Character Rigs for Motion Exaggeration

Ting Ting Wu

August, 2006

School of Computer Science
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA

Thesis Committee

Prof. Nancy Pollard, Chair
Prof. Jessica Hodgins

Submitted in partial fulfillment of the requirements
for the Degree of Master of Science

Abstract

Currently, the most common ways to obtain compelling 3D motion is using motion capture technology or employing the skills of an animator to hand keyframe it. Motion capture is fast, easy, and yields very realistic human motion. However it restricts its results to the realm of the realistic, unable to create cartoony motions that are more expressive. Hand keyframed motions, though, are expressive and appealing, but takes much longer to obtain and requires the skill of a trained animator. Hence, it is desirable to be able to add expressiveness to the large amount of motion capture data that is availble to us. In this thesis, we outline a method to add keyframed exaggerations to motion capture data with the use of character rigs. Along the way we introduce a motion transfer algorithm that turns joint angle motion data into a more animator friendly form, which allows editting operations to be performed more easily. We also investigate whether warping and blending algorithms yield better results in the rig control representation or the rig joint angle representation, and we perform perception studies to evaluate our results.

Acknowledgements

I would like to express my deepest thanks to my advisor, Dr. Nancy Pollard, for her consistent support, encouragement, patience, and guidance to help me direct my all unfocused thoughts into research that I was really interested in.

I would also like to thank Dr. Jessica Hodgins for being on my committee, and the advice and help she has given me. I also owe thanks to Leo Covarrubias for the lovely animations, Paul Reitsma for relaying everything he learned through trial and error, saving me countless hours of headache, Andrew Cove for helping me pass all my classes, and Christopher Twigg for taking care of everything else that went wrong.

Thank you to Mom and Dad for all the love, support, and wonderful home-cooked food when I'm home. And last but not least, to God, whose love humbles and strengthens me all at once.

Contents

List of Figures

Chapter 1

Introduction

Currently, the two most common ways to obtain compelling 3D motions are through motion capture or hand keyframing. There are advantages and disadvantages to both methods.

Realistic human motion is easy to obtain using motion capture systems like Vicon (http://www.vicon.com). The process is fast, and makes it easy to yield realistic motions. The disadvantage, however, is that motion capture does restrict us to the realm of realistic motion. In his 2002 paper, Bregler and his colleagues defined the range of motion style as ranging from "robotic" to "expressive", and marked "realistic" as much closer to "robotic" [8]. This leaves the large range of expressive motions as an area reachable only by skilled animators.

This brings us to hand keyframing, which can yield expressive and very appealing results. However, we would only get results as good as our animator. Hand keyframed motions also take much more time to produce even with the most gifted animators.

To highlight the difference between the results that the two methods can produce, we provide the following example. Suppose we want to duplicate Mickey Mouse's cheerful and bouncy walk in 3D (Figure 1.1).

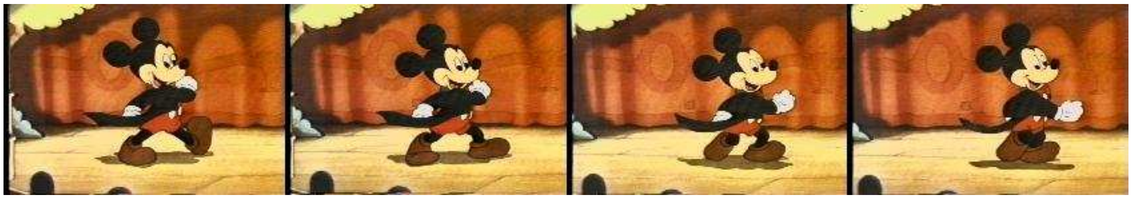


Figure 1.1: Mickey Mouse walk example.

If we tell a trained actor to mimic this walk and motion capture his movement, we get a result as in Figure 1.2. This result does capture the cheeriness of the walk. All result movies are available online at

`http://www.cs.cmu.edu/~ttwu/thesisMovies/`

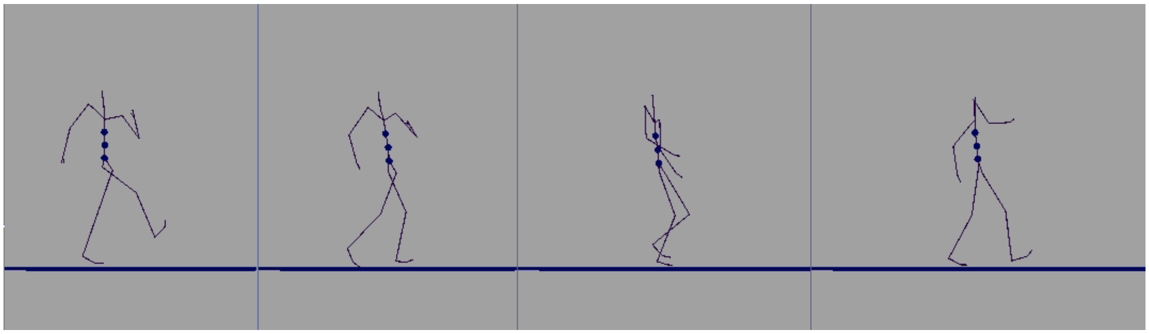


Figure 1.2: Motion capture result of Mickey Mouse walk.

