Laurenz Theuerkauf, BSc

Authoring and Training Human Motion in Virtual Reality

MASTER'S THESIS

to achieve the university degree of
Diplom-Ingenieur

Master's degree programme
Software Engineering and Management

submitted to

Graz University of Technology

Supervisor

Univ.-Prof. Dipl.-Ing. Dr.techn. Schmalstieg Dieter
Institute for Computer Graphics and Vision

Ass.Prof. Dipl.-Ing. Dr.techn. Kalkofen Denis
Institute for Computer Graphics and Vision

Graz, Austria, Jan. 2019

Abstract

Training in virtual reality shows promising benefits for anyone who wants to train at home without the direct help of a professional trainer. It can improve the learning experience, by decreasing the necessary time required to learn new motions. Moreover, accuracy is improved while performing by providing reliable and accurate feedback in real-time to the user. All of these benefits can be enjoyed from the comfort of your own living room. We create a virtual reality (VR) training environment, focusing on visualizing and authoring full-body motions. In this thesis we investigate and evaluate the advantages of first-person perspective (1PP) and third-person perspective (3PP) and their respective validity in different use cases. We introduce a low-cost motion capture solution, fusing sensor data from a Microsoft Kinect and an HTC Vive Lighthouse tracking, to create stable and accurate avatar animations in real-time. Using sensor fusion in conjunction with an inverse kinematics (IK) system, we can greatly improve the animation quality, when compared to a system solely using the Microsoft Kinect skeleton tracking. The same tracking solution is used to record motions by instructors and we also use the recorded motion capture for our visual guidance systems. We further explore visualizations to engage users and help them to accurately and reliably learn complex motions, using our system, by providing them visual guidance and real-time feedback to intervene and correct their poses. Our visual guides include path instructions with 3D glyphs, an animated virtual instructor and an avatar-in-miniature.

Kurzfassung

Training in der virtuellen Realität bietet allen, die zu Hause ohne einen professionellen Trainer trainieren wollen, vielversprechende Vorteile. Es kann die Lernerfahrung positiv beeinflussen, indem es dabei hilft, die Zeit zum Erlernen neuer Bewegungen zu reduzieren. Zudem wird die Genauigkeit der Bewegung durch zuverlässiges und exaktes Echtzeit-Feedback erhöht. Wir erstellten eine virtuelle Trainingsumgebung, mit dem Fokus auf der Visualisierung und dem Erstellen von Ganzkörperbewegungen. Hierbei werden die unterschiedlichen Vorteile einer *1PP* und einer *3PP* und ihrer jeweiligen Anwendbarkeit in unterschiedlichen Anwendungsszenarien untersucht und evaluiert. Wir stellen eine kostengünstige Motion-capture Lösung vor, durch Sensorfusion der Trackingdaten einer Microsoft Kinect und dem Lighthouse Trackingsystem einer HTC Vive, um einen Avatar in Echtzeit zuverlässig und genau zu animieren. Durch die Einführung einer Sensorfusion in Kombination mit einer inversen Kinematik (IK) Lösung, sind wir in der Lage die Qualität der Animation deutlich zu verbessern, verglichen mit Systemen, welche nur auf das Tracking der Microsoft Kinect zurückgreifen. Das selbe Trackingsystem kommt zum Einsatz, um die Bewegungen von Instruktoren aufzunehmen und die Daten dieser Aufzeichnungen für unsere visuellen Anleitungen zu nutzen. Wir untersuchen weitere Visualisierungen, um den Nutzer zu motivieren und ihm dabei zu helfen, komplexe Bewegungen zuverlässig und akkurat mit Hilfe unseres Systems zu erlernen, indem wir visuelle Anleitungen nutzen und dem Nutzer Feedback in Echtzeit geben, um somit eingreifen zu können und seine Bewegungen und Posen zu korrigieren. Unsere vorgestellten visuellen Hilfestellungen inkludieren 3D-Glyphen, einen animierten virtuellen Instruktor und einen avatar-in-miniature.

Affidavit

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly indicated all material which has been quoted either literally or by content from the sources used.

The text document uploaded to TUGRAZonline is identical to the present master's thesis dissertation.

Date	Signature

Acknowledgments

First and foremost I would like to thank Univ.-Prof. Dipl.-Ing. Dr.techn. Dieter Schmalstieg and Ass.Prof. Dipl.-Ing. Dr.techn. Denis Kalkofen for their guidance throughout the creation of the thesis. Without the continuous support and great suggestions from Ass.Prof. Dipl.-Ing. Dr.techn. Denis Kalkofen this thesis would have not been possible. Also, I would like to thank Dipl. Ing. Peter Mohr-Ziak for his support throughout the creation of the thesis and his valuable input. I also thank the institute for the opportunity to work at the institute. I also thank my fellow students and friends for the fruitful discussions and coffee breaks. I thank my family for their financial support during my studies. Lastly, I thank Julia, my better half for her moral support and proof reading the thesis.

Contents

List of Figures

List of Tables

1 Introduction

Contents

1.1 Motivation

Learning new body movements, either for the purpose of sports, or physical rehabilitation, can be expensive, time consuming and requires constant feedback and intervention by a professional trainer. To reduce the cost factor, many people resort to tutorials provided for free by community portals. These tutorials are in most cases two dimensional, either using video instructions, or depictions. Throughout history, two-dimensional instructions like manuals were the most used and best way to communicate with the user. Creating instructions with depictions in 2D is a challenging task, but so is understanding and abstracting them. Conveying the desired information from 2D space to 3D space is difficult for humans and is prone to errors.

The most common form of a learning instruction is in video form. Nowadays, these tutorials are mostly provided on media sharing platforms in large amounts and readily available to anyone in the world with access to the internet. Time or financial constraints make these kinds of training attractive to a wide audience. People offer tutorials in a broad spectrum of fields, like sports, technical skills, arts and crafts.

With the upcoming of user portals like YouTube or Pinterest, where a user can share vast quantities of information, tutorials for every situation imaginable became available. The available tutorials range from easy tutorials for beginner levels, up to complex ones targeted at professionals. In the case of learning human motion, there are many tutorials available, for example for sportive activities like martial arts, yoga, gymnastics and dancing. Users also created content for re-creative motions or curative treatment like physiotherapy, or motor rehabilitation for patients after serious injuries or strokes. These complex motions are often difficult to follow and with the lack of a real instructor often times wrongly imitated because the users has no feedback on their performance and no intervention takes place to correct movements, when doing them wrong. These mistakes can lead to a bad learning experience, frustration, or in the case of physiotherapy or physical training, even risk injuries. For motions, the spectrum covers from full body, to pinpoint accurate motions. We will be focusing on full-body and upper-body motions, used in martial arts, dancing, yoga, physiotherapy and sports. We implemented a virtual TV-set allowing for video playback, similar to the experience one would have at home standing in front of his or her TV-screen.

We want to introduce a system to convey three-dimensional human joint information from training videos and use that information in a cost-effective user-friendly 3D virtual environment. We track a human body with a motion capture system and convert their joint information to 3D joint information. We use the motion tracking system to animate an avatar representing the user in real-time in our virtual environment. The motion tracking can also be used to record the movement of a person, like a professional trainer and apply these recordings to a virtual instruction, assisting the user later via visual guides to learn a movement. This data can be used, for example, to animate an avatar to act as a live instructor and to measure and compare it to the live user's performance. By doing so, we can evaluate a user's performance and give live feedback to either correct or reward him, or her for accurate or wrong movements. We want the user to get an immediate feedback and the ability to correct his, or her pose all by himself, or herself, reducing the need of a trainer who may not be available at all desired times. A virtual training system can greatly enhance the learning experience and complement the training with a professional, by providing a system, usable at home to repeat learned movements in a session led by a professional.

A virtual reality (VR) environment allows us, as the designer of a training system more freedom than the real world. As mentioned above, we can augment the user and support him or her with immediate feedback. We also obtain the freedom to change the perspective and freely move around the scene. We can therefore move the user out of his or her body and give him or her a third-person experience. Viewing their body from this third-person perspective (3PP), improves the visibility of their own body and the understanding of their body pose. In this thesis, we want to evaluate the advantages and limitations of a *3PP* for arm motions, following two recorded instructions viewed from both, a first-person perspective (1PP) and a *3PP*.

1.2 Structure of the thesis

We will introduce the reader to *VR* in general in chapter 2 and how it evolved over time, from a simple stereoscope using hand-drawn pictures for each individual eye to create the illusion of depth, to today's head-mounted-displays using small high density displays with distorting lenses. In chapter 2.2 we give a comprehensive review on related *VR* training environments, using similar approaches like we did. Most of these systems also use the Microsoft Kinect for tracking the user and either an head-mounted display (HMD) or a CAVE-like system. We also investigate augmented reality (AR) systems like 'YouMove' and 'Physio @ Home' [35] [2], where the users' augmentation is done in mirror setups. We also take a look at the importance of an avatar within a virtual space, the effect of virtual body ownership (VBO), and how changing the users' perspective can impact the performance and immersion for them in chapter 2.3. Lastly, we will investigate findings on the tracking performance of the used tracking hardware in this thesis, like the Microsoft Kinect and the Lighthouse tracking of the HTC Vive. Tracking performances of both generations of the Kinect are being compared, both to each other and also to motion capture systems, like PhaseSpace and OptiTrack.

Chapter 3 will deal with the process evaluating the available tracking solutions, and how we selected our hardware and software for that purpose. We will further present the required methods for creating our avatar, its relevance and what methods we will use to animate it in section 3.2. We will also introduce the reader to Unity, a game engine used for our implementation in section 3.3. How we record motions via motion capture and how we play back that data, using different visual styles will be the topic in section 3.4. Chapter 4 will give a detailed description on how we implemented all presented topics in chapter 3. Finally, we will present our user study in chapter 5, our findings and analyzed data from recording twelve participating users. We will end this thesis in chapter 6 with our conclusion.

2 Related Work

Contents

For a virtual reality (VR) system designed for full-body motion training sessions, or clinical applications, reliable and accurate tracking is required. In our work we want to immerse and stimulate the user using such a system. To achieve that, we need a realistic avatar to give the user a feel of virtual body ownership (VBO). To guide the user's movement in the *VR* training environment and to give them proper feedback on their performance, we need some means to record motions from an instructor, or to convey instructions from video sources to our 3D environment. Chapter 2.1 will give the reader an extended introduction on the technology of *VR*. We will discuss its history and how it evolved to what it is today. In chapter 2.2 we will look at *VR* training systems and other topics and explore the impact of their findings and implications on our own solution. We will investigate the importance of having an authentic avatar in a virtual world to convey the natural movement of a user. The effects of *VBO* on self-perception and immersion will be discussed in detail in chapter 2.3. We will also address relevant work on the different advantages of changing the perspective between first-person perspective (1PP) and third-person perspective (3PP) and the effects they have on the users and their performance. We will cover findings of different tracking systems, their accuracy and viability for various scenarios in chapter 2.4.

2.1 Virtual reality in general

This chapter will give the reader a short overview of the origins and evolution of virtual reality from 1960 until today. It will give a small insight in the development and the challenges of the creation of a head-mounted display (HMD), why it didn't succeed in the 1990s and what challenges still exist today. A detailed summary of the origins of today's most important displays, the Oculus Rift and HTC Vive will be done in a later chapter 3.5.

The very first VR *HMD* was developed in 1960. Its earliest predecessor is the stereoscope (figure: 2.1) invented in 1838 by Charles Wheatstone, Professor of Experimental Philosophy in King's College, London [42]. The Stereoscope is designed to create the illusion of depth from flat images using small changes in angle. The earliest version of the Stereoscope used a pair of mirrors at 45-degree angles, so they would reflect an image to the viewer's left and right side. Wheatstone had to rely on hand drawings for his invention, because he preceded the invention of photography. This method of creating a 3D image and a sense of immersion, by using only two flat images, is still valid today and continues to be used by modern VR *HMD*s. All current leading *HMD*s, like the hand-held based Google Cardboard, the HTC Vive, the Oculus Rift and the Microsoft Mixed Reality headsets use two separate images for a stereoscopic effect.

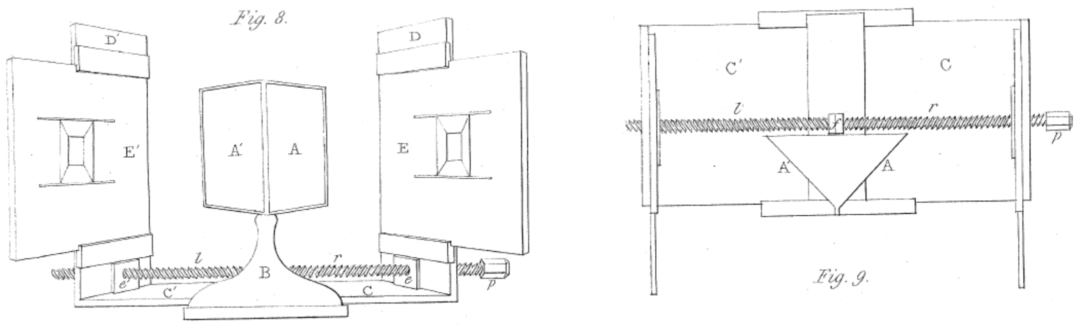


Figure 2.1: The mirror stereoscope viewed from the front (left) and from the top (right) (Image courtesy Charles Wheatstone [42])

The first patented *HMD* suitable for *VR* was the Telesphere Mask by Morton Heilig in 1960. The device provided a non-interactive 3D movie experience with stereo sound. The first device which also included motion-tracking followed only one year later in 1961 by Comeau and Bryan, two engineers at Philco Corporation. They called their device the 'Headsight'. It had a magnetic motion-tracking system, which was linked to a closed circuit camera. The Headsight allowed users to remotely view hazardous situations and was mainly targeted towards military use, like most early VR and simulation systems.

In 1968, Ivan Sutherland, together with his student Bob Sproull, created the "Sword

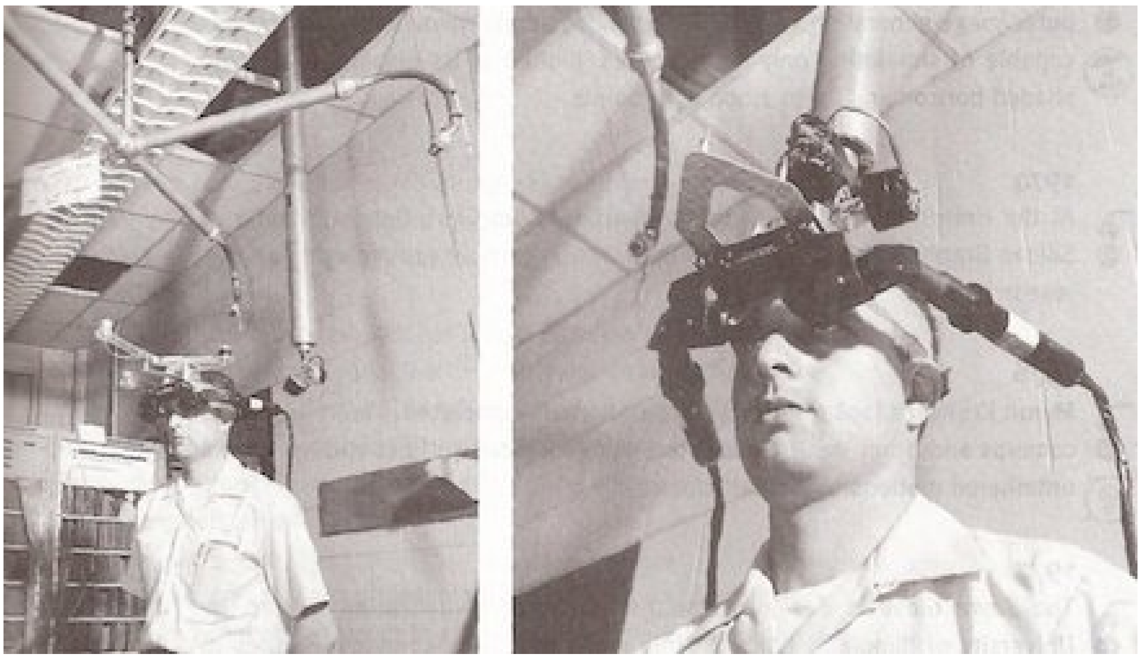


Figure 2.2: Sword of Damocles head mounted display, as demonstrated by Ivan Sutherland (Image courtesy Ivan Sutherland)

of Damocles" shown in figure 2.2, a head mounted VR display, connected to a TX-2 computer, showing simple wireframe renderings of a room. It earned its name because of its bulky nature. To properly wear the device it had to be supported by a ceiling mounted construction, which also supported motion tracking of the headset, allowing the test subjects to look around the environment. The construction was a counterweight mechanical arm suspended from the ceiling. Damocles is a Greek mythological figure featured in a story about Dionysius, his king, who offers to switch places with Damocles for one day, who admires his king's fortune. Dionysius, though also installs a Sword hanging above Damocles' head, only held by a single string of horse hair, to demonstrate that all his fortune comes at the cost of the constant fear of threats to overthrow him. The name for the mechanical arm is inspired by that image of a sword hanging above Damocles' head. Ivan Sutherland described the concept of the "ultimate display" 3 years before he created his head mounted device. According to Sutherland, "[the] ultimate display would, of course, be a room within which the computer can control the existence of matter"[34]. Sutherland envisions the ultimate virtual reality experience to be one, wherein the user is not only visually immersed inside a digital world, but would also be able to physically interact with it. In his description he talks about the display's capability, to not only visually present a chair, but to also be able to physically sit in it. Since everything that is digital inside this world would have physical properties extending to the real world, it would also have real world consequences, like when the user would get hit by a bullet, that wound would be fatal.

By 1981 NASA's Ames Research Center began a research program, lead by Michael McGreevy in dimensional information transfer. They pursued the creation of an immersive virtual training system for pilots. Based on his proposals from 1984 McGreevy published his work "The Virtual Environment Display System"[22] in 1991. This system was the first, that combined computer generated 3D graphics, 3D sound, voice recognition, a *HMD* and a data glove. McGreevy already had the vision of every virtual object being interactive, so that a user could manipulate it by touching it, either with their real hands or a controller, using gestures, or using their voice to issue commands to the system. For every virtual object, to be interactive, the users would also have to be tracked, which would ultimately mean, that "[...]the entire body could be tracked."[22]

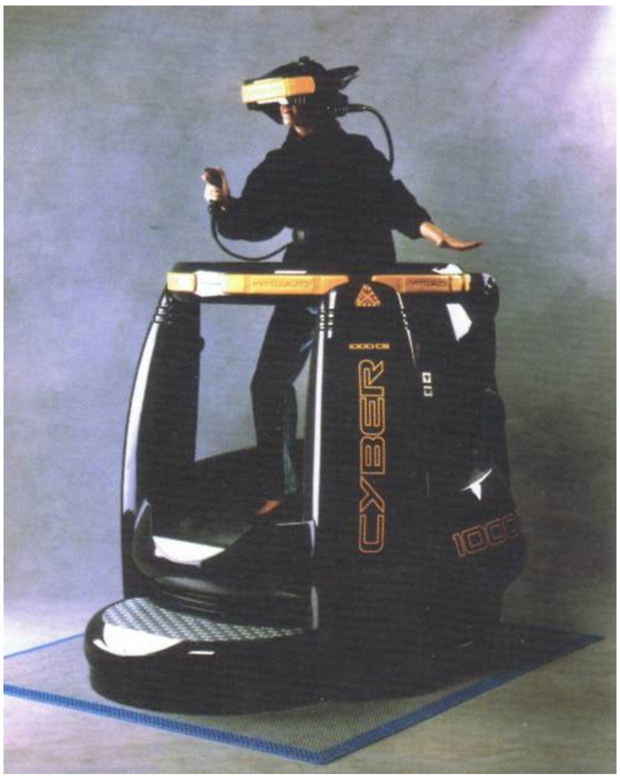


Figure 2.3: Standing user inside the Virtuality system, wearing the Visette

Jaron Lenier founded VPL Inc. in 1984, which is one of the first companies that developed and sold virtual reality products, like the data glove and the data suit. He reputedly coined the term virtual reality in the mid to late 1980s. Sega was the first company that made an attempt to launch a *VR*-headset for consumers. They planned to release their headset for Sega Genesis and arcades. Their development started in 1991, but only the arcade version hit the market. By the early 1990s virtual reality became accessible to a wider audience through arcade machines, like the "Virtuality", shown in figure 2.3. This was produced by the Virtuality group, founded by Dr. Jonathan Waldern. It featured a *HMD*, the "Visette", using two LCD screens with a resolution of 276 x 372 pixels for each eye and giving the user a 65° Field of View (FoV) and a magnetic-tracked joystick. Two different units were developed. One is shown in figure 2.3, where the user could stand upright, whereas while using their second one, the user had to sit down.

Nintendo released their "Virtual Boy", which they called a "portable table-top 3D video game console" in 1995. They announced it on November 14, 1994 in Tokyo [10]. It was marketed as the first console being able to display stereoscopic 3D images. The LED display was monochrome and had a resolution of 384 x 224 pixels and a refresh rate of 50.2 Hz. The portability of the system was in reality limited, because the console was mounted on a tripod and weighed 760 grams. The monochrome screen, eyestrain during play, lack of head tracking and the lack of actual mobility was not well received by the market. This lead to it being considered a commercial failure, although it sold approximately 770,000 units[1], but had an initially projected target of 3 million hardware units. This led to its discontinuation and was therefore only shipped in Japan for ¥15,000 and in North America for $179.95, but never reached the European market.

Movies like *"The Lawnmower Man"*, directed by Brett Leonard in 1992, *"Hackers"* directed by Iain Softley in 1995, *"Johnny Mnemonic"* by Robert Longo in 1995 and most famously *"The Matrix"* in 1999 directed by the Wachowski brothers, helped introducing the idea of virtual reality to a wider audience. Their work increased the hype around the technology in the late 90's, but the industry wasn't able to deliver on those promises due to a lack of high resolution displays, proper portability, the integration of low latency tracking solutions for head movement and the heavy weight of systems of that era.

Major advancements in the 21st century, especially the development of smart phones and therefore the introduction and improvements in small high density displays, let the idea of virtual reality reemerge for consumers. Smartphones feature many technological requirements necessary for the development of a feasible *VR* headset. They offer small light-weight screens with high resolutions, resulting in a high pixel density, quick response times, low latency and tracking technology for rotational tracking in the form of accelerometers and gyroscopes. Another necessary development for *VR* to achieve an immersive experience was the improvement and evolution of hardware. Rendering a scene for *VR* means, that the entire scene has to be rendered twice, from two different perspectives for both eyes. To reduce motion sickness, the experience also has to achieve a high frame-rate, preferably above 90 frames per second (FPS), and a resolution of at least 1080p. However, 1080p can still be considered too low of a resolution. Because the screen is just a few centimeters away from the eyes, a user can still see individual pixels on the screen at that resolution. Desired target resolutions would require the hardware to render an experience above 4K resolution.

2.2 Virtual reality training

Virtual reality with sensor-based tracking has positive effects on training patients and professionals alike. A virtual reality training system is beneficial and low in cost for a wide variety of training applications, like therapeutically training for patients with motor impairments or training for learners without a professional trainer. VR training systems

[1]`http://nintendo.wikia.com/wiki/Virtual_Boy`. Visited on June 24th 2018.

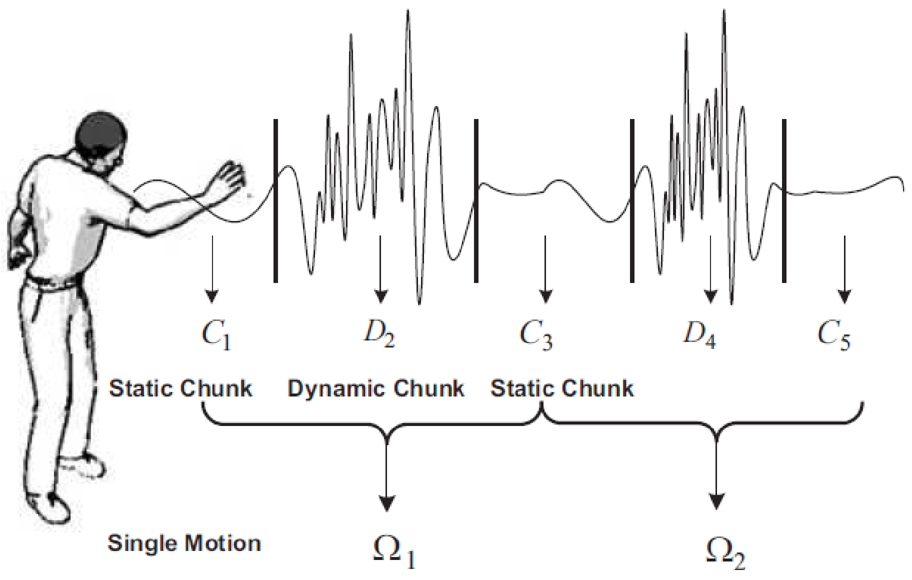


Figure 2.4: Structure of a motion chunk. A recognized single motion consists of two static chunks and one dynamic chunk. Image taken from Kwon and Gross [17]

do not only offer an application in training persons at home, but to also train and improve the performance of professional personnel.

