Online Full Body Motion Reconstruction

Mats Donselaar

July 1, 2014

Contents

1 Introduction

In the past, research has been done into the capturing and processing of user input, varying from simple input such as key or mouse button presses from to complex input such as the motion of a human body. Motion capture was traditionally only used to record motions which would be stored in a huge database, after which the motion capture data would be used to create the animations we see in modern video games and movies. Only relatively recently did we see game controllers emerge which attempt to capture more human motion than just hands, and also focus on real-time motion capture as input. The Wii and Kinect are two well-known examples of commercial applications of motion capture interfaces for consumers to use in their own home. However, these devices only use a small portion of human motion, and are thus limited in accurately capturing human motion. It should however be noted that it is simply not feasible to fit the complex systems normally required to facilitate proper motion capture in a normal living room. Hence, there is an ongoing field of research which attempts to create methods which eventually allow simple systems, and thus feasible for home consumers, to better capture and process human motion input data in real-time without losing much accuracy. While part of this research focuses on better and more advanced input devices, another part of the research focuses on the more efficient and better processing of the available motion data.

This thesis will focus on the latter: investigating what is required to produce good looking animations. The research will determine how much motion capture data is needed in the form of marker positions in order to be able to accurately portray the motions of the actor. For this accurate representation, we also look at which joints we need in a virtual skeleton and which degrees of freedom are essential to maintain naturalness in the reconstructed motions.

Part of this thesis will describe the pipeline required for performance animation. When converting raw motion capture marker position data into the animation of the reconstructed motion, you will need some form of data smoothing and filling, skeleton fitting, and inverse kinematics. Each of these parts of the pipeline will be discussed in this thesis, to explain the design choices made during implementation as well as elaborate on how the encountered problems were solved. Before we cover the implementation details, we will first look at some of the relevant research done in the past in chapter 2.

The final part of this research will focus on identifying meaningful experiments, where users will be able to rate the naturalness of varying motions. These experiments consist of different sets of markers and joints, both in order to identify the minimal possible adequate set and to identify which parts of the body have the most impact on the overall naturalness. This will be done by comparing the naturalness of the experiments to that of the ground truth animation, which contains a full marker and joint set and should thus be as accurate as possible. Parallel to this, this project will look at different metrics to determine the error between the animation of the experiments and the ground truth. Finally, the resulting data from the subjective nature of the naturalness will be compared to the objective error metric, to determine whether there ex-

ists a correlation between the two. The results from these experiments would ultimately also allow to propose a sort of 'optimal' configuration of markers and joints, which produces high quality results with as little data required as possible.

As the entire process of transforming raw motion capture data into character animation is done online, there is no room to apply any post-processing methods. This is why all the methods described in following chapters of this thesis are designed to work online.

2 Background Research

In this section there will be a brief discussion of the papers that are related to this research or helped determine which techniques to use.

2.1 Motion data processing framework

For general guidelines on how to implement the overall part of the pipeline which covers the processing of the motion data, I used the report of the experimentation project of Schuddeboom [Sch11]. He already implemented the real-time processing of motion capture data, and describes the framework and techniques he employed. His pipeline consisted of four primary stages: smoothing, gap filling, marker identification, and interpolation.

For smoothing, he compared a Butterworth low pass filter to a Kalman filter, and explained the advantages of both techniques. Because the Butterworth low pass filter displayed some inaccuracies, he judged the Kalman filter to be more suited for application as a real-time data processing method. The downside of these techniques is that they are not directly applicable to my own problem, as these methods rely on a limited data set.

For gap filling, he analysed the differences between average velocity prediction, polyfit prediction, and spline interpolation. While spline interpolation is not fit as a real-time interpolation method, it was included in the discussion as comparison. Polyfit reduction displayed a large error, even for short moments of missing data. This resulted in Schuddeboom expressing his preference to the average velocity prediction.

While the marker identification is already mostly done by Vicon Nexus in my own program, Schuddeboom makes suggestions to use a prediction method such as the one used by the Kalman smoother filter to help identify markers. This could be used in the case where Vicon Nexus mislabels markers. His interpolation involves linearly converging from the gap filling extrapolation to the rediscovered marker data.

The report from Schuddeboom only captures the online processing of motion data, and does not address skeleton fitting or inverse kinematics. For this, other papers will be referenced in the following sections. Nevertheless, the report by Schuddeboom gives a good idea of a framework, and provides references to the research from which he employed several techniques. The techniques I will use are those for estimating the marker positions of occluded markers [LM06, PLKF09]. Piazza et al. presented a cost-efficient extrapolation algorithm with

the specific purpose of being able to produce adequate results while having the constraint of being real-time.

Their algorithm differentiates between a linear and circular case of movement. Human movement is typically best approximated by linearly combining the linear and circular types of movement. For the purposes of my thesis, I focus primarily on their linear case of extrapolation. Their prediction algorithm creates a prediction vector looking at the history of the position. In their system, they looked back 30 frames, which was $250ms$ back in time. They then used the vector from the position 30 frames back to the most recent position as prediction vector, which should be less vulnerable to local high-frequency motion.

2.2 Computer puppetry

As an early example of a real-time pipeline from motion capture to virtual character animation, Shin et al. [SLSG01] managed to create a system which transformed motion capture to animated character performance in real-time. They demonstrated their techniques by applying them to create virtual character motion live in a televised broadcast. With this, Shin et al. were one of the first to implement all the techniques required for what was defined as Computer Puppetry in 1998 by Sturman [Stu98].

To implement the pipeline from motion capture to character performance, they implemented several steps. Using an optic system with motion capture cameras, they applied a Kalman filter to clear the motion from any artefacts created by the motion capture system. After filtering the noise from the data, they look at the end-effectors and their relations to the environment. If a hand or foot is simply somewhere in the middle of air, not close to any objects or obstacles, calculating its exact position is less important. If the position is off by a small margin, the error will not be as noticeable. However, when a foot is about to touch the ground or a hand is about to hit an object, tracking the exact position becomes more important. This is why end-effectors near an object have a higher importance than end-effectors which are relatively free. While it is possible that something other than an end-effector has a collision, bumping your knee into a table for example, the assumption is made that the space is relatively open and such complex collisions will not occur. The final step of the pipeline would be to let an algorithm compute a pose of the virtual character in a way such that it will satisfy as many of the features which have been determined to be important for the pose to be recreated as characteristically as possible.

The problem with this, however, is that there is no single set of features which is always important. The set of important features has to be decided for each frame. A simplistic way of doing this is by only looking at the positions of end-effectors and not setting any constraints on joint angles, but this is prone to errors. There can generally be more ways of fulfilling end-effector constraints, as is demonstrated in the paper. This is why they conclude that the set of important features needs to be determined depending on context. They discuss several techniques which can be used to determine importance

[LS99, Gle97, Gle98, PW99], but those are all offline or inapplicable to the problem at hand. This is why they determine a heuristic where the importance of the end-effector is inversely proportional to its distance to the nearest object in the environment.

An issue Shin et al. ran into was that their system would not work as well for characters which did not resemble human anatomy close enough. In order for characters to be fit for motion retargeting, and importance-based metrics to be relevant, humanoid anatomy was required. During their analysis of inverse kinematics solvers, they conclude that analytic methods cannot incorporate measures required for retargeting, while numerical solvers, although they can include the importance metrics, are just too slow. They created fast IK algorithm is specialised for humanlike articulated characters.

The algorithm starts off by attempting to move the root of the body into a position from which all limbs can reach their goal positions, which is calculated as in intersection of all the spheres of reach. Should such a position not exist, a combination of the root position, root orientation, and posture of the upper body will be attempted instead. After that attempt, a configuration of the arms and legs is found to reach the targets. The calculating of the joint orientations themselves is partially done by numerically solving some equations while the rest is done analytically, which is inspired by a hybrid solver which was demonstrated by Lee and Shin [LS99]. The reason for this is that numerical methods, even though they produce more accurate results, they take too much time for a real-time application.

What my own approach intends to do differently is to fit the virtual skeleton nearly perfectly to the pose of the actor, as indicated by the markers on his body. My method will likely have less interaction with the environment, decreasing the need for importance-based metrics. Whereas the method by Shin et al. was demonstrated by applying it to character swhich were humanoid cartoons, which could have completely different body proportions than humans, the intention of my research is to match the human performer as closely as possible. If this is to be achieved, the importance-based metrics that serve to satisfy as many of the kinematic constraints, such as root position, joint angles and end-effector positions, which describe the characteristics of the pose, become less meaningful. While Shin et al. presented good and fast methods, they were specialised to work in unison with importance-based metrics which are not fully applicable to this research.

2.3 Low-dimensional control signals

A study which contained relevant information was the paper written by Chai and Hodgins [CH05]. In their research, they aimed at creating a robust optical motion capture system using as few as 6 to 9 markers and only two cameras. This paper is relevant in multiple ways. The first and most obvious reason is that they used an optical motion capture system for their research, opposed to inertial sensors. Any techniques they have devised which target optical motion capture system specifically will most likely work in my own research as well. The second reason is that their techniques also work in real-time.

All the processing of the motion capture data is done online, and produces results which are qualitatively barely inferior to the results of offline methods. The third point where their research is relevant to mine is that they focus on reducing the amount of markers needed for motion capture, which is something this research will be focusing on during the experimental stage.

Chai and Hodgins use databases of pre-recorded human motion data for their methods. The motion is recorded and processed in real-time by comparing it to motion fragments in the database. The downside to this approach is that you need a very fast algorithm to traverse a motion database, as finding the segment which matches the movement of the actor as close as possible becomes more and more difficult as the size of the database increases. As the entire purpose of the algorithm is to work online, there is also a time constraint which needs to be taken into account.

This time constraint is satisfied by a combination of different approaches. The first aspect is the reduced marker set. Chai and Hodgins noticed an important aspect of human motion, as they tried to make their system with only 6 to 9 markers resemble the high-dimensional human motion which has around 60 degrees of freedom. What they noticed was that this high-dimensional human motion displays a relatively high level of dependence between the degrees of freedom. This means that, even when using a much reduced marker set, you would still retain sufficient motion information.

This opened up the possibility of online local modelling. By only having to evaluate and optimize a much smaller set of constraints, they were able to develop a lazy learning algorithm. This algorithm constructs a series of simple local models which sufficiently constrain the solution space, and postpones computation until an explicit request for information is received, saving precious processing time.

There is a slight point in their research which might result in some of their findings not being applicable to my own research, being their motion capture system. While it is an optical system like the one I will be using, they only use two frontal cameras. The reason for this seems to be because they attempt to create a relatively simple system which can replace complex motion capture systems which require difficult suits and expensive cameras to operate. They are essentially targeting the consumer, perhaps even more specifically aimed at gaming. In these applications, the user will be facing the screen most of the time, and the only two cameras required are ones facing the front of the motion actor. As a result of this, the only markers required were the ones on the front of the actor, as the ones on the back add very little information.

As I will be using a full room of cameras, I will have more information to work with. This means there is less chance that a marker is occluded and relatively less data loss when eventually one marker does become occluded. Because these experiments use cameras and markers all around, there is enough information for full body motion synthesis, without restricting the actor in its movement. Due to the amount of information, my own research will not be using a motion capture database. While Chai and Hodgins managed to reduce the amount of markers they needed for their system, as well as only requiring a single camera, their results are largely dependent on their use of the database,

and therefore not very relevant to my thesis.

2.4 Motion control with inertial sensors

Another paper on performance animation, written by Liu et al. [LWC+11], is slightly similar to the paper by Chai and Hodgins [CH05]. In their research, Liu et al. developed a system for online full-body motion synthesis. Their system uses inertial motion sensors opposed to the more traditional optical sensors. They chose to do this because their focus, similar to that of Chai and Hodgins, was to create a system which was more viable for consumers to purchase for their own homes. Their inertial motion capture system uses six well-placed inertial sensors on the head, hands, waist and ankles, and produces motion synthesis of quality similar to that of a full motion capture suit.

As the research done by Liu et al. was more or less a continuation of the research by Chai and Hodgins, Liu et al. also employed a database consisting of varying types of motion, which would later be used during the processing to search the database for several of the motion fragments closest to the actor's motion. They improved on the algorithm by Chai and Hodgins, by providing their own method of finding the K nearest neighbours. Going further than merely assuming that the degrees of freedom in human motion are correlated, they assume that a human motion segment can actually be described as an m-order Markov chain. This means that the current pose in a dynamic human motion is dependent on the previous m poses. Using this as a basis, they developed a model which can be dynamically updated opposed to being completely recalculated with each new pose. This means that they use a previously calculated pose as initial approximation of the next pose, and use their algorithm to change joint orientations of the current pose until it fits the new pose.

Nevertheless, Liu et al. also use a database with pre-recorded motion, something which I do use for my own research. My research involves neither inertial sensors nor pre-recorded motion data for pose matching. As such, the value their research offers is their analysis of noise and how to clear the motion data of any noise.

2.5 Distance metrics for animation blending

As the final part of this research focuses on error metrics for determining the quality of animation, the research done by van Basten and Egges [vBE09] can serve as a guideline for the sort of error metrics that can be used. Their research focuses on identifying similar motions, to determine which motions in the motion graph could allow for a smooth transition between the two. To identify this similarity, they propose three different metrics: one that compares joint angles and derivatives such as velocity and acceleration, a second metric that calculates the geometric distance between two point clouds, and a third one that performs principal component analysis. They evaluated these metrics in two ways: quantitative and qualitative. The quantitative component looked at attributes such as path deviation, foot skating, and running time. The qualitative part was in form of a user study, where people were asked to rate the

motion realism of some animations. The results of their research showed that each metric had its advantages and disadvantages, and that there seemed to be a correlation between the summed distance and the grade given a motion. For all three distance metrics, there seemed to be a plateau at the smaller distances, where slight increases in distance did not have a noticeable effect on the perceived naturalness. This plateau was different for all metrics.

The research done by van Basten and Egges differs from this research on several points. Some minor ones are that the foot placement will not be critical in this experiment as this is fully governed by the marker data of the actor, rather than a path along which a walking motion has to be generated. Running time of the metrics is irrelevant as well, given that this research will not analyse the motions in real-time, but during offline post-processing. However, the most important point of difference is that this thesis is not focused on motion blending, where the goal is to find the best point in two motions where they could transition from one to the other. Instead, the error metrics will evaluate the error between two motions for the entirety of those motions. The metrics are used to calculate the error between an experiment with a reduced marker and joint set and the ground truth motion, which are both constructed from roughly the same data set. The experiments will not use different raw marker data, simply *less*. While the exact reconstructed motions will be slightly different, the sort actions they perform will be same, opposed to blending from one action into the other.

