VR Multimedia Analytics Software Usability: Implementing and Assessing Novel Application Features in ViRMA

Jonas Sylvain Jersild Balin (jbal@itu.dk)
Katarzyna Honorata Toborek (ktob@itu.dk)

IT UNIVERSITY OF COPENHAGEN

Master Thesis
STADS: KISPECI1SE
IT University of Copenhagen
Denmark
01/06/2022

Abstract

This paper examines ViRMA, a Scalable Multimedia Analytics (SMA) virtual reality application based on the M^3 data model, with the purpose of examining its usability, proposing potential improvements, implementing these, evaluating them via user tests, and presenting implications and guidelines for future development of similar SMA VR applications.

Based on established literature on VR usability, several potential improvements were identified, such as implementing passthrough technology to allow users to view their physical environment, adding physical keyboard interaction to allow for a familiar text input method, and adding hand tracking to allow for controller-free interaction with the UI. Due to various reasons presented in this paper, the implementations suffered some necessary compromise that resulted in an expected lower usability.

Usability tests were conducted with the aim of evaluating the usability of ViRMA pre- and post-changes. Broadly, the usability study found that the changes generally improved the usability of the software, with some caveats due to sub-optimal implementations.

These findings lead to discussions on the future development of ViRMA and similar VR applications, including recommendations for future development of the initial implementation of ViRMA, future development of the post-changes ViRMA, and a potential Oculus Quest 2 native ViRMA port.

Contents

1 Introduction **1**

2 Background **2**
2.1 Previous works . . . 2
2.1.1 ObjectCube - A Generic Multi-Dimensional Model for Media Browsing . . . 2
2.1.2 PhotoCube: A Multi-Dimensional Image Browser . . . 2
2.1.3 A Web Browser Based Implementation of PhotoCube . . . 2
2.1.4 ViRMA: Virtual Reality Multimedia Analytics at LSC 2021 . . . 3
2.1.5 Layered VR-User Interfaces - Onboarding of users in ViRMA . . . 3
2.2 SMA and VR - related work . . . 3
2.2.1 Towards Interactive, Intelligent, and Integrated Multimedia Analytics . . . 3
2.2.2 Integration of Exploration and Search: A Case Study of the M^3 Model . . . 4
2.2.3 Comparing Approaches to Interactive Lifelog Search at the Lifelog Search Challenge (LSC2018) . . . 4
2.2.4 Exploring Intuitive Lifelog Retrieval and Interaction Modes in Virtual Reality with vitrivr-VR . . . 5
2.3 The M^3 Data Model . . . 5
2.3.1 Objects . . . 5
2.3.2 Tags . . . 5
2.3.3 Tag-sets . . . 5
2.3.4 Hierarchies . . . 5
2.3.5 Dimensions . . . 6

3 Virtual reality and usability **7**
3.1 VR input devices . . . 7
3.2 Hand input device classes . . . 7
3.2.1 Bare hands . . . 8
3.3 Text inputs in virtual reality . . . 10
3.3.1 VR keyboard integration . . . 11
3.4 Motion sickness and virtual environments . . . 12

4 ViRMA **15**
4.1 API and Hardware . . . 15
4.1.1 Project Scene . . . 15
4.2 Using the Software . . . 16
4.2.1 Main Menu and tag search . . . 16
4.2.2 Dimension Explorer and filter projection . . . 17
4.2.3 Visualization Controller and image browsing . . . 19
4.3 Data-set . . . 20

5 Analysis and implementation of potential usability improvements **21**
5.1 Hardware: Oculus Quest 2 . . . 21
5.2 Identifying potential usability improvements . . . 21
5.2.1 Environment . . . 22
5.2.2 Hand- and text input . . . 23

5.3 Implementation . . . 23
5.3.1 Porting to Oculus VR . . . 23
5.3.2 Passthrough . . . 25
5.3.3 Tracked Keyboard . . . 26
5.3.4 Hand Tracking . . . 27
5.3.5 Other changes . . . 28
5.3.6 Interaction . . . 28
5.3.7 Initializing query search . . . 29
5.3.8 Interacting with search results and projecting filters . . . 29
5.3.9 Limitations and impacts . . . 31

6 Usability testing **32**
6.1 Methodological considerations . . . 32
6.2 Structure . . . 33
6.2.1 Tasks . . . 33
6.2.2 Test users . . . 34
6.3 Testing structure . . . 35
6.3.1 Introductory conversation . . . 35
6.3.2 Introduction – slides . . . 35
6.3.3 Test A & B . . . 36
6.3.4 Post-test interviews A & B . . . 36
6.3.5 (Optional) break . . . 36
6.3.6 Semi-structured interview . . . 36
6.4 Concluding survey . . . 37
6.5 Test iteration 1 and 2 . . . 37
6.5.1 Quest 2 changes for second iteration . . . 38
6.6 Hardware . . . 39
6.7 Set-up or test errors . . . 39

7 Usability test analysis **41**
7.1 Immersive environment / Passthrough environment . . . 42
7.2 Hand-held controllers / bare hands . . . 44
7.3 Virtual / physical keyboards . . . 47
7.4 Holistic review of findings . . . 48

8 Discussion and reflections **52**
8.1 Methodology, practical considerations . . . 52
8.1.1 Implementation faults . . . 52
8.1.2 Usability testing vulnerabilities . . . 52
8.2 Usability test result implications . . . 54
8.3 Future work . . . 55
8.3.1 Hands API . . . 55
8.3.2 Native Quest 2 . . . 57
8.3.3 SteamVR iteration of ViRMA . . . 57

9 Conclusion **58**

A ViRMA Usability testing mail **62**

B Usability testing presentation **63**

C Usability testing concluding survey **69**

D Usability testing concluding survey responses **71**

E Running the software **76**

F Usability testing notes **77**

G Usability Schema **122**

List of Figures

1 Tag-set vs Hierarchy . . . 6
2 Projected filters on X and Y axis . . . 6
3 Comparison of hand and non-hand input device classes [16] (p. 312) . . . 8
4 Hand tracking in Quest 2 home environment [30] . . . 9
5 Different types of text input methods [17], (p. 6) . . . 10
6 Usability statistics for VR text input methods [17] . . . 11
7 Tracked Logitech K830 keyboard with passthrough hands in Oculus Home environment [23] . . . 12
8 VR boundary illustration [29] . . . 13
9 ViRMA `MainMenu` . . . 16
10 ViRMA Controllers (Valve Index Knuckles) . . . 17
11 ViRMA Keyboard (`DimExKeyboard`) . . . 17
12 ViRMA `DimensionExplorer` . . . 18
13 ViRMA Tag Projection . . . 18
14 ViRMA `VisualizationController` . . . 19
15 ViRMA `Timeline` . . . 20
16 ViRMA Active Filters in `MainMenu` . . . 20
17 Oculus Touch controllers . . . 24
18 Room as seen with passthrough layer [3] . . . 26
19 `SearchBar` . . . 26
20 Hand tracking in ViRMA . . . 27
21 Populated `SearchResults` menu with active Axis Projection and Filter clearing menu 28
22 Populated `SearchResults` menu . . . 29
23 Menu placement on flat surface illustration (original image from [9]; blue surface is an original addition) . . . 30
24 Usability test structure . . . 35
25 `SearchResults` menu with colour coded buttons . . . 38
26 SteamVR and Quest 2 identified usability defects related to environment . . . 42
27 Passthrough environment survey question chart . . . 43
28 SteamVR and Quest 2 identified usability defects related to environment . . . 45
29 Controls survey question chart . . . 46

30 Hand tracking survey question chart 47
31 SteamVR and Quest 2 keyboard-related identified usability issues 48
32 Treemaps of frequency of usability issues classified by heuristic. Numbers in blocks represent usability heuristic and percentage of total occurrences, respectively 50
33 Participant enjoyment survey question chart 51
34 Oculus Passthrough settings demonstration [34] 55
35 Hand iteration comparison, clapping operation [14] 56
36 Usability testing concluding survey form 69
37 Usability testing concluding survey form - open questions 70
38 Usability testing concluding survey question 1 responses 71
39 Usability testing concluding survey question 2 responses 71
40 Usability testing concluding survey question 3 responses 72
41 Usability testing concluding survey question 4 responses 72
42 Usability testing concluding survey question 5 responses 73
43 Usability testing concluding survey question 6 responses 74
44 Usability testing concluding survey question 7 responses 75

List of Tables

1 Superficial differences between test iterations 37
2 Q: Being able to see my physical environment was beneficial to my experience of the software 43
3 Q: I preferred the controls in the SteamVR implementation 46
4 Q: I preferred using physical controllers to hand tracking 46
5 Identified usability heuristic frequencies 50
6 Q: If I had to use ViRMA for work or personal enjoyment, I would prefer using the Index version over the Quest 2 51

1 Introduction

At the time of this thesis' publication, Virtual Reality headsets have reached extensive adoption rates, with analytical forecasts projecting sales growing the coming years. While predicting future states of technology is inherently a volatile endeavor, betting on VR becoming a mainstay in the same manner as mobile phones and personal computers is becoming safer by the year. This push is aided in no small part by the world's largest social media conglomerate Meta (formerly Facebook), which is putting its efforts to create a future where entertainment, productivity, and fitness applications are going to be served to consumers through VR headsets [21].

On a separate front, both private and commercial multimedia collections are growing rapidly. This has pushed researchers within the niche academic field of Scalable Multimedia Analytics to devise innovative and usable data models and software that allow for the exploration and analysis of vast data-sets of multimedia content. Among these data models is M^3 ("Emm-cubed"), which facilitates multi-dimensional exploration and analysis of any type of media content, along with traversing tags ordered in a hierarchical tree-like data structure.

Practical implementations of the M^3 data model include PhotoCube. Developed in 2011 but furthered in 2019, PhotoCube currently exists as a web-browser based implementation of the M^3 data model, allowing users to navigate hierarchies of tags and project photos tagged with these onto multiple axes in a three-dimensional space.

Recognizing the increased adoption and potential mainstream breakthrough of virtual reality, researchers at the IT University of Copenhagen have developed the Virtual Reality Multimedia Analytics software (ViRMA). This software can also be considered a realization of the M^3 data model, building upon the aforementioned PhotoCube.

However, being not just an application in a developing market (VR), but also one of the first virtual reality scalable multimedia analytics applications, ViRMA is facing unique challenges. Among these challenges are questions of how to translate complex features and theory of the M^3 data model and PhotoCube software to utilize the unique affordances the virtual reality medium holds.

In this thesis, we will present ViRMA as it exists currently, seek to evaluate potential avenues for improvement, design and implement these, and evaluate their impact on usability with testing. In doing so, we will implement several features that are novel to ViRMA and potentially similar VR applications. We will seek to ground every design decision in established literature on VR design, attempting to improve the usability of ViRMA without sacrificing functionality.

The goal of this thesis is to supply future development of ViRMA and similar VR applications with guidelines or recommendations for design choices, as well as propose direction for further research.

2 Background

2.1 Previous works

This thesis builds upon numerous previous works that will be described here. This section will prioritize literature that is closely related to this thesis and ViRMA. Therefore, certain works such as face recognition plugins for ObjectCube will not be described.[1] In chronological order of release, the texts that will be examined are:

- Grímur Tómasson's ObjectCube thesis (2011)
- Hlynur Sigurþórsson's PhotoCube thesis (2011)
- Peter Clausen's web browser based implementation of PhotoCube thesis (2019)
- Aaron Duane and Björn Jónsson's presentation of ViRMA (2021)
- Gina-Josephine Fischer, Sebastian Matthew Lajciak and Ida Maria Aunsbjerg Villadsen's work in progress thesis about the onboarding in ViRMA (2022)

Following this section, an in-depth explanation of the ViRMA software will be given.

2.1.1 ObjectCube - A Generic Multi-Dimensional Model for Media Browsing

The initial thesis for ObjectCube was released in January of 2011 by Grímur Tómasson. The motivation for developing ObjectCube was due to an identified issue of growing organization and collection of growing multimedia collections. Based on online analytical processing (OLAP), this thesis establishes the foundational concepts of the data organization and exploration ViRMA and PhotoCube rely on. These include (but are not limited to) the concepts of tag-sets, hierarchies, filters, and hypercubes. [33]

2.1.2 PhotoCube: A Multi-Dimensional Image Browser

Hlynur Sigurþórsson released their thesis on PhotoCube in June of 2011. The thesis, similar to Tómasson's thesis, was motivated by the growth of personal photo collections making efficient photo browsing less feasible. Sigurþórsson developed a prototype of PhotoCube, the architecture of which was based on the ObjectCube data model. This thesis iterated upon concepts introduced in Tómasson's thesis and introduced further important concepts such as browsing modes. [26]

2.1.3 A Web Browser Based Implementation of PhotoCube

As part of their thesis released in April 2019, Peter Clausen developed a web browser based implementation of the previously mentioned PhotoCube client. This was primarily motivated by improving the usability, maintainability, and functionality of the software. The thesis concluded that the web browser based implementation was superior to the prior ones in terms of maintainability and portability, but suffered some usability issues. [4] As part of our preparation for this thesis, we conducted a minor research project wherein we improved this client's usability. [1]

[1] Though their history in shaping the overall research should not be dismissed

2.1.4 ViRMA: Virtual Reality Multimedia Analytics at LSC 2021

In this paper for the Lifelog Search Challenge 2021 (LSC), Duane and Jónsson presented the virtual reality prototype of PhotoCube. The motivation for developing the ViRMA prototype arose from what Duane and Jónsson perceived as the emphasis on interactive analysis, availability of novel access mechanisms, and increasing collection scales. As the purpose of the text is to present the software for the Lifelog Search Challenge, it does not make conclusions on usability or future of the software.[7][2]

2.1.5 Layered VR-User Interfaces - Onboarding of users in ViRMA

This thesis is written by Ida Villadsen, Gina Fischer, and Sebastian Lajciak. At the time of writing, it has not yet been finished, but it is expected to be during September of 2022. Due to researching the same topic, our two research teams have collaborated on exchanging data. The purpose of this thesis is to explore and improve the onboarding of users in the aforementioned ViRMA software. While it is not finished, usability studies with ViRMA have been conducted and from these they can determine some usability issues with the software. Namely the dimension browser (described in section 4.2.2) and onboarding experience.[10] Though it is noted that the conclusions of the thesis may change by release, hence why this paper can not fully accept its conclusions.

2.2 SMA and VR - related work

Due to the niche nature of the scalable multimedia analytics (SMA) field, particularly with regards to VR, there is limited literature available on the topic. While SMA is not a central topic of this thesis, it is important to draw the connection between SMA and ViRMA, therefore in the following section we briefly introduce other work that has been conducted in relation to it. First we present a paper from 2014 which sets the ground for the development of SMA applications. Next, we outline a paper presented at the 25th International Conference on MultiMedia Modeling (2019), which draws on conclusions formulated in the aforementioned paper. As it pertains the M^3 model, which is the theoretical foundation of ViRMA, it is highly relevant. Then a research article which compared six interactive search approaches to the LSC 2018 is presented. Finally, we introduce a different system presented in the LSC context, which like ViRMA employs virtual reality to facilitate multimedia retrieval. Altogether, these papers contextualize SMA in relation to ViRMA.

2.2.1 Towards Interactive, Intelligent, and Integrated Multimedia Analytics

In this study, Zahálka and Worring conducted an extensive analysis of one hundred research papers related to multimedia analytics, compiling the existing techniques for user interaction with multimedia collections. They present the human and machine perceptions of multimedia, pointing out two main gaps between the approaches:

- **semantic gap** - the human is able to retrieve much more relevant information from a multimedia item than the machine, which relies on algorithms and having been trained on multiple data-sets among other things
- **pragmatic gap** - media categorization model as performed by human is much more flexible than that performed by a machine.

[2]The aspects of usability and its implications for the future of the software will be thoroughly examined and presented in this thesis.

These gaps cover some of the most important drawbacks in the human and machine perceptions of multimedia. Bridging these would allow for more focused and effective searching and browsing of large multimedia data-sets. The authors of the paper conclude with presenting a multimedia analytics model which incorporates the surveyed techniques, attempting to bridge some aspect of the semantic and pragmatic gaps. An important outcome of this work is an exploration-search axis, that in between these two extremes contains tasks such as "Browsing", "Finding relevant items" or "Ranking", which users of such system should be able to manage.[39]

Towards Interactive, Intelligent, and Integrated Multimedia Analytics identifies the obstacles that need to be overcome to achieve an impeccable multimedia analytics systems. Significant part of the described issues relates to the immense sizes of modern multimedia collections, thus the scalability aspect is very important in these applications. All in all, the model proposed in this paper lays the groundwork for the future development of efficient SMA systems.

2.2.2 Integration of Exploration and Search: A Case Study of the M^3 Model

This paper focuses on extending the exploration-focused M^3 model (described in detail in section 2.3) in a way that supports search aspects, more precisely what the authors call *"searching within an exploration context and exploring within a search context"*. In the process the authors draw upon the work of Zahálka and Worring [39] outlined in the previous section. They state that a multimedia analytics system capable of supporting the growing needs in that field must contain both exploration and search aspects. The goal of extending the M^3 model to allow search is achieved, however with certain state of the technology limitations related to lack of dynamic support for subcollections of media defined by users. [11]

Integration of Exploration and Search: A Case Study of the M^3 Model is an important text that sheds more light on some practical issues related to SMA. The authors evaluated the performance of a prototype based on relational database on collection sizes ranging from 1K to 1M objects. The results led them to rule out systems based on relational databases as being suitable for multimedia exploration, due to lack of interactive performance with collections which are still relatively small in the SMA context. [11]

2.2.3 Comparing Approaches to Interactive Lifelog Search at the Lifelog Search Challenge (LSC2018)

This text describes, compares and evaluates approaches to multimedia analytics applications presented by 6 different teams at the Lifelog Search Challenge (LSC) 2018. The participants of the LSC submit systems that are then subjected to completing challenges of retrieving the relevant items among lifelog multimedia data, consisting of large image collections. The evaluation is based on performance in the different lifelog search tasks, where time taken to solve a task and number of incorrect submissions are used to calculate the score. The evaluation has shown that the three teams that performed the best had used existing retrieval systems in their solutions. According to the authors, this has significantly reduced the probability of having technical difficulties and thus contributed to the better overall performance. [12]

The systems competing in the LSC bring in different approaches to the problem of searching in large multimedia collections. Collecting and comparing data about various solutions and their performance is an important work for the SMA, as it highlights the features that contribute to advancing in that field or indicates the directions which might not be optimal.

2.2.4 Exploring Intuitive Lifelog Retrieval and Interaction Modes in Virtual Reality with vitrivr-VR

This paper, written with focus on participation in the LSC, describes a multimedia retrieval system called vitrivr-VR. The system can be used for searching through 3D model, audio, images and video collections. The authors concentrate on creating novel multimedia retrieval user interactions that take advantage of the 3D space and interaction methods unique to VR. It is concluded that despite the good efforts to translate traditional interfaces to those VR-specific, there is still a lot of room for improvement when it comes to utilizing the possibilities stemming from the accessibility of virtual space. [27]

Exploring Intuitive Lifelog Retrieval and Interaction Modes in Virtual Reality with vitrivr-VR similarly to ViRMA tackles some of the challenges still present in modern SMA.

2.3 The M^3 Data Model

ViRMA is built upon the M^3 data model - an interaction model used to explore multimedia collections. M^3 allows for the utilization of metadata attached to any given media to organize it in a multidimensional space. In this section we will describe the concepts of the data model.

2.3.1 Objects

Objects represent singular files or pieces stored within the data model. While the format of multimedia data is not specified in the model, ViRMA is built with specific focus on photographs.

2.3.2 Tags

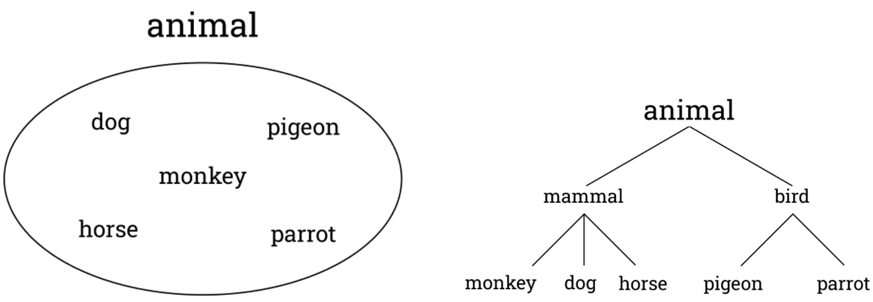
Tags represent meta-data that are associated with any objects. Any object can have an unlimited amount of tags. Tags and objects are in an `M:M` relationship meaning that tags or objects can be associated with `M` amount of the other.

2.3.3 Tag-sets

Tag-sets are sets of tags that are in some manner related to each other. Examples could be a tag-set of `[zebra, rhino, cow, pig, goat, sheep]` representing animals. Tags can also appear in multiple tag-sets. For example, the tags `[cow, pig, goat, sheep]` could appear both in a tag-set of animals, mammals, and farm animals. Tag-sets can also aid in clarifying ambiguous tags, such as the tag `speaker` being a part of the tag-set of persons or audio electronics.

2.3.4 Hierarchies

Tag hierarchies are collections of tags ordered in a hierarchy resembling a tree data structure. Each node is a tag contained in the collection, and the node is related to one or more nodes by being parent or child. Figure 1 illustrates the difference between a hierarchical collection and tag-sets.



(a) Tag-set "Animals" (b) Hierarchy "Animal"

Figure 1: Tag-set vs Hierarchy

2.3.5 Dimensions

Dimensions describe an axis upon which a tag-set or hierarchy can be projected. In the current iterations of PhotoCube and ViRMA, a maximum of three simultaneous dimensions (X, Y, and Z) can be used for projection. Using the LifeLogging data-set as an example, a user could choose to project people on the Y-axis and types of beverage on the X-axis. This would result in a two-dimensional chart where each entry contains the collection of photos associated with tags that fit within both values on the X and Y coordinates. Figure 2 illustrates projected tag-sets and hierarchies on dimension axes. Since the values on liquid (beverages) are noted to be hierarchies, each value may itself contain subordinate hierarchies that could be "drilled down" into. For example, the hierarchy of "Soda" could contain children of different soda brands.

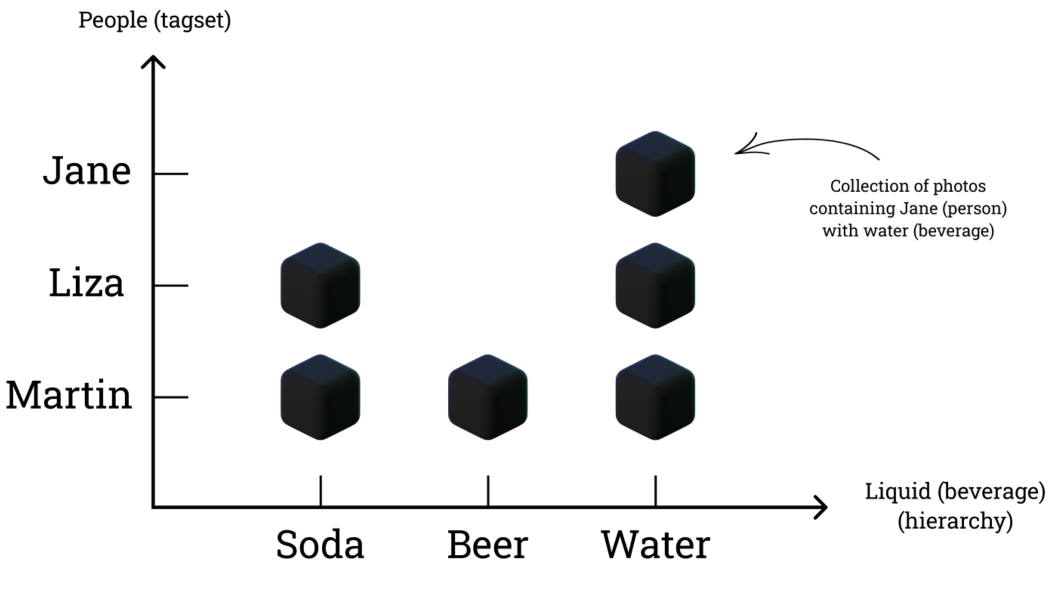


Figure 2: Projected filters on X and Y axis

In addition to the *projected* filters which can be directly applied to either axis, there exists a concept of *direct* filters. To illustrate the concept, applying a direct filter "Monday" to the structure in figure 2 would limit the displayed results to those that were taken on Monday.

3 Virtual reality and usability

This section will examine usability in VR in relation to topics that are relevant for this thesis. The section will cover topics such as VR input devices, text inputs in virtual reality, and mixed reality "passthrough" environments. This is done with the purpose of grounding future developments of ViRMA in established theory and literature.

3.1 VR input devices

In the 2016 edition of his VR handbook, Jerald presents a classification of VR input devices. This classification will be used as a point of reference for examinations and discussions of VR controls.

Jerald delineates between two overall categories of VR input devices. Hand input device classes and non-hand input device classes. As the name implies, one set of input devices is manipulated by the user's hands while the second is not. Figure 3 shows Jerald's classifications of VR input devices along characteristics these devices usually have. For the purposes of this thesis, we are only going to be critically examining hand input device classes. This is because all but the microphone integration of non-hand input device classes rely on external technology that was inaccessible to us. It was deemed that the microphone implementation would not be interesting in comparison to other device classes, as its integration in VR would likely not be much different from integrations in other types of software - as VR generally does not influence one's ability to use their voice.[3]

3.2 Hand input device classes

Hand input device classes were mentioned briefly in the previous section. This set includes classes of devices in VR that are operated with the user's hands. The tracked hand-held controllers are perhaps the most frequently associated with consumer grade VR headsets. Most purchasable consumer grade VR headsets include a set of tracked hand-held controllers, such as the Valve Index's Knuckles, or the Oculus brand's Touch controllers. Tracked hand-held controllers often include buttons, thumbsticks, triggers, and haptic motors (e.g. vibrations on button pres). Due to being tracked, the controllers almost always contain full degree of freedom (6DoF). 6DoF means that the controllers' can transmit a full translation of their position and rotation in 3D space. This is unlike non-tracked input devices such as a traditional console controller, whose highest degree of freedom is 2, the X and Y axes on the thumbsticks. A simple example of using VR controllers' full DoF in software could be grabbing objects and moving them in a natural manner, allowing for movement along three positional axes and rotation of yaw, pitch, and roll.

Hand-tracked controllers' primary limitation is that they usually occupy a full hand, limiting potential interaction with the real world until the controllers are placed [16]. These limitations are rectified by other types of input device that also allow for full DoF. Hand worn controllers, often in the form of a glove, allow for hand interaction with the real world, while carrying some benefits of other types of input, such as haptic feedback. Unlike the aforementioned hand-held controllers,

[3] Unlike, for example, world-grounded devices such as physical keyboard that carry unique difficulties in VR due to lack of visibility not found in other media [16]

	Proprioception	Consistent	Usable in Lap or the Side	Haptics Capable	Unencumbered	Physical Buttons	Hands Free to Interact with Real World	General Purpose
Hand Input Device Class								
World-Grounded Devices	✓	✓		✓	✓	✓	✓	
Non-Tracked Hand-Held Controllers		✓	✓	✓		✓		
Bare Hands	✓				✓		✓	✓
Tracked Hand-Held Controllers	✓	✓	✓	✓		✓		✓
Hand Worn	✓	✓	✓	✓		✓	✓	✓
Non-Hand Input Device Class								
Head Tracking	✓	✓					✓	✓
Eye Tracking							✓	
Microphone			✓		✓		✓	✓
Full-Body Tracking	✓	✓	✓	✓			✓	✓
Treadmills	✓	✓			✓		✓	

Figure 3: Comparison of hand and non-hand input device classes [16] (p. 312)

hand worn devices are primarily relegated to commercial or academic purposes. As noted by Jerald, the technology needs improvement before it could be ready for mass-adoption [16] (p. 316).

The chief input method that will be examined in this paper is "bare hands". This classification of devices covers the user's physical hands with no addition or modification. Sensors track hand motion which then facilitate the translation of physical hand gestures to 3D objects with full DoF. As this class of input device is highly relevant to the thesis, we will explore them in greater detail.

3.2.1 Bare hands

Prior to the increasing popularity of consumer-grade VR headsets, bare hands control had been attempted in other media. Famously, Microsoft launched the Xbox Kinect product in November 2010 [35]. The Kinect was an entertainment product primarily designed for video games, though Microsoft advertisements had also claimed it carried potential for research and commercial purposes [37]. The Kinect was a motion-sensitive camera that when connected to an Xbox 360, Xbox One or Windows PC allowed for full-body tracking, with hand movements and gestures being used for navigating the user interface.[4] Around the time of the Kinect's popularity, other motion sensitive cameras such as the Intel Creative, Leap Motion, and the ASUS XTION cameras existed (though primarily commercial or research oriented).

The first major implementation of hand tracking in VR came from Leap Motion. Leap Motion was a company specializing in hand tracking hardware and software, and in 2014 they released a mount

[4]The Kinect was unlike prior entertainment motion cameras such as the PlayStation EyeToy, as they did generally not include hand tracking

that allowed their motion sensitive cameras to be attached to the front of VR headsets. Utilizing an accompanying SDK, developers could implement bare hand tracking in virtual reality applications [6]. Bare hand tracking in VR approached widespread adoption when Oculus introduced it as a built-in input method for the Oculus Quest, and published an SDK allowing developers to integrate it in applications [2]. The Quest hand tracking relies on four HMD-mounted cameras which uses motion sensing technology to track the hands, similar to the Leap Motion camera mount. Figure 4 depicts hand trancking in Quest 2 home environment.

Figure 4: Hand tracking in Quest 2 home environment [30]

One of the biggest benefits of hand tracking via HMD-mounted cameras is portability, since it does not rely on external tracking stations or static cameras. At the same time, this approach incurs significant cost in precision. Despite multiple cameras on the HMD, their point of view is rather limited. This can lead to issues of occlusion. If one hand is blocking the view of the cameras to the other hand, the other hand will stop being tracked. The lack of precision can also impact gesture recognition. Pinching the index finger and thumb together is a standard interaction in Oculus hand tracking, often being a "select/press" action. However, if the hand is positioned in such a way that either the index finger or thumb are occluded, these gestures may not be recognized. Further, any use of hand tracking necessitates that the user's hands are in front of the cameras at all times, as Jerald notes, this can result in physical fatigue [16].

While Leap Motion has been defunct since 2019 [20], Oculus has continued to improve upon hand tracking and shipped a major revision of API in May 2022. Due to the recency of the update, its technology has not been implemented in this thesis. However, details and potential implications of the revision will be discussed in section 8.3.1.

Perhaps the biggest limitation of bare hand tracking is the lack of haptic feedback. A typical use-case for haptic feedback for hand-held controllers is to provide a rumble. Rumble can be efficient at providing feedback on certain actions. For example, providing a rumble upon input accept informs the user that their input has been submitted without any audiovisual feedback.

Further, the lack of tactility is another issue. UI that is affixed in floating space does not provide any tactile feeling or resistance and precise inputs may be problematic. Research has been done on how the lack of feedback could be mitigated. Experiments with mixed reality applications where the UI is projected onto surface and controlled by bare hands have been conducted. Among these, it was found that projecting such UI onto a surface allows for a sense of tactility that is not provided by UI "floating" in space, overall improving usability [38]. This effect can be comparable to using a touch screen.

3.3 Text inputs in virtual reality

Text inputting in VR has been marked as a challenge in prior research [17]. The lack of visibility for users in VR is a chief consideration when tackling this challenge, as it means more familiar forms of input such as the physical keyboard are not ideal. Different typing methods for VR have been created and evaluated. According to Kipp [18], the most standard methods of text input are:

- Head pointing
- Controller pointing[5]
- Controller tapping
- Freehand technique
- Pad-based cursor control[6]

These input methods are visualized in figure 5.

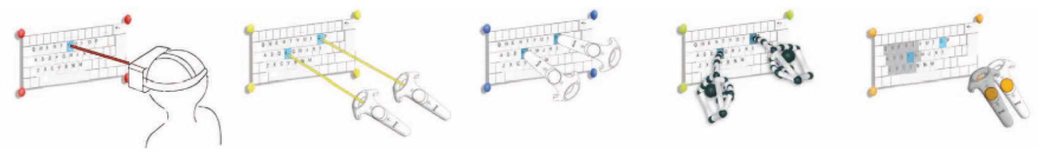


Figure 5: Different types of text input methods [17], (p. 6)

As part of his paper, Kipp also presents quantitative data for the performance of these different methods. The conclusion was that both the words per minute and user experience was best in the controller pointing method, as can be seen in figure 6.

ViRMA can be said to use a variation of controller tapping, i.e. the controller has to physically move over the input buttons in the space. While this input method has the second highest WPM of all the methods, it is fourth in user experience, middling in frustration and subjective performance, and the most physically demanding of all the tested input methods [18]. However, even the most efficient VR typing method is severely outperformed by physical keyboard typing. A comprehensive 2018 study of keyboard typing speed and accuracy found that even the slowest keyboard typists had a significantly higher WPM than controller pointing with comparable error rates (46.5 WPM) [5].