However, if we tell an animator to reproduce this walk, we would get a result like the following in Figure 1.3. There is a significant amount of stretching and squashing in the legs as well as the torso. The trunk of the body also curves and leans back quite a lot. The character also drives its body up and brings itself down very forcefully, making the walk appear very energetic. These are some of the characteristics that are not possible to duplicate in a motion capture walk, and the characteristics that are possible to duplicate are not always safe to perform.

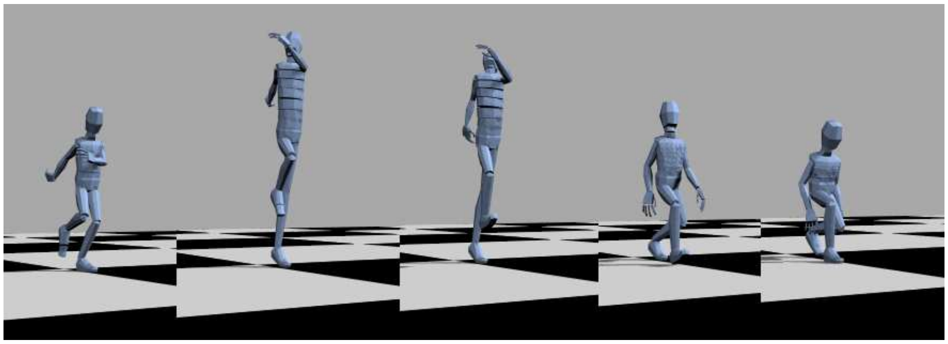


Figure 1.3: Keyframed result of Mickey Mouse walk.

Ideally, we would like a method that generates motions with the expressive appeal of hand keyframed motion, but has the convenient availability of motion capture data.

We chose to approach this problem by blending hand-keyframed exaggerations or “cartoonyness” into motion capture data. This approach has the potential to not only greatly expand a library’s range of motion style, but also vary motion mood because we can choose to exaggerate a motion in an angry way or a happy way.

In this thesis, we are interested in comparing the use of joint angles with the use of character rigs for this task of blending exaggerations into motion capture data. Of these two different ways to work with motion, the former method is preferred by those working on the technical side of the computer animation, while the latter is used by their artistic counterparts.

1.1 Joint Angle Representation

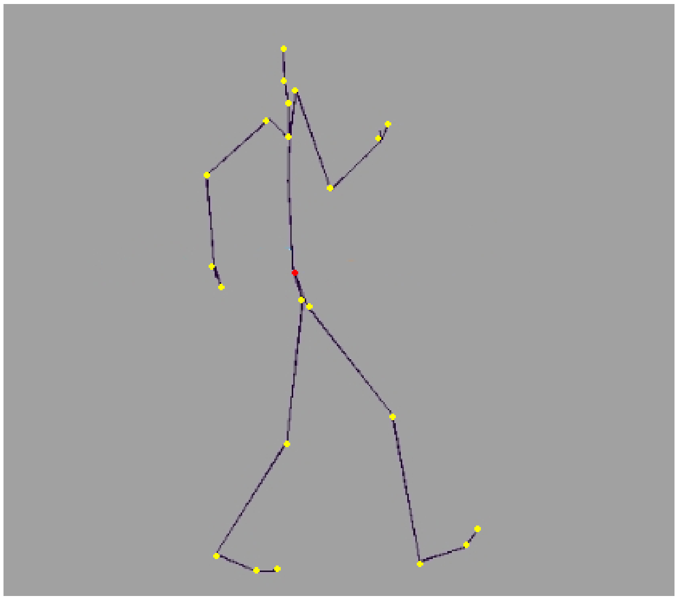


Figure 1.4: Example skeleton for joint angle system. Red marks root node. Yellow marks other joints.

The animation research community is accustomed to working with motion represented by joint angles. The joint angle system, or skeleton, defines joints in a hierarchical manner. An example of a skeleton is in Figure 1.4. The location and orientation of each joint is specified with respect to its parent joint. The location and orientation of a root in the skeleton is defined with respect to the global coordinate system.

The advantage of working in this space is that it is simple and quantifies motion neatly. A single joint is a building block for the skeleton, hence any arbitrary skeleton can be created and its motion defined in a similar fashion as long as the structure of the skeleton can be organized in the form of a directed graph. In addition, motion capture data is already encoded in this space, in the ASF AMC format. A disadvantage, however, is that we run into the inherent problems of working with Euler angles, including gimbal lock and the multiple sets of Euler angles that represent the same orientation [1]. Some of these problems can be fixed by using quaternions, but because euler angles are needed for a representation such as Maya, we have to move into the quaternions space for blending and editing operations, then move back to euler angles. This move can cause difficulties for keeping smooth euler angle curves near gimbal lock. If euler angle curves are not smooth, motion blur performs incorrectly, as does any interpolation that may be used to create intermediate keyframes. Also, the motion can no longer be represented easily with sparse keyframes.

