E(x|g)o: Comparing Visual Perspectives on Guidance Visualisations for Motor Learning in Virtual Reality

Stefan Paul Feyer
HCI Group, University of Konstanz, 01/725883

Maximilian Dürr
Supervisor

Prof. Dr. Harald Reiterer
Reviewer

Master's Project Report
1.10.2020

Abstract

Motor learning is an important aspect to master physical movements and usually includes a teacher. If a teacher is not available or affordable, Virtual Reality has proven to be a suitable replacement for a real teacher. In the real world, the visual perspective on the teacher is natively exo-centric. In Virtual Reality, the perspective on the guidance visualisation can be ego-centric, too. However, the influence of the visual perspective on virtual guidance visualisations is sparsely researched. This report describes the design and implementation of E(x|g)o, a system capable of conducting a study to investigate the influence of different visual perspectives on guidance visualisations. E(x|g)o uses Virtual Reality and full-body motion tracking for motor learning from multiple visual perspectives.

Contents

1 Introduction

1.1 Introduction

Motor learning is necessary for various activities, like the ergonomic conduction of working routines, sports, arts, or dancing. Training such movements usually include a teacher that performs the movements, the learner (further called **student**) can mimic. However, a teacher is not always accessible because of the lack of availability, economic reasons, or time. In this case, technical solutions exist to step into this role. For example, YouTube[1] and other video platform turned into a great source of learning videos.
Research proved that Mixed Reality (MR) could also be a valuable technology to teach students. MR systems can provide teachers no only in two dimensions (2D) but in three dimensions (3D), which provides a better grasping of the movement in question. Also, MR systems can be interactive and provide feedback on the performance of the student. On this assumptions, researcher developed training systems for arts [4, 7, 12], dance [2, 6, 23], sports [5, 11] and rehabilitation [3, 16, 20], in the ego-centric [10, 17, 24], exo-centric [8, 13, 21] or both [9, 18, 19] visual perspectives. These systems showed that the choice of the visual perspective on a guidance visualisation (further also called **teacher**) could potentially influence motor learning. However, how the visual perspectives on virtual guidance visualisation influences motor learning is widely left out in these works. The fact that the visual perspective on virtual guidance visualisations potentially influences motor learning in combination with the limited knowledge of how the visual perspective influences motor learning shows the necessity of investigations.
On this basis, this work aims to answer the following research question:

> Does the visual perspective on a virtual guidance visualisation have an influence on motor learning in MR environments?

The preceding seminar thesis investigated on several aspects of motor learning and revealed requirements a motor learning system must fulfil to train motions

[1] youtube.com

effectively. In the following, a brief overview of scope and requirements is provided. For a more detailed elaboration, see seminar thesis. The scope was set to:

- Serial movements: a sequence of discrete movements, best suited for a study and widely used by other researchers [1, 2, 9].
- Cognitive stage: highest increase of proficiency, therefore best suited for a study.
- Closed skills: investigation in a controlled environment.
- Single error measures: proven suitable by other researchers to generate data to evaluate motor learning [12, 18, 20].
- Virtual reality (VR): AR provides a limited field of view, VR more suitable for the study in question.

For this scope, the requirements are as follows.

R1 Provide at least four different perspectives: ego-centric, exo-centric, ego & exo-centric and augmented exo-centric seem most promising to deliver insights to answer the research questions. Further elaboration in next section *Perspectives*.

R2 Provide different viewpoints on the guidance visualisations: mirror effect and occlusion of the guidance visualisation makes it necessary to provide different viewpoints on the guidance visualisation [4].

R3 Avatar guidance visualisation: a person-shaped avatar will be used as virtual guidance visualisation like used in [4, 8, 12].

R4 High realism degree, person-shaped avatar. Weber [22] showed that high realism guidance visualisations are more suited for virtual guidance visualisations.

R5 No additional feedback: evaluation is focused on the perspective; feedback will not be part of the evaluation.

R6 Scaling of the teachers height to the student's height: in case of teacher an students have different body sizes, especially in the ego-centric perspective, scaling of the teacher is mandatory [9].

R7 Provide measures: the system must be capable of producing reliable data for the evaluation.

R8 Provide full-body motion tracking: real-time tracking of the student.

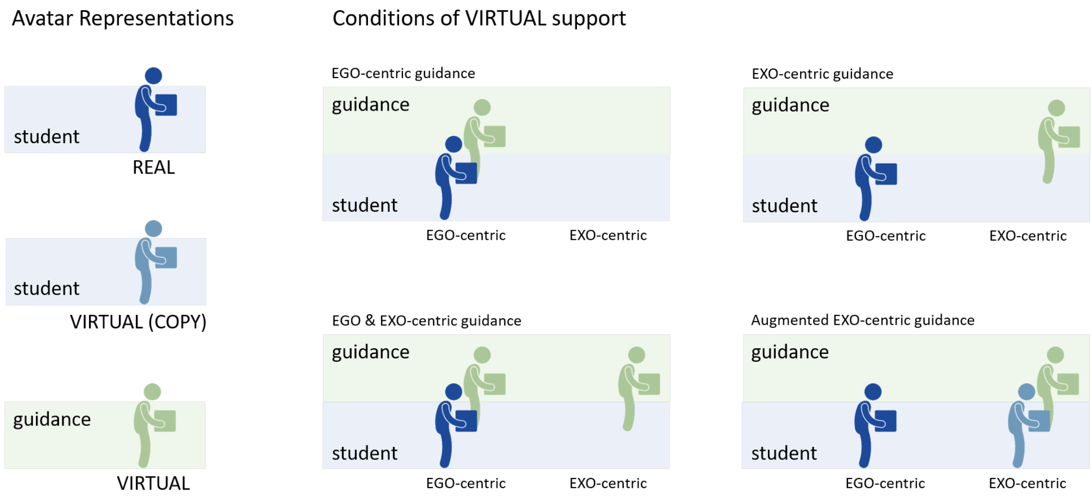

Figure 1.1: The four perspectives the system is capable of. Human figure taken from thenounproject.com, accessed: 19.06.2020. Icon: created by Ghan Khoon Lay from Noun Project.

1.2 Perspectives

In a scenario with one teacher and one student, five different visual perspective are possible: ego-centric, exo-centric, ego & exo-centric, augmented exo-centric and ego & augmented exo-centric, compare 1.1. To reduce complexity, the focus is set to the fist four perspectives. These four visual perspectives seem most promising to deliver insights to answer the research question.
In the **ego-centric** visual perspective the teacher stands inside the student, in the **exo-centric** visual perspective outside the student. In the **ego & exo centric** perspective, the teacher stands as well as inside and outside the student. In the **augmented exo-centric** visual perceptive the teacher stands inside a virtual copy of the student.

1.3 Task

The task will relate to how to handle physical load for the following reasons: it might help to address critical health issues by allowing the guidance of the correct ergonomic conduct. Furthermore, there is view research on guidance on how to handle physical load.
Physical load hereby could be represented by a box. The virtual guidance visuali-

sation shows a sequence of movements like lifting, turning, pushing the box on or onto a table. Examples can be found in the following video segment: click.

1.4 Structure

This masters project aims to implement a VR motor learning system that can be used in a study to evaluate the above-mentioned research question. This document describes the implementation of E(x|g)o, which claims to be such a system. E(x|g)o fulfils the requirements developed in the preceding seminar thesis. The development starts with an analysis of hard and software in chapter 2, which serves as a fundament for the implementation described in chapter 3. Chapter 3 points out the limitations of E(x|g)o, too. In chapter 4, an outlook provides insight into the next steps.

2 Technology Evaluation

The hardware consists of three main parts. First, an Virtual Reality (VR) Head Mounted Display (HMD) which serves as the window to the virtual world for the study participants. Secondly, a motion capturing system, which serves as a translation of the real world movements in the virtual world. Thirdly, the assets with which the participant of the study will interact. This chapter discusses the requirements these three aspects have to fulfil to have a reliable and suitable study design.
In the early stages of planning a study, it is vital to choose the hardware wisely. The hardware is the base on which the software is built, and both need to fulfil requirements to be usable in a study. For example, the accuracy must be high enough to measure errors in the motion of the student and also it must be as reliable as possible, to not disrupt the study procedure. For this reason, hard and software were analysed. Finally, the study setting overview depicts the interplay of these three components.

2.1 Hardware

Head Mounted Display

In 2013, the first next-generation HMDs was on the market for sale. The Oculus Rift DK1 had given the first glimpse on what will be possible with next-generation Virtual Reality headsets. The next big step was the evolution of graphics cards, and with the GTX 1000 series released in May 2016, enough graphics power was widely available to run state of the art headsets. Since then, more and more headsets of multiple companies made it to market maturity, and today we are spoilt of choice. Hence, a comparison of VR HMD's was necessary. Four VR HMD's were part of this evaluation: Oculus Quest, Pimax 8K, HTX Vive Pro and Valve index. The aspects these four headsets were evaluated are: the screen resolution they provide, the field of view they have, the tracking method they use, if they are bound to a cable and how pleasant they are to use with glasses beneath the headset.

Headset	Cable bound	Resolution	Field of View	Tracking method	Usable with Glasses?
Oculus Quest	No	2880 x 1600, 72 Hz	110°	Inside-Out, 4 Cameras	comfortable
Pimax 8k	Yes	7680 × 2160, 80 Hz	200°	SteamVR Tracking 2.0	?
HTC Vive Pro	Both possible	2880 x 1600, 90Hz	110°	SteamVR Tracking 2.0	uncomfortable
Valve Index	Yes	2880 x 1600, 144Hz	130°	SteamVR Tracking 2.0	comfortable

Table 2.1: Comparison of VR HMDs. Sources: https:// developer.oculus.com/ design/oculus-device-specs/, https:// www.pimax.com/ pages/pimax-8k-series, https:// www.vive.com/ eu/product/vive-pro/, https:// www.valve software.com/ de/index/headset, all accessed: 19.06.2020

All headsets are suitable to teach motor learning in virtual reality. They have low latency, are exact, have plenty of pixels with reasonable refresh rates and a wide field of view. However, they differ in the tracking technology utilised to identify their position and orientation in space, as well as other aspects like the information transfer to a PC or the possibility of wearing glasses beneath it. The possibility to wear glasses beneath the HMD is important because no one is excluded from participating in the study for the reason of wearing glasses. For this project, the cable does not play a big role. Since all headsets are suitable to conduct a study on motor learning, the decision was mostly made from the possibility to wear glasses and have the same coordinate system as the motion tracking technology, which guarantees easier and a more stable environment. The Valve Index headset, compare figure 2.1 currently has with the best performance and fulfils the additional demands. Though the choice here was easy[1]. A comparison of VR HMDs can be found in table 2.1.

