Virtual Reality Operating System User Interface

Bachelor of Science Thesis in Computer Science and Engineering

WILLIAM FALKENGREN
ANDREAS HÖGLUND
MALIN LILJA
ANTON SOLBACK
ANTON STRANDMAN
JOHAN SWANBERG

CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERISTY OF GOTHENBURG
Department of Computer Science and Engineering
Gothenburg, Sweden 2017

BACHELOR OF SCIENCE THESIS

Virtual Reality Operating System User Interface

WILLIAM FALKENGREN

ANDREAS HÖGLUND

MALIN LILJA

ANTON SOLBACK

ANTON STRANDMAN

JOHAN SWANBERG

CHALMERS UNIVERSITY OF TECHNOLOGY

UNIVERISTY OF GOTHENBURG

Department of Computer Science and Engineering

Gothenburg, Sweden 2017

Virtual Reality Operating System User Interface
WILLIAM FALKENGREN

ANDREAS HÖGLUND

MALIN LILJA

ANTON SOLBACK

ANTON STRANDMAN

JOHAN SWANBERG

Supervisor: Daniel Sjölie, Department of Computer Science and Engineering
Examiner: Arne Linde, Department of Computer Science and Engineering

Department of Computer Science and Engineering
Chalmers University of Technology
Univeristy of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Images and figures are captured or created by the authors, if nothing else is stated.
Department of Computer Science and Engineering Gothenburg 2017

Abstract

New virtual reality (VR) technology has been developed and is reaching a broader consumer-base than ever before. However, no consumer operating system (OS) has been released optimized for VR. In this project, a concept VR OS user interface (UI) was developed, and different interaction patterns were developed and evaluated. Several concept applications were developed and evaluated in user tests. The project resulted in a concept VR UI application with basic OS features such as viewing files and opening programs. The tests conducted suggest that some interactions, such as throwing away programs to close them and opening programs through a hand menu with 3D icons, are efficient. The tests also suggest that new interaction patterns for interacting with legacy 2D applications in VR should be researched.

Sammandrag

Ny virtuell verklighets (VR) -teknik har utvecklats och når en bredare konsumentbas än någonsin tidigare. Dock finns ingen konsumentversion av ett VR-anpassat operativsystem (OS). I detta projekt utvecklades det ett användargränssnitt för ett OS i VR där olika interaktionsmönster utvecklades och utvärderades. Flera konceptapplikationer utvecklades och utvärderades i användartester. Projektet resulterade i en koncept-VR UI-applikation med grundläggande operativsystemsfunktioner som till exempel att visa filer och öppna program. De tester som utförts tyder på att vissa interaktioner, som till exempel att kasta bort program för att stänga dem och öppna nya program via en handmeny med 3D-ikoner, är effektiva. Testerna föreslår också att nya interaktionsmönster för interaktion med äldre 2D-applikationer i VR bör undersökas.

Acknowledgements

We would first like to thank our supervisor for this project, Daniel Sjölie who helped us throughout the course of the project. We would also like to thank Visual Arena in Lindholmen Science Park who let us use their VR equipment during the development. Last but not least we would like to thank all our test participants who gave us invaluable feedback in our tests.

Contents

List of Figures

1

Introduction

Over the last two decades, the use of computers have constantly been growing and become a fully integrated part of our society. With the latest consumer friendly virtual reality (VR) hardware, such as Facebook's Oculus Rift and HTC's Vive, consumers are introduced to new ways of experiencing and interacting with computer systems through virtual environments. This allows for interaction that to a greater extent resembles the interaction with objects in the physical world.

There are VR-applications, such as Big Screen Beta (Epic Games, 2017), which provides the user with a way to use and interact with the underlying operating system (OS) by duplicating and projecting its two dimensional (2D) graphical user interface (GUI) inside the VR-environment. Therefore, even though the interaction is taking place in a fully immersive 3D-environment, the interactions within the OS is 2D and optimized for the windows, icons, menus, pointer design, shortened WIMP (Markoff, 2009). Due to that the WIMP design is based on windows which are two dimensional, and pointers which do not normally exist within VR space, WIMP can be considered unoptimized for VR use. There is currently no commercial VR-application designed as an operating system utilizing the new possibilities VR provides.

1.1 Purpose

The purpose of this project is to create a user interface (UI) where the user can, in a VR environment, get access to the most essential features taken for granted in a 2D operating system, such as starting applications and viewing files. Several UI concepts related to opening, managing and closing programs, files and legacy applications are to be developed and evaluated to find efficient and intuitive solutions for the UI. Several applications will be implemented to allow the user to interact with the UI in a realistic way. The UI shall use the new opportunities that VR entails, in forms of utilizing the 3D environment and intuitive interactions with virtual objects and tools.

1.2 Scope

Within the scope of the project is to create a UI which gives the user access to the most common features of an operating system, such as starting applications and viewing files. The project will include proof-of-concept interaction implementations for starting other VR applications and traditional 2D desktop applications. All these functions should be accessible from within the VR application, without taking the VR headset off.

The UI will allow the user to multitask, i.e. use multiple running programs simultaneously, and customize the work area by moving the different programs around in the 3D space. The UI will also allow the user to find and close currently running programs.

The UI should be usable by people with normal variation of height, arm length, sitting, standing, without causing the user discomfort.

1.2.1 Delimitations

The project will not produce a full fledged operating system, but a UI concept with emphasis on the implementation of the user interface and the interaction design of the system. The operating system UI will be constructed in the game engine Unreal Engine 4 (UE4) developed by Epic Games. Programs built with UE4 requires a compatible operating system to run. Therefore, the project will be limited to exploring the possibilities of the front end design of a VR operating system. Backend components such as drivers and kernel will not be implemented. Frameworks for developing applications for the OS will not be developed. The project will not attempt to support applications designed for running on 2D operating systems, instead prototype applications will be developed to represent legacy applications.

The UI will be developed for the VR and computer hardware that is available to students at Chalmers University of Technology. The computers are Windows computers, and the graphic cards are nVidia Gtx 1070 or higher performing equivalents. The VR technology available is Oculus Rift and HTC Vive head mounted displays (HMD), Oculus Touch and Vive motion controllers. The application will not be tested on other hardware or software configurations.

The UI will not be developed with accessibility features. Common user disabilities such as color blindness will not be considered during development. This is due to time constraints. Reading longer texts in VR can be very stressful for the eyes due to the low resolution of existing hardware (Applebee and Deruette, 2017). This project will not attempt to create any alternative reading methods, but instead minimize the amount of text.

1.3 Project overview

The project will use an agile development process consisting of four major steps: Requirement analysis, solution design, development and testing. Each step will use established methods, introduced below, which are explained in further detail in section 3.3. The steps, or phases, were performed twice in an iterative fashion according to the agile methodology, as can be seen in Figure 1.1, where the initial step was "requirement analysis".

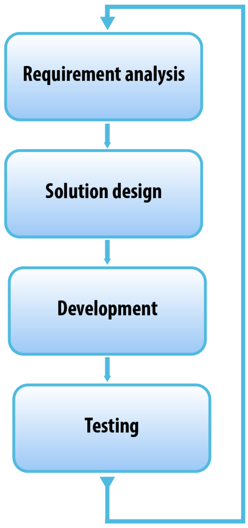

Figure 1.1: *An overview of the major development steps of the project.*

In order to confirm that most of the necessary components of the system are identified, the starting point will be a requirement analysis. The requirement analysis step will use semi-structured in-depth interviews with experienced computer users from different backgrounds. In the second iteration the results from the test step is used as input for the requirement analysis. These results are then aggregated and evaluated using the KJ analysis method, information about the KJ analysis method can be found in chapter 3.3.2.

Once the requirements are identified, solutions are proposed by developers and designers in close collaboration during brainstorming sessions. The proposed solutions are put into a backlog and prioritized for the development phase.

The development phase will be split into sprints according to a modified version of SCRUM, where the daily scrum meetings were omitted. Each sprint is one week long. At the end of each week there is a sprint review, consisting of retrospective and dividing tasks among users and planning for the next sprint (Scrum, 2017).

Tests will be conducted in order to receive input from users and make sure that the needed features are included and design solutions are perceived as intended. In total two tests are conducted, one for each iteration. The results of the first test is used in the next iteration of the requirement analysis phase, and thus the process is repeated.

2

Background

This chapter contains background information about user interfaces, operating systems and VR technology relevant to this project.

2.1 Introduction to operating systems

Computer system architectures are often built upon abstracting lower levels of data, making the implementation of the complex systems manageable. The same concept is applied to the front end implementation. By moving many commands away from the end user, the system becomes easier to interact with and in turn accessible for a broader audience. At the core of personal computers, no matter if the physical implementation is a desktop computer, laptop, mobile phone or gaming console, there is an operating system managing the interaction between processes and hardware, handling networking, I/O etc. (Dhotre, 2008). All these features are abstracted away from the end user by introducing a UI, helping the user perform various tasks without the need to know about how it is done behind the scenes.

2.2 Traditional OS interaction

In a traditional OS the default input actions are left and right click on the mouse, in combination with keybinds on a keyboard. For mobile phones there is the touch, long touch and swipe actions. But none of these actions can be directly applied to a VR, as you do not have any surface to touch nor a mouse to click. Looking at previous implementations in programs such as Oculus Home (Oculus VR, 2017a), Steam VR (Valve Corporation, 2017), Big Screen beta (Bigscreen Inc, 2017) and many VR games, the traditional WIMP interface is introduced by extending a laser pointer from the hand controller to a menu option, and simulating the left (and sometimes right) click with a button on the controller. Although this solution is very familiar to any user experienced with a desktop computer, it does not utilize the familiarity humans have with our hands.

2.3 VR hardware

A few years ago, access to VR technology was limited to specialized developers (Lily Prasuethsut, 2017) as no consumer VR hardware was readily available. Recently several new developers of VR technology have appeared. Facebook, HTC and Sony all have released powerful VR products, consisting of HMDs and hand tracking controllers, and made them widely available to the public (Durbin, 2016). Today consumers that are curious can get hold of high quality VR hardware (Jerald, 2015) for less than $1000 (Amazon, 2017).

2.4 Current VR UI limitations

The move from a 2D to a VR interface naturally leads to changes in the way users interact with, and handle programs and files. Motion sensitive hand controllers makes it possible for the user to interact with the environment through movement and gestures in 3D space. These new ways of interacting makes a new computer experience possible, and if they are utilized correctly they could lead to a more intuitive and efficient interaction between man and computer (Jerald, 2015).

However Facebook and HTC's products, which are the major consumer version VR suppliers for desktop computers, both require that the user interacts with the OS using a traditional 2D UI to start their specific VR applications. There is software that gathers VR applications and places them in a VR environment, such as Oculus Home (Oculus VR, 2017a) and Steam VR (Valve Corporation, 2017). These environments also allow the user to download apps which can perform other tasks such as look at pictures and play games. However, they lack the feature of managing files, offer limited support for running applications concurrently and offer limited customization options for the environments. Consequently, the user is required to have a traditional operating system and monitor to operate their computer efficiently. There are compatibility environments such as BigScreen Beta (Bigscreen Inc, 2017) and VirtualDesktop (Virtual Desktop, 2017) which allows users to interact with 2D applications in the VR environment, however due to these 2D applications being designed with WIMP design, usage is not optimized for VR controllers. A picture of a user interacting with 2D applications using BigScreen can be seen in figure 2.1 (Bigscreen, 2017) and the UI of Oculus Home can be seen in figure 2.2 (Simplicity Compass, 2016). Currently, there is no OS optimized for use with VR headset and hand tracking controllers.

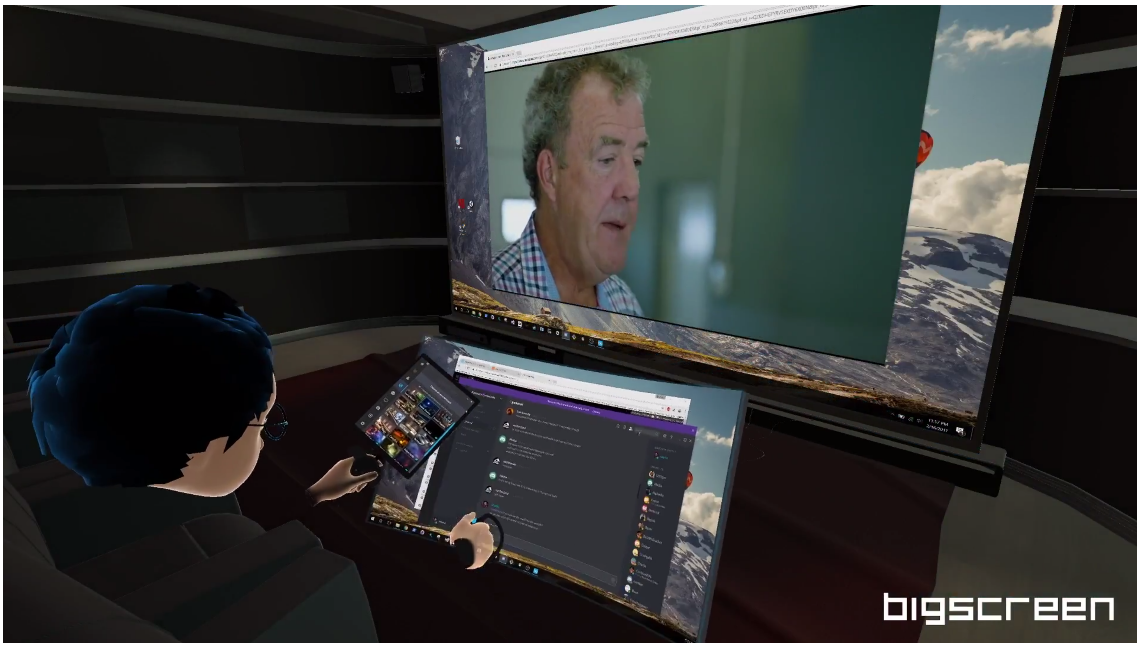

Figure 2.1: *A screenshot from Bigscreens youtube channel showing a user interacting with two 2D screens in a VR environment.*

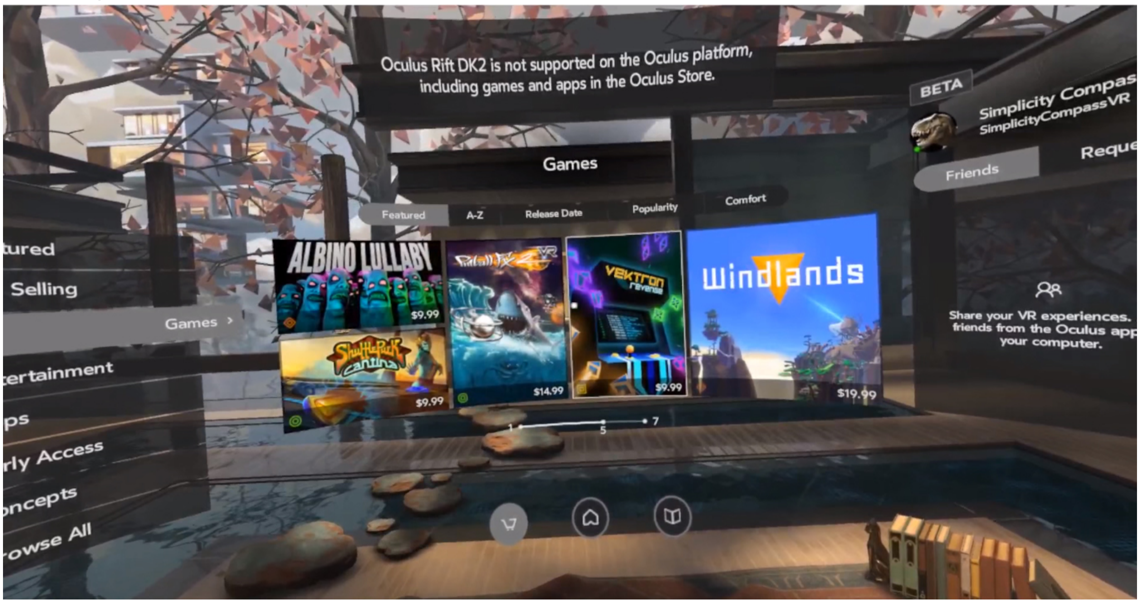

Figure 2.2: *A screenshot of the UI of Oculus Home.*

3

Theory

To make a UI well designed, several factors have to be taken into consideration. According to Jerald (2015), a well designed VR experience can create the excitement and awe of being in another world. He also shows that it can also improve performance, cut costs and create better understanding of the information presented to the user. However, poorly designed VR experiences can cause the opposite and induce anger and irritation, even physical sickness if not properly implemented and calibrated (Jerald, 2015). Research into the field of usability in VR has only recently started, and there are still no established best practices for interaction (Handman, 2017). On the other hand much of the established research on usability can be applied to VR as well (Mike Alger, 2017). The following section will present human factors and design theories, which were considered throughout the development of the UI to make a well designed VR experience based on relevant theory about human factors. It will also present theory about the methods and tools used to create the VR UI.