Our system offers training by gamification and a vision-based tracking and feedback system. Tracking is done by the Lighthouse tracking system from an HTC Vive, supplemented by two additional Vive trackers and a Microsoft Kinect for skeletal tracking. We created a system for training within a virtual environment, that offers the user a multitude of features. The training system itself can record and playback motions. Motions tracked in real-time by the system that are executed by the user and visualized by the virtual avatar within the scene can be recorded for later use. In that respect our system is similar to what Kwon and Gross did in 2005.

Kwon and Gross [17] created an immersive VR training system for Chinese Tai Chi motions. Their system allowed both, recording and playback, suiting the needs of a trainer and a trainee. They recorded motions using Inertial Measurement Units (IMUs) and a camera. Hidden Markov Models were used to automatically segment the motion sequences into chunks, by identifying static and dynamic chunks for playback as shown in figure 2.4.

In a study conducted by Kim et al. [15] over a time period of 8 weeks, 15 stroke survivors were monitored to show the effectiveness of tailor-made virtual reality rehabilitation programs. Each patient who participated trained with the tailor-made program for 40-50 minutes and 3 days per week to improve upper-body motor functions. A total of 22 tailor-made programs were developed for the 15 participants, who were between 52 to 77 years old, both male and female and community dwellers. All participants could improve their upper motor functions during the 8 weeks of therapy guided by the virtual reality system. The training suited the requirements of each participant. For rehabilitating the motor functions of each patient, they implemented diagonal shoulder flexion and extension patterns, as they would have been instructed by a therapist. These patterns either begin in an extended position slightly out to the side, about one fist width from the hip, or in a lifted position at about eye-level and end diagonally at either eye- or hip-level respectively. These patterns were incorporated into games like ”Whack-a-mole” or ”Nutcracker” to engage and motivate the patients in their training exercises. Despite the small sample group, the system was able to show a significant improvement in upper limb motor functionality and also that it is safe, feasible, and beneficial for physical functions of chronic stroke survivors [15]. Since the Microsoft Kinect, a low-cost sensor, and other consumer hardware, like a consumer-grade laptop and TV screen were used, a system like this would be easy to supply to clinics or community health centres.

Tianyu He et al. [11] also introduced an immersive system to learn Chinese Tai Chi motions in three different virtual environments in 2017, called ImmerTai. A Kinect was used to record motions by a Tai Chi expert. They implemented and compared a Cave Automatic Virtual Environment (CAVE), a *HMD* (Oculus Rift) and a PC learning environment. The student’s performances were scored by a Quaternion-based similarity assessment using a Tai Chi expert’s scoring as ground truth. During training sessions, the virtual avatar was tracked by a Microsoft Kinect. In their study, the *CAVE* system was the preferred system and achieved the highest scores. The *HMD* environment was a first-person experience with no avatar for the user, whereas the *CAVE* system offered the user a real-time avatar of the student’s motions as reference next to the trainer’s avatar, additional to the ability to see their own real world bodies. Being able to in addition see their own real-world bodies, appeared to be a great factor for preferring the *CAVE* system.

Jiann-Der Lee et al. [19] demonstrated the advantages of an immersive training system during physical rehabilitation for patients with movement disorders. They presented a Kinect-based physical rehabilitation system assisting patients, who performed unassisted Tai Chi movements at their homes. Their method included movement recordings of a Tai Chi instructor, who was invited to perform an exercise. The performed exercise was recorded using their Kinect-based system. For users, using their system, they normalized the skeleton of the user to match the recorded one, and aligned the recording to the torso of the patient. For evaluation, they compared the correct motion behaviour of a patient to the instruction. Alternatingly, they compare the performance of a patient during their study after they were instructed by a therapist, without visual guidance by their system and by measuring the performance with usage of the proposed rehabilitation system. The measurements were taken over a period of several days. When using the visual cues of the proposed system, without the therapist being involved, the patient performed significantly better, than when only being instructed by the therapist. This study showed the potential learning and intervention effects of an virtual guided system for body movement.

'YouMove' was a novel system to record and learn physical movements [2]. Anderson et al. used a floor-to-ceiling mirror setup for the user, where he or she could watch themselves as they move, similar to ballet studios. A mirror gives the performer an immediate visual feedback on their movements. To extend the information gained by the mirror, they created an augmented reality mirror, providing the user additional information and feedback. Their setup of the augmented reality mirror consisted of a 3.2 m x 1.8 m pane of glass. They applied a half-mirror film on one side and a diffuse film on the other side. The reflective mirror film was facing the user and a rear-mounted projector projected the augmented information onto the backside with the applied diffuse film. To track the user, a Kinect was mounted at the bottom of the mirror. The augmented skeleton was aligned with the user's hip. The rotation was not changed, because they assumed that the person standing in front of the mirror keeps facing the mirror in the same direction. They conducted a user study comparing the learning effect of a video-based approach and their augmented mirror system. The participants practiced a motion 45 times during their training phase. Five minutes after completing the training phase, a short-term retention test was completed, consisting of watching a demonstration video and then repeating it five times. The results were recorded using the Kinect. Their work showed that their augmented mirror training technique results in a better short-term retention score than a video-based learning approach.

In Physio @ Home, Tang et al. [35] designed an augmented reality (AR) training environment composed of a TV set and two cameras. The two cameras recorded a subject from a frontal and a top-down view. They used marker tracking to track the arm in the real world. They crafted an arm sleeve, where they put on retro-reflective markers of a Vicon tracking system, to track the shoulder, elbow and wrist joints of a user. To guide the user's arm movement they designed, what they called a 'Wedge', shown in figure 2.7. Their 'Wedge' is a colorized arc, indicating the desired motion, consisting of visual guides for already completed segments and a directional arrow for the remaining path. The two cameras, that record the subject, stream to a TV set in front of the user. The TV set shows both recordings (top-down and front), like a split screen. The 'Wedge' was augmented on top of the video stream and can be seen from the top-down and front view.

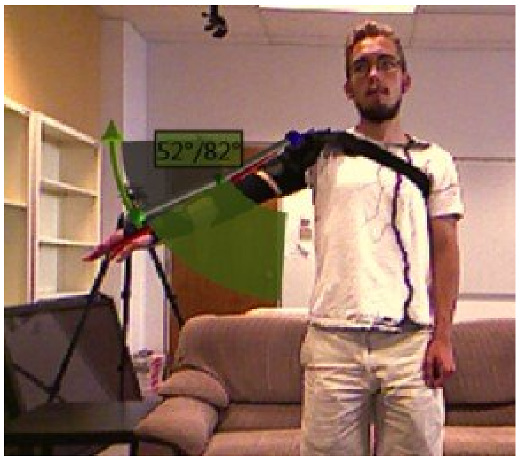


Figure 2.5: Wedge visualization. The dynamic visual guide is overlaid on top of a user's arm. Image courtesy Tang et al. [35]

In contrast to traditional computer games, in a virtual reality environment the user's scale is determined by the actual person's body size. This may present new challenges for designers and developers who are experienced in virtual reality. To even the odds and present a fair challenge for every user, one might scale the user's avatar to always be the same size. At the University of Georgia, Burgh and Johnsen [7] investigated the effects of tracking scale on user performance. Other studies show that the visual senses are dominant over the kinesthetic ones, implying that scaling the tracking data has little effect on the user. Their user study investigated whether scaling every person to the same avatar size would be noticed and effect the user's performances. In their game, participants had to classify bad bread on a conveyor belt. Each user had to do two trials, both in real tracking size and scaled, where the average scale for users was 4.8% and the biggest scale was 114%. The conductor of the study expected a negative effect on the overall performance when scaling the tracking data. This was not confirmed. After the first trial, performance of each participant greatly improved in the second, and where a negative correlation between scaling and performance was present in the first one, it didn't persist through to the second trial. In their study they established that the biggest factors of influence were previous experience with video games of each individual and practice within the game itself throughout trials.

Iguchi et al. [13] experimented to use *AR* for evacuation training of teachers. Earthquakes and other natural disasters are a common occurrence throughout Japan. To develop proper evacuation strategies, Iguchi et al. designed an evacuation trainer targeted at teachers. The goal was to present the teachers an *AR* scenario, with virtual children superimposed onto a real classroom, during a simulated natural disaster. The scene was seen through a smartphone based *HMD*. The teacher was tasked to guide a student with vocal instructions, who behaved unexpectedly, like standing still, or running outside, instead of hiding under a desk. Their preliminary study showed to be promising on the effectiveness of virtual training to prepare adults for disasters.

2.3 Avatar

In an *AR* environment, or when training in front of a mirror or even unguided at home, one can always see their own body. Tools like mirrors or cameras streaming to a screen can help users by giving them additional perspectives and accurate representation of their current pose for self-evaluation. Within a *VR* environment however, the visual field is completely occluded by a *HMD* and only a virtual environment with a reduced *FoV* is visible to the user. To support self-evaluation, a virtual avatar has to be present in the virtual scene environment. Furthermore, the user has to be convinced into perceiving this avatar to be their very own. This can be achieved, for example, by outside stimulation as presented by Botvinick and Cohen [6], or increased realism to represent the actual user as close as possible [18]. This effect is know as *VBO*. The virtual avatar also has to be convincing in his movement. To achieve this, a robust, fast, accurate and reliable tracking has to be in place. To get the best possible results one might usually use a high-end motion capture suit, where several high frequency cameras track a user wearing a suit equipped with tracking markers at every important joint of a human to enable full-body motion tracking. Since this thesis is focused on a low-cost system, we used the skeleton-tracking capabilities of the Microsoft Kinect and utilized the available Vive tracking to enhance the tracking results, to animate the avatar similar to previously described systems. Despite the disadvantage of *VR* to use a virtual avatar, it offers several advantages over *AR*, like the possibility to freely move the camera in the environment and therefore change the perspective of the user at will. This enables us to use a *1PP* and a *3PP*, investigating their advantages and even combining the strengths of both.

2.3.1 First-person perspective and third-person perspective

As evaluated by Salamin et al. in 2006 [31] and 2010 [30] first- and third-person perspective both have their advantages. The first-person perspective's primary strength lies within small hand manipulations in front of the user. Third-person perspective is an out-of-body

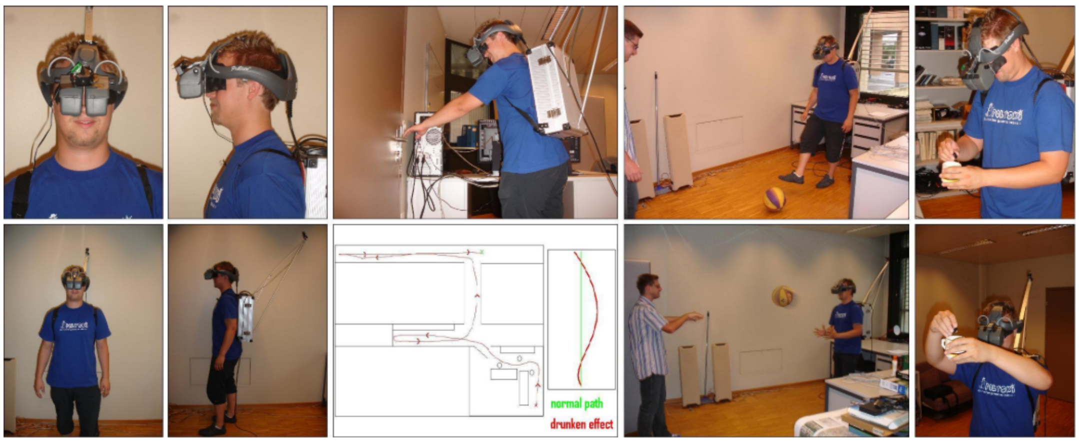


Figure 2.6: Setup and tasks for users in the user study by Salamin et al. [31].

experience, where the users see themselves, or a virtual representation of themselves, from behind. The user's viewpoint is usually offset by 1-2 meters to the back and a few centimeters to the side. This camera perspective is widely used in computer games and the user can chose in most cases if the camera is offset to the left or right side of the game character's body, depending on personal preference. Viewing yourself from a *3PP* does provide several advantages, like a better overview over the environment, better understanding of your positioning and posture and even the possibility to see what's directly behind and around the person. It also allows the user to see body parts that are not visible from a *1PP*, like your own head, or multiple limbs at the same time. Salamin et al. used for their study a *HMD* and placed one camera on the headset itself for a *1PP* and a camera on a swiveling pivot point fixed on an aluminum bar displaced 80 cm behind and 60 cm upper the user eye position, with an 7 degrees tilt towards the bottom to record the user from a *3PP*, as shown in figure 2.6. Users had to complete five tasks using both perspectives in a randomized order. They had to open a door, walk 50 meters through a gallery, put a ball into a coffee mug, receive a rolling ball with their feet and a thrown ball with their hands. Their study showed that the third-person perspective was usually preferred favourable for displacement actions and to track moving objects. *3PP*, as an out-of-body experience, has the drawback of a lack of immersion [8]. It usually is the designer's choice and intention to decide if the loss of a small degree of immersion is desired or tolerable. Salamin et al. mounted in his study a camera on a stick, recording the wearer from behind, to achieve a *3PP*. The user was shown the recorded video stream in real-time on a *HMD*, enabling him or her to view himself or herself from behind, with a slight offset towards the upper left, similar to a game character in video games. In their *VR* environment study participants were able to quickly adapt to the *3PP* and were able to maintain an accurate estimation for distances, which was proven in the study by the participants catching a ball thrown at them from different distances. A study by Denisova et al. [8] with 40 participants evaluated the level of immersion for *1PP* and *3PP* within Skyrim, a role-playing game. Participants had to finish an early in-game quest in both perspectives. Their study supported the claim that the first-person point of view (POV) is more immersive than the third-person *POV*, regardless of player preference.

2.3.2 Virtual body ownership

Acceptance in terms of *VBO* and immersion are strongly connected to the appearance and liveliness of the avatar. Latoschik et al. [18] and Waltemate et al. [38] studied the impact of avatar personalization and immersion on *VBO*. The *VBO* is similar to the classical rubber hand illusion introduced by Botvinick and Cohen in 1998 [6], where users, after receiving synchronized visual and physical stimulation, accept a rubber hand as their own. In this experiment an illusion is created by seating a subject upon a small table with their left arm resting in front of them, on that table. A standing screen was used to obstruct the subject's view of their real arm. A life-sized rubber hand replica was then positioned in front of the participant and next to their real, but occluded arm. To trigger *VBO*, the participant was then stimulated, using two paintbrushes synchronously striking both, the real and the rubber hand, while the subject was concentrating on the rubber hand. It was reported that the subjects felt ownership over the replica and the illusion could interfere with their judgment of alignment of their arms when blindfolded.

Body ownership can strongly affect a person's reaction, usually confirmed by triggering a stress reaction, by threatening the replica. In 2010, Slater et al. [33] investigated the *VBO* illusion phenomenon within virtual reality. In their experiment there were 24 male participants, who experienced being a female child sitting on a chair, being stroke by an adult woman. They compared the effect on *VBO* from a *1PP* and a *3PP*. During the experiment, all participants felt tactile feedback synchronized with the visual strokes inside virtual reality to induce *VBO*. To measure the effect, a stress reaction was triggered by the virtual woman slapping the child three times to measure the subject's heart drop rates. Their study concluded that *1PP* has a significant higher impact on *VBO* than *3PP* and that male subjects can take *VBO* over a female child.

The avatar can have various psycho-physical effects on a user within the virtual space, described by Yee et al. as the "Proteus effect" [43]. Users tend to change and infer their personality from their virtual avatar. This effect was shown by Pena et al. [29] in 2009 where users changed their behaviour, level of aggression and their attitude according to their avatar and stereotypes associated with that. Further studies of body ownership show the effect of changing the appearance of the avatar in a fully immersive *VR* environment, like changing a person's gender, the appearance of one's figure, skin color, or size. Normand et al. [26] demonstrated in their paper in 2011 that an illusion of body distortion can be achieved in virtual reality with synchronous visual tactile stimulation. The experiment showed an impact on the self-perception and on judgment of the subject's own belly size. A social study by Peck et al. [28] investigated the effect of virtual embodiment on racial bias and to which extend *"[...] virtual embodiment can change implicit biases such as racial prejudice, that are likely rooted in our personality and that can be modified only with great effort."* In this study, a Racial Implicit Association Test was conducted and found support for the hypothesis that body ownership of a light-skinned person over a dark-skinned virtual avatar can reduce racial bias. Banakou et al. [4] 2013 found significant impact in perception and size estimation by putting an adult into the body of a child in *VR*. They substituted a user's body with a sex-matched virtual child proportioned body or a scaled down adult body. Their study showed that putting an adult into the body of a child led to overestimating sizes. This effect was also more significant when compared to a scaled down adult proportioned avatar, matching the height of the child avatar.

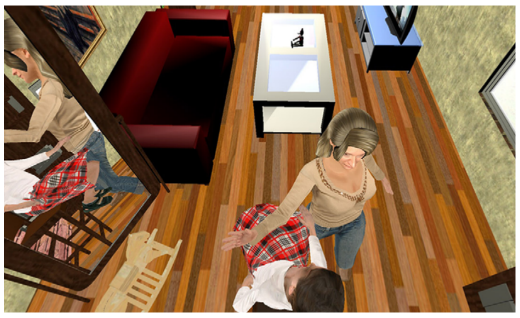


Figure 2.7: Woman virtually slapping the child as seen from the participants in the study conducted by Slater et al. [33] (Image courtesy Mel Slater

Studies from Latoschik et al. [18] in 2017 and Waltemate et al. [38] in 2018 showed the impact of avatar realism in *VBO*. Compared to a generic but gender-matching avatar, a personalized avatar significantly increased *VBO*, presence and dominance. They conducted a study using twelve different avatar types. They used six for each gender. A generic one created with Autodesk Character Creator, a scanned non-individualized avatar, and three 3D scanned versions. They were able to confirm an improvement in *VBO* if the avatar is customized. The customization included a 3D scan of their body, their faces and their clothes. All three had an impact on the resulting immersion.

2.4 Tracking

Immersion in virtual reality is a faceted topic and influenced by many factors. One factor contributing to an increased immersive feel is the avatar representing the user within that environment. As described in the previous section 2.3.2, immersion is influenced by the realism and resemblance of the avatar used. This realism has to be conveyed by a natural animation of said 3D character. To achieve a most realistic transfer of real body motion to the virtual space, robust, precise and reliable tracking has to be available. The advancements in sensor technologies, like infrared depth sensor cameras, influenced and advanced markerless tracking of human motion.

For a high tracking accuracy, sophisticated motion tracking solutions, like OptiTrack[2] motion capture technology, must be used. With this tracking technology, the user has to wear a full body suit with tracking markers attached, tracked by high speed cameras (up to 240 Hz). It is capable of sub-20 µm accuracy in optimal conditions and achieves sub millimeter accuracy consistently. Markerless and low-cost tracking technology is readily available for human tracking, like the Microsoft Kinect generation one and two, an RGB-D camera, originally intended for hands free interaction with the Microsoft Xbox 360. This sensor achieves good and satisfying results for most cases, but lacks in reliability and precision. "*An important milestone for wide adoption of these technologies was the release of Microsoft Kinect camera for the gaming console Xbox 360 in 2010, followed by the release of Kinect for Windows with accompanying Software Development Kit (SDK)*" [39]. In an extensive examination Wang et al. [39] compared the skeletal tracking accuracy of the Kinect 1 and Kinect 2.

Wang tracked 10 users, using simultaneously a Kinect 1, a Kinect 2 and an optical motion capture system by PhaseSpace. To keep all data comparable only those joints common to all three capture systems were used in the evaluation and the PhaseSpace motion capture data was used as a baseline. Their study showed that the offset and standard deviation (SD) are considerably smaller for the Kinect 2 compared to the Kinect 1. The Kinect tracking algorithm does not calibrate of pre-define lengths of anthropometric human bones. Since the human body can be approximated as a rigid kinematic system with determined sizes, we can assume that bone lengths do stay constant. The Kinect skeleton tracking algorithm however, does change bone lengths over time. Wang et al. interpreted this peculiarity as a measure of robustness and measured the variance of bone lengths additionally to the joint positions accuracy. This measure was also in favor of the Kinect 2 with a *SD* between 4 mm and 21 mm in bone length when facing the sensor straight.

Wasenmuller and Stricker [40] made a comparison of depth image accuracy and precision obtained from Kinect v1 and Kinect v2 depth sensor systems. First of all, they found out that the Kinect v2 requires to pre-heat for at least 25 minutes, since they found a correlation between depth accuracy and device temperature. Similar to Livingston, they

[2] `https://optitrack.com/products/motion-capture-suits/` . Visited on October 30th 2018.

also found that the Kinect v1 depth accuracy decreases exponentially with increasing distance. They also found a correlation between depth accuracy of the Kinect v2 and the scene color.

In an experimental analysis on the accuracy of the Microsoft Kinect Webster and Celik [41] compared the Kinect skeleton tracking accuracy to an OptiTrack system. Their comparison concluded in an acceptable level of accuracy for the Microsoft Kinect of a normalized root mean squared error (MSE) of 1.74 cm. They state: *"The results of this study support the Kinect as a sufficiently accurate and responsive sensor for gross movement-based impairment assessment and rehabilitation progress tracking system for both clinical and in-home settings."* [41]

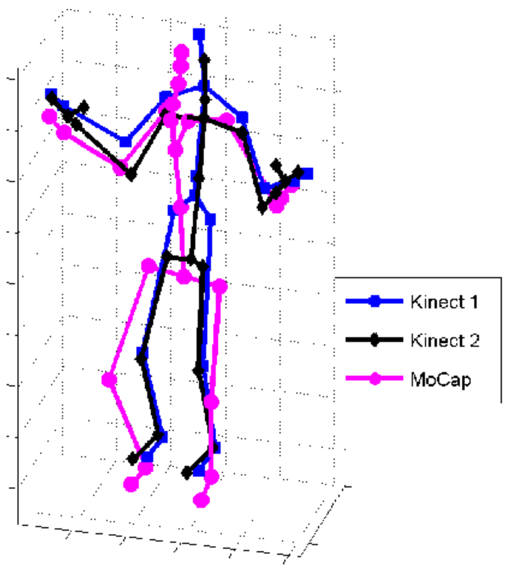


Figure 2.8: Three skeletons captured by Kinect 1, Kinect 2, and motion capture (extracted via Recap2 software) after geometric and temporal alignment. Source: Wang et al. [39]

In an evaluation of the Kinect skeletal tracking, Tao et al. [36] compared the tracking precision for hand position, trunk position and elbow angles with an OptiTrack tracking system. They use the OptiTrack as a gold standard for their comparison. Their goal was to evaluate the tracking accuracy of a Microsoft Kinect and its validity in a clinical setting for upper limb hemiparesis. They concluded that the Kinect tracking capabilities are sufficient and that the optimal distance for the Kinect sensor from the user should be between 1.45 and 1.75 m with an offset of 0.15 m either to the left or the right.

Automated evaluation of the performance of a user in 3D in real-time and comparing it to a gold standard (for example pre-recorded professional) is a daunting task. As stated by Alexiadis and Daras [1], there are several technical challenges including temporal synchronization and spatial alignment. They demonstrated this problem by evaluating the performance of professional dancers. They presented an automatic system based on quaternionic signal processing. In a real-time guidance system motion data has to be spatial-temporal aligned with the user. To account for user delay, Shen et al. [32] conducted experiments to evaluate existing and proposed methods to align motion sequences between a pre-recorded avatar and an exercising user using Microsoft Kinect skeleton tracking. On a real-time guidance system human reaction has a delay, and comparing recorded motion data with the current motion vector of a user is inaccurate. User motion data should therefore be compared to an earlier stage of the instruction and not to the currently active one. Many exercises require to move multiple body parts at the same time. Reaction delay for these parts can vary. Shen et al. [32] therefore spatially segmented the body into five separate parts: head and torso, left arm, left leg, right arm and right leg. In their final method, they temporally and spatially segment human delay for a better estimation of the delay to align the current user's action to the trainer's pre-recorded motion.

