Markerless Motion Capture for Real Time Animation

Abderrahman Mhadden, Asmus Eg Nielson Andresen, Marc Bisgaard Petersen

Supervisor: Henrik Schønau-Fog

Medialogy group 610, Bachelor Project, Copenhagen, 2021



Copyright © Aalborg University Copenhagen 2020



The Technical Faculty of IT and Design
Aalborg University
http://www.aau.dk

AALBORG UNIVERSITY
STUDENT REPORT

Title:
Markerless Motion Capture for Real Time Animation

Theme:
Interactive Systems Design

Project Period:
Spring Semester 2021

Project Group:
610

Participant(s):
Abderrahman Mhadden
Asmus Eg Nielson Andresen
Marc Bisgaard Petersen

Supervisor(s):
Henrik Schønau-Fog

Copies: 1

Page Numbers: 51

Date of Completion:
May 27, 2021

Abstract:

This project will focus on making markerless motion capture for real time animation and making it cheaper and easy to use. This was done based on researching the state of the art and related technologies. RGB-D camera was the tool of choice but due to lack of documentation following the newly developed *RealSense L515*, a switch to an RGB camera paired with an AI powered repository *ThreeDPoseUnity-Baracuda* that estimates human body positions was used to test the project. The results show that it could work to some extent compared with the state of the art Rokoko Suit while performing none complex movements. This shows that improvements to usability and AI would potentially make the product easier to use and equally accurate compare to the state of the art for real time motion capture devices.

The content of this report is freely available, but publication (with reference) may only be pursued due to agreement with the author.

Contents

1 Introduction **2**
1.1 IPS . . . 2

2 Analysis **3**
2.1 What is motion capture? . . . 3
2.2 Motion Capture Solutions . . . 4
2.3 Technical Issues with Motion Capture . . . 5
2.4 State of The Art . . . 8
2.4.1 Motion Capture Suits . . . 8
2.4.2 LiDAR Camera for Motion Caputre . . . 9
2.4.3 RGB Cameras for Motion Capture . . . 9
2.5 Artificial Intelligence . . . 10
2.6 Real-Time Animation and Recording Motion . . . 11
2.6.1 Motion Retargeting . . . 12
2.7 Delimitation . . . 12
2.7.1 Final Problem Statement . . . 13
2.7.2 Design Requirements . . . 13

3 Methods **15**
3.1 Methodological approach . . . 15
3.2 Agile . . . 16
3.3 Usability test . . . 17
3.3.1 Likert Scale . . . 17

4 Production **18**
4.1 First Iteration . . . 19
4.1.1 Design . . . 19
4.1.2 Implementation . . . 20
4.2 Second Iteration . . . 20
4.2.1 Design . . . 20

4.2.2 Implementation . . . 21
4.3 Third Iteration . . . 24
4.3.1 Design . . . 24
4.3.2 Implementation . . . 25
4.4 Fourth Iteration . . . 26
4.4.1 Design . . . 26
4.4.2 Initial Attempt of fourth iteration . . . 27
4.4.3 Implementation . . . 29

5 Evaluation **31**
5.1 Initial testing . . . 31
5.2 Testing In A Non-Covid19 World . . . 31
5.3 First test . . . 32
5.3.1 Setup . . . 32
5.3.2 Procedure . . . 33
5.3.3 Participants . . . 34
5.3.4 Results of the usability test . . . 34
5.4 Comparison test . . . 37
5.4.1 Setup . . . 37
5.4.2 Procedure . . . 37
5.4.3 Results of the comparison test . . . 37

6 Discussion **41**
6.1 Evaluation discussion . . . 41
6.1.1 Comparison test discussion . . . 42
6.1.2 Work flow and planning . . . 43
6.1.3 Development discussion . . . 44

7 Conclusion **45**

8 Appendices **47**

Bibliography **49**

Chapter 1

Introduction

Motion capture technology is the technology that can transform a persons movements into a 3D environment, this data can then be used to control a virtual character [15]. This technology is used for many purposes, but especially in the film and game industry. As the technology has evolved more people have gotten their hands on the equipment to create motion capture at home, but at a hefty price tag. Prices can be as low as $200 for independent self-build options and these can scale op to $1500. It is also possible to buy kits that range from $500 up to $15000 [2]. Although, for a professional kit the price ranges can reach upwards of 25000$. During the covid-19 pandemic the need for home setups has increased as professional and amateur animators and artist can not go to their work space and studios to record. this has lead to people seeking cheaper alternatives for motion capture, but often these alternatives are complicated to use, require self building or lack accuracy. Therefore, we want to further investigate the possibility to create a cheaper alternative to motion capture that can preform real time animation.

With this initial information and motivation in mind, an initial problem statement will be formulated:

1.1 IPS

IPS: *"How can we create a cheap alternative to a motion capture suit for real time animation?"*

Chapter 2

Analysis

In this chapter we will explore and investigate motion capture in greater detail and expand on the Initial Problem Statement. This will be done by taking a look at what motion capture is, how it is used and what the technical difficulties are. Then we will take a look at the State of The Art of motion capture and the kinds of technologies that are related to motion capture. Finally, the chapter will end with a formulation of a final problem statement and a list of technical and design requirements, which will be based on the knowledge gained in the analysis.

2.1 What is motion capture?

Motion capture is the process that translates movement from the physical space[1] to the virtual[2] world in real time [19]. This process is achieved by recording physical motion and rendering it into data using tracking points in a four dimensional environment (space over time). This data is mapped on a three dimensional (3D) character rig. The points could be sensors or markings on the subject being recorded, and they should be located on pivot points between inflexible parts, for example; joints for full bodies or all around the lips for faces. In conclusion, as said by Alberto Menache in the Understand Motion Capture for Computer Animation and Video Games;

"Motion capture is the collection of data that represents motion [19]."

There are different ways of collecting motion data. Some use sensors in an electromagnetic or ultra sound field to track the motion. Others use cameras by either tracking physical markers on the subject or using image processing to generate

[1] Real world performance
[2] Digital world performance

a body rig with markings over the captured images. Some use a combination of both or other technologies like depth camera. But over all, all these methods aim to accomplish one goal, which is capturing accurate motion and providing data from it [19].

2.2 Motion Capture Solutions

As motioned above, when working with motion capture there are a verity of different solutions available to achieve different levels of motion capture. This is both in regards to the capture of the movement and the process of interpreting the information recorded and translating it into a 3D or 2D environment. As the technology has evolved different methods have been developed to suit these different levels of motion capture and use cases. These use cases span from bio mechanic recording, military development to entertainment. As the technology evolved it saw even more uses, such as in cartoons, Hollywood and games. The technology became broader and new tools were developed [1].

As mentioned, Depending on the use case, different forms of motion capture was developed taking advantage of different technologies and setups. In the more professional setting, the captures have to be accurate within the smallest of movements by the actors and must be observable from every conceivable angle. Therefore, motion capture recording stages were developed, where within a stage the actors every movement could be accurately recorded using suits with markers for reference on key placements on the actors body (joints and expressive facial muscles.). These recordings could then be translated to characters in a 3D environment [28]. Generally these markers are considered in two groups of markers, passive and active, and work by either reflecting or emitting light back to one or more high refresh rate cameras. This data is then used to process the relative location of the body in real space to the 3D space [11].

Mechanical tracking systems are also an option. A mechanical tracking system refers to a lightweight "skeleton" that is worn by the actor. This type of solution is typically quite restricting for the actor to move in and often bulky and expensive.

Inertial tacking or sensor based motion tracking uses IMU (Inertial measurement units) to calculate the location and movements of the actor. This is done by the sensors recording the movement using gyroscopes, magnetometers, and accelerometers to determine the direction, orientation and speed of the movements.

Marker-less tracking is another option which is quite popular and seems to have cheaper options in home built systems. It is considered a less accurate option, but typically it works by using special cameras with depth sensors or heat mapping to determine the location and movements of the actor. Although, regular web cam-

eras can also be used, using RGB technology.This technology is easier to setup for beginners, amateur's, but professionals seem to prefer more setup in return for a much more accurate recording. The main difficulties relate to feet, hand and facial tracking, but this goes for most if not all the motion capture solutions available today. The potential of marker-less tracking is nowhere near its final potential, and it may be in the near future that marker-less tracking will see a break through. As development continues for marker-less motion capture, content creators are seeking to find cheap quality motion capture solutions to use for their private projects and work related tasks [11]. An example of how marker-less tracking is visualised can be seen in figure 2.1.

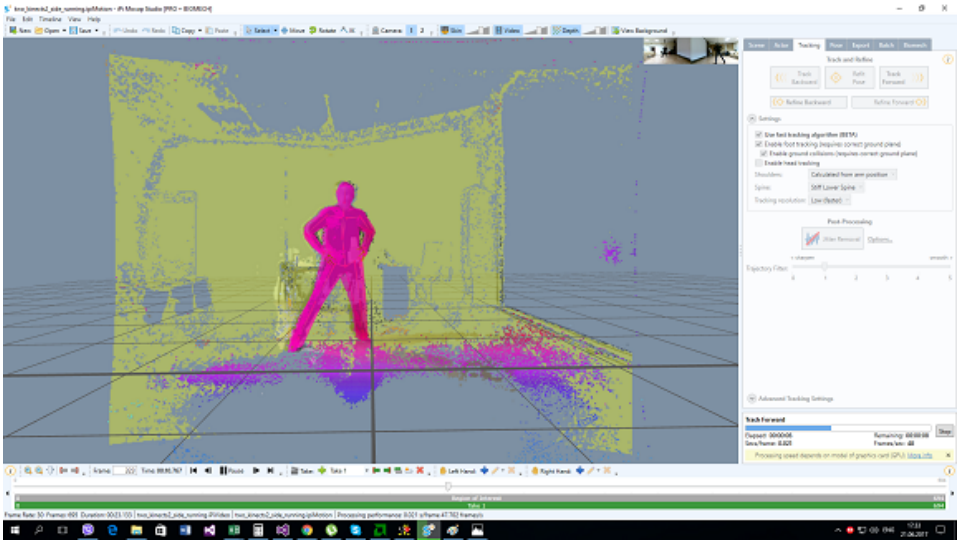

Figure 2.1: The image shows how the data that is being recorded by the Kinect depth camera is interpreted and visualised [29]

Furthermore, other lesser known motion capture technologies exists, which specialise in more accurately recording hands and feet movements. Face tracking is in a league of its own, but it most often refers back to the marker based systems. Where they use markers on the actors face with a head mount (lights and high speed camera) to track every detail of the actors expression. Other technologies are also being developed such as using floor mats with sensors and other creative solutions to capture motion capture [22].

2.3 Technical Issues with Motion Capture

Now that we have taken a look at the different kinds of motion capture solutions that are available, we want to investigate the challenges and technical issues that

may occur when working with motion capture.

When working with motion capture, the "stage" has to be setup. For most motion capture solutions, this can be a tenuous task as a dedicated space has to be cleared and rigged to accurately record the actors movement with as little interference as possible. This can make it difficult for users to set up the motion capture and space in their own homes or in new locations. The main obstacles are lighting, removing magnetic interference (sensor based motion capture), uniform backdrop (for marker-less), securing space to move safely within, and the rigging of the equipment within the defined area [11].

Depending on the kind of motion capture solution that is being setup, different levels of setup and construction has to be done. Marker-less motion capture is the easiest to set up, as it requires clearing floor space and then depending on the cameras and other technologies that are being used, setting up the recording equipment and calibrating the cameras. For marker-less tracking it is important that the lights are placed behind the cameras, to ensure each camera has clear view and the user is well illuminated [28].