Lighthouse

The so-called *Lighthouse 2*, compare figure 2.2, by Valve, also called *Base Stations* are rectangular boxes in size of 7.5 x 7.5 x 6 cm. They emit a fast-moving laser beam. With two base stations, the HMD, Vive Trackers and Controller can determine the exact position and orientation in 3D space in real-time. The Lighthouse enables E(x|g)o to track all actors. The setting includes 4 Base Stations in each

[1]Meanwhile, the Pimax 8K is on sale. From a technical view, this is a headset outperforming the Valve Index. A test of this hardware is planned, and if it proves to be suitable, the study will be conducted with the Pimax 8K

Figure 2.1: Valve Index

Figure 2.2: Light House 2.0

corner of the tracked volume. Two Base Stations would be enough to track the actor in the volume, but since the system is optical, it is prone to occlusion. To overcome this issue, 4 Base Stations are used.

Motion Tracking

In contrast to the VR HMD, the choice for motion tracking hardware is more difficult.

The requirements for the motion capturing system are as follows:

- **Accuracy**: to compare movements, the accuracy should be at least under 1cm
- **Large movement area**: motor learning includes movement in space. The tracked area should be at least 20qm
- **Capable for humanoid movements**: tracking of objects and movement differ fundamentally. For motor learning, humanoid movements are necessary.
- **Freedom of movements**: during motor learning, a human can move in all directions and all possible postures conceivable. The motion tracking system must cover this.
- **Sufficient tracking points**: for measuring movements, enough tracking points must be provided for comparing the movements.
- **Extendable**: assets must be able to be tracked as well.
- **Reliable, study safe setup**: jitter or calibration loss can void a study.
- **No coordinate system matching**: matching of the different coordinate system, e.g. HMD and the human body is an error source which should be excluded.

There are two main classes of motion tracking systems: inside-out and outside in. Both of them have pros and cons. Inside-out tracking uses active sensors on the body to track. Perception Neuron[2], Rokoko, [3] uses the magnetic field of the earth, accelerometer or a gyroscope to identify the position and orientation changes. However, e.g. magnetic sensors are prone to metal, and there is still the

[2]https://neuronmocap.com/
[3]https://www.rokoko.com/ or https://www.xsens.com/

Rq/sys	Accuracy	Movement area size adequate?	Humanoid movements supported?	Enough freedom of movements	Expandable with additional objects?	Reliability	Coordinate system matching required?
Inside out technologies	drift	Potentially unlimited	✓	Even "hidden" markers	No	mid range	Yes
Kinect	acceptable	6m²	✓	Only front view	No	migh	Yes
OptiTrack	Yes but prone to noise	Setup specific	✓	All visible markers	native	Hard to achieve	No, with workaround
Vive Tracker	Highest	Ca. 100m²	✓	All visible markers	native	mid- high	native

Table 2.2: Comparison of evaluated Motion Tracking technologies.

drift problem: a position is determined by the previous position, and the error adds on the error before. The longer the capturing, the greater the error. On the plus side, the tracked area is potentially infinite, and the sensors are rather small. On the other hand, outside-in tracking systems use external tracking devices to track points on the body in question. These systems deliver an absolute position and orientation in the tracking space and do not have the drift issue. If accuracy is the measurement in question, inside-out tracking systems are always second to outside-in tracking systems. For this reason, in the following only outside-in tracking systems are discussed. An overview of the evaluated motion tracking technologies is depicted in table 2.2.

Kinect

Microsoft Kinect 2.0 is the successor of the 1.0 version released in 2014. It uses an RGB camera and an IR camera to calculate a skeleton of a human body in front of it. It is effortless to set up and use. Also, the tracking area is large enough. However, the participant needs to face the Kinect at all time to have a reliable, jitter-free tracking. For this reason, the accuracy and reliability had to be rated low, as can be seen in table 2.2. Therefore, the evaluation of the Kinect 2.0 revealed that it is not suitable for motor learning, compare test video. The new version *Azure Kinect*[4] was not available for evaluation.

OptiTrack

2.4 OptiTrack is an optical motion tracking system. Multiple infrared cameras track reflective markers on the human body, calculating a skeleton. This skeleton is sent to the game engine in use where it can be used to show an avatar of the

[4]https://azure.microsoft.com/en-us/services/kinect-dk/

tracked person. The movement area is big, depending on the camera setup and has high accuracy as long as there is no noise. Additionally, with an external tool called OpenVR[5] the HMD can be natively tracked, too and thus no coordinate system matching is necessary. On the opposite side, there is the setup itself, compare figure 2.4 left. The cameras are connected by USB to the PC, there the OptiTrack driver calculates the position in the room. The OptiTrack driver loops this calculated data back on localhost where the OpenVR driver takes it, makes his calculations to include the HMD then and loops it again back on localhost. Here the SteamVR driver takes the data up and sends it to the game engine in use. Unfortunately, this process is time-consuming and leads to a mentionable and rather reasonable delay in the visual representation. This delay can lead to cave sickness of the participant. Another issue is the calibration process, compare figure 2.4 right. The marker design must be very elaborated and is the base for good tracking. OptiTrack must be calibrated with the Lighthouse of, the rigid body must be calibrated, then the Lighthouse must be switched on, and the pivot point of the HMD marker set mus to be calculated. In the end, the streaming of the data must be specified, and the Lighthouse must be switched off again. First, every step is a possible source of error. For a study considering accuracy, these are too many points of uncertainty. Secondly, this process is very time consuming because every step except the marker design must be conducted before every participant in the study. Eventually, the marker sets of the body and marker sets of the HMD are woven into each other, compare figure 2.3, which is hard to distinguish for OptiTrack and though another source of error. For these reasons, OptiTrack is hardly usable for the task in question.

Vive Tracker

Vive Tracker are devices that utilise the Lighthouse which is also used to track the HMD, see next chapter for further details. This allows tracking points in the room. If these trackers are placed on a human body, the position and orientation of the body part can be digitalised and utilised in an engine. With Inverse Kinematics (see next chapter), these tracking points can be formed to a human body.

Vive Trackers are very accurate, easy to set up and the most important point is that the Lighthouse natively supports them. Because of this, no coordinate system matching is necessary. Furthermore, they are reliable and not prone to noise. The tracking volume can be up to 110 m^2 depending on the number of base stations. Besides, they have the same low latency as the HMD, and the calibration process must only be conducted once and not before every participant. On the other hand, much more must be done by hand, see chapter *Implementation*. Furthermore, the

[5]https://v22.wiki.optitrack.com/index.php?title=OptiTrack_OpenVR_Driver

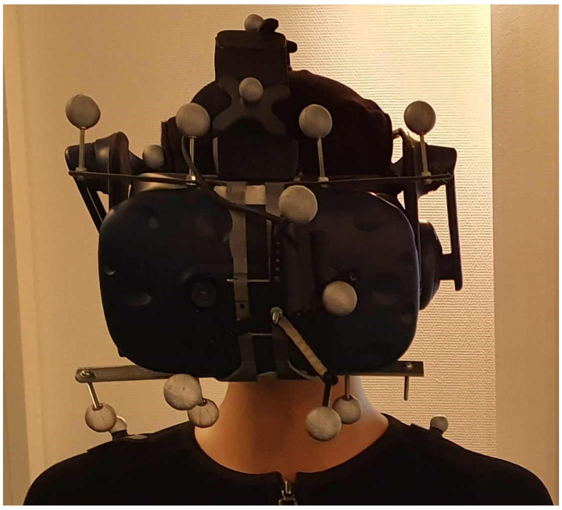

Figure 2.3: The marker set of the body (top and side) and the marker set of the HMD (at 3D printed mount) are too close to each other to be distinguished by OptiTrack.

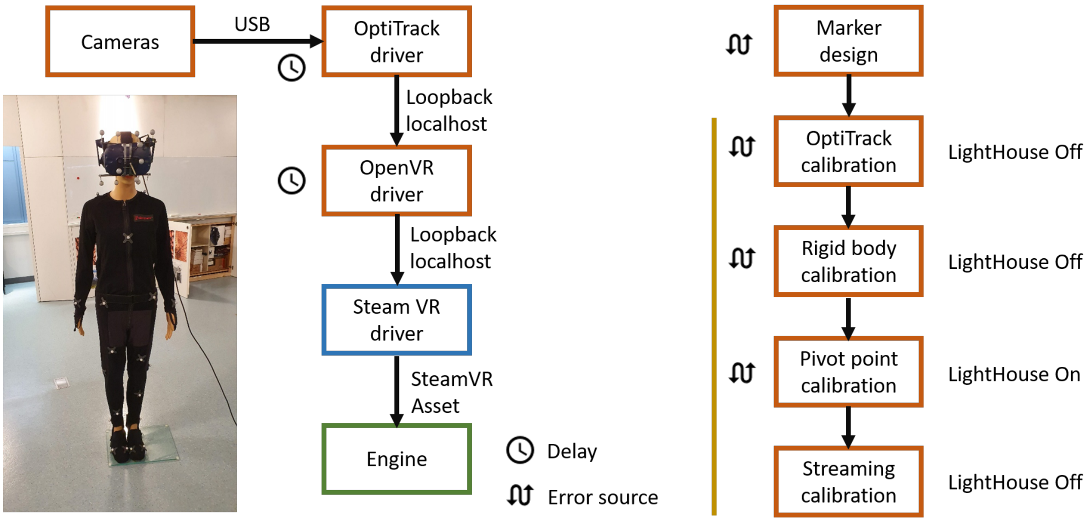

Figure 2.4: OptiTrack data flow and calibration process.

