Interaction with Cuboids in Virtual Reality

- a Kitchen Design Scenario

Nicolai Krogh Jørgensen
Christian Aagaard Larsen

VGIS, 10^{th} semester
VGIS Group 1004
Aalborg University

Supervisor:
Claus B. Madsen

June 8^{th} 2017

AALBORG UNIVERSITY
STUDENT REPORT

VGIS 10
Det Teknisk-Naturvidenskabelige Fakultet
School of Information and Communication Technology
http://sict.aau.dk

Title:
Interaction with Cuboids in Virtual Reality - a Kitchen Design Scenario

Theme:
Interactive Systems

Subthemes:
Virtual Reality, HTC Vive, Head Mounted Display, Interaction

Project period and hand-in:
Wednesday, February 8^{th} 2017 - June 8^{th} 2017

Project group:
VGIS Group 1004

Participants:
Nicolai Krogh Jørgensen
Christian Aagaard Larsen

Supervisor:
Claus B. Madsen

Number of pages: 69

Number of appendices:
5 with a total of 11 pages

Synopsis:

Head-Mounted Displays (HMD) and motion tracked controllers have been rising steadily in popularity in recent years, with more and more companies trying to find ways of utilising the technology for improving processes. Many applications have been created for utilising the technology for visualisation of 3D environments, however using the technology in a creative capacity is still in its infancy, mostly due to tradition methods such as mouse and keyboard typically being faster and more feature rich. Proper techniques for content modification and creation need to be discovered in order for this technology to move beyond being primarily a visualisation tool. The aim of this project is to explore various interaction methods with the aim of being able to move and place objects in Virtual Reality in a comfortable and fast way. To accomplish this, several interaction methods were designed and implemented, primarily aimed at aiding the user in the task of placing and aligning objects. Once the methods were implemented, user tests were conducted where participants had to copy pre-placed arrangements of objects as best they could, in order to discover strengths and weaknesses of the various methods. The experiment revealed that some methods lent themselves better to scenarios where a user would have more time to learn the system, whereas other methods were better for situations where users had to be able to pick up the system and use it.

PREFACE

This masters project was written by group VGIS1004, consisting of two 10^th semester Vision, Graphics, and Interactive Systems (VGIS) students at Aalborg University

The work in this report is a continuation of the authors 9^th project, which revolved around product assembly and menu design in VR, in collaboration with a company called PDM Technology. This report stands on the shoulders of this previous project, and instead focuses on interaction systems in HMD-based VR. As in the previous project the current project is developed with support from the company. This report covers the design, implementation, and testing of different interaction methods.

This report, along with the additional material found together with this upload covers the knowledge needed in order to reproduce the results of the project. The report references to the portfolio when more information on the topic is available. It is structured in chapters and sections. Chapters always start on a right-hand side (odd-numbered) page, and sources are cited using the bracket citation style. If a figure is referenced in the text, it happens before the figure is shown or on the same page as the figure is shown. It may sometimes appear on a later page. To understand the report easier, make sure to find the referenced figure before moving on to the next paragraph.

The project files consisting of: scripts, libraries, and scenes,- exist as a Unity project which was placed in the digital exam system at AAU. The data acquired through the experiment were also made available as raw data presented in Excel sheets. This report is also accessible through the exam system, and through the AAU student project database.

ABSTRACT

This report presents the process of creating and evaluating different interaction methods for virtual reality applications using Head-Mounted Display (HMD) technology. Virtual Reality (VR) has, in recent years, moved from being primarily available to academics and research to be something that is available to consumers at a reasonable price. Virtual reality with HMDs has many use areas, and as the sales continue to rise and more and more consumers, and companies, get their hands on a VR-system, the requirements for these systems will steadily increased, also increasing the need for more, and better, interaction possibilities. Currently, virtual reality with HMDs is primarily used for visualisation and is missing something that computers have had almost since their conception: a common way to interact with it, i.e. mouse and keyboard for computers. The aim of this project was to explore various interaction methods with the purpose of being able to precisely move, place, and align objects with one another in virtual reality in a fast and reliable manner. We identified strengths and weaknesses of the interaction methods through user experiments gathering both quantitative and qualitative feedback from the users. The results of this experiment showed that some interaction methods are faster and easier to use than others, and that the learning rate differs between them. We also attempt to show correlation between variables, such as the age of participants and how fast they were. These results showed that some methods lend themselves better for certain situations; some being better if the users have more time to to get accustomed to them, whilst other are a better choice should the system require the user to be able to pick it up and use it without much training. We also showed that there did indeed seem to be some correlation between age, hours spent on computers, playing games, and participants completion times. However no causation was proven. Important to note is that the project was only able to cover a portion of possible interaction methods in virtual reality and a lot more could be explored in the future.

CONTENTS

CHAPTER 1

INTRODUCTION

Virtual Reality (VR) technology has, to date, largely been limited to academics and research, however, with new consumer oriented systems becoming more readily available this is expected to change in the near future. Consumer priced systems are flourishing with many products emerging from various companies, and with the increase in computing power over the last couple of decades VR is no longer limited to in-house research and academics. Some industries have already successfully been using VR for many years in areas such as architecture, flight simulation, military training, product design etc. for e.g. digital mock-ups which provide a reduction in development time and cost of the design process. VR provides new possibilities for interacting with digital media in a way that provides seemingly limitless direct access engaging more of the human senses than more traditional interaction techniques. This makes VR good for providing a better understanding of the task at hand due to a more active participation in the task; be it viewing models or training a specific task, such as assembling a product, thus providing a more engaging overall experience [8]. The consumer oriented VR systems which are gaining the most recognition in the VR community is the Oculus Rift and the HTC Vive head-mounted displays (HMD), especially when looking at interaction in VR, as both these systems are supported by tracked wands technology (Oculus Touch and Vive Controllers). In October 2016 it was revealed that 140.000 HTC Vive headsets had been sold since it launched in April the same year [11]. Comparatively Apple sold approximately 3.3 million iPhones during its debut year in 2007 (six months) [5] while the PlayStation sells roughly 1 million consoles a month. By the end of 2016, research done by Canalys [6] suggests that 500.000 HTC Vive were sold and 400.000 Oculus Rift headsets along with 800.000 PlayStation VR. If this trend continues the requirements of what is possible with VR systems will increase and as such a need for more, and better, interaction possibilities will likely arise.

This project revolves around interaction in HMD-based VR utilising wand-based technology, also called Motion Tracked Controllers (MTC). Specifically, we are using the HTC Vive VR headset mainly due to the availability of the headset but also, in part, due the increased mobility it provides with room scale tracking (an

image of the headset is displayed in Figure 1.1). Virtual Reality is most popularly

Figure 1.1. An image of the HTC Vive with the vive controllers and the two lightowers making roomscale possible.

used by the general population to describe imaginary worlds that exist inside computer games or in movies. Immersive virtual reality includes technologies that give users the psychophysical experience of being surrounded by a computer-generated environment, where immersion is the sensation of being present giving the user the sense of reality - of "being there" - in the computer-generated environment [24]. In this project Virtual Reality is used to describe an interactive artificial environment generated by the computer providing the same visual stimuli as the real world; and we generally only consider VR with interaction through HMD-based technology.

Reading through VR literature and looking at some of the existing software solutions utilising HMD based technology it is apparent that the technology is primarily used for visualisations. Interaction is generally limited to being able to pick up objects in proximity and moving around in the environment. Using VR to design and build Virtual Environments (VE) is still something that is sparsely seen, most like due to the difficulty of interacting with objects of different sizes at different distances seamlessly [27]. Another issue is providing good application control inside a VE using the limited buttons available for interaction on MTCs [13]. Creating quality interactions that works for multiple purposes and with multiple types and sizes of objects is currently one of the biggest challenges for VR [8]. Furthermore, a thing that is missing from VR interaction is a common acceptance of a single type of interaction. Using a computer, more or less everyone accepts the mouse and keyboard interaction and how it works, even when something might not work exactly the way one wants; that is simply "how it is". This is something VR is still missing and something we try to work towards with projects like this.

1.1 Project Delimitation

FiXme Note: Add example of technique or pattern

In this project we present different interaction patterns and a variety of techniques within each pattern. To clarify, an interaction pattern is a generalised high-level interaction concept that can be utilised in different ways. An interaction technique

is more specific and usually technology dependent and these techniques can usually be grouped under a common pattern that they share. Another way to think of it could be a computer program with inheritance, where the parent class has some common functionality shared by all the children classes; this is not entirely the same, but close to it. Trying to solve interaction, specifically manipulation, for every shape of object would require a larger time-frame than what was available for this project. As such, this project was focused on cuboid shapes; and as user testing was something that was required to verify the interaction quality in the project, we decided to work with a kitchen design scenario. The problem with most non-cuboid shapes, especially non-symmetric shapes, is that it is harder to make an automatic model of how they align next to one another. The placement should be more or less the same no matter the shape of the object, so for placement it is mostly the size that matters, but alignment is a more difficult task. To align non-symmetric and abstract shapes to one another one would likely have to manually define how they align by e.g. setting points or faces on each object to be eligible for alignment.

The purpose of working with a kitchen design scenario was to have something users would be able to relate to, and most kitchen objects (cupboards, drawers, stove, etc.) have cuboid shapes and are placed and aligned precisely next to one another. Working with a kitchen design scenario also meant that we could restrict rotation to cardinal directions (i.e. 90-degrees increments around the Y-axis - up/down), and that objects had a distinct forward direction (front face of objects). Additionally, kitchen elements typically only sit on the floor or the wall, providing a useful limitation on how items can be placed, which simplified the problem area even further. By applying these limitations, more time was saved and a foundation for precise manipulation of objects was made, opening up for more complex shapes in the future.

1.2 Problem Statement

The problem statement this project attempts to answer is:

> What strengths and weaknesses do different interaction patterns/techniques have for cuboid placement and alignment in VR when rotation is restricted to cardinal directions?

This problem statement is twofold in the sense that, first, the best pattern for cuboid placement and alignment has to be identified, in order to further narrow down the scope to investigate, and then techniques within that pattern has to be evaluated to identify the strengths and weaknesses; and preferences.

1.3 Report Structure

The report is focused on the exploration of interaction patterns and interaction techniques for VR. In Chapter 2 the work related to this project is described with examples of other applications using interaction in VR as well as ample examples

of interaction patterns and techniques that have been developed in the past. In Chapter 3 the design of the overall system system is described as well as the patterns that were explored in this project. In Chapter 4 a preliminary test is discussed in order to figure out which pattern is preferred by users to then further develop techniques for that pattern. Crefchap:desimpb covers the design and development of techniques as well as the thoughts behind them and Chapter 7 presents and discusses the results of an experiment conducted to identify the strength and weaknesses of the developed techniques. Finally, Chapter 8 we conclude on the findings from the experiments and the project as a whole.

CHAPTER 2

RELATED WORK

This chapter discusses existing 2D applications for room configuration, as well as examples of VR used in existing 3D applications, and previous research related to interaction in VR. When mentioning 2D applications and 3D applications in this report we are referring to the dimensionality of the interaction interfaces. A 2D application in this sense is an application where the interaction is done with e.g. traditional mouse and keyboard input on a regular computer monitor, and a 3D application uses interaction in three dimensions using e.g. wand based technology like the Vive and uses a peripheral capable of displaying 3D. The purpose of discussing existing applications for room configuration is to gather inspiration from what users might have tried before and thus be used to experiencing. Furthermore, it gives an idea of what the current technology is capable of. The same argument can be used to researching previously implemented interaction patterns and techniques used in VR, where the focus will be on patterns and techniques that can be used with an HMD. The interaction patterns and techniques will be considered when designing and implementing patterns and techniques later in this report in Chapter 3 and Chapter 5.

2.1 Related Applications

Looking at related applications, two different types are explored. 2D room planners and 3D applications usings HMD-based VR. The 2D room planners explored in this project are free to use and are all browser-based. It is possible, probably likely, that better applications exist if one were to pay the price and get a stand-alone application, however, as previously mentioned, the primary purpose of looking at existing 2D planners is to gather inspiration and to see what some of them do well and in what areas we imagine that VR could improve upon this process and vice versa. The 3D applications are analysed from video and textual information only, meaning we have not tried these applications ourselves. The 3D applications are all applications that currently use HMD-based VR and here the interaction in the applications will be analysed based on the available information.

2.1.1 2D Room Planners

Looking at some of the different 2D room planners, some traits are common, and they generally have a few things they all can do. However, there are also some clear distinguishing factors as some of them have quite a few more features to ease the process of 2D room planning. As there are quite a few 2D room planners, we will primarily focus on three of them while also focusing on what is common features for the different planners. The three planners we will focus on are the Planner 5D [18], the Ikea Home Planner [7], and the Autodesk Homestyler [1]. The choice fell on these three as they provide the most broad functionality as well as Ikea and Autodesk, both being well known companies. Each of these planners provide quite a few features, thus explaining and discussing all of them would be out of scope of this project; as such we focus on the **room layout**, **view controls**, and **object placement** as these are the most relevant features with this project in mind. To show an example of what a planner looks like, the IKEA home planner is displayed in Figure 2.1 with an example of a kitchen setup. Most of the room planners come

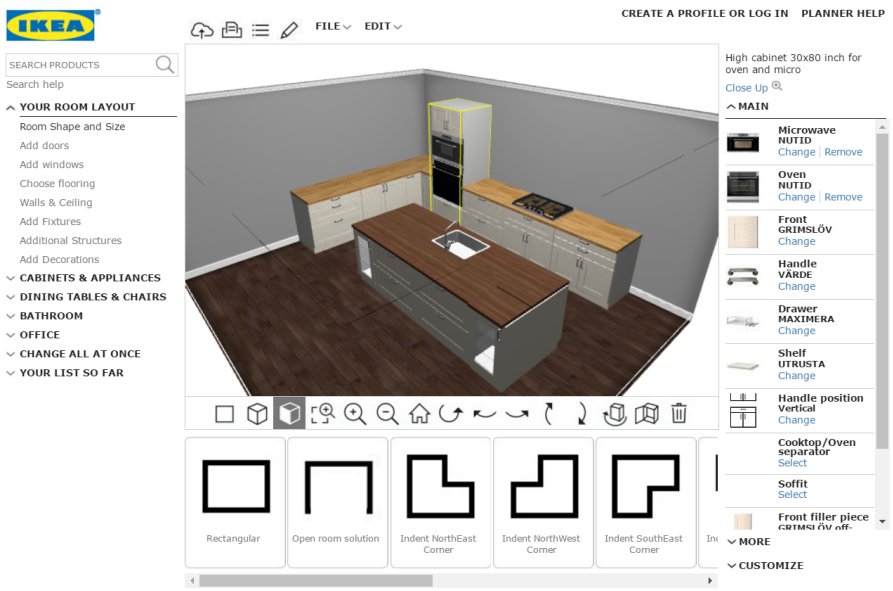


Figure 2.1. A screenshot of the IKEA home planner

with a **room layout configurator**, and the ones supplied in the three mentioned applications allow for an almost complete custom room layout. It is possible to control everything from room size, to where doors and windows should be placed, as well as how the room should be shaped to some extend; some only have presets to select from, such as the IKEA planner, where only the lengths of the walls are changeable. The ceiling height is also adjustable. The rooms are all configured from a top down orthographic projection.

The **view controls** are one of the main differences between the applications, and something that greatly influences the usability of the room planning. Having the ability to freely chose which perspective and which projection to view the room from makes it a lot easier to place items exactly as wanted. Here the Planner 5D and

Autodesk Homestyler both use a top down orthographic projection as well as a 3D perspective projection as their only two options, whereas the IKEA room planner allows for several orthographic projections where it is possible to also view it from all sides in 2D. This is especially useful when placing objects on walls and above other objects, as the top down view provides no useful information in that scenario, and interacting and moving objects precisely in the 3D view can be difficult at times where the 2D view provides more precise control. The view controls are controlled through the applications interfaces which have a similar style to them with a side menu for selecting different types of objects to add to the design. Menus at the top or bottom have settings for things such as controlling the perspective, zoom and other general application settings as well as settings for the currently selected object. Some other functionality included through the interface is the option to take pictures of the room for printing as well as adding a ruler to the planner in order to get a proper measurement of the setup in the planner. There is also a few manual input options in most cases in order to control e.g. the height of an object or if it should be raised from the floor (depending on the application).

Object placement in the different applications also vary slightly. In the Planner 5D the placement of objects is completely freehand and there is no collision checks which means you can move objects through one another as well as through the walls. Both the IKEA and Autodesk planners use collisions and objects snap to both walls and other objects when in close proximity, with the IKEA planner using more aggressive snapping. Visually, they are also quite different, where the Autodesk Homestyler has a discrete snapping contrary to the IKEA planner which uses a very visual indicator of how the object is going to snap when released. All the planners allow for rotation which is also guided by snapping at 45 degree- or 90 degree angles. Though configuration is not a focus of this project, all the planners offer varying degrees of product configuration. The Autodesk Homestyler and the Planner 5D allow for change to the size of objects, whereas the IKEA planner allow for changes such as which handles should be on e.g. a drawer; this is something that should be explored more in the future.

2.1.2 3D Applications

As previously mentioned, examples of complex interaction in VR are still quite limited, but some have made an attempt at using HMD-based VR as a creation tool. The main examples we use here is the MARUI plug-in [15] for Autodesk's 3D modelling tool Maya and the EditorVR for the Unity3D game engine (VRtisan [26] was also an interesting candidate). Other plug-ins and programs exist, but these either have limited interaction capabilities compared to the MARUI plug-in and EditorVR, which allow you to create content while wearing the HMD, or they are not publicly available. Other plug-ins, such as the VR-Plugin Viewer by VR-Plugin [25] (also for Maya), are mainly meant to assist in viewing the content you have already created in VR. When looking at these 3D applications we look at how the **interaction** is handled in the application and the **functionality** provided and in the end they are compared from an ease of use standpoint.