3.1 Human factors

Most human beings share common senses, and our brains react to certain stimuli in similar ways (Schacht and Mulder, 2012). It is possible to optimize UIs based on the phenomena that humans tend to react in similar ways to certain types of stimuli, and that humans have common patterns of searching when attempting to solve problems (Schacht and Mulder, 2012). Optimizing an interface to accommodate these common patterns can help the user perform tasks efficiently (Cooper, Reimann, and Cronin, 2007).

3.1.1 Mental model

According to Norman (1990), a good *mental model* allows the user to predict the response of their actions in a way that is correspondent with the outcome in reality. This is of importance since the user needs to feel in control of their actions to feel comfortable. On the contrary, a bad mental model will cause the user to be unable to predict the consequences or results of their actions. This means their predictions will not be correspondent with what actually happens, which can cause confusion and break flow. The user creates their mental model by simulating the outcome from the system or product with the help of its visible structure - in particular from its affordances, constraints and mapping. A good use of these structures makes the user create a good mental model which leads to the user understanding what actions are possible and their effects on the system.

Affordances is described by Norman (1990) as the perceived and actual properties of the thing in question, mainly those fundamental properties deciding what the thing can possibly be used for. Affordances provide clues for the user on how things can be operated. Using and taking advantage of affordances reduces the need of instructions. An example of an affordance could be the ear of a mug, it allows the user to pick it up firmly without getting burnt. *Constraints* are described as the perceived and actual properties of the thing which determines the restrictions in the possible use of the thing. Physical constraints rely on properties of the physical world, while semantic constraints build upon the knowledge and situation of the world. This kind of knowledge can be an important and powerful clue to successfully handling a system (Norman, 1990). *Mapping* is the relationship between the controlling action and their result in the world. For the mapping to be successful the controlling action has to correspond accordingly to the result in a logical way. This could be by the combined factors of being visible, close to the desired outcome and by providing feedback immediately (Norman, 1990).

A correct mental model, according to Jerald (2015), is necessary for the interface to be perceived as intuitive. To create a quality mental model the user's assumptions should be made explicit by the help of signifying clues, feedback and constraints. The user will eventually form filters to generalize interactions by their consistency, leading to understanding and effectiveness if it is used correctly. Once the user starts to form a mental model of the interface they are very likely to stick to decisions already made and are unlikely to change them later on. This leads to the importance of creating an experience that forms a correct mental model and is easy for the user to like from the very beginning.

3.1.2 Usability

According to Nielsen (1993), *usability* is the extent to which a product's functions can be used with ease. Good usability results in efficient use and few errors for a user, and it makes the interface more subjectively pleasing. This definition of usability does not include utility, as it is the second part in the aspect of usefulness.

Jordan (1998) has complemented definition by adding five aspects of usability. These aspects can be used to evaluate usability. *Guessability* is the first time users' understanding of the interface. Good guessability is characterized by interfaces that first time users need no instructions for. *Learnability* is the degree of ease the user has to learn the interface. An interface with good learnability is easy to learn. *Experienced user performance* (EUP) is the efficiency a experienced user can accomplish with the interface. Often it is features that make the interaction quicker, or give the experienced user access to more functions that are not available for the first time user. *System potential* is the potential of the interface's usefulness that is reachable. *Re-usability* is the users ability to reuse the interface after already learning it a first time. Good re-usability should not require any new learning and depends on the recognition of the user.

3.1.3 Attention

An understanding of human attention is beneficial to creating an intuitive UI. According to Salvendy (2012), attention can be conceptualized as having three modes. These modes are selective, focused and divided. Selective attention is most relevant for the VR UI and will be the one presented here.

According to Salvendy (2012), selective attention is used to discover what to process in the environment. This kind of attention can be described as the influence of the combined force of four factors: salience, effort, expectancy and value, which are described below.

Salient features of the environment works to "capture" and attract attention. Sound is usually more attention grabbing than visuals, leading to sound being the first choice used in alarms. Although, visual display can also be used to grab attention, with the onset of a stimuli tending to be the most attention grabbing feature. The onset of a stimuli is a change of an element in the environment, for example increased brightness or the appearance of an object.

Expectancy is the knowledge regarding the probable time and location for information to become available. A frequently changing environment is scanned more often than slowly changing one, due to the expectations of new important cues of the changing environment. This can be of importance when guiding attention, for example an auditory warning may direct attention towards the display indicator because the user is expecting a significant change.

Effort may influence attention allocation in a negative way. Small movements requires a little effort, however larger movements require considerably more information access effort. For example, glancing down requires far less effort than turning around, leading to high requiring information access effort may go unnoticed.

Value refers to the importance of knowing what information is relevant or not. Relevant information will result in a higher cost if it remains unnoticed, like a driver needs to focus on the road ahead despite a lot of perceptual actions happening in the side view. The view ahead is relevant and has a higher value, while the information to the side has already passed and therefore has a lower value.

3.2 VR health effects

To be able to use the VR comfortably and safely several factors have to be taken into consideration. Some are directly linked to the hardware, while others are affected by the experience of the VR UI design. In this section *VR sickness* and relevant ergonomics related to the UI will be presented.

3.2.1 VR sickness

VR sickness is a phenomenon that is a specific case of motion sickness. VR sickness primarily occurs when a person's vestibular system, the sense of balance and force, does not match up with their visual perception of movement (Benson, 2002). In normal cases of motion sickness, such as reading while sitting in a car, the user's body feels the forces of moving, but does not see the moving environment. This conflicting feedback can cause motion sickness. In VR, the situation is reversed, the user can see the environment moving, while sitting still. This leads to the user not experiencing forces of movement, but receiving visual cues of movement (Benson, 2002). The intensity and duration of the experienced motion sickness varies between individuals (Jerald, 2015).

3.2.2 Ergonomics

An understanding of ergonomics will help to design a human centered UI that can be used for longer periods of time without causing any sustaining damage for the user. As the VR UI opens up for new ways of interaction, especially compared to a classical workstation, ergonomics can be used to predict and evaluate the long term use. The following section will present some risks with certain body positions and work, but also how to reduce those risks.

Repetitive light static work, on as low as 2-5 percent of total muscle strength, can be a risk for muscle damage. These low static loads are usually constant during the workday and are common in modern work, for example in continuous office work sitting by a computer. The damage occur due to muscles not being properly relaxed before they are activated again. The risk of damage by repetitive tasks is increased if no customization or variation of the task is available, which are effective ways to prevent damages caused by light repetitive physical work (Hägg, Ericsson, and Odenrick, 2011).

To avoid strain, especially in the back area, it is recommend to vary the body position as often as possible. Work should be possible while standing up to relief the back, although it is necessary to make sure the neck is not bent forwards or arms reaching to far from the body. Aim to keep the arms close to the body, hands above the shoulders should only be present for short periods of time (Hägg et al., 2011).

Although standing is a good body position for work it is also important to design VR experiences for sitting. Designing VR experiences for sitting reduces the risk of certain kind of injury caused by the users blocked senses. While the user is wearing a HMD they are both blind and deaf to the real world, which leads to a higher risk of falling down and collision with real-world objects as they are moving around. A sitting experience also helps to reduce VR-sickness since it provides a reference frame and creates better postural stability (Jerald, 2015).

3.3 Design tools and methods

In this chapter, some of the methods used during the development of the project are described.

3.3.1 Function analysis

A function analysis is used to specify functions of a product and to divide them into different categories and grade them in order of importance. Österlin (2016) writes that figuring out the main purpose of a product and exploring different solutions to filling this purpose is important when designing a new one. According to Österlin there are three categories of functions of a product, main functions, subfunctions and extra functions. Main functions are the functions vital for the product to work. Subfunctions are functions required for the main function to work. Extra functions are functions that could be removed and still leave a functional product.

When performing a function analysis all functions of a product are divided into the three above mentioned categories. The functions are then graded on a scale from one to five based on their importance. Performing a functional analysis and assembling these into a functional specification will give you a good overview of what the important and less important functions of a product are.

3.3.2 The KJ method

The KJ method is used for analyzing data, e.g. from interviews, user tests or focus groups and categorize this data so that the most common and most important problems, demands and opinions are easy to detect (Perry and Bacon, 2006).

The KJ method is commonly executed with help of whiteboards and post-it notes. Comments from interviews or user tests are written on a post-its and categorized so that related comments regarding e.g. design flaws or problem areas are grouped together. When all the data deemed significant from the interviews or user tests has been analyzed and categorized, the distribution of items in each category provides an indication of problem areas to focus on. The creation of the categories and the classification of interview data is subjective.

3.3.3 Usability testing

The usability testing technique is a technique where the usability of a product or a system is evaluated through user testing. Testing the usability with actual users is fundamental to be able make a well designed UI. It is the only way that you with certainty can find the out the exact problems and difficulties the user has with an interface (Nielsen, 1993). According to Nielsen, usability testing can be done with two purposes, as a formative or a summative evaluation. A *formative evaluation* is done to help improve an interface as a step in an iterative design process. A *summative evaluation* is done to determine the quality of an interface.

The test is performed by letting users perform tasks and while studying if the user is able to solve the tasks in the intended way. The test tasks for the user to perform are based on a realistic scenario created for the product or system. The test is commonly observed by an observer or by cameras so that notes can be taken.

3.3.4 Use case

A use case is a list of actions made by a user on a system and can be based on a scenario. The use case is often created to showcase some kind of goal for a project. By studying this use case, important requirements for a software development project are easily detected (Jerald, 2015). The developed use cases can be seen in appendix H.

3.3.5 Brainstorming

Brainstorming is a method for coming up with as many ideas as possible. There are many different methods of brainstorming but they all have the same ground rules (Nilsson, Ericson, and Törlind, 2015):

- Do not criticize any ideas.
- No ideas are to wild.
- Combine and improve ideas
- Aim for many different ideas rather than few well thought out ideas.

A brainstorming session is often executed by letting the members of the session freely spawn ideas regarding specific topics, for some duration of time. This process is done, one topic at the time, iteratively until all topics been processed.

3.4 Interviewing

There are several interview methods but only two has been utilized in this study. These two are described below.

Structured interview - An interview method where all interviewees are asked the same predecided questions (Ideas, 2012).

Unstructured interview - Unplanned interviews, questions are made up as the interview goes on (Ideas, 2012).

A combination of these where utilized in this study called a semi-structured interview (Cohen and Crabtree, 2017).

4

Method

In this chapter the execution of the project is described. The different phases of the project are described chronologically.

4.1 Analysis of functionality and features

The first tasks of the development process was to decide what features were desired in an OS UI. Since the scope of the project just cover the most essential features an analysis was needed to determine which the most essential features were. To find these features and determine the overall OS UI specifications, interviews and a function analysis were conducted.

4.1.1 Function analysis of OSs

To decide what features that an OS UI requires and to evaluate their respective importance, traditional 2D OSs were analyzed. A function analysis was conducted to structure the functions of an OS UI and analyze which of these that are the most important.

The function analysis theory described in 3.3.1 was applied. The OSs Windows and OSX were studied and analyzed, and the results were assembled in a document. A summary can be found in the result section 5.1.1 and the full function analysis document can be found in appendix A.

4.1.2 Interviews on OS features important to users

After the functional analysis was completed, interviews were conducted to gather knowledge of what functions users consider important in the UI of an OS. Semi-structured personal interviews were held. The subjects were chosen with the intention of forming a representative sample of different types of experienced computer users. All interviewees had in common that they all use computers in their everyday life. A total of seven subjects were interviewed. In appendix C information on the interviewees can be seen. The interview form used for the interviews can be found in appendix B.

The questions asked had the purpose of exploring what the interviewees' most common computer tasks are, their perceived biggest differences, disadvantages and advantages of different kinds of OSs and how they use and organize their computers.

The data gathered from the interviews were analyzed and important remarks regarding structures, interactions and general design of OSs was summarized and analyzed with the KJ method, see chapter 3.3.2 for in depth info on the KJ method.

4.2 Concept development and user testing

The KJ method resulted in a first iteration of concepts for a VR UI. In this section the development of the first concepts and the testing of these is described.

4.2.1 Developing concepts

The initial analysis of the functionality and features of an OS suggested some features that were essential. An initial brainstorming session was conducted with the goal to find as many ideas on implementation on the above mentioned features as possible. A range of solutions and implementation ideas regarding the different topics were developed. More information about brainstorming can be found in chapter 3.3.5.

Three implementation ideas on each feature were selected and assembled into three different concept levels in UE4. Several applications was also implemented and combined into the levels.

4.2.2 Concept test setup

The three concept levels created were to be tested by real potential users. A test for every important feature of each concept level was created. See the chapter 5.2 for a full description of the concept levels and see appendix D for the test protocol. The test was divided into three parts where the user tried one test level at a time, followed by a short interview with questions regarding the specific test level. The full interview protocol can be seen in appendix E. After the user had tried all three test levels, some comparative questions were asked, these can also be found in appendix E. The test was modeled around the usability testing technique, this technique is described in depth in chapter 3.3.3. The test was performed with the tester standing.

The order of the test levels were shuffled between each user to decrease the risk of faulty indications due to the factor of learnability, the users getting accustomed to the system and therefore performing better and better in the tests.

Test subjects were obtained through advertising the test to acquaintances of the project group. A total number of eight tests were held. Information on the test subjects can be found in appendix F.

This test is referred to as "Test 1" in subsequent chapters.

4.3 Concept evaluation and further development

After the testing of the concepts was finished the results in form of notes from the tests and interviews were studied. Common mistakes were listed and the relevant comments from the interviews were summarized, see appendix G. The summarized results were evaluated and important comments and notation regarding the different implementations of the above mentioned important features were particularly acknowledged. The most successful parts of the implementation concepts were chosen for further development. This was used as the foundation of a final OS UI concept.