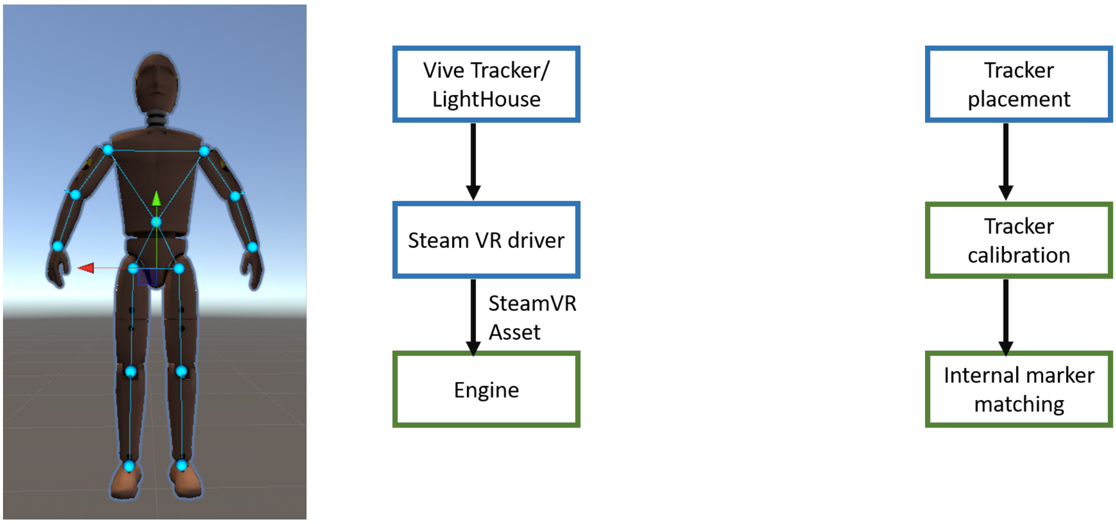

Figure 2.5: Vive Tracker data flow and calibration process.

Figure 2.6: Left: Vive Tracker. Right: 3D printed Vive Tracker Mounts with Velcro.

Vive Tracker are rather big and are attached to the body. The weight and size could influence the movements of the participants. Nevertheless, for the sake of the study and the above reasons, the Vive Trackers are used for this project.

Vive Tracker 2

Vive Tracker 2, compare figure 2.6 left, are star shaped elements, that utilise the Lighthouse to determine its position and orientation in the tracking volume. The Vive Tracker 2 transmits its data via Bluetooth. For each Vive Tracker a special dongle mus be plugged in the PC on a USB port. To mount the Vive Tracker on a human body comfortably, 3D printed mounts are used, compare figure 2.6 right. On the back side Velcro tape is applied. Velcro holds perfectly on an OptiTrack

Figure 2.7: Left: evaluation of different box shapes and sizes. Right: Box and table with Vive Tracker. Both shoes with Vive Tracker in the middle.

suit. In future, special Vive Tracker straps can be used to make wearing the tracker more comfortable. The mounts are fixated by a M6 screw to the Vive Tracker.

PC

The PC must be capable of handling the VR HMD. The main bottleneck is the graphics card. The used PCs graphic card is a GTX 2080. The PC must be connected to the HMD. Furthermore, the PC must provide enough USB Ports for HMD, periphery devices, one per Vive Tracker, but better not more than two per USB controller, compare next section.

Box & Table

To conduct a study that includes physical load, a physical load prop must be designed. For this, several shapes and sizes of boxes were collected and placed on a table, compare **??** left. After interacting with all these boxes, the decision was made for a box with the dimensions 27 x 21 x 21 cm. For developing a cardboard box was used. Also, to make scenarios possible where the participant lifts the box up and down, a table is part of the setup. Both are tracked by a Vive Tracker, compare figure **??**.

2.2 Software

The choice of software is strongly bound to the hardware choice and less complex. The two main engines on the market to use are Unreal[6] and Unity[7]. Both are

[6] https://www.unrealengine.com/en-US/
[7] https://unity.com/

suitable but differ in the assets and plug-ins available. Unity has a well developed native market with independent developers, which already produced valuable plug-ins this project can benefit from. Finally, in my opinion, Unity is much more intuitive to use than Unreal, though the Unity Engine will be used.
Because the choice was made for the Valve Index HMD, SteamVR is mandatory to use.
For the Inverse Kinematics tool, the choice was made for Final IK[8]. The functions are promising and cover more scenarios like other tools like Fast IK[9] or Dynamic Bone[10]. Final IK supports person-shaped avatars natively in VR, which was the pivotal argument.

Inverse Kinematics & FinalIK

Inverse Kinematics emerged from the field of robotics. Imagine a robot arm tip needs to be at an exact point in a 3D space. Inverse Kinematics is the process to calculate the angles of the joints of the arm to match the endpoint of the arm with the desired point in space. Every joint has a minimum and maximum value and degrees of freedom. This process can also be utilised to visualise a human body. The end effectors are here the Vive Tracker located on a real human body. Inverse Kinematics calculate the angles of joints in between to render a person realistically. FinalIK is a Unity package providing this process.
VRIK is a part of FinalIK with the focus of rendering human bodies for virtual reality. VRIK utilises 5 Vive Tracker: 2 on the feet, 1 one the hip and 2 on the hands. The HMD is the last reference point. Based on these 6 points, it solves the limbs of a human body and renders a humanoid character.

SteamVR Principles

SteamVR is a driver handling the connected Tracker, HMD and Controller. After a calibration, up to 16 devices can be added to one instance. In this case, these devices are 4x Lighthouse, 1x Valve Index HMD, 7x Vive Tracker and 1x controller. The driver provides the position and orientation to programs utilising this information. In this case, the Unity asset SteamVR access this information.

[8] https://assetstore.unity.com/packages/tools/animation/final-ik-14290
[9] https://assetstore.unity.com/packages/tools/animation/fast-ik-139972
[10] https://assetstore.unity.com/packages/tools/animation/dynamic-bone-16743

2.3 Study Setting Overview

In this chapter, the hardware and software components were discussed. This section shows the interplay of all these components to create the study setup, compare 2.8. The hardware components are:

- 4x Lighthouse 2: tracking of Valve index and Vive Tracker.
- VR HMD: Valve Index: window for the student to the virtual world.
- 7x Vive Tracker 2: tracking devices placed on the student for full-body motion tracking, on the table to locate the table, on the box to determine the position and orientation in the tracking volume.
- 7x Vive Tracker mounts: for attaching the Vive Tracker on the prop or student
- Velcro OptiTrack suit: easier attachment of Vive Tracker mounts on the student.
- PC (SteamVR, Unity, FinalIK): software infrastructure, brings together the tracking data, HMD data and produces output on the HMD
- box & Table: during the task, the student will move the box onto, off and on the table.

The four Lighthouses define the tracking volume, where the study will take place. The Lighthouses track the markers on the student, the box and the table as well the Valve Index. This data is transferred to the PC were Steam VR handles the tracking data and serves it to Unity. Unity generates the output for the Valve Index.

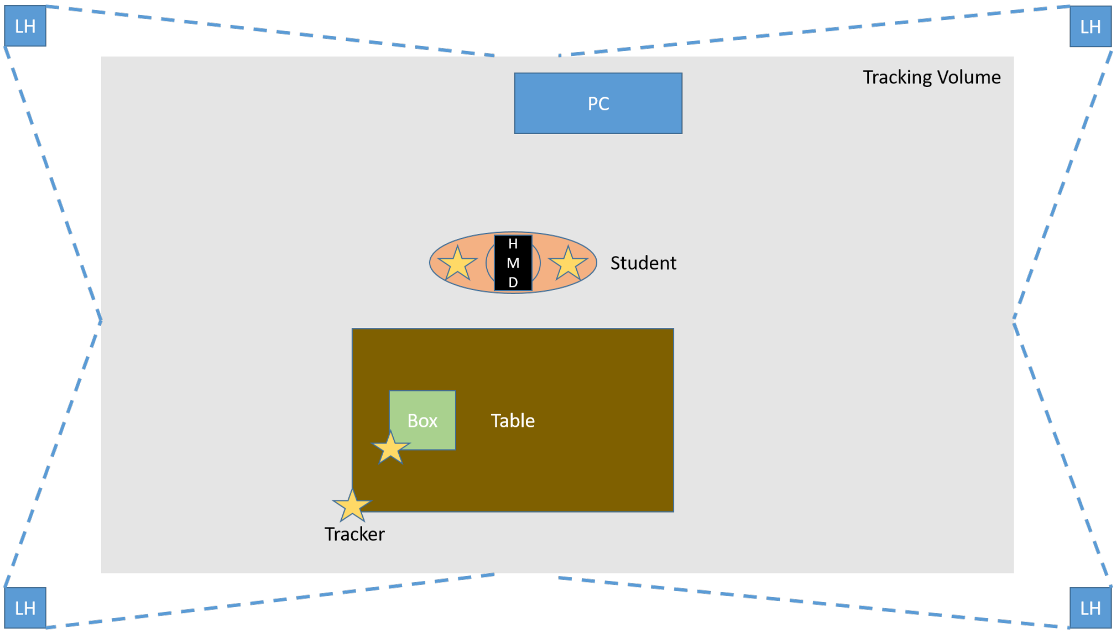

Figure 2.8: Study setup; shows the interplay of hard and software components.

3 Implementation

The preceding chapter describes the software and hardware, which serves as the fundament the implementation builds on. To create a system to teach motor learning from multiple perspectives, several steps have to be made: the Vive Tracker matching, VRIK calibration, the rendering of the student, teacher, table and box. With these steps completed, the movements of a teacher can be recorded. The recorded teacher then needs to be resized to the student's height. With all this set up, the visual perspectives themself need to be developed.
This chapter describes in detail the implementation of these steps. The last section shows a comparison of the perspectives with their mechanics utilised to achieve the visual perspective in question, as well as their limitations.

