Providing essential feedback for the self-directed learning of Kinaesthetics-based patient transfer movements in virtual reality

Master Project Report

submitted by

Tanveer Singh Mahendra (01/867038)

at the

Universität Konstanz

Human-Computer Interaction Group

Department of Computer and Information Science

Advisor: Maximilian Dürr

Reviewer: Prof. Dr. Harald Reiterer

Konstanz, 12. October 2020

Abstract

Patients with functional disabilities are often cared for by Geriatric nurses. While assisting with their daily activities, nurses are often required to lift and move the patients from one place to another. To reduce the physical deterioration caused by these movements, courses based on kinaesthetics care conception are taught in medical schools in Germany. However, these academic courses lack enough practical experience. With the goal to promote self-directed learning in virtual reality, we developed a system for providing essential feedback to the learner training for kinaesthetics-based patient transfers. The feedback is based on the four risk-metrics related to frequent injuries amongst healthcare workers. By integrating this system into a virtual training task, the learner can get real-time feedback that allows them to fixes errors in their techniques. The system also helps the learner to review their training in a 3D interactive environment from different perspectives. In this report, we go through the design and implementation process of creating this system. We also take a look at the challenges faced and alternative solutions that were adapted. Further, we examine the advantages and limitations of such a feedback system. The long term research goal is to integrate this system into a task and conduct a qualitative usability study.

Contents

List of Figures

List of Tables

1 Introduction

Recently, mixed-reality technologies have been improved to an extent that they would allow learners to be trained in virtual training environments. This successively has led to an improvement in wearable technologies. These wearable technologies can increase awareness and aid in the correction of mistakes in a learner's posture or body movement for various tasks. One of the use-cases facilitated by these technologies could be the training of patient-transfer movements for the nurses. In this chapter, we will talk about the motivation behind developing such a virtual training feedback system. And later we discuss how we plan to fulfill our overall goal from such a system.

1.1 Motivation

The elderly population (people aged > 65 years) in European countries is rising at a rapid rate. Germany is among the top three countries with the highest elderly population of 21.45%. This has led to an increase in the number of nursing-care personnel in Germany [Figure 1.1]. One substantial part of nursing care is patient transfers, which comes under mobility care. Although transferring is an important part of taking care of the elderly, it is also a major cause of risk in nurses. The physical risks majorly include lower back pain(LBP) due to musculoskeletal strain and other physical detrimental effects such as Musculoskeletal Disorders [1, 2, 3]. Musculoskeletal Disorders or MSDs are injuries and disorders that affect the human body's movement or musculoskeletal system (i.e. muscles, tendons, ligaments, nerves, discs, blood vessels, etc.).

With the intention of reducing the physical deterioration of the nurses, patient-transfer courses based on kinaesthetics care conception are taught in medical schools. However, figure 1.2 highlights the limitations of the current educational system for the nurses. The red mark highlights the pain-points in the system. In the existing curriculum, nursing-care students mostly only take part in one basic kinaesthetics practical training course over three nonconsecutive days in a formation of three years. This shows that opportunities to gain practical knowledge are quite limited. Since, from this course, typically no further support for learning kinaesthetics transfers is provided by these educational institutions [4]. The knowledge transfer in the field from experienced nurses is also lim-

ited by the factor that the curriculum changes over time. The investigation conducted by Dürr et al. (2019) clearly states that nurses report further need for self-directed practical training. As per the implications, this need is characterised by support for interactive learning by instruction, feedback, and reflection mechanisms [4].

Staff in care homes and home care services[1]

Specification	Unit	2007	2009	2011	2013	2015	2017
In care homes	number	573,545	621,392	661,179	685,447	730,145	764,648
including							
full-time staff	number	202,764	207,126	212,416	203,715	209,881	220,958
In home care services	number	236,162	268,891	290,714	320,077	355,613	390,322
including							
full-time staff	number	62,405	71,964	79,755	85,866	96,701	109,657

1: As at 15 December.

As at 24 July 2019

Figure 1.1: The increasing number of nursing care staff in home care services [5].

Previous researches have effectively demonstrated that interactive technologies like smart mirrors [6], virtual reality [7] and vibrotactile feedback mechanisms [8] can be used for training of movements. Although these implementations are limited by training movement which includes a single-entity(oneself), the learnings can be carefully translated for a system with two entities. There exists a virtual patient transfer system developed by Daniel Schweitzer as a part of his master project. The aforementioned system tries to mimic the process of a real-patient transfer setting in a virtual reality environment. This system suffices some part of the self-directed learning by enabling the learner to experience the patient transfer in a virtual reality environment. However, there are other factors like - feedback related to posture based on kinaesthetics concepts, revisiting and reflecting on errors made during the training which are not addressed as the inherent part of the virtual patient transfer system. Hence, there is a possibility to extend system based on the aforesaid factors. The next section gives a brief overview of how I plan to develop a feedback system considering various aspects of patient-transfer mechanics based on the findings from the literature research.

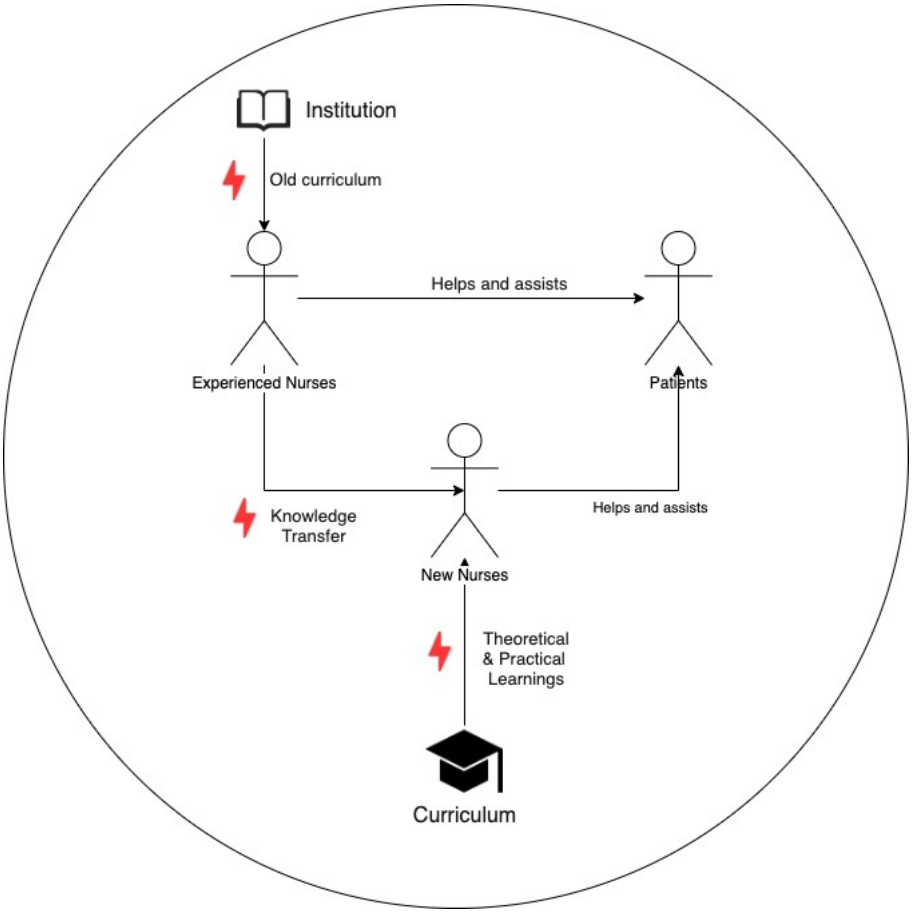

Figure 1.2: Limitations of the current system.

1.2 Goal - An implemented proof of concept

Previous researches in nursing care have developed an understanding of how MSDs impact health-care workers, specifically geriatric nurses. They also suggest that proper training including reinforcement of lifting techniques and body mechanics are the most important factors to reduce injuries [9]. To bring the real-world procedure close to the effective procedure, frequent and timely feedback to the nurses is critical [10, 11]. In my seminar research, I have discussed the benefits of various feedback modalities depending on the learning stage of the learner. Similarly, other researches in the field of virtual movement training have highlighted how different types of feedback can be beneficial [12]. The literature research has led to an understanding that the system would require two types of feedback. The two types of feedback are necessary to facilitate two factors namely - facilitating feedback related to posture errors during the patient-transfer and helping the learner to reflect on the errors made after the patient-transfer is completed.

These two categories of feedback mechanisms can be categorised as a function in relation to time.

- Concurrent feedback
- Terminal feedback

The overall goal of our system would be to facilitate the learner(user of the system) with these two types of feedback while performing a patient-transfer in virtual reality. From now on-wards, we address the user of our system as the 'learner'. The learner here is considered to be an geriatric nurse who has basic knowledge of kinaesthetics-based patient transfer and also, is motivated for self-directed learning. All through the report, you will find sections of the chapters divided amongst these two categories. Each category will discuss the topic of the section in regards to the particular type of feedback. Now, we further discuss in brief how these feedback goals can be realised.

One of the important studies conducted by Muckell et al. (2017) is directed towards reducing the injuries in Direct Care Workers(DCWs) employed by nursing and residential care facilities. This study highlights four key risk-metrics which are common in different lifting and carrying techniques [Figure 1.3]. These risk-metrics are considered important since they are related to stresses on the disc, vertebra, muscles, and ligaments of the low back. As per the earlier section, lower back pain is one of the most recurring musculoskeletal disorders among the nurses carrying patient-transfers. I decided go ahead with these four-risk metrics which were also mentioned as a part of my seminar research. These risk-metrics are as below:

1. Detecting Wide Support Base
2. Detecting Squat
3. Detecting Good Posture (Upright Stance)

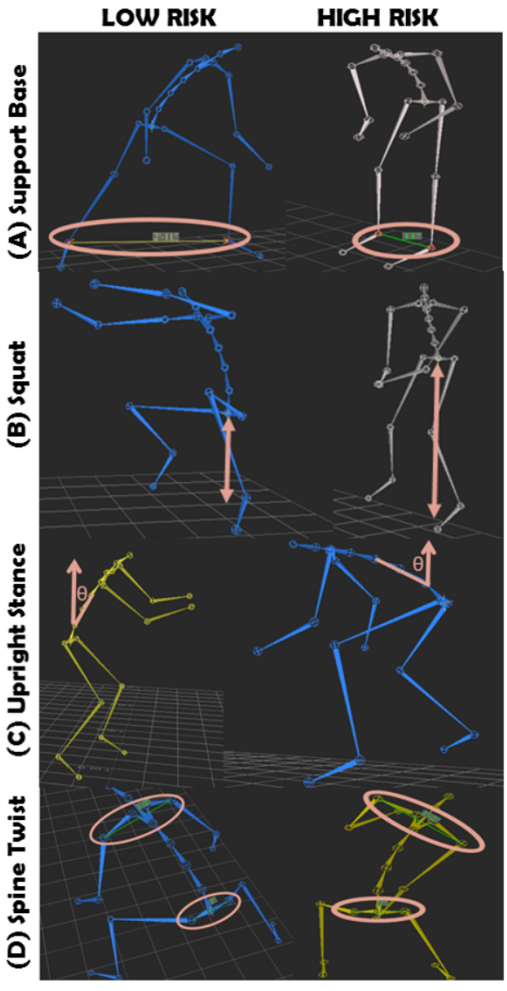

Figure 1.3: Low vs. high risk motions for each metric [13].

4. Detecting Good Posture (Avoid Spine Twist)

The first goal of the system is to make the learner aware of the errors they have in their posture while performing the patient-transfer movements. These posture errors will be based on four risk-metrics mentioned above. The system should have a means of tracking different body-part locations. And also, in-turn means to inform the user about the error using appropriate modality. The comparison values to be used for measuring the accuracy of the movements based on kinaesthetics concepts can be found in the study by Muckell et al. (2017). These values for comparison are stated as gold-standard since they are performed by highly experienced physical therapists with a Doctorate in physical therapy and over 15 years of experience [13].

The second goal of the system is to help the learner reflect upon the anomalies that happened when they were performing the virtual patient-transfers. This type of terminal feedback is very much necessary for complex tasks that consist of a second entity/actor. Many movement training systems developed in the past use varying kind of terminal feedback methods. Some of these methods can help the learner to reflect. These methods include recordings [6, 14], metric values (like time and score) [15], expert evaluation score [16] to name a few. Taking these two goals as a basis, we move ahead to discuss what you can expect further.

1.3 Outline

The focus of this report is to provide a detailed understanding of how the envisioned system was created. First, we go through the design thinking processes of how we collected learnings from the literature research. We apply those learnings to our use-case and visualize them as a part of a sketching process. Further, by analysing the storyboard, we define the hardware requirements for such a system. Next, we take a look at existing technologies that suit our requirements and see their comparison which helped us to make an informed choice. Once the technology platform is decided we dive into the system and implementation details. Further, we address the limitations of our system. And finally, we conclude by discussing potential future works and study design for the thesis.

2 Design thinking

For the initial part of the master project, it was necessary to dive more into the concept of the system. This was essential so as to provide a decent foundation for the implementation. The design thinking process helped me to review and improve the learnings carried forward from the seminar research. For this phase, I focused on the first two stages of the UX design life-cycle, namely - understanding user needs and creating design concepts [Figure 2.1]. The sections of this chapter discuss the details related to these stages of the life-cycle.

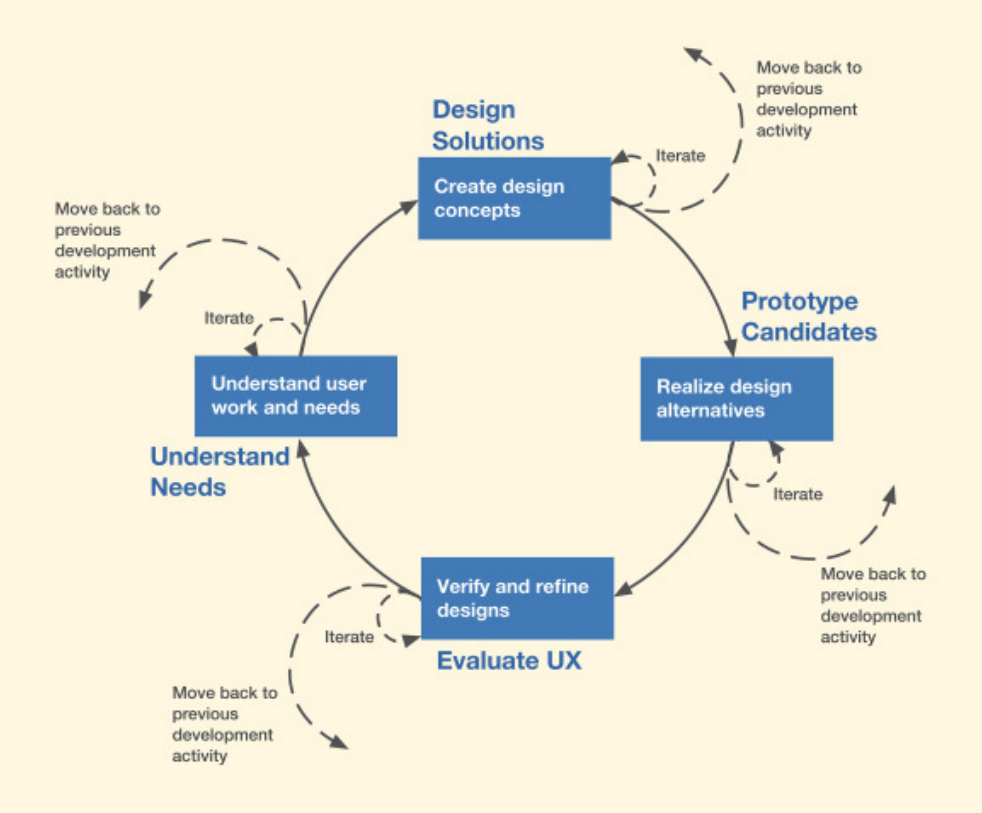

Figure 2.1: UX Design lifecycle [17].

2.1 Revisiting the learnings

As a part of master seminar work, a storyboard was created which visualised an envisioned system. Figures 2.2 to 2.9 provides a glimpse of that scenario. But from the feedback received during the final seminar presentation, it was easy to perceive that the concept required more refinement. This came from a perspective, that details of the concept storyboard made as a part of my seminar were unclear. The feedback received after the seminar presentation was documented so as to act upon it. On analysing the feedback comments, it was realised that there are some shortcomings in the concept like,

1. Feedback methodologies from previous implementations [6, 14, 16, 18, 19], might not work well into virtual reality.

2. The task complexity significantly impacts how and what kind of feedback the learner perceives.

3. The feedback will not be directly related to the task at hand (i.e. transferring the patient).

Also, the storyboard neglected the important details related to the terminal feedback [Figure 2.8]. The terminal feedback had relevant elements from previous related work. But, it lacked the elements of how the virtual reality environment can be leveraged so as to improve the patient-transfer training feedback experience.

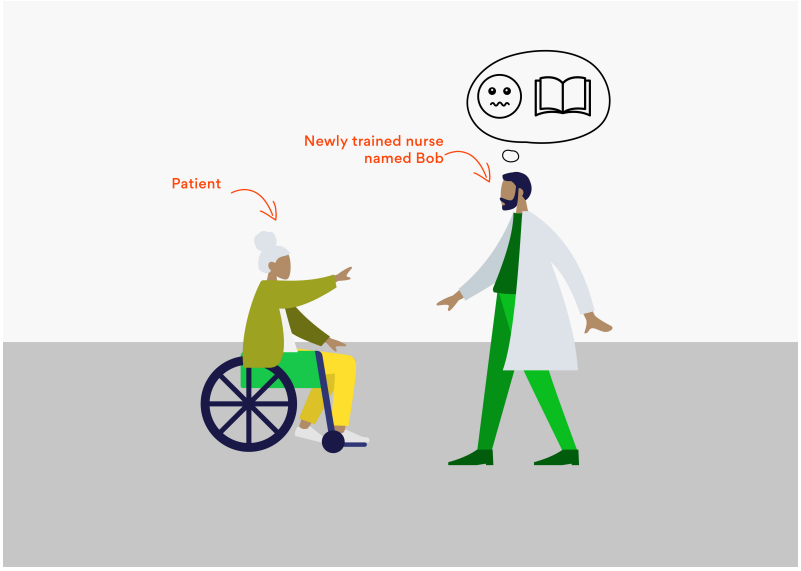

Figure 2.2: Storyboard introduction frame.

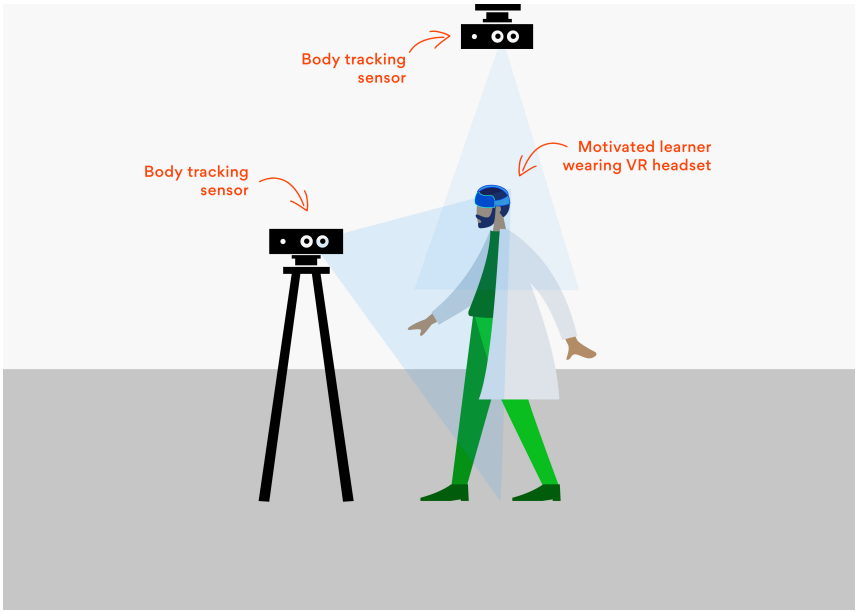

Figure 2.3: Storyboard - User wears the HMD and the system is initiated.