Some systems require as little as 1 camera to function and the amount of cameras can be scaled up to add more accuracy and the number of individuals that can be tracked depending on the use case. The camera must be placed so that it can capture the actor directly from the front to ensure the accuracy of the movements in their full motion [11]. When working with marker based motion capture, more setup is required. A suit with the aforementioned passive or active markers has to be put on the actor, with the markers placed on key reference points on the body (joints). The suit then has to be calibrated to the position of the cameras. Sensor based tracking is similar, but instead of having to wear a skin suit, the actors Velcro straps with sensors attached to their joints. The actor then has to stand in a reference pose to most accurately record the movements [28]. An illustration of how such a setup may look can be seen in the figure 2.2.



Figure 2.2: Screenshot of the setup of motion capture using camera from OptiTrack [24]

While setting up the motion capture suit or cameras, the computer specs have to be taken into account, for most of them have a minimum requirement to be met in order for the system to work.

Rokoko Studio	OptiTrack PrimeX 41
Windows 10 (64-bit)	Windows 7, 8, 10 (64-bit)
Intel Core i5 3.00 GHz	3.00+ GHz i5 Processor
8GB RAM	8GB RAM
NVIDIA GTX 980	1 GB Ethernet port

Figure 2.3: Minimum requirement for Rokoko Studio [3] and OptiTrack PrimeX 41 [4]

As seen in the figure 2.3, both motion capture suits and camera based motion capture have a minimum requirement that has to be met in order for it to work in the intended way. This also means that the user has to invest into a good computer set up before investing in the motion capture set up, which adds up to the total cost. Furthermore, when working with marker-less tracking a higher amount of computational power is needed, which require newer and better hardware. This specifically relates to the Graphics card.

When working with an RGB camera, the footage is required to run through a com-

puter vision application, since the computer can be taught to detect body parts via machine learning [26]. By doing so, it is possible to translate the motion recorded in two-dimensional view to a 3-Dimensional space. An issue that can appear using this concept, is that rotation of the body can not be recorded as easy, since the body or an object may obstruct the view of the body part. This can be handled by using an extra camera view from another angle (90 degree from the original camera) or muptile angels. Another way is to use machine learning and/or artificial intelligence by training it to fill out the gaps in the recording or guesstimate the location [17].

2.4 State of The Art

In the following subsections we will explore the state of the art for motion capture suits, cameras and motion capture technologies. We will take a look at the different kinds of products that exists and observe their strengths and weaknesses.

2.4.1 Motion Capture Suits

The state of the art for motion capture suits generally work with markers and sensors. The sensor can be found in the motion capture suits, where each joint has a sensor attached. When working with markers they are placed on the outside of the suit on the same positions. In general, sensor based suits use IMU sensors, which contains a gyroscope for rotation, an accelerometer for movement and a magnetometer for direction. These sensor will help the computer to understand where each joint is pointing, how fast they are moving and their rotation. The amount of sensors can vary from 10 sensors like the Tesla Suit all the way up to 32 sensors, like the Perception Neuron (figure 8.1). The data from the sensors has a frequency between one to 300 hertz(Hz) where the Xsens suit has a 1000Hz internal (figure 8.1). So, for the user, the higher the frequency is, the more accurate the data from the sensor is. The suits from the list that we compiled (figure 8.1), all of them have a wireless connection which means that a wi-fi connection between the suit and computer is possible. their range can be up to 100 meters and can have a latency as low as seven milliseconds. All of the suits except Tesla Suit, has compatibility with the game engines, Unity and Unreal as well several 3D editing programs like Autodesk Maya and Blender. Lastly, the pricing for the suits is in the $1000 range. The Perception Neuron is the cheapest at $1499 where the Xsens suit is at $5190. These suit are on the more pricey side, but the data from the suits make up the hefty pricing. As it has more smooth movement between the joints.

2.4.2 LiDAR Camera for Motion Caputre

Based on the Motion Capture Solutions section stated preciously in the analysis (2.2), it is clear that motion capture industry is growing and using various methods and different hardware to get a better marker-less motion capture system. In this research, we will be focused on making this process more accessible to amateur content creators and indie game developers, with the highest accuracy possible for a reasonable price that is easy set up and use. After looking into the different types of suits and cameras, a couple of options where available; RBG Camera, Phone, LiDAR Camera (RealSense L515).

LiDAR Cameras or also known as RGB-D cameras use a combination of an RGB camera and an infrared projector. The IR projector projects a speckle pattern that is then captured by an IR camera that is located next to the RGB camera [16]. The data from the inferred sensor is then added and calibrated with the data from the RGB camera. This gives the outcome of an RGB image with a depth pixel. This data is represented as *point cloud* which is a collection of points in a three dimensional space, where each of these points have additional proprieties associated with it.

The Intel RealSense LiDAR Camera L515 uses a combination of sensors; an IR laser, an IR photodiode, a MEMS (Microelectromechanical systems) that is a micro actuators and sensors, that react to environmental change under microcircuit control [30], a MEMS contoller; which scan the laser beam over the field-of view, a vision ASIC; that will process the data from the reflected IR beam that is captured by the photodiode and will output the accurate distance from a specific point in the scene to the camera, and finally an RGB Imager. The gathered depth points will then form a point cloud depth data that represents the entire scene [13]. In simpler words, the LiDAR Camera shoots an infrared beam that bounces back and then it calculates the distance between camera and the bounce by calculating the time it took the laser beam to hit the camera back. Each laser beam creates a point in a three dimensional space and all the points that have been gathered from the field of view create a replica of the object that the beams bounced of. This is then combined with the RGB image coming from the RGB Imager. This process will be further explained in the following section about how motion capture is made using only an RGB Camera.

2.4.3 RGB Cameras for Motion Capture

Recent Digital Cameras and video Camera or also known as RGB cameras, are made with a combination of parts that consist of a lens (Imaging Optics) to capture the desired scene and focuses on the Microlens Array, a Color Filter Array and an Image Sensor. This divides the captured scene into smaller parts that are color filtered and then read by their respective image sensors [8]. RGB Cameras started

using complimentary metal-oxide semiconductor image sensors or also known as CMOS that produce high quality images. The CMOS are composed of many photodiodes; the more diodes it consists of the more higher the quality is. These photodiodes convert the light waves into voltage that is then sent to an AGC (Automatic Gain Control) or an ADC (Analog-to-Digital Converter) that converts the voltage signals into digital signals that are then either stored or displayed as pixels into an output device [3]. The RGB Camera is one of the first tools used for computer vision, using the images detected with image processing tools like OpenCV. This combination of RGB Camera and image recognition libraries is used to detect objects, shapes or even people [10]. This can be done in many ways, like color recognition or shape recognition. This is also usable in human body detection by detecting joints locations that are marked on the image and grouping them to form a skeleton [23]. The marks have a two dimensional coordinate each that is updated depending on the frequency of the camera and the software used to analyse the pictures.

2.5 Artificial Intelligence

To help with detecting objects or humans through a camera, artificial intelligence or AI can be a great assistant to identify what is within the camera view. To use AI, a better understanding of how it works is needed.

AI is when a human is teaching the computer to perform any task that previously would have required human intelligence according to John McCarthy and Marvin Minsky back in 1950 [12]. Nowadays, the AI researcher, Francois Chollet says that intelligence is tied how well a systems can adept and improvise in a environment through generalising its knowledge and use it in unfamiliar situations [12]. When taking about AI, it can be split into two broad types: narrow and general AI. Narrow AI are taught or have learned to perform specific a task without being programmed towards the said task such as recognise certain objects in a video feed [12]. General AI has a fixable form of intelligence that can help it learn how to perform a variety of task through experience, similar to humans [12].

Although, for each of the models it preforms differently, but they can be taught using similar methods. One of these methods is machine learning where the AI is fed a huge amount of data to carry out a certain task, such as recognise objects or living creatures [12]. This means that the data should contain information about the key points that are valuable for the given task, such as position of body parts and joints.

Another method of learning would be the neural network. The neural network has a similar structure as a human brain, where each neurons is feeding information

to each other. The neural network uses layers of algorithms that feeds the data to the other layers as seen in figure 2.4.

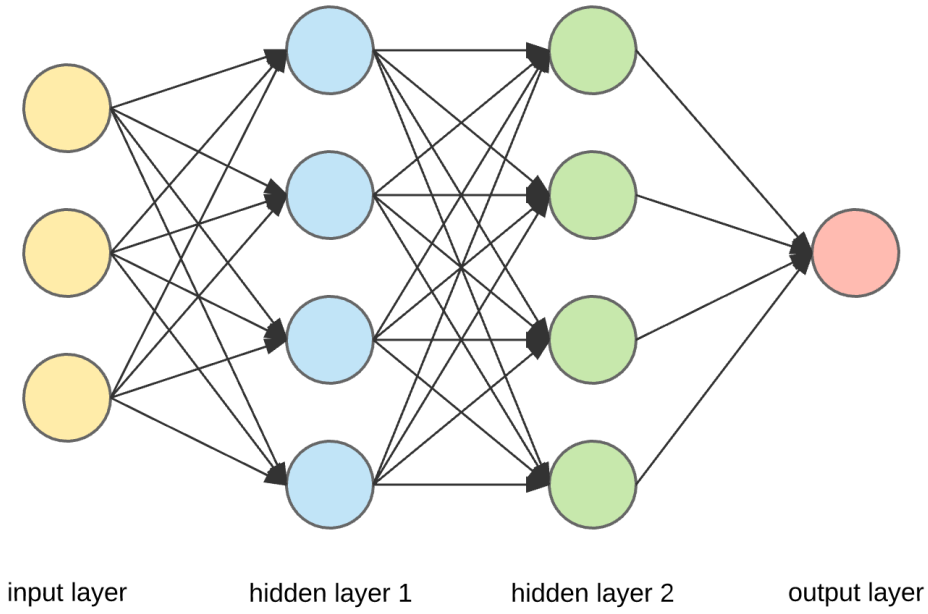

Figure 2.4: Neural network system [5]

By doing so, the main task will be split into smaller tasks for each layer. So the task for the system could be to detect a body, where each layer would search for certain body part which will contribute to overall task. With neural networks, different optimizations in the layers can give better results. By setting up the neural network in a convolutional setting, the accuracy of detecting the right object is increased. While both methods of learning are good, the machine learning is more time consuming in the long run where neural network can be fast to set up and use. Therefore, it is likely that we would go for a convoluational neural network for the time saving and it's fast setup.

2.6 Real-Time Animation and Recording Motion

There are different ways of using captured motion for animating a 3D character, one is *Real time animation*, also known as *Real-Time Performance Capture* or *Live Animation* or *Live Performance*. It is the act of using a motion capture system live with a real time feedback. This animation method is used more frequently as motion capture systems are more relevant and accessible. When working with animated 3D characters and making them come to life in a performance for an audience or director [6].

The second one is known as *Recorded motion* or *Recording Motion*, which is the most common way of animating 3D characters, due to it's ability to be altered or to retake different takes of a single movement. This is often used by animators in big animation studios, as it is easy for the animator to use a motion capture system whenever they need. The animation can be recorded, then altered to be adjusted based on the requirements of the scene [21].

Each of these motion capture techniques are different and good in their own way depending on the situation. Prerecorded motion can be used when animating in many situation, like a hand wave animation or a walking animations.

Real-Time Animation is good for live shows like online streaming and the new trend of the V-Tubers; which are Virtual YouTubers or live streamers that create a virtual character and control via motion capture. They make the character the star of the show and not the person behind it.