Looking at the MARUI plug-in, displayed in Figure 2.2, it provides an example of how **interaction** can be done in VR. In the figure its shown that the plug-in makes use of the HTC Vive and the trackpads it provides which in this case are used for radial menus. To interact with the menu and the objects in the scene two red arrows are used, sprouting from the top of the controllers; like the standard laser provided in the Steam VR user interface, except shorter. The arrows are able to interact with everything visible in the scene through a projection from the origin of the view to the tip of the arrow selecting whatever is visible at that point in screen space. As we did not have access to the application it is hard to say exactly how the interaction works in terms of button presses, accuracy etc. However, if it follows general convention of the HTC Vive and applications developed for it, then the trigger underneath the controller is used for selecting objects and the two buttons on the side (grip buttons) are used for moving objects. In the example given in the YouTube video the

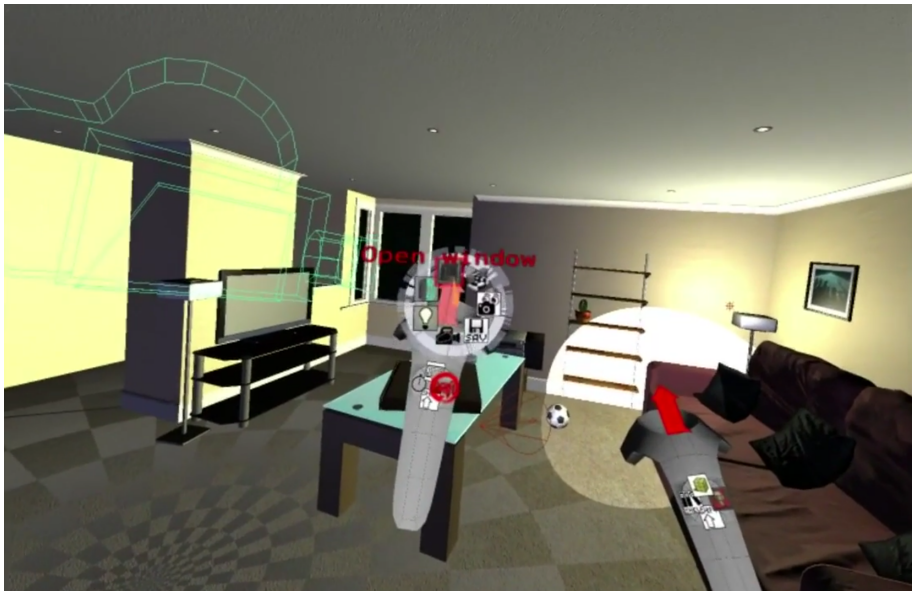


Figure 2.2. A screen shot of the MARUI plug-in for Autodesk Maya from a YouTube clip presenting light setup in the plug-in [14]

user creates a new cone light which is scaled and moved around while also switching between different types of views (shaded, wireframe, etc.). Some of this **functionality** does not exactly advertise the strengths of content creation in VR as most of it could have been done just as easily from the 2D editor. However, the strength is really displayed when the user constrains the camera to the Vive view and records key-frames of the movement done with the headset. This could drastically cut down on the time needed to move a camera around in the 2D editor and set key-frames to get the desired recording of the virtual scene. Another advantage of VR content creation is that you get a better feel of the scale of objects compared to one another and compared to the size of the user/character, and if the content is meant for VR, being able to directly preview and change things from VR could potentially save a lot of time. One of the potential issues with content creation in VR is precision and functionality as one would want to be able to do everything possible in the 2D editor inside VR as well so it is not necessary to take the VR head-set on and off all

the time. Whether or not that has been achieved in the MARUI plug-in is unclear, however, it also unlikely that every functionality of the 2D editor would be fitting or even functional in VR. For instance, in the ending of the video presentation of the plug-in the recorded key-frames are displayed in the 2D editor and not in VR, which is sensible.

Another example of interaction in VR is the EditorVR by Unity Technologies in which they adapt their regular 2D editor of the Unity3D application to VR. An example of the EditorVR is displayed in Figure 2.3 where the user is interacting with a 3D widget containing a World In Miniature (WIM) model of the scene currently being edited. Again the HTC Vive is the HMD of choice here, most likely due to its superior room scale tracking. The **interaction** in this application is handled using two lasers from the top of the controllers as in the Vive standard menu configuration as well as two cones at the base of the laser pointers. If one is familiar with the HTC Vive, the interaction is more or less as expected. The trigger on the controller is used for selecting objects and menu points while the grip buttons on the side of the controller are mainly used to grab objects or 3D widgets and move them around when they in close proximity of the cones on the controller. The lasers are mainly used for menu interaction and are not enabled for object manipulation. The main menu in the EditorVR is a 3D menu placed above the left controller when the Unity Logo on that controller is pressed. The main menu is four-sided and can be rotated using the trackpad on the left-hand controller. This is where all the 3D widgets of the editor is located. In the 2016 Unite demonstration of the EditorVR tool the

Figure 2.3. A screen shot of the EditorVR from a presentation at Unite 2016 [21]

user added some new objects to the existing scene and moves them around a bit through some 1-to-1 direct manipulation as well as through WIM. Again, most of the interaction and **functionality** demonstrated does not portray any particular strength of using EditorVR over the regular editor (other than for VR content creation), until a camera gets added and some footage is recorded. In this scenario, the free 1-to-1 movement provided in VR once again speeds up this process by quite a

margin. Other than that, only the general benefits of using VR are in effect.

Common for both the examples given here is that they are not oriented towards new users; we say this based on the complexity of the interface and the plethora of possibilities. Previous experience has taught us that even something as fundamental as how to hold the controller properly has been an issue for some first time users of the HTC Vive and then adding a few interactions on top of that and users got overwhelmed pretty quickly. Another trait they have in common is the fact that both applications only seem to primarily allow for 1-to-1 manipulation of objects, and in the EditorVR you are mainly able to manipulate objects that are close to you (except using WIM). The MARUI plug-in does allow for moving objects not in proximity through its screen projection selection technique, however the manipulation is still 1-to-1. Neither of the two applications allow for large movements nor any precise alignment of objects next to one another as far as we were able to tell from the material we had access to; except of course through direct value input in the EditorVR through their floating number pad which while usable still feels inconvenient to use. Having tried the EditorVR we are intrigued by the opportunities it presents in the future, but it is also evident that it is not quite ready for common use yet.

2.2 Interaction Patterns and Techniques

When it comes to interaction there is rarely a single right way to handle it; it always depends on the problem one is trying to solve. As such, a lot of interaction patterns and interaction techniques have been developed over the past several years. Here, we take a look at what interaction patterns and techniques are, based on the book [8] by Jason Jerald, and provide some examples of what has been developed in the past.

An interaction pattern is a high-level concept that can be generalised and used across different applications and likely across different platforms as well. An interaction pattern would be able to achieve the same goals independently of the application and platform it is used on. Interaction patterns are also largely implementation independent and described in a simple way which states the interactions that occur between the user and virtual world.
Interaction techniques are more low-level and can be highly technologically dependant and very platform specific. Interaction techniques can usually be grouped under a common interaction pattern if they have similarities. Examples of interaction techniques grouped under the same pattern are given in Chapter 5. The reason for this distinguishing is to make it easier to communicate interaction concepts as a lot of different interaction techniques exist, each with their own name, so, by grouping the similar techniques under a unifying interaction pattern it makes it easier to organise and remember.

Jason Jerald organises the patterns into categories based on what function they are used for which results in the following categories: selection patterns, manipulation

patterns, viewpoint control patterns, indirect control patterns, and compound patterns. In this project we are mainly concerned with selection and manipulation of objects and as such we do not cover viewpoint control patterns nor indirect control patterns; we also only concern ourselves with patterns revolving around user input through hands or a tracked controller, and not voice or other modalities. Furthermore, all the interaction patterns and their techniques described in the book are not included in this section.

2.2.1 Selection Patterns and techniques

Selection is the process of specifying which object(s) one wants to apply an instruction to, be it changing an objects colour or deleting the object etc. In traditional mouse and keyboard input this corresponds to clicking a folder which marks the folder as selected. In VR this process becomes slightly more complicated as a third dimension is added (depth/z contrary to x,y input of a mouse) resulting in the fact that all objects are not guaranteed to be at the same distance. In the following sections, some selection patterns and corresponding techniques are described and discussed.

Hand Selection Pattern

This pattern requires direct contact with the object being selected akin to how selecting and picking up items works in the real world - assuming one were to use ones hand to make physical contact which is rarely the pattern that occurs most. Once contact has been made an action triggers the selection such as a press of a button or a gesture. As this pattern requires direct contact, it limits how far an object can be selected based on how far one can reach which means that objects outside of reachable space are not selectable without first relocating. Another potential limitation presents itself when selecting small objects as the hand in some cases would occlude the object attempting to be selected and if several small objects are in vicinity of one another it could become difficult to select the desired object. One of the upsides of the pattern is its realistic properties which could provide a more immersive experience.

The techniques within this pattern include a **realistic hands** approach, **non-realistic hands** approach and a technique called the **Go-go** technique. The **realistic hands** approach is a technique that attempts to mimic the human hand visually and functionally to provide a realistic immersive experience. However, this has not yet been fully accomplished by anyone if we include precise tracking of the entire arm. With this technique, the head and torso would ideally need to be tracked as well to give a complete immersive experience. The **non-realistic hands** approach replaces the visually realistic hand with a handheld 3D tool of some sort - much like the virtual controllers displayed by the HTC Vive. When realism is taken out of the equation it leaves more room to make the interaction easier. With this technique there is no need for arms either which means the reach of the interaction can more easily be extended beyond the reach of the users arms. As this is not realistic it also requires some getting used to, as you break the users mental model of how the

world works. Breaking this model enough one gets to something like the **Go-go** technique by [19], in which the arm can be extended well beyond its normal reach. The techniques works by extending the user's reach non-linearly when the hand goes beyond 2/3 of the arms reach with a normal 1-to-1 mapping in the first 2/3 of the arms reach. The preliminary results obtained by Poupyrev et al. through user tests showed that the technique was intuitive for users to use; being preliminary the results are not statistically verified or reproducible. Other revisions of this technique have been made in which the mapping can be controlled by the user to potentially allow infinite reach by continuously growing the user's reach when the arm is beyond a certain threshold [2]. However, physical contact is rarely the model used for selection in the real adult world where pointing and speaking is a more common way for selecting e.g. pointing in a PowerPoint presentation.

Pointing Pattern

Pointing is probably one of the most universally used gestures across the world and the same is case for VR applications. Pointing is one of the most popular patterns applied for selection purposes. It works by casting a ray from a point on e.g. a controller or the head out into the distance and the first object that intersects with the ray will be selected when triggered by the user; much like a hand held laser-pointer. This pattern is usually better for selection whenever realistic interaction is not a requirement and it can handle selection in both close proximity as well as at a distance. One of the challenges with the pattern is the selection of small objects at a distance due to small hand tremors caused by the very accurate tracking in the technology. Almost no one can keep their hand perfectly steady and most current tracking technologies will register any tiny movements. Some techniques do attempt to mitigate that problem however.

As it is the most common selection pattern, a lot of techniques exist including:**hand pointing**, **head pointing**, **eye gaze selection**, **two-handed pointing**, and many more. **Hand pointing** is mainly used when the hands are being tracked with technology such as the Leap Motion[12]. In this scenario a ray will usually be extended from the tip of a finger or the centre of the hand and the selection triggered by a gesture or by voice commands. **Head pointing** is also a fairly used technique, where the centre of the field of view is used as a pointer so the user simply has to turn their head to centre the object they want to select in their field of view. The user then triggers selection either by dwell selection which selects the object after looking at it for a certain amount of time, or by pressing a button. **Eye gaze selection** is a technique that is more rare both because eye tracking technology is more scarce, especially in commercial VR, but also due to some of the problems present in the technique. As with head pointing, the user would need to trigger a selection by either button press or by dwelling on an object, which causes a problem as users need to be able to look at things without selecting them. Another problem is providing the user with feedback of what they are looking at, meaning the eye-tracker might not be completely on point, so a sort of reticle would be needed to show this, which can be distracting to the user; especially because our eyes constantly saccade all over the place. **Two-handed pointing** uses both hands to generate a ray that originates in

one hand, the one closest to the origin of view, and passes through the other hand, the one furthest from the origin of view. This technique was originally proposed by Mine et al. [16] for navigation by flying, but it can be adopted for selection across longer distances. With this technique the distance of the selection can be controlled by increasing or decreasing the distance between the hands, and the precision and speed of direction can be controlled through this factor as well. The closer the hands are together the faster the direction can be altered, however, with lower precision and vice versa.

Image-Plane Selection Pattern

This pattern is the one used in the MARUI plug-in described in Section 2.1.2. It greatly resembles the way we interact with a 3D space on a 2D monitor with a regular mouse input. With a regular mouse the position of the mouse is projected into the 3D scene in order to figure out which object one is hovering over with the mouse. In VR the pattern uses a combination of eye position and hand or tool position for selection as described by Pierce et al. in [17]. It differs from regular mouse input by having two points of reference, as changing viewpoint inside VR would also change what is being pointed at, which is not the case with a mouse on a regular 2D screen. Specifically, the pattern works by placing the hand or tool in front of the object one wants to select so there is a straight line formed from the centre of view to the hand/tool to the object and then pressing a selection signal. The idea with this pattern is to simulate direct touch at a distance which should make it easy to use as long as the object is clearly visible [3]. As VR is in stereo this pattern is slightly limited as it only works with one eye at a time for maximum accuracy and it can result in fatigue when used repeatedly due to holding the hands up in front of the head for an extended amount of time; a fatigue known as gorilla arms. Occlusion can also be a problem.

Pierce et al. present four techniques within this interaction pattern in their work [17], namely: **Head Crusher**, **Sticky Finger**, **Lifting Palm**, and **Framing Hands**. In the **Head Crusher** technique, users position their thumb and index finger below and above the object in 2D space. A ray is then cast from the centre of the dominant eye (user determined) through the midway point between the two fingers and into the scene. The system then provides feedback by clearly indicating the object that is the candidate for selection and the user can then issue a selection command to select the object. The feedback, selection command and origin of the selection ray is common for each of these four techniques. The **Sticky Finger** technique uses a single outstretched finger to select objects that are underneath it, much like a touch screen interface. This works best for large or close objects as the finger would have to occlude part the objects to be selected and with small enough object the finger would occlude it completely. The **Lifting Palm** technique works by placing a flattened outstretched hand below the object (in 2D space) you want to select so that the top of palm is level with the bottom of the object. A slight offset can also be added to select an object higher above the palms surface. The **Framing Hands** technique is a two-handed technique that requires users to form the two corners of a frame in a 2D space, the corners being created by their thumb and

index finger, surrounding the object they want to select. Though these are all selection techniques, Pierce et al. also mentioned how the pattern can be used for manipulation of objects as well as navigation of the scene. However, their ideas for manipulating mostly revolve around scaling either the world or the object, and the navigation is also always related to the selected object, with no option of using navigation without it being related to an object, which seems quite restrictive. The potential problem with scaling objects or the world is that users can loose sight of the size of objects which can cause misalignment and wrongly sized objects. A problem also mentioned by Pierce et al. is that the user must be able to interpret and distinguish between the 3D environment and the 2D frame used for selection with one eye. The techniques would also rely strongly on the quality of the hand-tracking used although the methods could be adopted to use a 3D tool as seen in the MARUI plug-in.

2.2.2 Manipulation Patterns

Manipulation of objects refers to the modification of an object's attributes such as its position, size, orientation, colour, etc. Manipulation is something that typically follows selection like when moving a folder on the desktop of a computer by holding down the mouse button while moving the mouse. The majority of this project revolves around the ability to be able to change an objects position and orientation accurately in relation to other objects. In VR this is, again, complicated by the third dimension of interaction and the fact that all objects are not placed at the same distance from the interaction point, contrary to mouse interaction where the depth is always zero. In the following sections, some manipulation patterns and their underlying techniques are described and discussed.

Direct Hand Manipulation Pattern

This pattern relates to how we move objects around in the real world i.e. by physically touching them and moving them around. This has been implemented in a few different ways in the different applications we have experience with. One way is to select an object which then attaches it to the virtual representation of the hand (be it a model of a hand or 3D tool). Another way is to have physically accurate virtual hands which allow users to push objects around in the virtual environment, and thus also allowing the to grab them (without having to select them first). More implementations exist and new ones are created regularly as new hardware gets developed. This pattern has been shown to be more efficient than other manipulation patterns and thus provide greater user satisfaction [2]. As with the **Hand Selection Pattern** from earlier this pattern is limited by the physical reach of the users as they can only manipulate objects in their proximity. In fact this pattern strongly relates to the **Hand Selection Pattern** in the sense that the techniques used are much the same, e.g. the **realistic hands**, **non-realistic hands**, and the **Go-go** technique. The difference being, that here instead of selecting the object one manipulates it instead.

Proxy Pattern

This pattern revolves around using a proxy to manipulate a selected object; a proxy being a local object that can be either physical or virtual. This proxy maps directly to the selected object in the sense that when you move or rotate the proxy the selected object does so as well. Naturally, this pattern works well when the object needs to be manipulated as if it were placed in the users hand. According to Jason Jerald (in his book on VR [8]) a limitation of this pattern is that objects can be hard to manipulate when there is an orientation offset between the proxy and the selected object being manipulated. However, this could be largely, or possibly even completely, circumvented by using a proxy item that has no visible orientation, i.e. a smooth sphere. We would argue that a more pronounced limitation, at least when using a physical proxy, is the fact that one hand will be occupied by the physical proxy instead of a potential 3D interaction tool with more functionality. A 3D tool can also be used as a physical proxy but most 3D tools, e.g. the Vive controllers, are not very handy to have as a physical proxy when handling rotation, and with a 3D tool there would likely be an orientation offset. **Tracked physical props** is the main technique associated with this pattern, in which the user holds a physical object that is mapped to one or multiple virtual objects in order to directly manipulate them.

3D Tool Pattern

This pattern makes use of intermediary 3D tools which users can directly manipulate using their hands or a motion tracked controller, and the intermediary 3D tool then manipulates a desired object. An example could be a virtual pointer allowing users to extend their virtual reach or a screwdriver which would enable a specific interaction with object such as mapping a small physical rotation to a large virtual rotation. This pattern is mainly used whenever an enhancement of a certain capability is needed. A limitation of this pattern is that it takes effort to keep track of different 3D tools and their functionality and applying to right appropriately to each object. The techniques within this pattern include **Hand-held tools**, **Object-attached tools** and **Jigs**.
The **Hand-held tools** technique makes use of virtual tools that are attached to the hand or held in a hand such as a paintbrush used to draw on surfaces of objects, or either of the two examples presented above. These tools can help improve understanding and can be easier to use than widgets in a menus due to their inherent directness; you just grab a tool and use it - once each tools functionality has been learned. **Object-attached tools** technique is very similar except for that fact that with this technique the tools are attached to the objects on which they can be used resulting in a more coupled signifier. The **Jigs** technique is meant to aid users in precise alignment and modeling of objects by providing virtual guides, such as grids, rulers, angle irons, etc, that attach to object vertices, edges, and faces. The users can adjust these jigs, e.g. the size of the ruler, and use them to snap objects into exact positions and orientations using the jigs on the objects they are attached to.

2.2.3 Compound Patterns

Compound patterns are the combination of two or more patterns working together to form more intricate patterns that can potentially take the strengths of one pattern to eliminate or reduce the weakness of another. Examples of compound patterns are the **Pointing Hand Pattern**, **World-in-Miniature Pattern**, and **Multimodal Pattern** described in the sections below.

Pointing Hand Pattern

Some of the patterns mentioned previously (**Hand Selection Pattern** and **Direct Hand Manipulation Pattern**) have limitations in the form of a limited reach, where the user is only able to select or manipulate objects that are within their reach. The **Pointing Pattern** is capable of selecting items at any range as long as the object is not too small and it requires only limited hand movement as a small change at the wrist results in a larger change out in the distance. According to Bowman et al. in [3] pointing is not good for object manipulation when it comes to changing the objects position (depending on the task) due to the arcing movement around the user. But by using the **Pointing Pattern** to first select the items and then the **Direct Hand Manipulation Pattern** to then manipulate it using the hand or a tracked controller as a proxy for the object, the **Pointing Hand Pattern** is born. One thing to consider though is that this would make the manipulation of the object 1-to-1 making the movements of manipulated object relatively small. If you need to move a far away object over a larger distance something else needs to be considered. Some of the techniques developed for this pattern attempts to solve this problem and create stronger signifiers to show how the manipulation works, such as the **HOMER** technique and the **scaled HOMER** technique, and the **extender grab** technique.

The **HOMER** technique by Bowman and Hodges [2] causes the hand to jump to the object being selected using the **Pointing Pattern** which creates a clear signifier of which object is being manipulated. The user is then able to position and rotate the object at a distance as if it was in close proximity. To solve the problem of not being to move the object over larger distances, Wilkes and Bowman [29] developed the **scaled HOMER** technique in which the objects movement is scaled based on how fast the users move their hand when manipulating the object. The faster they move their hands, the more the object moves and vice versa, allowing for both crude and precise placement and potentially also rotation. This type of interaction is called velocity-based interaction or velocity-based scaling. The **extender grab** technique by Miné et al. [16] is used to scale objects translation based on the distance from the user to the object. So the further away an object is grabbed the more it moves when moving the hand with which the object was selected, and the scale is determined on selection, i.e. based on the original distance to the object. The rotation remains a direct mapping in this technique setting the objects rotation to the orientation of the users hand.

