 84.3% of them have at least heard about this term[2]. In a few years many manufactures have introduced their own solution packages, products or devices into the market. Among them are also some big companies. Google, Samsung and Facebook have been pushing the boundaries of the definition of this emerging hype further and further. To most people nowadays, this technology seems to be novel, but actually it is not completely new. Dating back to 1960s, a VR system was firstly created[3]. Back in 1989, Webster's New Universal Unabridged Dictionary[4] defines virtual as "being in essence or effect, but not in fact" and reality as "the state or quality of being real. Something that exists independently of ideas concerning it. Something that constitutes a real or actual thing as distinguished from something that is merely apparent." In this sense, virtual reality sounds confusing and contradicted. However, a more modern definition to this word comes from Merriam-Webster[5], it is "an artificial environment which is experienced through sensory stimuli (such as sights and sounds) provided by a computer and in which one's actions partially determine what happens in the environment." This is already a decent definition, but there is another that provides a special perspective. According to Jerald in The VR Book, virtual reality is considered as a way to communicate, it is "defined to be a computer-generated digital environment that can be experienced and interacted with as if that environment were real."[6] In this way, an ideal VR system will provide

---

[2] Ballhaus, Bruns, Deligios, Graber, Kammerling, Lorenz, Schink, Wipper, and Wilke, "Digital Trend Outlook 2016."
[3] Jerald, *The VR Book: Human-Centered Design for Virtual Reality*, 9.
[4] *Webster's New Universal Unabridged Dictionary*.
[5] "Definition of VIRTUAL REALITY."
[6] Jerald, *The VR Book: Human-Centered Design for Virtual Reality*, 9.

understanding to the user as an ideal medium to offer immersive experience to the virtual objects. It will connect a range of entities by providing multiple modalities and stimuli and therefore, does a much more appealing impression to the user comparing to other forms of communication. So, it becomes clear that VR will be an ideal way to reveal meaningful information enclosed in a fussy dataset because it brings the audience one more dimension closer to the nature of the data, hence to unveil the useful information that can be much better understood by the end user.

By taking the advantage of the immersive experience, it is natural that this technology will be firstly brought to the industries or areas that aim to provide stunning perceptions to the audience such as gaming, entertainment, education, professional training or exhibition, just to name a few. However, VR should not be just be limited to those areas. Since it also provides a world of perception to the audience and reveals even deeper insights of fussy data, it could also be applied to scientific researches to visualize large datasets that would be otherwise hard to comprehend. What interests this thesis, is the potential of applying VR technology in business sectors, especially in visualizing and reporting of business data.

Already are there applications existing in the marketing departments as it is apparently an eye catcher to customers when using this technology to introduce new products. For example, car manufacture like BMW offers virtual test drive[7] on specific car models and IKEA offers a virtual kitchen that allows potential customers to have a "real" feeling on how their furniture can fit with each other and how they would work with customized setups[8]. However, when going up along the

[7] "BMW I Samsung Virtual Reality Experience."
[8] "Virtual Reality - IKEA."

value chain, it is still a green land waiting for VR to prosper. For example, in the traditional business intelligence applications, after all the data having been collected, cleaned, transformed and well aligned, multi-dimensional data models will need to be presented to the end user in forms of reports, dashboards, infographics so on and so forth. Users with different levels of knowledge and positions will need to have different viewpoints on those data and they will have various ways to manipulate the presentation of data. This demand aligns well with the characteristics of VR technology. Since the technology can provide immersive way to perceive the data, learning from the datasets will become considerably more intuitive and convenient. And since the presentation of data can be manipulated easily, understanding and analyzing the information underlying it from different perspectives and viewpoints will also be much easier. This can be one of the joining points of business analytics and VR technology applications. It can be called immersive business analytics. Another aspect of merging business data analytics and VR technology will be a concept that looks at the analysis of the VR technology itself. Since more sensors like haptic mapping and eye tracking technologies are being continuously integrated, there will be a tremendous amount of sensor data available and to be collected, thus providing a possibility to leverage those tracking and mapping data to gain the insights from how the users operate the VR devices and furthermore to support the user behavior analysis on specific products or service scenarios.

To summary, Virtual Reality as a new form of human computer interaction medium with immersive experience, can provide a new way for consumers and enterprise to communicate with data and business. When combined with analytics functionalities, VR would help people understand and interact with information and knowledge better[9].

---

[9] Jerald, *The VR Book: Human-Centered Design for Virtual Reality*, 12-13.

1.2 Current Technology Offerings and Market Profile

Although Virtual Reality is not a completely new technology and concept, its marketplace has just opened to welcome the first wave of mass customers. Dated back to the 1990s, there were already a few existing products on the market but since the technology then was not as powerful as today and the price was high enough to keep most of the customers away. At that time, a pair of new VR glasses costed as good as a new computer and the resolution of the image was just high as 384 x 224 pixels[10]. Therefore, the first try of pushing VR to the market was failed. However, in 2012, a company named Oculus appeared on Kickstarter.com and made VR back on the stage in a modern plot.

2 years later in 2014, Facebook acquired Oculus and Google published Cardboard. In 2015, Samsung launched Gear VR together with Oculus, the VR headsets were available across the mobile platform and more powerful desktop platform. In the first half of 2016, Oculus Rift was available to customers and more companies came in to market offering their solutions in VR. In December 2016, Oculus also made its product line complete as it shipped its controller Oculus Touch and its room-scale VR solution. As this thesis being written, Oculus was trying to find a new product category as a device that comes between smartphone and desktop so that this device will be able to provide more power than smartphone + headset solution and mobility that the desktop solution will not deliver. HTC also debuted its Vive to compete with Oculus Rift. Aside from that, Sony announced its PlayStation VR to supplement its successful console platform, PlayStation 4, and it became available to customers on October 2016. It is also worth of mentioning

---

[10] Ballhaus, Bruns, Deligios, Graber, Kammerling, Lorenz, Schink, Wipper, and Wilke, “Digital Trend Outlook 2016.”

that Microsoft also released its AR (Augmented Reality) product HoloLens. With AR, the headset can place a computer-generated layer onto the real physical environment, instead of creating a completely digital immersive environment as in VR. However, AR is not the focus of this thesis.

With all these big companies stepping in to take the lead and pushing the VR technology, what are the mainstream technologies available right now in the market? How would this market grow? What could the ecosystem look like? What are the available contents? What could be challenges?

1.2.1 Technologies

1.2.1.1 Hardware and Platforms

According to PricewaterhouseCoopers(PwC)[11], the current main market offerings of the VR systems and platform are show in the Figure 1:

	High-End-VR			Mobile-VR
	Oculus Rift	**HTC Vive**	**Sony PlayStation VR**	**Samsung Gear VR**
Displayauflösung	2160 x 1200 (1080 x 1200 pro Auge)	2160 x 1200 (1080 x 1200 pro Auge)	1920 x 1080 (960 x 1080 pro Auge)	2560 x 1440
Bildwiederholungs-frequenz	90 Hz	90 Hz	120 Hz (Reprojection), 60 Hz	60 Hz
Diagonale	unbekannt	unbekannt	5,7 Zoll	5,7 Zoll
Positions-Tracking	externe Kamera	Laser-Tracker	externe Kamera	bisher noch nicht möglich
Integrierte Kamera	nicht vorhanden	Frontkamera	nicht vorhanden	Smartphone
Mikrofon	ja	ja	ja	nein
Gewicht	470 g	600 g	610 g	318 g
Geeignet für Brillenträger	ja	ja	ja	nein (Bildschärfe aber einstellbar)
Anschlüsse	HDMI 1x USB 3.0 Kopfhörer (3,5 mm)	HDMI 1x USB 2.0 Kopfhörer (3,5 mm)	HDMI 1x USB 3.0 Kopfhörer (3,5 mm)	keine (mobil)
Systemvoraussetzungen	• Prozessor: Intel i5-4590 • Arbeitsspeicher: 8 GB • Grafikkarte: Geforce GTX 970 • USB-Anschluss: 3x USB 3.0, 1x USB 2.0	• Prozessor: Intel i5-4590 • Arbeitsspeicher: 4 GB • Grafikkarte: Geforce GTX 970 • USB-Anschluss: 1x USB 2.0	• PlayStation 4 • PlayStation 4 Kamera • PlayStation Move (optional)	• Galaxy S6 (Edge) • Galaxy S6 Edge Plus • Galaxy S7 (Edge) • Galaxy S7 Edge Plus • Note 5
Kooperation	Facebook und Microsoft	Valve	keine	Oculus VR
Preis	699 Euro	799 Euro	399 Euro	80 Euro

Figure 1 Current market offerings

There are basically two types of categories of VR right now in the market, one is the High-End VR, or desktop VR, which requires very high performance of a desktop PC as a host to render the images and then output them into the head-mounted display(HMD). It is important that the rendering PC could handle such a performance because once there is any noticeable latency, the user will lose the immersive experience and quickly perceive nausea or motion sickness and less

[11] ibid.

latency means more immersive and comfortable in VR. According to Oculus, a best practice would be that the target of motion-to-photon latency should be 20ms or less. The latency has always been a technological challenge for the VR systems. Luckily for both High-End and the Samsung Gear VR mobile VR systems the latencies have been decreased to ~20ms[12], if not less.

Because of the scale of using the VR system, those High-End VR solutions normally include position tracking technologies such as stand-alone sensors to offer room-scale movement and internal sensors such as accelerometer, gyroscope and magnetometer. For example, HTC's Vive utilizes two laser-based "Lighthouses" to scan and map out the room and position the HMD and controllers. Controllers for VR are specially designed to align with the interactions with higher degrees of freedom than the normal gamepads, although they are implemented in different ways trying to differentiate from the other. For example, Oculus Touch controller is such designed that there is a ring with LED sensors that covers the fingers so that it can even pick up the gestures.

Now all the controllers and HMD from Oculus, HTC and Sony can provide 6 degrees of freedom(DoF), which means the controller can make the following movements:

- Moving up and down (elevating/heaving)
- Moving left and right (strafing/swaying)
- Moving forward and backward (walking/surging)
- Swivels left and right (yawing)
- Tilts forward and backward (pitching)
- Pivots side to side (rolling)

---

[12] Bourque, "Spec Comparison: Can the Plucky PlayStation VR Upset HTC's Vive?."

6 degrees of freedom is important in VR as it would provide the most realistic experience in a virtual world. However, due to different technologies used in the tracking systems, they will have their own advantages or disadvantages in regarding to immersive experiences in the virtual world. The main differences lie in the play areas.

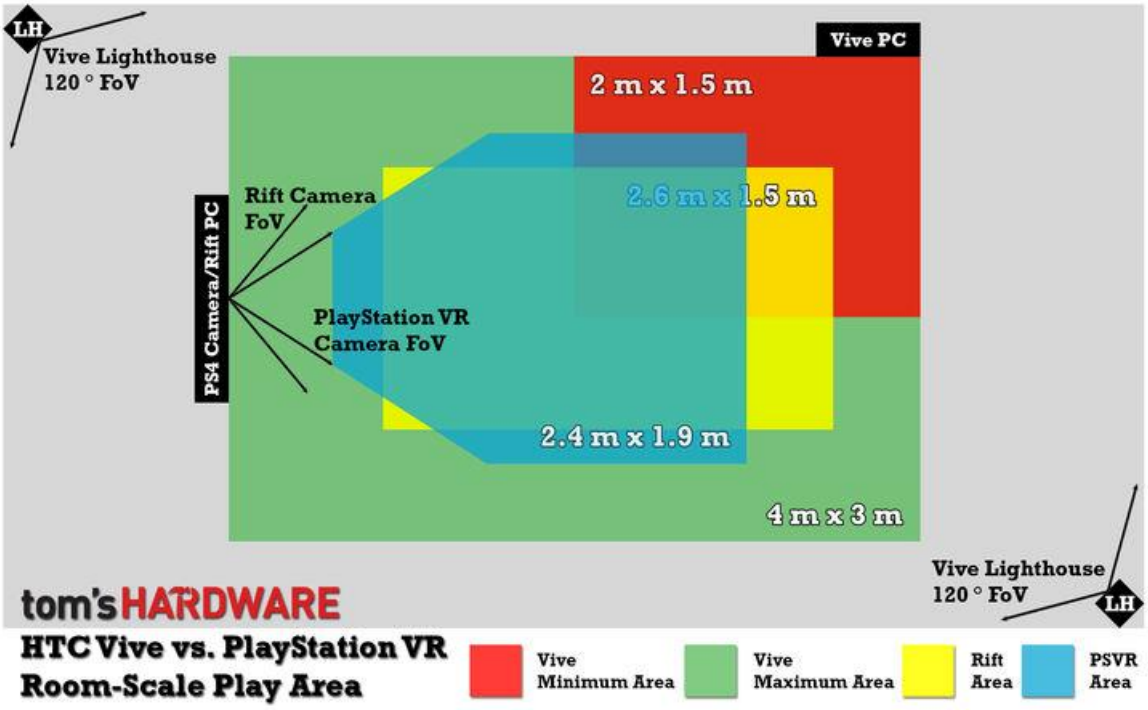


Figure 2 Play areas of different VR systems

From the Figure 2[13] above it can be seen that HTC Vive has the largest tracking area. Because it is using the laser beams omitted from the "Lighthouses" to scan the room and picks up the LED signals set off from the HMD and the controllers. If there are no obstacles, there would be theoretically no zone that cannot be tracked within the suggested size of the area. Compared to Vive, both PlayStation VR and Oculus Rift utilize cameras to capture the movement of the HMD on user's head and controllers. The cameras have dead zones, which means they cannot pick up

[13] Davies, "Oculus Rift, HTC Vive, PlayStation VR - Tracking \& Controls."

the signals once the HMD comes too close to the camera due to the optical limitation: The Oculus Rift has a dead zone for camera tracking as 90cm where the PlayStation VR has 60cm. Other than the dead zones, the latter two systems also have narrower FoVs and shorter cords, which lead to smaller size of the play areas. However, due to the lack of tracking sensors on the back of the HMD on HTC Vive, the tracking will not work in optimal if the Lighthouses are both placed in front of the user, which is not a concern for both Oculus Rift and PlayStation VR because they both provide sensors on the back of the HMD. That makes the setup of HTC Vive require more planning in advance.

1.2.1.2 Software

When it comes to software, there are mainly two aspects to be discussed, one is the content itself and another is the creation of the contents. Since this is still a young and fast-developing industry, it is faced with a "chicken or egg" question: killer application or growth of market[14]? It basically means: VR experiences are yet to be familiar to mass consumers who yearn for a killer application to persuade everyone else to jump into the virtual world. However, the market is currently still trying to catch up with the hype. Therefore, it is crucial that the device makers can integrate well with content provider, they even have to act the role as content provider so that the market can keep on growing.

1.2.1.2.1 Contents

Oculus

Oculus is laying out a larger picture in the VR industry showing more ambitious moves in various perspectives. Apart from the Oculus Store that hosts VR applications developed from third-party

[14] "A Critical Look at Virtual Reality."

developers, according to the Oculus Connect 3 conference in October 2016, this firm pulled off a couple of initiatives and projects that caught people's eyes. It was increasing the investment in VR contents with $250 million and committing to spend $250 million[15] more to jumpstart the VR content ecosystem. It will continue expanding the already existing mobile platform, Gear VR, by adding functionalities to Mobile SDK and enabling Facebook Live broadcast from any Gear VR app. Another important part of Oculus' ecosystem is the social VR. It announced Avatars, Parties and Rooms to allow users to have their own avatar in the VR world, to host voice calls with 8 people and to invite friends into a virtual meet-up to share movies and apps. A third aspect for Oculus to differentiate itself from the others is media and storytelling. It sees VR as an indispensable media in the future, hence it announced couple of creative tools for artists who are interesting in creating masterpieces in the VR world. With those tools artist can make sculptures, illustrations and paintings, also with animations. Other than that, Oculus has also founded its own studio to make short films trying to innovate a completely new way of storytelling that never existed before. Apparently, Oculus is making an effort to bring VR experiences in multiple forms to the mass market and aiming to spread this technology as wide as possible. It utilizes new mechanics called Asynchronous Spacewarp (ASW) to "allow games to run at half frame rate and look nearly as good as native 90Hz rendering"[16] so that computers which have lower technical specifications can also run the VR applications and deliver similar experiences. More than that, it is committing $10 million to support diversities which include women and people from underrepresented groups so that they could also take part in producing contents in the VR platform.

---

[15] "OC3 Reveals: Touch Launch, Santa Cruz Prototype, Min Spec, 400+ Mobile VR Apps, and More."
[16] ibid.

It seems Oculus has broader interests in different aspects of VR content creation in a longer term and that it dedicates to delivering this technology to as many people and as soon as possible.

HTC

With no exception, HTC also needs a strong platform for publishing contents in order to support its VR system. Before August 2016, HTC Vive relied on the partnership with Steam, a platform known for game distributions and hosting some VR gaming and non-gaming applications, to provide contents for its users. But afterwards, HTC introduced Viveport as its own application distribution channel. It works as an application store that provides all kinds of VR contents across different categories more than typical gaming and entertainment, including "information, edutainment, social media, 360˚ video, news, sports, health, travel and shopping[17]". This strategy brought HTC to shed lights on non-gaming sectors, especially in so-called edutainment, where, for example, children can discover various 3D objects in a VR museum. HTC also wants to speed up the evolution of the technology and spread it to non-gaming applications. In order to achieve that, it initiated Vive X and Vive Business Edition. Vive X is an accelerator or an incubator where HTC will provide $100 million[18] as a fund to support start-ups of great VR ideas by offering technical expertise, access to VR technology, financial investment, mentorship and marketing support and make them finally a content producer or enablers[19]. On the other hand, Vive Business Edition targets on technology support for commercial use cases where it provides not only devices and systems but also extra customer services and license. With all these considerations HTC hopes to build up its own ecosystem.

---

[17] "Introducing Viveport."
[18] "Chap23-Transitioning to Vr Content Creation."
[19] ibid.

Google

Since Google announced its Cardboard in 2014, it kept developing its own platform and new devices. At a live event on October 2016, Google largely expanded its 2-year-old Cardboard platform. Now Google has a VR initiative that includes a device-and-content-distribution platform called Daydream and a content-create-and-device platform called Jump. Daydream consists of a brand-new designed headset with high quality and user-friendly details and comfortable materials that can be paired with a new controller, a content provider service which distribute VR-related applications, games and videos and last but not least, a Daydream-compatible phone that would run all the VR content through the new headset. As for the Jump platform, it is built for VR content creators. Right now, it is comprised of a camera rig that can hold 16 camera modules as a circle to capture videos and a post process computing unit that can assemble all the videos captured from the 16 cameras and output a 360-degree video in high resolution. Finally, this platform can connect to YouTube to publish the 360-degree video. With Daydream and Jump, Google is aiming to finally build up a complete VR ecosystem.

Sony

For Sony, the strategy is relatively focused and the clear. Since it is dependent on the gaming console PlayStation 4, most of the VR content for PlayStation VR are games or videos. As it is written, after a few weeks of the shipping of PlayStation VR, in Sony's application store, PlayStation Store, dozens of VR applications or games have been published. It is worth to be mentioned that Sony also integrated social element in the VR gaming world, from a demo[20], it is seen that three players are sharing the same world and interacting with objects like musical instrument and sport equipment together. It is also interesting to see that Sony seemed to have

[20] Jeffrey Grubb, "PlayStation VR Social Demo Amazes Crowd of Developers."

integrated eye-tracking technology in the PlayStation VR therefore players can see the movements of eyeballs from each other's eyes. Apparently, for the most feasible user group, Sony is prepared to make profits out of the gamers who are willing to give it a try. Other than gaming and video contents, it is not clear if Sony will push the contents producing into other areas.

1.2.1.2.2 Development

The platforms and ecosystems mentioned above are all prepared by the device manufactures. As for developers, frameworks are already being built, tools are being prepared, more software development kits (SDKs) are on the way.

Game Engines

A game engine is similar compared to integrated development environment (IDE) for software development. In a game engine, a developer can combine resources like 3D objects or models, transformation of those objects, position and movement of the objects, audio objects, scene setups of a game and most importantly, the codes and mechanics that act as a "director" who controls those objects and pull off a "show" on the stage. With the help of the user interface of a game engine, developer can easily create mechanics that direct the movement or transformation of an object and various kinds of events. In comparison to web development as an example, a game engine will help sorting out more complex events that happens on 3D assets while developer can have an intuitive view on the scenes as show in the Figure 3[21]. As a result, efficiency on workflow is fulfilled, which is essential to game development.

[21] "Unity 5 vs Unreal Engine 4."

Figure 3 One could have a view on the current "scene" where the objects are placed in a way accordingly and other assets well organized.

Plugins / Software Development Kits (SDKs)

There are two popular game engines[22] that are well supported by VR manufactures, one is Unity and another is Unreal Engine. These two engines both have plugins or SDKs supported by Oculus, HTC (SteamVR), Google and Sony. Once they are imported to the project, developers can easily call relevant interfaces or classes or can attach those scripts onto objects as wish for further manipulations. Except Sony, other VR system providers also provide with abundance of documentations upfront and they will walk the developer through the usage of the libraries provided in the plugins. To become a developer for Sony PlayStation, one has to go through a process of validation, then can he or she receive the support for development from Sony.

Except those two game engines, some other game engines also support VR development. For example, CryEngine from Crytek and Lumberyard from Amazon provide built-in supports for the Oculus Rift. But they are not discussed here.

---

[22] "This Engine Is Dominating the Gaming Industry Right Now."

1.2.1.3 Other Technologies and Platforms

The technologies described above can be considered as a base and mainstream that developed from big players in the market. However, they are not the only players in the market of course, there are also various smaller companies that keep pushing the boundaries of VR technologies and enrich the whole ecosystem.

Eye-tracking

Eye tracking technology tracks the movement of the eyeballs inside of the HMD, which can bring up interesting applications. Many would think of using it as a selecting tool in the VR, however, it does not seem to be a very good application for such technology. According to Jacob[23], eyeball movement can be misleading and meaningless because there is such problem called Midas Touch problem referring to the fact that people expect to look at things without that look "meaning" anything. Therefore, unless eye-tracking input is combined with other types of input or the gaze information is leveraged, it is not an ideal interaction method. But there can be other usages of this technology. As implemented by Oculus, eye tracking in interactions among players in a VR world is much more appealing and delivers richer information in the social aspect. Eye tracking can be also helpful in optimizing performances or reducing delay in an HMD. For example, implemented by FOVE[24], an HMD embedded with eye tracking technology can do selective rendering depending on where the eyes are looking at so that unimportant or unfocused details would be left out and therefore improve the performance and reduce the delay.

---

[23] Jacob, "The Use of Eye Movements in Human-Computer Interaction Techniques - What You Look at Is What You Get.."

[24] "Home - FOVE Eye Tracking Virtual Reality Headset."

Hand as an Input Device

When considered as an input device, bare hands can provide high intuition and unencumbered user experience in a VR world hence improves the immersion and sense of presence. One of the leading vendor of this technology is Leap Motion. With only bare hands and the tracking device on an HMD, as shown in the Figure 4[25], a user would be able to see his or her own hands in the VR world.

Figure 4 Bare hand input from Leapmotion

Compared to physical controller on hand with buttons, there are pros and cons. Obviously the user is unencumbered when trying to interact with objects in VR with their own hand and it looks extreme compelling. However, the user also will not be able to feel physical feedbacks that can be

[25] "Orion."

commonly experienced from physical controllers and buttons. Also, the user may also have problem of fatigue after a long time of interactions as the hands have to be in the certain range of the tracking sensor. This technology is also faced with usability issues. The recognition of the fingers sometimes can be difficult, movement information will be also misinterpreted from time to time and when the accuracy of recognition becomes poor, user would feel frustration and the experience will be hugely discounted. However, once those challenges are overcome, this way of input will dramatically improve the experience in VR.

Nosulus Rift

This piece of technology is quite special in comparison to the ones mentioned before and it is surprisingly interesting. In VR, the more stimuli are projected onto the sensory receptors, the better immersion will be created for the user. So far for most of the VR devices, visuals, audios and even physical force stimuli were taken care of. However, one more dimension has been left out, the olfactory stimuli, until the summer of 2016.

Figure 5 Nosulus Rift simulates the smell

During the Gamescom 2016, a device that to be worn on the nose was brought up by Ubisoft as seen from Figure 5[26]. Basically, this muzzle-like device carries two scented capsules and will disperse smells depending on the game.

WebVR

As mentioned before, there are now mainly two platforms for VR, one is the desktop high-performance VR and other is portable mobile VR that relies on native VR applications. However, there is another platform under development, growing and blurring the line between desktop and a mobile device, which is Web VR. Web VR can be achieved by WebVR API[27], which is an open and experimental Javascript API that provides access to various popular VR devices including

[26] "Nosulus Rift."
[27] "WebVR - Bringing Virtual Reality to the Web."

Oculus Rift, HTC vive and Google Cardboard and more. With this access, developers can finally build VR applications that sit in the browser with connection to any VR devices. It is also imaginable, with a more mature WebVR API and expected growing extensions and open Javascript APIs of visualization frameworks like THREE.js and D3.js, or reporting tools like Tableau and QlikView, VR data visualization and reporting in web could be very appealing and promising.

1.2.2 Market Profiling

Different institutions have different ideas about the future of VR market, but in general they all have optimistic projections. Prediction comes from Deloitte[28] in a report claimed that the VR sales will reach its first billion in US dollars in 2016 with about $700 million in hardware sales and $300 million in software and content sales. They also estimated sales of 2.5 million VR headsets and 10 million game copies. According to Goldman Sachs[29], by year 2025 there are three kinds of estimations: base case (Figure 6), accelerated uptake and delayed uptake.

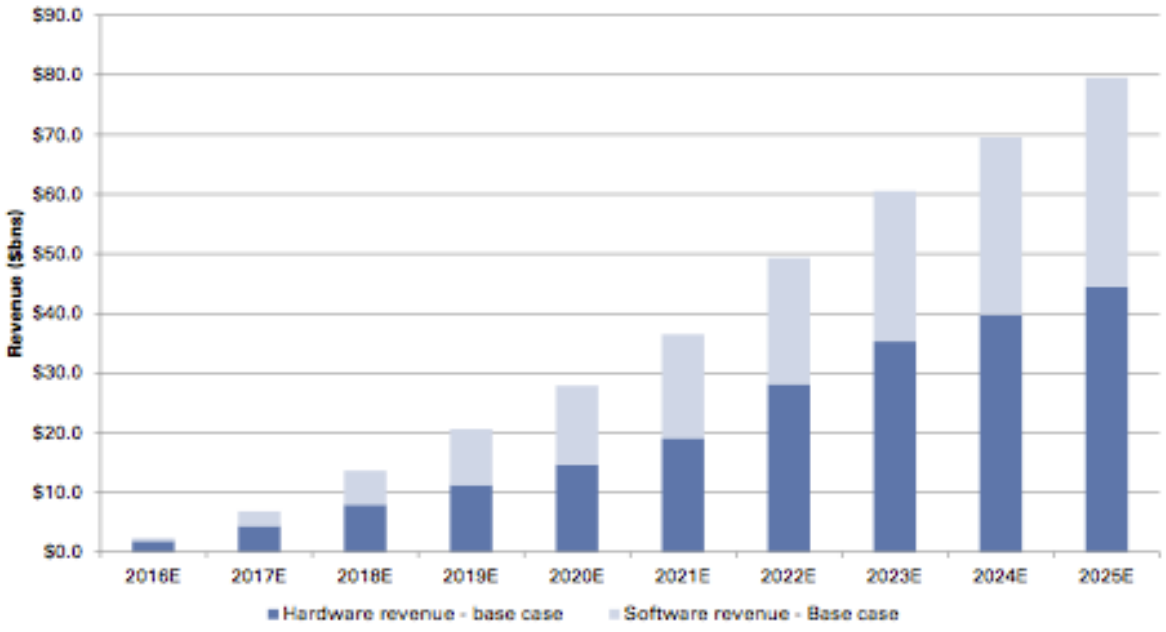


Figure 6 Base case forecast from Goldman Sachs

The "base case" scenario means the HMDs gains popularity as VR/AR technology improves overtime but limited by the mobility and battery life, under this scenario the revenue of total

[28] "Technology, Media & Telecommunications Predictions 2016."
[29] "Profiles in Innovation - Virtual & Augmented Reality."

available market (TAM) would be $80 billion. The “accelerated uptake” would mean a brighter future for HMDs to become a generic computing platform with less challenges on mobility and battery technology, the revenue TAM would be $182 billion. As for a more pessimistic point of view of “delayed uptake”, the revenue TAM was estimated to be $23 billion as the HMDs are faced with challenges from latency, display, safety, privacy and other issues so it would not be widely spread. As shown in the Figure 7, Goldman Sachs sees VR and AR together as revolutionary visual technologies so it estimated the revenues from these two technologies as a whole. In terms of software, Goldman Sachs estimated that 75% of the software market will be generated from VR use cases.

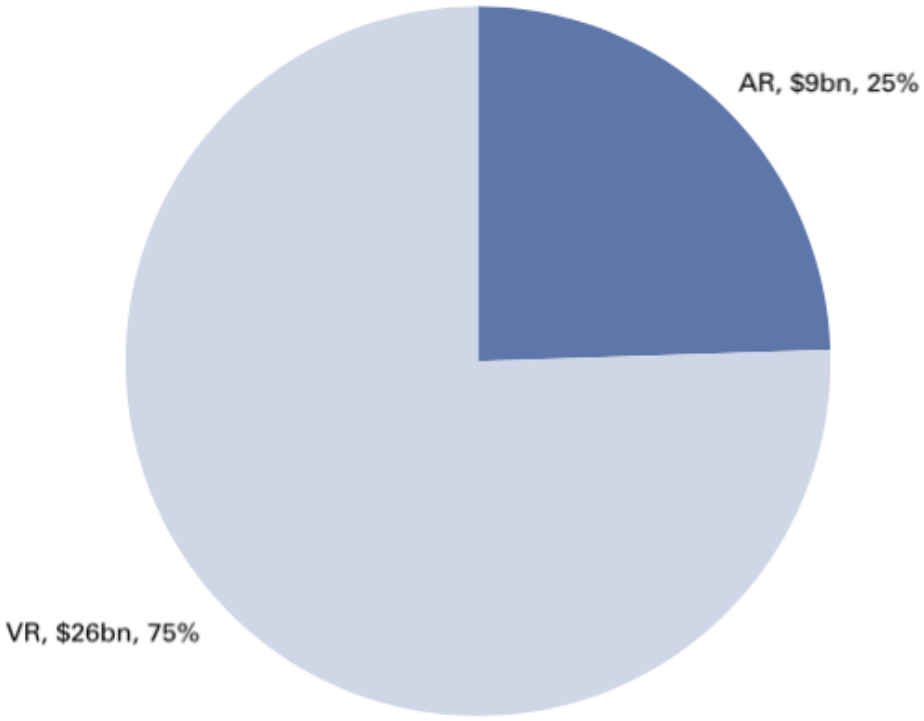


Figure 7 Software revenue estimates by VR and AR

PricewaterhouseCoopers (PwC) also made its predictions on the VR market with a timespan from 2014 till 2018, shown in the Figure 8[30].

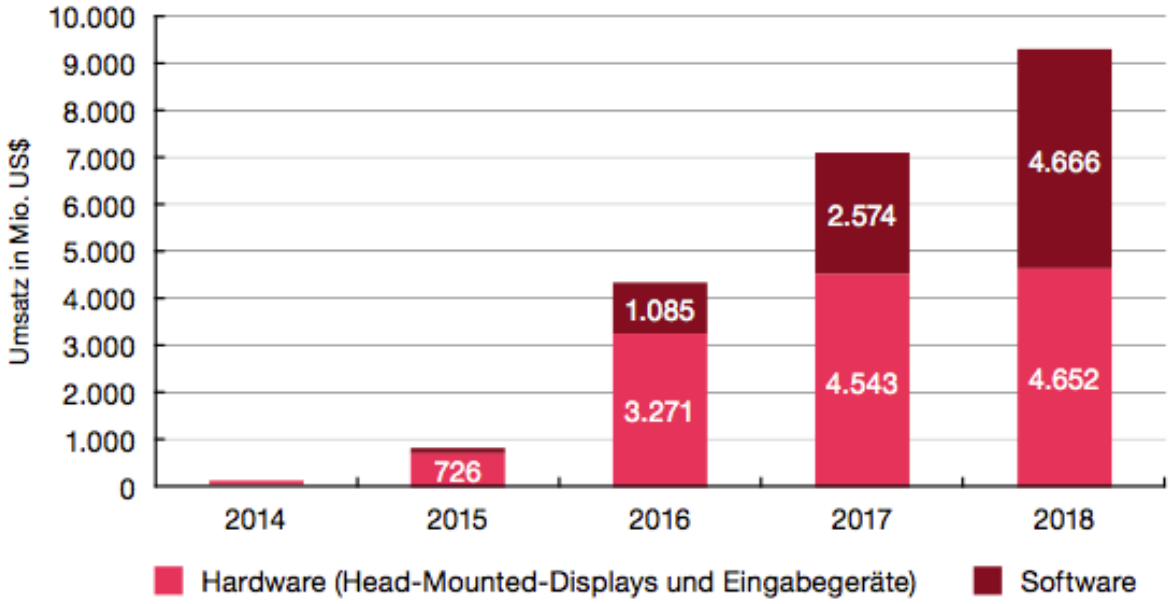


Figure 8 World-wide VR revenue prediction in from year 2014 to 2018

According to them, the revenue TAM of VR will reach $4 billion in 2016 with around $3 billion in hardware and accessories and $1 billion in software, which reflected the current situation that the HMDs and other device manufactures are trying hard to evolve their technologies as fast as they could and they are pushing those devices to as much mass audience as possible. Then gradually the content producers will step in to complement the ecosystem. After a few years the hardware will become mature and the price will sink, but the contents will be more of a substantive factor. Only polished contents and applications will bring in the long-term momentum of growth. But before VR contents are made, different use cases have to be identified.

---

[30] Ballhaus, Bruns, Deligios, Graber, Kammerling, Lorenz, Schink, Wipper, and Wilke, "Digital Trend Outlook 2016."

1.2.2.1 Use Cases

In general, there are two kinds of use cases, one is non-gaming and another is gaming use case because of the major proportion that non-gaming use case can occupy.

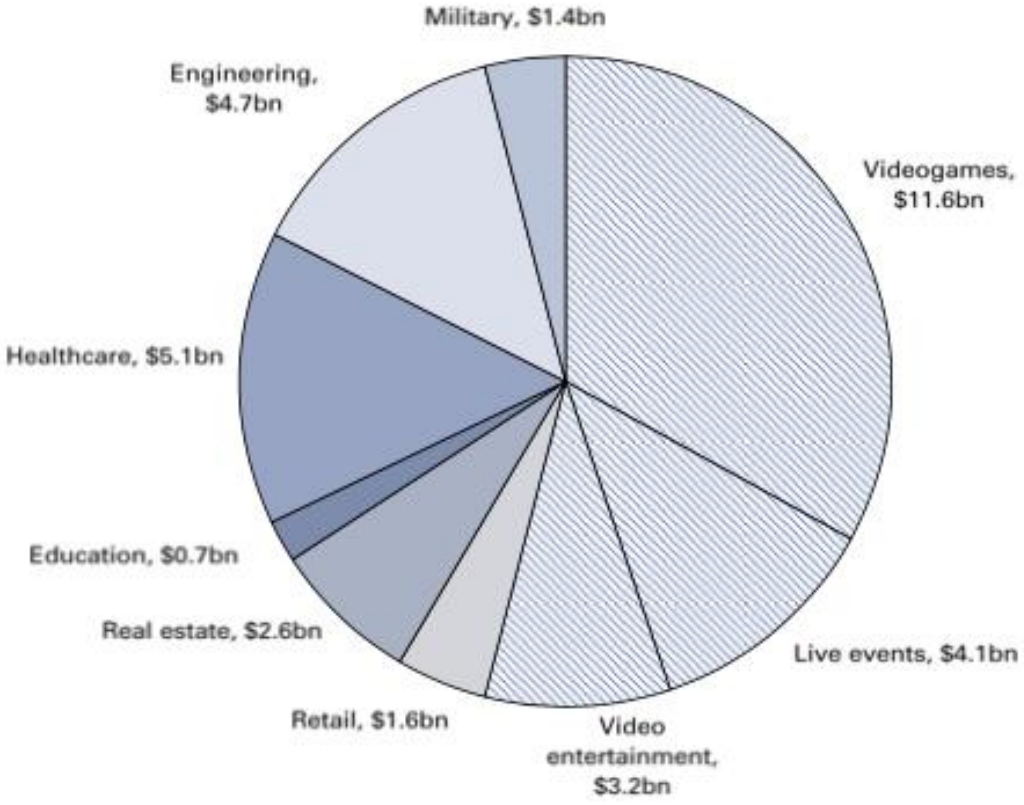


Figure 9 VR/AR use case estimates by 2025

As shown from the Figure 9, according to Golman Sachs' estimation[31], by 2025, the use case of video gaming in VR/AR will take almost 1/3 of the total software market value. From PricewaterhouseCoopers' report[32] shown below, most of the audience between 18 and 30 years old would also see video gaming as the major application of VR.