In order to identify additional functionality, that would have to be implemented, three use cases were developed. The use cases were based on three probable scenarios of where a VR OS could be used. The full use cases can be found in appendix H. The use cases led to the development of several integrated VR applications that could be interacted with through the use of the OS UI.

A new, final UI was developed with an increased amount of functionality and applications, to evaluate several types of interactions and observe genuine user behaviour. Functional applications were implemented when permitted by time, but some applications were developed as simple nonfunctional concepts.

4.4 UI evaluation testing

To evaluate the final OS UI design another usability test was conducted. This test is subsequently referred to as "Test 2".

The test consisted of several tasks that the test participant were asked to perform. The tasks were constructed so that most features and applications were tested. The full test procedure can be found in appendix J. This test was slightly less structured than test 1, meaning that the test participants were encouraged to play around with the functions and the environment, and the test tasks could be solved in any order.

The test was performed with one test participant at a time being guided by one test guide while one observer documented the test through notes. The observer documented all feedback spoken by the tester, but also problems perceived by the observer that the tester did not mention.

The test was performed by a total of 8 test subjects, were where two test subjects had also participated in the previous test. The test subject were again obtained through advertising the test to acquaintances of the project group. Information on the test subjects can be found in appendix K.

The results, consisting of detailed notes, from test 2 were summarized and analyzed with the KJ method, see chapter 3.3.2 for further information on the KJ method. The results of the analysis were then used for a final evaluation of the program. The results from the analysis can be found in appendix L.

5

Result

In this section the results of the interviews and test observations are presented as well as the developed test environments, the final OS UI result, and how the different parts of the system were implemented.

5.1 Analysis of functionality and features

During the early development phase of the project, three different types of applications were identified. These were 2D applications, integrated VR applications and fully immersive VR applications.

The fully immersive VR applications are applications which completely replace the UI of the operating system, similar to a fullscreen application in a 2D operating system. An example would be a VR game developed by a professional game studio. While the fully immersive VR application is running, all the interaction and the display of the application will be managed by the application itself.

Integrated VR applications can be compared to traditional window mode applications, in a desktop OS, or even widgets in systems such as Android. The difference between these and the fully immersive is that the integrated VR applications run inside of the OS UI as a component. The application does not require the whole screen of the HMD but only a part of the virtual space within the OS UI. An example could be a calculator that functions and looks like a real calculator.

2D applications are applications as we are used to seeing them on our computer or smartphone screens. Typically a two dimensional UI designed to be interacted with through WIMP.

To not limit the user all three types of programs, 2D applications, VR applications and fully immersive VR applications needs to be supported in the OS UI. A discussion reasoning about this can be found in section 6.2.1.

5.1.1 Function analysis

The result of the function analysis indicated that the most used functionality of an OS was found in the taskbar or the start menu, as it is named in Windows. It provides easy access to tools and programs. The ability to access frequently used apps globally was a feature identified as important. Having applications statically placed in the virtual environment could potentially cause situations where the user is working on something and needs to use a specific tool, and then have to move across the environment to get this tool, and then move all the way back to the work area, which would disrupt the user's work flow.

Another important function of an OS is letting the user manage files and folders. The user should be able to open folders, make copies, delete files etc. The complete list of functions can be found in its entirety in appendix A.

5.1.2 Interviews

The data obtained from the interviews showed that there is a great variation in what different users expect from their OS.

Most interviewees frequently used the desktop of their computer as a place to organize shortcuts to files, folders that they are currently working on. They found that the desktop was a vital component in their way of managing their computer work. While most interviewees organized their files in a folder and subfolder hierarchy some felt that navigation through these folders can be difficult, visual clarity of folder structures could help with making navigation easier. The need for customization of the OS was quite low for the exception of being able to change the desktop background picture, almost all interviewees felt this was an important feature. Some interviewees felt that the OSs of smartphones, while it lacked much of the functionality of a desktop OS, was easier to use because of the motions used to control it were simpler than using a mouse and a keyboard.

5.2 Concept prototypes

The three prototypes developed for test 1 were intended to test different interaction concepts for the most important features in the OS UI. The three test prototypes are referred to as test level 1, 2 and 3. The primary test variables, each representing a tested aspect, in each level were categorized as environment, global menu, navigation and interaction.

Figure 5.1: *To the left, the environment used in test level 1. To the right, the tool belt menu.*

Environment: A picture of the environment can be seen in the figure 5.1. The environment depicted a large open field, modeled to be similar to a grassy mountain range.

Global menu: A picture of the menu accessible in test 1 can be seen in figure 5.1. A tool belt menu that statically floats around the user's hips. The menu contained several 3D icons representing program shortcuts. The menu contained a kill zone for shutting down programs.

Navigation: The player cannot navigate in the world more than the player's actual movement in the real world, movement enabled by the motion capture cameras. No teleportation or alternative movement was enabled.

Interaction: Programs can be started by dragging 3D icons from the tool belt menu, and dropping the icons where the user intends the program to start. Dropping an application in the kill zone would cause the application to close. Programs can be moved around by grabbing and dragging them, this was possible in all the three test levels.

5.2.1 Test level 2

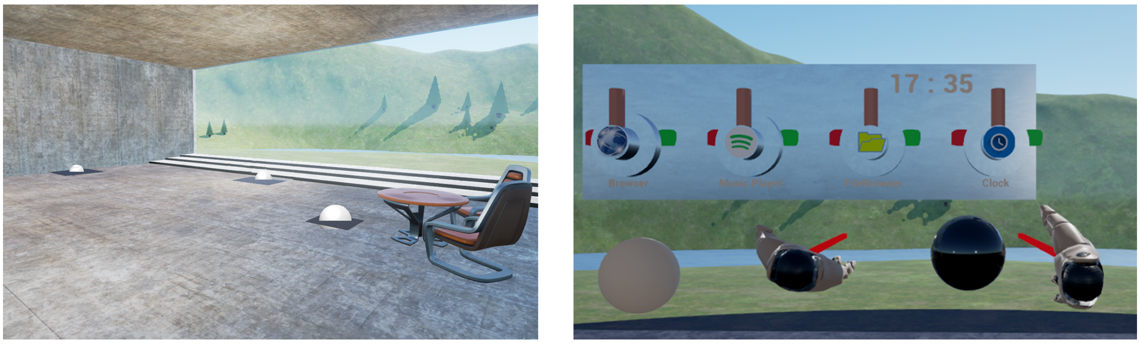

Figure 5.2: *To the left, the environment and to the right, the lever-based menu used in test level 2.*

Environment: A picture of the environment of test level 2 can be seen in figure 5.2. The environment is a single scarcely furnished large room with concrete walls. One of the walls were replaced with a window, overlooking a hill with some trees and a pond. Specific locations where teleportation is allowed were marked by differently colored squares.

Global menu: A picture of the menu accessible in test level 2 can be seen in figure 5.2. The menu is attached to the player's left hand and appears dynamically through rotation of the wrist. The circular knobs with icons on are grabbable handles that can be rotated towards the green or red dots to start and close applications.

Navigation: The player can use teleportation but can only teleport to certain allocated "snap points" in the room, previewed as white balls on the floor of figure 5.2.

Interaction: Programs are started through a 60 degree twist of the levers towards the green dot, and closed by rotating the levers towards the red rot.

5.2.2 Test level 3

Figure 5.3: *To the left, the environment and to the right, the 'taskbar' menu used in test 3.*

Environment: The environment, shown in Figure 5.3, consists of a furnished house with two smaller rooms. A city skyline can be seen out of a window placed in a corridor between the two rooms.

Global menu: The menu, shown in Figure 5.3, is a menu bar floating in front and above the users, consisting of three buttons and a clock, not visible in the Figure 5.3, mounted onto a black panel surrounded by a half transparent border.

Navigation: The player is allowed to freely teleport around and can also utilize a crawl movement by tilting the thumbstick on of the motion controller in any direction. The crawl movement causes the camera to slowly move in the direction the of the tilted thumbstick.

Interaction: A program is started by touching one of the buttons on the global menu, shown in Figure 5.3. A running program is indicated by its corresponding button being lit up and is closed by touching the same button again.

5.2.3 Test application prototypes

This section briefly details the prototype applications that were used during the testing. The applications shown in figure 5.4, 5.6, and the left clock in figure 5.5 were accessible from the users menu during testing.

The filebrowser displayed files in a grid. It has a handlebar on one side for the user to grab in order to move the window. On the other side of the handlebar there is a folder tree structure allowing the user to navigate from the root folder and any subsequent subfolder leading to the currently displayed folder, this is called breadcrumbs. The folders are displayed as folder icons and files as paper icons while the images are displayed as image thumbnails.

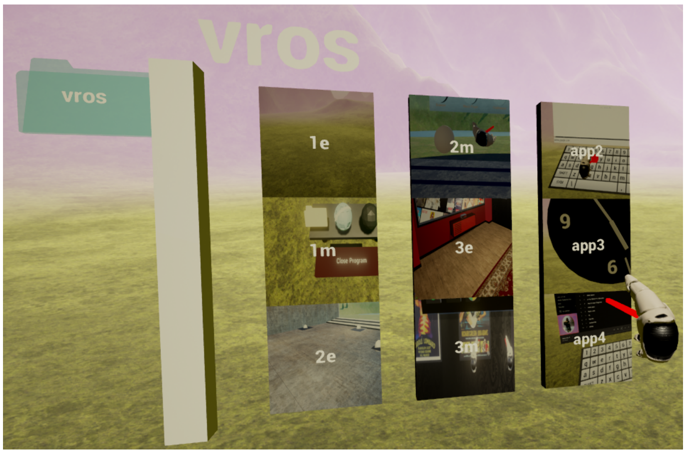

Figure 5.4: *The first implementation of the file explorer, used in test 1, consisting of a navigation tree to the left and the content of the current folder to the right, divided by a handle.*

At the point of the first test the filebrowser only displayed a small selection of all files in a folder, and the user was only able to navigate the system and create shortcuts to files or folders to place in the world. Interacting with a file did nothing while interacting with a folder displayed the content of the folder in the same window. Interacting with a folder shortcut placed in the world would open a new window.

Figure 5.5 shows the two versions of clocks used in test 1. One analogue and one digital. Both showing current time in hours, minutes and seconds. The first is accessible through the global menus in test levels 1 and 2. The latter, not accessible through any of the global menus, but instead integrated as a part of the interior of test 3.

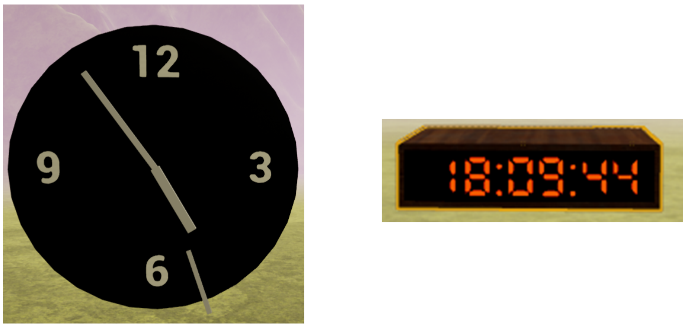

Figure 5.5: *The two clock applications accessible in the test.*

Figure 5.6 shows the concept music playing application used in test 1. The application was not an implemented music player, but rather a screenshot of the windows application Spotify (Spotify Ltd, 2017). The figure 5.6 also shows a web browser utilizing the built in web-component in UE4, combined with a functioning keyboard activated by a button on the hand controller. The url bar was kept below the screen for reachability purposes.

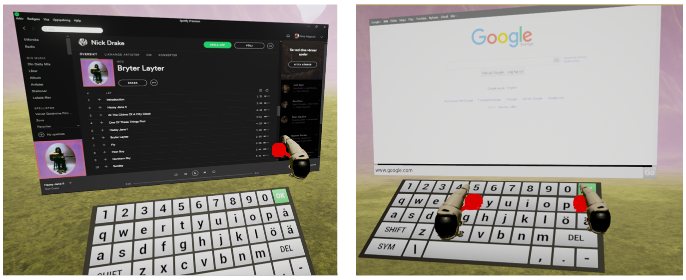

Figure 5.6: *To the left, a concept music player, and to the right a web browser. These applications conceptualize interaction with legacy 2D apps.*

5.3 Feedback from concept tests

This section details a summary of the feedback received from the first tests. The results from the first testing round were used to evaluate the tested implementations and interactions, and which of these to develop further during the project, were selected. Information regarding the test procedure, the interview questions and the test users can be found in appendix D, E and F. All users found that the optimal UI would have consisted of different parts from each of the three test levels.

5.3.1 Environment

There was no consensus on which environment was the best one. A few users preferred the open environment of test level 1, because they felt that the open environment would not limit them when working with programs. At the same time a lot of users disliked it because of its lack of detail, such as objects like plants in the environment.

The large room in test two was the least preferred. User felt that the wall and floor materials, which resembled concrete, were cold and rough. One user even described the room as a bunker.

Environment consisting of multiple rooms in test level three was preferred the most. The users described this level as the most detailed. The cause of this is probably due to the larger amount of furniture and decorations compared to the other levels. This higher degree of detail made the users perceive the level as cozy and homey.

5.3.2 Navigation

The vast majority of users preferred being able to move through teleportation, they felt trapped when not able to move around more than they could in test one. They expressed a desire to be able to explore the environment. The users generally did not use the teleport function in order to complete any task in any practical way, but rather utilized it when they first started a new level to explore it, once the level was explored the users stood still and performed their given tasks. Users also reported that it was hard to teleport to objects suspended in mid air, as the teleporter only indicated the target locations on the ground.

5.3.3 Interaction

The most well received interaction solution was the drag and drop feature to start programs. The users expressed that they liked being able to control where programs opened by dropping it in the environment.

The twist-to-start function in level two was the least preferred since the users really disliked the twist motion and felt it was too complicated.

The interaction with the buttons in test 3 were found easy for the user to grasp. However, the users generally found it quite inefficient since the buttons were not very responsive. Another downside pointed out was the fact that the menu, onto which the buttons were attached to, was quite far away from the user.

5.3.4 Global menu

The opinions about which menu was the best varied among the users. Certain traits of the menus were unanimously considered as useful and these trait are described in the text below.

The tool belt menu of level one was placed too low and the fact that it did not rotate with the player was a source of frustration among users. The tool belt's placement often obstructed windows or programs the user worked with. Although the concept of grab and drag introduced in this menu, to start programs, was well received, as mentioned in 5.3.3.

The hand menu of level two had both good and bad aspects. The twist of the wrist motion needed to get the menu to appear was disliked, it was complicated and a little too sensitive. The fact that the menu followed the user's hand was very well received by the users, as they felt that they had more control over the menu.

The large floating task bar of test three was considered to obstruct the view and be hard to reach. Most users expressed a desire to be able to toggle the menu on and off, as they felt that the menu broke their immersion.

5.4 Use cases

After the first test, use cases were created to get testable goals to showcase some practical usage of the UI. The use cases were created based on three different personas which had the intention of representing different kinds of computer users. Here follows a short description of the three personas:

- A user utilizing the computer with specific software as a tool in their work and using it as a means of communication.
- A user focusing on using the computer for gaming and looking at gaming related content.
- A social media interested user, using the computer for communication through different means, such as messages, pictures, videos, and social media.