World-in-Miniature Pattern

This pattern was utilised in the EditorVR as described in Section 2.1.2. A World-In-Miniature (WIM) is an interactive 3D representation of the environment the users are immersed in that is updated live when any changes are made to the world (or vice versa), and the users are able to see themselves and the movements they do inside the WIM; as well as what direction they are looking in. As indicated by the name, the 3D representation is relatively small akin to something like a child's dollhouse. WIM for VR was explored by Stoakley et al. in [22], where an informal user study found that WIM for VR was intuitive to use and was able to provide easy manipulation and selection of objects at a distance. To interact with the WIM other patterns are used, e.g. some of the ones described above, and by using WIM the **Direct Hand Manipulation Pattern** makes users able to move objects through the entire environment by manipulating the WIM representations of the real objects. WIM can also be used to move oneself through the environment.

In this chapter, we took a look at some applications related to what is being developed in this project, in the shape of room planner applications and existing applications utilising interaction in virtual reality. We also took at look at some of the interaction patterns and techniques that others have developed and tried throughout the existence of interaction in VR. All of the information presented is meant to give some insight into some of the things that currently exists within interaction in Virtual Reality and 2D room design. Its second function is to serve as inspiration for the patterns and techniques that have been designed and developed throughout this project. Some of the patterns and techniques will be referenced later in the report, and the ones that are not were not considered to viable contributors to solving the problem being addressed in this project. One thing noticeable after reading through all this information is that only a few of the techniques described in the literature are used in current virtual reality applications and it is not the same ones; more or less confirming that a common consensus regarding which pattern and technique is most suitable for VR has yet to be reached. Granted it is also possible that with the complexity of VR there wont be only one pattern and technique that fits all types of applications.

CHAPTER 3

DESIGN AND IMPLEMENTATION OF PATTERNS

Based on the information gathered, and on experience from the previous project [9], we will try to improve the previously created interaction system. This chapter presents the ideas and design behind the overall system as well as the design of the three patterns developed in this project and the thoughts behind them. These three patterns are used for the preliminary test described in Chapter 4 and are all manipulation-based compound patterns. Due to time constraints these were the only patterns explored more in depth, although ample patterns exist besides the ones explored in this project. We settled on these three patterns through discussions and personal experience with VR as well as how well each pattern would fit within the chosen kitchen design theme. The first part of the chapter covers some of the considerations made for the interaction fidelity of the system and then the design we envisioned for a finished product is detailed. Finally, in Sections 3.3 to 3.5 each of the three patterns are covered by first describing the design behind each of the patterns and then how they were implemented.
Plasticity, a term that refers to any one things' (i.e. menus, interaction patterns, interaction techniques, etc.) ability to automatically fit a set of hardware and environmental constraints, was something that was not greatly considered during the design of these menus. On the contrary, we were focused on making something sure to work well for the Vive without putting much consideration into other HMD technologies. In the future other HMD technologies would have to be considered to improve the flexibility of the system. The system is implemented using Unity [23], a free 3D game engine that supports the HTC Vive. As previously mentioned, the system is an extension of a previous project that focused on menu design and interaction and here the focus is on interactions with objects.

3.1 Interaction Fidelity

Interaction fidelity is a term used to describe the degree to which physical actions used for a virtual task correspond to the physical action used in the equivalent real-world task [4]. In other words it is a term used to describe the realism of the interaction inside a system. There is no common right or wrong answer when it comes to deciding what the interaction fidelity in an application needs to be like; it all depends on what the goal of the application is and what task you are trying to solve. VR interaction has seen it all, interactions trying to imitate reality almost to a fault and, oppositely, applications attempting very futuristic interaction that has almost nothing to do with reality; although most applications are somewhere in between. Each end of the spectrum has its advantages and disadvantages.

With realistic interaction (high interaction fidelity), users tend to be more immersed as the environment works in a way akin to the real world. Realistic interactions are typically used in cases where realism is important, such as training applications for real world tasks or simulation for e.g. flying a plane. If the interactions are not realistic in such applications then the experience gained from using it cannot be transferred properly to the real world. An advantage of realistic type interactions is that there is usually less learning required for users making an application easier to simply pick up and use. However, in some cases the real world has some serious limitations to what is possible, which can be circumvented inside virtual reality through non-realistic interactions or magical interactions.

With non-realistic interactions (low interaction fidelity), users will be capable of doing things that are not at all possible in the real world, such as manipulating objects from afar and change the environment with the push of a button. This has a higher potential of breaking the immersion of the users as they are able to do things that are not real, however, by allowing these magical interactions it is possible to increase performance and decrease fatigue by making some actions a lot easier and a lot faster [8]. An issue with these magical interactions is that users have to learn how to use them and developers also need to learn how best to develop them. With realistic interactions one simply tries to mimic something that already exists whereas non-realistic and magical interactions first have to be designed more or less from scratch.

In this project we aimed to have both realistic and non-realistic interactions in order to achieve the most performance from the system, both in terms of speed and usability. The base system used (see next section) was made with elements of both realistic interactions and non-realistic interactions to use depending on the use case. This project attempts to refine these interactions by taking a look at different possibilities, most of them erring towards the non-realistic interactions, as the options are somewhat limited within the realistic interaction spectrum. We also wanted it to be possible for users to always be able to interact with objects no matter the distance to the object (within reasonable parameters considering the kitchen design scenario). This alone pretty much rules out most types of realistic

interaction. The controllers for the HTC Vive also have a few buttons on them which makes sense to utilise for some types of interaction, and as such the fidelity is quite dependant on the hardware you are using.

3.2 Overall Design

Since the project had a limited time frame, a fully functional system could not be developed, however, that did not mean that we did not consider what a fully developed system could look like, and what it should contain in a kitchen design scenario (though kitchen design is only one possible use area). In this section we take a short look at what we imagine would be included when the system is fully developed while also discussing how the lack of some of these features might influence the user experience. First, to understand the motivation behind some of the choices made in the design and implementation of the system as well as the experiment conducted, we first give an example of what we imagine a common use case might be like.

When someone wants to buy a new kitchen they usually have a few choices depending on what they are going for, be it a premade IKEA kitchen or a custom design from a kitchen retailer (e.g. Invita in Denmark). They can either go to a store and find something that fits their house or invite a designer to their home and have them draw something up and maybe be able to show some 3D renderings of the final product. In either case the customer rarely gets the full picture of what a kitchen would look and feel like when it is installed and ready to use, which is what we want to get closer to. We imagine the system would be installed at a retailer of some sort where users would then be able to use it. Now, there are a few ways the system could be initiated. Either the user will provide information about the room, in which the kitchen is to be, dimensions, window and door placements, wall colours etc. (or a blueprint) which will then be reflected in the system so that the user has an empty replica of the room he or she wants to build the kitchen in. Another possibility would be to have a bare-bone square room the user could then edit to resemble the wanted setup. Alternatively, an option that was also considered possible was that a user could start out with an example kitchen (possibly even a kitchen designed for the user) and then edit it from there, however, that is a slightly different scenario. In either case the user will equip the Virtual Reality HMD (in our case the HTC Vive) and start out by being guided through a virtual tour of the possibilities available to them in the virtual environment. At a retailer there might even be a clerk that can offer assistance, however, the idea is that the user should be able to use the system after a thorough virtual tutorial. Users would select kitchen inventory from a virtual 3D catalogue which would spawn objects in the virtual environment for the users to place as they choose. From there on, users would be able to configure the kitchen as they pleased, within the parameters of the retailer; a configuration could include the change of colour or handles on drawers etc. or even changing entire objects i.e. from a stove to a refrigerator. While this is something that is relatively simple to describe in words, the implementation thereof is more elusive due to its uncharted nature, especially when it comes to ease of use.

As previously mentioned (see Preface), we are not starting from scratch in this project. A system foundation is already in place from a previous project which is used as a basis for the design and development in this project. To give an idea of what that system includes, we provide a short description of the functionality that system provides. In Figure 3.1 a schematic of the controls is displayed which is used to describe the functionality and to show the layout of the controls on the two controllers.

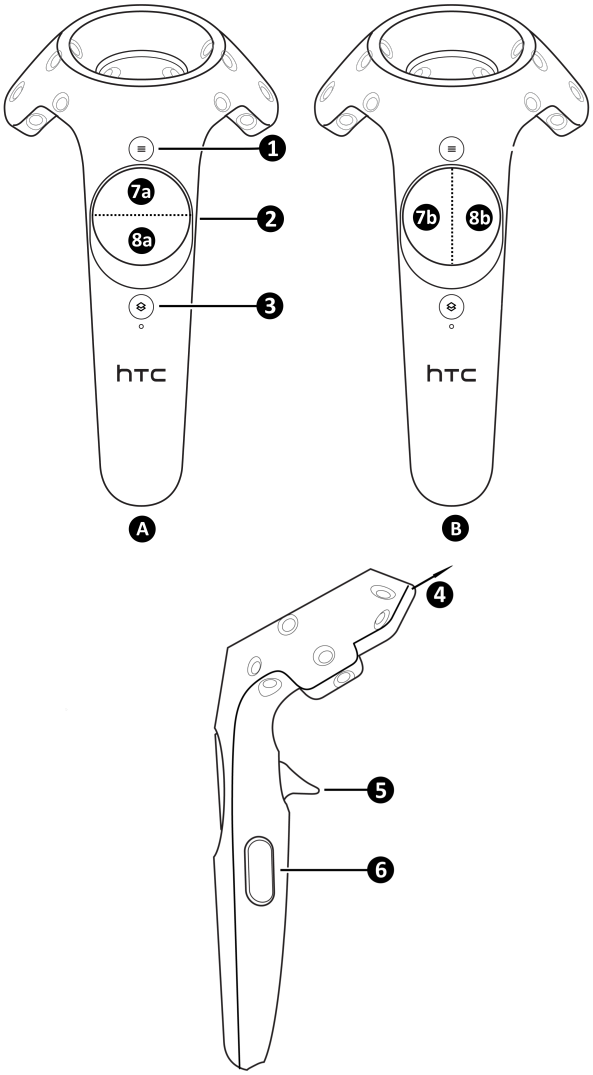
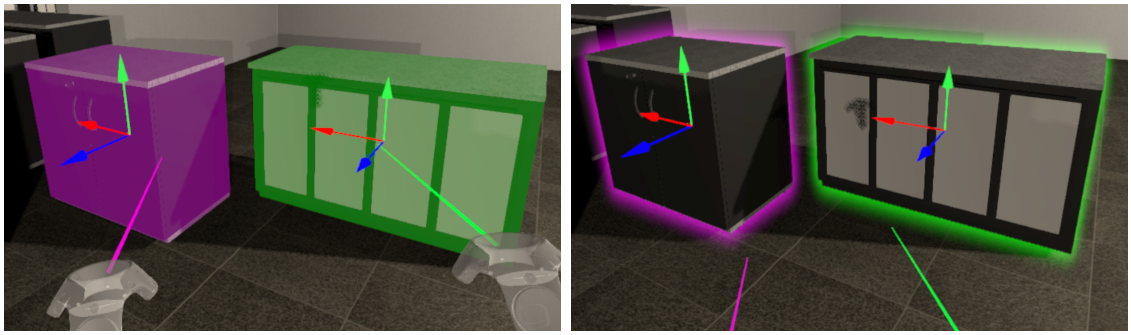


Figure 3.1. Layout of the controllers.

The first thing important to cover is that the system has two different interaction states: proximity interaction and laser interaction. Looking at Figure 3.1, number **(4)** shows the position and the direction of the laser when it is turned on. When the laser is turned off, the system follows the **Hand Selection Pattern** (see

Section 2.2.1 and **Direct Hand Manipulation Pattern** (see Section 2.2.2)for selection and manipulation where a virtual representation of the Vive controller is displayed inside the virtual world and this is used for both selection and manipulation of objects when in proximity (controller has to be inside the objects collider). When the controller is in proximity of the object, the object will be highlighted with a colour specific to that controller (each controller has its own colour - see Figure 3.2). To select an object, the trigger **(5)** is used and to manipulate the position of the object (to grab it) the grip-buttons **(6)** on the side of the controllers are used, where the user has to hold them in while moving the object. In case the reader is unfamiliar with the Vive controllers it is worth noting that while the grip-buttons are plural, they function as one-button able to be pressed from both sides of the controller, and when both sides are pressed simultaneously the button(s) becomes easier to press. When the user releases the grip-buttons, the object is released. When an object is selected or grabbed it becomes highlighted by being outlined with the colour of the controller selecting or grabbing it (not to be confused with the highlight when in proximity which highlights the entire object). Furthermore, when an object is selected a 3D gizmo is displayed to show an objects local coordinate system as shown in Figure 3.2b: the blue arrow is the objects Z-direction, the red its X-direction, and the green arrow its Y-direction.



(a) *(b)*

Figure 3.2. Figures showing the highlighting of objects when *(a)* hovering over them or being in proximity of them and *(b)* selecting them

When the laser is turned on, the system follows a variation of the **Pointing Pattern** (see Section 2.2.1) which is used for both selection and manipulation in this system. The user can point at an object, which will highlight it, and use the trigger or grip-buttons to once again either select or manipulate the object. While selection functions the same, manipulating the position is handled by attaching the object to the end of the laser (which is on the object where the laser intersects it). This will cause the object to move whenever the user moves his hand, like when doing proximity interaction, but more so when the user rotates his/her hand which will cause the object to move in an arc as the laser will continue to point straight forward from the controller with the object attached to the end of it. To control the length of the laser (which is potentially infinite when an object is not hit) when an object is grabbed the user can use the trackpad **(2)** as a scroll wheel by moving

a finger across is (up and down for scrolling) which will cause the laser to change length based on how far away the object is. When the object is further away the scroll will be faster than when the object is close in order to allow for better control when objects are in close proximity. Also different from proximity interaction is the fact the users do not have to hold the grip-buttons down while moving an object, instead, they have to press it once to grab the item and press it again in order to release it. The reason behind this is that we feel it reduces the strain on the forearm during long movements of objects, whereas in proximity interaction the manipulations are usually relatively short and meant to feel more realistic. Other than these differences, the two interaction types provide the same overall functionality, and the reason for using two different interactions is to allow users to interact with object at a distance to do some gross manipulation and in proximity for the more realistic feel and precise interaction.

Looking at Figure 3.1 it is split into **(A)** and **(B)**, and the differences between the two is the functionality of the trackpad. **(A)** is the functionality provided when no object is selected or grabbed whereas **(B)** is when object are either selected or grabbed. The upper part of the trackpad **(7a)** can be used to turn on and off on the laser on press to allow for either proximity interaction or laser interaction, and **(8a)** is used for teleportation i.e. navigation. When a user puts a finger on the trackpad it will display a pie menu, that shows these options as well, and when the user has a finger on the lower part of the trackpad **(8a)** it will also display a bezier curve between the controller and a point on the floor or an object (see Figure 3.3). When

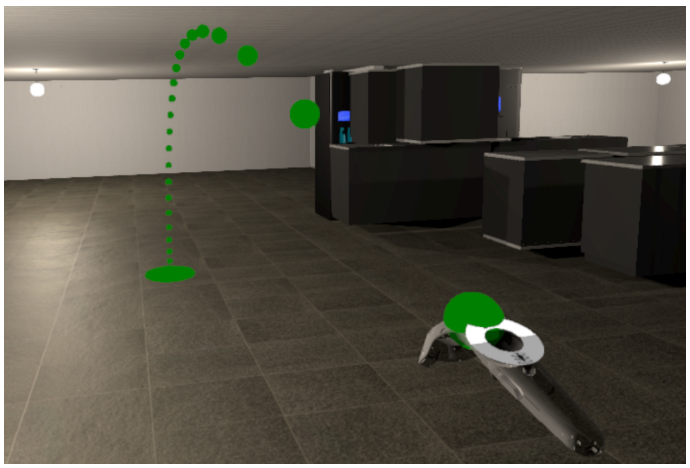
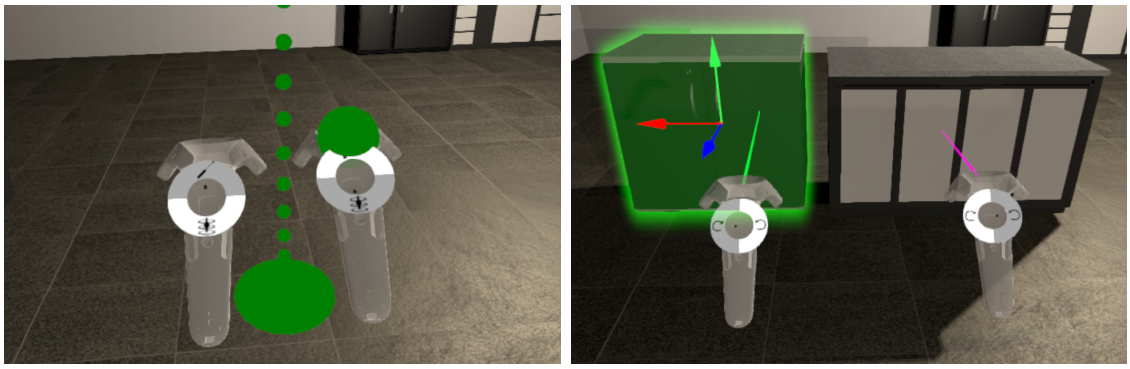


Figure 3.3. Visualisation of the Bezier curve based teleporter.

the curve is green the user will teleport to the end of the bezier curve when pressing the trackpad and if the curve is red it means the endpoint is ineligible for teleportation and nothing will happen on press. When an object is grabbed or selected the pie menu also changes and can then be used to rotate the selected or grabbed object, **(7b)** causing it rotate 90 degrees left (clockwise) and **(8b)** 90 degrees right (counterclockwise). The two variations of the trackpad pie menu can be seen in action in Figure 3.4.

The remaining functionality is not important to the current project scope and not fully implemented in the base system either, but we will cover it briefly here anyway. The button at the top of the controller **(1)** (also called application-button) is meant to be used to open menus. When no object is selected it would open an overall systems menu where the user would have access to settings such as deciding whether or not the grip-buttons should be a hold or a toggle or setting the sensitivity of the trackpad scroll. When an object is selected the button would open a context based object menu, based on the type of object being selected, where different types of objects would have different options. The final button below the trackpad **(3)** is a Vive-controlled button, in the sense that it is used to open the Vive menu and when pressed down long enough they turn on and off the controllers; and as such, we have no control over this button as such but it has its uses as it also allows users to go into desktop mode where they can use the VR headset to navigate the PC they are using it on.



(a) *(b)*

Figure 3.4. Figures showing the pie menus displayed on the controller; *(a)* showing **(7a)** and **(8b)** and *(b)* showing **(7b)** and **(8b)** where the left object is being selected and the right object is being grabbed

Important to mention is that, while a system foundation was in place, a lot of improvements have been made to the core system, outside of the patterns and techniques developed for this project, which is not well reflected in the contents of this report as it would require the addition of a lot of source code which we feel would not fit well in this report. These improvements also apply to some of the functionality mentioned above, such as the rotation of objects which has been made so that objects cannot be rotated into walls or other objects. This also applies to many of the patterns and techniques mentioned throughout the rest of this project as many improvements were made to the implementation of each of them which is not documented in great detail for the same reason as mentioned above. However, we do try to mention some of the greater changes to the implementation of each of them whenever appropriate.