Now that we know two different ways of using the captured motion, how can it be applied to different kinds of 3D characters? In the following section we will take a look at this.

2.6.1 Motion Retargeting

Animating 3D characters can be done in different ways as mentioned in the previous section 2.6. When applying the animation to the rigs, the animation could look incorrect due to the difference in rigging depending on the 3D model. Because different rigs have different joints in general, depending on their size and if they are human 3D models or other things like Mickey Mouse or The aliens from the Alien movie.

They have different rig structures and joint positioning, which makes translating this data from a human rig into a none human character look incorrect [14]. This can be solved by using an AI driven tool called *Motion Retargeting*. Motion retargeting works by comparing the rig used by the motion capture tool and the rig of the animated 3D model and finding the key point joints in between. It then creates a new rig based on these key point joints and applies the animation onto it. After this it uses the newly created rig to animate the 3D model [27].

2.7 Delimitation

The report started with an introduction and initial motivation of this project going over the inaccessibility of quality motion capture equipment and a decent price for consumers (chapter1). This lead to the question-forming of the Initial problem

statement - *"How can we create a cheap alternative to a motion capture suit for real time animation?"*

Through the analysis we have investigated and researched the motion capture industry and technology in order to expand on the IPS. This was done by taking a look at what motion capture is and how it is used 2.1. Then we took a look at the different technologies and approaches that are used to create motion capture solutions 2.2. Within this chapter, knowledge on different forms of motion capture was discovered and the basics of what is required of them to function. Then we took a look at the technical and preparational aspects of using motion capture, where it was discovered that within the different kinds of motion capture solutions, problems and obstructions would occur for each of them 2.3. At this point, we wanted to take a closer look at the state of the art for motion capture, to gain an understanding of how the best of the industry does motion capture, but also to grasp what the requirements are for making a good motion capture solution 2.3. The SOTA chapter, takes a look at both optical and suit solutions. We then used the knowledge gained to narrow down the scope of the technology that we wanted to use and further investigate. It was decided, that the solution should be cheap, accurate, easy to use and run in real time. Therefore, some technologies were picked out to stand as examples of the level at which we will judge our further developments 2.4.2. Finally, in the analysis we took a look at how the technical aspects, including how the data and feedback loop works when working with motion capture.

With the knowledge gained through the analysis and delimitation in mind, we formulated our Final Problem Statement.

2.7.1 Final Problem Statement

"How can a program that utilizes LiDAR technology to achieve cheaper, markerless and easy to use motion capture be created for real time animation?"

This was later changed, as the LiDAR technology did not end up working as expected. This was due to inability of continuing to work with the LiDAR technology, as the compatibility was limited. This will be explained in greater detail in the first iteration 4.2. It was changed to:

"How can a program that utilizes webcam RGB technology to achieve cheaper, markerless and easy to use motion capture be created for real time animation?"

2.7.2 Design Requirements

In this section we will outline the design requirements for our product development. These requirements will stem from our research in the analysis. Functional

Requirements:

- Ability to track body movement
- Translating the body movement into data that can be used in Unity
- Ability to animate in real time

Non-Functional Requirement:

- Should utilise LiDAR/RGB technology
- Utilising image processing library OpenCV
- Should be made using marker less technology
- Should be easy to use/install by users
- Should stand as a cheap Alternative to other motion capture solutions.

Chapter 3

Methods

In this chapter, we will outline the methods that will be used in this project regarding development and testing in particular. This section should stand as an explanation of which methods will be used, why we are using them and how. In order to do this, the research onion model will be used.

3.1 Methodological approach

When planing a study the research onion can be used to determine the methodological approach for the study. Generally speaking, it works by selecting an approach from the most outer layer available and then moving in a layer and picking one of the surrounding approaches. Each layer that is closer to the core determines a more specific approach regarding how the project will proceed. An example of how the research onion looks can be seen in figure below 3.1.

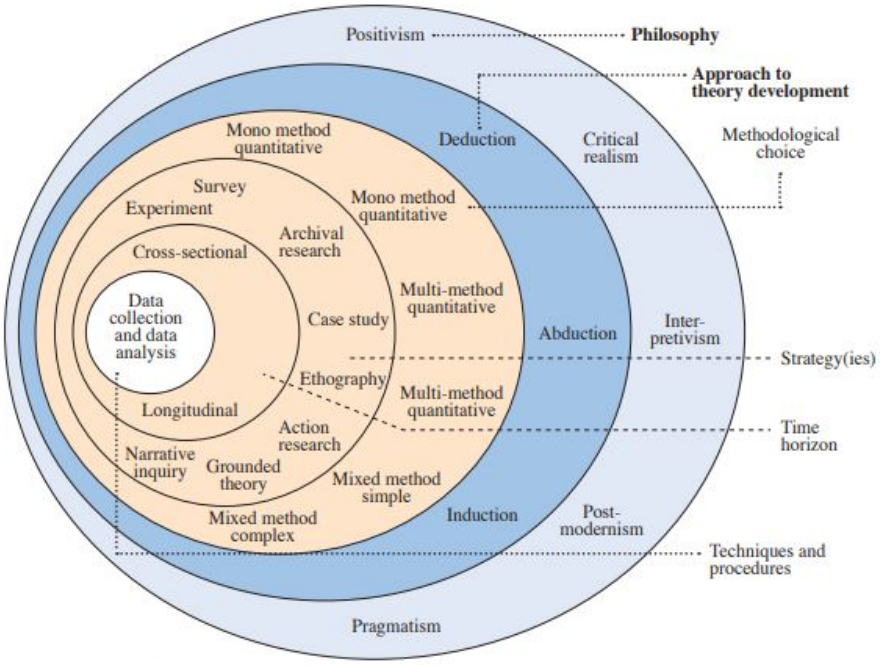

Figure 3.1: Research onion [18]

From the first layer, the group philosophy best resounds with critical realism, which relates to the possibility, or likeliness that the future is certain but may be altered and changed by people taking action. In the second layer, we will be working with deduction as the base for research theory in this project. Although, it may occur that some induction will be used, when data has been gathered. This second layer relates to how theories are formed and worked with. In the third layer, it can be said that we will be working with multi-method quantitative, as the current plan is to preform a survey with likert scale and collect data from a comparative test. In the fourth layer, both the survey and experiment strategies are intended to be used. In the fifth layer, we will use the cross sectional approach, as we do not intended to continuously run these tests over time to see how the feedback alters. Now that the core of the onion has been reached it is time to go into further detail about the specific techniques and procedures that will be used while developing and testing in this project. These approaches will be further outlines in the sections below.

3.2 Agile

For this project we will use an iterative work flow approach, which consists of sprints in which design, implementation, and testing is done simultaneously. The

method is called "Agile" and it is often used in it development as it is an efficient work flow method that allows the developers to understand the impact of the changes that are being made to the product. The agile method will be used in the design and implementing phase, with internal testing and a usability test near the end of the initial development phase. In comparison to other work flow methods such as the "water fall" method, the agile approach allows for a more dynamic relationship between the development of the product and the final end goal of the product. Furthermore, the agile approach allows for the developers to catch more mishaps and faults before the users get to try the product. The sprints will be run continuously until the final test has been completed, thereby creating a cycle of development [9].

3.3 Usability test

In this project, we will preform a usability test in the early stages of testing. A usability test is a test that examines and evaluates how well a product is performing and how easy and well functioning a user thinks the product is. For this project a usability test will be done to test if the product is functioning as intended, if the users think it is intuitive to use and if there are any outstanding issues with the program or setup. Furthermore, a usability test can be used to ensure that we are focusing on the right aspects of the product and development. The usability test will be preformed using some other methods, such a a likert scale, observations and the speak out loud approach [25]. This will be gone over in greater detail in the design of the tests.

3.3.1 Likert Scale

During the usability test a quantitative test will be run. A part of this test will be a questionnaire with a likert scale. This will be done in order to better understand how users feel about the product, how it preforms and how easy it is to use. The test participants will be asked a range of questions that will help us determine what is done well and where there are potential faults. Overall, the likert scale will consist of both negative and positively charged questions, that should in a best chase scenario, produce an average, which is the middle value of the ranking system. The ranking system should be uniform in amount for each of the questions. For this project, a 7 point scale will be used. This will be done in order to ensure accuracy in opinion and the rankings will range from "Strongly agree" to "Strongly disagree". The questions will be posed as statements, such as "The program accurately tracked my movements" [25].

Chapter 4

Production

In this project we will be working with productions. A production consists of a design, implementation and evaluations phase. Furthermore, each production will have an overview of what is made in the production as well as what we were unable to make. For the first three iterations testing will be done internally due to covid-19. Although, plans had been made to test on actual people in a non covid-19 world.

The following sections of the production chapters will be split into four iteration. From each iteration, considerations will be noted and taking into account for the next iteration and discussion. The final iteration will include the testing that has been preformed and it will seek to answer the FPS 2.7.1.

The list of requirements from the design requirements section 2.7.2, will be used as the guide line to fulfill the FPS and making the product. However, each of the requirements will be introduced accordingly into the iteration. In order to get a visual concept of when each goal will be introduced into the iteration, a flowchart was made.

The flowchart (figure 4.1) shows the elements that should be in the final product and they are shown as a progression system, so that it is easy to see how far the product has come between each iteration. When an element is being worked on it is coloured in with green, while if and element is not yet being worked on it will be shown in red.

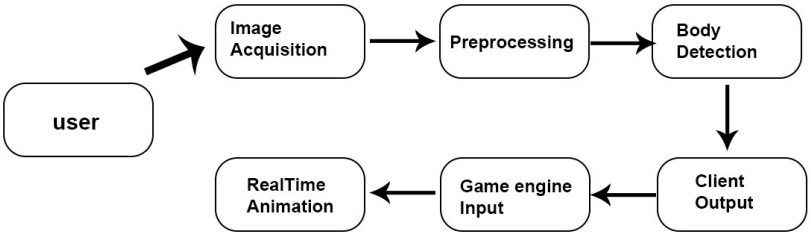

Figure 4.1: Flowchart of the iterations

4.1 First Iteration

For the first iteration, the goal is to setup the LiDAR camera and use image processing to find the body parts as can be seen in figure 4.2.

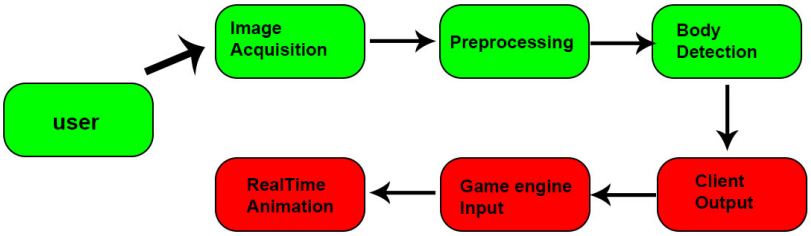

Figure 4.2: Goals for first iteration of the flowchart

This will help us learn how to use and work with the LiDAR camera and it will introduce the first functional requirement to the iteration 2.7.2.

4.1.1 Design

Initially, we want to make general body tracking, so that each body part can be detected. For this, the camera of choice is the Intel RealSense L515, since it was the cheapest solution that we found in the analysis (2.4.2). For body tracking, OpenCV is used since we have previous experience with the library for image processing.

The RealSense has a LiDAR sensor build-it, which shows the depth in a color range from blue to red. Through image processing, it is possible to isolate the blue colors which leaves the body untouched. That process uses a mask that is applied to the RGB camera output which will help the OpenCV to better detect the body parts at that point 2.4.2. By following this procedure, the functional requirement for tracking the body movement will be fulfilled. Additionally the non-functional requirements like using LiDAR technology, OpenCV and markerless technology will also be fulfilled 2.7.2.