The full use cases can be found in appendix H.

5.5 Final OS UI design

A new UI based on the information gathered from the first concept test was created. The design specifications based on the data from the concept testing is described in this part.

5.5.1 Environment

The first test showed that the preferred environment varied between users. Most test subjects found the environment of test level 1, seen in Figure 5.1, and test level 3, seen in Figure 5.3, most appealing. The perceived sense of freedom given by the first level and the level of details of the third level was taken into consideration in the design of the final level.

The final environment, partly shown in Figure 5.7 and Figure 5.8 consists of an open-planned house surrounded by an archipelago and some trees. The latter modelled by an UE4 community member, Fighter5347, available for free for anyone to use. The indoors area is divided into two rooms, through the use of plateaus. The rooms are partly furnished and several applications are placed at specific areas, aiming to add more realism to the environment at the same time as further supporting the users spatial memory. The latter by dividing the open space into smaller areas which can be utilized for specific tasks.

Figure 5.7: *View outwards of the user accessible area in the VR UI environment.*

Figure 5.8: *View inside of the user accessible room in the VR UI environment.*

5.5.2 Navigation

The feedback given regarding the navigation in test 1, see subsection 5.3.2, showed that the vast majority of test subjects preferred being able to move around through teleportation. Based on this knowledge the final navigation is implemented as a combination of free teleportation, tested in test level 3, and teleportation to fixed locations, tested in test level 2. The first with the aim of providing the user a way to fully explore the environment, the latter with the aim to make it easier to navigate to specific areas within the environment. These specific areas, which were called Navigation snappoints, are highlighted by green bubbles when the user aimed the teleport, which can be seen in figure 5.9.

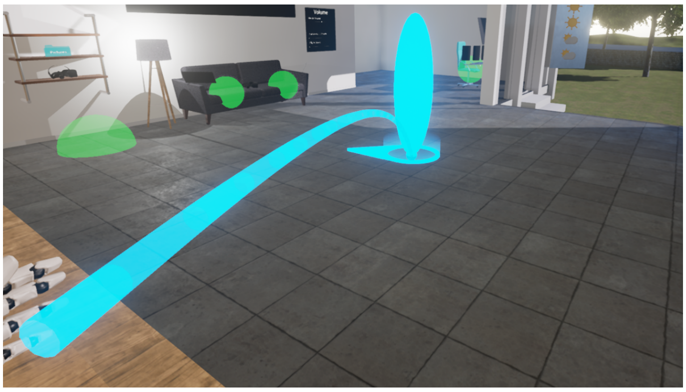

Figure 5.9: *A teleporting user, the green spheres represent snappoints. The blue line, extending from the users hand, preview where the user will be located after teleporting.*

The teleportation is executed by tilting the thumbstick of the motion controller in any direction and aiming towards the destination. By tweaking the direction of the thumbstick while aiming, the user can choose the direction to face after the teleportation. This feature is indicated by an arrow at the location of the destination, pointing in the direction the user will face. This can also be seen in figure 5.9 as a blue line, and an arrow on the floor indicating the direction the user will face after the teleportation. As can also be seen in 5.9, a stretched sphere was added to indicate where the teleport location was in relation to objects floating in mid air, as test users reported that they often teleported beyond these objects.

5.5.3 Interaction

The feedback given regarding the interactions in test 1, see subsection 5.3.3, showed that the test subjects preferred grabbing and releasing a 3D icon, over turning a lever or pressing buttons, in order to start an application. This action is preserved and refined in the final set of interactions. An image of a user grabbing a cube icon from the hand menu can be seen in figure 5.10.

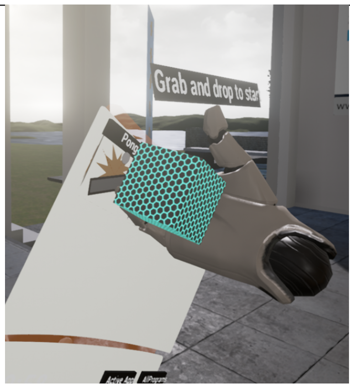

Figure 5.10: *A close look at the users right hand which is starting to grab a hologram 3D icon, which is appearing over the Pong hand menu shortcut.*

As in the first test, the grab interaction is also used to move applications and objects around. This action is performed by pressing and holding the index finger button on the motion controller, see 5.8.3. Grabbing a running application and then throwing it away will cause the execution to stop, as well as making its visual representation disappear.

By pressing and holding the palm button on the motion controller, the hand will perform a pointing gesture, see 5.8.4. This activates the point mode, primarily used to interact with the web browser, the filebrowser and the keyboard, but also to press buttons and drag sliders.

When interacting with an application using point, alternative actions can be performed by pressing and holding the alternative(alt) button, located on the top of the motion controllers. The alternative action depends on the context. While interacting with the filebrowser, pressing the alt-button will display a context menu for the poked object. While interacting with the Web browser, pressing the alt-button will produce the same action as a left mouse button click would on a regular PC. To trigger more alternative actions the alt-button can be pressed while grabbing something, such as copying files from one folder to another, while interacting with an object using grab.

In test 1, short laser pointers, used to interact with the keyboard, were attached to the hands, which can be seen in figure 5.6. In order to increase visibility close to the hands, these are not visible all the time in the final implementation. The interaction with the keyboard is done by solely pressing the buttons while in point mode, similar to how one would type on a physical keyboard. In the final implementation, the laser pointers are only visible while pointing towards a web browser, to enable interaction at greater distances than the arms length.

To enhance the sense of presence, haptic feedback is given through vibration of the motion controllers when interacting with objects.

5.5.4 Global menu

The feedback given in test 1 regarding the three different global menus, summarized in subsection 5.3.4, showed that each version had both good and bad qualities. The most prominent negative feedback was the absence of a way to hide the menu, while the most prominent positive feedback was the placement of the hand menu, shown in figure 5.2, and the action of grabbing and releasing 3D objects in order to run an application. Based on this information, a new global menu was implemented with the aim of removing what was considered bad, while refining those parts considered good.

The final global menu, shown in Figure 5.11, is hidden by default and shown by pressing and holding one of the buttons on top of the motion controller. It can be shown on either hand, taking into account that the dominant hand might differ from one user to another. The menu consists of two panels mounted onto a back plate. The top panel displays icons representing shortcuts to applications that are frequently used, analogous to the task bar found in Windows 10. The bottom panel displays shortcuts to the applications "Active apps" and "All programs" in order to minimize navigation to these commonly used applications.

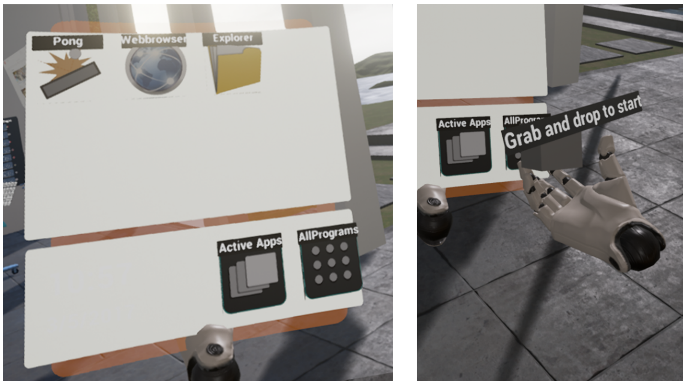

Figure 5.11: *To the left, a user looking at the hand menu on the left hand. To the right, a hint text telling the user what can happen when 3D icon is grabbed*

While hovering the hand over one of the icons, a small 3D representation of the application is displayed in front of the icon. In order to run one of the applications, the user grabs this 3D representation and release it at the position where it wants the application to be started.

5.5.5 Hints

As mentioned in subsection 5.3.3, in test 1, several test subjects had a hard time understanding how to interact with the global menu. However, when explained, the majority of test subjects preferred this way of interacting with the global menu.

In the final implementation hints about how to perform interactions are introduced, in order to guide the user in the right direction.

The hint consist of a small text box displayed above the hand close to an interactable object, shown in Figure 5.11, providing concise instructions.

5.6 Application concepts

This section details the applications that were implemented to be used in the second VR test session. An overall theme was implemented for several applications. Application windows such as "All programs", "Running programs" and the filebrowser all had similar looks and color schemes. They all contain two bent rectangular areas, a back plate and a front plate. The front plate contains all the information while the back plate simply acts as a border.

5.6.1 Filebrowser

Compared to the filebrowser used in test 1, the overall design has been preserved, but refined in order to improve navigation and clarity. The filebrowser can be seen in figure 5.12

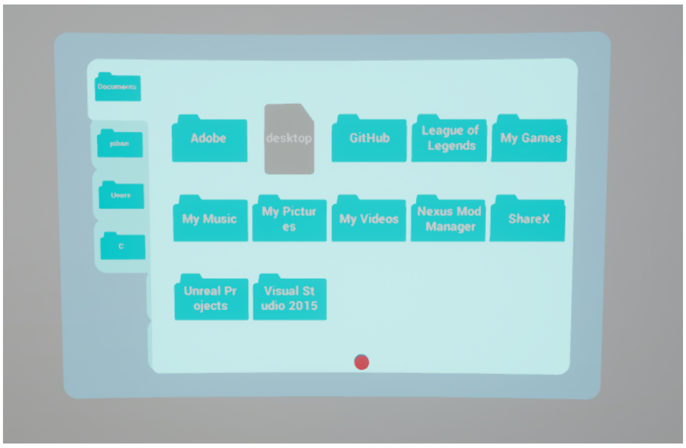

Figure 5.12: *The file browser application. An overview of where the user is in the folder hierarchy is displayed to the left, while files in the current folder is displayed in the larger central area. The red dot represents the current page, which is the only page for the current folder.*

The final filebrowser consists of a main content area in the center and a navigation tree to the left. The main content area can fit 15 icons and can be swiped to the left and right in order to navigate through files if there are more. A small indicator in the bottom shows how many pages are available and which one the user currently is viewing. The indicators can be pressed to directly change to that page. The tabs to the left shows a folder tree were the active folder always is on top.

Besides the refinement of the visual design, the final filebrowser provides the user with the features of moving files between folders, duplicate files as well as deleting files. The action of moving an object is done by grabbing a specific file or folder and release it upon another window. The action of duplicating a file is done in the same way, while simultaneously pressing and holding the alt-button. The deletion of a file is done by alt-poking the specific file, and choose the option of deletion from the displayed context menu.

Another feature added to the final filebrowser is an image viewer. When poking an image icon within the filebrowser this image will be expanded and displayed in front of the user. This process was made easier by using the Victory Plugin, created by the community member Rama. This plugin enabled us to load the image as a texture and then display it.

The filebrowser was implemented by creating a blueprint library in C++, to provide file management functions in blueprints. The application consist of two parts, the display window and the files. Both the display window and the files utilize the GrabInterface for moving, but the files also use the GrabInterface to create shortcuts to files and folders. The program interface was implemented by the display window to provide a startup path for the window. The display window also implemented the Touchable interface, to provide a swipe function, and the Context menu button that caused the window to display the next page of files in a folder. This swipe feature was implemented by measuring the distance and speed that the hand traveled when touching the window. The IHint interface was implemented by files to provide hints to the user of what would happen when the file was grabbed, released or poked.

5.6.2 All Programs

The purpose of the "All programs" application, shown in Figure 5.13, is to give the user a quick and easy way to access all installed programs. The application is reached from the global menu and consists of a main content area populated with icons representing each and every installed application. The 3D icons are described in 5.8.8

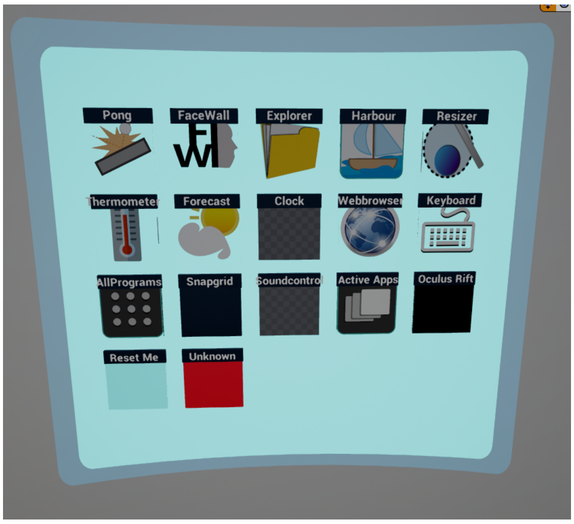

Figure 5.13: *The application holding shortcuts to all installed programs. Each icon creates a 3D hologram when the user puts his hand in front of it.*

5.6.3 Web browser

The web browser application, shown in Figure 5.14, consists of a window showing the current web page and an address bar. The interaction is done in a way that heavily resembles the interaction with a traditional web browser, i.e. using mouse or fingers and a keyboard.

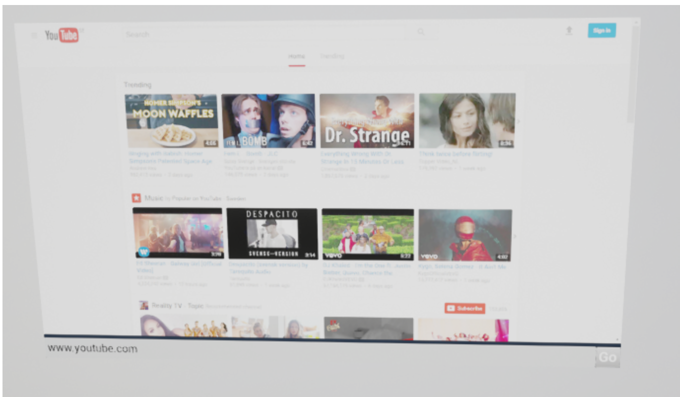

Figure 5.14: *The web browser application.*

When pointing at a browser a laser beam appears, connecting the tip of the finger with the point on the screen the user is pointing towards, creating the equivalence of the traditional mouse cursor. Continuing the analogy with a mouse, the alt-button is the equivalence to the left button mouse click. The text input is done by selecting a text field and type on the virtual keyboard.

The interaction with the web browser application also conceptualize how interaction with 2D-applications, such as Microsoft Word and Adobe Photoshop, in general could be made.

The web browser core functionality was provided by UE4. Due to a bug that exist in the version of UE4 that was used, the out of the box implementation of the web browser scales in a way that makes it impossible to interact with. This problem was solved, partly, by rescaling the the widget several times. It was also modified to increase readability, extended with interfaces to provide grab and poke interactions, as well as generating force feedback on the hand controllers when poked. The laser pointer was implemented by creating a raytrace from the hand when pointing. When the ray trace hit a web browser the laser pointer would be displayed and the alt-button would trigger a left click event to the browser, done by moving the widget interaction component from the finger to the point of hit.

5.6.4 FaceWall

The FaceWall application, shown in Figure 5.15 is designed to conceptualize how a user, in an effective way, could share images with its friends. The concept is built upon the idea of having multiple users subscribing to each others FaceWalls. When an image is added to one FaceWall, users subscribing to this FaceWall will be able to view the same picture on their own FaceWalls. This application was developed based upon one of the use cases.

Figure 5.15: *The social application concept FaceWall. Users can stick photos to the wall by grabbing them and throwing them at the FaceWall.*

The implementation illustrates how the FaceWall would behave for one particular user. The application consists of a canvas onto which users can throw images. When an image is thrown onto the canvas, the image will be attached at a random location. Multiple images can be shown on the canvas simultaneously. In this version, the features of subscriptions as well as viewing friends pictures is omitted.