[5]This is the standard for both Oculus and SteamVR interfaces

[6]Standard for PSVR where users often have a pad-based controller such as the DualShock

Method	WPM	Error Rate (%) (corrected)	User Experience	Physical Demand	Frustration	Subjective Performance
Head Pointing (HP)	III: $10.20_{\pm 1.91}$	II: $1.15_{\pm 2.14}$	II: $0.67_{\pm 1.16}$	IV: $41.43_{\pm 0.00}$	II: $41.07_{\pm 0.00}$	II: $32.86_{\pm 0.00}$
Controller Pointing (CP)	I: 15.44 ± 2.68	I: $0.97_{\pm 1.19}$	I: $1.17_{\pm 0.78}$	III: $37.86_{\pm 27.82}$	I: $28.10_{\pm 24.42}$	I: $28.33_{\pm 20.88}$
Controller Tapping (CT)	II: $12.69_{\pm 2.27}$	III: $1.94_{\pm 2.22}$	IV: $0.56_{\pm 1.17}$	VI: $51.90_{\pm 26.05}$	III: $42.86_{\pm 32.12}$	III: $38.81_{\pm 21.67}$
Freehand (FH)	IV: $9.77_{\pm 4.78}$	VI: $7.57_{\pm 7.69}$	III: $0.55_{\pm 1.18}$	V: $46.43_{\pm 26.28}$	IV: $50.71_{\pm 27.85}$	IV: $40.00_{\pm 29.41}$
Discrete Cursor (DC)	VI: $5.31_{\pm 1.05}$	V: $2.79_{\pm 3.02}$	VI: $-0.40_{\pm 0.88}$	II: $30.71_{\pm 24.15}$	VI: $62.14_{\pm 23.64}$	VI: $54.05_{\pm 28.31}$
Continuous Cursor (CC)	V: $8.35_{\pm 1.58}$	IV: $2.15_{\pm 2.93}$	V: $-0.07_{\pm 0.92}$	I: $28.81_{\pm 21.62}$	V: $57.38_{\pm 28.44}$	V: $47.86_{\pm 24.22}$

Figure 6: Usability statistics for VR text input methods [17]

Based on the prior research, two relevant conclusions on VR text inputs can be made.

- Choice of VR text input seeking to maximize WPM and user satisfaction should prioritize controller pointing
- A significant majority of people have significantly higher typing speeds on traditional keyboard inputs compared to VR inputs

3.3.1 VR keyboard integration

Among the classifications for VR input devices described in 3.1, Jerald defines world-grounded input devices. This set contains input methods that are "designed to be constrained or fixed in the real world and are most often used to interact with desktop systems", such as keyboards and mice. The primary challenge of integrating world-grounded input devices is their lack of visibility in the VR environment. Due to this, Jerald noted that using such devices in VR is not optimal, with the possible exception of augmented reality scenarios where the user can see their environment [16] (p. 311).

In recent years, integrating physical keyboards into VR in a manner similar to hand-held tracked controllers has been introduced in consumer grade VR. In 2017, Logitech announced the BRIDGE Kit, which contained hardware and software that would allow developers to add a virtual representation of a physical keyboard to a VR environment.[7][32] Following the release of HMDs with front-facing cameras such as the Quest 2, integrations of keyboards and VR have continued. Productivity software such as ImmersedVR allowed for projecting keyboards in VR environment in the Quest 2 version [36], and Oculus itself introduced an official trackable keyboard feature to their Quest 2 OS [23]. The feature renders a 3D model of a connected Bluetooth keyboard in the virtual environment, maintaining the keyboard's position in the environment relative to the position of the HMD. When the user moves their hands over the keys to type, a partial passthrough effect is enabled such that the user's hands become visible (see figure 7 for illustration). Similar to the tracked Oculus Touch controllers, the 3D render reacts to button presses made on the actual keyboard.

Rendering and tracking a physical keyboard's model and position in VR could alleviate the visibility issues mentioned in Jerald's work. Prior research indicates that rendering and tracking could have a positive impact on users' typing efficiency in VR [15]. The rising prominence of tracked keyboards in VR also carries potential implications for Jerald's classifications examined in section

[7] Though after the initial supply of hardware was exhausted they have not been renewed and the project's Git repository has not been updated since 2018

3.1, as these devices are neither fully world-grounded devices (due to being tracked) nor tracked hand-held controllers. Perhaps it could be described as another class of VR devices, tracked non-hand-held controllers, or a super-set of both called "tracked hand controllers". This implication will not be discussed and for this thesis, tracked keyboards are going to be considered a set of tracked hand-held controllers due to their high degree of similarity.

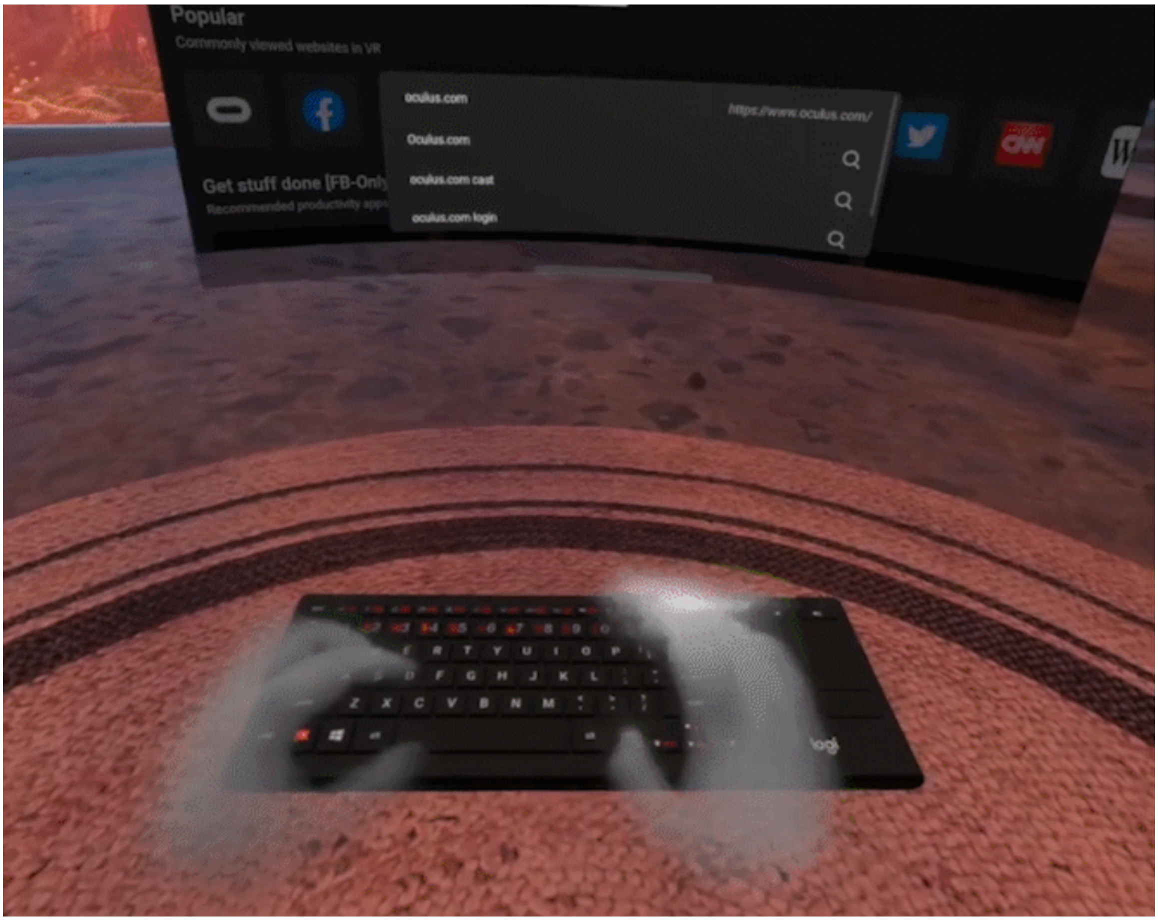


Figure 7: Tracked Logitech K830 keyboard with passthrough hands in Oculus Home environment [23]

3.4 Motion sickness and virtual environments

Extensive research has been done on the exact causes of motion sickness in VR, but these will not be examined in this section.[8] This section will primarily relate to virtual environments in VR and how these can impact usability. Important aspects of designing virtual environments in relation to ViRMA include motion sickness and physical space awareness.

Virtual environments in this context can be understood as the 3D representations of a space that a user may find themselves in while using a given VR application. How a virtual environment is constructed and used can have a massive impact on the usability of a VR software. Moreso than

[8] Jerald's VR book includes several long chapters detailing the cause and potential remedies to VR motion sickness

most other media, VR carries the potential of inflicting adverse health effects on users. For example, a convincing virtual environment that places users at a high altitude may induce a natural fear of falling. Scene motion, either when the scene or the user within it moves, is also a major cause of motion sickness [16]. Established research on the topic indicates how motion sickness in VR could be mitigated, along with what could cause it. A 2022 survey on cybersickness[9] attempts to establish both causes and remedies to the issue. Certain variables are dependent on the users, such as age and gender, which are correlated with higher rates of motion sickness. Elements that designers can use to mitigate sickness include controls, lowering field of view, and reducing graphic realism [25].

Another concern with virtual environments is naturally the lack of visibility. Many VR applications require or encourage moving around in the virtual environment. Physical movement in discorded physical and virtual environments can result in collisions. For example, a person moving their body physically forward to move in a wide open VR space only to hit a physical wall. Standard solutions for movement in VR applications is to rely on the thumbsticks on the controllers in a manner similar to console video games.

To mitigate these issues, consumer grade VR headsets usually allow users to define "play-space boundaries". As HMDs allow for tracking a user's position in physical space (which is then translated to the virtual space), the HMD can also detect when a user may be exceeding a pre-defined virtual boundary. Usually, the closer the user moves to this boundary, the more explicitly clear it becomes. An illustration of this can be seen in figure 8.

Figure 8: VR boundary illustration [29]

Certain modern VR HMDs also include onboard IR cameras to handle boundary tracking. These cameras can also be used in application purposes, essentially turning the VR HMD into a mixed reality machine. This can alleviate most adverse health effects related to immersive VR environ-

[9]Cybersickness specifically refers to motion sickness induced by HMD VR content

ments, as the user has a general match between visual stimuli and physical movement and space [16]. However, implementing passthrough naturally comes at the cost of immersion.

4 ViRMA

This section will describe the initial implementation of ViRMA.

4.1 API and Hardware

ViRMA is developed in the Unity Engine and all the scripts are written in C#. ViRMA makes use of the SteamVR API.[10] For ViRMA, the SteamVR API is primarily used to handle controller inputs, action sets, and mapping the HMD and controllers to game objects in the Unity scene. ViRMA was developed exclusively with the Valve Index HMD and controllers. However, due to generic controls,[11] and the platform-agnostic feature of SteamVR, porting the controls and functionality over to other HMDs is very simple. Therefore the initial change to use Quest 2 headset with ViRMA required only minor setup changes in the SteamVR application.

The project consists of several main parts contained within the Assets folder. Resources folder contains the folders Prefabs, Materials, Images and Texture used in the Scene. All of the scripts which define the software's custom behaviour are found in the Scripts folder. To easily find custom scripts a naming convention has been followed, with each custom file starting with `ViRMA_`. The scripts make use of UnityEngine package and each class derives from `MonoBehaviour`, which is a base class for every Unity script and provides the basic functions such as `Start()` or `Update()`. All SteamVR related files are contained within own folder in the Assets folder.

4.1.1 Project Scene

Project Scene is made up of a few main elements which comprise the ViRMA software. `Player` at the top of the hierarchy window in the inspector contains all the elements and references to SteamVR (including eg. VRCamera, Left Hand and RightHand) and references the HMD.
`MainMenu`, `VisualizationController`, `DimensionExplorer`, `DimExKeyboard` and `Timeline` are the main elements of ViRMA UI, which allow for user interaction with the software. What these are and how they are used to interact with the software is described in section 4.2. Each of these game objects has a `ViRMA_` script attached to them that defines the element behaviour and user interactions. `Timeline` is an empty game object that gets populated with images when a cell is submitted for query. Similarly `DimensionExplorer` contains nothing until a user makes a successful query search after which it is populated with `DimExplorerButtons`. Each button corresponds to an individual tag and is labeled as such. The same applies to `VisualizationController` which, when populated, displays the spatial axes with tag-names labels, and image `Cells` .
There are two mains scripts which coordinate much of the software behaviour and assign the actions defined by the other scripts:

- `ViRMA_APIController` - contains methods for communication with the server, controls and processes the requests and the data retrieved from the server and the database
- `ViRMA_GlobalsAndActions` - main script which accesses almost all the other scripts and maps SteamVR actions with the right controller input and software functions. It is attached to the `Player` object in the scene.

[10] Also known as OpenVR [24]
[11] Generic in this sense meaning that the controls do not use Valve Index exclusive features such as full finger-tracking

4.2 Using the Software

This section describes the possible user interactions with ViRMA, including how to use the controllers. ViRMA only functions using hand-held tracked controllers and features a few interaction modes for facilitating the exploration of large multimedia data-sets. ViRMA makes use of an "immersive environment", presenting a completely blank, white space for the user to interact with the application in. ViRMA features no manipulation of user positioning. Instead, the application can be used while entirely stationary, and any movement would be by the user physically moving their body (and not the application moving the virtual user entity).

4.2.1 Main Menu and tag search

By pressing the "B" face button on any of the two controllers (assuming Index Knuckles), the user can summon (or dismiss, if already summoned) the `MainMenu` (figure 9). The `MainMenu` is positioned directly in front of the camera, but it can be repositioned to be anywhere in the virtual space. This can be done by holding the reposition button located in the lower right corner of the menu.

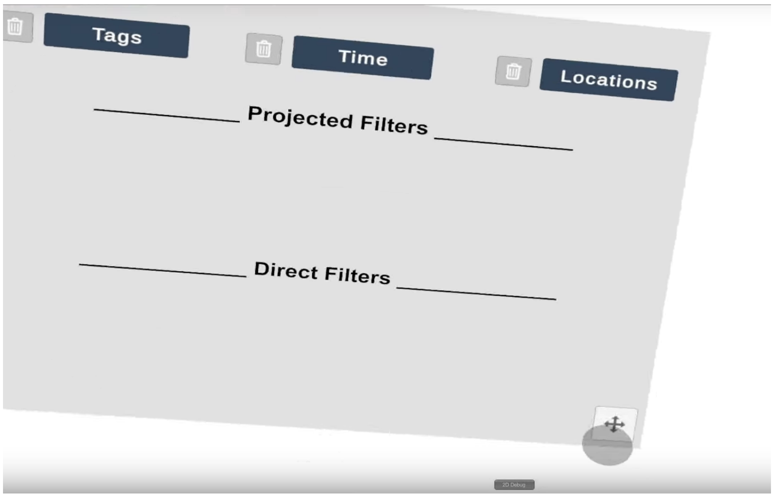


Figure 9: ViRMA `MainMenu`

Attached to each controller is a *hover-point*, as can be seen in figure 10. The hover-point is a small ball that acts as a pointer for the UI elements in ViRMA. Colliding a hover-point with any interactable button in ViRMA will highlight the button and a press of the interaction button ("A") will activate it. For example, opening the `Time` tags menu is done by colliding the hover-point with the button, and pressing the "A" button on the controller while the button is highlighted.

From the `MainMenu` there are three options to choose from: `Tags`, `Time`, `Locations`. `Tags` encompasses all types of tags and tag-sets as described in section 2.3.2 and 2.3.3 respectively. `Time` allows for applying temporal limits to the generated output, e.g. specifying only photos of a certain month or day. `Locations` is not functional in the initial iteration of ViRMA, but it would use geographical metadata to allow users to specify locations for the output.

Upon pressing the `Tags` button the `MainMenu` is replaced with a virtual keyboard - `DimExKeyboard`

Figure 10: ViRMA Controllers (Valve Index Knuckles)

(figure 11). In order to search, the user must hover over individual letters and press the "A" button each time. Once the search string has been entered, the user can initiate the search by pressing the green check mark. If the action is successful, the user is then presented with a `DimensionExplorer` where results of the search query are presented. Searches that fail to connect with the server or return no result give no feedback beyond the debug log in Unity.

Figure 11: ViRMA Keyboard (`DimExKeyboard`)

4.2.2 Dimension Explorer and filter projection

The `DimensionExplorer` is structured as can be seen in figure 12. Any tag which starts with the search string is present in the results. As an example, if the entered search query is *cat*, the results will contain words such as *cattle*, *cathedral*, *caterpillar*. The order of the buttons is determined in an ascending order by tag-id.

Each element of the `DimensionExplorer` consists of up to 3 columns of buttons, each containing a single string. The white central button of each element is the word which contains the search string, henceforth referred to as the main tag. Buttons in the same column are siblings of the main

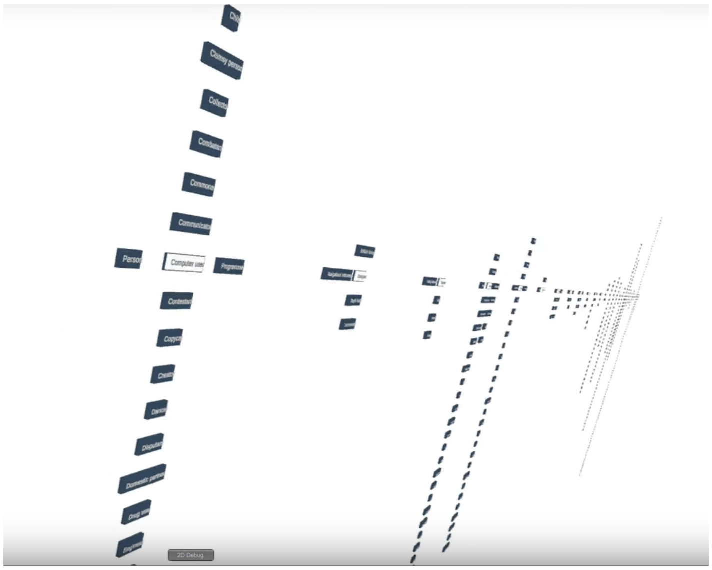


Figure 12: ViRMA `DimensionExplorer`

tag. Left side of each element contains a single button which is the parent of the main tag. The right column presents the children tags of the main tag. If absent, this indicates that the main tag is located lowest in its hierarchy and therefore contains no further children.
Users can interact with the `DimensionExplorer` in several ways. The index trigger button combined with horizontal controller movement allows for moving the entire structure on the x-axis to browse through the results. Moving the hover-point over the columns containing siblings or children puts all buttons from a given column in a highlighted state, in which the column can be scrolled up and down. To interact with a specific button the pointer has to collide with it. When a single button is highlighted in white, it is possible to click it and open the `ContextMenu` (figure 13). The `ContextMenu` is used to apply the projected and direct filters. Project to axis buttons have colours corresponding to the axes on which the tags can be projected.

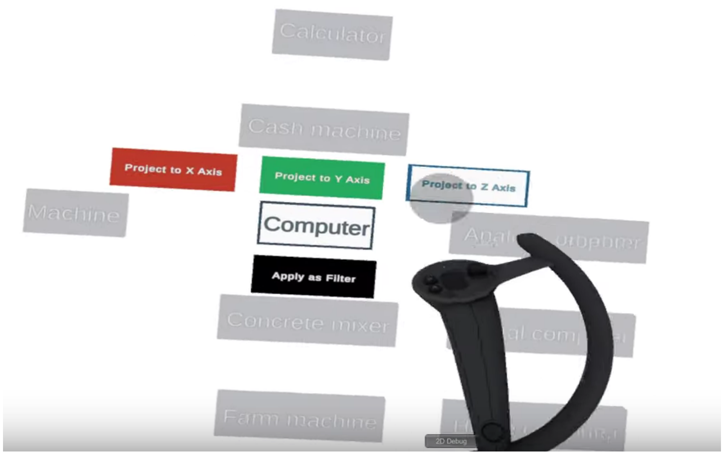
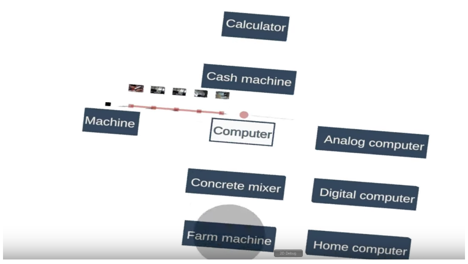


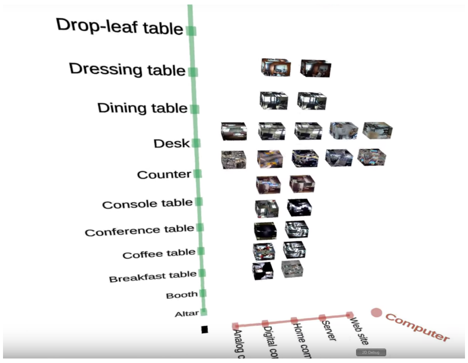
Figure 13: ViRMA Tag Projection

4.2.3 Visualization Controller and image browsing

Applying filters which contain child tags on any of the three axes results in the `VisualizationController` appearing behind the `DimensionExplorer` (see figure 14a). This element can be interacted with only after the `DimensionExplorer` and `DimExKeyboard` are hidden, which can be done by pressing the red button in the lower left corner of the `DimExKeyboard`. The `VisualizationController` with tags projected on the X and Y axis can be seen in figure 14b.



(a) ViRMA `VisualizationController` behind the `DimensionExplorer`



(b) ViRMA `VisualizationController` with filters projected to axis X and Y

Figure 14: ViRMA `VisualizationController`

The values on the axes are the children of the projected tag. Users can interact with these labels by colliding them with the hover-point to highlight the values and pressing the "A" button to confirm the choice. Axis labels which have a number in parenthesis when hovered over can be pressed with the "A" button to drill down. The number indicates how many children the given tag has. Similarly, pressing the parent tag at the end of the axis allows to roll up. The position and yaw can be manipulated with the controllers. Holding the index trigger on the right controller allows for pulling the `VisualizationController` closer or moving it away. Using the left controller it is possible to manipulate its yaw. Within the `VisualizationController`, any `Cell` that contains corresponding axis values is generated.[12] Colliding the hover-point with any cell in the `VisualizationController` highlights it and displays the corresponding axes values. It is important to note that these interactions are only possible once the `DimensionExplorer` and `DimExKeyboard` are hidden. The components of the `VisualizationController` can be highlighted with these UI elements still present, but it is not possible to manipulate them or do any of the other actions like drilling down, moving the `VisualizationController` or submitting `Cells` for `Timeline`.

When a single `Cell` is pressed, the `VisualizationController` is automatically hidden and the `Timeline` appears. The `Timeline` contains images from the hypercube which are displayed chronologically in a single row. This can be seen in figure 15. Index trigger can be used again to drag the `Timeline` on the X axis. Depending on the size of the query, the images may be separated into "chunks" which can be navigated between by pressing the "Next" button at the end. At any time the user can click "B" button to hide the `Timeline` and go back to the `VisualizationController`.

[12]Using M^3 terminology, the cells are hypercubes

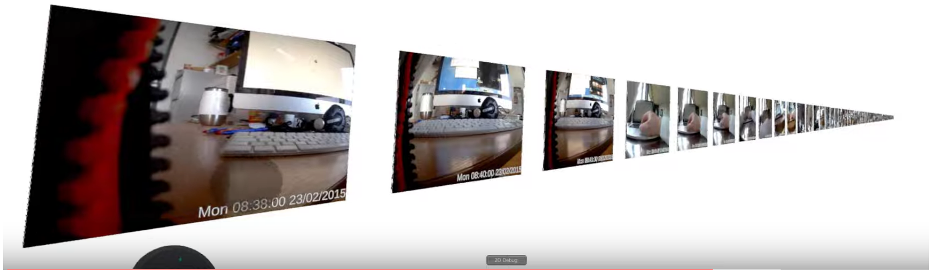


Figure 15: ViRMA `Timeline`

The `MainMenu` contains a button with a trash can icon which can be used to clear all the applied filters and projected tags. The projected tags and applied direct filters can be seen in the `MainMenu` as shown in figure 16. Users can scroll through these by putting the controller close and using the controller thumbstick.

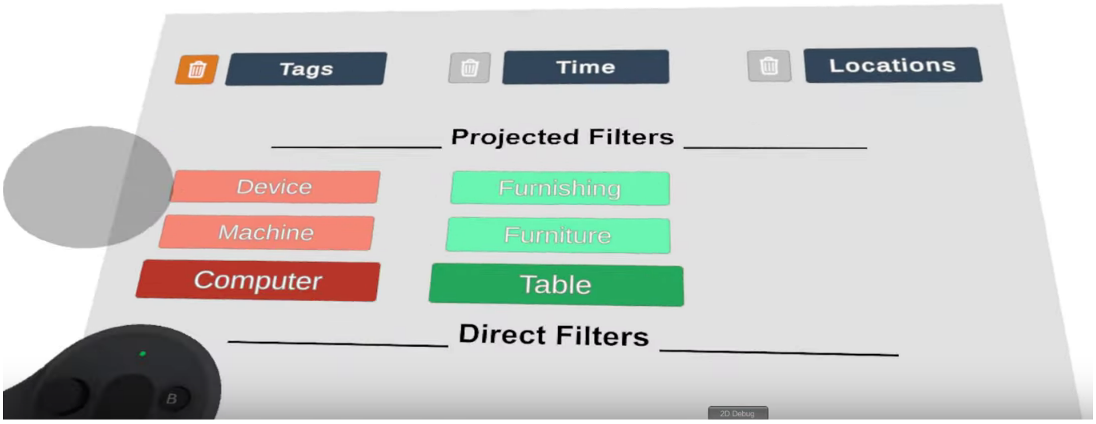


Figure 16: ViRMA Active Filters in `MainMenu`

4.3 Data-set

During the progress of this thesis, ViRMA has only been used and developed with one data-set. Choice of data-set is highly relevant for a database analysis and exploration software. The data-set available is from the LifeLog Search Challenge (LSC) 2021. This data-set comprises around 40GB of image files from a "lifelogger". In this instance, the lifelogger wore a wearable camera that automatically took pictures throughout the day. The data of the set were collected over periods in 2015, 2016, and 2018 [13]. The metadata (here mainly *tags* used for querying) associated with these images has been generated by an AI program.

5 Analysis and implementation of potential usability improvements

In this section we will evaluate potential usability improvements to ViRMA, and describe their implementation. Before doing this, we will describe some technical aspects of the Oculus Quest 2 as the implementation changes have been designed with this HMD, including using some of its proprietary technology.

5.1 Hardware: Oculus Quest 2

As described in section 4.1, ViRMA has been developed with the Valve Index. All the work for this thesis (except some usability testing) has been conducted on the Oculus Quest 2. This section will briefly explore the Oculus Quest 2 headset from a technical standpoint. Attention is drawn to the fact that Oculus is currently the former name of the popular VR HMD producer. In January 2022, as part of a rebrand, Oculus' parent company Facebook changed their name to Meta. This name change included Oculus which is now also known as Meta [31]. However, due to the name recognition of Oculus, and to avoid potential confusion between Meta the HMD producer and Meta the social media conglomerate, this paper will retain the Oculus naming.[13]

The Oculus Quest 2 is the second iteration of the Quest series of headsets, released in October 2020. The Quest series of headsets are "standalone", meaning that they can function without being attached to an external device such as a PC or gaming console. Facilitating this ability is the on-board OS which is Android-based, and the hardware which is running on the Qualcomm Snapdragon XR2 mobile CPU, 6GB memory. The headset features four IR cameras that allow for inside-out tracking, meaning that the headset can continuously infer the user's position in a room, warning users if they are about to overstep the boundaries of their play-space. The inside-out tracking also allows for tracking of the wireless controllers without external tracking monitors such as the ones used by the Valve Index.

Oculus offers an integration package to the Unity game engine which has been used for this project. This integration contains game objects, prefabs, and APIs for developers to use. The API package includes facilitating passthrough, tracked keyboard functionality, and bare hand tracking.

While the Quest 2 is capable of running Android applications non-tethered, all development of ViRMA for this paper has been conducted with the project build settings set to Windows PC. There are multiple reasons for this, primarily relating to how the application communicates with the server and data storage. In section 8.3.2, we describe potential future endeavors to create a Quest 2 native ViRMA port.

5.2 Identifying potential usability improvements

This section will identify potential software features/implementations that could serve to improve the usability of the ViRMA software. We will describe potential solutions and justify their effectiveness based on aforementioned literature.

[13] It is also noted that the Oculus naming currently persists internally in numerous methods, such as all of the formal documentation and plug-ins for the Unity Engine.

Before doing so, it is advisable to clarify one's general design philosophy or direction. How usability concerns are tackled is ultimately guided by what goal one has for the software. For these purposes, we clarify that our vision of a hypothetical finished version of ViRMA is for it to be used in a multitude of professional contexts. For example, medical, law-enforcement or content analysis/exploration. Due to this, we envision future users would engage with ViRMA in a professional setting, potentially around colleagues, near or at a desktop PC. It is important to be aware of this expectation as it guides some of the coming decisions. Therefore, it is also understood that a vision for ViRMA that presupposes, for example, use-cases in a private environment with no people nearby could prioritize different design decisions.

5.2.1 Environment

As described in the existing usability tests conducted in *Layered VR-User Interfaces - Onboarding of users in ViRMA* [10], certain participants expressed dissatisfaction with the environment in ViRMA. ViRMA features a solid white environment absent of any objects. It could be fair to conclude that while a blank environment could provide extensive immersion (as the only visual stimuli present are the controllers and all the elements that allow for querying data), it can have negative usability effects.

First, a bright white environment could have negative effects on comfort during prolonged sessions. While this has not been discovered in usability test, prolonged visual exposure to bright screens have been correlated with numerous health issues such as eye strain, fatigue, and headaches. This is especially relevant for VR where users' field of view is engulfed by a screen. It can be concluded that the environment of ViRMA may elicit negative subjective experience and potential health risks.

The health risks could be mitigated in a number of ways. Popular methods of reducing potential adverse health effects on eyes from screens include changing the hue of a regular blue-light screen to be warmer, reflecting the setting sun. However, the efficacy of such methods, for example on sleep patterns, are still contested [8].

Further, in a professional environment with colleagues present and moving about, being immersed in VR could present spatial challenges. Being unaware of one's physical surroundings is potentially dangerous. In a practical setting, this limits both the range of movement the VR user can achieve, and it limits colleagues and similar from entering the boundary of the VR user.

To mitigate both health issues and potential boundary conflicts, we propose implementing passthrough technology to allow users to see their environment while using the application. The Oculus Passthrough API allows for tinting of the environment such that it could emphasize any choice of hue that most users find visually pleasing. While the cameras' output is admittedly not high resolution nor color-correct, they are clear enough that we estimate a large majority of users can move through any lit physical space without collision due to being in an immersive VR environment.
Using passthrough may also decrease potential VR fatigue that can be felt when immersed in VR for longer periods of time. Testing for long term exposure is out of the scope of this thesis, however.

5.2.2 Hand- and text input

While the existing usability test of ViRMA did not reveal major issues with the controllers or virtual keyboards, current literature suggests that ViRMA's "touch-and-click" solution is suboptimal [18]. This method of text inputting described in section 3.3.1, while having a good WPM output, ranked very low in terms of user experience and had high physical demand compared to other text input methods. A re-evaluation of the text input method could be beneficial for ViRMA. To this, we propose two solutions of which we will select one and justify our choice.

The first solution is to implement what is, in VR, the most subjectively satisfying text input method, controller pointing, as described in section 3.3. This would be a "safe" option given the research and the fact that the largest major HMD OS' (Oculus and SteamVR) already implement this method for typing.

The other solution would be to implement physical keyboard integration. As described in section 3.3, even slow keyboard typists significantly outperform the average controller pointer typist in terms of WPM. As noted by Jerald (mentioned in 3.3.1), implementing world-grounded devices in VR is usually not advisable due to the lack of visibility. However, implementing Oculus' keyboard API with an Apple Magic keyboard,[14] the keyboard could be tracked and rendered in the environment in a manner similar to the tracked hand-held controllers.

In extension of this, we also propose integrating hand tracking technology to allow using the software entirely without controllers. Provided the tracking is acceptable, we hypothesize that users not familiar with VR or gaming controllers will find it more intuitive to use their hands. Further, this will alleviate potential encumbrance felt by placing and picking up controllers to interact with the keyboard. This lack of encumbrance coupled with passthrough could also result in a more pleasant experience, as users could easily hydrate or snack during longer sessions without removing the HMD.

To conclude on these sections, we hypothesize the following would increase ViRMA's usability:

- Passthrough functionality to view one's environment during use
- Tracked keyboard for increased typing efficiency
- Hand tracking controls for increased flexibility

In the following sections we will describe the implementations of these. Each of these changes carry implications for ViRMA as a software that necessitate some degree of change or tuning to other parts of the software. These will be discussed as the implementation process is described.