---

[31] "Profiles in Innovation - Virtual & Augmented Reality."
[32] Ballhaus, Bruns, Deligios, Graber, Kammerling, Lorenz, Schink, Wipper, and Wilke, "Digital Trend Outlook 2016."

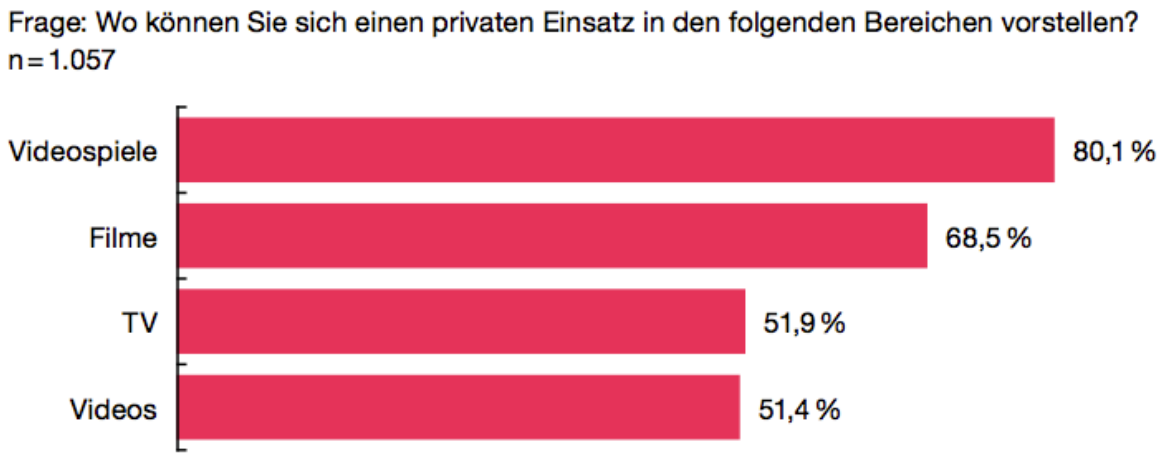


Figure 10 VR use case survey

Then the remaining non-gaming use cases will mostly include engineering, professional trainings, live events, entertainments and marketing and retailing. It is easy to imagine that with the help of VR/AR, engineering can benefit from fast-prototyping and modelling; professional trainings like medical practices and surgical trainings will also benefit from VR/AR technology in a great extent, not to mention the already-popular military training that uses cockpit simulations, which is also a type of VR; entertainment in VR/AR is also a brand new category to be created as it brings film-making, storytelling, social networking in a whole new level and perspective; live events and marketing and retailing will also prosper with the technology of VR/AR, because of the immersive and intuitive experience that can be delivered to end consumers, which will apparently invoke desires of event or product engagement.

But how about analytics? Some VR/AR enthusiasts also mentioned a term “immersive analytics” or “VR/AR analytics” meaning to vision the merge of analytics technologies, for example data analytics, big data, business intelligence, together with VR/AR. This is a bold and visionary idea

and has the potential to be revolutionary. According to their view[33], VR/AR is a new way to interact with users and it senses users' movements on limbs, eyeballs and even possibly expressions and speeches, by collecting those data generated from VR/AR, analysts can analyze user behaviors, emotions or sentiments and languages of certain reactions to specific VR scenarios in order to have special insights on the VR/AR experiences they are delivering. This is only one way of the analytics, VR/AR-to-data analytics. However, this could go both ways as data could also be visualized in the VR/AR worlds. With the world or scenario created by VR/AR and innovative data visualization techniques, users can experience their data in a whole new way that could never have been done in a plain 2D world.

[33] Osarek et al., *Virtual Reality Analytics: How VR and AR Change Business Intelligence*.

Part 2 VR Data Visualization Design

Visualization is essential for communication between human beings and the nature of data. A proper visualization of data will not only simply be plotting data in a meaningful way, but will also provide new perspectives for investigating data and thus alternative angles to perceive the world that is constructed with data. It plays an even more important role especially in the world right now where digitalization transforms every day's business and life. The term "Big Data" brings the recognition of velocity, variety and volume as three factors of today's data[34]. In this era of Big Data, how can the humongous volume of data with multiple dimensions or features be visualized and deliver the information that is of importance to the users? As the technologies developed till today, human computer interactions (HCI) has also been dramatically changed. Nowadays even visual interaction is not enough. People use gestures like pinch and swipe on their smartphones for basic interactions, video game players use kinetic gestures to control actions and receive haptic feedbacks from the game, multiple-modality interactions is now becoming the mainstream of HCI. Therefore, as a new emerging technology that could provide immersive environment which tracks user's kinetic behaviors and offers intuitive interactions with virtual objects, VR is the confluence of the need to visualize complex models, spatial relationships and the rich affordances of interactions from the objects to the human beings.

Data visualization in VR can map multiple dimensional data models to the virtual world and provide multi-modality interactions to the datasets. As experimented[35], because human beings are biologically sensitive to visual clues and patterns, the VR environment enables more effective

---

[34] McAfee and Brynjolfsson, "Big Data: the Management Revolution."
[35] Donalek et al., "Immersive and Collaborative Data Visualization Using Virtual Reality Platforms.."

analysis as a method of visualization, especially in revealing valuable information in spatial and locational datasets. In addition, VR can also visualize multi-dimensional data easily. Since it is in 3D world, where human beings are naturally accommodated to, data points can contain more information than just a 2D graph by incorporating variations not only in size, color, textures, but also shape or form, spatial sound, spatial alignment on XYZ axes and even perhaps smell. Thus, multiple dimensions of data can be brought to the virtual world and shown to the data analysts. With advanced interaction technologies mentioned in the last section, a data analysts and then walk through the dataset, interact with the data subsets that are of special interests and get closer look at a data point by approaching to it. Last but not least, aside from visualization and interactions, VR also brings possibilities for data analysts an opportunity to collaborate with others albeit they are physically in different spaces. By sharing the same virtual world and taking advantages of VR interactions, they will help each other find information patterns that are not directly seen. Thus, efficiency and effectiveness can be achieved with less efforts comparing to traditional data visualization techniques.

In the following section, user experiences and interactions in VR will be discussed followed by techniques and guidelines of visualization in 3D, finally a preliminary discussion on collaborative technologies and possibilities will be presented.

2.1 VR User Experience Design

VR provides a different kind of 3D world than what it is only seen from a plain screen. VR emphasizes on immersive experiences and creating a virtual world that blocks out the real physical world surrounding the user. Therefore, the user experience in VR have to be considered differently. Aside from simply creating objects that can be interacted with, there are several other points to be considered thoroughly. For example, bad designs may not only misguide the user behaviors, but may also quickly draw the user out of the immersive virtual world. Additionally, careless design will also bring adverse health effects to the user causing motion sickness, physical fatigue or even worse.

The following part will start to discuss about the major adverse health effect — motion sickness, and how to avoid them. Then it will go through some design guidelines so that smooth user experience can be delivered.

2.1.1 Adverse Health Effects

It has been always a great challenge for VR systems and applications to deal with the uncomfortable feelings the user may experience after trying out VR. Those feelings could include nausea, dizziness, eye strain, physical fatigue, headache and so on. Additionally, it may even bring user's health at risk, for example physical injury, hygiene issues or even transmitted diseases. Those effects can all be considered as adverse health effects[36]. In order to bring the optimal VR experiences to the user with as less adverse health effects as possible, identifying the causes and further to find the countermeasures are crucial. Among all the adverse health effects, motion sickness is a major one.

2.1.1.1 Motion Sickness and Factors

Motion sickness is one of the major factor that causes user to feel nausea, dizziness, headache, discomfort, disorient, sweating, drowsy or even sick[37] during the VR experience. It refers to adverse symptoms and readily observable signs that are associated with exposure to real (physical or visual) and/or apparent motion[38]. Motion sickness appears actually in daily life, for example people will feel sick when travelling in a boat, car or plane. In VR, mostly user will experience simulator sickness which is induced by motion sickness when the visual motion appears in a simulation but the actual physical motion is absent. In this case, there are conflicts among the visual, vestibular (balance), and proprioceptive (bodily position) senses that will produce discomfort[39]. More than nausea and dizziness, this discomfort can also induce strain and fatigue. Although the factors that are contributing to the motion sickness are rather complex and often

---

[36] Jerald, *The VR Book: Human-Centered Design for Virtual Reality*, 159.
[37] Kennedy et al., "Virtual Environments and Product Liability.."
[38] Ben D Lawson, "Motion Sickness Symptomatology and Origins.."
[39] "Oculus Best Practices."

interrelated[40], Oculus summarized a handful of factors that can be seen as instances of the theories of causes of motion sickness in VR[41]:

- speed of movement and acceleration
- degree of control
- duration
- altitude
- binocular display
- field of view (FoV)
- latency and lag
- distortion correction
- flicker
- experience

Among these factors, some are creating sensory conflict: too fast of the movement or unstable and long-duration acceleration, lower altitude of the user viewpoint, high latency and too wide of a FoV — because they will create conflict information between vestibular and visual sensory and will also affect how the brain choosing the rest frame without enough motion cue; other factors such as: lack of control on viewpoint and flicker are more relevant to eye movement theory — because they will cause passive eye movement as the eyes are always trying to find a fixate point to focus due to unexpected changes.

---

[40] Jerald, *The VR Book: Human-Centered Design for Virtual Reality*, 197.
[41] ibid.

Latency

According to Jerald, latency is the major contributor to the motion sickness. Oculus also identified it crucial to be comprehended by developers so that user may enjoy the VR. Latency is defined as the total time between the movement of the user's head and the updated image being displayed on the screen ("motion-to-photon")[42] because of delays. The delays are from tracking, application, rendering, display and synchronization among components as shown in the figure below.

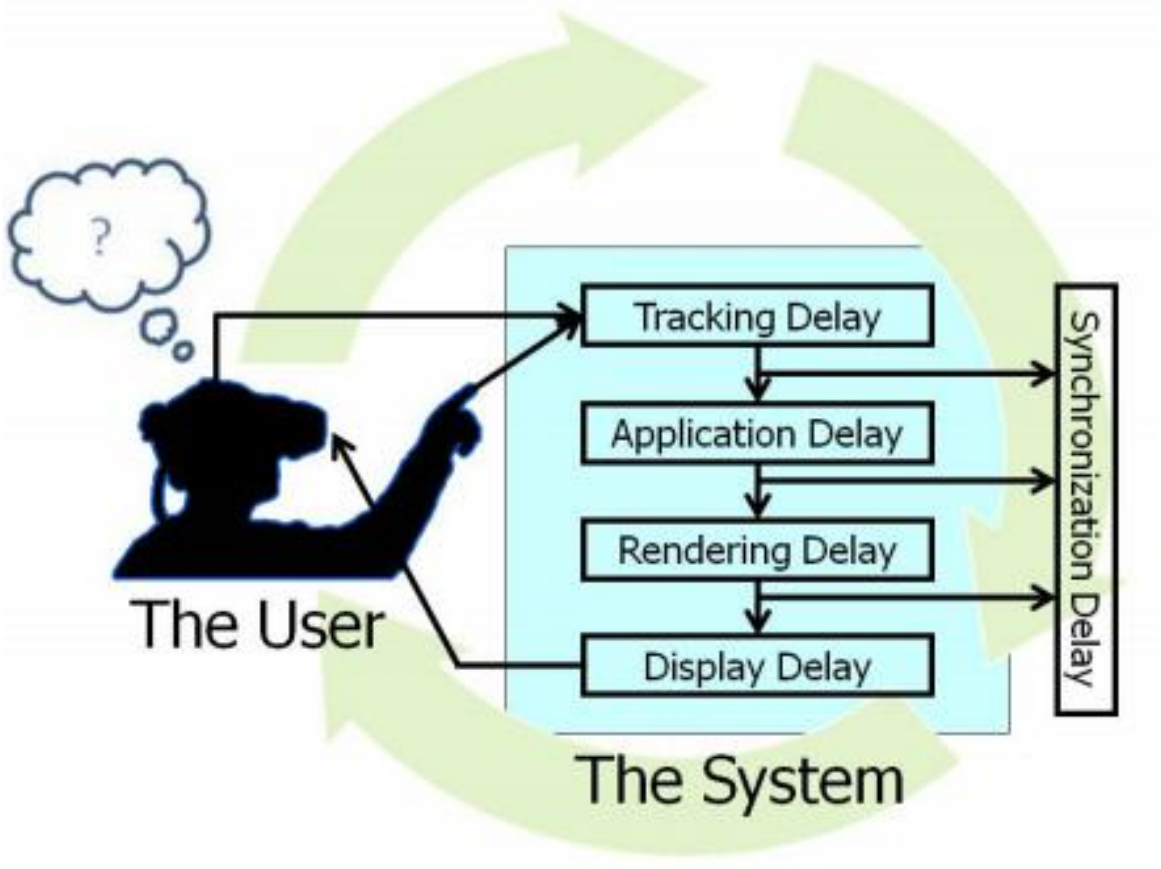


Figure 11 VR system end-to-end delay[43]

[42] ibid.
[43] Jerald, "Scene-Motion- and Latency-Perception Thresholds for Head-Mounted Displays."

So how low the latency is acceptable in a VR system? While Oculus gave a number of 20ms as a threshold under which the VR experience is to be compelling, once the latency is higher than 60ms it would cause discomfort and sickness[44]. According to a research carried out at NASA Ames Research Center, it is reported that individual threshold of latency would varies by 85ms due to many factors such as bias, head movement and experimental conditions. Jerald further found that as head motion increases, the sensitivity to latency also increases[45]. He found out that for latency lower than 100ms, users do not perceive latency directly, but rather the consequences of latency that a static virtual scene appears to be unstable in space when users move their heads[46] makes the latency perceivable.

---

[44] "Oculus Best Practices," 22.
[45] Jerald, "Scene-Motion- and Latency-Perception Thresholds for Head-Mounted Displays."
[46] ibid.

2.1.1.2 Countermeasures

How to avoid the motion sickness in the course of design and setting up the system? To tackle motion sickness, there are mainly two aspects to consider. In terms of hardware devices, since latency is a major contributor that causes motion sickness, it is crucial to keep the refresh rate high and frame rate high, as well as the latency consistent. As suggested by Oculus[47], developer should try to make the application run at a frame rate equal or greater than the refresh rate of the HMD, which is 90Hz for Rift, vertically synced on and unbuffered. The same requirement should be applicable to HTC Vive too as it has the refresh rate also at 90Hz. In order to achieve this, the developer should organize the code and optimize the application as much as possible. Meanwhile the device manufactures are also developing technologies that could minimize the latency. For example, Oculus' SDK supports predictive tracking and so-called TimeWarp that could reduce latency after rendering. It is also useful to adjust the inter-camera distance as some people will experience discomfort using stereoscope thus reducing this distance may help[48]. FoV (field of view) is also a variable to adjust when some user feel sickness when it is too wide. Another aspect to consider is how the application is designed, namely: movement, interactions, world and so on. Jerald makes some suggestions on this aspect: according to the theories mentioned before on the causes to motion sickness, a proper rest frame should be provided in the scene if the user is just sitting or standing while using the VR system, thus a "real-world stabilized cue" will be perceived and hence the vestibular and visual system would work well with each other[49]. For example, if there are movements in the world required, then a vehicle or a cockpit that is stable relative to the real world would be a helpful cue to provide a rest frame. If not a full cockpit, even a stable

---

[47] "Oculus Best Practices."
[48] ibid., 25.
[49] Jerald, *The VR Book: Human-Centered Design for Virtual Reality*, 208.

platform or a series stable arrows pointing the direction the user could go would be helpful reducing the motion sickness brought by the conflict of visual and vestibular cues. Another point to consider when designing the application is acceleration and rotations. Accelerations in any direction could cause the vestibular-visual conflict and should be minimized and shortened as much as possible. For example, when giving the user the ability to teleport his/herself in the scene, it is obvious that the motion sickness is much perceivable when the teleportation is in "dash mode" where the user will accelerate to the destination than just a "fade-in and fade-out" effect where there is just a direct change of position with a dark-out transition. Passive movements with acceleration and rotation is even worse and could easily cause nausea and dizziness, they should be avoided. As mentioned before, the height of the viewpoint is also important because when it is positioned to close to the ground, when motion appears, the relative speed could be too high that it will cause sickness. Viewpoint movement should also be in the control of user with no fast movement or continuous acceleration.

2.1.1.3 Summary

In summary of this section, it is clear that for many new users, sometimes experienced users, VR applications can induce adverse health effects, however the developer could always reduce those effects to its most extend by bringing in carefully designed UI and objects, streamlined codes, well-adjusted hardware and extra attentions to user's interactions with the applications. The adverse health effect could be less of a problem in the case of data analytics in VR because the analysts may not need to move him/herself as quickly as in a game. However, when the data amount become huge and it is easy to become overwhelming to an analyst and thus he/she may feel disorienting and confusing. Therefore, well-rounded human-oriented designs on contents and interactions are crucial. It will be discussed in the next section

2.1.2 Content Design

Fundamentally, VR is a new medium of communication between machine and human being. Users are longing for a useful platform that can make daily business easier, VR offers the chance. As an immersive communication platform, the world created in VR becomes so subjective as never before. Always as the first person, users expect something more interesting, engaging and that stronger bonding are built between machine and the user. As a VR developer, it is the duty to make the experience a compelling one that can retain the user and make the user keep coming back, thus this new technology can be utilized to the most extend. Since VR is genuinely an intersect of multiple disciplines and a combination of art and science, many concepts from different disciplines are brought in to make the best content design possible in VR. Researchers have found 4 elements crucial to VR user experiences[50]: strong emotion, deep engagement, massive stimulation and escape from reality. Strong emotions include feelings such as joy, excitement surprise and so on, sometimes may become extreme. By fulfilling dreams or achieving goals, strong emotions would be aroused. Deep engagement occurs when users are in the moment and experience a state of flow as described by Csikszentmihalyi[51]. When users are engaged, they forget about time and space and extremely focused. Massive stimulation refers to multi-modality stimulation that are perceived by the user so that he/she can immerse in the experience in different ways. Escape from reality means the user is so indulged in the VR and pay no attention to sensory cues from the real world or passage of time. Apart from the 4 elements described, Lindeman and Beckhaus also characterized the concept of experiential fidelity. It is the degree to which the user's personal experience matches the intended experience of VR creator. It is not always the case that the user's personal experience

---

[50] Lindeman and Beckhaus, "Crafting Memorable VR Experiences Using Experiential Fidelity.."
[51] Csikszentmihalyi, *Flow: the Psychology of Optimal Experience*.

could align with the intended experience of VR creator because it is highly subjective, but a successful story telling with high experiential fidelity in VR could nevertheless better fulfil the 4 elements and thus an impressive experience. It is no exception for data analytics.

Data analytics also requires an alignment between information hidden in the data and the question that is to be answered by the analysts, which means a good data story in VR can make the analysis more effective, comprehensible and compelling, especially when presented to the final reader of the result. There is a method called skeuomorphism that could help with alignment of the intentions between the content creator and the end user. It means the incorporation of old, familiar ideas into new technologies, even though the ideas no longer play a functional role[52]. In other words, when incorporating real-world metaphors into the VR world, it would be much easier for new users to adapt to the scene hence a better experience.

Those are the general design principles shall be kept in mind when designing the contents in VR, the next section will discuss the content design principles in two more specific perspectives: environment and user.

2.1.2.1 Environment Elements

Scenes

Scene is the fundamental element in a VR world or environment. In a scene, every object is included although not all may be scene. There can be multiple scenes for a complete VR experience and transitions in between. A user will be set in a scene to start from status 0 and then perceive and

---

[52] Norman, *The Design of Everyday Things: Revised and Expanded Edition*.

interact with the different levels of scene. A scene can be divided into a few different layers and they serve different purposes and should be designed differently, they are: background, contextual geometry, fundamental geometry and interactive objects[53]. Background is scenery in the periphery of the scene located in far vista space. It is normally the sky, mountain and the sun. In Unity3D, it is rendered as "skybox" and there is no cue of depth in this layer, therefore a rest frame or a reference is needed to tell how the objects are positioned. Contextual geometry is the layer where the objects that provide a certain context to the scene for example, trees and terrains. They are usually not interactive and they define the environment the user is in. Fundamental geometry is the layer where objects provide basic experience and they are usually static components, for example table, chairs and walls. Normally they are not interactive but serve the purpose for supporting other objects or constraining the actions of the user and they are close to user. Interactive objects are the objects that can will provide the most affordance to the user and are usually within reach. Since the layers described above contain different objects in various distances and serve different purposes, for a VR experience, it is important to provide different levels of details accordingly, otherwise it could on one hand provide too much or less of information or on the other hand when the scene is overcomplicated, the performance of the application would be affected severely and thus adverse health effects are induced. For example, when visualizing a huge dataset, it is helpful to simplify the background and contextual geometry and focus on fundamental geometry and interactive objects because they are in the action area of the user, as a result, the user will not feel overwhelmed.

---

[53] Jerald, *The VR Book: Human-Centered Design for Virtual Reality*, 237.

Color and Lighting

Colors and lightings are the elements provide direct visual cues and affects the emotions. Different use of color and lighting can help user distinguish objects. For example, when the user wants to select an object or highlight a data point during analysis, the target can be colored in a bright color and high lighting while other irrelevant objects can be grayed out and dimmed. People also tend to associate color to emotional reactions and meanings. It is a common approach in business reports that the "traffic light" indicators are applied to show whether a status is good, average or bad. Therefore, wisely choosing color codes could effectively convey the information to the user with intended emotions. For example, in the case of analysis, it is important that the user could focus on the visualized data and perform meaningful interactions, then it would not be a good idea to choose colors of too many variations with too much brightness since it introduces unnecessary noise information and conflicting feelings.

Audio

Audio is also a major element in the VR, especially when spatialized audio is used. Spatialized audio makes the sound perceived from a location in a 3D world, which means it could be used as a clue for navigation or direction. Apart from this, audio could deliver information in combination with visual cues in order to emphasize important contents and could also provide interactions cues like confirmation, notification etc. When those features combined together, it provides user with multi-dimensional information. For example, in data analysis, an introduction speech could serve a good purpose for providing context of the data; when the user successfully built a visualization model out of selected parameters, a success notification sound could be played and reveal the next possible actions; when the user is wandering through the data, the spatial audio cue could tell the

user in which direction to go in order to find the target data. Audio experience is crucial in VR since it could be a powerful compensation to visual cues and provide information in another dimension.

Environmental Wayfinding

Apart from these elements, a mechanism in VR plays an especially important role than that in other kinds of medium, it is "wayfinding aids" [54]. This mechanism assists the user in virtual world to locate and position him/herself, to identify the goals and to stick to the plan to get to the objectives. Darken and Sibert pointed out, if not given enough source of directional cues, then wayfinding performance would be inhibited and the user then will feel disorienting. Imagine a data analyst fly through a space where cubes are plotted in a 3D space with only a background layer but without any references and axes, since the background layer does not provide with any depth clue, it would not take long before the analyst becomes lost and confused. In VR, this is specially a problem because in the virtual world the user does not have the clue how far he/she has gone because of missing of physical body movement. The sense of direction would be also missing if a controller, instead of the head is used to turn. Fortunately, depending on what the content is, there are many ways to provide the user with extra cues in order to be able to navigate through the virtual world. For example, if data is presented on top of a 3D geographical map because geolocational information of the data is to be revealed, then landmarks or simple markers on the map with noticeable colors and lightings would be helpful as an aid for wayfinding. For a flying experience (among the data), paths and routes may serve similarly to a landmark[55]. Other elements like routes,

---

[54] Darken and Sibert, "Wayfinding Strategies and Behaviors in Large Virtual Worlds.."
[55] Jerald, *The VR Book: Human-Centered Design for Virtual Reality*, 244.

channels, nodes and edges are also helpful in providing constrains and boundaries to the navigations thus the user would always have references for where to go. Apart from these, using breadcrumbs or trails could inform the user where he/she has been to. But too much of these would also clutter the scene. For example, as shown in the Figure 12 in an application which visualize the tweets in a campus is realized on top of a map texture layer and a 3D campus model[56]. By this way a user would know where to navigate and what data is related to what location even if the campus scale could be real-world size.

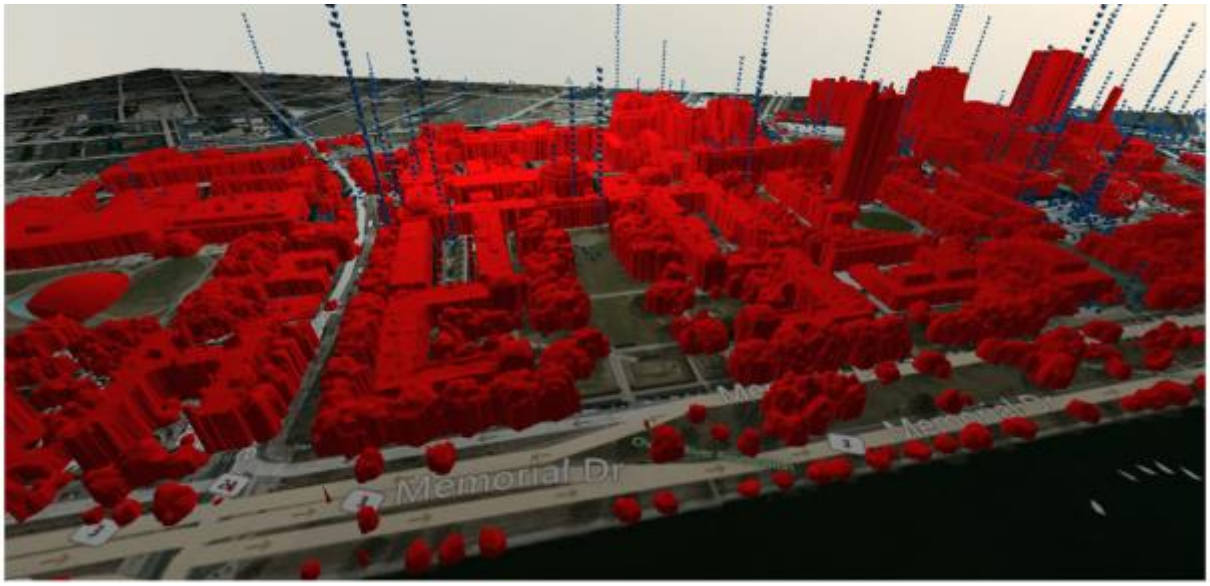


Figure 12 Elements like routes, markers on maps helps with spatial comprehension and providing constrains and boundaries

Real-world Contents

Last but not least, real-world contents can also be integrated in order to create a mix-reality scene. For example, the background of the scene could be captured using a 360-degree camera or a panorama photo taken from the real world and rendered in the scene as a "skybox", by which the realism would be highly improved.

---

[56] Moran et al., "Improving Big Data Visual Analytics with Interactive Virtual Reality."

2.1.2.2 User Elements

Personal Wayfinding

As described before, wayfinding is a significant mechanism to assist the user to locate him/herself. Other than applying this on the environmental elements, it is also extremely helpful when applied on user's personal sphere. For this purpose, mainly map and compass are applied. The map here is not the same as in the environment. It is a map that is carried around or follows the user. It is mini map that provide location information about the user. This map could be dynamic or in terms of rotation and scales. It could be always facing to a direction for example north, or could rotate as the user turns. The mini map could also make the navigation more effective when the environment scales so large that the user lose the direction and position. For example, the Google Earth VR application shows a mini earth (Figure 13[57]) on one of the controller that is always easily accessible by the user and it could be rotated with just pressing on the touchpad on the controller and user could then mark a favorite place or jump into a point on the earth using another controller. This is especially helpful when the user dives too deep into an environment.

[57] "Earth."

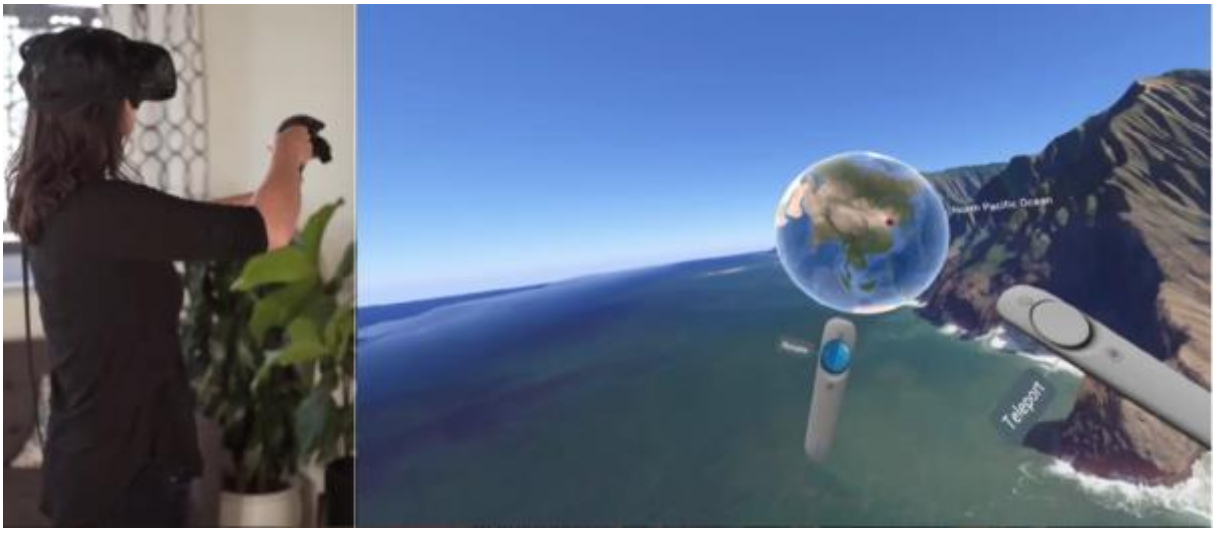


Figure 13 Wayfinding technique in Google Earth VR

Action Areas

Action areas is another point that should be considered when designing the contents from the user's point of view. Visionary VR and Mike Alger both proposed the concept that in VR, viewer would have different focuses on different areas around him/her. Visionary VR defines primary, secondary and tertiary viewing directions around the viewer (Figure 14) and suggests that the most important actions take place in the primary direction[58]. According to them, it is important that the contents should respond to the changes of the direction of view when it crosses the boundary and goes into another area.

[58] "Visionary VR Is Reinventing Filmmaking's Most Fundamental Concept to Tell Stories in Virtual Reality."

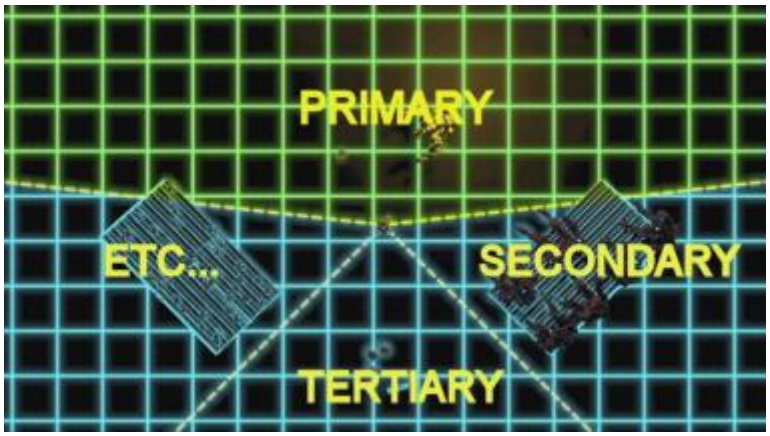


Figure 14 The action areas with different focuses

Alger proposed interaction areas that is optimal and comfortable for interactions with bare-hand controller[59], as seen below:

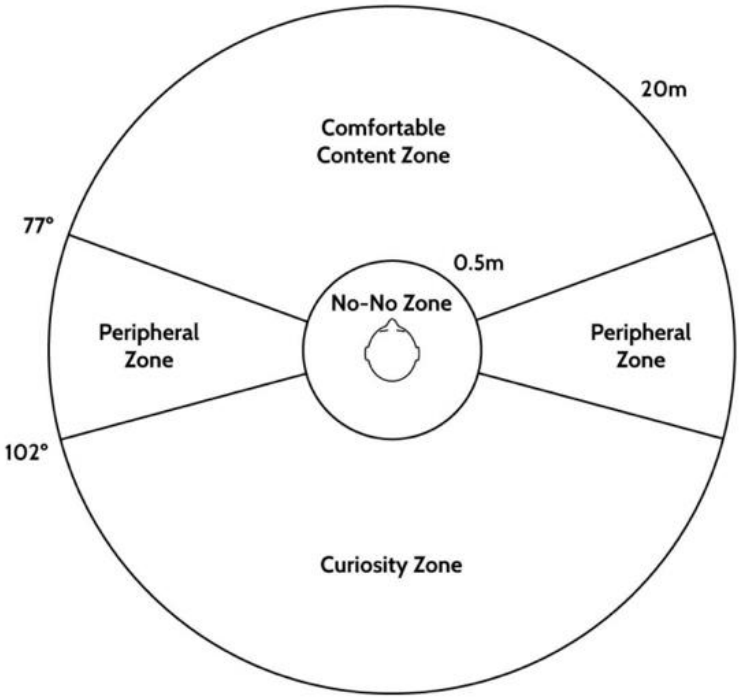


Figure 15 Different interaction areas for bare-hand controller - above

[59] Alger, "VisualDesignMethodsforVR_MikeAlger.Pdf."