4.1.2 Implementation

This section will go through the steps that were taken to implementing what have been described in the design chapter of the first iteration.

Initial Attempt

After getting hold of the Real Sense L515, an attempt to use the built in Unity plugin has failed. After following the step by step instruction to install it, a critical file called "realsense.unitypackage" wasn't created, which led to the unity project not having access to the intel real sense libraries, meaning the scene could not load in Unity. After trying this on two computers and in different hard discs within one computer, the decision was made to use OpenCV with Python since it is another tool that could be used to feed the same data to unity by creating a python script. Unfortunately the python library isn't up to date with the latest Real Sense L515 and still had many errors that have not been fixed. This put us in a position where the critical decision of moving on and working with an RGB camera instead, while simultaneously trying to find solutions to the technical errors we have encountered.

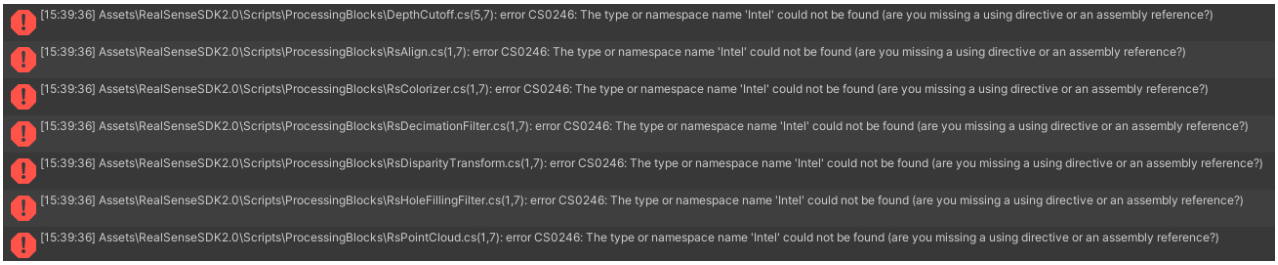

Figure 4.3: Unity Errors

For the next iteration, the requirements from this iteration will stay the same since they were not fulfilled. The requirement were 'Ability to track body movement' from the functional requirements and from the non-funtional requirements were 'utilise LiDAR/RGB technology', 'using marker-less technology' and 'utilising image processing library OpenCV' (2.7.2) as can be seen in figure 4.2.

4.2 Second Iteration

4.2.1 Design

In the second iteration, we will start working with an RGB camera, which is the most regular type of camera. This is a less accurate version than the LiDAR camera for our purposes, but as we were unable to make the LiDAR camera work, the RGB

camera was seen as the second best option for availability, price and ease of use. The goal of this iteration is the same as the previous version, which was to get a functioning body detection ready. This will be accomplished using a combination of Python and C#. Python was chosen due our previous experience with it along with the library called OpenCV. Since we have chosen to work in the game engine Unity, The default code language for this is the C#.

For this iteration we will use a regular RGB webcam, since this is something almost everyone has as a standalone or built-in on a laptop, however the webcam that will be used will follow similar technical requirements from the LiDAR camera such as the resolution and the frame rate.

In order to detect the human body with the information from the RGB camera, we will be using AI. This is due to its accuracy when detecting body parts while keeping performance stable. For this iteration Tensorflow will be used, which is a Machine Learning algorithm that is made to detect body parts. As mentioned in the analysis (2.5), AI can improve the workflow of how the program detects body parts and therefore it was chosen to be used.

4.2.2 Implementation

The first task is to get the output of the webcam into the python script. This is done through the OpenCV library, where a while loop will check if the camera can be read. If the camera can not be read, it will break the while loop and exit the program as seen in figure 4.4.

```
36  cap = cv.VideoCapture(args.input if args.input else 0)
37
38
39  if not cap.isOpened():
40      print("Cannot open camera")
41      exit()
42
43  while True:
44      # openCV section
45      ret, frame = cap.read()
46      # if frame is read correctly ret is True
47      if not ret:
48          print("Can't receive frame (stream end?). Exiting ...")
49          break
50
```

Figure 4.4: Reading the cam via OpenCV

To get the feedback from the camera, the command "cv.imshow" can be used to create a window with the camera feedback. This window is also going to show where the joints and limbs are, by using the Tensorflow Human Pose Estimation machine learning algorithm.

This pose estimation needs to know what body parts needs to be found and the connection between them. The Tensorflow Human Pose Estimation can detect up to 17 body parts and can create the skeleton based of those parts. The body parts and their connection can be seen in figure 4.5.

```
BODY_PARTS = { "Nose": 0, "Neck": 1, "RShoulder": 2, "RElbow": 3, "RWrist": 4,
               "LShoulder": 5, "LElbow": 6, "LWrist": 7, "RHip": 8, "RKnee": 9,
               "RAnkle": 10, "LHip": 11, "LKnee": 12, "LAnkle": 13, "REye": 14,
               "LEye": 15, "REar": 16, "LEar": 17, "Background": 18 }

POSE_PAIRS = [ ["Neck", "RShoulder"], ["Neck", "LShoulder"], ["RShoulder", "RElbow"],
               ["RElbow", "RWrist"], ["LShoulder", "LElbow"], ["LElbow", "LWrist"],
               ["Neck", "RHip"], ["RHip", "RKnee"], ["RKnee", "RAnkle"], ["Neck", "LHip"],
               ["LHip", "LKnee"], ["LKnee", "LAnkle"], ["Neck", "Nose"], ["Nose", "REye"],
               ["REye", "REar"], ["Nose", "LEye"], ["LEye", "LEar"] ]
```

Figure 4.5: Body parts and their connection

By using the heatmap and blob detection from the Tensorflow algorithm, it is possible to get the estimated area of where the body part would be. In the first for loop, the frame will be saved and split into a 15x15x17 grid. Each section of the grid will then be run through the Tensorflow algorithm to detect areas that resembles a body part. Then each area will be put together to create the body itself and then highlight the body parts' positions. Afterward, the body parts' positions will be save onto an array as can be seen in figure 4.6

```
    for i in range(len(BODY_PARTS)):
        # Slice heatmap of corresponging body's part.
        heatMap = out[0, i, :, :]

        # Originally, we try to find all the local maximums. To simplify a sample
        # we just find a global one. However only a single pose at the same time
        # could be detected this way.
        _, conf, _, point = cv.minMaxLoc(heatMap)
        x = (frameWidth * point[0]) / out.shape[3]
        y = (frameHeight * point[1]) / out.shape[2]
        # Add a point if it's confidence is higher than threshold.
        points.append((int(x), int(y)) if conf > args.thr else None)

```

Figure 4.6: Splits the heatmap and gets the position of the body parts

These positions will then be used in the second for loop where the joint and limbs will be drawn onto of the camera output as seen in figure 4.7.

```
76      for pair in POSE_PAIRS:
77          partFrom = pair[0]
78          partTo = pair[1]
79          assert(partFrom in BODY_PARTS)
80          assert(partTo in BODY_PARTS)
81
82          idFrom = BODY_PARTS[partFrom]
83          idTo = BODY_PARTS[partTo]
84
85          if points[idFrom] and points[idTo]:
86              cv.line(frame, points[idFrom], points[idTo], (0, 255, 0), 3)
87              cv.ellipse(frame, points[idFrom], (3, 3), 0, 0, 360, (0, 0, 255), cv.FILLED)
88              cv.ellipse(frame, points[idTo], (3, 3), 0, 0, 360, (0, 0, 255), cv.FILLED)
89
```

Figure 4.7: skeleton drawn on person in camera view

The joints will take the position of the estimated point between two body parts where it thinks a joint is located and it is then drawn as an ellipse on that given location. the body parts themselves will be presented as a line between each joint as seen in figure 4.8.

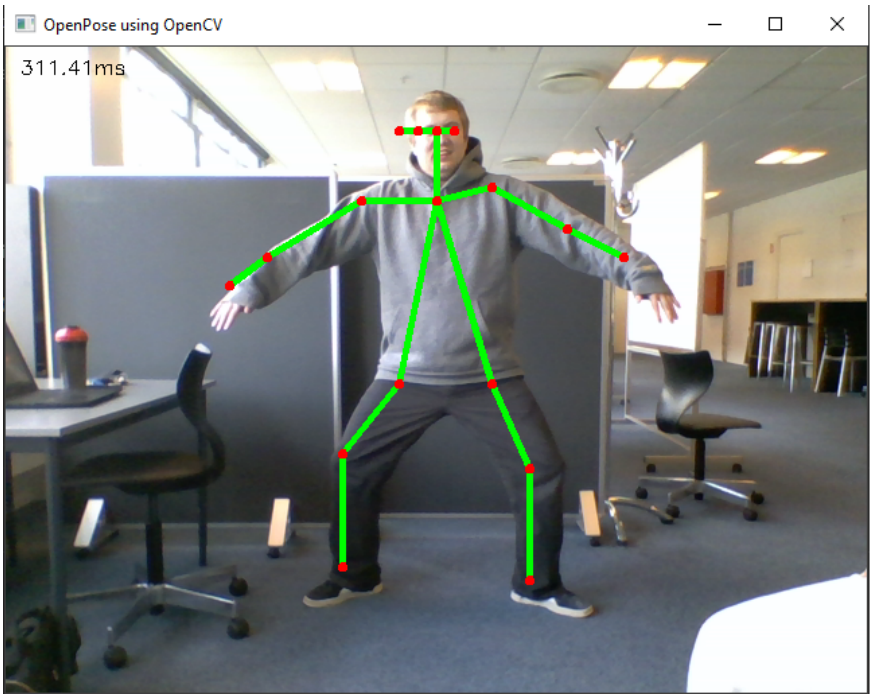

Figure 4.8: Camera view with skeleton

With this detection of body parts and joints through a RGB camera, the goals for this iteration has been met. The next iteration will relate to how the location of the joints can be sent to a game engine on the same computer. This is going to fulfill

the following design requirements (2.7.2):

- Translating the body movement into data that can be used in Unity

Since the data is currently only available in the python script, the data is not being transferred to Unity, so it can be used as an input for further iterations.

4.3 Third Iteration

4.3.1 Design

For the third iteration, the goal will be to send the information from the RGB camera with the body tracking to a game engine. This will build upon the second iteration and fulfill more of the requirements (2.7.2). The next goal will be to send the information and obtain it in a game engine as an input as can be see in figure 4.9

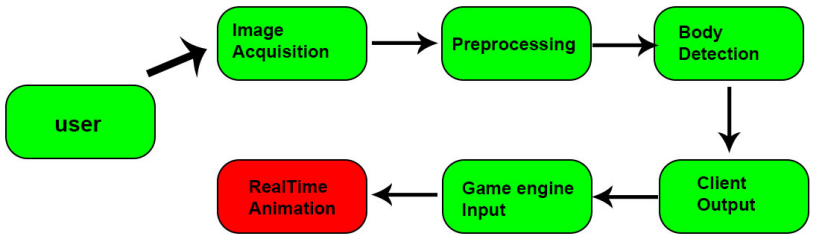

Figure 4.9: Goals for second iteration of the flowchart

The first step will be to send the data to a game engine. This can be done by creating a client-server structure connection to a game engine with the python program. Since the OpenCV code is all-ready written in python, the client part of the code can be added here as well. The server side will be added to the game engine since it will receive the data from the client. The game engine will be Unity since we are the most experience with it. To create this client-server connection, a bit of socket programming, will be used to convert the python output of joints into string inputs that Unity can read.