During the experiments, the goal is to determine which variables influence the perceived naturalness the most. Viewers could base their rating of naturalness on any number of things, such as jerking motion, limbs bending the wrong ways, or even single 'spike' of error that only lasts a few frames. Any metric will have to be tested in several ways, to be able to properly capture whatever criteria a viewer has for determining naturalness. Given that the error metrics are evaluated during post-processing and thus completely disjoint from the performance animation pipeline, the implementation details of the error metrics will be discussed during the experiments in chapter 5.3.

3 Online Motion Data Manipulation

Before going into detail regarding the actual implementation of the online motion data manipulation pipeline in the following chapters, I will first give a short overview of all the stages and the solutions to the important problems with the processing of the raw data. In the raw motion capture data, we could have mislabeled markers and noisy data, with possible gaps. In order to avoid the mislabeling, a two-staged solution was implemented. Firstly, mislabelings of the markers by Vicon are checked as is explained in section 3.2. Secondly, motion capture can produce noisy motion data, which has to be dealt with. Judging from several papers [SLSG01, AB94, FSP92] that have dealt with the same issue, a Kalman filter produces the best results for an online method. The Kalman filter will be discussed in section 3.3. In case of a gap in the data, the prediction vector created for the Kalman filter is used to extrapolate, after

which the previously missing marker is interpolated smoothly back to the actual trajectory.

Given that the motion capture is done by the Vicon Nexus system, the start of the full pipeline is technically the marker placement and calibration process of Vicon Nexus. From there on, the raw marker data will be sent over the socket to my program, where it goes through a series of operations before being passed on to the inverse kinematics part of the pipeline. The operations performed on the raw data are the denoising and gap filling, but also the detection and relabeling of incorrectly labeled markers. An illustration of the entire frame-by-frameprocess can be found in figure 1. Each of the steps of the process will be discussed in the order in which they take place. After these data processing steps, the data will be further used by Inverse Kinematics in chapter 4.

3.1 Raw motion capture data

To do motion capture, the system I use is 14 motion capture cameras, using Vicon Nexus software to drive the cameras and provide the initial marker labeling and position values each frame. Vixon Nexus is used to interpret the data feed from the cameras and turn it into 3D-coordinates with sub-millimeter accuracy, although still being subject to noise. The Vixon Nexus system also allows the user to calibrate to a specific marker set. The markerset I have selected as a base line for all following experiments contains 38 markers, as illustrated in figure 2. This markerset is calibrated in Vicon Nexus by using a T-pose, and then manually assigning each of these names to each of the markers the cameras registered while holding the T-pose. These marker labels are then tracked internally by Vicon Nexus during the live recording. While this internal Vicon Nexus labeling is not flawless, it gives a good general measure of which markers are which, something I therefore did not have to reproduce in my own program.

Another functionality Vicon Nexus has is that it attempts to relabel if it has reason to believe that something has gone wrong with the internal real-time labeling. The inner workings of the Vicon Nexus labeling and tracking are not part of this thesis, but dealing with the occasional errors produced by them is. For example, in the case where two markers come very close to eachother, they might get swapped around. This occurrence is marker mislabeling. As the markers move further and further apart the Vicon Nexus system will attempt to correct the labeling itself, which unfortunately is not guaranteed to always work. A more elaborate explanation of how the mislabeling occurs, how to detect it and how to deal with it will follow in chapter 3.2.

An important thing in handling the raw marker data from Vicon Nexus is that I will need to use the raw data multiple times for the purposes of running experiments. By storing the raw data with all the labeling in files, I will later be able to run several experiment configurations using the same raw marker data. This serves to eliminate any variation caused by the actor having to attempt to recreate the same motion for each experiment, as it is impossible to perform the exact same motion multiple times. While this is not relevant to the online portion of the performance animation, my program is also able to use file input as offline alternative to the live motion capture stream. The offline alternative

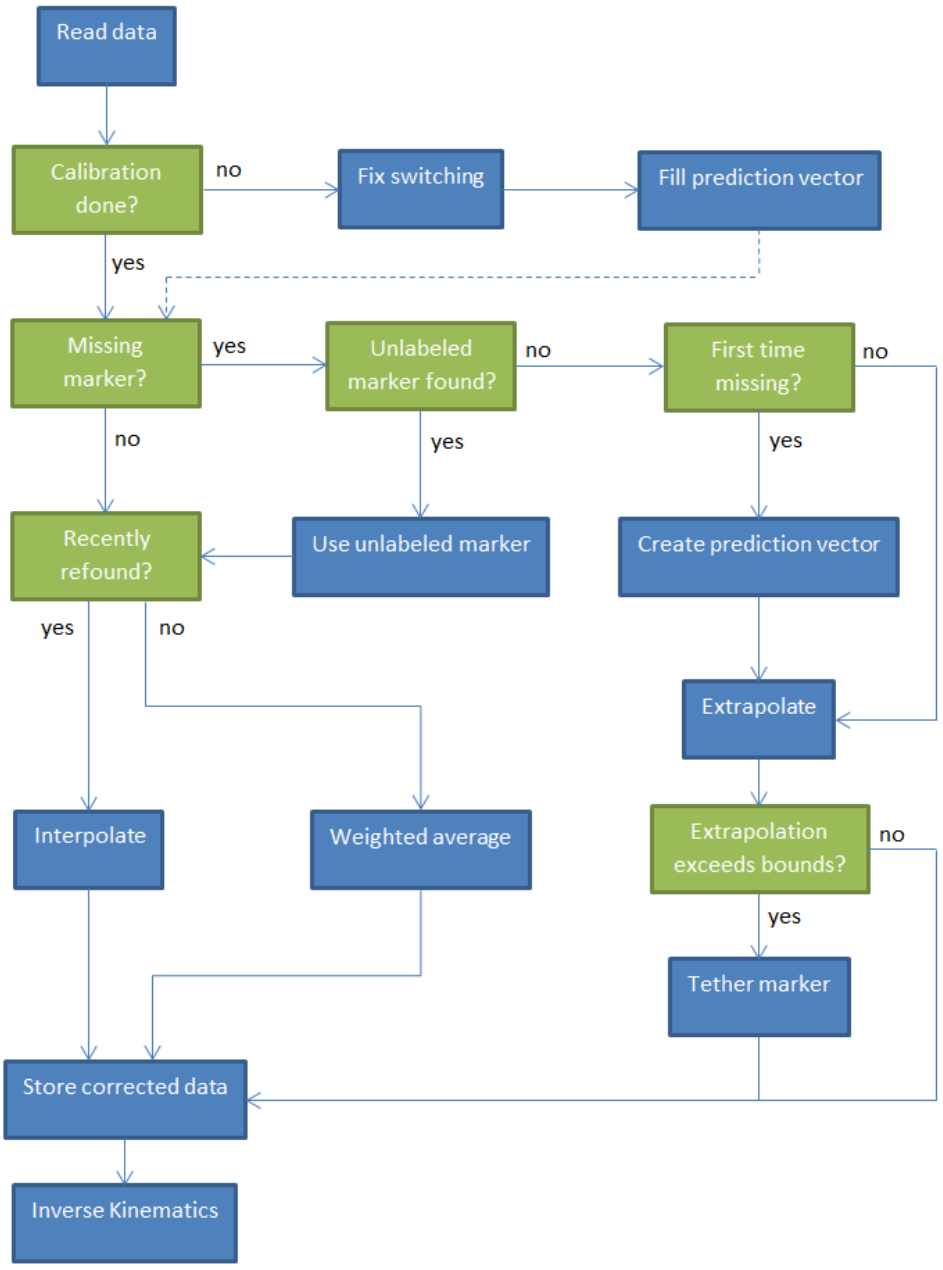

Figure 1: Flowchart of the data correction process which is executed every frame. Calibration is done by filling the prediction vector, after which all the other steps of the process are executed from that point onwards.

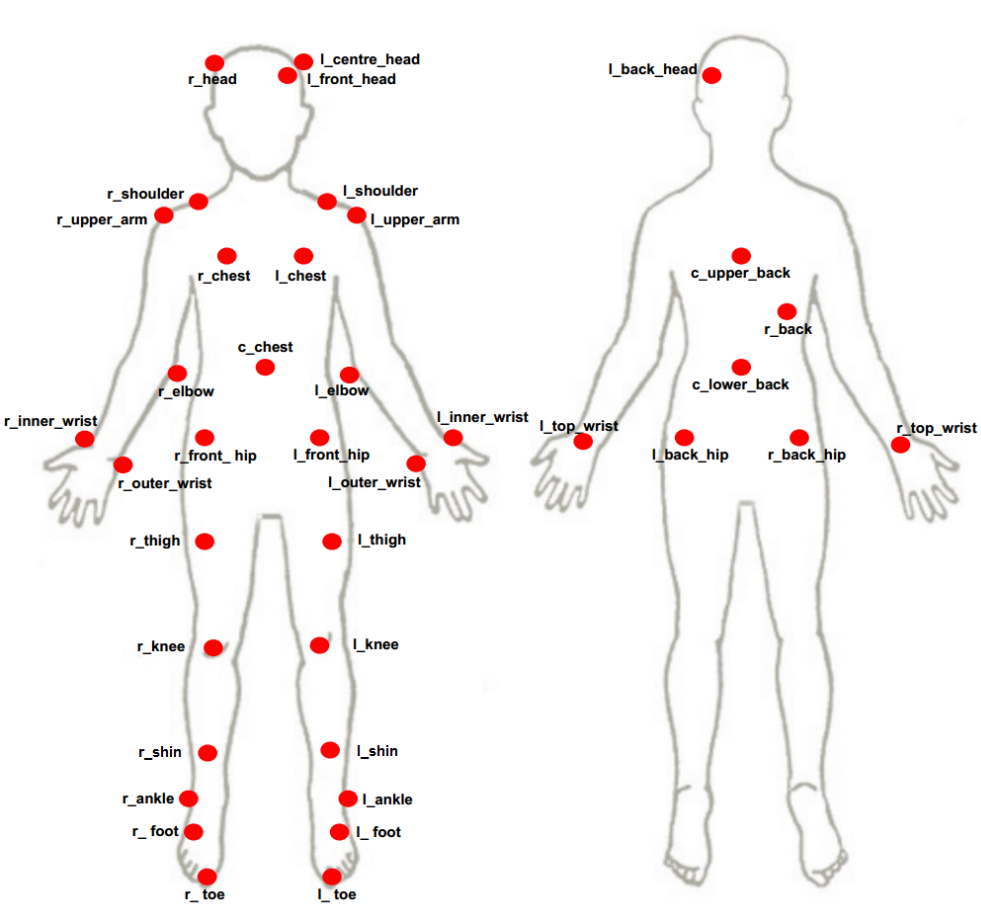

Figure 2: Positions and names of the markers

uses the rest of the pipeline as normal, as the raw data provided to the pipeline remains the same, barring several omitted markers for particular experiments. More on how the file input is used for the experiments in chapter 5.1.

To summarize, there are issues presented by the motion capture system, which I will have to solve to the best of my ability before attempting any further steps down the pipeline. The marker positions as given by Vicon Nexus are subject to noise, which will have to be smoothened out. Markers can become occluded, and a method of extrapolation for prediction will have to be implemented. Remember, as the goal of the application is to have the pipeline run in real-time, the place an occluded marker will reappear is not known during gap filling. The last and most impactful issue is marker mislabeling, something which can have a cascade effect on an entire limb, causing multiple markers being incorrectly labeled. Whereas noise and gap filling can be done locally using the data of only one marker, marker mislabeling can have consequences throughout the body and can only be solved by looking at the whole.

3.2 Marker calibration and correction

The motion capture cameras are, unfortunately, not 100% accurate. Next to the aforementioned noise from camera system, another big problem is marker occlusion, something which is inevitable given any normal walking gait where arms are moving alongside the body. Hip markers are very prone to being occluded, be it very briefly, by the arms passing by them. Given that Vicon Nexus has been calibrated to keep track of labeled markers, it is able to detect if markers have gone missing. A missing marker will return the position $(0, 0, 0)$, a position that typically never occurs for any given marker. This is typically an easy test to determine whether marker data is missing, but unfortunately will not suffice completely.

While the motion capture data is recorded with sub-millimeter precision, cameras sometimes get tricked into seeing additional markers. Fake reflections from dust or other things may cause the cameras believe there exist more markers than the user specified during the calibration of Vicon Nexus. These fake markers are called 'ghost markers'. The method I use of connecting my program with the Vicon Nexus system is the Vicon DataStreamSDK [Vic] provided by Vicon. Next to the normal marker data, I also keep track of the ghost markers. The reason for this is that Vicon Nexus sometimes has a double error, in the sense that one marker is occluded and then falsely concludes data from a completely different marker or even a ghost marker to be correct.

Marker mislabeling exists in different forms. The simplest one occurs when two markers get very close to eachother. At this point, the Vicon Nexus automatic labeling might make a mistake given that the margins of error are sufficient. This could happen when someone, for instance, moves their wrist too close to their hip. Even if the hip marker doesn't become occluded, the values are close enough to be confused and we have a marker labeling swap. While pretty much harmless at first, seeing as the markers being very close caused the swap in the first place. However, the labeling swap might persist as the markers continue on their trajectory, and the distance between the two

swapped markers becomes noticeably large. At this stage, the marker labels should get swapped back again.

The problem just mentioned, a marker labeling swap which occurs when two markers come too close to eachother, sadly also occurs occasionally even when markers *aren't* close to eachother. A commonly observed marker swap is the one where two knees might simply swap labeling. The problem often manifests itself by simply swapping the knee markers around, but leaving the rest of the legs unaltered. Such errors should generally be easily corrected, as the entire body structure as well as the history of the trajectories of the markers indicates that a marker labeling swap has occurred. Unfortunately, it is also possible that one of the two knee markers may become unlabeled entirely. In this situation, we have one marker taking over the data of another marker, with the other marker simply having no data anymore. A third manifestation of this problem sometimes produces a cascade effect. Occasionally, one marker swap between the left and right shoulder marker, or the left and right thigh markers might result in *all* the markers on the left and right side become swapped around, making part of the character's body seemingly instantaneously turn 180 degrees. As these cases of marker mislabeling very often doesn't solve itself by randomly switching back normal, we have to employ a generic solution that indefinitely tracks and fixes the marker swap, which will be discussed in section 3.2.2.