5.3 Implementation

5.3.1 Porting to Oculus VR

Before implementing any of the elements described in the previous section, it was necessary to introduce some basic changes. To begin with, the `Player` object in the project scene has been replaced by `OVRCameraRig` prefab which can be found in the Oculus Integration SDK. This element provides

[14] Currently, the Oculus API only supports tracking Apple Magic Keyboards and the Logitech K830

an object that represents the Oculus tracking space, and contains left and right anchors for both controllers and eyes as well as the center eye anchor (main Unity camera). In a later stage this has been replaced by `OculusInteractionSampleRig`, which comprises `OVRCameraRig` and `InputOVR`, the latter of which was necessary to introduce hand tracking and constituted the foundation for Oculus control system in ViRMA. `ViRMA_GlobalsAndActions` script was then attached to the `OVR-CameraRig`.

A significant amount of time has been spent on adjusting ViRMA to work solely with Oculus input controls, detaching it from SteamVR. It was a necessary foundation to transfer the controls and have a working base software version before beginning any of the intended implementation changes. This proved to be challenging and time consuming mainly due to the following reasons:

- Lacking or outdated documentation on Oculus input
- Lacking knowledge on SteamVR implementation in ViRMA

At the beginning we attempted using Unity Engine XR Input, where the `XR.InputDevices` class enables accessing any input devices (such as controllers) connected to the system. This method of assigning controls has been abandoned later in favor of the `OVRInput` class. At first, custom laser pointers were implemented. These served as a way to interact with UI buttons without the need to move the hand with controller physically closer to the objects, acting as controller pointers described in section 3.3. However, when implementing hand tracking, several issues were encountered with this implementation such as flickering and collision issues. Therefore the pointer was discarded and controllers with poke interactors, which are part of the aforementioned `InputOVR`, were used instead. These were also more in line with the original ViRMA controllers. A major difference between the SteamVR implementation and Oculus VR implementation of the controllers is that the hover-points on the Oculus VR controllers in some situations act the same as an index finger on the bare hand controllers. This results in most menu interactions with the Touch controllers being purely by colliding the hover-point with buttons rather than colliding and clicking. The reason for this is provided in section 5.3.4, where we describe the menu implementation. Touch controller interactions of colliding and clicking with the elements of the `VisualizationController` remain the same. See figure 10 for original controllers and figure 17 for Oculus Touch controllers with poke interactors.

(a) View on poke interactor

(b) General view

Figure 17: Oculus Touch controllers

To handle object grabbing `OVRGrabbable` and `OVRGrabber` were used on objects and controllers respectively. This allowed for grabbing any ViRMA menu or `VisualizationController` by pressing the index trigger. However at a later stage when hand tracking was implemented several issues related to grabbing appeared. Therefore we decided to implement our own script `ViRMA_NaiveGrabbable`, where we defined very basic interactions supporting grabbing, moving and rotating objects. Using `OVRInput` provided the functions to retrieve a real time controller position and rotation necessary for the grabbable to function. This naive implementation eliminated any noise from `OVRGrabbable` which was previously interfering with hand tracking.

The `ViRMA_NaiveGrabbable`, true to its name, is a highly simple implementation of grabbing objects in VR. When the index trigger is pressed while controller is found within an activation distance, the object can be grabbed. We implemented a global variable that checks if any element is being grabbed before allowing for this action. This ensures that only one object can be grabbed at a time. The object's position is then being set to the position of the controller, allowing the user to place it anywhere in the virtual space. In addition this script implements a special case for rotating and scaling the `VisualizationController` while grabbing. Based on the thumbstick movement, which can be retrieved in form of a float with the use of `OVRInput` class, the `VisualizationController` rotation and scale are transformed accordingly. One problem with this naive implementation of grabbable is that a grabbed object "jumps", anchoring on controller in one specific place, as opposed to sticking with the specific part that was grabbed. For example, grabbing the `VisualizationController` always results in the controller being found in the center of this element, despite grabbing eg. the corner of it.

Due to time constraint moving the `Timeline` as in the original ViRMA was not implemented. To mitigate the issue introduced by this limitation, we introduced a lower maximum number of images displayed at once, where only 5 images are shown such that a user can comfortably view them without the need for scrolling. The `Timeline` in this version still has a button present that allows to load the next bunch of images.

5.3.2 Passthrough

The simplest of our proposed usability improvements to implement was the passthrough functionality. This required attaching the `OVRPassthroughLayer` script available in the Oculus SDK to `OVRCameraRig`, adjusting the opacity and edge color settings accordingly and enabling passthrough in the Oculus app. Upon application launch the user is then presented with environment similar to this shown in figure 18.

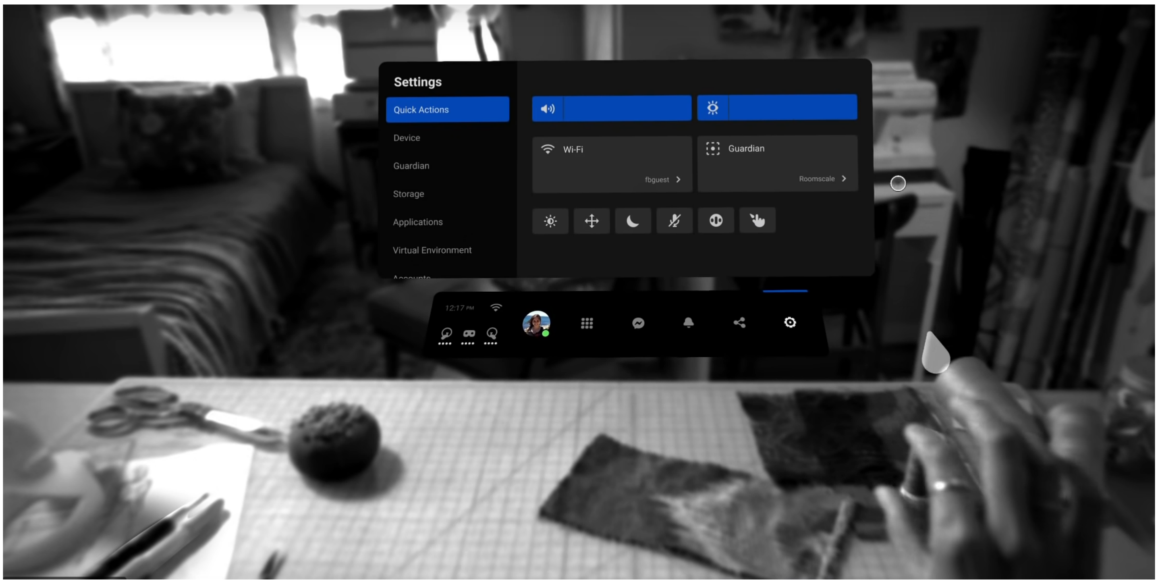


Figure 18: Room as seen with passthrough layer [3]

5.3.3 Tracked Keyboard

Before gaining access to Apple Magic keyboard we implemented processing keyboard input in the software. This was done by using `Input` class in the `ViRMA_Keyboard` script. Having the physical keyboard as an input source eliminated the need for a virtual keyboard, therefore we modified `DimExKeyboard` UI element to simply contain a search bar and submit button, as can be seen in figure 19. While we succeeded in connecting the Apple Magic Keyboard to Oculus where the 3D rendered version of the keyboard was clearly visible, we encountered issues related to passing keyboard input to ViRMA. Due to time constraints we did not manage to find a solution to this problem. The intended idea of having a 3D rendered version of a physical keyboard to ensure a good typing experience had to be discarded. Nevertheless we decided to leave the physical keyboard input in ViRMA, as with the passthrough layer the surroundings, including the keyboard, are visible to the user. This introduced issue of its own, as the environment rendered with passthrough is not completely clear and the keys appear slightly blurry. Being aware of this shortcoming we expected to receive substantial feedback related to typing during the testing phase. We assumed a stance that it will be still beneficial to test another input method in ViRMA. For a picture of the intended implementation of a 3D rendered tracked keyboard we refer the reader back to figure 7.

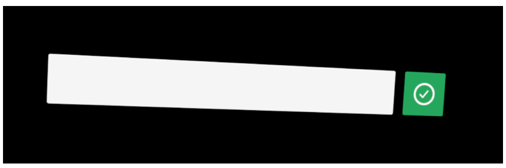


Figure 19: `SearchBar`

5.3.4 Hand Tracking

In an attempt to ensure a seamless software use, especially when transitioning between typing on keyboard and interacting with virtual UI, we decided to integrate hand tracking technology to ViRMA. As mentioned in section 5.3.1, introducing to the scene Oculus Interaction Sample Rig with `InputOVR` was the first step to implement hand tracking, as `InputOVR` provides the prefabs and settings for Hand controllers. The `OVRInput` class allows for easy action mapping for both Touch and Hand controllers. Some of the controller buttons have their corresponding action recognized in hand gestures. For example, having `OVRInput.Button.One` will trigger an action both when the "A" button is clicked on the right controller, or if a pinch (using index finger and thumb) with the right hand is detected. The same is true for `OVRInput.Button.Three`, which responds to the "X" button on the left controller or pinching with the left hand. Due to this, several interactions with bare hands were immediately possible without the need for any explicit changes. Oculus' API only allows for one type of controller to be active at a time - Hands or Touch controllers. When Touch controllers are put down and remain static for a while, and the user holds their hands in front of the headset camera, hand tracking is activated. Due to this it is also possible to add additional check and allow for action to happen only if the specific type of controller is active. However, this process may present some challenges as switching between input methods is not instantaneous (sometimes hands may need to be kept in front of the camera for 5-10 seconds). Hand models as seen in ViRMA are presented in figure 20.

(a) Hand tracking (b) Pinch gesture (c) Pointing with index finger

Figure 20: Hand tracking in ViRMA

To better accommodate the software to hand tracking functionality we made additional design changes related to ViRMA UI elements. The most prominent alteration was eliminating the `DimensionExplorer` and replacing it with a much smaller, scrollable menu that has a fixed size. `SearchResults` menu in the Oculus implementation is the equivalent of `DimensionExplorer` in SteamVR implementation. Since it serves the same purpose, we sometimes refer to this menu as `DimensionExplorer`. We deemed the `SearchResults` menu more usable with hands than the `DimensionExplorer` that can potentially span a massive area both horizontally and vertically. The newly introduced menu consists of several elements and incorporates the functionality of `DimensionExplorer`, `ContextMenu` (used for projecting filters) and clearing the projecting filters. We chose to combine these features, as they are closely related. The idea behind this was to limit the physical exertion required to operate the `DimensionExplorer`. Placing them in a small scrollable flat menu, we sought to evoke a sense of familiarity with the user, reminding them of touch screens they may have used on their mobile device. New `SearchResults` menu can be seen in figure 21. Due to this menu being designed specifically with hand interactions in mind, instead of having regular buttons, "toggle" buttons were implemented. These can be interacted with easily by poking the index finger or poking the hover-points on Touch controllers. The `SearchResults` menu and the interactions with it are described further in detail in section 5.3.7.

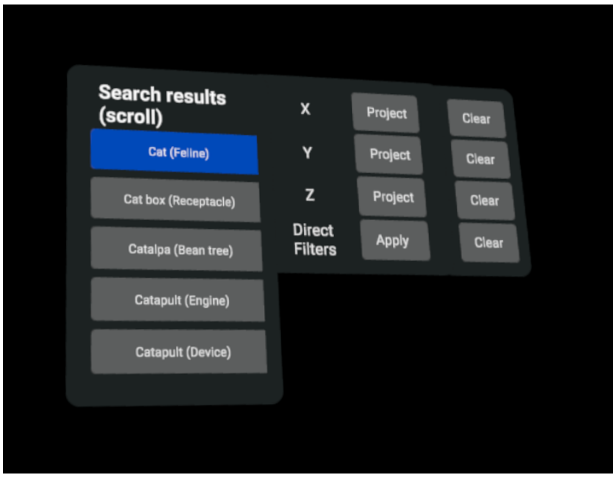


Figure 21: Populated `SearchResults` menu with active Axis Projection and Filter clearing menu

5.3.5 Other changes

Among other changes introduced to ViRMA UI was removing the `MainMenu`. Clearing active filters can now be done with the `SearchResults` menu and the `Search Bar` where users can type their queries is appeared using the keyboard. The `Time` filter has not been implemented in this iteration, mainly due to time constraint and low perceived additional benefit to the usability testing. Hence with the `Time` and `Locations` buttons not functional, the `MainMenu` was deemed unnecessary extra step in this software iteration.

Due to failing to implement trackable keyboard, and hand tracking being "hidden" we decided to create the `InformationMenu` that is visible upon launch. This menu was meant to give the user the introduction necessary to be aware that they can use both their hands, Oculus controllers and keyboard to interact with the UI, particularly because the keyboard is necessary to proceed through the software. We recognized that this was not an ideal solution, but we lacked time and resources to create more intuitive methods of communicating keyboard and hand tracking to the users.

Each `ViRMA_` script had to be edited to accommodate the change from `Player` to `OVRCameraRig`. Furthermore, these scripts responsible for actions/interactions with their corresponding game objects were adapted to respond to Hand and Touch controllers. This was done using the aforementioned `OVRInput` class. The newly introduced UI elements have their own scripts defining the behaviour.

5.3.6 Interaction

This section outlines in detail all interactions possible in the second ViRMA iterations, with the changes implemented as described in the previous section. To sum up, this iteration gives the user possibility to use:

- Oculus Touch controllers (see figure 17)
- Hands (limited interactions)
- Keyboard (necessary to make a query)

Due to technical limitations, the screenshots of interactions have black background as seen in the Unity editor and don't show the passthrough environment that is active at all times and visible to the user. To see example of passthrough we refer to figure 18.

5.3.7 Initializing query search

Upon the launch of the software the user finds themselves seeing the physical surroundings with a passthrough layer. From the start there are two elements present: the `InformationMenu` and empty `SearchResults` menu. The `InformationMenu` contains all the basic information about interacting with the software, such as how to use the keyboard, Touch controllers and hands. This menu can be toggled on or off at any time by pressing the "X" button on the left controller. In order to make a query, the user is required to press Enter key on the keyboard. This action results in a `Search Bar` appearing in front of the camera. `SearchBar` can be hidden or shown at any time by pressing Esc or Enter keys respectively. To enter a desired tag the user simply needs to type the word using the physical keyboard. While the `SearchBar` is present the Enter key serves submitting the typed word for query. This action populates the `SearchResults` menu with buttons containing resulting tags along with a parent tag written in parenthesis as can be seen in figure 22.

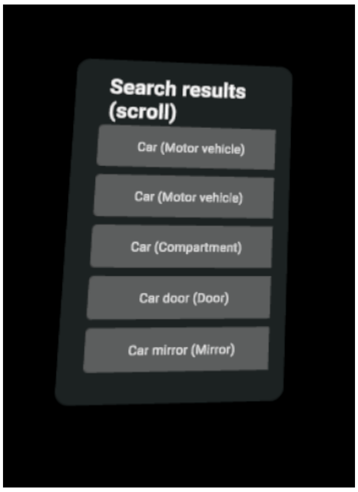


Figure 22: Populated `SearchResults` menu

5.3.8 Interacting with search results and projecting filters

Users can interact with the `SearchResults` menu in two ways:

- with active Controllers: by tapping the buttons with the poke interactor at the end of controller
- with active Hands: by tapping the buttons with index finger

These interactions toggle the pressed button on, changing the colour to blue. On the right side of the `SearchResults` a second element of the menu appears (see figure 21). This part allows the user to project the chosen tag to the X, Y, Z axis or apply it as a direct filter, as well as clear any individual axis from projected filter and remove the applied direct filters. Once the filters are projected or applied, the same `VisualizationController` element as described in section 4.2.3 is loaded (figure 14b). When using controllers, interacting with cells, rolling up and drilling down is done in the same way as in the original implementation - by hovering the poke interactor over the elements and pressing the "A" button. The user can grab and move the `VisualizationController` by pressing and holding the right hand index trigger. In addition, while grabbing is taking place it is possible to rotate it by pushing the thumbstick left or right, or scale up and down by pushing the thumbstick correspondingly. Pressing a `Cell` loads images in the `Timeline`. Pressing "B" on the right touch controller hides the `Timeline` and shows `VisualizationController`.

The condensed dimension explorer was generally designed with hand tracking in mind. Due to this, we sought to mitigate the lack of tactile feedback bare hand tracking on menus is usually associated with (described in section 3.2.1). To do this, we allowed the user to grab the menu with the Oculus Touch controller and easily align it with any flat surface, such as a table, which would then serve as a tactile boundary for the menu. This alignment can be difficult to explain via text, so we have created figure 23 to illustrate it. Figure 23 shows two Touch controllers on a flat surface, with the blue plane being an illustration of the menu attached to the controller.

The use-case for this would be a person sitting at their desk, grabbing the menu and placing it on the table next to their keyboard after which they could use both the keyboard and dimension explorer in tandem.

Figure 23: Menu placement on flat surface illustration (original image from [9]; blue surface is an original addition)

Hand interaction with the `VisualizationController` is limited to viewing `Cell` axis labels values and highlighting labels with right hand index finger and drilling down, rolling up and submitting a `Cell` by pinching with right hand. Hiding the `Timeline` and going back to viewing `VisualizationController` is not supported with hand tracking due to limited input and using the "B" button on a controller is necessary for this interaction.

5.3.9 Limitations and impacts

Before proceeding to the usability test, we want to summarize and clarify the limitations encountered during development along with our hypothesis for their impact. The major limitations include:

- Lack of keyboard tracking
- Lack of hand grabbing
- Controller hover-to-activate
- Naive grabbing
- Lack of Time filter
- Lack of image context

We expect the lack of keyboard tracking to have a significant impact on user satisfaction due to low visibility in the passthrough environment.

While we think hand tracking can offer an intuitive control method, the lack of feature parity with controllers presents a challenge. However, we expect that users may set up their "workspace" by positioning menus with controllers whereafter they will rely on their hands.

Controllers hover-to-activate is an unsatisfactory implementation of controller-menu-interaction. It breaches VR conventions by not requiring an explicit click to activate. We expect participants who are familiar with VR to find it especially dissatisfying.

Naive grabbing is generally usable but presents a tremendous challenge on larger queries. If participants want to move down a long axis they must do so physically (which may be possible with the passthrough functionality). However, it is far from ideal as it requires physical movement.

As part of limiting the scope, time filtering and image contexts were not incorporated to the Quest 2 implementation of ViRMA. By image context we describe viewing the contextual timeline of the selected image. While the lack of these features is not ideal and future development efforts should seek to reimplement them, they are not essential to testing the usability of the added features, which is ultimately the goal of the study.

6 Usability testing

To evaluate the usability of the changes to ViRMA developed during this project, two sets of usability tests have been conducted. The first round of usability tests were conducted on the SteamVR implementation of ViRMA. The second round of tests were conducted on the Quest 2 iteration of ViRMA which is described in section 5.3. This section will detail and examine the structure of the usability tests, and evaluate their proceedings and findings.

This is done with the goal of testing the hypothesis that the changes to ViRMA improve its usability.

6.1 Methodological considerations

Most of the prescriptions on usability testing referred to in this thesis will be from the book *User Interface Design: A Software Engineering Perspective*, by Søren Lauesen (2005). For evaluation of usability tests we will rely on Nielsen's Ten Heuristics (1994), developed from Nielsen & Molich's heuristics [19]. For this paper, we are specifically using the 2020 iteration of the heuristics as defined by the Nielsen Norman Group [22].

Due to the very frequent use of Nielsen & Molich's heuristics, their explanation will be very brief. The ten heuristics are broad guidelines that can be used to inform design decisions to optimize for usability or to analyze usability issues. The heuristics follow a specific order and are as such:

- Visibility of system status
- Match between the system and real world
- User control and freedom
- Consistency and standards
- Error prevention
- Recognition rather than recall
- Flexibility and efficiency of use
- Aesthetic and minimalist design
- Help users recognize, diagnose, and recover from errors
- Help and documentation

Brief descriptions of each can be found with the usability issues schema in appendix section F. For the purposes of this thesis, we are going to apply the heuristics to identified usability issues, to gather an overview of which specific heuristics the software is lacking on. Heuristic evaluation of usability is identified as a qualitative approach to usability testing, rather than a strictly quantitative approach that may prioritize quantifying data such as time taken to complete tasks and clicks per task [28]. We have chosen this approach due to the experimental nature of ViRMA where every feature is subject to change, and efficiency optimization is not necessarily the top priority. Usability evaluations of ViRMA in latter stages where the direction and development of the software has furthered could benefit from a quantitative approach.

6.2 Structure

6.2.1 Tasks

Tasks describe a set of objectives test users should complete during a usability test. Tasks are defined based on what the user should accomplish with the software. For example, a task for medical prescription software could ask the test user to prescribe a specific type of medication for a person. There are guidelines to how tasks should be defined. Among these are:

- Tasks must be enclosed - completion of each task "deserves a coffee break"
- Small related tasks are described as sessions without breaks
- Hide who does what, use imperative language
- Do not program: "`if x then y else z`"

However, ViRMA is complicated for creation of tasks. First, tasks should be defined in conjunction with the demographic of whoever is going to use the software. As described earlier, we envision ViRMA being used in a multitude of professional contexts. However, for the purposes of these tests we are limited by the available data-set.

Another concern in defining tasks for ViRMA is the fact that the software currently has an exploration-based purpose. Tasks could be argued to be inherently contradictory to exploration. By formulating a task a user should strive to complete, their exploration would be limited to an explicit destination. This could perhaps consciously or subconsciously result in the test user limiting their exploration to paths that would have them complete their tasks, rather than allow for spontaneity.

To account for the exploration-based purpose of the software, tasks have not been rigidly defined. The only explicit task users are met with is to generate a query (generally on multiple axes). What the query should contain is going to vary between participants and should take their person into account. For example, a participant who expresses an interest in football could be prompted to generate queries on sports, and explore instances of the data-set containing football. This approach is not perfect. Some participants' hobbies or interests may not be represented in the software and a query generation from this would not yield anything. In cases like these, the facilitator will engage in conversation with the participant in order to brainstorm further query ideas. This is done to keep some element of exploration and spontaneity. Usually, the facilitator will ask "We have this insight into a person's life, what could be interesting to know about them?", appealing to a sense of curiosity.

In cases where both approaches fail, the facilitator has generic queries ready. These would only be used if no "spontaneous" queries generate results. However, at this point it would be prepositionally true that the exploration aspect would be severely limited therefore, while not ideal, we consider it beneficial.

Due to time querying not being implemented in the Quest 2 version (as described in the previous section), there were no tasks related to adding time filters or viewing contexts of images.

6.2.2 Test users

Test users in this scenario describe participants of a usability test. The ideal test users are almost always people who represent the intended demographic of the software. Currently it is anyone who has an interest in browsing and searching through large heterogeneous multimedia data-sets. The data-set could be used to define a target demographic, e.g. a data-set containing images of illnesses or medical procedures would lend itself well to testing with medical scientists. Unfortunately, the data-set available (described in section 4.3) is quite limited in this regard. The primary target demographic of interest in this would be the lifelogger them-self or subjects of interest in their life.

Lack of a clearly defined target demographic does not make choosing suitable test users impossible. However, it does create the need for extra stipulations in the usability test. Chief among these can be thus formulated as: *Given the lack of a target demographic for the software in question, it is understood that the usability test is based on a hypothetical target demographic which the test users fit into.* With this stipulation follows the understanding that any future usability tests regarding ViRMA *not* following these hypothetical user groups could yield different results.[15] To mitigate this, we will carefully justify our hypothetical user group.

In choosing participants, there are two major challenges:

1. Understanding the software, data, and data model
2. Testing in the VR medium

In attempting to account for each, we have made the choice to give each participant an introductory presentation on the data model and data-set. We consider this information the absolute minimum information a user must contain to use (or even comprehend) ViRMA. The presentation would be shown prior to the test and would focus on brevity and relatability. Having a full understanding of the data model is not necessary, but basic concepts such as hierarchical ordering of tags should be understood. The full slides can be found in the appendix B.

It was decided that there would be no selection bias in regards to familiarity with VR. This means that participants' familiarity level with VR could range from non-existent to highly familiar. While this approach broadens the potential "range" of findings, it was considered beneficial as it could allow for interesting observations such as observing potential differences in expectations between experienced VR users and beginners. Further, ViRMA has no prescriptions on the VR familiarity level of future users. Therefore, usability experiences from all ranges are valid for consideration.

The usability tests have been split into two sub-sets, which will be described in sections 6.5. Both iteration 1 and 2 of the tests were conducted with 8 participants each, totaling 16. All the tests were facilitated at the IT University of Copenhagen's campus, in private rooms. Measures were taken to benefit participant comfort, such as the presence of requested beverages and snacks.

[15] Future adaptations of ViRMA for specific purposes should have a new set of usability tests conducted with whatever target demographic the adaptation carries

6.3 Testing structure

Figure 24 shows the structure of the usability test. In the following sub-sections, each process will be described in detail. This section details the structure of the first iteration of tests.

Process	**Minutes**
Survey	5
Introductory presentation	5
Test A	15
Post-test interview (A)	5
Break (optional)	5-10
Test B	15
Post-test interview (B)	5
Semi-structured interview	10
Concluding survey	5
Total	**65-75**

Figure 24: Usability test structure

6.3.1 Introductory conversation

The introductory conversation is a brief informal dialogue between the facilitator and participant. The participant has been made aware that the testing has begun, and they will be posed some questions. The conversation can arguably be considered a semi-structured interview, but in the interest of participant comfort and natural dialogue, it is not presented as such. Typical questions in this conversation include:

- What is your name?
- What is your age?
- How would you describe your familiarity level with VR?
- What kind of hobbies/interests do you have?

The facilitator often asks follow-up questions. For example, specific questions about VR experiences or brief conversation about the choice of hobby. As described in section 6.2.1, questions about hobbies/interests are to gauge potential queries that will be created in the actual test.

6.3.2 Introduction – slides

This part of the tests cover the introduction to data model mentioned in section 6.2.2. Participants are shown a brief PowerPoint presentation that covers the basic idea of the data model. This includes descriptions of the data-set, hierarchical tags, tag-sets, multi-dimensional browsing, and drilling-down and rolling-up. Great care was taken to present these topics in a friendly and approachable manner. The full presentation slides can be found in appendix B.

6.3.3 Test A & B

As the purpose of the tests is to conduct a comparative usability study, each participant will be subjected to both versions of ViRMA. These tests are labeled A and B. As the two implementations are highly similar, participants may experience fewer usability issues in the B test due to a familiarity bias. This can lead to compromised data validity.

To mitigate this, half of the subjects' A-test will be the initial SteamVR implementation, whereafter their B-test is the Quest 2 version. For the other half of the subjects, this order is reversed. Due to this choice, a hypothesis could be formulated that each implementation's A-test findings would highlight more issues than its B-test findings. As part of the findings section, we will attempt to falsify this hypothesis. Regardless of the result, it remains an aside that due to being accounted for via split-testing, this should not compromise the validity of the study.

6.3.4 Post-test interviews A & B

Immediately after each test, a brief unstructured interview/conversation takes place. The subject is asked to discuss their perception of the software they used. The subject is free to go in any direction they feel is most important (e.g. focusing on controls). The interviewer is allowed to pose questions or prompts if the subject finds it difficult to relate their experience, or if the subject undertook a particularly interesting action during the test. The interviewer's questions are going to vary wildly and will depend on aspects noticed during the tests.

Both post-test interviews are exclusively about the implementation just tested. It may prove challenging for subjects in the B-phase to relate purely to the B test, avoiding statements such as "I think X feature was annoying, the other implementation did that better". This is the type of implicit bias described in section 6.3.3 that is attempted to be accounted for using the split-test structure. It is in this case the interviewer has the responsibility to steer subjects to only talking about the B test to the extent possible.

It is important that these conversations happen immediately after each test, as the experience is still at the forefront of the subject's mind. Eliminating the post-test interviews and instead enlarging the scope of the final interview could skew data towards the B-test, given that it would be the most recent for the user.

6.3.5 (Optional) break

After the post-test interview A, subjects are invited to take a small break. This is to allow the subjects to rest, hydrate, and clear their heads. Crucially, this break is set between the A and B test to allow the subject to reflect on their wellbeing, as some people may experience adverse effects from VR use as briefly described in section 3.4. The break is marked optional as subjects who do not wish to take it may proceed to the next step.

6.3.6 Semi-structured interview

The final interview is designed to allow subjects to compare the two tests and judge their experience based on these. The interview is semi-structured, allowing for getting concrete answers to specific questions while allowing the subjects to elaborate and explore tangents if deemed relevant. The interviewer has a few fixed topics that must be approached at some point during the interview. Time spent on the fixed topic is proportional to what the participant or interviewer found most interesting. For example, the topic of controls would likely be longer for a participant who expressed frustration at the keyboard integration. The fixed topics cover the biggest changes between the ViRMA implementations and can be seen below.

Fixed topics

- Passthrough compared to virtual environment
- Control differences (keyboard, controller, hand tracking)
- Menu/presentation differences

6.4 Concluding survey

Finally, before the participants leave they are presented with a very brief survey asking them to rank their experience with the different iterations on a digital survey. This step primarily serves as a quantitative representation of the subjects' experiences, as extensive qualitative data will already have been captured by the think-aloud tests and the interviews. The questions and responses to this survey can be seen in appendix C, and D respectively.

6.5 Test iteration 1 and 2

This section will very briefly outline the differences in testing between iteration 1 and 2. Table 1 shows a superficial comparison on the main changes between the two testing iterations.

Element	Iteration 1	Iteration 2
Software walk-through	×	✓
Index HMD	✓	×
Quest 2 HMD	✓	✓
(Q2) Emphasis on hand tracking	×	✓

Table 1: Superficial differences between test iterations

The motivation for two iterations and the differences between them came during the first iteration of tests. While beneficial, it was found that a lack of guidance would result in users only using the tracked hand-held controllers during the Quest 2 implementation test, despite hand tracking being described in the `InformationMenu`. This resulted in a lack of data on bare hands. As this was one of the primary subjects of interest, it was deemed necessary to adjust and conduct a second iteration of tests, should a conclusion on bare hand controllers be justified.

Bare hand grabbing (moving objects) was disabled in iteration 1 due to a perceived excess of false input readings during development. This lack of feature parity resulted in users preferring controllers as they could do more. To mitigate this and gain further data on hand tracking as input, bare hand grabbing was enabled in iteration 2 despite potential false input readings.

After iteration 1, the software defect that caused the Quest 2 HMD to be incompatible with the SteamVR implementation was rectified and this allowed for testing on only one HMD. For reasons of practicality, this was pursued. Since iteration 1 was done with both HMDs in their respective versions and this data had been captured, usability defects potentially caused by using the SteamVR software with a Quest 2 could be easily controlled for.

Crucially, iteration 2 featured a very brief (2 minutes) walkthrough of each software before participants were to use it. This was determined beneficial as the usability of the "onboarding" process of both implementations had been established at this point. The walkthrough was short enough to give participants an idea of the software, but not so thorough that they would be able to recall the process.

Finally, reminiscent of iterative design methodology, the Quest 2 version had changes to minor aspects to rectify usability issues that had a high or ubiquitous frequency. All of these changes were very small in scale, not affecting any of the primary subjects of interest (bare hand tracking, passthrough, keyboard inputs). These changes are briefly described in the following section.

We consider these changes defensible as both iterations of the tests had enough participants to justify some level of conclusiveness. The second iteration allowed for adjusting and isolating some variables to gain a better understanding of them - especially bare hand tracking.

6.5.1 Quest 2 changes for second iteration

Due to the second interaction containing a brief software walk-through, the `InformationMenu` was deemed obsolete. The `SearchResults` menu has also been hidden on software launch and it appears first when a user presses Enter to open the `SearchBar` and initiate search. This was done to mitigate confusion some participants encountered when seeing the `SearchResults` menu prior to searching. In addition, the buttons used for projecting and applying filters were colour coded as can be seen in figure 25.

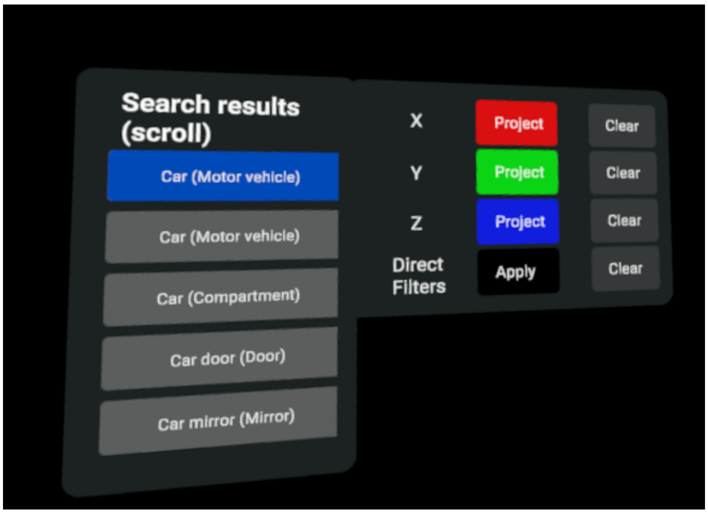


Figure 25: `SearchResults` menu with colour coded buttons

The most important addition was enabling left-hand grabbing of objects with the `ViRMA_Naive-Grabbable` script attached. This allowed users to move all menus as well as the `Visualization-Controller` around.

