Technology
Arts Sciences
TH Köln

Project Documentation

Interaction Lab

written by

Vera Brockmeyer	(Matrikelnr. 11077082)
Anna Bolder	(Matrikelnr. 11083451)
Britta Boerner	(Matrikelnr. 11070843)
Laura Anger	(Matrikelnr. 11086356)

Interactive Systems in SS 2017

Supervisor:
Prof. Dr. Stefan Michael Grünvogel
Institute for Media- and Phototechnology

Contents

1 Introduction

The main idea of Virtual Reality (VR) is to ensure a totally immersive user experience and the first important milestone was the development of the first Head-Mounted-Display (HMD) in 1966 by Sutherland and Goertz. This HMD offered the possibility to walk around in each virtual scene and to look at the scene from every view point. Despite this possibility, a totally immersive experience requires user interactions with virtual objects like selecting, grabbing, manipulating, movement and indirect controlling via widgets, gestures and voice input in virtual scenes. Especially a realistic grabbing and positioning of these objects is required which should come close to human motion sequences and fits to the human cognition and experiences. These kind of immersive interaction methods should include further requirements as well. A high precision rate is unavoidable and must be guaranteed. Another condition is that the method offer an ergonomic usage which is not tiring for the users and can be learned easily.

The number of VR devices and applications increased heavily during this decade and the relevance of consumer and business VR applications rises steadily. Several suppliers currently offer various methods and devices for grabbing interactions. The most common are the *Oculus*-HMD, *HTC Vive*-HMD, *HTC Vive*-Controller, data gloves and motion capturing systems for hand-tracking like the *LeapMotion*-Controller. Most devices offer a system with two hand controllers at this state. These hand controllers enable the use of virtual menus and offer a robust hand tracking. The latter is urgently needed to implement common grabbing interactions. Due to this rising amount of VR devices, it is vital for the development and improvement of grabbing methods to evaluate their usability and performance in an adequate environment as well as to compare them with the state of art methods.

In the following sections, we describe and discuss an interactive VR system to compare different controller-based grabbing methods to evaluate their usability, error rate and time consume. This VR system runs with the *HTC Vive*-HMD and two *HTC Vive*-Controllers. Thus, all offered grabbing methods are controller based which could be learned in a special learning room and evaluated in a predefined VR supermarket with all required measurement tools.

1.1 Motivation

Currently there no interaction laboratory exists that compares different grabbing interaction methods. Similar laboratories [17][24][**?**] have been developed but they do not allow the user to compare different kind of grabbing methods. In case of [**?**] and [24], it provides only some VR applications or devices to experience the methods in different scenarios. Another [17] laboratory evaluates natural grabbing methods without a credible evaluation.

This circumstance calls for a virtual laboratory where different interaction methods can be compared, demonstrated, or tested in adequate virtual test scenes. Furthermore, all tests and compares should be based on scientific standards to allow factual

and useful results. Thus, this laboratory should provide all required measurement tools to start credible and scientific studies of pre-implemented grabbing methods. This helps to standardise the evaluations and to increase their comparability of common studies. Another achievement is that the user friendliness of interaction methods which are not tiring and do not destroy the immersion could be improved by researcher.

Another aspect of those user-friendly methods is the increasing usability of VR applications which will yield to a higher consumer preference of devices with their implementation. Therefore, the profit of VR device suppliers will be squeezed and the relevance of those products will expand worldwide.

In an educational context it could be used for demonstrations during lectures. This will give the lecturer an effective tool at hand to explain the importance of usability as well as the advantages and disadvantages of each grabbing method. Another aspect of this laboratory is to give students a tool for the technical realisation of interaction studies in virtual (or augmented) reality environments.

1.2 Project Goal

Hence, a virtual interface laboratory with an environment to test and compare grabbing methods will be be developed. It will offer a possibility to develop new ones and it provide a use for teaching purposes as well. Thus, one task is to develop sophisticated test scenes for testing the interaction methods. These scenes will implement test exercises in different difficulty levels and represent typical and well-known environments like shops. All relevant parameters for the evaluation of the methods will be measured automatically and saved in an output comma separated value file. The latter must be easily imported in common statistic tools. Additional the laboratory will provide digital questionnaires to evaluate the usability of each provided method as well as one that enquire the system relevant properties like motion sickness, immersion, and latency.

2 State of the Art

In the following sections the fundamentals and scientific knowledge of all parts of the *Interaction Lab* will be described. Publications of similar projects are described in section 2.1. Furthermore, an overview of methods of (grabbing) interactions (compare section 2.2), as well as their evaluation methods (compare section **??**) are given.

2.1 VR Labor

Due to the progress in the development of grabbing interactions, various evaluations of those interactions were needed and implemented. There are a few papers and other articles. In the following sections three different works will be described shortly.
In the year 1999 Pouryrev and Ichikawa categorized and evaluated different grabbing interactions in their paper [22]. In their project they used hand tracking to measure the position of the real hand and to visualise a virtual representation of the hand. The participants than had to select different simple test objects, like cubes, spheres and cylinders. Also a positioning task was available.
A few years later Lee et al. evaluated different raycast grabbing methods [15]. Lee at al. used a 3D mouse for tracking the position and orientation of the hand. With this mouse the participants selected spheres on various and random positions. Therefore the participants were ask not to change the position of their head.
In the last year Eriksson published his master thesis, which is relative similar to this project [9]. Erikson used the *Oculus Rift* [30] with the Oculus Touch to provide the tracking and visualisation. In this project the participants also had to complete tasks in a virtual shop. The tasks had similar requirements, related to the selection, but also refer to manipulation and translation of the objects.

2.2 VR Grabbing-Interactions

To create an immersive virtual environment it is, inter alia, necessary to use adequate grabbing methods [2]. As mixed reality was not taken into consideration for implementing the *Interaction Lab*, all objects, the user can interact with, are completely virtual.
The interaction methods provided by the *Interaction Lab* have no haptical feedback [1], which would allow a conclusion about the surface quality of the gripped object. Furthermore, they only use the hardware described in section 3.1.2 for the execution of the interactions.
Many of the common methods of interaction, which are available in the *Interaction Lab* (compare section 4.3) are explained in the paper by Bowman and Hodges [2]. In their explanations they also mentioned the *Go-Go* technique, which presents a conceivable extension of the laboratory.

2.3 VR Grabbing-Interactions Evaluation

To determine usability of computer based applications has been a problem from the start of the computer development. Many usability questionnaires are available but short and easy to use ones are preferred. A widely used usability questionnaire is the *SUS: A quick and dirty usability scale* [3]. It has been developed in 1986 by John Brooke. The ten questions are phrased in such a way that the questionnaire can be used generally for any digital application or even websites. Corresponding to that, since virtual reality applications started to appear, there needed a way to test how it affected the users. Kennedy et al. developed a *Simulator sickness questionnaire* in 1993 [14], which is still in use today. It tests sixteen symptoms that the user might feel and give a total score as the result. Furthermore, three sub results for nausea, oculomotor and disorientation are recorded as well. Even though these questionnaires are not the newest, the frequent use of them show that they are still up to date.

3 Materials

The following section will give an overview over all hard- and software that was used in this project.

3.1 Hardware

The hardware consists of a head-mounted display, the HTC Vive and a computer.

3.1.1 Computer

CGPC6	Beschreibung
Prozessor	Intel Core i7 6700 CPU @ $4 \times 3.4 - 4.0$ GHz
Arbeitsspeicher	16 GB
Grafikkarte	NVIDIA GeForce GTX 980
Betriebssystem	Windows 10 Education 64 bit
Schnittstellen	2× USB 3.0, 5× USB 2.0, 1× HDMI

Table 1: Summary of the technical specifications of the used computer for the *Unity* simulation.

The hard- and software requirements for using *Unity* with the *HTC Vive*, which are listed in Table 2, are being exceeded by the used computer.

HTC Vive	System requirements
Prozessor	mindestens Intel Core i5 4590 oder AMD FX 8350
Grafikkarte	mindestens NVIDIA GeForce GTX 1060 oder AMD Radeon RX 480
Arbeitsspeicher	mindestens 4 GB
Videoausgang	1× HDMI 1.4 Anschluss oder DisplayPort 1.2
USB	1× USB 2.0-Anschluss
Betriebssystem	Windows 7 SP1, Windows 8.1 oder Windows 10

Table 2: *HTC Vive* System requirements [11].

3.1.2 HTC Vive

The *HTC Vive* is a head-mounted display which is being produced by *HTC* in cooperation with *Valve* [29]. It was introduced on the 1st of March 2015 at the *Mobile World Congress* [20].The resolution of the integrated display is 2160×1200 pixels which equals 1080×1200 pixels per eye. The head-mounted display offers a field of vision of 110° with a refresh rate of $90\,Hz$ [10]. The system requirements can be

seen in table 2.
Hier wird dreimal "the" als Satzanfang benutzt.

To determine the head-mounted displays position within the room, the lighthouse technology [4] by *Valve* is used. Evtl. Furthermore, the lighthouse technology by .... used to determine ... room. Additional, the head-mounted display is equipped with a gyroscope, an acceleration sensor and a laser position measurer. The interaction with virtual objects is made possible with a proprietary hand controller.

3.2 Software

3.2.1 Unity

Unity is a so called game engine, meaning a developing and runtime environment, which was created for the development of 3D games. The software was released on the 6th of June 2005 [**?**] and is being developed and marketed by *Unity Technologies* [25]. Unity is widely used in the game development industry. 34 % of the Top 1000 mobile free to play games are developed with *Unity* [27]. Kann es sein dass dieser Satz keinen Sinn ergibt?

Unity offers broad platform support [26] and also allows the development for head-mounted display, like the *Oculus Rift* [28] or the *HTC Vive* [28], which was used in this project.

3.2.2 Visual Studio 2015

Micosoft Visual Studio 2015 is a widespread development environment (IDE). It supports the programming languages Visual Basic, Visual C#, and Visual C++ amongst others. With the help of the IDE a developer can program and compile Win32/ Win64 and web applications. For *Interaction Lab* version 14.0.25123.00 update 2 was used. Hier fehlt noch die Referenz. Ich glaube ich habe damals einfach auf die Homepage verwiesen.