Vive Tracker Matching

In Unity, a Vive Trackers position and orientation can be assigned to a GameObject with the script *SteamVRTrackedObject* compare figure 3.1 middle. In this script, a device ID is used to be associated with a physical Vive Tracker. Unfortunately, every time SteamVR driver is started, the device IDs change. For example, device ID 1 was associated with the Vive Tracker placed on the left feet of the user. On the next start-up, device ID 1 can be associated with Lighthouse base station. This leads to no study-safe setup and must be corrected. To solve this issue, first, all physical Vive Tracker needs a label. The label is a sticker with the numbers B1-B12. After that, device hardware IDs of the used devices must be read out. The device IDs per tracker label in use are:

B1: LHR-67E402D1 Hip

B3: LHR-32C38603 RFoot

B5: LHR-4E4C94A4 LFoot

B6: LHR-B925C963 Table

B8: LHR-89131158 BOX

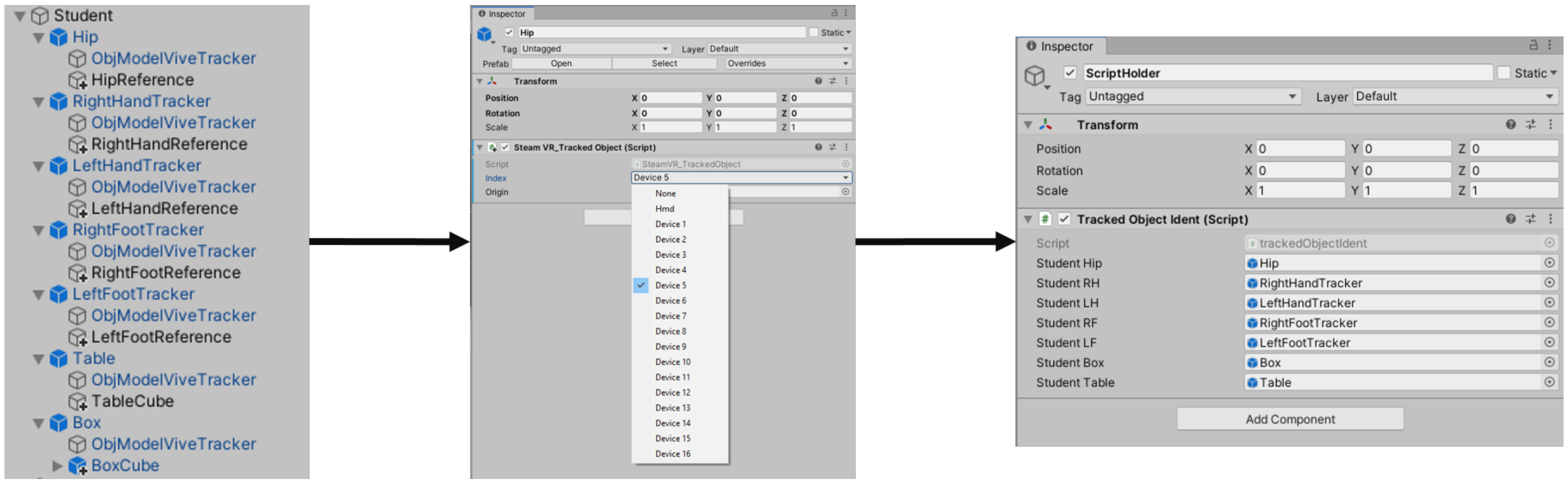

Figure 3.1: Vive Tracker matching. Left: reference holding all Objects to be transformed by a tracker, middle: device ID list, right: script which reads the hardware IDs and sets the device IDs.

B9: LHR-31D0CDF2 LHand

B10: LHR-CAC69A3C RHand

Secondly, the hardware id must be matched to specific devices. Then, at the start-up of the system, an algorithm reads out all hardware IDs and assign it to the correct GameObject by associating the correct device ID. Since this is a common issue across many projects, this code illustrates the solution:

```
for(uint i = 0; i < 16; i++){
 ETrackedPropertyError error = new ETrackedPropertyError();
 StringBuilder sb = new StringBuilder();

 OpenVR.System.GetStringTrackedDeviceProperty(i,
  ETrackedDeviceProperty.Prop_SerialNumber_String,
  sb, OpenVR.k_unMaxPropertyStringSize, ref error);

 var serial = sb.ToString();

 switch (serial)
 {
  case "LHR-67E402D1":
   // "Found device with ID LHR-67E402D1 (Hip).
   // Assinging Hip with device index: " + i
   studentHip.GetComponent<SteamVR_TrackedObject>()
    .SetDeviceIndex((int)i);
  break;
  case "LHR-32C38603":
   // "Found device with ID LHR-32C38603 (RFoot).
   // Assinging RFoot with device index: " + i
   studentRF.GetComponent<SteamVR_TrackedObject>()
    .SetDeviceIndex((int)i);
  break;
...

}
```

VRIK Calibration

VRIK provides natively to track a person in Virtual Reality. However, the setup is rather complex and needs several calibrations for the head, hands and feet. The

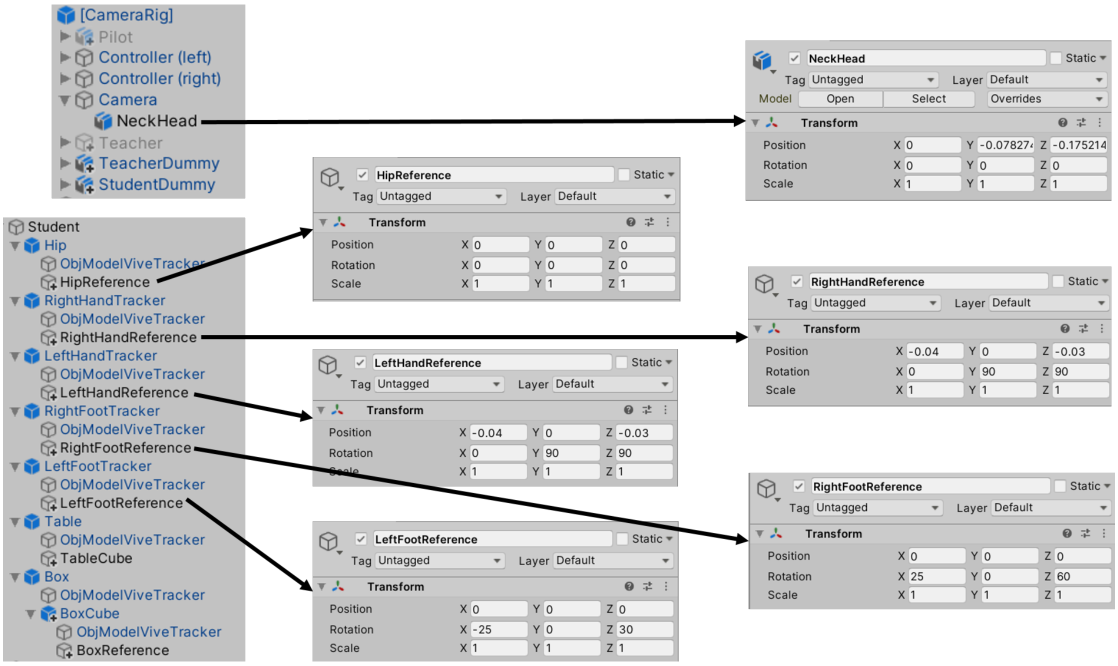

Figure 3.2: All elements and their relations to calibrate one student. Other projects with a similar setup can reuse the values of shifting and rotating the markers

VRIK Calibration Controller brings all calibrated parts together. The following sections describe the calibration steps needed to generate natural-looking representations of the student and teacher. This process is extensively time-consuming because all values need to be determined by experimentation. The values for calibration can be reused in similar setups for other projects.

Head

VRIK solves the limbs based on the tracking points mentioned above. The head reference is a child of the *camera* provided by the SteamVR asset. It holds position and rotation and is the essential input for the solver. The VRIK solver takes this position as absolute and solves everything else based on this position. The position but not rotation overrides even the parent transforms. This fact influences the further handling in Unity massively: to transform a VRIK, the GameObject can not be translated into a specific position. This translation can only happen by transforming the position of the references of the Vive Tracker itself. Because of this, a student avatar never holds the original head transform of the head. The avatar holds a GameObject that copies the transform of the head transform.

Hands

Left and right hand position need to be adjusted because their root lies inside the body in the middle of the joint. To transform the References accordingly, they are a child to the tracker reference. From this root, the references are shifted 4 cm towards X and 3 cm towards Y. Additionally, the rotation needs to be adjusted by 90° in Y and Z. With this, the tracking point is located in the middle on the back of the hand. The values used for hand calibration are depicted in figure 3.2

Feet

Left and right feet need adjustments in terms of position and rotation, too. The Vive Tracker is located at the top of the feet, which is fine with the root of the VRIK reference. However, the slope of the trackers and forward direction must be adjusted. The left reference needs a tilt of -25°, the right reference by +25°. For the correct direction, the left reference needs a rotation by 30° on Z-axis, the right by 60°.
The hip does not need further shifts. The values used for feet calibration are depicted in figure 3.2

VRIK Callibration Controler

With the correct position of the references, the references can now be assigned to a VRIK instance. This works with a VRIK calibration controller shown in figure 3.3. The calibration controller takes the references and assigns it to the VRIK instance. In the same step, the humanoid character is sized to the references.

Student Rendering

The rendering of the student's humanoid character is based on the live tracking data of the Vive Trackers. As seen above, the Trackers get associated to a GameObject, e.g. hip. Relative to this GameObject, a reference is created. This reference is passed to the calibration controller. The calibration controller sets the corresponding references in the VRIK solver. So far nothing new.
Now the VRIK takes a humanoid character and binds its muscles. Then the solving begins, and the algorithm animates the associated humanoid character. This character itself consists of a hierarchy of bones and muscles. These bones and muscles are overlaid by a mesh renderer which produces the desired output, compare figure 3.5.