6.6 Hardware

For test iteration 1 the participants would be using both the Index and Quest 2 VR HMDs. This was for practical reasons as the SteamVR version of ViRMA did not reliably work with the Quest 2 headset at the time. This split of headsets can both have positive and negative effects. The potential negative is that the subject of interest is the software, but with the hardware acting as an independent variable this may skew the data.
An argument in favor could be that both versions of the software were designed with separate HMDs, and testing on these HMDs would lend a higher level of consistency with the designers' intentions. At any rate, observable disparities due to HMDs will attempt to be accounted for.

For the second iteration of tests, the software defect had been rectified and all tests were running on the Quest 2 for practical reasons.

6.7 Set-up or test errors

Overall, most of the tests went by without any issues. However, there were two instances that bear mentioning.

The first user test conducted took place morning of the 2nd of May 2022. Just prior to the interview, a server fire occurred in the basement of the testing location necessitating evacuation. This resulted in the first test not testing the SteamVR implementation. For the sake of consistency, data from this participant was discarded and a replacement participant was found.

The second instance of a process issue was with participant 1.06 (notes from this test can be found in appendix F). Part of the usability tests entail the participants not being aware of the intentions behind each implementation, including which implementation was designed by us. This is done to maintain impartiality. Usually, after the tests had concluded and the survey filled out, the conversation opened up to allow for facilitators and participants to discuss intentions, implementations and so forth. With participant 1.06, the facilitator accidentally revealed the Quest 2 version was designed by us and some of the intentions behind the choices. It was determined that the participant being aware of this could influence their answers on the survey, therefore it was skipped.

However, 1.06's answers on the survey could still impact averages and distribution. Therefore, based on the conversation and notes we have decided to answer the survey based on 1.06's feedback. In the interest of full disclosure, the rationale behind each answer can be found with 1.06's testing notes. This is not ideal, but with the presented mitigation strategy we do not consider it a foundational issue either.

Another potential issue with the set-up is the lack of a chaperone system for the first iteration of tests on the Index hardware.[16] The facilitator would warn participants if they were about to

[16] SteamVR chaperone is a boundary system that warns users when exceeding a designated play-space

collide with anything, but it could be the case that the chaperone system would have introduced a greater sense of safety for participants, thereby skewing the data. The second iteration of tests on the SteamVR version of ViRMA was done on the Quest 2 with its boundary system enabled, so we can use that data as a control.

7 Usability test analysis

This section will contain an analysis of the usability test proceedings for the first set of tests. Each of the potential usability concerns found in the tests have been classified on the following parameters:

- Functionality
- Description
- Heuristic (Nielsen & Molich)
- Participants

"Functionality" describes which functionality in the application the usability defect is concerned with. This classification has been made to allow for direct comparison between the two implementations' usability defects in the functionalities that have been altered. Examples of functionality could be "keyboard" or "controllers".

"Description" very briefly describes the usability concern. These can range from system defects to design choices that impede subjective enjoyment.

The basis of the heuristic evaluation is the Ten Usability Heuristics as defined by Jakob Nielsen and Rolf Molich in 1990 [19]. Classification based on these heuristics are beneficial to describe the type of usability defect. This can be used to establish a pattern of defect types and potentially inform further development.

"Participants" describe the specific participants that encountered or expressed this usability concern. They are ordered chronologically, following which iteration they used. `1.XX` and `2.XX` indicate participants from the first and second testing iterations respectively. This specification exists due to the set-up differences in the two iterations, as the changes may alter some variables and therefore perceived usability concerns. For example, due to the higher focus on hand tracking in iteration 2, it is expected to see more participants from the second iteration detect usability concerns related to those.

Including a severity rating was considered. It was deemed unnecessary as the following detailed analysis is going to describe the impacts of the usability defects to a greater level of detail.

The usability defects may cover both defects participants expressed themselves, and defects participants may have been unaware of, but were noted by facilitators. The full table can be found in the appendix section F. This also includes a brief definition of the ten heuristics.

Note that the findings will also refer to expressions made by participants regarding positive experiences, such as finding a specific implementation intuitive to use. However, there will not be a formal schema or tabulation of positive attitudes (except for survey results). Further, the appendix includes the notes taken for the session for each section of the usability tests, serving as justification for both the usability concerns schema and positive expressions. These can be found in the appendix F starting with section 1.01.

The following sections will detail findings from the usability study divided into the three primary subjects of interest: virtual environment, hand-held controllers/bare hands, and keyboard implementation. The final section will examine the findings of the usability study on a holistic level.

7.1 Immersive environment / Passthrough environment

The usability tests found both benefits and issues to the changes made to the VR environment. Data from the concluding survey regarding whether subjects found the passthrough environment beneficial to their experience, a majority of subjects answered positively. The qualitative data indicates why. Most participants who enjoyed passthrough expressed that it gave a sense of security to be able to see their surroundings. It was also noted that participants in passthrough were less hesitant to move their physical body, for example to reach over and grab something, whereas some participants without passthrough expressed concern about physical movement.

It was mentioned that the Index iteration 1 tests did not have chaperone enabled, and that this could skew subjective enjoyment of the SteamVR version environment. It can be observed that a higher degree of participants in iteration 1 had a preference for the passthrough environment. This could be due to the lack of chaperone in iteration 1. However, despite iteration 2 featuring a lower number of positive responses towards passthrough, every positive response was marked *strongly agree*.[17] Therefore, in our opinion it is safe to conclude that the passthrough feature can be correlated with stronger subjective enjoyment (with higher rates to be determined). Figure 26 lists the environment usability defects for both implementations. From here it can also be seen that almost as many in iteration 2 expressed insecurity about boundary/physical space as participants in iteration 1.

STEAMVR IMPLEMENTATION			
GROUP	DESCRIPTION	HEURISTIC	PARTICIPANTS
ENVIRONMENT	Boundary / physical space insecurities	8	1.03; 1.05; 1.07; 1:08; 2.03; 2.05; 2.06
	Potentially vertigo-inducing (when looking down at a large queries or hierarchies)	2	1.02; 1:08; 2.03

(a) SteamVR environment usability issues

QUEST IMPLEMENTATION			
GROUP	DESCRIPTION	HEURISTIC	PARTICIPANTS
ENVIRONMENT	Pass-through environment can be distracting, obscure what is part of the software	8	1.03; 1.04; 1.05; 1.06; 1.07; 2.02; 2.04
	Difficult to parse software elements in non-ideal backgrounds (e.g., black text on black surface)	8	2.04

(b) Quest 2 environment usability issues

Figure 26: SteamVR and Quest 2 identified usability defects related to environment

However, not all participants preferred passthrough. Among those who did not, their main concern was that passthrough added a layer of clutter. They found that locating software elements

[17] Whereas iteration 1's positive responses were distributed between *agree* and *strongly agree*

(such as menus) was more difficult than the immersive VR environment. It has not been possible to establish a correlation between participant properties such as VR familiarity and enjoyment of the passthrough feature.

The chief criticism of the passthrough feature was the camera clarity. As described, the Quest 2's cameras have several limitations. Because the cameras prioritize a wide field of view to support hand and controller tracking, the passthrough has a fish-eye effect. Beyond that, the IR cameras do not capture color, and they are quite low resolution. These hardware limitations had a significant impact on the physical keyboard integration, as participants had difficulty discerning the keys. These usability defects are described in section 7.3.

Table 2 shows the scores on the concluding survey question related to the passthrough environment. Note that the results delineate between iteration 1 and 2 due to the changes in set-up structure.

	1	2	3	4	5
Iteration 1	0	2	0	3	3
Iteration 2	0	3	1	0	4
Total	0	5	1	3	7

Table 2: Q: Being able to see my physical environment was beneficial to my experience of the software

The tables represent the output of answers to how much participants agree with the statement shown in the table caption. 1 represents "Completely disagree" while 5 represents "Completely agree". For each table there will also be a corresponding chart, helpful to illustrate the distribution of answers.

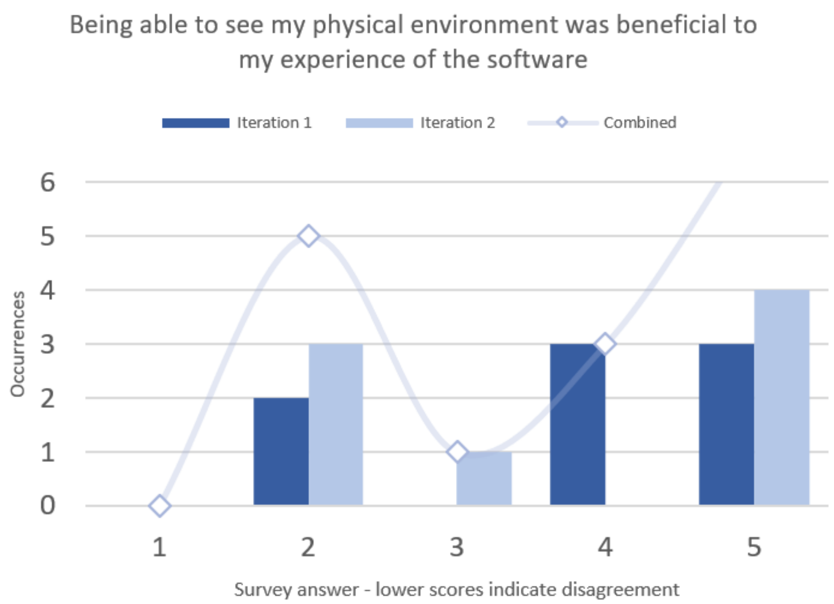


Figure 27: Passthrough environment survey question chart

From table 2 and figure 27, the claim that most participants preferred the passthrough environment can be supported. The slight discrepancy between iterations could be the result of the lack of chaperone, though this cannot be concluded safely. Further, participants in both iterations expressed an overall preference for the passthrough implementation.

7.2 Hand-held controllers / bare hands

In the first iteration of tests, the possibility of bare hand tracking was only communicated by the `InformationMenu`. Because most participants didn't fully read the it, none used hand tracking unless prompted.

Once using it, some participants (incl. 1.03, 1.04) returned to using controllers after a while for two primary reasons. First, they had already become familiar with controllers and to several of them, removing a familiar method of control was considered a challenge they were not ready to undertake as they didn't know the software well. Second, bare hand tracking did not allow for grabbing in the first iteration. Participants did not want to use bare hands due to a lack of feature parity.

In the second iteration of tests, grabbing had been re-introduced to the bare hands control scheme. For these tests, generally, the inverse of the above happened. Participants who started using the software with hand tracking and were prompted to use controllers frequently expressed a desire to revert to the prior. This is not necessarily a usability defect, but it is important to note the potential impact of familiarity bias on the results.

Due to technical limitations, hand grabbing only worked with the left hand and item selection with the right hand. All participants who used it noted it and expressed that it breached expectations (exception 2.07 who related it to operating machinery). Most had the opinion that left and right hands should carry the same affordances since there was no signifier that they were different.

In the implementation section, we described implementing a menu-grab functionality that allowed for grabbing the menu and aligning it with a flat surface easily, to allow for tactile feedback on touch inputs. No participant who used the grab functionality on the menu realized the intent behind it, instead finding the rotation and placement by the controller unintuitive and frustrating.[18] This resulted in a lack of any data on the tactile feedback aspect of the `MainMenu`, and frequent criticism of the grab-behavior.

For VR beginners, menu interaction with bare hands was much more efficient. Due to the lack of buttons (and therefore possible input), participants frequently intuited how to interact with the scrollable menu. Some participants noted that the search tags menu was reminiscent of a phone touch screen, matching between the system and the real world.

For technical reasons, it was not possible to implement clickable buttons in the Quest 2 iteration's UI while retaining intuitive hand tracking gestures. Instead, the small poke interactors located at the end of the controllers acted like index fingers, which could manipulate the UI by touch. Out of

[18] Placing the menu in front of one's field of view requires holding one's hand out similar to a point or handshake gesture, and then rotating one's hand 90 degrees such that the thumb faces downward.

all three control methods (SteamVR controllers, Quest 2 controllers, Quest 2 bare hands), this was most frequently associated with erroneous inputs (1.01, 1.03, 1.05, 2.03). Those who had used the SteamVR implementation before the Quest 2 would frequently assume that hover-and-click was the interaction method for the scrollable UI as it was in the SteamVR version. Conversely, users who started with the Quest 2 would frequently attempt to interact with the SteamVR implementation purely by hovering. Some experienced VR users preferred the SteamVR's hover-and-click method due to increased precision and higher consistency with their expectations (2.03, 2.04).

Figure 28 shows a cut-out of the usability schema related to controllers and bare hands for both versions. For the Quest 2 version, the bare hand inputs are associated with a high number of usability concerns, with three of them being ubiquitously experienced.

STEAMVR IMPLEMENTATION			
GROUP	DESCRIPTION	HEURISTIC	PARTICIPANTS
CONTROLLERS	Initial confusion about button mappings	10	ALL
	Persistent confusion regarding button mappings	6, 10	1.03; 1.04; 1.06; 1.07; 2.01; 2.02; 2.04
	Confusion about hover point / menu interactions	10	1.03; 1.04; 1.07; 2.01; 2.02; 2.04; 2.06

(a) SteamVR controller usability issues

QUEST 2 IMPLEMENTATION			
GROUP	DESCRIPTION	HEURISTIC	PARTICIPANTS
CONTROLLERS	Snap-grabbing very unintuitive and contrary to expectations	4	1.01; 1.04; 1.05; 2.04
	Sometimes controller face buttons are used to activate, other times hover-point collision activates (no communication)	4, 9	1.01; 1.03; 1.05; 2.03
	Unclear functionality (scale and rotate when grabbing items)	10	1.07; 2.04
BARE HANDS	Not clear that it can function as a control method	10	ALL
	Unintended input readings (general)	7	1.04; 2.01; 2.03; 2.04; 2.05; 2.06
	(Iteration 2) Arbitrary differences between hands; one can grab and the other select	4	ALL
	(Iteration 2) grabbing when the user didn't intend to	5	2.01; 2.04; 2:08
	Ambiguous/inconsistent controls (sometimes index finger activates, other times pinch does)	4, 5	ALL
	(Iteration 1) Missing features compared to controllers (grabbing)	4	1.03; 1.04

(b) Quest 2 controller and bare hand tracking usability issues

Figure 28: SteamVR and Quest 2 identified usability defects related to environment

Coupled with all of the above, part of the concluding survey asked respondents about which version they perceived had better control, as well as whether they preferred to use physical controllers over hand tracking. Below, we present two tables that show these.

	1	2	3	4	5
Iteration 1	0	3	0	0	5
Iteration 2	2	3	0	2	1
Total	2	6	0	2	6

Table 3: Q: I preferred the controls in the SteamVR implementation

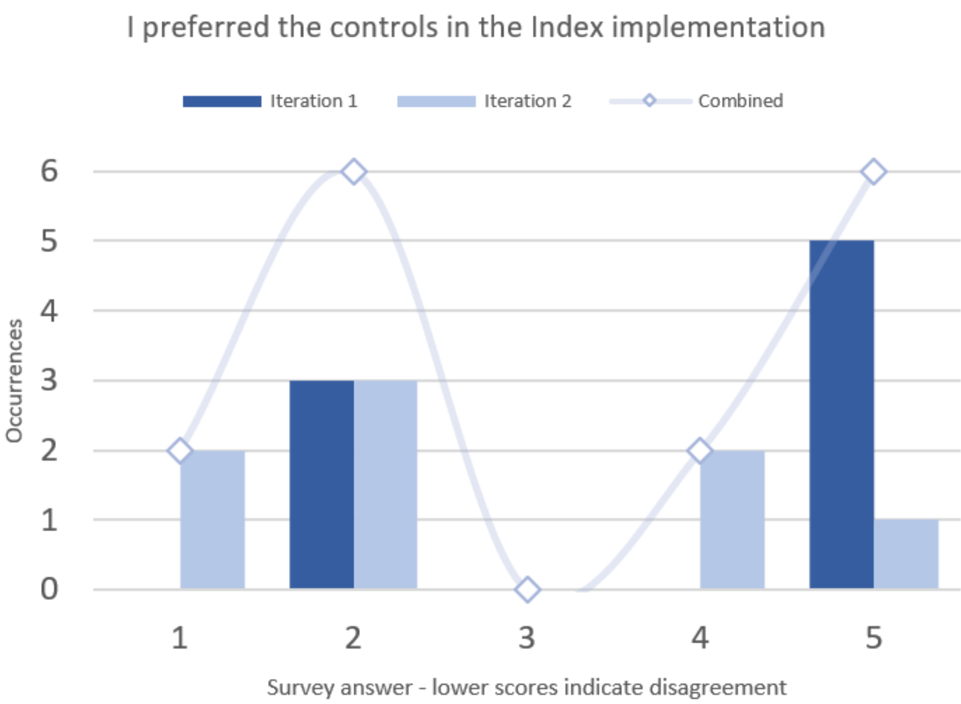


Figure 29: Controls survey question chart

	1	2	3	4	5
Iteration 1	0	1	4	2	1
Iteration 2	3	2	2	1	0
Total	3	3	6	3	1

Table 4: Q: I preferred using physical controllers to hand tracking

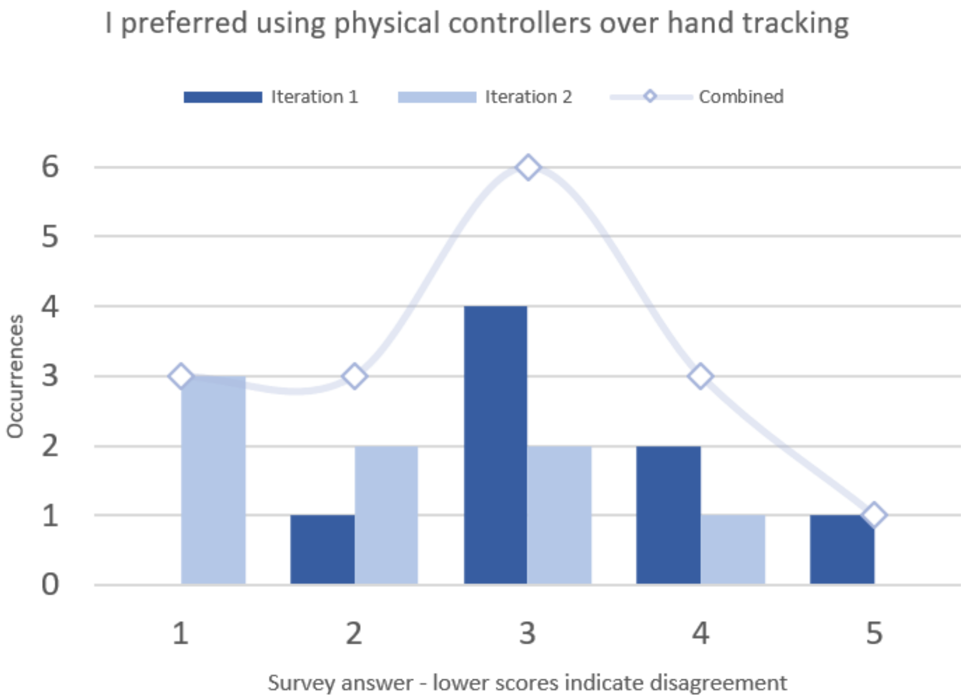


Figure 30: Hand tracking survey question chart

It can also be observed that, compared to all the other data from the concluding survey, the question regarding hand tracking in iteration 1 has disproportionately many neutral responses. Based on our interviews with people, many people who selected a neutral disposition towards hand tracking cited that while they would ideally prefer it over controllers, the implementation issues led to them preferring the more precise and efficient controllers.

7.3 Virtual / physical keyboards

The chief usability concern for participants regarding the physical keyboard was the lack of key visibility (experienced by all participants). As mentioned, the intention was for the keyboard implementation to integrate Oculus' tracked keyboard API. This would render the keyboard fully and mitigate this issue. While this analysis will provide useful data, it is undeniable that this shortcoming may have resulted in other usability issues not being detected.

As was predicted, all users expressed difficulty with distinguishing the keys on the keyboard. Once participants had "charted" the keyboard by locating the positions of a few keys, they would generally be successful in creating queries very fast. However, participants who did not have strong QWERTY layout familiarity had significant issues with using it (1.04, 1.05, 2.02).

The virtual keyboard in the SteamVR implementation received both criticisms and praise. The layout correlated with real world expectations and was received well. Some participants without VR experience expressed confusion at having to press the "A" button while hovering over to enter an input, as they expected it to be touch-based. Users who were familiar with VR would very quickly engage with the keyboard, although noting that it was more physically demanding than pointer-based inputting, which is what they are used to (consistent with Kipp's findings mentioned in section 3.3).

Generally, participants preferred the virtual keyboard's implementation, but several stated that they would prefer using a physical keyboard if visibility was not an issue (eg. 1.04, 1.06, 2.01, 2.02). Few participants stated that even with a tracked physical keyboard, they would prefer a virtual implementation (eg. 2.04). Their reasoning being that the virtual keyboard allows for greater freedom of placement (and in extension, movement).

STEAMVR IMPLEMENTATION			
GROUP	DESCRIPTION	HEURISTIC	PARTICIPANTS
KEYBOARD	No feedback on empty queries	9	ALL
	Inefficient typing compared to other methods.	4	1.01
	Keyboard spawn may not be noticed at first	1	1.03; 1:08

(a) SteamVR keyboard usability defects

QUEST 2 IMPLEMENTATION			
GROUP	DESCRIPTION	HEURISTIC	PARTICIPANTS
KEYBOARD	Low key visibility in pass-through camera	2, 4, 8	ALL
	Frustration at (subjective) slow input compared to normal conditions	2, 4	1.04; 2.02; 2.05; 2.06
	Necessitates keyboard layout memorization for efficient inputting	6	1.04; 1.05; 2.02
	Not clearly marked as part of the software (no render, blends in with other "real" objects)	2, 5, 4, 8	ALL
	Confusion about how to start (enter key)	10	ALL

(b) Quest 2 keyboard usability defects

Figure 31: SteamVR and Quest 2 keyboard-related identified usability issues

7.4 Holistic review of findings

This section will cover findings from the usability studies of both various specific (but minor) aspects, as well as a holistic review of the ViRMA's usability.[19] The benefit of the holistic approach is to contextualize potential usability concerns within the greater whole they exist in.

Both implementations received praise and criticisms from participants regarding the `DimensionBrowser`[20]. The SteamVR version was frequently criticized for being too expansive in size. Generally, participants would be interested in the tags that were close to their search term, and the SteamVR dimension browser separates these with some horizontal distance. This is done to allow showing parent, children, and sibling tags. Participants who were browsing specifically for their search terms would express annoyance at the physical requirement necessary to progress to the next search string match (1.02, 1.03). This was not aided by the search results not ordering alphabetically as many

[19] By holistic it is meant that the review does not delve into specific features, but gauges each implementation's usability as a whole

[20] In the Oculus implementation was also called `SearchResults` menu

expected (1.01, 1.03, 1.05; most 2nd iteration testers).

The Quest `DimensionExplorer` was significantly scaled down (to `SearchResults` menu) and features only tags that are associated with the search string. This boosted efficiency in generating queries, but some participants (1.03,1.06, 2.02, 2.03) noted that the compressed nature of the menu resulted in a lack of context.

When asked about a preference, most participants expressed that they would prefer a combination of the two implementations' solutions. One that emphasizes both the Index's contextual approach while having the Quest 2's smaller scale and lesser physical demand. This may have potential implications for future development of SMA VR applications (including ViRMA). These implications will be discussed in section 8.3.

As described, iteration 1 featured testing on both the Index HMD and the Quest 2 HMD whereas iteration 2 featured only the Quest 2. While this variable could impact the data, it was not possible to determine any specific correlation between HMD and experience of the SteamVR implementation of ViRMA. Participants' comments were chiefly related directly to the software.

While the concluding survey was very simple to not burden participants who already donated time and effort, it can be used to make some broader quantitative conclusions on the software. On a holistic level, slightly more participants expressed a favor for the Quest 2 implementation rather than the SteamVR. The survey also asked participants to (very briefly) justify this answer. Most of these answers, based on the interviews conducted, were in line with our expectations. A significant portion of participants prioritized the bare hand controllers (citing them as intuitive) and passthrough environment (citing it as "safe") as their chief motivations for their reasoning. While this generally fits with the motivation for their implementation, the demographic of our participants being largely unfamiliar with VR may skew the results in one way. This will be discussed further in section 8.1.2.

Participants were also asked to clarify what they thought the SteamVR implementation did better. Here, based on the survey and qualitative interviews, it is also possible to gain useful information. Out of the 15 survey responses, 9 expressed some degree of favorability for the SteamVR version's UI design and interactions. Based on Nielsen's heuristics, it can be argued that participants found a strong sense of efficiency of use with the SteamVR implementation. False input readings, as can be seen by the usability schema, were also encountered less often than in the Quest 2 version.

Among other findings, this naturally also carries implications. It could be sensible to assume that when presented the same base software, participants would prefer the one that allows them to move through it faster. Despite participants' preferences for bare hand tracking, the SteamVR implementation is frequently cited as the more efficient system.[21]

Finally, we thought it beneficial to create an overview of the heuristic issue each implementation generally faces. This can be used to reflect on implementation implications and potential foci for future developments. Figure 32 shows a treemap chart of both implementations and the frequency

[21] In fact, only one participant expressed the Quest 2 iteration as "faster" in the survey

of usability issues associated with different heuristics, proportional to the total amount of usability issues identified. It bears noting that, when accounting for iteration differences in the Quest 2 version, both implementations had a similar number of identified usability concerns. Table 5 shows the percentage values of the treemap data in a condensed manner.

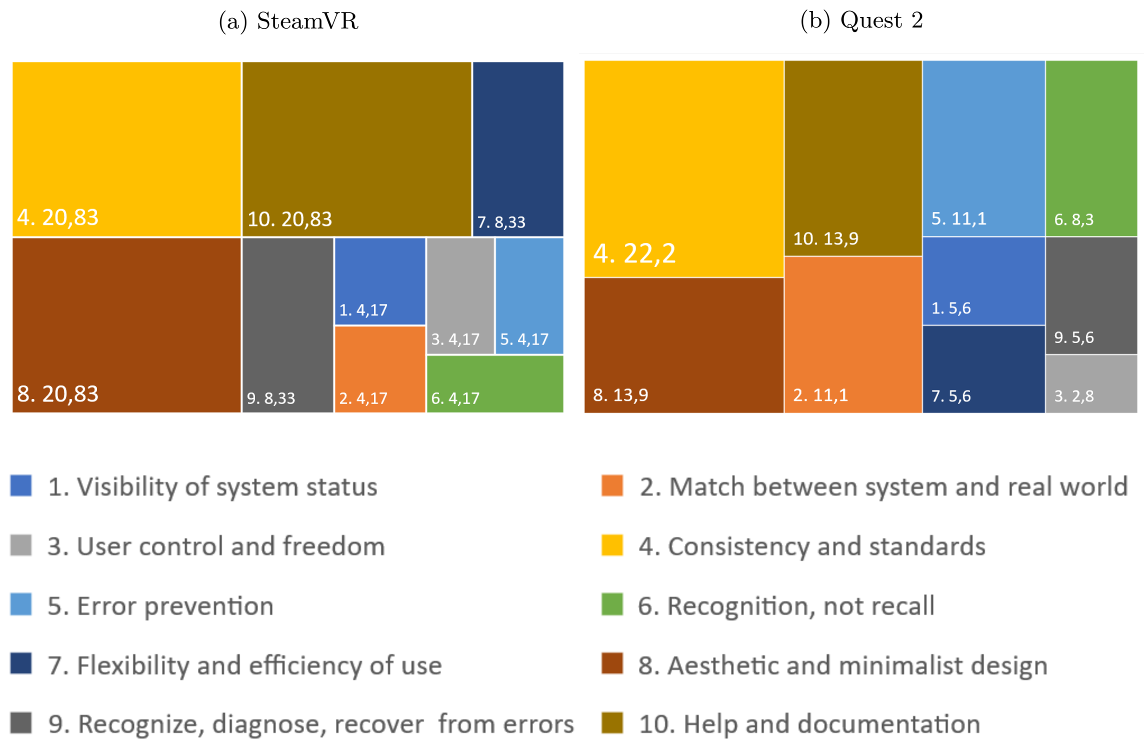


Figure 32: Treemaps of frequency of usability issues classified by heuristic. Numbers in blocks represent usability heuristic and percentage of total occurrences, respectively

Heuristic/Implementation	1	2	3	4	5	6	7	8	9	10
SteamVR	4.2%	4.2%	4.2%	20.8%	4.2%	4.2%	8.3%	20.8%	8.3%	20.8%
Quest 2	5.6%	11.1%	2.8%	22.2%	11.1%	8.3%	5.6%	13.9%	5.6%	13.9%

Table 5: Identified usability heuristic frequencies

It can be observed that both iterations have disproportionate issues with heuristics 4, 8, and 10, which revolve around consistency and standards, aesthetic and minimalist design, and help and documentation respectively. Help and documentation could be argued to be purposefully lacking. Both iterations of ViRMA are very early in development, and onboarding as well as support tools have not been a priority for either version. Consistency and standards manifests in different ways for both implementations. The SteamVR implementation has slight internal inconsistencies as well as breaches of expectations from experienced VR users (1.06, 2.04). The Quest 2 implementation generally has a larger share of internal inconsistencies. Most of these are manifested in the aforementioned controller disparities. Finally, aesthetic and minimalist design for both versions are generally not optimal, mostly relating to presentation of information, environment visibility (or lack thereof), and UI design.

Before concluding we want to relate to the final Likert-scale question of the survey, querying participants about their subjective enjoyment. This data can be viewed on table 6. Figure 33 provides visualization of it.

	1	2	3	4	5
Iteration 1	2	3	0	0	3
Iteration 2	2	2	2	2	0
Total	4	5	2	2	3

Table 6: Q: If I had to use ViRMA for work or personal enjoyment, I would prefer using the Index version over the Quest 2

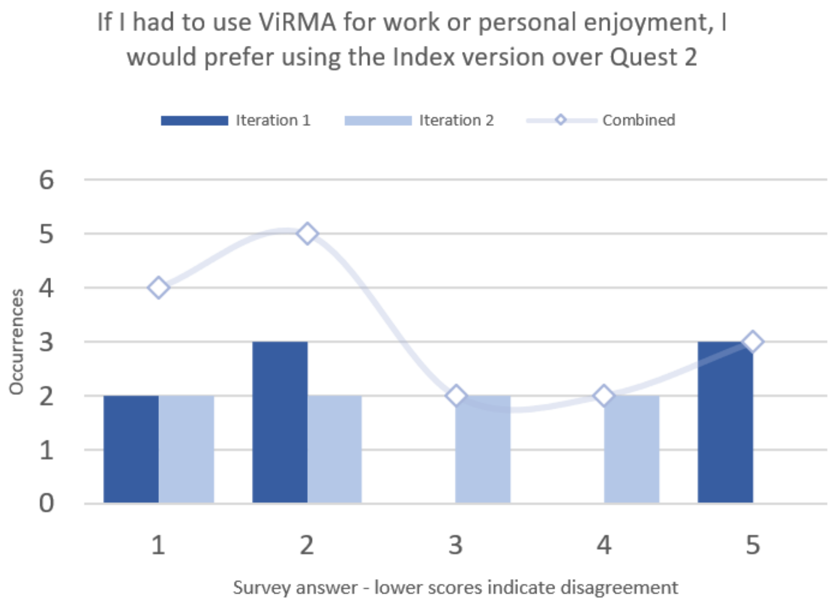


Figure 33: Participant enjoyment survey question chart

Overall, we find it justifiable to conclude the usability study with the following statement:

The usability study generally found the Quest 2 implementation to be more friendly, primarily citing intuitiveness of hand tracking controls and perceived safety of the passthrough functionality. However, many participants expressed that the SteamVR implementation featured more usable UI design and consistent/precise input readings.

8 Discussion and reflections

In this section, we will discuss and reflect on several aspects of the thesis. These include

- Potential implications of usability test results
- Reflection on methodology
- Future work

8.1 Methodology, practical considerations

Before discussing the implications of the usability test results, we will examine factors that could have impacted the outcome of the tests, including the potential implications and conclusions thereof.

8.1.1 Implementation faults

First, the development of the Quest 2 implementation was volatile. Due to a lack of prior experience with VR development and transitioning from SteamVR to Oculus' VR API, it was not possible for us to implement all features without some necessary and severe compromises.

The bare hand inputs were highly limited. While pose recognition could allow for further input mappings (e.g. making a flat hand to reset a query), it was not feasible to implement. Instead, the hands are interpreted as Oculus VR controllers that have two potential inputs, pinching the index finger with the thumb on either hand. This resulted in the compromise of making the right hand the "interactor" while the left hand is the "grabber". From a usability perspective, we did not consider this satisfactory which the usability study confirmed through participants' negative experience of it. This was compounded by persistent tracking issues that incurred multiple false input readings.

Another issue was the lack of keyboard tracking. This was not ideal due to the limited visibility of keys, which we were aware of during development. While participants generally managed to succeed with the keyboard, and many expressed that they would enjoy using a "tracked keyboard" more than the SteamVR implementations' virtual one, it is not possible to safely conclude this without testing an actual implementation. This results in a lack of some degree of data validity regarding the keyboard implementation aspects of the usability tests. In other words, we're not necessarily measuring the usability of a tracked keyboard implementation (as we would want), but rather world-grounded input devices in a passthrough environment. This information can also be valuable and we will use it to formulate some conclusions, but the shortcomings bear mentioning nonetheless.