3.2.3 Steam VR

Steam VR [23] is an interface between the *HTC Vive* and *Unity*. *Steam VR* has to be installed on the computer to use the head-mounted display (HMD). A small GUI element can be seen on the monitor, which show the current status of the *HTC Vive*. It communicates error messages and is used for the system calibrating.
Within *Unity*, *Steam* provides a plug-in, which can be embedded into the *Unity* scenes. The developer could integrate the VR options easily per drag-and-drop into a existing *Unity* scene. The provided *Unity* prefab includes all necessary elements to communicate with the hardware. A determination of position and a camera rig for the stereoscopic display is provided as well as the possibility for hand controller input and further use of this data.

3.2.4 Virtual Reality Toolkit (VRTK)

Anna

The "VRTK" (Virtual Reality Toolkit) is a free packet with different scripts and prefabs for interaction in the virtual reality [18] [19]. In this project different parts of this plug-in are used.
The *radial menu* Muss hier noch ne Dateiendung hin? Haben das sonst immer so gemacht :)script is used for the controller menu, see section 4.2. This script creates menu buttons over the touchpad of the *HTC Vive* controller. By scrolling over the touchpad a button can be selected. The selection will be shown by highlighting the button. If this button should be executed= For executing this button, the touchpad has to be pressed, when the button is selected. The number of buttons per menu can be changed within the *Unity* editor and each button can get a different icon. Through a link to a function of a script this function can be executed if the button is pressed. Furthermore the prefab for *controller tips* is used for visualisation of the selfteaching (compare section 4.4), and the tasks (compare section 5.1). This prefab creates an area next to the controller and a line from this area to the selected button of the controller. This area is parented to the controller and moves according to it. Different settings can be changed within the *Unity* editor or by a script.
Besides that there is a very similar prefab, which is called the *object tips*. This is also a canvas, in which information can be displayed, but this can be fixed on any object you like. This prefab is used for the labelling of the target object as well as the visualisation of additional information in the supermarket, see section 5.1.

4 System

The actual application can be divided into two types of VR rooms. The first room is designed as a learning room (compare section 4.1.1), in which the user can get in touch with the different interaction methods. Afterwards different tasks will be presented to him in a VR supermarket scenario (compare section 4.1.2), which is the second room. In the learning room the user will be supported in his learning process by a selfteaching system (compare section 4.4), which he can switch on and off, when he is in the supermarket. The user can make this setting among all other settings in a Menu (compare section 4.2), which is controllable with the *HTC Vive*-controller. In this menu he can also choose between all provided interaction methods, which are described in section 4.3.

4.1 VR Labor

In this section the different rooms of the VR Labor will be described. Each room is realised as a *Unity* scene.
There are two different kinds of rooms. First there is a simple room to get familiar with the system and the methods, which were implemented. This room will be specified in section 4.1.1.
The second room is created to look like a small supermarket, in which the user has to solve different tasks (compare section 5.1). This room will be described in section 4.1.2.

4.1.1 Learning room

The learning room is the first room where the user will get to know the system. On one hand the user will get familiar with the virtual experience and the *HTC Vive* system (HMD and controller) and on the other hand the user will learn the different interaction methods.

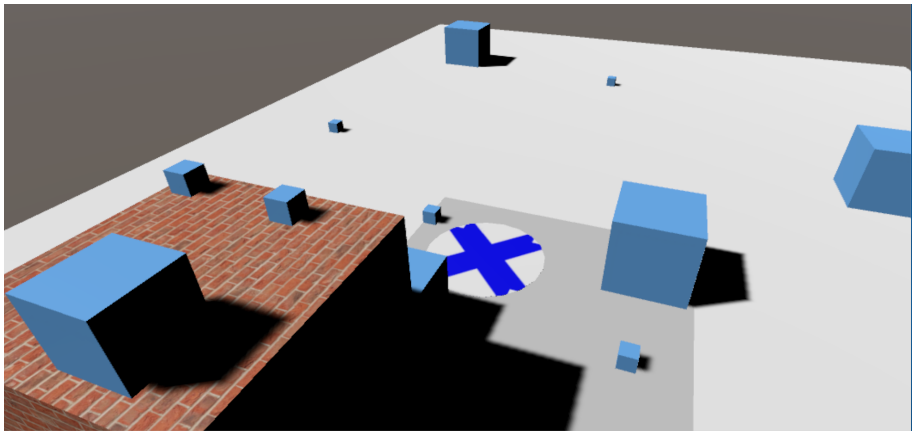

Figure 1: Overview of the Learning Room.

The room is designed very clean and simple. In different sizes and distances simple objects, in this case cubes, are distributed randomly. This is shown in figure 1. The virtual room is slightly bigger than the usable room in reality. So the user is forced to user interactions, which are designed for larger distances, to move some objects (compare sections 4.3.4 and 4.3.5). Due to different colors of the floor the user know how big the real room is approximately. In addition the *HTC Vive* system offers a coloured wire, which is shown in the HMD, whenever the user gets too close to the border of the calibrated area.
The user should already know the different interactions as well as the menu settings, when he enters the supermarket scene. So the menu and interactions are similar in both rooms. Also the labelling of the target object and the reaction of the target area to the target object will be established. This will be described in section sec:tasks. To get to know all settings and methods the user will be led to the entire learning room by a selfteaching system. This selfteaching is be described in section 4.4.

4.1.2 Supermarket

The second room is a small supermarket.

Figure 2: Overview of the supermarket.

Like it is shown in figure 2 in this supermarket are different sized objects in various distances. All objects could be found in a real supermarket, from fruits to milk. This objects were dowloaded from the asset store [21] [16] [13].
The size of the real room is also displayed by using a slightly darker colour on the floor of the supermarket. This is necessary, because the supermarket is bigger than the real room. So interactions for far distance have to be used according to the tasks.

Figure 3: Closer view of the Shelves.

In some tasks the participant has to grab objects which are hard to pick. On one side they could be at a lower position, on the other side they could have to move other objects before they reach the labelled object. To implement this into a supermarket naturally shelves, like in figure 3, are appropriate for a supermarket. This shelves were build with different scaled cubes.

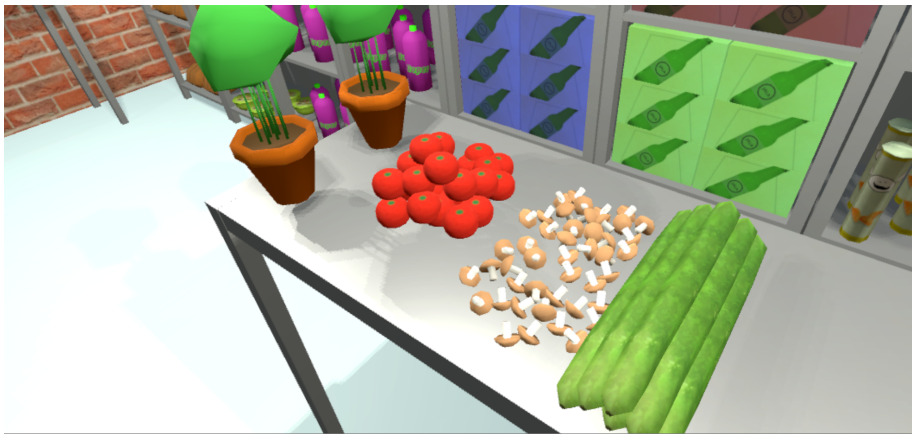

Figure 4: Closer View of the Table.

In a real supermarket objects are not always arranged in shelves. Due to this fact and to have a variance for the tasks, some objects where placed on tables as one bulk, for example fruits and vegetables. There are two tables in this supermarket. A closer look to one of this tables is given in figure 4.
Depending on the task different target areas can appear in the supermarket. In this situations different objects will be blend out. Also the target object of every tasks will be labelled. The first target area and target object are shown in figure 2.
The structure of the labelling of the target object are specified in section 4.4. The different tasks as well as the reaction of the target area to the target object will be described in section 5.1.

4.2 Controller Menu

The controller menu is the main menu of the system. It allows to change settings without leaving the virtual environment. Within this menu interaction methods can be changed and settings like snapping for the interactions can be enabled. In the learning room the user is able to turn the selfteaching informations on and off. In the same way the user can decide if the tasks will be shown or not in the supermarket. In the supermarket scene the user can restart a single task via the controller menu. With the reset button the scene will be reloaded. The main menu is shown in figure 5.

Figure 5: Main Menu of the Controller Menu.

If the button "select grab" is available, the user is able to choose every interaction method he likes. The button leads to an other menu. In this menu the methods are shown as icons, see figure 6. The user learns the meaning of those icons with the help of the selfteaching, while they are in the learning room.

Figure 6: Select Grab Menu.

For the measurement, which are described in section 5.2, the user has to start and stop every tasks. The start button will be the first menu button in every scene.

Only if this button is pressed the user can interact with the environment and will be led to the next menu. When the user presses the stop button the switching area will appear and the corresponding *BoxCollider* [5] will be enabled. If the user reaches this area the next scene will be loaded.
The structure of the menu is built on the plug-in "VRTK" which is described in section 3.2.4.
In this project there is a script named *Menu.cs* in which all functions for the menu are collected. Due to this script different settings can be chosen.

4.3 Interaction Methods

Of course there were different interaction methods required to make the *Interaction Lab* suitable for the testing described and evaluated in section 5. Also all interaction methods are implemented to realise the grabbing of virtual objects, they can be separated in the two categories, described in the next two paragraphs:

Close Range (CR) Interactions: The CR can be interpreted as a synonym for the natural interaction radius of the person. Due to this definition it is excluded that those interactions can be used outside an area, which the person can not reach with his arm, or to be more precise: with the controller in his hand. In other words: the CR combines all interactions which can be used to pick up objects in the direct reach of the user.

Far Range (FR) Interactions: Due to a limitation of the range of motion in VR applications, it is common to have interactions designed for grabbing objects which are normally seen as out of the reach of the user [12]. Interaction methods allowing such an acting are called FR interactions. It is not excluded, that a FR interaction is used in the actual reach of the user.

Whereas the CR interactions differ mainly in the accuracy of the selection of an object while grabbing it, the FR methods differ in their usability. All characteristics of the various interaction methods can be traced in their descriptions (compare sections 4.3.1 - 4.3.6).