Each body part will get a pointer which contains the coordinates of that body part and these pointers will be placed near the joints. By doing that, it is easier to track the position of the pointer and use them as reference on the Unity side. By doing these steps, the following design requirements will be, translating the body movement into data that be used in Unity.

4.3.2 Implementation

First a socket connection is created between the Python Script, which is the client in this case, and the Unity Project which is the server. The script then feeds Unity constantly with the two dimensional position of the joints as a string. In the Client, the library called socket is imported and therefore the socket connection between client and server can be made. First the IP address and port needs to be defined so both sides know where to look for the connection. In this case, the IP address will be '127.0.0.1" to be local and the port will be '25001' is due to its common use in communication programming.

```
98      s = socket.socket()
99      s.connect((host, port))
100     for x in range(len(POSE_PAIRS)):
101         pose = x
102         joints = points[x]
103         combinedString = str(pose)+str(joints) +'|'
104         print(combinedString)
105         s.sendall(combinedString.encode())
106
107     s.close()
```

Figure 4.10: Sending the data locally in python

Then using the function 'connect' from the socket library, the connection for the IP address and port has been established. The next step would be to get the data and send it via the given connection. Since there are multiple positions of the joints, a for loop was made to send each of the positions. Each position will be translated to a string where the joint's ID and its position for x and y will be encoded into a uft-8 which makes the string readable for the unity server. Lastly, it will close the connection and reconnect once it reaches the same section of the code again as seen in the figure 4.10. The next section will go into detail on how the server works, which is where the data is received in unity.

The server on the unity side will start out similar to the client with defining the IP address and the port. Instead of sending information, it will instead be listening through the connection, so it can pick up the data from the client. Since the server is going to receive a string with the data of the ID of the joints and its position, a vector2d array was made to save the received data. The ID would be the entry in the vector2d array and the x and y coordinates will be placed in the x and y of the array. But, to get the ID, x and y coordinates of the joint and the string that are being received, needs to be split. This is done through a double for loop, where

the first loop will iterate through each string and the second loop through each char of the string. The reason for iterating each char is due to the string's setup. when the ID of the joint is above 10 it means the string will be one char longer, which can resolve in wrong data. Therefore, a check is made to see if the string is a certain length and afterwards assign the data into local variables. First the Id, x and y coordinates will be saved as a string which then will be converted into an integer. These coordinates can then be saved into the public vector2d array where the ID will be the joints' ID as seen in the figure 4.11.

```
119                 testsubject = new char[singlebodyPart.Length];
120                 testsubject = singlebodyPart[j].ToCharArray();
121                 if (testsubject.Length == 12)
122                 {
123                     int xvalue;
124                     int yvalue;
125                     int id;
126                     string xstring = testsubject[3].ToString() + testsubject[4].ToString() + testsubject[5].ToString();
127                     string ystring = testsubject[8].ToString() + testsubject[9].ToString() + testsubject[10].ToString();
128                     string idstring = testsubject[0].ToString() + testsubject[1].ToString();
129 
130                     int.TryParse(idstring, out id);
131                     int.TryParse(xstring, out xvalue);
132                     int.TryParse(ystring, out yvalue);
133                     positionForJoints[id].x = xvalue;
134                     positionForJoints[id].y = yvalue;
135                 }
136                 else if ( testsubject.Length == 11)
137                 {
138                     int xvalue;
139                     int yvalue;
140                     int id;
141                     string xstring = testsubject[2].ToString() + testsubject[3].ToString() + testsubject[4].ToString();
142                     string ystring = testsubject[7].ToString() + testsubject[8].ToString() + testsubject[9].ToString();
143                     string idstring = testsubject[0].ToString();
144                     int.TryParse(idstring, out id);
145                     int.TryParse(ystring, out yvalue);
146                     int.TryParse(xstring, out xvalue);
147                     positionForJoints[id].x = xvalue;
148                     positionForJoints[id].y = yvalue;
149                 }
```

Figure 4.11: Double for loop of managing the data

Now the data has been sent from the client and has been processed so it is ready to be used on a rig for realtime animation in the next iteration. The next iteration is going to focus on the realtime animation and therefore the last functional requirement which is 'Ability to animate in real time' (2.7.2).

4.4 Fourth Iteration

4.4.1 Design

In the final iteration of the product development we will focus on the realtime animation step. This is the final piece of our product. It will start with attempting to apply the data onto the rig. This would results in movement that is a replication of the user of the program.

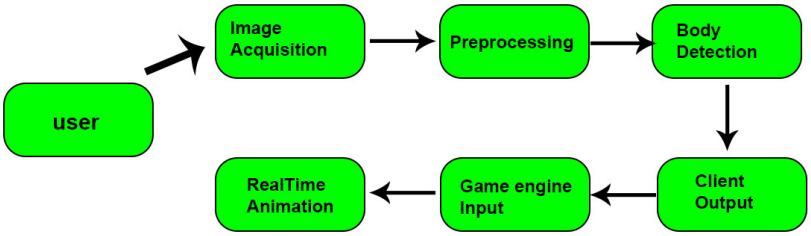

Figure 4.12: Goals for final iteration of the flowchart

The ideal method would be to apply the raw data from the client to the joints in the rig since that would change the position of the joints. Once the rig is corresponding with the client output, the 'Ability to animate in real time' requirement will be met and therefore all the design requirements have been met (2.7.2).

As an additional feature, we wanted to add a chromakey effect to the background of the scene so that the user can change it depending on the user's needs. This would allow the user to get a unicolored background that they can use as a green/blue screen in case they want to record the character. Exemplary, while shooting a video or live streaming it. This feature will allow for more flexibility and let the user be more creative knowing they can use the character in different settings.

4.4.2 Initial Attempt of fourth iteration

With the data saved in Unity, we wanted to test if the positions of the joints were equal to the client side. For that test, a simple instantiation of an image in the user interface with the joint's coordinates was presented in the form of white squares. Once we saw that the server placement of the joints was equal to the client side. This showed that the data worked as individual objects and the next step would be to apply the data onto a rig. Applying the data on the rig resulted in the rig's joints to be sent very far away from its origin. This is due to the data being set as world position and it is not being converted to local space since each joint would be a child object. To try to fix this problem, we tried to detach the joints from the rig, then set their position and reattach them to their parents in the rig. That resulted in the same issue as before and a new method was attempted instead. The method this time would be to use the unity function "Transform.InverseTransformPoint" since this function will convert world space position to local space position.

```
19          Vector2 inversPos = server.positionForJoints[0];
20          arrayOfJoints[0].transform.position = server.positionForJoints[0];
21          arrayOfJoints[0].transform.InverseTransformPoint(arrayOfJoints[0].transform.position);
22
23          /* float newshoulderX = remap(server.positionForJoints[0].x,0, 368, 0,3);
24           float newshouldery = remap(server.positionForJoints[0].y, 0, 368, 0,3);
25
26           float newelbowX = remap(server.positionForJoints[2].x, 0, 368, 0, 3);
27           float newelbowy = remap(server.positionForJoints[2].x, 0, 368, 0, 3);
28
29           float newwristx = remap(server.positionForJoints[3].y, 0, 368, 0, 3);
30           float newwristy = remap(server.positionForJoints[3].y, 0, 368, 0, 3);*/
31
32          arrayOfJoints[0].transform.parent = null;
33          arrayOfJoints[0].transform.position = server.positionForJoints[0];
34          arrayOfJoints[0].transform.parent = parent[0];
35          arrayOfJoints[0].transform.position = server.positionForJoints[0];
36
37          arrayOfJoints[1].transform.parent = null;
38          arrayOfJoints[1].transform.position = server.positionForJoints[2];
39          arrayOfJoints[1].transform.parent = parent[1];
40          arrayOfJoints[1].transform.position = server.positionForJoints[2];
41
42          arrayOfJoints[2].transform.parent = null;
43          arrayOfJoints[2].transform.position = server.positionForJoints[3];
44          arrayOfJoints[2].transform.parent = parent[2];
45          arrayOfJoints[2].transform.position = server.positionForJoints[3];*/
46
47          //arrayOfJoints[3].transform.position = new Vector3(0,0,0);
48
49          /*for (int i = 0; i < joints.Length; i++)
50          {
51              if (joints[i].name != "shoulder")
52              {
53                  joints[i].transform.parent = null;
54              }
55
56              joints[i].transform.position = newpos[i];
57
58              if (joints[i].name != "shoulder")
59              {
60                  joints[i].transform.parent = parents[i];
61                  joints[i].transform.localPosition = newpos[i];
62              }
63
64          }*/
```

Figure 4.13: Troubleshoot attempt

By using this method, it should have made the joints' position in world space to local but unfortunately it resulted in similar issues as before as the code shows in figure 4.13. While looking for ways to fix the displacement of joints in the rig, we came across a GitHub repository from Digital- Standard Co., Ltd. that is called "ThreeDPoseUnityBarracuda" [7]. This repository has a better tracking system, it is able to apply the tracked joints onto a rig and it is doing this inside of unity. It was decided that we would use this repository for the final iteration since it was very similar to the FPS that we had set and it was ultimately an optimised version of what we had attempted to implement.

4.4.3 Implementation

The repository [7] from Digital- Standard Co., Ltd. uses the Barracuda library from Unity which is a lightweight cross-platform neural network interface. It can take an open neural network exchange (ONNX) model as its pipeline for learning. In this case, the Tensorflow model for pose estimation was chosen since it is a machine learned model to estimate the pose of a person from the webcam's output.

In the Unity scene, there is an avatar, two UI images along with general objects like the camera and directional light. The first UI image is the general camera feedback and the other UI image is a focus area from the camera feedback. The focus area will be where the detection will happen and once it finds one of the joints, it will start mapping the skeleton from it. This will result in the avatar and the skeleton made from the focus area to match the movement from the user.

As an addition, The background was made to a solid color instead of a sky box. By doing so, it will be easier to chromakey it out if it is needed. Since the avatar can match the background color, extra UI in form of sliders were made to change to background in a RGB setting. That way, the user can change the background to what they need and the type of chromakey they want. The sliders themselves can be seen in figure 4.14.

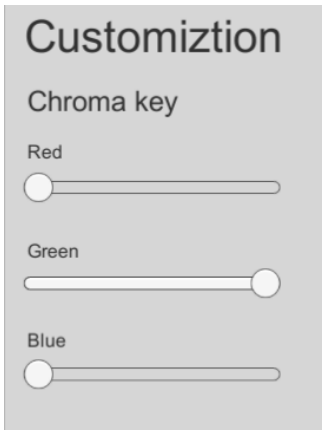

Figure 4.14: Sliders for chromakeys in scene

The switching of character is fairly simple since the two characters in the scene to begin with, have all the components that are required by the 'VNectBarracudaRunner' script. The 'VNectBarracudaRunner' script requires a gameobject that has a 'VNectModel' attached. The 'VNectModel' is a script that takes the trained data from the ONNX model in 'VNectBarracudaRunnner' and applies it to the 3D character in the scene, giving the character realtime animation. Both of these scripts requires the 3D character to have a rig with Unity humanoid avatar to apply the

data and perform the realtime animation.

This will fulfill the last requirements from the design requirements and therefore it can be tested and be evaluated upon.