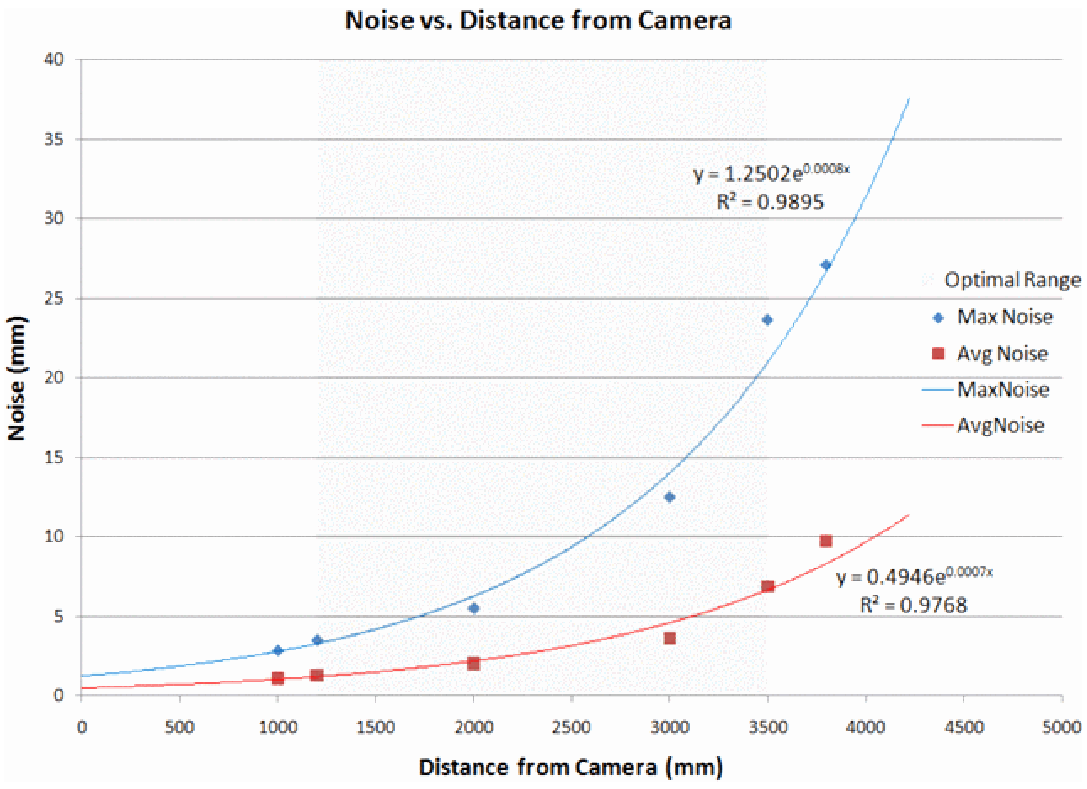


Figure 2.9: The mean and maximum noise as a function of distance from the sensor showed an exponential fit. The shaded region denotes the optimal depth range. Source: Livingston [20]

Mark Livingston et al. [20] measured and evaluated the accuracy and noise of the Kinect for Windows skeleton tracking. They found 3D noise at a distance from the sensor of 1.2 m to be 1.3 mm with a standard deviation of 0.75 mm. At a distance of 3.5 m, the noise increased to 6.9 mm with a standard deviation of 5.6 mm. As shown in figure 2.9 they measured that the mean and maximum noise had an exponential fit to its distance. They also found that the noise varied by dimension, with the x-axis being the most accurate with an average of 4.1 mm, the y-axis averaging at 6.2 mm and the z-axis averaging at 8.1 mm.

Depth sensing cameras like the Kinect or the Asus Xtion did a major contribution to low-cost, real-time motion tracking. This technology is not only cheap to acquire, but also easy to set up. In comparison to more sophisticated body motion tracking systems like the ones offered by OptiTrack or Vicon, RGB-D camera tracking technology does not require the user to wear a special suit, covered with retroreflective markers, calibration of the markers and calibration of the required cameras. While the possibilities of depth sensing cameras are great, they also have some shortcomings. Their full-body motion tracking often fails in outdoor scenarios due to sunlight interference, they have a much lower resolution than modern RGB cameras and they are not very common among consumers. Regular RGB cameras, on the other hand, can be found in almost every single pocket nowadays. Skeletal pose estimation from a single color camera is a challenging task and usually done offline. As a novelty, Mehta et al. [23] presented the first method to solve a stable kinematic skeleton solution in real-time from a single RGB video. They built their approach upon state-of-the-art 3D pose estimation methods utilizing convolutional neural networks (CNN)s. The results of their new method showed to be comparable to RGB-D methods and even surpass them in outdoors scenarios. It also can be applied on community videos, or even video streams from mobile phone cameras, making it much more applicable and versatile. The presented method shows itself to be as accurate as a Microsoft Kinect and a solid alternative for human skeleton pose tracking, despite having some disadvantages. Similar to the Kinect, their system also has problems with occlusions and self-occlusions of the tracked person. Furthermore, depth estimation from a monocular image is a difficult task, and even slight inaccuracies in the depth estimation can lead to large errors, a problem a RGB-D sensor can prevent, because of its additional depth image.

3 Method

Contents

In this chapter, the creation of a virtual reality (VR) system implementing several techniques for visualizing virtual reality training systems will be explained in detail. Using the crafted *VR* environment, pros and cons of described techniques for *VR* training will be evaluated, improvements and suitable alternatives will be inspected. For an immersive training environment accurate and reliable tracking is important. We inspect two different consumer-grade tracking solutions and their reliability.

Henderson et al.[12] showed the advantages of augmented reality in a cramped environment by reducing head movement and increasing effectiveness of localization. A *VR* environment gives the system more flexibility and use cases not available in a traditional learning environment like 2D videos, or text-based instructions, or an augmented reality learning environment. The separation of the users and their body, by replacing them with a tracked and 3D captured avatar and allowing a free movement of their perspective within the scene gives us the option of a third-person perspective (3PP).

As evaluated by Salamin et al.[31] first-person perspective (1PP) and *3PP*, both have their advantages in different situations, whereby *1PP*'s strengths are small hand manipulations in front of the user. These actions are small motoric movements contained within a small working area and would be obstructed from the user's avatar in *3PP*. Visualization techniques for overcoming the problem of occlusion in the *3PP* would have to be examined and evaluated, like the introduction of an occlusion-shader, which was used in our system. The third-person perspective is favorable for displacement actions and to track moving objects. It allows users to better evaluate distances and anticipate respectively extrapolate trajectories. The larger field of view also helps the user to better asses the situation and to understand his or her body posture.

Our hypothesis is that the *3PP* has an advantage over the *1PP*, by increasing the users' overview of the overall environment and their own body. Contrary to the *1PP*, where only a limited part of the users' body is within their perceived and visible Field of View (FoV), the *3PP* allows us to place the scene camera, so that the whole body is within our *FoV*. We also mitigate occlusions by the virtual avatar using an occlusion shader for our 3D glyphs, so that they stay visible in *3PP* when being in front of the avatar. We expect to reduce cognitive load and required head movement.

3.1 Tracking

For our purpose of real-time tracking we integrated and studied two different low-cost consumer-level tracking systems with varying precision, cost and usability. For our system, a reliable and stable tracking, supplying enough trackable points to animate an avatar is desired. We will discuss the different approaches, their advantages and disadvantages and which was finally chosen for this thesis.

3.1.1 Microsoft Kinect

The Microsoft Kinect is a consumer-level device initially designed and shipped with the Microsoft Xbox 360 in 2010. It is a low-cost RGB-D sensor. The Kinect is a motion-sensing input device released by Microsoft for the Xbox 360 and Xbox One game console. The sensor is a peripheral, intended to work as a hands-free natural user interface (*NUI*)

Feature	**Kinect 1**	**Kinect 2**
Color Camera	640 x 480 @30fps	1920 x 1080 @30fps
Depth Camera	320 x 240	512 x 424
Max Depth Distance	4.5 m	4.5 m
Min Depth Distance	40 cm	50 cm
Horizontal *FoV*	57° degrees	70° degrees
Vertical *FoV*	43° degrees	60° degrees
Infrared (IR) Stream	No IR stream	512 x 424 11-bit dynamic range
Audio capture	4-mic array returning 48 Hz audio	4-mic array returning 48 Hz audio
Latency	$\sim$ 90 ms	$\sim$ 60 ms
Skeletal tracking joints	20	25

Table 3.1: Comparison of camera specifications of Kinect v1 for Windows and Kinect v2 for Windows

for the console. The sensor bar features an RGB camera, a depth sensor and a multi-array microphone. Its depth sensing camera and integrated microphone support the use of gestures and spoken commands to interact with the game console. Exploiting the capabilities of the depth sensor, enables it to provide full-body motion capture, hand gestures, facial and voice recognition. The first generation of the Kinect has a color camera with a resolution of 640 x 480 pixels, and the second generation supports a high definition (*HD*) resolution of 1920 x 1080 pixels. The depth camera's resolution was increased from 320 x 240 pixels up to 512 x 424 pixels for the second generation. Both cameras on both generations have a refresh rate of 30 Hz. They both have the same tracking depth distance of 4.5 m, but the second generation increased the *FoV* from 57° horizontal to 70° and 43° vertical to 60°, allowing the user to stand closer to the device, while still being fully visible and therefore increasing the total volume captured by the device. For a detailed comparison of the specifications see table 3.1

The first version of Microsoft's Kinect used a structured light approach to acquire depth data. The technique for acquiring depth data was changed for the second version of the Kinect and was substituted by the Time of Flight (*ToF*) principle. The accuracy of depth is very different between the two versions of Kinect. Where the accuracy in the first version decreases with the square of the distance, the Kinect 2, in contrast, is relatively constant within its tracking volume [39]. The accuracy drop of the Kinect 1 is shown in figure 2.9.

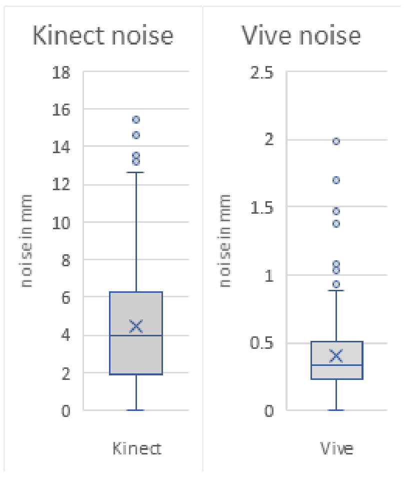


Figure 3.1: Tracking noise for Kinect (left) and Vive tracker (right) measured in millimeter for a stationary hand.

At the Naval Research Laboratory, Mark Livingston [20] measured and evaluated the accuracy and noise of the first Kinect tracking system. As previously mentioned in chapter 2.4, he found 3D noise at a distance from the sensor of 1.2 m to be 1.3 mm with a standard deviation of 0.75 mm. At a distance of 3.5 m, the noise increased to 6.9 mm with a standard deviation of 5.6 mm. He measured that the mean and maximum noise had an exponential fit to its distance. He also found that the noise varied by dimension, with x-axis being the most accurate with an average of 4.1 mm, y-axis averaging at 6.2 mm and the z-axis averaging at 8.1 mm. Another noticeable and important find for our system was, that the average noise for the wrist joints, which are the outer most limbs and our most desired tracked joint, showed the highest noise over all tracking points, with 31 mm for the right wrist.

These findings were validated with a measured average noise of 4.49 mm for a rested left wrist joint, 2.5 m away from the sensor with a standard deviation of 3.23 mm. In our test, a Vive Tracker was attached to the left wrist and used as a reference point for the Kinect. The wrist was rested on a stationary tripod to stabilize the arm and minimize movement. Both the Vive and the Kinect wrist joint were tracked simultaneously and spatially and temporal aligned for comparison. Similar to the findings of Oliver Kreylos [16] and [25], an average noise of 0.39 mm and a standard deviation of 0.28 mm for the Vive tracker, when stationary could be measured as shown in figure 3.1.

Microsoft SDK

To get the best possible results for our tracking we used the Kinect for Windows SDK 2.0 together with a Kinect 2 for a better, more stable skeletal tracking, more tracked joints for a better visual result and a wider tracking area when compared to the Kinect 1. The increase in joints gives the avatar a more natural and fluid looking movement. Also, the improved sensor slightly reduces noise and judder for each tracked joint since its depth camera is more accurate and robust as examined by Wang et al.[39]. Furthermore, the position of the pelvis joint for the Kinect 1 skeleton tracking has a large offset towards the core of the human torso, as shown in figure 2.8, whereas the Kinect 2 pelvis does not.

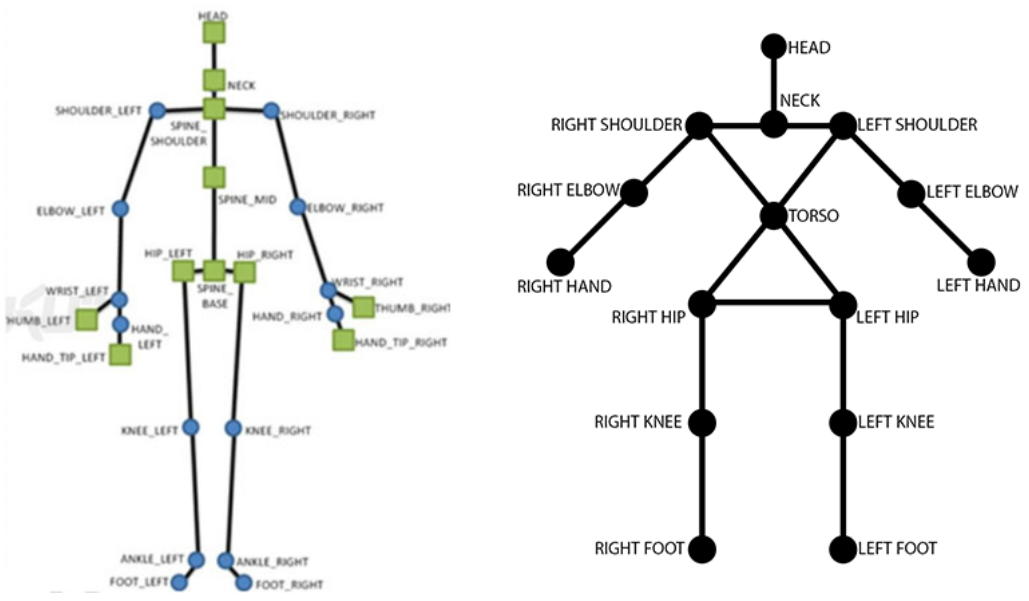


Figure 3.2: Tracked joints of the Microsoft SDK Durango skeleton (left) and tracked joints of the NiTE framework skeleton (right)

OpenNI and NiTE

OpenNI (Open Natural Interaction) is an open source SDK created by an industry-led non-profit organization. One of its main founding members was PrimeSense, a company acquired by Apple on November 24, 2013. The OpenNI framework includes an open driver to access motion sensing hardware like the Microsoft Kinect, which was used for full-body motion tracking in a previous project, or an Asus Xtion Pro. The framework is used for the development of 3D sensing middleware libraries, like NiTE, which we used in a previous application for skeletal tracking to animate a lifelike avatar. The OpenNI driver gives access to the sensors and streams like infrared (IR), depth, and color streams.

NiTE (Natural Interaction Middleware) is a middleware developed by PrimeSense built upon OpenNI, for skeletal full-body motion tracking. It provides an API for hand-based or full-body control. For hand and body tracking it uses the depth, color and infrared (IR) streams, provided by the Kinect. The middleware was used in a preliminary project for tracking a user in conjunction with the first version of the Kinect sensor, and later on, replaced by the more advanced Kinect for Windows SDK 2.0 utilizing the full capabilities of the Kinect 2. The NiTE skeleton is composed of 15 joints as shown in figure 3.2. Every joint has its individual rotation and position, whereby the distance between them and the length of their corresponding bones can alternate throughout time. Also, the joints for shoulders and hips cannot move freely irrespective of each other. They are a stiff plane, which leads to the torso always being on the same plane with the shoulders and hips. Also, the joint for the neck always lies in the middle between the left and right shoulder. Because the distances between joints can alternate, the decision was made to only use their rotational values and apply them on a custom avatar prepared for avateering. The avatar had a custom rig to access those joints and to apply rotational values of the NiTE skeleton to them. The rotational values obtained through NiTE had to run through some post-processing, because every joint has its own rotation irrespectively of its parent joint. Since the joints of the 3D rig always inherit their parent's rotation, these had to be applied to every child.

3.1.2 Lighthouse tracking

In our project we used the HTC Vive. It ships with three trackable devices: the head-mounted display and two controllers. The tracking system used is a laser-based inside-out positional tracking called Lighthouse, developed by Valve for SteamVR and HTC Vive. Key component of the tracking system are two base stations. These base stations contain an IR beacon and two laser emitters, that spin 60 times per second and constantly sweep the room. One laser projects a horizontal line of light and the second a vertical, sweeping through the tracking space in front of the base station. Both rotating lasers and the LED array used for synchronization can be seen on figure 3.3. The laser rotations are interleaved because there can only be one laser sweeping the tracking space at a time. To support this interleaving, the two base stations need to be synchronized. This goal is either achieved by the two LED arrays inside the base stations, flashing a wide angle synchronization pulse, or by using the synchronization cable, if both base stations are too far apart or cannot see each other. Measurements taken by Oliver Kreylos, the author of the online article ”Lighthouse tracking examined” [16], show that although the refresh rate of the Lighthouse base stations is 120Hz, the positional update rate of the *HMD* could be measured at 1006Hz and at 366Hz for each controller. These high refresh rates lead to the conclusion, that the built-in Inertial Measurement Units (*IMUs*) [24] are used in-between Lighthouse sweeps for positional tracking. By integrating linear acceleration and angular velocity measurements from each device's built-in *IMU* via dead reckoning.

As another important part of positional tracking, residual noise and tracking jitter of the headset while being mounted on a fixed position. In this setup, the Lighthouses were placed 4 m apart and 2.4 m above the floor. The noise was isotropic and had a range of 0.3 mm. This error was to be expected. The camera-based nature of the tracking system connotes that the lateral-to-camera error grows linearly with respect to the camera's distance. In practice, the residual jitter and slight inaccuracy in tracking are not noticeable by the user.

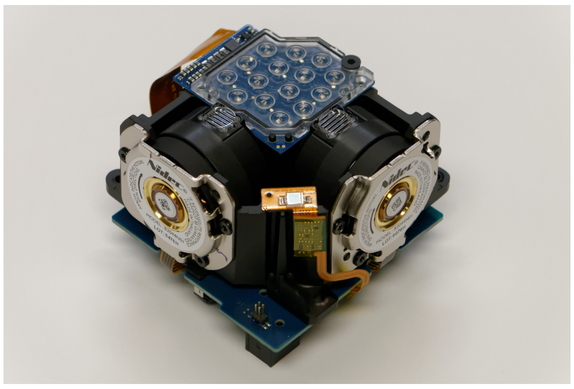


Figure 3.3: Lighthouse base station disassembled. Source by Allyn Malventano [21]

3.2 Avatar

A self-avatar for the virtual environment was created and implemented in this thesis to represent the real user. This was done with the goal to improve the feel of immersion and create a sense of virtual body ownership (VBO). We will discuss the creation of an avatar, using consumer grade hardware and accessible software. The goal was to investigate the required steps necessary to undertake the creation of a virtual self-avatar without employing any external services or professionals and keeping it cost-effective. There is a certain prowess and a certain amount of knowledge required for a satisfying result. Hardware requirements are minimal. For the author's approach a camera and a PC were required. Software was used to minimize manual labor and to automate the process of creating the avatar and rigging the mesh in preparation for animation, as much as possible. The camera was used to take a set of photographs of a person with the goal to create the mesh of the avatar using photogrammetry. We used a mesh reconstruction software by Autodesk called ReCap, which uses pictures taken from different angles to create a 3D mesh. The process of creating and preparing the avatar is described in chapter 4.3 in more detail.

Inverse Kinematics

Inverse kinematics (IK) was initially a system designed to solve the problem of moving a robotic arm, with specific degrees of freedom, when only the target position is known. The technique later also found its way into computer graphics and animation. It is a great tool in computer animation to articulate subjects and create poses without the need to manipulate every single joint individually. The animator is able to only position an effector, like the hand, to its desired position, and the subject's joints will be manipulated by the inverse kinematics (IK) system to reach the position, properly changing child joints' positions and rotations. In a survey, Aristidou et al.[3] presented a comprehensive review of the solutions developed over the years. The survey covers four solutions developed over time. The first family are the analytical methods. Analytic solutions takes an end-effector's pose as an input and outputs joint position solutions as a function of the lengths of the mechanism. Analytical IK solvers are usually much faster than numerical solvers. Numerical solutions are the second method presented. They require a set of iterations to achieve their result, like the Jacobian methods. The Jacobian methods change the configuration of a complete chain of joints, until it brings the effector satisfactorily close to the target position and orientation. More recent approaches are data-driven approaches, which rely on previously recorded motion capture data to solve the *IK* problem. They use databases of motion capture data to guess the correct pose. The data-driven approach is further improved by the recent popularity of deep learning networks. The disadvantage of this method is, that it needs a large amount of recorded data and therefore cannot be applied to solve uncommon chains of joints. Its best use is for human body motions.

3.3 Unity

Unity is a game engine, suitable for 2D and 3D projects. The engine supports cross-platform deployment and a user-friendly, easy to learn development environment. Projects can be published to Windows, Mac OS, Linux, WebGL, mobile platforms, like Android, iOS, Universal Windows Platform and consoles, like the PlayStation (PS) Vita, PS4 and Xbox One. Scripts are written using C# and the .NET run time. The developer is provided with a large collection of tutorials, suited for beginners and experts alike. "Easy enough for the beginner and powerful enough for the expert."[1]

Unity has a thorough and well-written documentation available which makes it accessible and easy to learn. The community is very active and a vast majority of hardware and software companies offer support and frameworks for their products within Unity. The main reasons for choosing Unity as a developing platform were primarily driven by three major factors. First, the accessibility, support and good documentation. The documentation manual[2] is extensive and well-written with good samples, paired with the availability

[1]`https://code.tutsplus.com/tutorials/introduction-to-unity3d--mobile-10752`. Visited on October 15th 2018.

of thorough tutorials for each main topic. The second important factor was the support of the used hardware. All vendors of the used hardware supported Unity with SDKs simplifying access to data provided by the hardware, like sensor data. Third, the engine's licensing plan offers a personal license free of charge, as long annual gross revenues or raised funds do not exceed $100,000. This and the engine's support for a wide variety of build platforms makes it attractive, both for new developers and scientific researchers. Unity is supporting C# and JavaScript as programming languages. This project was written using the C# language.

3.3.1 Unity GameObject and Components

This section explains how GameObjects, components and the Scripting API fit together, and how they were used implementing an environment scene, an avatar, tracking methods, motion recording and visual playback of instructions.

The GameObject is the most important concept in the Unity Editor. Every object in the Unity game engine is a GameObject. Every 3D object, avatar, environments, lights, the camera and other scene elements, like the user interface, logging, recording and playback in our system are GameObjects. GameObjects themselves, however, don't have any properties or functionality. They can be seen more like an empty container. GameObjects get their purpose and functionality by adding components, the second core element of Unity, to them. Components are special classes that inherit the MonoBehaviour class which give the class the possibility to be added to the GameObject, and expose variables to the Unity inspector so they can be changed in the inspector. By deriving from the Monobehaviour class we inherit events and a life-cycle in each component, as seen in figure: 3.4. The most important and most used methods of the MonoBehaviour life-cycle in executed order are Awake, Start, FixedUpdate, Update, OnDisable and OnDestroy. Awake and Start are initialization methods and only called once when the application is run, or the GameObject is being activated for the first time at application run-time. Awake is called after all objects in a scene are initialized, which makes it safe creating references between objects. Awake should be used to set up references between scripts, while Start should be used to pass any information back and forth.

FixedUpdate and Update are game loops. FixedUpdate is re-occurring every fixed milliseconds for accurate physics-based calculations, whereas Update is called every frame and can therefore vary in time passed between each call.

OnDisable is a decommissioning method, being called every time a component or a GameObject with a MonoBehaviour attached is disabled. Disable events and object destruction are also called after OnApplicationQuit, an event called when the application is closed, which is also part of the Monobehaviour life-cycle.

[2] `https://docs.unity3d.com/Manual/index.html`. Visited on October 15th 2018.