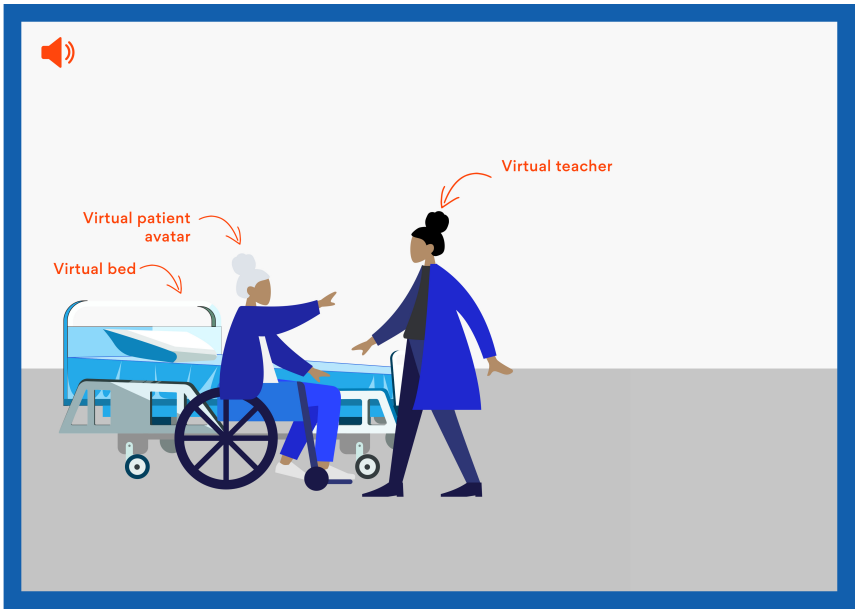

Figure 2.4: Storyboard - User observes a virtual patient transfer recording.

Figure 2.5: Storyboard - User starts to perform the guided patient transfer.

Figure 2.6: Storyboard - User gets notified about the error in his posture

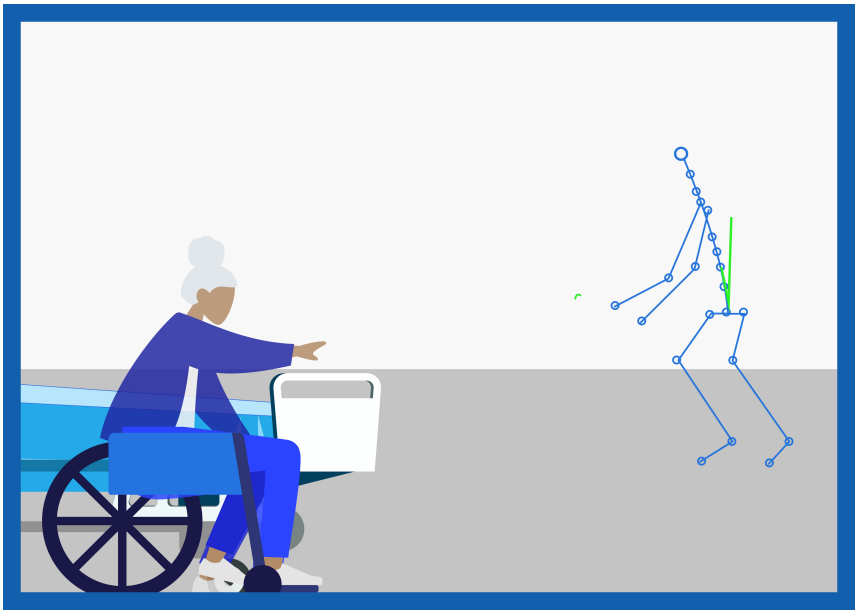

Figure 2.7: Storyboard - User fixes the erroneous posture and completes the movement.

Figure 2.8: Storyboard - User analyses his movements recordings and reflects on his learnings.

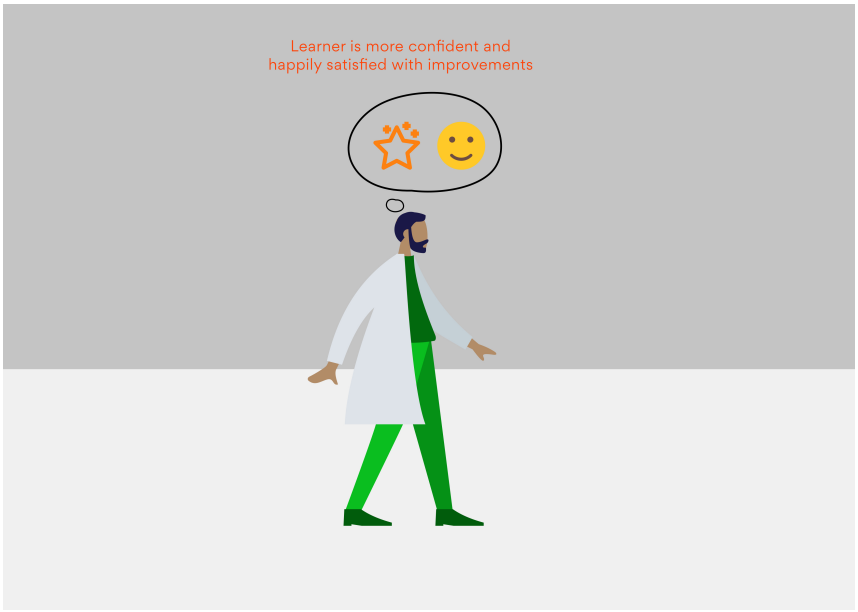

Figure 2.9: Storyboard - Satisfied and more confident user

The next section explains the process of how the design exploration was conducted by looking into literature from other relevant domains and brainstorming ideas.

2.2 Design exploration

Since potential shortcomings were realised, it was necessary to address them with proper methods. The methods used for the corresponding first two stages of the UX design life-cycle were:

1. Literature research in the field of data visualization and feedback mechanisms for complex tasks.

2. Sketching ideas for a virtual reality system to create detailed concepts.

2.2.1 Literature research

In regards to concurrent feedback, researches have shown that using multi-modal feedback for complex tasks could be beneficial [4]. However, previous implementations talk

about the benefits of these feedback in relation to performing the task. This is not exactly the case in our system. To elaborate, the virtual reality movement training is related to patient-transfer. The main goal of the learner is to successfully transfer the patient (ex. from bed to chair). What we want as a part of this process is, that the learner should perform this task with proper kinaesthetics-based concepts. These concepts in layman's terms would mean that the learner follows certain physiological guidelines so as to avoid detrimental effects on their body. Our system, which will potentially keep track of the learner's body will inform the user of anomalies related to the four-risk metrics mentioned in the introduction [Figure 1.3]. The issue here is that the learner is already being instructed during the patient transfer in regards to the next step to be performed. This presents us with a challenge that the user's visual and auditory senses are pre-occupied with the task at hand. One potential solution to this is vibrotactile haptic feedback.

Vibrotactile feedback mechanisms have been explored in a number of different contexts, including collision avoidance in virtual reality games [20], navigation systems for pedestrians [21] and rehabilitation exercises for stroke patients [22]. However, a commonly cited benefit of vibrotactile feedback is that it is useful when other modalities, such as hearing and vision, are under cognitive load (e.g., [23]). Also, vibrotactile feedback has shown to induced faster reaction times in the learners as compared to the same instructions given verbally. System developed for complex tasks such as playing the violin [24], has shown to be effective in terms of delivering silent hints to the learner. The In-the-Wild study conducted for it concluded that vibrotactile feedback used in moderation can be really beneficial for the initial stages of learning. Moreover, one of the learnings from the ERTRAG project states that during training experts don't usually interrupt the session when the students are performing patient transfer [4]. However, they tend to provide hints to the students by asking indirect questions like, "How does your back feel?". This analogy correlates to the one mentioned earlier, that vibrotactile feedback can provide 'silent hints' to the learner.

Moving forward, we take our attention to another important aspect of the feedback system - The terminal feedback module. As a part of seminar research, it is already well established that terminal feedback plays a vital role in reflection. It helps a learner to understand their ingrained errors after the task has been completed. Most complex tasks can be benefited from a combination of concurrent and terminal feedback. Earlier we talked about how concurrent feedback should be succinct for a complex task. However, terminal feedback works well when it comes to retention [12]. Since the training task has already been performed, this type of feedback can be more verbose. And the learner now has a single point of focus - i.e. to analyse their performance. Earlier systems developed for movement training had varying types of terminal feedback. The one we adopted for our use-case was shown a video recording of the user performing the task in virtual reality. The video recording would highlight the errors with respect to the four risk-metrics our system will be tracking. This idea was one of the takeaways in

the seminar research from the YouMove system [6]. Later, it was realised that being is a 3D interactive environment like virtual reality, the system should tend to avoid two-dimensional representation. There's nothing wrong with a 2D representation of the recordings, it just doesn't leverage the improvements in the technologies like virtual reality. Thus we came up with an idea, of recording the learner's body movements in three-dimensional space. Since we will already be tracking the points of interest on the learner's body, it should be feasible to store those points as a function of time and replay them later.

With an overall approach for terminal feedback in mind, it is important to look into works that might help us to conceptualise a detailed scenario. Literature related to contextual interfaces and data visualisation provides an understanding of relevant concepts. Previous research on contextual interfaces has established that users need to interact with more information and with more interface components that can be conveniently displayed at one time on a single screen [25]. The works talk about four approaches that are categorised based on interface mechanisms used to separate and blend views. These approaches are:

- Overview+Detail - uses a spatial separation between focused and contextual views
- Zooming - uses a temporal separation
- Focus+Context - displaying the focus within the context
- Cue-based - selectively highlight or suppress items

These approaches can help us to provide necessary information to the learner based on their actions and surrounding situations.

Another work of interest which improved the conceptualisation phase was from the field of information visualisation. Work written by Tamara Munzner [26] as a part of designing visualisations provides key insights into core principles of how information visualisations could be designed. The chapter 5 of the book talks about two fundamental kinds of sensory modalities, namely, the **identity** and the **magnitude** channels [Figure 2.10]. We perceive information about *what* something is or *where* it is by means of the identity channels. In contrast, the magnitude channel tells us *how-much* of something there is. Further, the chapter presents an interesting medium to present two different types of data - table data-set and network. Although from the naming convention these two types might sound irrelevant, they implicitly define the data we need to present to the learner. Table data-set contains data that can be categorised. We have already mentioned that we need to track four-types of risk-metrics. Hence the errors representing these different risk-metrics is categorical data. And when it comes to replaying a 3D

motion capture of a learner in virtual environment - the risk metrics values shown must be visually linked with the relevant body locations. This is analogical to a network data-set. According to the literature, these data-types can be well depicted using *Marks* - for table data-set and *link-nodes* - for representing a network data-set.

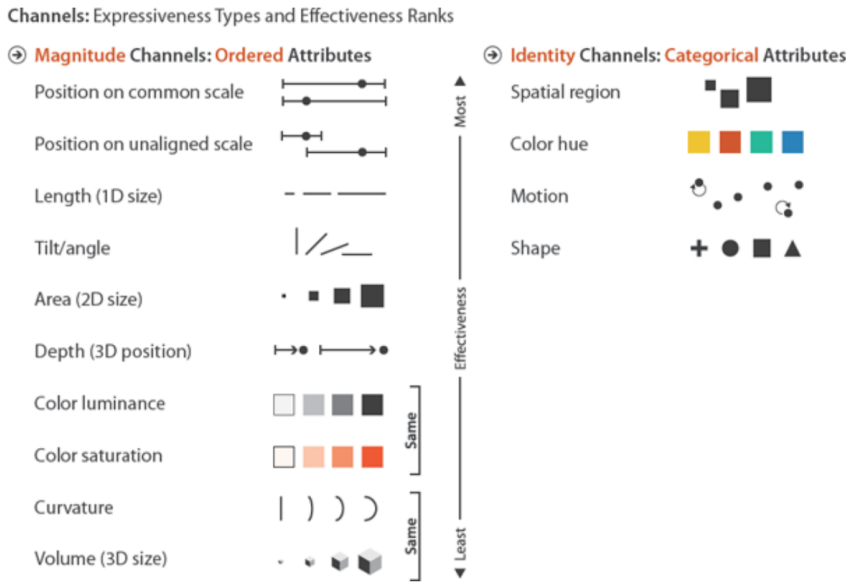

Figure 2.10: Expressiveness and effectiveness rankings of visual channels [26].

Furthermore, it is said that the most common beginner's mistake in visualisations is violating the expressiveness and effectiveness principles [26]. We took our learnings from the literature research and applied them in the sketching phase which is explained in the next sub-section.

2.2.2 Sketching ideas

There are multiple ways in which solutions can be designed. One of the common ways to visualise the ideas occurring in the brainstorming sessions is sketching. The *Sketching User Experiences* book by Bill Buxton clearly suggests approaches that are useful for putting ideas onto the paper. For the design exploration, I sought out to sketching the ideas. Although a pen and blank paper are usually sufficient, it is limited by the fact that it is difficult to generate ideas in a 3D perspective. Talbot et al. (2020) in their paper, explains why free movement and immersive 3D pose challenges to the traditional storyboarding methods. Virtual reality being a 3D interactive space, a sketch on blank paper lacks in maintaining spatial efficiency and intuitive reading of storyboard. A powerful suggested solution is being able to bind the several perspectives together to represent

a specific point in time [27]. This would help balance the three-dimensionality, spatial efficiency, and ease of creation. This would in turn also help for an easier understanding of the virtual reality scene to the reader. After consolidating these ideas, I came up with a template variation on which storyboard related to virtual reality can be sketched. The aspects taken into consideration were the field of view, comfortable head-turn zone, limits of depth perception (min. and max. distances).

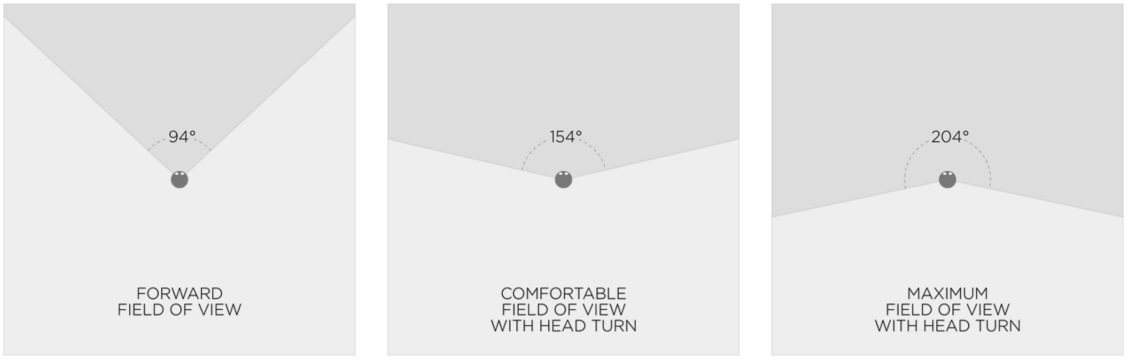

Figure 2.11: Field of view based on comfortable head rotation ranges.

Most of the commercial virtual reality headsets have a field-of-view of around 94-degrees [28]. The person wearing the headset can rotate their head comfortably to up to 30-degrees to the side. The maximum a head can be rotated without over-stressing the neck muscles is 55-degrees. The visual representation of this data is shown in figure 2.11.

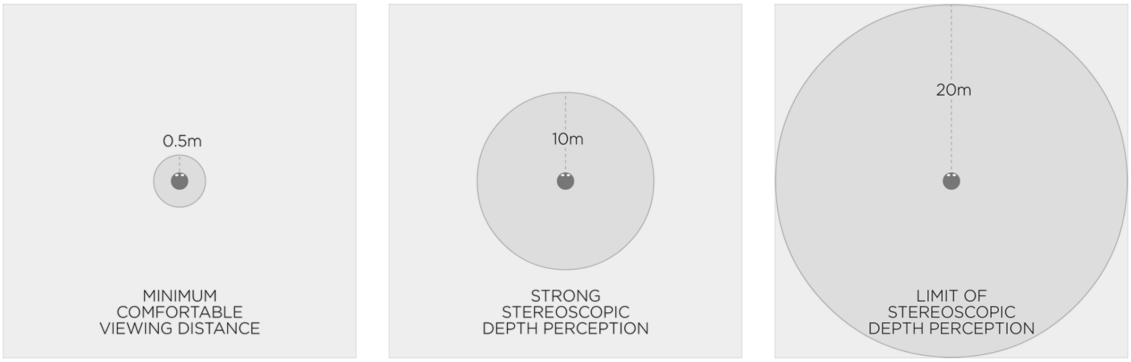

Figure 2.12: Viewing distance based on comfort and strength of stereoscopic depth perception.

Next, we look at the distance. Humans have evolved to pay more attention to objects that are closer. The minimum comfortable viewing distance in a Head-Mounted Display(HMD), before a user starts going cross-eyed, is 0.5-meters (Oculus now recommends a minimum distance of 0.75-meters [29]). Beyond 10-meters the sense of 3D stereoscopic depth perception diminishes rapidly until it is almost unnoticeable beyond

20-meters. So this gives us a sweet spot between 0.5-meters to 10.0-meters where we can place important content [Figure 2.12]. Furthermore, figure 2.13 shows a two-dimensional representation of a virtual scene by stitching several frames together. The labeling in the figure explains what each section of the 2D image represents in 3D.

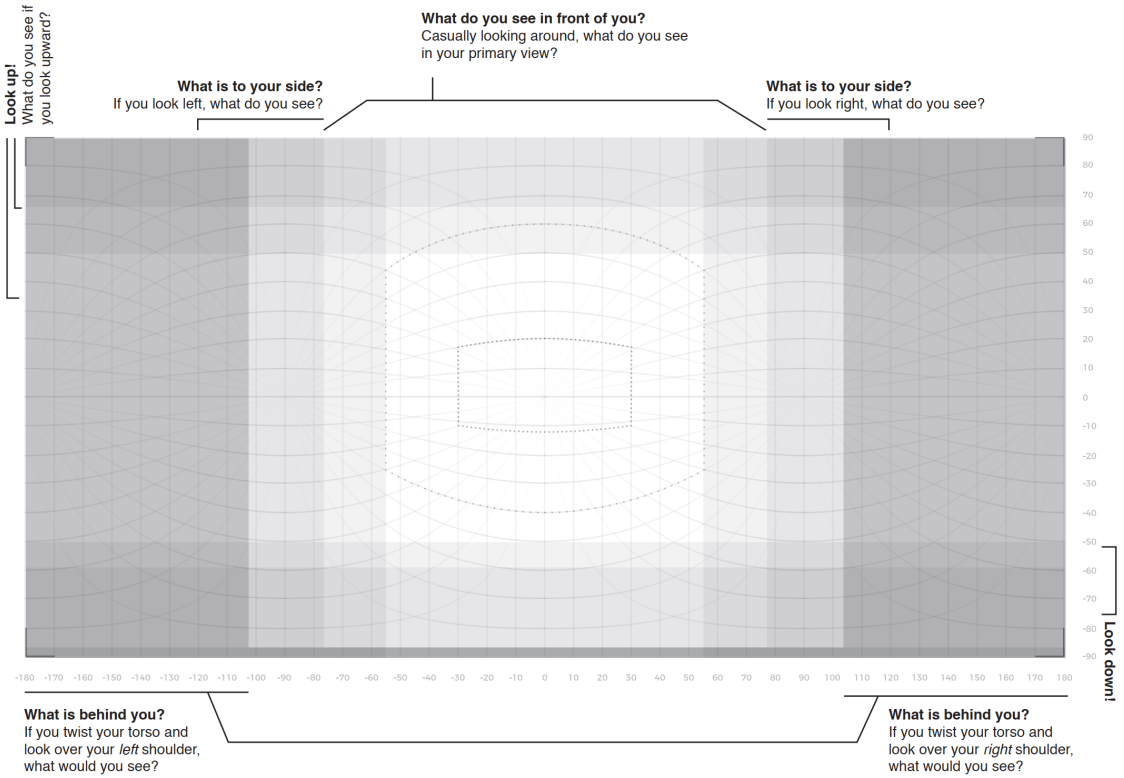

Figure 2.13: 2D representation of the field of view sketching template with labelling.