With the functionality of the ideal system, as well as the functionality of the current system, explained, we will move unto the different explaining the patterns that were implemented on top of this system in the following sections. As mentioned we

did not have the time to try and implement all of the techniques for the patterns before the preliminary test used to root out the pattern we want to explore further. As such, only the techniques partly implemented are described in these sections whereas the other techniques for each of the patterns (which were only conceptually designed) are described in Appendix A.

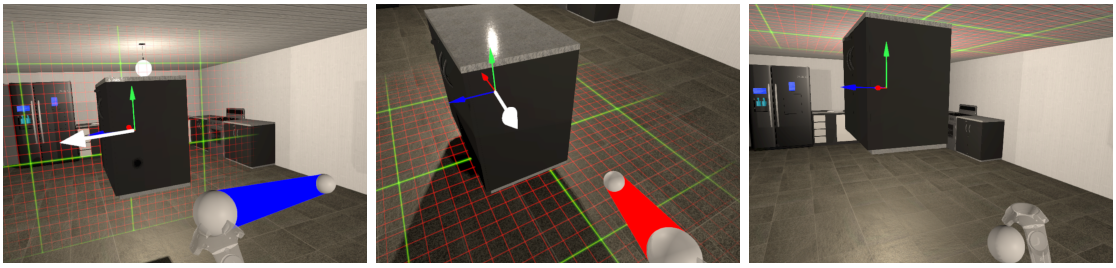
3.3 Grid-based Pattern

This pattern is a compound pattern inspired by the **Proxy Pattern** and the **3D Tool Pattern** (see Section 2.2.2, specifically the **Jigs** technique. In this pattern users move objects in a world-based grid with a specific grid-spacing which should ideally be controlled by the user based on the level of precision required. The objects can be moved along single axis in the grid, as well as in in two-dimensional planes (XY plane and ZY plane etc). In order to be able to accurately align objects next to each other, they cannot move through walls or other objects, which means that when an object is pushed up against another and the wall behind it, it is aligned. Objects are manually rotated to face the desired direction using the rotation functionality described in the previous section. One of the strengths of this pattern is that it can potentially be used for any shape and size of object, no matter if it is symmetric or not. Since the entire placement is based on a grid it almost completely disregards what type of object is being placed, however, collision could become a problem for aligning certain concave or complex shapes. As we did not feel the laser interaction would work for moving objects in a grid, we decided to look at a few possible interaction techniques instead.

The technique used for this pattern used a virtual proxy-based manipulation instead. When the user selects an object and small sphere will appear at the position of the controller used to select the object. The user would then need to grab the proxy (the sphere) in order to move the selected object which is moved based on the direction one moves the proxy in. The speed of the movement is based on how far the proxy is moved from its original location. Furthermore, when an object is being moved, a white directional arrow is displayed in the centre of the object in order to indicate the direction the object is currently being moved in. This is also indicated at the proxy item as a line appears between the original position (which will show a smaller sphere) and the current position of the proxy, and this line will have a colour based on the direction the proxy is being moved in. This colour corresponds to the axis colours displayed on the 3D gizmo of the selected object. When an object is moved in a plane, the colour of the line turns into a gradient of the colours the plane is made up of. When the proxy is released it will reset its origin to the position at which it was released, meaning it is possible to re-position the proxy. This interaction is displayed in Figure 3.5, where one can also see that the visual placement of the grid changes.

In Figure 3.5a the grid is placed behind the object occupying the ZY plane (relative to the object as indicated by the objects gizmos). In Figure 3.5b the grid is placed below the object and in Figure 3.5c above the object. This positioning is based

on a few variables, namely, the position of the object relative to the users head, the viewing angle between the head and the object, and the direction the object is currently moving in; the latter being the most dominating factor.



(a) *(b)* *(c)*

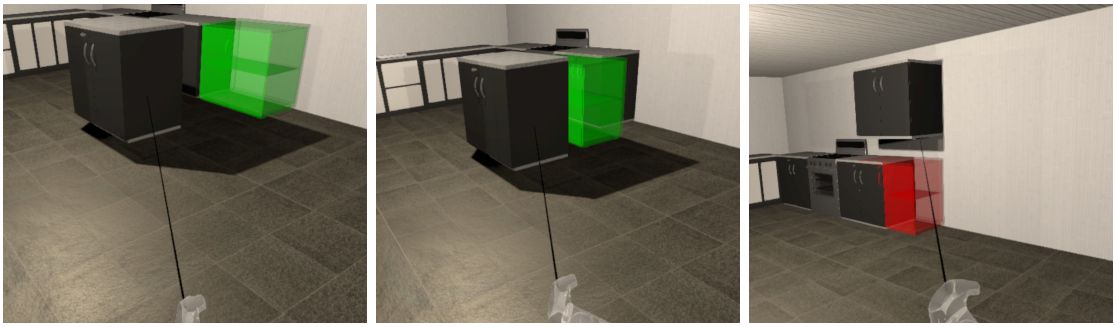
Figure 3.5. Figures showing the interaction with objects when using the **Grid-based Pattern** with the **Crane-control** technique. The green line in the pattern indicates 1 unit (metre) and the red lines indicate the grid-size (the amount the cube moves each time). *(a)* Showing sideways movement where the grid places itself behind the object. *(b)* Showing the user looking down and moving the object towards him/her, with the grid placing itself underneath. *(c)* Showing the user looking up with the grid aligning itself on top.

3.4 Snapping-based Pattern

This pattern was inspired by the wish to make our laser interaction a feasible method for precise placement and alignment of objects. The laser interaction is essentially a **Pointing Hand Pattern** (Section 2.2.3) with some twists as one does not move objects in a 1-to-1 manner. Instead objects are moved as described in Section 3.2 where objects are attached to the end of the laser and can be moved closer or further away by using the trackpad as a scroll-wheel. As with the previous pattern, objects cannot move through each other or the wall in order to make it easier to place and align them. With this method, placing items relatively close to each other is not immensely difficult, however, aligning them next to each other with precision requires the user to be standing very close to the objects and preferably right in front of them. The precise movement required to exactly align objects is something that is close to impossible using only the basic laser interaction of the system. With this pattern, the idea is to offer users assistance to both place and align objects through the use of transparent ghost models and snapping. A transparent ghost model is a copy of the model being placed which is shown in the position the object is going to snap to when prompted. An example of this interaction is displayed in Figure 3.6 where one can also see that the ghost model can be coloured red, which means that the object will not snap when prompted, as it does not fit at that position. The system does try to account for this, by looking at nearby available positions to assist the user in the placement. Objects in this pattern are able to snap some of the surfaces together based on their cuboid shape and the surrounding

objects. As we are doing a kitchen design scenario, the front of all objects, and the top of most others, are considered ineligible surfaces for snapping. It has to be mentioned that this pattern is made a lot easier to design and implement with cuboid shapes in mind, as their complexity is relatively simple, however, if one had to use a pattern such as this for complex shapes it is likely that manual snapping parameters would have to be used. With cuboids it is possible to model how they should best snap to each other in most cases, especially when walls and floors are taken into consideration as well; as we know objects should not be floating and should be all the way up against walls (or the back and sides of other objects). One of the strengths of this pattern is that users are guaranteed that the objects are perfectly aligned when using the ghost models to snap them together. A weakness is its lack of flexibility when multiple shapes of objects are considered.

The technique used for the pattern is the **Proximity** technique which is covered in full detail in Chapter 5 as well as the other techniques used for this pattern.



(a) *(b)* *(c)*

Figure 3.6. Figures showing the interaction with objects when using the **Snapping-based Pattern** with the **Proximity** technique. *(a)* Placing the object close to a wall shows the ghost. *(a)* Placing the object in front of another object places the ghost in front of that object. *(c)* Placing the object in an invalid location shows a red ghost.

3.5 Free-hand Pattern

As with the previous pattern, this pattern is inspired by a desire to make the laser interaction more accurate for precise placement and alignment of objects by combining even more together to make a strong and slightly more complex pattern. This pattern combines our laser interaction with the **Proxy Pattern** (see Section 2.2.2) to make it possible to do rough placement with one hand and precise placement and alignment with the other hand. An example of this interaction is displayed in Figure 3.7 where Figures 3.7b and 3.7c are an example of a proxy based movement (of course hard to properly display in still images). The technique implemented for this pattern was the **two-handed** technique in which whenever an object is grabbed and released, the other hand is able to grab the thin air and function as a proxy for the object that was last placed. This technique is the equivalent of placing an item

(a) *(b)* *(c)*

Figure 3.7. Figures showing the interaction with objects when using the **Free-hand Pattern** with the **two-handed** technique. *(a)* Showing rough placement using the laser. *(b)* The object is slightly misplaced. *(c)* Showin the use of the precise movement to finish placement of the object.

using the laser interaction of the system and then teleport to the item once grossly placed and then use the proximity interaction to do the final precise placement and alignment. Except with this technique the users are able to do it at a distance.

CHAPTER 4

PRELIMINARY TEST

In this chapter the design of an informal preliminary user test is described. First, the purpose of the test is discussed followed by information regarding the equipment used for the test and the its participants. Then, the procedure of the test is detailed followed a discussion of the expectations we had for the test in terms of participation and results. As the test was informal and with only a few participants, the results do not merit a chapter of their own and are as such covered in the end of this chapter. This test used a basic within-subjects (repeated measures) design as all participants tried all the experiment conditions, which in this case was the three different patterns, and they were all asked the test questions. By having every participant try every pattern they were able to make a subjective decision on which pattern they prefer which is the goal for this test. The order in which they tried the patterns was randomised in order to limit any carryover effect there might be between the different patterns and to remove, or at least reduce, any bias the order might have.

4.1 Purpose

To test which pattern we want to continue with and develop multiple techniques for, we first have to figure out which pattern is most broadly accepted as the best. To do this, we made a small test using each of the three patterns using one of the designed techniques for the patterns which was implemented in a prototype stage. The idea with the test is to keep the pattern in focus as the technique used should influence the test as little as possible.

The three patterns being tested are a grid-based pattern, snapping-based pattern, and a free-hand pattern. The epitome techniques for the patterns are the crane-control technique for grid based pattern, the **Proximity** technique for the snapping-based pattern, and the two-hand technique for the free-hand pattern. The techniques are all epitomes and thus only function as well as needed to properly test the different interaction patterns.

Expectations

The expectations from doing this test were quite limited, other than discovering which pattern was preferred by users as well as why they preferred it. We expected that there would be a preference towards one pattern which would allow us to focus our attention that specific pattern.

4.2 Equipment and Stimuli

The test was executed in the corner of an office with a mostly square open space as displayed in Figure 4.1. The test area was relatively small compared to the size the Vive is capable of tracking, but users were able to move around slightly and teleport whenever the physical space was insufficient. The test design did not require much physical movement in any case, as the virtual room users were placed in was much larger than their physical space, meaning that users had to use the teleporter when navigating the environment. The virtual environment was running using an HTC

Figure 4.1. A picture of the physical setup used for this preliminary test as well as the experiment

Vive (see Figure 1.1) on a desktop PC with the following specifications:

- Intel(R) Core(TM) i5-4460 CPU @3.2 GHz
- 32 GB RAM
- NVIDIA GeForce GTX 970 with 4GB of VRAM

These are some of the minimum requirements to be able to run the HTC Vive system with the required frame-rate of at least 90 frames per second (the screen's refresh rate). This is something that was always being considered during the design and implementation of the system (and the virtual environment), as anything less than 90 frames per second can make the environment feel sluggish and give users motion sickness. That being said, we felt that users needed to have a better visual experience than what the default standard shading of Unity was able to provide using the basic shapes in Unity. To achieve that, we decided to spend some time locating some 3D

models of domestic appliances, sinks, and other things that fit inside a kitchen and gave them and the room they were placed in some decent shading. The result of this is displayed in Figure 4.2, which also shows the kitchen users were asked recreate in the procedure below. Using more or less realistic models was also meant to aid users in their understand of the environment and to make any tasks they had to perform in VR easier.

Figure 4.2. Screenshot of one of the kitchens the users were asked to replicate

4.3 Participants

As this was an informal user test meant to narrow the scope of our further design and implementation, we did not spend much time gathering participants. As such, the participants for this test were all people sitting in the same office the test was conducted in. They all had knowledge of the system but no experience using it which was fine for the purpose of this test. As for the amount of participants needed for this test, we were aiming to get a clear picture of what pattern users prefer, and once that was possible to establish, that would be the number.

4.4 Procedure

To test these patterns, two scenarios are explored. The first scenario will be a sandbox scenario where users can build whatever they want with the models made available to them. . The second scenario will be a mock-up scenario where users will have to recreate an existing kitchen setup, using the same models. Each participant will try each pattern one after the other in a randomised order.

As mentioned this is only a small test meant to determine which pattern to continue with and as such the data collection will be minimalist for this test. A small set of questions were prepared to have some structure for each participant. The randomisation of the pattern order will be handled by the application. The questions were asked between the test of each pattern, and are as follows:

1. What did you most like about the system?
2. How long did it take you to get good at using the system?
3. Were you able to create what you wanted to create?
4. Could you have created a more interesting scene given more time?
5. What did you dislike about the system?
6. What suggestions or ideas do you have for improving the system?
7. Did you feel tired from using the systems? If so, in what areas of your body did you feel fatigue?
8. Any other comments?
9. (After all patterns were tried) Which pattern did you like the best?

4.5 Results

This section will cover the main points of feedback that was received for each pattern in order of: the **Grid-based Pattern**, the **Snapping-based Pattern**, and the **Free-hand Pattern**.

Grid-based Pattern

In general participants were positive about the pattern, noting that the grid is nice, especially if it could be displayed in more directions at a time. They also found it easy to place items, with the exception that, due to how the grid works, small gaps could be present between objects. They all felt that they were more or less able to create what they wanted to using the pattern. The one common complaint was that the proxy did not move with the user as they teleported around; this caused some confusion among participants as they had to re-select the object. A suggestion to solve this was proposed by several participants; that one should simply be able to grip in anywhere mid air and the proxy should appear there as one started to drag the controller.

Snapping-based Pattern

Participants felt that this method was very good due to the fact that the snapping helped set rules for how things could be placed, as well as finding it quite fast to use. They especially liked the fact that the pattern automatically handled the rotation of objects. When asked if they felt like they could create what they wanted the answer was a resounding yes. Some negative comments included that moving items along the floor was hard, and that the grip button hard to use.

Free-hand Pattern

The consensus for this pattern was that it was very nice to have the option of more precise movement as opposed to just having the laser, although some participants rarely ended up using it. Participants also found the technique easy to use, with the exception of one who found the it complicated for first time use. Additionally it was mentioned that it was hard to remember that it was always the last object you had picked up with the laser that the precise movement affected.

Conclusion

Generally the most positive feedback was given to the **Snapping-based Pattern**. Meanwhile the other patterns received a lot of constructive criticism that could be used to improve them. When asked which pattern the participants preferred, they all agreed that the **Snapping-based Pattern** was the best. Based on this, we decided to focus on techniques using this pattern.

CHAPTER 5

DESIGN AND IMPLEMENTATION OF TECHNIQUES

Based on the results of the test in Chapter 4, three different techniques were designed and implemented using a snapping-based pattern (see Chapter 3 and Figure 3.6). This chapter will, for each technique, first explain the process of designing the technique, thereafter, the implementation of the technique is detailed. The design will cover not only the final system, but also things that were considered but not implemented, either due to time constraints or due to being excluded for reasons that are explained, whereas the implementation will cover only important parts of the final system. If no argument is presented as to why something was not included, it was due to time constraints. The implementation of the following techniques was a lengthy iterative process and in this chapter only the final iteration is presented. As with the decision to only include three patterns in Chapter 3, we decided to limit ourselves to three techniques as well, mainly due to time constraints of the project. The three techniques were created through discussion and examination of previous techniques, and as with the patterns, the kitchen scenario was also taken into account, and finally personal experiences with using VR.

5.1 Proximity Technique

The **Proximity** technique was the technique that was prototyped for the **Snapping-based** pattern in Section 3.4. We called it **Proximity** technique due to the nature of how the technique works, as objects attempt to snap to appropriate positions based on its proximity to nearby surfaces.

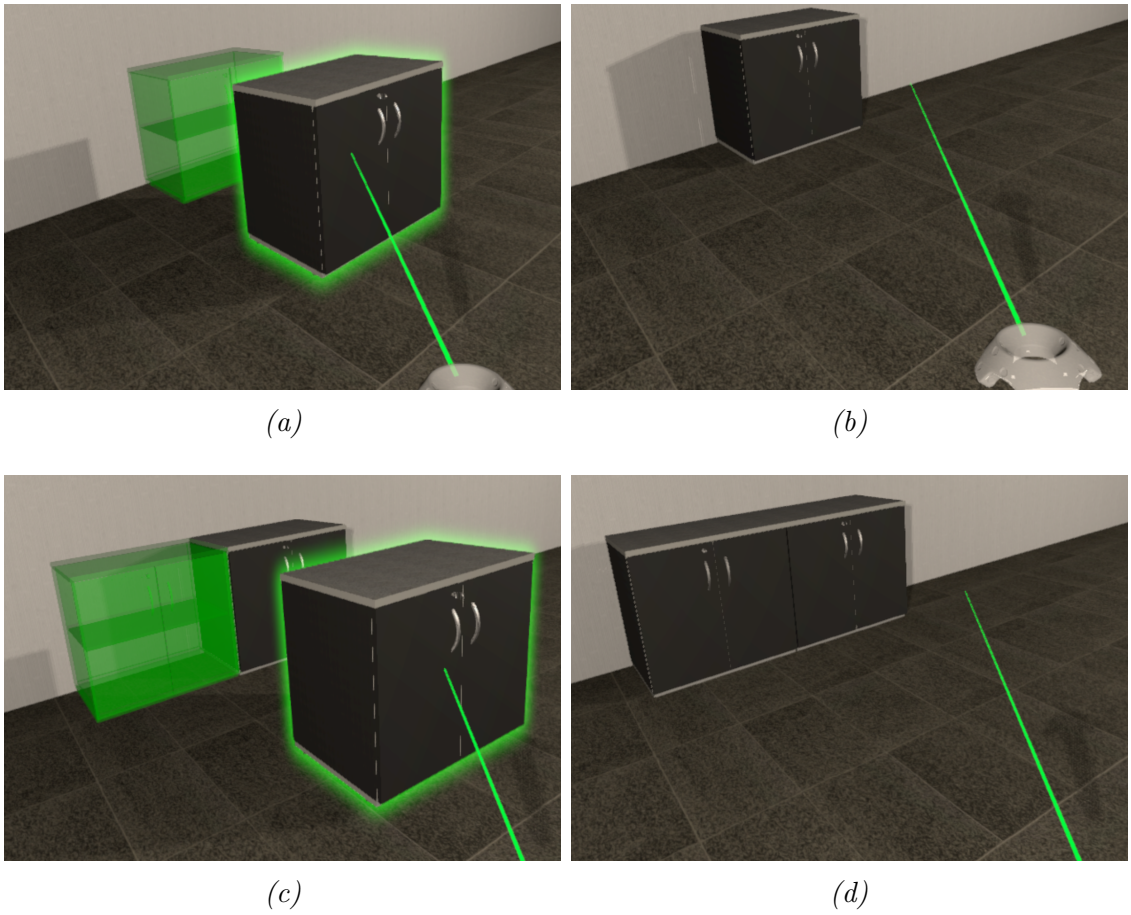
5.1.1 Design

The idea for the **Proximity** technique came from the IKEA kitchen planner (see Section 2.1.1), where the user can interact with objects on a 2D plane, and as objects come close to walls or other suitable faces, a transparent replica of the object appears

that will rotate and move to the closest suitable surface it can be placed at, indicating that if you release the object at that time, it will snap to that ghosts location and rotation. If you hold the object in place the transparent replica will disappear and nothing happens when the object is released. This is something we felt could help to improve some of the accuracy problems with our laser interaction, where objects are attached to the end of a laser pointer, as it can be hard to place objects accurately due to e.g. hand tremors and the slight hand movement that happens when you press the grip-button to release objects.