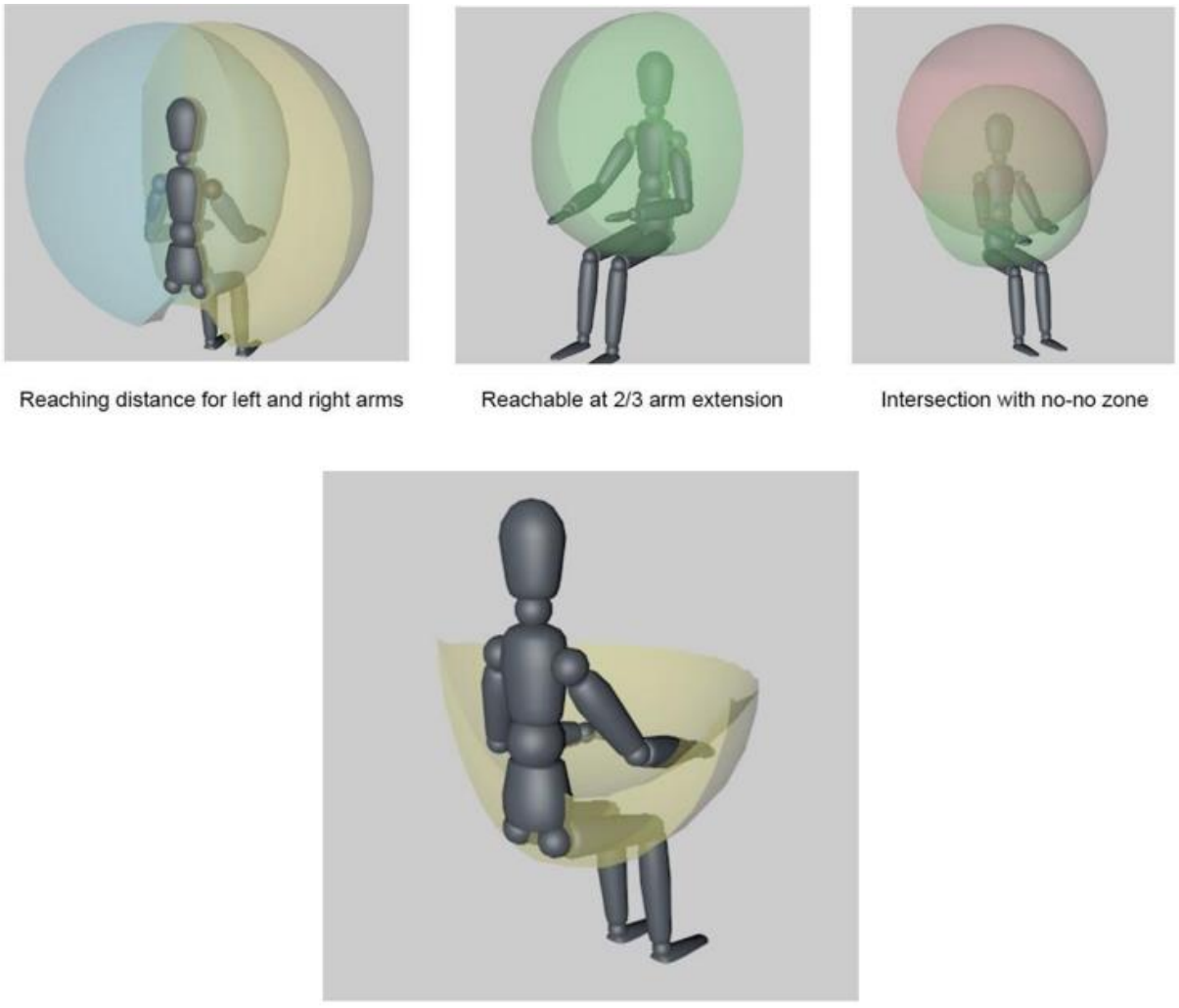


Figure 16 Different interaction areas for bare-hand controller - 3D

Those areas described above then suggest, when designing the contents, it is important to consider the way of interactions and how the interactions would change the course of the storytelling. If it is a sit-and-fix experience, then it is particularly important to put the major objects into the interaction-friendly zones and then position the secondary objects into peripheral zones or secondary zones. In the situation of data analytics, data visualizations that are to be interacted or viewed for longer time should be put in the comfort content zones and other assistive contents like

UIs, reference panels or information cards could be put in peripheral or second zones so that they will not confuse the user. If movement of dataset is included, then consider pausing the movement when the analyst looks away so that the changes of the data would not be ignored.

2.1.2.3 Summary

In summary, content creation in VR is far more versatile and substantial than in the 2D world. Simply porting the designs and contents from 2D will not be sufficient. Bearing the considerations regarding to environment and user itself should bring the contents design to the right direction. Additionally, in VR, contents are more interactive and have richer affordance to the user, so what are the right ways to incorporate interactions into the objects and user? What could the interactions in VR help data analysis? It is discussed in the next section.

2.1.3 Interaction Design

According to the definition mentioned in the section 1.1 What Is Virtual Reality, VR is a way of communication, is a medium that brings intuitiveness and immersion to the receptor. Interaction is the communication that occurs between a user and the VR application that is mediated through the use of input and output devices, according to Jerald[60]. Since human beings have certain mental models to perceive how the world works in a simplified way, when a person interfaces him/herself to VR, an intuitive and effective interaction design should be critical, otherwise the experience would raise confusion and frustrations which should apparently be avoided. More intuitive the interactions are, easier could the user jump into the VR world with the mental models that are shaped by existing past experiences. But this does not necessarily mean the design should always be realistic, because sometimes it could make problems more complex, unless it is an application that is used for training and requires for high real-world fidelity. Although for the purpose of data analysis realism is not essential, the intuitiveness of interaction is crucial because of the block-out effect in VR. A VR experience is expected to block out the real world because of the requirement of high immersion, therefore an intuitive interaction design that is understandable, predictable and easy to use is essential to keep the user feeling immersed and engaged. In this section, some interaction design concepts and patterns that are specifically applied to VR will be discussed

---

[60] Jerald, *The VR Book: Human-Centered Design for Virtual Reality*, 275.

2.1.3.1 VR Interaction Concepts

Interaction Fidelity

VR is a virtual world, where everything could be on one hand realistic and on the other hand abstract. For interaction, the degree to which physical actions used for virtual task correspond to the physical actions used in the equivalent real-world task is defined as interaction fidelity[61]. On the continuum of interaction fidelity, one side is realistic, on the other side is non-realistic. Although high interaction fidelity means high realism in the virtual world and it is understandable and familiar to users, it also makes the interactions cumbersome and unnecessary. Imagine flipping through a bunch of virtual document objects only to find a sales number in a financial statement in VR, it would be quite frustrating and meaningless. Low interaction fidelity on the other hand may have steep learning curve for the users to adapt to, but it is efficient and simple. For example, in the VR application in Google Cardboard, users could interact with objects by gazing at them or simply slide the button on the side of the headset. Somewhere in between sits the "magical interactions" where users make physical movements but the techniques make the user more powerful and give them enhanced abilities or intelligent guidance[62]. For example, in Google Earth VR shown in Figure 17, the user could fly over the earth by directing the controller. Or in data analysis the user could remotely grab a data cube for more information.

---

[61] Bowman, McMahan, and Ragan, "Questioning Naturalism in 3D User Interfaces."
[62] ibid.

Figure 17 A good balance between high and low interaction fidelity yields a "magical interaction"

Reference Frame

The second concept that fits the VR interaction is the reference frame. It is a coordinate system that serves as a basis to locate and orient objects[63]. A complete VR experience include these reference frames: real world, virtual world, torso, hand, head, and eye. Interactions and contents can be designed to be hosted in any one of them. By using high-end VR equipment like HTC Vive or Oculus Rift, the position and rotation of the user and controllers are tracked therefore real-world reference frame is used. However, since it is still static and will not be changed from the virtual world, it is recommended to design corresponding static rest frame in the virtual world in order to shield the user from motion sickness as discussed in the earlier chapter. But in the case of data analysis, since there will not be too much movements but more manipulations on the virtual objects or data points, it is more important to have a look at the virtual-world reference frames, torso reference frames, hand reference frames and head reference frames.

---

[63] Jerald, *The VR Book: Human-Centered Design for Virtual Reality*, 292.

Virtual-world reference contains a global view of the scene. In analytics, it is the world that shows the wide view of the data with global position, scale and rotation information. Analysts should be able to have a general perspective on the data in this reference frame and to dive into the data if he/she will. But how does the user interact with a graphical interface? Where should it sit? Torso reference frame is a good place for it follows the user and it is always accessible. For example, "body relative tool" is a technique, with which handful of frequent interact tools are bound to the body of the user, so that the user could easily access them and even develop a muscle memory. A fixed-location interface is not suitable in the virtual world because if the user moves to another position or dive into data, then the interface will not be access anymore. However, if the controller is designed to be always held by the user, then a user interface could also be placed in hand reference frame. But too much information or interface on the hand would also become overwhelming. What more suitable to be placed on controllers are tooltips and signifiers which gives basic information of the controller or possible interactions of a data point. For some VR applications that hand tracking is not possible like Google Cardboard, head tracking is more important. In this head reference frame, a reticle is usually in the center of the view and an invisible "ray cast" would be shooting through the reticle from the head so that it could detect what the object is being looked or gazed at, then further reactions are to be evaluated or triggered. Last but not least, the eye reference frame is where eyeball movements exist. Although eye movement may not be an ideal tool to select object because it could be misleading and meaningless[64], it is extremely helpful to reveal emotions in social scenarios.

---

[64] Jacob, "The Use of Eye Movements in Human-Computer Interaction Techniques - What You Look at Is What You Get.."

Speech and Gestures

Speech and gestures are other components that could be especially helpful in VR than in other medium. If using gesture detecting devices such as Oculus Action controller and Leap Motion Orion, gestures on one hand can simplify the interactions with objects because the user does not always need to reach for a control button or alike. On the other hand, when including other users in the virtual world, social communication would be dramatically improved by allowing users to express quick responses like "ok", "stop", or sizes in gesture. Speech does the similar job. It provides interactions in other dimensions and simplifies the communication. Not only conversations or instructions could be delivered through speeches, when speech recognition is implemented, the user could manipulate objects with speeches.

Multimodal Interaction

Multimodal interaction is especially interesting in VR. It combines multiple input and output sensory modalities to provide the user with a richer set of interactions[65]. There are 6 types of combinations: specialized, equivalence, redundancy, concurrency, complementarity, and transfer[66]. Specialized input limits the modalities that could be used, it is aimed for effectiveness of modality. Equivalent input modalities give users a few modalities to choose from for the same interaction. Redundant input modalities will combine multiple same modalities together in order to achieve an interaction goal. For example, if the user wants to delete a visualization from the scene, he/she has to say "delete" while press and hold the delete button. Concurrent input modalities make user

---

[65] Jerald, *The VR Book: Human-Centered Design for Virtual Reality*, 302.
[66] Laviola et al., "Whole-Hand and Speech Input in Virtual Environments;" Martin, "TYCOON: Theoretical Framework and Software Tools for Multimodal Interfaces."

available to perform more than one interactions simultaneously. For example, the user could aggregate the data cubes while changing their colors. Complementarity input modalities merges different types of inputs into a single command. For example, Bolt's "put-that-there" interface requires combination of voice command and gesture to move on object[67]. The last one is transfer, it occurs when one input modality transfers to another in order to complete the whole interaction, for example, a "push-to-talk" interface[68]. Multimodal interactions are especially useful when manipulating with data objects and selecting objects. When speech is included, it is also very helpful in shared virtual world because of unreliability of network connections.

---

[67] Bolt and Bolt, *"Put-That-There": Voice and Gesture at the Graphics Interface*.
[68] Jerald, *The VR Book: Human-Centered Design for Virtual Reality*, 303.

2.1.3.2 VR Interaction Patterns & Techniques

With the principles and concepts explained from the previous section, it is time to have a look how those concepts could be applied. When implementing the interaction designs, Jerald suggested a few interaction patterns which categorizes specific techniques that depends on available technologies into different purposes. In each pattern, it is a comprehensive application of the theories mentioned above through different techniques. By classifying those techniques, the following patterns are suggested: selection, manipulation, viewpoint control, indirect control, and compound[69].

Selection Pattern

When select, there are different ways to select, to summarize, there are the following sub patterns that are interesting: hand selection, pointing selection, and volume-based selection. Hand selection is realistic and straight forward, user uses the hand to touch a virtual object before the object is picked or grabbed. It is good for selecting controls or interface elements within the comfortable reach of the user as suggested by Alger in the "Action Areas" section in Content Design part. But this selection pattern has limited accuracy because it depends on the size of the hand and how user is used to the mapping of the real hand in the virtual world, therefore big buttons and large UI elements is more suitable for this kind of interaction. Pointing selection is quite often used in VR because objects could stand in further distance beyond arm reach. Even if the user could move in the scene, a pointer is still efficient to select the object that is far away. As Bowman suggested, pointing is faster when speed of remote selection is important[70]. There are also different kinds of pointing, for example, in Google Cardboard, user could use gaze pointer but in high-end VR

---

[69] ibid., 324.

[70] Bowman, *Interaction Techniques for Common Tasks in Immersive Virtual Environments: Design, Evaluation, and Application*.

system like HTC Vive, developer could attach a "laser pointer" onto the controller. There are also variants of pointing selection: object snapping that allows selection ray to snap/bend towards objects that are wanted[71]; and precision mode pointing that includes a zoom lens or makes the movement of the pointer slowly[72]. These variants allow pointing selection to be applied for smaller or further objects. A volume-based selection is especially useful for selecting group of data in 3D world. It uses a cube/volume to select the content that is included inside of the cube. It is similar to a box-selection in 2D. The Figure 18 below shows the cube selection technique in a game[73].

Figure 18 A volume-based selection pattern helps with selecting objects in 3D world

Manipulation Pattern

For manipulation, there are three sub patterns to consider: direct hand manipulation, proxy, and 3D tool. Direct hand manipulation is intuitive and efficient in terms of manipulation of positioning

---

[71] Haan, Koutek, and Post, *IntenSelect: Using Dynamic Object Rating for Assisting 3D Object Selection*.
[72] Kopper et al., "A Human Motor Behavior Model for Distal Pointing Tasks."
[73] "Red Alert 2 Remake in Unreal Engine 4 - Allies Gameplay in VR Using HTC Vive.."

and orientation, it would also result in greater user satisfaction than other manipulation patterns[74]. Proxy manipulation means the user would manipulate a proxy rather directly on the target object itself. It is helpful when the target object is too large or the user's viewpoint is within the object that the user does not have a global view of the target object but still wants to manipulate it. For example, in Google Earth VR shown in Figure 19[75], the user could rotate the earth proxy on the controller rather than directly the gigantic earth appearing in front of the face or within. 3D tool on the other hand allows user to manipulate an intermediary 3D tool with their hands that in turn directly manipulates the objects in the world, according to Jerald. It is convenient when the target object is far from the user or it is unintuitive to manipulate. For example, user could use a brush and palette to change an object's color, or use a remote controller to change the behavior of the object.

---

[74] Bowman and Hodges, *An Evaluation of Techniques for Grabbing and Manipulating Remote Objects in Immersive Virtual Environments*.
[75] "Earth."

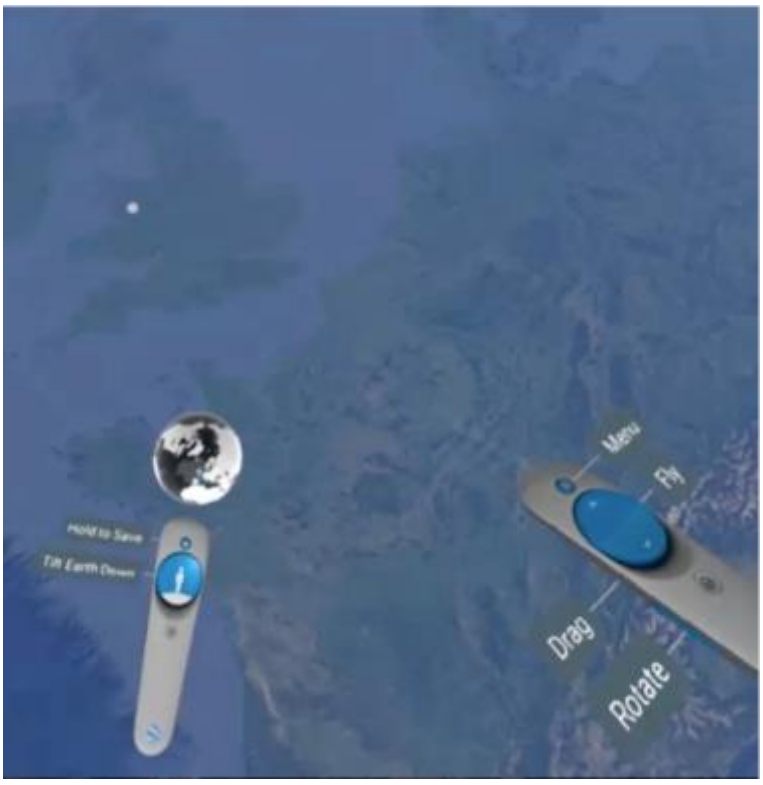


Figure 19 Proxy manipulation pattern allows user to manipulate a large object easily

Viewpoint Control Pattern

Viewpoint control pattern allows user to manipulate viewpoint's orientation, perspective, scale and so on. Controlling one's viewpoint is equivalent to moving, rotating or scaling the world[76], it includes the following sub patterns: walking, steering, "3D multi-touch" and automation. With the feasible technologies, walking would only be available in a room-scale high-level VR system. This sort of interaction allows the system to track the position of the user when he/she is walking in a certain area. It is intuitive and efficient, also enhances presence and ease of navigation[77]. It would be helpful when user wants to exam an object in virtual world while walking around it. Steering

---

[76] Jerald, *The VR Book: Human-Centered Design for Virtual Reality*, 336.
[77] Usoh et al., *Walking > Walking-in-Place > Flying, in Virtual Environments*.

pattern is continuous control of viewpoint direction that does not involve movement of the feet[78]. Since the high-end VR systems track the position of HMD, therefore the user could change the viewpoint freely by actually changing the physical location, but it is not the case of mobile platform such as Google Cardboard because of lack of body tracking. If moving a long distance is required for example, as described before, the user could fly through the world in Google Earth VR by directing the controller. For mobile platforms, it is better to use gaze-directed steering pattern, which moves the viewpoint when the user is looking at a direction with or without trigger a button. There are other techniques to steer viewer's viewpoint by using external controllers such as gamepad or joysticks, but those techniques could probably induce motion sickness if not tested intensively. 3D multi-touch viewpoint control is quite interesting according to an implementation by Digital ArtForms[79]. As shown in the Figure 20 (adopted from Jerald[80]), user could use one-handed dragging to change the viewpoint in the world, two-handed zooming to change the scale of the world, and two-handed steering to change the rotation of the world. It is convenient for user to navigate through the world by easily changing the viewpoint and how the world is presented. Last but not least, automation sub pattern also allows changes of viewpoint, but in passive. Since it changes the viewpoint passively, it has to be carefully designed otherwise motion sickness will occur. Teleportation is a common technique to implement. It teleports user's viewpoint when triggered. It is important to control the fade-in and fade-out effect when teleporting, if there is no such effect, user may feel startling. Transitional movements should also be voided when

---

[78] Jerald, *The VR Book: Human-Centered Design for Virtual Reality*, 338.
[79] Schultheis et al., *Comparison of a Two-Handed Interface to a Wand Interface and a Mouse Interface for Fundamental 3D Tasks*.
[80] Jerald, *The VR Book: Human-Centered Design for Virtual Reality*, 341.

teleporting because it would noticeably increase the risk of feeling nausea and dizziness although if implemented well, teleportation could actually decrease the motion sickness.

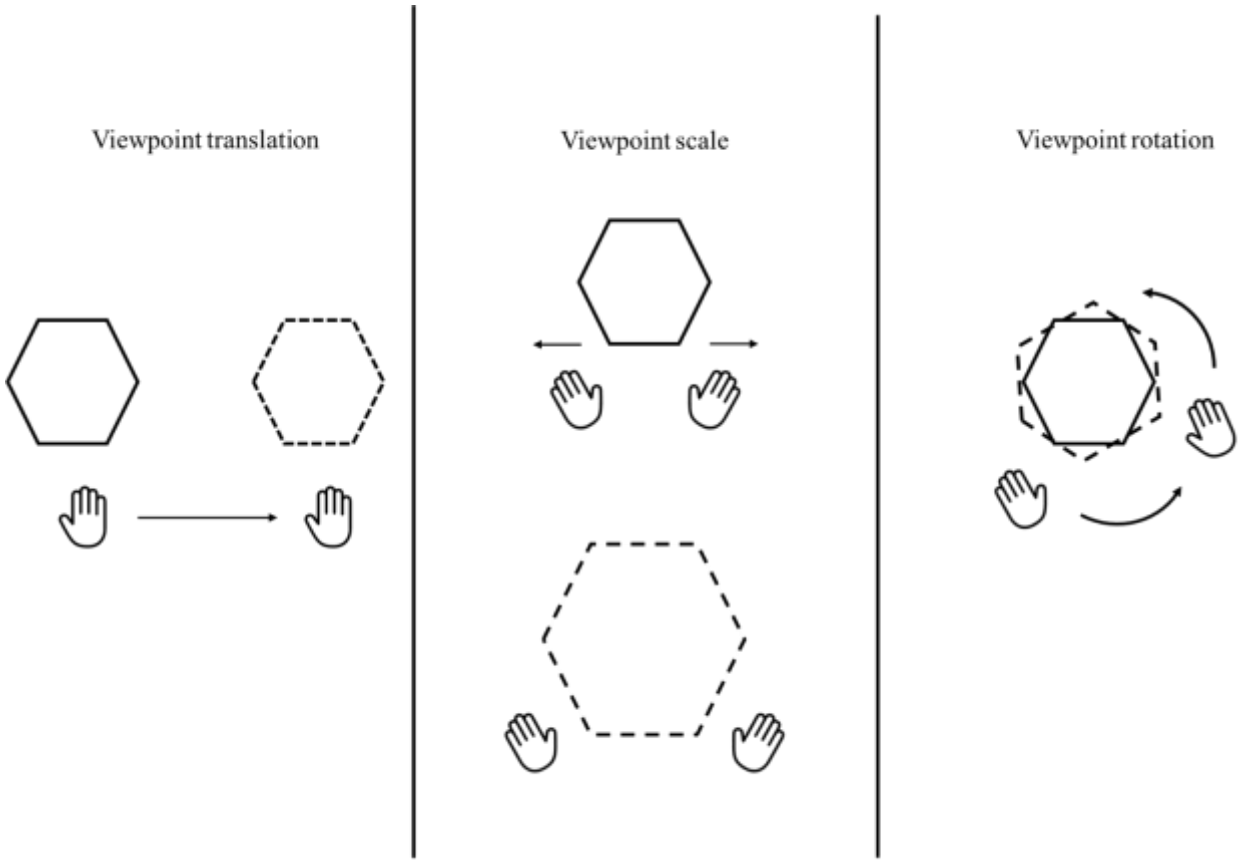


Figure 20 3D multi-touch view point controls

Indirect Control Pattern

In many cases, direct control over an object is not feasible, or the object is abstract and not visible, then intermediary control is in need. Indirect control interaction pattern then covers such situations. Jerald suggested two sub patterns for indirect control. One is widget and panels pattern. In this pattern, the user would use small widgets or control panels to interact with abstract objects such as toggling modes, changing values of properties, and issuing commands. Because the target is already infeasible, here by the intermediary object the signifiers have to be obvious and

informative. Techniques could include menus floating on the controllers or by the body, or widgets that are easily accessible and convenient to interact with. For example, the user could use ring menu on the controller to change audio clips or turn up/down the lighting of the scene. Another sub pattern to have indirect control is through non-spatial interactions. In this case, techniques such as speech or gesture control would be used. However, if the speech recognition is not accurate, the user would be easily frustrated, and if the gesture control is too often used, user could become fatigue. It is not a good idea to heavily depend on only the non-spatial indirect control without complimentary techniques, which brings to the compound patterns.

Compound Pattern

By using compound interactions, some drawbacks of certain interaction patterns could be compensated by others thus they could deliver better experience. Pointing hand pattern is one of the compound patterns. As described before, in order to balance between realism and usability of interaction, "magic interactions" are used. "Magic interactions" could be combination of hand pattern and pointing pattern. User could use real hand model in VR for manipulation meanwhile one of the fingers, the index finger for example, can cast pointing ray to interact with objects in the distance. By this way the interaction flow would be more smooth, efficient and enjoyable. Other compound patterns include for instances, world-in-miniature pattern, as shown in Figure 21[81] below, and multimodal pattern. The former combines proxy, multi-touch and automation so that user could switch between exocentric and egocentric viewpoints in order to manipulate objects

[81] "Figure1.Gif."

in a world from a broader perspective. Multimodal pattern combines different sensory modalities together to perform one or more interactions as described in the last section.

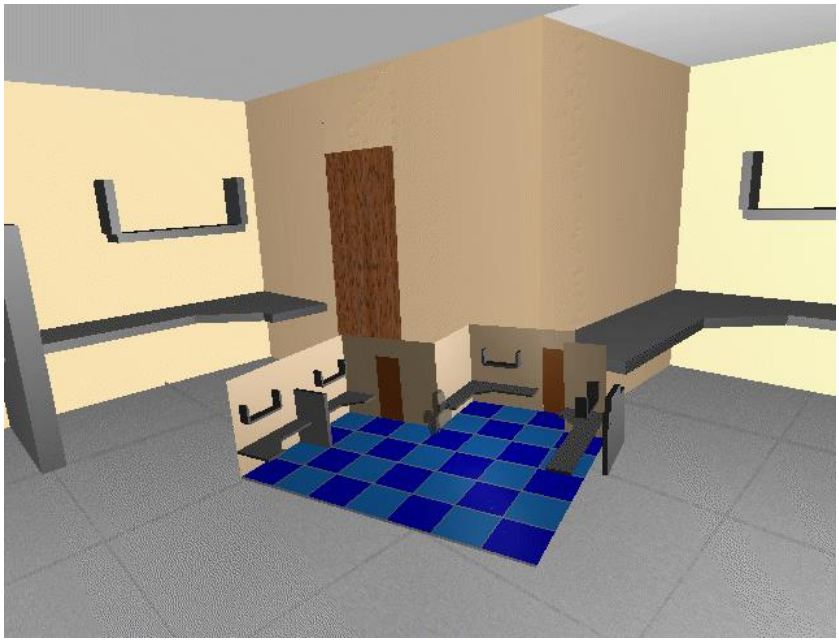


Figure 21 World-in-miniature pattern combines proxy, multi-touch and automation interaction patterns

2.1.3.3 Summary

In this part, interaction design in VR was discussed. Starting with high-level principles of general interaction, some specific concepts that are applied to VR application developments are then described. With those concepts in mind, practical implementation patterns and techniques were described. With all those principles, concepts, practical patterns and available technologies, it is then possible to deliver an enjoyable, rewarding and engaging experience in VR, even if it is about data analysis, certain level of gamification could keep the analysts focused and feeling joyful, thus curiosity shall be aroused and then motivate the user to discover more about data in the virtual world. In the next part visualization of data in 3D world will be discussed.

2.2 3D Data Visualization

VR world is a 3D world where the user could freely position the view point as in the real world. If data is to be visualized in this world, then it can be quite different from that in the 2D world because it could facilitate the extra third dimension thus provide another new perspective to data. Before discussing about the data visualization in 3D world, let us have a look at the data visualization concept in general. Then advantages and necessities of 3D data visualization in VR would be discussed with the drawbacks and the solutions on the other side.

2.2.1 Data Visualization Principle

According to Ward, a data visualization can be understood as a pipeline, as shown in Figure 22, containing the following stages: data modeling, data selection, data to visual mappings, view transformations and generation of the visualization.[82]

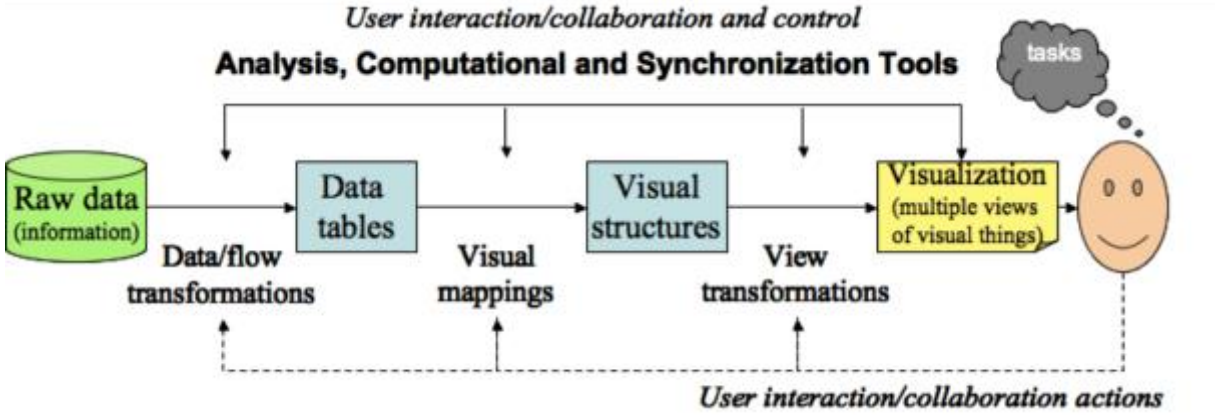


Figure 22 The data visualization pipeline

Data modeling is the data to be visualized as, but should be structured or modeled so that it could be accessed easily and mapped. Among those data that to be visualized, a certain range of data could be selected to be displayed while another part of which becomes hidden but still loaded. For example, user could zoom in into a range of data while the data that is out of focus is not shown. Then the selected data should have visual mappings to the world that is to be shown, including positions, rotations, scales and so on. Conventionally the data then would be transformed in terms of color, sound, lightings and so on that are relative independent attribute of data. In VR, these information means the potential affordance or signifiers of data objects. In the next step, the

---

[82] Ward, Grinstein, and Keim, *Interactive Data Visualization*.

visualization should be rendered for display, textures or shadings would be applied. Finally, the viewer would perceive the visualization. Ideally, the information that flows from the beginning of the pipeline to the end should maintain a high level of fidelity that it delivers the information the user is expecting to know without misinterpretation, distortion and noises. People's perception to visual signals depends on simplicity of the object and the perception has different sensitivities to different level of attributes. The whole perceptual process would first focus on low-level attributes (color, hue, 3D depth cues, size etc.), then turn to higher level ones (patterns, groups, interested information etc.), and finally put it all together with memory models[83].

[83] ibid.

2.2.2 Data Visualization in 3D World

3D world can not only utilize depth cues to catch attention, the 3D visualizations in which can also assist in better understanding when the model becomes complex. According to a research done by Amini et al., as a visualization of spatiotemporal data shown in Figure 23 and Figure 24, "3D space-time cube style of the visualization is beneficial specially when users have to examine sequences of events to identify a complex behavior within object movement data sets" [84]. Although some users are feeling the 3D style "busy" and "messy", they also responded that, questions about the data are easier to be answered with 3D and the technique was perceived "very cool". In VR world, the interactions can be even easier because the user have to rotate and pan the 3D visualization in a 2D display using mouse and keyboard, but it would be more intuitive in VR.

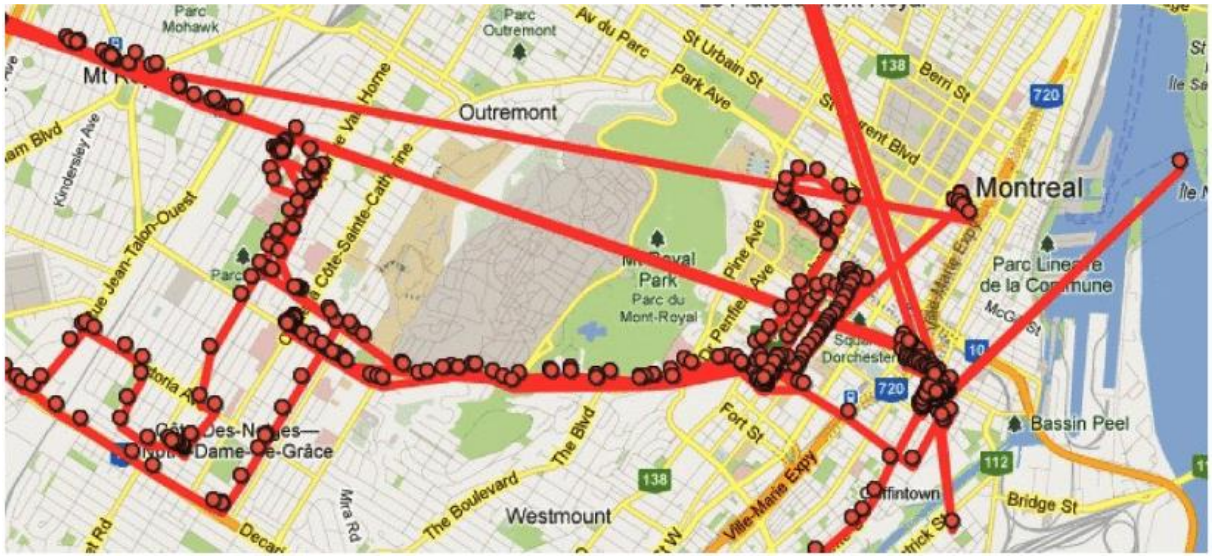


Figure 23 Movements of a smartphone over seven days, as captured by the Backitude application for Android, sampling at 15 second intervals, and subsequently visualized with Google Latitude. Gaps and discontinuous jumps in the data are caused by traveling through

[84] Amini et al., "The Impact of Interactivity on Comprehending 2D and 3D Visualizations of Movement Data."

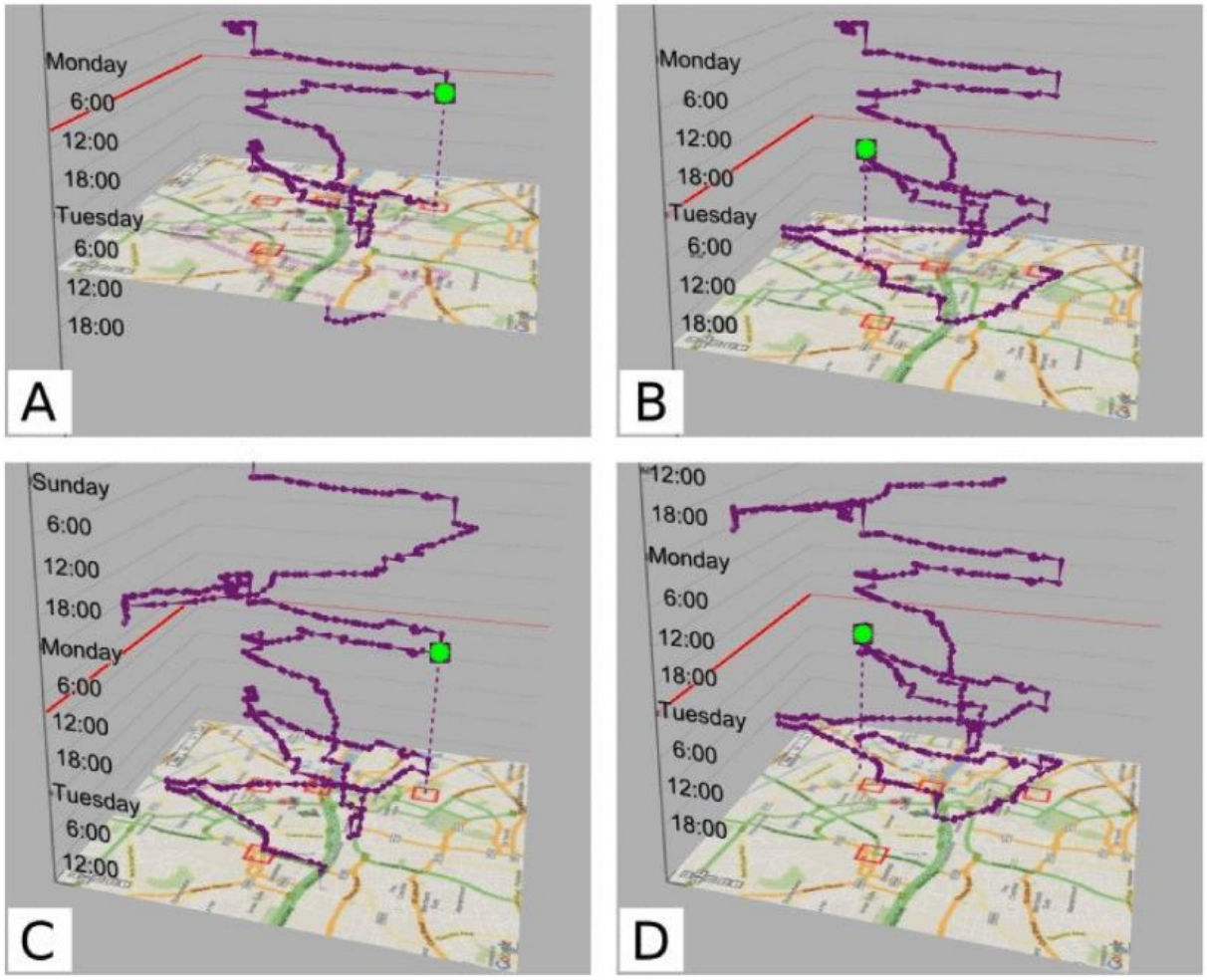


Figure 24 Two design options of the spatiotemporal visualization. A and B show a "moving plane": the ground plane, red reference line, and vertical position of the green icon all move downward as time advances. C and D show a "moving trajectory": the ground plane, red reference line, and vertical position of the green icon are all fixed, but the trajectory moves upward as time advances. A and C show Monday morning at 10:00, B and D show Monday evening at 23:25

When the data analysis involves data that is of huge amount, high complexity, and high dimensions, visualization of this kind of data can also facilitate 3D visualization in VR. As mentioned before, the team led by Donalek tried to visualize a dataset with 8 dimensions in VR, as seen in the Figure 25, by utilizing parameters: XYZ spatial coordinates, colors, sizes, transparencies and shapes, textures, orientations, rotation and pulsation. Additionally, user could have an avatar in the VR world and an interface to change the dimensions of the data. By a map-drawing experiment, they

also found that scientists seem to perform better in the immersive environment compared to a 2D condition[85].