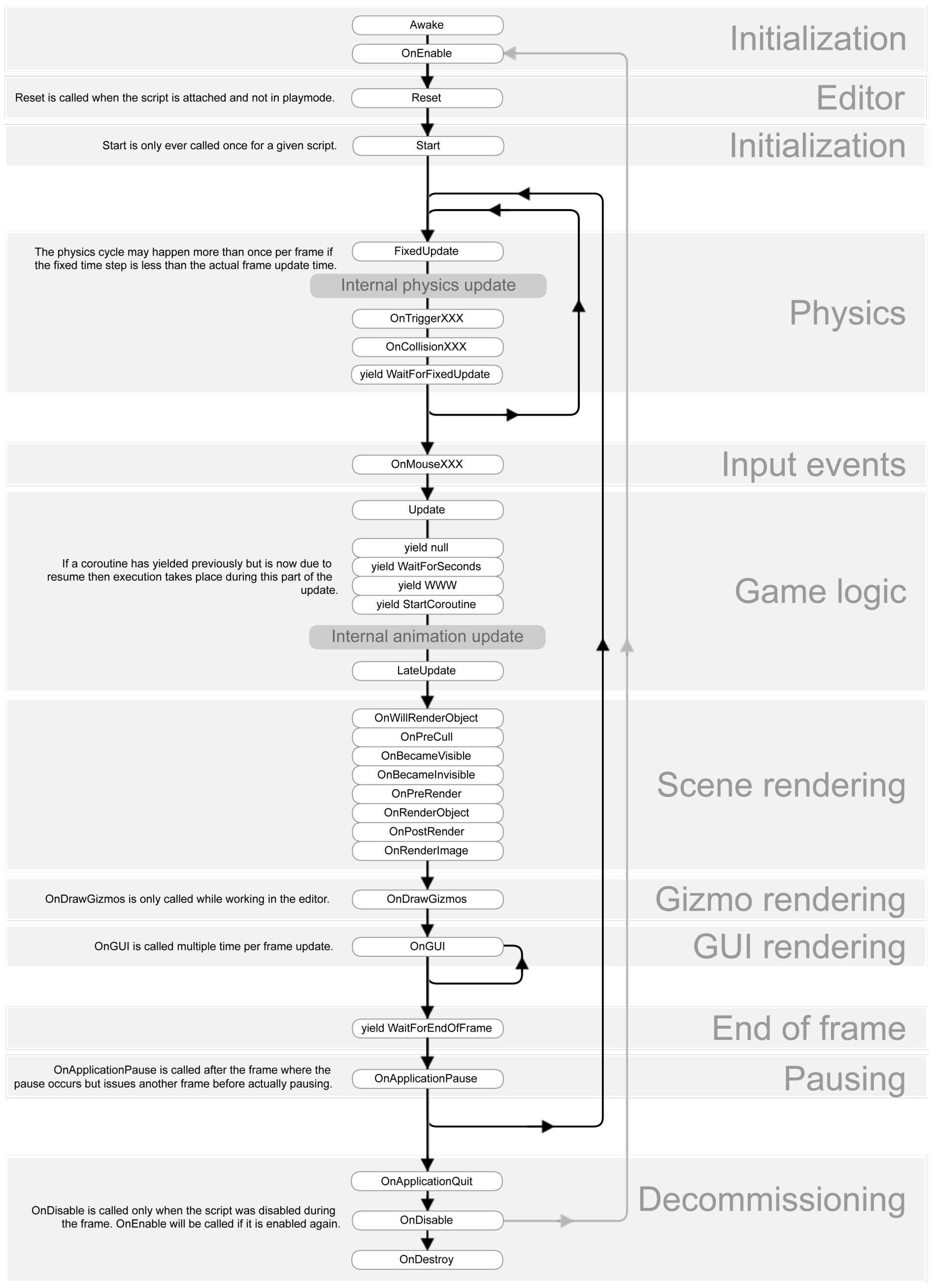

Awake
OnEnable
Initialization
Reset is called when the script is attached and not in playmode.
Reset
Editor
Start is only ever called once for a given script.
Start
Initialization
The physics cycle may happen more than once per frame if the fixed time step is less than the actual frame update time.
FixedUpdate
Internal physics update
OnTriggerXXX
OnCollisionXXX
yield WaitForFixedUpdate
Physics
OnMouseXXX
Input events
Update
If a coroutine has yielded previously but is now due to resume then execution takes place during this part of the update.
yield null
yield WaitForSeconds
yield WWW
yield StartCoroutine
Internal animation update
LateUpdate
Game logic
OnWillRenderObject
OnPreCull
OnBecameVisible
OnBecameInvisible
OnPreRender
OnRenderObject
OnPostRender
OnRenderImage
Scene rendering
OnDrawGizmos is only called while working in the editor.
OnDrawGizmos
Gizmo rendering
OnGUI is called multiple time per frame update.
OnGUI
GUI rendering
yield WaitForEndOfFrame
End of frame
OnApplicationPause is called after the frame where the pause occurs but issues another frame before actually pausing.
OnApplicationPause
Pausing
OnApplicationQuit
OnDisable is called only when the script was disabled during the frame. OnEnable will be called if it is enabled again.
OnDisable
OnDestroy
Decommissioning

Figure 3.4: Flowchart summarizing the ordering and repetition of event functions during a script's lifetime derived from MonoBehaviour.

3.4 Visualization recording and playback

The emphasis of this thesis is on visual feedback and different techniques of playing back recorded instructions to the user. The system implements two different ways of recording human motions. Several approaches were pursued to visualize and present them to a user in a meaningful way that is easy to follow, requires low cognitive load and allows an accurate reenactment by someone following the instruction.

3.4.1 Motion Capture

Our Unity project implements a custom-built motion capture solution, supporting real-time animation, which enables us to record and replay the movements of the user or an instructor for multiple purposes. The main use of the capture system is to drive the avatar in the scene representing the user in real-time. This gives the user a high level of immersion and a great understanding of his own movements. It also provides an accurate representation of the user's current pose. Similar to a large mirror wall at studios where people train and learn for dancing or sports where accurate body movement is required. As one dancer in [9] states *"With the mirror it is good to see my placement [...] sometimes I look at the mirror and see I am doing a movement wrong, I need to fix my sway back, or whether my flat back was as it should be."*. The usefulness of a mirror is supported by several interviewed dancers. The indication of the usefulness of a simple mirror to correct the body pose supports our presumption, that an immersive 3D virtual environment with decent motion capture can help to do the same. Additionally, we have many advantages within a virtual environment. We have free control over the user's perspective and can freely navigate the camera and the user's view, which allows them to watch their movement and poses from any angle. We also can add guidance and feedback to the scene to further improve the performance of a person using the system.

Our motion capture system can also be used to record the movement and later use it for replay. This gives us the possibility to record accurate movements of an instructor and use such recording for visual instruction later with different independent visualizations, like our 3D glyphs.

3.4.2 3D glyphs

To properly guide a person in virtual reality to successfully follow an instruction and give them an intelligible visual feedback, 3-dimensional-glyphs were used by the system. The purpose of the glyphs is to encourage precise and temporal accurate reenactment of a provided motion instruction, without cluttering the visual field of the person using the system. The glyphs are being placed with a uniform spacing along the recorded path. As glyphs we used directional 3D arrows, pointing in the direction of the movement. The glyphs also incorporate visual feedback for the user, continuously visualizing their accuracy through color-coding and scaling the glyphs when being close to them. Color coding and scaling are done with an linear interpolation, using the measured distance between the user's wrist joint and each glyph in the virtual space, with the interpolation working between 0 cm and 25 cm. To reduce visual clutter, we dynamically segment the shown path at run-time, enforcing multiple thresholds, like total visible distance, and a change in direction. Glyphs of successfully finished steps of the instruction are deleted from the scene, therefore only showing parts of the instruction the user still has to finish.

3.4.3 First-person perspective

We showed the users the motion instruction from an egocentric first-person view wherein the virtual camera is placed at the avatar's eye position. Having a virtual avatar animated by skeleton tracking, the users can see their own body and their movements in real-time. This gives a natural, latency-free feedback to the user of their tracked body movement. We found that the tracking has to be reliable and fast in the first-person view, so the users won't feel dizzy or lose balance. Especially a deviation from the velocity or position of the feet movement could lead to losing balance. Being in first-person view, a very accurate sense of scale and depth perception is achieved. Our user study has shown that test subjects are able to easily hit and point at targets within their reach since movement and distances feel very natural. To further increase the accuracy of the tracking system we used the controllers provided by the HTC Vive setup for wrist positioning, rotation and lower arm roll rotations and two additional trackers for the feet. The high accuracy of the Lighthouse tracking system used by HTC, greatly improved the accuracy of our limb movement and reduced Kinect infused jitter and inaccuracy.

3.4.4 Third-person perspective

We introduced a third-person perspective to give the user a better overview of their whole body and the full instruction presented to them. The camera was placed half a meter behind the avatar on eye-level, similar to the camera placement in a third-person game, so the user could see the whole avatar. Contrary to a *3PP* in a computer game the camera was not offset to either side, to avoid that the users would have to constantly tilt their head slightly to look at their avatar. Our expectation was that the *3PP* improves the overall visibility of the avatar and the instructions. Because the whole body was within the viewer's FoV, we expected to gain several advantages. The user should require less head movement and should be able to concentrate on more than just one part of their body at the same time and the instruction as a whole. Without the constant need of turning their head and changing focus on the other arm, we expected that *3PP* would reduce the level of frustration and the time required to follow a motion.

3.4.5 Multiperspective mirror

Inspired by the augmented reality (AR) proposed system Physio @ Home by Tang et al.[35], we implemented a multiperspective mirror. We implemented two virtual mirror screens, placed in front of the user's view. The right panel was a top-down view and the left one a frontal view. Similar to the *3PP*, this method gives a good vision of the instruction as a whole, but was limited to only two plains. Furthermore, the user has to look back and forth between both screens, which causes them to lose focus of the second one. The nature of 2 flat mirrors is the loss of proper depth perception, reducing accuracy and speed. This system was tested against a *1PP* in a preliminary study, similar to the current implementation.

3.4.6 Video instruction

Since many instructions are already available in video form, and because we are able to extract positional joint information for our visual feedback system, we also implemented a virtual TV set for video playback. This form allows the users to watch a video instruction and with the freedom of changing the camera perspective, can also view themselves and better evaluate their performance.

3.4.7 Avatar-in-miniature

The avatar-in-miniature (AIM) is a similar approach to the world-in-miniature concept introduced by Pausch et al.1995 [27], where a miniaturized version of the virtual environment is used for locomotion. The world-in-miniature paradigm is a hand-held miniature representation of the surrounding environment. It shows a localization of the operating user and an exact copy of the virtual world. Manipulations in the virtual environment are conveyed to the miniature. Another goal of the paradigm is to introduce a novel locomotion technique, to allow the user fast travel across large distances within the virtual space. In contrast to teleportation locomotion paradigms direct line-of-sight of the target location from the viewer's perspective is not required. In our case the miniature is intended to combine the advantages of *1PP* and *3PP*. The miniature is a clone of the virtual avatar mimicking all motions. By dynamically positioning the *AIM* within the user's peripheral vision, the whole body posture can always be conceived independently of the used perspective and view direction. This, for example, allows a user to use the *1PP*, focus on one arm and still perceive the whole body posture and instruction on the rest of the body that would not be visible to the user in *1PP*.

3.5 Head-mounted displays

For this thesis we looked at the two most suitable head-mounted display (HMD)s available at the current consumer market. The Oculus Rift and the HTC Vive both are similar and capable *VR* devices, with an outside-in tracking system to track the headset worn by the user and two controllers. In this section we compare both headsets and which headset got chosen for our system, for which reasons.

3.5.1 Oculus Rift

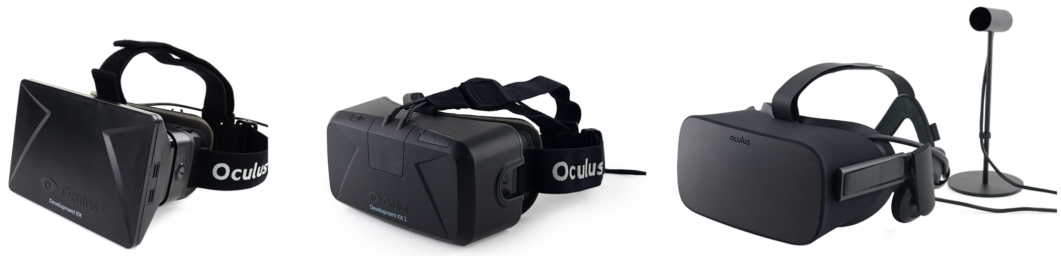


Figure 3.5: (left) Developer Kit 1 (DK1), (middle) Developer Kit 2 (DK2), (right) Consumer Version 1 (CV1) (source: iFixit.com [3])

Most products before the announcement of the Oculus Rift were too expensive and not feasible enough from a consumer's point of view.

Entrepreneur Palmer Luckey created in 2010 his first prototype of the now known Oculus Rift by the age of 17, with a considerable field of view of 90°. The main incentive for creating his own head-mounted virtual reality device, was his dissatisfaction with existing displays. He couldn't find one satisfying his needs. He continued his development of head-mounted devices and shared his progress on the open web forum "Meant to be seen"[4], which eventually caught the attention of the famous game developer John Carmack, co-creator of the *"Doom"* series. Carmack became involved in the project early on and presented a prototype at E3 in June 2012 [37] with Luckey's assent. Luckey started a Kickstarter[5] campaign on August 1, 2012, surpassing its initial goal within 24 hours and collecting a total amount of $2,437,429 from 9,522 supporters by the end of the campaign. The first kits, called *DK1*, reached their customers in late 2012. Although the resolution of 640 x 800 pixels per eye was low, and the tracking wasn't good enough to avoid nausea, it was very well-received, both by the press and consumers. Spurred by the positive reception and financial success, Palmer Luckey founded Oculus VR® and continued to improve his headset with the development of the *DK2*. Oculus VR® gained a lot of attention through the generated hype. Facebook CEO Mark Zuckerberg bought Oculus VR® in March 25, 2014 for 2 billion dollars, which he announced on Facebook [44]. The *DK2* began shipping

[3]https://de.ifixit.com/Anleitung/Image/meta/HT5RXqJQwnwEQkUv. Visited on June 24th 2018.
[4]https://www.mtbs3d.com/phpBB/viewtopic.php?f=140&t=14777&start=0. Visited on June 24th 2018.
[5]https://www.kickstarter.com/projects/1523379957/oculus-rift-step-into-the-game. Visited on June 24th 2018.

in July 2014 and delivered big improvements in specs[6], like a better display with a higher resolution of 960 x 1080 pixels per eye and an increased refresh rate of 75 Hertz (Hz) compared to its predecessor working at 60 Hz. It also featured better persistence and an improved tracking system. The field of view decreased from 110° to 100° and the headsets weight increased slightly by 60 grams.

The next step in Oculus VR® 's plans was the development of their third iteration of the headset. Its working title was "Crescent Bay" and it was later titled as *CV1*. While the previous units were mainly bought by developers, researchers and enthusiasts, this was the first *HMD* by Oculus VR® targeted at the consumer market. Crescent Bay was first revealed in September 2014. In May 2015, Oculus VR® announced their plans to ship the *CV1* in the first quarter of 2016. The CV1 uses an OLED panel for each eye and further increased the display's resolution to 1080 x 1200 pixels per eye and the display's refresh rate to 90 Hz and globally refresh as seen in table 3.2. The low persistence, which means that an image is only displayed for 2ms, is necessary, so that the user won't experience any motion blur or judder, as with regular displays, like many used in smartphones used for a Google Cardboard experience. Oculus integrated headphones with the CV1 to provide 3D audio, licensed from RealSpace 3D Audio.

Shipped Hardware

As the DK2 already did, the CV1 also had a six degrees of freedom (6DoF) rotational and positional tracking. The headset has a built-in gyroscope, accelerometer and a magnetometer for rotational tracking which is supplemented by Oculus VR® 's own "Constellation" tracking system. "Constellation" is an inside-out tracking system used for positional tracking. The Headset is shipped with only one "Constellation" tracker meant to be placed on a table facing the user. This setup limits the user's tracking space and does not provide a full 360 ° experience.

Oculus VR® couldn't finish their development of their motion controller by the time they shipped their CV1 in the first quarter of 2016. They partnered with Microsoft to bundle their units with an Xbox One controller. Their controllers are called "Oculus Touch" and started shipping in October 2016, half a year after the *HMD*'s release. Each controller provides an analog stick, three buttons, two triggers and a system for detecting finger gestures, which allows the user to point with his index finger or give a thumbs up, which increases the possible interactions and immersion within *VR* experiences. The default Oculus bundle was modified and the Xbox One controller was replaced by the Oculus Touch controllers and an additional "Constellation" tracking sensor was supplemented in August 2017 to support tracking of the Touch controller.

[6] `https://riftinfo.com/oculus-rift-specs-dk1-vs-dk2-comparison`. Visited on June 24th 2018.

Display	Dual PenTile OLED Panels
Resolution	2160 x 1200 (1080 x 1200 per eye)
Pixel density	455.63 PPI per eye
Refresh rate	90 Hz
Persistence	Low
Field of View	110° diagonal
Optics	Fresnel lenses
IPD (Interpupillary distance)	58-72mm
Audio	Integrated 3D audio headphones
Tracking	6 degrees of freedom
Rotational tracking	Gyroscope, Accelerometer, Magnetometer
Positional tracking	Oculus Constellation Sensor (outside-in tracking)
Update Rate	Rotational: 1000Hz, Positional: 60Hz
Tracking Volume	100° H x 70° V (over 18 feet range)
Connectivity	4m custom cable that integrates USB and HDMI connections
Weight	360 grams (0.8 pounds)

Table 3.2: Specifications of the Oculus Rift *CV1*

3.5.2 HTC Vive

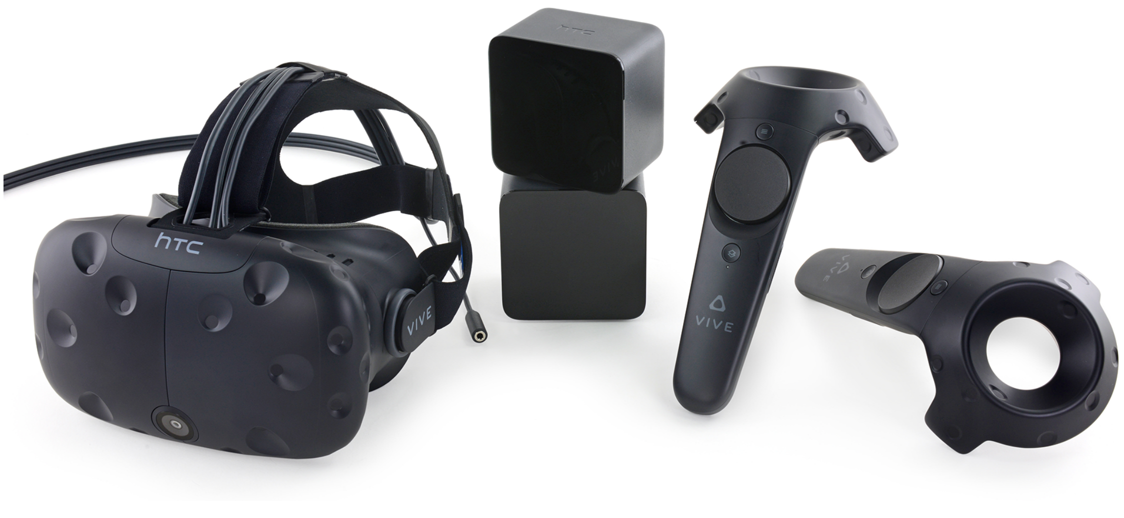


Figure 3.6: (left) HTC Vive *HMD*, (middle) Lighthouse tracker, (right) tracked controller (source: iFixit.com[7])

[7]`https://de.ifixit.com/Anleitung/Image/meta/r4YklkkyQiA165QM`. Visited on June 27th 2018.

With the release of the HTC Vive, an accurate and fast tracking system was introduced to the system, further improving the tracking of the user. The HTC Vive as shown in figure 3.6 was developed by HTC and sold by Valve. It was released on April 5, 2016. The display's specs are almost identical to those of the Oculus Rift *CV1*. The *HMD* has two displays with a resolution of 1080 x 1200 pixels for each eye, giving a total resolution of 2160 x 1200 pixels at 90Hz and a field of view of 110° (see table 3.3). Implemented safety features include a Chaperone play area and a front-facing camera. The front-facing camera can be turned on and used to view the outside world from within the headset. The Chaperone system is a safety feature that automatically displays a virtual wall, or a customizable feed from the front-facing camera. The play area is set up in advance and is a user-defined boundary to minimize the risk of hitting a real wall or walking into obstacles. It weights 470 grams which makes it significantly heavier than the Oculus weighing 360 grams.

As a tracking solution they use the Lighthouse tracking, which is an outside-in tracking system as described in section 3.1.2, with sub millimeter accuracy and low latency.

The newest version of the HTC Vive at the time of writing this thesis, is the HTC Vive pro with an increased screen resolution of 1440 x 1600 pixels per eye, 90Hz and 110 degrees of field of view, and integrated headphones.

Shipped Hardware

Vive got shipped with the headset, two tracked controllers and two lighthouse sensors, enabling room-scale *VR* with a maximum range of 5 meters between base stations. Included is also a link box with a 3-in-1 cable for connectivity to a computer and in-ear headphones for audio. The shipment includes a second interchangeable face cushion with a thicker cushion to support different head sizes and for hygienic reasons.

Additional Hardware

The HTC Vive is designed as a modular hardware with the option to upgrade and add additional hardware. Additional hardware includes a wireless adapter, additional tracker and a deluxe audio strap. The Vive Wireless Adapter is a wireless solution for the headset to remove the cables attaching the headset to the PC. It is attached to the head strap and weights 129 grams, adding to the total weight of the headset. It has a battery life of up to 2.5 hours and requires a spare Peripheral Component Interconnect Express (PCIe) slot. The adapter supports a play area of 6 m x 6 m and allows for up to 3 adapters in a single-room-scale environment, allowing multiplayer in a single room. The Vive Trackers are additional track-able devices one can attach to any object or surface with the purpose to allow additional tracked objects in the virtual environment, like custom controllers, or like in our case, tracked joints of a human body.

We opted to use the HTC Vive for our system for primarily two reasons. First, the Oculus Touch controller were not shipped and available at the start of the development

Headset Specs	
Screen:	Dual AMOLED 3.6" diagonal
Resolution:	1080 x 1200 pixels per eye (2160 x 1200 pixels combined)
Refresh rate:	90 Hz
Field of view:	110°
Safety features:	Chaperone play area boundaries and front-facing camera
Sensors:	SteamVR Tracking, G-sensor, gyroscope, proximity
Connections:	HDMI, USB 2.0, stereo 3.5 mm headphone jack, Power, Bluetooth
Input:	Integrated microphone
Eye Relief:	Interpupillary distance and lens distance adjustment
Controller specs	
Sensors:	SteamVR Tracking
Input:	Multifunction trackpad, Grip buttons, dual-stage trigger, System button, Menu button
Use per charge:	Approx. 6 hours
Connections:	Micro-USB charging port

Table 3.3: Hardware specifications of the HTC Vive[8].

of this thesis' system. We wanted to have the additional position tracking of the provided controllers to track the user's hand position. The HTC Vive, in addition to that also offered the possibility to introduce additional trackers to the system, which we used for tracking the user's feet. Our second reason was the larger room-scale tracking provided by the Lighthouse tracking system. This allowed us to move more freely in a larger area than it would have been, when using the "Constellation" tracking environment offered by the Oculus Rift system.

[8]`https://www.vive.com/eu/product/#vive-spec`. Visited on June 27th 2018.

4 Implementation

Contents

Through iteration we developed two different approaches to animate a virtual avatar via tracking techniques. Both use a Humanoid rigging system and are animated by manipulating individual joints. The first version takes rotational values from the Kinect tracking system or from a previous recording and applies those to the corresponding joints of the avatar. This version is less flexible and has a less accurate placement of the body's extremities. Because only rotational values are used, errors along the hierarchical structure of joints add up and differences in user size lead to incorrect placements of extremities. The latest version utilizes SAFullBodyIK, an open-source inverse kinematic solution from StereoArts[1] , achieving a good and convincing result.

The user is rendered in a 3D environment with a 3D reconstructed avatar of the author. The avatar is controlled in real-time using the game engine Unity and life-tracking. The user's movements are either tracked solely by the Microsoft Kinect or the Kinect with additional tracking info supplied by the HTC Vive. This method is called avateering. For the motion-tracking, we used two different approaches. First, a Kinect for Xbox One and second the HTC Vive tracker with inverse kinematics. We used the Microsoft SDK's skeleton consisting of 25 individual tracked joints for the human body. The user wears an HTC Vive setup for a full immersive virtual reality (VR) experience. To account for different body types, shapes and sizes and to not deform the used avatar, we only applied the tracked rotational data to the avatar. While tracking, only angles of the participant's

[1]https://github.com/Stereoarts/SAFullBodyIK. Visited on October 15th 2018.

joints were used to drive the avatar. Therefore, different body proportions didn't influence the results. But because of the highly accurate sense of scale within a virtual environment, users were able to notice a difference to their own body height. To show the users the motions they had to learn, we implemented different visual methods. Movements were pre-recorded within the system using the same tracking system and avatar. All skeletal positional and rotational data were recorded and used for playback in the study. The system allows any user to record movement and use it as an instruction later on within the same system. There is no further motion capture software or hardware required.

4.1 Tracking

In this thesis we have two different implementations available to animate our virtual avatar. In chapter 4.1 we describe a method using only the available tracking data provided by the Kinect, by applying rotational tracking data to the avatar's joints. Chapter 4.1 describes our second and final approach, where we used positional tracking data from both, the Kinect and the HTC Vive's Lighthouse tracking to animate the character using an inverse kinematics (IK) approach.

Rotational

The system can be used with only the Kinect's skeleton tracking, achieving a similar but less accurate result with an increase of jitter. In this version of the avateering system, only the Kinect's skeleton tracking rotational values are used. Those get applied on the matching joints of the 3D avatar, rotating them individually. This method leads to some inaccuracies and issues induced by the Kinect for Windows SDK 2.0 skeletal tracking combined with the use of a rigged avatar. Some joints' rotations, especially the lower arm's and leg's, aren't stable and reliable. The x-axis of the lower arms, representing the direction of the arm and its roll axis, tend to uncontrollably flip by up to 180 °. This leads to a very disruptive experience for the users, forcing them to wait for the tracking to recover. This causes to different behaviours by the users, who then either try to recover it by randomly moving around, or by standing still and waiting. Both reactions are undesirable, because they either decrease the achieved accuracy of the user, or increase the required time to finish the instruction.

Another issue with using a rigged avatar and animating it by only using rotational information from the tracking system, is the peculiarity of introducing a positional offset to the extremities end positions. The joints of a provided digital avatar's rig are only an approximation and never a detailed and perfectly matching clone of the real world person's bone structure. Only using the rotations and applying them to their corresponding joints, is incrementally adding an offset error, which means that the position of the outermost extremities, like the user's hands, do not end up at their actual real world or tracked positions.