The interaction of this technique is displayed in Figure 5.1, where an example of placing two objects adjacent to one another is depicted. When an object nears a suitable surface to snap to, and there is space for it, a ghost model appears at the snap position. Currently, the appearance of this ghost model is instantaneous, however, it was meant to be animated. By animated, we mean the ghost model would spawn at the objects current location instead and then move to the position the object will snap to when released, also rotating to the correct rotation in the process. By not having the ghost models animated, it meant that they could jump around, moving from one location to another instantaneously, which could be confusing and annoying to look at for the users; this would be prevented by animating it. However, by animating the ghost models it would also introduce a small delay before users can see where the item is going to snap to, as well as raise some interesting questions regarding when an animation should start, stop, or reset etc. when moving objects around as you e.g. would not want the animation to stop just because the used moved the object by a millimetre (which would happen a lot due to hand tremors). The technique is also designed to assist users in placing items as much as possible by checking nearby suitable placements whenever an immediate placement is invalid, like expressed in Figure 5.1 where an object automatically suggests snapping to the side of another object as the front-face of objects is unsuitable for snapping.

With the **Proximity** technique, objects do not only have the ability to snap to other objects, but also the wall and the floor. Currently, the system only supports objects that are meant to be placed up against the wall and along the floor (we call them floor-objects). However, the system was designed with multiple type of objects in mind in order to specialise the snapping for each type of object, something that makes this technique less flexible as each object requires its own implementation. Other types of objects could include e.g. corner objects, wall objects, and kitchen-island objects which would each have different parameters to how and where they can snap. A wall object would for example not snap to the floor as floor objects do, but instead likely snap to a specific height on the wall that would be controlled by the user. Common for all objects is that nothing can snap to the front-facing surface of the objects, a design that is specific for the kitchen scenario as cupboard etc. would need to be able to open. Ideally, the system should also check whether or not the placement of an object would cause another object to be unable to be opened (e.g. a drawer), meaning objects' bounding box should be expanded in the front-facing direction, something that is currently not the case.



(a) *(b)*

(c) *(d)*

Figure 5.1. Figures showing the interaction with objects when using the **Proximity** technique. *(a)* The user places the object near a wall and the ghost appears. *(b)* The user releases the object and it snaps to the ghosts location. *(c)* A new object is placed in front of a placed object, showing how the ghost will suggest placing the gripped object next to it. *(d)* The new object is placed next to the old one when it is released.

5.1.2 Implementation

As mentioned the technique is named by its functionality as objects check their proximity for places to snap to. Here we take a look at how the objects do that, and what is considered to be in proximity of the item.

Since we wanted a functional system we decided to create a simple, but naive, method of calculating where objects should be placed. This was implemented by calculating the outer bounds for the cuboid objects in all four cardinal directions, and then doing both ray- and box-casts two unity units in these directions from the edges of the object. The reason for using both normal raycasts as well as boxcasts is that if, for example, a wall is hit by a boxcast, it might not hit directly at the centre point, which would cause the ghost to be placed incorrectly. The normal raycast is guaranteed to hit exactly in the middle since it is cast from the midpoint of the object. Meanwhile in the case of placing an object in front of another object with the intent of having the ghost appear next to the desired object, a regular raycast would mean the user would have to have at least the centre of the held object overlap with the edge of the object one wishes to place it next to. This is solved using the boxcast since it covers the full width of the object. Once all the casts have been done, the one with the shortest distance is selected as the starting point of calculating where the ghost should be placed. It is important to note that this could be made much more robust by using a scoring system so that distance was not the only factor, but also things such as what the ray hit.

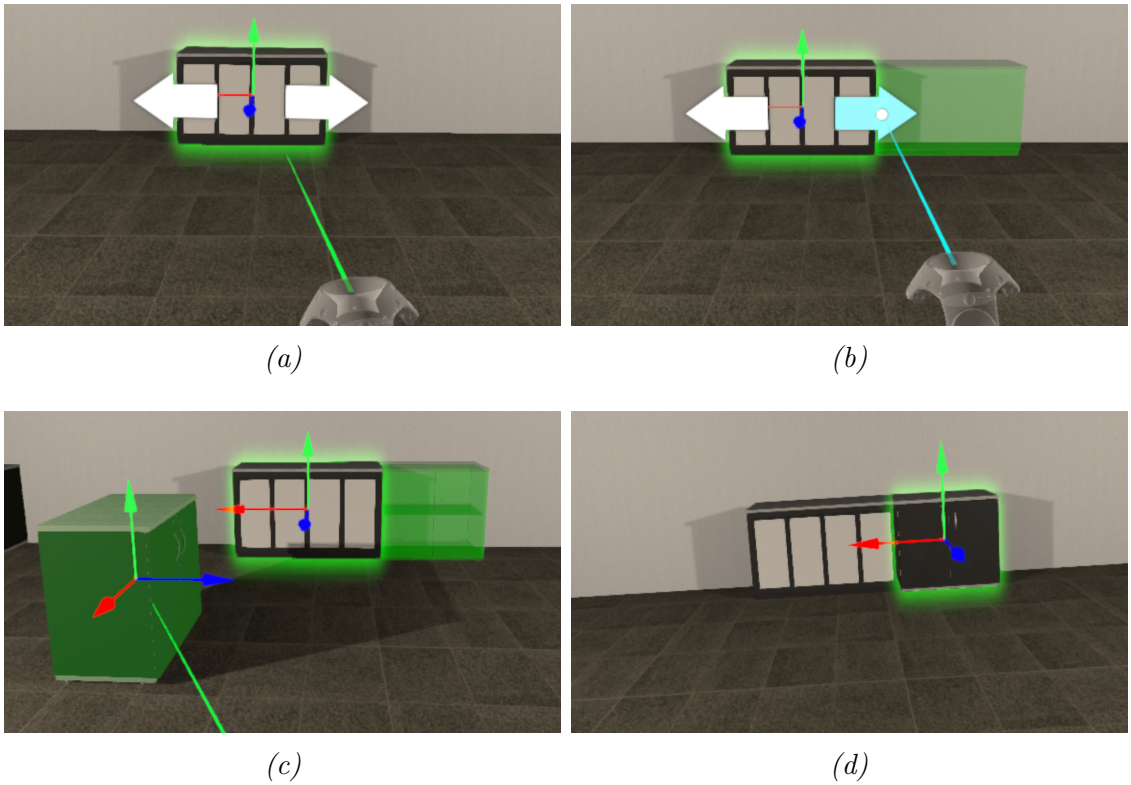
Once the shortest ray is determined, we check what has been hit. If a wall has been hit then the ghost is placed with its edge up against the wall, and then moved down to floor height. If an object is hit several checks are in place to find out which side of the object was hit, allowing different things to happen. If the front is hit, then we calculate which side (left or right) of the front was hit, if it is hit on the left side of the centre, the ghost is position to the left, with its forward direction being the same as the object and vice versa for the right side. If the back is hit the ghost is position with its forward being inverse of the object, and place at the centre, and made so the bottom of both the ghost and the object align. If sides are hit then we simple place the ghost on the side of the object, with the forward direction of the ghost being the same as the object. Currently rays are never cast downwards or upwards, meaning we cannot calculate cases where the top or bottom of an object is hit.

5.2 Continuous Technique

The **Continuous** technique was named so due to its ability to put rows of objects together in a fast continuous manner. It was meant to feel as one continuous process from placing the first object to the last.

5.2.1 Design

The idea for the **Continuous** technique came from considering what could be useful in a kitchen design scenario, and not so much considering only cuboid; although the **Continuous** technique could probably find a purpose in multiple scenarios. Considering the kitchen design scenario, most kitchens are basically just a straight line of objects along a wall, or a couple of walls, aligned next to one another. With that in mind we wanted to make a technique that made it possible to build rows of objects as simple and fast as possible. Most kitchens occupy at least two walls, sometimes with a kitchen island as well, meaning there is at least one corner in the kitchen. This is something the technique should take into account by automatically adjusting for corners by building along the next wall whenever there is no more space for items along the first wall and adjusting the already placed items to align with the corner however desired by the user. However, the implemented technique did not have this functionality and as such users had to manually start each row separately. The first item in the row is placed manually by the user using the system's laser interaction or proximity interaction and then the row is built as depicted by Figure 5.2. Whilst



(a) *(b)*

(c) *(d)*

Figure 5.2. Figures showing the interaction with objects when using the **Continuous** technique. *(a)* The user selects an object and is presented with directions to build in. *(b)* A building direction is selected by clicking an arrow. *(c)* The user hovers the laser over a new object, showing a ghost where it will be placed. *(d)* Clicking the new object places it at the ghosts location, and a new object can now be selected continuing the row in the same direction until a deselect occurs.

this technique is able to handle both floor object and wall objects, it would not be very purposeful for kitchen islands, as most kitchens islands consist of one big item or few items placed next to one another, so not much row structure to be found. As for wall items, the are made possible as the beginning of the row is based on the positioning of the first item, which in this case is not snapped to any specific surface. And this technique does not align objects against the wall or the floor as such, but rather the object it is being placed next to, by aligning both the backwards-facing surface and the bottom-facing surface with one another.

5.2.2 Implementation

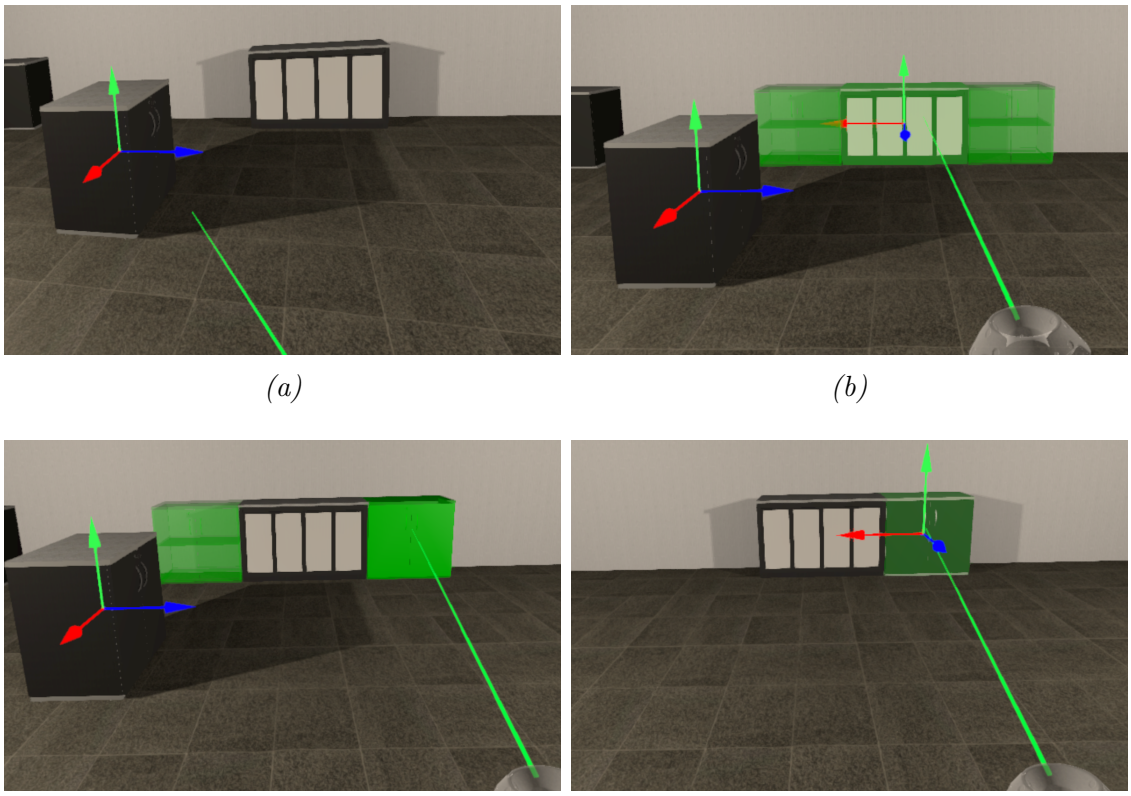
This technique used the basis of the **Proximity** techniques system for placing ghosts, namely the use of bounding boxes and directions. However the main different in implementation here was that the ghost had to be placed when the user hovered over buttons in an interface. This interface was implemented as a simple canvas item in Unity, and was always placed on the front of the object. Once an arrow is clicked the direction is locked, and once an object is clicked it will be placed to that side. Once the object is placed it is automatically selected, this allows the user to keep building without selecting the object that was just placed.

5.3 Preview Technique

The name of this technique is inspired by its functionality of showing users multiple options for placement prior to placing the item; multiple previews.

5.3.1 Design

The idea for the **Preview** technique came from a desire to be able to place items without needing to move them around much, so that items could be quickly relocated, but also from a desire to have multiple options when placing an item. The design of this technique went through a few iterations as the original idea was to have ghost models displayed (previews) at every possible location at item could be placed, however, it quickly became clear that this would leave the virtual environment cluttered when more than a few possible locations were available. This idea morphed into an idea of only displaying possible locations near the cursor position (i.e. where users are pointing), but this could potentially lead to overlapping ghost models, or at least confusion as to which preview to display. In the end we decided to keep the preview to one object at a time based on what the user points at, as depicted in Figure 5.3 where an interaction example is given. With this technique, the first item currently has to be manually placed by the user, using either the system's laser interaction or proximity interaction. An idea during the design was to make it possible to simply select an item and point to a place on the wall or on the floor to place the item there. This could potentially cause some problems as that is currently how items are deselected in the system and as such this is probably something that would have to be implemented through the use of a context based menu. While the



(a) *(b)*

(c) *(d)*

Figure 5.3. Figures showing the interaction with objects when using the **Preview** technique. *(a)* The users selects an object. *(b)* The laser is pointed at a new object showing possible placements for the selected objects. *(c)* Pointing the laser at one of the suggested ghosts. *(d)* Clicking the ghost places the selected item at the location.

design of the **Proximity** technique and the **Continuous** technique does not handle placement and alignment of kitchen islands very well, this technique inherently supports this, as well as wall object, through its use of interfaces as described below.

5.3.2 Implementation

The main difference in the implementation of this technique was the creation of a form of interfaces. Each object using this technique can have a number of interfaces defined which tell other objects which directions ghosts can be placed in, and which direction they should face when placed there. So when a user selects an object, and hovers over another object, several ghosts are created based on the defined interfaces, the interface can then be disabled once an item has been placed there. Additionally, when the ghosts are placed, they check if anything is blocking them, and in that case they turn red and cannot be clicked.
Whilst we were not able to implement all of our ideas for all the techniques, three different, working, techniques were nonetheless implemented with their core functionality in place. With these complete, a test was designed in order to figure out the strengths and weaknesses of each technique, this process will be explained in Chapter 6

CHAPTER 6

EXPERIMENT DESIGN

In this chapter the design of an experiment is described. First, the purpose of the experiment is discussed followed by the hypotheses being tested and a description of the participants of the experiment. Then, the procedure of the experiment is detailed followed a discussion of the expectations we had for the experiment in terms of participation and results. Information information regarding the equipment and setup used for the experiment can be found in Section 4.2. Worth noting is that multiple locations for used for this experiment, as opposed to the preliminary test, however, the setup and stimuli remained the same.
This experiment used a within-subjects (repeated measures) design as all participants tried all the experiment conditions, which in this case was the three different techniques, and they were all asked the same questions. This design was chosen as it requires fewer participants, which is something that would otherwise be a struggle based on previous experience with gathering participants. At the same time, by having every participant try every condition it limits the influence of individual differences might have on the results of the experiment. Furthermore, it saves a lot of time both due to less participants needed, meaning fewer tests, but also in terms of the planning and scheduling needed to properly conduct a between-subjects design experiment. There are also drawbacks of using a within-subjects design as each participant will have to spend longer for each test which can potentially introduce some fatigue. Carryover effects are also a concern with within-subjects design which occurs when testing one condition has an unwanted influence on the other condition tested afterwards. To counterbalance this effect the test order of the conditions is randomised as described further below.

6.1 Purpose

The purpose of the experiment was to find out which interaction technique provided the best interaction experience for the users. For this purpose three interaction techniques were developed: **Proximity technique (A)**, **Preview technique (B)**, and **Continuous technique (C)** as described in chapter X. Furthermore, the experi-

ment was meant to identify strengths and weaknesses through summative evaluation of the techniques, besides the already known shortcomings, as well as which method users found most easy to just pick up and use i.e. intuitiveness. The data logged in this experiment was:

- The **completion time** for each task. This was logged manually on a mobile device and the completion time started when a participant manipulated the first item.
- The **frustration rate**. The frustration rate was influenced by how many times users pressed a frustration button during each technique. This was a button on the controller which the user could press whenever they experienced a mismatch between their expected outcome of performing an action and the actual outcome.
- **Subjective user feedback** based on any comments users might have had while using the application as well as from the post experiment short interviews.
- **Observations** we made during the experiment.

6.2 Testing Order

The order in which the techniques were tested was pseudo random in order to remove any bias the order might have caused as inexperienced users were likely to gradually improve their performance after each technique. To limit the influence of the increased experience carried over from each test, each participant was first subjected to a **practice sequence (P)**. In this practice sequence users would some of the interactions that are common for every technique developed in this project, such as selecting objects, moving objects by grabbing them, and the trackpad interactions. As such, there are six different orders for this test as displayed in Table 6.1.

6.3 Hypothesis

Interaction technique **(A)**, **(B)**, and **(C)** have a difference in time for completion of task Y.
Interaction technique **(A)**, **(B)**, and **(C)** cause a difference in amount frustration for completion of task Y

PABC	**PACB**
PBCA	**PBAC**
PCAB	**PCBA**

Table 6.1. The different orders used for the test

The independent variable of the experiment is the interaction technique used i.e. **(A)**, **(B)** and **(C)**. The dependant variables are the task completion time and the task frustration rate. There are some confounding factors that could influence the dependant variables, such as:

- Having to repeat instructions.
- The test being conducted in different places on different dates, however, the influence of this factor should be minimal due to the nature of VR, where they are immersed in a virtual world.
- The time of day the experiment was conducted (users can be more exhausted at the end of a day).
- Retesting bias as some participants might have tried something similar before.
- Experimenter bias, as people are very rarely perfect, and some participants might get a piece of information others did not.

It is hard to account for all of these confounding factors and managing them in the experiment design, however, by acknowledging that they are a possibility one can notice when they might occur.

6.4 Participants

The participants for this test were a mix of current and previous students from Aalborg University as well as people from the offices adjacent to the one the experiment was conducted in. Some of them had knowledge of the system but no experience using it which was fine for the purpose of this test, while others had neither knowledge or experience. Their ages varied from 23 to 61 with the average age being around 35 years old. To get enough data, the goal was to gather at least 18 participants, and if possible only, make increments of 6 participants, i.e. 19 would not be great due to an uneven distribution of the order, so instead we would need 24, 30, 36 etc. The data gathered was all submitted anonymously.

