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

Stefan Paul Feyer
HCI Group, University of Konstanz

Maximilian Dürr
Supervisor

Master's Project Report
March 2020

Contents

Abstract

Motor learning includes a teacher that a student can mimic. In case a teacher is not available Mixed Reality systems proofed to be suitable for motor learning, too. There is many research in several aspects of motor learning in Mixed Reality. But the influence of the visual perspective on a guidance evaluation has not be considered much. E(x|g)o is a system designed to fill this research gap, providing different perspectives on guidance visualisations. With E(x|g)o a study can be conducted to investigate the influence of visual perspectives on guidance visualisations for motor learning.

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

[1] youtube.com

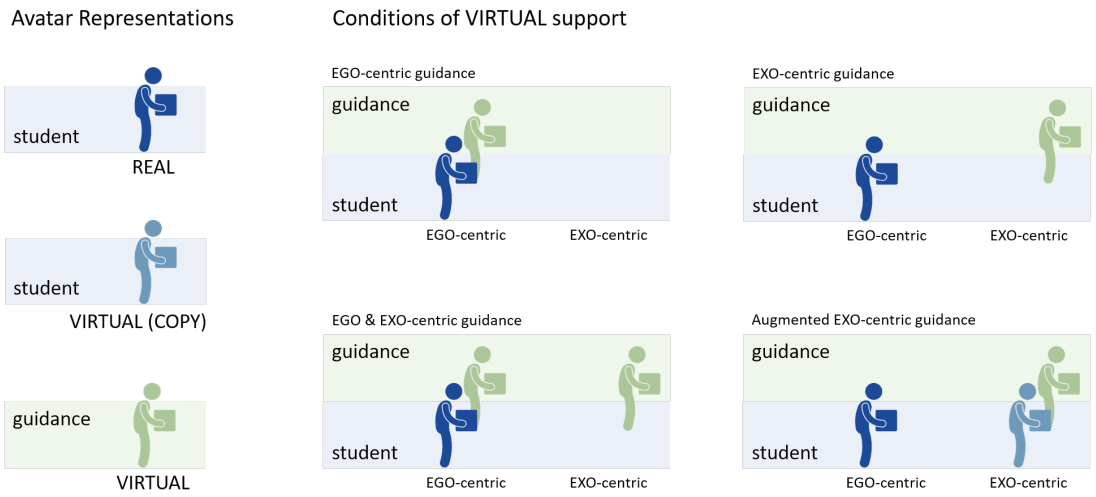

Figure 1.1: The four perspectives the system is capable of. Human figure taken from thenounproject.com, accessed: 19.06.2020. Icon: created by Ghan Khoon Lay from Noun Project.

teach students. MR system 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, this system can be interactive and provide feedback on the performance of the student. On this assumptions, researcher developed training systems for arts [3, 6, 11], dance [1, 5, 20], sports [4, 10] and rehabilitation [2, 14, 18], in the ego-centric [9, 15, 21], exo-centric [7, 12, 19] or both [8, 16, 17] 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. 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 showed that investigation on this topic is valuable because knowledge about how the visual perspective on a guidance visualisation influence motor learning is limited.
Furthermore, the seminar thesis investigated on several aspects of motor learning and revealed requirements a motor learning system must fulfil to train motions effectively. Additionally the scope was set:

- serial movements
- cognitive stage
- closed skills
- single error measures
- virtual reality (VR)

For this scope, the requirements are as follows.

R1 provide at least four different perspectives

R2 provide different view points on the guidance visualisations

R3 avatar guidance visualisation

R4 high realism degree, person shaped avatar

R5 no additional feedback

R6 scaling of the teachers height to the students height

R7 provide measures

R8 provide full body motion tracking

The four perspectives will be the ego-centric perspective, the exo-centric perspective, as well the combinations ego & exo centric perspective and the augmented exo-centric perspective compare 1.1.

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.

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.

This work starts with an evaluation of the hardware and software, followed by the implementation of the system and finally, the perspectives. In the end, an outlook is provided.

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 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 what want to be measured and also it must be as reliable as possible, to not disrupt the study procedure. For this reason, hard and software were analysed.

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 this new 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. Meanwhile, many headsets fulfil the requirements 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 wear-

Figure 2.1: Valve Index

ing glasses. For this project, the cable does not play a big role. Since all headsets worth considering fulfil the requirements 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 requirements. Though the choice here was easy[1]. A comparison of VR HMDs can be found in Table 2.3.

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

[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 2.2: Light House 2.0

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 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

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

Figure 2.3: 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.valvesoftware.com/de/index/headset, all accessed: 19.06.2020

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] uses the magnetic field of the earth to identify the position and orientation changes. However, magnetic sensors are prone to metal. Other systems [3] use accelerometer or gyroscopes or a combination to track the movements. However, there is still the drift problem: a position

[2]https://neuronmocap.com/
[3]https://www.rokoko.com/, 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	Plug and Play	Yes
OptiTrack	Yes but prone to noise	Setup specific	✓	All visible markers	native	Poor	No, with workaround
Vive Tracker	Highest	Ca. 100m²	✓	All visible markers	native	Ok*	native

Figure 2.4: Comparison of evaluated Motion Tracking technologies.