8.1.2 Usability testing vulnerabilities

While we generally consider the usability test defensible and the data thereof valid, it could be argued that some of the circumstances could lead to unintended biases or compromised data validity.

Sample population

The sample population for the usability tests were vague to the point where, theoretically, it could be applicable to most of the population. The motivation for this choice was that a broad demographic may reveal more usability issues, even if the conclusion can't be as secure. However, in practice, the demographic the usability tests ended up with can broadly be described as:

- Young adults (18-30)
- Primary occupation: Student at higher education
- Higher than average degree of tech-savvyness (several pursuing education at the IT University of Copenhagen, 5 familiar with VR)

As described in section 5.2, the improvements were designed with productive work in mind. For example, the software could be used in an office surrounding colleagues. One could argue that it would then have been more beneficial to limit the sample population to participants whose primary occupation are in office environments.

We consider this criticism valid to some extent. An explicitly defined sample population that attempts to adhere to the defined demographic which guided the design decisions could have stronger data validity. However, practical considerations such as time and resource constraints led us to prioritizing sample size and participant availability.

When taking several of the usability issues in mind, we consider this approach defensible. The tests uncovered numerous potential usability issues with both high and low hit rate. We think it is justified to assume that many if not all of the high hit rate issues (such as keyboard visibility) would also be encountered by the specified demographic. Since most of the conclusions primarily relate to the high hit rate usability issues, we do not consider the demographic mismatch a severe weakness. However, the relatively high degree of participants with VR experience has most likely impacted the overall evaluation to some extent.

Lack of onboarding / Software walkthrough

The first iteration of tests may also suffer methodological weaknesses. Most of the justified criticism of this would, in our opinion, pertain to the fact that participants did not receive any introduction to the software. This has the effect of them not just evaluating the software, but also the onboarding of the two implementations. Further criticism could note that the Quest 2 iteration of the software introduced an information menu that gave users some advice on controls. While this could easily have been added to the SteamVR implementation, it was not. It could be argued that the information menu presented an unequal advantage to the evaluation of the software, without aiding in testing what was pertinent (hand tracking, keyboard implementation, passthrough). The motivation for including the information menu was an expectation that users would not otherwise be able to identify the keyboard and hand tracking as part of the software otherwise.[22] This idea was essentially confirmed by the fact that no participant attempted to use hand tracking even in spite of the information menu explicitly stating its feasibility.

We consider this concern justified to some extent. Had the testing purely consisted of iteration 1, it would have severely compromised the data validity and served as a weak foundation for future

[22] Unlike SteamVR ViRMA where all aspects of the software are clearly identified

conclusions. That is not to say that iteration 1's testing of the onboarding process was unfruitful. Without it, it would not have been possible to determine how users perceive an "invisible" input method such as hand tracking. Even with explicit guidance from the information menu, most participants either did not understand the hand tracking capabilities or preferred not to use due to a lack of familiarity.

Further, we think all of the main criticisms of iteration 1 have been taken into account and corrected for in the second iteration. Iteration 2 removes unequal onboarding measures between the two implementations and introduced very brief software walkthroughs for both. The software walkthrough clarified input methods such as choosing between hands and controllers. Finally, minor implementation changes that are reminiscent of the iterative design cycle resulted in more nuanced data, as we could now also measure the impact of hand controls with higher feature parity with Touch controllers. As shown in section 7.2, this may have had a measurable impact on participant enjoyment of the feature.

8.2 Usability test result implications

In this section, we will discuss some of the implications of the usability test findings, along with how these could guide future developments of ViRMA or similar VR SMA applications.

First, for ViRMA it can be somewhat safely concluded that for exploration purposes, allowing users to view and navigate through tag hierarchies is beneficial. This feature provides some degree of contextualization and allows for easy "side-tracking", which is useful in an exploratory context. On the other hand, the concrete implementation of the `DimensionBrowser` in the SteamVR implementation of ViRMA is physically demanding and overwhelming to some. Further, as a search results browser, it does not allow for efficiency of use. The context surrounding each tag that matches the search string occupies space and results in further distance between other search result tags.

A potential improvement to this could include condensing the dimension browser similar to the Quest 2 iteration, but allowing users to expand the context for each tag, traversing the hierarchy it is present in. Such a method would allow for directed exploration of search results without physical exertion, while allowing for greater exploration. However, given the vast potential of VR, innovative methods of displaying and traversing tag results could be devised. The chief conclusion from this thesis on the matter is that such an implementation should seek not to be too physically demanding (allowing for longer-term use without fatigue), and allow for fast retrieval of information pertinent to the search result, if the user is only interested in this.

The following implications are not necessarily directly related to ViRMA or SMA VR applications. With this we mean that they are not intrinsic to applications related to the theoretical foundation that ViRMA is built upon, described in section 2. Instead, these could likely be applicable to many types of VR software that rely on productivity apps using passthrough environments, world-grounded input devices, and bare hand tracking.

The chief criticism of the passthrough was a perceived increase in "distraction" or "noise" added by being able to see one's environment. As participants generally preferred the passthrough environment, it could be concluded that in spite of this criticism, it is the optimal choice (see table 2). However, in fitting with Nielsen's sixth heuristic of flexibility and efficiency of use, adding options to allow users to change their environment could be beneficial. Figure 34 from Oculus demonstrates how such a passthrough settings menu could look.[23]. We recommend implementing such a feature.

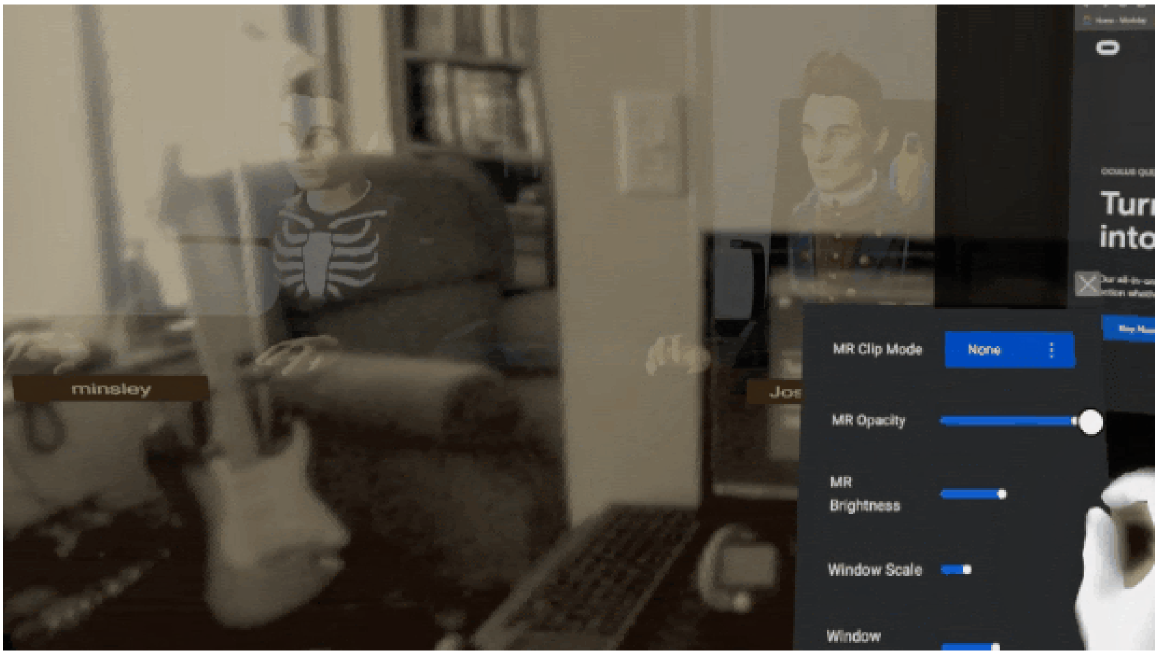


Figure 34: Oculus Passthrough settings demonstration [34]

As described in the previous section, it is difficult to make any conclusions on keyboard implementation, taking into account we considered the implementation inherently faulty. Nonetheless, the analysis findings can be used to make some conclusions. Possibly most important is the idea that MR/VR applications should seek to either 3D render all input devices, or none of them. Combining tracked and non-tracked input devices breaches Nielsen's fourth heuristic of standards and consistency, and may lead to users not recognizing non-tracked input devices, or interpreting unrelated devices as input devices.[24] Further, the Quest 2's passthrough visibility is quite low, meaning that even if it can facilitate MR applications, it is not useful for use with world-grounded input devices that require clear visibility for the reasons Jerald mentioned [16] (p. 311).

8.3 Future work

Based on research done for this thesis, we propose a number of directions future work could take. In this section, we will discuss some of these directions, what shape they could take, and what the benefits could be.

8.3.1 Hands API

During the development of the Quest 2 iteration, Oculus released a major iteration of the hand tracking API. This is highly relevant for future work of ViRMA/SMA VR software that emphasizes

[23] Follow the citation for a video demonstration
[24] Such as some participants trying to use the mouse in the Quest 2 iteration as they associate it with the keyboard

hand tracking. Due to this, we will briefly explore what integrating the new revision of the API could mean for future developments.

As described, Quest bare hand tracking works via the headset detecting the user's hands through the front-facing IR cameras. This is contrary to the Oculus Touch controllers that have internal sensors and use simultaneous localization and mapping to track their position. The reliance on visibility results in most of the hand tracking technology's most severe issues. For example, any action that includes occlusion of one hand may result in the hand not being tracked anymore. This could be common in applications that require users to pass objects between their hands, clap, or perform similar gestures.

The Hands 2.0 API reportedly improves both tracking accuracy and occlusion issues. While at the time of writing it has not been possible to find details on the implementation changes, demonstration videos compare the old and new implementation, with the newer one being significantly better at handling occlusion.

Further, the new iteration seems more precise, allowing for more nuanced operations. Further development of such an API could increase usability. Not just in for reducing potential input errors, but also increasing the amount of recognized gestures, allowing for greater control potential. Figure 35 shows a comparison between Hands 1.0 and Hands 2.0 during a clap operation. Hands 2.0 shows both hands whereas the lower hand has lost tracking in Hands 1.0.[25]

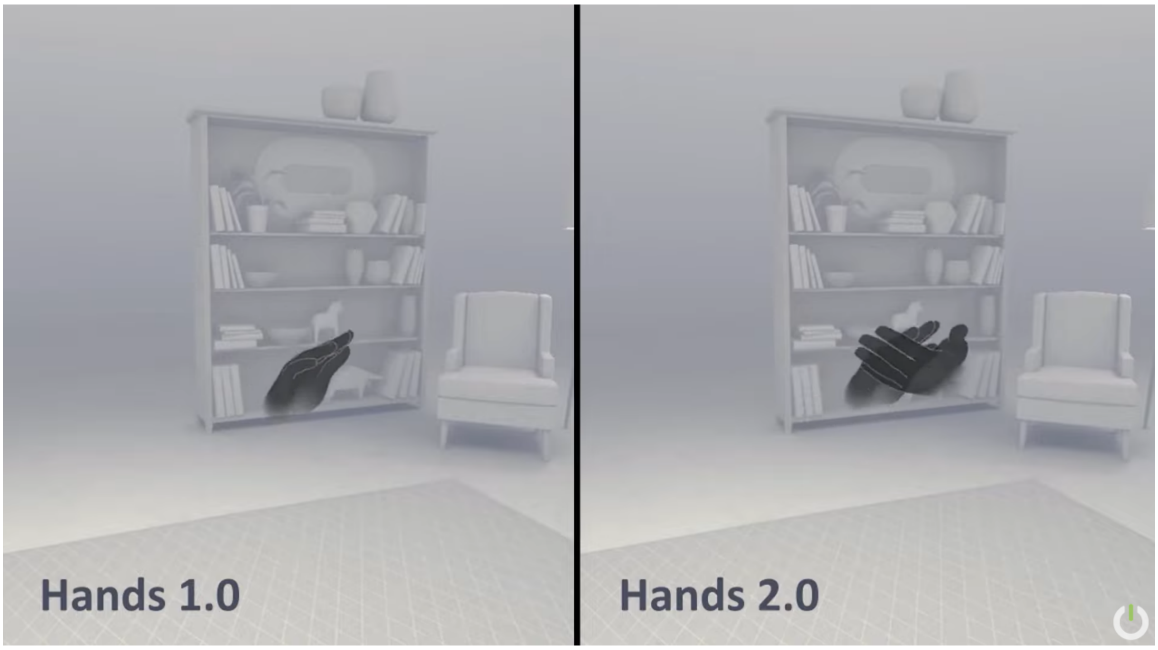


Figure 35: Hand iteration comparison, clapping operation [14]

While it does not seem that Hands 2.0 implements any new control methods that were not possible prior, integrating it could potentially improve several of the usability issues related to hand tracking in the Quest 2 implementation. Further accurate readings could also be used to experiment

[25]Refer to [14] for a link with a video demonstration

with novel hand tracking inputs such as pose recognition.

8.3.2 Native Quest 2

Another direction for a Quest 2 application could be to port ViRMA natively to the Quest 2. As described in section 5.1, there were practical considerations that made porting ViRMA to the Quest 2 unfeasible within this thesis. However, future research could potentially attempt this. ViRMA running natively on a standalone headset could lead to interesting possibilities for SMA VR applications. Most importantly, such an approach would completely de-tether the user, allowing for using the software in any environment without physical restrictions such as cable length and external PC requirements. This could be relatively feasible with our implementation of ViRMA, as it has moved to using the Oculus API exclusively.[26] However, future research that attempts this must take the following matters into account:

- Currently the tag-server runs locally on the host machine. This tag-server must be established externally of the Quest 2 (which then fetches via HTTP requests), or be set up to run internally on the ViRMA application in Unity
- The Quest 2 must be able to access the data-set. If this is stored on local storage, the size of the application may exceed 64GB Quest 2s. Storing these on an external server that the app fetches upon query introduces new complexities as images would have to be streamed in as game objects in Unity.
- The Quest 2 is significantly weaker in terms of processor and graphical power compared to the desktop PCs ViRMA has been developed and run on. The performance impact of larger queries on the Quest 2 is unknown and possibly not feasible without major performance optimizations

8.3.3 SteamVR iteration of ViRMA

Future work on the SteamVR iteration of ViRMA could also be informed by findings in this thesis. Due to the proprietary nature of some of the technologies used here, elements such as the passthrough API and hand tracking are not applicable to the SteamVR implementation as SteamVR does not currently support these. Nevertheless, the usability tests revealed potential issues with the SteamVR iteration of ViRMA that could be rectified. Some of these include:

- Changing the environment to be more welcoming
- Improving onboarding experience[27]
- More consistent controls
- More intuitive menu-grabbing
- Implementing point-and-click rather than hover-and-click
- Re-evaluating dimension browser and tag-traversal. Limit physical exertion required and information overload while maintaining context and exploration.

[26] Although changing build settings to Android would almost certainly break some ViRMA implementations

[27] This is already the subject of the forthcoming thesis *Layered VR-User Interfaces - Onboarding of users in ViRMA* [10]

9 Conclusion

In this thesis we have presented the history and state of virtual reality Scalable Multimedia Applications, with ViRMA being in particular focus. Based on analytical work conducted with established VR usability literature, we have identified potential usability issues with ViRMA, including but not limited to the immersive virtual environment, controller implementation, and "Dimension Explorer".

Building on existing guidelines for VR usability, we designed and implemented several novel features to the ViRMA application, with the goal of creating potential guidelines for future development of similar applications. Chief among these designs were:

- An implementation of passthrough technology, allowing users to see their physical environment while using the application
- Introducing physical keyboard implementation, allowing users to rely on a familiar input method
- Implementing bare hand tracking, allowing for use of the system completely free of VR tracked hand-held controllers

Due to time and resource constraints, certain severe compromises had to be made for many of the features. This led to an expected decrease in usability.

The original and newly changed iterations of ViRMA were subject to user testing, where their usability was compared directly to each other on both individual features and a holistic level. Based on findings of the usability test analysis, it can be concluded that, the changes to the Quest 2 implementation improved the overall usability of the software on a holistic level. The biggest perceived benefits of the changes was implementing passthrough, which was related to a high degree of subjective satisfaction. While hand tracking received moderate-to-positive reception, primarily citing suboptimal implementation as the chief criticism. Of the main changes, the keyboard implementation received most criticism due to low key visibility making visually-oriented typing difficult. Finally, the paper makes suggestions to future work within both ViRMA and SMA VR applications, offering multiple directions for these to go in.

References

[1] Jonas Sylvain Jersild Balin and Katarzyna Honorata Toborek. *Improvements to the Web-Based Implementation of PhotoCube*. Dec. 2021 (cit. on p. 2).

[2] Oculus Blog. *Thumbs Up: Hand Tracking Available on Oculus Quest This Week*. Dec. 2019. URL: `https://www.oculus.com/blog/thumbs-up-hand-tracking-now-available-on-oculus-quest/` (cit. on p. 9).

[3] Mitchell Clark. *Oculus' Passthrough API will enable experiences that mix VR and the real world*. Screenshot taken at timestamp 0:44. July 2021. URL: `https://www.theverge.com/2021/7/23/22590794/oculus-mixed-reality-api-quest-2-development-kit-unity` (cit. on p. 26).

[4] Peter Øvergård Clausen. "A Web Browser Based Implementation of PhotoCube". MA thesis. IT University of Copenhagen, 2019 (cit. on p. 2).

[5] Vivek Dhakal et al. "Observations on Typing from 136 Million Keystrokes". In: *CHI '18: Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* 646 (2018), pp. 646–658. DOI: `10.1145/3173574.3174220` (cit. on p. 10).

[6] Unity Documentation. *Leap Motion Sets a Course for VR*. Aug. 2014. URL: `https://web.archive.org/web/20160530090125/http://blog.leapmotion.com/leap-motion-sets-a-course-for-vr/` (cit. on p. 9).

[7] Aaron Duane and Björn Þór Jónsson. "ViRMA: Virtual Reality Multimedia Analytics at LSC 2021". In: *Proceedings of the 4th Annual on Lifelog Search Challenge*. LSC '21. Taipei, Taiwan: Association for Computing Machinery, 2021, pp. 29–34. ISBN: 9781450385336. DOI: `10.1145/3463948.3469067` (cit. on p. 3).

[8] Kara M. Duraccio et al. "Does iPhone night shift mitigate negative effects of smartphone use on sleep outcomes in emerging adults?" In: *Sleep Health* 7.4 (2021), pp. 478–484. ISSN: 2352-7218. DOI: `https://doi.org/10.1016/j.sleh.2021.03.005`. URL: `https://www.sciencedirect.com/science/article/pii/S2352721821000607` (cit. on p. 22).

[9] Jamie Feltham. *Oculus quest 2 controller battery life lasted me 3 months*. Nov. 2020. URL: `https://uploadvr.com/oculus-quest-2-controller-battery-life/` (cit. on p. 30).

[10] Gina-Josephine Fischer, Sebastian Matthew Lajciak, and Ida Maria Aunsbjerg Villadsen. *Layered VR-User Interfaces - Onboarding of users in ViRMA*. Work in Progress Thesis. 2022 (cit. on pp. 3, 22, 57).

[11] Snorri Gíslason, Björn Þór Jónsson, and Laurent Amsaleg. "Integration of Exploration and Search: A Case Study of the M^3*Model*". In: *MultiMedia Modeling*. Ed. by Ioannis Kompatsiaris et al. Cham: Springer International Publishing, 2019, pp. 156–168. ISBN: 978-3-030-05710-7. DOI: `10.1007/978-3-030-05710-7_13` (cit. on p. 4).

[12] Cathal Gurrin et al. "[Invited papers] Comparing Approaches to Interactive Lifelog Search at the Lifelog Search Challenge (LSC2018)". In: *ITE Transactions on Media Technology and Applications* 7.2 (2019), pp. 46–59. DOI: `10.3169/mta.7.46` (cit. on p. 4).

[13] Cathal Gurrin et al. "Introduction to the Fourth Annual Lifelog Search Challenge, LSC'21". In: *Proc. International Conference on Multimedia Retrieval (ICMR'21)*. Taipei, Taiwan: ACM, 2021 (cit. on p. 20).

[14] David Heaney. *Quest 2 Hand Tracking 2.0 brings dramatic improvements.* May 2022. URL: https://uploadvr.com/quest-2-hand-tracking-2-0-upgrade/ (cit. on p. 56).

[15] Adrian H. Hoppe et al. “qVRty: Virtual Keyboard with a Haptic, Real-World Representation”. In: *20th International Conference, HCI International 2018, Las Vegas, NV, USA, July 15-20, 2018, Proceedings, Part II* 851 (Apr. 2018), pp. 266–272. DOI: 10.1007/978-3-319-92279-9_36 (cit. on p. 11).

[16] Jason Jerald. *The VR Book: Human-Centered Design for Virtual Reality.* Association for Computing Machinery, Morgan Claypool Publishers, 2016. ISBN: 978-1-97000-113-6 (cit. on pp. 7–9, 11, 13, 14, 55).

[17] Janis Gisel Jimenez. “A Prototype for Text Input in Virtual Reality with a Swype-like Process Using a Hand-trackingDevice”. MA thesis. UC San Diego, 2017 (cit. on pp. 10, 11).

[18] Dr. Michael Kipp. *Radial Text Input in Virtual Reality.* How it was published. Augsburg University of Applied Sciences, 1993 (cit. on pp. 10, 23).

[19] Søren Lauesen. *User Interface Design: A Software Engineering Perspective.* Pearson Addison Wesley, 2005 (cit. on pp. 32, 41).

[20] Lucas Matney. *Once poised to kill the mouse and keyboard, Leap Motion plays its final hand.* May 2019. URL: https://techcrunch.com/2019/05/30/once-poised-to-kill-the-mouse-and-keyboard-leap-motion-plays-its-final-hand/ (cit. on p. 9).

[21] Casey Newton. *Mark in the Metaverse.* July 2021. URL: https://www.theverge.com/22588022/mark-zuckerberg-facebook-ceo-metaverse-interview (cit. on p. 1).

[22] Jakob Nielsen. *10 usability heuristics for user interface design.* Nov. 2020. URL: https://www.nngroup.com/articles/ten-usability-heuristics/ (cit. on p. 32).

[23] Oculus. *Introducing Oculus Air Link, a Wireless Way to Play PC VR Games on Oculus Quest 2, Plus Infinite Office Updates, Support for 120 Hz on Quest 2, and More.* Apr. 2021 (cit. on pp. 11, 12).

[24] *OpenVR (Steamworks Documentation).* Valve (cit. on p. 15).

[25] Ananth N. Ramaseri Chandra, Fatima El Jamiy, and Hassan Reza. “A Systematic Survey on Cybersickness in Virtual Environments”. In: *Computers* 11.4 (2022). ISSN: 2073-431X. DOI: 10.3390/computers11040051. URL: https://www.mdpi.com/2073-431X/11/4/51 (cit. on p. 13).

[26] Hlynur Sigurþórsson. “PhotoCube: A Multi-Dimensional Image Browser”. MA thesis. Reykjavík University, 2011 (cit. on p. 2).

[27] Florian Spiess et al. “Exploring Intuitive Lifelog Retrieval and Interaction Modes in Virtual Reality with vitrivr-VR”. In: *Zurich Open Repository and Archive* (2021). DOI: 10.1145/3463948.3469061 (cit. on p. 5).

[28] Debbie Stone et al. *User interface design and evaluation.* Morgan Kaufmann, 2005 (cit. on p. 32).

[29] Oculus Support. *Oculus quest 2 VR headset setting up Guardian | Oculus.* Screenshot taken at timestamp 02:11. Oct. 2020. URL: https://www.youtube.com/watch?v=d73PfTXZDvo (cit. on p. 13).

[30] Eric Switzer. *Quest 2 Hand Tracking Is A Neat Trick, But Far From Game Ready.* Oct. 2020. URL: `https://www.thegamer.com/oculus-quest-2-hand-tracking-waltz-wizard/` (cit. on p. 9).

[31] Paul Tassi. *Meta unwisely kills off the oculus name for VR headsets once and for all.* Jan. 2022. URL: `https://www.forbes.com/sites/paultassi/2022/01/27/meta-unwisely-kills-off-the-oculus-name-for-vr-headsets-once-and-for-all/` (cit. on p. 21).

[32] VIVE Team. *Introducing the logitech bridge SDK.* Feb. 2017. URL: `https://blog.vive.com/us/2017/11/02/introducing-the-logitech-bridge-sdk/` (cit. on p. 11).

[33] Grímur Tómasson. "ObjectCube – A Generic Multi-Dimensional Model for Media Browsing". MA thesis. Reykjavík University, 2011 (cit. on p. 2).

[34] Oculus VR. *Mixed reality with Passthrough.* July 2021. URL: `https://developer.oculus.com/blog/mixed-reality-with-passthrough/` (cit. on p. 55).

[35] Matt Weinberger. *The rise and fall of kinect: Why Microsoft gave up on its most promising product.* Jan. 2018. URL: `https://finance.yahoo.com/news/downfall-kinect-why-microsoft-gave-183900710.html?guccounter=1&guce_referrer=aHR0cHM6Ly93d3cuZ29vZ2xlLmNvbS8&guce_referrer_sig=AQAAAJNVZH-KSS55WtC2RE4bxol3CwlFWfE2xe67bnEEtlYvLl99aNFipZ_s2VZIm-XBjxlSYNjP1xSIGVhrzgtSha3nxHv3FA9Hj0TymTDyO1gEAyLutr9pR9jhf0HvMTs0YHAzWrpyuGkRnD7aB9j8` (cit. on p. 8).

[36] *Work Faster in VR Than in Real Life.* 2022. URL: `https://www.immersed.com/` (cit. on p. 11).

[37] *Xbox - Kinect effect.* The video in question is a re-upload of an original Microsoft advertisement. Nov. 2011. URL: `https://www.youtube.com/watch?v=FCw38l3rHW4` (cit. on p. 8).

[38] Robert Xiao et al. "MRTouch: Adding touch input to head-mounted mixed reality". In: *IEEE Transactions on Visualization and Computer Graphics* 24.4 (Apr. 2018), pp. 1653–1660. DOI: `10.1109/tvcg.2018.2794222` (cit. on p. 10).

[39] Jan Zahálka and Marcel Worring. "Towards Interactive, Intelligent, and Integrated Multimedia Analytics". In: *2014 IEEE Conference on Visual Analytics Science and Technology (VAST)* (2014), pp. 3–12. DOI: `doi:10.1109/VAST.2014.7042476` (cit. on p. 4).

A ViRMA Usability testing mail

Template text sent to participants who fulfilled the initial survey and/or expressed interest in participating.

Hello `name`, We are writing because you have expressed interest in participating in a usability test of an experimental virtual reality software, as part of a master's thesis being conducted at the IT University of Copenhagen

We are conducting the tests next week, between the 2nd and 4th of May, at ITU's or Univate's premises (located across the street from ITU) and would like to include you as subject in a session. The session is expected to take 65-75 minutes total, with a five to ten-minute break in the middle. During the session, you will try two different implementations of the same base virtual reality software. The VR testing portion of the session is split in two, with each lasting roughly 15 minutes. The remainder of the session will be spent on interviewing you about your experience.

If you are still interested in participating, please let us know when it would be best for you to conduct the test. Let us know if you have any wishes that would make your experience more pleasant, for example the presence of a specific drink or snacks.

Best regards,
Jonas Balin & Katarzyna Toborek

B Presentation

The data

So, ViRMA is all about exploring data. In order to do so, it is good to very quickly talk about what data we have.

The data you will browse through is from a “lifelog” data-set. Basically, a person logged their life by attaching a camera to themselves and taking a photo every 30 seconds. We have *a ton* of photos!

How do we organize this?
All the photos have extensive metadata that describe what is on the photo (example follows)

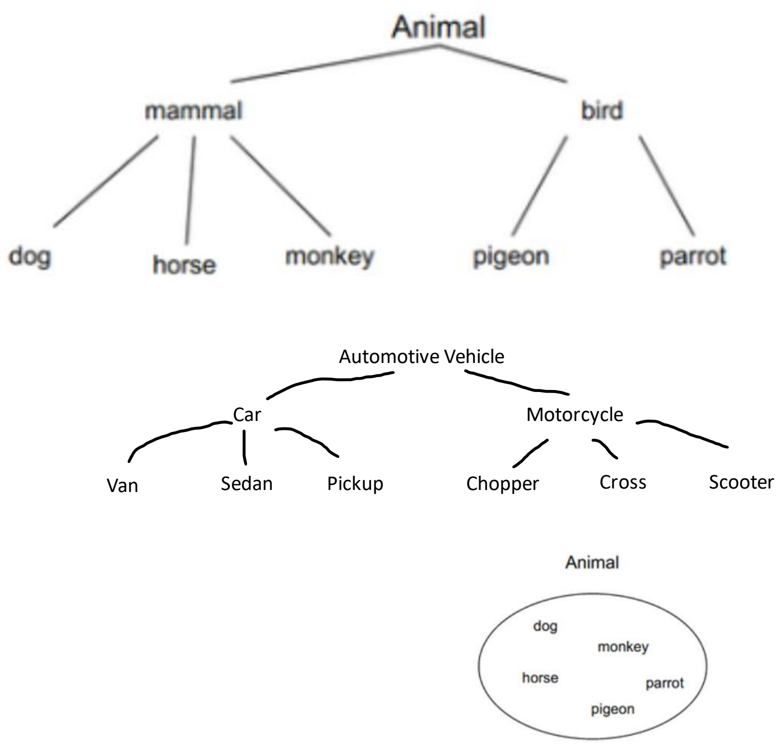

Tags:
Hills,
Wood,
House,
Front loader,
Clouds,
Dirt
Datetime:
09:30:30
23/11/2013

Tags

Tags can be ordered into "hierarchies" where there are children, siblings, and parents.

You can kind of imagine it as family trees.

Tag-sets are similar, but are non-hierarchical orderings of tags.



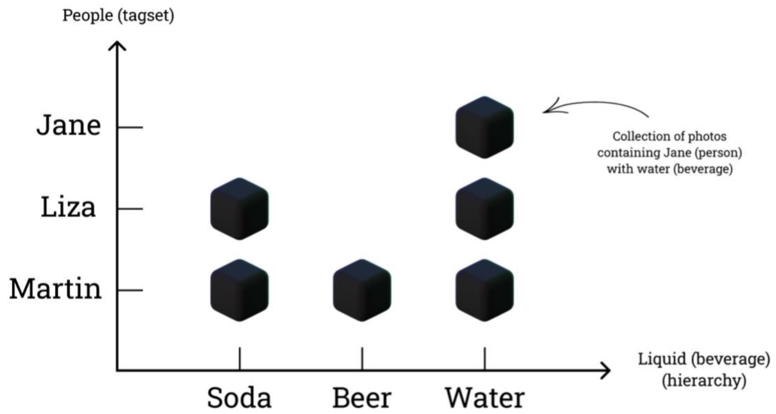
Ordering data on axes

You are probably familiar with two-dimensional axes (X and Y).

The example to the right has liquid (beverages) on the X-axis and people on the Y-axis.

Each value within this coordinate system just represents one or more photos that have both of the tags.

Naturally, ”empty spaces” indicate that there are no photos with both the X and Y value



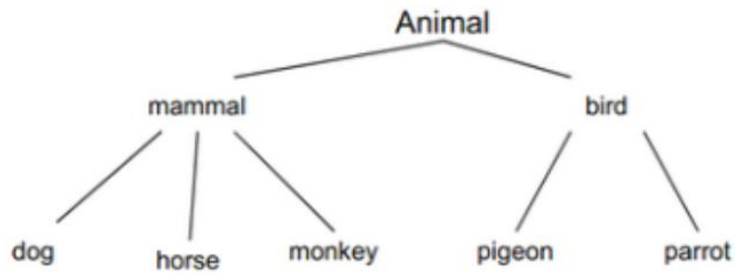
“Drilling down” and “rolling up”

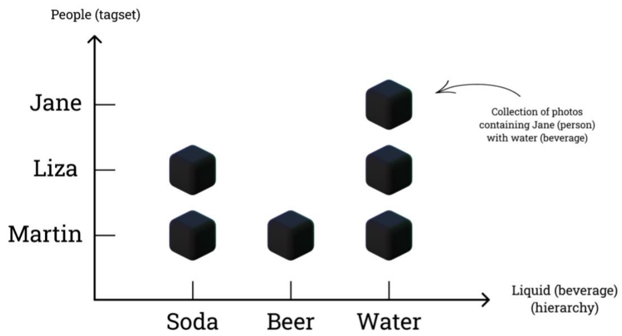
Let’s imagine you have the list of hierarchies from before.

“Drilling down” simply describes going from a parent to a child and “rolling up” means you are going from a child to a parent.