For a better understanding it should be mentioned, that all interaction methods can be controlled with the *HTC Vive*-controller. Even there are plenty of different possibilities to grab an object all methods have in common that the grabbing is caused by pressing the trigger on the *HTC Vive*-controller. The releasing of the object is than triggered by letting it go. Whenever there is a divergent usability necessary, it is described in the respective section (compare 4.3.5).

Due to an easier integration into the learning room (compare section 4.1.1), as well as the actual supermarket scenes (compare section 4.1.2) all methods using a ray (compare sections 4.3.5, 4.3.4 and 4.3.3) are summed up in one script called *AllRaycastMethods.cs*. All other methods have their own script in which the grabbing

and releasing is implemented. Also the *Raycast Head Mounted Display*-method, described in section 4.3.6, is using a ray it is not included into the script mentioned above. This is caused by remaining problems during the implementation of this method which lead to an unfinished work. Further explanations on why this method is not available in the *Interaction Lab* can be found in the according section.

The two interaction methods, described in sections 4.3.1 or rather 4.3.2, can be used with snapping or without it. This technique is used to reassign the position and orientation of a grabbed object in hand. By assuming that the middle of the ring of the *HTC Vive*-controller is the new center of the grabbed object the actual grab could appear more realistic to the user.

In the application all objects, which can be grabbed are tagged as moveable.

In the following sections all available interaction methods of the *Interaction Lab* are presented. To guarantee a better overview they are sorted by their interaction range.

4.3.1 Close Range: Touch Grab

When the *Touch Grab* interaction method is selected the user can make use of the *HTC Vive*-controller to pick up objects directly by touching them. An object can be grabbed whenever it is tagged as moveable and collides with the *HTC Vive*-controller. This collision is detected by giving the object a collider, which fits its form best [5][7] and applying a *BoxCollider* to the controller. The interaction can be seen in figure 7.
In the script *TouchGrab.cs*, in which the interaction method is implemented, is checked frequently, whether there is a overlap of the collider of the controller with the collider of a moveable object or not. Whenever they collide, the respective object is coloured green to show the user that he could grab it by pulling the trigger.

Figure 7: Grabbing a virtual object by using *Touch Grab*.

4.3.2 Close Range: Proximity Grab

When it comes to the CR interactions the *Proximity Grab* is by far the most inexact selection.
The functionality is provided by the script *ProximityGrab.cs*. The basic idea is that the object can be grabbed, whenever the object triggers the *BoxCollider* [5] placed at the end of the *HTC Vive*-controller. A more detailed description can be found in section 4.3.1. In contrast to the *Touch Grab* described in section 4.3.1 this *BoxCollider* is bigger than the actual size of the controller. To show the user, which object collides with the controller and can therefore be grabbed, the respective object is coloured green, as shown in figure 8. The small gap between the actual grabbed object and the controller shows the difference between the interaction method shown in figure 7.

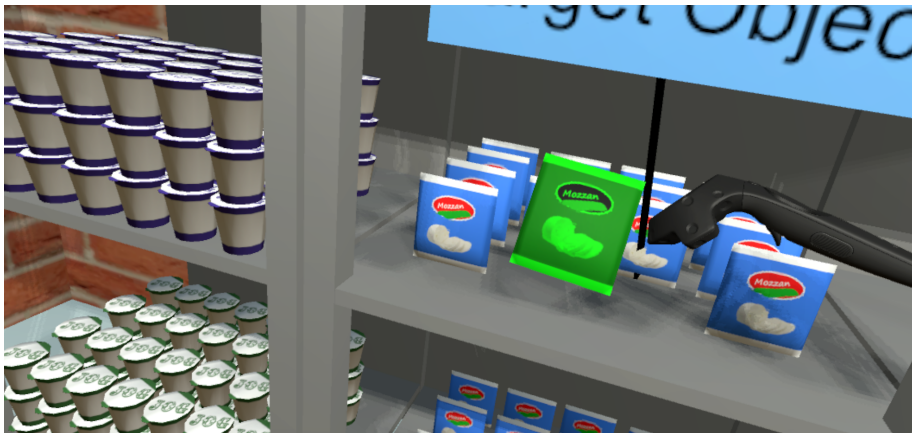

Figure 8: Grabbing a virtual object by using *Proximity Grab*.

4.3.3 Close Range: Wand Grab

In contrast to the interaction method described in 4.3.2 this method can be used to grab very tiny objects. Thereby it is not needed that the target object is very isolated from other objects. To give the user such an high grade of accuracy a stick is added to the controller like shown on figure 9.
The implementation can be found in *AllRaycastMethods.cs*. The wand consists of two elements: a ray [6] and a cube. The cube is only for the visualisation and has a fixed size in all three dimensions. The collision detection, which is necessary for the actual grabbing, is done with the ray. That means that an object can be grabbed if the ray, which has the same dimensions like the cube, touches this specific object. The stick will then turn from black to green to show the user that there is an object which can be grabbed.

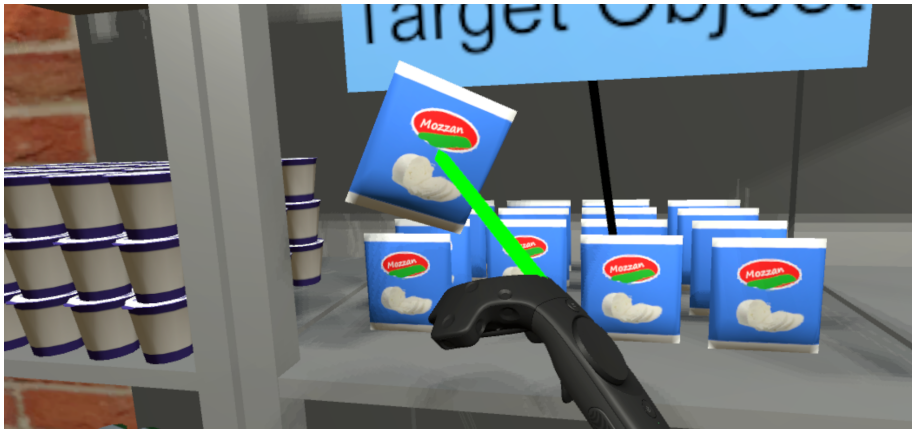

Figure 9: Grabbing a virtual object by using *Wand Grab*.

4.3.4 Far Range: Raycast

By using the *Raycast* method the user can grab virtual objects which are further away as well as objects in his CR. As shown in figure 10 a ray is coming out of the *HTC Vive*-controller pointing away from the user. At the end of the ray is a small sphere, which turns green, if it collides with an moveable object. Whenever the ray hits an objects, like for example the floor or a product in the supermarket (compare section 4.1.2), the ray is shortened to the distance between the controller and the respective object.
The implementation can be found in the *AllRaycastMethods.cs* script. As already explained in section 4.3.3 a cube and a ray are combined to reach the intended functionality. In contrast to the *Wand Grab*, the ray and the visible cube have no fixed length and there is a sphere added to the end of the cube.

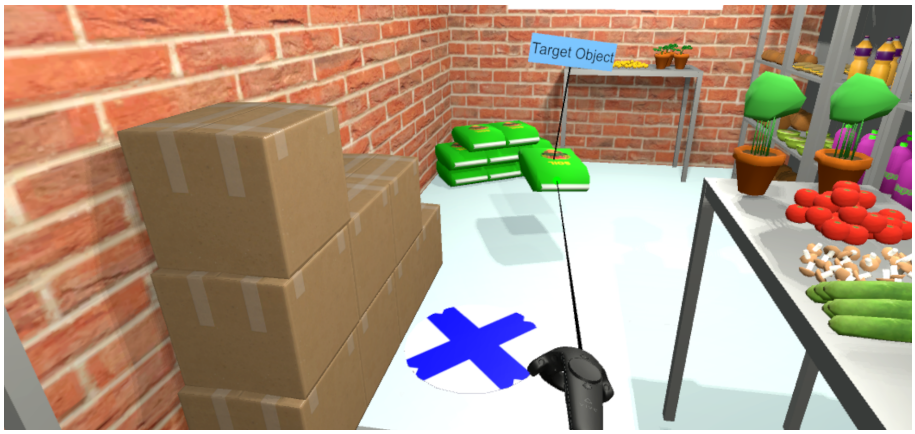

Figure 10: Grabbing a virtual object by using *Raycast*.

4.3.5 Far Range: Extendable Ray

The actual ray is build as described in section 4.3.4. The ray [6] is complemented by a cube with a sphere at the end, to make it visible for the user. In contrast to

the normal *Raycast* method the length of the ray is set to a start value of 3 meters. The user can shorten and lengthen the ray by pressing the touchpad of the *HTC Vive*-controller in the lower or rather upper area. This subtracts or adds a constant value to the length of the visible ray. The behaviour of the sphere remains, which turns green, whenever a moveable virtual object is brushed. The *Extendable Ray* is one of the three interactions methods (compare sections 4.3.3 and 4.3.4) which are combined in the *AllRaycastMethods.cs* script. This interaction method is shown on figure 4.3.5.

Figure 11: Grabbing a virtual Object by using *Extendable Ray*.

4.3.6 Far Range: Raycast Head Mounted Display

It was planned to realise an interaction method where there is a ray coming out of the HMD which can be used similar to the method described in section 4.3.4. Due to the low accuracy of this method it did not become a part of the *Interaction Lab*. There was a try to parent [8] the position of the HMD to the starting point of the ray. It turned out that the parenting-method is not successful when it comes to the position and rotation of the HMD. Even this method was effective for adding a ray to the *HTC Vive*-controller, there is a irregular shift when you try to implement it with the HMD. To proof that an easy example (compare source code 1) was observed. In this example a simple cube should be rendered at the front of the HMD. In reality this cube was rendered in a position which can be seen as random.

```
1 cube.transform.SetParent(HMDEye.transform);
```

Source Code 1: Test on the parenting of *HTC Vive*-HMD and an virtual object.

All effort on this method can be found in the script *RaycastingMethodeHMD.cs*.

4.4 Self-Teaching

Anna

To guarantee an easy and smooth introduction to the system, an automatic selfteaching is integrated into the learning room. When the program is started all necessary instructions will be shown next to the controller, like shown in figure 12.