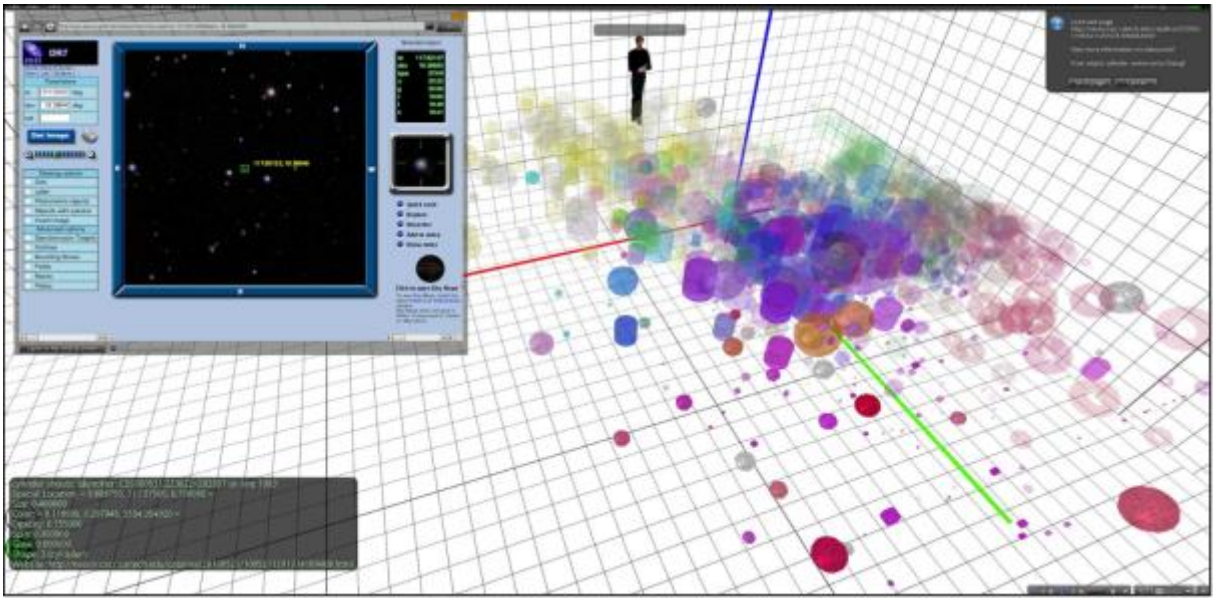


Figure 25 A 3D visualization in VR visualizes 8 dimensions of a dataset

3D visualization has been applied to many scientific researches where multidimensional data is commonly used, until recently, there was an application that tried to find out the convergence of 3D visualization and business intelligence, specifically in Online Analytical Processing (OLAP) cube development. According to the research from Su Mi Ji et al., they were utilizing 3D data visualization to build the OLAP cube (Figure 26) from a 2D user interface where they could select cube dimensions and measures, then on this cube user could do common operations in business applications such as slice and dice, drill down and up, until the final reporting visualization is presented. Then in this visualization there are 3D bar maps that can be interacted with selecting, emphasizing, relevance comparison or searching[86]. In this way, a business analyst could easily and

---

[85] Donalek et al., "Immersive and Collaborative Data Visualization Using Virtual Reality Platforms.."
[86] Ji et al., "A Study on the Generation of OLAP Data Cube Based on 3D Visualization Interaction."

intuitively build the data cube with less technical knowledge because the visualization greatly simplified the cognitive process. It would be even better when in VR since user could have more intuitive interactions with the cube and visualization.

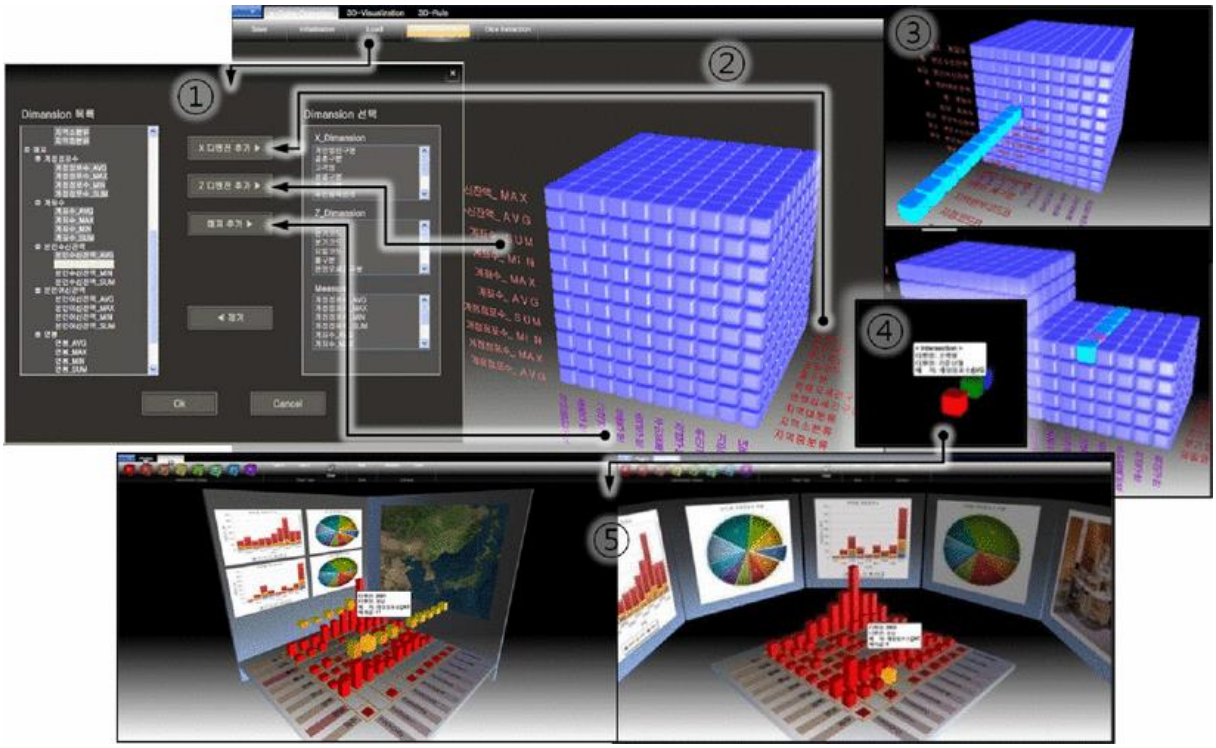


Figure 26 3D visualization of OLAP cube

However, traditional 3D visualization does not come with no drawbacks. On one hand, according to a perceptional theory called Stevens' Law, using attributes such as the volume of 3D object to convey information is much less effective and much more error-prone than using area or, length[87], as seen from Figure 27.

[87] Ward, Grinstein, and Keim, *Interactive Data Visualization*.

Figure 27 3D object conveys less effective information

On the other hand, perspective and occlusion in 3D world could also bring troubles when investigating the visualization, seen from Figure 28[88]. Perspective may distort the size of the object and occlusion may simply block the view.

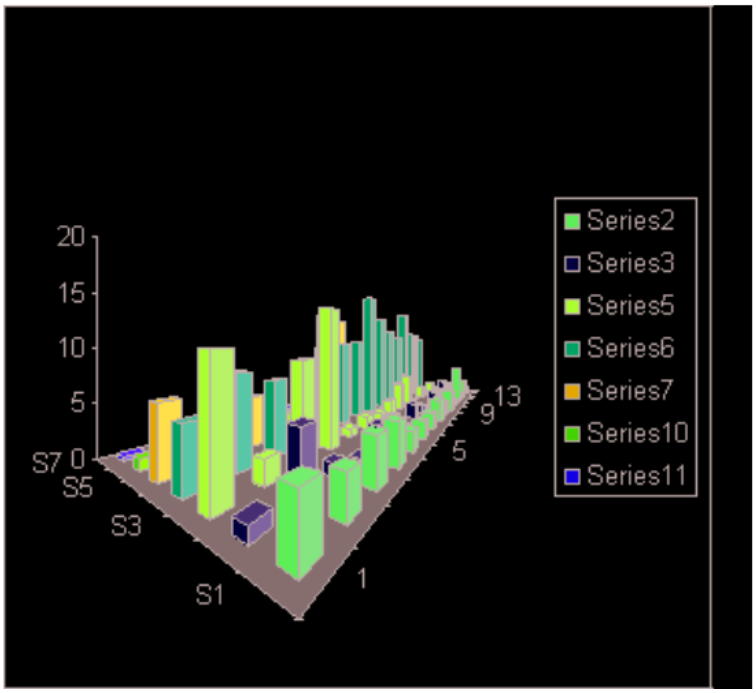


Figure 28 Perspective distorts the size of the object and the objects can block each other in the view

[88] ibid.

But those problems are resolvable. For example, if user finds the perspective distort the shapes, or the depth cue is misleading, it is better to introduce a "cutting plane" or "cutting cube" mechanism which it could cut through the 3D visualization to show the 2D interpretations on planes where they collide. By this way, the user will not lose the bigger picture where shows the patterns in a 3D world, but could also investigate deeper with high perceptional fidelity. As for occlusion, texture changes can be applied to signify the information that is to be required from user, which can be achieved by highlighting on the objects of interesting. For example as the Figure 29[89] shown below:

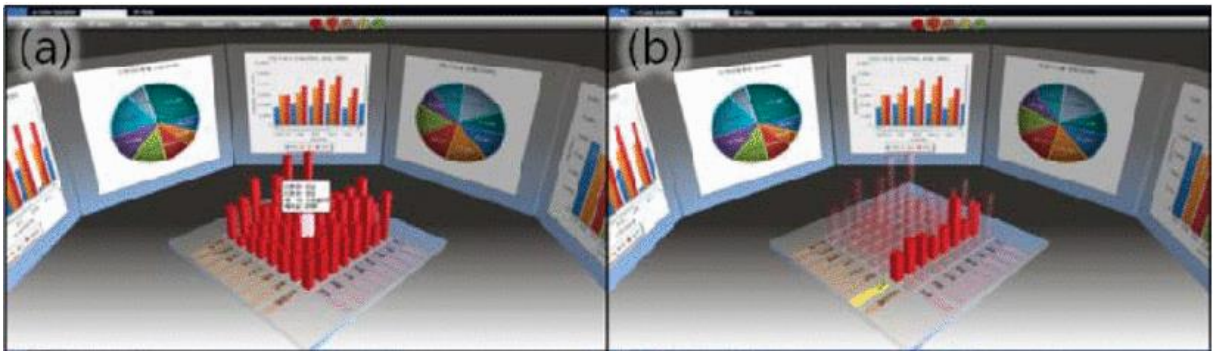


Figure 29 Cutting plane and cutting cube is helping to solve problems brought by 3D visualization

[89] Ji et al., "A Study on the Generation of OLAP Data Cube Based on 3D Visualization Interaction."

2.2.3 Summary

To summarize, 3D visualization is a compelling way to visualize data, combined with VR interactions that are well designed, the comprehension of complex, multidimensional abstract data can be greatly improved. Apart from scientific applications, business application with highly interactive data visualization is also regarded as a positive influence on daily work practice as reported by some IT experts[90]. A well implemented 3D visualization with VR interactions is expected to deliver a more positive experience and better performance in data analysis than traditional reporting and charts. Other than that, VR as a new way of communication can greatly enhance how people comprehend data in 3D world by including social factors, and it plays a unique role comparing to normal 3D data visualization. In the next part, social factors in VR are discussed.

[90] Aigner, "Current Work Practice and Users' Perspectives on Visualization and Interactivity in Business Intelligence.."

2.3 VR Social and Collaboration

VR is a medium that allows multimodal communications that offers possibilities for multiple users to collaborate within a 3D virtual environment. Different from other mediums for communications such as voice and video callings, VR can provide relative high realism or sense of presence when compared with others by embodying user into an avatar that could interact with objects and other users. By telepresence, users could share the same virtual environment as in the real world and could have better comprehension in communications because they are able to share the same view angle and viewpoints. Additionally, VR collaboration could also incorporate verbal communications to boost the effectiveness of communication, when multimodal stimuli are present, the interactions would be greatly compensated as described before. So, what decides a good and effective VR social and communication and what technologies are required to deliver such experience?

VR realism is an important factor to consider in VR social and collaboration. According to Bailenson et al., the realism of avatars in collaborative VR is critical[91]. The realism can be divided into form- or physical realism and behavioral realism. Form-realism means how the avatar physically looks same to real human and behavioral realism focuses on if the avatar acts the same as human, including emotions and movement. It is found by both Bailenson and Guadagno and their colleagues, behavioral realism is more important, because when it increases, the social presence in VR world also increases[92], and people tend to disclose more when the behavioral realism is high, thus more effective the communication with the avatar is. By more effective

---

[91] Bailenson et al., "The Effect of Behavioral Realism and Form Realism of Real-Time Avatar Faces on Verbal Disclosure, Nonverbal Disclosure, Emotion Recognition, and Copresence in Dyadic Interaction.."

[92] Guadagno et al., "Virtual Humans and Persuasion: the Effects of Agency and Behavioral Realism."

communication, or with higher behavioral realism, people tend to also perform the task better[93] and response easier[94]. Ideally, if both high form realism and behavioral realism need to be achieved, then there should be perfect 3D model of the users and perfect mappings of the behaviors including facial expressions and movements in the virtual environment. However, most of the time it is not feasible and just not necessary. To achieve an effective co-presence or telepresence with persuasive inter-avatar communications, a lower level of form realism can be designed. However, it should not fall into the Uncanny Valley shown below in Figure 30[95]. The x-axis indicates the degree "human likeness" or form realism, the y-axis indicates the familiarity of the object. As seen from the figure, 100% human likeness will bring the highest level of familiarity, thus the user will feel he or she is talking with a real human being. But the "uncanny valley" indicates that, although an object is relative similar to a human, but it is scary or not familiar at all, such as corpse or zombie. To avoid this, it is better to have even lower realism to maintain a better balance between human likeness and familiarity, such as humanoid robot or stuffed animal as an avatar.

---

[93] Garau, "The Impact of Avatar Fidelity on Social Interaction in Virtual Environments."
[94] Bente et al., *Bente: Measuring Behavioral Correlates of Social... - Google Scholar*.
[95] "Corpses, Androids, and the Polar Express: a Social Neuroscience Perspective on the Uncanny Valley."

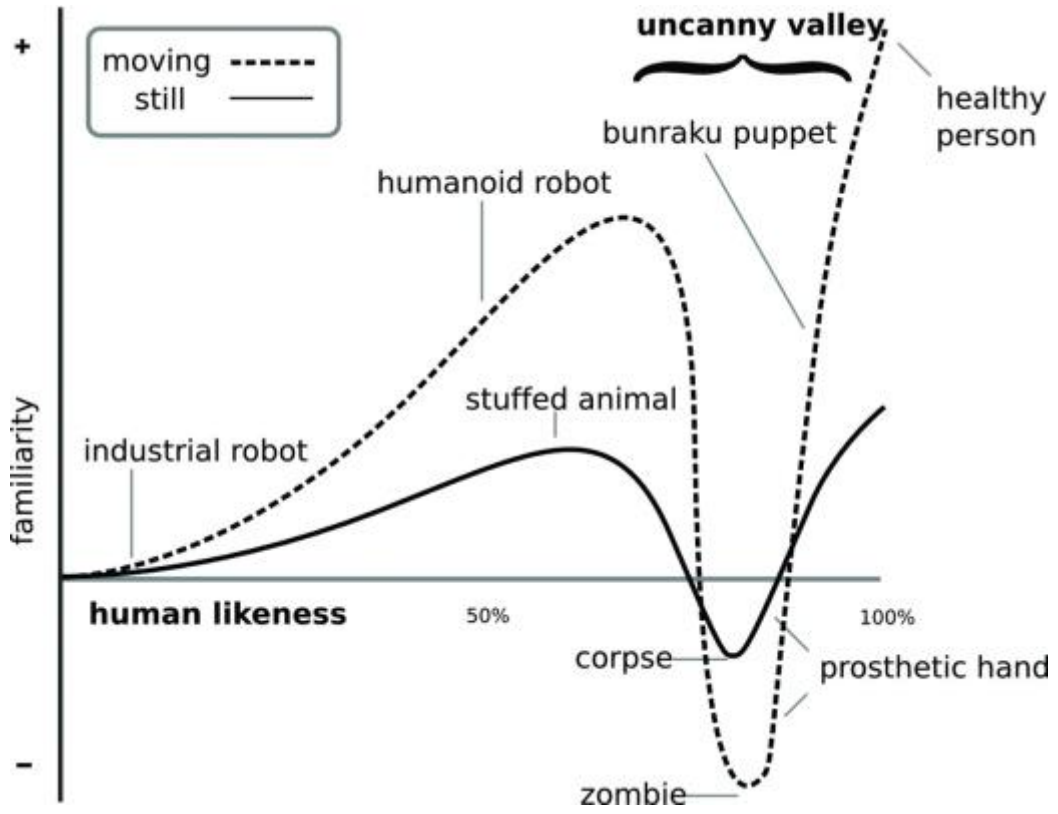


Figure 30 The Uncanny Valley should be avoided in the relationship between familiarity and human likeness of an object

However, a perfect system is not available in the market now that could track the whole body, face and eyeball movements to provide every social cue in VR. Among the technologies in the market now, Oculus' mother company Facebook has made its way in designing VR social and collaboration. After some experiments, it concluded its approach in developing a comfortable, charming and effective social experience with the following elements: speech that has spatialization, 1:1 tracking on user's behaviors, gesture detecting from the controller, eye contact or blinking (not tracked for now), gaze following to the people that is referred to, lip synchronization, emotions that are triggered by gestures (because of lack of tracking), arms and

body that provide physical realism and sense of presence[96]. Facebook's solution (Figure 31[97]) also includes a few other interactions such as video calls in VR, shared games, shared teleporting, shared video, shared space marker, and even group selfie in VR. Other than that, Oculus also introduced its own avatar solution (Figure 32[98]) that embodies the user in a diminished humanoid form with a "projection" metaphor telling the position. It also finds a method to avoid the problem of lacking eye tracking by giving eyewear for its avatars. As for lips' movement, it uses lighting/wave signifiers to represent the ongoing speech from the user. Oculus Avatars is available as an SDK and can be implemented in Unity3D.

Figure 31 Taking selfies with video calls in VR by Facebook

Figure 32 Oculus Avatar SDK that embodies the users in a projection metaphor

96 "Facebook Details Social VR Avatar Experiments and Lessons Learned – Road to VR."
97 "Facebook Social VR Demo - Oculus Connect 2016."
98 Oculus, "Oculus Avatars: Maximizing Social Presence."

Part 3 Project Implementation

After the description of the concept of VR and its design principles and techniques, this part will demonstrate a prototype implemented with guidance from the theories. This part comprises of the following aspects: project introduction, hardware solution, software environment setup, application architecture, and content design and implementation. To help explain the implementation better, this part will include snippets of codes. The complete project can be found in the attachment submitted with this paper.

3.1 Introduction

In the previous parts, it is concluded that 3D data visualization will benefit from VR user experiences and be enhanced through this revolutionary way of communication. In virtual world, spatial relationship is one of the relationships of data that can be visualized most intuitively and can take full advantage of the third dimension. Data points mapped in the virtual world can also be better perceived by audience if there are strong spatial relationships among the data points. With regard to this consideration, the data with geographical information are chosen to be visualized in this implementation, accompanied with two different forms of map. The goal of the implementation is to enable user to choose a dataset that is desired to be visualized and then plot the data onto one of the two sorts of maps hence to achieve better understandings on the geospatial data. With VR interaction techniques, the user can easily change the viewpoint and interact with the data points to explore more details that are concealed. From this example of realization, some advantages and disadvantages of VR application in data visualization and analysis comparing with it in 2D will be discussed.

The current implementation is based on an earlier Google Cardboard VR application which was used as a technical showcase It only included simple interactions because of the limitation of this mobile platform. The latest implementation used the full-fledged high-end room-scale VR technology and delivers an experience that are greatly differentiated from a simple mobile VR application. It is also a complete system that contains a client or front end and a server or a backend that feeds the raw data. In the next sections hardware components are introduced, followed by the development environment, system architecture, and content design and realization.

3.2 Hardware Solution

For the Google Cardboard VR application, it runs on mobile platforms, in this case an iPhone 6, and can be easily ported to Android with some adjustments on graphical effects due to performance requirements. The more interesting setup is the room-scaled VR application. As mentioned before, a complete VR system is comprised of the following parts: tracking, application, rendering, and display.

As described in the first part, this room-scaled application runs on HTC Vive VR system which provides head and body tracking through HMD and laser sensors, and hand tracking through handheld controllers. The refresh rate of the HMD is 90Hz, which means the frame rate of the application during runtime should be kept above at least 90 frames per second (FPS) for optimal experience. Those are the tracking and display components.

The application and rendering components are hosted in a computer, in this case, a high-performance laptop due to business requirements that the system should be portable in order to demonstrate in different locations. The current machine used for running the application and rendering is Schenker with specification[99] that meets the minimum requirements of HTC Vive VR system[100]. The computer is connected with the HMD through a link box that has HDMI and USB ports, and it is also connected with laser trackers (Lighthouses) by another two USB ports.

The HTC Vive is more favorable than Oculus Rift is because HTC Vive provides a complete solution that includes handheld controllers and better tracking technologies. Until recently and

---

[99] "U716."

[100] "VIVE™ Vive Ready Computers."

during the development of the application, Oculus Rift did not support handheld controllers that are designed specifically for VR usage, instead, an XBOX gaming controller is included in the box for basic steering. As discussed in part 2 of this paper, because of this indirect control mechanism, user may easily experience adverse health effects and also will break out from the immersive experience. Although the Oculus Rift could pair with Leap Motion's bare hand interaction solution, the technology was not mature enough and the tracking on bare hands is not accurate and stable, meanwhile it did provided better immersive experiences. Another issue of Oculus Rift is the position tracking technology. Since it uses only external camera as an optical solution, the camera has much smaller field of view, it is not stable and accurate as HTC Vive's laser tracking system. The next section will introduce the development environment.

3.3 Development Environment Setup

As mentioned before in part 1, the VR applications are commonly developed using a game engine. A game engine is an integrated development environment (IDE) that consolidates all the resources that are to be used in a game-like program and finally build run-time executables for end users. In this VR application, Unity 5 is used for development. It is easy to learn, well documented and continuously supported by both Unity and developers' community. It also provides good native supports for VR development by integrating plugins and SDKs into the environment.

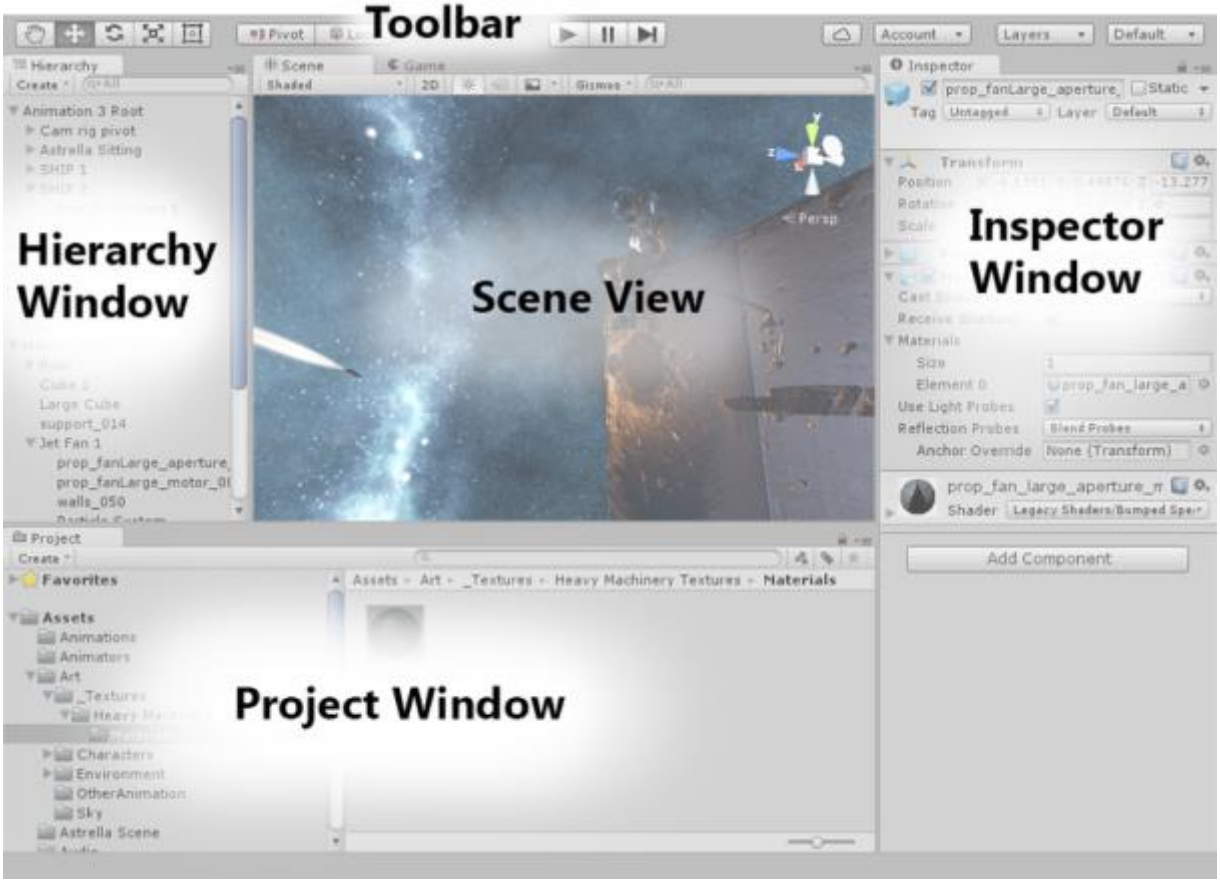


Figure 33 Unity editor provides an integrated user interface[101]

[101] "Editor-Breakdown.Png."

Unity has a user interface that integrates all the useful and frequently used components for developers. As seen from the Figure 33 above, it provides views on the current scene, the project resources or “assets”, component hierarchy, and the properties of components. During the development, the developer will design the scene by dragging 3D/2D objects from the Assets folder into the scenes and position them, manipulate and modify them accordingly. Basically, assets in Unity are just objects, they include 3D/2D objects, multimedia objects, animations, textures, scripts and any other kinds of objects that can be used in the game or application. Unity also provides a AppStore-like marketplace called Asset Store that allows other developers to publish their pieces work of any kind of assets so that they can share with the community and also monetize their efforts. In this way, Unity democratizes game development and becomes one of most supported game engine. In the editor, developer can see how elements in the component hierarchy are related to each other in the current scene. Since Unity uses the component-based architecture, the relationships of components are hierarchical. Each component also has its own properties, and it follows the object-oriented concepts in computer science. More details about components, properties and application architecture will be introduced in the next section.

After the developer sets up all the objects in the scene like a choreographer or director, he or she can then continue to design how the objects will behave by going into codes and writing scripts in either C# or Javascript. Once behaviors and logics are designed and attached to the objects or placed in the scene, the developer can run the application directly from Unity’s editor to see how it functions just like run time, although the application is not built and packed up to be published. Unity also provides multiple target platforms and various settings for building the application. For example, if the target is mobile platform like iOS, Unity will use the lower graphical settings and

then build to convert the whole Unity project into an Apple's XCode project that could be opened by XCode and then deployed to an iOS device; but if the target is a standalone PC application, Unity will build the project into a Windows executable file with a folder of resources, then the user just need to double click the .exe file to fire up the application. This approach also makes Unity a game engine that supports multiple platforms ranging from web, mobile to desktop and console.

So how does Unity work in the context of VR development? It is time to have a look at the supports provided by SteamVR and VRTK.

3.3.1 OpenVR and SteamVR

Unity started to provide native built-in VR supports since version 5.1, in the latest version it already supports the main VR devices in the market. That means, from Unity, a developer can develop a VR application just by adjusting a few settings from the editor, the Unity will take care of the rest such as: automatic rendering to a HMD, automatic head-tracked input, and position of the camera in the scene. In the Unity version 5.5.2 it already supports OpenVR and Oculus SDKs. However, if developer wants to utilize more features, such as spatial audio, controller model in VR, or other extended functionalities such as social/multiplayer supports as mentioned in part 2, device-specific SDKs or utilities packages are usually implemented on top of the base support.

OpenVR is one of the base APIs that are integrated in Unity in order to provide a device-independent support for interfacing the software and hardware. It "provides a game with a way to interact with Virtual Reality displays without relying on a specific hardware vendor's SDK" [102]. In order to access more functionalities and supports, the current project also integrated SteamVR as a Unity asset package which includes more scripts that wraps OpenVR. As an essential part of VR development, it provides a single interface that will work with all major VR headsets and will provide access to device controllers, chaperoning, and render models of tracked devices. Figure 34 provides an overview of how the SDKs work together in order to enable VR development.

With the support from Unity, OpenVR and SteamVR, a normal 3D application can be easily ported into VR settings, but to realize more mechanisms such as physics, interaction mechanisms among objects and other higher-level primitive manipulations, it is better to utilize an interaction/physics

---

[102] "Openvr: OpenVR SDK."

library that actually provides a convenient way to achieve solid immersive experiences. In this project, Virtual Reality Toolkit (VRTK) is chosen as an interaction library.

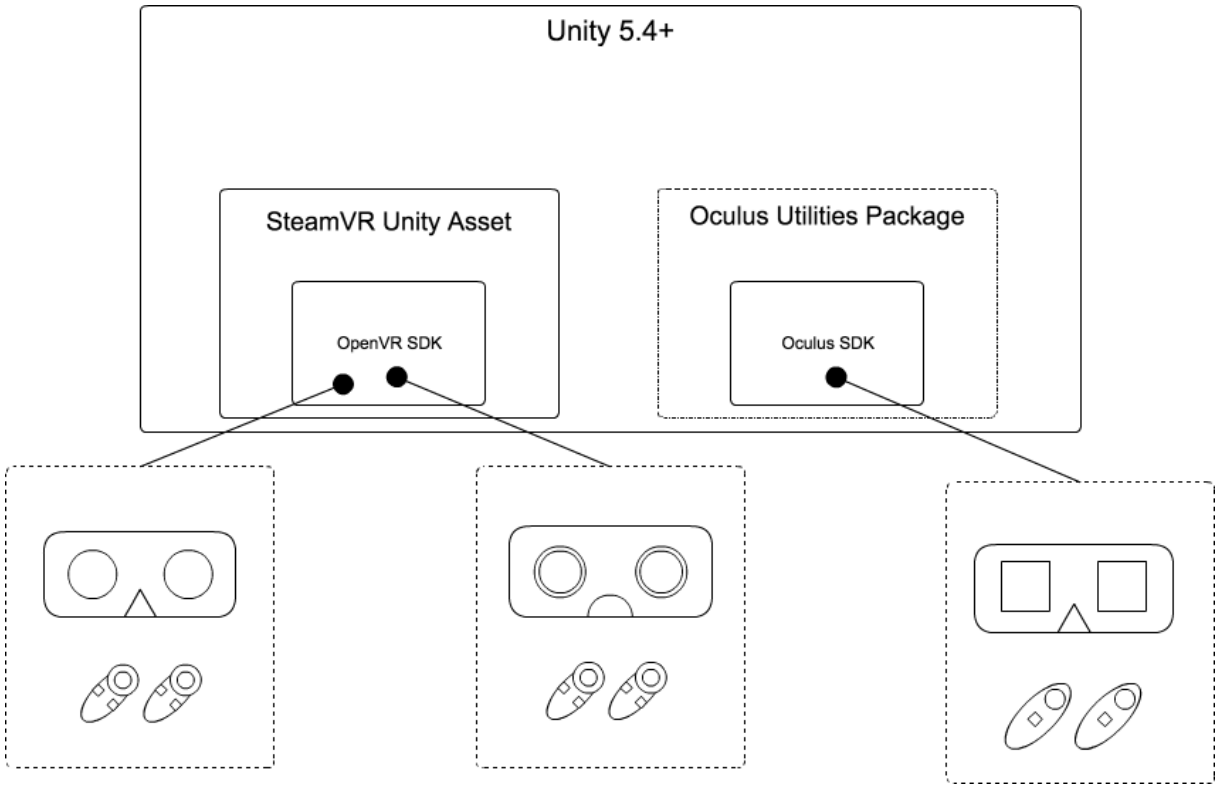


Figure 34 . Base SDKs connect the hardware and Unity assets or packages provides methods for high-level reference

3.3.2 VRTK and NewtonVR

Since VR development is still at an early stage and the threshold for entering this field is relatively high because of the cost of the devices, there are not many choices out there as free library that provides a robust, concise and easy-to-maintain solution package for VR developers. Two of the great choices as a VR development essential toolkit is VRTK and NewtonVR. These two packages both provide essential physical solutions regarding to interactions among objects. For example, with either of them integrated in the project, developer can easily implement basic interactions like grabbing, throwing, touching and rotating objects using VR controllers. Another great reason to use the library here is flexibility. As mentioned in the last section, VR devices are driven by the base API provided by Unity and wrapped by hardware vendor's utilities. Then the interaction library references the utilities and provide high-level APIs for developers to implement interaction methods. Since developer can easily configure which vendor's utility to reference through the interaction library, it actually decouples the device from the application, in other words, by using the interaction library, developer does not have to worry about the hardware platform anymore and can focus more on how to realize interaction mechanisms, which makes the future development more agile and flexible. In comparison of VRTK and NewtonVR, although NewtonVR provides slightly more physics mechanisms, VRTK does better in other features other merely just physics, it is a more complete package. Figure 35 depicts an overview of how an interaction library wraps up the vendor-based utilities and provides high-level implementations in the Unity scenes.

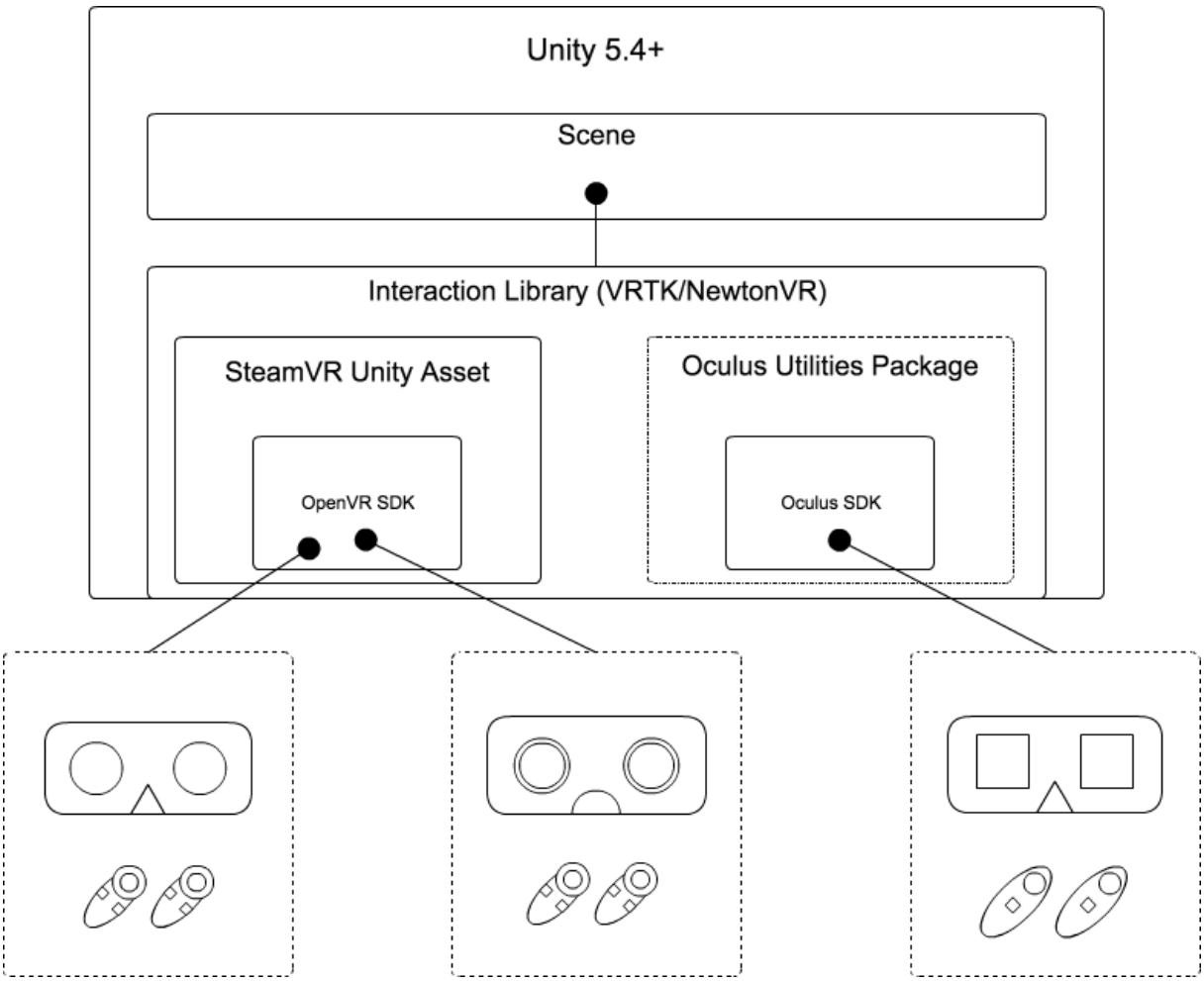


Figure 35 Interaction library provides high-level interaction implementations which are decoupled from the vendors

Other than basic interactive physics, VRTK also provides simplified implementations on locomotion in VR, controller pointers and other useful VR integrated functionalities such as tooltips on objects, VR-friendly user interfaces and controls, event-triggered haptics settings, and so on. Under the hood VRTK depends on either SteamVR Unity Asset or Oculus Utilities Unity Package (with Google VR SDK as experimental support). It means that developer can directly implement APIs provided by VRTK instead of deeper SteamVR's APIs to realize a series of interactions, which makes development more efficient and less prone to errors, meanwhile also

less customized. Another reason VRTK is chosen as a higher-level library is that it provides rich tutorial examples and well-maintained documentations, it flats out the learning curve dramatically.