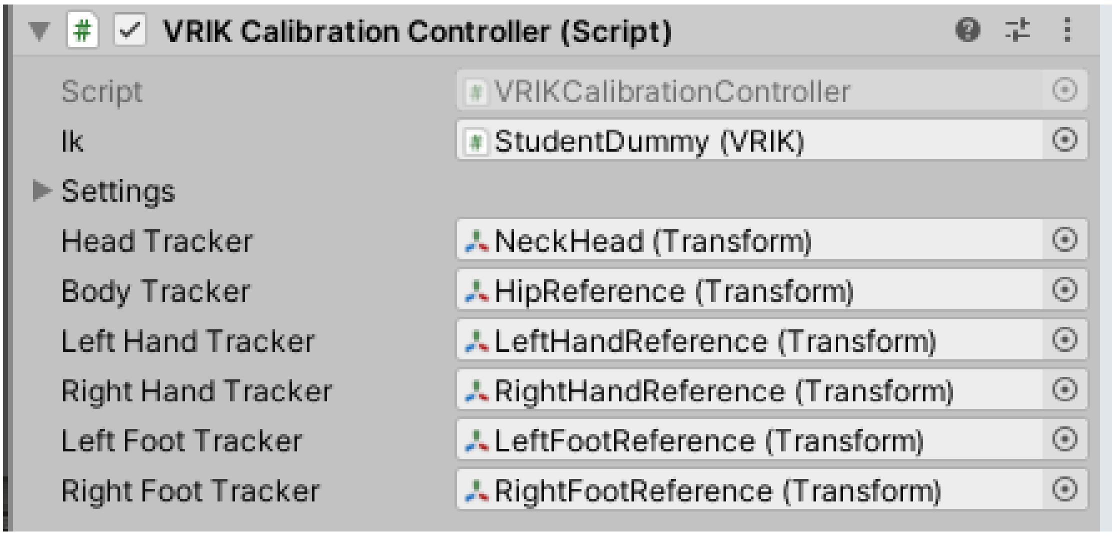

Figure 3.3: VRIK Calibration Controller. Sets the references necessary to calibrate the student.

Teacher Rendering

The naive approach of rendering the teacher is to record a teacher and then just play the animation. However, recording a humanoid animation is not natively supported by Unity and would require a third-party tool to do so. This leads to unwanted differences in the representation of the teacher. To ensure, that teacher and student produce comparable renderings; the same technique has to be utilised. To accomplish this requirement, the movements are recorded by recording the student's tracker position and then pass this as animations to a similarly shaped hierarchy of *GameObjects*, that in turn are passed to the VRIK solver. This is why the rendering pipeline of the teacher differs from the students rendering pipeline. The process for the teacher is as follows, compare figure 3.5: For the teacher, a reference hierarchy needs to be created. The calibration above applies, but the head must be extracted from the camera and transformed into an own *GameObject* with the heads reference turn by 90° on the Y and Z axis. On the parent *TeacherReference* an *AnimatorController* is attached. This *AnimatorController* plays the pre-recorded animation and transforms the references of the children accordingly. Now, these children can be used like in the students rendering pipeline and passed to the *CalibrationController*. Afterwards, the rendering pipeline is as seen in the student's pipeline.

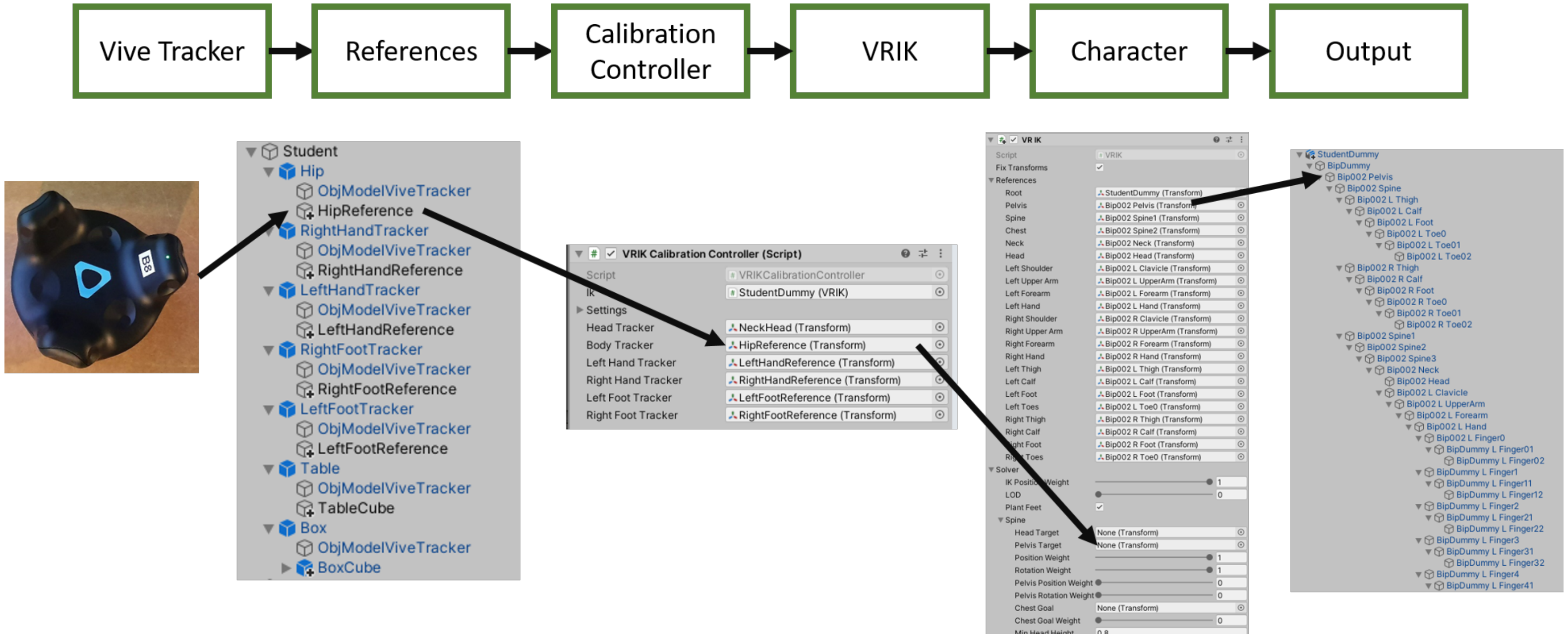

Figure 3.4: Top: data flow to generate the visual representation of a student. Bottom: Example of data flow from the Tracker placed on the hip of a student until the visual representation.

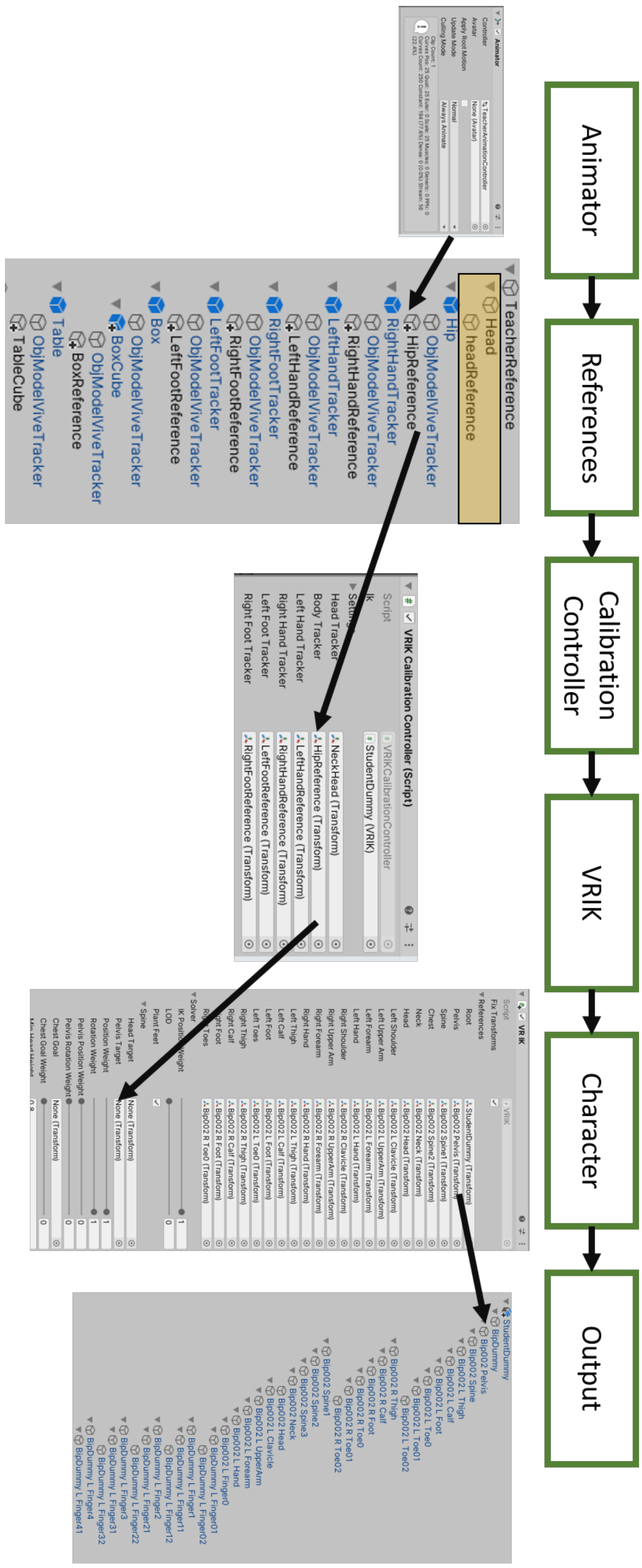

Figure 3.5: Top: data flow to generate the visual representation of a teacher. Bottom: Example of animating the hip of a teacher.