Chapter 5

Evaluation

5.1 Initial testing

During development, we will be preforming continuous internal tests to ensure that the technology continues to function and that the new additions are properly implemented. This will be done using an iterative approach, where a design plan is made, implementation is done and then a technical test is preformed. From here the test reveals if there are any mistakes, obstacles or missing components that should be developed. The cycle then continues until a functioning version is available for the usability test. An outline of how that cycle functions was described in the beginning of the design chapter 4.1.1. The internal tests will be preformed on group members, by group members in their respective homes.

5.2 Testing In A Non-Covid19 World

If testing could have been done without the fear of Covid-19 and the restrictions set by the danish government, some elements of testing would have been done differently. Ultimately a different form of testing would have been preformed as well as additional tests. In this section we will explain how testing would have been preformed if Covid-19 was not a factor.

Initially, proof of concept testing would have been preformed, along with initial design concepts. This would have been done using low fidelity prototyping using paper cutouts, displaying design concepts for UI.

As a second test we would preform a usability test, which would involve the test participants doing the entire process of booting the program from their own devices and setting up the program. This would help us gain a better understanding

of the "ease of use" element, while also giving us more insight into the UI needs of our test participants. Otherwise, the test would include the same questions from the current installment of the usability test. This kind test would then aim to receive approximately 30 participants, as that would be a large enough number of participants for qualitative responses and data for data collection.

With the knowledge gained from those tests we would then expand further on the comparison test, which is done in this project, so that we could definitively answer the final problem statement. This would involve having participants trying both versions, and comparing the two measurable variables. How easy is it to use? and How well does it preform? As an addition to this test a "TAM" Technology Acceptance Model, would be preformed. A TAM test can provide insight to developers about the attitude of users towards the product as well as, their opinions and limitations related to the product. Furthermore, in relation to this project it can provide information such as, usefulness, ease of use, and the likeliness that users would use this product [4].

Alternatively, we would have reached out to professionals withing the motion capture business and ask for their expert opinions. This would involve showing them the current product and receiving feedback from them and then compare this program with their product.

Unfortunately, testing under these conditions was not an option so we had to adapt to these conditions and attempt to keep the quality of the tests to a higher standard. Therefore, the tests that were preformed will involve as little contact to the participants as possible, while simultaneously trying to receive their valuable information and insight.

5.3 First test

The goal for this test is to receive initial feedback on our first prototype and concept. The test will involve the users attempting to use the program and evaluating how well it functions in their opinion. This will be done by having the participants evaluate the experience using a questionnaire which contains a range of questions and a likert scale. The questions will relate to the ease of use in regard to the program, the quality of the program and participants feedback on the program overall.

5.3.1 Setup

In this section, the setup of the test will be gone over. The test setup is made to ensure that the test participants have as close to the same conditions as possible while testing. This is done in order to limit the amount of bias and interference with the

test and test results. Although, it was considered to make the test an online test, so that people could test from their homes. Ultimately, the logistics of ensuring that the test participants had similar and properly setup testing conditions was to difficult. This would have included lighting, space, camera access, knowledge of operating Unity and asking participants to focus on the testing.

In between each test we will make sure to the wipe down the equipment and keyboard before the next participant tests to limit the risk of spreading covid-19. Furthermore, the tests will take place at two different locations, one will be at a designated room on campus where in which to get access, participants will have to have taken a covid-19 negative test before being able to enter. The second location will be at a students apartment where a testing environment is set up, following the same covid-19 safety procedures. To test, participants will be interacting with a computer with a webcam attached. The camera will be facing a wall that participants will stand against to be within the camera view. Furthermore, the lighting in the room should be even or placed behind the camera illuminating the participant.

5.3.2 Procedure

The testing will be done over the span of two days. On the first day a group member will go to a student organised event where other students can help each other test. On the second day the other members of the group will go to school and try to find participants for the test. The first step is to setup the room and calibrate the lights and environment. The potential test participants will then be approached and asked to help us test. If they are willing to test they will be brought to the setup, where we will ask them to sign a consent form. They will also be told that the test is for motion tracking, but that they will not be recorded and their movement data will not be stored. The participants will then be asked to answer some preliminary questions on the computer, such as, "Have you tried working with motion capture before". At this point, The participant will be asked to start the program and a group member will make sure it is done properly. The participant should then start moving and try testing the limitations of the prototype. After 5 minutes the participant will be asked to answer the final portion of the questionnaire, which involves a few open questions and a likert scale. When they are finished with the questionnaire, they are done testing and we will say thank you. A group member will then wipe down, laptop and other equipment. This will be done again, until we have enough participants for the test (approximately 10 due to circumstances).

5.3.3 Participants

The test participants will be sampled using convenience sampling, which in this case means that the participants will be other students that are available for testing. This may lead to increased bias, but given the circumstances, and the inconvenience of having people testing from their own homes with varying conditions and setups, this was the option that was best available to us. Overall, We are aiming to receive around 10 participants for this test.

5.3.4 Results of the usability test

To see all the results go to the appendix 8, in this section we will go over the most important results and feedback received from participants. Based on the knowledge gained we will work to improve the product. All the test results for the usability test was gathered on the days of 10th-11th of may 2021, between the hours of 12-16 and 9-10:30.

In total we got ten participants, seven on the first day and three on the second. Out of these ten participants six had no prior experience with motion capture before and none of the participants felt experienced with motion capture. Of the four that had tried motion capture before one had used a Rokoko suit, while the others had used the Kinect and tried it with Just Dance. In the second part of the questionnaire nine participants responded yes to the question 'Is the lighting even in the room?', while one participant responded that it was "Decent". All ten responded 'Yes' to the question 'Do you fit in the frame of the camera?'. And for the final preliminary question, the seven participants from the first day of testing responded 'no' to the question 'Is the background uniform or same coloured?', while two from the second day responded 'yes' and one 'mostly'.

In the final section of the questionnaire, the participants were asked open questions and to respond to a likert scale. The first question they were asked was 'In your own words, what did you just experience/test?'. To this the participants responded using words such as "motion controlling, motion tracking, motion capture", while one participant responded "How the model on the screen reacted to my own movements in front of the camera" and another "I tested a motion capture avatar to see how well an how nuanced it matched my movements, and the limitations of the animation.". This feedback is nice to have it confirms that they understood the purpose of the program.

As a part of the questionnaire eight likert items were made. These likert items were posed as statements that the participants had to choose the level of agreement they had. "1" represented "Strongly disagree" while "7" represented "Strongly agree". On two of the questions we wanted to know how the participants felt about the

accuracy of the product. Here the questions were in one case negatively charged, while in another, it was positively charged. Another difference was the words "software" and "program" which are the same thing. But, to the surprise of us the responses do not seem to overlap. This can be seen in 5.1 & 5.2.

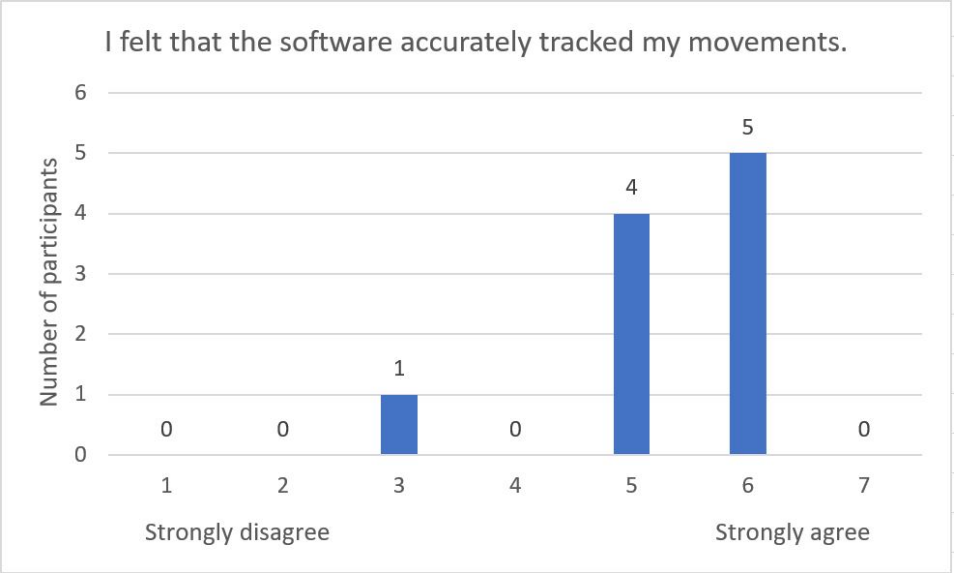

Figure 5.1: How participants responded to the question 'I felt that the software accurately tracked my movements.'

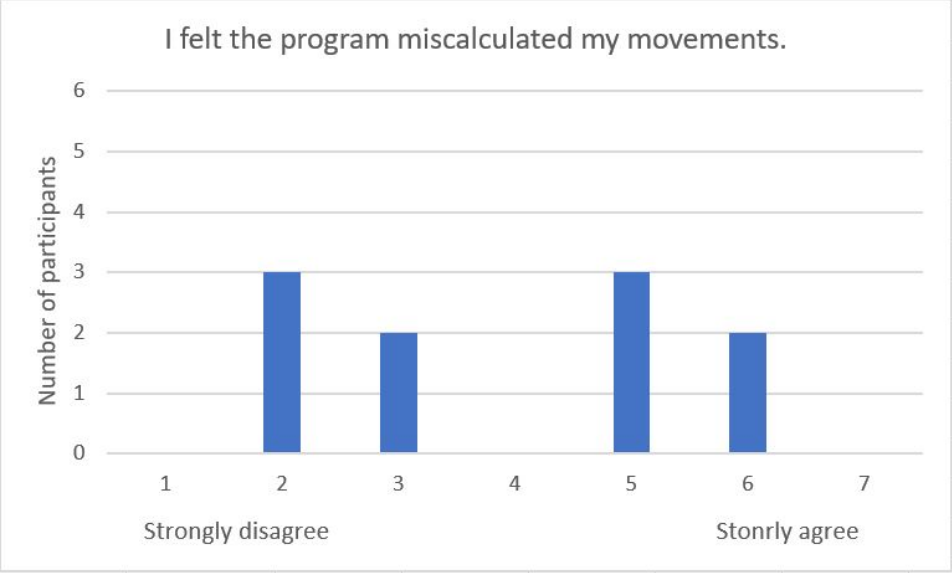

Figure 5.2: How participants responded to the question 'I felt the program miscalculated my movements.'

In another likert item, participants were given the statement "felt that the program was easy to use" to which most participants responded that they "strongly agree". This can be seen in figure 5.3. This is a nice result as this is linked to our FPS and design requirements 2.7.1 & 2.7.2.

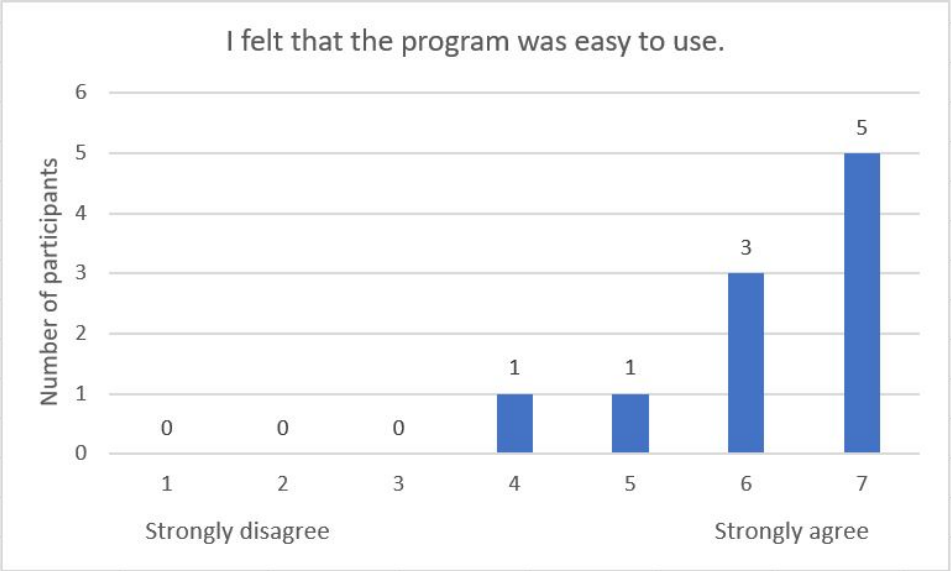

Figure 5.3: How participants responded to the question 'I felt that the program was easy to use.'