Inverse Kinematic (IK)

The inverse kinematic system uses both the HTC Vive tracking system and the Kinect tracking supplementary. The HTC Vive controllers are used as positional control points for the avatar's wrists. Two additional Vive trackers are tied to the user's ankle and are used for the avatar's feet IK control points. The Vive head-mounted display (HMD) is utilized for a smooth head animation. The Kinect skeleton tracking from the Windows SDK is used for the position of the user's hip, controlling the user's placement in the 3D space. The elbow and knee positional values are used to guide the avatar's knee and elbow pointing directions with the IK system to further improve the IK result. The Kinect could be fully eliminated from the system by providing 1 to 5 additional Vive trackers.

4.2 Avatar

An avatar was created to represent the user using a photogrammetric approach, as described in chapter 4.2.1. We will describe its creation and integration into our scene using the Unity game engine. In chapter 4.3 we will give a detailed description on how the avatar was structured and animated with the engine.

4.2.1 Creation of a life-like Avatar

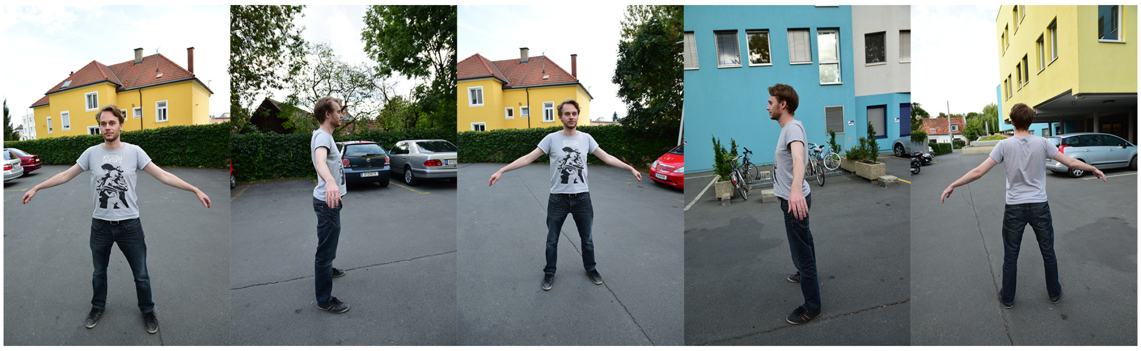


Figure 4.1: A sample from the series of pictures taken used for photogrammetric reconstruction

As an avatar a 3D reconstruction of the author was used. The model was created using 3D photogrammetry and a series of post processing steps. The first step in the process in the creation of the avatar was to take a series of pictures. A Nikon DSLR D3300 camera with 24.2 megapixels and a maximum resolution of 6000 x 4000 pixels was used to capture a person standing in a T-pose. The camera was set to an ISO 800, 1/500 exposure time, F-stop of f/6.3, a focal length of 16 mm and a maximum aperture of 3.6. Fig. 4.1 shows a small subset of the total of 36 pictures taken. For the lack of a proper setup with multiple cameras like Latoschik [18] used, the pictures were taken in a consecutive manner. As environment, an outdoor setup was chosen during a cloudy day for a diffuse lighting and a feature rich background. Having a camera rig with 40 Digital Single-Lens Reflex (DSLR) cameras has significant benefits over shooting all pictures consecutively. The pictures were taken in two full circles at different heights for different angles with approximately 18 degrees between each picture. For additional detail, some close-ups were taken from the face. The whole process took around 15 minutes, resulting in small movements by the photographed person. Hands and the head were most badly affected by that. For the reconstruction ReCap, an online service by Autodesk was used. The resulting mesh was dense with 1,152,994 vertices, good textures, but multiple issues with the topology. To use it for our system, we had to reduce the polygon count of the model and re-model the mesh where the reconstruction algorithm failed. Most issues arose because of the previously mentioned movement of the person during the consecutive photo shooting.

The mesh was refurbished using Mudbox, a 3D digital painting and sculpting software by Autodesk. It was used to reduce the polygon count down to 60,286 polygons, a reduction by 3825%. Furthermore, it was utilized to re-model the hands and most parts of the head and arms, which were not smooth, had excessive volume, lack of detail and no individual finger, as seen in Fig. 4.2. After these re-modeling steps, the model had to be unwrapped again, since its initial UV coordinates were lost due to the re-topologizing process and the texture provided by Autodesk ReCap couldn't be used anymore. After unwrapping, Mudbox's 3D texture painting was used to repaint a texture for the model. The painting tool allowed the user to place a 2D image like our photographs on top of the 3D model in the editor. The image could then be painted onto the model by aligning the model with the image.

To animate the avatar it was necessary to rig the model. 3D rigging is the process of creating a skeleton for a 3D model similar to bones and joints within a living creature, to make it move. Characters need to be rigged before they are animated to be able to be deformed and moved around. The two main processed involved in this step are creating the joint hierarchy and skinning the mesh.

- Joints

  These are also called "bones". Joints are similar to bones in the human body. They work in a similar way. Joints are the points of articulation you create to control the model. They represent the rigid objects underneath the skin controlling the movement and deformation of the soft outer shell, either the human skin or a 3D mesh. Therefore, if you think of rigging a character's arm, you would want to place a joint simulating the clavicle, the humerus and one for both the radius and ulna. Because a rig can be seen as a simplified version of a real skeleton, we only need one instead of two joints to replace the radius and ulna for the lower arm.

- Skinning

  Skinning is the process of taking the joints or bones of the rig and binding them to the actual 3D mesh. This binding process determines over which vertices each joint has an influence. Every vertex can be influenced by one or even multiple joints, which need to be weighted. This process allows the model's mesh to be moved and deformed by manipulating the bones.

To simplify and speed up the whole process of creating an avatar, an automated solution provided by Adobe's Mixamo was utilized. This software is an online tool to create a humanoid rig for a human mesh. It automates the process of creating the joints and skinning the mesh for the user using machine learning methods. On top of that, the platform additionally offers a wide variety of motion capture .bvh recordings, which can be used to animate the rigged model inside the engine. The platform offers to export the model in an fbx file format in a T-pose. This T-pose is also required in our scene, when we import an avatar if we want to use Kinect skeleton tracking.

Figure 4.2 shows the mesh in its initial state after the reconstruction by Autodesk ReCap on the left. The model in the middle shows the untextured refurbished model after it was reduced, re-sculpted and re-textured, but for better comparison and a better visual understanding of the underlying mesh without textures. On the right the final textured result is shown, as it is used in the training system.

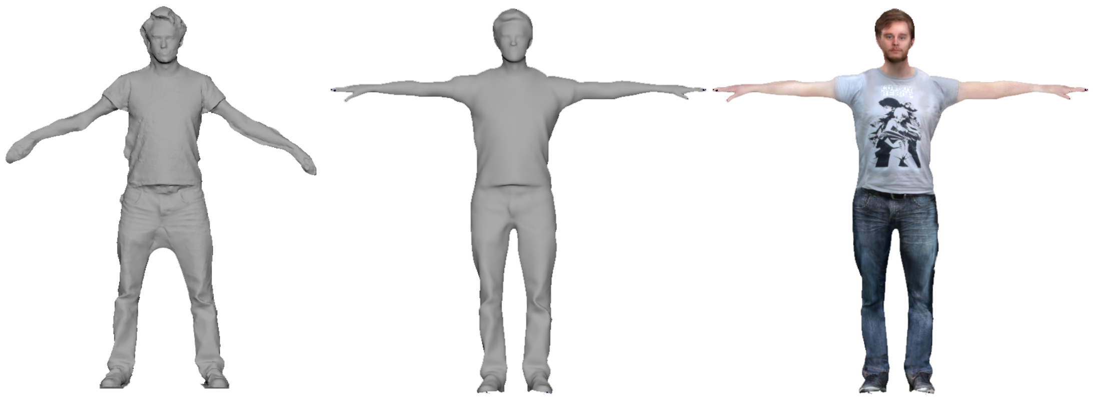
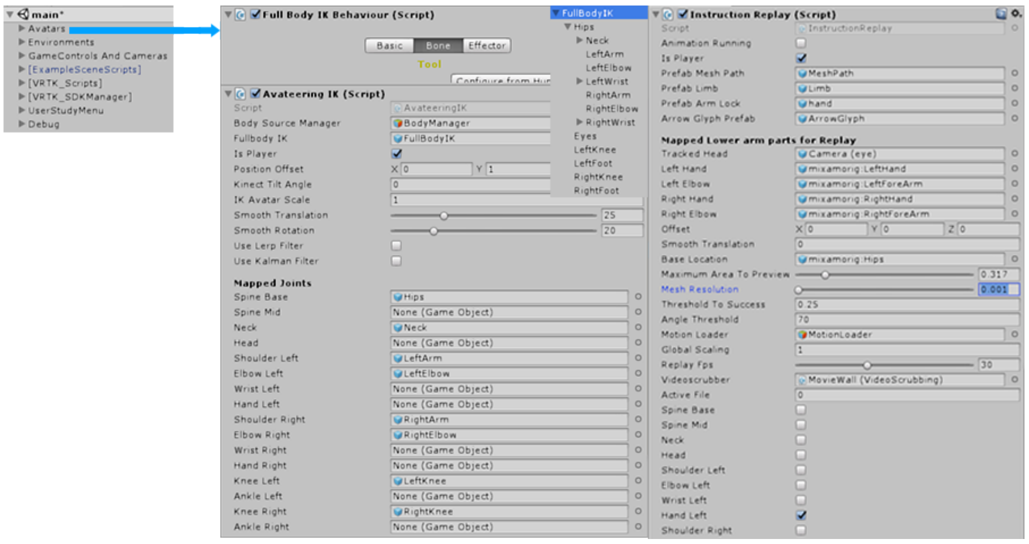


Figure 4.2: Comparison of the initial reconstructed mesh using photogrammetry by Autodesk ReCap (left), the refurbished one un-textured (middle) and the refurbished textured (right).

4.3 Virtual Environment

The main GameObjects in this thesis' system are objects for the avatars, the tracking, logging, virtual reality support, environments and the recording and playback system. The main structure is shown in Fig. 4.3. The scene avatars have components attached for handling the tracking and animation of the avatar itself and a component responsible for the synchronized playback and visualization of the playback with glyphs.



```
main*
  Avatars
  Environments
  GameControls And Cameras
  [ExampleSceneScripts]
  [VRTK_Scripts]
  [VRTK_SDKManager]
  UserStudyMenu
  Debug
```

```
Full Body IK Behaviour (Script)
  Basic | Bone | Effector
  Tool
  Configure from Hu

FullBodyIK
  Hips
    Neck
    LeftArm
    LeftElbow
    LeftWrist
    RightArm
    RightElbow
    RightWrist
  Eyes
  LeftKnee
  LeftFoot
  RightKnee
  RightFoot
```

```
Avateering IK (Script)
Script                  AvateeringIK
Body Source Manager     BodyManager
Fullbody IK             FullBodyIK
Is Player               ✓
Position Offset         X 0    Y 1
Kinect Tilt Angle       0
IK Avatar Scale         1
Smooth Translation      25
Smooth Rotation         20
Use Lerp Filter         ☐
Use Kalman Filter       ☐

Mapped Joints
Spine Base              Hips
Spine Mid               None (Game Object)
Neck                    Neck
Head                    None (Game Object)
Shoulder Left           LeftArm
Elbow Left              LeftElbow
Wrist Left              None (Game Object)
Hand Left               None (Game Object)
Shoulder Right          RightArm
Elbow Right             RightElbow
Wrist Right             None (Game Object)
Hand Right              None (Game Object)
Knee Left               LeftKnee
Ankle Left              None (Game Object)
Knee Right              RightKnee
Ankle Right             None (Game Object)
```

```
Instruction Replay (Script)
Script                  InstructionReplay
Animation Running       ☐
Is Player               ✓
Prefab Mesh Path        MeshPath
Prefab Limb             Limb
Prefab Arm Lock         hand
Arrow Glyph Prefab      ArrowGlyph

Mapped Lower arm parts for Replay
Tracked Head            Camera (eye)
Left Hand               mixamorig:LeftHand
Left Elbow              mixamorig:LeftForeArm
Right Hand              mixamorig:RightHand
Right Elbow             mixamorig:RightForeArm
Offset                  X 0    Y 0    Z 0
Smooth Translation      0
Base Location           mixamorig:Hips
Maximum Area To Preview 0.317
Mesh Resolution         0.001
Threshold To Success    0.25
Angle Threshold         70
Motion Loader           MotionLoader
Global Scaling          1
Replay Fps              30
Videoscrubber           MovieWall (VideoScrubbing)
Active File             0
Spine Base              ☐
Spine Mid               ☐
Neck                    ☐
Head                    ☐
Shoulder Left           ☐
Elbow Left              ☐
Wrist Left              ☐
Hand Left               ✓
Shoulder Right          ☐
```

Figure 4.3: Main structure of the Unity project, scripts attached to the scene avatar for our motion capture animation system and the effector GameObjects for the IK system.

Avatar

The avatar is a rigged humanoid model created using photogrammetry as described in section 4.2.1. The three components attached to the avatar are the FullBodyIKBehaviour, the AvateeringIK and the InstructionReplay scripts. FullBodyIKBehaviour is a sophisticated *IK* solution and described in detail in section 4.3.1. In combination with the provided tracking data, it solves the pose of the humanoid avatar by only being required to provide 9 tracking points out of a maximum possible 24 *IK* handles. The second part of the animation is the attached AvateeringIK script, which provides the additionally required Kinect skeleton tracking data. Section 4.3.1 shows how tracking data for each available joint is retrieved from code in more detail. For proper use, the retrieved tracking data had to be adjusted. The data which is retrieved is mirrored and therefore the x value of each joints position got inverted to reverse the mirroring. The rotational values also had to be mirrored which is achieved by inverting the y and z values of the quaternion.

```
new Quaternion(rot.X, -rot.Y, -rot.Z, rot.W);
```

The script additionally has two filtering methods to smooth tracking data, which is only used for the Kinect tracking. We don't require to smooth tracking data received by the Lighthouse tracking system. As elaborated in section 3.1.2 the Vive tracking noise is only on a sub-millimetre level and therefore accurate enough without any additional filtering. The implemented filtering techniques are a linear interpolation (LERP) and a Kalman filter as proposed by R.E. Kalman in 1960 [14]. The performance impact of linear interpolation filtering is negligible, whereas the Kalman filter has a noticeable impact and made it not possible to maintain the required 90 frames per second (FPS) consistently. While the linear interpolation had no negative effect on our performance, its nature is to cause lag. Depending on the strength of the interpolation, the tracking becomes less responsive and inert. The script is versatile and customizable to change the amount of *IK* handles driven by the Kinect skeleton tracking, where other trackers are not available. The tracked skeleton joints can alternatively be mapped directly to the avatar skeleton joints of its humanoid rig. The rotation of the Kinect skeleton joints and of the humanoid rig don't match in T-pose. Therefore, the rotational tracking data of the Kinect is being adjusted to match the initial rotations of the avatar. Adjustments have to be made for the upper left, upper right, lower left and lower right body hemisphere.

- upper left

```
rotation = Quaternion.Euler(0, -90, -90) * rotation;
result = new Quaternion(-rotation.y, rotation.z, -rotation.x
    , rotation.w);
```

- upper right

```
rotation = Quaternion.Euler(0, 90, 90) * rotation;
result = new Quaternion(rotation.y, rotation.z, rotation.x,
    rotation.w);
```

- lower left

```
rotation = Quaternion.Euler(180, 90, 0) * rotation;
result = new Quaternion(-rotation.z, -rotation.y, -rotation.
    x, rotation.w);
```

- lower right

```
rotation = Quaternion.Euler(180, -90, 0) * rotation;
result = new Quaternion(rotation.z, -rotation.y, rotation.x,
     rotation.w);
```

The avatar we use in this project is called *Laurenz_clean*. It uses the Unity tag *Player* to make the GameObject more accessible via script. The avatar itself has all the necessary scripts attached for animating the character in real-time and for visualizing the instructions via glyphs. To animate the character we use two scripts called **FullBodyIKBevahiour**, which is a third party extension sypplied by SAFullBodyIK, as described in section4.3.1, and **AvateeringIK**. FullBodyIKBehaviour is a script built upon Unity's inverse kinematic solution to improve its result. Although it is free, it yields good, visually appealing results comparable to other solutions in the asset store for which you have to pay. It is also easy to set up. This script adds *IK* control GameObjects in a child hierarchy. These children are the manipulation objects for the inverse kinematic solution. The joints for the Head, LeftWrist, RightWrist, LeftFoot and RightFoot have an additional script attached IKTarget. This is needed for special treatment of those joints, because they are controlled by the Vive tracking system instead of the Kinect skeleton tracking. For the hands, there is a rotation correction in place to match the hands' rotation with the Vive controllers. Furthermore, a system is in place to support the third-person view, to allow the avatar arms to move correct while the Vive controllers are out of place. The hand joints are controlled by the controller positions. Therefore, we need a proper offset for the third-person view. The wrists and feet are mapped to the corresponding GameObjects, representing the Vive controllers, and the Vive trackers used for the feet.

The AvateeringIK script attached to the avatar controls the remaining specified joints using the Kinect skeleton tracking to further improve the IK result for the knee and elbow positions. This addition, compared to a solution only using tracked controllers, yields a correct placement of the subjects elbows and knees, whereas without that additional positional information the *IK* system would have to calculate and guess these positions. It is used for the tracking of the user's hip joint, because the system is missing an additional Vive tracker for that purpose. The Kinect skeleton tracking is not as accurate or fast as the Lighthouse tracking solution for Vive, but it is sufficient enough for the purpose of tracking the user's elbow and knee positions for our inverse kinematic solution. The visual result is accurate and convincing enough to the user. The hip-tracking of the Kinect tracking is reliable enough for our design.
In case there are no Vive trackers available, or if the system is used in a different VR environment, like the Oculus Rift, we implemented an avateering solution solely relying on the tracking capabilities of the Kinect.

BodyManager

The BodyManager GameObject is the integration of the Microsoft application program interface (API) and responsible for the Kinect tracking. *"An API is a set of routines, protocols, and tools for building software applications. Basically, an API specifies how software components should interact."* [5] We can track 25 different joint positions and rotations measured in meters. The GameObject is also used on the attached scripts of the avatar responsible for the avateering. The BodyManager acquires the latest frame from the Kinect live stream and takes the body data of the first detected skeleton from the sensor. Support for only one tracked skeleton was implemented although the Kinect is capable of tracking up to 6 individuals simultaneously. This tracking data is passed to our avateering scripts, AvateeringKinect and AvateeringIK. The sensor has a refresh rate of 30 Hz, limiting the tracking and animation of the avatar to 30 *FPS*, whereas our system is targeted at a frame rate of 90 *FPS* to match the refresh rate of our *HMD*. Furthermore, the tracking data is also consumed by our recording script responsible for recording the motion capture of the scene avatar.

Motionloader

The Motionloader has two scripts attached responsible for loading saved motion capture-data. This data can either be recorded by our system in a previous session by a user or a trainer, or it can be reconstructed skeletal data using a 3D de-projection method consisting of 2D data from OpenPose and a re-projection into 3D space. The motion capture data is read from a custom txt file storing rotational and positional data of all 25 joints, for each recorded frame and is stored in memory as an array. Each recording is recorded with 30 frames per second. The reconstructed data using OpenPose and our deprojection does not have any rotational data. This data can therefore only be used in conjunction with our *IK* avateering system.

Environment

The environment object is structured to hold all environment specific GameObjects and models. For our interior we implemented a few different sceneries. During our user-study we used our cosy living room model, filled with furniture. It features 2 couches,a couch table, a cupboard and a working table with a PC. In front of the wall the user is facing towards, we placed our virtual TV set, which can be hidden when not required. Alternatively the virtual mirrors can also be turned on and off to be shown to the user on the same spot. The scene is lit by two directional lights and an high-dynamic-range (HDR) image used as a skybox.

Game controls and Cameras

This GameObject controlls all user input and stores the two cameras necessary for the multiperspective mirrors. We support multiple input sources, like the HTC Vive controllers, the keyboard and an Xbox 360 controller for PC. The KeyboardInputHandler component attached to this object, is responsible for all inputs and is also triggered by our radial menu attached to the left Vive controller.

It does switch the camera between first- and third-person, switch between loaded instructions on-the-fly. The camera offset for the third-person camera is set with this script, which in our case is 0.2 m in the y-axis and 1.5 m in the z-axis. It also controlls all visual instructions. We can show or hide each visual style, like the Video, the multiperspective mirrors, the avatar-in-miniature (AIM) or the 3D glyphs. It also toggles the recording of a new motion-capture. The motion-capture script handling actually storing the data is attached to the same object. It allows to select a file name, and which individual joint the user wants to record.

4.3.1 Frameworks

The application for this thesis uses multiple open-source frameworks for *VR* and Kinect tracking support within Unity and a solution for inverse kinematics. The SteamVR library is required to fully support rendering on the used Vive *HMD*. SAFullbodyIK is an open-source solution for inverse kinematics, which is used to animate the virtual avatar fusing Vive and Kinect tracking. The Kinect for Windows SDK 2.0 is required to access the Kinect. It includes the driver for the Kinect, an *API*, samples and Unity support.

Kinect for Windows SDK 2.0

The Kinect for Windows SDK 2.0 is required for its driver and interface to communicate with the Kinect. For our purposes we only needed the provided BodySourceManager script to expose body data from the Kinect skeleton tracking. The script establishes a connection to the plugged in Kinect 2. In each frame update, it acquires the RGB frame and the skeleton data for each tracked body within the frame. The SDK allows to track up to eight bodies at the same time, but for our study we only support the tracking for one individual.

```
 _Sensor = KinectSensor.GetDefault();
Body[] data = new Body[_Sensor.BodyFrameSource.BodyCount];
foreach (var body in data)
  {
    if (body == null)
    {
      continue;
    }
    if (body.IsTracked)
    {
      var rot = body.JointOrientations[i].Orientation;
      //mirror X Axis
      kinectRotation =
      new Quaternion(rot.X, -rot.Y, -rot.Z, rot.W);
    }
  }
```

SteamVR

SteamVR is an Software Development Kit (SDK) released by Valve with the intended goal to provide a single interface to support all major *VR* headsets. The *SDK* for Unity provides the developer access to all tracked devices, like the headset, the controllers and additional trackers. It also provides access to the chaperone, which is a safety feature to warn the user, when leaving the defined safe play area. The package also includes interactive, high quality models for the tracked devices. We need the SteamVR *SDK* to get access to the *HMD* and the controllers. It also provides us an already set up prefab for the camera and interactive 3D models for the controllers. Unity and the SteamVR

SDK take care of proper stereo rendering. The scene is rendered twice, once for each eye, with a default eye distance of 64 mm. It provides scripts for the tracking of the headset, the controllers and the additional trackers. These scripts are attached to the respective GameObject. The system does not automatically choose the correct index of the correct tracker in the script when starting the application, except for the HMD tracker. Left and right Controller can be swapped and the correct index for the trackers attached to the feet have to be set manually.

VRTK

The VRTK asset found on the Unity asset store, is a valuable bundle for our system. It provides scripts for many default interactions within *VR*. In our system it took care of setting up the camera environment, with the proper hierarchy and would allow to setup another SDK, like the Oculus SDK and switch between both during run-time. In the hierarchy, the parent object is a representation of the allowed tracked space, also displaying the chaperone. The chaperone is a security system provided by the HTC Vive, displaying a blue wire-frame wall to warn the user when he is getting too close the wall in the real world, to prevent injuries. The child objects are the controllers for the left and right hands, our camera, used for our first-person perspective (1PP) and third-person perspective (3PP) and objects for the feet trackers. The tool kit provides an event system, for all inputs of the controllers, like button presses. We also used the VRTK for the implementation of a radial menu on the left controller, to provide full control over the scene from within the *VR* environment. The controls include the visibility settings for the multiperspective mirrors, the video instruction, the *AIM*, activating the 3D glyph instruction animation, switching between first- and third-person perspective, scaling the avatar and starting and stopping a new motion capture recording.

SAFullbodyIK

As an *IK* solution we used the SAFullBodyIK[2] system found on GitHub, a development platform for hosting projects. All use of this software is without charge and released under the MIT license. It provides an IK solution for the body, fingers, head and limbs. We used the solution for the whole body. When providing a humanoid rigged model, like we did, the system automatically detects the correct joints and no further set up steps are required. If required, all joints can also be set manually. The system provides effectors for the limbs and the head. We used the effectors for the hip, left and right wrists and elbows and left and right feet and knees.

[2] `https://github.com/Stereoarts/SAFullBodyIK`. Visited on October 15th 2018.

4.4 Visualization, recording and playback

The system developed for this thesis handles the visualization of instructions within a *VR* environment. To obtain the necessary data required for these instructions, capabilities to record natural movement of an instructing person can be recorded by the system itself, using the same tracking technology used to animate the virtual avatar. The recorded data can then be utilized in a number of ways, for playback, or as positional data for our multiple visualization techniques.

4.4.1 Perspective

In our environment we used two different fixed perspectives. Our user can chose between a *1PP* and a *3PP*, depending on the requirements of the task. In our study, our participants had to use both perspectives to follow both our introduced instructions. We implemented a toggle, controllable both by the user from within the environment and by the conductor of the study from outside. The toggle switches the user camera between the two described perspectives.