1.2 Character Rigs

Character rigs, on the other hand, have been heavily used by animators as a more convenient way to control character skeletons. A character rig is a combination of a skeleton, a set of controls designed to manipulate that skeleton, and, usually, also a character mesh. Animators manually design rigs for characters, and the types of handles and the number of handles may be influenced by the skeleton configuration. The three components are designed to work together such that the controls anticipate the range of motions that particular character would make, and the mesh deforms properly when the skeleton moves within that range of motions. Because we are working only with the motion data we will only be concerned with the rig controls and the rig skeleton. Rig controls are also treated as blackboxes, where the joint angle calculations are unimportant as long as we can obtain the desired pose.

Rigs can be created in software such as Maya (http://www.autodesk.com/maya) or MotionBuilder (http://www.autodesk.com/motionbuilder). There are also a number of free rigs available for download online.

The rig handles combine many different types of control handles. While there are no set rules about what controls make up any character rig, the most commonly used types are inverse kinematic handles, directional handles and rotational handles. IK handles are used to control the absolute position and orientation of a joint at the end of a chain of joints, like a foot or a hand, and they manipulate the other joints along the chain to be able to satisfy constraints on the end effector. Directional handles, usually used on knees, elbows and eye gaze, control the absolute orientation of a part of the character such that that part of the character will point towards wherever the directional handle is located in space. Rotational handles control the rotation of parts of the body much like joint angles. An animator can choose to create any kind of control, so this makes it hard to have a universal format to store motion in terms of rig controls.

Figure 1.5 shows the rig we worked with and we will refer to this rig specifically for subsequent experiments.

Here, we will introduce a few of the key controls to give an idea of the variety of rig handles. There are IK handles on the feet (1 on Figure 1.5), which set the position and rotation of each foot, altering hip, knee, and ankle joints as necessary. The control on the back (2) is a more complicated one that combines the use of IK with a nurbs curve and an equation that takes the nurb radius of curvature as a parameter. It manipulates all the back joints in a natural looking way by allocating the bend among the joints, approximating the

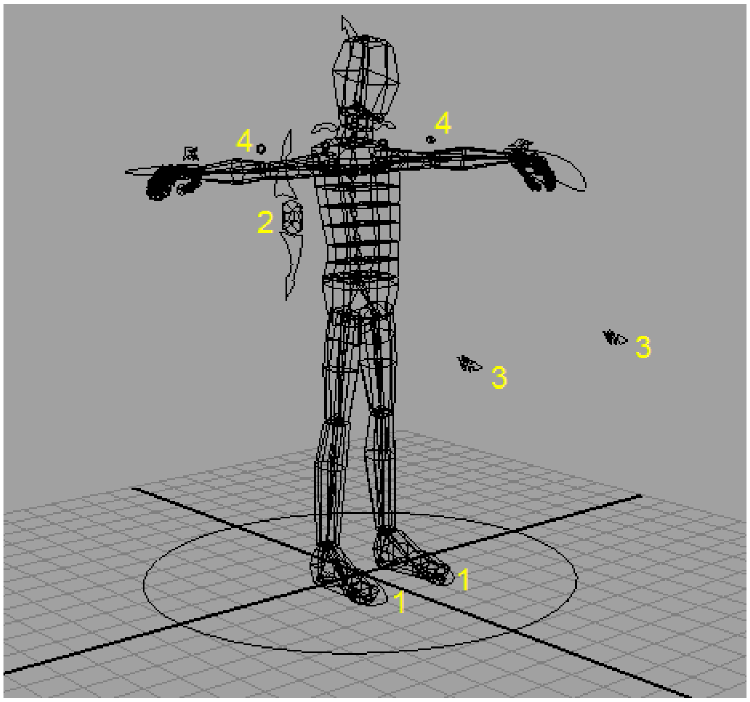


Figure 1.5: (1) IK controls on feet, (2) control for back, (3) directional controls for knees, (4) rotational controls for arms.

vertabrae. There are also directional controls to manipulate where the knees will point (3). This type of control is many times used to control the eye gaze as well. The knee will point into the direction of where the control handle is translated to, hence all locations on the ray created from the knee to the control have the same affect. There are also rotation controls (4) on the arms, which are very much like joint angle controls. There are also extra attributes added to the foot controller to allow manipulation of the curl of the foot and stretching of the legs. These extra attributes are not always intuitive to work with and tweaking may not contribute too much to overall look of the pose. For a full list of the rig controls see Appendix A.