6.5 Procedure

Each participant in the experiment was submitted the the following procedure:

1. Participants were given a brief introduction to the project as well as an explanation of what they were about to test, as described in full detail in the test script in Appendix B

2. A brief background/experience questionnaire was filled out by the participant, displayed in Appendix C, regarding their prior experience with computers, video games, and 3D in general.

3. Participants were given instructions to the system in preparation for their practice sequence prior to trying the different techniques. When the instructions were given users were equipped with the HTC Vive and the two controllers, and then placed inside the virtual environment.

4. After the practice session, the first interaction technique the participants would be trying was introduced, and a small tutorial was given. Inside the virtual environment the users were given a task to replicate two separate kitchen designs, displayed in Figure 4.2, with the objects that were available to them in the environment, using one of the three interaction techniques.

5. Once the task was completed with one interaction technique, a short questionnaire was filled out, see Appendix D, regarding the experience the participants just had using that specific interaction technique.

6. Steps 4 and 5 were then repeated for the remaining two interaction techniques, and in the end the participants filled out an extra questionnaire regarding their overall experience and state of mind, as well as a short follow-up interview/discussion about their experience, see Appendix E.

During the experiment the participants were observed to look for any mishaps that might happen in the application as well as any noticeable difficulties the users might have had. Whenever users seem to be either off track or forgetting the controls for a given technique, the facilitator would step in to provided the needed assistance - like a clerk would at a shop.

6.6 Expectations

Through this experiment we expected to find the following:

- Despite the practice period, experience tells us that the participants were likely to get better as the experiment progressed, meaning the first technique they tried would probably have the highest **completion time**; and possibly also a higher **frustration rate**.

- With this implementation and experiment design we expected that the interaction technique that would provide the most positive feedback would be the **preview item** technique. However, we were also aware that the fastest technique, if utilised properly, was likely to be the **continuous item** technique.

- Though we had a small simulator sickness questionnaire, we did not expect anyone to experience any kind of simulator sickness from using this application, as previous experience has shown that, even people who have not tried VR before has had no issues with previous versions of the application.

- We expected that the results would show a quite clear correlation between how much participants have spent playing computer/video games and the completion time for the interaction techniques and the frustration level as well. A part of playing video games is the ability to learn new controls in order to play a new game, not to mention that they are likely to be more familiar with controllers.

CHAPTER 7

RESULTS & DISCUSSION

In this chapter, the results of the experiment, which was described in the previous chapter, are discussed. The results are presented in the following order: the quantitative results, such as completion time and scale responses; thereafter the qualitative results, such as questionnaire responses and our observations during testing. Whilst a lot of data was collected via questionnaires in order to be thorough, we choose to only present the subset of data that we found most interesting and relevant. The quantitative data are evaluated using tables and figures, as well as statistical tests. The qualitative data, in the form of questionnaire responses and observations from the testing, are presented and then discussed in order to see if any knowledge can be gained. The test was conducted on 18 people in total, over a period of 4 days at various locations.

7.1 Quantitative Results

In this section, the quantitative part of the data is analysed. These data are comprised of: the completion time, answers to various questions on a discrete visual analogue scale from 1 to 5, and background information, such as how many hours participants spend using a computer or playing games as well as their age.

7.1.1 Completion Time

Figure 7.1 shows how long participants took to finish building the two kitchens for each technique. From visual inspection the **Preview** technique seems to be the fastest technique, having the lowest median value, it is also skewed right, suggesting that the data is not normally distributed; this is determined by looking at the size of the boxes on each side of the median as well as looking at skewness values. The fact that it is skewed right suggests that the 50% of participants who had low times were closer together, or less varied, than the participants in the upper 50%. The opposite is true for the **Continuous** technique, which has the highest median value, and is instead skewed left. Meanwhile, the **Proximity** technique is not particularly

skewed, with the median being quite close to the mean, within around 5 seconds; this suggests that the times are potentially more normally distributed than for the other techniques.

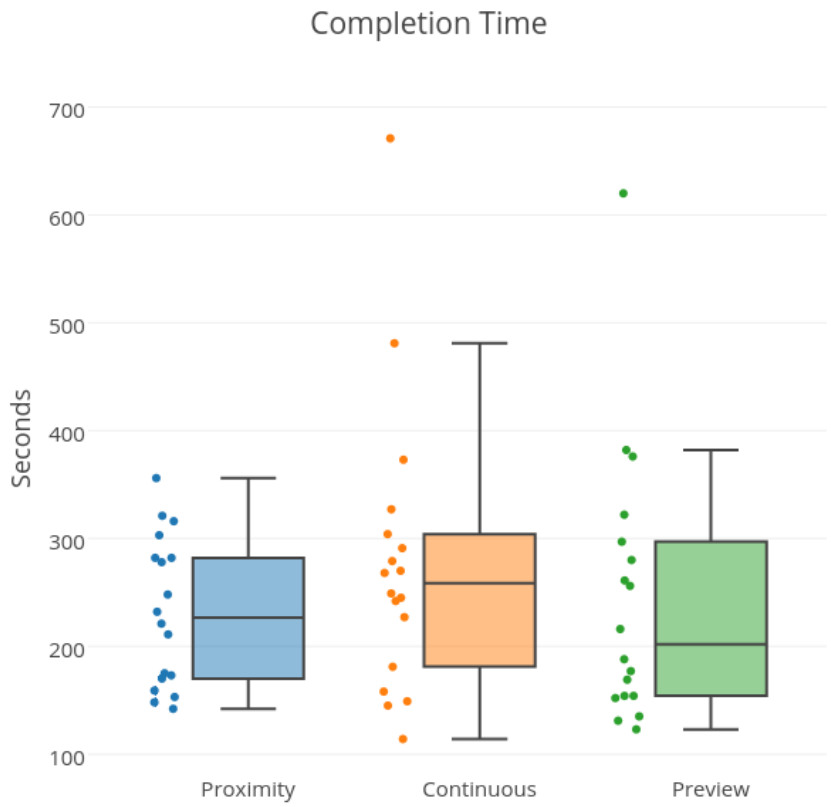


Figure 7.1. Box plot of the completion time for each technique.

Based on the box plots one might be inclined to also say that the **Continuous** technique has more variation than the other techniques, but looking at the data points this is mainly due to two larger than usual values, whilst the rest of the data looks similarly varied to the other two techniques. A thing to note however is how the **Proximity** technique features no outliers, and since our data is paired this means that some test participants, who struggled with the **Continuous** and **Preview** technique, did just fine with the **Proximity** technique. It is hard to conclude anything from this both due to low sample size, but also due to the impact of which order participants used the techniques; something which will be discussed in the next section. However, whilst it might be hard to conclude something, we suspect this might be due to the **Proximity** technique being the easiest to pick up and use, on the basis that it requires the least amount of memorisation of buttons out of all the techniques.

One of our expectations for the test, mentioned in Chapter 6, was that the **Preview** technique would have the best user feedback, and the **Continuous** technique would have the fastest time if participants utilised it properly. The feedback of the

Preview technique will be discussed in the next section, but for now, we can see that it seems to have the lowest median time, whilst the one we expected to be the fastest has the highest median with the **Proximity** in between. In order to see if there was a statistically significant difference between the times, we decided to run a repeated measures one-way ANOVA test for correlated samples [20]. In our case, the dependent variable is the completion time for each test, and the independent variable is the technique. The assumptions for the ANOVA test with correlated samples are:

- That the measures within each group are independent.
- That the population from which the samples are drawn be assumed to be normally distributed.
- That the groups have roughly equal variances.

It should, however, be noted that the test should be quite robust in regard to the last two assumptions, normality and variance. This robustness is especially present in the case where each group is the same size, which for a correlated test, they must be. Taking this into account, we can say that our data satisfies the criteria of being independent, however, we cannot say with certainty that the population is normally distributed, as only our **Proximity** technique data hints at this. However, this might be due to the low sample size, and it could be assumed that as the sample size increases our results would be more normalised. Finally, the variance between our groups is high when comparing the **Proximity** technique to the other techniques, however, the **Continuous** and **Preview** techniques have similar variance. With all this in mind we still decide to use the ANOVA test, however since we are aware that we might not completely satisfy the assumptions, we choose an alpha value at $\alpha = 0.01$. In this case the null hypothesis is:

H_0 : *the means of all groups are equal.*

And the alternative hypothesis is:

H_A : *at least two means are significantly different.*

After having run the test we end up with a p-value of: $p = 0.347$. This value is above our chosen α value and thus we cannot reject the null hypothesis, meaning that, statistically, the means of all groups equal. Whilst we cannot statically prove it, for our data set, the fastest method seems to be the **Preview** technique, and the slowest the **Continuous** technique when looking at the median, however, the **Proximity** technique is faster when looking at the mean, in part due to lack of outliers. This seems to suggest that the **Proximity** technique is the more consistent technique, whereas the **Preview** technique might be faster for users who can figure out the button scheme to utilise it properly, as this seemed to be the biggest problem for users in regards to the technique; namely confusing the actions of different buttons. We also meant to analyse the mentioned "frustration" rate, gathered by users clicking a button if they felt frustrated using the same statistical test, however, many users forgot to use the button so this idea was discarded.

Order influence

As mentioned in the last section, we expected that the order in which the test participants used the techniques would influence their completion time, a reason for why we chose to randomise the test. But despite that, we thought it might be interesting to investigate if this expectation held true. In order to investigate this, Figure 7.2 was created; this figure shows the completion time for participants for each technique they tried, with the lowest value being indicated with green, middle value with yellow, and worst time with red. Figure 7.2(a) shows the times ordered by technique, and it is hard to see any immediate pattern, but with some close examination one appears. When comparing the group that took the experiment in the *"ABC"* order, with the group that did it in the *"CBA"* order, one can note that the first group performs best in the **Preview(C)** technique, worst in the **Proximity(A)** and somewhere in between with the **Continuous(B)** technique, whereas the "CBA" group is an almost exact mirror. This seems to suggest that our expectation was correct, participants kept learning well beyond the practice session they completed before the actual techniques.

Order	Interaction technique A	B	C
ABC	321	245	131
	282	181	154
	356	304	169
ACB	148	227	123
	278	291	216
	248	279	280
BCA	142	145	204
	232	373	256
	170	481	297
BAC	282	114	188
	303	327	152
	175	268	135
CAB	221	242	177
	153	158	382
	173	149	261
CBA	159	270	376
	211	249	322
	316	671	620

(a)

Order	Interaction technique *(ordered by sequence)*		
ABC	321	245	131
	282	181	154
	356	304	169
ACB	148	123	227
	278	216	291
	248	280	279
BCA	145	204	142
	373	256	232
	481	297	170
BAC	114	282	188
	327	303	152
	268	175	135
CAB	177	221	242
	382	153	158
	261	173	149
CBA	376	270	159
	322	249	211
	620	671	316

(b)

Figure 7.2. This figure shows the completion time for each participant per technique, with vertical groups showing the order in which participants, in that group, tried the different techniques. **(a)** shows the data ordered by technique, whereas **(b)** shows the data ordered by the sequence in which test participants tried them.

Ordering the data by the sequence in which participants tried the technique presents

an even better understanding, as illustrated in Figure 7.2(b). The thing to take note of here is that the first column contains more red, the middle more yellow, and the final column more green. This is a strong indicator that test participants, in most cases, were faster with the final technique they tried, with one notable exception: groups "ACB" and "CAB", both being the only groups where the last technique is the slowest, in this case, the **Continuous** technique. This suggests that, even with practice, participants did not do well with this technique. Granted, the sample size is not huge, but the fact that it appears in both, and not just one, of the two groups that tried the **Continuous** technique last, suggests that it might be a trend. All this seems to suggest it was indeed a good idea to randomise the test in order to get an average of how well participants performed with each technique despite the learning bias. One could also argue that a better approach might have been to not use a repeated measures design; participants could instead only use one technique but have more time to learn how to use it. This approach would give a better idea of the true mean of the completion time of each technique with potentially less varying data, it would, however, require a much larger number of participants; something which was not feasible for this project given time constraints.

7.1.2 User Rankings

In order to get a better understanding of the subjective opinion, as opposed to completion time, that test participants held towards techniques, a series of statements were presented after each technique had been tried, and participants asked to score how much they agreed with the statement on a scale from 1-5 where 1 is "Strongly Disagree", and 5 is "Strongly Agree". This section will cover a subset of these statements found to be most relevant. Some statements not covered here were found to be biased by other factors, such as a statement regarding if they enjoyed the experience; participants often mentioned they enjoyed it due to it "being VR", and not specifically due to the method. This could potentially have been solved by wording the statements better, something which was done for other statements, ensuring they were referring specifically to the interaction technique. Another issue was the chosen range of 1-5 for the replies, some participants voiced their disappointment that they had given the first method they tried a 5, when they liked a later method more, this might have been less of an issue with more values to choose from. Another thing that could be improved in this regard is to consider adding inverse questions to pair with the positive questions e.g. "I found the technique easy to learn" and "I found it difficult to understand the technique" as a simple example; this would allow us to verify answers by checking the replies to both.

Statistical tests were run on the different rankings, specifically the same ANOVA test mentioned in Section 7.1.1, however, no statistically significant difference could be found between the means, thus all discussion is based on visual inspection of the figures.

Ease of learning

The first statement participants were asked to give their opinion on was: "I found the interaction technique easy to learn.", the answers for which can be seen in Figure 7.3. Here we expected that the **Proximity** technique would receive the highest score, due to the fact that it has fewer buttons to worry about when learning how to use the technique. Another factor we suspect might increase the ease of learning for the **Proximity** technique is that the concept is comparatively simpler, one picks up items and places them. This is in contrast to the **Continuous** and **Preview** techniques where a user is forced into a more specific pattern they must follow in order to place objects, something which can either be a benefit if used properly or a disadvantage if not used properly or not fully understood. This seems to nicely correlate with the completion times, as the **Proximity** technique had less varying times, suggesting that the technique is a nice middle ground between speed and ease of learning.

Figure 7.3. Box plot showing test participants questionnaire responses regarding how easy they found the technique to learn.

Looking at the numbers for the **Continuous** and **Preview** techniques it is evident that they are almost identical, except strangely the **Preview** technique has more varying scores, yet it has better completion times. It is important to note that this may be due to outliers, but nevertheless, it might also mean that participants find the technique slightly harder to learn than the **Continuous** one, but once they do it is actually faster. Interestingly, whilst we expected the **Continuous** item to be the fastest, we at the same time expected the **Preview** technique to garner the most positive feedback, which in this case it does not. We can find no reason for this, as the two techniques should be roughly similarly in terms of complexity.

Ease of use

The second statement presented to participants was: "Once I learned the interaction technique, I found the object manipulation easy and intuitive to use.", Figure 7.4 shows the replies. Once again the **Prixmity** technique seems to be the most consistent performer, with all participants ranking it either 4 or 5 in terms of ease of use, with a 4.5 median value, albeit with one outlier at a score of 2. The **Continuous** and **Preview** techniques look quite similar, but one thing to note is that the median is at 4 for the **Continuous** and 5 for the **preview** technique, suggesting that participants found the preview technique the easier one to use; this is quite odd looking back at Figure 7.3, where the **Preview** technique scored the lowest in terms of how easy it was to learn. This difference seems to imply that while participants found the technique the hardest to learn, it was the easiest to use properly once learned. This makes sense when one considers that the **Preview** technique does a lot of the work for the user, in that it automatically shows where the object can be placed, in the most likely locations a user would want to place the object; at the cost of having to learn how to use the technique. Comparatively, the **Proximity** technique can require a lot more precision in its current implementation, helping to explain why the **Preview** technique seems to fare better.

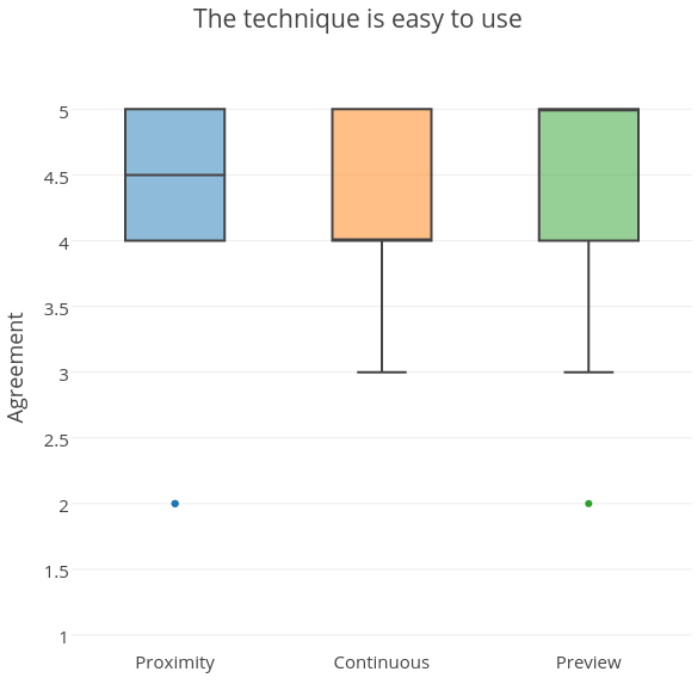


Figure 7.4. Boxplot showing test participants questionnaire responses regarding how easy they found the technique to use. (Note that the median line overlaps with the first and third quartile lines in the case of the Continuous and Preview technique, respectively).

The worst scoring of the three is, as mentioned, the **Continuous** technique, which goes contrary to our expectations. Ideally, based on our expectations and how few actions it requires, the technique should be very easy to use as it takes even more

work out of the user's hands, in that one simply selects and object, a direction and then click the items you want to place. However some confusion as to what item was selected, and how to stop building caused issues for some users. This might help explain why it was generally found harder to use, but it is possible that solving these problems could increase the ease of use of the technique; this might also improve for other techniques as well, however, since it, to some extent, was a problem for all techniques that users were unclear as to what was selected and how to deselect something. We have discussed ways to solve this, like having a head tracked GUI that would point an arrow to the selected object, or show a ghost of the object so that it is always clear something is selected, even if you are not looking in the direction of it.