Now that each part of the template has been explained, it will be easier for you to understand the upcoming sketches. Figure 2.14 unifies all the modules of the virtual reality storyboarding template. This template provided with a necessary base that made it easier to portray the ideas before implementing them. Next, this sub-section shows some of the hand sketches made during the design phase. Some of these sketches were iterated upon and were taken forward to be then translated into a scenario. Certain aspects of the hand-drawn sketches were later highlighted on a computer to give the reader a better understanding.

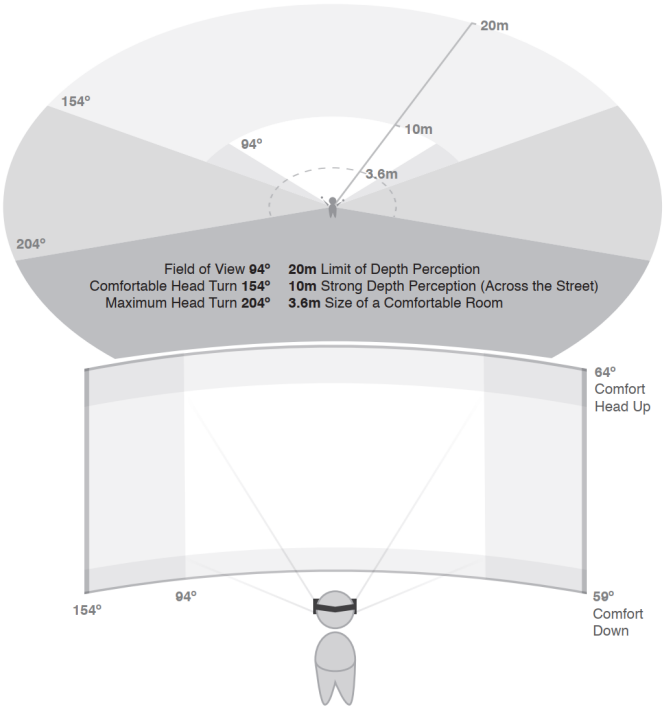

Figure 2.14: Consolidated storyboard sketching template with guidelines.

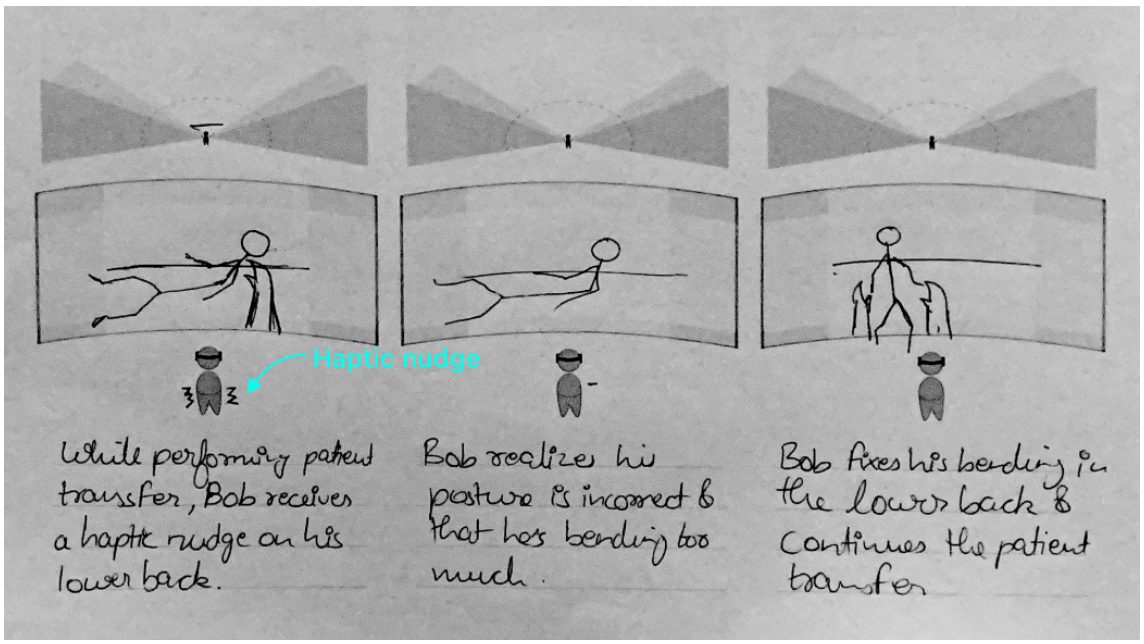

Figure 2.15: Sketched concept for terminal feedback.

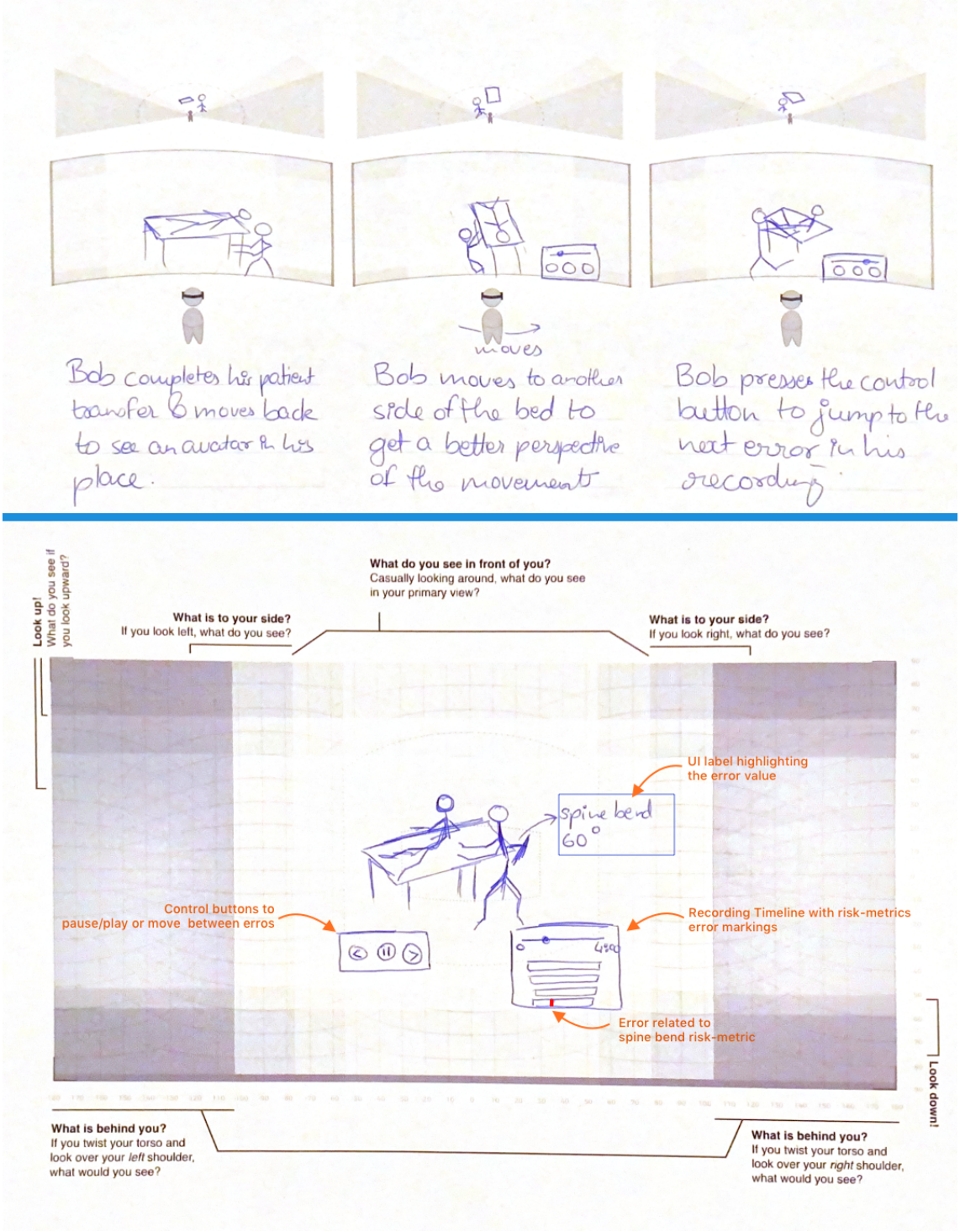

Figure 2.16: Sketched concept for terminal feedback.

Multiple drawings and concepts were created during the sketching phase. Some ideas which were translated into the final storyboard are presented here. Figure 2.15 shows how the user behaves on receiving a haptic nudge while performing the patient transfer in an erroneous way. While figure 2.16 shows a combination of frames representing the terminal feedback concept. The idea here is that, once the learner completes the virtual patient transfer, he can step back and see his own recorded movements. During this recording playback, the user has the ability to move in the virtual space and observe his recording from different perspectives. Also, during this playback, the learner is presented with two user interfaces(UIs). The UI canvas on the right is an animation timeline with spaces provided for error markings related to the four risk-metrics. While the UI canvas on the left is an animation controller that has options to play or pause the recording or traverse between the errors shown on the animation timeline.

Storyboard

The storyboard explains a scenario consisting of the target user. It also highlights the key features of the envisioned system. To provide some user-background, Bob is a medical student learning caregiving during his academic course. But, Bob is not satisfied with the only 3 days of practical training. He feels a need to practice patient transfers by himself. At medical school, he's introduced to a new virtual training system using which a person can practice patient transfers by themselves. Bob decides to give this training system a try.

1. Bob enters the room that has the virtual patient transfer training system already setup. He starts the application and wears the HMD, the gloves along with all the sensors. (Image on next page)

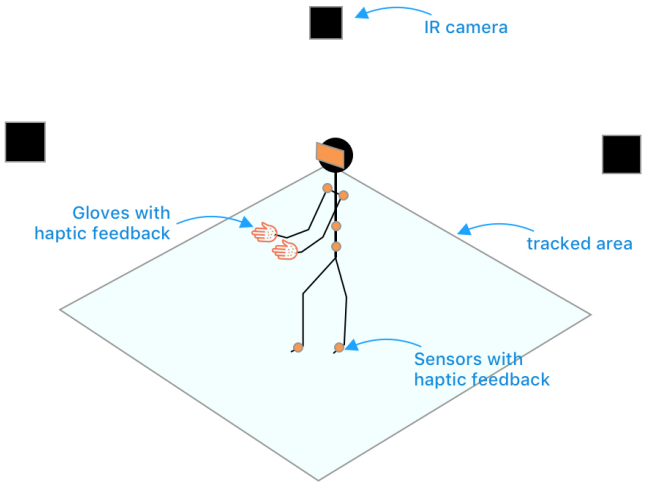

Figure 2.17: Bob starts the virtual training application and wears the hardware.

2. Bob enters the VR environment and sees a virtual patient lying on a bed. He sees some instructions and gets prepared.

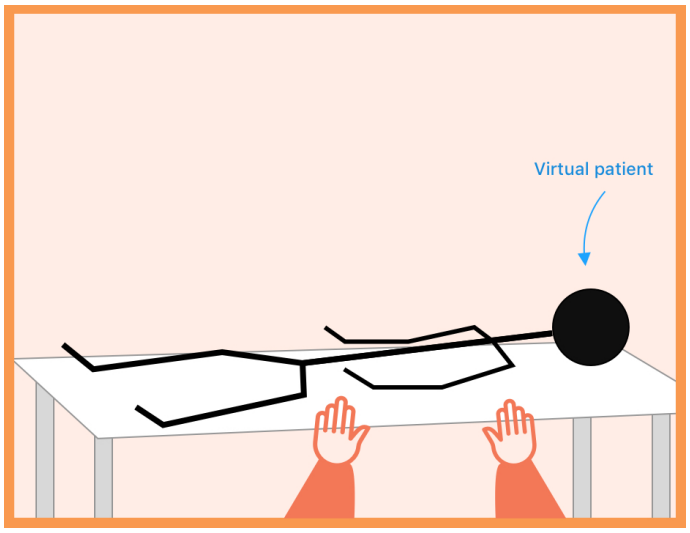

Figure 2.18: Bobs enters virtual reality and sees a virtual patient laying in the bed.

3. Bob moves towards the patient and starts following the instructions to transfer the patient from the bed to chair.

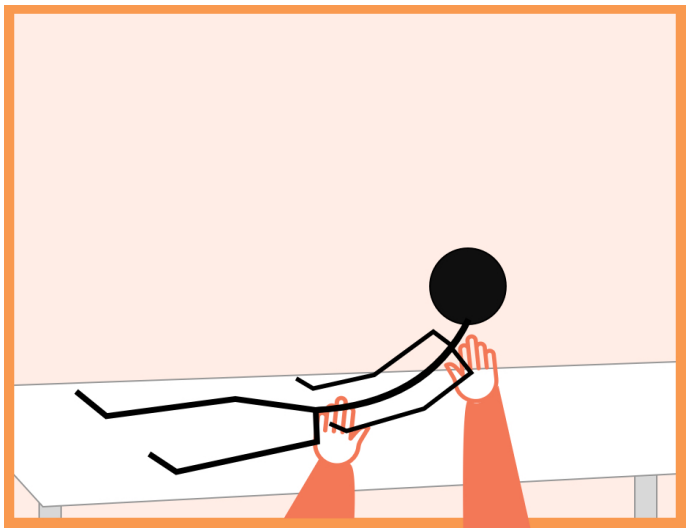

Figure 2.19: Bob moves towards the virtual patient and start performing transfer movements.

4. As Bob bends to move his hands further under the patient, he feels a haptic nudge via the sensor mounted on his lower-back. He immediately realises that the feed-

back is because of his spine bend. He fixes the error instantaneously and continues with the patient transfer.

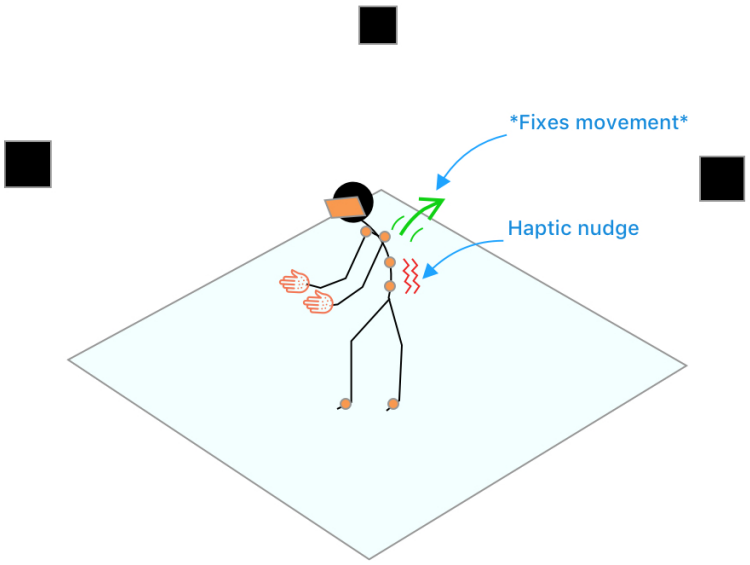

Figure 2.20: Bob receives a haptic feedback on his lower back because of too much bending.

5. Bob successfully moves the patient in a seating position and completes the transfer.

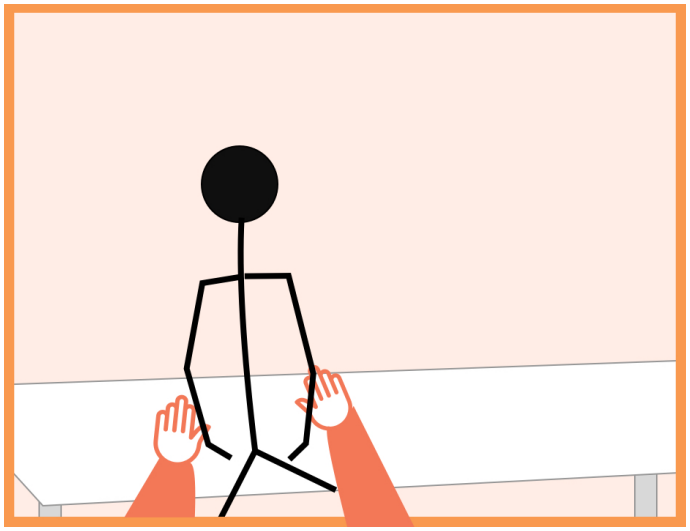

Figure 2.21: Bob successfully completes the patient transfer.

6. As Bob steps back from the virtual bed, he sees an avatar spawning at the initial location of Bob. At the same time a recording timeline UI pops up. Bob plays the animation and reaches his point of spine bend error. Here Bob pauses the 3D recording reconstruction.

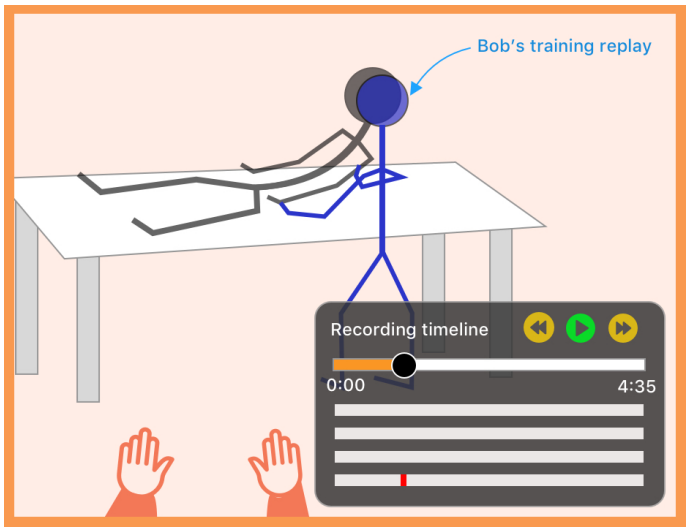

Figure 2.22: Bob starts seeing his complete patient transfer movement in virtual reality.

7. As Bob moves towards the bed again, the UI controls fade away and the system displays more contextual information. Bob can now see a tool-tip anchored to his lower back showing the error value of 60 degrees of spine bend. This was reported as a critical error by the system.

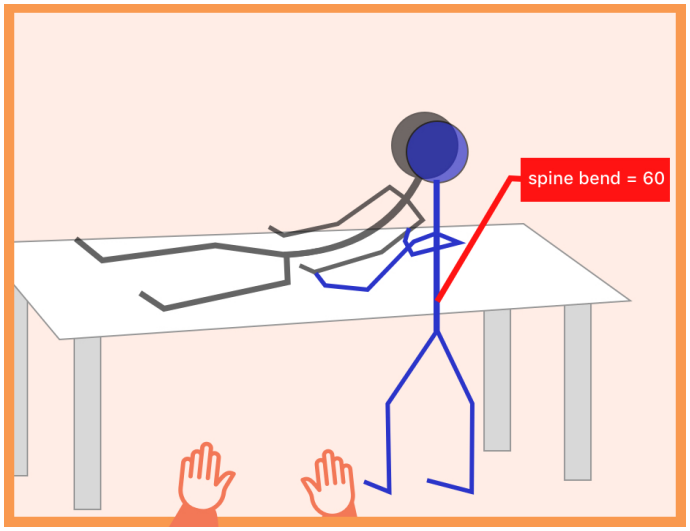

Figure 2.23: Bob starts the virtual training application and wears the hardware.

8. Bob reflects on his mistakes during the training session and now has a better understanding of applying kinaesthetics concepts in patient transfers. He feels more confident and is happy to practice by himself.

Overall, the outcome of this chapter will help us to define the hardware requirements required by our system. The storyboard also facilitates as the basis for starting the development.

3 Requirement gathering and hardware analysis

This chapter includes the hardware analysis that were performed at the start of the project development. The analysis is based on requirements that were observed as a part of the design thinking process. These requirements helped us to compare the existing technologies. And also assisted us in choosing a feasible platform for developing our system. The comparison in the later section of this chapter helps us to determine which hardware suits better to the project requirements.