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 figure 2.4.

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. An evaluation of the Kinect 2.0 revealed that it is not suitable for motor learning. Compare test video.

OptiTrack

2.6 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 engine in use where it can be used to train movements. The movement area is big, depending on the camera setup and has high accuracy as long there is no noise. Additionally, with an external tool called OpenVR[4] the HMD can be

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

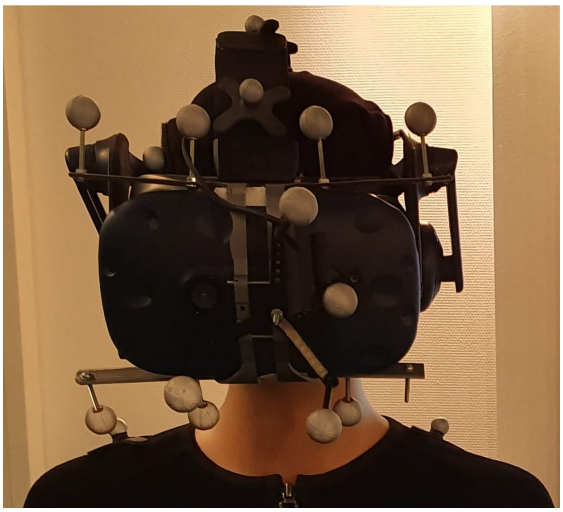

Figure 2.5: The marker set of the body (top and side) and the marker set of the HMD (at 3D printed mount) are too close to each other to be distinguished by OptiTrack.

natively tracked, too and thus no coordinate system matching is necessary. On the opposite side, there is the setup itself, compare figure 2.6 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 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 2.6 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 2.5, 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.

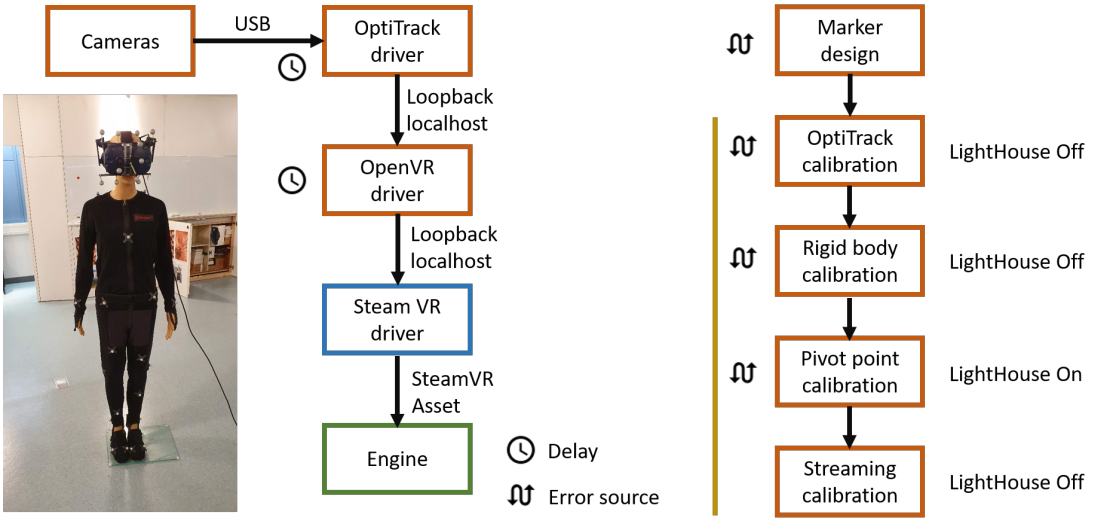

Figure 2.6: OptiTrack data flow and calibration process.

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*. 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 2.7, 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 2.9. On

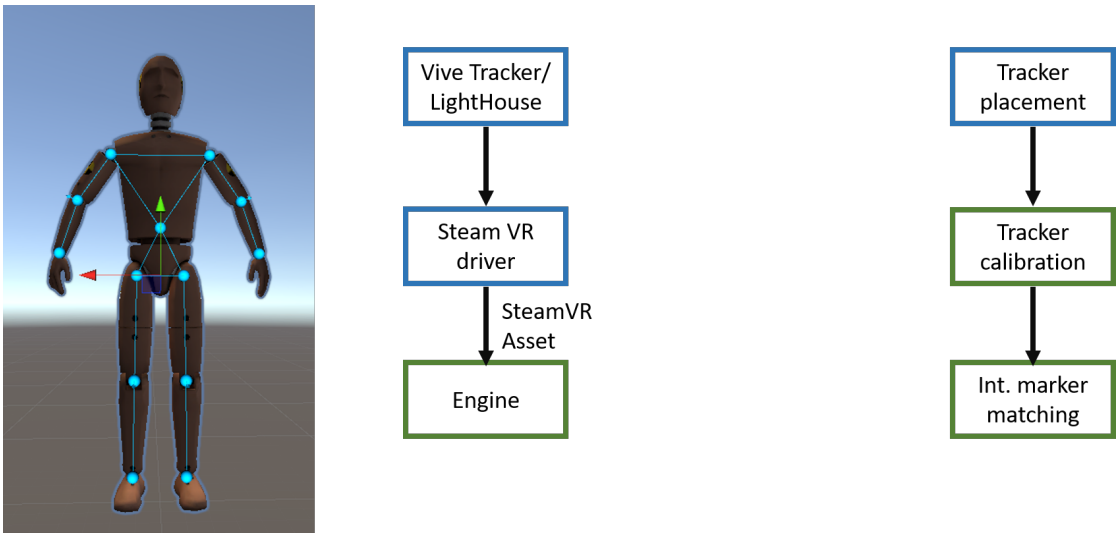

Figure 2.7: Vive Tracker data flow and calibration process.