Ease of aligning objects

The third statement participants were asked about their opinion on was: "Once I learned the interaction technique, I felt it was easy to properly align objects with each other.", the answers can be seen in Figure 7.5. Both the **Proximity** and **Preview** received a median ranking of 5, with the **Proximity** having slightly more varying rankings. Meanwhile, the **Continuous** technique has a median of only 4.5, with the rankings varying as much as the ones for the **Proximity** technique. These results go against our expectations since the **Continuous** technique should be the fastest and easiest technique for aligning the items, due to the user merely having to click items and they get placed in a line (objectively less actions), potentially halving the amount of actions required to place each item. These results might be influenced by the fact that users found the technique harder to use and more frustrating due to previously mentioned reasons, and thus gave it lower score for alignment as well. Another interesting thing to note is how the **Preview** technique scores extremely well in this measure, even more so considering users found it the hardest to learn. This seems to support the reasoning in the last section; that whilst the **Preview** technique is harder to learn, it is more rewarding to do so than for the **Continuous technique**. The only technique to feature outliers for this question is **Proximity**, this is most likely due to the fact that, at times, the technique can be quite finicky to use depending on how accurate a user wants to be in placing the objects. The reason for this is not due to user error, but more an incomplete implementation of the technique. Since placing objects next to each other can be done in two different ways: 1. Moving an object in front of another object will place the ghost at the side closest to where you are holding the object and 2. Moving an object to the side of another object and, hopefully, placing the ghost where the user wants it. The problem with the first method is that some users forget it, and preferred to move the object to the side instead of in front, which might feel more natural since this is the intended final location. The problem with this is that the implementation of the snapping prefers whichever surface is closest, so if a user pushes the object into the wall, it will prefer this instead of the object right next to it, where the user actually wants it to be placed. This is likely what causes the higher variation in rankings for the **Proximity** technique. Contrary to this, the **Preview** technique

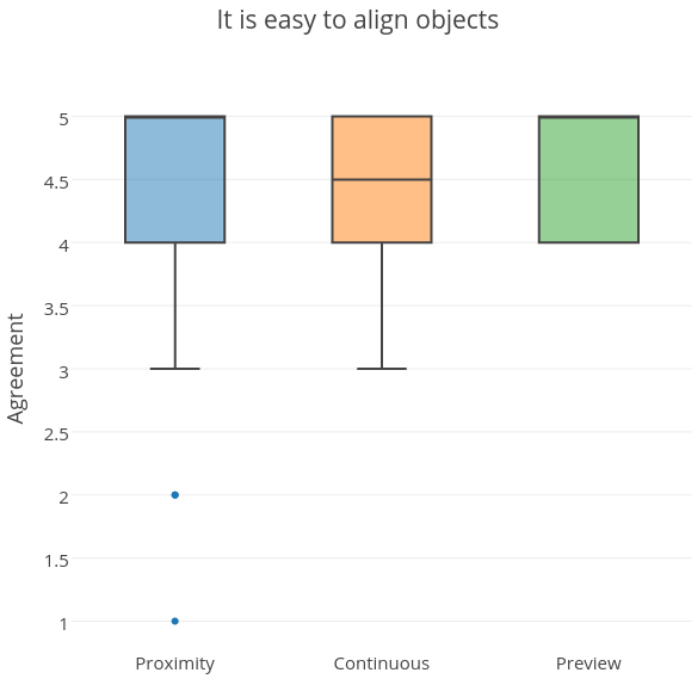


Figure 7.5. Box plot showing test participants questionnaire responses regarding how easy they found it to align items using the technique.

had few problems in this regard, and worked quite well, allowing users to easily be shown where they can place an object, as well as placing it perfectly aligned simply by clicking the ghost.

Preferred technique

After participants had completed all three techniques, they were asked this question: "Which of the three techniques you just tried would you prefer? (If you had to choose only one)", the results are presented in Figure 7.6. Here we can see that the technique most users prefer, is the **Preview** technique, followed by the **Proximity** technique, and lastly **Continuous** technique. We definitely expected the **Preview** technique to gather the most positive feedback, and between rankings in the previous sections and 44.4% of users picking it as their favourite, it is quite clear that this is indeed the case, although the **Proximity** technique is a close second.

7.1.3 Data correlation

Another assumption we had in regards to the test was that certain factors would impact the completion times of users, such as their age, how much they use computers, both playing games or using CAD software as examples. Our expectation is based on the assumption that users who are younger or play more games will be more used to learning new control schemes for applications or be more technologically minded in general. Our experiments sample size is quite low meaning that correlating variables is quite hard, and any correlation might be pure coincidence, and definitely,

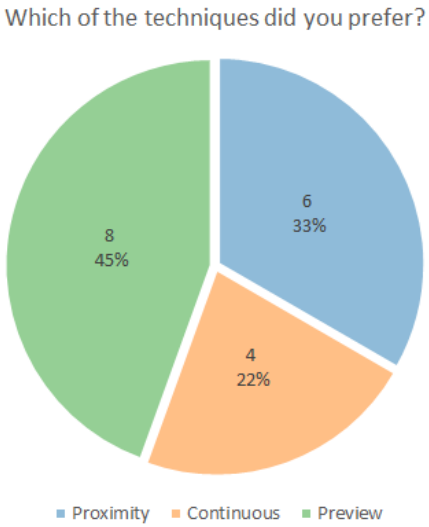


Figure 7.6. Pie chart depicting the results of asking test participants which method they preferred, if they had to pick one.

does not mean causation. With this in mind, Figure 7.7 shows completion times against participant age, with a linear trend line. We can see that there seems to be a positive correlation between the two variables, meaning that as age increases, so does completion time.

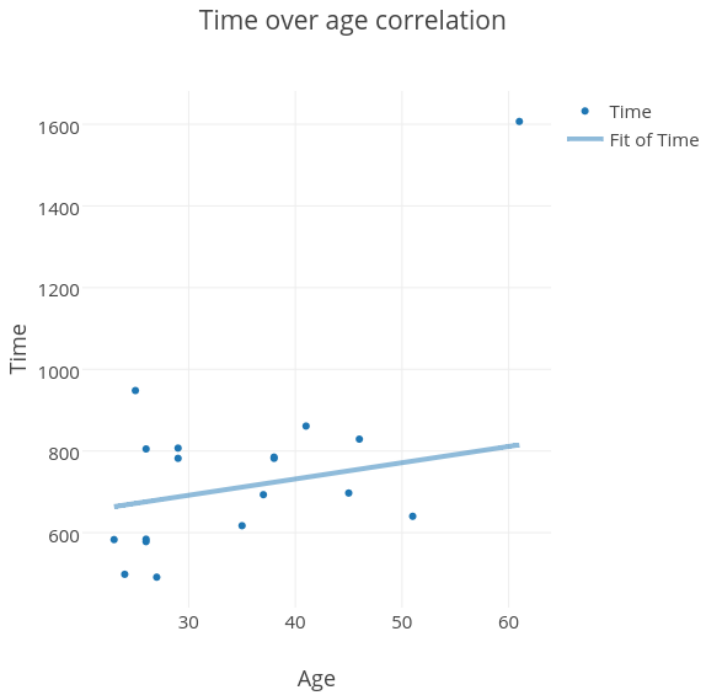


Figure 7.7. Scatter plot depicting the completion time over the age of the participant, as well as a trend line showing a potential slight positive correlation. (Outlier is disregarded by the trend line).

Time Correlations						
Test type	**Pearson**		**Spearman**		**Kendall**	
Value	r	P-Value	rho	P-Value	tau-b	P-Value
Age	0.6286	0.0052	0.4682	0.0501	0.3400	0.0523
PC hours	-0.4823	0.0427	-0.0531	0.8344	-0.0370	0.8532
Gaming hours	-0.4667	0.0509	-0.3328	0.1771	-0.2873	0.1133
3D Controller	-0.3659	0.1353	-0.3680	0.1330	-0.2963	0.1083

Table 7.1. Shows the correlation coefficient and p-values for correlations between time and: Age, hours spent using a PC per week, hours spent gaming per week, and how much experience users had with 3D controllers.

In order to see if there is any validity to this we decided to run three different measures for correlation, namely Pearson's r, Spearman's rho and Kendall's tau-b. These all return a coefficient between -1 and 1, with -1 being a strong negative correlation, 1 being a strong positive correlation, and 0 being no correlation. The purpose for running different tests is simply to be more confident in the results, and since the different tests are good at different things. The Pearson test is mainly for interval or ratio data, whereas the Spearman and Kendall tests also accept ordinal data. Aside from this, the Pearson test shows linear relationships, whilst the Spearman and Kendall support monotonic relationships. We choose an α-value of 0.05 for these tests. The tests produce a p-value for the alternative hypothesis:

H_A : *true correlation is not equal to 0*

The results of running the tests are shown in the first row of Table 7.1, and as predicted from looking at Figure 7.7, we do indeed see positive correlations in all measures for age, particularly in Pearson's test; this makes sense given that it primarily looks at linear relationships, which it what we expected. This means that there is a strong, positive correlation between completion time and participant age, which is statistically significant with a p-value of 0.0052. We do not get a significant result for either of the other tests, but they both have positive coefficients as well, with this in mind we are quite confident that there is a correlation.This could mean, among other things, that our techniques are not user-friendly enough for all age groups, which could be considered weakness.

Another assumption we had was that various other variables might correlate with the completion time, namely how many hours participants spend using a computer, how many hours they spend playing games and how much experience they had with non-standard computer input devices such as 3D mice. Looking at the second row of Table 7.1 one can see that values from running the three tests for correlation between completion time and how many hours the participants spent using a computer. Most participants picked the top two choices for how much time they spent using a computer, this makes it hard to correlate anything as we lack samples in the low end. However, as can be seen, mostly negative correlation coefficients are present, suggesting that more hours spent using a computer decreases completion

time; this was expected, however, due to the aforementioned problem of too few samples, we decide not to conclude anything despite getting a significant p-value using the Pearson test, as it is not designed for ordinal data. The third row of the table shows the correlation coefficients for completion time and how many hours participants played games each week. For this we also see negative correlation coefficients, however no significant results using either test, the same is true for the last row, showing the correlation between use of 3D controllers and completion time. Overall we see a trend that all of these things correlate with a reduction in completion time , however, due to low sample size, it is hard to conclude whether this is true, and especially if there is a direct causation.

7.2 Qualitative Results

This section focuses on the presentation and discussion of responses to a range of qualitative questions that participants were asked after the experiment, as well as observations we made during the experiment. The discussion will cover topics by order of relevance in terms of how many test participants commented on them, as well as how important we feel they are.

One of the most repeated positive comments regarding the system was that the immersion is very good. Participants particularly liked that they were able to move around and inspect things in three dimensions, and they felt like it gave them a better idea of the environment than, for example, a drawing would. This was definitely one of the things we expected to hear, and one of the main strong points for VR; one could even go so far as to say it is the main difference between VR and traditional monitors/interaction. However, as observed during testing, the main cost for this new immersion is that users are not used to interacting in this manner, most likely due to how ingrained mouse and keyboard interaction is in most users, not to mention how many years this interaction method has had to evolve. The main problem is the strongly reduced amount of buttons compared to a keyboard, and the lower accuracy of the input devices compared to a mouse; since the tracking of the Vive controllers is so precise it actually causes problems at times due to picking up a users hand shaking slightly, as mentioned by one user. This feeling of immersion is one of the reasons for doing a project like this, attempting to find some new way of interacting in the virtual world that lets users get this immersive experience, while also being able to create something in VR. In this regard, we received some positive feedback regarding our three interaction techniques, but also a lot of constructive criticism. The most mentioned techniques in the feedback, as well as discussions with participants during testing, was the **Preview** and **Proximity** techniques. Users really liked the freedom of the **Proximity** technique compared to the alternatives, noting that it would be preferable to combine it with the other techniques. This desire was driven by the fact that placing the first object, when using the **Continuous** or **Preview** technique, one needed to use unassisted placement, which users did not like; this in itself suggests that just using basic interaction with, no snapping, ghosts or some form of aid, is not a good idea for precise placement and alignment. Several users did, in fact, specially note that combining the **Proximity**

and **Preview** technique would provide a very nice combination of the strengths of both methods, namely ease of placing the first object, and easy of placing objects thereafter.

Overall, users liked the concept of snapping, suggesting that our choice of pattern was good. The only comment, in this regard, was one user who wished that instead of picking up the actual object, one would pick up a ghost/representation of the object, which could then be moved to locations to snap. This could be an interesting concept to explore due to how we handle collisions for objects in the current system. It can be hard to move objects when they are stuck between other objects, or are behind objects, forcing a user to move them around the object blocking the path. This could potentially be solved by implementing the suggested change, so a user instead picks up a ghost of the object, which can logically move through other objects, and if dropped inside another object it could simply disappear. If implemented as the system is now (where one picks up the actual object), then either the object would have to be stuck inside another object, which would be neither pleasant to look at nor logical, or it would have to move to some random location, or back to where it was before one picked it up; none of these solutions are very satisfying compared to using a ghost. Another related comment was the lack of an undo button, letting users retrace their steps. Using the aforementioned idea of picking up ghosts of objects instead of objects, this problem could be partially solved without dedicating a button to it, by simply letting to user drop the ghost. Alternatively, the problem could be solved by, for example, adding a 2D menu to the back of one of the controllers which would only be visible when facing towards oneself, this menu could then contain common actions like an undo function, which would work for all three interaction techniques.

One common complaint with the general interaction was how the laser behaved when trying to place items close to walls using the scrolling function on the trackpad. Users found that, depending on how they had picked up the object, it tended to push objects outwards from walls when trying to move it close to the wall. This is mainly an implementation problem due to how the system tries to correct for users attempting to scroll objects into walls. Since scrolling moves the endpoint of the laser, which objects follow, then, if a user scrolled the point too far behind a wall or the floor, then objects would exert more and more force trying to get to the point. This issue causes objects to, seemingly, get stuck to walls and floors. Whilst our current implementation solves this "stickiness" problem, it is overly aggressive in some cases. Solving this would significantly improve users opinion on the basic interaction; it is, however, not an easy thing to solve and we currently have no solution.

We also had comments from users regarding the physical controllers which, whilst not directly related to the interaction techniques, is still relevant since our implementation is directly dependent on the controllers available. Users tended to get very confused in regards to the grip buttons and the trigger, often confusing the two even after having been explained several times what each button did. For some

the problem was memory of which button did what, for others it was the physical location of the button that was the problem and it felt hard to use. Related to this is the confusion between "grabbing" objects, and "selecting" objects, the former of which the grip-buttons are used for, and the latter the trigger. One solution to this problem would be to alter the techniques so there is no distinction between the two actions. Alternatively double-clicking or time dependant buttons might be utilised; such as making a distinction between a long click and a short click. The problem with this kind of system is that a trained user might be annoyed with having to wait for actions to happen, if they know exactly what they want to do. For double clicking the problem might be that, as mentioned by users, the controllers can be too accurate and clicking buttons can move the laser, so as one tries to double click, one might end up missing the second click. There are several possible solutions, but as to which works the best, more experimentation would be required.

As stated in Section 6.6 we did not expect users to feel any simulator sickness from our system as we had taken measures to reduce this, such as blacking out the screen momentarily when teleporting. This fact combined with the excellent tracking and refresh rate of the display should severely limit any kind of simulator sickness symptoms. Despite this, we still decided to ask participants some questions regarding this. The large majority of participants reported that they felt none of the symptoms of simulator sickness and felt just as fine after the experiment, as they did before. A few participants noted eye strain and slight fatigue, but also said it was very minor. This is most likely not due to our system and instead due to using VR in general.

7.3 Summary

This section summarises the various results of the experiment, by providing an easy to read structure of what we consider the strengths and weaknesses of the various techniques. Whilst there may more more strengths and weaknesses, we find these to be the main ones.

Proximity technique

Strengths

- Easy to learn, requiring the use of only one button for placement.
- Very consistent in terms of completion times.
- Provides more freedom compared to other techniques.
- Well liked technique by most users, with few negative comments.

Weaknesses

- Can be confusing/annoying for users if the snapping is not implemented properly.
- Due to more freedom, the technique can require more fidgeting when placing items.

Continuous technique

- Objectively the fastest technique when building one long row, since only one click is required per item.

Weaknesses

- Requires manual placement of the first object.
- Slowest in our experiment.
- Not entirely mode-less, users got confused how to stop building the row.
- Users did not like it very much.
- Less flexible.

Preview technique

- Users found it easy to use.
- Potentially very fast if utilised properly.
- Preferred technique by most users, with very few negative comments.

Weaknesses

- Requires manual placement of the first object.
- Users found it hard to learn.

CHAPTER 8

CONCLUSION

In this project we set out to discover the merits of various interaction patterns and techniques in HMD-based VR, limited in scope by using a kitchen building scenario. This was done so that the project could focus on interaction with cuboids, and at the same time rotation was limited to cardinal directions (90 degree increments around the Y-axis, the upwards axis). It also gave users a relatable task when testing the system, as opposed to testing it on arbitrary tasks. In this report, three different patterns, with accompanying techniques, were designed, implemented and tested in order to determine which overall pattern was preferred by users. Afterwards, three different techniques were designed, implemented and an experiment was conducted on 18 participants in order to gather feedback. The experiment resulted in both quantitative and qualitative data to which both exploratory data analysis, as well as inferential statistics, were applied in order to discover which techniques performed well in certain measures, and which did not. Through this testing we discovered that the **Preview** technique, whilst being harder to learn, ends up being the one with best performance in terms of completion time, although statistical tests failed to prove this. Based on this we conclude that this technique lends itself better to systems where users have more time to get used to the controls. Meanwhile, we discovered that the **Proximity** technique is quite fast, that users find it easy to learn and use, and that it is quite flexible. A conclusion to this could be that it lends itself well to situations where users only need to use a system for a short while, and need to quickly learn how to use it. The least liked technique was the **Continuous** technique. Whilst this technique should objectively be the fastest due to the low amount of actions required to place each object, users were confused as to how to stop building and what they had selected. In conclusion we find that the technique most likely requires more work, and that some of negativity surround it is more due to lacking implementation, than the technique itself.

It is important to reiterate the fact that the patterns and techniques covered in this project are only a small portion of the options available in virtual environment with HMD-based VR. Furthermore, the patterns and techniques in this project are only first iteration, as multiple user tests would have to be conducted to perfect the

design and implementation of each pattern and technique. In the next section we take a look at some of the future work required to further discover the best patterns and techniques for VR as well as the improvements needed to better the currently implemented patterns and techniques.

8.1 Future Work

Whilst we feel that the implemented techniques are highly functional, testing did reveal many flaws and possible improvements. Considering this, several more design cycles would be needed to fully realise the potential of the various techniques mentioned in this report. On a related note, it would be interesting to implemented more techniques and compare them to the results of this report to see if we have found the "best" solution to interaction in this scenario with this pattern (**Snapping-based Pattern**). Furthermore, more patterns could be explored and the patterns explored in this project (**Grid-Based Pattern** and **Free-hand Pattern**) need to be further explored as well, and the techniques for these patterns need to be improved and developed further. Another interesting prospect would be to take a look at a World-in-Miniature implementation, using the already developed proximity interaction, or an improvement hereof, to see how WIM could influence some of the problems faced when doing interaction in VR, e.g. interaction with objects over greater distances.

One very important feature that was missing in our system was a form of tutorial that walked users through how to use the interaction system as well as the specific functionality of each technique. While we did explain to users how to work the controls, this would not be feasible if the system was to be deployed and there was no supervision of users. We also got feedback from users requesting a type of tutorial which walks users through each step of the process of using a technique, not letting them make a mistake. Investigating how to make such a tutorial for our techniques could be an interesting prospect.

One suggestion that appeared several times during testing, was that the **Proximity** technique should be combined with the others to allow the first item to be placed with some assistance, rather than using just the free-hand laser. It would be interesting to create this combination as it provides a nice freedom, which could complement the more rigid nature of the other techniques. It would not be hard to do this as the functions of the techniques do not overlap, and they use different buttons.

Another thing that could be interesting to investigate is having users pick up ghosts instead of objects from the beginning. This would allow us to disregard the problem if collision whilst users are moving objects around, as well as letting users cancel their actions, something which is very practical, especially given the current lack of an undo function. This could potentially be easier to rationalise as well, since users are not picking up actual heavy objects, but merely representations of them.

BIBLIOGRAPHY

[1] [Autodesk]. MARUI Maya VR PlugIn. `https://www.marui-plugin.com/`, 2017. Last accessed at 2017, April 5^{th}.

[2] Doug A. Bowman and Larry F. Hodges. An evaluation of techniques for grabbing and manipulating remote objects in immersive virtual environments. In *Proceedings of the 1997 Symposium on Interactive 3D Graphics*, I3D '97, pages 35–ff., New York, NY, USA, 1997. ACM.

[3] Doug A. Bowman, Ernst Kruijff, Joseph J. LaViola Jr., and Ivan Pouprev. *3D User Interfaces: Theory and Practice*. Addison-Wesley Professional, July 2004.

[4] Doug A. Bowman, Ryan P. McMahan, and Eric D. Ragan. Questioning naturalism in 3d user interfaces. *Commun. ACM*, 55(9):78–88, September 2012.