3.1 Hardware requirements

As an outcome of the design thinking process, it was realised that the requirements of the system can be segregated into two main criteria:

1. Body tracking

2. Output to the user

Body tracking

Body tracking means the positional measurement of bodies in a defined space. It is required since we need to calculate values for the four risk-metrics. These values can be calculated only if, data points for the body are available to us for development. When it comes to body-tracking, there are various specifications that needs to be addressed. These requirement specifications are as follows:

- Quality of tracking
  This represents how accurate/precise the tracking system is in varying conditions.

- Calibration
  Does the system require calibration steps? If yes, how extensive is the calibration process? and is it feasible to be a part of the system?

- Motion capture
  Can the tracked data points on the learner's body be stored in a way that they can be retrieved later.

- Experience
  Does the body tracking hardware makes the training experience unnatural in any way?

- Flexibility of development
  Due to COVID-19 restrictions, it is important to understand whether the development and testing will always require a lab setting or can it be done remotely as well.

- Reliability
  If the hardware can track reliably for longer periods of running times.

- Clothing restrictions
  Does the clothing of the user affects the tracking in a negative way. Also, if there are any clothing restrictions imposed by the hardware working mechanism.

- Portability
  If the tracking system be easily carried, setup and used without hassle.

Output to the user

Output to the user covers the visualisation part. This requirement criterion covers how the risk metrics data is displayed to the user. Since we are going to develop a feedback system for a virtual reality environment, it is obvious that a head-mounted display would be necessary. Hence, this criterion mainly describes how well the tracking system could integrate with the HMD output. In our case, the HMD used will be HTC Vive or Valve Index, as it is used as a base by Daniel Schweitzer for his master project. The requirement specifications for this criterion are:

- Latency
  It covers the time delay between the instruction given by the learner to the instruction acknowledged by the system. This is necessary since there should be a minimum delay between the tracking of error related to the risk-metrics and

the user getting informed about them. Output latency significantly impacts the realism and overall experience of the system.

- Playback of recorded movement
  It covers whether the motion capture data from the body-tracking be easily translated to the user's output environment.

3.2 Existing technologies and their comparison

There are multiple body-tracking technologies that can work in conjunction with a virtual reality system. We take a look at the most relevant ones and analyse them on the basis of the requirements described in the previous section. Although there are many available tracking technologies to work with, the selection was based on the availability of the hardware in the department. Altogether, four different tracking platforms were analysed so as to make an informed decision. Two of the tracking technologies that we tested works by tracking markers/trackers on the user's body using multiple infrared-based cameras. While the other two use single infrared(IR) depth-camera data so as to generate a virtual skeleton. Both types of approaches towards body-tracking has its own advantages and disadvantages. The table 3.1 mentions the names of the body-tracking systems that were tested according to their working principles. We will dive further individually into what we learned by comparing these tracking technologies according to our requirements. The overall key factor in deciding a platform would be how many and how well we can track the data related to the four risk-metrics [13].

Depth data from single IR camera	Body markers with multiple IR cameras
Microsoft Kinect	HTC Vive
iOS device using ARKit3	Opti-track

Table 3.1: Tracking platforms based on their working principle.

The figure 3.1 provides an overview regarding which points on the body needs to be tracked. We'll analyse the tracking technologies by looking at the feasibility of keeping track of these points of interest on the learner's body.

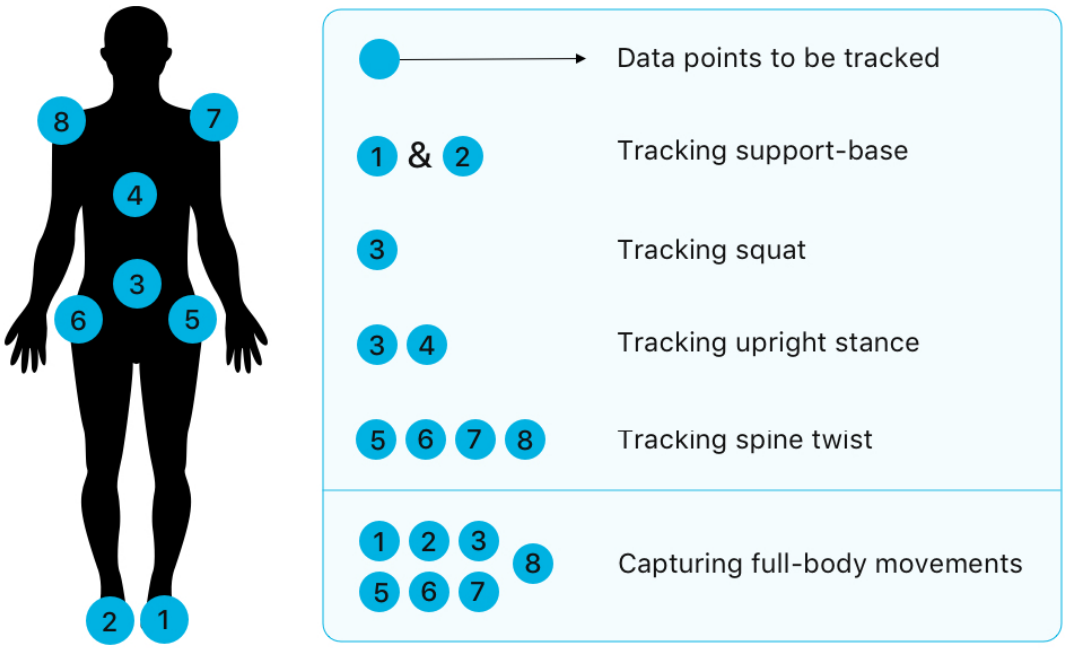

Figure 3.1: Minimum required tracked data points from the body required to analyse the errors related to four risk-metrics [13].

3.2.1 Body tracking based on depth data from single infrared camera

The two options under this category are *Microsoft Kinect sensor* and *ARKit3* on a iOS13 device by Apple. Based on their working principle, there are lots of similarities in what these platforms can offer. So I decided to first test the Kinect sensor as it has a native support on Windows and their SDK is available for Unity on Windows. This was done to see if such an approach would suffice our implementation requirements.

Microsoft Kinect

Initially the depth camera data using Unity API was tested [Figure 3.2]. After successfully able to get the Kinect SDK working with the Unity engine, I explored the the skeleton tracking data. The figure 3.3 shows the output of the skeleton tracking from different angles. The following are the analysis from an implementation perspective.

Quality of tracking (Precision)
When it comes to tracking the points of interest mentioned earlier [Figure 3.1], I tried to get the relevant data point information from the tracked skeleton. On comparing the values provided by the Kinect SDK to the real-world measurements, it was realised that

the tolerance values for the z-axis values (depth) were really high. These values works well for spatial segregation but when it comes to comparing minute changes between two data points, the system lacks the accuracy to do so.

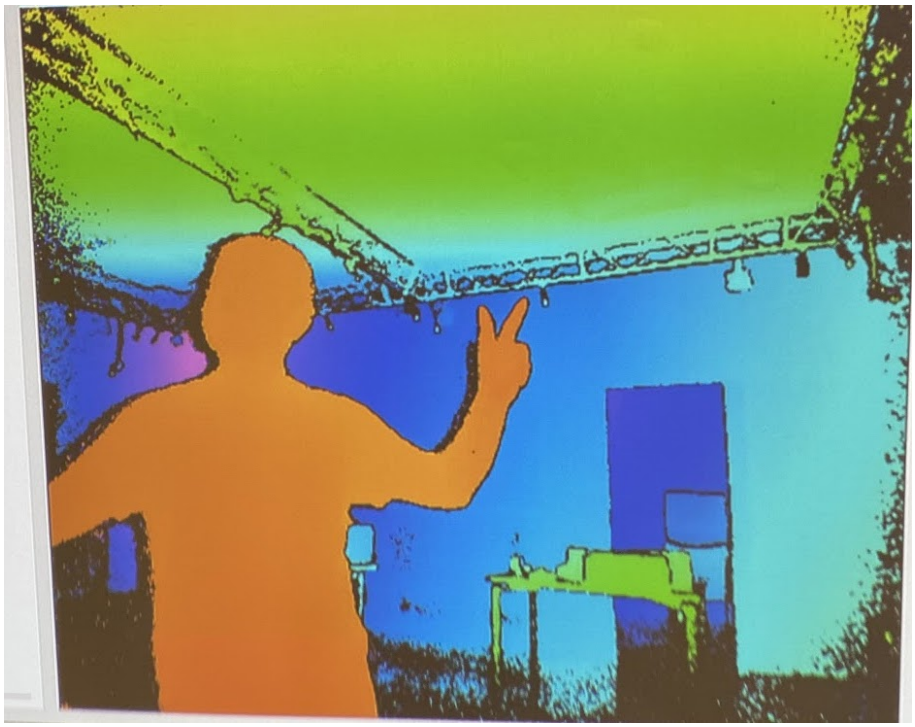

Figure 3.2: Depth data from Microsoft Kinect SDK visualised in Unity.

Latency
The Microsoft Kinect offers high resolution because of the abundance of data points it provides to the developer. However, the sensor sacrifices temporal resolution for richer spatial data. *Temporal resolution* is defined as the amount of time needed to revisit and acquire data for the exact same location [30]. Although this might be benefit for some other applications, it doesn't provide use with any advantage. Our system would prefer higher refresh rates so that the patient transfer movements are well captured in time.

Reliability
On testing further, it was observed that the tracked skeleton points sometimes jumps to a random location. This usually happened while performing squats or on rotating the body. Kinect sensors were originally made for tracking frontal skeleton of the user's body. However, when the body is rotated for for more than 70 degrees, the tracking is unreliable. Lots of false positives are detected in the space. This makes it highly susceptible to bodily movements which might also have occlusion due to overlapping of body parts. This could be sorted by introducing two Kinect cameras, one in the front

and one at the side. But, this significantly increases the implementation complexity. The Kinect SDK does not natively support body-tracking using multiple cameras. There are some workarounds possible which are mentioned over the internet, but it raises a lot of concerns with the stability of the system and later interfacing it with a virtual reality HMD environment.

Calibration
To work properly, the sensor requires a 3 step calibration process every time the application is started. This is necessary because the sensor needs to derive the peripheral points of the body for continuous tracking.

Motion capture
As mentioned earlier, the accuracy and reliability of the tracked data points is questionable. Though frontal body capture would work fine for our use-case, overall the motion capture is not that good.

Playback of movement in virtual reality
Even though the motion capture is not highly accurate, the playback of movements in a VR environment like HTC Vive or Valve Index poses some independent challenges. Once the depth-data from Kinect sensor is different from the data expected by a VR system. The co-ordinate systems of HTC Vive and Valve Index both are based on SteamVR tracking technologies. This includes an overhead for us to translate the values from the Kinect sensor to be understandable by the SteamVR coordinate system.

Tracking risk-metrics
The sensor provides a lot of data points on the skeleton. These points align with our requirements from figure 3.1. Hence, the tacking system can ideally be used for calculating values related to the four risk-metrics.

Tracking experience
The experience for the user while being tracked is natural since there are no trackers or markers mounted on the body.

Clothing restrictions
Since the skeleton tracking is based on the image generated from the depth camera, wide-pants or loose jackets are not allowed to be worn while using the system. This would generate incorrect tracking values for the the skeleton structure.

Portability
The sensor is compact and can be easily carried with oneself to change the location of training. This makes it highly portable and independent of the HMD setup.

Flexibility of development
Since the sensor can function independently without the HMD setup, it provides high flexibility for developing features outside the lab.

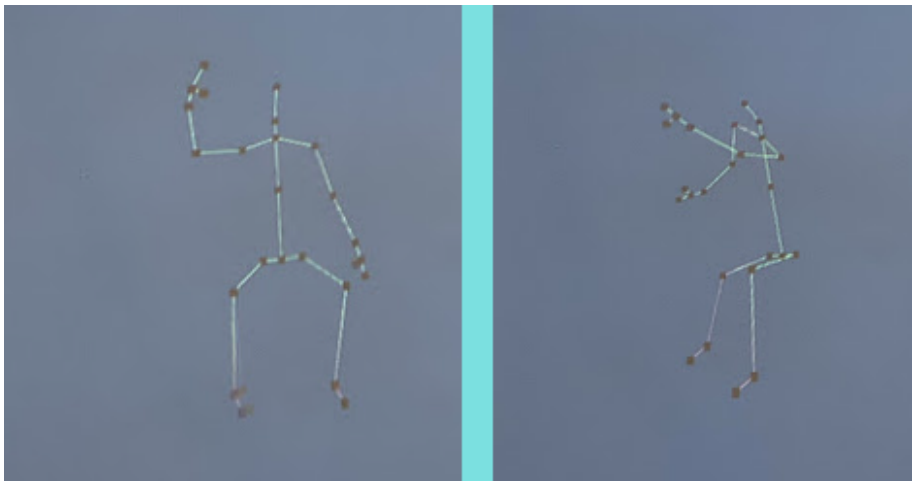

Figure 3.3: (Left to right) Skeleton data points generated from frontal body tracking, Skeleton data points generated from side body tracking.

Overall, Kinect sensors were made for a purpose of skeleton tracking in gaming. Hence extending its use-case for an application like ours imposes a lots of restrictions and would not be an ideal first choice for our implementation.

ARKit3

ARKit3 is Apple's supporting SDK for devices running iOS13 or higher. The SDK provides libraries for skeleton tracking, anchoring and many relevant features for mixed reality development. There is no native support for Windows or Unity engine platforms. The following are the analysis from an implementation perspective.

Quality of tracking (Precision)
The quality of skeleton tracking is better than Microsoft Kinect sensor. The tolerance values seems acceptable when comparing data points in space. The z-axis depth values were close to the real-world measurements.

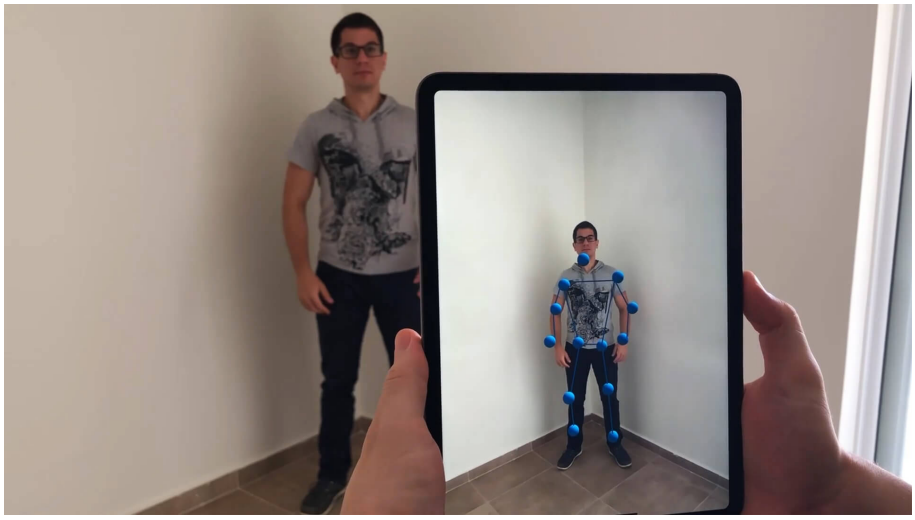

Figure 3.4: Skeleton tracking using ARKit3 on an iPad Pro.

Latency
Since the ARKit3 is only available for iOS devices, it interaction with Unity application for virtual reality is limited. The implementation would require a network interface layer to communicate and exchange data with the Unity application running on a Windows PC. This introduced high latency in data points being tracked and reflected over in the Unity's user output

Reliability
Surprisingly, ARKit3 handles occlusion pretty well. Even when the body was turned for more than 75 degrees, the data points being tracked didn't generate false positives. The observed improvements were more on the iPad Pro since it uses a dedicated LiDAR sensor to acquire depth values from the environment.

Calibration
Auto-calibrates the scene withing seconds of detecting a floor type base entity.

Motion capture
Capturing skeleton co-ordinates in real-time and applying them to an avatar is natively supported by Apple's SDK. The code I tested was a sample provided by Apple developers from the WorldWide Developers Conference(WWDC) 2019 [31].

Playback of movement in virtual reality
Though the motion capture is highly accurate, the playback of movements in a VR environment like HTC Vive or Valve Index poses some independent challenges. The depth-data from ARKit3 is different from the data expected by a VR system. The co-ordinate

systems of HTC Vive and Valve Index both are based on SteamVR tracking technologies. This includes an overhead for us to translate the values from the Apple's proprietary coordinate system to be understandable by the SteamVR coordinate system.

Tracking risk-metrics
The ARKit3 SDK provides information regarding all the data points necessary for implementing tracking of the four-risk-metrics.

Tracking experience
The experience for the user while being tracked is natural since there are no trackers or markers mounted on the body.

Clothing restrictions
Since the skeleton tracking is based on the image generated from the depth camera, wide-pants or loose jackets are not allowed to be worn while using the system. This would generate incorrect tracking values for the the skeleton structure.

Portability
Devices running iOS can be easily carried with oneself to change the location of training. This makes it highly portable and independent of the HMD setup.

Flexibility of development
Since the ARKit3 implementation can function independently without the HMD setup, it provides high flexibility for developing features outside the lab. The initial testing of the hardware was done in my personal space due to the COVID-19 restrictions.

3.2.2 Body tracking based on tracking markers on body with multiple infrared cameras

The two options under this category are *Optitrack* and *HTC Vive trackers*. There are similarities in what these platforms can offer. Since multiple cameras are used in these types of settings, the capital cost (cost of the setup) is high. But, this leads to a significant improvements in certain aspects of our requirements. Lets take a detailed look into these systems.

Optitrack

Optitrack system is a 3D tracking system developed for use-cases such as: virtual reality, robotics, animation, and movement sciences. Stefan Feyer performed a detailed analysis of this tracking system as a part of his masters project. The learnings from

the analysis were studied and used to determine whether this system is suitable for our implementation.

Quality of tracking (Precision)
As the system is originally meant for motion capture, it is good for short instances. Because on tracking body for longer times, the system was prone to noise. The noise generated overtime led to cascading of errors in the tracked data. Hence, precision is highly dependent on the time required by the task to be performed.

Latency
The latency is low and the performance of the system is snappy. There were no delays, but the system processing power required is quite high.

Reliability
As mentioned earlier, the reliability is a major concern. Since error cascading can lead to inaccurate data points over time.

Calibration
Multiple steps are required by the user to calibrate the system with the body suits markers.

Motion capture
Motion capture is difficult with some workarounds in place.

Playback of movement in virtual reality
Playback of movements in the native Optitrack coordinate system works the best. However,translation to SteamVR coordinate system would still be required which raises concerns.

Tracking risk-metrics
The system provides all the body points required by us for calculation of the four risk-metrics. However, reliability is a concern because of the introduction of noise.

Tracking experience
The body movements feels unnatural while being tracked. This is because the user is required to wear a dedicated suit during the total duration of tracking [Figure 3.5].

Clothing restrictions
It requires the user to wear a suit which has multiple markers mounted on it, so as to effectively track a person's body [Figure 3.5].

Portability
The system has a dedicated industrial mounting setup which requires bolting the cameras to a chassis mounted on the ceiling. This makes the tracking setup immovable.

Flexibility of development
The tracking system is mounted by professionals in the lab. This significantly restricts the implementation possibilities during COVID-19 restrictions.

Figure 3.5: Body suit required by the Optitrack system to track and capture real-world movements.

Considering the limitations such as reliability, precision and flexibility of development, the Optitrack system does not provide a decent foundation for our implementation. Also, considering the timeline of development, the development being restricted only to the lab setting is a major concern.

HTC Vive

The HTC Vive tracking is based on SteamVR tracking technologies [32]. It has three main components: base stations, sensors on tracked objects, and a host.

Quality of tracking (Precision)
The tracking precision was really good and had a tolerances of +-1cm for distances and +-2 degrees for angle calculations. Figure 3.6 shows how the test was performed. The top part of the image consists of 2 Vive trackers in a real-world setting. They were

kept at a distance of 39cm in a lab environment. The bottom part of the figure shows 2 cyan colored spheres in a virtual reality environment (screenshot taken from Unity editor). These two spheres are attached to the Vive tracker movements using the SDK provided by SteamVR. Hence the relative distance between the spheres provided us with the precision of the tracking system. The distance of the two blue spheres was shown to be 0.40m in the virtual environment. This test was performed multiple times and the average tolerance value for distance between tracked points was found out to be +-1cm.