When adding together the responses from the likert scale and calculating the mean, a relatively low score is found at 1,4285, which is an indication that there are some areas which are lacking. This was mainly for questions that related to performance, which can be attributed to lack of sufficient testing hardware. Overall, it can be said that participants tended to score negatively charged question closer to "1" while positively charged questions rarely received a "7". Furthermore, the questions consisted of three positively charged questions, while five were negatively charged.

When participants were asked the question 'If you were looking for a "cheap", "easy to use", "home setup" motion capture solution, would you be satisfied with the current version of the software?' To this participants responded four 'yes', four 'no', and two responding that with better precision they would. Those that said 'no', were then asked 'If no, what were the areas that were lacking?', to which they responded; "Accuracy and responsiveness", "Not really a user friendly interface right now" and overall about the accuracy and precision. Although, this was a know problem when running the tests as the sub optimal performance of the computers the test was being run on. It should also be mentioned, that participants could have interpreted this question different from each other. Some may

have thought of a professional setting, while other may have thought of it as recreational or personal use.

We then asked the participants 'Which features would improve your user experience?', to which participants had a varied responds, some which are out of our current scope ("finger tracking"), while others relate to the lack of proper testing equipment (generation of Graphics card). Participants also wanted like to see upgrades in UI, and the ability to change character to try other models.

5.4 Comparison test

As a secondary test to the usability test, a comparison test will be done. This test will compare the current software to the state of the art Rokoko suit. The test will involve doing poses and movements to test the accuracy of the two products and determining how well they preform comparatively. The Rokoko suit, is borrowed from the company called Amunet, which have agreed to borrow us their suit for our testing purposes[1].

5.4.1 Setup

As a part of this test, we will need a room with enough space to setup the Rokoko suit, secondary we will need a plain wall with sufficient space for the RBG camera to record the tester. We will also need a room with even lighting and minimal interference for the Rokoko suits signal sending, which is being done to ensure the best quality results. Finally, we need to decide on the poses and movements for the test. With this in order the test can proceed [20].

5.4.2 Procedure

We will first setup the equipment, room and calibrate it to the testers measurements. With both systems running the test person will then do a series of movements and poses to test the capabilities of the programs. This test will be done in the living room of one of the group members, where we have the ability to clear the floor space and have a neutral background.

5.4.3 Results of the comparison test

After running the test and recording the results, we went through the footage to compare the poses of the actor with the poses of both Rokoko Studio's skeleton and the developed prototype's skeleton. This has show that there are some differences between the products, as the Rokoko suit captures complex movements better. We

[1] Amunet studio website: https://www.amunetstudio.com/ & AAU CPH equipment inventory

have followed a couple of poses that have been done by researchers that tackled 3D Human poses recovery based on RGB Video Sequences [20]. A couple of examples will be shown in this paragraph as the rest of the footage will be provided in the appendix 8.

The poses were performed as following:

1. **Performing a simple move slow and then fast**: In this step a simple wave movement was done slowly and then fast, which both motion capture tools have shown to pick up with no problem as shown in the figure 5.4.



Figure 5.4: First movement comparison between our prototype and Rokoko's Smartsuit Pro

2. **Walking forward, backwards and to both sides while looking forward**: In this step, both tools have performed similarly except when the actor moved forward his head and feet (ankles) moved out of the camera frame which led to the model losing calibration of the head and feet.
3. **Performing a 360 degrees rotation vertically**: In this step both the Rokoko Suit and the developed prototype accurately tracked the body's movement rotating around itself within the program.
4. **Performing a hand cross where both palms touch the opposite hand's elbow**: In this step, the developed prototype showed to recognise the hand cross better than the Rokoko Suit as showen in the figure 5.5.



Figure 5.5: Fourth movement comparison between our prototype and Rokoko's Smartsuit Pro

5. **Performing a tying shoes position while facing forward**: In this step the movement is more complex showing that the developed prototype had a harder time tracking the joints as for the Rokoko Suit performed it without any issue.



Figure 5.6: Fourth movement comparison between our prototype and Rokoko's Smartsuit Pro

6. **Performing a jump without moving the position**: This step again shows that both the tools have tracked the jumping motion, but when the actor was jumping his head and forearms left the camera frame. This led to a slight glitch in the developed prototype's rig where the body got squished.
7. **Performing random complex movement**: This step is where it shows that

complex movement isn't picked up well for our prototype. When the actor was walking around freely, both the Rokoko suit and the developed prototype functioned well. Although, when he tried a hand stand or a frog position the developed prototype shows that it cannot predict the move and by losing some joint's position, it performs some random and unpredictable moves as showen in the figure 5.7.



Figure 5.7: Tenth movement comparison between our prototype and Rokoko's Smartsuit Pro

Chapter 6

Discussion

In this chapter we will discuss the choices that have been made during this project, the process in which we developed and implemented the product and the results we received.

6.1 Evaluation discussion

In this section we will discuss the results gathered from the two tests we ran. First the usability test then the comparison test.

Usability test

While testing we knew that the testing would not be at the optimal running capacity as the tests were done on outdated graphics cards or integrated graphics card on laptops with generations old cards. This was not explained to the participants as we saw no use. Although, if they asked we would let the participant know that on newer graphics card this program would run far greater. Many of the responses related to these performance issues, but as mentioned in the repository the optimal range for running these test would be to use the nvidia 2080 or newer [7]. Other participants wanted to see a better UI or have more options, which is something that was considered. As we currently were looking at other issues with the program, the UI became a nice to have issue compared to other pressing issues, such as performance and accuracy.

Some of the faults and mistakes we made while testing could also have influenced the performance. These were things like not having a simple background while testing and making sure that the participants were without any distractions other than testing the product.

As for the participants, we could have also improved here, with more participants and a larger variety of people in regards to age and occupational background. This is due to us only testing on fellow Medialogy students, with prior knowledge of related project work. This also creates a high bias as the participants are co-students that are also looking to test. Although, it can also be said that their bias could lead them to be more critical of the product, as well as more positive.

In the first test, we would like to discuss, how the questions could have been posed better. First off, the likert scale could have been more balanced and evenly divided in regards to positively and negatively charged questions. Secondly, the questions could have been more specific, such as "I found the UI was easy to read and navigate.". This could have ultimately helped us better triangulate the issues that our participants related to.

Overall the test went well, as the results indicated that participants were happy with the performance and the overall accuracy, though with notable faults. The participants were also overall happy with the ease of use, though here too, the participants would have liked to see more features and quality of life upgrades. This including, a recording button to store the movement sequence within the program. Other participants wanted more characters, which is available, but the character we used was the most optimal as the high number of joints and their position was best suited for the software.

Furthermore, the obstacle of testing with an ongoing global pandemic looking over our heads, made it so the quality of our tests was worsened. This was in part due to the fact that the testing conditions that were available to us was limited but also who we could test on. This also made it difficult to create a hypothesis as the kind of tests we could run did not allow us to either reject or accept any hypothesis as it would be based on limited data and vague information.

6.1.1 Comparison test discussion

As mentioned in the non-covid testing sections 5.2, a more informative and data analytical test would have been run using the TAM test. This would have provided us with data to analyse and quantify, with the goal of affirming the points in our FPS. Specifically, relating to ease of use it could have helped us in finding out if we had a product that can compete with other technologies at a cheaper price, with minimal drop off in performance.

In the results of the comparison test it can be said that when preforming non-complex movements in the optimal environment (good lighting and mono-colored background contrasting the user) our program could keep up with the state of the art Rokoko's Smartsuit Pro and software. While with complex movements the pro-

gram could not keep up, as complex movements were too difficult for the program to recognise what the actor was attempting to do. It can be said that using an RGB-D camera could improve the accuracy of the program as the depth aspect is likely where most of the fault comes from and relate to when doing complex movements. Another aspect to improve the program for preforming complex movements would be to optimize the current AI by feeding it better quality data and using another learning model 2.5. Another way to improve the accuracy would be using motion retageting, since it would make complex movements adapt better to the used rig 2.6.1.

6.1.2 Work flow and planning

In this section we will discuss the work flow and planning for the project. We will discuss the ups, downs and difficulties we encountered while making this project.

In the beginning, we had a slow startup in form of finding materials for the analysis and writing about it, since we were unsure what to look for regarding research. Once we settled on a topic to research, we decided to set up a management system to keep us motivated.

We then setup a "clickup" group, which is a website for management of tasks and meetings. Here we would set up the task for each section of the report as well with the project itself. This allowed the other members to know what was currently being worked on and what the next task would be. We also tried to have daily meetings where we would discuss the next step of the project along with progress on each members current task. Slowly over time, we stopped going to the "clickup" page to update the task there, but we kept informing the others about the tasks and the progress in general. Instead to keep up the management on the clickup website, we should have assigned one person to update the task in there as well as adding new tasks if we felt that they were needed. During this time the group were also focusing on classes and keeping up with homework and projects.

To counter this lack of engagement problem, we kept having the daily meeting when they did not interfere with the other courses. Additionally, we went to the university whenever we were allowed to as a part of motivating each other. This was near the end of the project time, about the last month, where gradual reopening would start. This was a sight of relief as the group would become far more motivated and we were able to get a break through in development. This will be further discussed in the following subsection.

6.1.3 Development discussion

Early in the development process, we decided work with iterations, since they allowed us to work in an agile methods 3.2. This turned out to be a success, because it allowed us to work using checkpoints that we needed to fulfill to complete the final iteration. One of the issue that occurred through this workflow, was prolonging of the said checkpoints. Once we had started on one of the checkpoints, they would extend the checkpoint due to complication with the device, code or a mixture of both. This would result in the deadline for each checkpoint ahead to be pushed. This could have been solved by having sprints instead. With sprints, it would have been possible to limit the time frame for each task and therefore, the task could been discarded if it did not work or if we would not use it.

In the first iteration, we tried to use the Realsense L515 from Intel since the depth sensor would help us with the accuracy and rotation of the body parts. But since the Realsense was fairly new, the documentation for it and support was limited, more so than expected (4.1). Since we wanted to use the Realsense, we spent more time on it, attempting to make it work. This put a pause on the development, but we were able to bounce back quickly when we switched to a RGB camera.

In the fourth iteration, the attempt to assign the positions of the joints from the client onto the rig on the server was more troublesome than we thought. This resulted in more time being spent troubleshooting the rig's joints rather than developing the program. As mentioned in 4.4, we found a repository from Digital-Standard Co., Ltd. [7] that fulfilled our design requirements (2.7.2), which were that it had to be user friendly and perform well. The decision to switch was immediate, since this repository provided us with a way to test the concept and the FPS.

Chapter 7

Conclusion

The project started off with the introduction and motivation for this project, which included creating motion capture for home use both in a professional and free time setting 1. This lead to the formulation of our IPS, which is stated as:

"How can we create a cheap alternative to a motion capture suit for real time animation?".