With these tools on hand, the development for VR specific application is ready to go, in the next chapter the architecture of the whole application will be introduced, from which some more detailed implementations of the VRTK will be described along with interaction events.

3.4 Application Walkthrough

Till now there are two versions or iterations of the prototype on HTC Vive platform. One is called “Poweruser Version” with advanced GUI on which users can choose aggregation methods and aggregated dimensions of the data that is to be visualized. Another is called “Showcase Version” and is a simplified version that just applies the SUM aggregation method on the dimensions. User will only need to select what dimensions to be visualized.

3.4.1 Version Poweruser

The VR data visualization application starts from the following scene in the first and more complex version for power users as shown in the figure below:

Figure 36 The application starts in a familiar office scene

The scene is in an ordinary office conference room, in front of the user there are two tracked and projected HTC Vive controllers with tooltips on them and a GUI on a computer to simulate the real-life situation. This design provides a relatively high interaction fidelity because it assembles the familiar real-life environments. Then the user could pick up the controller and press the button on the controller to toggle an on-controller GUI (Figure 37). This design utilizes the reference frame of hand. In this way, the GUI would be able to follow the user everywhere and can be easily accessed. The two GUIs are synchronized. User uses the virtual laser pointer on the controller to

interact with objects and GUIs. Here the proxy selection pattern is used because therefore the user can select with distance and at the same time with high accuracy.

Figure 37 User can toggle on/off the on-controller GUI and use the laser pointer to interact with it

In either of the two GUIs user can start to choose the way of visualization of the data. The user can choose data source, visualization type (on a plain map or on a globe), the dimensions indicating the geolocation information of the data, the way to aggregate the data and what dimension to be aggregated, and extra dimension to include in as detail information of the data. Notice that user can choose different dimensions and different aggregation methods for two dimensions of the data visualization: the data bar height and the data bar color. On the GUI there are two buttons, one is for submitting request to the backend server for visualization, another is for reset and reload the user input.

Then the user utilizes the virtual laser pointer from the controller to confirm the input and request data from server by pressing the confirm button using the trigger on the controller. And then the data will be visualized accordingly as shown in the figure below:

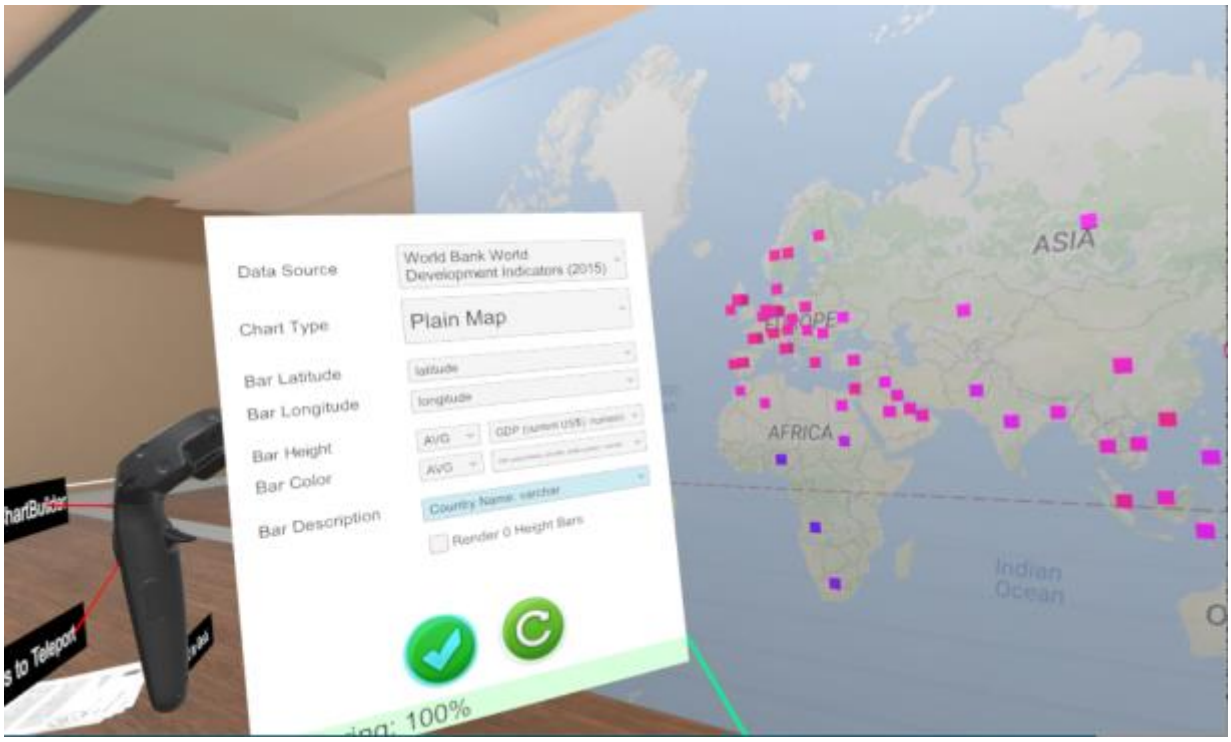


Figure 38 After user selects what and how the data to be visualized and presses the confirm button, the data is visualized

User can also change dimensions of data to be visualized as shown below:

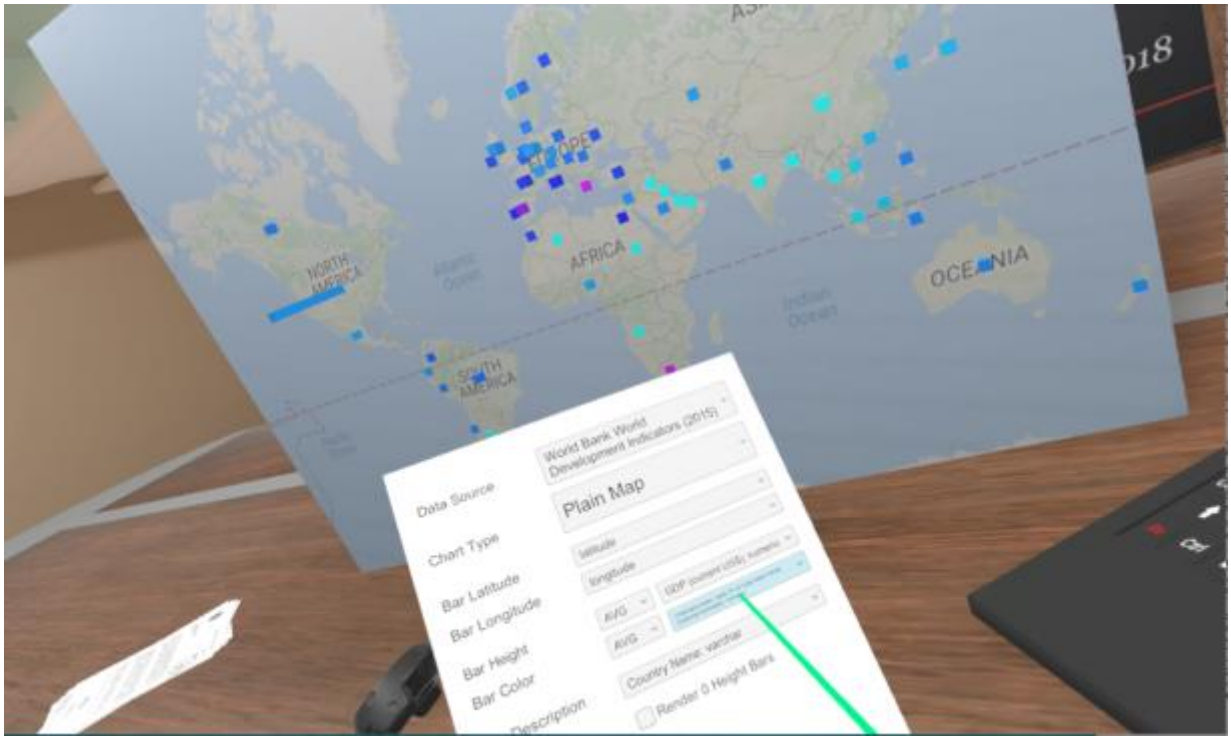


Figure 39 User can change what dimension to be visualized and refresh the data

The type of the map can be changed to provide a different perspective. User can also use the controller laser pointer to manipulate the map so that the data bars can be seen from different angels. The effects are shown in the figures below:

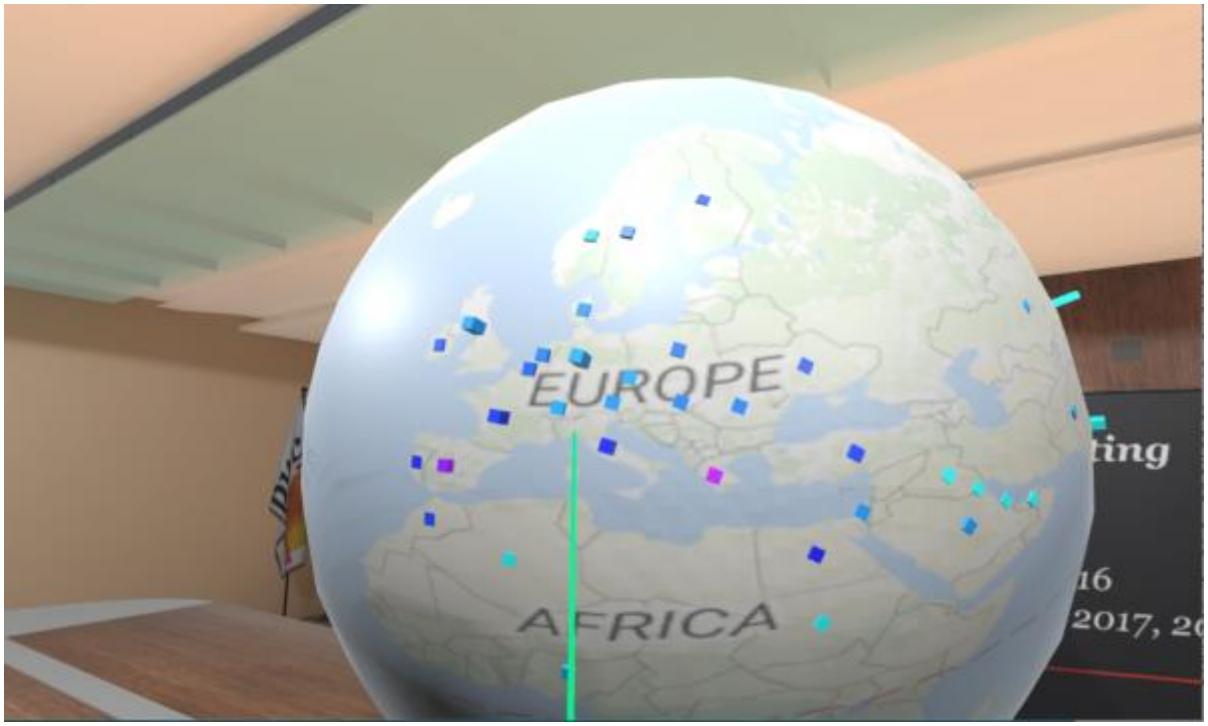


Figure 40 User can also change the type of the map object on which the data points is placed

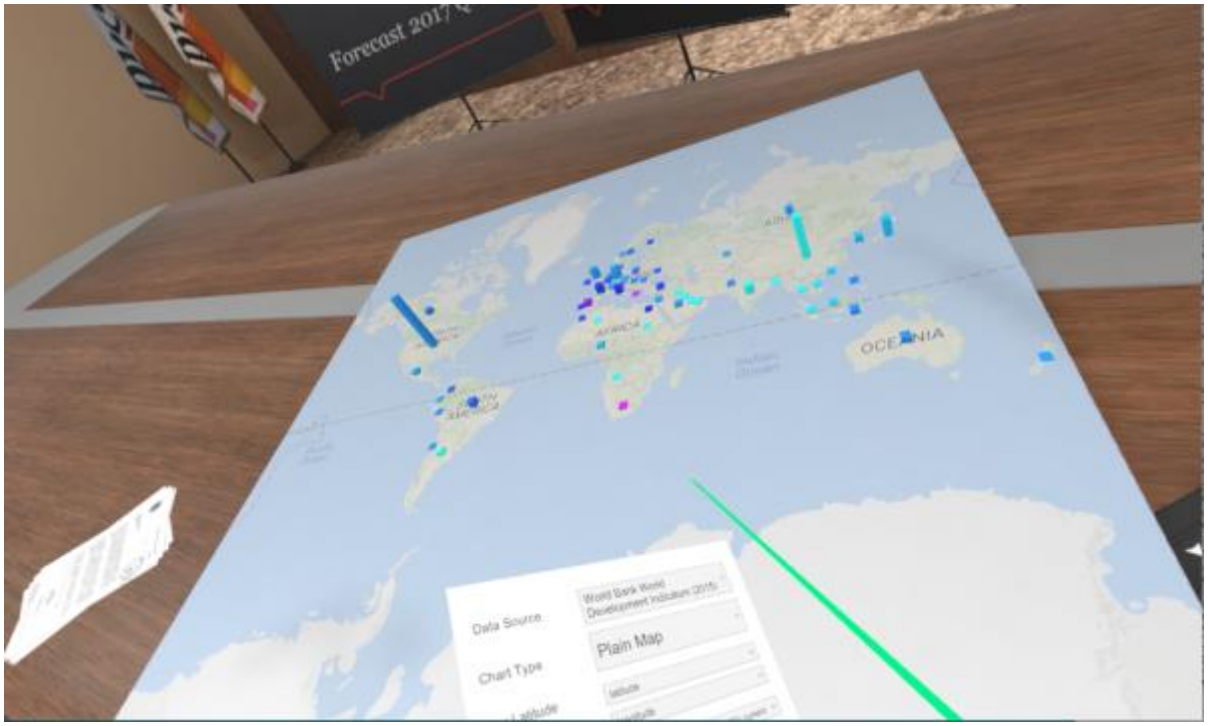


Figure 41 User can use the controller to manipulate the perspective of the map object so that the data points can be observed from another angle

The color code here is not designed to be traffic light style in order to preserve neutrality of data because the data source or contents are dynamic. It is better just to show the hue difference to represent the high and low values as the figure below:

Figure 42 The legend that indicates two dimensions of the data points: height and color hue

Notice how the multi-dimensional visualization helps with understanding the data more efficiently. For example, in the figures above they show the GDP and Unemployment rate at the same time from the "World Bank World Development Indicators" dataset. The height of the bar shows the low and high of the GDP meanwhile the hue of the data bar shows the Unemployment rate. An extra dimension brings a more efficient and effective understanding of data. Since it is in VR, it means the data disclose more details in a special interactive way. For example, when the user utilizes the laser pointer to point to select one of the data bars on the map or globe, detail information will be toggled on (Figure 43), both on the data bar and on the controller:

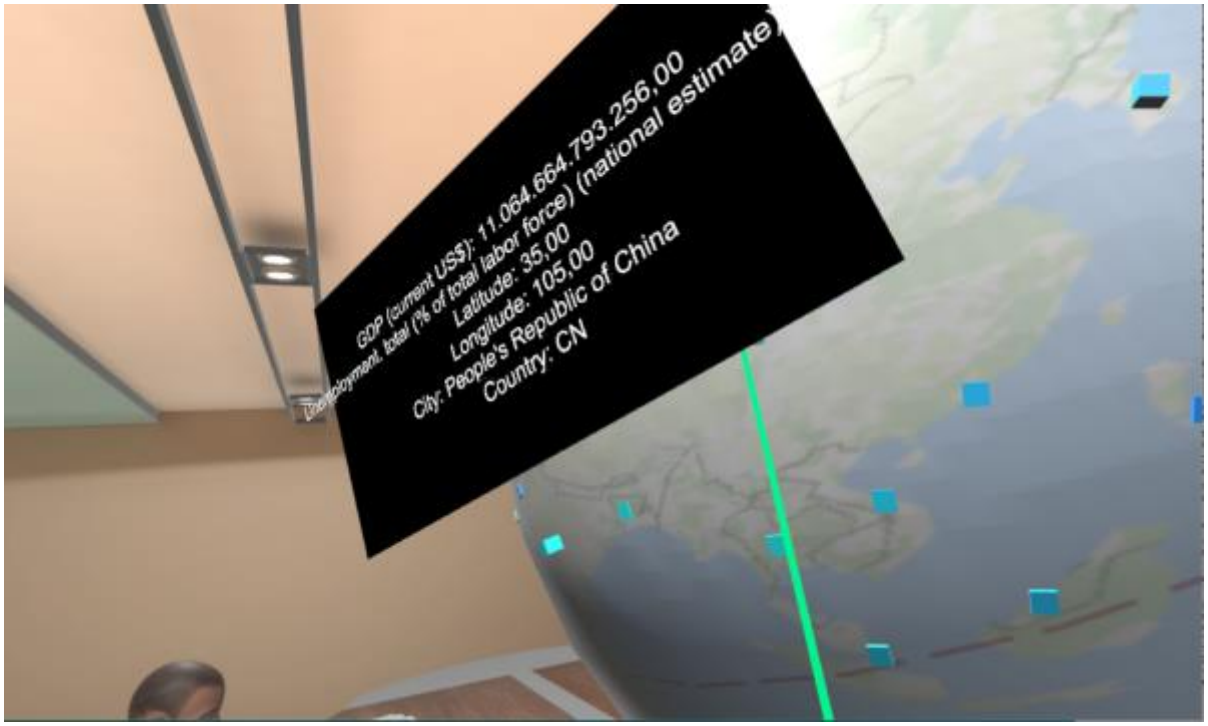


Figure 43 When user points at the data point using the laser pointer, the detail information will be toggled on

Figure 44 The detail information of the data point is also shown on the controller so that the user can see it easily from the hand

As shown from the Figure 44, if the user is standing from a further distance, he or she can still have this information shown at the controller. Again, the detail information panel utilized the reference frame of hand so that the information will be easily picked up by the user even though he or she may be standing far away from the data itself as in the figure above.

Figure 45 Teleportation and HMD tracking allow user to change the viewpoint easily

Another thing to be mentioned here is the viewpoint control pattern. In section 2.3.2 “Viewpoint Control Pattern”, the different methods of controlling the viewpoint were discussed and here in the application the teleportation technique is implemented. Because it is intuitive, easy to learn and also causes less motion sickness. On the other hand, since the HMD is precisely tracked, the movement of head and change of the torso position can naturally change the viewpoint, that is one of the benefit of using the high-end VR platform. So, no extra techniques are used for viewpoint control except teleporting.

By now, the application visualized the data on the maps and provides extra information through information panel on the controller. By changing the dimensions that are to be visualized and the data aggregation methods, can the power user build Ad-Hoc multidimensional data visualizations. In the next iteration — the showcase version of the application, more focuses on VR interactions and communications are implemented.

3.4.2 Version Showcase

The showcase version of the application is designed for entry-level users, they do not have to have the knowledge of data dimensions and aggregation methods, they just want to visualize data fast and retrieve information that is hard to perceive in 2D diagrams. And since they are entry level users, they may need more modal assistance so that they can have more emotional bonding and thus obtain more impressive immersive experience. With this in mind, the scene starts like the figure shown below:

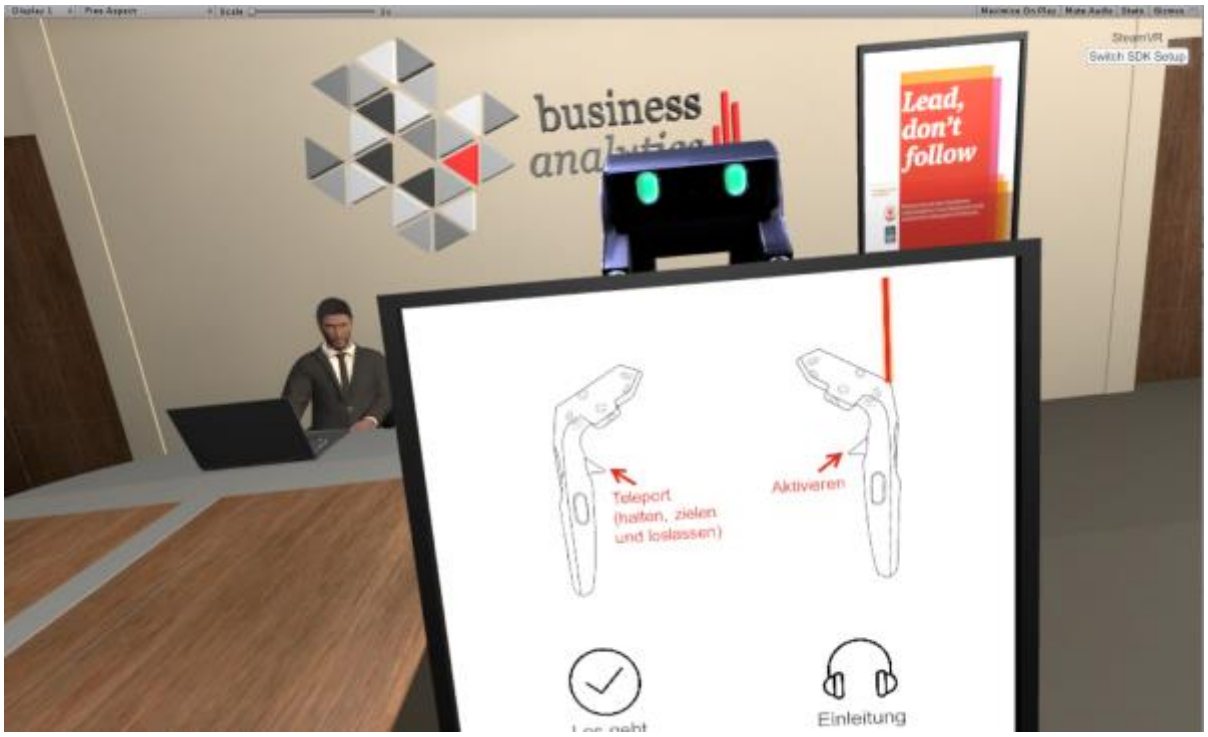


Figure 46 A start scene with a robot drone

It is still in the same office scene as the power user version, but this time there is a robot drone flying in near the user and holding the GUI which shows a launch guidance page. The drone is positioned in the peripheral area as described in “Action Areas” section, so that it will not block

the user's working area but still accessible. Notice that the drone follows the position of HMD, or it stays in the reference frame of torso, so that the user can always access the GUI. Starting with the scene, an audio guide will be played explaining what the application is about and how it can be interacted. And an introduction of operations is shown here as the first guidance. In this version of application, the operations are simplified as well. User just need to use two triggers to finish most of the tasks. The laser pointer is always on, compared to the power user version, in which the user has to press the touch pad to toggle the laser pointer. The grab buttons on the side of the controllers are set to be a hidden feature to stimulate the curiosity of the user so there is no hints on them. The user can toggle on and off the audio guide freely by pointing and clicking on the headphone icon. When the user is ready, he or she can use the laser pointer to click on the check icon to switch to the data visualization panel. One small thing to mention here is the flying height of the robot drone, it is set to be a little bit lower than the eyesight of the user, in this way the user will feel less stressed and welcoming. The emotional bonding is built up.

Figure 47 A simplified version of GUI

In this version, the GUI is also redesigned as seen above. Since the functionalities are simplified, the aggregations are left out. The user just need to choose what two dimensions of which data need to be visualized in what form. The three buttons are also redesigned to assemble a clear, simple and friendly GUI. After the user confirms the input and click on the check icon, the data will be visualized the same way as in the power user version. The difference is, there will be only one detail information panel shown on the map object, and it will always face the user by referring to the head reference frame. In this way, the detail information is more readable. Another add-on for this version is, when the data is plotted on the map object, a speech will be played stating what kind of data is being visualized and what data dimensions the bar height and bar color represent. As discussed in the section 2.1.2 "Content Design", audio helps providing extra context and emphasize the information.

Another point to be discussed in this version is the static visualization. When user uses the laser pointer to trigger any of the flip charts, a 3D bar chart will pop up and sits on the desk in the scene. The source data of the charts come from the PricewaterhouseCooper's report "Digitale Abschlussprüfung – Realität, Erwartungen und Trends"[103]. By visualizing in 3D in VR, user can perceive information in a more effective and efficient way. Consider the following comparison in Figure 48 and Figure 49:

[103] Jacob, "The Use of Eye Movements in Human-Computer Interaction Techniques: What You Look at Is What You Get."

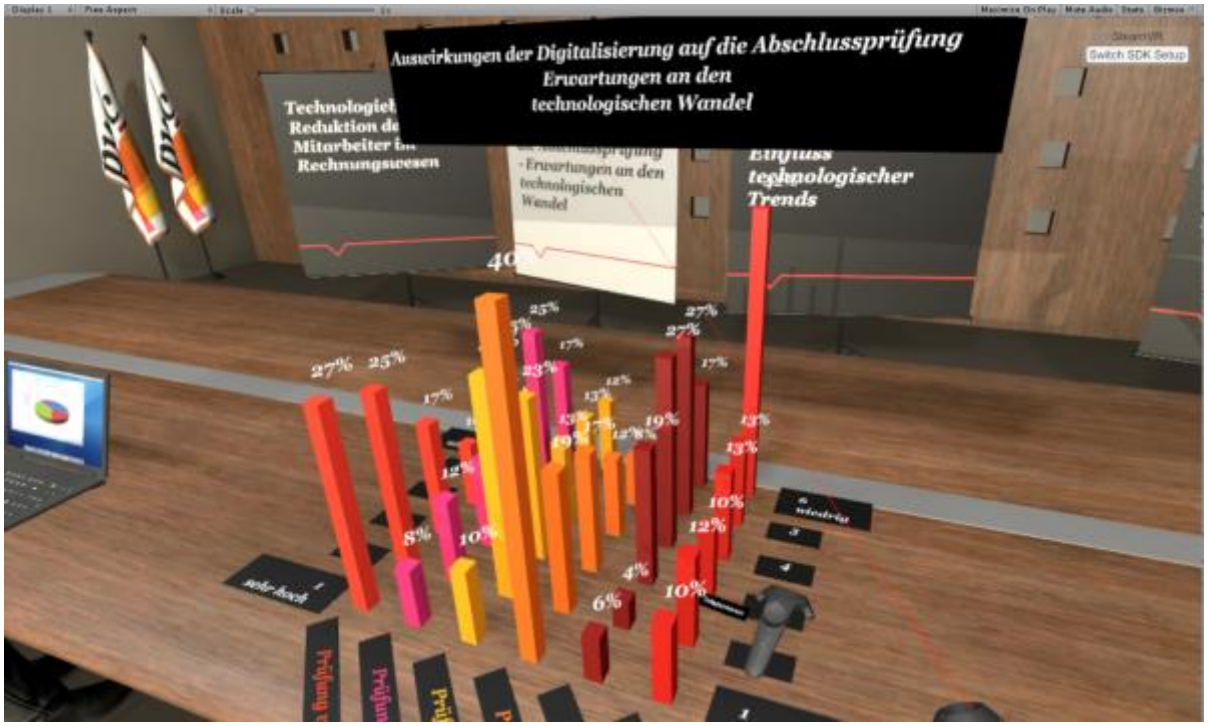


Figure 48 A 3D bar chart shows a survey result which has multiple dimensions

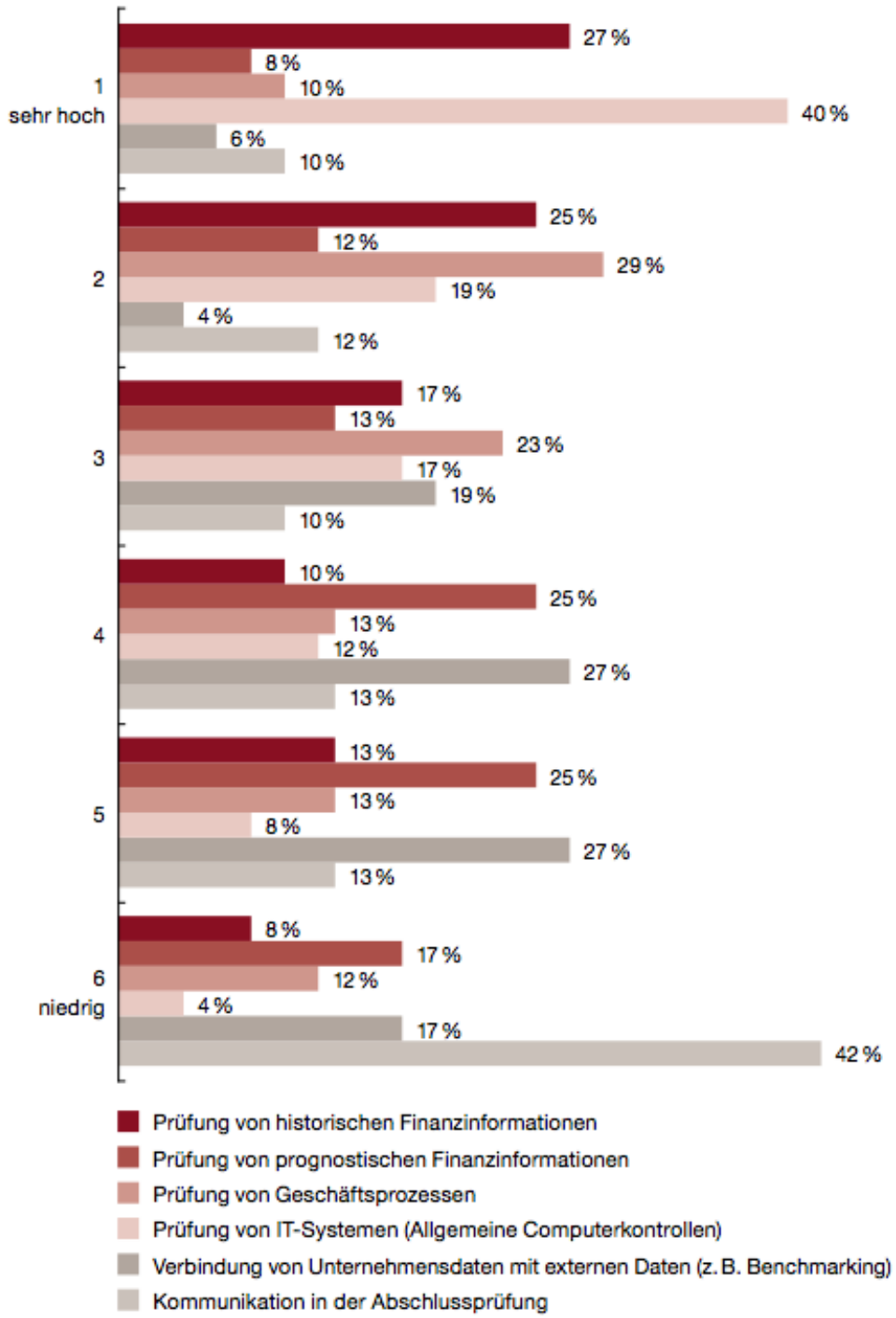


Figure 49 A 2D version of the same set of data from the survey: "Erwartungen durch technologischen Wandel"

From the 2D version of the visualization, it is not easy to see the distributions of the expectation percentages of one kind of technological evolution because when the reader scan through the report vertically trying to find the same color, it is prone to lose track of the number and the categories, it is even harder to perceive pattern under a category. But when the user observes the 3D chart in VR, things are different. User can easily identify the distribution of technological changes in each level of expectations as shown in the figure below.