A big plus of working in this space is that most character rigs use IK handles, directly addressing the foot sliding problem. Rigs often simplify the control of the character by having controls that modify multiple joint angles simultaneously to obtain visually pleasing results. Rigs also make some interesting motion editing operations trivial. For example motion retargetting from one rig to another rig can be as simple as copying and scaling motion data directly. The task of changing the step size of a motion can be done by scaling the root translation by the translation of the feet IK handles. The operations become messy and complicated in joint angle space.

1.3 Comparing the Two Representations

We implement motion warping and motion blending in both of the representations of rig controls and rig joint angles. Below is an overview of our system.

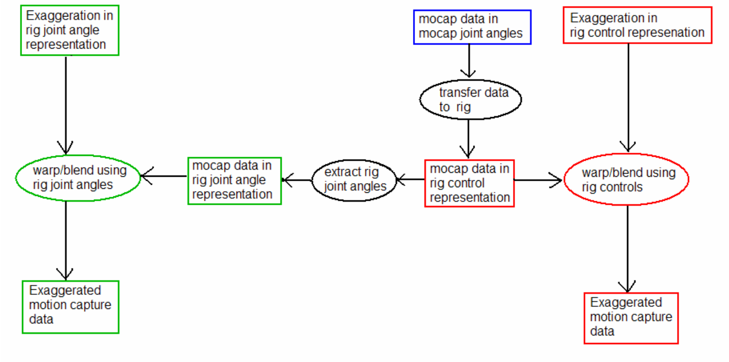


Figure 1.6: Overview of our system. Blue indicates data represented in motion capture joint angles. Red indicates data represented with rig controls. Green indicates data represented in rig joint angles.

We begin with motion capture data as input in its original representation of motion capture skeleton joint angles. In order to be able to blend in the exaggerations that are in the rig space, we transfer the motion capture data into the rig control space with our motion transfer method. At this point, we can split off and use the motion capture data in the rig control form (red part of the diagram), or extract the joint angles of the skeleton and operate in the rig joint angle form (green part of the diagram).

We perform similar motion warping and blending operations in both representations, then compare the results in the end, and also evaluate them with perceptual studies. After analyzing our results, we further discuss the strengths and weaknesses of working with both representations, and what this implies for the potential of using character rigs in other motion algorithms.

Chapter 2

Related Works

There are a number of different areas that are relevant to understanding this problem. First of all, it is important to have a grasp on the notion of cartoonyness, and for this we turn to references on traditional 2D animation. There has been research into quantifying cartoonyness and introducing it into existing motion data, much like we are trying to do. Works on motion style extraction and translation are also relevent because we can think of cartoonyness as a style of motion. Some of the existing problems with using joint angles is issues with foot sliding and also its awkwardness in representing motion. While we chose to address these problems with the use of rigs, previous research found different ways.

2.1 Exaggeration in Traditional Animation

Cartoonyness is a well established art in traditional 2D animation and there are a number of universally recognized resources on the subject. “The Illusion of Life: Disney Animation” [2] provides some history into developments by the studio over the years in bringing characters to life. Richard Williams’ “The Animator’s Survival Kit” [3] thoroughly explores issues of timing, squash and stretch, and even lists features that can be varied to add personality to a walk cycle. It was from Williams’ work that we found the different sets of rules for working with different classes of motions (walks, runs, sneaks). As a result, we decided to focus only on walking motions. We anticipate that we can work with other classes of motion capture data in the same way but would need a corresponding set of hand keyframed exaggerations. John Lasseter’s landmark paper [5] reviewed the classic animation principles for a new generation of animation in 3D.

2.2 Character Rigs

Character rigs are widely used by 3D animators and is a well developed area of 3D animation. Most instructional books for commercial animation software such as Maya have at least a chapter devoted to the setup of various controls to make a character easy to animate [4]. There is also software such as MotionBuilder (http://www.autodesk.com/motionbuilder) that have features designed specifically to create character rigs.

2.3 Algorithms to Add Exaggerations to Motion

There have been a few different approaches to obtain cartoony motions. Motions of simple objects can be enhanced by applied volume-preserving sinusoidal-based stretch and squash effects [6]. For fully articulated characters, though, the concept of adding style is not so clearly defined. Li et al. approached it by creating an interface in which a user can aid the system to stylize existing motion by drawing out the character in two stages [7]. The first stage modifies the underlying skeleton, and the second stage deforms the surface of the mesh. While this technique works well with simple characters, it would be very difficult and time-consuming for detailed characters, because features that are not specified by the artist can move in undesirable ways. Bregler et al. developed a method to extract motions from cartoons that involves affine transformations and interpolation between shapes at keyframes, enabling the transfer of motion from one 2D character to another 2D character, and even from 2D to 3D [8]. Most recently, Wang and his colleagues formulated the Cartoon Animation Filter, which applied the animation ideas of anticipation and follow-through to exaggerate joint angle signals over time [13]. The idea is simple and is enough to make motion capture data more interesting. However, this method provides only one dimension to make a motion more cartoony, hence it is unable to produce exaggerated walks of a certain mood. It often introduces wild foot sliding when applied to motion capture data.