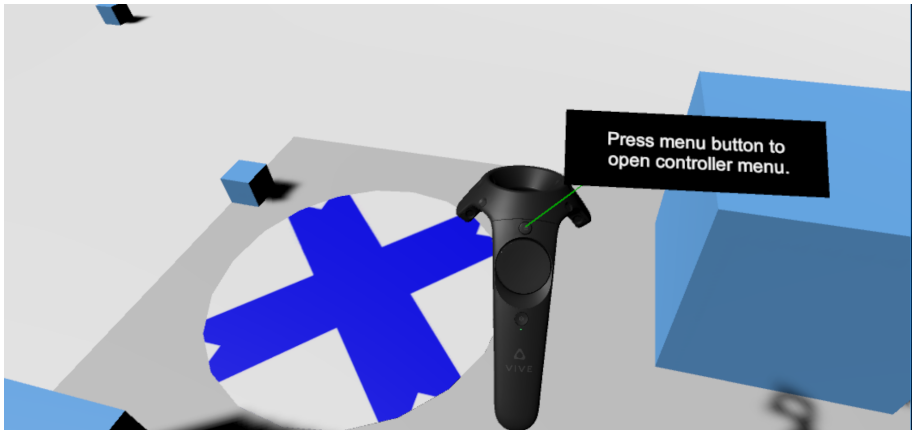

Figure 12: Start of the Selfteaching.

This instructions led the user step by step through the system. He will get to know how the interaction methods can be chosen and what settings can be made. Of course the actual grabbing and releasing of an object is taught as well, for every interaction method.

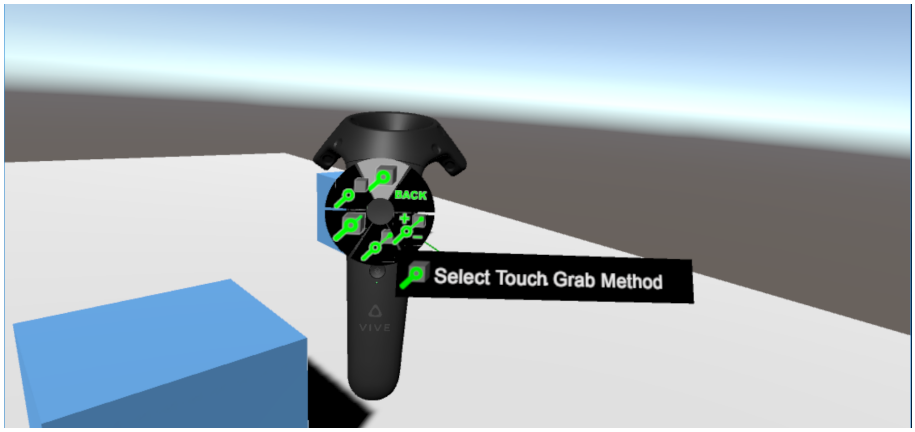

Figure 13: Selfteaching recommends to select *Touch Grab* method.

The instructions will only be visible in the learning room. The position of the information area changes depending on which button is important for the interaction.
In the first grab method, the *Touch Grab* method, two terms will be established, which are important for all tasks. This terms are the labelling of the target object and the target area. Here the user learns how the target object is marked, compare figure 14, and how the target area reacts to the target object.

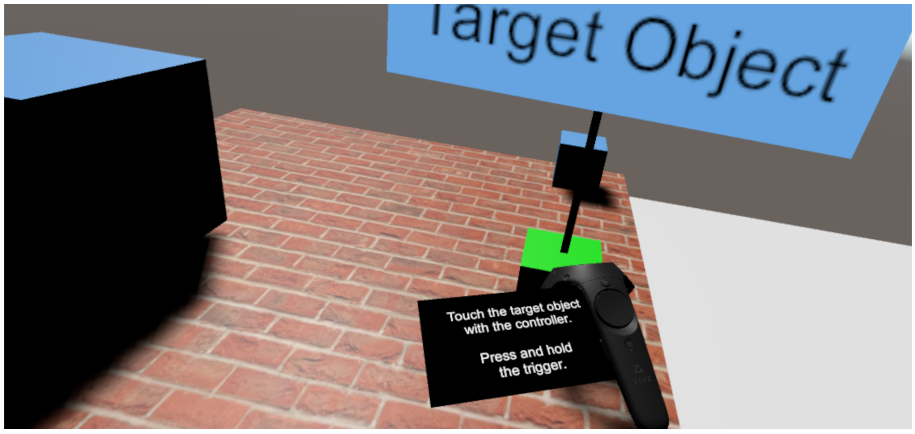

Figure 14: Selfteaching with Target Object.

All instructions and steps of the selfteaching are saved in an external CSV-file, see following figure.

```
0;Press menu button to\n open controller menu.;5;-1;60
1;Press START to begin\n the Learning Process.;3;-1;60
3;Press SELECT GRAB Button;3;-1;30
4; . ;3;0;30
5;Touch the target object\n with the controller. \n \n Press and hold \n the trigger.;1;-1;110
6; Hold trigger and move \n object to target area. \n\n Release trigger to drop \n the object.;1;-1;100
7;Press menu button to \n open controller menu.;5;-1;60
8;Press USE SNAP to \n enable the snapping mode.;3;-1;60
9;Press menu button to \n close controller menu.;5;-1;60
```

Figure 15: Extract of the CSV-file of the instructions of the selfteaching.

Due to an additional script *WriteMeasureFile.cs* this file will be read line after line and be saved into different lists. In each line are four main information. The first number has no effect any more. The first important information is the text which should be shown in this step. Second is the Button ID, which defines on which button the information should be fixed. Third is a number which characterises if a material should be loaded.Vielleicht hier die Nummerierung nach den Spalten machen also stringent von 1-4? Auch wenn die erste Zahl nicht mehr notwenig ist? If this number is a -1, no material will be loaded and the text will be displayed on its own. If this number equals 0 a specific material will be loaded into the canvas. Which material will be loaded is defined in the *SelfTeaching.cs* script. The fourth When ja dann hier fifth :)and last information of the CSV-file is the height of the information area. This height changes depending on the amount of information. For each of this information a list is implemented in the *SelfTeaching.cs* script.
In this script a counter is implemented. This counter controls which line of the file will be displayed. It will be increased or set from different scripts depending on an action. This increasing is still a problem, if the user does not follow the instructions of the selfteaching step by step. For example, if the user grabs an object twice and the selfteaching plans to grab it just once, the selfteaching will go on and the user misses instructions.
The instruction canvas is established by the *controller tips* prefab of the "VRTK" plug-in, see section 3.2.4. This plug-in has also a prefab for *object tips*. With this prefab the target objects are labelled.

5 Evaluation

Hier fehlt die Einführung, Britta

5.1 Tasks

The supermarket (compare section 4.1.2) is confronted with four tasks. In every task, an object has to be selected and placed on a target area. If the correct object is placed, the target area will change the color to signal that the tasks is successful done. In the script *TargetTest.cs*, which is added as a component to the target object, is recognized when the target object hits the target area. At this moment the texture of target area is changed and the measurement (compare section 5.2) is stopped.
In the first three tasks the user has to select different objects. Therefore the user can decide which method he/she uses to grab the object. The selected method should differ depending on the tasks. In these tasks objects far away as well as in the CR should be picked.
The tasks will be shown next to the controller similar to the selfteaching (compare section 4.4), which the user is already aware of. The placement of the tasks is shown in the following figure.

Figure 16: Task shown net to the Controller.

The text for the tasks are saved in a CSV-file, similar to the selfteaching, see section 4.4. The implementation is also comparable to the selfteaching and implemented in the script *showTasks.cs*.

Figure 17: Additional Information on the Wall.

To give the user some more instructions a information board within the supermarket is established, see figure 17. For the first three tasks there are only shown some basic information.
The last tasks will be repeated with every available method. That means, that the user has to pick up the same object five times. This object is placed, so that the user could use CR as well as FR methods (compare section 4.3. The methods are implemented in a fix order which can be seen in table 3. The methods will already be activated as soon as the user presses start.

Number of subtask	Method
1	Close Range: Touch Grab
2	Close Range: Wand Grab
3	Far Range: Extendable Ray
4	Close Range: Proximity Grab
5	Far Range: Raycast

Table 3: Order of methods in the last tasks.

To help the user to figure out what method is activated the name of the method will be shown on the information board as soon as he is in the new scene (compare figure 18).

Figure 18: *Touch Grab* is activated.

For the last tasks a usability questionnaire needs to be answered by the user, after he finishes the task.

5.2 Measurement

5.2.1 Time and Precision Measurement

Vera

As mentioned in section 5.1, the users has to fulfil several tasks which are designed to evaluate various aspects. Thus, three different sets of measurements will be saved automatically in a task. All results are stored in a comma separated value text file that is named after their subtask and stored directly into the unity project folder. These files can be imported in all common statistic applications like *MS Office Excel*, for example. Hence, the provided template is a *Excel* file with the required basic statistic computations and visualisation of the results. Here, the output files could be easily integrated in the designated table fields. The template computes then all useful mean values and standard deviations of the measurements. In other cases, it calculates the percentagewise proportion like the commonness of a method use. Each calculation is displayed in a corresponding diagram. Further, the ones that visualise the mean values includes an error bar that depends on the standard deviation. If necessary, other static computation like a significance test needs to be extended because the calculation highly depends on the study conditions and are not predictable.

The first measuring set is made for the learning procedure and evaluates the affordable learning time of a grabbing method. Their usage time is recorded during the complete learning process and saved in milliseconds when a user presses the stop button.

Afterwards, the first task with its three subtasks is started. Here, the user's method preference of a close or far range task will be evaluated. Thus, the application measures the usage time of every selected method and saves the id of the current

method when the target object is placed successfully on the target area. The listing of this measurement shows significantly which methods were preferred and if a user had comprehend their scope.

Finally, the precision as well as the grabbing and positioning time is measured in the second task for every provided method. The grabbing time is started when a user presses the start button and stopped when the target object is grabbed. At this point, the positioning time is started immediately and stopped when the object is placed on the target area. These time measuring shows the workload and where the users have problems with the method. The additional selecting and positioning error rate is recorded during the whole process, too. Thus, this recorded data gives conclusions about the precision. The success of this task is recorded and the use of the snapping mode as well and so all relevant informations are provided to get all important conclusions about the applicability of each method.