In order to solve the mislabeling problem, we need to analyse how mislabelings occur, whether Vicon Nexus fixes them automatically or not, and how to handle the mislabeling. The two different cases of mislabeling, finite and indefinite, will be treated separately.

3.2.1 Mislabeling detection

To start off, the case where two markers become swapped because they are too close to eachother will be ignored. That case is not relevant because the error does not become noticeable until the markers once again go apart, and one might go in the completely wrong direction. As this will go very gradually, mislabeling detection based on analyzing the prediction error will not be effective here. Instead, there are several things we know about the human body. One very important characteristic that we humans have is that our bones don't usually resize that much without breaking. Given that someone breaking a bone means that we should stop recording an animation, it is implied that throughout the animation, markers which are positioned on the same bone should not move relative to eachother. Unfortunately, markers are not placed directly onto the bone but onto the skin or even onto clothes, so they will still move slightly relative to eachother. However, we can still use the distance between these markers are a relatively constant guideline throughout the real-time animation recording.

We can use a configuration file which describes the human skeleton, all the joints that are relevant to this project in it, and the position of the markers relative to these joints. Markers that are attached to the same bone are grouped together. While the exact use of this skeleton file will be further explained in

chapter 4.1, the information that certain markers are positioned on the same bone means that we can use those markers to imply eachother's position. During the beginning of the recording, we can measure the distance between each of the markers in the same group. To make sure we have a bit of a reliable distance calibration, we take the average distance over the first 50 frames. Then, if a marker's distance later on during the recording becomes too large to the other markers it was calibrated with, there's undoubtedly something amiss.

All we need to do now is find suitable thresholds over or under which a marker position is considered to be invalid due to bone length. A static number for distance thresholds might be used, but given that bones generally not have the same length, an arbitrary number might not cover all cases sufficiently. Through tweaking and experimentation, I've found that summing a percentual factor of the calibrated distance and a static addition works best. The percentual part consists of upper and under thresholds of 1.05 and 0.95 times the measured distance between markers, where the static part consists of the distance traveled since last frame plus $0.025m$. In other words, distance d is accepted when it fits the constraint formalized in equation (1), where i is another marker in the same group, d_i the calibrated distance to that marker, and $\vec{m}$ and $\vec{m}'$ are respectively the current and previous marker positions.

$$\forall i : 0.95 * d_i - (0.025 + ||\vec{m} - \vec{m}'||) \leq d \leq 1.05 * d_i + (0.025 + ||\vec{m} - \vec{m}'||) \quad (1)$$

The second part of our mislabeling detection is by checking the prediction, as mentioned before. We want to define a sort of acceptable volume, in which a marker should be located. A possible center for this volume would be to use the marker's predicted position. We could create a prediction of a marker's position in the new frame by adding the prediction vector to the previous position. In this thesis, there was chosen to use the previous marker position as the center, and expand the volume to compensate for the fact that we're not taking the trajectory into account. We can make the volume dependent on length of the prediction vector. By doing this, we can accommodate for errors in prediction when someone makes a high-frequency motion that completely goes against the current trajectory.

We define the acceptable volume as a sphere with a radius the size of 2 times the prediction length plus a static $0.01m$. If the new marker position is not within this volume, it is flagged as possibly mislabeled data. This factor $f = 2$ seems slightly arbitrary, but is the result of tweaking parameters until a configuration is found that produces good results. The reason for this seems to be that the acceptable volume needs to err somewhat on the cautious side to be able to accommodate for both noise and rapid acceleration of the speed at which the marker is moving. In the case someone is keeping one body part complete still before suddenly accelerating, which could for example happen when someone starts sprinting, a prediction based on the previous trajectory of the marker would be completely unable to predict this valid human motion. The value of variable f is able to be changed real-time by the user, along with the upper and under threshold factors and the static additions in both equation 1 and the definition of the acceptable volume.

The combination of biomechanical constraint violation and severe prediction defiance should be able to detect all marker mislabelings which are noticeable enough to be distracting in the animation. Tiny mislabelings of markers such as the ones on the foot or on the wrist will be relatively easily detected by the bone length violation, as any swapping around is guaranteed to violate either the upper or under threshold. Knees being swapped around should quite severely defy the prediction. Labeling a ghost marker should trigger both the bone and the prediction detection. Markers swapping due to proximity will be picked up by the bone length violation as soon as the markers move away from eachother again. Trivially, complete limb swaps will raise numerous prediction defiance flags. Now remains the task of actually handling the detected mislabelings.

3.2.2 Marker switching

Now we've established some criteria for detecting possible mislabelings, our first problem to solve will be the mislabeling case where some if not all the markers in the limbs get swapped around indefinitely. For certain parts of the body such as the legs, it can occur that only parts of the leg will get swapped around. As such, the marker switching algorithm will have to be able to deal with swapping around individual markers. Given the fact that the markers on the hip are used to determine the root position and orientation, a mislabeling during calibration of the hip markers could be considered a fatal error. In the case of a mislabeling of hip markers, the entire calibration should be redone Vicon-side. So, even while we technically cannot guarantee the hip markers are correctly labeled, this assumption is the base of the algorithms.

When we observe the calibration T-pose, there are certain facts about the markers on the human body. The left thigh marker should be closer to the back and front left hip markers, and the same holds for the right side. Using this fact, we can test whether the thigh markers are closer to their respective hips, or to the opposite hips. When the latter is true, we have detected a mislabeling and swap the markers around. After now having made sure the thighs are correctly labeled, we repeat this process by comparing the knee positions to the thigh positions, and work our way down from there all the way to the toe. In a similar fashion, we can check whether the chest markers are closest to the correct torso, whether the shoulders are closest to the correct chest marker, and so on. Another occurrence of an indefinite marker swap is where the toe and foot markers of the same foot get swapped around, due to their proximity. We can use the same principle here, but instead of comparing the left and right side, we compare the length inside the foot itself. The foot marker should be the closer of the two to the ankle marker, and we can make a swap should the foot and toe marker not follow this constraint. We store each of these marker swaps in a list, which we can use to swap the data from Vicon around every frame.

One important aspect to this swapping correction process is that the swapping detection part of it only works on user demand while the actor is standing still. As such, the swap list is constructed at the very start of the recording process, and only updated whenever the user specifically gives the command

for it. The reason for this is that these constraints will only function properly during the calibration T-pose. An example of the constraints giving a false result would be someone sitting cross-legged. In this example, it is clear that the left ankle marker is closer to the right knee, and the right ankle is closer to the left knee. Should the swapping detection still be active, it would incorrectly swap the ankle markers of both legs around. As such, the detection only works during the distance calibration frames, which were mentioned in chapter 3.2.1. After the calibration, the incoming raw marker data will simply be swapped around according to the list that was constructed during calibration. In the case that one of the swaps fixes itself, or a new one occurs, the user is able to manually give the command recheck the swapping without the need to fully calibrate again.

3.2.3 Marker relabeling

While the switching algorithm in chapter 3.2.2 was designed to solve the indefinite swappings that can occur, that algorithm can only detect and solve the swaps during calibration. We also need an algorithm which detects the short-term mislabelings that can occur at any time during the recording. The detection for these mislabelings is as was described in chapter 3.2.1.

What we can do is check the positions of all the unlabeled markers, and see if any of them are within the acceptable volume. If there are multiple unlabeled markers which fit the criterion, we pick whichever one would be closest to where we predicted the marker should have been. We take the marker data of the unlabeled marker out of the list of unlabeled markers, and replace it with the marker which data defied our prediction too much.

It's very important to point out that, in the case of there being no valid unlabeled substitutes, we do *not* consider the marker data missing. There are situations imaginable, as was suggested in chapter 3.2.1, where marker data can be perfectly fine yet still defy the acceptable volume. As such, we only really consider a marker to be missing when the position is $(0, 0, 0)$. In the case of actual missing data, we still check to see if there is an unlabeled marker which fits the criterion. In the case where we actually find such an unlabeled marker, we will instead consider the marker to not be missing. Only when Vicon gives the marker position $(0, 0, 0)$ and we cannot find a suitable substitute do we initiate the gap filling procedure of chapter 3.4. Other than that, any incorrect marker data should be able to get smoothened out by the Kalman filter.

3.3 Kalman filter

We now get to the part of the data processing pipeline that deals with the actual smoothing and noise filtering. At this stage, we should have correct labeling of all the markers. This part of the pipeline deals with the situation where there is currently no marker occlusion. The case of a marker occlusion will be discussed in chapter 3.4. As such, this entire section is written under the assertion that we have marker data available, and the main objective is to smoothen out the noise.

The filter used by the program to do smoothing is an Extended Kalman Filter, which is an extension of the regular Kalman Filter. Whereas a normal Kalman Filter attempts to find the exact value of a single, static variable by smoothing down noisy measurements which are roughly around the actual value, an Extended Kalman Filter is designed to also work for dynamic variables [WB06]. Dynamic variables differ from static variables in that dynamic variables are 'online'; the Extended Kalman Filter is designed to remove noise from a signal as you're receiving it. As this project is online and we receive a continuous flow of marker data, we need the Extended Kalman Filter.

The nature of a Kalman Filter is to make a prediction for the upcoming values of your variable based on the previous values, and then adjust your values based on the prediction. Each of the previous states of our variable will have influenced the prediction for the upcoming state, be it very slightly. However, as we are having a large amount of data for a short timeframe due to our high framerate ($\sim$ 100 frames per second), only looking at the last x states will give a very local prediction. Such a local prediction will very likely be heavily influenced by high frequency noise, which is the exact thing we are trying to remove with our smoothing algorithm. This is why we have to sample our states over a larger period of time. If we take a large enough time window with samples spaced out across the interval, we should obtain a much more accurate long-term prediction.

While the exact length of the sample window and the amount of samples in that interval is fully customizable by the user in real-time, the standard configuration for these parameters is set at a 1.0 second sampling interval, with a sample being taken every 0.1 second for a total of 10 samples evenly spaced out across the interval. In the case of 100 frames per second, every 10th frame will be stored in the history array, only keeping the 10 most recent samples. The reasoning behind this standard configuration is fairly simple. Through simple tests, it appeared that a 1.0 second interval was just right to eliminate most of the high-frequency noise while still maintaining some of the actual high frequency movement to stop it from looking unnatural. If too little of high frequency movement was preserved, the limbs would display strange gliding motions. Similarly, 10 samples proved through tweaking and experimentation to be a configuration that produced good results. More frames would seemingly grant no additional information and would occasionally allow noise to persist, less could instead result in too little information to make accurate predictions.

The algorithm used to remove the noise looks at the sampled states in the history array and makes a prediction based on those, valuing the more recent states heavier than states further back down the timeline. This is done by calculating all the difference vectors between the previous states in the sampling history, and performing a weighted average between them. To put this in an equation, the prediction vector $\vec{p}$ for any marker $\vec{m}$ is the following:

$$\vec{p} = w_1(\vec{m}_1 - \vec{m}_2) + w_2(\vec{m}_2 - \vec{m}_3) + ... + w_{n-1}(\vec{m}_{(n-1)} - \vec{m}_n) \qquad (2)$$

In this formula, w_i is the weight for the state $\vec{m}_i$, where i indicates the index in the history array, and n is the amount of states sampled. Lower values of

index i indicate more recently sampled states, and n in this case would be 10, as mentioned earlier. The function used in my program to determine w_i is an exponential one, namely $\frac{1}{2^i}$. The reason I chose for this specific function is that the summation of vectors multiplied by this function will be very close to a normalized vector. To make sure the vector is normalized, the weight w_{n-1} is equal to w_{n-2}. To illustrate this, here is the weight function for $n = 5$:

$$\frac{1}{2^1} + \frac{1}{2^2} + \frac{1}{2^3} + \frac{1}{2^4} + \frac{1}{2^4} = \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \frac{1}{16} + \frac{1}{16} = 1$$

So, here we have a prediction algorithm which is very fast and produces normalized prediction vectors very efficiently. There is however one little thing that still needs to be pointed out. The n states were sampled with an interval of t, where t is dependent on the framerate, sampling interval, and amount of samples. Given that the framerate is 100 frames per second, that means that the prediction we currently have in the standard configuration is based on differences of 10 frames, while we are trying to predict *every* frame. This means that we have to divide our prediction vector by the amount of frames between sample states in order to create a prediction vector for a single frame. As such, the formula for the predicted position of our current marker $\vec{m}$ looks like this:

$$\vec{m} = \vec{m}_{prev} + \frac{\vec{p}}{t}, \tag{3}$$

where $\vec{m}_{prev}$ is the marker position in the previous frame, $\vec{p}$ the prediction vector from (2), and t being the size of the sample interval.

As we now have a nice prediction for the current location of each marker based on both the previous position and their positions further back in time, we should not forget the essence of an Extended Kalman Filter is to correct measurements based on past results. The goal was to reduce the noise from the Vicon marker data, but our prediction algorithm as it is right now is simply a form of extrapolation. In order to reduce the noise while still generally following the marker data, we take the weighted average between the predicted marker position and the marker position as received from Vicon. This weighted average could be anywhere between 90% raw data 10% prediction and 5% raw data 95% prediction. Through several tests, the algorithm behaves as one would expect. When the weight for the raw data is high, noise will be more apparent. Alternatively, when the weight for the prediction becomes too high, high frequency motions become blurred out, resulting in very unnatural gliding animations. Through further testing, I've come to the conclusion that trying to fit a single weight for all the markers across the body is not sufficient. For some markers, a weight anywhere above 15% for the raw data provides very poor results as the noise becomes apparent. Alternatively, if the weight for raw data is lowered below 10% for certain other markers, very little of their original motion is retained. When the weight is lowered to 5% or below, even low-dimensional signals begin to fade at that weight. As such, it was necessary to give individual weights for each of the markers used. The weights for each of the markers are stored in a configuration file, which can be changed by the user at any time.

Hip and torso markers	0.10
Head markers	0.20
Shoulder and upper arm markers	0.15
Elbow markers	0.20
Wrist markers	0.25
Thigh and knee markers	0.15
Shin and ankle markers	0.20
Foot and toe markers	0.25

Table 1: Marker weights for prediction averaging. The value represents the weight given to the raw marker data for the weighted average.

The actual selected weights which have resulted in the best looking animation for this project, were all between 10% and 25% of the original raw data. The weights for each of the markers were determined initially by reasoning, and then confirmed by testing. The weights are assigned proportional to the distance away from the root. The further away from the root, the higher the weight given to the raw data. The reasoning behind this is that we want to maintain the motion details of the feet and hands during high frequency motion, while the parts closer to the root generally move more smoothly. The weights chosen for each of the markers is shown in table 1.