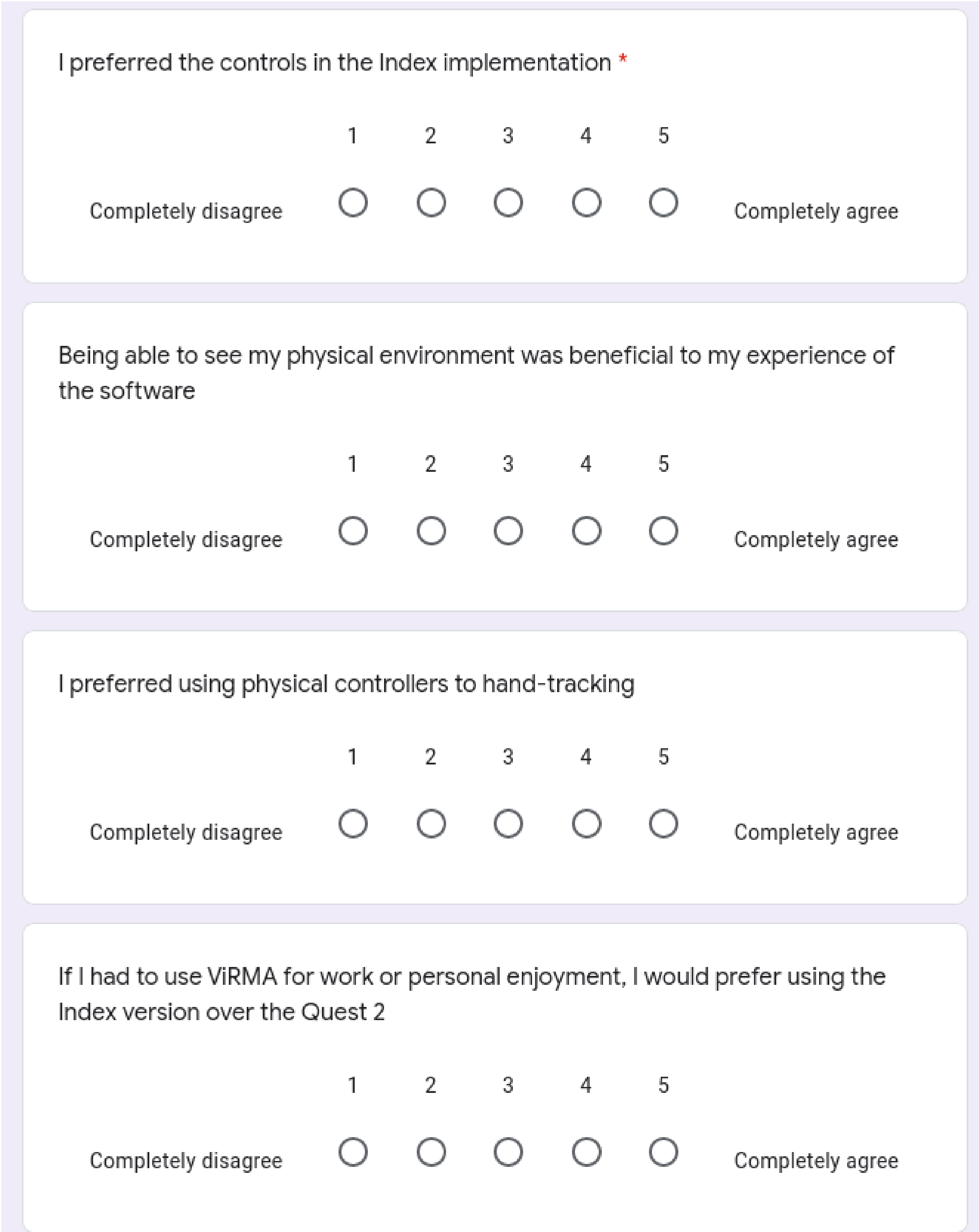
Using the schema to the right, if you are ”standing” on mammal and you want to view pictures only of dogs, you would be ”drilling down”.

Instead, if you wanted to generalize and view pictures of all animals, you would be ”rolling up”





C Usability testing concluding survey



I preferred the controls in the Index implementation *

	1	2	3	4	5	
Completely disagree	○	○	○	○	○	Completely agree

Being able to see my physical environment was beneficial to my experience of the software

	1	2	3	4	5	
Completely disagree	○	○	○	○	○	Completely agree

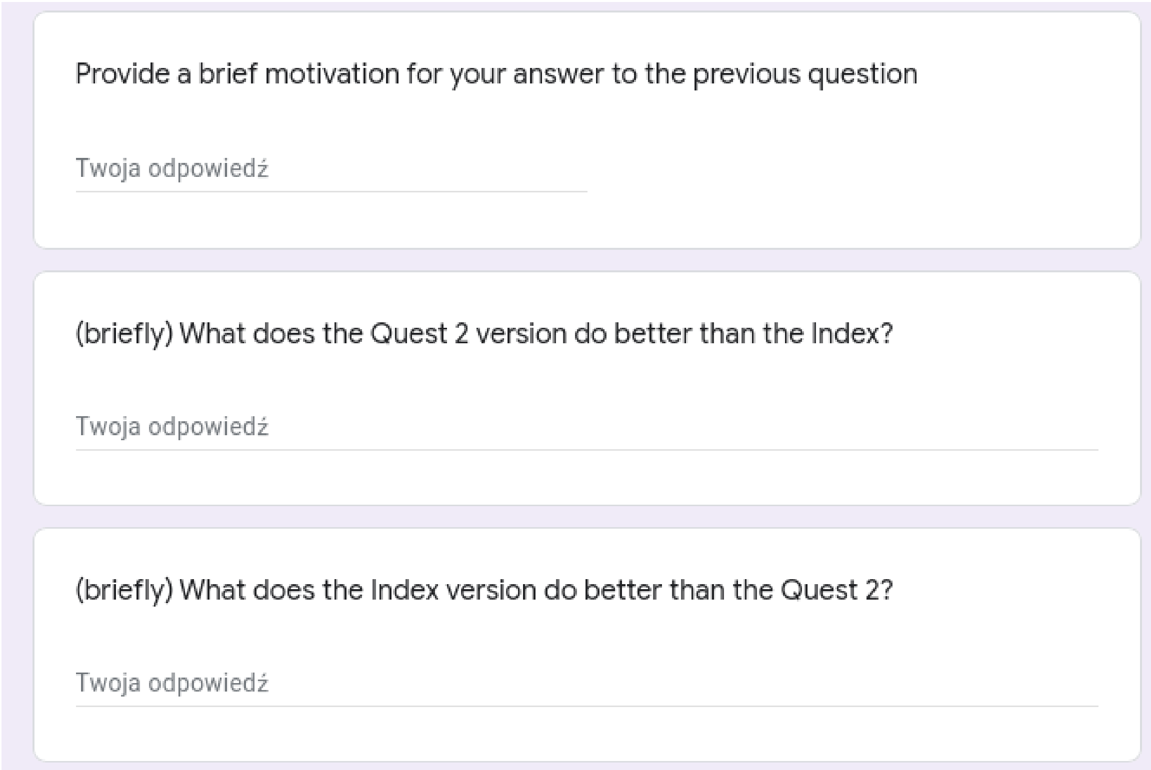
I preferred using physical controllers to hand-tracking

	1	2	3	4	5	
Completely disagree	○	○	○	○	○	Completely agree

If I had to use ViRMA for work or personal enjoyment, I would prefer using the Index version over the Quest 2

	1	2	3	4	5	
Completely disagree	○	○	○	○	○	Completely agree

Figure 36: Usability testing concluding survey form



Provide a brief motivation for your answer to the previous question

Twoja odpowiedź

(briefly) What does the Quest 2 version do better than the Index?

Twoja odpowiedź

(briefly) What does the Index version do better than the Quest 2?

Twoja odpowiedź

Figure 37: Usability testing concluding survey form - open questions

D Usability testing concluding survey responses

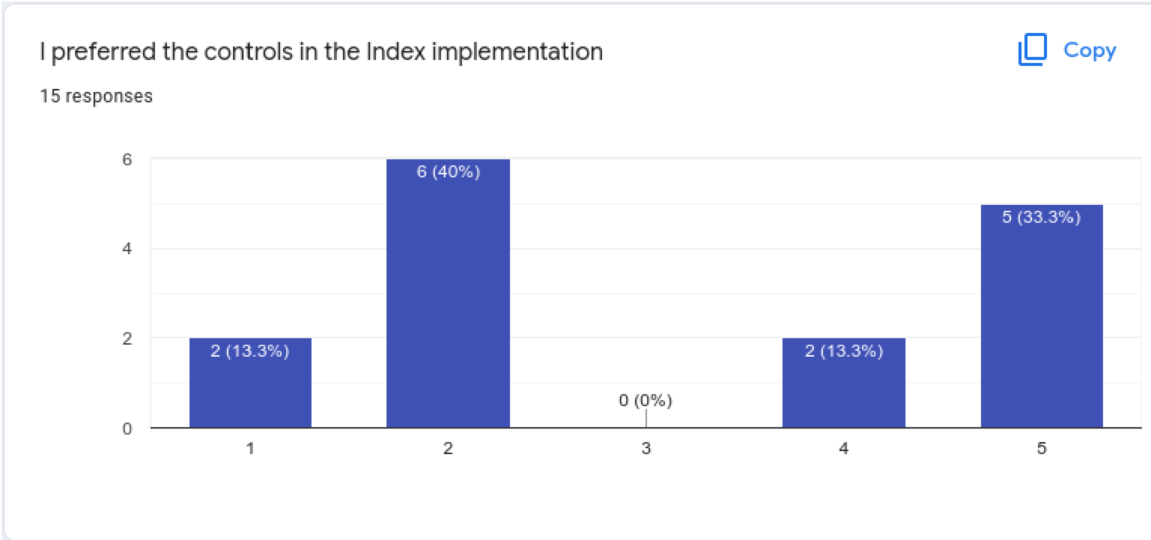


Figure 38: Usability testing concluding survey question 1 responses

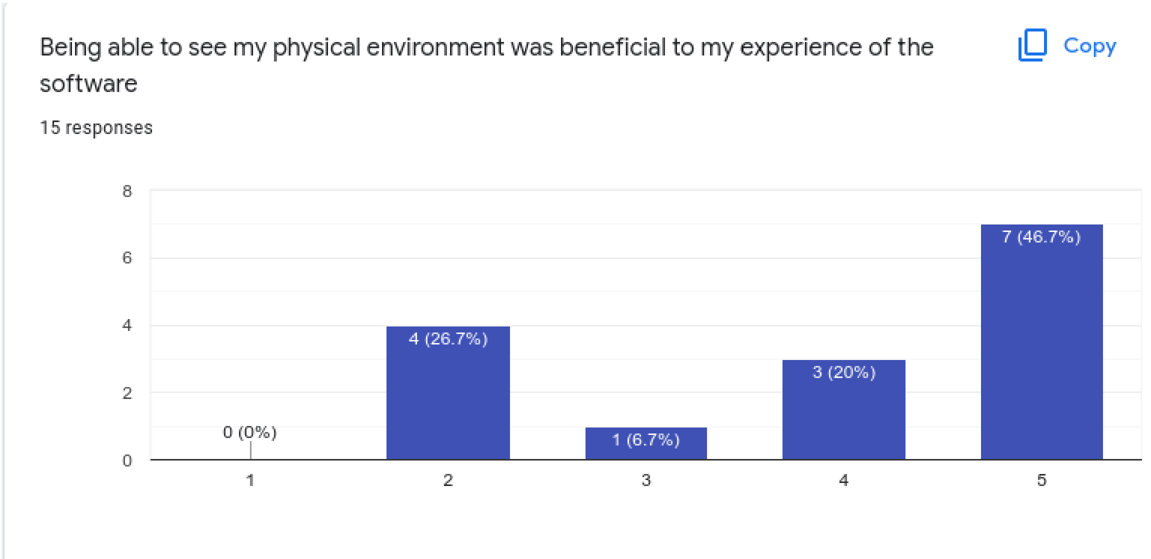


Figure 39: Usability testing concluding survey question 2 responses

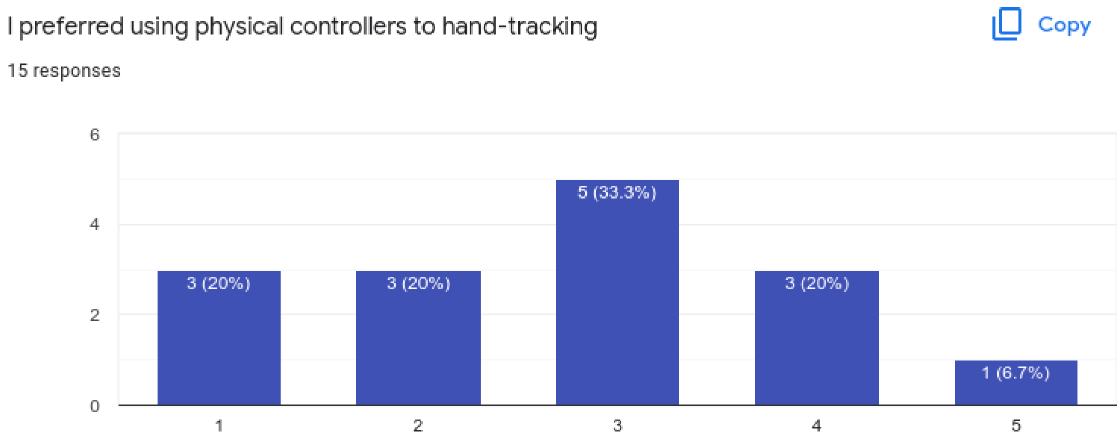


Figure 40: Usability testing concluding survey question 3 responses

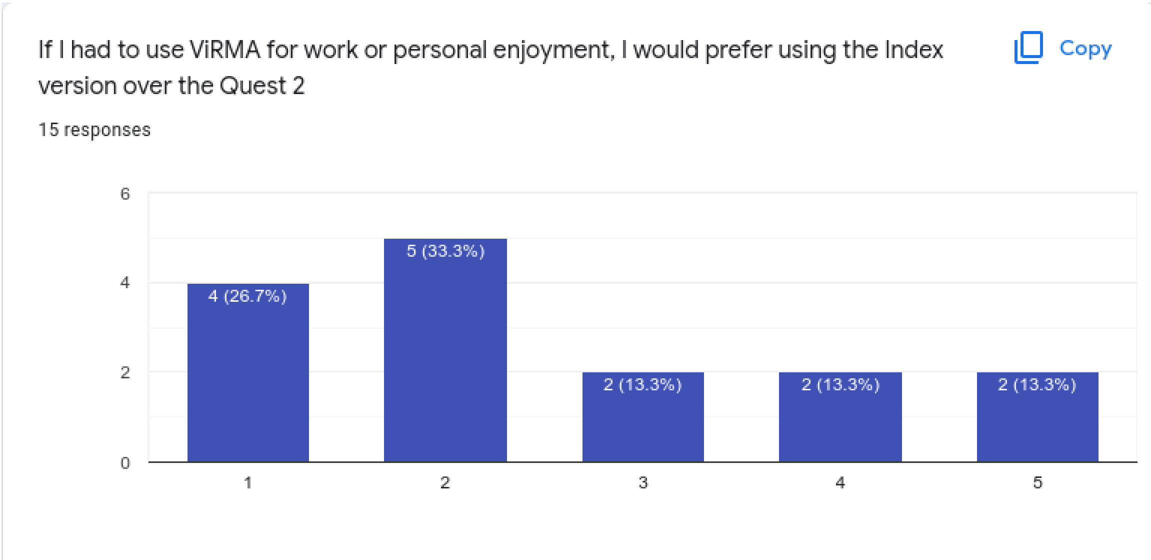


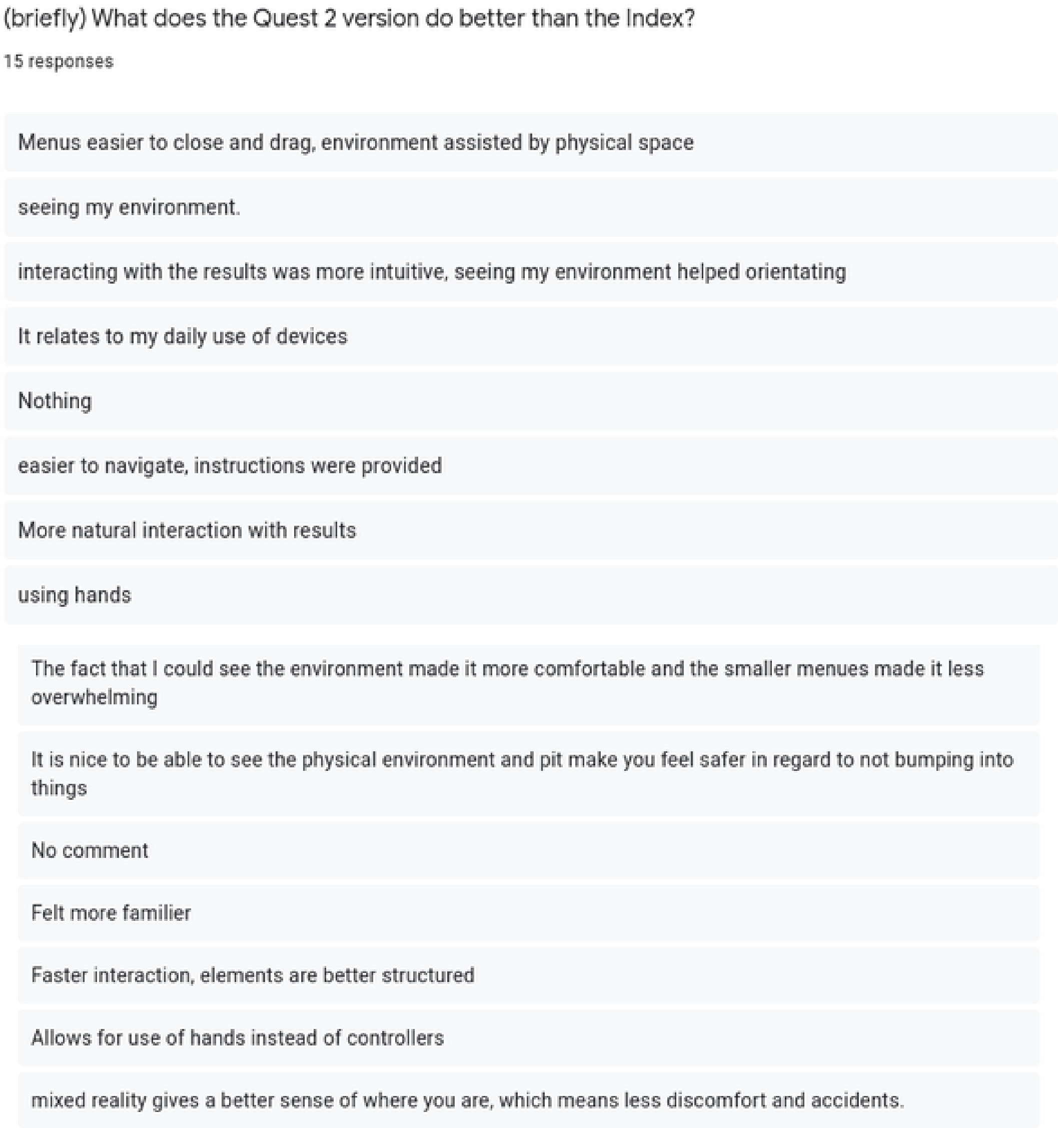
Figure 41: Usability testing concluding survey question 4 responses

Provide a brief motivation for your answer to the previous question

15 responses

- Adds to physical environments easy to navigate
- Being able to see my environment felt a lot better.
- seeing the query results was helpful
- Quest 2 was more intuitive
- Responsiveness and usability
- more intuitive to move stuff in 3d with your hands as used to
- Prefer being able to naturally interact with results
- harder to use controllers than hands
- Both had their benefits. Index was faster to move through but the other was more comfortable
- I think it was easier to read the labels in the index version and it was nice to have a rendered keyboard
- The enviornment
- Using was more intuitive and reminded me of how I use my other computer, iphone etc.
- Effeciency
- Preffered design of ViRMA but felt more comfortable using Quest 2.
- quest is less embarasing and userfriendly in an office.

Figure 42: Usability testing concluding survey question 5 responses



(briefly) What does the Quest 2 version do better than the Index?

15 responses

Menus easier to close and drag, environment assisted by physical space

seeing my environment.

interacting with the results was more intuitive, seeing my environment helped orientating

It relates to my daily use of devices

Nothing

easier to navigate, instructions were provided

More natural interaction with results

using hands

The fact that I could see the environment made it more comfortable and the smaller menues made it less overwhelming

It is nice to be able to see the physical environment and pit make you feel safer in regard to not bumping into things

No comment

Felt more familier

Faster interaction, elements are better structured

Allows for use of hands instead of controllers

mixed reality gives a better sense of where you are, which means less discomfort and accidents.

Figure 43: Usability testing concluding survey question 6 responses

(briefly) What does the Index version do better than the Quest 2?

15 responses

Had nice rotate functionality

controls and color coding.

displayed search results, clearer structure

Not sure

BEtter responsiveness better menues and less clutter

grouping of the images, hierarchy

Virtual keyboard

menu

Faster to move through, often felt more accurate and more intuitive

It is way easier to read the labels and just see the UI

Enviornment

The menu worked better with this one with the scroll etc

Cancels out possible distractions

It is visually more appealing.

moving around in the system is easier and fits better with what&how i want to solve a task intuitively.

Figure 44: Usability testing concluding survey question 7 responses

E Running the software

The Git repo for the project can be found at this link: `https://github.com/katarynka/ViRMA`

The branch is called OculusQuest.

This section will very briefly explain how to run the software. It presupposes the local machine has an active and correctly set-up tag-server.

The software has some pre-requisites:

- Quest 2 HMD
- Oculus PC Software
- USB-C cable

To run the software with passthrough functionality, developer mode must be enabled on the Oculus PC software. Prior to attempting to run ViRMA Quest 2, it is recommended to establish an Oculus Link connection between the headset and PC via a USB-C cable.

Next, ensure that the `ViRMA_APIController` file in the Unity project has the correct path to the set of images.

Once Unity and the Oculus software is configured, one simply needs to enable Oculus Link and start the Unity software when this is done. There have been no major bugs associated with starting the software, so it should work every time. The software should also function with any standard keyboard attached to the PC.

F Usability testing notes

Starting with the next page, notes taken for all participants during the usability testing can be found.

Subject 1.01

SUBJECT: 1.01		
Datetime		**10:48-11:40** **02/05/2022**
Process	**Minutes**	**Process description (include timestamps if possible)**
Survey	X	24, student, no experience with VR
Introductory presentation		
Test A		**<u>SteamVR</u>** How come there isn't any data. Looks like directory. It's weird that he was looking up on the menu, after clicking tags, couldn't see anything, has to look down to find the keyboard. Typing kind of hard, but gets used to it. Surprised by the dimension explorer appearing, thinks its the same thing repeated. Issue with distance before he can select the button. Searches for dog, notices that the axis is still there. Looks at the menu, sees projected filters, wants to remove basketball player. Clicked on the trash, noticed everything got deleted. Proceeds to project just dog. Searches for animal, is surprised when sees animal tissue first instead of animal, or a list of search results related. Q about interpreting the results. Says first one is parent. Too big space with results. Hint about moving the result to browse them. Plays with the browser, dragging, scrolling. Gets a ton of results, notices the categories. Note that the resulting queries are a lot. Tries to go to menu and limit the results somehow. Plays with moving the menu around. Expects to be able to grab the axis, reaches out and notices that he can interact with it, but can't see what due to the distance. Expects to be able to click and see the categories or something.
Post-test interview (A)		Liked the search results once he figured out how the dragging works (Dimension Explorer). But lack of some sort of boundry on the Dimension Explorer, went on forever. It was nice exploring the Visualization Controller, with 3 axes, the depth perception). Typing could be improved, something you can get used to but it was very slow, especially if you have to enter a longer query. It was a bit problematic that you had to get very close to some things to click on them instead of being able to just point and click. Also counterintuitive having the menus overlaying (Visualization Controller appearing behind Dim Explorer)
Break (optional)		

Test B		**Oculus** Immediately proceeds to move the scroll menu. Reads the instructions Looks at the controllers while reading. Impressed with seeing the room. Thinks about using the keyboard right away. Grabbing the menu is cool. Is confused what to do. Tries to type immediately. Reads the instructions again. Presses enter and initiates search. Searches dog, sees the difference between the categories the dog fits in (parents). Pokes the buttons, but at the same time clicks – doesn't notice poking is enough. Tries to drag and rotate it. Accidentally drills down. Clears the direct filter. Clicked on one cell, looks are the displayed pictures. Tries to drag them. Clicked B gets back to the axis, then expects that one more click would go back again. Projected to y axis. Wants to rotate it, but can't figure out. Hint about thumbstick. Mentions mirror effect, flip fast, text still faces you. Tries to grab the photos. When asked, seeing the environment is nice, gives a safer feeling. Likes the idea that you can put the pictures/menus on real surface. Environment helps. The white space in the previous one could be more confined, might make it better.
Post-test interview (B)		What could be improved? Maybe makes more sense that on valve you could rotate more easily, felt more intuitive. Oculus had mixed controls – rotating with thumbstick. Information menu – different priority, information overload. Press enter to first search is essential for use, should be at the top.
Semi-Structured interview		Passthrough vs white void – maybe if void was limited, had nicer environment could be fine, but for work, at a desk, the passthrough is nicer. Felt uncomfortable to switch hands. Typing method Prefers to type on keyboard- efficiency reason. Downside – blurry, has to sit very still. But still relied on the visuals to type. 3D rendered keyboard would be great. Feeling the real keyboard was also nice, the other keyboard didn't feel tactile. Used to regular keyboard, able to type much faster. Grabbing menus You should be able to grab it anywhere, quickly, not have to hover over specific point. Felt awkward with the scroll menu flipping around, having to fix it all the time. After having queries and menus Second one was better, but mainly because the axis was blocked by the huge dimension explorer. Drilling down is intuitive, but for rolling up he would

Subject 1.02

<table>
<tr><th colspan="3">SUBJECT: 1.02</th></tr>
<tr><td colspan="2">Datetime</td><td>12:00-13:15
02/05/2022</td></tr>
<tr><td>Process</td><td>Minutes</td><td>Process description (include timestamps if possible)</td></tr>
<tr><td>Survey</td><td>X</td><td>24y, software developer, ITU software development student on 6th semester.
No VR experience. (tried once for 2 min 6 years ago)</td></tr>
<tr><td>Introductory presentation</td><td></td><td></td></tr>
<tr><td>Test A</td><td></td><td><u>Oculus</u>
intuitively grabs the controllers
starts out by reading the information menu
Comments A confirm, B go back, makes sense. Correctly identifies index trigger. Reads about enter showing search field, proceeds to try it out right away. Correctly presses enter, searches for car. The search results make sense. Tries to go back with B. Wants to select one of the results. Expects to be able to point at the menu, or ”scroll” through the results. Pulls out information menu to see if he can figure something out. Tries to use hands, presses one button with it but then goes back to using controller.
Looking at Visualization Controller, figures out he can grab it, move, roatte and scale quickly. Says that it’s nice, but the text is not rotating, so it’s difficult to read the labels.
Query: car/ car mirror
click on picture, tries to scroll through the results, can’t do it. Tries to follow the instructions. Confused about which axis is which. When clearing the axis, the Visualization Controller moves to different place, which is not optimal. Cleares both axes.
Turning around of the search result menu is weird. Car to X axis. Search for old man. Tries to get this on Y, but notices there are no pictures. Animal on Y axis, there are pictures.</td></tr>
<tr><td>Post-test interview (A)</td><td></td><td><u>Dislikes?</u>
Lack of colour coding – seeing which axis is which – menu – axis, so you know which to clear.
Turning the stuff around to to read (Visualization Controller).
Feedback for when there is no picture results – otherwise wondering if it’s a technical issue, didn’t press something or something else.
Can’t tell what’s going on the cells – what do the photos contain – maybe preview would be nice – maybe when hovering over one it shows a larger preview.

Searching for tags with keyboard was nice. A bit blury but being used to keyboard it wasn’t a big issue.</td></tr>
</table>

		Information menu not good – wall of text and then it didn't really help with all the things, didn't have all the information.
Break (optional)		
Test B		**<u>SteamVR</u>** technical issue – mismatch between where the controllers are and how they are shown in the system. Fixin the issue… fixed… Small confusion if he's in the software. Gets the menu, immediately proceeds to press tag and notices the keyboard. Types car, tries the button and sees he can move it around. Searches for car, says it doesn't make sense that the "careerist "is first. Tries to move the dim explorer. Finally succeeds. Looks for car, but thinnks maybe it's not there, no clue how far he has to go to find the right thing. Asked about the dim explorer, all the buttons – kind of says the same as before, but is confused what "amphibian" or some other thing has to do with a car, Thinks maybe he can do something with the other tags. Proceeds to project car to X axis. Tries to get the Visualization Controller, after a moment succeds. Tries to rotate as before. Noticed that labels on axis are selectable, but can't interact with it (no children, but the subject doesn't know that). Placement of menu when pulled out was a surprise. Tries to project something else, here buttons are colour coded, so he checks which axis has somehting projected on it, not to overwrite it. Tries to select a picture cell. Can't do it for some reason. Pulls out the menu, notices the projected tags. Clicks on parent of male, has more results. Manages to click on one picture to view it. Noticed that axis has alphabetically ordered tags. Cumbersome to go up all the way to results on y axis if there are very many.
Post-test interview (B)		<u>Dislikes</u> with long lists of results, very difficult to find the thing you're looking for. Might be fine for exploration. Not many particular dislikes, kind of used to the software by now, so controls also feel easier. Nicer that you don't have to move to the physical keyboard, feels unintuitive when using the headset.

Semi-Structured interview		General – passthrough vs void Liked passthrough a bit more. Helps to orient and not feel like you flow in space, where with the white space it made him feel a bit dizzy. Nice to be able to see your environment, especially if you are in the office eg, and can still look at people around you. Typing speed-wise – physical keyboard, general experience – more intuitive and a bit easier with a keyboard that's part of the VR. Lack of connection between physical keyboard and VR – if there was a 3D rendered keyboard it might be better. Hands seemed necessary to use the controllers anyway. Hard to use hands, but there was much to take in, trying still to master the controllers instead of trying out a new thing. Barely got used to using the controllers. If only hands with all the same functionality, hands would be prefered.
Concluding survey		

Subject 1.03

SUBJECT: 1.03		
Datetime		09:00-10:00 03/05/2022
Process	**Minutes**	**Process description (include timestamps if possible)**
Survey	X	VR experience – played some games (used Quest), not a frequent user though age 24 student at ITU, Data Science
Introductory presentation		
Test A	~9:11	**<u>Oculus</u>** Reading through the instructions, trying out buttons hint about the physical keyboard being a part of the software, having to press enter to open search field Not easy to see the keyboard, problem to type Figuring out how to scroll, scrolling, thinking how to press a button. Accidentally grabs the menu a couple of times trying to press a button. Is given a hint about ”poking” the button to press it. Doesn't notice the query at first. Has now arthropods on X and Y axis, wants to project something else on one of the axis. Thinks it might be nice to have the hierarchy, to know what things might be connected. Tries to type a different query. Not sure if she can type right away, tries to use the controller to activate the search field. Figures she can just type. Tries to use hands to scroll through the results Comment again about having access to how the structure looks like (hierarchy). Projects something that has no children, not sure what happens, explaining the situation. Quite intuitive to grab. Pictures (cells on axes) a bit messy – can't see what is on them. Q of how to roll up. Controls not intuitive, tries to poke the tags on axis.
Post-test interview (A)	~9:27	Tried a lot to explore the tags, but it wasn't intuitive (would assume that if you search for transport, you would see the children of transport or something like that). Didn't go back to see menu, lots of information, tried to explore. Remembered that A and B should be confirm/go back, tried and it didn't work – not consistent controls. Handtracking – tried and quickly went back to controllers – though that the hands don't have all the functions, that you probably can't grab

		with them, so used the controllers. But hands still felt more intuitive in a way. Hands would be preferable if they were fully functional. Now she is used to knowing how the controllers work and that they can be used for everything. Keyboard not great – visibility being the issue. 3D keyboard would be faster for her personally. Opinion – lack of feedback and seeing the hands might be harder for people used to typing without looking.
Break (optional)		
Test B	9:37	**SteamVR** Is in white space, searches for menu, some description, anything. Pressed random things trying to do something. Trying to click stuff to start typing/searching. (hint that here you have to always click, which buttons to try). Clicked tags, didn't notice the keyboard at first. Types bus, notices a lot of query results. Confused by the results. Business person is first, then other words next to it which don't have anything to do it with it. Quickly figures out how to scroll through it. Tries to figure out the order/ what are the tags, it's a lot b-starting words. Intuitively bus should be first. Not sure if there even is a bus in this list. So long, she would assume there is no bus. Finds it after knowing it is there, very confused about the order. Identifies parent tag, children and siblings correctly. Projects bus to x axis, tries to reach it and grab it and move it closer. Tries to interact with the text on the axes. Notices the children. Projects group to y axis, there are tags on it but no cells. Identifies that there are no common results. Asked about rolling up. Says it would be easy having the previous result (Dimension Explorer), just apply the parent tag to X axis. Tries to interact with the axis instead. The structure in the menu (projected filters) is not clear (something that was a parent is in line with something else). Prompted to close the keyboard to be able to interact with the Visualization Controller. Is able to select a cell and get pics displayed.
Post-test interview (B)	9:51	very confusing search results, would assume alphabetic sorting, with the exact match at the beginning etc. interacting with search result – not intuitive, not knowing that you have to hide the keyboard first to be able to interact with it. Hints about controls – wasn't sure which buttons should be pressed when. Would be good to have feedback, knowing that something is happening/ there are no results when pressing some buttons (stuff was still highlighted).