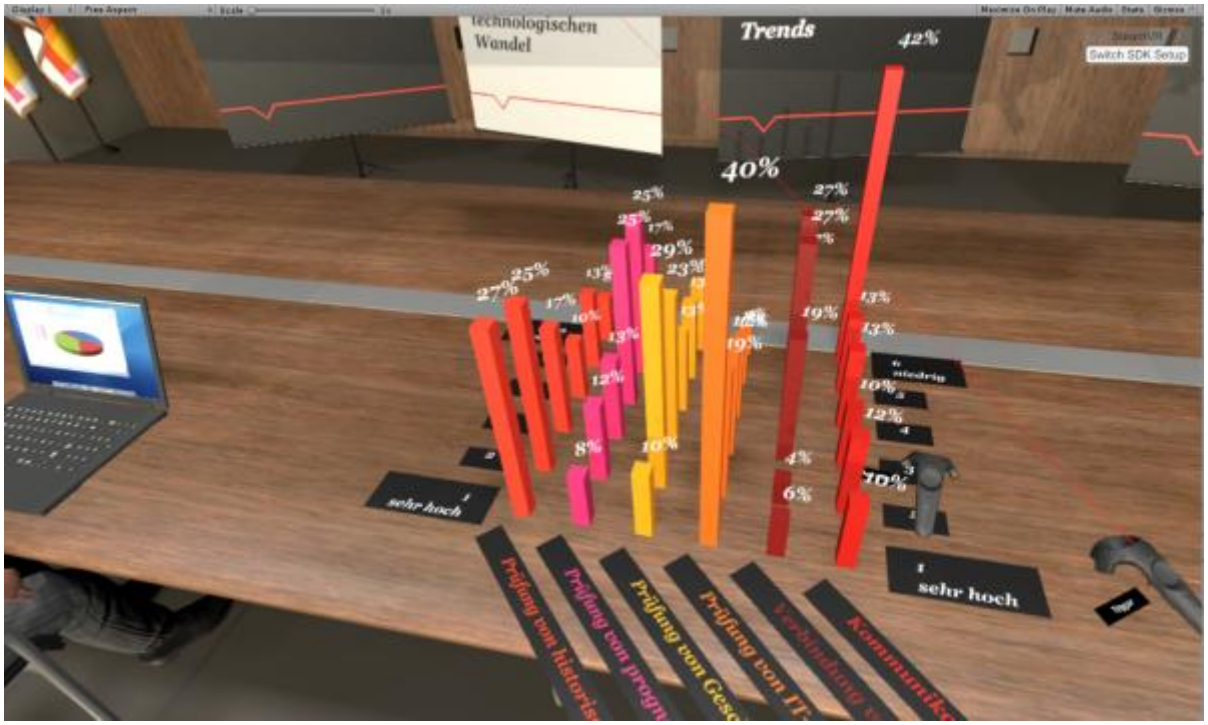


Figure 50 User can easily identify the category dimension and its outstanding values

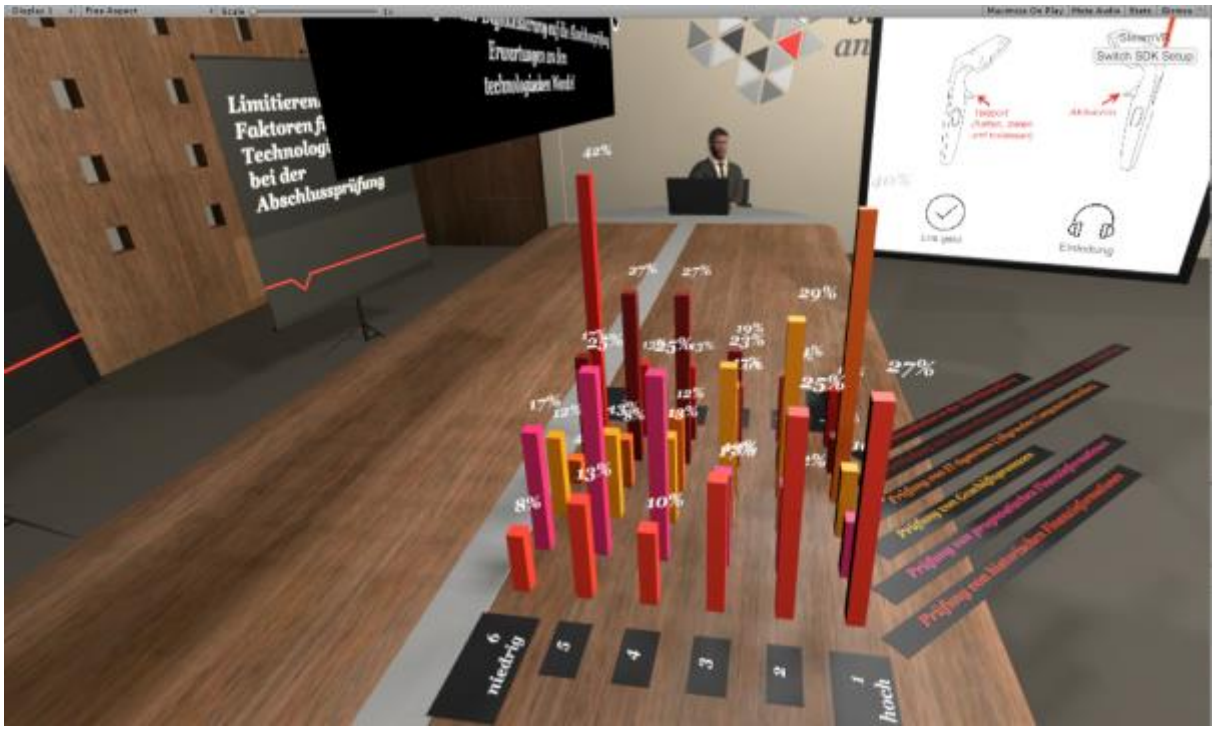


Figure 51 User can easily identify distribution of value dimension by changing the perspective or the point of view

Since it is in VR, user can freely and intuitively move the viewpoint to gain a better view for pattern recognizing. For example, as shown in the Figure 51, when the viewpoint is set as below, it is easier to identify, the orange technological change “Prüfung von IT-Systemen” has the highest expectation compared to others and most of the expectations on this issue are high. The rightmost technological change (“Kommunikation in der Abschlussprüfung”) however is the opposite. The patterns of other distributions of expectations are also noticeably comparable than the plain 2D chart. It will be more helpful when the number of dimensions becomes larger.

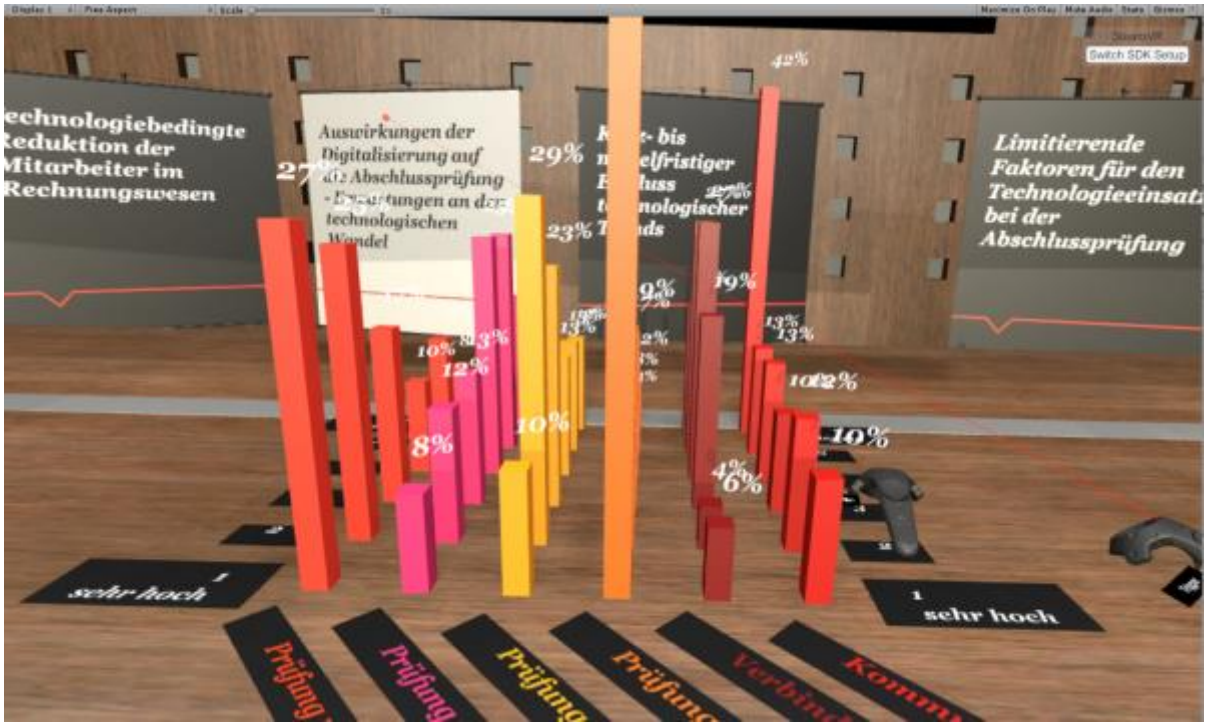


Figure 52 User can easily identify distribution patterns by moving the point of view intuitively

Although these visualizations are static in the scene because they are not loading any backend data, they are built with dynamic scripts. Which means, it is possible that these static models are also to be built dynamically by taking in user's inputs.

3.5 Prototype Architecture

In the previous sections, the hardware solution and development environment were described. In this section, the architecture of the prototype and some key classes or methods will be introduced to explain how the application runs under the hood. First of all, let us take a look of the application from the user's point of view, then the underlying logics and structures will be revealed.

3.5.1 Application Architecture

This application consists of mainly two parts. One is the "frontend" that present all the visual and provide the virtual world where the user is placed and he or she can interact with virtual objects. Another is the "backend" that feeds the data into the virtual world on request. On the backend, the data is stored in persistency. In another word, on the highest level, the application is a client-server architecture where VR visualization is the client and the server will provide the data to be visualized.

3.5.1.1 Client

The following diagram shows the overview of the client side. Basically, the design of the architecture follows a Model-View-Controller (MVC) pattern. The virtual world or the scene in Unity is considered to be the view, and there is a controller that is responsible to process the requests sent from the scene and mappings of the data into the scene. Finally, there is a model part that deals with the raw data received from the backend server and wrap them into data objects so that it can be further processed by controller.

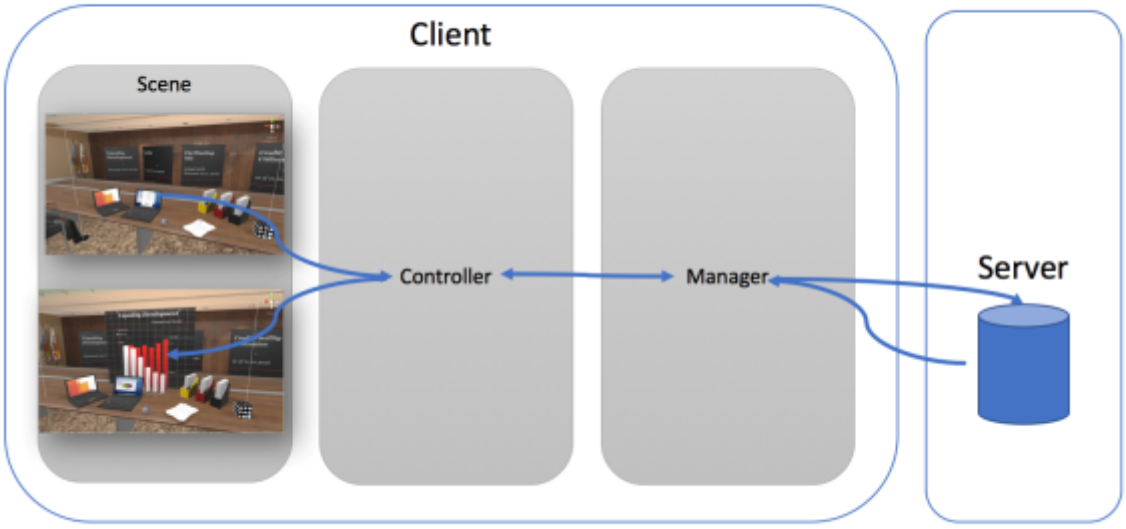


Figure 53 A client-server architecture with MVC design pattern

To understand how each part of the client works, it is necessary to dive deeper into Unity to understand how it drives the whole application.

Unity Basics

Basically, Unity uses a component-based architecture. With this architecture, the components will have its own behavior independently while keep the possibility to interact with other components. In Unity, the base component is GameObject. Everything in Unity is a GameObject and it is a container. GameObject could be attached with many other children components, together they will make each GameObject an independent object and have its own behavior, properties, and interactions with other objects. A GameObject always has a Transform component attached, this component defines the position, rotation and scale of the GameObject. For other kinds of components, it really depends on what kind of GameObject it is defined to be: a light will have a Light component attached which dictates properties of a light such as intensity, color, range and

so on (Figure 54[104]); a solid cube object has a Mesh Filter and Mesh Renderer component which draws the surface of the cube, and a Box Collider component that represent the object's volume in terms of physics so that it will reacts to physical effects (Figure 55[105]).

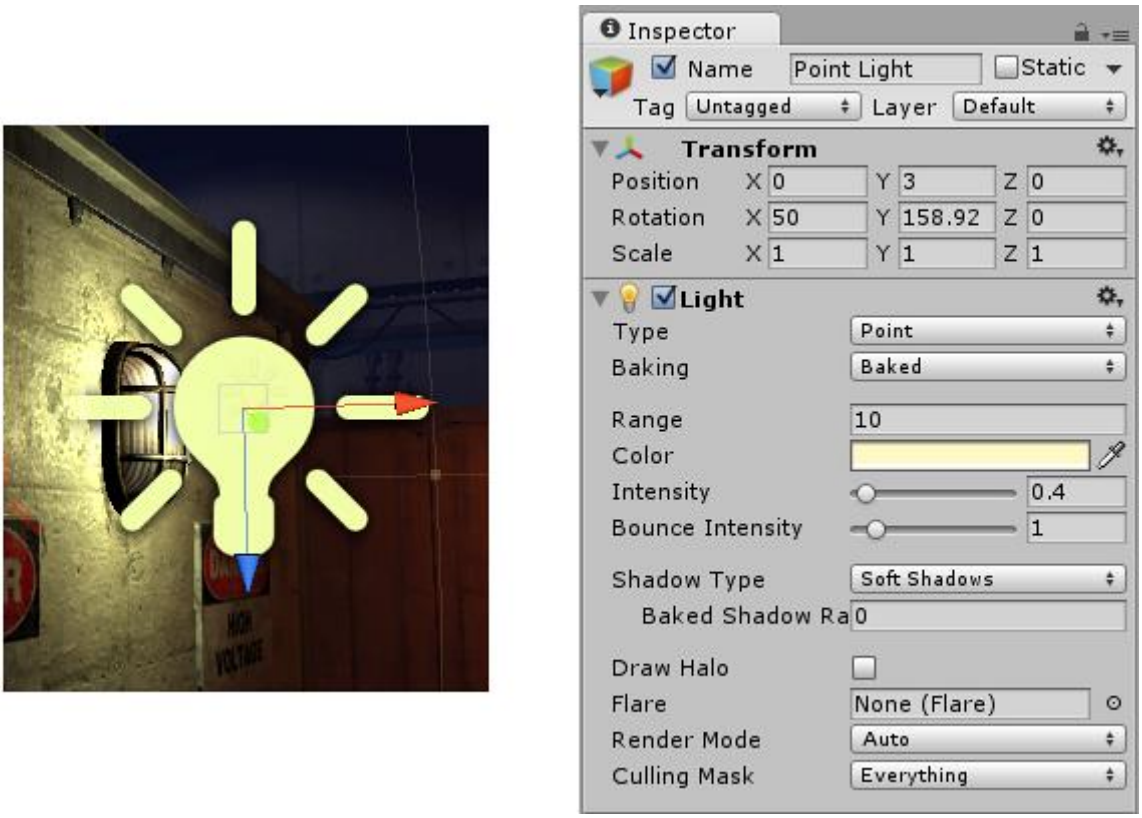


Figure 54 A GameObject as light

[104] Unity, "GameObjectLightExample.Png (600×404)."
[105] Unity, "GameObjectCubeExample.Png (582×457)."

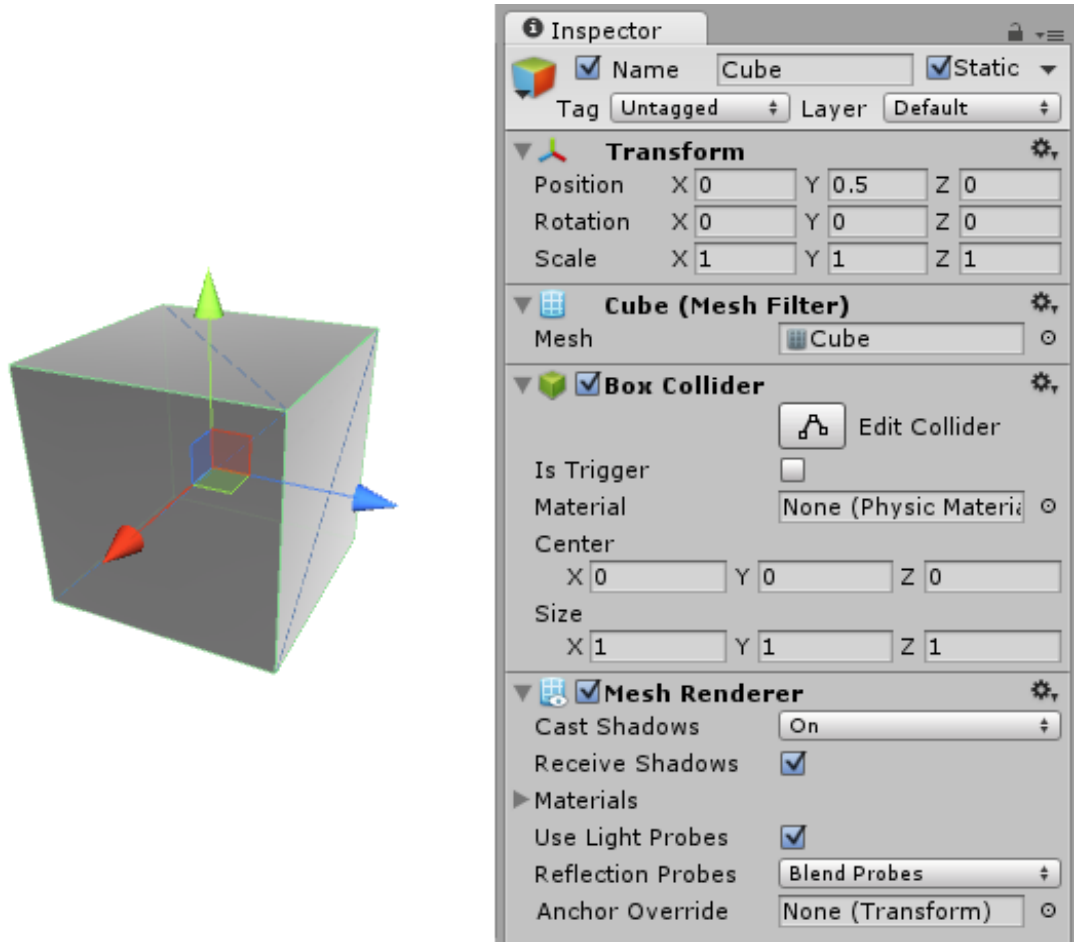


Figure 55 A GameObject as a normal cube

GameObjects and its components define the objects in Unity. Behavior scripts can also be attached to the GameObject as a component. When the GameObject to be utilized, both the GameObject and the attached components can be referenced and manipulated by code. As mentioned earlier, both C# and Javascript are supported as scripting languages in Unity. So how are the objects referenced and how their behaviors are directed? Since the current application is developed using C#, the further discussions of codes are in the C#.

Whenever a new script is created in Unity, a class inherits from MonoBehaviour is created. MonoBehaviour is the base class from which every Unity scripts derives and it provides a series of event functions that direct what the class should do in various circumstances. Some events are executed in order, some events happen during the interaction with other objects. But where does this MonoBehaviour class come from? When traced back, it inherits from Behaviour class, then from Component class, and finally from Object class. Therefore, Behaviours are also one type of component in Unity's component based architecture. As for GameObject class, it inherits also from Object, therefore, it is clear that from Object class inherit GameObject and MonoBehaviour these two most important classes in Unity's application. With GameObject class, all the objects in Unity can be access, and with MonoBehaviour class, all kinds of behaviors of objects can be defined. Now the game engine can assemble all the resources it has and direct them as a choreographer. To better understand the structure, let us take a look at an example shown in Figure 56.

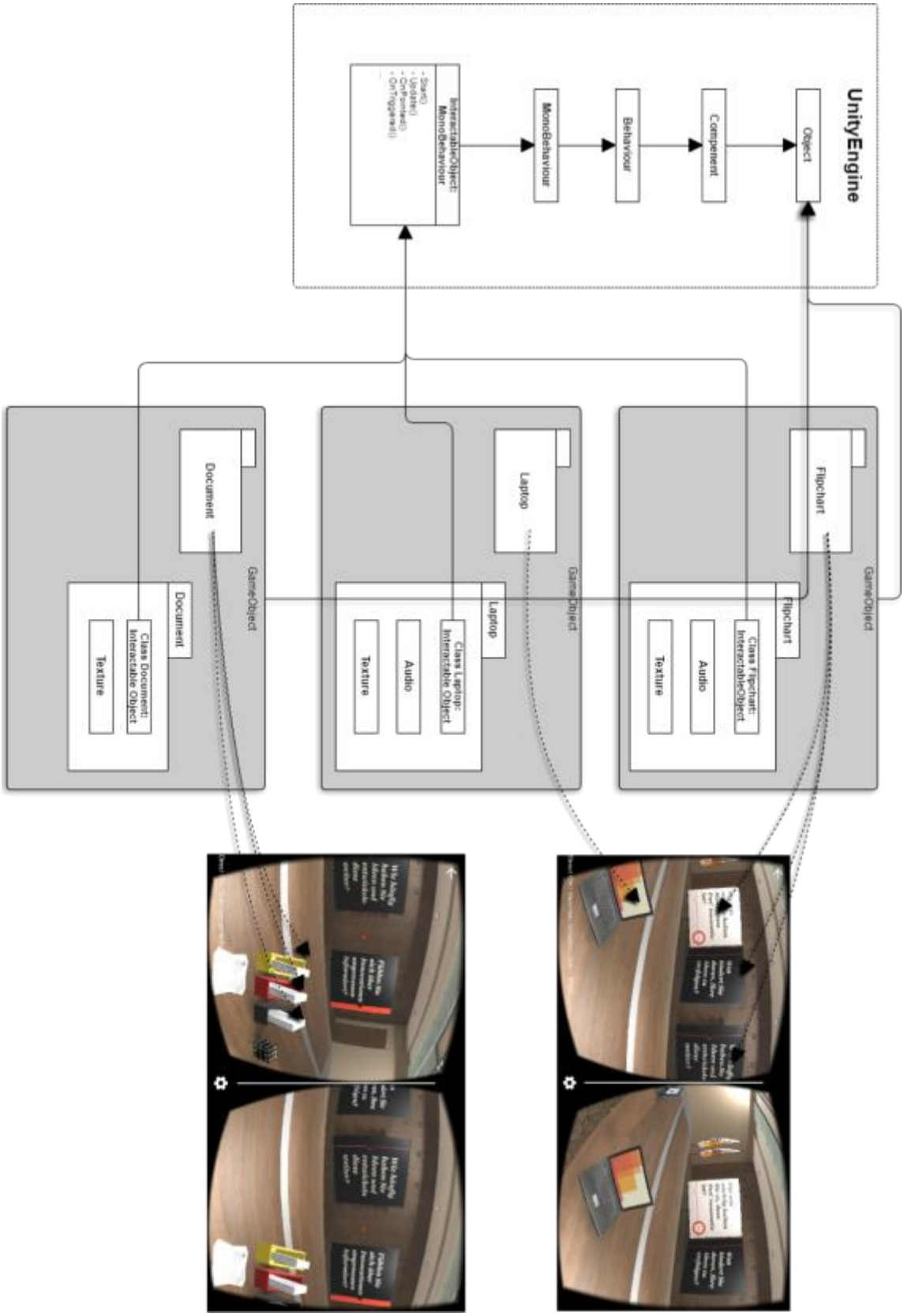


Figure 56 The structure of Unity's classes

On the right side of the figure, the stereoscopic view of the virtual world is the scene at runtime created by Unity. There are a couple of objects: some flip charts, a laptop, some document folders and some other stuffs. Since everything in Unity is GameObject and they have components attached, the structure is shown at the middle part of the diagram. Now each of the object will react to some certain interactions, which is realized with the script attached to it. Each object has its own class but those classes all inherit from a customized class called InteractableObject which, by nature in Unity, inherits from MonoBehaviour class. This relationship is shown on the leftmost side of the diagram. In the InteractableObject class, some event functions are defined, such as Start() and Update(). These two functions are standard MonoBehaviour event functions and dictates the behaviors during the lifecycle of the object which the script is attached to. Like other standard event functions, it is called by Unity. Notice that there are other customized event methods in the InteractableObject class, those methods are to be called by other objects or method. In this way, a customized GameObject and its behaviors are defined.

In the scene, objects are created and they can operate on their own behaves. In order to utilize the VRTK library mentioned before, those objects that are to be interacted with need to include script components that implements VRTK library, then during the runtime all the scripts components that are attached to the object will affect the behaviors, including the VRTK behaviors. From the figure below for example, a laptop in the scene is attached with two script components, one is LapTopLogic class which inherits from VRTK_InteractableObject provided by VRTK library, and it includes also some other properties that are not part of the base class. Another script component

is a direct integration of VRTK_FixedJointGrabAttach class, which will enable grab interaction using the HTC Vive controller in the virtual world.

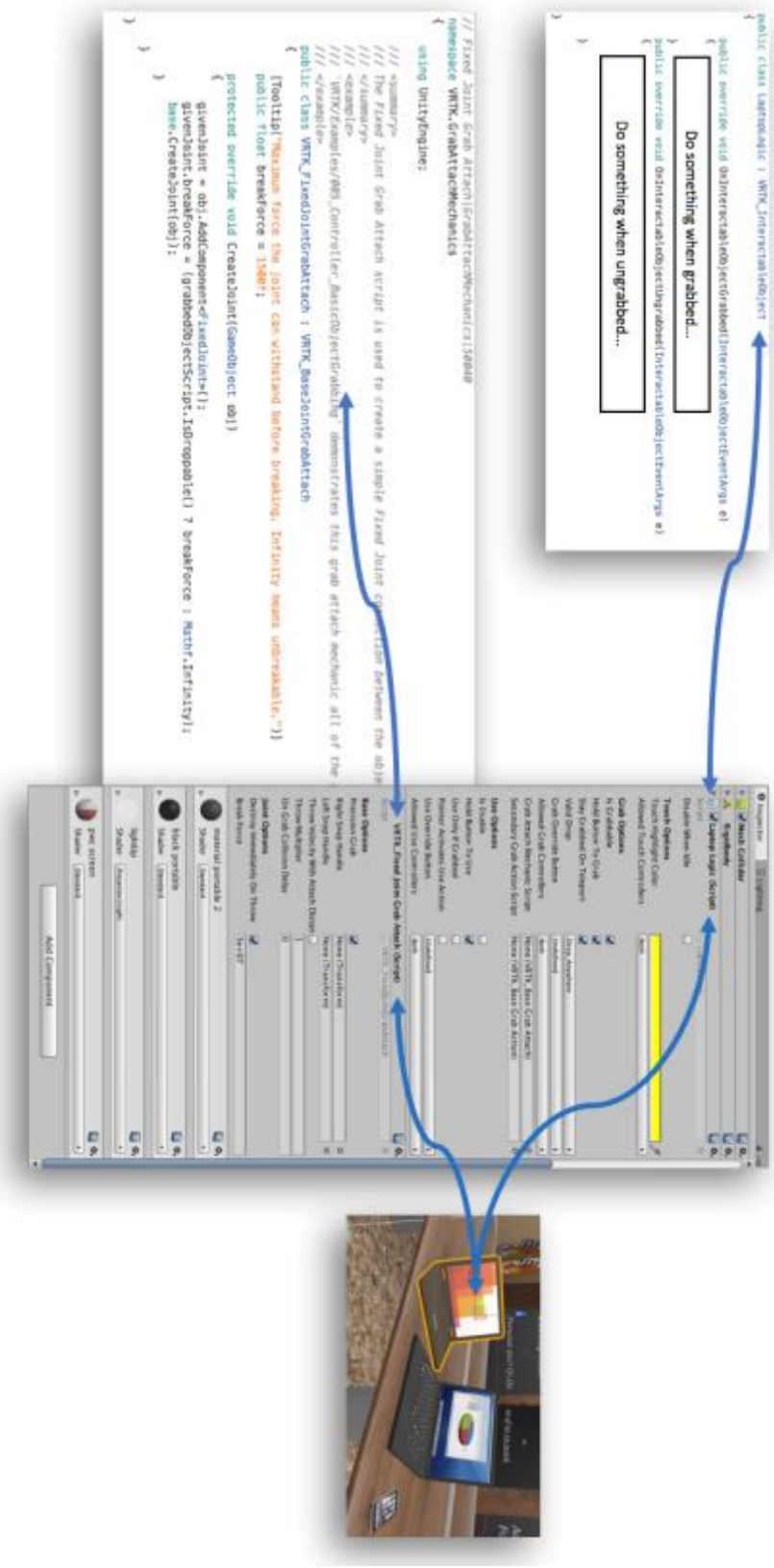


Figure 57 A GameObject that has its own behavior script and integrates with the VRTK behaviors

With the understanding of how GameObjects and scripts work in Unity, it is ready to discover the next part of client logics: Controller.

Controller

Through attached script components, the objects in the scene can behave on themselves and react to user's actions. Then the next step is to request the backend and consume the data, and finally visualize them in the scene. This responsibility is taken by the controller. In this prototype, as mentioned before, the form of the visualization is geographical data on a plain map or on a globe according to user's choice. As seen from the application walkthrough, the user will send request through a 2D user interface, and the data will be mapped into the virtual world. So, the controller will take in the user inputs from the GUI, pack them up as parameters, then the controller will send those parameters to the manager. The manager will send a request to the server and receive the data objects from it before the controller consumes the data objects and get ready for visualization. Finally, as mentioned in the data visualization pipeline in section 2.2 "3D Data Visualization", those data objects will be unwrapped, the geographical coordinates will be remapped into the 3D virtual world coordinates, and the data values that are to be visualized will be transformed and presented to the user.

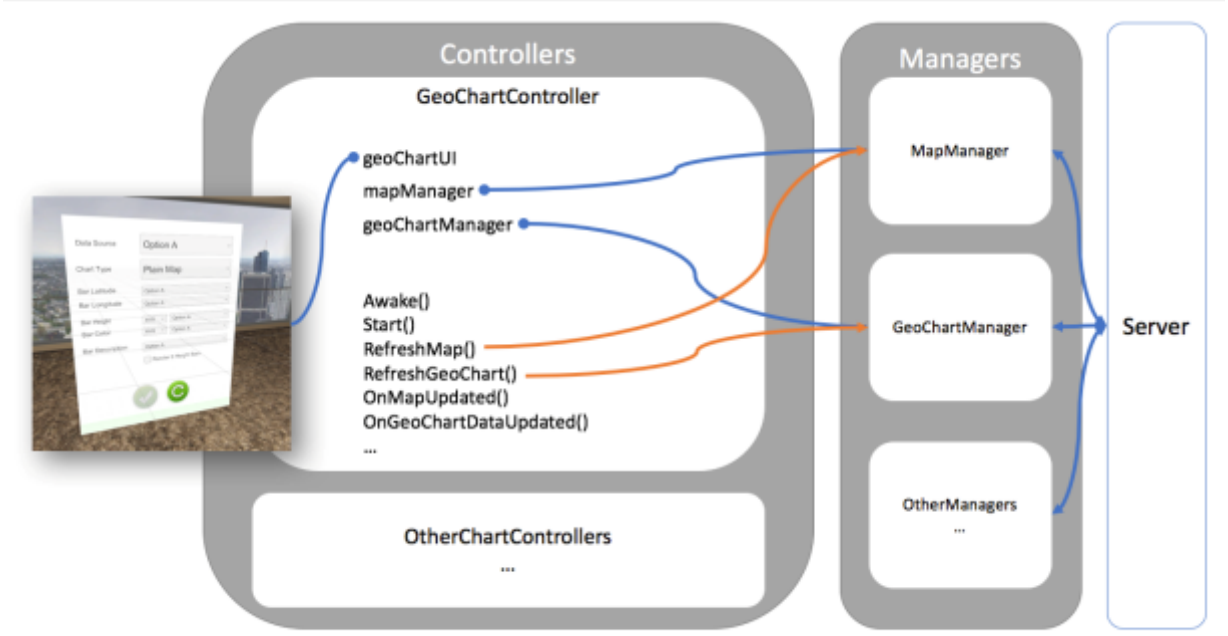


Figure 58 Overview of the operations between Controller and Manager

As seen from the Figure 58, this application is designed to be able to visualize various types of charts, therefore there are GeoChartController and OtherChartControllers (as a placeholder in the diagram). Same applies to GeoChartManager and OtherManagers due to the consideration of further extensions on the server requests. In this section GeoChartController will be the focus, the Managers will be discussed in the next section. In order to communicate with both the user and the backend, there are references from both the GeoChartUI, the MapManager and the GeoChartManager in the GeoChartController. As the application starts, Unity will call the standard MonoBehaviour methods Awake() and Start() in sequence. They are automatically called by Unity since the GeoChartController is placed in the scene as a GameObject. In these two methods variables are initialized. In the Awake() method, there are a series of AddListener() methods from Messenger class (Figure 59). It is a utility class that broadcast or receive event messages in a global scope and react to the event messages accordingly.

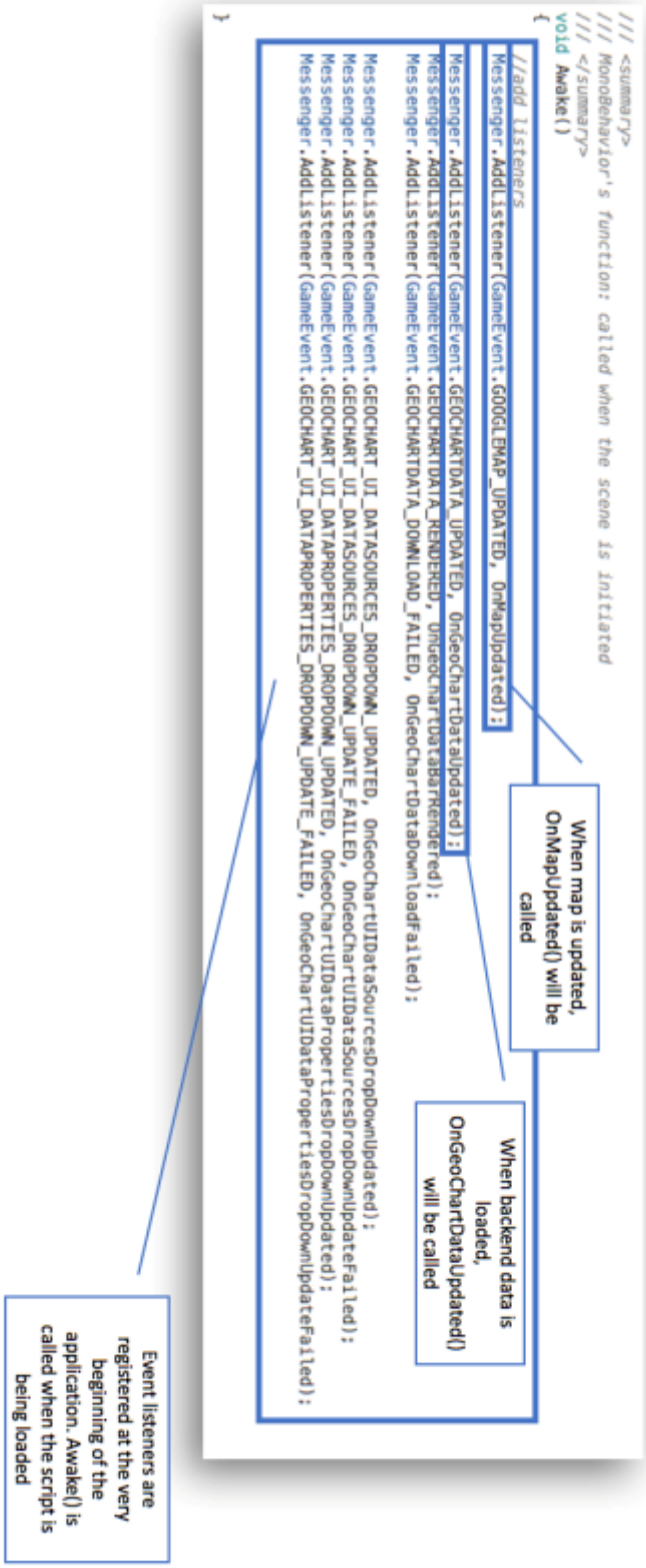


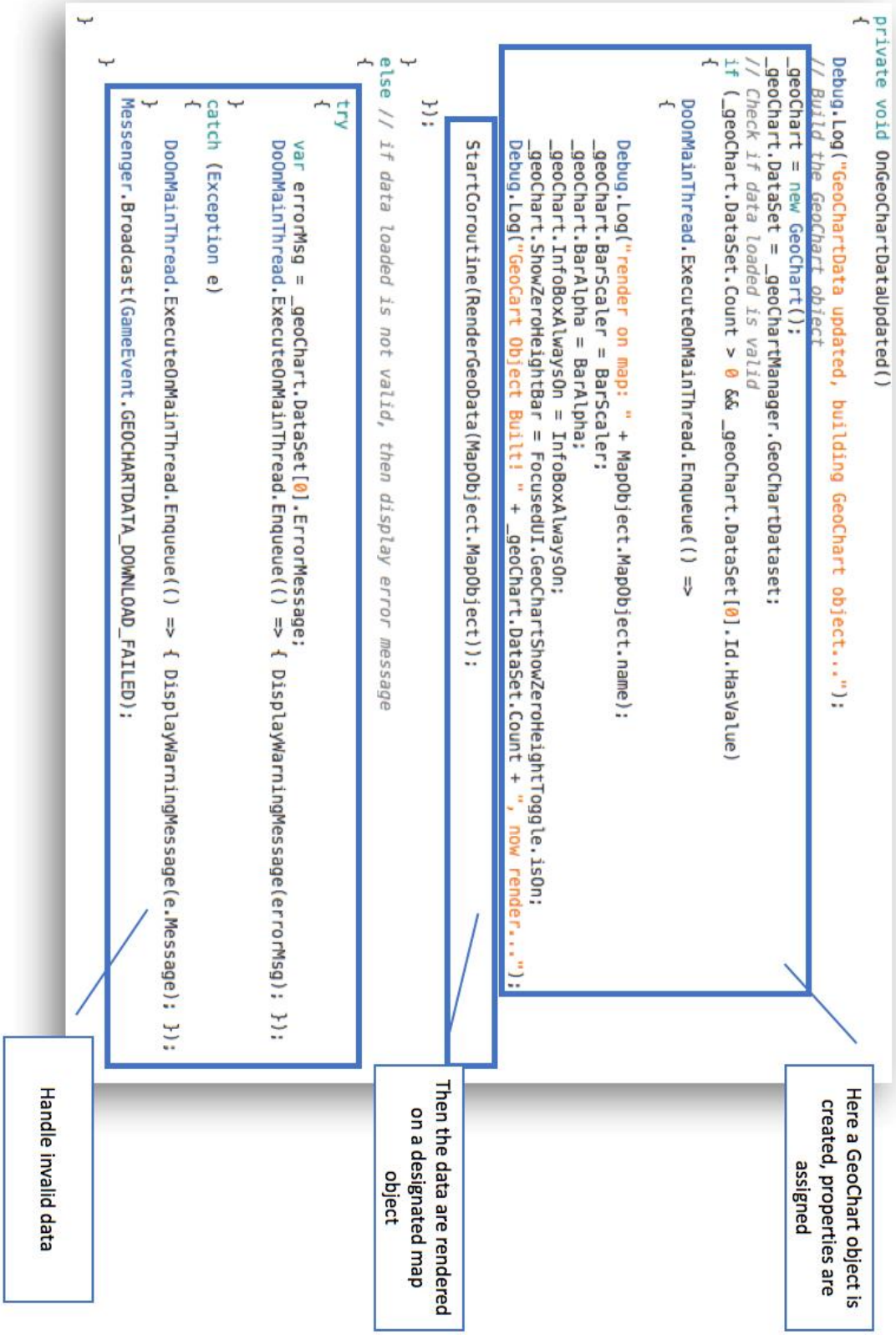
Figure 59 Register listeners in Awake()

Figure 60 Initiate variables and Manager references in Start()

Then the controller will utilize the MapManager to refresh the map using the RefreshMap() method so that the world map texture will be downloaded and saved as a texture. Once the map texture is downloaded and cached, the controller will then call OnMapUpdated() method, in which it will initialize the MapObjectFactory. The MapObjectFactory will produce the MapObject later depending on user's choices of map type: whether it is a plain map or a globe (Figure 61).