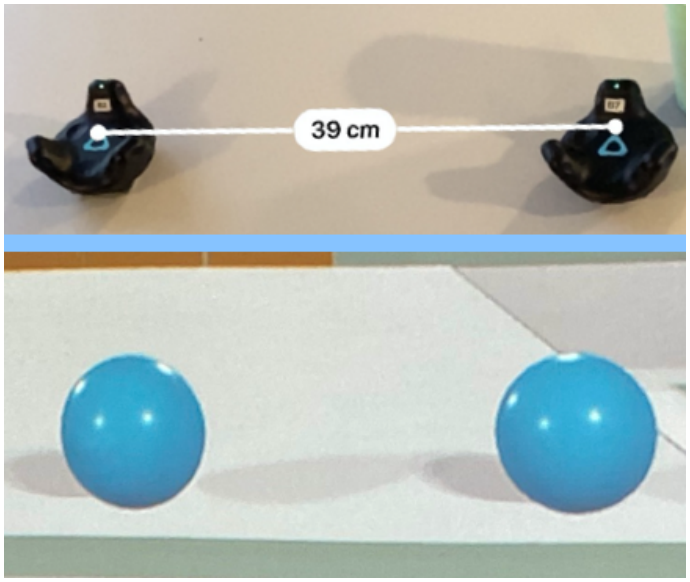

Figure 3.6: Testing the accuracy of distance between the trackers in real world (up) to the tracked objects in virtual world (blue spheres).

Latency
The HTC Vive trackers [Figure 3.7] have a higher capture rate of 200Hz/device/second which results in really low delays [32]. During the testing latency was never an issue with Vive trackers.

Reliability
A minimum of 2 base stations are required to track the sensors in HTC Vive environment. The reliability of tracking increases with the increase in the number of base stations. For our testing, a setup containing 4 base stations was used. The tracking was not lost across multiple testing process and was resistant to occlusion. This is also because of the fact that each base station contains a 120° multi-axis laser emitter.

Figure 3.7: The palm-sized HTC Vive trackers.

Calibration
Easier one-time calibration of the system which retains the data and can be used multiple times until the location of setting is changed.

Motion capture
HTC Vive tracking environment facilitates maximum tracking of 9 data points. The motion capture of these points is supported by the VR environment and can also be translated to humanoid movements.

Playback of movement in virtual reality
Since the Vive trackers are native tracking solution of HTC Vive, the playback of movements does not require any coordinate system translation. This removes the necessity of writing an interfacing driver for the coordinate systems to function together.

Tracking risk-metrics
We need 8 tracked data points on the learner's body to extract the risk metrics [Figure 3.1]. Since Vive tracking environment can support up to 9 trackers, we should be successfully be able to track the risk-metrics.

Tracking experience
Although Vive trackers are palm-sized, mounting 8 trackers on the body will make the bodily movements feel unnatural.

Clothing restrictions
There are no clothing restrictions because Vive trackers can be mounted on the body over the learner's clothing.

Portability
The system is not highly portable but still can be moved from one place to another. The base stations can be easily removed and can be packed together to be taken to another location.

Flexibility of development
There are frameworks available such as Virtual Reality Toolkit(VRTK), which can allow standalone development of modules without requiring the constant access to a virtual reality HMD.

The table 3.2 summarizes the comparison of all the tracking technologies that we discussed so far. The table helps to gain a holistic view of the pros and cons of each tracking methodology.

As an overall conclusion, the HTC Vive trackers were found to be a good alternative to take the development of the feedback system ahead. On discussing the details of with my supervisor Maximilian Dürr, we agreed that although the Vive tracker system is less portable the pros of the platform provides a better trade-off than the rest available options. The virtual patient transfer task implemented by Daniel Schweitzer as a part of his master project is also developed using SteamVR. It would be later be feasible to integrate the feedback system into it and extend its functionalities. Also, initially focus of the system development should be on the implementation of the core features. The first core feature being providing concurrent feedback to the learner using real-time risk-metrics calculations. Another core feature being delivering terminal feedback where the learner can retrace his movements in a 3D interactive virtual environment. HTC Vive tracking system helps to focus on these core features, since the coordinate system doesn't require any translation/conversion. In the next chapter, we discuss the implementation details of the development using HTC Vive trackers and Valve Index HMD.

	HTC Vive	Microsoft Kinect	ARKit3	Opti-track
Quality of Tracking	High precision	Lower but acceptable	Medium precision	Prone to noise
Latency	Low	Medium	High	Low
Reliability	High	Low	Medium	Low
Calibration	Easier to calibrate	Multiple steps required	Easier to calibrate	Multiple steps required
Motion Capture	5 Vive trackers covers whole body movement	Good for the frontal body. Loses lower body tracking on occlusion and rotation	Decent for frontal body. Side view tracking is better than Kinect.	No, only some workarounds
Playback of movements	Feasible (native HTC)	Complicated - requires translation	Complicated - requires translation	Quite decent natively. Difficult with Vive or Valve HMDs
Tracking risk metrics	High accuracy with less tolerance	Not-feasible	Feasible but restricted	Unreliable because of noise
Tracking experience	Unnatural (Weight of Vive trackers)	Natural	Natural	Slightly unnatural
Clothing restrictions	None	No wide pants or loose jackets	No wide pants or loose jackets	Suit required with attached markers
Portability	Less	More	More	Less
Flexibility of development	Less	More	More	Less

Table 3.2: Comparison of existing technologies from implementation perspective.

4 Prototyping

On successfully choosing a body-tracking approach for the development, I went ahead with prototyping the feedback system for the storyboard scenario. This chapter has two sections, the first section provides an overview of the system which will help to gain an overall picture of the various modules. The second section goes into the details of the technicalities of each development iteration.

4.1 Overview of the system

A holistic view of the system is provided from two perspectives. Firstly, the figure 4.1 shows the overall setup necessary for the system to work as expected. The physical setup of the system requires a 2 meters by 1.5 meters of free space (6.5ft x 5ft), and the maximum distance between base stations of 5 meters (16ft) [32]. These are the official guidelines provide for the SteamVR tracking which is used by HTC Vive trackers. For our application, four base stations are recommended for reliable tracking through-out the virtual patient transfer scenario. Although the system would also work with two base stations placed diagonally. During testing with two base stations, there were cases when the tracker location was detected incorrectly. The learner/user is expected to be in the area of tracking at all times. During the virtual kinaesthetics-based patient transfer training, the learner is expected to wear 6 HTC Vive trackers [Figure 3.7] on their body while wearing the Valve Index HMD. The Vive trackers are held onto the learner's body by means of *TrackBelt* and *TrackStraps* mounts [Figure 4.2]. These mounts are easily available on a commercial website like *Amazon*. Apart from this, the learner is expected to hold the Index controllers in their hands(these will be replaced by ManusVR gloves in future task integration).

Secondly, the figure 4.3 shows the 4 most important modules of the feedback system. Each module is represented by a block. These blocks contain classes and storage files relevant to the correct functioning of each of the modules. As we have already discussed in the introduction, the main goal of the system is to deliver concurrent and terminal feedback. Hence, these two modules take most of the space and time in terms of complexity. Although these two are standalone implementations, most of the communica-

tion between the *concurrent feedback module* and *terminal feedback module* happens via the manager and database modules. The manager module is also responsible for the synchronisation of timing between the two main modules.

The technical details of the functions of the modules and what each class inside these modules do is covered in the next section. Along with the technicalities, the upcoming section also presents explanations of how certain challenges were solved.

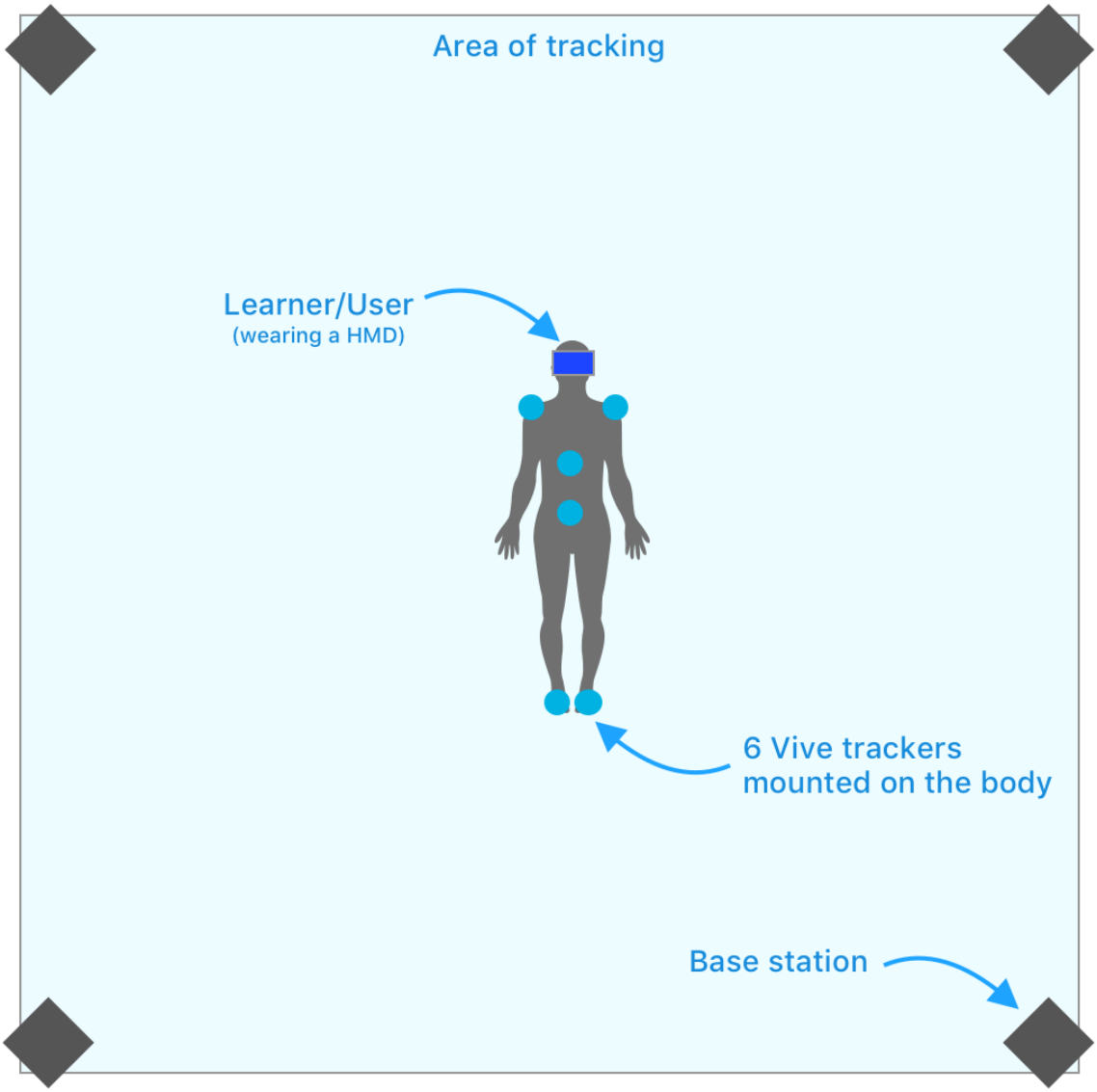

Figure 4.1: Physical setup for the VR feedback delivery system.

Figure 4.2: Belt and straps for mounting the Vive trackers on body.

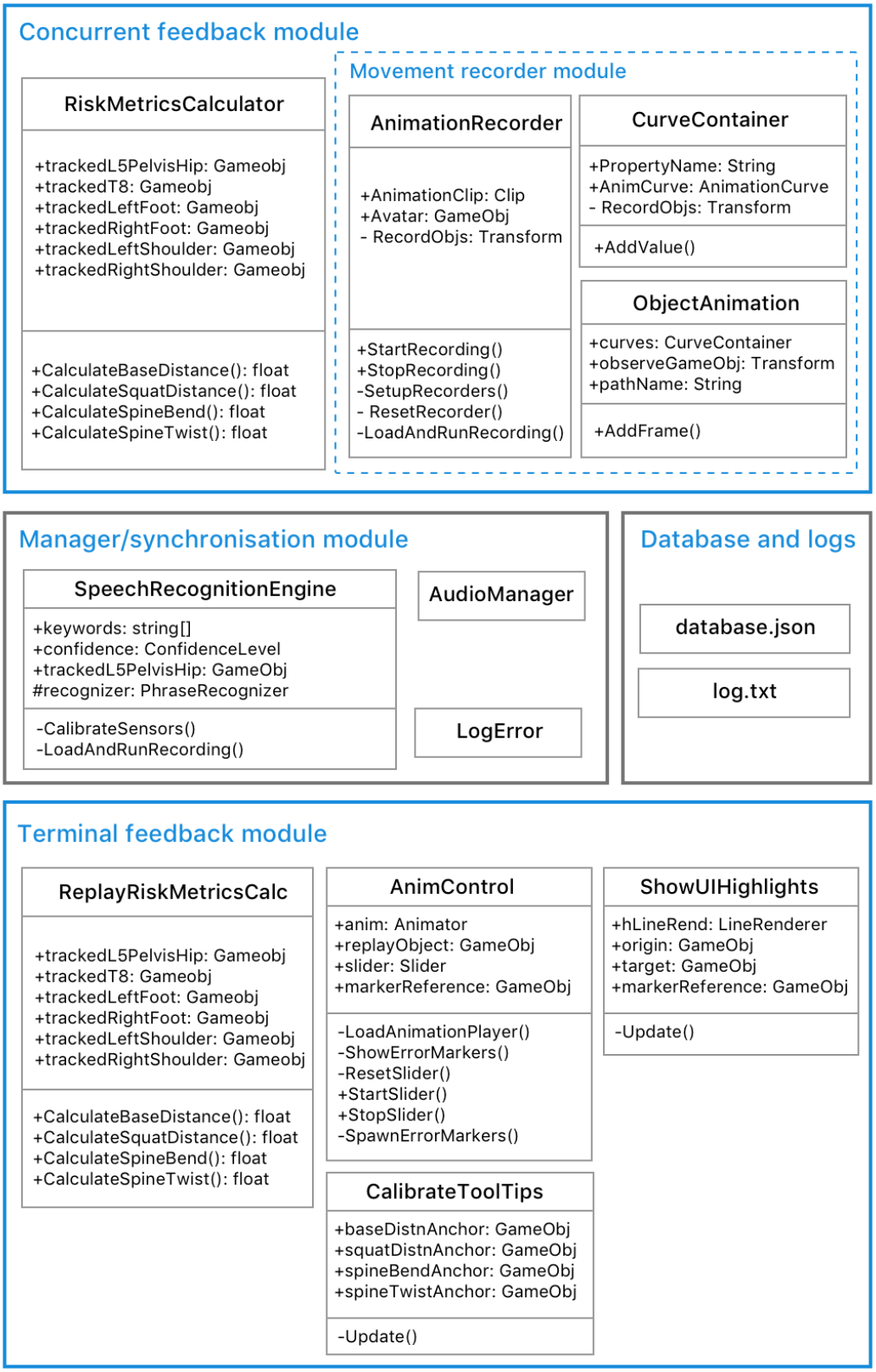

Figure 4.3: Technical system overview.

4.2 Iterative implementation details

We now dive into the technicalities and implementation details of how the development of this system was achieved. To make the content of this section easier to understand we would talk about it according to the modules [Figure 4.3]. The modules would include an explanation of what all it contains and how it was subsequently developed. Some of these iterations would also include challenges that were faced and how they were overcome. Apart from addressing the challenges, there were certain tests that were performed regularly to test the system. These will also be included in the subsections under with relevant headings. Let's start with the first module.

4.2.1 Concurrent feedback module

The main objectives of this module are:

- Track & store movement data
- Analyse the user movements according to the four risk-metrics

These objectives are to be performed synchronously in real-time. In chapter 3, we already analysed the HTC Vive trackers for its precision. However before moving forward with risk-metrics calculations, testing was needed so as to see if the trackers can be reliably used for angle measurements. Angle measurements are required for analysing the two risk-metrics namely - spine bend and spine twist.

Unit testing - Measuring accuracy of angle measurements by Vive trackers

Figure 4.4 shows the setup used for comparing real-world measurements to the measurements obtained in the VR environment. The top part of the figure shows two Vive trackers in the real-world which are linked to the two blue spheres in the virtual reality(VR) environment. The two pink spheres are stationary game-objects in the VR environment. An invisible vector is drawn between the two pairs of the spheres. Then a Unity math function is used to calculate the angle between the joining vectors. On analysing the calculations, the values were found to be close to the real-world measurements. The maximum tolerance found was of +-2 degrees while comparing the geometric angle calculation.

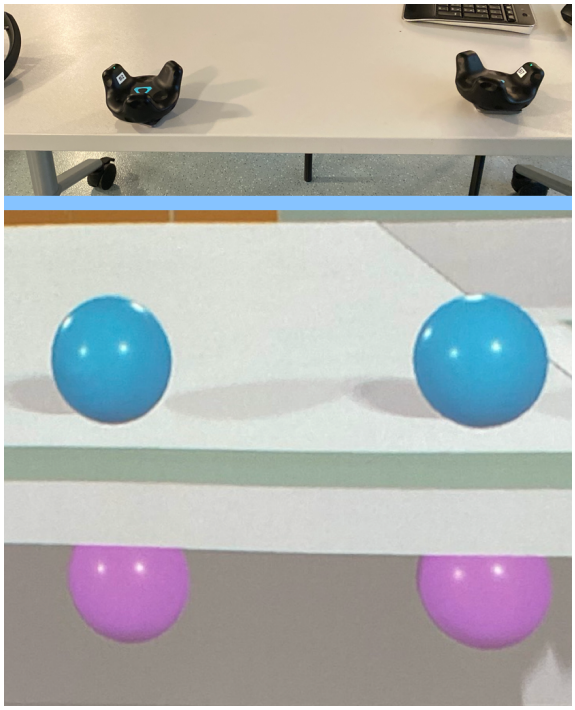

Figure 4.4: Measuring accuracy for angle measurements.

Implementation - Risk metrics calculator

Now that we know that the values related to the four risk-metrics can be accurately extracted from the real-world, we go ahead with the implementation of the risk metrics calculation functions. For tracking the points of interest from the learner's body, Vive trackers are to be mounted. It was crucial to limit the number of Vive trackers on the body so as to reduce the unnatural feeling. After looking into literature related to human physiology, it was realised that the locations of the L5 lumbar spine vertebrae and pelvis are very close to each other [Figure 4.5]. Also, it was come to attention that an another name for pelvic bones is hip bones. This literature review helped us to reduce the number of Vive trackers required for tracking the risk-metrics. To clarify, according to Muckell et al. (2017) the four risk metrics are tracked as follows:

1. Base distance - The distance between the two feet.

2. Squat distance - The distance of *pelvis* from the floor.

3. Spine bend - The angle of bend between the vertical axis and the line joining the L5-T8 vertebrae.

4. Spine twist - The angle between the line joining the shoulders and the line joining the *hips*.

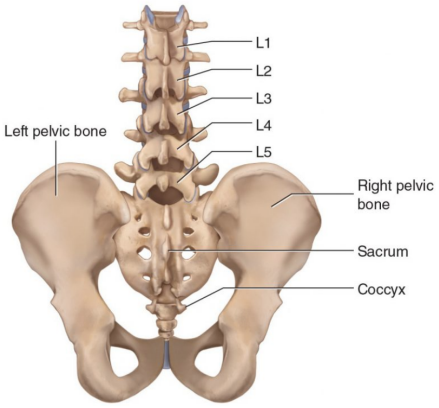

Figure 4.5: Human skeletal structure showing the close proximity of L5 vertebra and pelvic bones.