First-person perspective

The *1PP* places the scene camera at the center between both of the avatar's eyes. Initially we enforced the camera to follow this center. A slight difference between the avatar's head position and the real person's head position, tracked by the *HMD* was noticeable by the user and led to some discomfort and motion sickness. We therefore only position the camera at the eye center when toggling the *1PP*. Because of the overall accuracy and natural behaviour of our avatar, the tracked viewer camera and the avatar's head center keep almost perfectly aligned and the camera never drifted outside during our study.

Third-person perspective

When toggling the *3PP*, the camera teleports the users out of their body, into a third-person experience. The user is placed 1.5 m behind his or her avatar and slightly elevated by 20 cm. Unlike a *3PP* in video games, the camera does not get an offset to either side. Whilst a user in a traditional video game is focusing their attention at a distant point in front of the avatar, and looking past the avatar, our focus is the avatar itself and its close surrounding. Since the user will be looking at the avatar most of the time we did not give the camera an offset to the left or right, to avoid the user constantly tilting his or her head or even rotating his or her body in the direction of the avatar.

4.4.2 Record and Replay

We wanted our system to be able to record motion-capture data from our real-time tracking. We implemented a recording system, which stores all rotational and positional data from our tracked avatar's humanoid skeleton joints. The file structure is built as follows: Each recorded frame is stored in a new line with an interval of 33 milliseconds for a target speed of 30 frames per second. To assure a consistent interval in the recording, Unity's built in FixedUpdate method was used. FixedUpdate ensures a fixed frame-rate and is usually used for physics-based calculations. The positional data is stored in a Cartesian coordinate system. The unit scale is in meters. The rotations are stored as quaternions. Values of all joints from the visible avatar within the scene are stored. The format looks as follows for every stored joint

$$p(C)r(Q)$$

where 'p' is the tag for the following position stored in 'C' as Cartesian coordinates, 'r' being the tag for the following rotation stored in 'Q' representing quaternions. A total number of 25 joints matching their counterparts from the Kinect skeleton tracking are saved for each recorded frame. These joints are SpineBase, SpineMid, Neck, Head, ShoulderLeft, ElbowLeft, WristLeft, HandLeft, ShoulderRight, ElbowRight, WristRight, HandRight, HipLeft, KneeLeft, AnkleLeft, FootLeft, HipRight, KneeRight, AnkleRight, FootRight, SpineShoulder, HandTipLeft, ThumbLeft, HandTipRight and the ThumbRight.

After the motion from an instructor has been stored, we can use the data in multiple ways. The first way to use the data for visualization purposes is to apply all recorded positional and rotational data to a second avatar within the scene and create a virtual instructor for the user. This instructor would be similar to a real one, demonstrating the desired body motion for a training user to imitate. With the recorded data we stored, we are able to animate the instructor avatar with the same both techniques also used for the user's avatar. This includes either animating him by applying the rotational values from our recording to each corresponding joint, or by also using the same *IK* system driven by our stored positional information, both with some advantages and disadvantages. Animating our virtual instructor using only rotational values for each joint, yields comparable results in posture for different sized models, but will lead to differences in extremities world positions for changing proportions. This does not happen when using the *IK* system. Since every user can be of different height, we need to address this and change the size of the avatar's model. After re-sizing the avatar, we also have to proportionally scale the recorded data for our 3D glyph visualization.

Recorded Data

We load stored motion capture data at run-time. Our designated script for that is LoadBodyMotion.cs. It supports loading of multiple files and stores the data in an array of nested lists. The data structure is as described below.

```
List<List<Quaternion>>[]
```

We create a new element in that array for each unique file. The inner list stores data of each of the 25 individual joints in a new element. Every element of the outer list is a new frame. Data[0][0][0] therefore accesses in the first file loaded by the script the first frame's first joint (SpineBase). (Data[file][frame][joint]). We use two different arrays for positional and rotational values, although introducing a tuple would also have been a valid solution. The data is used by all visual guidance systems. Most systems use the positional data only, whereas rotations are only used by the avateering script built upon the Kinect data to animate a virtual instructor represented by an arbitrary humanoid avatar.

OpenPose Data

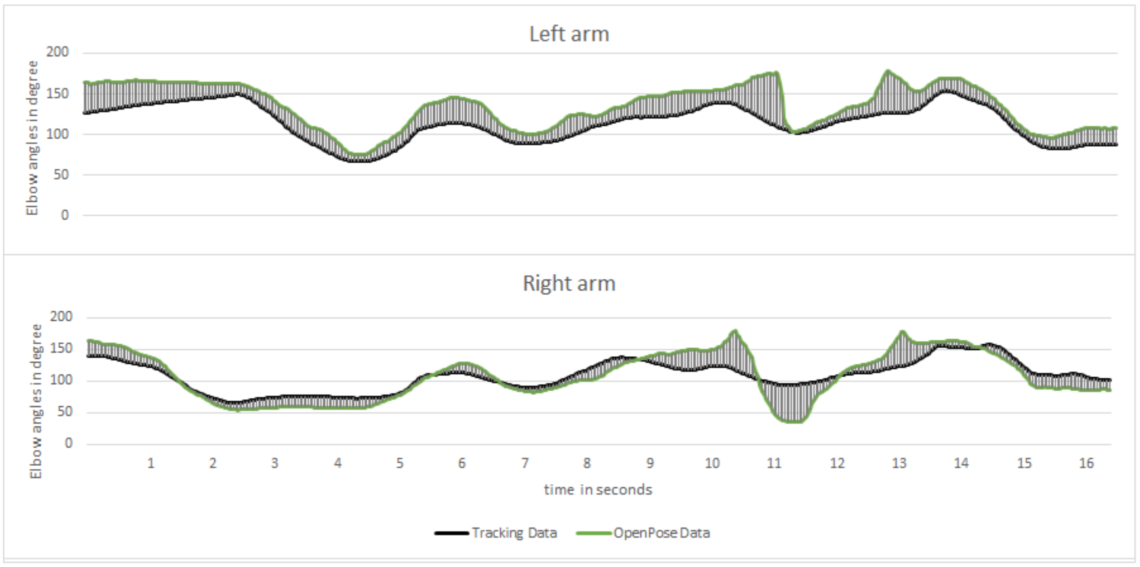


Figure 4.4: Comparing the elbow angle between extrapolated OpenPose data and our *IK* motion-tracking

The system also supports the possibility to use positional data for avatar joints from OpenPose, extrapolated to 3D. Figure 4.4 shows how the angle between the upper arm and lower arm of a recorded avatar using our recording system changes compared to OpenPose data for the exact same motion. The instructing person performing the motion was simultaneously recorded by our recording system and by the Kinect's RGB camera, ensuring a temporal and spatial synchronization. A constant offset between the two angles can be expected, as the disadvantage of our *IK* system is, that it does not account for the user's natural bend of their elbows, but instead it fully stretches the arm, generating a 180° angle between upper and lower arm. This results in an overstretched and less natural relaxed elbow angle.

For the OpenPose data, we used a video recording of the exact same movement, which was used for the recording with our motion capture system. We recorded the color stream of the Microsoft Kinect, therefore ensuring that we use a recording of the same motion for OpenPose, from the same perspective the user is tracked by the skeleton tracking with the same frame rate. The graphs show a clear correspondence between both animations.

The two spikes are caused by moving the hands towards the camera, which results in a bad guess of the angles by the OpenPose system.

4.4.3 Animating the avatar

To animate a 3D avatar, the system supports two different implementations. In the first one, the avatar is animated using only the rotational values of the Kinect skeleton tracking provided by the Kinect for Windows SDK 2.0. This version does not require any additional tracking. The script allows the user to individually map each tracked skeleton joint of the Kinect *API* to the corresponding joint of an avatar. This avatar has to have a humanoid skeleton and must be imported into the scene standing in a T-pose to be properly mapped to the Kinect data.

The newer version uses the Unity *IK* system, further enhanced by the open-source SA-FullBodyIK framework. Of the available *IK* effectors, the thesis' system only uses those for the hip, head, left wrist, left elbow, right wrist, right elbow, left foot, left knee, right foot and right knee. Head, wrists, and feet effectors are driven by the more accurate Vive tracking. Hip, knees and elbows derive their positional information from the Kinect tracking. The system considers different user body statures by adjusting the tracker's position accordingly. The scene avatar is only scaled to match the current user's body height, but not different body proportions to avoid the avatar's mesh to unnaturally stretch or compress. Therefore, the avatar's and user's height might match, but their hand positions might be different. This also has to be implemented because of the possibility that different users might hold the controller in a slightly different way. Since the controller's position is the effector position for the avatar's hand, different offsets in holding the controller would cause a varying effect on the solution of the *IK* algorithm. The necessary offset for the controller tracking position is calculated once and initiated by the user. The user has to strike a T-pose with fully stretched out arms and an approximate elbow angle of 180°. The offset vector is then calculated for the controller to exactly match it with the avatar's wrist joint effector. Knee and elbow effectors use positional tracking information provided by the Microsoft Kinect to allow the *IK* system to properly position the avatar's knees and elbows, so the *IK* does not have to guess them and give the user a more accurate, natural and immersive representation.

4.4.4 Instruction visualization

This thesis implemented multiple possible ways to present instructions to a user. First, we have a stylized visual instruction system focusing on specific joints of the body and on precision, while providing feedback to the user. For that we used 3D glyphs in our system. Second, a virtual TV set is present in the application allowing playback of any video tutorial. Pre-recorded motion capture data can furthermore not only be utilized by our 3D glyph visualization, but can also be applied to another arbitrary avatar, for example substituting an instructor. Moreover, the system is able to combine any of these instructions, giving us the possibility to show an original video instruction on our screen and supplement it with our 3D glyphs, while also controlling playback speed by the user's advancement in following the tutorial.

3D glyphs

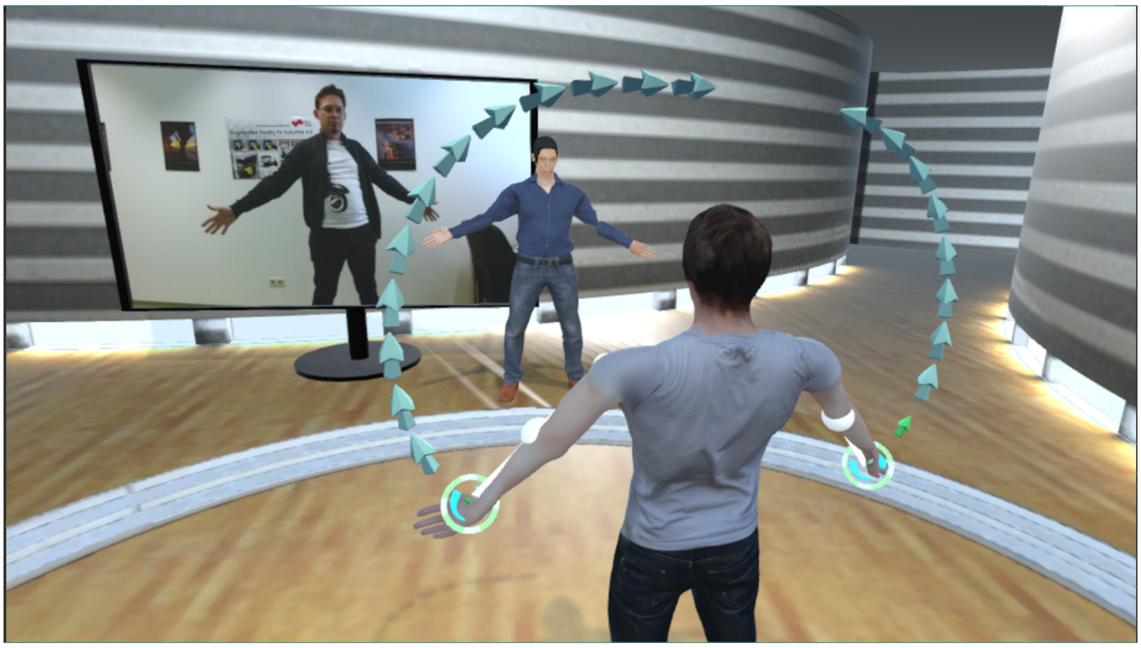


Figure 4.5: The instructions as seen by the user in *3PP* with three different instruction methods turned on. Visible are the 3D arrow glyphs, the video instruction and a virtual trainer as an avatar.

The motions of the hands are instructed to the user using 3D arrows following the recorded path. We use the recorded hip position to map the visualization to the current avatar's hip position. This ensures proper placement of the instruction at all times, avoiding misalignment or wrong spatial positioning of the training user. As mentioned, we use the visible avatar's joints in our motion capture system. This ensures that all target points are always within reach of the user following the instruction. At the start of a training session, both the Avatar and the recorded data can be scaled. To scale the avatar we ask the user to stand in a T-pose. We measure the height of the user, using the *HMD* as a reference point. We then scale the avatar to match the current user's body height. When the avatar is scaled, the data used from our recording is scaled accordingly. The arrows are using the recorded wrist positions and during our evaluation process we compare them to the life avatar's wrist positions. The arrows can be adjusted on the fly to either increase or decrease the spacing between them. Visualizing the motion as a whole would clutter the scene and would be confusing for the user, which is why we implemented possibilities to limit the previewed distance shown by the glyphs. We implemented two automated thresholds to control how much of the coming path are indicated by the arrows to the user at a time. We limited the maximum preview distance of the path and we checked the angle between three consecutive arrow positions, if the two vectors created by them have an angle above 70°. At an angle below this threshold, we assumed a change in direction and limited the preview to this point. The next preview segment will be shown, after the user has reached the current limit.

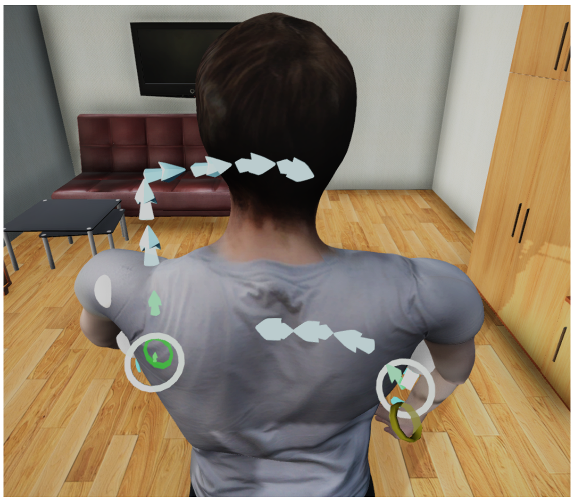


Figure 4.6: The occlusion shading seen from a *3PP* used for the 3D glyphs, wristbands and target circles to help the user find the target path and estimate his or her own hand position, while the instruction is occluded by the avatar.

The circles indicate the current desired position of the wrists and are camera-aligned. They are indicating the current target position of the instruction and are moving along the path with the target velocity of the original recorded motion. Points along the path, where the user already moved along successfully, are not shown and disappear from the preview. An additional help in *3PP* for the user are stick-figure like arms. They are made up of spheres for the shoulders and elbows and lines with a width of 3 cm for the lower and upper arms. This stick-figure like arms show the arm poses for the target pose of the current instruction frame. Their main goal is to help the user understand the pose, for the elbow placement, the arm bend and shoulder placement.

Some motions like our ”Yoga” instruction include upper-body movement. Figure 4.5 shows the glyphs for the first visible segment of the yoga instruction. With this visual help we intend to motivate the user to use his or her whole upper-body, and not only arm movement. The circles indicating the current instruction frame, only continue to move forward if the user is within a 25 cm area around them with both of their hands at the same time. Moving out of this area with only one hand at a time will pause the animation of the circles and stick-figure arms, and will wait for the user to regain the correct posture. Vicinity to the instruction glyphs is indicated by shrinking the arrow’s size linearly within a range of 25 cm and also by linearly interpolating their color from teal to green, as seen in Fig. 4.5 at the avatar’s right hand.

As an additional help to indicate the correct hand position within our 3D environment, the avatar has two wristbands around both of his wrists, rendered with an occlusion shader. Movement in front of the user completely occludes the avatar’s hands while being in a *3PP*. To give the user an approximate indication of their hands’ position during these periods of a motion, we help them with this additional visual guide. The arrows are also occlusion shaded, like our wristbands, so they can be seen when in front of the avatar as shown in figure 4.6. The shading is used in both perspectives.

Multiperspective mirror

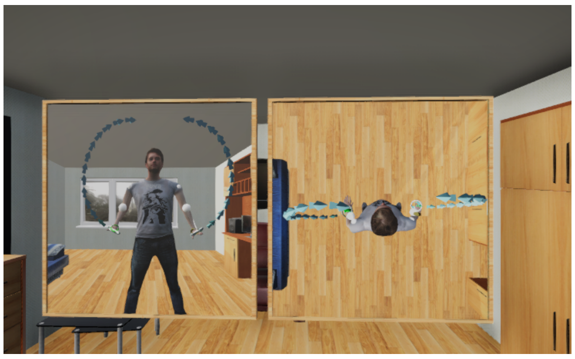


Figure 4.7: The multiperspective mirror seen from a *1PP*. The users see themselves from the front like in a real mirror on the left and from a top-down view on the right panel.

The multiperspective mirror shown in figure 4.7, imitates the augmented reality training environment of [35] where two cameras were placed in the real world to record the user and play back the video stream on a TV set with augmented instructions. The two cameras are facing top-down and the world's z-axis is facing towards the user. Both cameras are played back in augmented virtual mirrors presented in front of the user. The left mirror shows the frontal stream, whereas the right one is showing the top-down. As an advantage of being in virtual reality instead of the real world, our top-down and frontal cameras are able to follow the user, while moving around and they avoid being obstructed by any obstacles in the scene by rendering only what is desired. The recording is furthermore augmented by our 3D glyphs. In a preliminary experiment, we tested the performance of users following an instruction path in the first-person perspective versus the augmented multiperspective mirrors. Showing the instruction only in these mirrors reduces the amount of visible layers to only two (top and front) and showing instructions on a virtual screen also reduces the ability to properly gauge depth and distance to the target. Users also had to switch their focus between both mirrors to realign their hands positions, because the mirrors only offer information of one axis. The first-person perspective showed itself to be superior due to these limitations of the virtual mirrors.

Video instruction

A video can be used to show the original video instruction which was previously recorded or provided by any other source. The visualization is a representation of the widespread use case in the real world of offline training methods without a personal trainer using video material as a resource, like training videos on DVDs or from the web. A 3D model in the shape of a TV set was created for a visually more immersive feel. The video is played back as a movie texture on a quad using Unity's built-in video player. The video player supports the following video codecs on Windows platform: asf, dv, m4v, mov, mp4, mpg, mpeg, ogv, vp8, webm and wmv. H.264 is the preferred supported video codec, because it offers the best compatibility across platforms. The video playback is temporal controlled and synchronized by the instruction sequence and the user's current progress.

Avatar-in-miniature

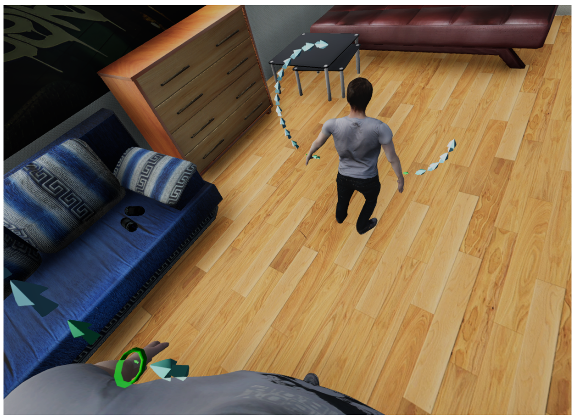


Figure 4.8: The *AIM* as seen from the *1PP*, looking down at the instruction on the left arm.

As a proposition of combining the advantages of *1PP* and *3PP*, we introduced an *AIM* for our *1PP*. Inspired by the concept of Pausch et al.[27], we created a miniaturized replica of our avatar, complete with all visual guides currently active for the user, as shown in figure 4.8. As described in chapter 3.4.7, the world-in-miniature paradigm is a hand-held miniature representation of the surrounding environment designed for locomotion and is an exact copy of the virtual world. While the world-in-miniature is a re-sized version of the virtual environment as a means of locomotion, our *AIM* is a shrunken copy of the user's avatar himself. Our goal with the introduction of a miniaturized avatar to the *1PP*, is to merge the benefits of *3PP* with the benefits of a *1PP*. The avatar is scaled down to 25% of the original's size. It is positioned half a meter in front of the user's eyes and 25 cm to the left, while following the user's point of view to ensure, that the miniature stays within the peripheral vision. The avatar is set half a meter in front of the eyes to stay within reach of an arm length. Interaction with the Vive controller and the miniature avatar was enabled, allowing for manipulation of the avatar's position by grabbing. To drag the avatar, the controller has to be moved over the *AIM*, until it is highlighted to indicate that the sphere of influence is entered by the controller. Holding down the trigger enables the user to grab the miniature and to drop it upon release anywhere in the virtual world, for re-positioning. The miniature copies all movements of the real avatar together with a re-sized copy of the current instruction visualization, if 3D arrow glyphs are currently active.

4.5 Hardware

This section covers the hardware used for the system developed for this thesis. We will discuss in detail how to set up the HTC Vive, the tracking devices and the necessary steps to ensure reliable tracking.

4.5.1 Microsoft Kinect

The Kinect is placed about 2.5 m in front of the user, facing the person in a straight line and approximately 1 m above ground. The Kinect sensor is mounted on top of a tripod absolute level to the ground. Tilting the sensor would lead to a tilted skeleton tracking, which we don't want. In most studies like those of Wang [39] and Livingston [20], this positioning is said to be within the optimal range and offset for human tracking. In respect to different heights of different users, the Kinect sensor can be height-adjusted using the tripod, to avoid the virtual avatar from floating above the environment's ground floor or being placed underneath it. The Kinect sensor has to be placed so that it does not directly point in the direction of a Lighthouse tracking sensor. The infrared Light Emitting Diode (LED) array of the Microsoft Kinect depth sensor is disruptive for the infrared light-sweep of the Lighthouse tracking sensor, disabling its tracking.

4.5.2 HTC Vive

The HTC Vive comes with 2 tracked controllers, which are held in both hands and a head mounted display, which already provides us three tracked joints of the human body. We further expanded that number of tracked body positions by adding two additional Vive Tracker to the environment to additionally track the user's feet. As previously described in subsection 3.5.2, the HTC Vive *HMD* has two OLED displays. One for each eye with a resolution of 1080 x 1200 pixels per eye. The display has a refresh rate of 90Hz and offers a field of view of 110°. The setup has two Lighthouse sensors for room scale tracking. In our setup we had a space of 2 x 3 m available for the user. The two controllers are wireless and support multiple input methods, which include a track-pad, a grip button, a dual-stage trigger and two additional buttons. We used the track-pad to implement a radial menu for the user to access controls to change the perspective, activate our different visualizations, start a recording session and start a training session. With the access to a very accurate and reliable positional information of all four extremities of the human user, we were able to also implement an accurate and visually convincing inverse kinematic solution. The tracking of the hip joint continued to be tracked by the Kinect sensor. The Kinect sensors skeletal tracking was furthermore also utilized to keep track of knee and elbow joints. These joints were used for the corresponding IK effectors.

4.5.3 PC

Our implementation is running at a stable 90 *FPS*, matching the refresh rate of the *HMD* on our hardware. We used a consumer mid-range level computer running on an Intel core i5 6600, 12GB of RAM, rendered by an Nvidia GeForce GTX 970. The render settings in our Unity scene were set to high.

5 User study

Contents

In this chapter we will show the results of a user study conducted to investigate the impact on changing the user perspective in virtual reality (VR) and their respective advantages in different scenarios. We will also show results by a preliminary study in a similar setup, which was conducted as part or a master project. We invited 12 users to analyze head movement while following an instruction and the accuracy of their hand movements, both for first- and third-person perspective using two different movements, with one focusing on wide sweeping gestures and the other one focusing on hand movement primarily in front of the user, occluding the instruction path by the virtual avatar, making it only visible by our occlusion shader. For the accuracy of the hands the offset of the wrist joints deviating from the target path were measured. Head movement is measured in angles for roll, pitch and yaw. Additionally, the total amount of rotation in degrees and the time used to finish the task were logged. In chapter 5.2 we will give a detailed description of the study setup. The results and analysis of the questionnaires and collected data will be presented in chapter 5.3.