Semi-Structured interview		From the previous version – intuitively reaching for the viz query to grab it, but it wasn't possible without removing the keyboard (not clear at all). Keyboard – very straightforward, much better than the previous version. Nicer with seeing the parent, children, siblings, closer to expectations expressed while testing the previous version. (but not necessarily always clear though eg. for business person it seemed more like random words, but because of experience with searching bus, transportation earlier, identified the correct structure right away. Environment the white void had no background noise, but the passthrough you are more sure about the environment. Eg saw the query viz right away (maybe due to expecting it form previous version, but also just more visible).
Concluding survey		

Subject 1.04

SUBJECT: 1.04		
Datetime		**12:00-13:15** **03/05/2022**
Process	**Minutes**	**Process description (include timestamps if possible)**
Survey	X	26, played plenty of videogames in vr, Oculus Quest and playstation VR, itu student, Msc DIM
Introductory presentation		
Test A	12:04	**SteamVR** Looks around, trying different buttons, trying to press tags button, gets keyboard, searches for coffee. Bunch of search results appear, subject wants to hide the keyboard, presses the button, is surprised that the query result also disappeared. Get results again, tries to use thumbsticks to scroll through them. Grabbing by pressing index trigger (is used to do that in VR. Tries to scroll vertically through the results, it's difficult. Moves away from the keyboard. Tries to press a button, accidentally opens a menu. Tries other things and distance, confused by all buttons highlighting. Applies direct filter, prompted to delete it. Confused by a black cell (no data). Thought that just search for coffee would make it be applied as a filter. Figures that the tags next to the "result" are related resets both axes as there are no results.
Post-test interview (A)	12:15	Difficulties: confusion about where does the query start, when is the next part, assuming searching a tag already queries something on an axis, not sure what the next step is. Thinking that the dim browser that appeared is things related to coffee that can be then applied. Didn't see the browser as separate column but as one thing. Confusion about which was the main menu expectation to be able to grab the query viz, not being able to do that (need to hide the dim browser and keyboard).
Break (optional)		

Test B	12:21	**Oculus** reading the information menu weird seeing both the VR and real version of the controller toggles the menu, tries different controls thinks it's cool to use just hands grabs the search results menu, thinks it's weird how it rotates. Wants to do the search, opens info menu to search for clues. Presses enter, notices the search bar. Frustrated with keyboard, can't see the keys well, is used to a different layout. Successful search. Tries to grab with hands, think should be able to do so (info menu says that). Tries to scroll with index finger, correctly, easily presses one of the results. Correctly identifies pressing project to apply it on the X axis. Is not sure if it was projected, tries to grab the Visualization Controller with a hand, accidentally clicks a cell and Timeline is open. Is confused how to go back, tries to scroll through the pictures. Pressed B (trying stuff), goes back, sees on the help menu that it was meant to do that. Tries to move the help menu, can't do that, proceeds to make a different query to apply to the Y axis. Expects that the axis which has something projected on it should be highlighted (in the menu), to know on which other to apply in case he forgets. Proceeds to grab the Visualization Controller with a controller. Notices that the tags on the axis are being highlighted, expects to be able to do some action. Pressed one tag, thinks it was a wrong button, because the Visualization Controller jumped back to the original spawn place (expects to have it still in the same place).
Post-test interview (B)		Confused about grabbing, messy because you have to keep switching between hands and controllers. Not sure if using things correctly, or there is the need for switching. Using hands preferred, more fun, like using the phone on a daily basis, more intuitive, no need to use the buttons. Weird grabbing (wherever you grab it, it jumps to a certain point).
Semi-Structured interview		Using physical keyboard – liked it better, it seems slower (but only compared to daily use of physical keyboard, frustrated at doing it slow despite being used to being good and fast at it). In a sense then more frustrating than using the software version. (if seeing the keys not issue – physical definitely preferred). Generally enjoyed that part, was a strange mix between VR and physical world, sitting in front of the screen (coincidence), thinking that something that appeared on the screen was part of the experience (confusing because of the passthrough). There is need to use the keyboard, so it seemed that screen might also be part of it.

Subject 1.05

SUBJECT: 1.05		
Datetime		**13:30-14:45** **03/05/2022**
Process	**Minutes**	**Process description (include timestamps if possible)**
Survey	X	38, student ITU Software Design No experience with VR (tried it 20 years before)
Introductory presentation		
Test A	13:45	<u>**Oculus**</u> holding the controllers feels weird. Looks at the instructions, tries controls, pressing buttons, says it's not clear what he has to do. Reads the menu some more. Thinks it says nothing about how to make queries. Hint to press enter. Sees the search bar, not sure how to interact with it. Confusion about controls. Tries to type, can't see the keys. Searches for dog. Tries to figure out how to interact with the search results. Pressing buttons, unsure how to interact with the menu, goes for the mouse, thinking it's part of the software perhaps. Thinks that pressing the trigger button actually pressed the button (another time as well). Projects dog on the X axis. Thinks what should be the next step, assumes maybe trying to search for a new thing and apply it on another axis. Does this. Thinks that because there is nothing to see there, there are no results (correctly). Correct interpretation of not having any results for this particular combination. Weird and awkward with menu rotation when you grab it. Tries to figure out how to rotate it. Opens info menu, reads how to do it. Rotates and scales but there are some issues (not able to rotate it the way he wants to). Prompted to use hands, says it's easier to interact with the menu with fingers.
Post-test interview (A)	13:59	Generally issues with controls, has an idea what to do but not able to figure out what controls to use. Wanted to grab and rotate the Visualization Controller same way as menu but it wasn't possible. Memorization of keyboard keys helped typing, wasn't a big issue due to that. Without it would be tricky. Says an overlay of the keys, to see them clearer would be much better. Skimming the big test, prefers to just try to figure it out, things should be intuitive enough. Says definitely using hands was much easier. Lack of feedback when clearing the axis was a small issue, had to look at the query viz to make sure the action actually happened. All things equal, hands would be preferable. Especially when working a lot with it. Things that manipulating with hands could be easier to manipulate rotating, Personal preference: hate for thumbsticks.

Break (optional)		
Test B	14:08	**Valve** presses some buttons, menu appears. Tries to press a button. Pressed tags, happy with the keyboard in the software. Thinks the controls are better. Searches for train. Figures out that A button confirms choice. Manages to move away the keyboard. Applies trainer to X axis, thinks it's train. Assumed train would be the first result, small confusion. Tries to figure out how to grab and move the dim browser, tries different buttons to grab and move it. Opens up menu cause of pressing different buttons. Thinks that maybe there should be the arrow like on the menu and keyboard to intuitively move the dim browser. Finally manages to move it. Says there seems to be two of the same exact results in the dim browser. So much easier to search with the in software keyboard. Searches for tree. Wants tree, but some other result is first. Finds tree, applies to Y axis. Tries to grab the Visualization Controller, it's very long, unable to do that. Opens the menu. Presses train and tree in projected filters. Is hinted to hide the keyboard. Then grabbing the Visualization Controller is possible, succeeds. Says it seems that there is no query that matches both tree and train. Figures out how to roll up, rolls up to public transport. Presses one of the cubes, amazed by the images. Pressed b to go back to the Visualization Controller. Thinks this is so much easier to control. Rolls up in the tree hierarchy. Keeps rolling up, exploring the hierarchy. Thinks that it's much more intuitive to interact with this.
(Post-test interview (B)	14:22	It was easier to navigate, felt more responsive. Grabbing and dragging was unresponsive at first, so he thought he couldn't do it (then hinted to remove keyboard), that it was possible then. The keyboard was much easier to use. Main menu – thinks it was super clear (told not to use the other keys). Easy to access, super responsive. Easier to navigate. Comparing to the other one which was not responsive, had to wait (note: the actual issue being thinking that you have to press a button, when it was actually poking, which happened accidentally). Ordering of the search results was weird. If you search for train, he wants to see train first, other things could follow, but exact much should be first.

Semi-Structured interview		Highly prefers the second implementation controls, more responsive etc. Environment comparison: prefers the second environment (white void), thinks that passthrough, seeing things was not a big value, other than knowing that he won't bump into anything. If knew that he wouldn't bump into anything, having safe space, definitely would prefer the white space, being fully immersed. Thinks now that seeing the environment around distracted him from using the software. Full hand tracking vs second (preferred controls): the weirdest thing about the first controllers was the snapping, rotating when grabbing (yeah fucking dumb). Thinks that he would prefer hands, which would be less frustrating than using the controllers. As for now, the second version's controllers have no need to be replaced with hands, because they worked so well, so responsive.
Concluding survey		

Subject 1.06

SUBJECT: 1.06		
Datetime		14:00-15:15 04/05/2022
Process	**Minutes**	**Process description (include timestamps if possible)**
Survey	X	35, researcher/educator in IT highly experienced in VR, frequent user (videogames), using Rift
Introductory presentation		
Test A	14:15	**<u>Oculus</u>** Participant notices both menus. Reads the information, toggles the menu, moving it to a more comfortable place. Looks away and toggles as a way to move it around. Comment about people usually not reading the information menu. Takes time to read it properly. Thinks aloud, guessing that the keyboard needs to be used to search for tags. Presses enter, search menu appears. Picks up a keyboard. Search bar spawned at a weird position for the participant, he picks up the keyboard to try to reposition it. Sees (in the information menu) that ESC hides the keyboard, proceeds to reorient. Is bothered a bit with the tilted search bar. Reason for pickup was an uncomfortable positioning of the search bar (due to having to lean across to press enter). Comment that it's really hard to see the keys. Searches for martial arts, but nothing appears. Participant assumes correctly that there are no results. But indicates that without his prior experience he might be uncertain if something happened at all. Dog search. Assuming such simple thing has to be in the dataset. Quickly identifies that the dot at the end of controller interacts with the menu. Talks about consistency, being briefly confused of how to interact with it. Small mismatch of expectations at the beginning (very brief). Slightly confused what the task is. Presses direct filter. Is prompted to clear. Notices that something still happened. (hypercube with direct filter applied – all pics). Not sure what happened. Confused about the box. Brief talk, let's project dog to one axis. Is asked about understanding the difference between dog (domestic animal) and dog (canine). Gets the difference by the way results are presented. Is prompted to project dog (domestic animal). Confused about how to do it. Small hint is given, projects dog to X axis. Tries to project again, thinking that maybe he can reposition it the same way it was with the information menu and the search bar.

Post-test interview (A)		Tried right away to grab with the Right grab button (intuitive thing), didn't try again. Very quickly figured out hiding and spawning the things to reposition. BIAS ALERT – knowing this is early stage development, thought that grabbing, although should be possible, maybe is not developed yet, hence he didn't try again. Professional comment – info menu should be further developed. Advising that step by step tutorial would be optimal, presenting one thing at a time, small tasks to teach how to interact with things.
Break (optional)		
Test B	14:40	**SteamVR** Intuitively holds out hands for the controllers, adjusts the cable. Looks around, comments that it's just a white environment, not sure if the app has loaded. Notices controllers, tries to press different buttons. Opens the main menu. Tests the range of the balls at the end of controller, tries to interact with the menu the same way as in Oculus (poke), then presses index trigger to click button, next tries A and keyboard appeared. Again tries the thing with moving with index finger (pressing button). Figured out after trying that A is the one doing the job. Notices there are no instructions like in the other one. Plays with the basic things – establishes that B opens the main menu, A allows to interact with things. Comments that it's nice to be able to grab with both hands (each hand one menu) and manipulate each separately, doing a setup. Searches for dog. Notices the dim browser, moves around to adjust the position, moves away a keyboard to see the dim browser better. Says it's nice that he can see the actual hierarchy. Comment that if you don't know the tags, this gives a lot of information about what to look for. Asked about the understanding of the columns in results – yes he does. Three categories (canine, domestic animal, dogfish shark). Then siblings, then breeds of dogs. Assumes that if pressed one sibling – domestic cat – it would maybe show the children of that. Projects cat and dog on X and Y axes. Notices he tries to move the Visualization Controller but it's not responding. Says it's a problem where it appeared (behind the dim explorer), which makes it problematic to see. Tries to interact with it, with button. The photos highlight what makes him think that it should be possible to interact with them. Is then hinted to remove the keyboard – press the disappear button on the keyboard). Then possible to play with the query. Pressed one cell. Tries to move the pictures, can't do it. Then succeeds, comment about inconsistency – grabbing this part is done using a different button then the menu/keyboard before. When asked about relation, notices the

		dates, the pictures are in order by date. Thinks intuitively to press B for back. Goes back to the Visualization Controller. Moves the menu, notices the projected filters, notices the different colors, after a moment remembers the colored axis, associates them with the colors of axis – color coding – but had to think about it for a second. Tries index finger again to move things – intuitive. Keyboard pulled out, notices the dog still in the search bar, wondering how to search again, comment that because there is a green tick you would have to press the orange one first, but you can actually just press the green again to search.
Post-test interview (B)		Cool thing about VR interactions in real world are reflected there – hence the intuitive trying of using the index finger or the movement simulating actual grabbing. Later the steamVr version did do what is ”intuitive”, but there was inconsistency. With the other menus, A worked, when tried to grab the photos with A, assumed, that grabbing was impossible, because it worked for the other menus. Also didn't think to try to move the dim explorer (again mostly due to inconsistency and the learned controls not working with the other things). Also very weird with seeing a feedback from the Visualization Controller when the dim browser open, but not being able to interact with it. In VR the experience is not like window open on a window (not layer upon layer), you can move and then they are separate, when you change the perspective. Question about the layers – would you expect the most recent element to be the top layer – yes, that's what you're used from other things. This would make it more intuitive. Jonas - Consistent issue of layering in both version, with the query viz showing behind other things.
Semi-Structured interview	14:53	Environment question comparison Regardless of all other things, just the environment void vs passthrough – for exploring, research – Valve version preferable, due to more focus on the actual thins you need to interact with. Valve index again, because the hierarchical context is very helpful. If I don't need it it's still there but doesn't get in a way. Input question – situation when you do a lot of searches If keyboard was clearer, much rather would use the keyboard. If blurry – don't know, suspects would want the physical keyboard. Own keyboard also would make it easier – knowing where the keys are. So still would prefer this keyboard. Question about controllers – didn't notice in info menu that hands are an option, but also very familiar with controllers. Felt he didn't have time to play with the hands. (also not familiar at all with hands tracking in VR). So didn't try now due to time constraint.

Concluding survey		**THESE ARE FACILITATOR'S ANSWERS AND NOTES CONDUCTED AFTER TEST CONCLUSION. See section "Set-up or test errors" for description** I preferred the controls in the Index implementation: 5 – due to following standards set by VR better Being able to see my physical environment was beneficial to my experience of the software: 2 – prefers SteamVR void due to increased focus, but only for research/exploration I preferred using physical controllers to hand-tracking: 3 – cannot make a conclusion due to not using it at all If I had to use ViRMA for work or personal enjoyment, I would prefer using the Index version over Quest 2: 5 – based on above answers the participant illustrates a general preference for the SteamVR implementation.

Subject 1.07

SUBJECT: 1.07		
Datetime		**15:30-16:45 04/05/2022**
Process	**Minutes**	**Process description (include timestamps if possible)**
Survey	X	22, Data Science student, working part time, no experience with VR
Introductory presentation		
Test A	16:00	**SteamVR** pressed something and the menu appeared. No idea what she is supposed to do. Thinks that pressing B was approving the button. Tries to press the tags button with the left controller, which doesn't work. Tries to press index trigger, doesn't work. Figures out that A is the confirm button. She wants to get keyboard away, so presses the red button, everything disappears, gets it back. Wants to move the keyboard but doesn't know how. Asked about the results, says that think they are related to her query. Hinted with grabbing and moving the dim explorer. Is a bit confused about the results, why some words are repeated. Is confused why everything is highlighted when hovering over one of the siblings. Thinks they look like buttons, feels that should be able to press them but can't. Grabs the main menu, moves it around, thinks it's cool. Searches for ball, wonders if she can have multiple tags. Says it looks like she has a different set of tags. Not expected to see the words related to ball, thought that maybe things related to both ball and sports. Confused about the result. Noticed that they contain the string "ball". Is prompted to maybe look for a dog and try to press the buttons by moving closer to them and try to project. Projected to Y axis, tries to get behind to see the results clearly, doesn't know how. Asked about the ordering of tags in the dim browser. Identifies the general tag on the right (parent). Identifies siblings and children correctly, just naming that something different.
Post-test interview (A)	16:14	Difficulties with controllers, noticed that things were highlighting when hovering over it, took some time to figure out you have to get really close to be very precise. Confusing feedback in dim browser. - column highlighting, but not able to interact with it (thinking you should just click it). You have to get closer to actually get a different level (only one button highlighted). Feedback too early. Thinks it's unintuitive that the menu and keyboard would move in one way but the other things in a different way (index trigger). Would expect consistency.
Break (optional)		

Test B	16:21	**<u>Oculus</u>** proceeds to click buttons, toggles the information menu, proceeds to read the information menu., commenting she's not gonna remember it. Grabs the menu and moves it, says it's weird (doesn't rotate). Expects to be able to rotate it with the thumbstick but it doesn't work. Figures only the right controller can grab. Grabs the search results which moves as expected, is intuitive. Annoyed with inconsistency. Thinking she should maybe use the physical keyboard for the query searching. (says that the info earlier that she will ned to use it helped her figure it out, otherwise would think about it). Proceeds to press enter, says the keys are a bit blurry but types dog and sees the search results. Thinks it's annoying that when she grabs the menu it flips. Pulls out the info menu, reads information. Gets rid of keyboard by pressing esc. Pressed one tag in search menu, think it was because she pressed a when hovering over it. Grabs the Visualization Controller, immediately rotates it, says now it behaves the way she expected. Projects dog to the x axis. Searches for ball. Says it's nice with the specification (in the parenthesis). Results more in line with what she expected. Tried to project ball on y axis. Confused about disappearing result (Visualization Controller moved away.) there were no results, she's confused about lack of feedback. Thinks she should have some if there are no result for her search. Keyboard – not sees clearly, relies a bit on memory. Searches for cat. Wants to project it to an axis, doesn't remember which one is which. Proceeds to project to x, kind of correctly identifies it. Prompted to use hands with the search menu. Scrolls, presses buttons, says it's just like a touch screen, easily and intuitively can navigate through it.
Post-test interview (B)	16:35	Mismatch of expectations (info menu vs reality) and also as the info menu said you should be able to rotate. Had an expectation from using Index version before that there will be no consistency of controls. Didn't expect that you could use both hands and controllers, since you use the controllers for everything. But hands were somewhat easy to use once there was knowledge that it's possible. When using the keyboard would be better if the keys were visible. Wasn't seeing clearly what button is being pressed., seeing that she's on a button, but actually being in between.
Semi-Structured interview		search results. In the first one she really like that she could see the context of the search result (parents, children). But the order of the results shown was better in Oculus, what she sees met the expectations. Hands would be preferred if they were fully functional. Environment preference – depends on what you have to do – the white void had the advantage of being able to see the elements of software more clearly. Passthrough environment made her feel much safer.
Concluding survey		

Subject 1.08

SUBJECT: 1.08		
Datetime		**14:20-15:20 02/05/2022**
Process	**Minutes**	**Process description (include timestamps if possible)**
Survey		26, Msc student at CBS, no prior experience with VR
Introductory presentation		
Test A		**<u>SteamVR</u>** Is confused about the white void, not sure what to do, thinks about just trying the buttons. Presses different buttons, the menu suddenly appears, then she tries the buttons slowly to figure out which were responsible for getting the menu up. Then tries the other buttons but the others do nothing. Tries to interact with the menu buttons but fails initially. (tries just poking them) Then click the tags button but doesn't initially notice the keyboard. Appears the menu again, then after a moment notices the keyboard. Thinks that she should try to type some tag. Types knit and searches (without bigger issues). Is a bit overwhelmed with the results, also gets a bit dizzy from the "heights". Looks at the results, reads the tags, then tries to reach over to the rest of the results. Notices that there is another button she can try to use (index finger). Tries interacting with the dimension explorer, after a moment figures out how to move it. Tries to project one to an axis but is not close enough to the button (but gets feedback – the column highlights). Is stuck with this, so gets a hint to get closer to the button. Projects to X axis but there is nothing there (no children). Is encouraged to try a parent instead. Projects to X axis, the Visualization Controller appears. She's a bit seasick again because of the "freezing" of the app while the Visualization Controller loads. Tries to reach for the visualization controller, gets a hint to remove the keyboard first. Very instinctively grabs the visualization controller right away and pulls it closer. Says it was natural and easy, having the previous experience of using the index trigger and hand movement to move the dim explorer. Looks a bit at the results, accidentally clicks on one cell and is a bit surprised when something happens. Immediately starts browsing, moving the pictures. IS disappointed that the time is up.

Post-test interview (A)		Says that it taken some used to and figuring out, but when she discovered that the index trigger and hand movement could move things, it was natural to use it for pulling the visualization controller closer. No actual thinking involved, just an intuitive reaction, intuitive expectation. Thought that when the hover point was visually on the button, she should be able to interact with it. Not clear how close to the button she had to get due to the feedback beginning too early, though she maybe had to try some other angles etc. The white environment made her feel very uncomfortable, if there was an imaginary floor it would make it better, but then also with the large, high dim explorer she felt as if she was floating very high. She was a bit uncomfortable with the menus being so close, but because she managed to get further with the software, she didn't try to figure out how to move the menus. Would assume that it should be the same way as with the other elements (index trigger and just moving, not interacting with one specific button).
Break (optional)		
Test B		**Oculus** The information menu is positioned a bit awkwardly, she starts with hands and immediately tries to grab the menus (which doesn't work). Proceeds to adjust and read the information menu, figuring out the information, trying the different buttons. Grabs the search results menu and moves it around. Starts typing, has some visibility issues. Is confused whether she should see what she types (she didn't appear the search bar). Then tries searching by pressing enter and sees that then the search bar appeared. Types a query, saying she relies more on her brain rather than visibility of the keys. Notices that there are results, grabs the menu, exclaims annoyance in the menu flipping in a weird way. Meanwhile opens the information menu a couple of times to check if she can get some clues. Tries to project something. Initially tries pressing the X word instead of the button. Then thinks that she maybe should press the project button. Each times she does the poke motion in the direction of the button, but also presses the A button, so it takes a while to project but it does happen. Examines the axis, accidentally clicks on a cell. Looks at the pictures, thinks she would like to project something else to another axis. Wonders for a moment how to do it, proceeds to put the controllers down and type another query. Same story with projecting to the Y axis. The visualization controller is hidden behind the pictures from clicking the cell in a previous step, so it takes a while to hide them and look at the query. She wants to grab the visualization controller but does it with the hand trigger and doesn't manage to do it.

Post-test interview (B)		Passthrough – had stronger sense of where she is, didn't feel dizzy when looking down. Dealing with the physical (keyboard) an virtual was more complex to her. The screen was a bit confusing, and because she had to use the physical keyboard, that made her think that maybe eg the screen is also part of the experience. If the keyboard was 3D rendered, it would be clear that it's part of the software and would make the experience better. It was also a bit confusing with the elements of software sometimes being "in front" of the keyboard – a bit unclear than if it still should be used or not (no 'connection' with the keyboard).
Semi-Structured interview		It was simpler to use only the virtual environment – no confusion as to what is part of the software, or how the tasks should be done. Minus was not knowing anything about the surroundings. In short – white thing: 'worse' idea but better execution, but the passthrough is more comfortable but the execution could be better (3D rendered keyboard). The disadvantage of white could be mitigated but having some sort of virtual floor. Hypothetical 3D rendered physical keyboard vs virtual keyboard – the virtual keyboard for these tasks here (typing very short words) would also be fine, for more work, where more typing – 3D rendered physical. And in general the 'perfect' 3D might also be preferrable.
Concluding survey		

Subject 2.01

SUBJECT: 2.01		
Datetime		09:00-10:00 11/05/2022
Process	**Minutes**	**Process description (include timestamps if possible)**
Survey		23, student job, master in computer science student very little experience in VR (tried it a few times – also Oculus)
Introductory presentation		
Test A	9:17	**<u>Oculus</u>** User is introduced to the software. Presses enter to get the menu. Says that the keyboard is not exactly readable. Searches TV, tries to pinch to grab the search results menu, but it's difficult, finger interacts with the buttons, but can't grab. When asked about results, recognizes that the word in parenthesis is a parent. Projects television to X axis. Grabs the query viz and moves it around. Says there is some getting used to (accidentally grabbing is activated and grabs the Visualization Controller involuntarily. Presses on the cell, gets the pictures shown. Intuitively would like to move around to look at the stuff, but the boundary is limiting. Says he would feel safe moving around because he can see the environment. Is not sure if he can interact with the pictures due to prior experience with grabbing the Visualization Controller but not the scroll menu (which should be possible). Would expect to be able to grab the photos and move them to a more comfortable position. Is given controllers. Expects that he should be able to aim with the controllers. Pressed B, guesses it must be back, has the query viz back. Comments that it only works when you go back from the group of pictures. Tries other buttons, presses A. Manages to grab the query viz with right controller, comments that it's different from hands – there the left hand was used, would expect the left controller to work to.
Post-test interview (A)		Prefers the hands, more natural interaction when moving away from keyboard. Once you search you can go directly to exploring the query. It would also be natural and intuitive to have the freedom to walk around the environment more freely, walking to a given part of software would be nice. Despite having some trouble with using hands (not being able to grab when he knows he should), he still prefers the experience with hands (much more, breaks the barrier, more interesting experience). If both things were equal, of course hands preferred as well.

		The passthrough environment was very nice. Keyboard was a small issue, not visible keys, but participant was used to typing so did it easily. Comments that a 3D keyboard would be nice – not an overlay, as he says different keyboard can have different layouts. But in general virtual keyboard would be nicer – completely moving from ”the physical world” might be a nice experience.
Break (optional)		
Test B	9:35	**<u>SteamVR</u>** Let's search for something. Comments that he sees nothing, is in white space. Tries out the controls, gets the main menu. Presses the wrong button, then tries again. Comments that maybe he needs to get closer ot the button to click it properly (correct assumption). Manages to get the keyboard. Searches for tree. Says that he wouldn't assume to have tree – plant first, but sees squirrel. Moves the dim explorer, notices the tree (plant), says he would assume this would be the first result. Scrolls through the children, has some small issues which he identifies is due to rotation of wrist. Wants to project oak. Thinks that maybe he can't because it doesn't react. Is hinted that he has to move the controller extremely close to the button. Projects to X axis. Tries to grab the Visualization Controller, tries a few times to move closer. Proceeds to get the keyboard and close it (says he remember this from the walkthrough hence he tried). Reaches out for the Visualization Controller, tries to grab it but accidentally rolls up. Proceeds to scroll with the controller, exploring, then drills down to oak. Interprets the numbers next to tag names as how many tags are in this given group.
Post-test interview (B)	9:45	Questions about pressing Hard to say how far you have to go unless you see the feedback, you have to move really close. In his experience if you have a controller you usually have some pointer which lets you reach things that are further away without you having to move over. Doesn't mind moving close when he can interact with hands – but when controllers are there he would prefer not having to do that (separating the 2 experiences). Environment – white void a bit weird experience, not being used to seeing this white environment. Thinks the white void goes to more prolonged experience in software – you are completely in the virtual world. You focus more on this thing. The environments each serve their purpose, thinks it depends on the usability. Nice to jump in quickly, quick search – passthrough nice, more extensive, longer use – better white void, emerge yourself in the virtual environment. Dislike of using controllers in ”a physical” way – why move close and stuff, when you could just point, press, using thumbstick instead of

		moving your hand with it – expectations for when controllers are to be used.
Semi-Structured interview		Would in general prefer using the hands over controllers, even if the controllers would behave as expected. Theoretical questions about both modes used in any of the 2 environments – still very much likes the idea of handtracking. Thinks it's much more natural to use hands and why wouldn't you use them if you could. Again even if hands were slightly worse in use than controllers, he would still choose the hands. Question about the keyboard – preference He would prefer a render of a physical keyboard. If you were walking around, the render of physical keyboard is more limiting in a way that if you want to search again, you have to go back. Virtual keyboard (original virma) can be just pulled out anywhere. Not really concerned with physical keyboard with perfectly visible keys vs overlayed. Is not concerned with having a slower typing speed, because how much are you going to type.
Concluding survey		

Subject 2.02

SUBJECT: 2.02		
Datetime		**10:30-11:30 11/05/2022**
Process	**Minutes**	**Process description (include timestamps if possible)**
Survey		25, student, msc Data Science not much experience with VR, tried it once/twice
Introductory presentation		
Test A	10:43	**<u>SteamVR</u>** Presses some buttons, manages to get the man menu. Tried to press the tags button, but pressed the button that hides the menu. Doesn't get close enough to the button, is prompted to get closer. Moves the controller closer and notices the change of colour. Tries again, hides the menu, tries other buttons, finally keyboard appears. Types car, is confused about the result – sees careerist first. Not sure if car is there. Is prompted to find the car, informed that it is there, notices the rest of the dim explorer. Tries to drag it with different buttons, succeeded. Scrolls vertically, then again drags the dim explorer. Hard to find "car". Give up, search for dog. Not sure what she should do. Clicks on the dog and projects to the X axis. Moves the dim explorer away, tries to grab the Visualization Controller and pull it but it doesn't work. Is prompted to go back to keyboard, grabs it on the button, moves it closer, hides it. Grabs the Visualization Controller, moves it around. Is asked about rolling up, does it in the main menu. Rolls up to animal. Is asked to search for something else to project on another axis. Clicks on tags, gets the keyboard. It overlaps with Visualization Controller. Moves it away, quite quickly types. No results (small confusion, doesn't see anything). Different search, projects Male to Y axis. Tries to interact with the cell, thinks she should click it. But the dim explorer is still there, is reminded to remove the keyboard. Clicks the cell.
Post-test interview (A)	10:55	Mismatch of expectations, not getting the depth – need to move closer to interact. Compares to playing game, where you just have to press buttons on a controller. It was more confusing to use the face buttons. Grabbing with index trigger was quite intuitive. Getting main menu up and down a couple of times due to not being sure if this button now maybe does something different. It was easy to use the virtual keyboard, thinks also because then she knew that she has to get close and press.

Break (optional)		
Test B	11:00	**<u>Oculus</u>** Presses enter, tries to type a word but the keyboard is blurry. Searches for car. Tries to grab the menu with the right hand. Remembers that it's not this hand. Grabs it with left hand, says it moves in a weird way (lagging a bit). Involuntary movement of the menu (pinch recognized when it was not intended). Easily presses and projects car on X axis. IS prompted to search for something else. Is not happy, has trouble seeing the keyboard. Searches for plant, projects plant (organism) to Y. Just remembers that car was on the X axis. Has a bit trouble moving the Visualization Controller, accidentally grabs it a few times. Is asked to roll up on car. Sees that interacting with index finger the stuff on axis highlights, thinks that poking it with the finger should work. Is hinted to pinch, does it immediately with an immediate result. Accidental grab again (trying to interact with cell with left hand). Then gets that it works with the right hand but not the left. Is prompted to get rid of plant form Y axis. Grabs the menu (comments again that it lags). Clears the Y axis with no issues. Is asked to search for beverage, but is intimidated with how long the word is and that it's hard to type. Searches for road instead (again small issue typing). Projects road to Y axis. Plays with grabbing the Visualization Controller, lagging again. Wonders if she can see the photos, tries to poke one of the cells. Remembers pinching, pinches it and succeeds in viewing the photos.
Post-test interview (B)	11:10	Keyboard big issue, kind of remembers the keys but still had problems, misspelling words. Grabbing also felt a bit problematic, did the gesture with pinching left hand without thinking, and the thing suddenly teleported in her face. It also was lagging sometimes when dragging. Environment – a bit distracting to see the surroundings when looking eg. at the pictures. Usually would be distracting – when you're in a new place that you don't know, you look at other things as well.
Semi-Structured interview		Preferred to use the controllers of the steam vr version, due to handtracking being to confusing (lagging, getting a thing suddenly in your face). Hypothetically, if everything worked perfectly, hands would be definitely preferred. If keys on keyboard were well visible, the physical keyboard would be preferred. All in all, visibility is the only issue. Other than that, real keyboard wins.(Frustration was due to familiarity with the keyboard and not being able to use it properly).