Figure 2.8: Vive Tracker

Figure 2.9: 3D printed Vive Tracker Mounts with Velcro.

the back side Velcro tape is applied. Velcro holds perfectly on an OptiTrack 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 section pitfalls.

Box & Table

To conduct a study that includes physical load, a physical load prop must be designed. After several shapes and sizes were evaluated, the choice was made for a box with 27 x 21 x 21 cm. For developing a cardboard box was used. In addition, 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 **??**.

Figure 2.10: Box and table with Vive Tracker. Both shoes in the middle.

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[5] and Unity[6]. Both are suitable, and the choice can be made out of personal preference. In my opinion, Unity is much more intuitive 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 also easy, Final IK[7] is the most elaborated tool for the Unity engine.

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

[5]https://www.unrealengine.com/en-US/
[6]https://unity.com/
[7]https://assetstore.unity.com/packages/tools/animation/final-ik-14290

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.

2.3 Study Setting Overview

The complete setup looks like depicted in figure 2.11. it consists of

- 4x Lighthouse 2
- VR HMD: Valve Index
- 7x Vive Tracker 2
- 7x Vive Tracker mounts
- PC (SteamVR, Unity, FinalIK)
- OptiTrack suit
- Table
- Box

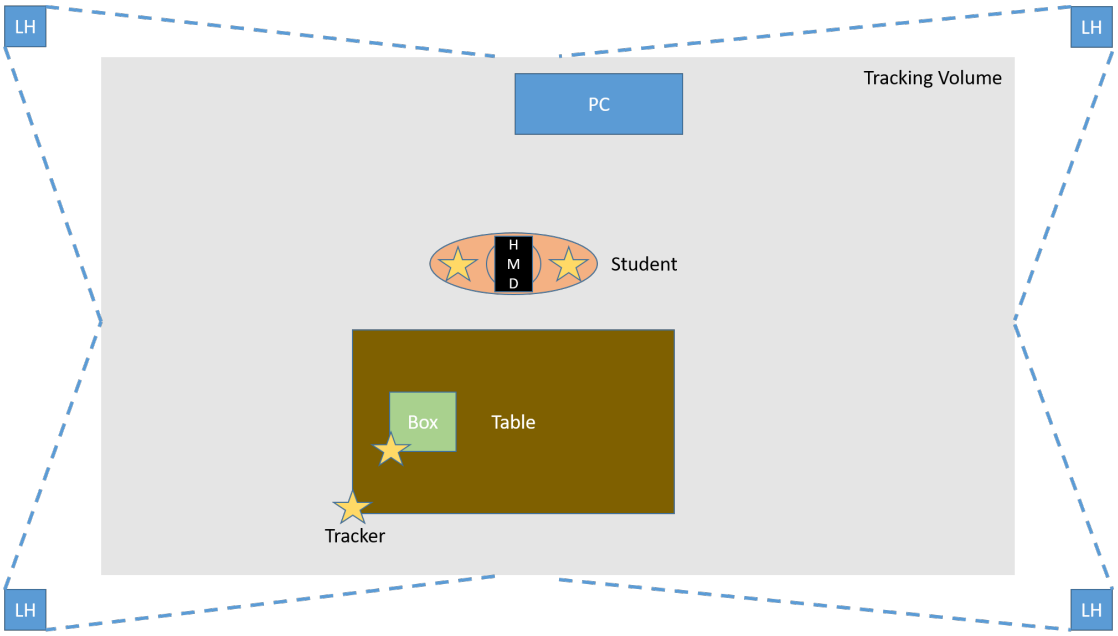

Figure 2.11: setup

3 Implementation

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 startup, 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 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

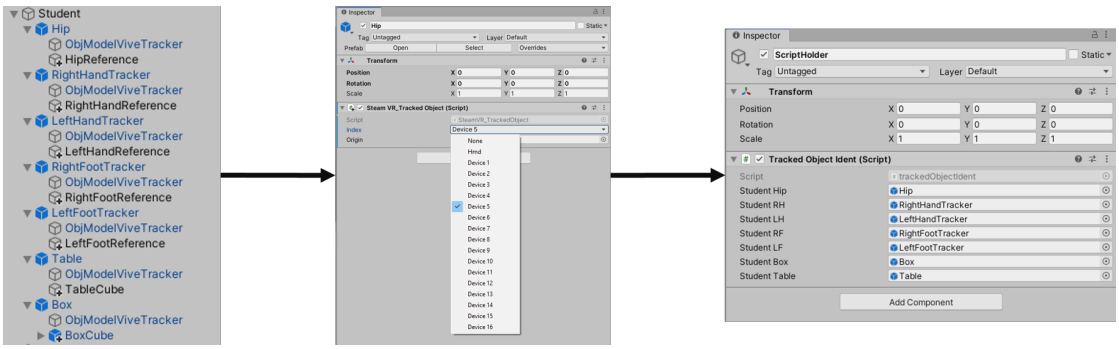

Figure 3.1: Vive Tracker matching. Left: reference holding all Objects to be transformed by a tracker, middle: device ID list, right script which reads the hardware IDs and sets the device IDs.