2.4 Cartoonyness as Style

Cartoonyness can be thought of as a style of motion. Hsu et al. presented a method for translating style from one motion to another using their Iterative Motion Warping algorithm and a linear time-invariant model [9]. An interesting idea in this paper is that the style of a motion can be encoded as a scale and offset of the motion data and a timewarp.

The Verbs and Adverbs paper of Rose et al. described a method to create an autonomous character in 3D space that can move smoothly within movements available in a verb graph (type of movement) which are controlled by an adverbs parameter (the style of motion) [16]. This is comparable to our idea of modifying motion capture data by adding different exaggerations to modify the mood.

Liu et al. parameterized style in a very different way in their 2005 paper [17] in which they defined style as a result of a person's preference to use certain muscles. With the use of information from biomechanical literature, they created a model were muscle preference parameters covered a wide range of motion styles.

2.5 Different Ways to Represent Motion

The idea of having different controls for motion other than just joint angles is not new to animation research. Many motion algorithms involve dimensionality reduction since many degrees of freedom of the skeleton are highly correlated [10] [11]. However, no previous research on motion representation is really comparable to the idea of transfering joint angle data to a character rig space.

2.6 Inverse Kinematics

Most motion editing algorithms ignore the foot sliding problem until the end, when post processing is needed to fix the problem [9]. One of the common post-processing methods is Kovar et al.'s footskate cleanup algorithm [12]. However, with the use of IK handles on character rigs, we can directly control the location of the foot and cleaning up foot sliding artifacts is much simpler.

2.7 Motion Retargeting

During the course of our algorithm, we transfer motion data from the original motion capture skeleton to the character rig. There are some related works on motion retargeting, such as Gleicher's method to retarget motion to characters with the same skeleton structure but different limb lengths [14]. While this is useful background, this is quite different than when the final desired representation is a rig.

2.8 Our Contribution

We describe a way to add hand keyframed exaggerations to motion capture data. Since the animator creates the exaggerations on a character rig, we choose to transfer the motion capture data over to the rig space as well before we can blend the exaggerations. Hence, we also outline our algorithm for this motion transfer process.

We also compare working with the rig controls versus working with rig joint angles to stylize motion capture data, and study the strength and weaknesses of both methods. Based on findings from our research, we assess the potential usefulness of character rigs as an alternative way to represent motion capture data.

Chapter 3

Transfer Motion into Rig Control Space

In order to transfer movements from the motion capture skeleton to the rig, which has a different skeleton with a greater number of degrees of freedom, we first establish a correspondence between the two skeletons. We must also be able to assess just how well we have set the parameters of a rig handle at any point as we are trying to match to the pose. Once we have these components, we can iteratively move the rig controls to minimize the difference between the rig pose and the motion capture pose.

3.1 Matching Skeletons

Figure 3.1 shows the correspondence between joints of the motion capture skeleton and the rig skeleton. Details and joint information about the two skeletons are available in Appendix B.

This correspondence will be used to create a mapping between a degree of freedom from the rig skeleton to another degree of freedom in the mocap skeleton. We will call this mapping $rigSk2MocapSk$. The two skeletons are quite different, and in general we would expect some change to the motion capture data as a result of the transfer. The fidelity of the match depends on the particular rig and how we choose to match up the two skeletons.

3.2 Evaluating Match

For each rig handle R, we calculate a $dPose$ vector, which is a measure of how well we have placed R. For any change to a given rig handle R, it affects m_R degrees of freedom of the rig skeleton. This m_R can contain all degrees of freedom that are affected, or it can also be selected algorithmically or manually. $dPose$ for rig R at time frame t is calculated as follows. Let the joint information

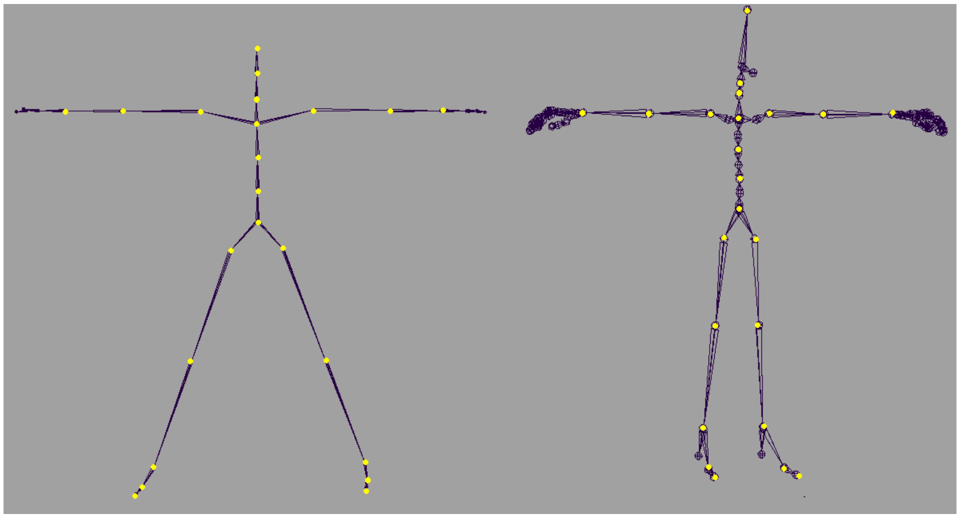


Figure 3.1: Motion capture skeleton (left), Rig skeleton (right). Joints used to match skeletons are marked in yellow.