Now it can be clearly seen that the second, third, and fourth risk metrics have one tracking point in common. In figure 4.6, the tracker number 3 can provide data relevant for 3 risk-metrics. This was a substantial improvement in design, as we reduced the redundancy and chances of interference amongst the data points. The hip vector points are represented by dotted circles in the aforementioned image. These points are extrapolated using the values from tracker number 3 during the calibration step.

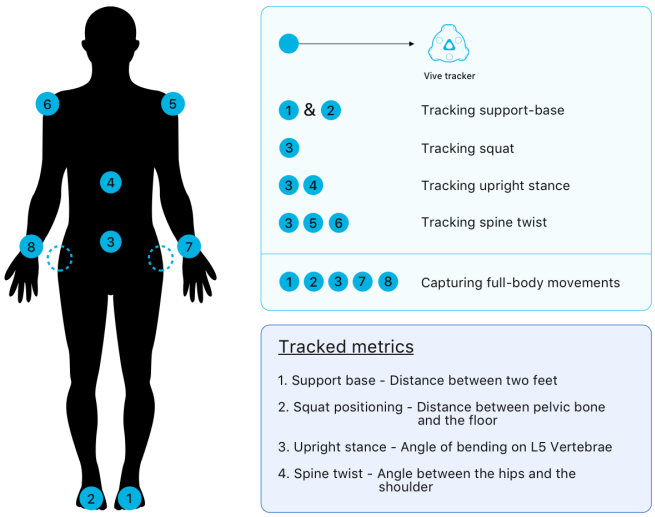

Figure 4.6: Updated position of the Vive trackers for risk metrics calculations.

Since we have the data points - the basis of calculations, we can look into the algorithms related to these risk metrics. The calculation for the first two risk-metrics are quite straight-forward. The base distance (first risk metric) simply applies the distance function to the tracked gameobjects linked to the feet of the user. *GameObjects* is a base class of Unity platform and all the entities inside Unity are of this type. From here on wards, you might encounter the word *'gameobject'* referring to as a type element in VR. The implementation looks similar to the testing process we showed in the HTC Vive technological analysis [Figure 3.6]. The second risk metric calculations are same as the first one. Instead of the calculating the distance between two gameobjects, the distance of the tracked pelvis tracker is calculated from the ground(x,0,z).

When it comes to the third and fourth risk metrics, calculations get a bit tricky. The third risk metric is for the upright stance. According to the paper by Muckell et al. (2017) - *"Upright stance metric is the angle of the lower back (L5-T8 vertebrate) compared to a perfect upright position"*. The paper provides an implementation explanation of how this angle calculation can be achieved. Figure 4.7 shows the process of how to calculate the spine bend angle. Tracker number 3 and 4 in figure 4.7 represent the location of L5 vertebra and T8 vertebra of the user respectively. The line joining these two tracked points is represented by 'a' on the triangle. We need to extrapolate a third point in the upward direction(in world coordinate space). The line joining the extrapolated point and tracked L5 vertebra is represented by 'b'. On joining all these three points, we get a triangle. All the sides of this triangle can be calculated using a simple distance formula. Furthermore, on applying the Law of Cosines on this triangle, we can find the angle 'C' which is the required value for spine bending.

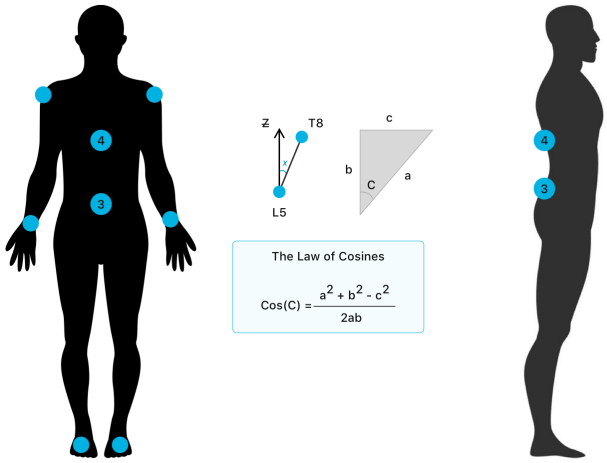

Figure 4.7: Visual explanation of the calculation related to the third risk metric (Upright stance).

For the fourth and the final risk metric, we need to calculate the degree of spine twist. Figure 4.8, visually describes how the spine twist calculation algorithm works. In the aforementioned figure, the blue spheres represent the gameobjects linked to the shoulder trackers and the pink dotted spheres are the tracked hip sensors which are extrapolated using the pelvis sensor location. The dotted line joining the same colored pairs of spheres are the vectors joining them. On translating these vectors along the Y-axis towards each other, there will come a point when they will overlap. During that instance, the angle between these two vectors is calculated so as to get the amount of twist present in the spine of the learner.

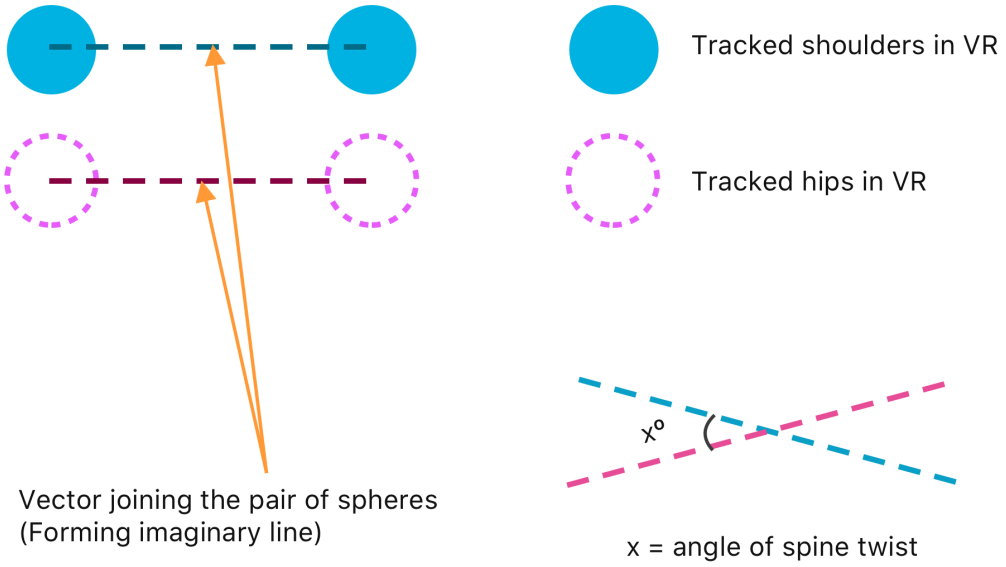

Figure 4.8: Visual explanation of how spine twist risk metric is calculated.

Unit testing - Risk metrics calculator

While testing the algorithm for the squat distance calculation (second risk metric), it was realised that the squat distance comparison values could be differ from person to person. For example, a learner who is shorter will have a short pelvis distance from the floor and might have a greater value detected by the system while performing a normal squat(in magnitude). This type of an edge-case would trigger false error feedback. However, this might not be the case for a tall learner. The literature reference related to the calculation of the four risk-metrics didn't contain an explanation for this [13]. It simply had one value to be compared for the squatting distance for all the cases [Figure 4.9]. Also, the code base of the study in a public repository on GitHub didn't provide any hints to how it was considered. So I wrote an email to the author of the research paper concerning

the issue. To which the author replied that the squat values are calibrated for each of the user initially by making them squat properly [Figure 4.10]. This introduced an additional step during the calibration process into our system. The goal of this step is to get a range of pelvis distance from the ground where the learner is properly squatting down. These values are stored and used as comparison basis for delivering error while the virtual patient transfer task is being performed.

Table 3: Totally Dependent

Metric	To / From	DCWs	Gold	Diff
Support Base	Bed-to-Chair	0.38m	0.66m	-0.27m
Support Base	Chair-to-Chair	0.36m	0.76m	-0.40m
Squat	Bed-to-Chair	0.85m	0.69m	-0.16m
Squat	Chair-to-Chair	0.85m	0.76m	-0.09m
Upright Stance	Bed-to-Chair	56.7°	43.70°	-13.03°
Upright Stance	Chair-to-Chair	49.28°	49.92°	+0.63°
Spine Twist	Bed-to-Chair	40.67°	16.51°	-24.16°
Spine Twist	Chair-to-Chair	33.17°	24.16°	-9.01°

Figure 4.9: Comparison values for the four risk-metrics [13].

RE: Query related to your research on patient transfer study from 2017

Friday, August 21, 2020 15:04 CEST

Muckell, Jonathan jmuckell@albany.edu

To

Tanveer Singh Mahendra

Hi Tanveer,

Thanks for your interest.

That should be calibrated by having the person squat properly and determining the height from the ground that occurs during a proper squat. The version code currently on GitHub doesn't include that parameter, but I'd like to include it in the next update.

Take Care,
-Jon

Jonathan Muckell, Ph.D.
Professor of Practice
IEEE Student Branch Counselor
Dept. of Electrical & Computer Engineering
Engineering & Applied Sciences
Phone: (518) 442-3167
Office: LI-69E

Figure 4.10: Email response from the author Muckell Jonathan.

Implementation - Feedback delivery to the learner

Our system now successfully tracks the risk metrics and can check the posture values for any error. Next, we need to implement an algorithm that answers *"What would happen if an error in risk metrics is detected?"*. Basically, we need a feedback delivery mechanism that can reflect the erroneous state in the output of the HMD. As per the literature research and storyboard in chapter 2, the ideal way to go about it is to provide a haptic nudge to the user at the body part associated with the error. This would act as a subtle hint and would be analogous to how the experts train the students during the 3-day workshop in academia. The teaching experts usually don't interrupt the patient transfer movement but rather simply asks the student question related to how a particular part of their body feels. For example, if the upright stance of the learner is incorrect (i.e. error related to spine bend), the teacher would ask, *"How does your lower back feel?"*. The haptic nudge could provide feedback to the students(learners) in a familiar way that they are used to.

However, there were some technical challenges that we faced. Vive trackers do not contain a haptic feedback motor. Carla Gröschel, as a part of her masters project used some sensors with integrated haptic motors while conducting a study with the nurses. These sensors could act as a workaround. But there were some major concerns associated with it. First, the motor inside isn't the sensors were small and hence would need to be placed very close to the skin of the learner. This would require to modify the mounting mechanisms of the Vive tracker. Second, the sensors with integrated haptics were made to order and would require additional time to come. Also, the delivery period was unsure because of the COVID-19 restrictions. To avoid any huge delays in the project development timeline, I decided to look at an alternative feedback delivery modality.

This took things back to sketching and I went through another iteration of stage 2 of the UX design lifecycle [Figure 2.1]. According to the literature, a combination of audio-visual feedback modalities is considered a decent replacement for haptics [33]. For complex tasks like ours, the audio cue helps to gain the attention of the learner since their audio-visual senses are already engaged. While the visual cue provides the necessary system message to the learner. Although, not perfect but this sounded like a feasible direction to move ahead with.

Figure 4.11 shows the process of the sketching phase. The idea is to provide an audio cue to the learner if any risk metric error value(s) exceeds a certain threshold. Along with this cue, a window pops up in the top left view of the HMD. This view highlights the part of the body which is affected(in the left half). While in the right half of the window the animation is shown as to how the error in posture can be fixed. Figure 4.12 provides a sketch of a detailed first person-view from the HMD.

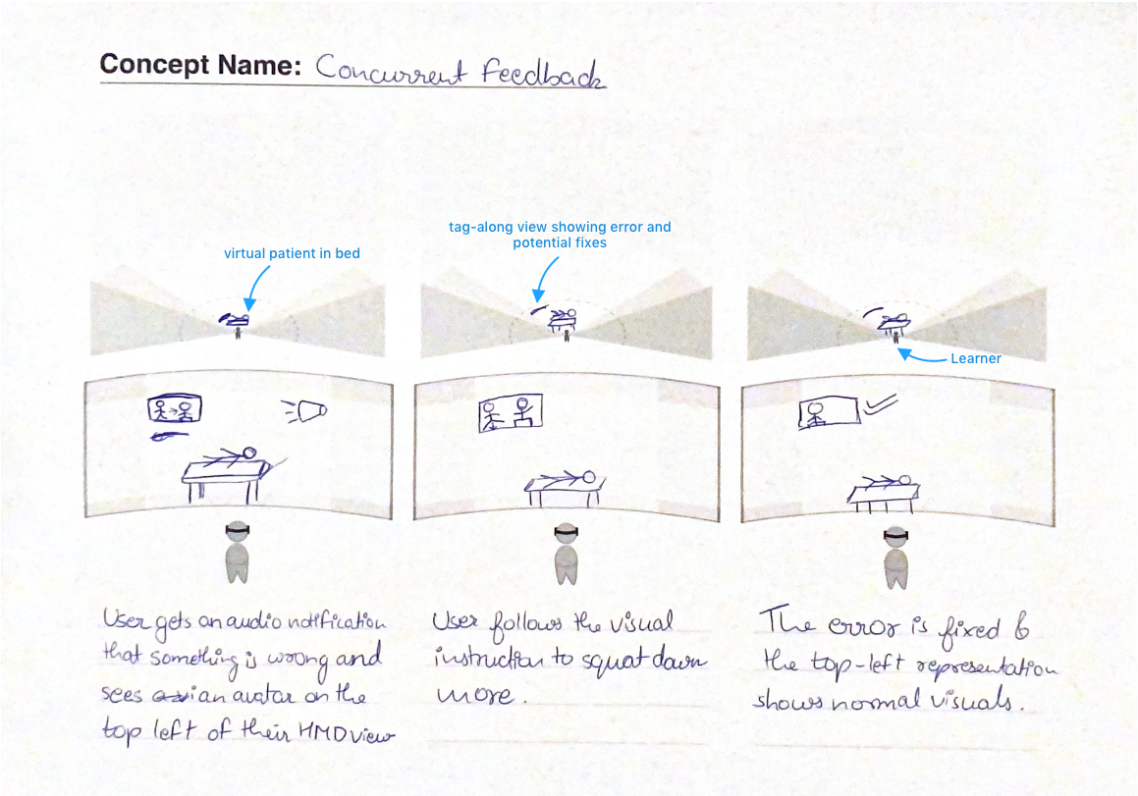

Figure 4.11: Sketching the process of delivering concurrent feedback.

This approach seemed good for showing error to the learner in real-time while performing the patient transfer. However, on discussing it with my supervisor Maximilian Dürr, we realised that the task is already too complex. Hence, showing so much information to the user at once would cause cognitive overload. Also, in the sketch we are looking at particular moment in time. However, while performing patient transfers, the movements are carried out in one single flow. Hence the animation wouldn't be able to account for the time synchronisation required. That is, the user would have already moved ahead until they animation of the potential fix would have been completed. So the right half of the tag-along window [34] was removed from the concept and just the highlighting body part concept was taken forward with the implementation. This is also a sufficient approach as the potential users of our system will be nurses who already have had basic training in the Kinaesthetics-based patient transfers as a part of their curriculum. Hence just knowing which error to fix would help the learner to improve their movements via training.

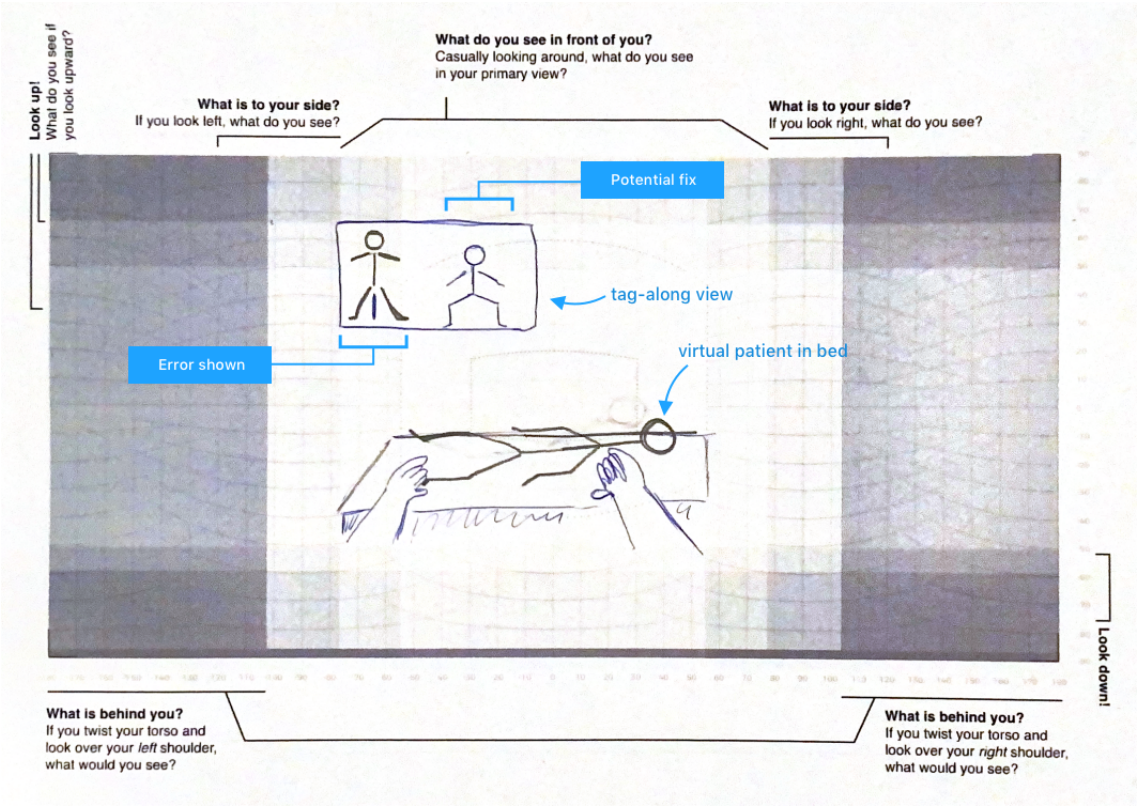

Figure 4.12: Sketch showing concurrent feedback shown from the learner's point of view.

To implement this concept, the system required an avatar with varying meshes for each body part to be highlighted. The process of creating a 3D avatar asset with required mesh selection was done using a tool named *Blender 3D*. Figure 4.13 shows a snapshot of the process of creating the humanoid avatar for delivering concurrent feedback. Each different mesh of the avatar body structure is linked to a type of motion error for each risk-metric. Figure 4.14 provides a glimpse of how does this error highlight look for the learner inside of the HMD. This type of element behaviour in mixed reality is known as tag-along. A tag-along object attempts to stay in a range that allows the user to view it or interact with it comfortably [34].

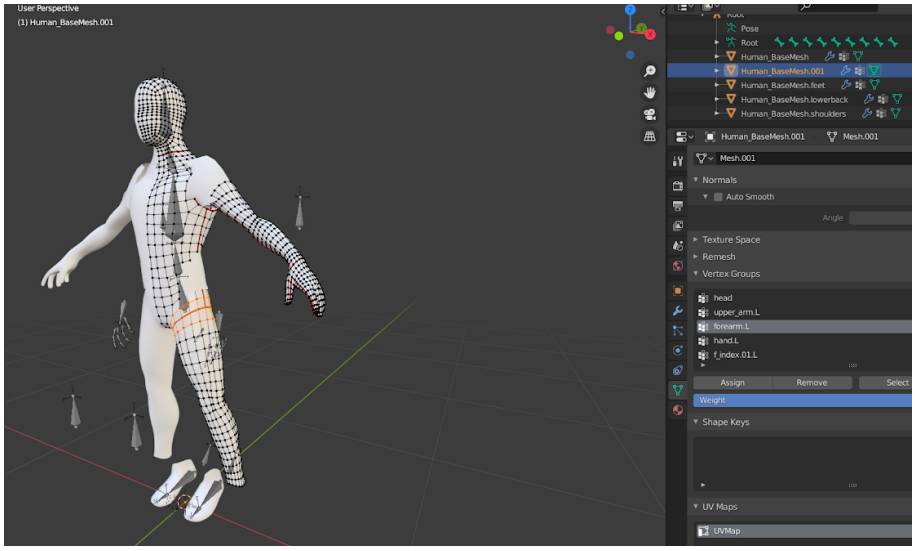

Figure 4.13: Creation of a humanoid avatar in Blender 3D with separate meshes.