5.3 Questionnaires

Hier wäre es generell schöner einen flüssigere Überleitungen zuschreiben. Meisten benutzt du drei bis viermal hintereinander "The" als Textanfang oder immer gleiche Satzmuster wie z.b. THe evaluation Formula... is intergrated oder "This is done ... !. To not only use measured values to evaluate the interaction methods but also use the participants input, the participants have to fill out two kinds of questionnaires. The first one is the *SUS: a 'quick and dirty' usability scale* usability questionnaire [3]. This questionnaire consists of ten questions. It is given to the user after each task, so each user fills out the questionnaires five times, once for each interaction method. Hier könnte man zum besseren Verständnis etwas genauer sein. Voralem, dass der Fragebogen im zweiten Task direkt nach der Benutzung der jeweiligen Methode ausgefüllt wird.This is done to determine the usability of each method. The investigator has to fill in two extra questions which are a testID (one number assigned to every participant) and which method was used Wann genau muss er das machen? oder andere Reihenfolge in der Beschreibung der Fragebögen.This is done to be able to draw an conclusion based on the used method Vorschlag: Hence, it is possible to draw a conclusion based on the used method.

The second questionnaire is a simulator sickness questionnaire by Kennedy et al. [14] which evaluates the general discomfort caused by the virtual reality application. The investigator has to fill in the testID corresponding to the testID used for the usability questionnaire Hier könnte man auch schreiben: This questionnaire has to be handed to the participant after all tasks are completed and the corresponding testID is entered by the investigator. Oder so ähnlich. Auf jeden Fall nicht dreimal "The" am Satzanfang.

The Durch All ersetzen? questionnaires can either be used in a printed paper form or the provided *Google Forms* can be utilized on a mobile device. When done in *Google Forms*, the result can be imported into *MS Office Excel*. Moreover, eva-

luation templates are provided as *Excel* files with basic statistic calculations and visualisations. The evaluation formula Besser: calculations or computations of the simulator sickness questionnaire are integrated and could be easily applied to larger study groups. The result consists of four values: nausea, optic, disorientation and the total score. The total score has a range from 0 to 160.7, but a theoretical result of over 100 would be so dangerous that an ambulance should be called.
The evaluation formula for the usability questionnaire is also integrated. The resulting value has a range from 0 to 100 whereas a value of 100 is considered for a perfect system. Above 70 are systems with good usability and a score under 50 indicates huge deficit in usability. Vorschlag: In general, systems with a value of more than 70 seems to have a good usability and systems wit a score under 50 have a huge deficit. Wieder Viermal The am Anfang :-(

6 Project Management

This chapter describes the project planning and management of *Interaction Lab*. It is divided into the different project phases. Each division includes all important facts of its project period.

6.1 Project Definition

This section describes the results of the project definition phase in detail. This includes a problem analysis, a list of objectives and requirements, a solution concept as well as a workability analysis.

6.1.1 Problem Analysis

The demand for Virtual Reality (VR) devices and applications increased heavily since the first consumer devices like *HTC Vive* and *Oculus Rift* were released during last years. One main difficulty of the current development of VR-applications is the lack of standardisation of the Software Development Kit (SDK) and interfaces. The most acknowledged suppliers *HTC* and *Oculus* do not work together or force standards for VR application development. Thus, all applications are system related and incompatible with other devices. Accordingly, each device offers different opportunities of interaction methods. These methods can be divided in the acknowledged categories selecting, grabbing, manipulating, movement and indirect controlling via widgets, gestures and voice input. Several suppliers currently offer different devices for interaction. And with focus on the grabbing and positioning methods, the most common are the *Oculus*-HMD, *HTC Vive*-HMD, *HTC Vive*-Controller, data gloves and motion capturing systems for hand-tracking like the *LeapMotion*-Controller.

As mentioned in section 1.1, an interaction laboratory which compares the different interaction methods in a scientific and credible way does currently not exist. Hence, the development of a virtual laboratory is highly requested to compare and test different interaction methods in adequate test environments. User friendly interaction methods which fulfil usability requirements could be improved by researcher which yields to a higher demand of VR devices and application. That will squeeze the profit of VR device suppliers which include those user friendly interaction methods.

6.1.2 Usage Context

According to the previous sections, the required laboratory has mainly two usage contexts. First, it can be used to run scientific studies in VR research or development. Second, it can demonstrate and exemplify the differences of grabbing interactions in education proposes or support the students to develop and test grabbing methods on their own during lectures.

6.1.3 Objective and Requirements

At least two scenes will be realised to provide a laboratory which allows to run scientific and reliable study as well as is useful for the education of students. In the first scene the user will be able to learn the offered grabbing methods. This room provides simple cubes of various sizes which are in different distances from the users. Every cube is moveable and can be placed at every place. Each user is forced to follow the introductions of a self teaching before every offered methods can be tested independently. The current user can only begin with the actual study after every method is trained to ensure equal preconditions.

The second room will be modelled after a supermarket because this model offers various options of grabbing and positioning tasks. In this room the participant will get different tasks which will differ in complexity, distance of grabbing and size of the objects. The user will be able to change the options of grabbing independently but not choose the current method. An optional extension of the project will be another type of task where the user decides which type of method is preferred for this task.

The grabbing methods can be categorised into close range and far range and include the grabbing, rotating, and positioning of an object. Possible types of close range methods, are the actual touching of a movable object to select it or by holding the controller in the proximity of it but without touching it. Another more precise option is the selection with a thin wand in front of the controller. This of collection of methods that includes close human cognition methods as well as less or very accurate ones. The far range interaction will have different options as well. One will be a ray that shoot out of the controller, another one will extend a ray from the head and the third one will extend the arm in the pointed direction. This means the user will be able to point at an object with the controller or to look in the direction of it.

The system offers two measurements and the related saving of the different parameters. First, the duration time is measured for every performed task to compare and validate the performance of the different interaction methods. Second, every single grabbing try of a task will be counted and saved to get a conclusion about the learn-ability, accuracy, and performance.

Furthermore, there will be a questionnaire designed to give the users of *Interaction Lab* an usability evaluation tool at hand. This questionnaire will test parameters as tiring, learnability, self-descriptiveness and fulfilling expectations.

6.1.4 Solution Concept

An interaction laboratory for grabbing and positioning interactions at close or far range will be developed in *Unity*. It includes two test rooms e.g. scenes, where the first is a learning room, in which the users can get familiar with the interaction methods. The second room is designed as a supermarket. This environment was chosen because it offers various possibilities of exercises under changing difficulties

like grabbing small mushrooms, fetching distantly placed tins or putting goods on provided target areas. The exercises are offered in form of a tasks that tells the participant what goods have to be grabbed and repositioned. These various tasks are predefined and cover all difficulties that a type of grabbing method could have. They are displayed on tables which are connected to the controller and could be shown or hide in the controller menu.

All rooms are implemented in Unity and the VR components are controlled by the same framework. Further, the *HTC Vive*-HMD and the corresponding controllers are used to run the interactions, imaging and orientation in the environment. It is planned to realise at least six interaction methods of grabbing and positioning. Additional the complete framework will be compatible with new test scenes and other interaction categories.

The system offers a measurement of the accuracy as well. A time measuring of duration and an error rate for every performed task is planned. Each measuring of every room is automatically saved in an output file which could be easily imported in common statistic tools. Furthermore, there will be a validated questionnaire designed to give the users a usability and simulator sickness evaluation tool at hand. This usability questionnaire will test parameters as tiring, learnability, self-descriptiveness and fulfilling expectations of each method. Whereas the simulator sickness evaluation asks for motion sickness and other system properties of the complete system. All questionnaires are validated and fit the requirement of VR systems and application. The results of each questionnaire will be saved in an output file as well.

6.1.5 Workability Analysis

There are several risks according to the concept in section 6.1.4. First, the measurements could be implemented incompletely or inaccurately. This can be avoided by a thorough testing before the final release with some external test persons. The tasks could be incomprehensible for them as well which have be proved. Another risk is that the system integration of future extensions could cause trouble, too. Therefore, the systems architecture should be designed wisely and consequently to avoid incompatibilities. It is also acknowledged that the implementation might be more costly and complex as recommended. After the implementation is finished, the interaction method performance or validation could be too expensive which results in a higher latency. These circumstances must be observed during the implementation and testing. Due to the high workload of the testing, the time slot for it and the trouble shooting might be underestimated. Another time risk is that there is limited access to the facilities and VR laboratory because of the huge number of running project at the current time.

Nevertheless, the concept is feasible and the project goals could be achieved during the time schedule because all the risk seems to manageable and could be observed during the scheduled testing.

The demand of the students project are satisfied and a financial profitability check is not necessaries due to the fact that the facilities of the university can be used and

no further purchases are affordable.

6.1.6 Project Organisation

The project manager is Vera Brockmeyer who mainly will manage the appointments and facilities as well as to communicate to the outside. The latter is done via email or in a meeting with the concerned persons. Another task is to create and maintain the project plans that includes to keep the overview of the complete project progress and to ensure the milestones. The current state should be hand out weekly to the team in form of an email or a team meeting.

All other team members have their own responsibilities. Anna Bolder is head of the scene building which includes the definition of the general scene design, research, and to inform the project manager about current problems and timing. The latter two points concern each head of a section. The other section is split into the close and far range interactions. The head of close range interaction is Britta Boerner and the other is Laura Anger. Both manage the implementation of their section.

The formal an informal non-verbal communication in the team is done via email with the subject *VR Interface Lab* and a *Google Calendar* is exclusively maintained by the project manager where all team appointments are intercalated. This calendar shows the availability of all team members and the VR laboratory, too. More complex problems or team decisions are made in the weekly team meeting with stringently required appearance. Due to the requirements and availability of the team members, the meeting is held via *Skype* or in personal.

All files belonging to the project are organized in a cloud folder of *Google Drive* or in two *GitHub* repositories. The first one is for all *LaTex* files and the second manage the complete framework. Whereas the cloud folder contains presentation files, graphics and images, To-Do-List, papers and more.

The required facility is one of the VR laboratories of the faculty which should have a minimum size of 15qm and be located in the university building. These laboratories have a complete *HTC Vive* system and a compatible computer (see Table 1).

6.2 Project Planning

This section list all required projects plans like a work breakdown structure, the work packages, a capacity and cost plan as well as a quality plan.