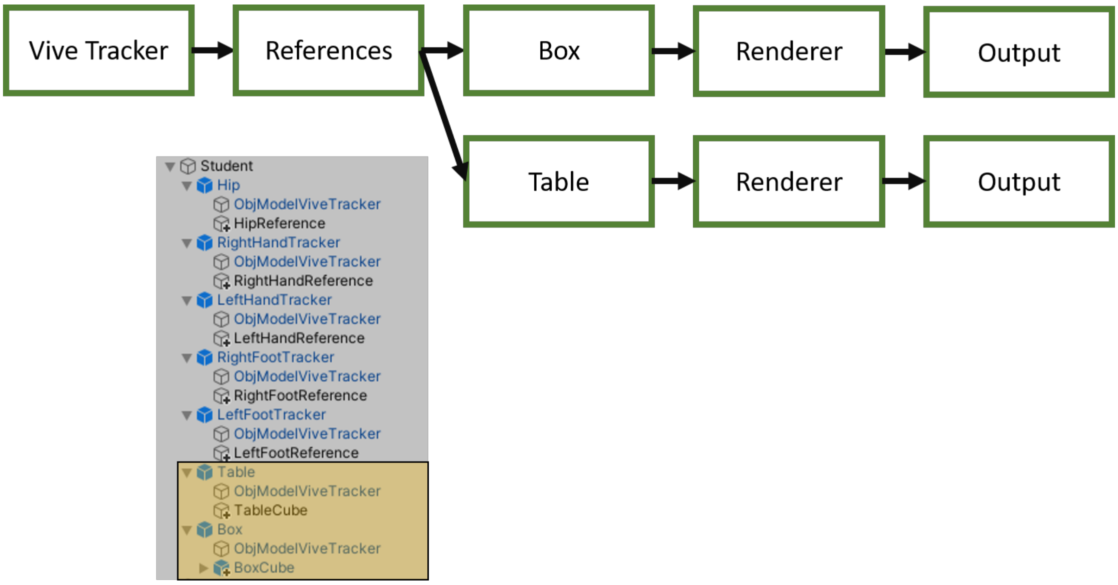

Figure 3.6: Data flow to generate the visual representation of the box and the table.

Assets Rendering

Finally, the objects the student and the teacher are interacting, namely the table and the box, need to be rendered. Rendering of these object is less complex. The student's objects get attached by a Vive Tracker. The position and rotation of the Tracker are bound to a cube, which is scaled to the size of the box or the table. The teacher's assets are also cubes scaled to the size of the cube and the table. But the position and rotation are set by the animator controller, compare figure 3.6

Recording of Movements

As mentioned above, the recording of humanoid movements is not trivial. The solution is to create a *GameObject* with a similar hierarchy, compare figure 3.7. This *GameObject* is called *Recorder*. It is similarly structured like the student. The references are holding a script which sets the transforms them into the transforms of the student. Additionally, the head reference, the box reference and the table reference. The transforms of these references are also copied from the real elements. This is necessary to have one single animation that controls the teacher. Otherwise, time shifts between animations could be possible. Now the complete *Recorder* hierarchy can be recorded by the *Unity Recorder*. The outcome is an animation

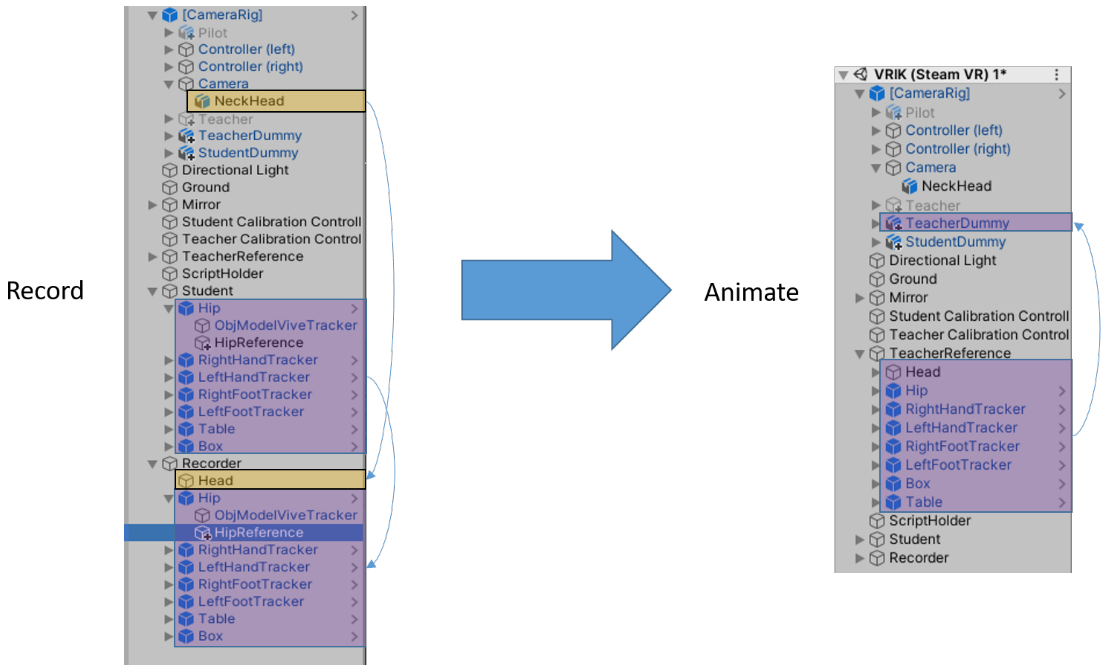

Figure 3.7: Left: setup of the recorder, GameObject that holds all points to record. Right: animated GameObject is used to steer the teacher's dummy.

that produces the references necessary to render the teacher like described above.

Resize

The teacher and the student are most likely to have different body sizes. Especially in the ego-centric perspective, the height of the teacher must match the height of the student. However, the naive approach of just resizing the teacher's humanoid character to the size of the student leads again not to a satisfying result. This is because the head position is absolute and not changeable for the VRIK solver. Resizing the teacher leads to floating or swaged representations. The solution for this is to resize the parent *GameObject* of the teacher's references, and then pass the already scaled references to the calibration controller. However, this leads to another issue: the size of the student can only be assessed after the calibration controller did his work, assigned the references to the VRIK solver and sized the representation. To overcome this issue, a 2-step body size calibration is necessary. First, the student's calibration controller performs the calibration of the student. Then the student's height is measured and compared to the teachers size and stored in Δy. Δy describes the percentage of the size difference between teacher

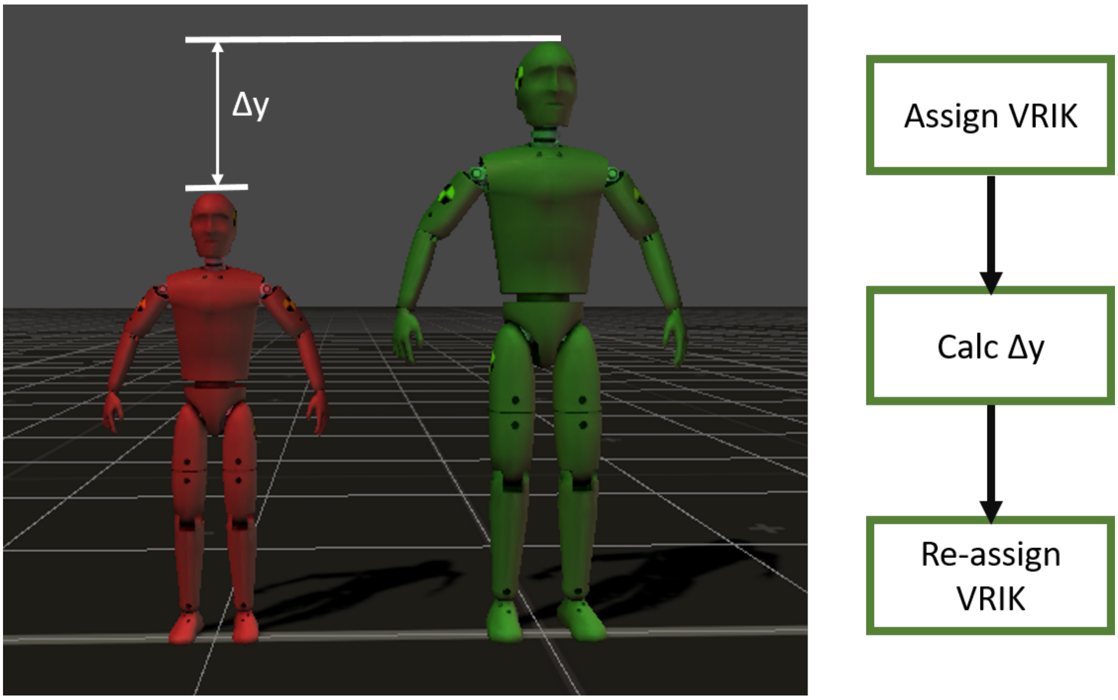

Figure 3.8: Calculating the height difference of Δy between the teacher and the student.

and student. Afterwards, the teacher's references root is scaled by Δy and gets reassigned to the calibration controller, which itself performs the task again. With this, the teacher has the same size as the student. However, now the box and table are also resized because they are in the same parent *GameObject*. However, with the knowledge about Δy the box and the table can be upscaled again by $\frac{1}{\Delta y}$ to have the original size.
The length of arms torso and feet are resized by VRIK Calibration Controller, which means a simple superordinate scaling is a valid resize method.

3.1 Visual Perspectives

With the previous implementations done, E(x|g)o is capable of showing a teacher and a student. This section describes the implementation of the visual perspective in which the student can learn movements. Namely, ego-centric, exo-centric, ego & exo-centric and augmented exo-centric, compare figure 1.1[1].

[1]Remark: E(x|g)o is also capable of the fifth possible visual perspective ego & augmented exo-centric. Because the focus is set to only the first four visual perspectives, the description of

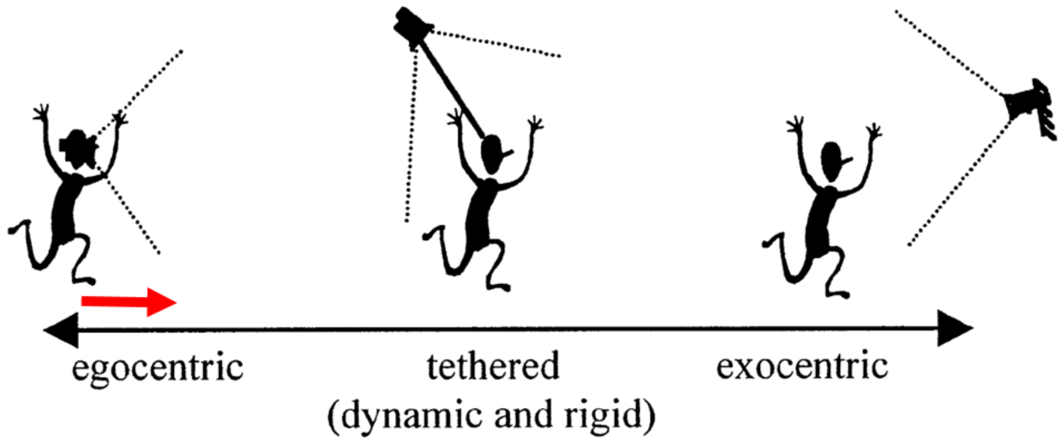

Figure 3.9: Continuum of perspectives according to Milgram [14]. The red arrow indicates the shift towards the exo-centric perspective.