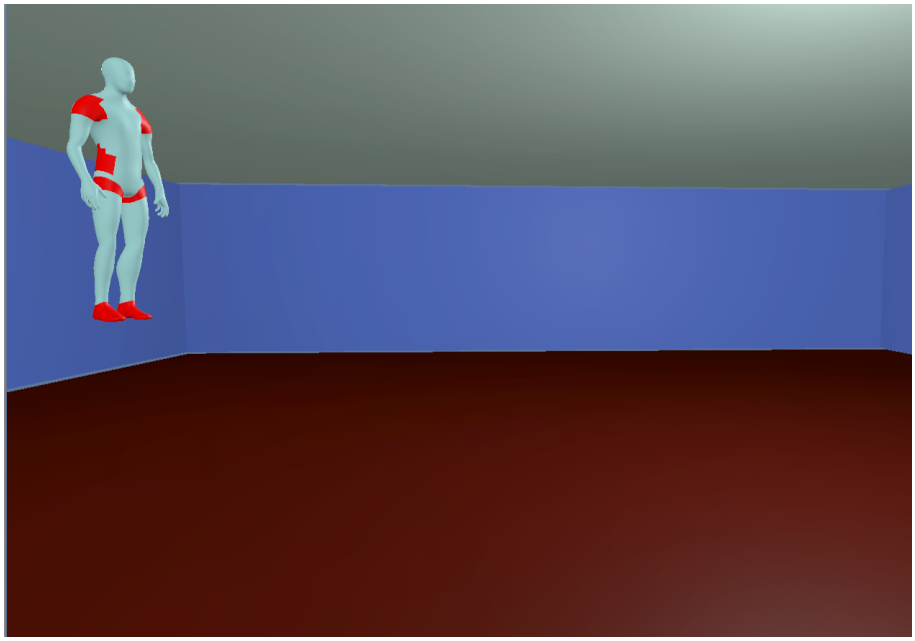

Figure 4.14: Highlights in the avatar meshes depicting error related to each of the risk-metrics.

Implementation - Movement recording in 3D space

An important section of the concurrent feedback module is the movement recorder. This part of the module lays the foundation that will allow the user to view a replay of their

movements in 3D virtual environment during the terminal feedback. Though movement playback is part of the terminal feedback delivery, all of the necessary information required for it is gathered during the virtual patient transfer is being carried out. For the ease of understanding the concept, let us clarify some terminologies.

- Gameobject - Each tracked part of the real world is represented as a gameobject in Unity. It is also b the base class for all entities in Unity Scenes. Since we are using Unity platform 2019.3.1f as our development platform we will be frequently using this term.

- Transforms - Each gameobject has 3 degrees of freedom namely Position(x,y,z), Rotation(x,y,z,w), and Scaling(x,y,z). These parameters are stored under transform attribute of a gameobject class.

- Animation - Each recording in Unity is stored as an animation component and is optimised for playback.

- Animation curve - Stores a collection of Keyframes that can be evaluated over time.

- Key frame - A keyframe is a structure optimised for storing data of gameobject as a function of time.

- Animation Clip - Stores keyframe based animations.

- Update - This function is called for every frame. Implemented as a part of Unity Systems.

Now that these terms related to movement recording are clear, we can move forward with how these classes and structures are used for recording movements. Figure 4.15 is a visual pseudo code explaining the logic of how things are implemented. An animation curve is created for the gameobject. The transforms values of gameobject are added every time to a new key-frame using a *Update* function. The key-frame structure stores the transform values and time of the recording. These key-frames are held together on an animation curve chronologically according to the time parameter. This process is repeated for each tracked gameobject. All the time synchronised animation curves for different gameobjects are stored together into an animation clip.

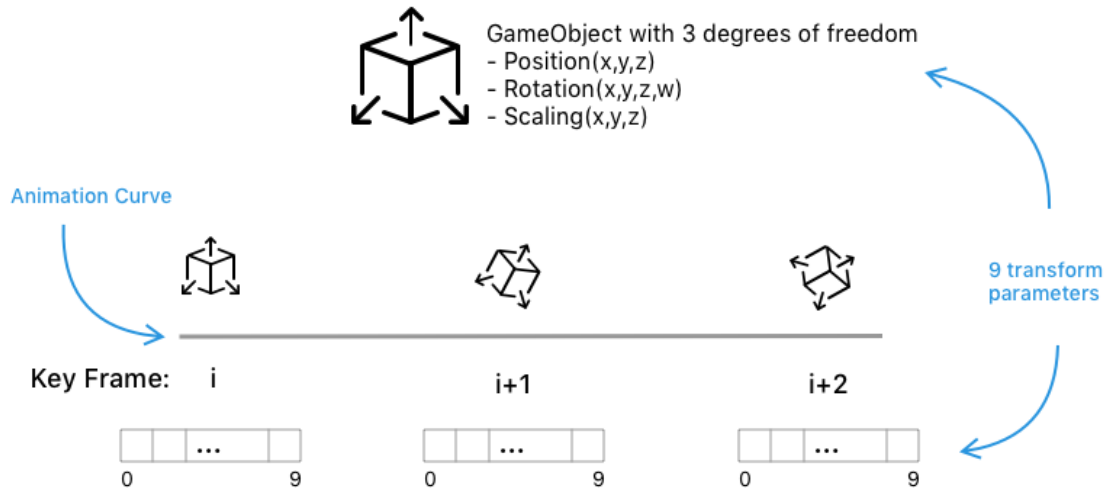

Figure 4.15: Visual explanation of data storage for movement playback.

This concludes one major module of our feedback system. Before moving to another major component of the code structure named Terminal feedback module, let's take a look into the helping modules namely: management & sync module and database storage & logging. These are smaller code implementations but are crucial for communicating data amongst the two major components of our system. Also, these module will help us to analyse how the user interacted with the system.

4.2.2 Helper modules

These modules help to manage and synchronise data between concurrent and terminal feedback. Once the patient transfer is completed the concurrent feedback module is on hold and terminal feedback module starts running. However, all the information gather during the patient transfer session is stored in the management module. Also, the session data is appended into a log text file for future analysis.

Implementation - management and synchronisation

Since the virtual reality patient transfer scenario by Daniel Schweitzer still needed some fixes, the speech recognition engine acts as a trigger for the movement completion. The speech recognition engine has four keywords defined as a part of it. The script uses input from the microphone which is part of the Valve Index HMD. When a relevant keyword is detected, the appropriate functions are triggered. Lets take a look at the various keywords supported by the speech recognition engine.

- Ready - This keyword is used to tell the system that the learner is starting the movements. It instantiates the hip calibration function and starts the movement recording mechanisms. It also tells the concurrent feedback module to start tracking and storing errors related to the four risk metrics.

- Done - This keyword defines the end of the patient transfer movement. It informs the concurrent feedback module to stop the risk metrics calculations and error logging. All the movement recording mechanisms are stopped and the animation clip generation is finalised. And data is forwarded to the terminal feedback module.

- Play/Pause - User can play or pause the recording in the terminal feedback module using these commands.

```
This function triggers functions base on keyword detected
by speech recognition engine //Pseudo-code

switch(keyword) {
    case "ready":
        Align Hip Sensors by extrapolation
        Start movement recording
        Start risk metrics tracking
        Start logging error
        Replace keyword variable with an empty string
        break;

    case "done":
        Stop risk metrics tracking
        Stop movement recording
        Stop logging error
        Save animation clip
        Save error markers log
        Send data to Terminal feedback module
        Replace keyword variable with an empty string
        break;

    case "pause":
        Get animator component of the avatar gameobject
        Change the animator speed to '0'
        Stop the slider movement on media control
        Replace keyword variable with an empty string
        break;

    case "play":
        Get animator component of the avatar gameobject
        Change the animator speed to '1'
        Start the slider movement on media control
        Replace keyword variable with an empty string
        break;
}
```

Further, we look into the ErrorLog element of the synchronisation module. Since arrays in Unity cannot be dynamically resized, the error logging system needed another flexible alternative to store information. The errors occurring during the patient transfer training are logged into Unity's Hash-table data structure. Hashtable are dynamic and also have value lookup time complexity of O(1). This highly improves the efficiency of error data retrieval in the terminal feedback stage. Because at the start of terminal feed-

back, the error marks are to be displayed on the slider of the animation timeline. The information in the Hash-table is stored every second with the *Key* parameter as time when the error occurred and *Value* parameter as the error code. The table 4.1 lists the errors associated to different error codes.

Error code	Risk metrics associated
1	1 (base distance)
2	2 (squat distance)
3	3 (spine bend)
4	4 (spine twist)
5	1 & 2
6	1 & 3
7	1 & 4
8	2 & 3
9	2 & 4
10	3 & 4
11	1,2 & 3
12	1,2 & 4
13	1,3 & 4
14	2,3 & 4
15	1,2,3 & 4

Table 4.1: Error codes and the corresponding risk metrics associated to them.

Next we look into the logging user information to local disk space and reading comparison from the database. To facilitate easier updating of comparison values in future, a JSON file is created. This file contains values for the various risk-metrics according to the gold standard. These values are categorised by the type of elderly patient such as partially dependent or totally dependent. Also, the value categorization take into consideration if the patient is being transferred form bed-to-chair or from chair-to-chair. Depending on the context of the patient transfer task scenario, relevant comparison values can be fetched.

For analysis purposes, these parameters are logged into the log.txt file for every second of the virtual session:

- Time in mm:ss format
- Trackers transform information
- Error occurred (0 for no error)
- Voice command triggered by user

```
database.json
{
    "overall":{
        "bedtochair":{
            "dcw":[0.40, 0.80, 55.95, 40.84],
            "gold":[0.83, 0.53, 55.79, 29.53]
        },
        "chairtochair":{
            "dcw":[0.37, 0.84, 50.83, 40.67],
            "gold":[0.65, 0.73, 43.19, 21.22]
        }
    },
    "partially":{
        "bedtochair":{
            "dcw":[0.42, 0.75, 55.17, 41.01],
            "gold":[1.00, 0.37, 67.87, 42.65]
        },
        "chairtochair":{
            "dcw":[0.38, 0.83, 52.68, 49.67],
            "gold":[0.55, 0.69, 40.45, 18.29]
        }
    },
    "fully":{
        "bedtochair":{
            "dcw":[0.38, 0.85, 56.70, 40.67],
            "gold":[0.66, 0.69, 43.70, 16.51]
        },
        "chairtochair":{
            "dcw":[0.36, 0.85, 49.28, 33.17],
            "gold":[0.76, 0.76, 49.92, 24.16]
        }
    }
}
```

Next we move to the final module of our system responsible for delivering terminal feedback functionalities to the learner/user.

4.2.3 Terminal feedback module

The main objectives of this module are:

- Parse and Load the data received from the concurrent feedback module.

- Load and update the VR user interface elements.
- Provide relevant risk-metric information to user in regards to their recorded movement.

This module will help the user to reflect on their patient transfer training. The implementation of this module enables the user to experience the post-session in a quite unique way. The system leverages the advancements in mixed reality technologies. It allows the learner to revisit a training session in their past by recreating the scene in virtual reality by the data we stored in the concurrent module. Let's take a detailed look into its various components.

Implementation - Animation control

This class receives the data from the concurrent module via the *ErrorLog* and *SpeechRecognitionEngine* components of the synchronisation module. The data received are the error values from the hashtable and the animation clip generated at the end of the patient transfer training. The animation clip is parsed and loaded into the Inverse Kinematics(IK) avatar [Figure 4.16]. Inverse Kinematics is the mathematical process of calculating the variable joint parameters that provides a desired configuration (position and rotation) for each body part of the humanoid 3D character. The Vive tracker data we collected from the learner's body joints is loaded into the IK avatar schema. On configuring the parameters like position weight, rotation weight and many more, the avatar replicates the movements very closely to how the learner performed those movements. The relative body part movements are calculated mathematically since Unity supports IK avatars natively.

Parallelly, the animation control class takes care of loading an animation player with an overview regarding the four risk metrics errors [Figure 4.17]. The slider UI component in Unity works on normalized values(range from 0 to 1). Animation clip length depends on the time taken by the learner to perform the patient transfer task. Hence, anim control class is responsible for calculating the iteration values per second and normalizing the values for the slider component to understand. Also, the error log values from the hashtable are reflected on the animation controller during this stage.

```
This function synchronizes animation time to the slider
void SynchronizeSlider()
    {
        timeElapsed++;
        totalTimeElapsed.text = timeElapsed.ToString();
        slider.normalizedValue += timer;
    }
```

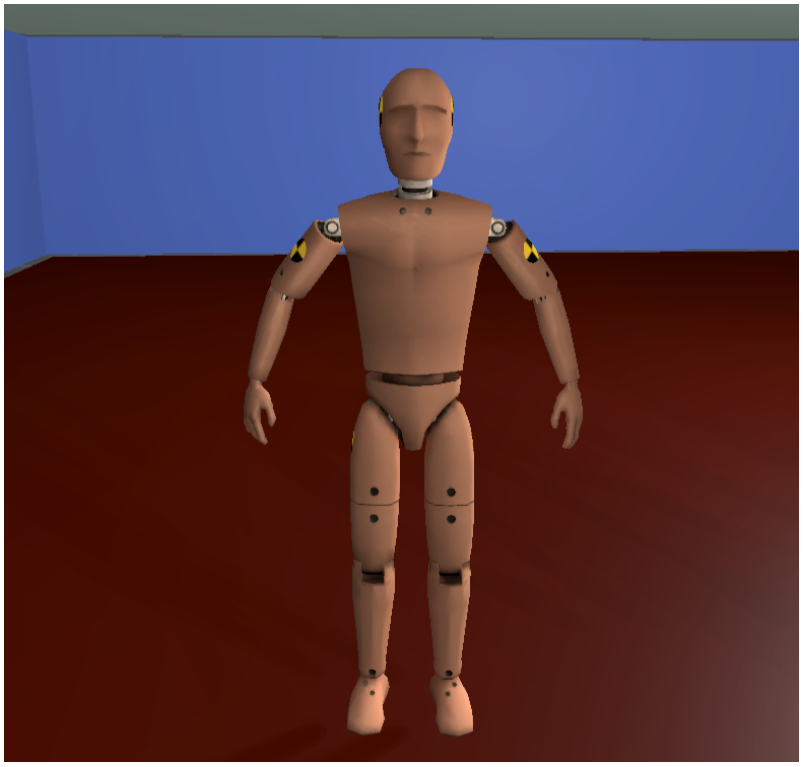

Figure 4.16: Correctly configured inverse kinematics avatar for movement reproduction.

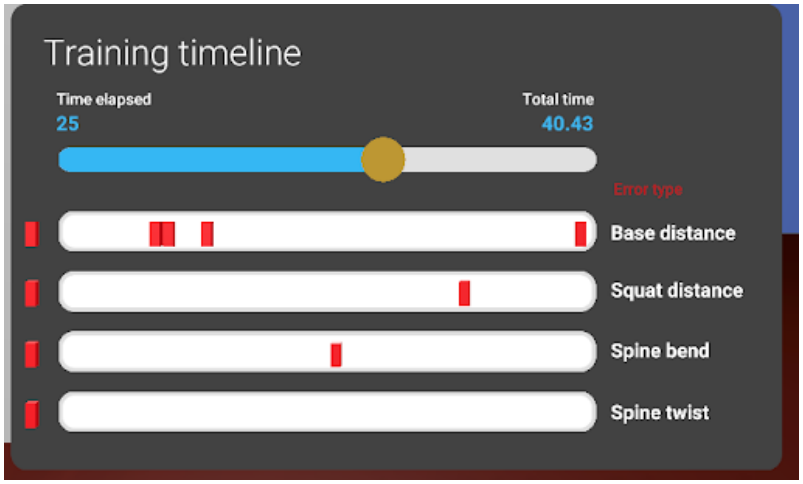

Figure 4.17: User interface of the animation controller with error markers.

```
This function spawns the error markers on the animation timeline
void SpawnMarkers(Hashtable) //Pseudo-code
{
  For all the values in the hashtable
  {
    Get reference location on Y-axis
    Calculate increment per second according to the length of animation
    Instantiate an error marker prefab in the scene
    Move the marker to the time of the error by calculating time*increment
  }
}
```

Description - Replay risk metrics calculator

It is an inherited class from the risk metrics calculator class. It keeps a track of replay movements and passes results to the UI Highlight component.

Unit testing - Replay recording user interface elements

Before implementing the final version of the animation controller interface, the first iteration was tested in a sandbox. The sandbox allowed for testing of scripts and implementation logic without a need of a VR HMD. Figure 4.18 shows a glimpse of such a testing process. The cube gameobjects in the figure represent the shoulder and hip sensors. The UI tooltip calibration worked well in this test. The values on the tooltip text elements were updated as per the animation timeline. During this test, the slider was successfully able to control the animation controller of the spine twist gameobjects. This implementation was later translated onto the final VR implementation.

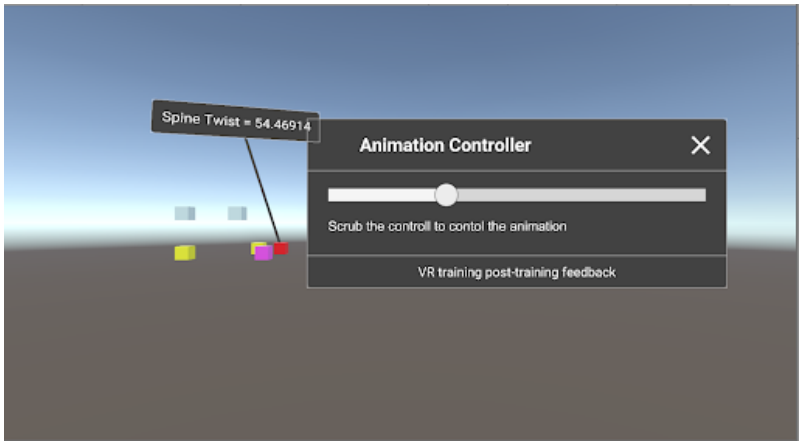

Figure 4.18: Sandbox testing for the first iteration of the animation controller.

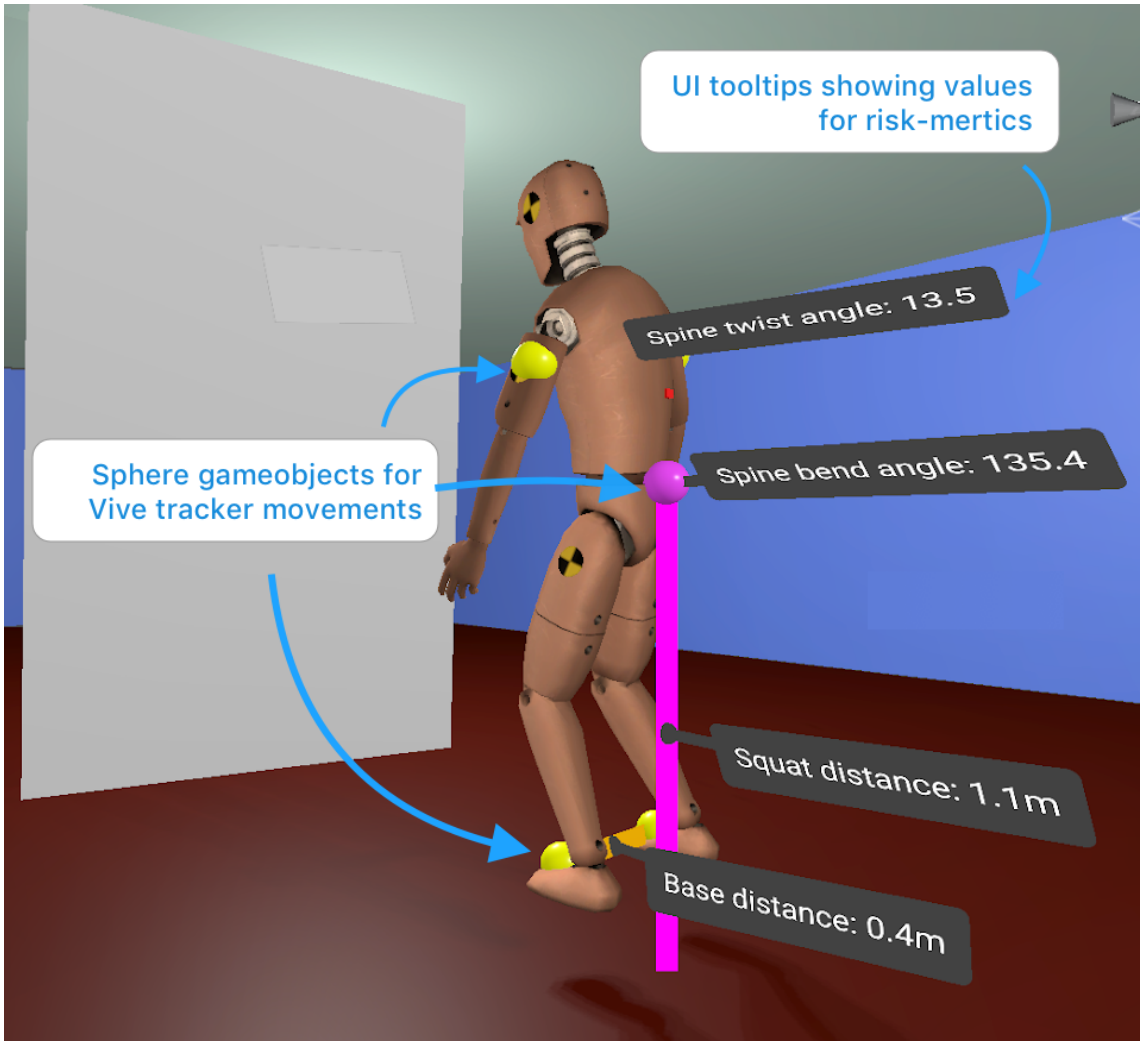

Figure 4.19: Elements user sees as a part of recording playback.