5.1 Preliminary study

A preliminary user study was conducted as part of a master project leading to this thesis. User performances following an instructed path from a first-person perspective (1PP) with 3D arrow glyphs and a mirror setup, comparable to augmented reality (AR) mirror setup by Tang et al.[35] were measured and compared. Goal of this preliminary study was to explore the potential of motion visualization in 3D space from a *1PP* point of

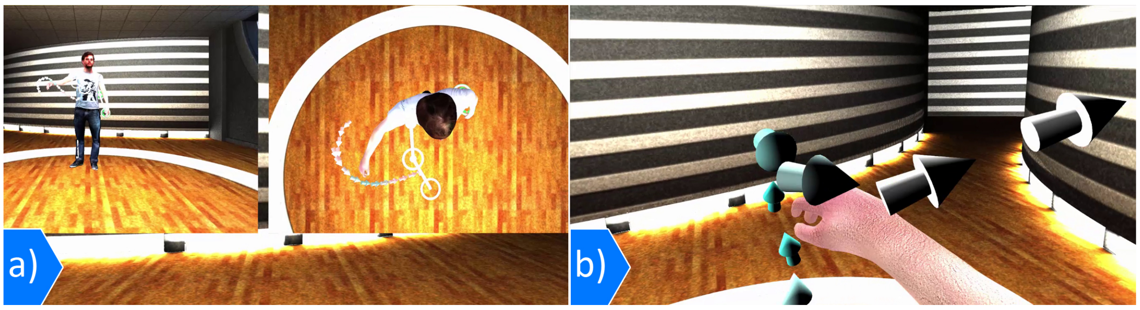


Figure 5.1: User evaluation. (a) We present motion instruction using the AR mirror visualization technique. To provide the user with the same hardware setup in both conditions, we render a virtual AR mirror in front of the user in a virtual environment. We use a split screen setup on the AR mirror providing a top-down and front view. (b) We compare the 2D projection AR mirror to our egocentric 3D visualization of body instructions. We present 3D body instructions using 3D arrows registered in 3D space to the user's skeleton.

view. We evaluated the effectiveness of egocentric 3D body instructions by measuring task completion time and error rate. Our egocentric AR (EAR) visualization technique shown in figure 5.1(b) was compared to a re-implementation of the AR mirror (ARM) by Tang et al., which presents body instructions on a large screen in front of the user from a front and top down view (figure 5.1(a)). The visualization used in EAR resembles the 3D arrow glyph visualization also used in the presented system. We attached the camera, which renders the environment to the user's head position and track their movement using a Microsoft Kinect. The visualization in ARM mimics an AR mirror, showing a live video with augmentations on a split-screen in front of the user (Figure 5(a)). The split-screen does provide the user with two views, a top-down and a front view.

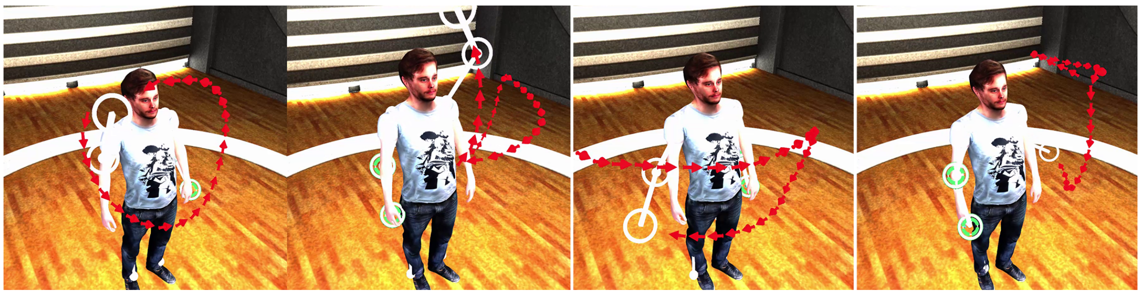


Figure 5.2: The four tasks we asked the users to follow for our preliminary user study, shown from a third-person perspective

Setup

To avoid any user bias, we use the same setup for EAR and ARM. The head tracker integrated into the Oculus Rift display maps the user's head motion to camera motion in 3D space. We track the user's skeleton with a Microsoft Kinect. The system was implemented in OpenGL using the open source tracking framework NiTE, providing rotational and positional tracking data for 15 joints. The rotational data was applied to the avatar for animation purposes. Our implementations ran at a stable 75 frames per second (FPS) on a PC equipped with an Intel Core-i7 CPU and an NVidia GTX 680 GPU.

Tasks

Each participant was asked to follow four motions, shown in figure 5.2, using EAR and ARM. As 3D motions are mostly demonstrated in video tutorials, we are specifically interested in testing the performance of our visualization for 3D movements. Therefore, we explicitly avoid motion within a single plane, such as used in the Physio @home *AR* example.

Results

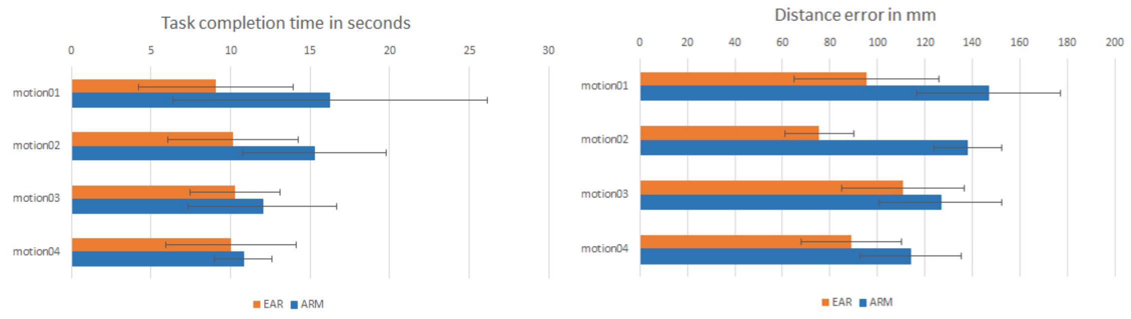


Figure 5.3: User Tasks. In our experiment we ask the users to follow the 4 illustrated motion paths in 3D space.

A total of 12 subjects (9 male, 3 female) aged 22-32 participated in our experiment. The average time taken to complete each task and the average error while following the four motions are shown in figure 5.3. The collected data shows, that a 3D instruction viewed from a *1PP* outperforms the ARM setup. Our quantitative measures indicate great potential of an egocentric visualization for 3D motion instructions. We also asked users about their personal preference for EAR or ARM. Eleven out of twelve participants preferred EAR over ARM, while the remaining participant was not in favor of either one. A common comment from the participants was that EAR allows to travel along the path much more self-reliant, while ARM requires looking at one's own body and the mirror concurrently. A few subjects noticed some jitter of the tracked skeleton, but did not feel obstructed in solving the task. When comparing the results from figure 5.3 and figure 5.8, an improvement on accuracy in our new system can be seen, due to the improved tracking, which removed the jitter, that was noticed by users in the preliminary study.

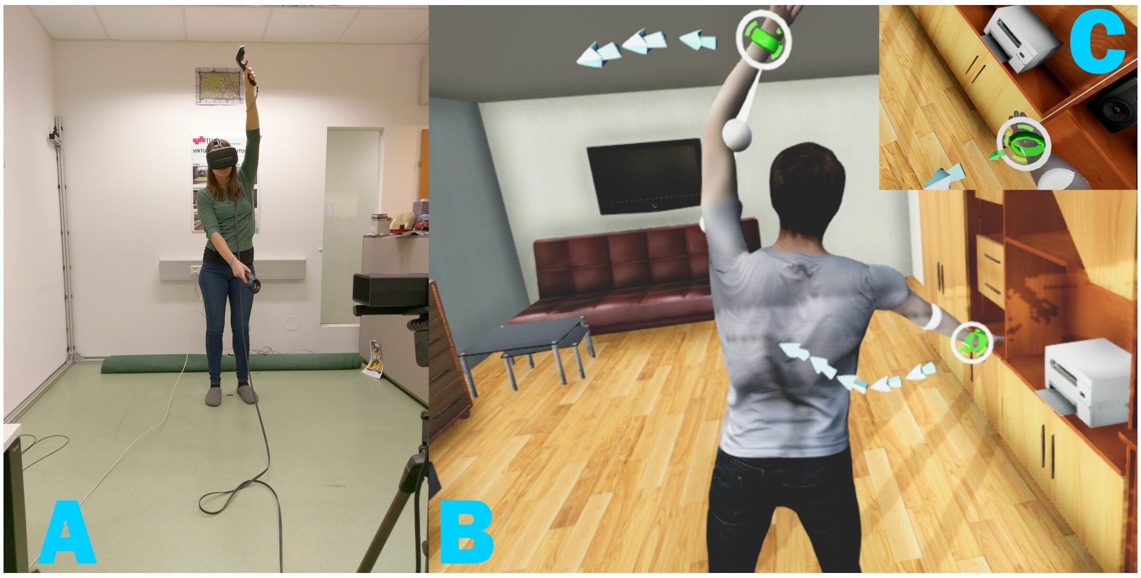


Figure 5.4: (A) Showing the study participant in the tracking environment, standing in front of the Kinect™ wearing the HTC Vive head-mounted display (HMD). (B) The user's perspective from within the *VR* environment during the yoga motion instruction from her third-person perspective (3PP) perspective. (C) The *VR* perspective of the participant as seen in *1PP*.

5.2 Study setup

In our study we tested our initial propositions. We defined *1PP* and *3PP* as our two conditions in this study. In both conditions, our two motion instructions 'Yoga' and 'Frontal' had to be successfully completed. We invited 12 participants to the study from the campus. As shown in table 5.1, out of these 12, two were female and the remaining ten were male. Their age ranged from 21 to 38 years, averaging at 29.4 years. Most participants had some experience with *VR*, meaning that they have previously at least worn and experienced a *VR* headset. Four rated themselves as very experienced, because of either working in the field of *VR*, or because of owning a *HMD* themself. Also, most participants (83.3%) were students studying in the field of computer science or a comparable program. All of our participants were right handed, and the majority (58.3%) of them had an impaired vision, requiring them to wear glasses. All participants gave written informed consent prior to inclusion in the study.

All participants, except for one, were able to wear their glasses underneath the *HMD*, therefore having no disadvantage in comparison to non-visually impaired participants. None of our subjects suffered from epilepsy or *VR* induced motion sickness. All participants had to fill out a pre-questionnaire before starting with the study (see table 5.1). To conduct our study, all participants were asked to wear the HTC Vive *HMD* and to stand in an open space, tracked by the Lighthouse tracking system provided by the HTC Vive and supplemented by the Kinect™ 2 tracking. The Kinect™ was placed 2.5 m in front of

Question	**Answer**
What is your age?	mean 29.4, SD 5.2, min 21, max 38
How experienced are you in *VR*?	mean 3.5, SD 1.1, min 2, max 5
What is your gender?	male 83.3%, female 16.6 %
Which is your dominant hand?	left 0%, right 100 %
Do you study computer science of anything comparable?	Yes 83.3%, No 16.6 %
Do you have an impaired vision?	Yes 58.3%, No 41.6 %
Do you suffer from epilepsy?	no one

Table 5.1: Pre-Questionnaire

the users' standing position at 1 m height. The participants didn't require to wear any additional hardware beside the provided *HMD* and were free to wear comfortable everyday clothes. Out of the 7 persons who wear glasses, 6 were able to comfortable fit them underneath the *HMD* headset.

After putting up the headset, we introduced our system to each participant. Everyone was instructed on how to operate the system and what to expect. At the beginning, we asked everyone to stand in an upright T-pose for calibration purposes. We scaled the virtual avatar to properly fit the user's real body height and their real arm length to provide an accurate arm behavior. After that, we gave everyone a few minutes to look around the virtual scene, experience both perspectives and to get accustomed to their avatar's movements. After the subject felt ready, we started showing them their first task. Every participant had to finish 4 tasks. We used 2 different motions, both in *1PP* and *3PP*. Both motions are described in more detail in sections 5.2.1 and 5.2.2 and are called 'Yoga' and 'Frontal' throughout the thesis.

The starting condition was randomized and each participant started with a different perspective and motion to diminish the learning effect influencing the results. Participants were allowed to start the logging and animation of the motions instructions to follow by themselves. They needed to press and hold both triggers on the Vive controllers to start the task whenever they felt ready for it. To allow our users to properly follow the instruction, we implemented a few mechanics waiting for the user to catch up with the instruction whenever necessary. We showed the users the proper velocity of the movement and an animated guide they had to follow with both of their hands. These targets had a 25 cm threshold implemented. Whenever one of both hands moved further away from its designated target position, the animation stopped and waited for the user to re-position themselves or catch up with the animation. This was mainly necessary in *1PP*, when the user concentrated on one hand, while the animated path of the other one was also moving. A paused animation indicated the user that the instruction for one hand went ahead of them and that they needed to re-adjust and catch up with it, or that they drifted away from the target with one of their hands, while they shifted their focus away from it. This ensured that both arms were performing the correct motion throughout the whole sequence. In the case of *3PP*, this technique was an additional help for the user, when they deviated from the path too often, because they misjudged depth. Navigating the arm on the z-axis in *3PP* proofed to be more challenging than from a *1PP* and could lead to a wrong depth placement of the hands.

Each task's difficulty was rated using a single ease questionnaire. After finishing each individual task, we asked every participant to rate the difficulty of that task on a scale from 1 to 7. After both motions of one condition were successfully finished, the participant was asked to fill out both, a NASA TLX questionnaire and our post-questionnaire as shown in 5.11. At last, after all four tasks were finished, we asked for the preferred perspective and additional remarks.

5.2.1 Yoga

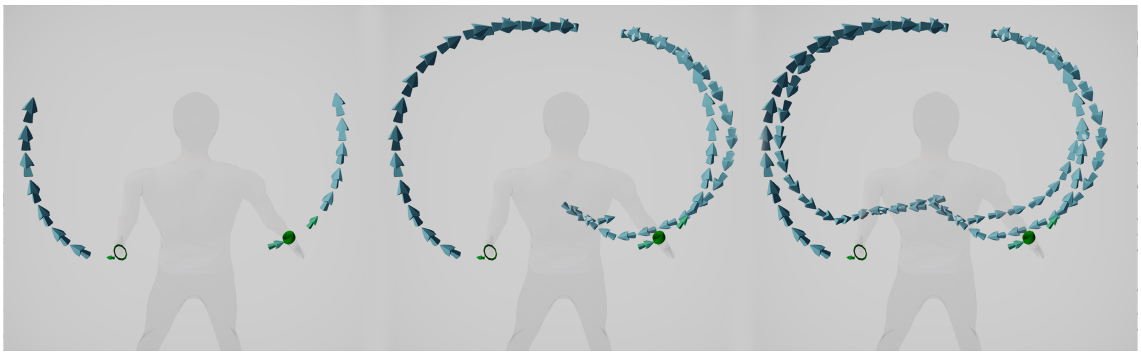


Figure 5.5: The yoga motion depicted with our arrow glyphs, split into three steps for better visual clarity.

The yoga motion instruction consists of a sweeping gesture using both arms simultaneously. The motion is inspired by a yoga movement. Both arms are mostly spread outwards and not visible at the same time from a *1PP*, within our *VR* headset due to its limited 110° Field of View (FoV). Even with a natural *FoV* of 210° , the hands would only be slightly visible within the outer limits of the peripheral vision.

5.2.2 Frontal

Our frontal movement is inspired by physio-therapeutically diagonal shoulder flexion and extension patterns, was extended to involve both arms simultaneously and keep the range

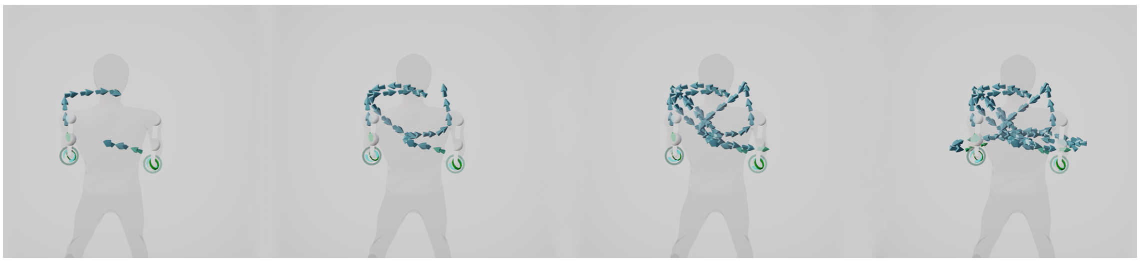


Figure 5.6: The frontal motion depicted with our arrow glyphs, split into four steps for better visual clarity.

of movement within a user's *FoV* of the worn *HMD*, with a diagonal *FoV* of 110°, from a first-person perspective's point of view. The frontal motion is designed to have a preference of *1PP*, by reducing the necessary head movement to a minimum and maximizing the occlusion by the avatar when viewed from a *3PP*.

5.3 Results

After finishing all tasks, we asked each participant about their preferred perspective in our presented system. Out of 12 participants 8 answered that they preferred the *3PP* and 4 preferred the *1PP*. Overall the decision for most participants was a very close one and strongly influenced by the main contributing factors, we expected to have the strongest negative contribution to the experience in each perspective. These were head movement, occlusion, depth perception and virtual body ownership (VBO). Many participants mentioned the required head movement in the *1PP* of the yoga task, and defined it as one of their main deciding factor, why they preferred the *3PP*. The two most commonly mentioned negative factors in regard of the *3PP* were the issue of occlusions caused by the avatar's body and the reduced depth perception. The effect of misjudged depth, was

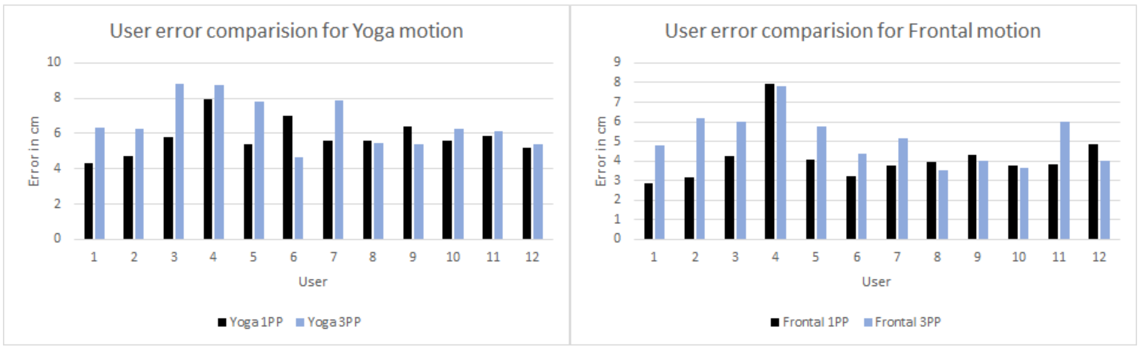


Figure 5.7: Average results of each individual user compared in both perspectives. Left the comparison for the Yoga instruction and right, the comparison for the frontal instruction.

most prominent while observing user 3. This user was also the only one who was not able

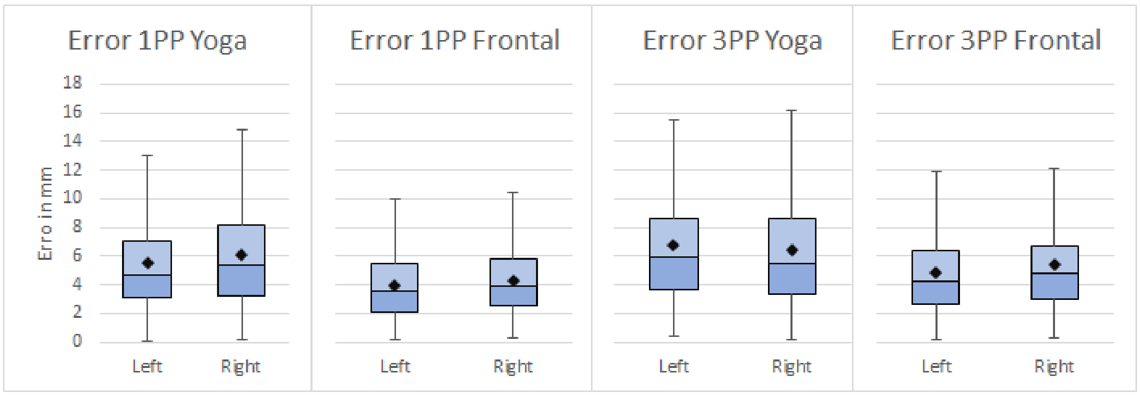


Figure 5.8: Average results of the participant's error for each task for their left and right hand positions measured in mm distance from the augmented instruction path.

to wear their glasses with the *HMD*. This caused a blurred vision from a farther distance as the one in the *3PP* and therefore further decreased the depth perception. This resulted in a much lower than average accuracy during the *3PP* condition. The average deviation of this user for the yoga instruction was 9.4 cm for the left hand and 8.16 cm for the right hand, compared to the overall average of 6.73 cm left and 6.44 cm right. For the frontal movement, the error was 4.05 cm for the left hand and 7.92 cm for the right hand, compared to the overall average of 4.83 cm left and 5.39 cm right. Figure 5.8 shows the resulting average error for each movement in both perspectives for both hands individually. As shown on the figure 5.8, on average the accuracy dropped during the yoga motion for the left hand by 1.2 cm and 0.4 cm for the right hand. Unlike the yoga instruction, the difference between both hands was not that big when comparing our data for the frontal motion. We measured an average increased distance to the target path of 0.84 cm for the left and 1.06 cm for the right hand. The means of the left and right hand errors for both motions were examined using a t-Test for 2 dependent means. During the yoga instruction the value of t was 2.029959 and the value of p was 0.03363 for the left hand. The result was significant at p <0.05. The right hand error means were not significant at p <0.05 for the yoga instruction, with a p value of 0.15882. For the frontal instruction, both hand error means were significant at p <0.05. For the left hand the value of t was 2.094325. The value of p was 0.03009. The result was significant at p <0.05. For the right hand the value of t was 1.820844. The value of p was 0.04796. The result was significant at p <0.05.

On average, the accuracy between *1PP* and *3PP* decreased by approximately 23.9%. In our Yoga motion the left hand's accuracy decreased on average by 25% and the right hand's by only 9.2%. As expected, the participants' accuracy dropped more during the frontal movement, with an average increase of 23% for the left and 38.4% for the right hand. Users 6 and 9 managed to be the only participants to be consistently better with both hands using the *3PP* for the Yoga instruction. The average error for each individual

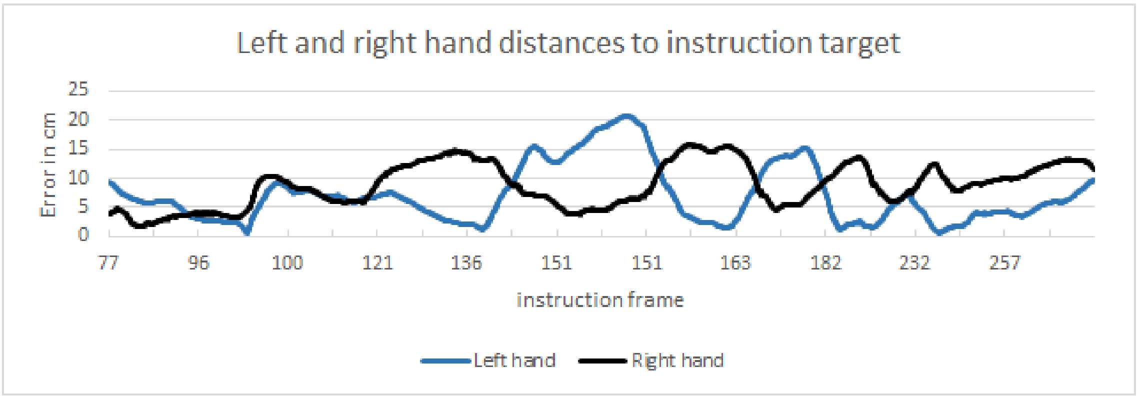


Figure 5.9: Error correlation between left and right hand depending on the side the user is concentrating on in first-person.

participant is shown in figure 5.7. An additional effect of changing the perspective, was a difference in time measured to perform each task. The frontal motion was completed on average in 57.5 seconds when viewed from a *1PP*, with a standard deviation (SD) of 11.3 seconds. The time to accomplish the frontal motion when viewed from a *3PP* increased to 70.5 seconds and a *SD* of 11.2 seconds. The same effect could be measured for the Yoga motion, but favouring the *3PP*. Participants required 74.6 seconds with a *SD* of 13.7 seconds when doing the yoga instruction from a *1PP* and reduced the time down to an average of 66.3 seconds with a *SD* of 8.7 seconds.