for the motion capture skeleton be stored in *mocapData*, and similarily, let the joint information for the rig skeleton be stored in *rigSkData*, both of which can be indexed by time and the degree of freedom.

```
//for rig handle R at time t
for(0<i<m_R){
        rigIndex = i-th degree of freedom of rig skeleton affected by rig R

        //get corresponding DOF in mocap skeleton
        mocapIndex = rigSk2MocapSk[rigIndex];

        dPose[i]= (mocapData[t][mocapIndex] - rigData[t][rigIndex])^2;
}
```

It is important to mention that the degrees of freedom of the two skeletons which we compare are not local but global positions and orientations of the joints. Motion capture data is defined in terms of relative joint angles, so we must calculate the global position and orientation information of a particular joint by traversing through all its ancestors and accumulating the transformations at each level.

3.3 Algorithm for Matching a Pose

Rigs usually have hierarchical structure for controls, much like that of a typical skeleton, such that children control handles are affected by the transforms of the parent. Figure 3.2 shows the hierarchical structure of the rig we used.

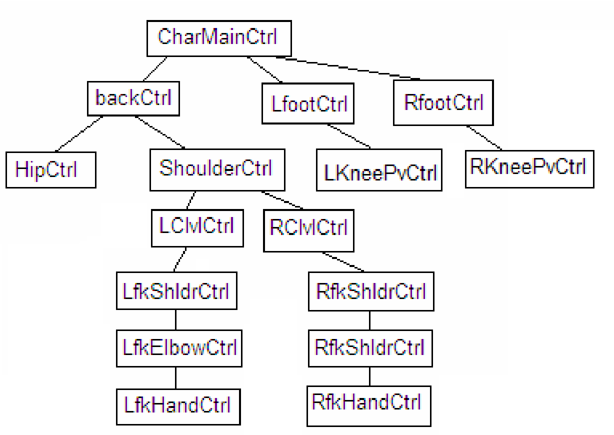


Figure 3.2: Hierarchical structure of rig handles.

We want to manipulate controls in a top down manner, such that we make big changes first, and gradually fine tune the pose. Hence we traverse through the rig controls in breadth-first order, adjust parent rig handles then the children rig handles.

3.3.1 Pose Matching Process

Below, we illustrate with psuedocode how the pose matching process works.

```
for(every frame){
        for(every rig handle, traversed in breadth-first order) {

                if(is a directional handle){
                        p = desired direction in target joint space;
                        T_p = target joint's transformation;
                        G_p = T_p*p;

                        move rig handle to G_p;
                }

                else{
                        for(set number of iterations){
                                dPose = calculate desired change in pose;

                                J = calculate Jacobian;
                                dRig = J^-1 * dPose;

                                move rig handle by a fraction of dRig;
```

```
                        }
                }
        }
}
```

The variety of different controls that can comprise a rig sometimes requires that they are distinguished between each other and some must be treated differently in the matching process.

3.3.2 IK and Rotational Controls

For IK, rotational and most other rig controls we can iteratively calculate a Jacobian rig matrix, where partial derivatives relate changes in the rig controls with changes in the pose, and drive the rig pose closer to the target motion capture pose. However, finding the Jacobian for a rig system is non-trivial, we will explain the procedure in the following chapter.

3.3.3 Directional Controls

Directional controls have three translational degrees of freedom, and its position in space dictates the orientation of the skeleton component that it controls. However, placing the controller on any point on the ray from the skeleton component to the controller would effectively be the same. This redundancy can be dangerous if we naively use the above iterative method to place this controller. The position values of the controller can blow up over the iterations as it moves further out along the ray which contains the desired position. There is a stable, faster and direct method to calculate where to place the directional control.

We look at the joint on the motion capture skeleton that we want to emulate with the directional control, and cast a ray in that direction with respect to the local coordinate frame. Since the local coordinate frame is centered at that joint, that ray can also be interpreted as a point in space with respect to the joint. Then it is easy to back out the global coordinate of that point and place the directional control there. An additional scaling term can be added to push that point further away before calculating the global coordinates, and this generally allows for smoother results and avoids popping artifacts in the motion.

3.4 Rig Jacobian Matrix

There are a large variety of controls that can make up a character rig, and nearly all the rigs handles, except for directional handles, can be incrementally matched up to the desired pose by constructing a Jacobian matrix.