B8: LHR-89131158 BOX

B9: LHR-31D0CDF2 LHand

B10: LHR-CAC69A3C RHand

Secondly, the hardware id must be matched to specific devices. Then, at the startup 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;
 ...

}
```

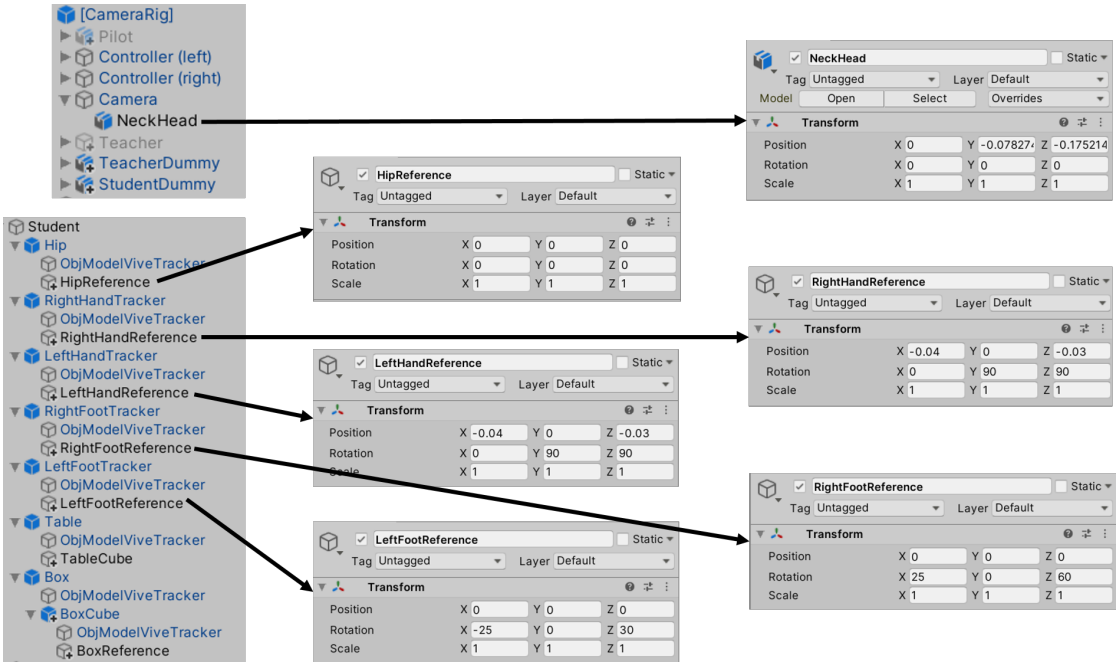

Figure 3.2: All elements and their relations to calibrate one student. The values of shifting and rotating the markers can be reused by other projects with an similar setup.

VRIK Calibration

VRIK solves the limbs based on the above mentioned tracking points. The head reference is a child of the *camera* provided by the SteamVR asset. It holds position and rotation and is the most important 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 overrides parent transforms. This fact influences the further handling in Unity massively: to transform a VRIK, the GameObject can not be translated to a specific position. This can only happen by transforming the references.

Left and right hand position needs 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.

Left and right feet need adjustments to in terms of position and rotation. The Vive Tracker is located at the top of the feet, which is fine with the root of the VRIK

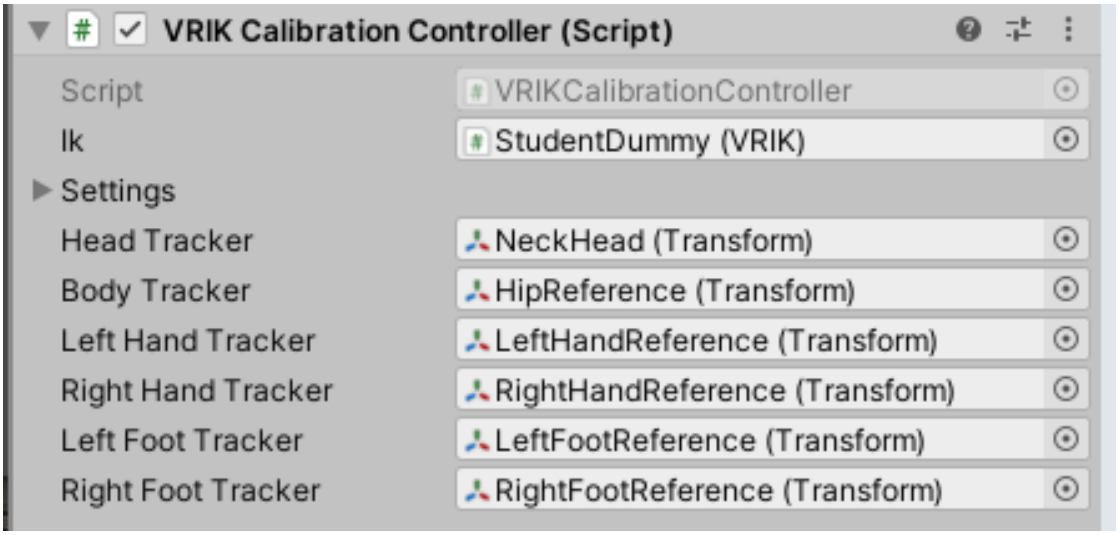

Figure 3.3: VRIK Calibration Controller. Sets the references necessary to calibrate the student.

reference. But 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 calibration are depicted in figure 3.2
With the correct position of the references, the references can now be assigned to an VRIK instance. This works with an 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 students 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 an humanoid character and binds it muscles. then the solving begins and the algorithm animates the associated humanoid character. This character itsself consists out of a hierarchy of bones and muscles. These bones and muscles are overlaid by a mesh renderer which produces the desired output, compare figure **??**.