Figure 61 MapObjectFactory will create a map object after the map is downloaded

The user then chooses what type of map to use and what data to be visualized through GUI, and click on the submit button to request for data. Once the data is returned, the OnGeoChartDataUpdated() method will be called and the data objects will be unwrapped and transformed, and finally mapped and visualized in the virtual world with a designated form: either on a plain map or on a globe. To make the understanding easier, the Figure 62 shows only an essential logic of the controller, some other methods are not included. There are other event methods which will dictates the interaction events triggered from the UI, for example the refresh button and the reset button. Other than these events, there are also visualization methods that will transform, map and render the data points in RenderGeoChartDataOnMapObject() method. Let us take a look on the OnGeoChartDataUpdated() method.




```
private void OnGeoChartDataUpdated()
{
    Debug.Log("GeoChartData updated, building GeoChart object...");
    // Build the GeoChart object
    _geoChart = new GeoChart();
    _geoChart.DataSet = _geoChartManager.GeoChartDataset;
    // Check if data loaded is valid
    if (_geoChart.DataSet.Count > 0 && _geoChart.DataSet[0].Id.HasValue)
    {
        DoOnMainThread.ExecuteOnMainThread.Enqueue(() =>
        {
            Debug.Log("render on map: " + MapObject.MapObject.name);
            _geoChart.BarScaler = BarScaler;
            _geoChart.BarAlpha = BarAlpha;
            _geoChart.InfoBoxAlwaysOn = InfoBoxAlwaysOn;
            _geoChart.ShowZeroHeightBar = FocusedUI.GeoChartShowZeroHeightToggle.isOn;
            Debug.Log("GeoCart Object Built! " + _geoChart.DataSet.Count + ", now render...");
            StartCoroutine(RenderGeoData(MapObject.MapObject));
        });
    }
    else // if data loaded is not valid, then display error message
    {
        try
        {
            var errorMsg = _geoChart.DataSet[0].ErrorMessage;
            DoOnMainThread.ExecuteOnMainThread.Enqueue(() => { DisplayWarningMessage(errorMsg); });
        }
        catch (Exception e)
        {
            DoOnMainThread.ExecuteOnMainThread.Enqueue(() => { DisplayWarningMessage(e.Message); });
        }
        Messenger.Broadcast(GameEvent.GEOCHARTDATA_DOWNLOAD_FAILED);
    }
}
```

Figure 62 OnGeoChartDataUpdated() called after data are returned from server

As mentioned, when the data is retrieved from the server, OnGeoChartDataUpdate() will be called as a registered event, then it will create a GeoChart object which contains a property called dataset and it would be assigned by the data received from the server, then it will start to render the data. It is worth mentioning that here some optimizations are applied as seen from the code through multi-thread operation and co-routine method: Multi-thread is used because of the latency of downloading data from server. In order to avoid the downloading thread blocking the frame rendering, multi-thread method is utilized. Here the multi-thread operation is not included in the standard Unity framework so it is used from a customized class. There is another method called StartCoroutine() which allows Unity to render the next frame without waiting, otherwise the current frame will be blocked if the downloading (or other action) takes too much time, therefore it also appears to be another thread, although it is just a co-routine in the same thread in CPU.

After all the operations mentioned above, the RenderGeoData() method will take in the map object that is created after the map was refreshed and call the render function RenderGeoChartDataOnMapObject() in GeoChart object as in the following figure:

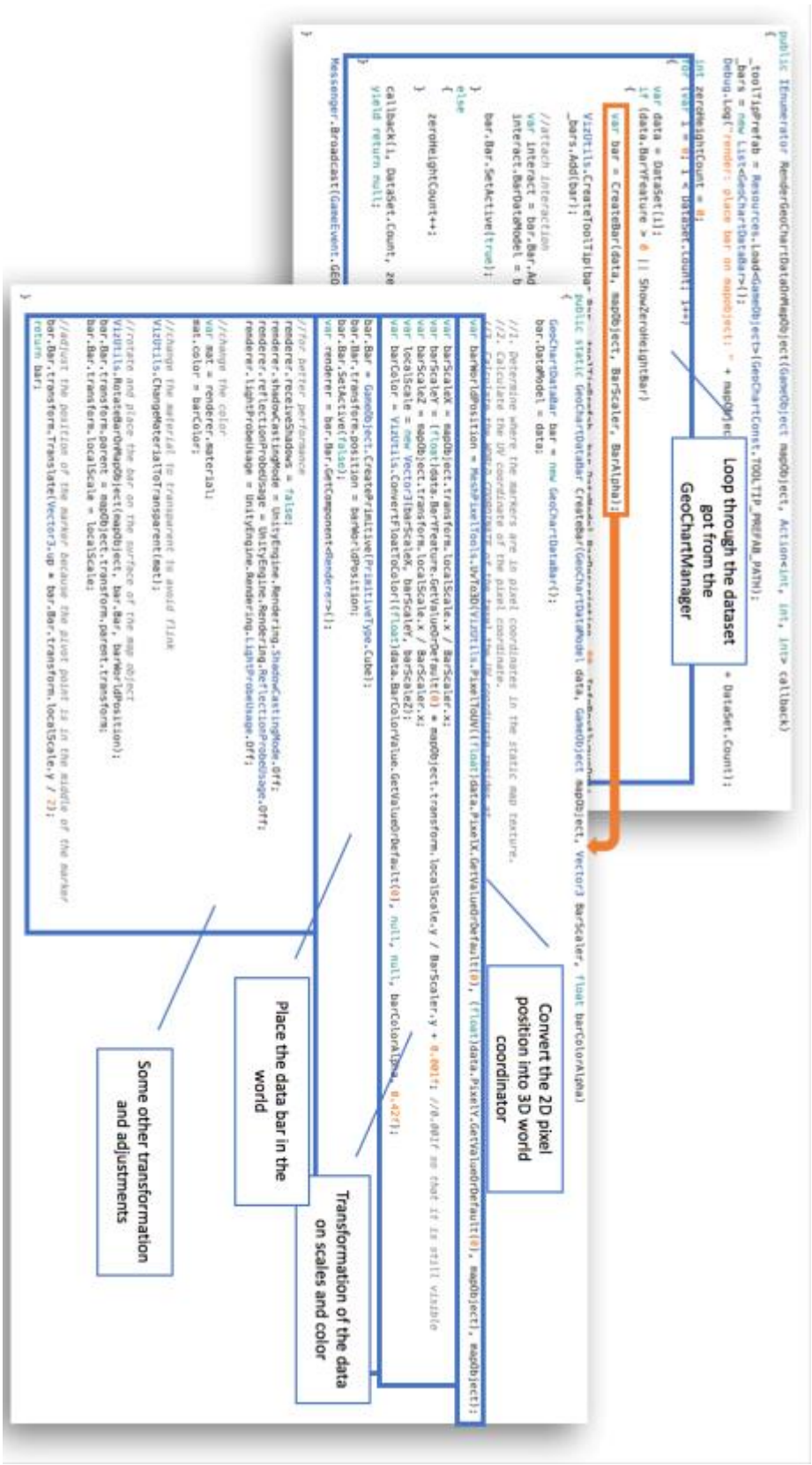


Figure 63 RenderGeoChartDataOnMapObject() will finally visualize the data in the scene

In this RenderGeoChartDataOnMapObject() method, it will loop through the dataset (which has been assigned to be the data received from the server by GeoChartController), and then go through the data visualization pipeline: mapping, transformation and plotting. The data object received from server already contains 2D pixel positions that are converted from geographical latitudes and longitudes by the server for the sake of performance, now the 2D pixel positions need to be converted into 3D world positions on the map object accordingly. Then scaling is applied and finally, other adjustments on textures, colors and rotations are implemented. After the data bar is created and set on the map, some other components are added on them to allow further interactions: tooltips that contains detailed descriptions and interaction script. At the end of the rendering loop, the RenderGeoChartDataOnMapObject() method will broadcast a message in global scope, other objects that are set to pick up the message will then react to it.

To summary, this is the most essential duty of the controller. It reacts to the user inputs and pull the data by utilizing the Managers, then unpack the data received from Managers and map the data into the virtual world to complete the pipeline of visualization. Let us see how the Managers communicate with the server.

Manager

Managers work closely with backend server. In the main scene of the prototype, there are three related classes: Managers, GoogleMapManager and GeoChartManager. Managers is the manager of the managers, it controls how the managers work and initiate them. The other two Manager classes are extended from the interface IGameManager which has only one Status variable and a Startup() method. The Startup method will take in a parameter of class NetworkService. Here

service injection is used in order to keep the manager class clean and maintainable. When the application runs, the Managers script is initiated, the Awake() method is called, then the managers will call the Startup() method in each of the manager so that they are prepared with NetworkService. NetworkService is the class that does the real communications with the server. The Figure 64 describes this process in a high level.

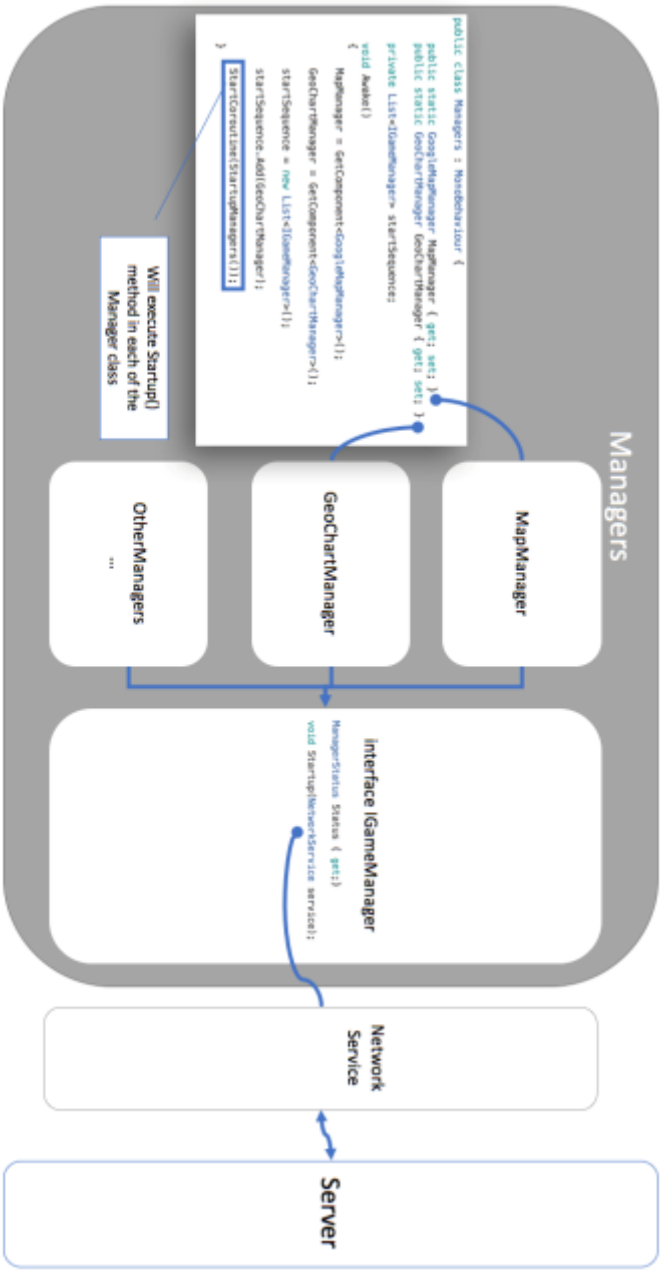


Figure 64 Managers are responsible of loading data from network by utilizing NetworkService

In the NetworkService class there are variables that defines the URL addresses, for example MapAPIUrl and GeoDataAPIUrl. There are also methods that will build the URLs with dynamic arguments which are passed from the Manager and originally from user inputs. Then those methods will call a base function CallAPI() which does the real job of sending request and receiving responds from server. One interesting parameter of the CallAPI() is the callback method. It is a method passed in as an argument of CallAPI() function, or in another word, it is a delegate function. The real action of this delegate is done by a declaration in manager. In this way the callback function will have access to all the variables in Manager and also called by the NetworkService class.

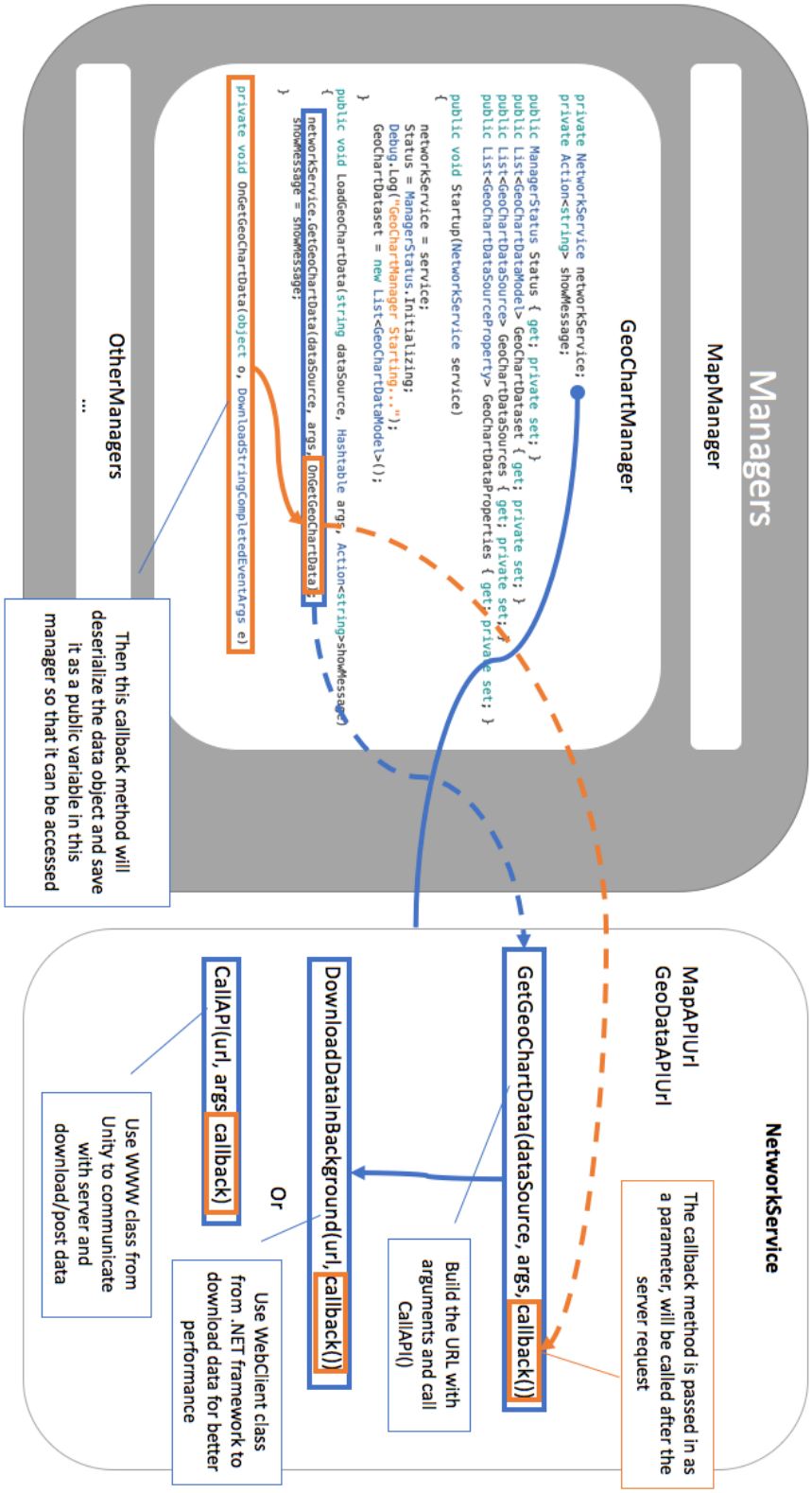


Figure 65 GeoChartManager utilizes NetworkService to load data that are to be visualized

For example, in GeoChartManager (Figure 65), it has variables that are to be accessed by GeoChartController and these variables hold the data downloaded from server. When the application launches, the Managers (manager of the managers) class will call the Startup() method in GeoChartManager to initialize the variables and instantiate the NetworkService class (injecting a NetworkService instance). At the same time, this GeoChartManager will also be referenced in the GeoChartController, as well as the GoogleMapManager. Then this GeoChartManager will be waiting for the calls from GeoChartController before it utilizes the NetwrokService instance to communicate with the server.

Suppose the user now triggers the submit button on the GUI to request for GeoChart data, firstly the GoogleMapManager will retrieve the map texture from the Internet and then holds it. Then GeoChartController build the map object by using MapObjectFactory, which will access the map texture from GoogleMapManager. After the map object is built, the controller will then call the GeoChartManager to load the data from the data server, which triggers LoadGeoChartData() in GeoChartManager. The following listings (Figure 66 and Figure 67) shows how the GeoChartController calls the managers to get the data needed.

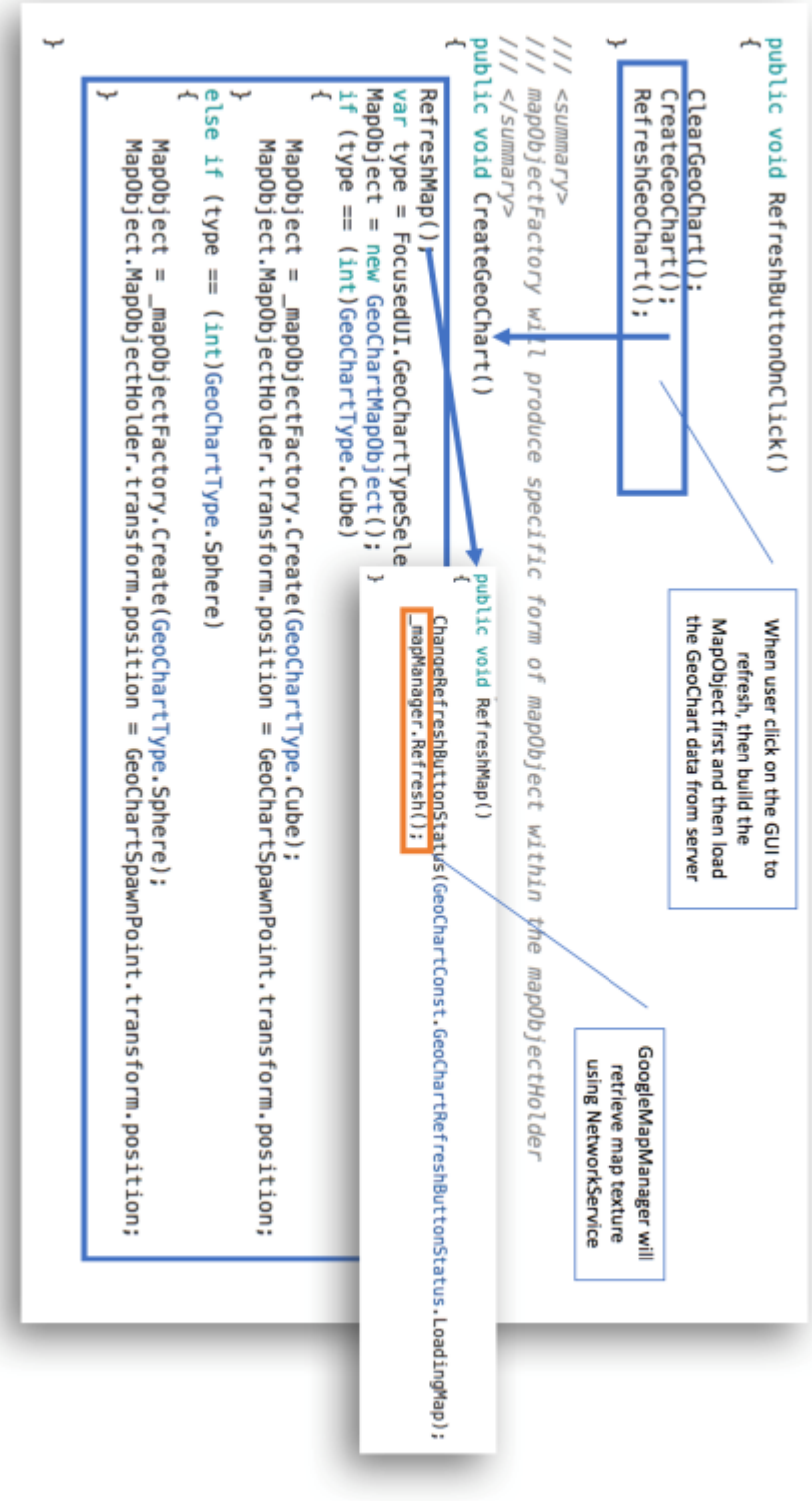


Figure 66 RefreshButtonOnClick() triggers map object creation and data loading

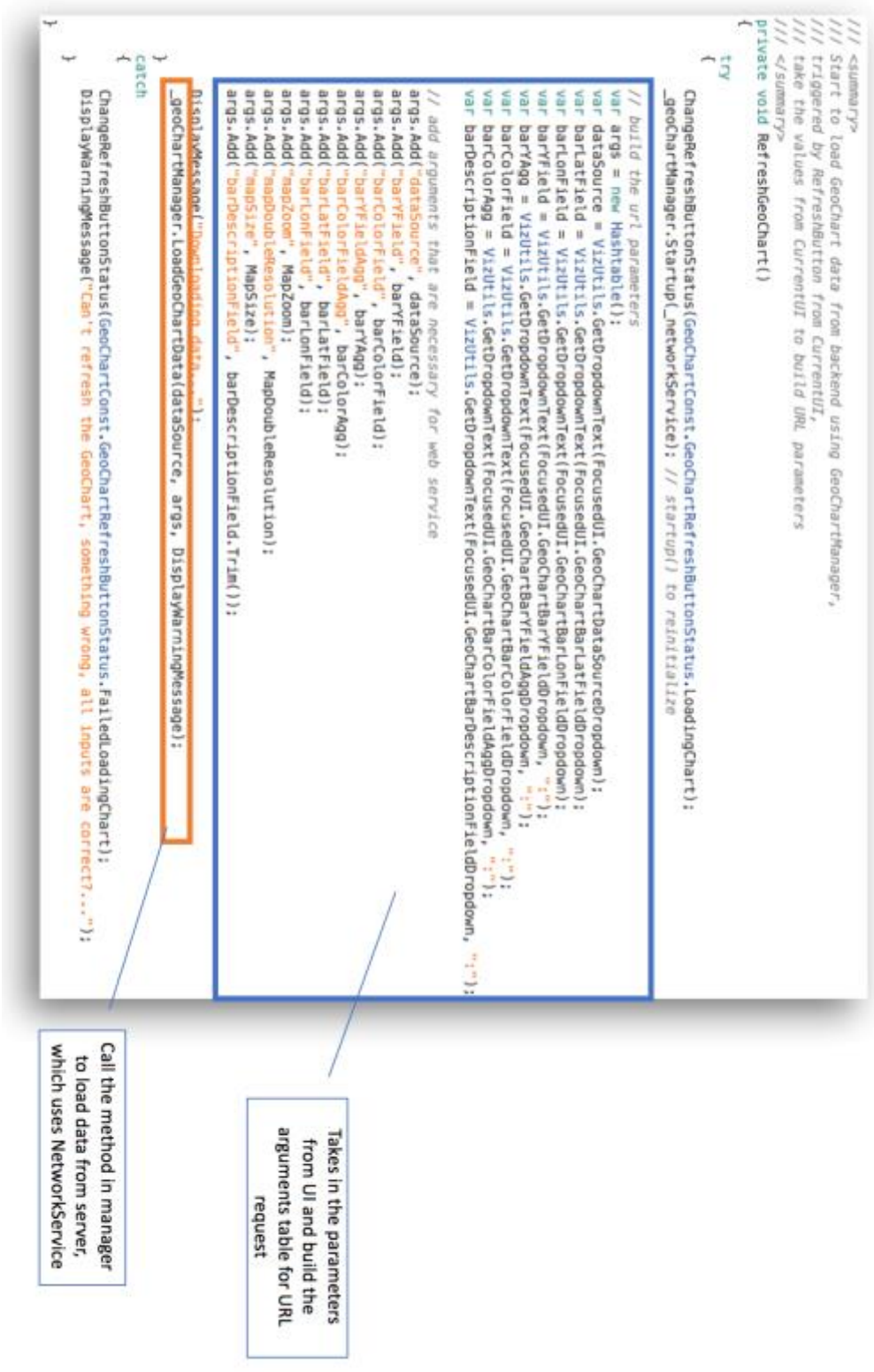


Figure 67 After the map object is created, RefreshGeoChart() is triggered to load the data from backend by passing parameters to the server

Then the GeoChartManager will call the GetGeoChartData() from within the NetworkService passing in a callback function delegate. Next, the NetworkService.GetGeoChartData() triggers the DownloadDataInBackground() method to download data from backend server[2]. Finally, the downloaded data will be stored back into GeoChartManager through callback function. The callback function does more than just storing the data, it will unwrap the data object, a JSON object in this case, then deserialize the data object into a data model. This data model, as shown in Figure 68, is a simple class that defines the properties of the data.

```
public class GeoChartDataModel {

    public string Id { get; set; }
    public decimal? BarYFeature { get; set; }
    public decimal? BarColorValue { get; set; }
    public decimal? BarColorAlpha { get; set; }

    public decimal? Latitude { get; set; }
    public decimal? Longitude { get; set; }
    public decimal? PixelX { get; set; }
    public decimal? PixelY { get; set; }

    public string BarDescription { get; set; }
    public string ErrorMessage { get; set; }


}
```

Figure 68 A data model that will form the JSON format data which received from server

Finally, the data that are needed for GeoChartController to visualize is ready to be accessed and sitting in the GeoChartManager, it is then controller's job to put them into the data visualization pipeline and finally let the user see how the data looks like in VR. At this point, the client side of the application has done its job of taking request from user and fetch the data from the backend. It is time to have a look at the server side to discover how it prepares the data for the client.

3.5.1.2 Server

Till now the architecture and the mechanism of the VR side or the client side has been introduced. Thanks to the flexibility brought by Unity Engine, the backend or the server side can be decoupled from the client side just by utilizing Unity's WWW class. The WWW class can load the external data by simply pointing to a URL. With this client-server architecture, the backend can be easily scaled, modified or migrated. In practical, this prototype has been successfully run on a cloud server, feeding the data that needs to be visualized through the Internet, back to the client.

Basically, the server is a classical .NET MVC web application only without a view. It has a controller that takes care of the HTTP requests sent from the client, and retrieve data from a SQL Server database. Finally, the controller builds the data model along as the business logic requires and return the data in JSON format. An overview is shown as the following:

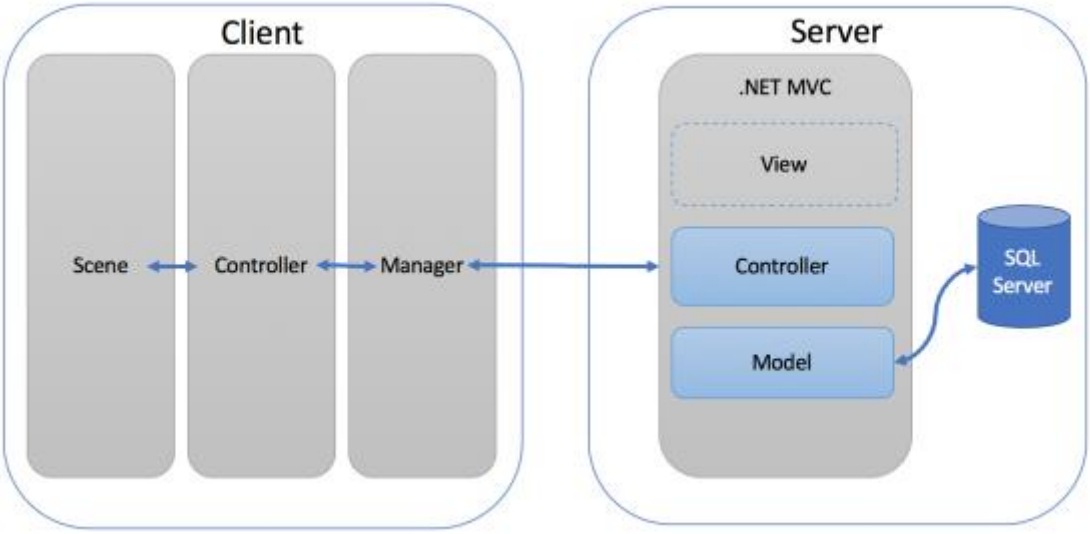


Figure 69 An overview of the server and its communication with client

Let us have a look at the server side layer by layer. Notice that the View layer is not implemented because it is not necessary.

Controller

There is only one controller for now in this prototype because only GeoChart type is implemented. Within this controller there are three actions to handle business logics. The first is the most important one called BuildChart(). This action will take the parameter from the client side and send to the database to trigger a stored procedure and generate a dynamic SQL query. After that, it will receive the query result from SQL Server and wrap it into the data model before it returns it back to the client side. The listing is shown as below in Figure 70:

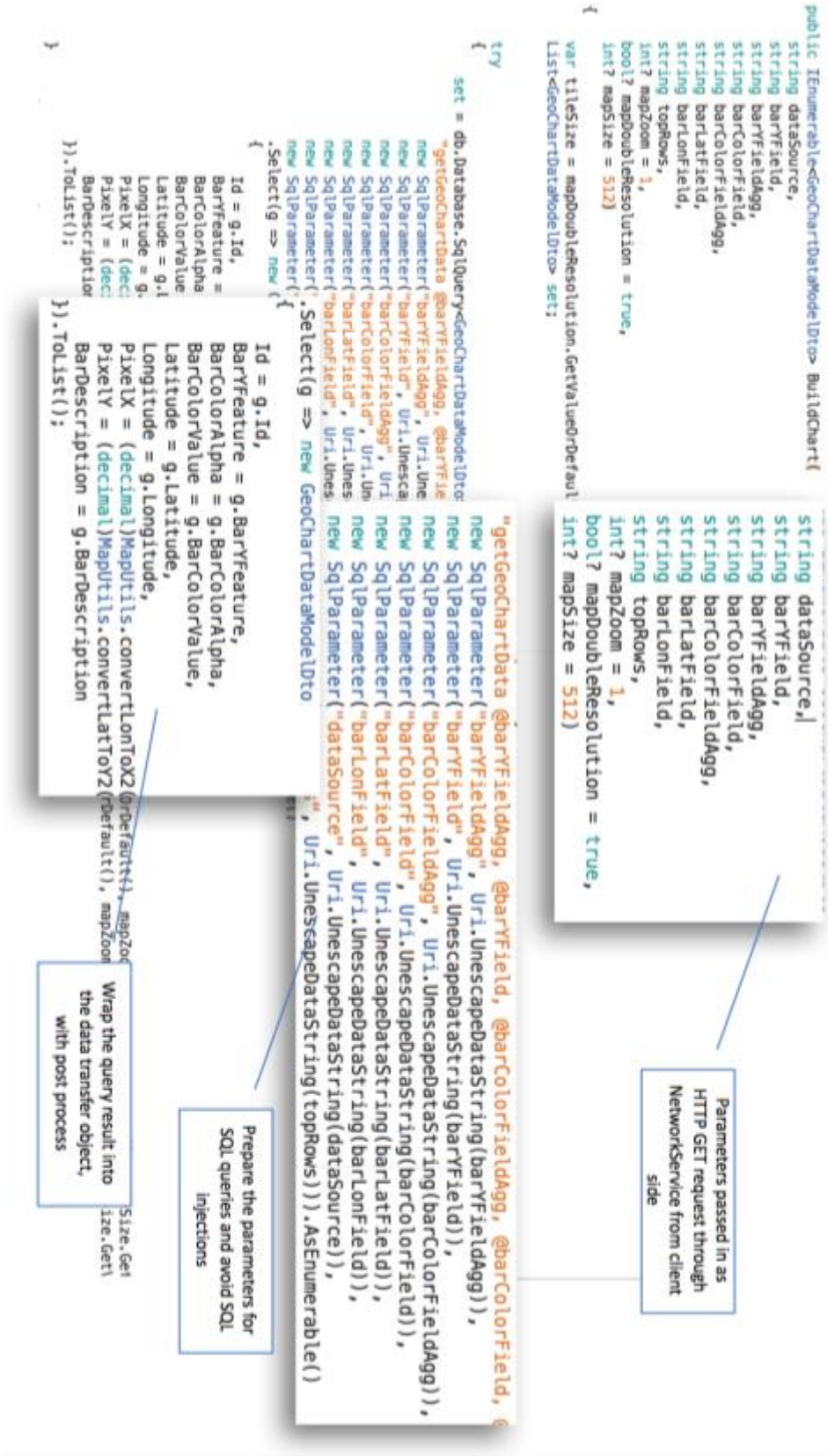


Figure 70 BuildChart() method takes in the parameters to generate dynamic SQL and forms the result as a data transfer object

One thing special here is the post process of the query result. Notice there are a MapUtils.convert() function. This is the helper function that converts the geolocation information: latitude and longitude into pixel coordinate, namely (X, Y). The conversion is done here because of the consideration of performance. In this way, the client side will have less computation, especially when the dataset is big.

The other two actions are relatively simple, they basically look for meta information of the database, therefore they take only one or no parameter for the query. The data model is also simple or is even a string list, so no complicated post processing is needed in this case. The listings are shown in the Figure 71:

```
public IEnumerable<DataSourceProperty> GetProperties(string dataSource)
{
    var query = @"select COLUMN_NAME as propertyName, DATA_TYPE as propertyTypeName
                from INFORMATION_SCHEMA.COLUMNS
                where table_name = @tableName order by COLUMN_NAME";
    var result = db.Database.SqlQuery<DataSourceProperty>(query, new SqlParameter("tableName", dataSource)).ToList();

    return result;
}
```

```
public IEnumerable<DataSource> GetDataSources()
{
    var tableNames = db.Database.SqlQuery<string>("select TABLE_NAME from INFORMATION_SCHEMA.TABLES").ToList();
    var query = @"select COLUMN_NAME
                from INFORMATION_SCHEMA.COLUMNS
                where TABLE_NAME = @tableName
                and(COLUMN_NAME like 'lat%' or COLUMN_NAME like 'lon%')
                and (DATA_TYPE in ('numeric', 'int', 'decimal', 'float'))";
    var result = tableNames.Select(n => new DataSource
    {
        TableName = n,
        LatLonProperties = db.Database.SqlQuery<string>(query, new SqlParameter("tableName", n)).ToList()
    }).ToList();

    return result;
}
```

Figure 71 Other actions in the controller of server

The GetProperties() action will take a dataSource as parameter then it will retrieve all the columns from this table. The GetDataSources() action will look for all the tables and guess out the possible latitude and longitude columns for each table.