6.2.1 Work Breakdown Structure

The work breakdown structure (see Figure 20) is split in four groups. First, the project planning which contains the actual project management as well as the research work packages. Second, the implementation group includes the development of the laboratory environment and the six interaction methods. Third, the evaluation

contains the work packages of tasks including measurements and the questionnaires. Finally, the last group includes the documentation and project profile packages.

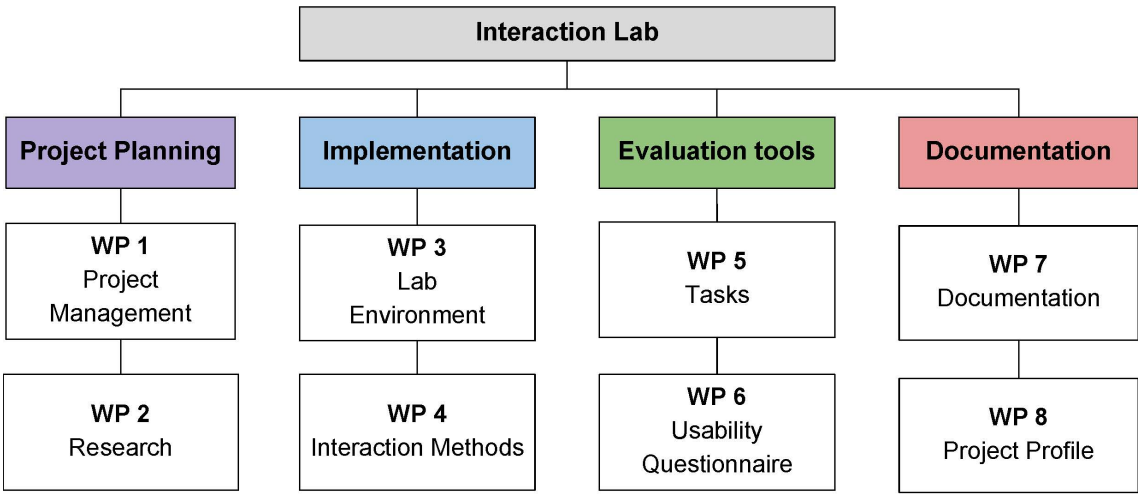

Figure 19: Work Breakdown Structure

6.2.2 Workpackages

A complete listing of the work packages according to Figure 20 could be seen in the Appendix A 10.1.

6.2.3 Project Schedule

Figure 20 shows the project schedule with the timing of all work packages. The most important deadlines are the project plan, research, first and second prototype as well as the release deadline. First, the project plan should be finished until 13th April 2017. Second, the research deadline is two and a half weeks later at the 27th April 2017. Followed by both prototypes which are set for the 31st may and 30th June 2017. The final release deadline is at the 15th July 2017.

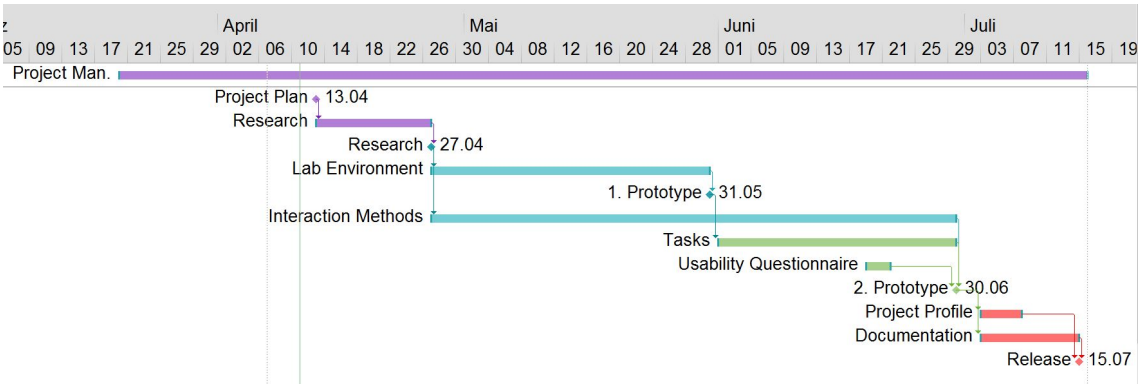

Figure 20: Project Schedule

6.2.4 Capacity Plan

The capacity plan lists the estimated time resources of each team member, the availability of the required laboratory with the *HTC Vive* system and the SLR camera for the documentation.

Resource	Estimated [h]	Actual [h]	Buffer [%]
Laura	166	180	7.78
Anna	170	180	5.56
Vera	182	180	-1.11
Britta	168	180	6.67
Total Personal	686	720	4.72
VR Lab 1	173	152	-13.82
SLR Video Camera	8	8	0.00

Table 4: Capacity Plan of personal and facility resources.

6.2.5 Cost Plans

The cost plan of personal resources in Table 5 gives a more detailed overview of the estimated time resources per work package. It is structured into the cost of each team member and the total cost of a work package. Whereas, the cost plan of the facility resources offers an overview of the material costs and their providers.

Name	**Laura [h]**	**Anna [h]**	**Vera [h]**	**Britta [h]**	**Cost WP [h]**
Project Management	12	12	45	12	81
Research	10	10	10	10	40
Lab Environment	0	55	32	10	97
Interaction Methods	65	0	0	65	130
Tasks	15	45	30	15	105
Usability Questionnaire	8 0	0	0	8	
Documentation	45	45	55	45	190
Project Profile	8 0	0	8	16	
Total Cost	**163**	**167**	**172**	**165**	**667**

Table 5: Cost Plan of personal resources.

Ressourcen	**Quantity**	**Unit Price [EUR]**	**Cost [EUR]**	**Act. Cost [EUR]**	**Comments**
HTV Vive	1	899	899	0	Price of Vive Online Shop, provided by TH Cologne
SLR Camera	1	500	500	0	estimated, provided by team member
VR Computer	1	1800	1800	0	offer from Computer Shop, provided by TH Cologne
MS Office	1	149	149	0	provided by TH Cologne
MS Project	1	195	195	0	provided by TH Cologne
3D assets of Food Beverages	2	15	30	0	Steam
Total			**3573**	**0**	

Table 6: Cost Plan of facility resources.

6.2.6 Quality Plan

Quality Goal	Criteria	Method	Controlling
Immersion and Presence	no distraction by real world	Presence questionnaire by Witmer and Singer; questions according to sound and haptic will left out	use and evaluate brief questionnaire during functionality test
Low Latency	under 20ms	keep the scenes as simple as possible	show the frames per second in the Unity console
No dropouts	no black frames or errors in the unity project	no expensive or parallel calculation	visual testing
Understandability of tasks	correct task performance by test user during functionality tests	use common objects and tasks; brief and precise task descriptions	use brief questionnaire during functionality test
robust system	Vive System does not crash during process	no expensive calculations; control and calibration of hardware; analyse and solve directly problems	observe during development
Correct saving of Usability Questionnaire Response	entire answers of study participants are correctly saved in an file	Use of Google Forms	functionality test and controlling of completeness and correctness
Understandability of Usability Questionnaire	no questions or uncertainties of test users	validated Usability Questionnaire	use and evaluate brief questionnaire during functionality test

Table 7: Quality Plan

6.3 Project Execution

The following sections summarizes the project progress as well as the problems that caused a delay of the schedule.

6.3.1 Project Progress

In general, the project progress went well and most of the project goals were achieved and are running as expected. The final version of the *Interaction Lab* is a complete working application that includes all required tools for a scientific study. It contains a learning room, a supermarket test scene with implemented task and measuring, the implementation of five different interaction methods to test as well as a completely set of validated usability questionnaires and excel templates for the evaluations.

However, some of the project goals were too ambitious for the resources capacity in the scheduled time window. Thus, not all project goals were completely achieved. First, the Gogo interaction method and the HMD raycast could not be developed due to the problems mentioned in section 6.3.2. Second, the number of tasks needed to be reduced significantly because it turned out that they are to time consuming during testing. Accordingly, only one task was implemented with all provided interaction methods and three tasks to figure out the preferences of the users for various kind of ranges and object sizes. Third, an additional goal was set after the first prototype. A self teaching mode has been added to the learning room to ensure an equal introduction of the methods and task for each test user.

The actual workload of most work packages covers their estimation beside some problems which are described in detail in section 6.3.2. All team members had a similar work load and the task allocation work efficient and well. Each team member knew their current tasks at every time and had an overview of the complete progress. This overview was achieved with weekly meetings via *Skype* or if affordable in personal. Furthermore, every few weeks we had a meeting with Prof. Grünvogel to give him an update and discuss current problems and their solution.

6.3.2 Problems and Solutions

A first problem was the additional work effort of the need to overwork all project plans which causes a one week delay. It should be noted that the planning was to detailed in the beginning which cause some unnecessary time. Thus, the research phase needed to be shorten and placed on hold for this week. This step brought the project back on time. Another delay of one and a half week causes the inevitable unity and *HTC Vive* system updates that were more difficult than expected. These technical interruptions occur several times when the laboratory was changed due to limited capacities of the designated VR Lab 1 or the periphery of the laboratories was not present. A further time consuming circumstance was the need of a recalibration of the VR system at every time the laboratory was changed.

The VR system caused trouble during the implementation of the HMD Raycast method, too. It turned out that the tracking of the HMD lacks precision and the ray drifts at the edge of the view field. At this point, the number of tasks that had to be done to guarantee a working application until the release was challenging. A main reason for the challenging timing was the additional self teaching which was set after the first prototype. It was much complex and time consuming than expected in the beginning and we decided to concentrate on a working application and discard

the HMD Raycast. However, it seems that the implementation of the Gogo method was too complex for this time schedule as well and it was changed to the Extenable Ray method which could be implemented successfully during the calculated period.

This decision and the resulting choice of methods had the advantage that their basic structure is adoptable. Regarding to this realisation, it could be said that the split of responsibility for Far and Close Range in the project organisation was a mistake. Accompanied with the decision to create a single C# script per method. This plenty of scripts caused a lot of trouble during their integration in the final application but these problems were solving by using only one script for similar methods like the ray methods, for example.

7 Reflection

Finally a running and nearly complete application was developed during this project. The test environments in form of the learning room and supermarket are working without a latency or blackouts. They provide an effective virtual laboratory that can be used by researchers or for educational purposes. Both environments integrate five perfectly working grabbing interaction methods. Firstly the close range methods Touch, Proximity and Wand Grab are implemented and the first two extends an optional snapping mode. Secondly the two far range methods Raycast and Extendable Ray are successfully integrated into the application.