We wanted to understand what motion capture is, how it can be used and if it was possible to create a cheaper, ease to use, marker-less motion capture solution for real time animation.

Therefore, in order to gain a better understanding of the technology we started researching motion capture and found that there are many different approaches to making motion capture. These approaches would vary in equipment, method of capture and pricing 2.2. At this point, it was decided that working with marker-less tracking was the most interesting approach as it would allow for relative low cost, high ease of use, but accuracy could be a potential downside. We then looked further into the state of the art of motion capture and we were surprised to find quite expensive options with multi layered purchase options to get all the equipment needed for optimal performance 2.3. We then looked at the technologies we would need to use and understand in order to create motion capture using marker-less tracking. This lead to the formulation of our FPS, which was stated as:

"How can a program that utilizes webcam RGB technology to achieve cheaper, markerless and easy to use motion capture be created for real time animation?"

Originally it was not *RGB technology* but *LiDAR technology*, but as we had to go in a different route due to the inability to make the LiDAR equipment function we changed the FPS 2.7.1. Based on the research we also created our design require-

ments, which would stand as the design and implementation principles we would create our product from.

During the development, we ran into some issues and overcame them by adjusting to the circumstances. The main issues related to equipment, and translating the data from our program into 3D space and on a rigged character 4.1 & 4.4. Ultimately, we had to use a Github repository [7] we had found while stuck as this point in development. This repository turned out to have accomplished in broad terms what we had wanted to achieve 4.4.

In order to answer our FPS, we ran some tests, which was a difficult process due to accessibility restrictions to participants and locations for testing. This meant that we had to alternate our plans for testing to a more simple testing structure 5. Ultimately, the testing conditions and results were not what we would have wanted, but in an overall sense we are happy with the response and feedback we received. In the discussion we mention how the testing was not setup and executed to an optimal standard and we believe that is reflected in the results we received. With this in mind and based on the results of our comparison test and the feedback we received from the usability test.

We believe that we created a motion capture program that uses an RBG webcam that stands as a cheap, easy to use, marker-less motion capture system alternative that can perform real time animation. This was done using artificial intelligence and machine learning to create a system that can estimate body position and translate this data into a 3D space where the coordinates are correlated to specific joints on a rig.

Chapter 8

Appendices

Suits	Price (minimum system Connectivity	Frequency (Hz)	Accuracy	components	Body Coverage	engine	additional info	link
Tesla Suit	15000 or 27500 (https://x.Wi-Fi 2.4 ghz	1-300 Hz / per channe 0.05 degrees		10 (Inertial sensor	Base - Body (No hands,	Real-time visualization	Wireless, Battery, biometry and Battery	https://teslasuit.io/
PrioVR	12000 for dev kit 7 msec typical (5 msec in wired mode) to 100 m	200 Hz	Not stated	3 axis gyroscope, 3 axis accelerator, 3 axis magnetometer, and Microcontroller w/ QUERAD™ firmware	19 (17 in suit + 2 for controllers)	- Unreal - Unity		https://priovr.com/product
Holosuit	9995(https://overlurebeat: WiFi/Bluetooth connectivity	unknown	3 degrees/entire solution = + 120 DoF) -upper body -lower body -hands	motion sensors	23 embodied sensors	- Unreal - Unity (https://www.kickstarter.co	haptic feedback	https://www.holosuit.com/
Xsens	4.390€ Wireless 60-80Hz	120 Hz (1000 Hz Alert	Orientation Dynamic Accuracy (Roll/Pitch) Dynamic Accuracy (Yaw) Static Accuracy (Roll/Pitch) Static Accuracy (Yaw) Dynamic Accuracy (Heading) 0.1 deg RMS Static Accuracy (Heading) 0.5 deg RMS	Inertial Measurement Unit (3-axis Gyroscope 3-axis Accelerator 3-axis Magnetometer)	Base MVN Animate, Base MVN Awinda, Pro, Full body Feet	- unreal engine -Unity	More expensive options e	https://www.xsens.com/products/mtw-awinda
Smartsuit Pro	WiFi 2.4 & 5 GHz - 802.11ac Wi-Fi wireless networking Battery: approx. 20 hrs $2,495 Range: Up to 100ft	Not stated	± 1 degrees Dynamic Range: 360 degree Angular Range: +/- 2000 degrees per sec gyroscope: +2000 degree per sec Resolution: 0.03 deg Repeatability: 1 deg	9-DoF Inertial Measurement Unit	19	- Unreal - Unity - Maya - MotionBuilder - Blender<- Motion - Cinema4D - Others		https://www.rokoko.com/products/smartsuit-pro
Perception Neuron	1.499$ WiFi	60 hz with 32 Neurons pitch +1 deg 120 hz with 16 Neurons pitch +2 deg	Inertial Measurement Unit (3-axis Gyroscope 3-axis Accelerator 3-axis Magnetometer)	32	- Unreal - Unity			https://neuronmocap.com/products/perception_neuron
Dynamikyz (only face) [set 4500 op / 250/month	Wired		Camera, gg and software	3	- Unreal - Unity - Maya - Cinema4D	Software, and highly com	https://www.dynamikyz.com/pricing	
OptiTrack	$7999 (for camera) Sdhr, Wired, USB	100 FPS	sub-millimeter accuracy - with an optima body license	6 (suggested minimum)	- Unreal - Unity	Has software for inverse	https://optitrack.com/	
Vicon	Unkown - Personal prices Wired			4			https://www.vicon.com/hardware/cameras/	
Ipisoft	195/695 for licence 2999 for full set 295 for gloves 495 for shoes 195 for hips 695 for markers (10) 495 for face	Wired	30FPS	depth sensor High quality tracking of realtively simple RGB camera	1-4 depth sensor 3-6 RGB cameras	-Unity		https://www.ipisoft.com/store
MoCap solution	Non-connectivity	None	(depends on amount of markers).	10 markers (standard)	full body (with all acces	-Unity	This is strictly a suit and r	https://mocapsolutions.com/

Figure 8.1: List of MoCap systems

Bibliography

[1] Motion Analysis. "Motion capture for animation: the fascinating history behind the movies we know today". In: (Dec 3,2020). URL: https://www.motionanalysis.com/blog/an-evolution-of-motion-capture-the-fascinating-history-behind-the-movies-we-know-today/#:~:text=Motion\%20capture\%20for\%20animation\%20was,to\%20paint\%20over\%20each\%20frame)..

[2] Site author. "Rokoko Review: High Quality Mocap for a Fractin of the Cost". In: (February 20, 2019). URL: https://nofilmschool.com/rokoko-review-high-quality-mocap-fraction-cost.

[3] Frederic Bastide et al. "Automatic gain control (AGC) as an interference assessment tool". In: *Proceedings of the 16th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GPS/GNSS 2003)*. 2003, pp. 2042–2053.

[4] Mohammad Y Chuttur. "Overview of the technology acceptance model: Origins, developments and future directions". In: *Working Papers on Information Systems* 9.37 (2009), pp. 9–37.

[5] Arden Dertat. *Applied Deep Learning - Part 1: Artificial Neural Networks*. Visited 25.03.2021. URL: https://towardsdatascience.com/applied-deep-learning-part-1-artificial-neural-networks-d7834f67a4f6.

[6] Motus Digital. *What is Real-Time Animation?* 2014. URL: http://motusdigital.com/real-time-animation.htm#text=Real\%2DTime\%20Animation\%2C\%20also\%20called,live\%20and\%20in\%20real\%2Dtime.

[7] Ltd. Digital-Standard Co. *ThreeDPoseUnityBarracuda*. January 19,2021. URL: https://github.com/digital-standard/ThreeDPoseUnityBarracuda.

[8] Abbas El Gamal and Helmy Eltoukhy. "CMOS image sensors". In: *IEEE Circuits and Devices Magazine* 21.3 (2005), pp. 6–20.

[9] Andy P Field and Graham Hole. *How to design and report experiments*. Sage, 2012.

[10] David A Forsyth and Jean Ponce. *Computer vision: a modern approach*. Pearson, 2012.

[11] Maureen Furniss. *Motion Capture*. Visited 25.03.2021. URL: http://web.mit.edu/comm-forum/legacy/papers/furniss.html.

[12] Nick Heath. "What is AI? Everything you need to know about Artificial Intelligence". In: (December 11, 2020). URL: https://www.zdnet.com/article/what-is-ai-everything-you-need-to-know-about-artificial-intelligence/.

[13] Intel. *Intel® RealSense™ LiDAR Camera L515 Datasheet*. Intel, Jan 1,2021. URL: https://dev.intelrealsense.com/docs/lidar-camera-l515-datasheet.

[14] Ning Kang et al. "Real-time Animation and Motion Retargeting of Virtual Characters Based on Single RGB-D Camera". In: *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. IEEE. 2019, pp. 1006–1007.

[15] Simon Kay. "WHAT IS MOTION CAPTURE?" In: (2021). URL: https://mocappys.com/what-is-motion-capture/.

[16] Krystof Litomisky. "Consumer rgb-d cameras and their applications". In: *Rapport technique, University of California* 20 (2012), p. 28.

[17] "Machine Learning: changing the game of Motion Capture, author=Foundry, url=https://www.foundry.com/insights/machine-learning/motion-capture, year=March 03, 2020, publisher=Foundry Insights Hub". In: ().

[18] Aleksandras Melnikovas. "Towards an explicit research methodology: Adapting research onion model for futures studies". In: *Journal of Futures Studies* 23.2 (2018), pp. 29–44.

[19] Alberto Menache. *Understanding motion capture for computer animation and video games*. Morgan kaufmann, 2000.

[20] Xin Min et al. "Motion Capture Research: 3D Human Pose Recovery Based on RGB Video Sequences". In: *Applied Sciences* 9.17 (2019), p. 3613.

[21] Meta Motion. *Motion Capture Recording vs. Realtime*. 2021. URL: https://metamotion.com/motion-capture/motion-capture-recording-vs-real-time.htm.

[22] Hossein Mousavi Hondori. "Different types of motion capture system." In: (2021). URL: https://www.researchgate.net/figure/Different-types-of-motion-capture-system_tbl1_268819026.

[23] Alejandro Newell, Zhiao Huang, and Jia Deng. "Associative embedding: End-to-end learning for joint detection and grouping". In: *arXiv preprint arXiv:1611.05424* (2016).

[24] OptiTrack. *OptiTrack*. Visited 25.03.2021. URL: https://optitrack.com/.

[25] Jennifer Preece, Yvonne Rogers, and Helen Sharp. *Interaction Design*. 4th ed. Wiley, 2015.

[26] José Rossi. *Hand Tracking & Body Pose Detection with Vision Framework*. URL: https://orangeloops.com/2020/08/hand-tracking-body-pose-detection-with-vision-framework/.

[27] Aliaksandr Siarohin et al. “Animating arbitrary objects via deep motion transfer”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 2377–2386.

[28] Leonid Sigal. *Human Motion Modeling and Analysis Fall 2012*. Visited 25.03.2021. URL: http://www.cs.cmu.edu/~yaser/Lecture-3-MarkerBasedMocap.pdf.

[29] iPi software. *heBroadcastBridge: iPi Soft Releases Motion Capture v3.5*. 8.07.2017. URL: https://www.ipisoft.com/2017/07/thebroadcastbridge-ipi-soft-releases-motion-capture-vers-3-5/.

[30] Vijay K Varadan et al. “their Applications”. In: *Pennsylvania State University, USA, Wiley Online Library* (2003).