The final step to do is to take the weighted average of our raw marker data $\vec{m}_{vicon}$ from Vicon and the calculated prediction position $\vec{m}$, using the appropriate weight s as found in the table as one would expect:

$$\vec{m}' = \vec{m}_{vicon} * s + \vec{m} * (1 - s)$$

Next frame, the averaged marker position $\vec{m}'$ becomes part of the prediction as described in equation (3). Given that this section was written under the assertion of available marker data, we should have nothing left to do but pass the marker data on to the inverse kinematics part of the pipeline. There is one exception to this rule, and that is when we have available marker data, but had recently dealt with a period of marker occlusion. In that case, the final steps are slightly different due to interpolation. This will be discussed in section 3.4.2.

3.4 Gap filling

While we have already attempted to detect and relabel markers in the previous chapters, marker occlusion is an inevitable fact for optical motion capture. Even during a normal walking motion, the swinging of the arms can occasionally completely obscure markers on the hip and torso from all the cameras, and even the marker on the top of the wrist might be obstructed from sight if the hand is turned upside down. As such, the periods during motion capture where there is no marker data available have to be filled in.

Gap filling algorithms are easier to use when dealing with an offline motion, as this essentially solves the two problems online gap filling faces: the length of

the gap is not known, and it is it not known where the marker will resurface. While there are offline algorithms to interpolate nicely, such as a spline, they require knowledge of both the start and the end of the gap. Given that this project deals with online motion capture processing, the only way to fill in the gaps in data is by means of extrapolating the trajectory of the marker before the gap, and to then interpolate back to the actual position once the marker is found again. This section will discuss the extrapolation and interpolation process, as well as the option to use biomechanical constraints of the human body to guide and slightly correct the extrapolation.

3.4.1 Extrapolation and tethering

For extrapolation, our work is already nearly done. To create an extrapolation vector, we use the prediction vector from the Kalman Filter, as was described in equation 3, and we keep this vector for as long as the occlusion lasts. Where the predicted marker position would then be averaged with the raw Vicon data, we can omit that step. This is due to the fact that there is currently no Vicon marker data to average with. This means that the final marker position $\vec{m}'$ is simply equal to $\vec{m}$. As a result, as long as there is no new marker data available, the prediction will continue on the trajectory it was on. The advantage of this method of extrapolating and filling the gap is that it's fast, as well as being consistent. The downside is that the prediction is essentially a linear prediction, which means that during longer periods of occlusion, the extrapolation has a chance to become inaccurate.

Regardless of the method of extrapolation, errors are inevitable should an occlusion last long enough. To counteract the possibility of a marker being extrapolated off into the distance, we can use constraints that we defined during the mislabeling detection, as described in chapter 3.2.1. During the calibration as the start of the recording, we will have determined these near-constant distances between markers in the same group. A violation of these constraints was defined to be any distance that did not satisfy equation (1). Should a marker at any time during the extrapolation fail to fall within these bounds, it will be tethered into place by iterating over all the other markers it is grouped with. During this iteration, the marker will be either pulled towards or pushed away from the markers of which it violated the calibrated distance constraint, until it reaches a spot where it no longer violates any constraints. An illustration of the process can be seen in figure 3. Iterating over the correct markers from the one that is the furthest away to the one that is the closest, the missing marker is moved to the edge of the calibrated acceptable zone as defined in chapter 3.2.1. This process is called every frame the marker is out of bounds.

One slight note on this process is that it is only applied to markers which are actually flagged to be missing. There might be a situation where there are markers of which the data is incorrect, but were not labeled as missing. In such a case, it could be the case that there is no stable state for the iterative process to converge to, because the acceptable areas of markers are completely disjoint. To prevent infinite iteration, there is a maximum limit of 5 full iterations. It is worth mentioning, however, that such a situation has never occurred during all

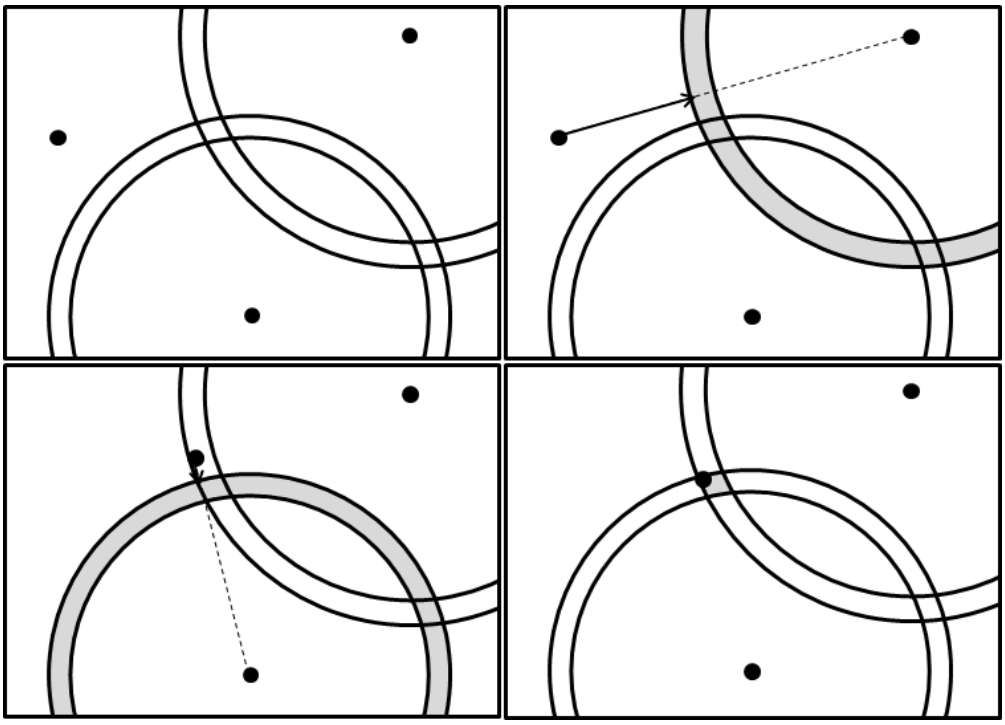

Figure 3: The iterative process of repositioning a marker back into the acceptable bounds.

experiments.

There are, however, slight drawbacks to the algorithm. The most obvious one being that we cannot tether to markers that are currently labeled as missing, as their position is unreliable. In the two-dimensional example, with two other markers to tether to, there are two areas. Logically, only one of these areas would actually be the correct one, but we are unable to distinguish between the two areas purely based on the calibrated distance. As such, it is possible that the tethering will actually send the marker to an incorrect solution. We could solve the problem by adding more markers to tether to, but that is the opposite of the thing we're looking to achieve with this project. Fortunately, the principle of the marker tethering algorithm is that a missing marker should always stay within the correct acceptable area. The reason for this is that as soon as it is flagged missing and leaves the acceptable area, it will be immediately moved back to the acceptable area.

With fewer markers, there will be even less certainty with the tethering, or even no possibility to tether at all. Nevertheless, the algorithm is still useful. By calling upon the tethering algorithm as soon as the extrapolated marker travels slightly outside the acceptable zone, it will be pulled or pushed back into it. This way, the extrapolated marker should theoretically never end up too far away from the actual marker position. Even when tethering is very limited, it reduces the error that can accumulate over longer periods of extrapolation. This should mean the distance between extrapolated position and reality is smaller after occlusion ends compared to without having used tethering, which

in turn produces smoother interpolation as the distance over which needs to be interpolated is smaller.

3.4.2 Interpolation

After the marker occlusion is at an end and new data is available, our prediction will no doubt still have been off by some amount, even with the tethering. This is why we will have to interpolate back to the raw data smoothly, to avoid very obvious 'snapping' into position which looks highly unnatural. Inspired by the work of Schuddeboom [Sch11], the interpolation method I use is a simple linear one. The interpolation will take place over n seconds, where n is once again fully customizable by the user in real-time during motion capture. For the purposes of this research, the value of n chosen which seemed to yield decent results is 0.1 seconds, which with the framerate of 100 frames per second results in 10 frames.

During these n frames, the interpolation will slowly go from the prediction to the raw data. During interpolation, the prediction will not be adjusted using the new data, although new sample states will be added to the sample history. At each step of the interpolation, the interpolated value $\vec{m}'$ of our marker m is a weighted average of the prediction and the raw data. Unlike the normal prediction, however, this weighted average changes linearly over the interpolation time. At each step i of the interpolation, the value for $\vec{m}' = \frac{i}{n}\vec{m}_{Vicon} + \frac{n-i}{n}\vec{m}$, with $i \in \{1..n\}$ and $\vec{m}$ denoting the raw data. Using this algorithm, you have a visually nice interpolation from prediction to actual data after occlusion, which is computationally very inexpensive when compared to more complex interpolation methods as splines [Ron10]. An illustration of this interpolation process can be found in figure 4.

Should a marker become occluded again at any point during the interpolation frames, the extrapolation process will start again as normal. Note that the prediction vector used as extrapolation might have changed slightly because of the way it is calculated. Both during the extrapolation and the interpolation, new frames are sampled and added to the sample history. These new frames affect the prediction vector more than the older ones, which can result in a slight adjustment of the direction the marker is being extrapolated in.

4 Inverse Kinematics

With the raw data processed, we now get to the part of Inverse Kinematics. In this chapter, we will explain the structure of our virtual skeleton, and how to scale the virtual skeleton to fit the body proportions of the actor. With the skeleton structure, we define how markers can be used to calculate the target world positions of joints. Calculating the joint orientations required to make a virtual character match the pose of the actor, we use the inverse kinematics algorithm Cyclic Coordinate Descent. We create a real-time animation by applying the joint angles we calculate every frame onto a virtual character mesh, which concludes the Performance Animation pipeline.

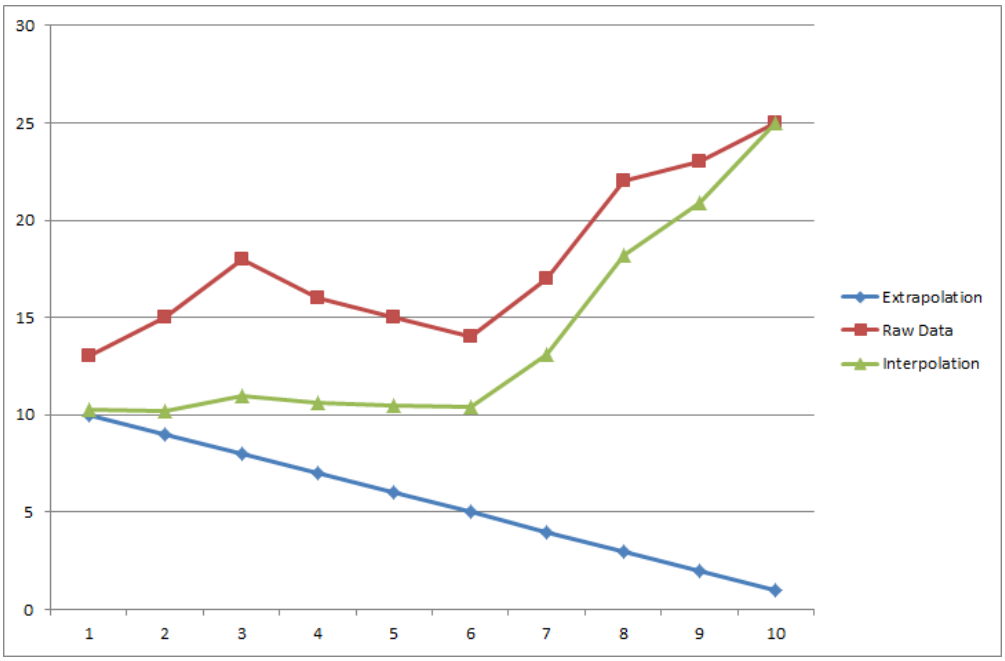

Figure 4: An illustration of the interpolation process over 10 frames

4.1 Skeleton reconstruction

After the marker data calibration phase described in chapter 3.2.1, we have a good indication of where the markers are relative to eachother. We can now use this information to measure our virtual skeleton. However, we first need to exactly define which joints and bones are in our skeleton and how they connect to one another. Too many joints, and we might not have enough marker data to accurately determine the orientation of each of the joints. Not enough joints, and the reconstructed animation might begin to look very similar to a stick figure, as there is not enough freedom to accurately apply the motions performed by the actor. We want to have a realistic human body animation, calculated in real-time from motion capture data. Given that the aim of the project is to ultimately reduce the amount of markers and possibly even reduce the amount of joints, the initial skeleton structure should only contain the bare necessities. The initial set of joints in our skeleton consists of the following joints: root, lower spine, clavicle, neck, skullbase, shoulders, elbows, wrists, hips, knees, ankles, and toes.

These joints already allow for most of the freedom we are able to see in a typical human body, and in the basic configuration we have the markers of the motion capture suit lined around those joints for optimal accuracy. While it is the goal to possibly move some markers around and to reduce the amount of markers on the body altogether, the idea for the base implementation of the pipeline is to have a set of markers which is 'complete', to act as a ground truth. Any experiments which change the marker set in any way will be compared to this configuration, to be able to see what the effects are.

Now that we have a list of joints, we still need to specify which markers

are attached to which joints, and where each of those markers is positioned relative to their joints. We need to know for each joint which marker is relevant for the calculation of the joint position. To illustrate, a wrist marker gives us no additional information about the location of a hip joint. The list of which markers correspond to which joints is all stored in a Joint Tree File. Discussed earlier in the section about mislabeling detection in chapter 3.2.1, this joint tree file is a user-customizable XML-style file in which all the joints in the body are given a structure and meaning. The skeleton is broken down into chains, each chain containing several joints. Most joints have a set of markers with the offset to the joint defined in the file. We can then use this offset in reverse to calculate the target joint positions based on the set of relevant markers. This target joint position will then be used during inverse kinematics to calculate the rotations required for the end effector to reach that position, which gives us the joint angles required to make the animation.

The last stage of the skeleton reconstruction is to properly scale the virtual skeleton to fit the body of the actor. In order to scale, we need to first decide how we can scale. We could attempt to scale per bone, per limb, or just the entire skeleton all at once. Scaling per bone would require a lot of marker information, as you would need to have at least one marker per joint to be able to have enough information to scale each of the bones individually. Given that the goal of this project is to remove markers that would be critical for scaling, this is infeasible. Scaling the entire body at once requires very limited information, namely only the length of the actor from toe to head, but does not really compensate for difference in body proportions. Keeping these things in mind, I have chosen to scale the virtual skeleton to fit the actor in 5 separate regions: the arms, the legs, and the spine from pelvis to head. This should allow the skeleton to adjust adequately to the actor while not requiring much information.