All measurements of the tasks are saved automatically in output files that could be evaluated with the provided *MS Office Excel* template. The usability questionnaire forms could be accessed by every mobile device or computer and their results are saved automatically as well. They are offered in English and German language. Certainly, they could be evaluated with the template, too. Additional, all required consent forms are delivered in both languages with the application as PDF file.

As mentioned in section 6.3.2, the HMD Raycast and Gogo method could not be realised but unfortunately there are more goals that are not yet achieved or do not work completely without faults. Firstly currently the self teaching mode only works if the introductions were strictly followed. In case of a mistake it will recover only when a new method is selected. However there is no reminder of the usability questionnaires during the last task for the study supervisor. Therefore he is forced to be totally focused during a test. Another failure is the missing question of the VR experience in these questionnaires and the lack of the general input of the test id in the application.

There are some technical mistakes as well which are not yet fixed. A main problem is the potential loss of script links in the unity project when the laboratory is changed or a git repository is pulled. A further problem is the lack of an easy method or scene extension in the application due to the complex measuring and interaction script architecture. Other small and rare troubles are the visible ray in front of transparent surfaces or object that fall sometimes through the ground.

8 Self-Assessment

This chapter includes the detailed descriptions of the team members work tasks and responsibilities as well as their personal reflections

8.1 Anna Bolder

Working Hours: 116

Written Sections: 2.1, 3.2.4, 4.1 (subsections included), 4.2, 4.4 and 5.1

Responsibilities:

My main task in this project were the menu as well as the displaying of different informations, including selfteaching and tasks, for the user.
At the beginning I created a shared *Unity* project and added the Steam VR plug-in. In this project I created different rooms for the testing of the close and far range interactions. In addition to that I designed the learning room, described in section 4.1.1, and discussed with V. Brockmeyer the structure of the supermarket (compare section 4.1.2). Furthermore, I researched the options to switch between the rooms and created the basis for the script *TargetTest.cs*.
After the creation of the basic scenes I researched for a solution of an radial menu on the controller. I implemented this with the "VRTK" plug-in, see section 3.2.4. To collect all functions of the menu buttons in one script I created the script *menu.cs*.
As soon as all interactions were implemented L. Anger and me integrated the interactions in the menu and all scenes. Here I helped to simplify the switching between different interactions with a ray.
In the last weeks I implemented the selfteaching together with V. Brockmeyer. This was time-consuming because the counter has to be increased or set on different parts of the project and the height of the canvas has to be set for every step.
With the knowledge of the selfteaching L. Anger and me implemented the tasks in the supermarket. Here we had to short the actual tasks texts.
With L. Anger I ran test surveys with five participants.

Self-Reflection:

In my opinion the project ran quiet well. Everybody had their part of the project and we also helped each other.
I underestimated some parts of my responsibilities. First the selfteaching took more time than expected. There were a lot more steps to do than I thought and each step had to be tested with respect to the visualisation. In addition to that the combination of all interactions in one menu and scene was more complicated. We did not think about the affection of the implementations among one another or that parts of the interaction have to be loaded at the start of the scene to work correctly.

8.2 Vera Brockmeyer

Working hours: 170

Written Sections: 1(subsections included), 5.2, 6 (subsections included), 7, 9

Responsibilities:
My main responsibility was the planning, organisation and management of the project. This includes the making of all project plans and the definition of it as well as to monitor the project progress and to guarantee the compliance of the schedule. Another task was the management of the internal and external communication as well as representation. This includes the making of appointments for meeting and required facilities as well as the steadily summary of the current project state. However the distribution of the work tasks was my responsibility, too.

Beside the project management, I supported Anna Bolder with the implementation of the VR environment. In detail I build the supermarket scene with 3D assets from the Steam Store and did the research for the tasks. Furthermore, I wrote all texts of the tasks and self-teaching and supported Anna Bolder with the implementation of the ladder. However, my other main task was the implementation, integration and output of the measurement. This includes the making of an *MS Office Excel* template to evaluate the measuring's.

Self-Reflection:

I feel that the project worked very well without big complications that afforded a time consuming solution or plenty of unnecessary work. According to the planning, it turned out that I planned to much in detail at the beginning which costs a lot of time resources and results in a complete refactoring of all plans. Another planning failure was the lack of enough research time during the complete project.

The communication worked quite well beside some small periods in the beginning where I planned too much and long team meetings and one where should have been more team meetings in May to avoid little misunderstandings. Beside this, I feel that I distributed the work tasks efficiently with regards to the skills of the team members. The external communication worked quite well beside some intermediately state of knowledges due to meetings without me. Especially the translation of the usability questionnaires into german and the parallel decision to run the whole application in the english language because of the tight schedule.

According to my task during the development of the VR environment, I implemented a working measuring with all required output files. Nonetheless the architecture of it do not allow an easy extension of further interaction methods or scenes.

Briefly, the project team worked very good together and solved each problem directly with focus on the entire application without rest on details. The results correspond to our expectations besides some little failures.

8.3 Britta Boerner

Britta

Working hours: xx

Written Sections: 2.3, 3 (subsections included), 5.3

Responsibilities:

I was head of close range interactions. This included researching as well as implementing the interactions. Further responsibilities where researching questionnaires for usability as well as simulator sickness in virtual reality.
The first few weeks of the project were used to research different interaction methods. After this phase I started the implementation. L. Anger and I worked together on the interactions. This was a benefit as the close range interactions and far range interactions have similarities that maybe were not as obvious in the planning stage. We could also benefit from each other as we could learn from each others mistakes and putting two minds on solving problems makes it easier to stay motivated. After this phased I researched different questionnaires. I found two standard questionnaires, one for usability in general and one for simulator sickness in virtual reality environments. I also created the consent form and wrote the introduction, which is given to every participant of the study.
After the small scale test I prepared a template evaluation of the results which can be used for further studies.

Self-Reflection:

I think over all the project went well. Due to a lot of planning prior to actually starting programming we could work quite structured. Of course, problems arose during programming but I was able to solve those also because we all worked well as a team and someone would always help. For future projects I think I would research the implementation of the interaction more in-depth so that during the programming process not so many questions come up that could be answered before hand.

8.4 Laura Anger

Working Hours: 125

Written Sections: 2, 2.2, 4 and 4.3 (subsections included)

Responsibilities:

Besides mandatory tasks, like for example doing research on specific problems and helping with the project management, I was mainly working on the implementation of the interactions. At the beginning of the project we divided the interactions into CR and FR (compare section 4.3). At that time I became responsible for the FR methods of the *Interaction Lab*. As it turned out it was not wise to distinguish between those two categories strictly, the borders of the jurisdiction of B. Boerner (Head of CR interactions) and me became more and more blurred. That is why we ended up working on most of the interaction methods together. This was more expedient because, for example, the *Wand Grab* (compare section 4.3.3, which is a CR interaction, is very similar to the FR interactions *Raycast* (compare section 4.3.4) and *Extendable Raycast* (compare section 4.3.5). Because B. Boerner went into holidays rather at the end of the implementation phase, I implemented the *Wand Grab* on my own.
Together with A. Bolder I integrated the implemented interaction methods into the actual scenes (compare section 4.1) of the *Interaction Lab*. We also took care of the visualisation of the tasks for the user. Therefore we created a display on the wall of the supermarket as well as next to the controller. On both of the displays the corresponding tasks are presented for the user.
I was also involved in creating the project video and all images for this documentation.
A. Bolder and I ran a test survey with 5 participants, which is described in section 5.

Self-Reflection:

All in all I felt like the project run smoothly and well planned. Of course I was confronted with unknown terrain as I never had implemented interactions of any kind. Maybe this was why I first started to create one script per interaction method. In the end of the project I noticed that both Raycast methods and the *Wand Grab* work after the same rules so I combined them in one script. If I had done that earlier I could have avoided a lot of trouble for me and A. Bolder as we needed to integrate them in to all scenes, separately first.
But all in all the workflow was well structured as anyone had her own work area. Whenever this areas collides (for example the integration of the interactions into our scenes), we managed that both responsible people worked together on this task.

9 Conclusion

The project progress went well with focus on a working application and without bigger complications. These were directly discussed and solved by the team. Each team member make one's contribution in nearly equal percentages. Hence, the result is a working application for the evaluation of grabbing interaction methods with the *HTC Vive* controllers. Most of the project goals were achieved beside the missing HMD Raycast and Gogo interaction method, the reduction of user tasks and the missing extensibility. Nevertheless, it has to mentioned that there is still a potential for improvement like the visible ray on transparent surfaces or the objects that sometimes fall through the ground.

10 Appendix

Vera

10.1 Appendix A

<table>
<tr><td>Project name:
Interaction Lab</td><td>WP- Nr.:
1</td><td>WP-Name:
WP 1: "Project Management"</td></tr>
<tr><td>Start:
20.03.2017</td><td>End:
15.07.2017</td><td>Responsible Team Member:
V. Brockmeyer</td></tr>
<tr><td colspan="3">Results:
• Project definition
• Project plans
• Resource and material reservation
• Weekly reports
• Milestone presentation
• Project process evaluation and reflexion</td></tr>
<tr><td colspan="3">Tasks:
• Organisation of meetings
• create project definition, project order, list of requirements, project organization
• prepare milestone and final presentations
• make Work breakdown structure (WBS), Project schedule, Capacity plan, Cost plan and Quality plan
• manage internal and external communication
• give weekly reports
• delegate and supervise tasks
• team meeting to reflex project process, compare plans and suggested cost with actually effort
• organise all required resources</td></tr>
<tr><td colspan="3">Requirements:
• MS Project
• Internet
• team meetings</td></tr>
<tr><td colspan="2">Signature Project manager</td><td>Signature Responsible Team Member</td></tr>
</table>

Figure 21: First Workpackage: Project Management

Project name: Interaction Lab	WP- Nr.: 2	WP-Name: WP 2: "Research"
Start: 13.04.2017	End: 26.04.2017	Responsible Team Member: L. Anger, A. Bolder, B. Boerner und V. Brockmeyer
Results: • Draft of implementation • knowledge of the required theory • Measurement methods • UML of Unity Project		
Tasks: • Research Far Range Interaction Methods • Research Close Range Interaction Methods • Research Test Scene Implementation and Assets • Research Measurement Methods and related Tasks • team meeting to plan integration of total system • create UML of Unity Project		
Requirements: • Project Definition • Project Plans • Internet • team meeting		
Signature Project manager		Signature Responsible Team Member