Ego-Centric

The nature of an ego-centric perspective raises one big issue to tackle: the student has "to be in the teacher" at any point in time. If the teacher now wants to indicate a movement, meaning a translation of the own position in space, it is indicating this movement by moving away from the student. This leads inevitably to a non-ego-centric perspective. To solve this issue, a closer look at the definition of the visual perspectives can help, compare figure 3.9: a continuum represents the visual perspectives. On one extreme, the ego-centric visual perspective is located, while on the other extreme, the exo-centric perspective is located. With moving from one extreme to the other, the tethering distance[2] is changing. A tethering distance of 0 indicates a pure ego-centric perspective, while a greater tethering distance means shifting towards the exo-centric extreme. Following the nature of a continuum, a slightly larger tethering distance than 0 is still an ego-centric perspective. The task is now to choose a value for the tethering distance that it still is an ego-centric perspective, but at the same time let the student be able to interpret the teacher's indication of movement.
To determine a reasonable tethering distance, a small study was conducted with two persons[3] In this study, the tethering distance was increased continuously from 5 to 50 cm. The best result was yielded between 15 and 30 cm. Based on this empirical values, the tethering distance is set to 30cm. It still feels ego-centric,

the fifth visual perspective is not part of this report.

[2]The tethering distance is the distance between the eyes anchor point and the camera observing the character in question.

[3]Due to COVID-19, a larger study was not possible.

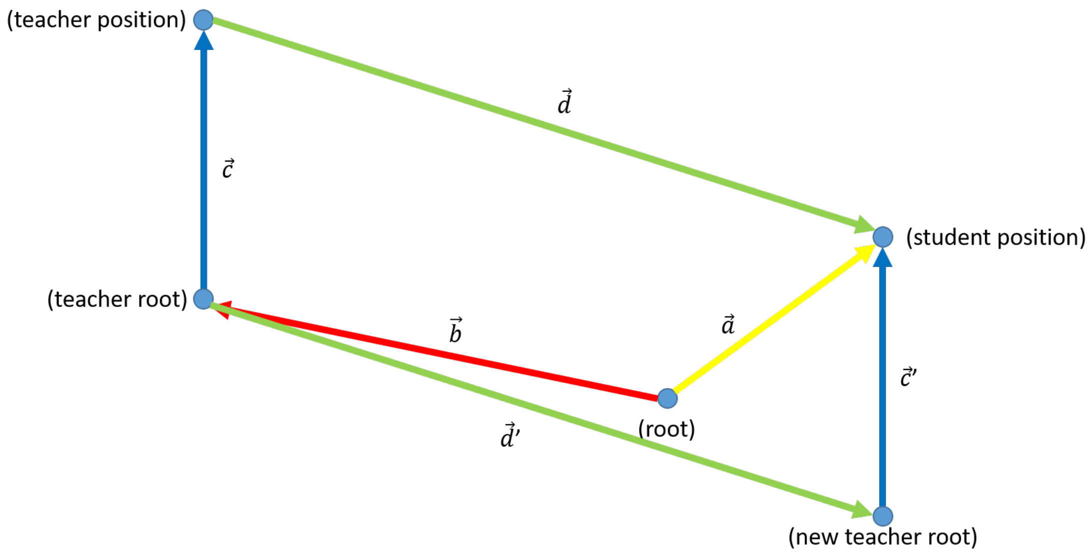

Figure 3.10: Process of shifting the teacher into the student.

but the indication of the movement of the teacher is clear to recognise. However, a hard threshold would lead to non-fluid demonstrations of the teacher, which is hard to follow for the student. For this reason, the tethering is set to a frame of 15cm to 30cm with a variable, interpolated speed of the teacher's demonstration. This means, between 0cm and 15cm, the teacher's movement demonstrations are shown at normal speed. Between 15cm and 30cm, the speed linearly decreases and stops completely at a tethering distance of 30cm. This makes the teacher's movements easily to follow by the student with no interruptions.
In the following, the implementation of this mechanic is described. First, the teacher must be translated to the position of the student. Again, the naive approach of just shifting the teacher's avatar on the student's position does not fulfil the requirements, because the VRIK solver takes the position of the references as absolute. Though, the teacher's references parents zero holding the animator has to be transformed.

For this, the distance between the parent of the references zero of teacher and student is calculated, and then the teacher's references parent is shifted by this distance. This process is shown in detail in figure 3.10: the teachers root must be shifted by $\vec{d}$, while $\vec{d}$ is determined by $\vec{a}$, $\vec{b}$ and $\vec{c}$. After shifting the teacher's root by $\vec{d}$, the animation ($\vec{c}$) of the teacher applies, and the teacher is placed directly in the student. Secondly, the animation speed is set by the distance between the student and the teacher. For this calculation, the hip references of both are taken into consideration. If the difference between these two points is below 0.15, the animation speed of the teacher is 1. If the distance is greater than 0.3, the animation speed is 0. Between 0.15 and 0.3 the animation is a linear interpolation. The following pseudocode achieves this:

```
float stopDistance = 0.3f;
float fullSpeedDistance = 0.15f;

void Update()
 {
  Vector3 deltaStudentTeacher =
  studentHip.transform.position -
  (teacherHip.transform.position -
  teacherZero.transform.position);

 if(deltaStudentTeacher.magnitude > stopDistance){
  setAnimationSpeedOfAllTeachersToZero();
 }
 else
 {
 allTeacherAnimators.speed =
  Mathf.Min(1f, (stopDistance / fullSpeedDistance -
  (deltaStudentTeacher.magnitude / fullSpeedDistance)));
 }
}
```

Exo-Centric

The exo-centric visual perspective is less complex than the ego-centric visual perspective. The teacher is located outside of the student, and the student can choose their position to observe the teacher. However, in this condition, the student can turn away from the teacher by following the instructions of the teacher. To visualise this issue, follow this scenario (demo video): teacher and student standing side by side and looking into the same direction. The student stands left of the teacher. If the teacher turns left and the student also turns left, the student can no longer see the teacher and follow the instructions. However, even when the teacher is turning to the right, the student sees the teacher from the back, unable to see what the teacher is doing in front of the teacher's body. For this issue, a solution must be found. Chua et al. [4] decided to introduce multiple representations of the teacher and found the mechanic as an aid for motor learning. Additionally, in real-world scenarios, the same issue exists. However, in real-world scenarios where this issue is likely, it is also likely that in the room are mirrors, allowing to see

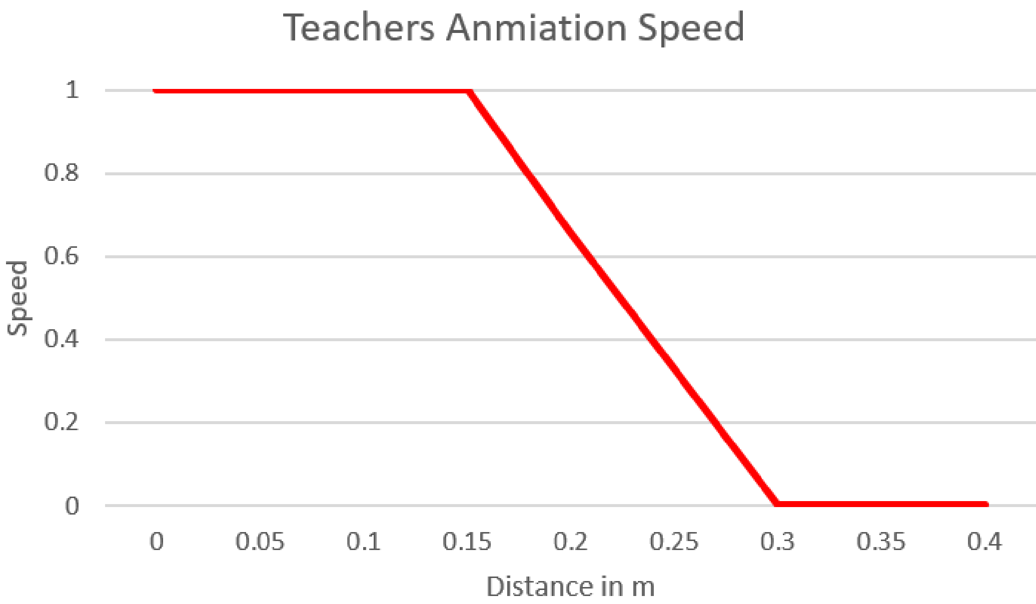

Figure 3.11: Teachers animation speed, linearly interpolated.

areas that are occluded by the teacher's body. VR provides the possibility to have multiple representations of the teacher around the student. With this argumentation, multiple representations of the teacher are introduced. Instead of having only one instance of the teacher, four instances on different positions around the student are present. To determine appropriate positions of the additional teachers, a small study was conducted with two people[4] During interacting with the box, the teachers were shifted around the student. After three iterations, the teacher's positions were set to:

$$\text{teacher0: } \vec{t_0} = (-0.2, 0, 1.5)$$

$$\text{teacher1: } \vec{t_1} = (-1.3, 0, -0.75)$$

$$\text{teacher2: } \vec{t_2} = (-1.3, 0, 0.75)$$

$$\text{teacher3: } \vec{t_3} = (-0.2, 0, -1.5)$$

Now the student has multiple angles on the teacher, which overcome the mirror issue[5], too.

[4] Due to COVID-19, a larger study was not possible.

[5] Mirror issue: a teacher is standing in front of the student forces the student to move the, e.g. left arm when the teacher moves the right arm. Compare seminar thesis for more details

Ego-Centric & Exo-Centric

The combination of the ego-centric and exo-centric perspective leads to the third visual perspective. As the name indicates, the teacher can be seen from the ego-centric perspective and the exo-centric perspective simultaneously: the teacher stands outside and inside of the student's body. To achieve this, the two above visual perspectives are combined. Starting from the above described exo-centric visual perspective with its four teachers, a fifth teacher is introduced and placed inside of the student with the same mechanics as in the above described ego-centric perspective. Eventually, the animation speed of the ego-centric teacher must be applied to the exo-centric teachers. This happens by the script *putTeacherIntoStudent.cs* which holds references to all animation controllers. The script synchronises all animations and changes the speed of the animation playback.