The FaceWall was implemented by checking all objects that begin overlapping an area in front of it. If the object was a file of the type Image the path from the file would be extracted and sent to the Victory Plugin to load the image as a texture and display this texture at a random location on the FaceWall.

5.6.5 Harbour workstation

The Harbour workstation is meant to conceptualize how business applications could be integrated.

The scenario modelled is a logistics company handling traffic and docking of boats in Gothenburgs harbours, this is based upon one of the use cases. The task is to guide each boat to a vacant and suitable dock by matching the incoming boats sizes and destinations to corresponding vacant docks. In case of any problems regarding the docking of a specific boat, the employee will have to inform the captain.

The implementation of the concept, shown in Figure 5.16, consists of one panel showing incoming boats, an overview of the harbour together with panels showing the status of each dock.

Figure 5.16: *A professional workstation concept application*

Each boat is grabbable and shows information about its size and its destination. Linked to each boat is also a telephone, which when held to one of the users ears will produce a dial up tune, illustrating the phone call to the captain of the particular boat. This sound asset was acquired at Soundbible (Soundbible, 2017).

The harbor system used GrabInterface for moving the ships, as well as moving the whole system to facilitate better ergonomics. The DropArea interface was used for the docks and checked if the dropped object was a boat, and would then destroy the ship and change the color of the dock. The telephone used the Lerp Helper to smoothly move the telephone back to the boat when released, and the IHeadCollision interface to trigger the dialing sound. Sound was played with functionality built into UE4.

5.6.6 Keyboard

The keyboard application, shown in 5.17 provides the user a way to type without using a physical keyboard. The keyboard is automatically linked to any active text field and consist of Swedish letters, numbers and symbols as well as special keys such as space, enter and backspace. The typing is done by pressing the keys while pointing.

Figure 5.17: *The system keyboard. The user can press a key by enabling the poke action on the controller and touching a key.*

The keyboard was built as a 2D-widget containing rows of buttons. The interaction is handled by a widget interaction component hidden in the index fingers of the virtual hands.

5.6.7 Pong game

The pong Game illustrates how a VR game can be integrated as a part of the home environment. The application, shown in Figure 5.18, consists of a game board, two paddles and one ball. One of the paddles are controlled by an AI, the other one can be grabbed by the user and used to smash the ball over to the other side of the board. The goal is to make the ball cross the boards edge on the opponents side.

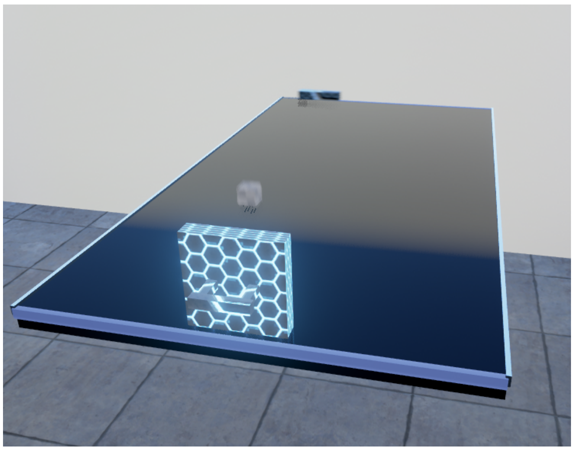

Figure 5.18: *The ping-pong game application with a grabbable user paddle and an opponent AI paddle.*

5.6.8 Resizer

The application resizer, shown in Figure 5.19, can be seen as a hand held tool giving the user a way to change the size of other applications and objects. By grabbing the Re-sizer and inserting one of the sides, marked by a plus- and minus sign, into another object, this other objects size will either decrease or increase depending on which of the two sides where inserted.

Figure 5.19: *The resizer-application. Touching the left side to a compatible object would shrink it, and the right side would expand it.*

5.6.9 Running applications

The purpose of the "Running applications" app is to give the user a clear overview of the currently running applications and a fast way of locating their position in the environment.

As seen in Figure 5.20, the application share the fundamental look and feel with "All programs", described in 5.6.2 and shown in Figure 5.13, i.e. a slightly bent frame consisting of a main content area surrounded by a border. The main content area is populated with icons representing all programs that are currently running, and does also provide information about the number of these currently running applications. The area shows up to 21 icons at the same time. If the number of running applications exceeds this maximum value, another page will be created, which can be reached by page indicators at the bottom of the main area. The 3D icons used is described in section 5.8.8.

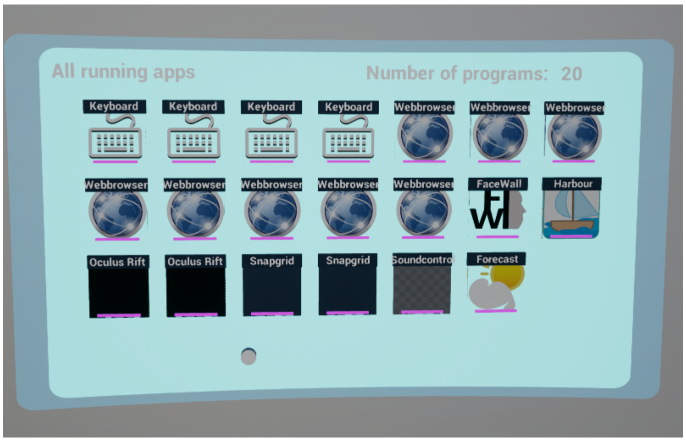

Figure 5.20: *The running apps indicator*

The indicator consists of a flow of small balls connecting the icon to the running application, giving the user a visual clue of where the application is located.

5.6.10 Snapgrid

The snapgrid application, shown in Figure 5.21, offers the user a way to organize objects and running applications and could be seen as an analogy to a pinboard.

Figure 5.21: *The application organizing app "snapgrid" with an attached folder*

The application consist of placeholders mounted on a slightly bent panel. By releasing an object or application close to the snapgrid, the object or application will snap to the closest free placeholder and attach itself to the board. The board can then be moved around as the user wishes, keeping the attached components grouped.

The snapgrid consists of a set of snappoints which use the IDropArea interface and Lerp Helper to smoothly align any object dropped on the snappoint.

5.6.11 Sound control

The sound control application, shown in Figure 5.22, consist of a panel holding three sliders, allowing the user to change the volume. This is done by pointing at one of the sliders and move the finger either to the left, to lower the volume, or the right, to increase the volume. The controls are divided into master volume, environmental sounds and effects sound. Environmental sounds only affect sounds that are produced depending on what the current weather status is. Lastly, the effects sound slider changes sounds that are produced by running applications.

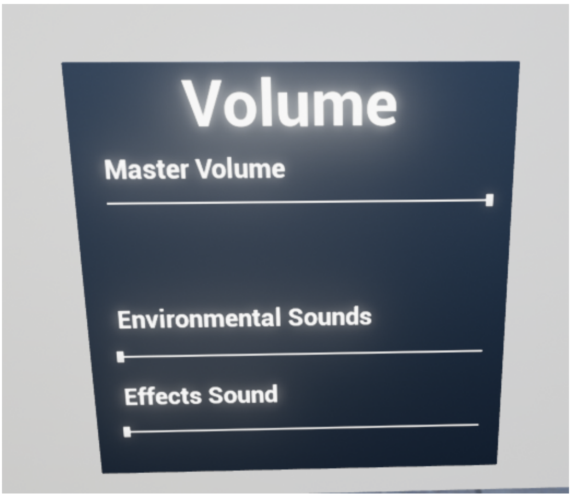

Figure 5.22: *The OS sound control.*

The way that this application works is that there are 3 different "sound classes". One for each different type of sound effect. In our case it is master, environmental and effects. After which, you assign a sound class to each sound asset. When you change one slider, only the sounds in that category will change.

5.6.12 Thermometer, weather forecast and dynamic weather

The thermometer application, shown in Figure 5.23, is designed to resemble an analog thermometer. By querying data, using JSON requests, from Yahoo Weather API (Yahoo Inc., 2017), the thermometer is able to show the current outside temperature in celcius at the location of the user, however in this version Gothenburg was set as a fixed location. This was made a lot easier by using the JSONQuery plugin made by community member Stefan Wijnker (Wijnker, 2014). This plugin was used in all other weather related applications.

The weather forecast application is an extension of the thermometer application. As seen in Figure 5.24, the application provides the user with information about the current weather and a four days weather forecast. Each day is represented by a date including an abbreviation for the weekday, an icon representing the weather and the maximum temperature during that day.

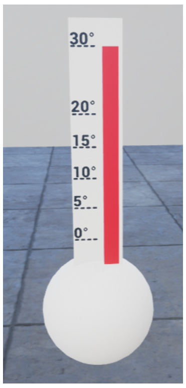

Figure 5.23: *The thermometer application.*

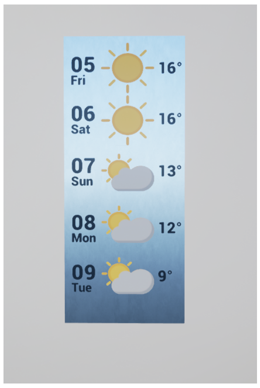

Figure 5.24: *The weather forecast application showing the predicted weather for the next 5 days, updated from the internet.*

The icons used on the weahter forecast was created by the users Freepik, Madebyoliver and Nikita Golubec and shared them for everyone to use Flaticon (Flaticon, 2017).

In addition to displaying the weather as a thermometer and a forecast, the current weather is also displayed as the actual weather inside the virtual world. This was achieved by changing the properties of the material that the skybox uses. If it is raining, the sky in the virtual environment changes as if it would rain in the real world. For example,the color and speed of the clouds can be changed, but also the amount. In order to boost the sensation of raining in particular, a particle system was added as well. The particle system and the corresponding materials was created by Youtube user PyroDev (PyroDev, 2017).

These visual effects were further enhanced with an audio of rain or birdsong that the user could control with the sound control. These audio assets was downloaded from Soundbible (Soundbible, 2017).

5.6.13 Portal

The purpose of the portal application, shown in Figure 5.25, is to conceptualize how a user can run a fully immersive VR application (FIVRA) from within the home environment without the excessive navigation through a portal such as Oculus Home.

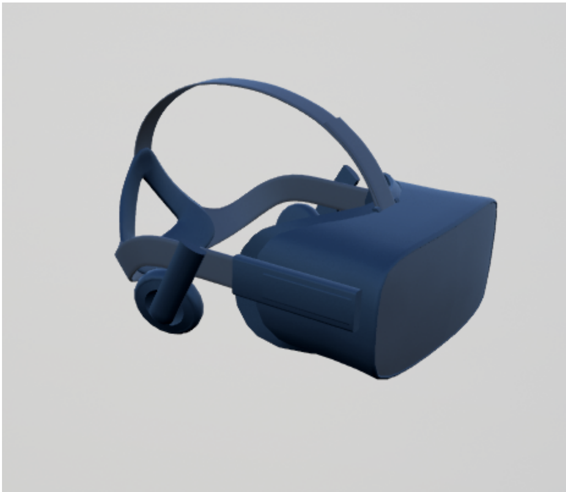

Figure 5.25: *An immersive VR application portal. Activated by putting on users head.*

The application solely consists of a 3D model resembling an Oculus Rift HMD and underlying scripts for changing levels.

By moving the model close to the head, the application will run the FIVRA instantly. Then, in order to exit the FIVRA, the reversed action is applied, i.e. grabbing close to the head and move the hand away from the head. This will take the user back to the home environment.

In order to test this concept, a fully immersive VR game was implemented, which is further described in section 5.6.14.

The Portal application implemented the GrabInterface and IHeadCollision interfaces to start a new program.

5.6.14 Throw balls game

The throw balls game, shown in Figure 5.26, illustrates how a FIVRA can be run from within the home environment by using the portal application, further described in section 5.6.13.

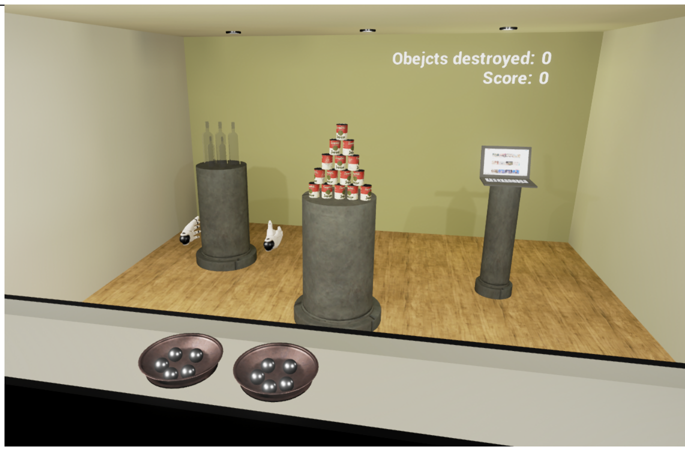

Figure 5.26: *An immersive VR application concept game.*

When running this application, the user will be taken to a new virtual environment where a game can be played. The game consisting of a closed room, a counter and three pedestals with various objects stacked upon. On top of the counter, in reach of the user, lies ten metal balls. The mission of the game is destroy as many objects as possible, by throwing the balls at them.

To exit the game, the user grabs close to its head and move the closed fist away from the head.

The throw balls game utilized the GrabInterface and the throw indicator to throw balls. It also utilized functionality built into the engine for destructible meshes and particle effects.

5.7 Polymorphism

Polymorphism (Bill Venners, 2017) was used extensively throughout the implementation of the system, through the use of generalized interfaces. In this way multiple concept prototypes could be easily created by implementing existing interfaces to provide different features. These interfaces were mainly called when the user made different actions, such as grabbing and dropping or pointing, but also for communication between different applications. All of the interface calls are done with events to avoid busy waiting and polling. The interfaces that were created can be found in appendix I.

5.8 Implementation of features and interactions

In this section the implementations of the features, which are not considered application concepts, are described.

5.8.1 Player pawn

The player pawn is the object that contains the camera and the hands in the application. These objects were provided by UE4 and were heavily modified during the development. The pawn was extended with keyboard and mouse input in order to be able to move the camera and hands when developing without access to a VR headset and motion controllers. They were also modified to facilitate all the interactions, such as grab, point etc., provided in the system.

5.8.2 Teleport

Unreal Engine 4 provided an out of the box implementation of a teleport feature. This provided feature was modified to change the input method as well as to introduce the teleport snappoint feature.

The teleport snappoint objects where implemented by overriding the teleport target location found when producing the arc for the teleport. If the arc end point overlapped with a teleport snappoint, the end point of the arc would be changed to the center of the teleport snappoint instead.

5.8.3 Grab

The grab action is triggered by pressing and holding the index finger button on the controller. This grabbing feature was provided by UE4 out of the box using an interface. The interface of this feature was replaced by the GrabInterface to make the feature more expandable, by moving the grabbing implementation from the hand to the grabbed object, as well as to implement the throwing feature.

The throwing feature was implemented by calculating the speed at which the hand was moved at the point of releasing an object. If the speed exceeded a specified threshold a boolean would be provided in through the interface that indicated that the object had been thrown. With the current implementation each thrown object must itself decide if it is to be destroyed when thrown, if so it will call the IKillable interface.