During the calibration of the subject, the skeleton reconstruction part of the program will use the marker positions to give a good estimate of the lengths of the bones between the joints. By looking at the average position of markers around each of the joints of for example the shoulders and wrists, a close enough estimate of the length of the arm can be calculated. By calculating the length of one of the regions of the actor and comparing it to the respective region in the virtual skeleton, we can derive a scaling factor for that region. This scaling factor can then be used to scale each of the bones in that region of the virtual skeleton, to ultimately fit the size and body type of the actor as closely as possible with very limited data.

The final part of the entire performance animation pipeline would be to actually produce the animation, which is done by deforming the mesh that represents our virtual character. The mesh used in this project already had an internally defined skeleton. This skeleton is the one that was used to determine the marker offsets and calculate the scaling. As such, the scaling we determined to make the virtual skeleton fit the length and body type of the actor can also be used to scale the mesh to fit the length and body type of the actor. The only thing left to do is that we need to calculate the joint angles we need to make the pose of the virtual character match that of the actor, which is done

by our Cyclic Coordinate Descent algorithm.

4.2 Cyclic Coordinate Descent

At this point, we should have processed the motion capture data to reduce the mislabeling of markers, filtered the noise, filled the gaps, and have a virtual skeleton made of bones and joints. What is left is to calculate the joint angles required for this virtual skeleton to match the pose of the actor. To be able to do this, we need to perform inverse kinematics. Inverse kinematics is about calculating the joint angles that are required for a chain to reach a desired point in space. Given that we need to update all our joint angles each frame to keeping matching the pose of the actor as closely as possible, with a framerate of roughly 100 frames per second, this means we have to do a lot of calculations. Due to this fact, we need a fast inverse kinematics algorithm that works online. The inverse kinematics used in my program is Cyclic Coordinate Descent [Wel93].

The working of the Cyclic Coordinate Descent algorithm is fairly simple. For any given chain of joint angles, CCD iteratively rotates one of the joints to make the desired end effector of the chain be as close to some desired point in space as possible, going from the start of the chain down to the joint closest to the end effector. This process can be repeated until a maximum number of iterations is reached, the error falls within a certain margin, or there is no improvement after one cycle over all the joints in the chain. Even if it's not perfectly possible to reach the desired point in space, CCD will very quickly give a result that is very close to the best solution. Because of this, it perfectly satisfies the constraints of being a fast algorithm that can work online.

The target point in space is dependent on the markers that are attached to the joint, as they were defined in the Joint Tree File. The way this point is calculated is by taking the average position of the markers relevant to the joint. Because the markers themselves are not necessarily located perfectly around the joint, we subtract the offset for each marker which we defined in the Joint Tree File as well. This offset itself is still not completely correct, as it was defined for the unscaled virtual skeleton. We use the scaling factor calculated during the skeleton reconstruction in chapter 4.1 to scale the offsets. Putting this all together, the final calculation for target position $\vec{j}$ of a joint is as follows:

$$\vec{j} = \frac{1}{n} \sum_{i=1}^{n} (\vec{m}_i - s * \vec{p}_i)$$

In this formula, n is the amount of markers attached to the joint, $\vec{m}_i$ the position of one of the attached markers, s the scaling factor for the region the joint is in, and $\vec{p}_i$ the rotated positional offset defined for marker m_i. There is however still a problem with this target position, which is that the marker positions are expressed in the world coordinate frame. To be able to calculate the rotations required for a joint to reach a target position, we need to have the target position in the local coordinate frame of the parent of the joint. We do this by subtracting the parent world position from the target world position, and rotating the resulting vector by the inverse of our current parent

joint orientation matrix. This gives the target end effector joint position in the local coordinate frame of the parent, which means we can calculate the required rotation to go from the current end effector position to the target.

Calculating the position of a joint is forward kinematics, which is slightly easier. The world position of a joint is defined as the world position of the parent joint, plus the local bone vector that separates them rotated by the joint orientation of the parent. This recursive definition eventually ends up at the root of the skeleton, of which the world position is simply defined as the average position of the hip markers.

Cyclic Coordinate Descent offers another advantage. Given that there's very little time between frames, we can assume that the differences in end effector positions between consecutive frames are rather small. As a result, the change in joint angles required to reach the target in the next frame should be very limited. CCD uses an initial estimate of the solution to then converge to a solution. Whereas CCD usually takes several iterations to reach a solution within typical margins of error, we now have a sequence of frames where the distance between markers in consecutive frames is close enough to use the pose of the previous frame as a decent estimator for the next one. By updating the joint angles every frame using CCD, we should not have to run CCD more than one or two cycles. We do, however, need to have our initial solution which we then iterate on the subsequent frame. While CCD should technically be able to find a solution for any pose, the actor performs the calibration of the system using a T-pose. Given that the virtual skeleton has a T-pose as base pose as well, finding this initial solution should be simple.

While CCD can function with chains of any length, we will initially have chains that are typically only two joints long, the joint we rotate and the joint we consider the end effector of that chain. CCD is able to work with chains of three or even more joints to calculate the angles required to bring the end effector as close to the target as possible, but this means that the position of the joints that are not the end effector are completely free. An example of what this means for the application of CCD to our virtual skeleton is that the chain from shoulder to wrist would still be able to position the wrist close to the target, but does not care where the elbow ends up to make this possible. Given that the aim is to produce natural-looking motion, there are limits to where an elbow could be in order to be considered natural.

The safest way to ensure that our motion looks natural is to stay as close as we can to the original motion of the actor. We do this by rotating every joint individually, which means every joint has their own target joint position which we attempt to solve for. This produces good results, but is unfortunately limited. The reason for this is that the purpose of this project is also to remove markers, and we will therefore not maintain a set which has at least one marker per joint. This means that the position of joints such as the elbows and knees might end up in unnatural positions in some experiments, as described in chapter 5.

5 Experiments

With the pipeline in place to transform raw motion capture marker data into an animated character, it is now time for the experiments. The goal of the experiments is to reduce the amount of markers in order to find the 'minimal' markerset. This minimal markerset is the set of markers which still provides enough data to produce decent looking results though the performance animation pipeline. Next to reducing the amount of markers, the experiments also focus on potentially reducing the amount of joints. While the base skeleton allows for rotational freedom around all three axes for joints such as the root joint located in the pelvis area, it is worth examining how much this freedom actually contributes to the overall naturalness of a motion. Similarly, locking joints like the lower spine and the neck may not result in a difference in the perceived naturalness.

The influence loss of marker data or joint freedom may have on the naturalness of a motion will be tested through a series of experiments. To limit the amount of variables changing in each experiment, each experiment will use the same raw marker data, but using a different Joint Tree File. The experiments are subdivided into three categories: spine, arms, and legs. What these categories indicate is that the only difference in setup from the ground truth motion is by having removed markers or joint freedom from that particular body area.

The animations produced by the experiments will then be tested in both a subjective way and an objective way. The subjective way is by performing a user case study in which users will be able to grade the naturalness of each animation. The objective way is by measuring the error between the ground truth and the experiment, using a distance metric described in chapter 5.3. The ultimate goal is then to see whether there is a correlation between the subjective analysis and the objective analysis.

5.1 Experiment setup

I want to preface this by saying that I am completely ignoring the process in which Vicon labels the markers. I assume that some markers are placed asymmetrically so that Vicon can detect which way the person in the motion capture suit is facing, but that is not the goal of these experiments. Starting out using the full markerset (figure 2 in chapter 3.1) combined with the base joint set (chapter 4.1) as ground truth, during my experiments I can omit markers or lock joints at will. By discarding the problems that may arise for Vicon when I attempt to use a reduced marker set, I am still left with a varied set of problems. By having fewer markers, I have less information as to the whereabouts of my body parts which leads to a set of difficulties, as was already briefly prefaced in chapter 4.2.

In case of marker occlusion, it can occur that I am left with no information at all. For a joint halfway along a joint chain (e.g. elbow) this is less of an issue, as there is not much freedom for such a joint to be given that I still have information about the position of the end effector. However, when I lose all the information for an end effector (e.g. wrist), I am left with a complete gap

and have to rely solely on the extrapolation of the position of that body part. Therefore, part of what I will be looking at during these experiments will be how far it is possible to go with omitting markers. I will also be investigating if errors such as marker occlusion or marker mislabeling occur more frequently in specific motions.

The second big part of the experiments was already briefly referred to in the previous paragraph. Given that the entire goal of performance animation is to produce natural-looking motions, naturalness is still a very important part of this project. To produce the motions, I rely on the inverse kinematics technique named Cyclic Coordinate Descent as it was explained in chapter 4.2. Given that the amount of data for the end-effector positions is less or even potentially incorrect due to the reduced markerset, CCD might produce results which are less natural than is acceptable. Not only is this project about whether I still have enough data from my reduced markerset, but whether this data is sufficient for CCD to produce a natural motion. Similar to before, different motions might produce different results with the same reduced markerset. The question is therefore not to find a single 'ideal' marker set, but to show which markers and joints are vital for correctness and naturalness of a varied set of motions.

One important remark would be that, similar to reducing the amount of markers used, it is also possible to reduce the amount of joints, or to limit their freedom. Given that there is less information available, it is entirely reasonable to assume that there is not enough information for all the finer details of the motion. Alternatively, certain joints might not even be required for the overall naturalness. It is possible to imagine that subtle differences in the movement of the pelvis and sternum do not impact the perceived naturalness during certain motions (e.g. calm walking motion). Keeping this in mind, we need to use varied motions for our experiments, to test the effects of locking such freedom on both high-energy and low-energy motions.

To be able to determine a relation between naturalness and amount of markers and joints I will first have to identify a couple of sets of reduced markers, as well as record a series of motions to test if there are particular motions where certain markers or joints are critical to maintain naturalness. I subdivide the experiments into three body parts, meaning each recorded motion will be used in an experiment more than once. To generate the animations for the experiment, I store all the raw marker data from Vicon. I then use this data to 'play' the motion again, allowing me to apply different settings each time around. These settings can be found in the unique Joint Tree Files, as was described in chapter 4.1. This way I can generate the ground truth animation and all the animations for the experiments using the same raw data, opposed to having to record each motion individually for all the experiments. This ensures that the only difference between the ground truth and the experiments is the change of the parameters.

This still leaves the process of choosing valid motions and experiments. First I narrowed it down to a selection of motions that covered different aspects, and focused in varying levels on the different parts of the body. Then, I used the distance metric which will be discussed further in section 5.3 to do initial filtering

	Spine experiments	Arm experiments	Leg experiments
Dancing	✓	✓	✓
Grabbing	✓	✓	
Jumping	✓		✓
Kneeling	✓		✓
Running	✓	✓	✓
Stretching	✓	✓	
Walking	✓	✓	✓
Waving	✓	✓	

Table 2: A table describing the motions that are used and which experiments applied to them.

of the sets, which will be discussed a bit more in section 5.1.1. Having filtered out some extreme cases, I was eventually left with a collection of experiments that should cover the range of different motions and different levels of reduction as well as possible, leaving further judgment open for people to give in a user case study. Using the results of that study, I should be able to determine which differences between the ground truth and the experiments are most distracting, and whether those differences show any correlation to the type of motion or the distance metric. As an example, for a simple walking motion the position and orientation of the hand, elbow, and shoulder might be less relevant to the perceived naturalness, whereas the correct placement of the feet might be less relevant during a running motion.

Before going more into detail what these experiments actually entail, an overview of all the motions and the experiments applied to them can be found in table 2. The motions are chosen because they should be easily identifiable for the people judging the naturalness. The reasoning behind this is that people might be inclined to rate the naturalness of a motion differently if they cannot instantly recognise the motion the actor is performing. As mentioned before, the motions feature a wide range of actions, which should cover both high-energy and low-energy motion, while emphasising different parts of the body in varying levels. As example, waving focuses on the smooth movement of the arm while the head is generally pointing in the same direction to look at the person you're waving at. Grabbing is more about the way the torso and arm together bend and stretch to reach a point in space. This shows that the parts of the body that are being focused on are slightly different throughout all motions.

The reason why the spine experiments are relevant to every motion is because they are the core of every motion. The grabbing, stretching, and waving motions are generally stationary and very arm-focused, which is why we will not get any useful data on the legs. Similarly, the arms are pretty much unused in the leg-focused jumping and kneeling motions, which means we do not run the arms experiments there. Dancing and running incorporate the entire body, and walking simply the most common motion, which means we run all three experiments on those.

Given that naturalness is a very complex concept with many variables, it is infeasible to draw conclusions from testing a reduced marker and joint set for the entire body at once. To simplify matters slightly, I divide the experiment by only changing the parameters of one area of the body at a time. These areas are the spine (from pelvis to head), the arms, and the legs. I will only remove markers and joints in one area, while keeping the rest of the body the same. This should allow me to be slightly more specific about the results of the experiments, as I can tie the loss of naturalness to a specific area.

5.1.1 Discarded experiments

The goal of the experiments was to be able to test certain variables of the marker and joint setup, by creating experiments that try to isolate one variable to compare with the ground truth at a time. However, to make experiment setups worth including in the user case study, they need to differ significantly enough from the ground truth. If one setup produces results that, according to the error metric from chapter 5.3 are too close to the ground truth, including that setup in the user case study will not provide much additional information. To illustrate, during the process of setting up the experiments it appeared that simply removing a few markers from a bodypart that already had multiple markers on it, such as the torso, would result in motion that was nearly identical to the ground truth. In general, the only markers that seemed to matter were those located on joints. Markers located on the bone inbetween joints, such as the upper arm markers, thigh markers, and shin markers, did not affect the accuracy of the motion at all. While this is not very surprising, there are two experiments which seemed promising yet ended up completely unusable for the user case study.

The first of these two experiments was an initial test to see how many markers could be removed, in an attempt to find the 'minimal' set of markers. Every bodypart that had multiple markers was reduced to one or two markers each, excluding the hip which retained three markers. Two markers was only allowed in the case the body part had a rotational degree of freedom for which more than one marker is needed to calculate. This meant that the torso was brought down from six to two markers, the head from four to two markers, and the hips from four to three. The shoulders, elbows, knees, and ankles kept their single marker, but the upper arm, top wrist, thigh, shin, and toe markers were all removed. This meant that every bodypart still had the required information to properly calculate the orientations using inverse kinematics. The removal of these extra markers did remove the safety of redundant information in case of marker occlusions, but was on average surprisingly close to the ground truth, only measuring an average average error (chapter 5.3) of $0.039m$ and an average largest error being $0.107m$ taken across all motions. While this experiment might have been interesting to compare to the ground truth in a user case study, it violates the setup of only testing the parameters of one bodypart at a time. Experiments that combine multiple body parts is something reserved for future work, which will be discussed in chapter 8. Nevertheless, this experiment served to illustrate that marker occlusion is not a very critical issue, and markers

hold little value if they are not directly relevant to determine joint position or orientation.