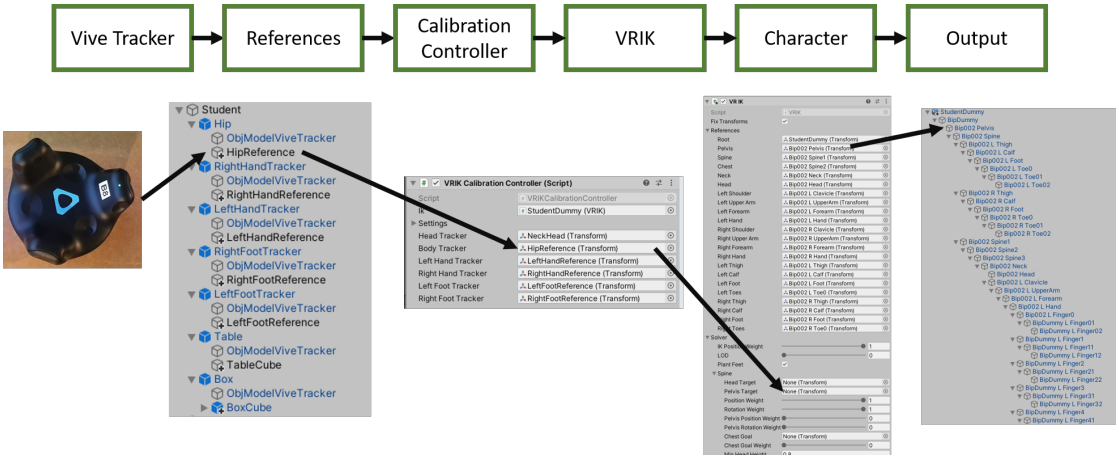

Figure 3.4: Top: data flow to generate the visual representation of a student. Bottom: Example of data flow from the tracker placed on the hip of a student till the visual representation.

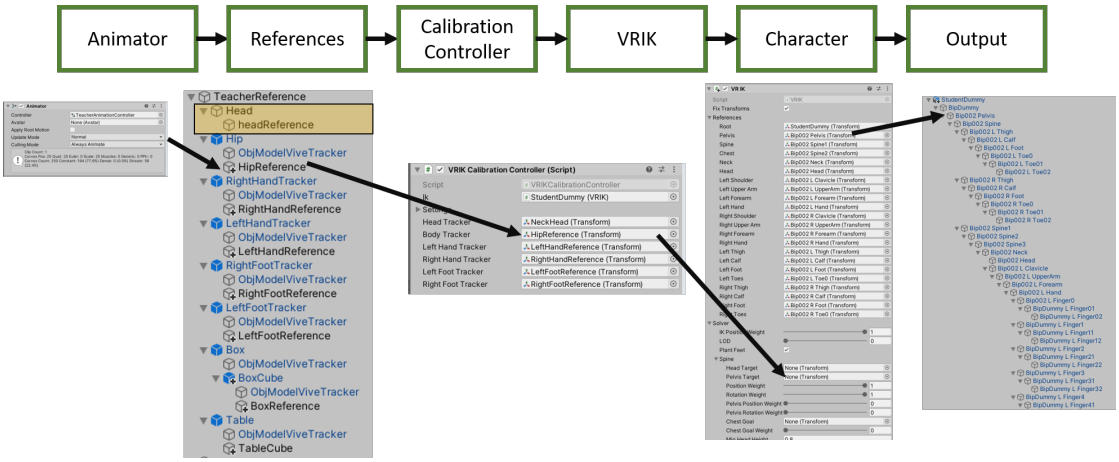

Figure 3.5: Top: data flow to generate the visual representation of a teacher. Bottom: Example of animating the hip of a teacher.

Teacher Rendering

The naive approach of rendering the teacher is to record a teacher and then just play the animation. But recording a humanoid animation is not natively supported by Unity and would require an 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 students

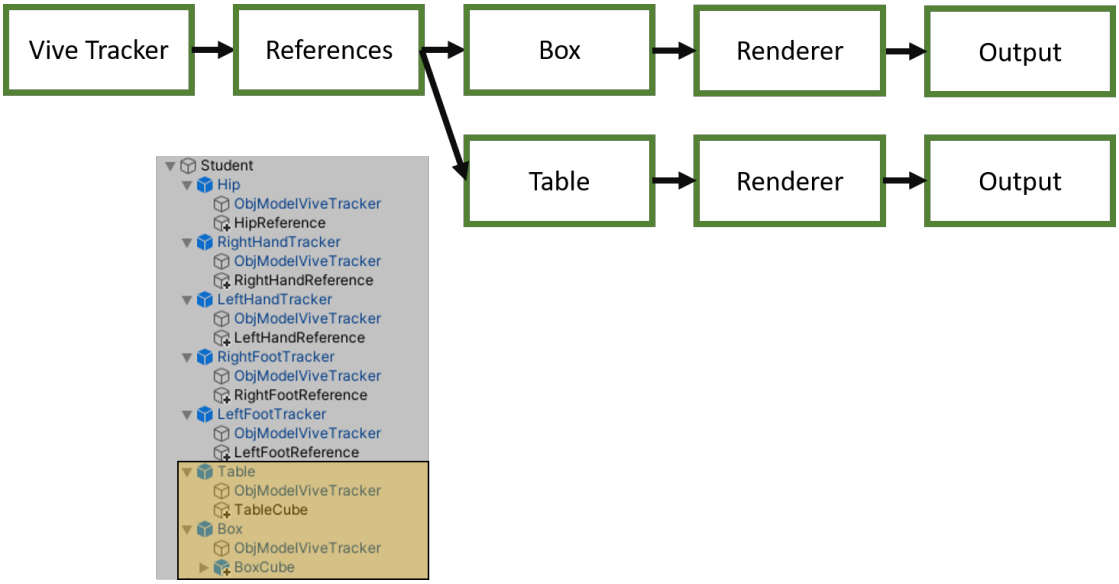

Figure 3.6: Data flow to generate the visual representation of the box and the table.

tracker position and then pass this as animations to a similar shaped hierarchy of *GameObjects*, that in turn are used are passed the an 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 students pipeline.

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 students objects get attached by a Vive Tracker. The position and rotation of the Tracker is bound to a cube, which is scaled to the size of the box or the table. The teachers 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

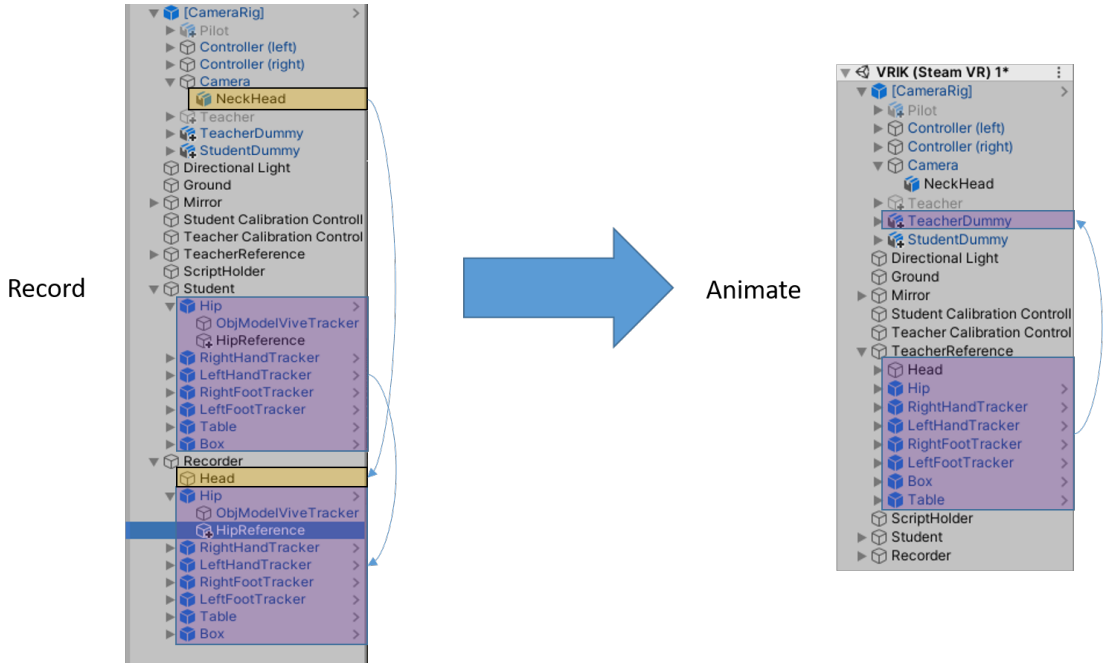

Figure 3.7: Left: setup of the recorder. GameObject that holds all points to record. Right: animated GameObject is used to steer a the teachers dummy.

Recording of Movements