5.8.4 Point

The IInteractable interface was introduced to provide objects with the ability to be poked with the virtual index finger. The hand will call on this interface when the point buttons is pressed and a collision box, located on the tip of the index finger, starts to overlap an object that implement IInteractable. The action is performed by pressing the palm button on the controller.

5.8.5 Alt-button

The alt-button was implemented by providing a boolean value indicating if the alt-button was pressed or not, to the IInteractable, GrabInterface and IHint when these were interfaces were called.

5.8.6 Hint

Hints were implemented by adding a text above the hands. Every frame both hands will call the IHint interface on any overlapping object and set the hint text if applicable. The object implementing IHint may also use this call to do internal hints as well.

5.8.7 Lerp helper

The LerpHelper was created to facilitate easy and smooth transformations of objects. It was implemented as a wrapper for the lerp function provided by UE4, in order to use lerp in function calls and have it report back once the transformation was complete.

5.8.8 3D icons

A generic class "Hologram Button" was implemented to allow the user to start and manage programs. The buttons were used in the applications "All programs" and "Running apps", and in the user hand menu. The buttons consisted of a 2D icon, a label, and a 3D icon which would appear when the user hand approached, by triggering the IHint interface. The hologram icons can be seen in figure 5.13, 5.9 and 5.11. The buttons were created to dynamically change their appearance to match a specific design by reading the definitions in a "progDef" object. All "progDef" objects were specified in a helper class "programLibrary". The 3D icons were implemented to listen to the GrabInterface detailed in 5.8.3. The holograms were also programmed to be able to attach to a specific program, and when attached, instead of starting a program, summoning the attached program, by calling the Unreal method Lerp on the object position.

5.8.9 Context menu button

A context menu button was created in order to facilitate an easy implementation of 3D buttons. The buttons used calls to the ButtonListener interface to report when they were pressed. Buttons provide both ID and name as identifiers. The IInteractable interface was used to trigger the buttons.

5.9 Evaluation testing results

The results from test two were mainly positive. The test subjects generally felt that the functions were easy to control and understand. The full test results can be found in appendix L. Here follows a summary of all the results.

Environment The test users felt that the environment was pleasant. Further customization in furnishing of the room was desired by a few users. The users were not observed to be disturbed by the environment when executing tasks.

A couple of users were annoyed by the office chair since it did not detect whether the test user was actually standing or sitting down and it did not rotate with the player. The blue chair can be seen in the figure 5.7. Because of this some test users felt that is was obstructing them.

Customization of the room by being able to place pictures was appreciated by the users.

Navigation Many users were observed to teleport by mistake by accidentally moving the joystick multiple times through the test.

The users described the rotation through teleport as somewhat difficult to control in the beginning of the test but the learnability was good so they quickly learned how to use it.

Menu The users were observed to use and interact with the hand menu without any problems. Some users at first tried to interact with the program icons in the menu through pointing and touching but upon seeing the holograms all users quickly tried to grab instead.

One problem observed was that the holograms was too motion sensitive and caused flickering of the holograms to occur when the user tried to grab a specific icon.

File browser The breadcrumb system was observed as being somewhat hard to understand since multiple users failed to navigate as they desired through it.

Web browser Users felt it was difficult to see small details and text because of the low resolution. The difficulty to see small details also made interacting with the browser more difficult.

Keyboard The users felt it was confusing that the connection between the keyboard and screen was not explicitly shown.

Some users felt that using the keyboard was difficult since they were unable to watch the keyboard to see which buttons they are pressing at the same time as they are watching the screen to see what they are writing.

Interaction Many users did not understand how and when they could utilize the alt-button. The users had to get exact instructions to be able to use it.

The responses regarding closing program windows were mixed. All users understood that they could throw programs to close them and a lot of users liked it however there were some users that felt that the throwing action would become tedious in the long run.

The users had a little bit of trouble remembering which button that opened the menu and which was the alt-button.

Many users were observed to grab program windows in the center rather than along the edge when they were going to move the window. This caused them to sometimes accidently grabbing icons or folders rather than the windows itself.

Applications All users liked the idea of being able to play games, both like widgets in the room, like Pong, and fully immersive VR games.

Almost all users tried to interact with the boats and the map rather than the boats and the parking chart, other than that the users handled the application as intended.

The facewall was commented by users to be fun and easy to use.

Starting fully immersive VR applications by putting on a headset was found intuitive for all users, however closing the applications through taking of the headset was not as intuitive.

Overall experience Some user felt discomfort in not knowing where they were in the actual room. Furthermore users consistently lost track of where they were in relation to the hand control sensors, and got frustrated when they turned their back to the sensors and got degrade tracking performance. Some users commented on that they were using the test-leaders position as perceived by their voice to orient themselves correctly.

6

Discussion

This chapter contains discussion about the test results, analysis of the proposed solutions, and reflections of possible improvements.

6.1 Design processes and testing

This section contains discussion related to executed design processes and tests.

6.1.1 Different levels of detail and theme when testing environment

In the concept testing, the environment of test level three, consisting of several small rooms was the most popular environment. One of the problems discovered with the testing conducted in this project was that the level of detail between the three environments of the test levels varied. The large open level contained only a vast green mountain landscape while the rooms of test level 3 contained detailed objects like movie posters and furniture. This varying level of detail might have caused the users to feel that test level 3 was more well-made and therefore prefer it over the other. If this has been the case, aspects such as the functionality of the different kinds of environments might not have had as big an impact on the user's choice of preferred environment as the amount of detail of the level. For this project the functionality preferably would have had the bigger impact. No noticeable difference was observed in user effectiveness in performing the tasks connected to the different environments, however it is difficult to tell as the menu and navigation system was changed together with the environment.

6.1.2 Limited testing of VR sickness and ergonomics

Due to the short testing sessions of the tests, lasting around 15 to 45 minutes, any potential VR sickness or other types of discomfort caused specifically by long term usage could not be determined. Some users did proclaim that they felt VR sick from using the crawl movement ability, this was anticipated from the VR sickness theory. For information on VR sickness see 3.2.1. But since no tests aimed specifically at testing comfort during longer usage was preformed it is possible that the OS could have been further optimized for long-time usage.

6.1.3 Inexperienced VR users

All users in test 1 felt that the ability to teleport was an important feature for the OS even though they rarely utilized this ability when performing tasks in the OS. The need for teleportation might be due the fact that none of the invited test users were experienced VR users, the initial intrigue of seeing environments in VR for the first time might have affected their desire to explore. It might also have been due to the fact that the users visited these specific environments for the very first time, if they would have visited an environment they already were familiar with, the need to explore might not have been as strong. However the reason for the user's wish to explore could also be due to the design of the levels. Levels could possibly be designed in a way that discourages exploration. This could for example be done by creating an environment with a simplistic and empty feel, without much real world connection an user would be less prone to use their mental model for the real world in which they can move in.

6.1.4 Limitations of testers and interviewees

Because of the limited time in the project, the limited access to VR equipment and the cost of the VR equipment, performing the tests with random people was considered inappropriate, as the test participants needed to be reliable and not damage the VR equipment. Therefore, the test participants consisted mostly of acquaintances. While there were some efforts to to get a diverse sample of acquaintances by intentionally picking people with varied age and computer knowledge, the resulting test sample was overly representing males aged 20-30. This may have led to the final results being more favorable to young males. Furthermore, due to the testers knowing the developers, there's a risk that some feedback was withheld for emotional reasons.

Switching test leaders, the one giving the instructions for the test participants, could also have affected the result. Although the test had a set order, different test leaders might have given different level of help and assistance to the test participants, which could have affected the reliability of the results.

6.2 Final UI

This section contains discussion of the final developed UI based on the feedback received and documented during the second testing session.

6.2.1 Ensuring user adoption

The supply of VR applications is still relatively low compared the vast amount of 2D applications available. Also the most commonly used computer programs for tasks such as word processing, spreadsheets and email fall into the 2D application category. At the time of writing there are very few, if any, VR optimized applications for any of the above mentioned tasks. Therefore in order to make sure the user can continue to complete these tasks even without a dedicated VR implementation, backward compatibility is required. Future development might not have to have this kind of 2D application support since more VR optimized application alternatives are probable to be released and therefore the need for 2D app compatibility decreased.

6.2.2 Launching applications

Feedback showed that the test subjects liked to open programs by grabbing 3D icons, and placing them somewhere, and thereby choosing where the application is started. This gives more freedom to the user to choose by themselves where they want their applications to be, giving them more freedom over their environment. However, with this freedom follows a risk of things easily getting cluttered and unorganized, which was observed during the tests. Using stronger constraints, explained in 3.1.1, guiding the users to a preset structure could significantly decrease these issues but at the same time, the possibility to customize the environment would be decreased. In order to find a good balance between the freedom of customization and order, further tests would be necessary.

The icons were made in the form of 3D objects rather than simple 2D graphic icons in order to get the user to grab them instead of just click them. This was based on forming mental models based on affordances, described in chapter 3.1.1. Regular 2D icons are normally interacted with through clicking, a 3D shape in a VR environment was thought to encourage the user to try other means of interaction with the intention to increase the guessability of launching applications. The 3D icons appearance was animated, so that they physically expand. Having this salient feature, as discussed in chapter 3.1.3, was intended to improve the user's ability to discover the feature of dragging to start programs. Furthermore, a hint text was added above the user's hand when in proximity of the 3D icon to explain the feature. It was deemed necessary to have, even though it is a sign of bad guessability to have explaining text. If the usability test had been done with more experienced users, who already were familiar with the grab function, the help text might not had been as needed.

A prevention area was added around the 3D icons to ensure that the user would drag the 3D icon a sufficient length away from the starting location. This would encourage the learning of the starting method, while simultaneously implementing a possibility for the user to cancel the start of a program.

6.2.3 Navigation

Teleportation was the preferred means of transportation in the VR environment for most of the users present in the first test. The only alternative was the crawl function that also tested in test one. Theory regarding motion sickness problems in VR 3.2.1, supports that teleportation is a good way to navigate, because the user does not expect to experience any force of acceleration when performing instantaneous movement, which, as discussed in 3.2.1, can cause motion sickness. The findings of this study support that teleportation is the best means for transportation in a VR environment.

Users did not consistently utilize the teleportation snappoints, but were quick to understand them, and seemed to handle exact teleportation better with its aid.

6.2.4 File browser

The design was very inspired by existing file browsers, in an attempt to allow the user to reuse mental models of previously learnt programs. However, the introduction of the new input in form of hand-controller buttons, caused the users to still have problems performing basic tasks in the file browser. This proves the difficulty of translating well known interaction patterns, like the ones of 2D computer interaction into new VR interactions. Relying only on the user's mental model to make for easy usability can be problematic since the user needs to rethink the interaction patterns.

6.2.5 Action feedback

The test users expressed confusion with our interactions as they did not get reliable feedback when poking or clicking objects, and which button did what was not clear. This might, to some extent, have been due to the lack of VR experience, but it might have been due to the lack of feedback and absence of a uniform interaction schema for applications. In each application the user was forced to use different buttons to do similar things. Such as the keyboard required a thumb button and laser pointer from a distance, while the file system poke action required the palm button to be held while the user touch the files and folders up close. Relating the interaction to real life situations would possibly help with the usability in these situations making it easier for the user the figure out how to interact correctly.

6.2.6 Hand menu

The discoverability of the menu was low, as it does not trigger the human selection attention factors as mentioned in 3.1.3. Neither does it have an obvious mapping of the relationship between the button and the activation of the menu. This was a conscious design decision motivated by the feedback received during test phase one. The value of having a visual cue for discoverability purposes was evaluated to not be worth it, considering the users during the tests did not have problems remembering such central features, but did complain about UI elements they found annoying. It could have been possible to improve the mapping with a better affordance, however such was not found.

Having a menu connected to the hand of the user could improve ergonomics compared to a static one, since it allows the user to move around freely. The user does not need to relate themselves to the UI and is given more opportunities for variation, which improves overall ergonomics, see 3.2.2. However further studies need to be done to investigate if the UI interaction movements can become too repetitive.

6.3 Further development

This chapter discovers potential and possibilities for further development of the VR OS UI.

6.3.1 Potential accessibility features

Many of the most popular OS's today offer accessibility features, such as Windows 10 (Microsoft Corporation, 2017a) and OSX (Apple Inc., 2017a). These features are there to allow as many people as possible to be able to use their products.

Therefore, in the future these types of features could also be implemented in the OS that has been developed. Such as, the ability to open the hand menu without having to press the buttons on the controllers, if you are unable, for whatever reason, to press them.

6.3.2 Potential implementation of the OS in augmented reality

With the release of Pokemon Go by Niantic in 2016 (Niantic Inc., 2016) the world was introduced to augmented reality(AR) and its possibilities. This increased the popularity of AR, as explained in this article where a Swedish company sells "smart glasses" that are used in the industry to teach personnel (Fredelius, 2016). Niklas Rengfors who is interviewed, continues by saying that with the introduction of Pokemon GO, people has got a better understanding of what AR or enhanced reality is which in turn has increased the popularity.

With the increased popularity in AR it could be a good idea to investigate the possibility to make the OS in AR as well, since the AR market is only expected to expand over the next years (Merel, 2017). One advantage with having an OS in AR is that you do not lose your real-world perception which you do in VR, due to the need to wear a HMD.

However, despite of the possible improvements AR could bring, some things would have to re-designed since the interaction in AR and VR are not the same. To give one example, currently, Oculus has their "Touch" controllers (Oculus VR, 2017b) and HTC has their own controller (HTC Corporation, 2017) for interaction with objects in VR. Presently, the OS takes advantage of these controllers for interaction with different elements. Therefore hand tracking and interaction with menus and different elements would have to be redesigned with the lack of these. Even though these problems exists the over all design choices of the UI could stay the same.

6.3.3 Multiplayer

A commonly used feature in enterprise computer usage is to remotely view or control another individual person's computer. The current implementation of the OS does not support this feature, but the VR OS does not strictly prohibit the feature. A study examining the best implementation of "VR sharing" could be made. Examples of problems could be that directly sharing the view of one user would force the viewing user to follow the head-movements of the other party, which might cause VR sickness due to the loss of control, see section 3.2.1. In order to avoid VR-sickness another solution would have to be thought of. One possible solution could be to allow an individual to join another person in their VR room as a separate entity. This would allow two users joining the same room to interact with the same content. Joining other people and playing games in the same virtual environment together is today a possibility in Bigscreen beta, where you can join up to 4 people (Bigscreen Inc, 2017).

6.3.4 Customization of the environment

Users repeatedly expressed a desire to alter the environment similar to how you'd change the desktop background in a traditional OS. The average user does not have access to 3D modeling tools and the experience to create such assets. Therefore, with the possible demand for customizable environments a marketplace could be created. These types of marketplaces already exists for other OS's, such as Microsoft with their Store (Microsoft Corporation, 2017b) and Apple with App Store (Apple Inc., 2017b) where you can download applications and games. A similar solution could be implemented in our case where entire environments and different types of assets could be uploaded for the users to download and let them customize their experience.

6.4 The developed UI compared to other solutions

This section compares the implemented solution to existing OS:s and other VR solutions that attempt to provide the similar functionality.

6.4.1 Compared to other OS:s

The OS GUI developed in this project is not feature-complete, and can therefor not compete with any existing OS. However, users reported that they experienced a deeper immersion in VR compared to other OSs, which could indicate that a VR OS could be optimal for users were immersion is important, such as gamer consumers.