The second of the two experiments had the problem that it was too unnatural to be used in the user case study. This experiment involved locking the spine to keep the torso fixed in a straight position, while leaving the complete head freedom intact to rotate along all three axes. This experiment was designed to isolate the variable and test the influence the freedom of the spine had on the perceived naturalness. The problem with this approach was because of the way the inverse kinematics algorithm attempted to solve for the target position of the head joint. The pose of the actor in most clips was relaxed, and somewhat slouching. This meant that the spine was slightly bent, with the neck somewhat bending forwards. By having fixed the torso in a straight position, this meant that the solution that would place the head joint as close to the target position, the neck had to be bent in a very extreme way, to compensate for the slouching pose of the actor. This unnatural head orientation was very distracting, which meant that it would be rated low consistently in the user case study. As such, it would provide no extra information, and was discarded. The result from this failed experiment was that locking the spine could not be done individually, and required to simultaneously lock the freedom of the head, only leaving the freedom to turn the head from side to side.

5.1.2 Spine experiments

One of the three parts of the experiment will focus on the spine, which was defined as the part of the body from pelvis to head. In an average motion, the most distance will be traveled by the limbs. This means that, relative to the limbs, the spine will move not as much. Therefore, one could assume that accuracy of the position of the chest area is not as important to the overall correctness of the motion. Given that the top half of the torso is a relatively rigid area with little freedom of joints for most people, the position and orientation of the chest is nearly explicitly indicated by the position of the shoulders. While this theoretically would imply that we do not need any markers on the chest area, and that we would be able to solely rely on the information given by the shoulder markers, this will not work as well if the target is not keeping his back straight as would be the case in a slouching gait. Nevertheless, we should not need more than just two markers on the torso: one on the front, and one on the back. This set of two markers for the torso will be compared to the set of six torso markers in the ground truth.

Another area where I feel there are not many markers required for an adequate level of detail is the head. To retain the full information required for all the freedom of the head, you would technically only need two markers, on either side of the head. Given that the head is fixed on the neck, you only need two markers to determine the rotation of the head, if we place our markers correctly. While it is already a valid experiment to test if this reduced amount of information is enough to calculate the position of the head, you could also take it one step further. For certain motions, the head is just facing the same direction as the rest of the body. This raises the question whether the freedom

of the head to tilt up and down or to rotate from side to side is really required for naturalness. If that is the case, one would only need a single marker on the top of the head to determine its position. I will test setups that include combinations of locking the rotation or the tilting of the head and compare them to the ground truth. A setup with only one head marker would be equivalent to locking the rotation of the head and having two head markers, with the exception of having no second set of data in case of marker occlusion. These head experiments focus primarily on the importance of the joint freedom rather than the risk of occlusions, so there has been chosen for to use two markers in the setups. The two markers we use on the head are one on the right side and another on the left side of the head, whereas there are four head markers in the ground truth.

The most important area of the spine where we can experiment with parameters is the pelvis area. Given that the pelvis is very important as that the root of our entire skeleton is located at the center of the pelvis, we can't reduce the amount of information too severely, as any error in the root joint would affect our entire motion. To cover the complete range of freedom for the pelvis, we would technically only need three markers. At the same time, it is worth questioning whether all degrees of freedom of the pelvis are actually required for a natural looking motion. I will therefore also experiment with locking the pelvis to only be able to rotate around the vertical y-axis. Theoretically speaking, if the freedom of the pelvis is not necessary, having only two markers would be enough to calculate the rotation around the y-axis. The downside to this is that two markers would become vulnerable to marker occlusion, as even a single occluded marker would make it impossible to accurately calculate the position or orientation of the root. Keeping this in mind, I will determine if it is possible to use a markerset containing only three markers on the hips, and compare this with the ground truth of four markers on the hips, as well as the aforementioned experiment where the degrees of freedom of the pelvis are locked.

One final part of the spine we can test is the spine itself. In the base skeleton, we have the lower spine joint to allow the upper torso to twist and bend slightly. However, when we observe the motions selected in section 5.1, it appears as if we do not use the freedom to twist and bend our spine very often. This is why the final part of the spine experiments allows to lock the rotational freedom of the spine, effectively locking the torso to be one rigid whole with the pelvis. The user case study will then show how much people pay attention to the freedom that is lost by doing this, and therefore how it affects the perceived naturalness.

From these possible parts of experimentation, I have assembled three sets of experiments. From initial testing, it appeared that any single experiment did not affect the motion enough to be considered interesting enough to use in a user case study. This is why I have combined the different experiments here to form the three sets as shown in table 3. A checkmark indicates that parameter is altered from the ground truth. As example, a checkmark for head rotation means the rotational freedom of the head is disabled for that experiment. Experiment S1 involves all the aforementioned reduction of markers, as well as

	Experiment S1	Experiment S2	Experiment S3
Chest markers	✓		✓
Head markers	✓		✓
Head rotation	✓		✓
Head tilting		✓	✓
Hip markers			✓
Pelvis freedom	✓	✓	✓
Spine freedom		✓	✓

Table 3: A table containing which individual spine experiments are combined to form the experiment sets used in the user case study.

the head and pelvis freedom, keeping the spine freedom intact. Experiment S2 does not reduce the amount of markers, but instead tests what happens when virtually all freedom is removed from the spine by locking the freedom of all joints except the head rotation. Experiment S3 combines experiments S1 and S2 to test all parameters being changed at the same time, in order to determine the loss of naturalness when dealing with the 'minimal' markerset for the spine region.

5.1.3 Arm experiments

The second of the three parts of the experiments will focus on the arms, from shoulder to wrist. Unlike the spine, the arms tend to be more mobile, and generally move at a higher velocity. This means that you will generally want to have multiple markers on the arms to keep proper track of the movements. However, due to the fact that the arms move around a lot more relative to the spine, there might be a lot higher leniency with regard to naturalness when it comes to positional errors. It might be the case that there are very noticeable errors in the position of the wrists, but they will not be perceived as unnatural because the resulting pose is still deemed plausible.

Starting with the wrists, we need to have enough information to know the orientation of the wrists. Without this information, we cannot tell which direction the hand is facing. For motions such as waving or picking up, the palms need to be facing a certain way. If we want to know the orientation of the wrists, we need to have a minimum of two markers. If we don't really care about the orientation of the wrists, such as would be the case for motions which are not focused on the hands, we could affix the wrist to the rest of the arm without losing much naturalness. In such a situation, we would only need to know the position of the wrists, which can be done with a single marker. To compare the naturalness and robustness under marker occlusion of these setups, we will compare both setups to the ground truth setup which has three markers positioned on the wrist.

Looking away from the wrists, we can argue that we do not need a lot of marker information on the rest of the arms. We discussed briefly during the spine experiments that we might not need the markers on the chest, as

the markers on the spine already define the limits of the chest. To confirm this relation, and to confirm that we could treat the upper half of the torso as one single rigid body, we will attempt to do the opposite here: we will use the position of the chest markers to estimate the position of the shoulder joints. This could mean that we can simply choose either of the sets of chest or shoulder markers, without losing significant information. While this is an interesting experiment, it was chosen to not be performed during this thesis because the setup of these experiments was to only test one body region at a time. This topic will be discussed a bit further in the future work section, chapter 8. For the purposes of these experiments, the shoulder markers were kept. The upper arm markers were considered optional for some experiments.

Having established positions for the shoulders and the wrists, the rest of the arm can be reconstructed using Inverse Kinematics. Given that we know the lengths of the bones in the arm, we can find solutions for rotations of the shoulder and elbow joint such that the wrist ends up where the markers indicate it should be. The only freedom we really have in this Inverse Kinematics problem is where the elbow ends up as a result of the roll of the upper arm. Technically speaking, there is a range of rotations an arm can be in while still being physically possible. This might lead to the idea that the elbow position should not affect naturalness too much. This means that, should my Inverse Kinematics method be accurate enough, any information provided by markers on the elbows or upper arms is superfluous. I will test this hypothesis during the user case study by comparing it to the ground truth marker set, which still has the added information of the elbow marker position. This elbow marker position can then be used by Inverse Kinematics to guarantee that the elbow joint is as close as possible to a position that should be very natural: that of the actor.

While there aren't as many parameters to experiment with as in the spine experiments, I was still able to form three sets of experiments for the arms, as can be seen in table 4. As before, a checkmark indicates that something is different from the ground truth. Experiment A1 focuses on what happens when the elbow is free as the elbow marker is removed, and whether two markers instead of three does not encounter problems due to marker occlusion. Experiment A2 maintains the full information of the elbow, but tests whether completely removing all but one marker from the wrist still allows it to look natural, or whether this locked wrist will distract people. Experiment A3 is designed in similar fashion as the third experiment of the spine experiments, which is to combine all the variables to test the performance of the minimal markerset.

5.1.4 Leg experiments

The final part of the three parts of the experiments is focused on the legs, from hip to toe. While technically both are limbs, the legs operate under slightly different constraints than the arms. In general, the placement of the feet is a bit more important than the position of the wrists, given that it is important that feet should be in contact with the ground during most motions. Whereas wrists might be slightly off, feet hovering at different heights and possibly penetrating

	Experiment A1	Experiment A2	Experiment A3
Upper arm marker		✓	✓
Elbow marker	✓		✓
Wrist markers	✓	✓	✓
Wrist rotation		✓	✓

Table 4: A table containing which individual arm experiments are combined to form the experiment sets used in the user case study.

the ground can become unnatural rather fast. Nevertheless, once the position of both the feet and the hips are known, the rest of the legs is simply in-between and can most likely be solved using Inverse Kinematics similar to the situation of the arms. Due to this, the leg experiments will still be reasonably similar to the arm experiments.

Using this reasoning, we can conclude that we only need to have markers around the foot. If we want to retain the full freedom of motion for the feet, I will have to keep into account that there is both some rotation possible around the ankle joint, as well as flexion of the foot. This means that we need three markers on the foot: on the ankle, the top of the foot, and the toes. We can use the ankle marker as an indication where our heel is supposed to end up, while we can use our foot marker for the rotation of the ankle, and the toe marker for the flexion in the foot. The correctness and naturalness of the rest of the leg as a result of the Inverse Kinematics solving will be compared to the ground truth set, as per usual. This allows me to always place the knee in a position that is physically possible and likely to be considered natural, just like in the experiments of the arms.

There are two additional experiments for the foot. Similar to how we might not need the rotational freedom in the wrist, we might not need all the freedom in the feet either. It is possible that we do not need the flexion in the foot, because it simply might not be sufficiently noticeable in our motions. If we lock the flexion of the foot, we can simultaneously omit the marker from the toes as it would give us information we no longer require. Whether the flexion in the foot is required for naturalness will be tested using a user case study. Similar to this experiment, we can also wonder whether we truly need the freedom around the ankle. If observers do not notice the flexion of the foot, it is reasonable to question how much the joint freedom of the feet play a part in the overall naturalness. By completely locking the ankle joint and thus eliminating all the freedom around the heel (and thus making the foot marker obsolete), the feet are essentially permanently locked in a 90-degree angle to shin bone. This experiment reduces nearly all the information from the legs, which possibly makes it near-impossible to provide good looking results from Inverse Kinematics. To ensure this is not affecting the result of the experiment of the feet freedom, I will also conduct this experiment with extra marker information for the knees, which guarantees that the Inverse Kinematics has enough information to calculate a relatively natural pose.

	Experiment L1	Experiment L2
Thigh marker	✓	✓
Shin marker	✓	✓
Foot flexion	✓	✓
Ankle freedom	✓	
Knee marker		✓

Table 5: A table containing which individual arm experiments are combined to form the experiment sets used in the user case study.

After the initial selection process using the distance metric described in chapter 5.3, it turned out that the thigh and shin marker had absolutely no influence on the position calculation of the joints, which means these markers are omitted in all leg experiments. Similarly, it turned out that the foot flexion did not matter enough to be included in the user case study either. As a result, there were only two interesting variables left for the user case study: the knee marker, and the freedom of ankle joint. As such, there are only two leg experiments. Experiment L1 has the ankle locked, and still has the knee and ankle markers. Experiment L2 has no knee marker, but still has both the ankle and the foot marker. The list of what the leg experiments entail can be found in table 5.

5.2 User case study

In order to properly rate the naturalness of the motions produced by the experiments, we will need to conduct a user case study. The motions used for this experiment are listed in table 2. Each of the three different types of experiments has either two or three experiment sets, which can be found in tables 3, 4, and 5. This means that, in total, the amount of experimental animations to be used in the user case study is 8 times the 3 different spine experiment sets, 6 times the 3 different arm experiment sets, and 5 times the 2 different leg experiment sets. This adds up to 52 total experiment clips. To have a base value to compare the user-given results of the experiment clips with, we also need to 1 ground truth clip for each of the 8 motions, resulting in a total of 60 clips.

The way the user case study will be conducted is by performing a survey online, where users are first given a quick introduction and explanation. They are asked to view every clip *once*, and to then rate the naturalness of every clip on a scale of 1 to 7. The term naturalness is explained to every participant as follows:

> The term 'naturalness' means how natural looking you think a motion is at first glance. Naturalness in this case primarily pertains to the actual pose and movement of limbs, rather than the type of motion. Example: dancing in an unusual way is not unnatural, while an arm taking an infeasible pose *is* unnatural.

The reason for this is due to the fact that not every participant in the survey can be expected to be familiar with the term naturalness. The reason why the participants are only allowed to look at the clip once is because the user case study focuses on the first reaction people have. Minor errors that don't instantly draw attention to themselves and go unnoticed if you only watch a clip once without knowing where to look can essentially be discarded, as this project focuses on performance animation. In a continuous animation as a result of performance animation, people would not be able to watch a clip more than once either.

The clips were put in a random order which does not include seeing any particular motion twice in a row, but the order is the same for all people. The individual clips have a duration between 5 and 10 seconds, which results in an overall survey duration of 10 to 15 minutes. Finally, to have any level of statistical relevance, we will need at least 30 responses.