As mentioned above, recording of humanoid movements are not trivial. The solution is to create a *GameObject* with a similar hierarchy, compare figure 3.7. This *GameObject* is called *Recorder*. It is similar structured like the student. The references are holding a script which sets the transforms them to 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 that produce 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 size of the teacher must match the size of the student. But the naive approach of resizing the teachers humanoid character to the size of the student leads again not to a satisfying result. This is because the head

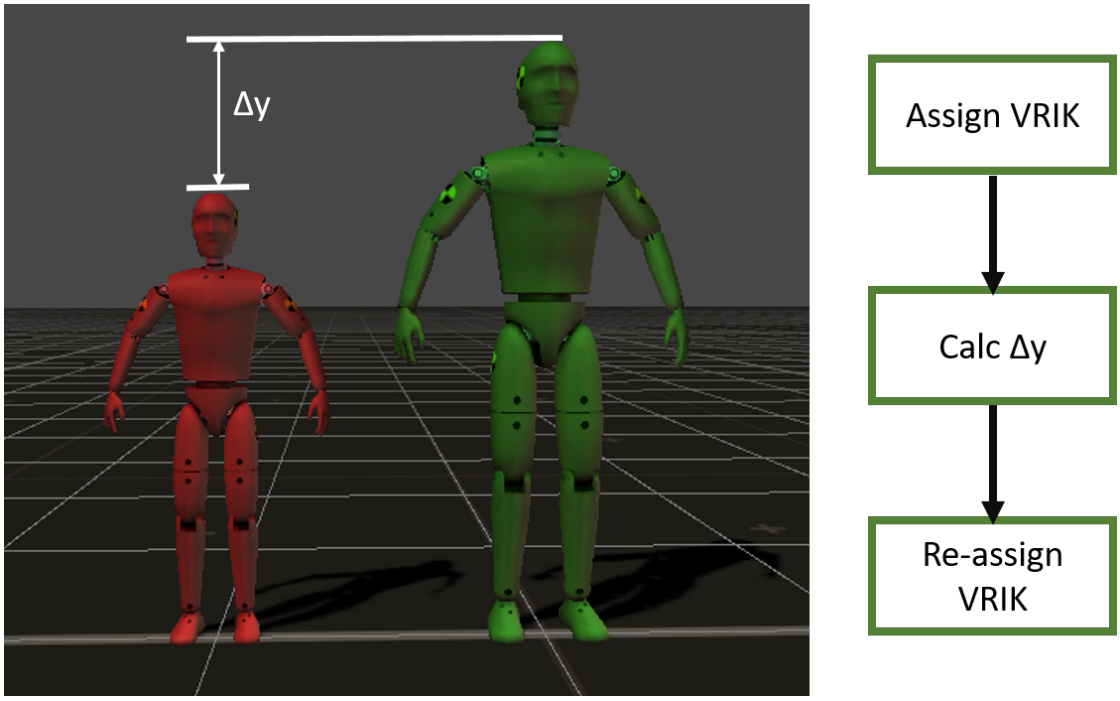

Figure 3.8: Calculating the height difference Δy between the teacher and the student.

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 of the the teachers references, and then pass the already scaled references to the calibration controller. But 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 the students calibration controller performs the calibration of the student. Then the students height is measured and compared to the teachers size and stored in Δy. Δy describes the percentage of size difference between teacher and student. Afterwards, the teachers 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.

3.1 Visual Perspectives

Ego-Centric

The nature of an ego-centric perspective raises one big issue to tackle: the student has "to be in the teacher" in any point of 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 inevitable to an non ego-centric perspective. To solve this issue, a closer look on the definition of the visual perspectives can help: the visual perspectives are represented by a continuum. 