		Thinks combination of the two search results might be better. Search results menu too small and not enough information. Dim explorer too much stuff, you have to drag it for so long to get it. And of course the order of results – search for car, but then have to scroll for idk how long to actually get to car.
Concluding survey		

Subject 2.03

SUBJECT: 2.03		
Datetime		**12:00-13:00** **11/05/2022**
Process	**Minutes**	**Process description (include timestamps if possible)**
Survey		24, student Msc Data Science has a VR headset (HTC Vive), quite experienced with VR
Introductory presentation		
Test A		<u>**Oculus**</u> At the start thinks it's a bit weird – new experience to be able to see the environment. Searches for cat, tries to project to X axis. Didn't press the button properly. Searches for dog, projects it to 2 axes. Clears the Y axis. Again searches for cat, projects to Y axis (says is not good at pressing the buttons). Rolls up twice on Y, gets to carnivore. Wants to see pictures, presses one cell. There is one image, it appeared quite far form the user, he immediately goes to try to grab the image with left hand. Is given controllers (only way to go back). Presses B quite intuitively. Tries to drill down, after one midclick finds out that the X button did the click. Tries to grab the Visualization Controller – but tries a few times with the left hand (knowing from experience with hands). Is hinted to use the right controller, then quickly figures how to grab with that. Projects something to Z axis. Explores cells, submits some cells, goes back. All easily
Post-test interview (A)	12:25	Intuitively tried to grab with the right hand, remembering the intro started using the left. Then after switching to controllers went for consistency. Once grasped, it was easy to use the controllers (easier than hands – sometimes hard to hit what you need with hands). Keyboard was not a big issue, knew where the keys are, is used to using the keyboard. If hands worked "perfectly" - what would be preferred? Depends – hands are much cooler to use, but as a person who is very used to using controllers – they are more intuitive to use. Familiarity a factor. But would like to use hands if they functioned completely, just needs some getting used to. Passthrough – nice experience, felt that he was still in the room. Didn't find it distracting (there is not much in the room, maybe if there was more could be more distracting, but here it was easy to distinguish the elements of the UI from the surroundings).

Break (optional)		
Test B		SteamVR Prefers to stand up, move away from table – can't see anything, is afraid to hit something (is used to that). Starts by pressing different buttons to see what they do. Figures that y is the only one that does something (gets menu). Presses tags, searches for building. Move the dim explorer. Is a bit confused about how to project (tried to poke). Then presses the button and projects building to X axis. Searches for traveller, goes fast with typing on the virtual keyboard. Projects traveller to Y axis. Wants to move to the Visualization Controller, but can't, is afraid to move and hit a table or something. Is hinted about removing the keyboard and dim explorer to interact with the Visualization Controller. Says it's intuitive to grab it and pull closer, but less intuitive to move it on the other axis with left hand (each hand serves different purpose). Clicks one cell, sees the pictures, Very intuitively goes back. Rolls up on one axis, gets a massive query. Has motion sickness and afraid of hights, so this gives a bit uncomfortable experience. Prompted to search for bicycle. Gets the main menu, types quickly. Is surprised by the result that was not bicycle but bicycle pump. Projects that to Z. Rolls up to wheeled vehicles.
Post-test interview (B)	12:42	Controls – very similar to VR experience that he knows. There was just with two hands where one was limited to doing this and the other that that was less intuitive, mismatch of expectations there. (assuming both controllers should do the same thing). Especially that for the most they did. Dim explorer – intuitive with the column ordering – this is subcategory of this, siblings are here – but a bit overwhelming – you're looking for this one thing but then you get a huge column of some other results. IT was fine with searching eg. the bicycle because there was not much results in general. If there was 100 that would be too bothersome.
Semi-Structured interview	12:47	Environment – it was nice with passthrough because you knew you wouldn't hit anything. It's always a bit uncomfortable when it (no passthrough) loads and you're in black void. Behaviour was also more loose, cause he knew where he was moving etc. So passthrough overall preferable. Also about the motions sickness and such – passthrough would make the experience much better in this regard – you know that you're standing on the floor and it's there, compared to not seeing anything and feeling like you're high up/floating. Hands vs controllers – enjoying hands but if wants to be efficient, controllers are faster to use. Thinks it's also easier to just use hands for people who don't use controllers much. Hand tracking with physical keyboard with perfect 3D overlay vs controllers and virtual keyboard – physical keyboard – it's faster, intuitive, don't have to look around for the keys. Moving hand to type on virtual keyboard is cumbersome (?).

		Search results menu vs dim explorer – search results menu is not as overwhelming, but the dim ex, has more context. SRM sometimes had trouble to see what was written. When it was full wasn't sure if there was more to scroll and back then didn't test that. But seeing the context with siblings etc was a good experience, maybe would be better if it was slightly more limited.
Concluding survey		

Subject 2.04

SUBJECT: 2.04		
Datetime		**13:30-14:30** **11/05/2022**
Process	**Minutes**	**Process description (include timestamps if possible)**
Survey		21, student – Bsc Software Development, 4th semester Has VR headset at home (Rift)
Introductory presentation		
Test A	13:48	**<u>SteamVR</u>** tries to get the menu – clicks some buttons, get the menu quickly, immediately goes for tags, clicks it right away without issues, types car, searches. Doesn't know what careerist has to do with cars. Starts browsing the dim explorer, expresses surprise over the results, can't see the car. Thinks it's maybe the tags where car is part of the word. Identifies the parent and it's subset (children). Tries to project car, can't do it. Somehow clicks on bloodmobile. Doesn't know what happens, but quickly figures out that you have to move the controller closer. (is used to having a laser pointer). Not sure how to project. Tries to move the keyboard away (the same way he moves the dim explorer). Manages to get the right interaction. Is hinted to get close to car and try to press it. Intuitively tried to press it with index trigger, but then tries A and succeeds. Project's to X axis, can't grab the Visualization Controller at first. Then tries things and manages to close the keyboard and dim explorer. Right away pulls the Visualization Controller close. Explores the tags, clicks cells, goes back and forth very quickly. Is prompted to project something to Y axis. Wonders how to go back to searching. Thinks maybe needs to the main menu, doesn't remember how but gets it quite fast. Searches for building (when typing tries to use index finger intuitively to submit key). Projects building to Y axis. Explores some more. Drills down on government building. Explores the results, thinks they are pretty cool. Tries to drill down to something on X, but can't (no children). But notices the difference – that there is nothing there as no extra text appears (number of children).

Post-test interview (A)	13:59	Mismatch of expectations when it comes to interactions – tried index trigger almost every time to press something. Then sometimes it worked, but sometimes you had to press A. (lack of consistency) Expectations of standard VR interactions vs how it was. Also not used to having this ball at the end of controller – used to having a laser pointer. Thinks these wouldn't be an issue if he wasn't used to the controls in VR. When trying to drill down, there was no immediate feedback, he was pressing the keys and accidentally rolled up. But then didn't know exactly what he did, had to look at the tags to figure out. Confusing on the axis – some things had number, some didn't figured that it means there is nothing there and that the number indicates the number of subcategories (children). The fact that when you hover over those which have no children it still gives you feedback – is highlighted – makes him think that he should be able to click it.
Break (optional)		
Test B		**<u>Oculus</u>** at first slightly confused if he's in the application. Given prompt to press enter to start. Quickly types tables, presses table(furniture), projects to X axis easily. Searches for computer, projects to Y axis. Thinks how to get the search bar away, immediately presses ESC. Also wants to remove the search menu but that's not possible. Hinted that removing it is not implemented. Tries to move the query viz, but with the right hand. (does it by grabbing and making a fist). Is hinted about pinching with left hand. Still has some issues, is hinted to move the hand closer. Manages to grab it, but there are still some small issues (small activation distance). Tries to read the labels (moves left hand above it – that doesn't do anything). Figures that his head with headset has to be closer to hands to grab and move things in a better way. Grabs and moves it to a more convenient position. (earlier the labels on axis where overlapping some dark background in the surrounding so it was less readable. Again tries to interact with the left hand. Is hinted to use right hand for this instead. Then it's easy to do, he tries to click picture – first by "tapping" motions (as it was with the buttons), then quickly tries to pinch and succeeds. Is given controllers to try and prompted to project something on the Z axis, Searches for coffee. Tries to project to Z, tries to use index trigger to click, instead grabs the menu. Then tries to click other buttons (after struggling with positioning the menu).Finally succeeds. Grabs the Visualization Controller. Comments on the troubles of reading the axis labels (the labels don't face the player), and then hard to read if there is dark background in surroundings. Tries to grab the Visualization

		Controller a couple of times. Thinks it's unintuitive that it snaps to the controller in one place and doesn't stick to where he tried to grab. Is asked to try to rotate and scale – what would be intuitive for him. Immediately things about using thumbstick, but that doesn't work on it's own. Is hinted that he also has to grab. Then it's quite easy to use.
Post-test interview (B)		Using keyboard was quite easy, didn't bother him, he is used to typing without looking (does it at home sometimes with his headset). Once he figures the layout, it's very easy, no problems). As to the issue with seeing the axis labels, thinks that having white background around the labels would be helpful. Handtracking - what was troublesome was that you had to get really close to the Visualization Controller. Tried also to move the Visualization Controller at some point with right hand as well, that's an expectations to have the same gesture on both hands doing the same. Similarly for the index finger interactions – tried to do it with the left hand but this didn't work, it only worked with right.
Semi-Structured interview		Thinks that for usability the white void was better. Passthrough interferes with the software experience, it's easier to see things you have to use in white space. But the passthrough was kind of cool, nice to see your surroundings and gives nice AR experience. Generally more comfortable with the controllers – more limited (in a way that you can only press buttons etc. - there is this amount of buttons and one of them has to work). With hands you have to know/figure out the gestures needed to interact. But the scroll menu was easy to use (just touching it). Some technical limitations, trying to figure out what to do with the hands for them to work. Hypothetical question - ”ideal” implementation of hand tracking (intuitive gestures) vs ideal controller – feels nice not having controllers, but you would need to get the hang of using the hands. Matter of familiarity. Not exactly sure what would be preferred in this hypothetical situation. It was easier to press with finger than with the ball at the end of the controller. Physical vs virtual keyboard (any) – when you don't have to write essays like in this software, thinks it's better with the virtual keyboard, because you don't have to move close to the keyboard (mobility). All in all, virtual still preferred – better having the entire application in VR, not having to use the physical things in your surroundings (it looks like part of the background and not part of the software).

Subject 2.05

SUBJECT: 2.05		
Datetime		**16:00-17:00 18/05/2022**
Process	**Minutes**	**Process description (include timestamps if possible)**
Survey		26, student of philosophy, tried VR a couple of times (videogames)
Introductory presentation		
Test A		**<u>SteamVR</u>** Looks around, tries pressing some buttons, the main menu appears. Intends to click on tags, expects the index trigger to do the trick because that's what he's used to from previous experience. This doesn't work, so tries something else. Gets the keyboard, types book. Dim Explorer appears, very quickly starts moving the explorer and the columns of queries. Takes some time to find book, quickly manages to project it to X axis. There are no results (black hypercube), the subject doesn't know if it didn't work. Is hinted to try something else, since this didn't have any result and he is stuck. Gets the keyboard, moves it closer and types noodle. No result. Tries soup. Searches through the results. Applies soup to the X axis, notices the query viz appearing. Tries different buttons to zoom in on the X axis. The dim explorer is still visible. The subject would expect to be able to manipulate the axis the same way as dim explorer. Is stuck with this, so gets a hint to hide the keyboard and dim explorer. Presses the exit button on the keyboard, immediately manages to grab the viz explorer and pull it closer. Explores the values on the X axis, clicks on some cells, exploring pictures of different kinds of soup. Browses through the photos easily and intuitively, goes back to the axis, chooses a different cell. Gets back the keyboard types beverage. Tries to move the keyboard but it overlaps with the dim explorer. Gives up on that and just projects ginger beer to Y axis. There are no results. Projects the parent of that – beverage to the Y axis. Rolls up to food from beverage.
Post-test interview (A)		The only encountered issue was accessing the Visualization Controller the first time. Otherwise the interactions were quite unproblematic. At first expected to be able to use index trigger to confirm/click buttons, but seeing it doesn't work, quickly adjusted to using A button.
Break (optional)		

Test B		**<u>Oculus</u>** Tries typing, then remember he had to press enter first. Shortly tries to move the menu but with the right hand. Types soup and initiates search. Tries to grab the menu again (with the right hand). Can't grab it, scrolls through the results. Selects soup (dish). Tries to project to X axis. Pressed apply (direct filter) thinking it's necessary, is prompted to clear it. Tries pressing project on X again, finally query appears. Tries interacting with the query viz (moving it closer), but can't again. Is hinted to try left hand after some time (was stuck, haven't though to try himself). Moves the query viz around. Tries to select some photos by tapping (like on the search results menu). Gets a hint to try pinching. Selects a cell. Tries to scroll through photos but that's not possible with the hands. Go back to query viz. Types beverage (has only small issues with visibility. Projects something to Y axis. There is a really large query. Tries to grab it but has some issues, finally manages to do it. Moves it along the X axis, but it snaps a bit and it's not optimal.
Post-test interview (B)		Small issues with the keyboard visibility. Handtracking – is right handed and intuitively tried to do the interactions with right hand only. Expected that moving the viz query should work with just pinching and moving it, where it doesn't jump like it did now. In reflection, thinks that he should be able to just slide/scroll (same interaction like the one with scroll menu). Pinching would be used then for grabbing and pulling it closer or further away. Sometimes was uncertain of whether a button was pressed, even though the interactions themselves were quite intuitive. Was more aware of the surroundings, but it didn't feel completely right – felt it wasn't very accurate because of the interactions with keyboard.
Semi-Structured interview		Assumes that if keyboard was tracked it would be easier and preferable to the virtual keyboard used with controllers. When doing the tests in safe environment, knowing that he would be warned if he was getting too close to hitting something, this white environment was much preferable, as gave a feeling of being more immersed in the software. Might be preferable with the passthrough if in unfamiliar place and alone. Hands vs controllers – controllers, decision based on familiarity with controllers, not being used to use hands in VR.
Concluding survey		

Subject 2.06

<table>
<tr><th colspan="3">SUBJECT: 2.06</th></tr>
<tr><td colspan="2">Datetime</td><td>13:15-14:15
19/05/2022</td></tr>
<tr><td>Process</td><td>Minutes</td><td>Process description (include timestamps if possible)</td></tr>
<tr><td>Survey</td><td></td><td>25, student of Msc DDIT, haven’t tried VR before</td></tr>
<tr><td>Introductory presentation</td><td></td><td></td></tr>
<tr><td>Test A</td><td></td><td><u>Oculus</u>
Presses enter, the search bar appears. Thinks that maybe she needs to use a mouse or poke with a finger to activate typing in the search bar. Then thinks that she should just try to type. Proceeds to do it. Has issues with visibility of keys. Types sports, gets plenty of results. Tries to move the results menu around, initially has small issues but then manages to do it. Searches for sport instead (previous results not what expected). Briefly tried to remove somehow results from the menu, then just proceeded to delete the previous typed query and typed sport. Got new results. Projects one of the results on the X axis. Grabs the query viz, moves it closer and examines the results. Tries to select one cell, but keeps using the left hand which causes her to grab the element instead. Tried to pinch with right hand and accidentally drilled down to racing. Is not sure how to back to her result, tries to project from the search menu again (pressing the search result button, not project). Tries clearing the axis and project again (again first with the button with tag name). Then tries clicking project again and succeeds. Wants to click cell again, interacting with it the same way as with the menu (pointing). Then tries pinching but with thumb and middle finger, which doesn’t work. After some attempts is hinted to use the index finger instead. Does this, notices that something is loading, then the pictures appear. Intuitively immediately tries to grab these to bring them closer/move it in general.

Is handed controllers, is not sure how to use them. Tries to press buttons, but doesn’t move the controllers – expects to interact with the things by pressing the buttons only. Tried to go back to hands, but there were issues, the controller model in vr kept appearing and it wasn’t possible to switch to hands.</td></tr>
</table>

Post-test interview (A)		Using keyboard was a small frustration – user though that she should just be able to just type right away because she's used to typing on keyboard. But once one key was localized, it went smoother. But typing relied on her being used to typing without looking. Using the scroll menu was quite intuitive. But there were some mismatched expectations and confusion about the difference between using right and left hand. Would expect to do anything with any hand. IT was the first time for her using VR and she felt secure in the environment, could see the surrounding and wasn't afraid that she would bump into something.
Break (optional)		
Test B		**SteamVR** Looks around, wonders how to start. Presses different buttons, doesn't know how to get to the search. Menu appears. Tries to press the tags button – uses index trigger for that, doesn't work. Then tries the left controller, pressed a button that made the menu disappear. Gets it back, tries different button. Tries a few times to press the tags button with different buttons, appearing and disappearing menu. Doesn't notice the keyboard at all at first. Is hinted to look around and then notices the keyboard. Starts typing, remembers which button on controller has to be pressed. When typing reaches over to the keyboard, doesn't try to adjust its position to a more comfortable one. Searches for concert, gets the dim explorer. Tries to move it but accidentally opens the menu. Tries different buttons and gets the right one, browses through the results. Tries to find concert but this particular word is not there, she wonders if she scrolled to far. Is not sure if her query is there. Tries to project concert-goer instead. Has trouble clicking the button tag. Projects to X axis, no results. Projects consumer to the X axis instead. Sees the query viz, tries to reach it and get it closer but it doesn't work. Wants to first project something to Y axis before continuing to grab the query viz. Projects piano to Y axis. Can't figure out how to get the query viz, is hinted to hide the keyboard. After a while manages to click the closing button, moves the query viz closer easily, examines the results. Clicks on the only cell. Goes back, wants to roll up from piano, is not sure how to do it. Gets the main menu back, removes the values from Y axis and tries to do the same search again.
Post-test interview (B)		For the controllers intuitively tried to use the index trigger almost every time. But also didn't memorize the other buttons, what they are for so just tried different things. But each time went for the index trigger first. Is not sure why she didn't try to move the keyboard, but instead reach over to it every time. Remembered from the walkthrough that there was some difference between using the left and right hand, so thought that this was relevant while using the other things. Thinking that using the right hand would work for most of the things and left maybe for just something small.

		Environment – thinks it was nice that it was all white, nothing else bothered her and wasn't distracted by seeing other people or things. The downside was the lack of orientation in her surroundings. Noticed that the elements were appearing in different places, is not sure that it is related to the environment. Would expect it to appear in the same place at all times. Is not sure if the things were moving around (random placement) or if it was because of her movement. At first was confused about the virtual keyboard, not being sure how to use it. Then would expect to just be able to type by tapping with the controller – it highlighted, but here you actually had to both hover it and also press a button. All in all, would expect to work just with touch. Scrolling the dim explorer was very intuitive as it reminded other interactions, like scrolling on iPad etc – same motion, using index finger, very familiar interactions.
Semi-Structured interview		Using controllers vs hands – she like using the hands more, felt it was more responsive. Thinks it might also be because she didn't know how to use the controllers. Is also not used to any kind of controllers, so thinks that this might be why she had such big troubles with it now and prefers hands. Virtual keyboard vs physical keyboard with 3D render – would prefer the physical keyboard due to being much more familiar with it, feels more safe (same as with hands vs controllers). Also with the second test when couldn't press something with controller automatically went for trying to use finger for interaction, forgetting that it's not possible in this version. In general trying familiar interactions, known from interacting with other media. Expecting right and left hand/controller to have the same behaviour.
Concluding survey		

Subject 2.07

SUBJECT: 2.07		
Datetime		**13:00-14:00** **23/05/2022**
Process	**Minutes**	**Process description (include timestamps if possible)**
Survey		24, Msc Data Science, no prior VR experience
Introductory presentation		
Test A		**SteamVR** Presses different buttons to see what they do. Got the menu but hidden it right away. Tries to press the tags button, interact with it with different buttons but none work (controller is not close enough to it). Is stuck for a longer time so is given a hint to move the ball of controller closer. Gradually gets closer and is surprised when she gets some feedback from the button at last. Thought that the ball is more of a pointer and she doesn't have to actually touch the button with it. Clicks tags, searches for Dancer (types quite easily). Projects Dancer to the Y axis (also quite easily getting close enough and clicking the buttons this time. Tries to reach out to the query viz, grab it and pull it closer but can't. Heavily stuck, so gets a hint about the exit button on keyboard. Hides the keyboard and dim explorer, quickly grabs and pulls the Visualization Controller closer. Looks at the results, clicks on Dancer to roll up to have more results. Pulls out the keyboard again to find query to project on another axis. Searches for Club, (meaning place where you dance) projects the first result (not noticing that it's related to Stick. Is confused about the results, takes her a while to realise that there are different options. Wants to replace the values on the X axis but is confused about how to do it. Thinks about it for a while, then decides to just make a new search and project something on the X axis again. Searches for Club again, trying to find the building. Browses the dim explorer, can't find the right type. When asked for interpretation of the columns on dim explorer, after thinking a bit interprets them correctly.
Post-test interview (A)		It was quick to figure out how to open and hide the menu, but had issues with interacting with the button. The subject thought that the ball, which was also in some distance from the controller was sort of a pointer and just had to "overlap" with the button to interact with it. Dim explorer – finding the right tag – distance was an issue, initially didn't notice that there were other results further down. Seeing club she didn't even think about the other columns but went straight to just projecting it to an axis.

		Controllers – movements were quite intuitive, a bit like using an ipad etc, with the direction of movement etc.
Break (optional)		
Test B		**<u>Oculus</u>** Wonders for a moment how to get the search bar up, but instinctively presses enter. Is disturbed by the lack of visibility of keys but thinks maybe she can just remember the layout. Goes to type a query and it's quite fast an easy for her. Searches for that, gets results, tries to grab and move the menu but has some trouble with that (pinches with her all fingers, which might interfere). Finally grabs it, moves it to a more comfortable position. Projects it to the Y axis. Wants to move the search bar away but can't. Moves away from that. Wants to project something to the X axis as well. Searches for another tag, where 2 results are present. Projects the first one to X axis (it has no children). Is confused about the result, thinks that maybe it's not working or something. Is hinted that there are just no results for that specific thing. Thinks a bit how to clear the axis, clicks the clear button. Projects the other result to X axis and gets some results. Goes on to grab the Visualization Controller, has a small issue with that. Finally grabs it and pulls it closer. Explores the results, looking at individual cell, poking the index finger to view the axis values and then to view the actual photos (doing the same thing as with buttons) but that doesn't work. Has to be hinted to try pinching. That works, she views a couple of pictures. Tries to move them the same way as the menu and Visualization Controller. Is given controllers to try out. Is confused, intuitively tries to use fingers again. Thinks it was easier and more natural to use the hands. Quite fast gets back by clicking the right button. Manages to click the buttons and find another tag. Takes some time to figure out how to grab etc.
Post-test interview (B)		Keyboard was usable despite not being able to see the keys (finding one reference point was enough to get her going easily.) Overall she says it didn't affect her overall experience negatively. Hands were more intuitive to use.

Semi-Structured interview		Controllers in the steam vr version vs hands – thinks that if already familiar with controllers might be faster, but hands are more fun to use. (there was some unexpected behaviour – grabbing when not indented eg). If hands where working ”ideally”, they would be much more preferable. They are more immersive for the VR experience, controllers you can also use with your console etc, so hands give more full experience. The inconsistency in controls (using right and left hand) is not an issue for her – thinking driving car where one hand is used to manipulate the gears and the other to stay on the steering wheel. Thinking about this as interacting with a system, so as in real life it's fine to use each hand for different things. (so grabbing with just left hand is fine). Passthrough vs white background. Thinks that the passthrough gave her better depth reference – made the interaction easier, because she had a better idea of how close something (software element) was to her. For using this software she would prefer the passthrough environment. She didn't have any issues with the passthrough, it was clear what was a part of the software and what has not. In general likes the oculus results menu better- could see more easily all of the results – faster and easier to get the result you want – the tags are all gathered in one place. Virtual vs physical keyboard – doesn't like the virtual keyboard at all. Also from previous experience with keyboard eg. on console, where you have to navigate through it with controller. IT was slightly better here where you could touch it with controller but still not ideal and definitely prefers the physical keyboard, even when working not perfectly (visibility issues).
Concluding survey		

Subject 2.08

SUBJECT: 2.08		
Datetime		**12:00-13:00 25/05/2022**
Process	**Minutes**	**Process description (include timestamps if possible)**
Survey		19, Bsc student at CBS, tried a VR game once but not experienced in general
Introductory presentation		
Test A		**Oculus** Thinks what she should search for, tries to remember how to make a query. Remembers that she has to press enter. Does that, the search bar appears. Looks at the keyboard to type and says that she can't see the keys. Types dog plays volleyball quite easily all the same. Hinted later that program doesn't support this kind of queries. Searches for just dog. Wants to grab and move the menu but continues trying with the right hand only. Wants to project dog to the X axis, does it quite easily. Tries to grab he query viz again with right hand. Accidentally rolls up to domestic animal. Gets a hint to try the other hand as well. Manages to grab with the left hand and move it. Wants to try to grab the menu as well, but it doesn't work and grabs query viz repeatedly instead (has large activation distance). Gives up on that and searches for volleyball. Projects that to the Y axis but there are no results. Few times again she accidentally rolls up or drills down. After a while of wondering why there are no more pictures gets a hint to project something else maybe to the Y axis, as volleyball doesn't have any children and therefore there are no results. Easily searches for sport and projects that to the Y axis. Explores the values on axes and notices there is just one picture in the results, Points the finger and reads the tags attached to it, then tries to pinch a few times to click it. When she does, there is one picture which she immediately tries to grab and bring closer.
Post-test interview (A)		Didn't really think about using the left hand, as she's right handed and it's natural to just use the right hand. The accidental grabs were annoying, especially when she wasn't even intending to grab anything and just doing gestures while talking. Keyboard was hard to see but she's very used to typing without looking and keys at all. Seeing the environment around her added to her enjoyment, as she felt more safe and wasn't afraid that she would hit something.
Break (optional)		

Test B		**SteamVR** She's looking at the controllers, wonders what are the balls at the end of the controllers. Tries to remember how to start the querying. Presses different buttons one at a time. Finally gets the menu to show up. Instinctively wanted to use a thumbstick to navigate through the menu. When she tries to interact then with a button she presses index finger (on index trigger) to try to activate the button. She notices that the buttons highlight when she gets the controller close enough. Finally she presses the tags and gets the keyboard. Thinks that she needs to press both the index finger and A button in order to press the virtual buttons. Moves the keyboard around, then types sport. Is surprised with the huge result she got. Manages to move the dim explorer and look at some results, but then decides to search for another tag. The keyboard is faded, she tries a few times to be able to interact with it again. Manages to do that. Comments that it takes a long time to type when she has to move her hand around for every letter. Is again overwhelmed with the amount of data she gets when searched for some tag. When presented with the dim explorer this time she wants to move it closer to her but it doesn't work for her. When interacting with the buttons there, she thinks that when the controller pointer is visually over a button she should be able to interact with it. Is hard stuck here so gets a hint to get even closer. Then easily projects 2 things on 2 different axis. Tries to grab the Visualization Controller, with A button which she knows from moving the keyboard and menu. Then figures out that she has to use the index trigger (after trying different things).
Post-test interview (B)		A few things were confusing at the beginning as coincidentally something happened, then she thought that this interaction was happening with these buttons, but they were used by accident (like with using both index and A to submit). It was frustrating at first to type on the keyboard, due to having to move the hand around so much, thinks that maybe if there was more type to familiarize with it might get easier. Thinks also that it might be better if the keyboard was smaller, having less distance between the keys would not require as much movement. Moving the dim explorer was intuitive.
Semi-Structured interview		Physical keyboard vs virtual keyboard – strong preference for the physical keyboard even without being able to see the keys, as she is used to typing without looking. Passthrough vs white environment At first like the passthrough, but at then in retrospective, thinks that white was more immersive etc. Having tried the oculus version first had an influence on that because she had time to get used to using the headset and VR, so later she felt more comfortable in general. Preferred hands even though it was a bit difficult to figure out at first – thinks it's faster to use once you get used to it – there is also just a limited set of interactions.

G Usability schema

Usability heuristics

1. Visibility of system status: If a user does something or the system is thinking, is this communicated?
2. Match between system and the real world: Terminology used
3. User control and freedom: Can users back out of an operation easily?
4. Consistency and standards: Are the designs internally consistent? Is it consistent with established standard in genre/medium?
5. Error prevention: Are users appropriately warned if they are about to make a mistake?
6. Recognition rather than recall: Does the system rely too heavily on memory to be appropriately used?
7. Flexibility and efficiency of use: Is the design efficient to use? Does the design allow for more efficient use if trained?
8. Aesthetic and minimalist design: Show only what needs to be shown
9. Help users recognize, diagnose, and recover from errors: If unintended behavior occurs, does the system inform the user?
10. Help and documentation: If the system needs explanation, does it provide it?

STEAMVR IMPLEMENTATION			
GROUP	**DESCRIPTION**	**HEURISTIC**	**PARTICIPANTS**
CONTROLLERS	Initial confusion about button mappings	10	ALL
	Persistent confusion regarding button mappings	6, 10	1.03; 1.04; 1.06; 1.07; 2.01; 2.02; 2.04
	Confusion about hover point / menu interactions	10	1.03; 1.04; 1.07; 2.01; 2.02; 2.04; 2.06
ENVIRONMENT	Boundary / physical space insecurities	8	1.03; 1.05; 1.07; 1:08; 2.03; 2.06
	Potentially vertigo-inducing (when looking down at a large queries or hierarchies)	2	1.02; 1:08; 2.03
MAIN MENU	Initial confusion about which path (if they want to generate query)	8	1.04; 2.04
	Initial confusion about how to trigger menu	10	ALL

	Unclear what clicking on projected tags does	8	
	Grab-functionality is not communicated well or not understood. Users assume grabbing anywhere on the object.	3	1.06; 1.07
DIMENSION EXPLORER	Overwhelming, information overload on large queries	8	1.01; 1.07; 2.02; 2.03
	Difficult to efficiently browse, physically demanding	7	1.01; 2.02
	Confusion about column-row search results	10	1.01; 1.02
	Incoherent search result ordering (not alphabetical)	4	1.01; 1.03; 1.05; 2.01; 2.02; 2.03; 2.04; 2.05; 2.06
	Confusion about how to move – relies on index trigger when it has been button so far	4, 7	1.04
	Uncertainty about column highlighting. Participants interpret it as the column being selectable like a button, when it indicates that the column can now be moved on the Y-axis.	4, 8	1.04; 1.07
	Search results are linked to keyboard. Disappearing keyboard hides results	4	1.04
KEYBOARD	No feedback on empty queries	9	ALL
	Inefficient typing compared to other methods.	4	1.01
	Keyboard spawn may not be noticed at first	1	1.03; 1:08
QUERY VISUALIZER	Must dismiss keyboard and dimension browser to grab Query. Attempt to “reach through” and grab instead	5, 9	ALL

QUEST 2 IMPLEMENTATION			
GROUP	**DESCRIPTION**	**HEURISTIC**	**PARTICIPANTS**
CONTROLLERS	Snap-grabbing very unintuitive and contrary to expectations	4	1.01; 1.04; 1.05; 2.04
	Sometimes controller face buttons are used to activate, other times hover-point collision activates (no communication)	4, 9	1.01; 1.03; 1.05; 2.03
	Unclear functionality (scale and rotate when grabbing items)	10	1.07; 2.04
BARE HANDS	Not clear that it can function as a control method	10	ALL
	Unintended input readings (general)	7	1.04; 2.01; 2.03; 2.04; 2.05; 2.06
	(Iteration 2) Arbitrary differences between hands; one can grab and the other select	4	ALL
	(Iteration 2) grabbing when the user didn't intend to	5	2.01; 2.04; 2:08
	Ambiguous/inconsistent controls (sometimes index finger activates, other times pinch does)	4, 5	ALL
	(Iteration 1) Missing features compared to controllers (grabbing)	4	1.03; 1.04
ENVIRONMENT	Pass-through environment can be distracting, obscure what is part of the software	8	1.03; 1.05; 1.06; 1.07; 2.02; 2.04
	Uncertainty about what is part of the environment and what is software	2	1.04
	Difficult to parse software elements in non-ideal backgrounds (e.g., black text on black surface)	8	2.04
KEYBOARD	Low key visibility in pass-through camera	2, 4, 8	ALL
	Frustration at (subjective) slow input compared to normal conditions	2, 4	1.04; 2.02; 2.05; 2.06
	Necessitates keyboard layout memorization for efficient inputting	6	1.04; 1.05; 2.02
	Not clearly marked as part of the software (no render, blends in with other "real" objects)	2, 5, 4, 8	ALL

	Confusion about how to start (enter key)	10	ALL
DIMENSION EXPLORER	Lack of context to search results (sibling, children tags)	8	1.03; 1.06; 2.02; 2.03
	(Iteration 1) No visual aids on axes project buttons, participant must remember which axis they projected to	6	1.02
	Confusion about when inputs were activated due to button hover effect	1, 5, 8	1.07
INFORMATION MENU (ITERATION 1)	Info overload, too much text, relies on recollection rather than recognition	6	1.01; 1.02; 1.04; 1.05; 1.06; 1.07
	Suboptimal ordering of information – does not start with what would be the user's first interaction (press enter to search)	10	1.01; 1.05
	Lacking information	10	1.02

BOTH IMPLEMENTATIONS

GROUP	DESCRIPTION	HEURISTIC	PARTICIPANTS
QUERY VISUALIZER	No communication if query does not contain data (must project and then revert)	5, 9	1.02; 1.03; 1.04; 1.05; 1.06; 2.02; 2.05
	Initial confusion about how to drill down and roll up (lack of feedback on tags with no further children)	8	1.02; 1.03; 2.04
	Application freeze when generating large queries, participants unsure of what happened	1	ALL
	Labels do not rotate to face user	4	1.01; 1.02; 2.04