5.3 Distance metric

The other part of the motion analysis is the objective analysis. We want to find a metric that is able to compare two motions and give some error as result. The two motions that are compared use the same raw marker data set, originating from the same motion. So, as an example, the distances calculated will be between two walking motions. Given that our goal is to determine the difference from an experiment to the ground truth, every comparison is made between one of the experiments, and the according ground truth.

The comparing is by taking the Euclidean distance between predetermined points on the body in both motions. These points need to be chosen so they take a combination of factors into account. We want to look in particular at two properties of parts of the body: the world position of joints, and their difference in orientation. Comparing the positions and orientations of joints would give us an objective measure of how different two poses are, as was already previously discussed in chapter 2.5. We assessed earlier that removing the marker information of the knees and elbows would result in them being able to be placed anywhere by the inverse kinematics algorithm, including positions that might not be very natural. Similarly, when we lock the freedom of the spine, our upper torso and head might be placed in positions that might not be completely considered natural.

At the same time, we also want to be able to somehow capture the orientation of certain joints. Joints that we are particularly interested in are the feet, wrists, and head. One could imagine that the joint orientation of the wrists for a waving or grabbing motion would be more important than for a walking motion, where the orientation of the feet should play a bigger role for the overall naturalness. It is also worth noting that we lose a bit of rotational freedom of the pelvis in some experiments.

Given that the subjective results of the user case study will be based on the difference in poses, our objective measurement should focus on the same variable that determines the poses: the joint orientations. We can use there orientations together with the bone lengths to use forward kinematics to calculate the world

joint positions of each of the joints used in an experiment, but this would omit the part where we also want to use the orientation of the joints. An example would be where the head joint world positions in two poses are exactly the same, but one involves a strong rotation of the neck. This would look completely different when observing the motion, but would be undetected by our current metric. To solve this, we can use a point displaced from the joint by a local vector, similar to how the marker offsets were defined during chapter 4.1. If the neck is rotated, the point is rotated along, which means we can once again detect the difference. The points we will be using are the markers, as we defined in figure 2 in chapter 3.1. The difference is that we are reconstructing them using forward kinematics.

We now have our points on the body which we can easily track, and calculating the Euclidean distance between two 3D-points is easy. All that remains is to determine how this Euclidean distance will actually determine our error metric. We can look at the average error between the two same markers, taken over all frames of the animation. We could also look at the largest error between two markers in a single frame, which would indicate a spike. Given that the purpose is to find a metric which correlates with naturalness, it is not entirely clear what people base their judgment of naturalness on. As such, it is impossible to discredit any of these error metrics before testing their correlation to the ratings of naturalness granted through the user case study.

One last option is to either combine the results of all the markers, again in two ways. We can either take the average distance of all the results of all the markers, or we can take the largest distance out of all the results of all the markers. This leaves us with a set of four possible metrics: the Average Average Error (AAE), the Largest Average Error (LAE), the Average Largest Error (ALE), and the Largest Largest Error (LLE). So to somewhat clarify: the AAE takes the average error of all the average errors of all markers across all frames; the LAE takes the largest errors of all the average errors of all markers across all frames; the ALE takes the average error of all the largest errors of all markers in a single frame; the LLE takes the largest error of all the largest error of all markers in a single frame.

The error metrics used during the selection of worthwhile experiments in chapter 5.1 were both the AAE and the ALE. The reason for this is to only include animations in the user case study that have a significant error. By taking the average of the AAE and ALE metrics across all motion clips for one experiment, we can discard experiments that will likely not yield any interesting data due to their strong similarity to the ground truth. If the average of all average errors was smaller than $0.04m$, while the average of all largest errors was also smaller than $0.10m$, then a clip was considered not interesting enough for the experiments. The average AAE and ALE values of the selected experiments taken across all clips can be found in 6. The complete overview of all the experiments can be found in appendix 8.

There is one however problem with these group of metrics, which is that they are somewhat biased to give higher errors for markers that are further away from their joints. As example, joint that determines where the torso markers end up is the lower spine, which is a lot further away from the torso markers

Experiment	AAE	ALE
S1	0.046	0.079
S2	0.091	0.213
S3	0.144	0.259
A1	0.045	0.106
A2	0.062	0.121
A3	0.075	0.181
L1	0.053	0.106
L2	0.107	0.261

Table 6: The average of the error metrics taken across all motion clips for each experiment

than the foot joint is from the toe marker. Seeing as we do not know yet what it is that determines naturalness, we may have to also consider a metric that is normalized based on the distance to the joint that directly determines the position of a marker. One particular case would be when the spine and neck are locked, which would mean that the reconstructed positions of the head markers are directly determined by the orientation of the root joint. In that case, the error of the head markers would be divided by the distance from the head marker to the root in order to normalize the error. To see the difference in correlation, the results of both sets of four metrics will be plotted against the results of the user case study.

6 Results

The user case study was performed by 40 people in total, from a varied background. The survey was taken both by people that possess some level of experience on the subject, namely students, as well as people who were completely new to the subject, both groups participating in roughly equal number. Across the grades awarded to the 60 clips used in the survey, there were no noticeable consistent differences in grades given between the two groups.

First, we discuss the results from the user case study itself, as there are several observations to be made about the importance of certain markers and degrees of freedom. Afterwards, we determine the correlation between the user case study data and the distance metrics by calculating the Pearson product-moment correlation coefficient. We show graphs of the error metrics being plotted against the results from the user case study, and highlight which metric gives the highest correlation.

6.1 Subjective results

The grades used in further analysis are the average of the grades given by all 40 participants. The average grades given for the ground truth motions ranged from 5.05 to 5.48 out of a possible 7.0, with the average of all the ground truth

	GT	S1	S2	S3	A1	A2	A3	L1	L2
Dancing	5.48	5.23	5.03	4.70	4.75	5.15	2.35	5.23	2.03
Grabbing	5.05	5.18	3.38	2.85	3.70	4.23	2.95		
Jumping	5.38	5.10	3.55	2.98				5.35	1.40
Kneeling	5.08	5.10	3.18	2.93				5.23	1.15
Running	5.35	5.20	4.33	3.85	3.78	4.30	3.13	5.35	1.95
Stretching	5.33	5.08	4.00	3.90	2.93	5.00	2.75		
Walking	5.13	4.03	5.30	3.30	4.88	3.55	3.80	5.10	1.90
Waving	5.18	5.05	4.20	4.63	3.05	3.95	2.23		

Table 7: The table containing all the results of the user case study. The table represents the motions used, and which experiments were applied to those motions. If an experiment was not applied to a particular motion, the entry is blank. The motion names and experiment codes are as they were defined in section 5.1, with the addition of GT standing for ground truth.

grades ending up at 5.25, as can be seen in table 7. For easier interpretation of the results, we can list the user case study result value of each experiment compared to the ground truth. These values can be found in table 8.

Looking at the results in table 8, we can point out some interesting results. The most obvious are the negative values, which are instances where an experiment was considered on average to be more natural than the ground truth. In five out of eight motions, the ground truth was either the highest rated motion or tied for highest. For two motions, the ground truth was rated second-highest, with a maximum difference to the highest rated motion of 0.18. In one particular motion, the kneeling motion, the ground truth motion was actually rated third-highest, with a maximum difference of 0.15.

Upon closer inspection of the motion where the ground truth was deemed less natural than two of the experiments, the kneeling motion, there turned out to be a valid explanation for this occurrence. Due to the nature of the motion, there had been marker occlusion as the top of the feet was no longer visible. As a result, the feet had been placed in a somewhat awkward position. The experiment which locked the rotation of the ankle, experiment L1, had avoided this problem altogether and was therefore graded as slightly more natural. While experiment S1 also scored better than the ground truth for the kneeling motion, this difference was only 0.02, which is not a significant difference.

Another thing we can see is that experiment L2, which had no knee marker information, was considered to be really poor performing across all motions. The same can be stated about experiment A3, which contained no elbow marker or wrist rotation. On the other side of the spectrum, experiment L1 managed to consistently perform nearly as good as the ground truth did every motion, only scoring an average grade of 0.03 less than the ground truth across all motions.

A more interesting result is experiment A1, which had no elbow marker but retained the wrist rotation information. It appears that the best case scenario for that experiment produces very good results which are almost as

Experiments	Max worse	Min worse	Average worse
S1	1.10	-0.13	0.25
S2	1.90	-0.18	1.13
S3	2.40	0.55	1.60
A1	2.40	0.25	1.40
A2	1.58	0.32	0.89
A3	3.13	1.33	2.38
L1	0.25	-0.15	0.03
L2	3.98	3.23	3.60

Table 8: A table containing the results of the user case study. Each experiment grade is offset against the ground truth grade.

good as the ground truth, but is rated very poorly in others. In fact, the motion where it was only 0.25 points below the ground truth, is the walking motion. The motions experiment A1 performs poorly at are the waving and stretching motions, which focuses strongly on the arms. The dancing motion also scored relatively well, only 0.73 below the score of the ground truth. The grabbing and running motion already performed a bit poorer, both scoring over a point less than the ground truth. It would appear as the motion focuses more on the arms, the correctness of the elbows becomes more and more important to the overall naturalness.

The rotational freedom of the wrists, as tested by experiment A2, seems to follow a different trend. The motion where experiment A1 performed best, the simple walking motion, is the worst performance for experiment A2. There is, however, an explanation for this somewhat unusual grade. Investigating the animation for experiment A2 on the walking motion reveals that there seems to be a strong case of occlusion going on with the right hand, as it is moving upwards in an unusual angle where the hand in the ground truth stays horizontal. If this is indeed a case of marker occlusion resulting in wrongly predicted hand movement, our distance metrics should be able to confirm this in the correlation analysis.

Having discussed the results of the leg and arm experiments, we also take a final look at the spine experiments. Experiment S1, which locked some of the rotational freedom of both the pelvis and the head, seems to perform quite well throughout all motions. The motion where this experiment performed worst, the walking motion, is in fact a sort of outlier when it comes to the error. Not counting the walking motion, the second-highest loss in naturalness experiment S1 registered when compared to the ground truth was only a minor 0.28 points lower. As mentioned prior, the walking motion seemed to be subject to some measure of marker occlusion. Because we removed a lot of markers from our chest area, there are slight problems towards the end of the animation.

Looking at experiments S2 and S3, we notice that they are not even performing that poorly, given that we essentially locked the entire torso into one rigid body. Removing as many markers from the head, torso, and hips ev-

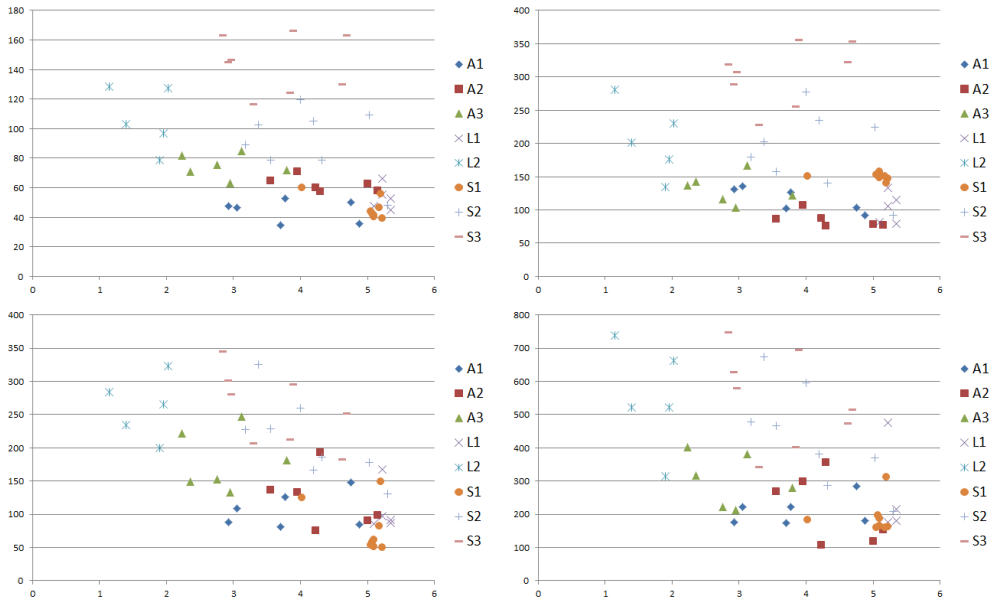

Figure 5: Error metrics plotted against user case study scores. Top left is AAE; Top right is LAE; Bottom left is ALE; Bottom right is LLE.

idently does not have as much of an impact as do the removal of the wrist or knee marker. Nevertheless, we still see quite a steep drop in naturalness when comparing the results to that of experiment S1. It would appear that the freedom to bend the torso still plays a major part in the naturalness of a motion. The only outlier comes from experiment S2 in the case of the walking motion. It would appear that maintaining the chest markers was essential for this particular motion, as there is no visible problem towards the end, unlike for experiment S1. Experiment S3 consistently scored lower than both S1 and S2, but given the fact that there are only 7 markers left on the entire spine area and only 1 degree of freedom which is the root, the results for experiment S3 are still quite impressive.

6.2 Correlation results

We will begin the correlation analysis by first plotting the distance metric we defined in chapter 5.3 against the user case study results. Given that we cannot easily determine which of the four variants (AAE, LAE, ALE, LLE) most accurately captures the details an observer uses to determine naturalness, figure 5 shows all the plots side by side. While LLE does not show much structure, the other three methods do seem to show some measure of correlation. When looking at AAE, the outliers detract slightly from the overall trend, but there appears to be a clear downwards trend. This would indicate that there is indeed a correlation between the perceived naturalness and the objective error metrics.