Data Model

There are three data models that handle the database queries. The first is the GeoChartDataModelDto. This is the data transfer object that wraps the data needed to be visualized. It matches the structure of the data model in client side, as seen below.

```
public class GeoChartDataModelDto
{
    public string Id { get; set; }
    public decimal? BarYFeature { get; set; }
    public decimal? BarColorValue { get; set; }
    public decimal? BarColorAlpha { get;  set; }
    public decimal? Latitude { get; set; }
    public decimal? Longitude { get; set; }
    public decimal? PixelX { get; set; }
    public decimal? PixelY { get; set; }

    public string BarDescription { get; set; }
    public string ErrorMessage { get; set; }
}
```

Figure 72 GeoChartDataModelDto entity class

The second is the DataSourceProperty. It has just two fields: PropertyName and PropertyTypeName. It provides information about columns of a table, as seen below.

```
public class DataSourceProperty
{
    public string PropertyName { get; set; }
    public string PropertyTypeName { get; set; }
}
```

Figure 73 DataSourceProperty entity class

The third one is the DataSource class, as seen in Figure 74. It also has just two fields: TableName and LatLonProperties. The LatLonProperties is for filtering out possible columns that contains geolocation information so that in the client side it is easier for user to choose.

```
public class DataSource
{
    public string TableName { get; set; }
    public List<string> LatLonProperties { get; set; }
}
```

Figure 74 DataSource entity class

Database

Before dive into SQL Server, let us have a quick look at the DbContext in the web application. Since the server side is implemented under the .NET MVC framework, the configuration of database is quite simple through scaffolding in the framework. The DbContext looks like this in Figure 75:

```
public class VR_backendContext : DbContext
{
    // You can add custom code to this file. Changes will not be overwritten.
    //
    // If you want Entity Framework to drop and regenerate your database
    // automatically whenever you change your model schema, please use data migrations.
    // For more information refer to the documentation:
    // http://msdn.microsoft.com/en-us/data/jj591621.aspx

    public VR_backendContext() : base("name=VR_backendContext")
    {
        this.Database.Log = s => System.Diagnostics.Debug.WriteLine(s);
    }
}
```

Figure 75 Database context class in .NET MVC framework

This class is then referenced in the controller, so that the controller can easily access the models through this DbContext.

The DbContext is configured by an XML file so that it can connect to the database. In the database, there are for now three tables and their structures looks like in the Figure 76:

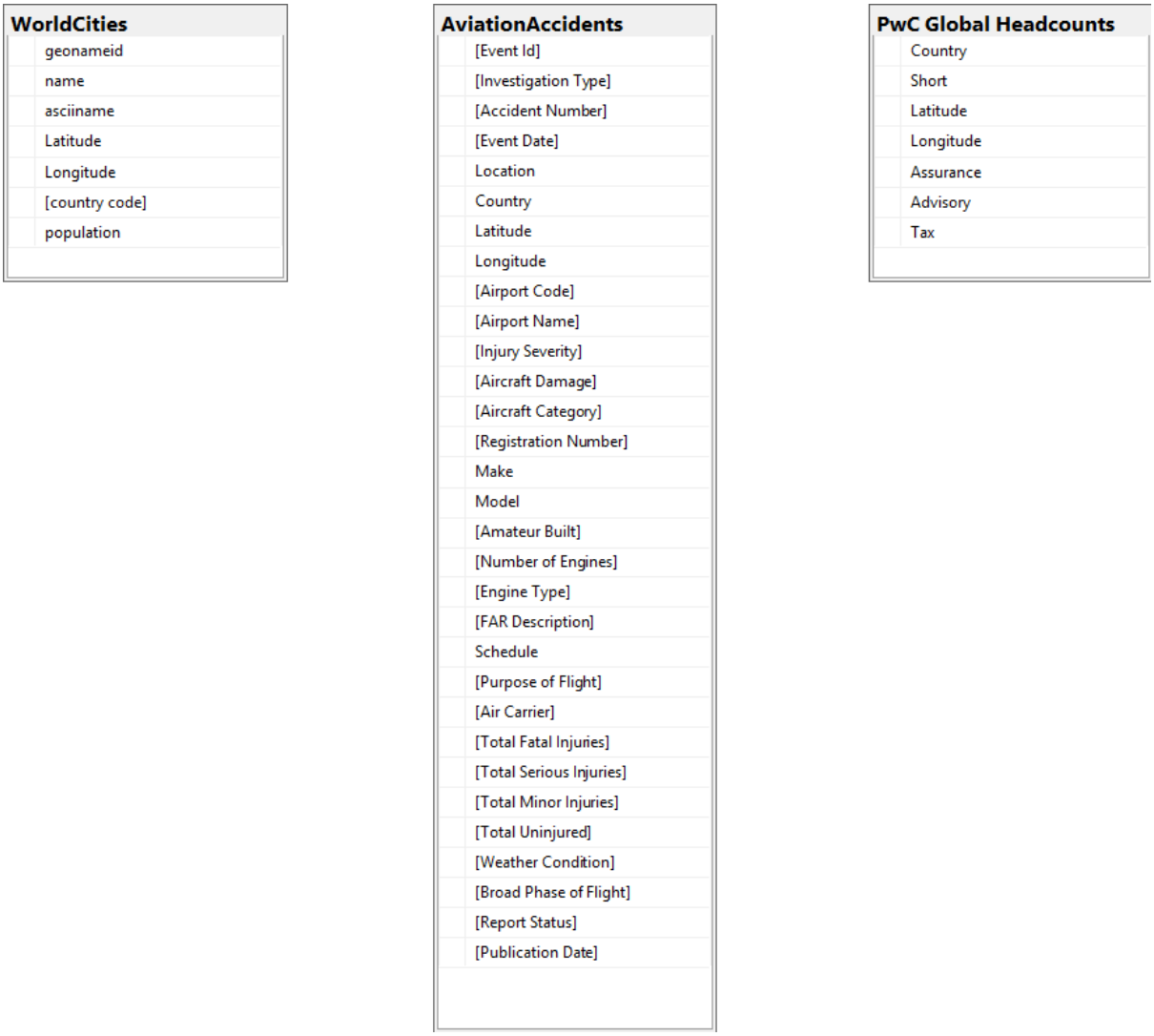


Figure 76 Current data tables in the database

The data are firstly imported as all string fields then value fields are cleaned and transformed into numerical types. For example, for all the three tables, "Latitude" and "Longitude" are the numerical fields because calculations will be done on these two fields. Same applies to the other columns in the tables, for example:

- "population" in WorldCities

- “Total Fatal Injuries”, “Total Serious Injuries”, “Total Minor Injuries” and “Total Uninjured” in AviationAccidents
- “Assurance”, “Advisory” and “Tax” columns in PwC Global Headcounts

They are all numerical columns in order to support aggregation methods. To retrieve the data for visualization, stored procedure in SQL Server is utilized. By using the stored procedure, the database will be able to just take in parameters and run through a series logics and produce the final results. Here it fits the purpose of building flexible SQL queries in terms of dynamic columns, which will be chosen by user. Let us now have a look at the stored procedure that retrieves the data.

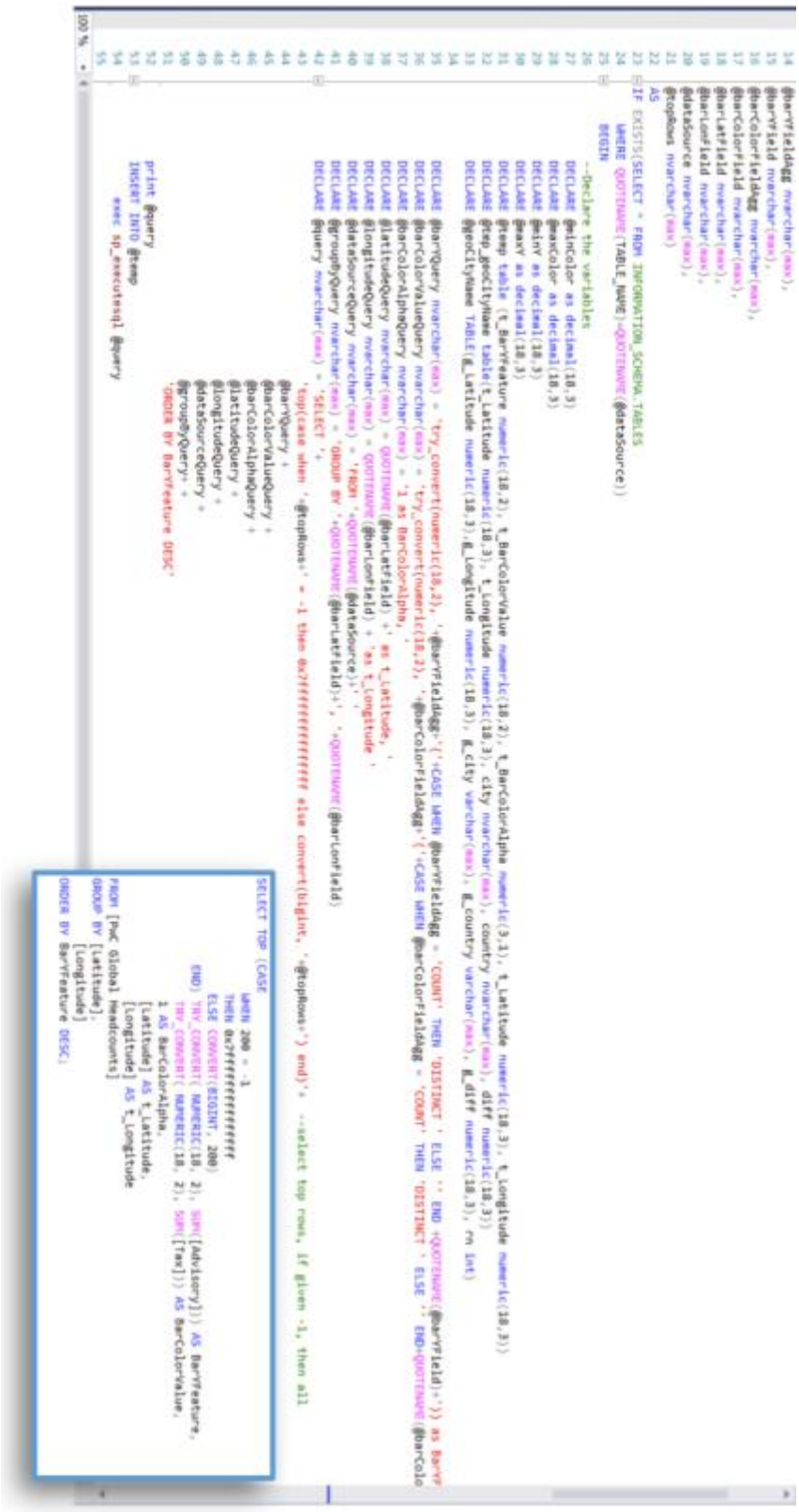


Figure 77 The stored procedure that generate an intermediate @temp table by executing a dynamic SQL query

As seen from the Figure 77, the procedure firstly takes in the variables that are sent from the web application, namely the controller in the server. Then it will check the availability of the required table (line 23 - 24). The next step is to create a temporary table @temp which will do the initial query with some conversions by executing a dynamically concatenated query (line 35 - 54). In this query, the variables from the web application will be inserted. For example, when the user wants to visualize the table “PwC Global Headcounts” with “SUM(Advisory)” as the height of the bar, “SUM(Tax)” as the color of the bar by geolocation, the generated query will be the one at the button right of the Figure 77.

```
set @minY=(select min(t_BarYFeature) from @temp)
set @maxY=(select max(t_BarYFeature) from @temp)

set @minColor = (select min(t_BarColorValue) from @temp)
set @maxColor = (select max(t_BarColorValue) from @temp)


--Create a geolocation-to-cityname translation table
insert into @tmp_geoCityName
    SELECT
        t.t_Latitude,
        t.t_Longitude,
        w.asciiname,
        w.[country code],
        SQRT(SQUARE(t.t_Latitude-w.Latitude) + SQUARE(t.t_Longitude-w.Longitude)) --Calculate the distance between the city and the geolocation
    FROM @temp AS t
    LEFT JOIN WorldCities AS w
    ON ROUND(t.t_Latitude,0) - ROUND(w.Latitude,0) AND ROUND(t.t_Longitude,0) - ROUND(w.Longitude,0) --Select the cities candidates
    WHERE (w.Latitude between t.t_Latitude-0.5 and t.t_Latitude+0.5 )and (w.Longitude between t.t_Longitude-0.5 and t.t_Longitude+0.5);

--Group the table by geolocation then choose the closest city
with cte
as(
    select *, ROW_NUMBER()
        over(partition by tg.t_Latitude, tg.t_Longitude order by diff) rn  --Group the cities candidates based on geolocation and rank them according to the distance
    from @tmp_geoCityName as tg
)
insert into @geoCityName
    select *
    from cte
    where rn=1 --Then select the closest city candidate
```

Figure 78 Prepare and reform the data that are to be selected

The second part of the procedure will be the preparation for normalization and reformation of the data (Figure 78). For example, the height of the bar and the color value of the bar will be normalized by calculating: value / (max - min), thus here the boundary values are calculated (line 56 - 60). In order to translate the geolocation into readable city names, extra information needs to be extracted from WordCities table. The WorldCities table contains basic information of cities such as city name, city latitude, city longitude and city populations. But the cities in the WorldCities are not 100% covering the geolocations in the source data in @temp table, therefore an extra step is to be taken: the city that has the closest geolocation to the geolocation in the source table @temp will be chosen as the corresponding city. Here is how it is done: the difference between the geolocation in source table and that in the WorldCities will be calculated to find the nearest city in the WorldCities, then the city information can be retrieved (line 64 - 86). To limit the selection by filtering out the cities that are too far away, a JOIN operation and a WHERE clause are utilized: The intermediate source table @temp table LEFT JOIN ON the rounded geolocations (line 72 - 73) WHERE the geolocation is within a range of ±0.5 (line 74). And this JOIN result will be saved in another temporary table @tmp_geoCityName with geolocation from the source table (t_Latitude, t_Longitude), city name from WorldCities (city), country code (country) and the geolocation difference (diff) columns (line 64). The next step is: choosing the nearest city as the information source: to repeat, the geolocations in source table @temp do not necessarily find the exact match in the WorldCities table because latitude and longitude are theoretically continuous numbers, therefore the intermediate table @temp_geoCityName is used to save the candidate geolocations from the WorldCities by having the [diff] column. To get the nearest city, the @temp_geoCityName needs to be partitioned and ordered by the [diff], before it is able to

determine the first record which will be chosen as the nearest city to the source geolocation from @temp (line 77 - 86). The result of selecting the nearest cities will be saved in the table @geoCityName of the structure: g_Latitude, g_Longitude, g_city, g_country and g_diff columns.

```
--Final query
select
    try_convert(nvarchar, ROW_NUMBER() OVER (ORDER BY t_BarYFeature DESC)) AS Id,
    try_convert(numeric(3,2),COALESCE((t_BarYFeature - @minY)/nullif(@maxY-@minY,0),0)) as BarYFeature,
    try_convert(numeric(3,2),COALESCE((t_BarColorValue - @minColor)/nullif(@maxColor-@minColor,0),0)) as BarColorValue,
    t_BarColorAlpha as BarColorAlpha,
    try_convert(numeric(18,3), t_Latitude) as Latitude,
    try_convert(numeric(18,3), t_Longitude) as Longitude,
    --t.t_Latitude,t.t_Longitude,--debug
    @barYfield+': '+try_convert(nvarchar,format(round(t_BarYFeature,1),'N', 'de-de'))
    + '\n'+@barColorField+': '+TRY_CONVERT(nvarchar, format(round(t_BarColorValue,1),'N', 'de-de'))
    + '\n Latitude: ' + TRY_CONVERT(nvarchar, format(round(t_Latitude,3), 'N', 'de-de'))
    +'\n Longitude: ' + TRY_CONVERT(nvarchar, format(round(t_Longitude,3), 'N', 'de-de'))
    +'\n City: '+ ISNULL(g.g_city,'-')
    + '\n Country: ' + ISNULL(g.g_country, '-')
    as BarDescription
from @temp AS t
LEFT JOIN @geoCityName AS g
ON t.t_Latitude = g.g_Latitude and t.t_Longitude = g.g_Longitude
where (TRY_CONVERT(numeric(6,1), t_Latitude) is not null and try_convert(numeric(6,1), t_Longitude) is not null)
        and try_convert(numeric(3,2),COALESCE((t_BarYFeature - @minY)/nullif(@maxY-@minY,0),0))>0 --Filter out the data with 0 height after normalization
    END
ELSE
    THROW 60000, 'Data table not found :(', 1
```

Figure 79 Form the result to match the data model

The final part of the procedure (Figure 79) is then forming the result into the corresponding data model that is to be expected by the web application. Here all the column names are assigned, number format are assigned, normalizations are calculated, strings are concatenated and the nulls are handled.

Back to the Controller

Now the data is freshly prepared, the controller wraps up the result as a collection and then dumps the result in the format of JSON to the client that sends the request. Till now, the work of the server side is done.

3.6 Summary of Project Implementation

In this part, the implementation of a VR data visualization prototype is discussed. Multiple design concepts, design guidelines and software design patterns are applied and implemented. The scene is designed to resemble daily working environment with familiar office objects and realistic skybox so that it follows the skeuomorphism methodology and makes the experience more realistic and intuitive. Multimodal principle also helps with creating GUI and interactions. The audio guide or speech, the vibration on the controller during the exploration of data visualization makes it easier for user to comprehend the rich information. Consistent interaction patterns also smooth out the learning curve for user. Selecting objects with a laser pointer and grabbing objects with the grab buttons on the controllers are all designed to provide an intuitive and smooth experience with the guidance of selection and manipulation patterns. And thanks to the utilization of different reference frames, the user can easily access the information or GUI that is in need. Carefully positioned objects in different action areas reveals different levels of priority of information. The user will see the data visualization in the primary action area; in the secondary action area user will have access to GUI which will always follow the position of user; in the tertiary action area, the hidden “Easter egg” is presented, which provides a high level of curiosity to make user come back or stay longer in the application.

During the implementation of the data exchange logics, MVC and element based design patterns are used so that the workflow of the application is optimized for better performance, which is essential in VR applications otherwise the delays on each stage of a VR system will incur motion sickness. On a higher level, the client-server architecture assures the flexibility of the application and the possibility of adapting more visualization logics in the future, because the backend can be

easily move to the cloud, where heavy-loaded jobs such as processing data, restructuring data and preparing data for the visualization will be done. And the cloud computing can be agilely scaled.

Part 4 Conclusion and Outlook

Virtual reality has come a long way. It is a revolutionary medium of communication that reveals information with an extra dimension. This thesis has just made a small step into discovering another possibility of utilizing this technology in non-entertainment areas. In the future, as new potential platforms of computation, virtual reality, augmented reality and mixed reality can play an important role in presenting information, either for entertainment, scientific researches or for businesses. With the prototypical implementation of this thesis, it is demonstrated that, a virtual reality application in data visualization is not only plausible and meaningful, but also feasible. With multimodal communication and high reality fidelity, user can discover underlying meanings, patterns and perspectives of data; with decreasing cost of the hardware equipment, data visualization and analysis in virtual reality with social factors will be more efficient, effective and accessible for more audience.

Of course, this prototype still has a long way to go to become a final product, there are still numbers of improvements to be made. For example, the VR social and collaboration is not yet implemented, however, as described, it is a thrilling feature in VR application as it brings co-exists into the scene. It is also not that difficult to implement this feature anymore. Game engine like Unity has already incorporated multiplayer networking high level APIs, developers can easily realize the basic multiplayer scenario by just using the native Unity supports. Other than that, there are of course other third parties developing SDKs and APIs continuously, for example Photon Unity Networking framework[106] and AltspaceVR[107] SDK, just to name a few.

---

[106] "Photon Unity 3D Networking Framework SDKs and Game Backend Photon: Multiplayer Made Simple."
[107] "AltspaceVR Developer Community – a Developer Community for AltspaceVR."

Another important aspect is the performance. During the actual development, when the application tries to visualize more than 700 - 800 data points, the frame rate drops noticeably, with 1000-2000 points the application is barely usable. It is due to the limitation of the hardware and the efficiency of the application. Optimization is so important regarding the VR experience. For smooth experience the application has been limited to just visualize 200-300 data points. Future improvements should be taken on streamlining and optimizing the application. For example, the data points or data bars right now are carrying scripts, when they all executed during the runtime, it can consume huge amount of CPU and GPU resources. And it will ultimately cause low frame rate and worse, motion sickness. Thus, it will make sense to improve this design to a more centralized and efficient design so that the data points do not have to be attached with scripts with heavy logics hence to reduce the consumptions of system resources and boost the performance.

More interactions and manipulations on the data can be implemented. As mentioned in the section 2.1.3.2, some more interaction patterns including 3D multi-touch so on and so forth can be incorporated so that user can zoom in/out on the map object and even dive deeper into the layers of data for more insights.

More emotional bonding can also be added into the application. In this way, the user would want to spend more time in the application to explore more about the data, the immersive experience would be improved so that the user will be more impressed, as well as satisfied about working on datasets. For example, expressions can be added to the robot in the scene. Spatial audio should be also included in the future improvement as it helps user identify where the sound comes from. If

the audio is attached to the robot and user can identify it, the spatial audio then helps not only build up emotional bonding together with the expressions on the robot, but also helps by acting as a way-finding aid.

To conclude, the discovery of the virtual reality application on data visualization and analysis has just started, it is a small step, but it seems plausible and promising. Its potential is worth to be further explored.

References

Aigner, Wolfgang. “Current Work Practice and Users' Perspectives on Visualization and Interactivity in Business Intelligence..” *Iv*, 2013.

Alger, Mike. “VisualDesignMethodsforVR_MikeAlger.Pdf.” *Aperturesciencellc.com*. Accessed June 28, 2017. http://aperturesciencellc.com/vr/VisualDesignMethodsforVR_MikeAlger.pdf.

Amini, Fereshteh, Sebastien Rufiange, Zahid Hossain, Quentin Ventura, Pourang Irani, and Michael J McGuffin. “The Impact of Interactivity on Comprehending 2D and 3D Visualizations of Movement Data.” *IEEE Transactions on Visualization and Computer Graphics* 21, no. 1 (2015): 122–35. doi:10.1109/TVCG.2014.2329308.

Bailenson, Jeremy N, Nick Yee, Dan Merget, and Ralph Schroeder. “The Effect of Behavioral Realism and Form Realism of Real-Time Avatar Faces on Verbal Disclosure, Nonverbal Disclosure, Emotion Recognition, and Copresence in Dyadic Interaction..” *Presence* 15, no. 4 (2006): 359–72. doi:10.1162/pres.15.4.359.

Ballhaus, Werner, Laura Bruns, Felicia Deligios, Tobias Graber Dr, Sina Kammerling, Manuel Lorenz, Nena Schink, Anna Kristine Wipper Dr., and Niklas Wilke. “Digital Trend Outlook 2016,” August 1, 2016. https://www.biu-online.de/2015/07/27/grosses-interesse-an-virtual-reality-brillen/.

Ben D Lawson. “Motion Sickness Symptomatology and Origins..” *Handbook of Virtual Environments, 2nd Ed.* 20143245 (2014): 531–600. doi:10.1201/b17360-29.

Bente, G, S Rüggenberg, B Tietz, and S Wortberg. *Bente: Measuring Behavioral Correlates of Social... - Google Scholar*, International Communication …, 2004.

Bolt, Richard A, and Richard A Bolt. *“Put-That-There”: Voice and Gesture at the Graphics Interface*. *ACM SIGGRAPH Computer Graphics*. Vol. 14. Voice and Gesture at the Graphics

Interface, ACM, 1980. doi:10.1145/965105.807503.

Bourque, Brad. “Spec Comparison: Can the Plucky PlayStation VR Upset HTC’s Vive?,” March 2016. http://www.digitaltrends.com/virtual-reality/playstation-vr-vs-htc-vive-spec-comparison/.

Bowman, Doug A, and Larry F Hodges. *An Evaluation of Techniques for Grabbing and Manipulating Remote Objects in Immersive Virtual Environments*. *The 1997 Symposium*, New York, New York, USA: ACM, 1997. doi:10.1145/253284.253301.

Bowman, Doug A, Ryan P McMahan, and Eric D Ragan. “Questioning Naturalism in 3D User Interfaces.” *Communications of the ACM* 55, no. 9 (September 1, 2012): 78–88. doi:10.1145/2330667.2330687.

Bowman, Douglas A. *Interaction Techniques for Common Tasks in Immersive Virtual Environments: Design, Evaluation, and Application*, Georgia Institute of Technology, 1999.

Csikszentmihalyi, Mihaly. *Flow: the Psychology of Optimal Experience*, New York: Harper Perennial Modern Classics, 2008.

Darken, Rudolph P, and John L Sibert. “Wayfinding Strategies and Behaviors in Large Virtual Worlds..” *Chi*, 1996.

Davies, Alex. “Oculus Rift, HTC Vive, PlayStation VR - Tracking \& Controls,” May 2016. http://www.tomshardware.com/reviews/vive-rift-playstation-vr-comparison,4513-6.html.

Donalek, Ciro, S George Djorgovski, Scott Davidoff, Alex Cioc, Anwell Wang, Giuseppe Longo, Jeffrey S Norris, et al. “Immersive and Collaborative Data Visualization Using Virtual Reality Platforms..” *CoRR* cs.HC (2014).

Garau, Maia. “The Impact of Avatar Fidelity on Social Interaction in Virtual Environments,” 2003.

Guadagno, Rosanna E, Jim Blascovich, Jeremy N Bailenson, and Cade Mccall. "Virtual Humans and Persuasion: the Effects of Agency and Behavioral Realism." *Media Psychology*, December 5, 2007. doi:10.1080/15213260701300865;wgroup:string:Publication.

Haan, Gerwin de, Michal Koutek, and Frits H Post. *IntenSelect: Using Dynamic Object Rating for Assisting 3D Object Selection*, The Eurographics Association, 2005. doi:10.2312/EGVE/IPT_EGVE2005/201-209.

Jacob, Robert J K. "The Use of Eye Movements in Human-Computer Interaction Techniques - What You Look at Is What You Get.." *ACM Trans. Inf. Syst.* 9, no. 2 (1991): 152–69. doi:10.1145/123078.128728.

Jacob, Robert J K. "The Use of Eye Movements in Human-Computer Interaction Techniques: What You Look at Is What You Get." *ACM Transactions on Information Systems (TOIS)* 9, no. 2 (April 1, 1991): 152–69. doi:10.1145/123078.128728.

Jeffrey Grubb. "PlayStation VR Social Demo Amazes Crowd of Developers." *Youtube.com*. Accessed November 9, 2016. https://youtu.be/sK8tMwlZLEM.

Jerald, Jason. *The VR Book: Human-Centered Design for Virtual Reality*, Morgan & Claypool Publishers, 2015.

Jerald, Jason J. "Scene-Motion- and Latency-Perception Thresholds for Head-Mounted Displays," THE UNIVERSITY OF NORTH CAROLINA AT CHAPEL HILL, 2009.

Ji, Su Mi, Beom Seok Lee, Kyoung Il Kang, Sung Gook Kim, Cheolwhan Lee, Oh-young Song, Joon Yeon Choeh, Ran Baik, and Sung Wook Baik. "A Study on the Generation of OLAP Data Cube Based on 3D Visualization Interaction," 231–34, IEEE, 2011. doi:10.1109/ICCSA.2011.40.

Kennedy, Robert S, Robert C Kennedy, Kristyne E Kennedy, Christine Wasula, and Kathleen M

Bartlett. “Virtual Environments and Product Liability..” *Handbook of Virtual Environments, 2nd Ed.* 20143245 (2014): 505–18. doi:10.1201/b17360-26.

Kopper, Regis, Doug A Bowman, Mara G Silva, and Ryan P McMahan. “A Human Motor Behavior Model for Distal Pointing Tasks.” *International Journal of Human-Computer Studies* 68, no. 10 (October 2010): 603–15. doi:10.1016/j.ijhcs.2010.05.001.

Laviola, Joseph J, David H Laidlaw, Robert C Zeleznik, William A S Buxton, and Peder J Estrup. “Whole-Hand and Speech Input in Virtual Environments,” June 12, 2000.

Lindeman, Robert W, and Steffi Beckhaus. “Crafting Memorable VR Experiences Using Experiential Fidelity..” *Vrst*, 2009, 187. doi:10.1145/1643928.1643970.

Martin, Jean-claude. “TYCOON: Theoretical Framework and Software Tools for Multimodal Interfaces,” 1998.

McAfee, Andrew, and Erik Brynjolfsson. “Big Data: the Management Revolution,” October 2012.

Moran, A, V Gadepally, M Hubbell, and J Kepner. “Improving Big Data Visual Analytics with Interactive Virtual Reality,” 1–6, 2015. doi:10.1109/HPEC.2015.7322473.

Norman, Don. *The Design of Everyday Things: Revised and Expanded Edition*. Revised and expanded edition., New York, New York: Basic Books, 2013.

Oculus. “Oculus Avatars: Maximizing Social Presence.” *Youtube.com*. Accessed June 25, 2017. https://www.youtube.com/watch?v=X6XOwtcscnY.

Osarek, Joerg, Carsten Frisch, Krzysztof Izdebski, Petr Legkov, Maximilian C Maschmann, Chuck Ian Gordon, Alexander Scholz, Frank Sommerer, and Kevin Williams. *Virtual Reality Analytics: How VR and AR Change Business Intelligence*. 1st ed., Gordon's Arcade, 2016.

Schultheis, U, J Jerald, F Toledo, A Yoganandan, and P Mlyniec. *Comparison of a Two-Handed*

Interface to a Wand Interface and a Mouse Interface for Fundamental 3D Tasks. *2012 IEEE Symposium on 3D User Interfaces (3DUI*, IEEE, 2012. doi:10.1109/3DUI.2012.6184195.

Unity. "GameObjectCubeExample.Png (582×457)." *Docs.Unity3d.com*. Accessed June 25, 2017. https://docs.unity3d.com/560/Documentation/uploads/Main/GameObjectCubeExample.png.

Unity. "GameObjectLightExample.Png (600×404)." *Docs.Unity3d.com*. Accessed June 25, 2017. https://docs.unity3d.com/560/Documentation/uploads/Main/GameObjectLightExample.png.

Usoh, Martin, Kevin Arthur, Mary C Whitton, Rui Bastos, Anthony Steed, Mel Slater, and Frederick P Brooks Jr. *Walking > Walking-in-Place > Flying, in Virtual Environments*. *The 26th Annual Conference*, New York, New York, USA: ACM Press/Addison-Wesley Publishing Co., 1999. doi:10.1145/311535.311589.

Ward, Matthew O, Georges Grinstein, and Daniel Keim. *Interactive Data Visualization*, CRC Press, 2015.

Webster's New Universal Unabridged Dictionary, Barnes \& Noble Books, 1989.

"2017 Global Digital IQ Survey: Virtual Reality," May 2017.

"A Critical Look at Virtual Reality," April 2016.

"AltspaceVR Developer Community – a Developer Community for AltspaceVR." Accessed June 25, 2017. https://developer.altvr.com/.

"BMW I Samsung Virtual Reality Experience." *Oculus.com*. Accessed June 25, 2017. https://www.oculus.com/experiences/gear-vr/876847565746067/.

"Chap23-Transitioning to Vr Content Creation," August 12, 2016, 1–6.

"Corpses, Androids, and the Polar Express: a Social Neuroscience Perspective on the Uncanny Valley," May 23, 2009.

"Definition of VIRTUAL REALITY." *Merriam-Webster.com*. Accessed June 28, 2017. https://www.merriam-webster.com/dictionary/virtual+reality.

"Earth." *Vr.Google.com*. Accessed December 5, 2016. https://vr.google.com/earth/.

"Editor-Breakdown.Png." *Docs.Unity3d.com*. Accessed June 28, 2017. https://docs.unity3d.com/uploads/Main/Editor-Breakdown.png.

"Facebook Details Social VR Avatar Experiments and Lessons Learned – Road to VR." *Roadtovr.com*, November 1, 2016. http://www.roadtovr.com/facebook-details-social-vr-avatar-experiments-and-lessons-learned/.

"Facebook Social VR Demo - Oculus Connect 2016." Accessed June 28, 2017. https://www.youtube.com/watch?v=YuIgyKLPt3s.

"Figure1.Gif." *Cs.Cmu.Edu*. Accessed December 12, 2016. http://www.cs.cmu.edu/~stage3/publications/95/conferences/chi/figure1.gif.

"Home - FOVE Eye Tracking Virtual Reality Headset." *Getfove.com*. Accessed June 25, 2017. https://www.getfove.com/.

"Introducing Viveport," August 2016. https://blog.vive.com/us/2016/08/05/introducing-viveport/.

"Nosulus Rift." Accessed November 9, 2016. http://nosulusrift.ubisoft.com/?lang=en-GB#!/introduction.

"OC3 Reveals: Touch Launch, Santa Cruz Prototype, Min Spec, 400+ Mobile VR Apps, and More." *Oculus.com*. Accessed June 28, 2017. https://www.oculus.com/blog/oc3-reveals.

"Oculus Best Practices." *Oculus.com*. Accessed November 2, 2016. http://static.oculus.com/documentation/pdfs/intro-vr/latest/bp.pdf.

"Openvr: OpenVR SDK," Valve Software, June 2017.

"Orion." *Developer.Leapmotion.com*. Accessed June 27, 2017. http://developer.leapmotion.com/orion/.

"Photon Unity 3D Networking Framework SDKs and Game Backend Photon: Multiplayer Made Simple." *Photonengine.com*. Accessed June 25, 2017. https://www.photonengine.com/en/PUN.

"Profiles in Innovation - Virtual & Augmented Reality." *Goldmansachs.com*. Accessed September 16, 2016. http://www.goldmansachs.com/our-thinking/pages/technology-driving-innovation-folder/virtual-and-augmented-reality/report.pdf.

"Red Alert 2 Remake in Unreal Engine 4 - Allies Gameplay in VR Using HTC Vive.." *Youtube.com*. Accessed December 9, 2016. https://www.youtube.com/watch?v=IGtrJaiUZaA.

"Technology, Media & Telecommunications Predictions 2016." *Www2.Deloitte.com*. Accessed September 16, 2016. http://www2.deloitte.com/content/dam/Deloitte/global/Documents/Technology-Media-Telecommunications/gx-tmt-prediction-2016-full-report.pdf.

"This Engine Is Dominating the Gaming Industry Right Now." *Thenextweb.com*, March 24, 2016. https://thenextweb.com/gaming/2016/03/24/engine-dominating-gaming-industry-right-now/.

"U716." *Xmg.Gg*. Accessed June 28, 2017. http://www.xmg.gg/u716/.

"Unity 5 vs Unreal Engine 4," September 8, 2015.

"Virtual Reality - IKEA." Accessed June 25, 2017. http://www.ikea.com/ms/en_US/this-is-ikea/ikea-highlights/Virtual-reality/index.html.

"Visionary VR Is Reinventing Filmmaking's Most Fundamental Concept to Tell Stories in

Virtual Reality," January 5, 2015.

"VIVE™ Vive Ready Computers." *Vive.com*. Accessed April 5, 2017. https://www.vive.com/us/ready/.

"WebVR - Bringing Virtual Reality to the Web." *Webvr.Info*. Accessed June 25, 2017. https://webvr.info/.

Declaration

I hereby declare that the present paper and the work reported herein was composed by and originated entirely from me without any help. All sources used from published or unpublished work of others are reported in the list of references. All parts of my work that are based on others' work are cited as such. The Master's Thesis has not been submitted for any degree or other purposes, neither at the TH Köln – University of Applied Sciences nor at any other university or college.

Düsseldorf, June 28th 2017

Xinzhou Zhang