Microsoft's OS called Mixed Reality Portal, which is currently under development (Hayden, 2017) shares some similarities with the developed OS. The most striking similarity is that Microsoft envision a house where the user interact with applications in the world around them, however most of these applications seem to be 2D screens, similar to Bigscreen. The major difference at this point is that the interactions in Microsoft's solution is gaze based in comparison to hand interactions in the implemented solution. On the other hand this may change as the Microsoft Hololens is assumed to use hand interactions (Hayden, 2017).

A better comparison might be to look at other 3rd party software available for the desktop VR platform.

6.4.2 Compared to other VR solutions

Two other examples of popular VR UI that allows users to interact with OS applications are SteamVR and Oculus Home. Both Oculus and Steam allow the user to watch movies, look at photos etc. by starting separate fully immersive applications, however they do not offer any of the basic features that you would imagine would exist in an OS, but rather allow the user to access the Windows OS features. When it comes to interaction, as explained in section 2.2, these applications offer basic interaction by extending a laser pointer from your hand. In comparison, the implementation of the OS developed in this project integrates applications that the users can interact with using gestures such as grab and poke, which were designed with VR in mind. However, the concept application developed in this project still lacks several features such as customizing the environment and features such as creating text documents, which is possible using applications such as BigScreen or VirtualDesktop.

6.5 Other game engines

There are a number of different game engines that could have been used in the project to develop a VR UI. Two alternative game engines that could have been chosen were Unity (Unity Technologies, 2017) and CryEngine (Crytek, 2017). For this project, Unreal Engine 4 (UE4) was chosen for the development of the UI. The reason why the engine was chosen was influenced by a number of different factors. Firstly, the supervisor for this bachelor project has previous experience in Unreal Engine making it easier if any questions about the engine would appear. Secondly, the official support of blueprints in Unreal Engine 4 was a major factor why it was chosen. The reasoning behind this was because the group members didn't have much experience in either C++ or C#. Therefore, the native implementation of blueprints in Unreal Engine 4 was preferred.

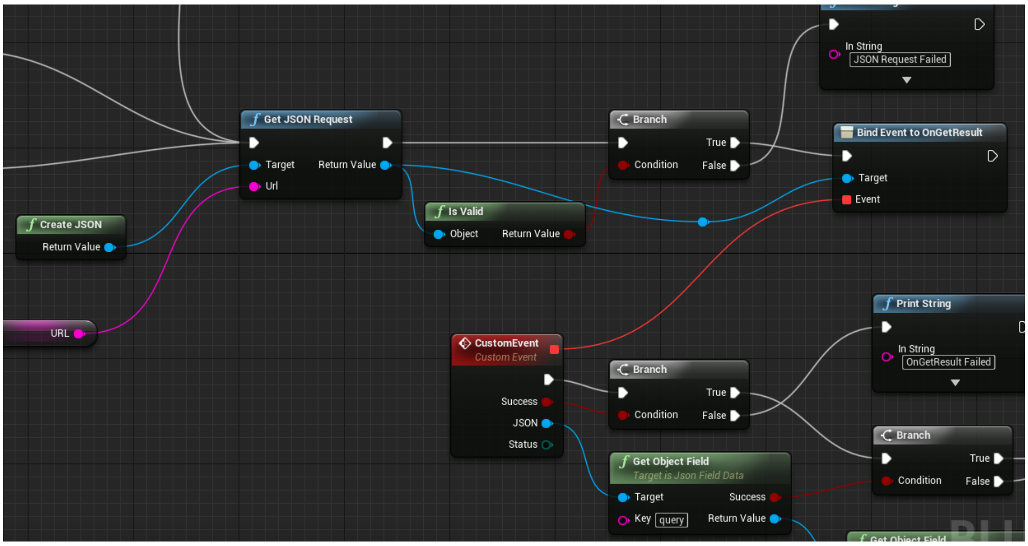

Figure 6.1: *An example of what a blueprint can look like. Allowing to control the flow without having to write code.*

7

Conclusions

The project resulted in a concept UI which was tested and evaluated. The program successfully allowed the user to view and move files, but did not include more advanced file management features. Several interactions to start and manage applications were tested. The action of closing applications by throwing them away was considered intuitive and useful. Overall the idea of a fully VR optimized OS was well received by the testers. Tests for interacting with legacy 2D applications by using laser pointers and a keyboard were developed, but were not well received. This can be partially explained by the unresponsive behaviour of the built in web browser-widget used in the implementation, and the low resolution of the VR headset. More research for alternative interactions with 2D applications from within a VR environment is encouraged. The ability to teleport with snappoints was well received as a means of moving around in a VR environment without experiencing motion sickness. Launching applications from a hand menu was considered intuitive and was easily adopted by the testers, and launching applications by grabbing 3D icons worked well, however the implementation of the 3D icons caused several situations where the testers had to re-attempt grabbing the icon.

Bibliography

Amazon. (2017). Retrieved May 11, 2017, from https://www.amazon.com/HTC-VIVE-Virtual-Reality-System-Pc/dp/B00VF5NT4I?th=1

Apple Inc. (2017a). Retrieved May 8, 2017, from https://www.apple.com/accessibility/mac/

Apple Inc. (2017b). Retrieved April 29, 2017, from https://itunes.apple.com/se/genre/ios/id36?mt=8

Applebee, S. & Deruette, A. (2017). Retrieved May 24, 2017, from https://www.smashingmagazine.com/2017/02/getting-started-with-vr-interface-design/

Benson, A. J. (2002). *Motion sickness.*

Bigscreen. (2017). Retrieved May 12, 2017, from https://www.youtube.com/watch?v=_m8xUrj0ZBE

Bigscreen Inc. (2017). Retrieved February 8, 2017, from http://bigscreenvr.com/

Bill Venners. (2017). Retrieved May 23, 2017, from http://www.artima.com/objectsandjava/webuscript/PolymorphismInterfaces1.html

Cohen, D. & Crabtree, B. (2017). Retrieved May 12, 2017, from http://www.qualres.org/HomeSemi-3629.html

Cooper, A., Reimann, R., & Cronin, D. (2007). *About face 3: The essentials of interaction design.* John Wiley & Sons.

Crytek. (2017). Retrieved April 28, 2017, from https://www.cryengine.com/

Dhotre, I. A. (2008). *Operating systems* (6th ed.). Technical Publications.

Durbin, J. (2016). Retrieved February 8, 2017, from http://uploadvr.com/superdata-heaset-sales-analysis/

Epic Games. (2017). Retrieved May 11, 2017, from https://docs.unrealengine.com/latest/INT/Platforms/VR/ContentSetup/index.html

Flaticon. (2017). Retrieved May 25, 2017, from http://www.flaticon.com/

Fredelius, A. (2016). Retrieved April 20, 2017, from http://www.nyteknik.se/digitalisering/arets-julklapp-ett-lyft-for-svenska-foretaget-6806612

Hägg, G. M., Ericsson, M., & Odenrick, P. (2011). *Arbete och teknik på människans villkor.* Prevent.

Handman, W. (2017). Retrieved May 11, 2017, from http://www.hammerandtusk.com/blog/article/why-you-should-quit-your-job-to-design-for-vr

Hayden, S. (2017). Retrieved May 9, 2017, from http://www.roadtovr.com/microsoft-shows-windows-vr-gives-acer-vr-headset-vision-summit-audience/

HTC Corporation. (2017). Retrieved May 6, 2017, from https://www.vive.com/us/accessory/controller/

Ideas, I. (2012). *Interviewing: Simple ideas for selecting the best* (First). Infinite Ideas Limited.

Jerald, J. (2015). *The vr book: Human-centered design for virtual reality* (First). S.N.

Jordan, P. W. (1998). *An introduction to usability* (First). CRC Press.

Lily Prasuethsut. (2017). Retrieved May 11, 2017, from https://www.wareable.com/vr/htc-vive-vr-headset-release-date-price-specs-7929

Markoff, J. (2009). Retrieved May 11, 2017, from http://www.nytimes.com/2009/02/17/science/17map.html

Merel, T. (2017). Retrieved May 2, 2017, from https://techcrunch.com/2017/01/11/the-reality-of-vrar-growth/

Microsoft Corporation. (2017a). Retrieved May 6, 2017, from https://www.microsoft.com/en-us/accessibility/windows

Microsoft Corporation. (2017b). Retrieved April 29, 2017, from https://www.microsoftstore.com/store/mseea/sv_SE/home

Mike Alger. (2017). Retrieved May 2, 2017, from https://www.youtube.com/watch?v=n3b8hZ5NV2E

Niantic Inc. (2016). Retrieved April 18, 2017, from http://www.pokemongo.com/

Nielsen, J. (1993). *Usability engineering* (First Edition). Academic Press Limited.

Nilsson, Å. W., Ericson, Å., & Törlind, P. (2015). *Design: Process och metod* (1:1). Studentlitteratur.

Norman, D. A. (1990). *The design of everyday things* (First Doubleday edition). Bantam Doubleday Dell Publishing Group.

Oculus VR. (2017a). Retrieved May 12, 2017, from https://product-guides.oculus.com/en-us/documentation/rift/latest/concepts/rug-2-hm-home/

Oculus VR. (2017b). Retrieved May 6, 2017, from https://www.oculus.com/rift/

Österlin, K. (2016). *Design i fokus* (4:1). Liber AB.

Perry, R. C. & Bacon, D. (2006). *Commercializing great products with design for six sigma.* Pearson Education.

PyroDev. (2017). Retrieved May 25, 2017, from https://www.youtube.com/channel/UCC8f6SxKJElVvaRb7nF4Axg

Salvendy, G. (2012). *Handbook of human factors and ergonomics* (4th edition). Wiley.

Schacht, R. & Mulder, M. B. (2012). *Human behavioural ecology.* Wiley-Blackwell.

Scrum. (2017). Retrieved February 8, 2017, from https://www.scrum.org/resources/what-is-scrum

Simplicity Compass. (2016). Retrieved May 12, 2017, from http://youtu.be/kWBhP0EQ1lA

Soundbible. (2017). Retrieved May 25, 2017, from http://soundbible.com/

Spotify Ltd. (2017). Retrieved May 25, 2017, from https://spotify.com

Unity Technologies. (2017). Retrieved April 28, 2017, from https://unity3d.com/

Valve Corporation. (2017). Retrieved May 12, 2017, from http://store.steampowered.com/steamvr

Virtual Desktop. (2017). Retrieved February 12, 2017, from http://store.steampowered.com/app/382110/Virtual_Desktop/

Wijnker, S. (2014). Retrieved March 27, 2017, from https://wiki.unrealengine.com/JSON_Query

Yahoo Inc. (2017). Retrieved May 25, 2017, from https://developer.yahoo.com/weather/

Appendices

A

Function analysis

Example	Purpose	Function	MF/SF/EF	Importance
Mouse pointer / VR hands	**Enable interaction**	**Enable interaction**	**MF**	5
Leftclick on icon		Start program	SF	4
Leftclick on cross		Close program window	SF	4
File explorer		**Enable navigation through folders**	**MF**	5
Leftclick on file/folder		Open/see content	SF	5
Mark multiple files/folder		Enable interaction with multiple object simultainiously	EF	3
Rightclick on file/folder		Change name	SF	4
		Delete	SF	4
		See properties	EF	2
		Open in new window		
		Copy	SF	3
Drag and drop		Move	SF	4
Rightclick in empty space		Create new folder	SF	4
		Sort	EF	1
File icons		**Graphical representation**	**MF**	5
Picture icons for pics and movies		Preview	EF	2
Start menu		Overview of apps/programs	SF	4
		Access to system tools	EF	3
		Access to power off	SF	5
Desktop		Act as startingpoint/ / home	SF	3
Shortcuts on desktop		Gather chosen programs	EF	2.5
Dekstop background		Customization of environment	EF	3
Taskbar		Show active programs	SF	3
		Enable fast start of programs	EF	3
		Show status info	EF	2
Taskbar tools		Gathering point for important tools	EF	3
		Start "fully immersive" VR programs	**MF**	5
		Return to OS when VR app closes	**MF**	5
		Start 2D-apps	SF	3
		Start VR-widgets	SF	3
		Change size of window	EF	2
		Move window	EF	2
		Statically place window	EF	2

B

Interview questions

How often do you use a computer?

What do you use your computer for? (why and how)

Which operating system do you use?

What operating systems do you have experience with? (Including mobile, tablet, etc.)

What do you think are the biggest differences?

Which do you think is the best of the various operating systems?

Which do you think is the worst?

When would you prefer to use one OS rather than the another?

How do you usually structure your files and programs? How does your desktop look?

Is there any difference in how you organize your computer at home vs. at your work?

How much do you personally customize the appearance (colors, themes, wallpapers, etc.) of your computer?

What do you want to be able customize to make your computer more personal?

Have you ever tested VR?

What did you try?

What do you think VR can be useful for?

C

Interviewee information

Interviewee	Age	Sex	Previous VR experience
1	26	Male	None
2	26	Female	None
3	56	Male	None
4	56	Female	None
5	19	Male	Yes, limited
6	50	Female	None
7	49	Male	None

D

Test protocol

Now I want to ask you to put on the glasses. There you will find yourself in an VR environment. Tell us when we have set the right focus. Are you ready to start?

Remember, you really should speak out aloud what you are thinking and what you are seeing.

Ask the test person to look around.

Ask the test person to test different hand movements.

Ask how it feels. Ask TP to test the teleport function (test 2 & 3).

Ask the test person to start spotify.

If the TP does not know Spotify, describe the icon.

If the TP does not find the menu bar, hint about the area where the menu is located.

If TP fails to start Spotify, hint how it can be done.

If TP happens to open the wrong programs, ask them to turn it off.

Ask if it would feel comfortable working with Spotify in its current position.

If not, ask TP to place it comfortably.

Give TP a scenario: You have a song on your head - Hey Jude - but you do not remember who made the song and you want to listen to just that artist on Spotify, google up it.

Provide hints if necessary if TP gets a problem.

Ask TP to return to the Spotiy window

Ask TP to close down all programs.

Give hints if needed.

Ask TP to start the file manager.

Ask TP to find a specific file.

E

Test interview protocol

Preform after each test part:

How was it?

Was there anything you reacted on?

Was there something you specifically liked?

Was there something you specifically disliked?

How did you experience the environment in this test?

Would you like to change the environment? How?

How did you experience the navigation (stand still / snap / teleport)?

Would you have liked it to be differently?

How did you experience the menu / quick start bar?

Was the menu easy or difficult to understand?

How did you experience with opening / closing programs?

Was the menu easy to reach?

Did you experience that the menu was in your way?

Would you have liked it in any other way?

After all part tests:

How was it to have multiple windows open at the same time?

How did moving around application windows work out?

Would you have liked it differently?

How did you experience the browser?

Would you have liked it differently?

How did you experience the file manager?

Would you have liked it differently?

If you had to choose one of the three environments you tested, which one would you choose and why?

If you had to choose one of the three menu systems you tested, which would you choose and why?

If you had to choose one of the three navigation systems you tested, which one would you choose and why?

Any other thoughts, questions or opinions?

End the test - thank you!

F

Test 1 tester information

Interviewee	Age	Sex	Previous VR experience
1	26	Male	None
2	25	Male	Some, limited
3	23	Male	None
4	24	Female	Yes, moderate
5	37	Male	None
6	25	Male	None
7	55	Male	None
8	29	Male	None

G

Test 1 results

Test 1 - 1 - Big open level, tool belt menu, start programs by drag and drop.