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[1] 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 an 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 still it still is an ego-centric perspective, but at the same time let the student be able to interpret the teachers indication of movement. Based on empirical values the tethering distance is set to 30cm. It still feels ego-centric, but the indication of the movement of the teacher is clear to recognise. But 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 teachers demonstration. This means, between 0cm and 15cm the teachers movement demonstrations are shown in normal speed. Between 15cm and 30cm the speed linearly decreases and stops completely at a tethering distance of 30cm. This makes the teachers movements easily to follow by the student with no interruptions.
In the following, the implementation for this mechanic is described. First the teacher must be translated to the position of the student. Again, the naive approach of just shifting the teachers avatar on the students position doesn't fulfil the requirements, because the VRIK solver takes the position of the references as absolute. Though, the teachers 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 teachers references parent is shifted by this distance. This process is shown in detail in figure 3.9: 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 teachers root by $\vec{d}$, the animation ($\vec{c}$) of the teacher applies and the teacher is placed directly in the

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

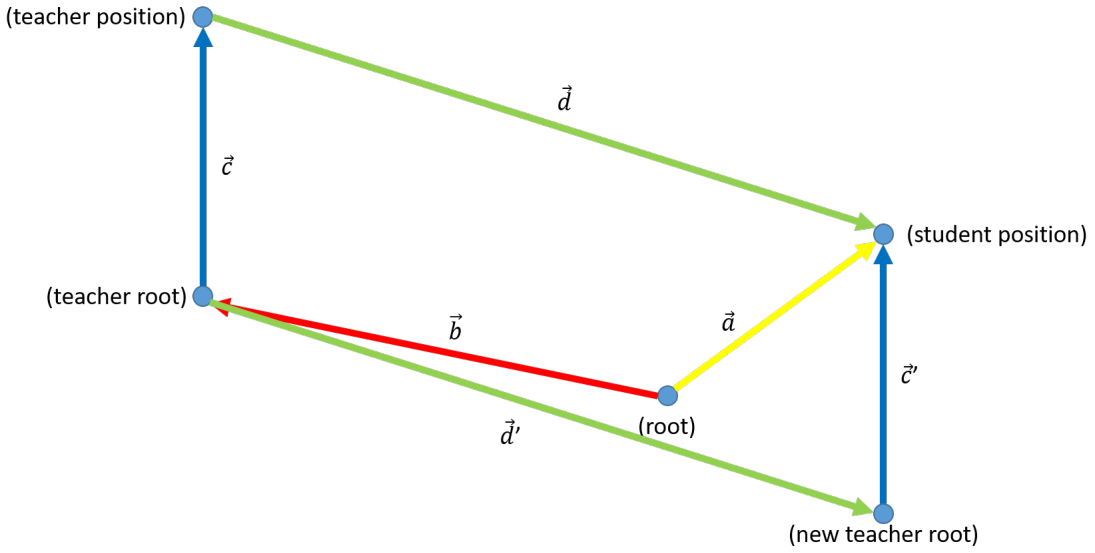

Figure 3.9: Process of shifting the teacher into the student.

student. Secondly, the animation speed is set by the by the distance between the student and the teacher. For this calculation, the hip references of both are taken into consideration. Is the difference of these two points 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 is used for that:

```
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 the own position to observe the teacher. But in this condition the student can turn away from the student 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. But 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 teachers body. One could argue that in real world scenarios the same issue exists. But in real world scenarios where this issue is likely, it is also likely that in the room are mirrors, allowing to see areas that are coved by the teachers body. 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. By empirical values, the teacher is translated by:

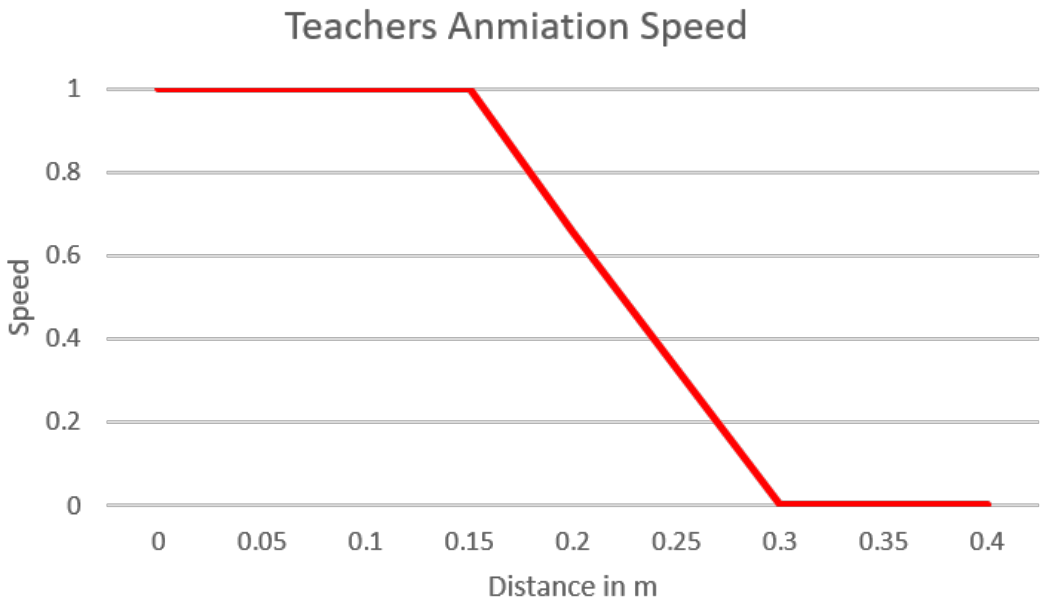

Figure 3.10: Teachers animation speed, linear interpolated.

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

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

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

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

Now the the student has multiple angles on the teacher, which overcome the mirror issue[2], too.

Ego-Centric & Exo-Centric

The combination of the ego-centric and exo-centric perspective leads to the third visual perspective. Like the name indicates, the teacher can be seen from the ego-centric perspective and the exo-centric perspective simultaneously: teacher stands outside and inside of the students 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

[2]Mirror issue: a teacher in 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

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 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 are not needed to have speed interpolation.

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. Comfort of study participants is important during a study. For this, special straps are planned to replace the OptiTrack suit. Furthermore, the box the student is handling needs to be replaced with more elaborated one.

4.2 Evaluation

The evaluation of E(x|g)o can be conducted in two ways. The first is a comparable 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. [13]. Additionally, a precision measurement can be applied. The precision is defined as the Euclidean distance between the students box and the teachers box. The study is a within subject design with counterbalancing resulting in at least 16, better 32 or 48 participants. Because of the current corona situation, an only initial study with less participants could be conducted.
The other possibility to evaluate E(x|g)o is to make a evaluation of the design decisions. In this case, the tethering distance in the ego-centric could be refined, the size and weight of the box, as well as the formation of the teachers in exo-centric perspectives.

Bibliography

[1] 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.

[2] 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.

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

[4] 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.

[5] 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.

[6] 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.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

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

[12] 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.

[13] 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.

[14] 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.

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

[16] 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.

[17] 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.

[18] 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.

[19] 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.

[20] 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.

[21] 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