For a rig handle R, there is a m_R x n_R Jacobian matrix, where m_R, is the number of degrees of freedom of the rig skeleton that are affected by changes to rig handle R, and n_R is the number of degrees of freedom of the rig handle which can be manipulated. We fill out the Jacobian Jas follows.

```
for(0<j<n_R){
        change rig R by d(rigDOF_j)

        for(0<i<m){
                find d(poseDOF_i) that resulted
                J[i][j]= d(poseDOF_i)/d(rigDOF_j);
        }

        undo d(rigDOF_j) change
}
```

The major limitations of this method is our limited ability to evaluate the difference between poses of two different characters. This evaluation is not always as simple as matching the absolute joint locations of skeletons. Factors that complicate this include the different geometry around each skeleton. For example, while the feet joints may match up well for two characters, one of the character's feet might be half through the floor.

Chapter 4

Motion Warping and Blending

4.1 Motion Warping

Once we have the motion capture data in rig representation, just like the keyframe data, we want to time warp the keyframed motion to the motion capture motion so that the exaggeration can be blended into the motion capture data. We computed and performed the warp in both the rig control space and the rig skeleton's joint angle space.

We originally implemented Hsu et al.'s Iterative Motion Warping algorithm [9], which alternates between a time warp step and a space warp step to shift and scale one input motion to better fit another motion. However, we found that while the method works well on the original motion capture data in the original motion capture skeleton, it does not work for either the rig control space data or the rig joint angle space data once the motion has been transferred to hte rig skeleton. However, when we removed the space warping part of their algorithm, and used only the time warping step, we were able to consistently get matches in the rig control space. This is a pleasant surprise since the space warping part is much slower due to a large linear system solve step.

From our experience with the exaggerated motions and the motion capture data, the motion capture walk is usually the longer sequence, because the keyframed motions we obtained run at 32 frames per walk cycle. This is due to animation timing rules. Walk cycles can be slower or faster, but 32 frames is a commonly chosen length. The time-warp setup described below assumes that the motion capture length is longer than the keyframed length. It this is not the case, we can simply repeat each frame of the motion capture data as necessary such that the motion capture length is longer by an integer multiple. We can perform the following time-warp as described, and simply down-sample the resulting final warped motion such that we get the original motion capture length.

Like the time warping step of Hsu et al. we do not allow any of the frames of the keyframed motion to be skipped. This is a constraint that we must be careful to enforce both when filling out the dynamic programming table and when we extract the warping information from the table.

First, we set up a dynamic programming table that is $keyframedLen + 1$ by $mocapLen + 1$ where an element at i, j is the optimal cost value of having matched the keyframed motion frames 0 to $i - 1$, to the motion capture data frames 0 to $j - 1$, where $0 < i \leq keyframedLen$ and $0 < j \leq mocapLen$. Once the chart is filled out, we can start from element $keyframedLen$, $mocapLen$ and traverse towards element 0,0 by picking the neighbor with the optimal cost. If we are currently at frame i,j then we only consider neighbors i,$j - 1$ and $i - 1$,$j - 1$. Picking element i,$j - 1$ corresponds to the choice of choosing to repeat the frame $i - 1$ of the keyframed data to match frame $j - 2$ of the motion capture data, and picking element $i - 1$, $j - 1$ corresponds to moving onto frame $i - 2$ of the keyframe motion to match with frame $j - 2$ of the motion capture data. Since we do not allow any of the frames of the keyframed data to be skipped, and since that corresponds to picking element $i - 1$,j, we do not consider that neighbor. Given the variable r, which is the limit on the number of repetitions of a frame of the keyframed data, we force the algorithm to move on to neighbor $i - 1$,$j - 1$ once this limit has been reached.

After we obtain the warp array, which tells us how many times we have to repeat each frame of the keyframed motion, we interpolate values for the repeated frames.

We use this same algorithm to time warp in the rig control space as well as the rig joint angle space. The differences between the algorithms using these two different representations are that different input data must be matched (rig control parameters versus joint angles) and different weights are used across different degrees of freedom in the two methods.

4.1.1 Setting Weights

Setting weights for the input data we want to match lets the algorithm ignore the degrees of freedom that we do not consider important and place more emphasis on matching the degrees of freedom that we think are significant.

When weighting the rig controls, we ignore directional handles by assigning weight 0 to those degrees of freedom. We also assign weight 0 to the degrees of freedom that remain constant over the keyframed motion because it does not contribute to the change of motion over time. The remaining degrees of freedom are divided into rotational degrees of freedom and translational degrees of freedom and weighed by their range of motion with respect to each individual group. We find the maximum range of motion of the rotational degrees of freedom, and also the maximum range of motion of the translational degrees of freedom, and set the weights of each degree of freedom as its range of motion normalized by the respective maximum range of motion.