Augmented Exo-Centric

In the augmented exo-centric perspective, the student stands inside of the teacher. While the number of teachers remains at 4, the student now needs to have multiple representations. One at the real-world location, representing the ego-centric perspective, and four more in every teacher. The formation of the exo-centric perspective remains. Because no ego-centric perspective is present, the animation is not needed to have speed interpolation.

3.2 Limitations: Perspectives and Mechanics

The four visual perspectives utilise two mechanics for teaching movements to a student. *Multiple representations* are necessary to overcome occlusion and the mirror issue. The *speed* mechanic is used to indicate transitional movements in the ego-centric perspectives. *Multiple representations* is used in all perspectives except for the ego-centric perspective, compare table **??**. In the ego-centric perspective, it is not necessary to show multiple representations. Therefore the missing of this mechanic only in this perspective is not study corrupting.
The *speed* mechanic is applied in all perspective except the exo-centric perspective. The reason for this is that the student has no visual indication on what point in space the movements should be performed. This makes it nearly impossible to the step into the threshold of 30 cm to trigger the teacher's movement instruction.
Since the *speed* mechanic is essential in the other conditions, the missing of this mechanic in only one condition could lead to evaluate the mechanic and not the movements of the student. A solution could be to provide a light spot on the ground to indicate the position. Another approach would be to replace the exo-

	Speed	**Multiple representations**
Exo-centric	No	Yes
Ego-centric	Yes	No
Augmented Exo-centric	Yes	Yes
Ego & Exo-centric	Yes	Yes

Table 3.1: Comparison of visual perspectives with applied mechanics

centric visual perspective with the ego & augmented exo-centric condition. The final solution to this issue will be part of the Master's thesis.

4 Outlook

4.1 Upcoming Tasks

E(x|g)o is a system capable of training a student the handling of physical load in Virtual Reality in 5 different visual perspectives, including the perspective which probably not be used in the study. For an evaluation of these perspectives, now the measures need to be implemented, namely the ergonomic measurement and the precision measurement described in the next section. The comfort of study participants is vital during a study. For this, special straps are planned to replace the OptiTrack suit, which only was used for its Velcro surface. Furthermore, the box the student is handling needs to be replaced with a more elaborated one.

4.2 Evaluation

The evaluation of E(x|g)o can be conducted in two ways. The first is a comparative study, with the conditions ego-centric, exo-centric, ego & exocentric and augmented exo-centric. Here the participants have the task to handle a box ergonomically. Measures for ergonomic handling the box give insights on how well the participant could follow the instructions of the teacher. These measurements consist of spine twist, spine bend and foot placements like described by Muckell et al. [15]. Additionally, a precision measurement can be applied. The precision is defined as the Euclidean distance between the student's box and the teacher's box. The study would be a within-subject design with counterbalancing resulting in at least 16, better 32 or 48 participants. Because of the current corona situation, only an initial study with fewer participants could be conducted.
The other possibility to evaluate E(x|g)o is to evaluate the design decisions. In this case, the tethering distance in the ego-centric visual perspective could be refined, the size and weight of the box, as well as the formation of the teachers in exo-centric visual perspectives.

4.3 Schedule

In October, the tasks and measures will be implemented. The resulting software adjustments and recordings of the teacher are scheduled in November. December and January, the study will take place. After that, the Master's thesis will be written, till in March the colloquium will take place.

Bibliography

[1] Fraser Anderson, Tovi Grossman, Justin Matejka, and George Fitzmaurice. YouMove : Enhancing Movement Training with an Augmented Reality Mirror. pages 311–320, 2013.

[2] J C P Chan, H Leung, J K T Tang, and T Komura. A Virtual Reality Dance Training System Using Motion Capture Technology. *IEEE Transactions on Learning Technologies*, 4(2):187–195, apr 2010.

[3] Winyu Chinthammit, Troy Merritt, Pedersen Scott, Williams Andrew, Visentin Denis, Robert Rowe, and Furness Thomas. Ghostman: Augmented Reality Application for Telerehabilitation and Remote Instruction of a Novel Motor Skill. *BioMed Research International*, 2014, 2014.

[4] Philo Tan Chua and Russ Schaaf. Training for Physical Tasks in Virtual Environments : Tai Chi.

[5] Alexandra Covaci, Anne-Hélène Olivier, and Franck Multon. Third Person View and Guidance for More Natural Motor Behaviour in Immersive Basketball Playing. In *Proceedings of the 20th ACM Symposium on Virtual Reality Software and Technology*, VRST '14, pages 55–64, New York, NY, USA, 2014. ACM.

[6] K Hachimura, H Kato, and H Tamura. A prototype dance training support system with motion capture and mixed reality technologies. In *RO-MAN 2004. 13th IEEE International Workshop on Robot and Human Interactive Communication (IEEE Catalog No.04TH8759)*, pages 217–222, 2004.

[7] Ping-Hsuan Han, Kuan-Wen Chen, Chen-Hsin Hsieh, Yu-Jie Huang, and Yi-Ping Hung. AR-Arm: Augmented Visualization for Guiding Arm Movement in the First-Person Perspective. In *Proceedings of the 7th Augmented Human International Conference 2016*, AH '16, pages 31:1—-31:4, New York, NY, USA, 2016. ACM.

[8] Ping-Hsuan Han, Yang-Sheng Chen, Yilun Zhong, Han-Lei Wang, and Yi-Ping Hung. My Tai-Chi Coaches: An Augmented-learning Tool for Practicing Tai-Chi Chuan. In *Proceedings of the 8th Augmented Human International Conference*, AH '17, pages 25:1—-25:4, New York, NY, USA, 2017. ACM.

[9] Thuong N Hoang, Martin Reinoso, Frank Vetere, and Egemen Tanin. Onebody: Remote Posture Guidance System Using First Person View in Virtual Environment. In *Proceedings of the 9th Nordic Conference on Human-Computer Interaction*, NordiCHI '16, pages 25:1—-25:10, New York, NY, USA, 2016. ACM.

[10] Nicholas Katzakis, Jonathan Tong, Oscar Ariza, Lihan Chen, Gudrun Klinker, Brigitte Röder, and Frank Steinicke. Stylo and Handifact: Modulating Haptic Perception Through Visualizations for Posture Training in Augmented Reality. In *Proceedings of the 5th Symposium on Spatial User Interaction*, SUI '17, pages 58–67, New York, NY, USA, 2017. ACM.

[11] Taihei Kojima, Atsushi Hiyama, Takahiro Miura, and Michitaka Hirose. Training Archived Physical Skill through Immersive Virtual Environment. In Sakae Yamamoto, editor, *Human Interface and the Management of Information. Information and Knowledge in Applications and Services*, pages 51–58, Cham, 2014. Springer International Publishing.

[12] Taku Komura, Beta Lam, R W H Lau, and Howard Leung. e-Learning Martial Arts. *Lncs*, 4181:239–248, 2006.

[13] J Lieberman and C Breazeal. TIKL: Development of a Wearable Vibrotactile Feedback Suit for Improved Human Motor Learning. *IEEE Transactions on Robotics*, 23(5):919–926, oct 2007.

[14] Paul Milgram and Fumio Kishino. A Taxonomy of Mixed Reality Visual Displays. *IEICE Trans. Information Systems*, vol. E77-D:1321–1329, 1994.

[15] Jonathan Muckell, Yuchi Young, and Mitch Leventhal. A wearable motion tracking system to reduce direct care worker injuries: An exploratory study. pages 202–206, 07 2017.

[16] Vijay Rajanna, Patrick Vo, Jerry Barth, Matthew Mjelde, Trevor Grey, Cassandra Oduola, and Tracy Hammond. KinoHaptics: An Automated, Wearable, Haptic Assisted, Physio-therapeutic System for Post-surgery Rehabilitation and Self-care. *Journal of Medical Systems*, 40(3):60, dec 2015.

[17] Giuseppe Scavo, Fridolin Wild, and Peter Scott. No Title. In *Workshop Proceedings of the 11th International Conference on Intelligent Environments*, pages 236–243, 2015.

[18] Rajinder Sodhi, Hrvoje Benko, and Andrew Wilson. LightGuide: Projected Visualizations for Hand Movement Guidance. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, pages 179–188, New York, NY, USA, 2012. ACM.

[19] Maur\'\icio Sousa, João Vieira, Daniel Medeiros, Artur Arsenio, and Joaquim Jorge. SleeveAR: Augmented Reality for Rehabilitation Using Realtime Feedback. In *Proceedings of the 21st International Conference on Intelligent User Interfaces*, IUI '16, pages 175–185, New York, NY, USA, 2016. ACM.

[20] Richard Tang, Xing-Dong Yang, Scott Bateman, Joaquim Jorge, and Anthony Tang. Physio@Home: Exploring Visual Guidance and Feedback Techniques for Physiotherapy Exercises. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, pages 4123–4132, New York, NY, USA, 2015. ACM.

[21] Eduardo Velloso, Andreas Bulling, and Hans Gellersen. MotionMA: Motion Modelling and Analysis by Demonstration. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, pages 1309–1318, New York, NY, USA, 2013. ACM.

[22] Rebecca Weber. Egocentric Motion Guidance in Mixed Reality - A comparison of visualizations guiding bilateral arm movements, 2018.

[23] Shuo Yan, Gangyi Ding, Zheng Guan, Ningxiao Sun, Hongsong Li, and Longfei Zhang. OutsideMe: Augmenting Dancer's External Self-Image by Using A Mixed Reality System. In *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems*, CHI EA '15, pages 965–970, New York, NY, USA, 2015. ACM.

[24] Ungyeon Yang and Gerard Jounghyn Kim. Implementation and Evaluation of "Just Follow Me": An Immersive, VR-Based, Motion-Training System. *Presence*, 11(No. 3):304–323, 2002.

List of Figures

List of Tables