Implementation - Calibrate user interface elements

Figure 4.19 provides the highlights of what the learner sees in the terminal feedback. The cyan text and arrow markers in the figure are added later for providing explanations. For the purpose of explanation, the yellow and pink spherical gameobjects are shown. These gameobject represent the Vive trackers placed on the learner's body. During the actual demonstration these gameobjects are hidden. This component of the terminal feedback module calibrates the UI tooltips and moves them according to the avatar's placement. The text on the UI tooltips always follows the gaze direction of the learner/user. The user can play/pause the movements during this playback and can walk around the virtual environment to analyse the errors from various perspectives. The UI tooltips colors are also highlighted in red or orange when the values enter the error range. The highlight

color is chosen as per the deviation of the learner's posture values from the gold standard. Orange means the posture is slightly incorrect, while red signifies critical errors.

This successfully concludes our section regarding iterative implementation details. All these modules integrated together made our feedback system complete. The upcoming section briefly concludes this chapter.

4.2.4 Summary

At the end of the modules integration process, the standalone feedback system was complete. The overall working application can be experienced as a part of the project presentation in November. The system fulfills most of main goals by having all the core features as part of it. However, to make it whole, this system needs to be integrated with a virtual patient transfer task. Due to the time and resources limitation this agenda of integrating a task can be taken forward to the next phase of my masters. To summarize, the system successfully performs these tasks:

- Tracks and store information related to the four risk-metrics important for Kinaesthetics-based patient transfer movements.
- Informs the learner who is training for these movements, about the potential errors.
- Provides the possibility to a learner to revisit their recordings in a 3D virtual space and reflect upon their learnings.
- Allows the learner to play/pause the recorded movements and provides contextual UI elements for observing critical errors.

5 Outlook

As a part of my master's project, I understood how various mixed reality(MR) technologies could work together and the advantages and limitations of implementing in MR. Also, a vast variety of tracking technologies available provided a foundation of what could be possible in the future. From time to time during the development process, I was able to perceive that multiple different MR technologies are unformed or are yet unable to coherently work well with each other. However, there were moments where the potential of VR was realised. For instance, revisiting a recording in 3D by reconstructing a scene opens endless possibilities for other use-cases. Apart from the main goal, a lot was understood about the UX design life-cycle as well. The importance of going back in the life-cycle seems quite underrated. Revisiting the sketching phase during prototyping really helped me to find a decent alternative to the feedback modalities. In the upcoming sections, we will discuss the limitations of the implemented system, what could be improved, the details related to the evaluation of the prototype, and at last the expected timeline for my master thesis.

5.1 Limitations & Future work

The system we have developed is limited in terms of scalability and flexibility. If more risk-metrics are to be added which will require more data points from the body, this system won't be able to scale. Because a maximum of 9 Vive trackers can be tracked at a time. Also, the system setup cannot be easily carried with oneself so as to train at their own personal space. Ideally, the system should provide concurrent feedback via haptics to the learner. However, we had to go with the implementation of a fallback option using visual-audio feedback. The system requires multiple calibration steps and also requires the user to mount the Vive trackers on the body. Based on the learnings acquired throughout the project, there are some important areas of future work that are mentioned below:

- **Integration of Virtual Patient Transfer Task:** The feedback delivery to the user would be more relevant while performing some task.

- **Detecting of Lifting/Transfer Stages:** Automatic detection of lifting and transfer stages, which might require integration with pressure sensors to identify when a learner is carrying significant weight.

- **Measuring more Risk Metrics:** The system can be extended by adding other risk metrics related to physical injury/stress. For example, detecting bending of arms, detecting the force and direction of the movement.

- **Reducing the System Complexity:** The training system would feel more natural when the full-body tracking can be performed just by the HMD [35]. There are also some researches that point in the direction of smart garments [36].

- **Integration of Haptics Mechanism:** Integration of accurate, powerful, and yet compact haptic motors can make the training experience more realistic. It can also act as a seamless feedback delivery mechanism for future systems.

- **Enter a Patient Transfer during Playback:** The system can allow the user while viewing his/someone's recording - to pause and step inside a particular patient transfer and carry forward the movements after a particular point in time. This would help to immediately reflect and correct the mistakes.

Although not all but some part of the future work will be considered during the initial phase of my thesis. I plan on integrating the feedback system into the virtual patient transfer task developed by Daniel Schweitzer. However if that is not possible due to unforeseen circumstances, then a task related to picking and transferring a box in VR can be considered as a fallback option.

5.2 Study details

After integrating the implemented feedback system into a task, the next step will be to conduct a participant study as a part of the thesis. The ideal option considered as of now is a qualitative usability evaluation study. The main research goal of the study would be, "Impact of concurrent and terminal feedback during the virtual training of kinaesthetics-based patient transfer movements". The research goal can be achieved by answering these two questions:

1. How does the feedback mechanism feel as compared to the expert training in the academy?

2. Does the feedback mechanisms in virtual training improve the overall understanding of the learner regarding the application of the Kinaesthetics principle in patient-transfer?

The participants of this study would be 10-15 students from the medical academy. The students who qualify as participants are the ones who have attended the 3-day workshop as a part of their course. The study would include a virtual patient transfer task to be performed by the participant. After the task is completed, a semi-structured interview would be conducted. The data acquired through observations and the interview question would then be qualitatively analysed to understand the impact of the feedback mechanisms.

The proposed plan is feasible only if the medical academy agrees to the participation of students in the study. However, due to the COVID-19 restrictions and limited access to other resources a fallback option would be to conduct a heuristic evaluation from UX experts. This would require us to adapt the heuristics or qualitative guidelines to our VR system. The heuristic evaluation would require at least 3 to a maximum of 5 experts from the field of Human-Computer Interaction. The answers recorded by these experts will then be compiled, analysed and segregated. The usability problems then found out by the experts will be used to propose an updated system design that would overcome the shortcomings of the previous one.

Another approach could be to ask the experts from the domain of Kinaesthetics(teachers), to evaluate the whole training system from their perspective. The feedback that will be received as a part of this study would also be qualitative. But it can provide some insights into how a system in the future could be improved and iterated further. Worst case, the qualitative study will be conducted with the help of student participants from the University.

5.3 Timeline

According to the proposed timeline [Figure 5.1], I can start the task integration one week after the report submission. This comes from a perspective that on talking to Daniel Schweitzer, it was realised that the patient-transfer task in VR will soon be working stably. In parallel, I plan to refine the study details and get in contact with the medical academy via my supervisor Maximilian Dürr. This is necessary so as to look into the feasibility of conducting the study as per the preferred study design in the earlier section. The timeline related to conducting of the study depends on the procurement of participants. As soon as the study is completed, the next steps will be to analyse the collected data and complete the writing of the thesis by the end of February, 2021.

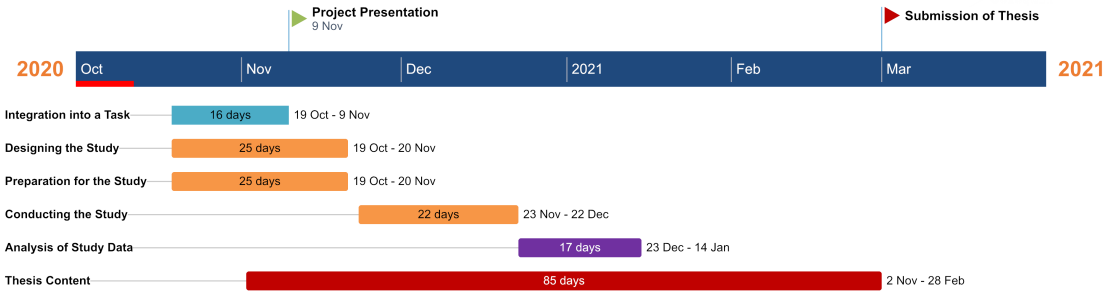

Figure 5.1: Master thesis timeline.

References

[1] A. GARG, B. D. OWEN, and B. CARLSON. "An ergonomic evaluation of nursing assistants' job in a nursing home". In: *Ergonomics* 35.9 (1992). PMID: 1387079, pp. 979–995. DOI: 10.1080/00140139208967377. eprint: https://doi.org/10.1080/00140139208967377. URL: https://doi.org/10.1080/0014013920896737.

[2] J Smedley et al. "Manual handling activities and risk of low back pain in nurses." In: *Occupational and Environmental Medicine* 52.3 (1995), pp. 160–163. ISSN: 1351-0711. DOI: 10.1136/oem.52.3.160. eprint: https://oem.bmj.com/content/52/3/160.full.pdf. URL: https://oem.bmj.com/content/52/3/160.

[3] "The physical workload of nursing personnel: association with musculoskeletal discomfort". In: *International Journal of Nursing Studies* 41.8 (2004), pp. 859–867. ISSN: 0020-7489. DOI: https://doi.org/10.1016/j.ijnurstu.2004.03.012. URL: http://www.sciencedirect.com/science/article/pii/S0020748904000628.

[4] Maximilian Dürr et al. "Learning Patient Transfers with Technology: A Qualitative Investigation of the Design Space". In: *Proceedings of Mensch Und Computer 2019*. MuC'19. Hamburg, Germany: Association for Computing Machinery, 2019, pp. 79–90. ISBN: 9781450371988. DOI: 10.1145/3340764.3340784. URL: https://doi.org/10.1145/3340764.3340784.

[5] Statistisches Bundesamt. "DE Statis | Statistisches Bundesamt". In: (July 24, 2019). URL: https://www.destatis.de/EN/Themes/Society-Environment/Health/Long-Term-Care/Tables/staff-nursing-care-facilities.html.

[6] Fraser Anderson et al. "YouMove: Enhancing Movement Training with an Augmented Reality Mirror". In: *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology*. UIST '13. St. Andrews, Scotland, United Kingdom: Association for Computing Machinery, 2013, pp. 311–320. ISBN: 9781450322683. DOI: 10.1145/2501988.2502045. URL: https://doi.org/10.1145/2501988.2502045.

[7] Ungyeon Yang and Gerard Jounghyun Kim. "Implementation and Evaluation of "Just Follow Me": An Immersive, VR-Based, Motion-Training System". In: *Presence: Teleoperators and Virtual Environments* 11.3 (2002), pp. 304–323. DOI: 10.1162/105474602317473240. eprint: https://doi.org/10.1162/105474602317473240. URL: https://doi.org/10.1162/105474602317473240.

[8] Janet van der Linden et al. “Buzzing to Play: Lessons Learned from an in the Wild Study of Real-Time Vibrotactile Feedback”. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’11. Vancouver, BC, Canada: Association for Computing Machinery, 2011, pp. 533–542. ISBN: 9781450302289. DOI: 10.1145/1978942.1979017. URL: https://doi.org/10.1145/1978942.1979017.

[9] Audrey Nelson and Andrea Baptiste. “Evidence-Based Practices for Safe Patient Handling and Movement”. In: *Orthopaedic nursing / National Association of Orthopaedic Nurses* 25 (Nov. 2006), pp. 367–8. DOI: 10.1385/BMM:4:1:55.

[10] Huiyu Zhou and Huosheng Hu. “Human motion tracking for rehabilitation—A survey”. In: *Biomedical Signal Processing and Control* 3 (Jan. 2008), pp. 1–18. DOI: 10.1016/j.bspc.2007.09.001.

[11] Huiyu Zhou et al. “Use of multiple wearable inertial sensors in upper limb motion tracking”. In: *Medical Engineering and Physics* 30.1 (2008), pp. 123–133. ISSN: 1350-4533. DOI: https://doi.org/10.1016/j.medengphy.2006.11.010. URL: http://www.sciencedirect.com/science/article/pii/S1350453306002633.

[12] Cynthia Marie Hadden. “Concurrent vs. Terminal Augmented Feedback in the Learning of a Discrete Bimanual Coordination Task.” In: 1998.

[13] Jonathan Muckell, Yuchi Young, and Mitch Leventhal. “A Wearable Motion Tracking System to Reduce Direct Care Worker Injuries: An Exploratory Study”. In: *Proceedings of the 2017 International Conference on Digital Health*. DH ’17. London, United Kingdom: Association for Computing Machinery, 2017, pp. 202–206. ISBN: 9781450352499. DOI: 10.1145/3079452.3079493. URL: https://doi.org/10.1145/3079452.3079493.

[14] Richard Tang et al. “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. Seoul, Republic of Korea: Association for Computing Machinery, 2015, pp. 4123–4132. ISBN: 9781450331456. DOI: 10.1145/2702123.2702401. URL: https://doi.org/10.1145/2702123.2702401.

[15] Nicholas Katzakis et al. “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. Brighton, United Kingdom: Association for Computing Machinery, 2017, pp. 58–67. ISBN: 9781450354868. DOI: 10.1145/3131277.3132181. URL: https://doi.org/10.1145/3131277.3132181.

[16] Azumi Maekawa et al. "Naviarm: Augmenting the Learning of Motor Skills Using a Backpack-Type Robotic Arm System". In: *Proceedings of the 10th Augmented Human International Conference 2019*. AH2019. Reims, France: Association for Computing Machinery, 2019. ISBN: 9781450365475. DOI: 10.1145/3311823.3311849. URL: https://doi.org/10.1145/3311823.3311849.

[17] Rex Hartson and Pardha Pyla. *The UX Book: Process and Guidelines for Ensuring a Quality User Experience*. 1st. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2012. ISBN: 0123852412.

[18] Ping-Hsuan Han et al. "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. Geneva, Switzerland: Association for Computing Machinery, 2016. ISBN: 9781450336802. DOI: 10.1145/2875194.2875237. URL: https://doi.org/10.1145/2875194.2875237.

[19] Thuong N. Hoang et al. "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. Gothenburg, Sweden: Association for Computing Machinery, 2016. ISBN: 9781450347631. DOI: 10.1145/2971485.2971521. URL: https://doi.org/10.1145/2971485.2971521.

[20] Aaron Bloomfield and Norman Badler. "Virtual Training via Vibrotactile Arrays". In: *Presence* 17 (Apr. 2008), pp. 103–120. DOI: 10.1162/pres.17.2.103.

[21] Martin Pielot and Susanne Boll. "Tactile Wayfinder: Comparison of Tactile Waypoint Navigation with Commercial Pedestrian Navigation Systems". In: *Pervasive Computing*. Ed. by Patrik Floréen, Antonio Krüger, and Mirjana Spasojevic. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 76–93. ISBN: 978-3-642-12654-3.

[22] A. Alamri et al. "Haptic Virtual Rehabilitation Exercises for Poststroke Diagnosis". In: *IEEE Transactions on Instrumentation and Measurement* 57.9 (2008), pp. 1876–1884.

[23] Kelly S. Hale and Kay M. Stanney. "Deriving Haptic Design Guidelines from Human Physiological, Psychophysical, and Neurological Foundations". In: *IEEE Comput. Graph. Appl.* 24.2 (Mar. 2004), pp. 33–39. ISSN: 0272-1716. DOI: 10.1109/MCG.2004.1274059. URL: https://doi.org/10.1109/MCG.2004.1274059.

[24] J. van der Linden et al. "MusicJacket—Combining Motion Capture and Vibrotactile Feedback to Teach Violin Bowing". In: *IEEE Transactions on Instrumentation and Measurement* 60.1 (2011), pp. 104–113.

[25] Andy Cockburn, Amy Karlson, and Benjamin B. Bederson. "A Review of Overview+detail, Zooming, and Focus+context Interfaces". In: *ACM Comput. Surv.* 41.1 (Jan. 2009). ISSN: 0360-0300. DOI: 10.1145/1456650.1456652. URL: https://doi.org/10.1145/1456650.1456652.

[26] Tamara Munzner and Eamonn Maguire. *Visualization analysis and design*. AK Peters visualization series. Boca Raton, FL: CRC Press, 2015. URL: https://cds.cern.ch/record/2001992.

[27] Thomas Brett Talbot, Katherine Elizabeth Thiry, and Michael Jenkins. "Storyboarding the Virtuality: Methods and Best Practices to Depict Scenes and Interactive Stories in Virtual and Mixed Reality". In: *Advances in Usability, User Experience, Wearable and Assistive Technology*. Ed. by Tareq Ahram and Christianne Falcão. Cham: Springer International Publishing, 2020, pp. 129–135. ISBN: 978-3-030-51828-8.

[28] Mike Alger. *VR Interface Design Pre-Visualisation Methods*. 2015. URL: https://vimeo.com/141330081 (visited on 10/09/2020).

[29] Facebook technologies. *Oculus documentation*. 2020. URL: https://developer.oculus.com/documentation/ (visited on 10/09/2020).

[30] Jérôme Théau. "Temporal Resolution". In: *Encyclopedia of GIS*. Ed. by Shashi Shekhar and Hui Xiong. Boston, MA: Springer US, 2008, pp. 1150–1151. ISBN: 978-0-387-35973-1. DOI: 10.1007/978-0-387-35973-1_1376. URL: https://doi.org/10.1007/978-0-387-35973-1_1376.

[31] Apple Inc. *Capturing body motion in 3D*. 2019. URL: https://developer.apple.com/documentation/arkit/capturing_body_motion_in_3d (visited on 10/09/2020).

[32] Mike Alger. *SteamVR™ Tracking*. 2018. URL: https://partner.steamgames.com/vrlicensing#Tracking (visited on 10/09/2020).

[33] Dominik Rausch et al. "Comparing Auditory and Haptic Feedback for a Virtual Drilling Task". In: *ICAT/EGVE/EuroVR*. 2012.

[34] Hololens Micorsoft. *Billboarding and taf-along*. 2018. URL: https://docs.microsoft.com/en-us/windows/mixed-reality/design/billboarding-and-tag-along (visited on 10/10/2020).

[35] Karan Ahuja et al. "MeCap: Whole-Body Digitization for Low-Cost VR/AR Headsets". In: *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*. UIST '19. New Orleans, LA, USA: Association for Computing Machinery, 2019, pp. 453–462. ISBN: 9781450368162. DOI: 10.1145/3332165.3347889. URL: https://doi.org/10.1145/3332165.3347889.

[36] Qi Wang et al. "Zishi: A Smart Garment for Posture Monitoring". In: *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems*. CHI EA '16. San Jose, California, USA: Association for Computing Machinery, 2016, pp. 3792–3795. ISBN: 9781450340823. DOI: 10.1145/2851581.2890262. URL: https://doi.org/10.1145/2851581.2890262.