Figure 22: Second Workpackage: Research

Project name: Interaction Lab	WP- Nr.: 3	WP-Name: WP 3: "Implementation Environment"
Start: 27.04.2017	End: 30.05.2017	Responsible Team Member: A. Bolder

Results:

- Setup of the Unity Project
- Setup of the GIT repository
- one simple test room to learn and test the different interactions
- one test room build as supermarket to implement tasks for user
- GUI
- First functionality testing

Tasks:

- Setting of the concrete Unity version
- Download and integrate SteamVR
- Build simple test room with cubes as big as the real room
- Build supermarket, which is bigger than the actual real room to force the user to use far range interaction
- ensure the user will not walk further than possible
- Implement switching between rooms
- Implement structure and position of the menu
- Implement different buttons and the functions behind
- Integration of the interaction methods
- Implement interface for further test scenes
- upload to GIT repository
- Perform first functionality test with 5 persons

Requirements:

- WP 1 Project Management
- WP 2 Research
- Plugins for Unity (Steam VR, 3D assets)
- VR Laboratory 1
- GIT repository
- Supporter: Vera Brockmeyer

Signature Project manager	Signature Responsible Team Member

Figure 23: Third Workpackage: Implementation of the Environment

<table>
<tr><td>Project name:
Interaction Lab</td><td>WP- Nr.:
4</td><td>WP-Name:
WP 4: "Implementation Interaction Methods"</td></tr>
<tr><td>Start:
27.04.2017</td><td>End:
29.06.2017</td><td>Responsible Team Member:
L. Anger (Far Range) and B. Boerner (Close Range)</td></tr>
<tr><td colspan="3">Results:
• Implementation of Close Range Methods
○ grab by contacting the object with controller
○ controller is in proximity with the object
○ controller is in proximity with the object and object snaps to the controller
• Implementation of Far Range Methods
○ Pointer to grab and control object
○ Pointer from the HMD to the Object
○ extension of the user's arm</td></tr>
<tr><td colspan="3">Tasks:
• implement scripts for each interaction methods</td></tr>
<tr><td colspan="3">Requirements:
• WP Research
• VR Room 1
• Git Repository</td></tr>
<tr><td colspan="2">Signature Project manager</td><td>Signature Responsible Team Member</td></tr>
</table>

Figure 24: Forth Workpackage: Implementation of the Interaction Methods

Project name: Interaction Lab	WP- Nr.: 5	WP-Name: WP 5: "Tasks"
Start: 01.06.2017	End: 29.06.2017	Responsible Team Member: A. Bolder
Results: • implementation of three tasks • integration of all task in the system • completed functionality test with 5-10 persons • output of measurements • evaluation of prototype and questionnaire • solution concepts		
Tasks: • implement training in first test room • implement three tasks in supermarket scene • design questionnaire of functionality test • run functionality test • evaluate questionnaire • implement solution concepts • implement final solutions		
Requirements: • VR Laboratory 1 • 1. Prototype • Research • 5-10 persons • Test schedule • brief questionnaire for test persons • Supporter: Vera Brockmeyer		
Signature Project manager		Signature Responsible Team Member

Figure 25: Fifth Workpackage: Tasks

Project name: Interaction Lab	WP- Nr.: 6	WP-Name: WP 6: "Usability Questionnaire"
Start: 19.06.2017	End: 21.06.2017	Responsible Team Member: V.Brockmeyer
Results: • Standard Usability Questionnaire for Interaction Methods		
Tasks: • research common Usability Studies • make Google Form to save automatically the results		
Requirements: • Google Form • WP 2: Research		
Signature Project manager		Signature Responsible Team Member

Figure 26: Sixth Workpackage: Usability Questionaire

<table>
<tr><td>Project name:
Interaction Lab</td><td>WP- Nr.:
7</td><td>WP-Name:
WP 7: "Documentation”</td></tr>
<tr><td>Start:
03.07.2017</td><td>End:
15.07.2017</td><td>Responsible Team Member:
L. Anger, A. Bolder, B. Boerner und
V. Brockmeyer</td></tr>
<tr><td colspan="3">Results:
• Two printed versions of the documentation of the “Interaction Lab”-Project
• Final presentation</td></tr>
<tr><td colspan="3">Tasks:
• Introduction and Motivation
• State of the art
• Description of Hard- and Software
• detailed description of complete System
• Evaluation / Results
• Project Management
• Reflexion
• Conclusion
• prepare final presentation</td></tr>
<tr><td colspan="3">Requirements:
• Latex
• MS PowerPoint
• This work package needs to be finished after all other work packages are ready
• It might be possible and useful to start before the final completion of all other work packages
• GitHub Repository for Latex files</td></tr>
<tr><td colspan="2">Signature Project manager</td><td>Signature Responsible Team Members</td></tr>
</table>

Figure 27: Seventh Workpackage: Documentation

Project name: Interaction Lab	WP- Nr.: 8	WP-Name: WP 8: "Project profile, project video and images"
Start: 07.07.2017	End: 07.07.2017	Responsible Team Member: L. Anger and B.2 Boerner
Results: • brief Project description and profile as Word document • 2-5 expressive image in adequate Quality • Image and Tutorial Video of System		
Tasks: • write brief Project description and profile in Word document • capture expressive images of project • make an image and tutorial video of system		
Requirements: • SLR Video camera • 2. Prototype • VR Laboratory 1 • MS Word		
Signature Project manager		Signature Responsible Team Member

Figure 28: Eighth Workpackage: Project Profile, Video and Images

References

[1] R. J. Adams and B. Hannaford. Stable haptic interaction with virtual environments. *IEEE Transactions on Robotics and Automation*, 15(3):465–474, Jun 1999.

[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] John Brooke et al. Sus-a quick and dirty usability scale. *Usability evaluation in industry*, 189(194):4–7, 1996.

[4] Doc-Ok.org. Lighthouse tracking examined. http://doc-ok.org/?p=1478. Visited: 30. june 2017.

[5] Unity Documentation. Boxcollider. https://docs.unity3d.com/ScriptReference/BoxCollider.html. Visited: 12. july 2017.

[6] Unity Documentation. Ray. https://docs.unity3d.com/ScriptReference/Ray.html. Visited: 16. july 2017.

[7] Unity Documentation. Spherecollider. https://docs.unity3d.com/ScriptReference/SphereCollider.html. Visited: 16. july 2017.

[8] Unity Documentation. Transform.setparent. https://docs.unity3d.com/ScriptReference/Transform.SetParent.html. Visited: 16. july 2017.

[9] Mikael Eriksson. Reaching out to grasp in virtual reality: A qualitative usability evaluation of interaction techniques for selection and manipulation in a vr game, 2016.

[10] HTC. Htc vive. https://www.vive.com/. Visited: 25. july 2017.

[11] HTC. HTC Vive – VIVE READY COMPUTERS. https://www.vive.com/eu/ready/. Visited: 18. april 2017.

[12] Jason Jerald. *The VR Book: Human-Centered Design for Virtual Reality*. Association for Computing Machinery & Morgan und Claypool, New York, NY, USA, 2016.

[13] Hit Jones. Unity asset store: Realistic cardboard boxes (pbr, hq). https://www.assetstore.unity3d.com/en/#!/content/58749. Released: march 2016, Visited: 17. may 2017.

[14] Robert S Kennedy, Norman E Lane, Kevin S Berbaum, and Michael G Lilienthal. Simulator sickness questionnaire: An enhanced method for quantifying simulator sickness. *The international journal of aviation psychology*, 3(3):203–220, 1993.

[15] Sangyoon Lee, Jinseok Seo, Gerard J Kim, and Chan-Mo Park. Evaluation of pointing techniques for ray casting selection in virtual environments. In *Third international conference on virtual reality and its application in industry*, volume 4756, pages 38–44, 2003.

[16] Lucie Lescuyer. Unity asset store: Free food pack. https://www.assetstore.unity3d.com/en/#!/content/85884. Released: april 2017, Visited: 3. may 2017.

[17] Jonathan Lin and Jürgen P Schulze. Towards naturally grabbing and moving objects in vr. *Electronic Imaging*, 2016(4):1–6, 2016.

[18] Sysdia Solutions Ltd. Unity asset store: Vrtk - virtual reality toolkit - [vr toolkit]. https://www.assetstore.unity3d.com/en/#!/content/64131. Released: June 2016, Visited: 17. may 2017.

[19] Sysdia Solutions Ltd. Vrtk - virtual reality toolkit (read me). Visited: 17. may 2017.

[20] Mobile World Congress. Mobile World Congress. https://www.mobileworldcongress.com/. Visited: 30. june 2017.

[21] OneCoinGames. Unity asset store: Food & grocery items - low poly. https://www.assetstore.unity3d.com/en/#!/content/75494. Released: november 2016, Visited: 3. may 2017.

[22] IVAN POUPYREV and TADAO ICHIKAWA. Manipulating objects in virtual worlds: Categorization and empirical evaluation of interaction techniques. *Journal of Visual Languages & Computing*, 10(1):19 – 35, 1999.

[23] Steam. Steam VR Internetauftritt. http://store.steampowered.com/steamvr?l=german. Visited: 21. april 2017.

[24] Technische Hochschule Chemnitz. Human-Machine Interaction Lab. https://www.tu-chemnitz.de/mb/ArbeitsWiss/forschung/labore/human_machine_interaction_lab. Visited: 28. june 2017.

[25] Unity Technologies. Unity. https://unity3d.com/de. Visited: 8. july 2017.

[26] Unity Technologies. Unity – Multiplatform. https://unity3d.com/unity/multiplatform. Visited: 14. july 2017.

[27] Unity Technologies. Unity – Public Relations. https://unity3d.com/public-relations. Visited: 14. july 2017.

[28] Unity Technologies. Unity – VR Overview. https://unity3d.com/de/learn/tutorials/topics/virtual-reality/vr-overview. Visited: 14. july 2017.

[29] Valve. Valve Software. http://www.valvesoftware.com/. Visited: 30. june 2017.

[30] Oculus VR. Oculus rift. https://www.oculus.com/rift/. Visited: 23. july 2017.