[5] Signe Brewster. Behind the Numbers of Virtual Reality's Sluggish Debut. *MIT Technology Review*, December 2016.

[6] [Canalys]. Who we are. `https://www.canalys.com/about#`, 2017. Last accessed at 2017, April 5^{th}.

[7] [IKEA]. Plan you kitchen in 3D. `http://www.ikea.com/ms/en_CA/rooms_ideas/kitchen_howto/NA/plan_your_kitchen_in_3d.html#`, 2017. Last accessed at 2017, April 5^{th}.

[8] Jason Jerald. *The VR Book: Human-Centered Design for Virtual Reality*. Association for Computing Machinery and Morgan & Claypool, 2016.

[9] Nicolai K. Jørgensen and Christian A. Larsen. System control in immersive virtual reality. Last accessed at 2017, June 1^{st}. 9th semester project report [in Group VGIS943]. Not published. (Write e-mail for access at cala10@student.aau.dk or njarge12@student.aau.dk), 2016.

[10] Robert S. Kennedy, Norman E. Lane, Kevin S. Berbaum, and Michael G. Lilienthal. Simulator sickness questionnaire: An enhanced method for quantifying simulator sickness. *International Journal of Aviation Psychology*, 3(3):203, 1993.

[11] Paul Lamkin. HTC Vive VR Headset Sales Revealed. *Forbes*, October 2016.

[12] [LEAP MOTION, INC.]. Leap Motion. `https://www.leapmotion.com`, 2017. Last accessed at 2017, May 05^{th}.

[13] Sotiris Makris, Loukas Rentzos, George Pintzos, Dimitris Mavrikios, and George Chryssolouris. Semantic-based taxonomy for immersive product design using vr techniques. *CIRP Annals - Manufacturing Technology*, 61:147–150, 2012.

[14] [MARUI - VR Plug-In for Maya]. Doing Light-Setup in VR. [Video File]. Retrieved from `https://www.youtube.com/watch?v=p4__Px9ZeTs`, November 2016. Last accessed at 2017, April 20^{th}.

[15] [MARUI-PlugIn]. MARUI Maya VR PlugIn. `https://www.marui-plugin.com/`, 2017. Last accessed at 2017, April 5^{th}.

[16] Mark R. Mine, Frederick P. Brooks, Jr., and Carlo H. Sequin. Moving objects in space: Exploiting proprioception in virtual-environment interaction. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '97, pages 19–26, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

[17] Jeffrey S. Pierce, Andrew S. Forsberg, Matthew J. Conway, Seung Hong, Robert C. Zeleznik, and Mark R. Mine. Image plane interaction techniques in 3d immersive environments. In *Proceedings of the 1997 Symposium on Interactive 3D Graphics*, I3D '97, pages 39–ff., New York, NY, USA, 1997. ACM.

[18] [Planner 5D]. Create home design and interior decor in 2d & 3d without any special skills. `https://planner5d.com/`, 2017. Last accessed at 2017, April 5^{th}.

[19] Ivan Poupyrev, Mark Billinghurst, Suzanne Weghorst, and Tadao Ichikawa. The go-go interaction technique: Non-linear mapping for direct manipulation in vr. In *Proceedings of the 9th Annual ACM Symposium on User Interface Software and Technology*, UIST '96, pages 79–80, New York, NY, USA, 1996. ACM.

[20] Richard Lowry. One-Way Analysis of Variance for Independent or Correlated Samples. `http://vassarstats.net/anova1u.html`, 2017. Last accessed at 2017, May 31^{st}.

[21] [Road to VR]. Latest Unity 'EditorVR' Demonstrated at Unite 2016. [Video File]. Retrieved from `https://www.youtube.com/watch?v=gV9rpwWWobc`, November 2016. Last accessed at 2017, April 20^{th}.

[22] Richard Stoakley, Matthew J. Conway, and Randy Pausch. Virtual reality on a wim: Interactive worlds in miniature. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '95, pages 265–272, New York, NY, USA, 1995. ACM Press/Addison-Wesley Publishing Co.

[23] Unity Technologies. Unity. `https://unity3d.com/unity`, 2016. Last accessed at 2016, November 11^{th}.

[24] Andries Van Dam, Andrew S Forsberg, David H Laidlaw, Joseph J LaViola, and Rosemary M Simpson. Immersive vr for scientific visualization: A progress report. *IEEE Computer Graphics and Applications*, 20(6):26–52, 2000.

[25] [VR-Plugin]. VR-Plugin Viewer. `https://moculus.io/`, 2017. Last accessed at 2017, April 5^{th}.

[26] [VRtisan]. VRtisan Architectural Visualisation. [YouTube Video Channel]. Retrieved from `https://www.youtube.com/channel/UCWMwwo51BUwnA81ZRIMpqLA`, May 2017. Last accessed at 2017, June 8^{th}.

[27] Jia Wang and Robert Lindeman. Coordinated hybrid virtual environments: Seamless interaction contexts for effective virtual reality. *Computers & Graphics*, 48:71–83, 2015.

[28] Daniel Wigdor and Dennis Wixon. *Brave NUI World - Designing Natural User Interfaces for Touch and Gesture.* Elsevier Science, 2011.

[29] Curtis Wilkes and Doug A. Bowman. Advantages of velocity-based scaling for distant 3d manipulation. In *Proceedings of the 2008 ACM Symposium on Virtual Reality Software and Technology*, VRST '08, pages 23–29, New York, NY, USA, 2008. ACM.

APPENDIX A

TECHNIQUES

In this Appendix the design of the techniques not developed is discussed. As previously mentioned it was not possible to develop and test all available techniques for each pattern within the available time frame, and as such a decision had to be made regarding which patterns and techniques to try. Those that did not make it to implementation were only conceptually designed, which is what is described below. The **Touch-control** technique and the **Gizmo-grab** technique belong to the **Grid-based Pattern** and the **Touch-precision** technique belongs to the **Free-hand Pattern**.

Touch-control Technique

This technique was designed to make use of the trackpad on the Vive controller to move objects in the grid. This would make the trackpad control the position of the object much in the same way the **Crane-control** technique does with its proxy-sphere. The centre of the trackpad would become the centre of the positional manipulation so that when users place their finger at e.g. the top of the trackpad the object would start moving away from the user along the grid; the speed depending on the distance between the position of the finger and the centre of the trackpad. To move objects along the Y-axis (up and down) the users will have to make the trackpad be vertical, i.e. pointing the controller towards the ceiling. Based on previous experience we know that the trackpad has some issues (hardware) that made users dislike it in certain situations which was one of the reasons we decided not to use this technique as the prototype for the pattern.

Gizmo-grab Technique

This technique was designed to try and use a Graphical User Interface (GUI) as a part of the placement manipulate. A bit similar to the **Continuous** technique, this technique would display a menu whenever an object is selected showing three different options for manipulations, namely: left/right, forward/backwards, and up-/down. The user would then need to point with the laser to the desired movement direction and press and hold the trigger down. Moving the laser away from the

point of contact on the menu would then move the object based on which gizmo (menu item) was selected. The speed would be controlled by how far away the user moves the laser from the position of the gizmo (based on its original position as the object would be moving). Another possibility is to use the rotation of the wrist to control the speed in either direction, but this might feel slightly uncomfortable over time. The problem with using the laser could be that, the further away a user selects an object the harder it might be to move objects accurately due to the distance and hand tremors; however, the techniques could be implemented to attempt and account for the distance to the object. The GUI would also have to be very informative and innovative to properly indicate direction and speed, where Natural User Interfaces [28] could be used as inspiration.

Touch-Precision technique

Same as **Touch-control** technique this technique was designed to try and make use of the trackpad to do precise placement and alignment as a part of the two-step process of the **Free-hand Pattern**. As with the **Two-hand** technique, an object is first roughly placed with on hand using the system's laser interaction, and when released the user would press the grip-button on the opposite hand to activate the precision control. The trackpad interaction works exactly like the **Touch-control** technique, except here the movement is not bound to a grid.

APPENDIX B

EXPERIMENT SCRIPT

Here the script for the experiment design is displayed. It was made in an attempt to give every user the same information prior to the beginning of the test.

Intro

Hello and welcome to our Virtual Reality (VR) user test. We are a group of two students from Vision, Graphics, and Interactive Systems, currently writing our master thesis on interaction in VR. For this purpose we have designed and developed three different interaction techniques based on a similar overall architecture and you will be trying each of these techniques. In the test you will be assembling a kitchen based on an existing design that we have made. First you will get a few minutes to practice the controls and interaction common for each technique, and afterwards you will try each technique one after the other. Before we start, and after you have tried each technique you will have to fill out a questionnaire, and once you have tried all techniques we have a follow-up questionnaire as well as short interview based on your experience with the application through this test. First we will let you fill out the background and experience questionnaire and then we will give you instructions for the short practice session.

(BACKGROUND/EXPERIENCE QUESTIONNAIRE FILLED)

Now we will give you the instructions for the practice session. First I will show you how to hold the controllers and how they work and then I will explain the basic functionality of the application.

(SHOW HOW TO HOLD CONTROLLERS - WE GIVE THEM ONE CONTROLLER WHILE WE HOLD THE OTHER)

There are four key components you will be using in this test:

- The **"grip-buttons"** on the side of the controller - most easily pressed by grasping the controller harder with the palm of your hand as well as the middle and/or ring finger from both sides of the controller at the same time. This will be used to move objects around by pointing at an object and pressing the button once and releasing to grasp the object, and press and release again when you want to release it.

- The **"trigger-button"** underneath the controller at your index finger, which works much like the trigger of a pistol. You have to press it all the way until it gives an audible and tactile click for it to register. This will be used to select objects and items in menus. To unselect an object, point to the ground/wall/ceiling and press the trigger (point at nothing).

- The **"trackpad/touchpad"** on top of the controller which is most easily operated by your thumb. It works more or less like the touchpad on a computer except you have to press it till there is an audible and tactile click for clicks to register. This will be used for a small pie menu which gives access to turning a laser pointer (upper part) on and off as well as navigating through the environment through a teleportation option (lower part). When an object is grabbed or selected this menu changes to give the option to rotate the by 90 degrees clockwise or counterclockwise (left and right part of trackpad). Furthermore, when an object is grabbed the trackpad also functions as a scroll wheel allowing you to move objects closer to you or further away by swiping your finger up and down on the trackpad.

- The **"application-button"** above the trackpad also most easily operated by your thumb. This is used as a frustration button, meant to be used whenever the application doesn't function as you would expect based on your action.

Any immediate questions? During the practice session you can ask any questions you might have regarding the general controls. You can now put on the HMD and we will get started.

(USER TAKES ON HMD AND PRACTICE SESSION BEGINS)

During the practice session the task of mirroring the two kitchen designs was more thoroughly explained so that users knew what to do when starting each technique. And in case the users had any critical problems with the controls they were refreshed or explained in greater detail depending on what their problem was. For each technique below, a short demo was given (corresponding to what would be in an in-application tutorial).

Now that you have had a short practice session you should be ready to try the first technique:

Proximity item

In this technique you will primarily be using the grip button to move and place

objects. When an object nears a suitable surface a transparent version of the model will appear. This transparent model will be either green or red. When green, the action of releasing the object will cause it to snap to the position of the transparent model - it functions as a placement indicator. When red, the action of releasing the object will cause it to stay where it is as it does not fit. This technique will automatically rotate objects so that their forward direction is facing away from the surface they are snapped to. When hovering a grabbed object in front of an already placed object, the system will suggest a location to the left or right of the placed object based on the position of the grabbed object. You will only need one hand to move objects around; the other hand can be used for navigation i.e. teleporting.

Continuous item

In this technique you will be using the grip button to move and place the first object in a row and the trigger to expand the row with the needed objects. With this technique you have to ensure the correct rotation when you place an object. Once the first object is placed, the trigger is used to select it which will open a small menu showing two directional arrows. Selecting one of these arrows will decide which direction to expand in. Once a direction is chosen you can use the trigger on the object(s) you want to expand the row with. You only have to chose the direction once to complete a row. You will only need one hand to move objects around and for selection; the other hand can be used for navigation i.e. teleporting.

Preview item

In this technique you will be using the grip button to move, place and rotate objects of your choosing. When an object is selected, you can point at an object you want to place the selected object next to. This will display transparent models at the possible locations the selected objects can be placed. The transparent model will be either green or red. When green, you can point at the transparent model and use the trigger to select it which will place the selected model at the position of the selected transparent model. If the transparent model is red, this means that the selected object does not fit next to the object you are pointing at. You will only need one hand to move objects around and for selection; the other hand can be used for navigation i.e. teleporting.

(QUESTIONNAIRE BETWEEN EACH TECHNIQUE, AND FOLLOW-UP QUESTIONNAIRE + SHORT INTERVIEW AFTER ALL TECHNIQUES HAVE BEEN TESTED)

APPENDIX C

BACKGROUND AND EXPERIENCE QUESTIONNAIRE

This Appendix displays the background and experience questionnaire in which we try an establish the age and experience of users in order to find whether or not either has an influence on the results gathered in the rest of the experiment. This was used to see if a correlation exists between e.g. age and completion time or video gaming and completion time. The start of the questionnaire was basically just the same introductory text as was given in the script, which was skipped by most users.

The rest of this page is left intentionally blank

VGIS10 User Test

We are a group of two students from Vision, Graphics, and Interactive Systems, currently writing our master thesis on interaction in Virtual Reality (VR). For this purpose we have designed and developed three different interaction techniques based on a similar overall architecture and you will be trying each of these techniques. In the test you will be assembling a kitchen based on an existing design that we have made. First you will get a few minutes to practice the controls and interaction common for each technique, and afterwards you will try each technique one after the other. Before we start, and after you have tried each technique you will have to fill out a questionnaire, and once you have tried all techniques we have a follow-up questionnaire as well as short interview based on your experience with the application through this test. First we will let you fill out the background and experience questionnaire and then we will give you instructions for the short practice session.

*Required

Background/Experience

All the questions marked with a star are required to answer.

1. **Age** *

2. **Gender** *
   Mark only one oval.
   - ◯ Male
   - ◯ Female

3. **What is your current occupation?**

4. **Have you ever used a Virtual Reality Headset before?** *
   Mark only one oval.
   - ◯ Yes
   - ◯ No

5. **How many hours a week do you use a computer on average?** *
   Mark only one oval.
   - ◯ 1 hour
   - ◯ 2 hours
   - ◯ 5 hours
   - ◯ 10 hours
   - ◯ 20 hours
   - ◯ 40 hours
   - ◯ more than 40 hours

6. **Over the past two years, what is the most you have played video games in a single week? (Approximately)** *
Mark only one oval.

- 1 hour
- 2 hours
- 5 hours
- 10 hours
- 20 hours
- 40 hours
- more than 40 hours

7. **How many times have you used systems where you used your hands in 3D space to control the computer application? (e.g. not using a mouse, keyboard and/or touchscreen)** *
Mark only one oval.

- Never
- Once
- Twice
- 3-10 times
- 11-20 times
- 20-100 times
- nearly every data as part of my work or my education, or for entertainment

8. **How much experience do you have with CAD (Computer Aided Design) software (e.g. Maya, 3D Studio Max, AutoCAD) in terms of how many times you have used it?** *
Mark only one oval.

- Never
- Once
- Twice
- 3-10 times
- 11-20 times
- 20-100 times
- nearly every data as part of my work or my education, or for entertainment

APPENDIX D

QUESTIONNAIRE ABOUT TECHNIQUES

In this Appendix the questionnaire regarding the three techniques are displayed. There was a similar questionnaire for each technique (3 questionnaires) in order to separate the data, and as the order was different for each participant, we felt this was the easiest way to do it. This questionnaire was used to gather quantitative data regarding the users' experience with the three techniques regarding the learnability and usability.

The rest of this page is left intentionally blank

Proximity Interaction

*Required

1. **I found the interaction technique easy to learn.** *
   Mark only one oval.

	1	2	3	4	5	
Strongly disagree	◯	◯	◯	◯	◯	Strongly agree

2. **Once I learned the interaction technique, I found the object manipulation easy and intuitive to use.** *
   Mark only one oval.

	1	2	3	4	5	
Strongly disagree	◯	◯	◯	◯	◯	Strongly agree

3. **Once I learned the interaction technique, I felt it was easy to properly align objects with each other.** *
   Mark only one oval.

	1	2	3	4	5	
Strongly disagree	◯	◯	◯	◯	◯	Strongly agree

4. **I prefer this new interaction technique over traditional mouse and keyboard interaction.** *
   Mark only one oval.

	1	2	3	4	5	
Strongly disagree	◯	◯	◯	◯	◯	Strongly agree

5. **I was able to easily replicate the two premade kitchen designs using this interaction technique.** *
   Mark only one oval.

	1	2	3	4	5	
Strongly disagree	◯	◯	◯	◯	◯	Strongly agree

6. **I enjoyed the experience.** *
Mark only one oval.

	1	2	3	4	5	
Strongly disagree	◯	◯	◯	◯	◯	Strongly agree

APPENDIX E

POST EXPERIMENT QUESTIONNAIRE

In this Appendix the post experiment questionnaire is displayed. This questionnaire was three-fold as the users first had to answer some questions regarding their overall experience after trying all three techniques. Then a short simulator sickness questionnaire was posted based on the Kennedy Simulator Sickness Questionnaire [10] where only a portion of the questions were asked (we do not go into details on the scoring of the questionnaire). Finally, the last part of the questionnaire was carried out as a short interview where the facilitator asked the questions to the participants.

The rest of this page is left intentionally blank

Post Questionnaire

A score of 3 means that you are indifferent (or don't know)

*Required

1. **Overall, I enjoyed the experience** *
   Mark only one oval.

	1	2	3	4	5	
Strongly disagree	◯	◯	◯	◯	◯	Strongly agree

2. **If I were to design my own kitchen I imagine I would prefer to do it in VR using a system like this.** *
   Mark only one oval.

	1	2	3	4	5	
Strongly disagree	◯	◯	◯	◯	◯	Strongly agree

3. **I would recommend the system to friends.** *
   Mark only one oval.

	1	2	3	4	5	
Strongly disagree	◯	◯	◯	◯	◯	Strongly agree

Simulator sickness

4. **Do you feel that you are in the same state good health as when you started the experiment? If not, please elaborate in the textfield below.** *
   Mark only one oval.

   ◯ Yes
   ◯ No

5. _____

For each of the following conditions, please select how you are feeling right now, on the scale of none through severe:

6. **General discomfort** *
Mark only one oval.

- None
- Slight
- Moderate
- Severe

7. **Fatigue (weariness or exhaustion of the body)** *
Mark only one oval.

- None
- Slight
- Moderate
- Severe

8. **Headache** *
Mark only one oval.

- None
- Slight
- Moderate
- Severe

9. **Eye strain (weariness or soreness of the eyes)** *
Mark only one oval.

- None
- Slight
- Moderate
- Severe

10. **Nausea (stomach distress).** *
Mark only one oval.

- None
- Slight
- Moderate
- Severe

11. **Blurred vision** *
Mark only one oval.

- None
- Slight
- Moderate
- Severe

12. **If you epxressed slight, moderate, or severe on any of the questions above, please state if you felt that way before using the system, and if so, explain how you felt worse after using the system.**

Interview

13. **What did you like most about the system?**

14. **Which of the three techniques you just tried would you prefer? (If you had to choose only one)**
Mark only one oval.
- (A) Proxy
- (B) Continuous
- (C) Preview

15. **What did you dislike about the system?**

16. **How long did it take you to learn to use the system?**

17. **What suggestions or ideas do you have for improving the system?**

18. **Any other comments?**