The range of angle movement is depicted in figure 5.10, showing the head motion in yaw, pitch and roll angles. The yaw axis is showing the left and right movement of the head, the pitch movement represents the back and forth tilt motion of the head and the roll movement represents the sideways tilt of the head. We show the range of head movement calculated in degrees angle. The yoga motion viewed from a *1PP* required the most left and right motion out of the 4 tasks with a range of 145.8°, followed by the frontal motion, also viewed from the *1PP* with a 106.4° angle. Both motions required significantly less rotational movement when viewed from a *3PP*, with only 30.4° during the yoga instruction and 29.3° during the frontal. Second most rotational movement was necessary for looking up and down, shown by the pitch movement, ranging between 119.4° during the yoga motion viewed from a *1PP* to 30.6° during the yoga motion, viewed in *3PP*. Pitch and yaw head movement did not vary much between all participants. Only roll movement showed a lot of variation between participants for all motions, with the highest variation during our frontal *1PP* motion, ranging between 29.4° up to 64°. Due to the bigger range of necessary head rotation during the *1PP*, the total amount of rotational movement measured in degrees increased significantly between *1PP* and *3PP*. When completing the yoga task in a *1PP* participants had to rotate their heads on average by a total amount 4422.6°. For completing the same task from a *3PP* they only required a total rotational

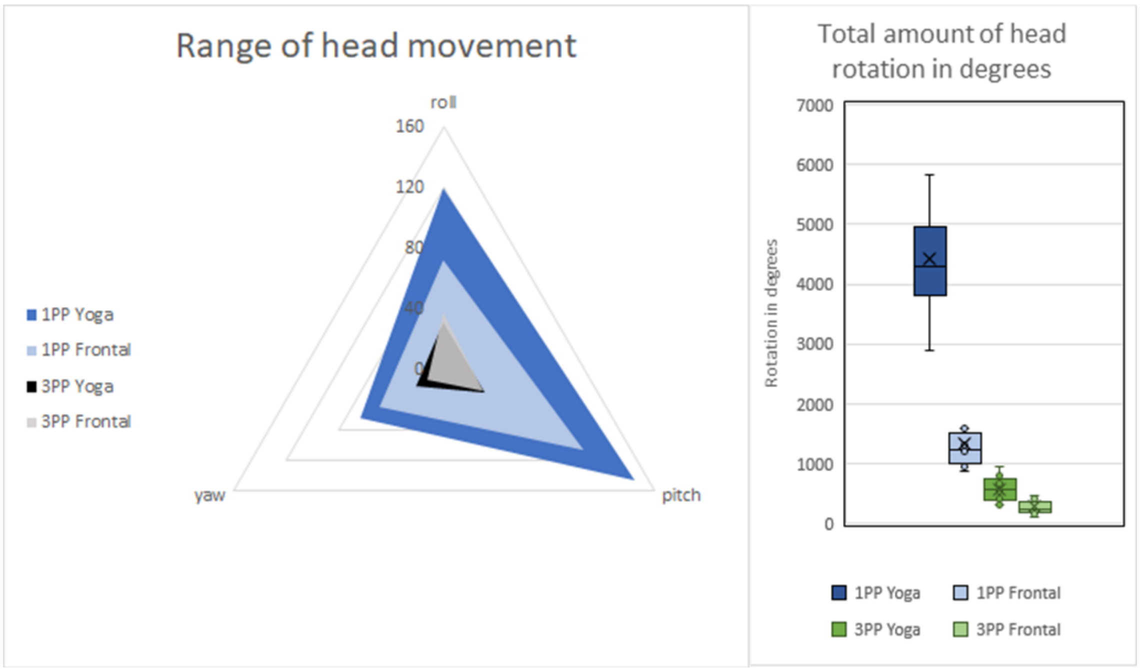


Figure 5.10: Range of the head movement during all 4 tasks in degrees (left). Total amount of head rotation in degrees during each task (right).

movement of 577.4°. Completing the frontal task had a comparable result, but slightly less pronounced. While users required to rotate their heads by 1344.7° from a *1PP* when completing the frontal task, that amount was reduced down to 285.7° when completed from the *3PP*.

Concentrating on one part of the body requires effort by the user and diverts their attention as can be seen in figure 5.9. This figure clearly shows a correlation between the deviation from the target as the user starts to focus on one arm. The graph is a recording of a user's left and right wrists error during the yoga training set within first-person. This effect highly increased in the *1PP*, especially when following the yoga instruction, since the user not only deviated his or her attention from one arm, but focusing on one arm meant, that the other one is not visible within his or her *FoV* in most cases.

5.3.1 Post-Questionnaires

We asked every participant to fill out a post-questionnaire after each condition. Everyone had to fill out two sets of post-questionnaires after each of our conditions. First, they filled out our questionnaire as shown in table 5.2. The answers to these questionnaires support our initial expectations. For both conditions, *1PP* and *3PP*, our test subjects answered similar to the question when they were able to follow the instruction with an average score of 4.30 for the *1PP* and 4.38 for the *3PP* on a Likert scale from 1 to 5, 1 being strongly disagreeing with the statement and 5 strongly agreeing. This indicates that there is no change in confidence per user if they feel able to accomplish a task, confirming our expectations.

Questions two and three confirm our expectations that occlusions make it more difficult for the user when using the *3PP*, if a motion is obstructed by their own body, while *1PP* increases the difficulty of the task when the instruction is not always inside the person's FoV. Questions four and five are geared towards examining the effect on the immersion of the different perspectives for the user and his or her *VBO*. Our data indicates a clear reduction of immersion and *VBO* when switching from a *1PP* to a *3PP*. Many users stated, that they didn't feel like they actually are the seen virtual person, when controlling the avatar from a *3PP*. They mentioned that it felt more like they remote control it, like a puppet. Question six indicates that even though people lost the *VBO* over the virtual avatar, their immersive feel of being in the virtual world did not suffer significantly. Answers in regard to our questions examining the usefulness of our visual assistance, and the ability to properly estimate distances and scale in the virtual environment did not vary between the two perspectives. We also asked our participants to fill out a NASA

Index	**Question**
1	I was able to follow the instruction.
2	Occlusions made it difficult to follow the instruction.
3	It was easy to keep the instruction within my field of view.
4	I felt like my avatar is part of my real body.
5	I believed that I was the character I was controlling.
6	I felt like being in the virtual environment.
7	Seeing my avatar helped in estimation of scale and distance in the virtual environment.
8	I found that the content in the virtual world was helpful in guiding my movement.

Table 5.2: Post-questionnaire give to each participant for both conditions.

task load index (TLX) questionnaire after each perspective. The TLX is a procedure for assessing subjective task workload and a used procedure for over 20 years for human computer interfaces. The TLX for *1PP* and *3PP* scored very similar with a score of 44.62 for *1PP* and a score of 43.28 for *3PP*, being normalized to a scale from 0 to 100. This indicates, that the workload differences between both perspectives is negligible small and that the workload does not play a big role for choosing one of either perspectives.

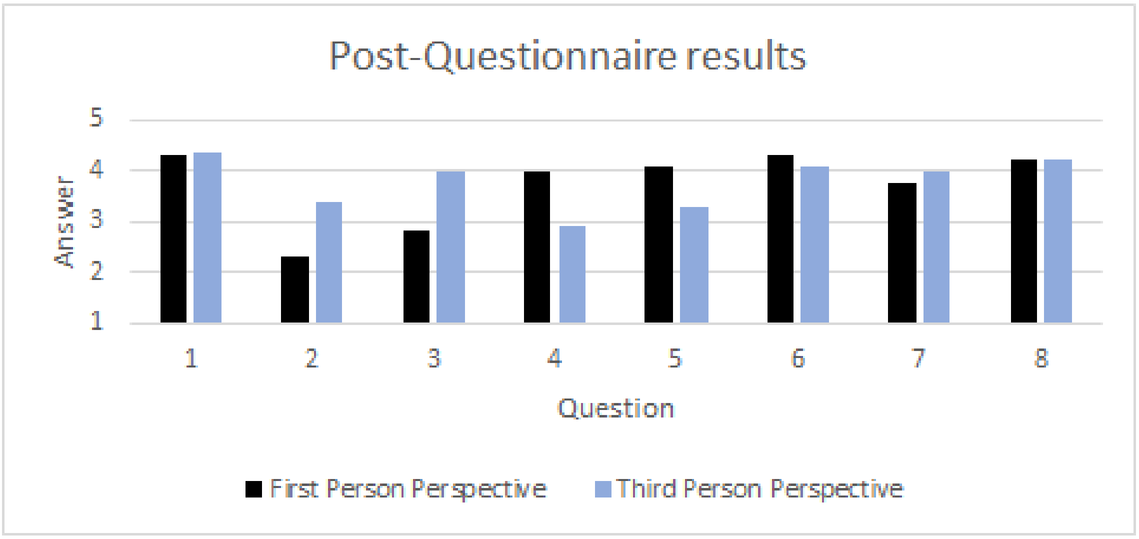


Figure 5.11: Average results of the participants post-questionnaire answers for each question shown in table 5.2 on a Likert-scale ranging from 1 to 5, where 1 is strongly disagree and 5 is strongly agree.

Single Ease Questionnaire

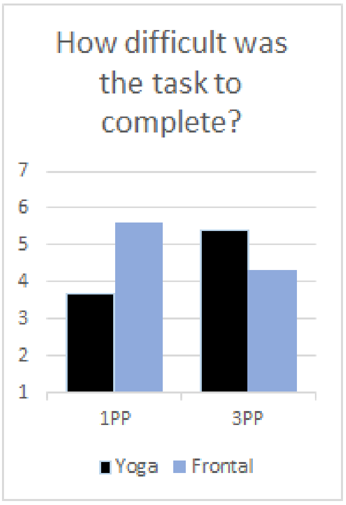


Figure 5.12: Difficulty of each task on average perceived by the users. 1 is very difficult and 7 very easy.

We used a single ease questionnaire for each task to evaluate the perceived difficulty of each individual task. As seen in figure 5.12, the average difficulty for our frontal task in *1PP* felt very easy with an average score on our single ease test of 5.38 on a scale from 1 to 7, where 7 is very easy. This score dropped for the yoga task significantly to 3.69 on average, mostly due to limited *FoV* and the resulting loss of visibility of one out of the two arms, further resulting in a significant increase of required head movement, as shown in figure 5.10. The difficulty score reversed in *3PP*, with an average score of 5.62 for the yoga instruction and 4.31 for the frontal movement, but being more balanced than the *1PP* scores, indicating that the *3PP* is the better choice for a setting, where the perspective cannot be changed to accommodate the situation.

6 Conclusion

In this thesis, we presented a virtual reality (VR) training system, which is able to record full-body movement and replay those recordings. We used the recorded motion capture for different visual guidance systems, for different training experiences, and also for evaluating the users' performance in real-time. With the example of our 3D glyphs, we guided the users' movement and corrected them by providing visual feedback. We assumed, that the first-person perspective (1PP) would require significantly more rotational head movement, due to the limited available Field of View (FoV) and the close proximity to the instruction. We also assumed that the third-person perspective (3PP) gives the user a better scene overview and understanding of their posture, allowing them to follow instructions more fluently with both arms simultaneously.

In our conducted user study, we investigated our initial assumptions regarding the advantages and disadvantages of the two different perspectives, the *1PP* and the *3PP*. We found that the limited *FoV* of the *VR* head-mounted display (HMD) requires significantly more head movement in a *1PP*. For most users in our study, this property outweighed the negative effects of the *3PP*, like the reduction in virtual body ownership (VBO), decreased depth perception and the problem of self-occlusion. Visual aides, like the used occlusion shader for the 3D glyphs and the wristbands, also helped the users to overcome the issue of self-occlusion and reduce its negative impact. Choosing a different perspective confirmed the assumptions that following an instruction, while watching a self-avatar from the *3PP* decreases the overall accuracy of the users. This was due to a more difficult depth perception, which was also stated by multiple participants. In our study, the error for hand placement increased on average by about 1 cm, from roughly 4.97 cm up to about 5.85 cm. The *3PP* also reduced the immersion and *VBO*, as many of our participants stated after the study and which was also confirmed by our post-questionnaire questions four and five. Increasing the available *FoV* for the *HMD* to match the natural human *FoV* of up to 210° could potentially reduce head movement and its negative effect, which was an important factor for most users, leading them to choose the *3PP* as their preferred perspective. During our study we measured an average left and right head motion of 17.5°

in the *3PP*, which increased to 56.4° in the *1PP*. Up and down head motion increased on average, from 29.9° to 129.1° in the *1PP*. Both perspectives showed their strengths and weaknesses for both instructions. When designing a virtual environment, choosing the perspective is strongly influenced by the intentions of the designers and their desire for *VBO*, immersion and small hand manipulations. Overall, most participants in our study preferred the *3PP*, which also seemed to be more balanced for varying tasks.

A

List of Acronyms

1PP	first-person perspective
3PP	third-person perspective
6DoF	six degrees of freedom
AIM	avatar-in-miniature
API	application program interface
AR	augmented reality
CAVE	Cave Automatic Virtual Environment
CNN	convolutional neural networks
CV1	Consumer Version 1
DK1	Developer Kit 1
DK2	Developer Kit 2
DSLR	Digital Single-Lens Reflex
FoV	Field of View
FPS	frames per second
HD	high definition
HDR	high-dynamic-range
HMD	head-mounted display
Hz	Hertz
IK	inverse kinematics
IMU	Inertial Measurement Unit
IMUs	Inertial Measurement Units
LED	Light Emitting Diode
LERP	linear interpolation
MSE	mean squared error
NUI	natural user interface
PCIe	Peripheral Component Interconnect Express
POV	point of view

SD	standard deviation
SDK	Software Development Kit
ToF	Time of Flight
VBO	virtual body ownership
VR	virtual reality

Question	Answer			
What is your age?				
How experienced are you in Virtual Reality? (1…5)				
What is your sex?	Male		Female	
Which is your dominant hand?	Left		Right	

	Yes	No
Do you study computer science or anything comparable?		
Do you have an impaired vision?		
Do you suffer from epilepsy?		

First Person Perspective: Yoga

Overall, this task was?

very difficult very easy

1	2	3	4	5	6	7
○	○	○	○	○	○	○

First Person Perspective: Frontal

Overall, this task was?

very difficult very easy

1	2	3	4	5	6	7
○	○	○	○	○	○	○

Third Person Perspective: Yoga

Overall, this task was?

very difficult very easy

1	2	3	4	5	6	7
○	○	○	○	○	○	○

Third Person Perspective: Frontal

Overall, this task was?

very difficult very easy

1	2	3	4	5	6	7
○	○	○	○	○	○	○

Figure 8.6

NASA Task Load Index

Hart and Staveland's NASA Task Load Index (TLX) method assesses work load on five 7-point scales. Increments of high, medium and low estimates for each point result in 21 gradations on the scales.

Name	Task	Date

Mental Demand — How mentally demanding was the task?

Very Low — Very High

Physical Demand — How physically demanding was the task?

Very Low — Very High

Temporal Demand — How hurried or rushed was the pace of the task?

Very Low — Very High

Performance — How successful were you in accomplishing what you were asked to do?

Perfect — Failure

Effort — How hard did you have to work to accomplish your level of performance?

Very Low — Very High

Frustration — How insecure, discouraged, irritated, stressed, and annoyed wereyou?

Very Low — Very High

Third Person Perspective

1 = Strongly Disagree **2** = Disagree **3** = Neither Agree nor Disagree **4** = Agree **5** = Strongly Agree

	Strongly Disagree	Disagree	Neither	Agree	Strongly Agree
I was able to follow the instruction.					
Occlusions made it difficult to follow the instruction.					
It was easy to keep the instruction within my field of view.					
I felt like my avatar is a part of my real body.					
I believed that I was the character I was controlling.					
I felt like being in the virtual environment.					
Seeing my avatar helped in estimation of scale and distance in the virtual environment.					
I found that the content in the virtual environment was helpful in guiding my movement.					

	1PP	3PP
Which perspective did you prefer?		

Bibliography

[1] Alexiadis, D. S. and Daras, P. (2014). Quaternionic Signal Processing Techniques for Automatic Evaluation of Dance Performances From MoCap Data. *IEEE Transactions on Multimedia*, 16(5):1391–1406. (page 18)

[2] Anderson, F., Grossman, T., Matejka, J., and Fitzmaurice, G. (2013). YouMove: Enhancing Movement Training with an Augmented Reality Mirror. *Proceedings of the 26th annual ACM symposium on User interface software and technology - UIST '13*, pages 311–320. (page 3, 12)

[3] Aristidou, A., Lasenby, J., Chrysanthou, Y., and Shamir, A. (2018). Inverse Kinematics Techniques in Computer Graphics: A Survey. *Computer Graphics Forum*, 37(6):35–58. (page 27)

[4] Banakou, D., Groten, R., and Slater, M. (2013). Illusory ownership of a virtual child body causes overestimation of object sizes and implicit attitude changes. *Proceedings of the National Academy of Sciences*, 110(31):12846–12851. (page 16)

[5] Beal, V. (2018). API - application program interface. `https://www.webopedia.com/TERM/A/API.html`. Visited on October 30th 2018. (page 46)

[6] Botvinick, M. and Cohen, J. (1998). Rubber hand feels touch that eyes see. *Nature*, 391(February):756. (page 13, 15)

[7] Burgh, B. and Johnsen, K. (2017). Effects of Tracking Scale on User Performance in Virtual Reality Games. In *2017 IEEE 3rd Workshop on Everyday Virtual Reality (WEVR)*, pages 1–4. (page 13)

[8] Denisova, A. and Cairns, P. (2015). First Person vs. Third Person Perspective in Digital Games: Do Player Preferences Affect Immersion? In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, pages 145–148. (page 15)

[9] Ehrenberg, S. (2010). Reflections on reflections: mirror use in a university dance training environment. *Theatre, Dance and Performance Training*, 1(2):172–184. (page 30)

[10] Harris, G. (1994). Nintendo introduces video game players to three-dimensional worlds with new virtual reality video game system. `https://www.planetvb.com/modules/advertising/?r17`. Visited on June 24th 2018. (page 9)

[11] He, T., Chen, X., Chen, Z., Li, Y., Liu, S., Hou, J., and He, Y. (2017). Immersive and collaborative Taichi motion learning in various VR environments. *Proceedings - IEEE Virtual Reality*, 2(c):307–308. (page 11)

[12] Henderson, S. J. and Feiner, S. (2009). Evaluating the Benefits of Augmented Reality for Task Localization in Maintenance of an Armored Personnel Carrier Turret. *Science and Technology Proceedings - IEEE 2009 International Symposium on Mixed and Augmented Reality, ISMAR 2009*, pages 135–144. (page 21)

[13] Iguchi, K., Mitsuhara, H., and Shishibori, M. (2016). Evacuation Instruction Training system Using Augmented Reality and s Smartphone-based Head Mounted Display. *3rd International Conference on In Information and Communication Technologies for Disaster Management (ICT-DM)*, pages 1–6. (page 13)

[14] Kalman, R. E. (1960). A New Approach to Linear Filtering and Prediction Problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45. (page 44)

[15] Kim, J., Lee, M., Kim, Y., Eun, S. D., and Yoon, B. C. (2016). Feasibility of an individually tailored virtual reality program for improving upper motor functions and activities of daily living in chronic stroke survivors: A case series. *European Journal of Integrative Medicine*, 8(5):731–737. (page 10, 11)

[16] Kreylos, O. (2016). Lighthouse tracking examined. `http://doc-ok.org/?p=1478`. Visited on June 3rd 2018. (page 24, 26)

[17] Kwon, D. Y. and Gross, M. (2005). Combining Body Sensors and Visual Sensors for Motion Training. *Proceedings of the 2005 ACM SIGCHI International Conference on Advances in computer entertainment technology - ACE '05*, pages 94–101. (page xiii, 10)

[18] Latoschik, M. E., Roth, D., Gall, D., Achenbach, J., Waltemate, T., and Botsch, M. (2017). The Effect of Avatar Realism in Immersive Social Virtual Realities. *Proceedings of the 23rd ACM Symposium on Virtual Reality Software and Technology*, 17:39:1–10. (page 13, 15, 16, 41)

[19] Lee, J. D., Hsieh, C. H., and Lin, T. Y. (2014). A Kinect-based Tai Chi Exercises Evaluation System for Physical Rehabilitation. *Digest of Technical Papers - IEEE International Conference on Consumer Electronics*, pages 177–178. (page 11)

[20] Livingston, M. A., Sebastian, J., Ai, Z., and Decker, J. W. (2012). Performance Measurements for the Microsoft Kinect Skeleton. *2012 IEEE Virtual Reality (VR)*, 298(0704):119–120. (page xiii, 19, 23, 58)

[21] Malventano, A. (2016). SteamVR HTC Vive In-depth - Lighthouse Tracking System Dissected and Explored. `https://www.pcper.com/reviews/General-Tech/SteamVR-HTC-Vive-depth-Lighthouse-Tracking-System-Dissected-and-Explored/SteamV`. Visited on June 4th 2018. (page xiii, 26)

[22] McGreevy, M. (1991). The Virtual Environment Display System. *Proceedings of the 1st Technology 2000 Conference*, 1:3–9. (page 8)

[23] Mehta, D., Sridhar, S., Sotnychenko, O., Rhodin, H., Shafiei, M., Seidel, H.-P., Xu, W., Casas, D., and Theobalt, C. (2017). VNect: Real-time 3D Human Pose Estimation with a Single RGB Camera. volume 36. (page 20)

[24] Morrison, M. M. (1985). Inertial measurement unit (US 4,711,125 A). `https://patents.google.com/patent/US4711125A/en`. Visited on October 29th 2018. (page 26)

[25] Niehorster, D. C., Li, L., and Lappe, M. (2017). The Accuracy and Precision of Position and Orientation Tracking in the HTC Vive Virtual Reality System for Scientific Research. *i-Perception*, 8(3):1–23. (page 24)

[26] Normand, J. M., Giannopoulos, E., Spanlang, B., and Slater, M. (2011). Multisensory Stimulation Can Induce an Illusion of Larger Belly Size in Immersive Virtual Reality. *PLoS ONE*, 6(1). (page 16)

[27] Pausch, R., Burnette, T., Brockway, D., and Weiblen, M. E. (1995). Navigation and Locomotion in Virtual Worlds via Flight into Hand-Held Miniatures. *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques - SIGGRAPH '95*, pages 399–400. (page 32, 57)

[28] Peck, T. C., Seinfeld, S., Aglioti, S. M., and Slater, M. (2013). Putting Yourself in the Skin of a Black Avatar Reduces Implicit Racial Bias. *Consciousness and Cognition*, 22(3):779–787. (page 16)

[29] Peña, J., Hancock, J. T., and Merola, N. A. (2009). The Priming Effects of Avatars in Virtual Settings. *Communication Research*, 36(6):838–856. (page 16)

[30] Salamin, P., Tadi, T., Blanke, O., Vexo, F., and Thalmann, D. (2010). Quantifying Effects of Exposure to the Third and First-Person Perspectives in Virtual-Reality-Based Training. *IEEE Transactions on Learning Technologies*, 3(3):272–276. (page 14)

[31] Salamin, P., Thalmann, D., and Vexo, F. (2006). The Benefits of Third-Person Perspective in Virtual and Augmented Reality? *Proceedings of the ACM symposium on Virtual reality software and technology - VRST '06*, page 27. (page xiii, 14, 21)

[32] Shen, D., Lu, Y., and Dey, S. (2016). Motion Data Alignment For Real-Time Guidance in Avatar Based Physical Therapy Training System. *2015 17th International Conference on E-Health Networking, Application and Services, HealthCom 2015*, pages 238–244. (page 18, 19)

[33] Slater, M., Spanlang, B., Sanchez-Vives, M. V., and Blanke, O. (2010). First Person Experience of Body Transfer in Virtual Reality. *PLoS ONE*, 5(5):1–9. (page xiii, 15, 16)

[34] Sutherland, I. E. (1965). The ultimate display. In *Proceedings of the IFIP Congress*, pages 506–508. (page 7)

[35] Tang, R., Bateman, X.-d. Y. S., Jorge, J., and Tang, A. (2015). Physio @ Home : Exploring Visual Guidance and Feedback Techniques for Physiotherapy Exercises. *Chi 2015*, pages 4123–4132. (page xiii, 3, 12, 32, 56, 59)

[36] Tao, G., Archambault, P. S., and Levin, M. F. (2013). Evaluation of Kinect Skeletal Tracking in a Virtual Reality Rehabilitation System for Upper Limb Hemiparesis. *2013 International Conference on Virtual Rehabilitation, ICVR 2013*, pages 164–165. (page 18)

[37] Totilo, S. (2012). Carmack being Carmack: A Dozen Minutes With One of Video Game's Smartest People. `https://kotaku.com/5916210/`. Visited on June 24th 2018. (page 33)

[38] Waltemate, T., Gall, D., Roth, D., Botsch, M., and Latoschik, M. E. (2018). The Impact of Avatar Personalization and Immersion on Virtual Body Ownership, Presence, and Emotional Response. *IEEE Transactions on Visualization and Computer Graphics*, 24(4):1643–1652. (page 15, 16)

[39] Wang, Q., Kurillo, G., Ofli, F., and Bajcsy, R. (2015). Evaluation of Pose Tracking Accuracy in the First and Second Generations of Microsoft Kinect. *Proceedings - 2015 IEEE International Conference on Healthcare Informatics, ICHI 2015*, pages 380–389. (page xiii, 17, 18, 23, 24, 58)

[40] Wasenmüller, O. and Stricker, D. (2017). Comparison of Kinect v1 and v2 Depth Images in Terms of Accuracy and Precision. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 10117 LNCS:34–45. (page 17)

[41] Webster, D. and Celik, O. (2014). Experimental evaluation of Microsoft Kinect's accuracy and capture rate for stroke rehabilitation applications. *IEEE Haptics Symposium, HAPTICS*, pages 455–460. (page 18)

[42] Wheatstone, C. (1838). Contributions to the Physiology of Vision.–Part the First. On Some Remarkable, and Hitherto Unobserved, Phenomena of Binocular Vision. *Philosophical Transactions of the Royal Society of London*, 128(1):371–394. (page xiii, 6)

[43] Yee, N. and Bailenson, J. (2007). The Proteus Effect: The Effect of Transformed Self-Representation on Behavior. *Human Communication Research*, 33(3):271–290. (page 16)

[44] Zuckerberg, M. (2014). Announcement by Mark Zuckerberg to aquire Oculus VR for 2 billion dollars. `https://www.facebook.com/zuck/posts/10101319050523971`. Visited on June 24th 2018. (page 33)