Looking at the outliers above the graph, all of these are produced by either experiment S2 or experiment S3. The reason for this seems to be that the head position is being too far off from the head position in the ground truth, which

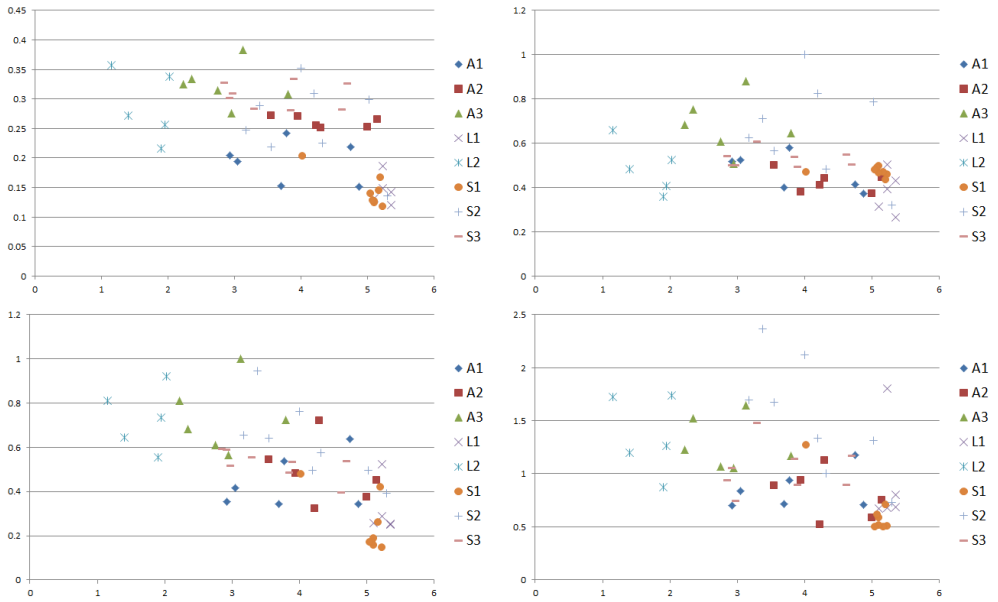

Figure 6: Normalized error metrics plotted against user case study scores. Top left is AAE; Top right is LAE; Bottom left is ALE; Bottom right is LLE.

is a result of the torso and neck being completely locked. As such, it appears people do not mind the rigidness of the torso as much as it is causing the head to deviate from the original motion. This brings us to the normalized version of the metrics, which we also previously defined in 5.3. By dividing the error by the distance of the marker to the joint it is directly rotated around, this could potentially solve some of the outliers. Figure 6 contains the normalized metrics plotted against the naturalness scores.

While our normalization did remove the outliers and seems to imply a correlation even more than before, we've come to a point where the graphs can't tell us everything. To find any correlation between the subjective user case study results and the objective distance metrics, we need to first choose our method of calculating correlation. The method chosen for this project is Pearson's product-moment correlation coefficient [Pea]. Pearson's correlation coefficient between two variables is defined as the covariance of the two variables divided by the product of their standard deviations. A strong correlation would be indicated by a r close to either 1 or -1, where r is given in equation (4).

$$r = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \bar{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i - \bar{Y})^2}} \tag{4}$$

If we calculate this value of r for all our four normalized metrics, we find that we obtain the highest correlation for our AAE and ALE metrics. The normalized Average Average Error metric has a correlation coefficient of -0.6295, while the normalized Average Largest Error shows a correlation coefficient of -0.6854.

7 Conclusions

At the start of this project, there were several goals set. The first goal was to use live motion capture data to produce an animated character in real-time. The challenges for this goal were to implement algorithms efficient enough to process all the marker data every frame. This processing involved the detecting of mislabeled markers, relabeling them where possible, smoothing out the noise and filling the gaps. An algorithm was designed to use the biomechanical constraints in the human body to assist in the gap filling process, by tethering markers in the case they were extrapolated too far.

For actually animating the character, the marker data had to be used to create a virtual skeleton which was scaled to fit the size and body type of the actor. The creation of the skeleton was done through configuration files called Joint Tree Files, which contained the skeletal structures with all the bones and joints. Each of these joints had markers defined onto them via offsets. The marker data could then be used to determine the target joint positions, which would then be solved for using the inverse kinematics algorithm Cyclic Coordinate Descent.

Through the successful implementation of performance animation, it was able to create a set of experiments to test the limits of the system. These experiments were created in order to find out which marker data is necessary for natural motion, and whether all degrees of freedom are equally necessary. This goal was to establish a sort of 'ideal' marker set, containing the fewest markers possible that would still produce decent looking animations.

To test whether a marker set was natural, a user case study was set up. In this user case study, 40 people participated in a survey where they rated the naturalness of motions on a scale of 1 to 7. In addition to the user case study, an objective distance metric was designed to analyse the motions by calculating the Euclidean distances between all similar points when comparing an experiment motion to its respective ground truth motion. The real goal was to try and show a correlation existed between the subjective naturalness and the objective distance metric.

From the user case study results, it has appeared that without further constraints on the degrees of freedom of the knee joints, it is near impossible to solve the inverse kinematics for the legs in a way that people find natural. The loss of the knee marker, and in lesser degree the elbow marker, indicated a serious drop in naturalness. Freedom in the feet was barely noticeable while ground contact is not necessary, which would indicate that the entire foot joint is essentially superfluous.

Other areas of the body that were not very impactful on the overall perceived naturalness were the wrists, head, and the pelvis. All three joints could have their rotational freedom severely restricted by reducing degrees of freedom, but this resulted in barely any loss of naturalness. The joint in the torso that seemed to be most important was the lower spine joint. The largest drop in naturalness of all the spine experiments was experienced when the lower spine joint became locked, and the torso was no longer able to bend.

Although marker occlusions do occur, they were only really impactful in one

out of the eight motions used for the user case study. The remarkable thing about the occlusions is that they occurred in one of the low-energy motions, which would suggest that there might have been a problem with the calibration of the motion capture system itself. Judging from the rest of the results, it would suggest that we could potentially reduce the amount of markers used from 38 to a low 17. This set would contain 3 hip markers, 2 torso markers, 2 head markers, 1 marker on both wrists, 1 marker on both elbows, 1 marker on both shoulders, 1 marker on both ankles, and finally 1 marker on both knees.

Finally, by plotting the normalized distance metrics against the naturalness scores from the user case study, we were able to find that the distance metric normalized Average Largest Error showed a strong negative correlation. By calculating Pearson's product-moment correlation coefficient, we were able to find a negative correlation of -0.6854. This suggests that there is indeed a correlation between the subjective nature of naturalness and objective error metrics such as the ones described in this paper.

8 Future Work

As the scope of this thesis is limited, there are several topics that are related but were not fully explored. These topics can be seen as an extension or improvement of the methods and the framework that has been described in this thesis.

The first change could be done at the very start of the pipeline. While Vicon Nexus does a decent job, there are occasionally some glitches where marker mislabeling and limb switching occurs. To have a bit more control over the marker labeling and tracking, the labeling part of the pipeline could be moved to inside this framework, leaving Vicon Nexus purely to read and send the marker data from the cameras. Alternatively, tweaks could be made to the marker positions and the skeleton Vicon Nexus uses.

Other areas of improvement are the prediction and interpolation methods described in chapter 3. Provided the marker labeling and tracking is still handled by Vicon, more advanced functions could be used instead. The function used in this thesis to predict and correct the marker position data currently uses a semilinear prediction vector (section 3.3, equation 2). Instead, higher order functions could be used to describe the marker trajectory as accurate as possible, thus providing a better prediction vector. Similarly, the process of correcting the marker position of a previously occluded marker currently uses a linear interpolation (section 3.4.2) to gradually interpolate from the extrapolated trajectory to the one of the raw marker data. Testing the actual impact the method of interpolating has on the perceived naturalness was not part of the experiments, but an alternative to the linear interpolation would be a sort of spline, as was briefly mentioned in section 3.4.2.

Perhaps the most influential and important part of the performance animation pipeline would be the inverse kinematics. While the Cyclic Coordinate Descent algorithm itself performs fine, there were no limitations set for any of the joints used in this thesis. All of the joints, unless changed in the setup of one

of the experiments, would have complete rotational freedom around all three axes. This is simply not physically possible for the vast majority of the joints in the human body. A strong improvement would involve using biomechanical constraints to impose limitations on each of the individual degrees of rotational freedom of the joints. By limiting the range joints can rotate around certain axes, this would mean any solution from the CCD algorithm would automatically be an anatomically correct one. One of the major factors that resulted in a low score for the experiments that removed all information from the elbow and knee joints was that the arms and legs would occasionally bend in completely impossible ways. Should these biomechanical constraints be added, the markers for both the knees and the elbows could potentially be removed as well without a significant decrease in naturalness. This would take the minimal markerset suggested in chapter 7 down to a low 13 markers.

Finally, the last area of future research would be the actual setup of the experiments themselves. The motions recorded for this thesis were all in a void, having no interaction with the environment. In the resulting animations, there character moved on a dark background, without even having a floor beneath. These extra interactions were not deemed necessary as they did not supply significant additional information to test the naturalness of the motion. Future experiments could contain more complex environments, where certain interactions such as correct floor contact become more important and importance-based metrics as discussed in chapter 2.2 might have to be incorporated. A second point of improvement would be setting up experiments that combine parameter changes across multiple body regions. While bodyparts were treated individually in this thesis in order to isolate the effects certain parameters have on the naturalness, further experiments might combine regions. This would allow to test full-body setups like the reduced marker set experiment mentioned in chapter 5.1.1 or the shoulder-torso marker exchange experiment mentioned in chapters 5.1.2 and 5.1.3.

References

[AB94] Ronald Azuma and Gary Bishop. Improving static and dynamic registration in an optical see-through hmd. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, SIGGRAPH '94, pages 197–204, New York, NY, USA, 1994. ACM.

[CH05] Jinxiang Chai and Jessica K. Hodgins. Performance animation from low-dimensional control signals. *ACM Transactions on Graphics*, 24:686–696, 2005.

[FSP92] Martin Friedmann, Thad Starner, and Alex Pentland. Synchronization in virtual realities. *Presence: Teleoper. Virtual Environ.*, 1:139–144, January 1992.

[Gle97] Michael Gleicher. Motion editing with spacetime constraints. In *Proceedings of the 1997 symposium on Interactive 3D graphics*, I3D '97, pages 139–ff., New York, NY, USA, 1997. ACM.

[Gle98] Michael Gleicher. Retargetting motion to new characters. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '98, pages 33–42, New York, NY, USA, 1998. ACM.

[KOF04] Adam G. Kirk, James F. O'Brien, and David A. Forsyth. Skeletal parameter estimation from optical motion capture data. In *ACM SIGGRAPH 2004, Technical Sketch*, pages 782–788, August 2004.

[LC10] Hui Lou and Jinxiang Chai. Example-based human motion denoising. *IEEE Transactions on Visualization and Computer Graphics*, 16:870–879, 2010.

[LM06] Guodong Liu and Leonard McMillan. Estimation of missing markers in human motion capture. *Vis. Comput.*, 22:721–728, September 2006.

[LS99] Jehee Lee and Sung Yong Shin. A hierarchical approach to interactive motion editing for human-like figures. pages 39–48, 1999.

[LWC+11] Huajun Liu, Xiaolin Wei, Jinxiang Chai, Inwoo Ha, and Taehyun Rhee. Realtime human motion control with a small number of inertial sensors. In *Symposium on Interactive 3D Graphics and Games*, I3D '11, pages 133–140, New York, NY, USA, 2011. ACM.

[Pea] Pearson product-moment correlation coefficient.

[PLKF09] Tommaso Piazza, Johan Lundstrm, Andreas Kunz, and Morten Fjeld. Predicting missing markers in real-time optical motion capture. In Nadia Magnenat-Thalmann, editor, *Modelling the Physiological Human*, volume 5903 of *Lecture Notes in Computer Science*, pages 125–136. Springer Berlin / Heidelberg, 2009. 10.1007/978-3-642-10470-1_11.

[PW99] Zoran Popović and Andrew Witkin. Physically based motion transformation. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '99, pages 11–20, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co.

[Ron10] Amos Ron. Lecture 13: Cubic hermite spline interpolation II, 2010.

[Sch11] R.J. Schuddeboom. Automatic processing of motion capture data, 2011.

[SLSG01] Hyun Joon Shin, Jehee Lee, Sung Yong Shin, and Michael Gleicher. Computer puppetry: An importance-based approach. *ACM Trans. Graph.*, 20:67–94, April 2001.

[SPB+98] Marius-Călin Silaghi, Ralf Plänkers, Ronan Boulic, Pascal Fua, and Daniel Thalmann. Local and global skeleton fitting techniques for

optical motion capture. In *In Workshop on Modelling and Motion Capture Techniques for Virtual Environments*, pages 26–40. Springer, 1998.

[Stu98] David J. Sturman. Computer puppetry. *IEEE Comput. Graph. Appl.*, 18:38–45, January 1998.

[vBE09] B. J. H. van Basten and A. Egges. Evaluating distance metrics for animation blending. In *Proceedings of the 4th International Conference on Foundations of Digital Games*, FDG '09, pages 199–206, New York, NY, USA, 2009. ACM.

[Vic] Vicon. Vicon DataStreamSDK.

[WB06] Greg Welch and Gary Bishop. An introduction to the kalman filter. Technical report, Chapel Hill, NC, USA, 2006.

[Wel93] Chris Welman. Inverse kinematics and geometric constraints for articulated figure manipulation. Master's thesis, Simon Fraser University, 1993.

Appendix 1: Experiment Selection Criteria Errors

Motion	Experiment	AAE	ALE
Dancing	S1	39.3109	50.1541
	S2	109.323	177.735
	S3	163.327	251.763
	A1	50.3938	147.627
	A2	57.8904	98.3135
	A3	70.8609	148.831
	L1	55.3436	97.459
	L2	127.25	322.982
Grabbing	S1	46.7346	81.7136
	S2	102.713	325.109
	S3	163.312	345.521
	A1	34.5792	80.864
	A2	59.8843	75.3155
	A3	63.175	133.49
Jumping	S1	40.5361	51.6402
	S2	78.5783	229.125
	S3	146.599	280.091
	L1	52.9685	91.4933
	L2	102.992	234.723
Kneeling	S1	41.5357	61.4234
	S2	88.9344	227.012
	S3	144.882	300.854
	L1	66.0841	167.462
	L2	128.23	283.702

The errors listed in the table are expressed in mm. Table continues on next page.

Motion	Experiment	AAE	ALE
Running	S1	56.1405	149.497
	S2	78.5361	185.954
	S3	124.228	212.139
	A1	52.9602	126.468
	A2	57.2055	193.227
	A3	84.7517	247.04
	L1	44.7885	86.8249
	L2	96.9566	265.959
Stretching	S1	42.588	57.6404
	S2	119.787	259.338
	S3	165.963	295.916
	A1	47.7081	88.2644
	A2	62.5272	90.2371
	A3	75.3477	152.261
Walking	S1	60.1408	125.396
	S2	48.1309	130.128
	S3	116.677	207.05
	A1	35.6471	84.1344
	A2	64.4348	136.682
	A3	71.8043	180.951
	L1	47.7541	86.1377
	L2	78.8707	199.673
Waving	S1	44.119	53.7602
	S2	105.207	166.808
	S3	130.075	182.576
	A1	46.4662	109.176
	A2	70.7077	133.485
	A3	81.5791	221.848

The errors listed in the table are expressed in mm.