Test 1 - 2 - Big room with window, menu on hand, program start from lever.

Test 1 - 3 - Several smaller rooms, menu bar floating, start through button push.

TP1

Test1-1

Nice to have the menu downstairs.

Liked that you could pull up the programs and start them where they let go of them.

The environment felt empty.

It's better when you can move, especially when you have a lot of programs open at the same time. But at the same time, if you sit down, maybe it's nice to be completely still.

Final Questions

Want to be able minimize application windows.

Missing an alt + tab feature to get the windows behind each other.

Difficult to reach the URL field.

The folder structure that was on the left side was a bit confusing. Tried to click on the top of it to come back.

The tool belt was easy to handle.

The top bar was hard to reach.

Liked to have different boxes you can teleport to, one could have game stuff on one and the job carrier on another.

TP2

Test 1 - 1

Close files did not work, did not work the way TP assumed.

The menu bar was too low but it was easy to understand.

The environment gave no structure, you did not knew where to do all the stuff.

When you opened things around you, you became trapped.

The menu - you think you should point things to open, not grab things, move them and place them somewhere.

How often do you take something, move it and place it somewhere to open it?

Terrible to close programs. You saw the box and realized that you would do something about it, but you did not know what.

Test 1 - 2

The hand menu was terribly sensitive.

Hard to understand how to open things (programs), did not see that it was a lever.

Strange to have the forest environment in front of you but you could not get there.

Teleport felt unnecessary, there was no need to move.

Test 1 - 3

Easy to open apps, not as easy to turn off. Not obvious how to do.

Nice room, well thought out.

Sickness when joystick is used.

The menu was far away, hard to reach.

Final Questions

Messy to have many windows at the same time. Very large boxes.

A advantage that you can turn around when switching between different windows. Hidden things I did not need i put behind my back.

The browser - hard to read, the keyboard worked alright.

File manager - Far away so I could not reach it. The window was not known as I reached the bar that would be used to move the window.

The red room felt best - safer environment because you are more used to such an environment.

Liked a menu below - more convenient accessibility. Easy to open apps in the top bar.

Fun to teleport, as long as there is something to teleport to.

TP3

Test 1 - 1

Hard to read small text. The pointer in the browser was very big, difficult to mark, for example, search fields.

Wished to have more stuff in the environment, such as flowers and other plants.

Could have liked to decide how the environment should look.

Missed a cross to close windows.

If there had been a more adaptable environment, you would have wanted to move around in it.

If you have a lot of stuff going on, it may be crowded if you can not move around.

Would rather have poked at programs to start them.

Would have liked the ability to customize the position on the toolbelt.

Test 1 - 2

Not so intuitive to bring up toolbar with a twist.

It was convenient and good to have full control over where the (menu) is somewhere.

The environment felt a little captivating.

Teleport with snappoint felt a little limiting.

Turn the levers to open, works but it was not that fun.

Test 1 - 3

Difficult to close programs, would like to have a cross or some way to close the program smoothly.

Not intuitive to turn off the same button as you started the program.

The toolbar was a bit in the way when using things.

I would have liked the programs to be in a 90 angle to me when I put them out.

The menu was too black and angular.

Final questions

When working with many windows next to each other, it was like getting inboxed.

Switch the focus between windows - just twisting your head.

When you made a copy in the file manager, you could not get rid of it.

Had wanted a small background behind the file manager.

Teleport was more comfortable than walking.

TP4

Test 1 - 1

Close program should have been a trash bin or something, need to clarify how it works.

The menu was tough, you got a little caught, the menu was in the way.

The environment - when you could not move around it, it became a bit uninteresting.

Navigation - Want to be able to move a little more.

Drop program to close - works ok, but also wanted to be able to point on a X to be able to shut down, drag and drop felt unnecessary.

Test 1 - 2

The menu was very sensitive.

Teleporting - works well, felt intuitive.

Strange to turn the levers.

The menu - good to be able to make it appear when u want it but not by twisting your hand.

Test 1 - 3

Turn off programs - started looking for a X in the corner to turn off.

Positioning windows was easy.

Joystick - worked well as long as you did not look down.

Navigation - had wanted to move objects instead of moving the player.

The menu - had wanted to get it with a button. Could be in the way.

Keyboard - Want to push your fingers instead of the laser pointers.

Final questions

Multiple windows open at the same time - can not have as many open as the computer lacks Alt + Tab.

Feel good to switch between programs if placed well.

The browser - difficult to use, sometimes you used your hands and sometimes the beams and then use the button on the controller.

File Manager - When you are used to pointing, it worked well. Hard to move the window.

The environment - likes level 2 but wants the joystick feature.

Menus - liked the belt but it was placed badly, liked the location of the top toolbar.

Navigation - like teleport + joystick I'd rather move objects around me. Walls good when teleported around to see where you are.

TP5

Test 1 - 1

The menu - felt very close to the body.

Close programs were a bit difficult - just thought one would press the bar.

Very easy to make copies by mistake in the file manager and then you had to shut down all copies individually.

Expected that the menu would accompany you when you turned around, it might be good to have it in a place so it does not come along and disturb.

Navigation - felt ok, did not feel like you had to move.

The menu - easy to use when you learned how, closing programs was hard to understand, just want to be able to press and close the active window, good distance on the menu, wasn't in the way.

Test 1 - 3

Teleporting - Disorienting.

Joystick was easier than teleport.

The menu - was very close and covered over the file manager, closing down was a little hard to understand.

Shut down - Would have been good if shutdown differs from start.

The menu - would have liked it a bit further away.

The final questions

Working with multiple windows - felt very easy, especially in level 1 when nothing bothered in the form of strange hand movements.

File manager - Very easy to make copies by mistake, otherwise easy-to-use, natural to have the breadcrumbs on the left.

Environments - liked level 1 the most (felt calm) or level 3 (homey) and could leave the house.

The menu - like the toolbelt most, did not feel the way. The hand menu was uncomfortable and weird to get out.

Launch programs - felt most natural to grab programs and let them go.

TP6

Test 1 - 1

Lots of stuff opened that was not on purpose, small and confusing (file manager).

Good things - when I realized that programs were closed by dropping in a special place, it felt smooth.

Can easily open up massive amounrs of folders in file manager, then had to drag each folder one at a time to the closet, extremely ineffective.

The environment felt like Svalbard, no trees, some water would have been nice, some plants.

Felt a bit uncomfortable that you could only stand still, I would have liked to be able to move, even if it was office work.

The menu - static menu bar located down where it was can be very annoying when opening a window - is in the way because the window opens directly on the menu bar (tp did not understand that he could drag the programs from the menu and let them go wherever he wanted).

Open the programs by grabbing was fun, the balls looked grippable but would have liked to be able to poke them.

Test 1 - 2

Turn to open - felt 19th century, not what one expects, expect more touch and point.

The menu - liked that you get it by hand twist.

Bad - Turn to start program.

The environment - too cold, one could see out the window but you could not access it, felt imprisoned, want to go out.

Navigation - Teleporting felt nice.

The menu - like to open it with your hand through twist, but do not like to open programs by twist, want to point, the menu was never in the way.

Test 1 - 3

Joystick crawl - got sick.

The menu bar had an unnatural location.

Do not understand the relation between open apps and documents, want consistent interaction.

Nice environment, lacking a plant.

Navigation - Cool to teleport.

The menu bar - bad location, want it in front of me, felt like it was outside the field of vision.

Interaction - The folder system felt easy, the bar difficult to understand how to interact with, however, understood the logic, missed X to close programs.

Final questions

Nice to work with multiple windows at the same time, just moving them wherever you want.

Environments - like the red room most, most characteristic.

Menus - liked the way the hand menu was hidden, but liked to have a combination of the ball system or pop-up menu where you could point to things.

Navigation - like joystick, felt weird to teleport and to stand still.

H

Use cases

Business Case

Kim is 38 years old and works as a coordinator for the boat traffic in Gothenburg's harbour. Kim's work is to guide and coordinatie ships that is coming to dock in the harbour of Gothenburg. Kim's work is entirely digital, the software used is specifically designed for the shipping industry. The program used is SeaFares and allows Kim to see a large digital interactive map with vacant and busy harbour spots as well as the ship's GPS positions. The program also includes a communication engine that allows Kim to communicate with the various ships through voice calls.

A typical work scenario for Kim looks like this: Kim is sitting in the office room in front of the computer with a headset on. Three screens are connected to the computer. On one of the screens, via SeaFares, Kim opens a list of all the incoming and outgoing ships of the day and the corresponding estimated times. A large ship is at the entrance and is estimated to reach the dock in ten minutes. Kim opens his digital dock chart on his computer screen and looks at the two places big enough for the ship to fit, both places are occupied even though one of them should be available. Kim opens the incoming ship contact information on the third computer screen and calls the ship to ask it to wait in the que. Kim changes the status of the ship on the map to "In Queue". Immediately after, Kim calls up the ship currently in the dock and is informed that the ship's departure has been delayed due to technical difficulties. Kim changes the ship's status on its digital map to "Delayed" and informs the ship in a docking queue about this. After 20 minutes, the delayed ship may depart, Kim is informed by a call in the program. Kim informs the queuing ship about the vacant dock and updates the status of the ships and the dock on the map.

Kim experiences the work as quite demanding, many windows are open simultaneously on the three screens and it is important that Kim has good control over everything. Kim often feels sleepy after a day of work when most of the day is spent sitting in front of the computer.

Gamer case

Alex is 22 years old, student and has computer games and computer technology as two major interests. Alex likes to use the computer for games, talking with friends, buying things etc. Alex has an Oculus Rift with Oculus Touch controls.

A typical user scenario for Alex: The computer is started immediately when Alex comes home from school. The program Alex starts first is Discord, where Alex can voice and text chat with friends. Alex likes to always have Discord running in the background so friends can always send messages or start voice chat. Alex plugs into Oculus Rift and starts the latest game, Robot Recall. While Alex he gets an notification from Discord, Alex has to take the Oculus-HMD off and open the Discord window on his screen to see the Discord chat. Alex gets hungry and turns off the game and puts away the Oculus-HMD. Alex decides to order pizza pizza from Online Pizza. Alex opens a tab in the browser and orders, online pizza has live tracking of the order so Alex keeps the tab open. While the pizza is baked and delivered, Alex goes to Twitch to watch his favorite streamers. Every now and then Alex switches to check out how the pizza works. The pizza is delivered, Alex eats it and then turns off his computer and goes to sleep.

Alex likes to do many things at the same time and wishes it could happen more smoothly. It's especially problematic when Alex has to take off the Oculus HMD to be able to check Discord when a game is running.

Social network case

Sam is 30 years old and works everyday in a store in the center. When Sam does not work, Sam likes to follow and stay up to date on various trend blogs and be active in social media.

Yesterday, Sam came home from a United States trip and would like to share these experiences. Sam had a very good trip and already did not want to be there anymore, so to remember Sam changed the background image on the computer to a picture from the trip. To do that, Sam opens the pictures and looks through them to find the best one. On the trip, Sam had a camera that could take pictures in 360 degree view, so of course, Sam wants to upload these pictures to get extra likes. For that, Sam has to plug in the memory card, locate them in the file manager and then log in to facebook and upload the pictures. Sam gets a little disappointed that it does not look as cool as it actually did to be there, but post them anyway.

While Sam is waiting for someone to pay attention to the pictures, Sam checks the weather next week by opening its weather widget and watching it. It looks like rain, but then it would be great to go to Ikea and buy new furniture. Sam does not really know what would fit into the home and would like to see the furniture in his room before Sam purchases it.

Sam notices a notification in the facebook tab and clicks in to view who has commented on the image and what the person has written. Sam writes back and then goes to the feed to check everything that has happened while Sam was abroad. Sometimes it seems a bit messy to have so many tabs open at the same time.

I

Interfaces

Interface	**Usage**
GrabInterface	Provide events for the actions Grab, Drop and Throw.
IInteractable	Provide events for the actions Poke/Point.
Touchable	Provide events for the Touch action when the hand overlaps the object.
IHeadCollision	Provide events when the object collides with the head of the user.
IDropArea	Provide events when another object is dropped ontop of it.
IHint	Provide events when the hand is near. Returns a string for the hint text.
IDisplayable	Displaying image files.
Program	Provides startup arguments and is called when the object is created.
IKillable	Used to shut down applications and objects, clean up memory and attached objects.
IResizeable	Provide implementing object with events when the resizer is overlapping.
LerpListener	Provides callbacks from the LerpHelper when the object has finished transforming.
ButtonListener	Provides callbacks from the ContextButton when the button is pressed.

J

Test 2 procedure

Test procedure test 2

Write down name, age, occupation and previous VR experience.

Explain the purpose - develop a UI for an OS, we want to test if it works.

Show the HMD and controllers, focus on showing all the buttons.

Introduction through OS tutorial.

Familiarize the user with the environment and play around with the programs.

Test functions of the hand menu.

Start programs.

See running programs.

Shut down program programs.

Test filebrowser.

Find a specific folder, VROS-folder on the desktop Open in new window.

Make a copy/move vacation pictures from “Mixed files” to “Pictures”.

Find out the weather through the widget.

Test Facewall.

Test Andreas’ game.

Order pizza.

Test pong.

Test boat parking widget.

Explain basics.

Ask user to park all boats.

Use phone if problems arise.

K

Test 2 tester information

Interviewee	Age	Sex	Previous VR experience
1	23	Male	Some, limited
2	55	Female	None
3	55	Male	Some, limited
4	26	Male	Some, limited
5	22	Male	None
6	24	Male	None
7	22	Male	None

L

Test 2 results

Emotions

A little unpleasant not knowing where you are irl.

Cozy with background sounds.

Environment

Hard to see small text

People think the environment is ok, some would like to be able to explore the outside.

One person would have liked a multiplayer feature.

People sometimes felt that the chair was in way for them when they rotated.

One person liked to personalize the surroundings by hanging up pictures.

Menu

The menu was smooth and understandable.

Some were trying to poke to launch the programs, but quickly figured out how to do it.

Problems around the hologram, sensitivity - precision, wrong programs started / difficult to grab the right program.

Browser

Hard to see, low resolution.

Generally difficult to mark / interact with.

File system

People had some trouble understanding / navigating with the breadcrumbs.

People had difficulty using the Alt button to copy files.

Keyboard

Hard to see what the keyboard was connected to.

People had difficulty checking if they were writing correctly, ie you did not know if you clicked the wrong key, and had to turn from the keyboard and pause the writing to see it.

Apps

People appreciated, thought it was fun with games, and it was easy to understand.

Everyone wants to put the boats on the map instead of in the boxes.

People thought facewall was easy to use.

People quickly understood how to start VR apps.

People did not understand directly how to shut down VR apps.

Interaction

Several people had difficulty finding and using the Alt button, and when it could do things.

Mixed response to closing programs by throwing, some thought it was quick and intuitive, some thought it was awkward.

Hard to remember which button was the menu button, which one was Alt.

Two people wanted to enlarge programs.

Resize stick did not work.

Teleport

People teleported by mistake, and people were struggling with the rotation.

People noted during the interviews that they in the end understood how teleport worked, including rotation and thought it worked easily then.

Mental model

People grab the center of windows to move, and then by mistake take things from programs that start things.