A similar algorithm failed on the rig joint angles. This is again because of the problem of using euler angles to represent the motion. In order to keep euler angle trajectories as smooth and continuous as possible we post process the data, however this can cause some rotational angles to continually increase past 360 degrees. This in turn gives the algorithm the illusion that this particular has the largest range of motion when in reality it may only have the largest range of values. Even more inconvenient is that each degree of freedom do not consistently give us good or bad weights, but vary depending on the exaggeration.

4.2 Blending in Exaggerations

After warping the keyframed data to the same time length as the motion capture data, we can blend the values for every degree of freedom by the chosen ratio. We chose to blend in the exaggerations at 25%, 50%, 75%, and 125% to cover a full range of possible exaggeration ratios.

Chapter 5

Implementation Suggestions

5.1 End Effector versus Joint Angles

It is a common problem of motion retargeting that if the target skeleton does not match the original skeleton well, one must trade off matching joint and end-effector positions with matching joint angles. [15] One example of this can be found when trying to match the lowerbody part of rig with motion capture data. It is not unlikely that in the process of matching the IK handles on the feet and matching the location of the hips, that the bend of the knees would look very awkward. We handle this by matching up the hips and the feet IK handles over all frames. When this step is done, we find the time at which there is the maximum amount of bend in the knee in the motion capture data, and we move the hips of the rig until we get a good match of the knee bend angles. Finally, we apply the same displacement to the hips in the rest of the frames. We found that this method worked quite well for us, but the solution to deal with this problem across different rigs may vary.

5.2 Technical Difficulties with the Rig

We would like to mention that we also encountered some problems in dealing with the arms of this particular rig. Upper arm controls affected elbow angles, and elbow controls also affected upper arm joint angles. Coupling the changes in these controls made it very difficult to get a good match to the motion capture data. We simply chose to not work with the elbow controls.

5.3 Interpolating in Rig Control Space

There is also an additional caveat for interpolating values in the rig control space. Many times animators design motion not anticipating that the time between two consecutive frames would be stretched out. This becomes a problem with degrees of freedom that make a sudden jump in value from one frame to the next while the pose does not change much visually. This is a sign that we were never meant to interpolate the values in between the two frames. We handle this by thresholding the change in value from one frame to the next. If the change is greater than 50% of the value range of that particular degree of freedom, then we simply repeat the frame, otherwise, we can interpolate. We have only encountered this problem in working with rig controls.

Chapter 6

Results

6.1 Animation Results

Below we have some motion results to compare operations using the rig control representation and the rig joint angles representation. To see videos of all results, go to

`http://www.cs.cmu.edu/~ttwu/thesisMovies/`

6.1.1 Motion Transfer

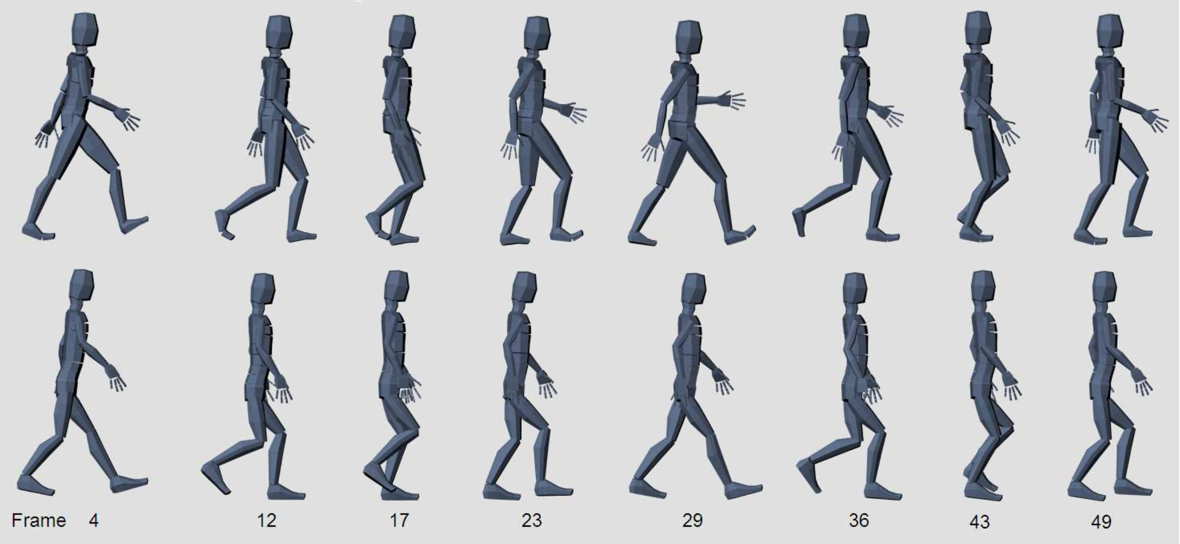


Figure 6.1: (Top row) Motion capture data in original motion capture skeleton. (Bottom row) Motion capture data transfered rig representation.

The resulting transfered motion is very similar to the original motion. We notice that most of the differences between the rig representation results of our algorithm and the original motion capture data is a result of the disparities between the two skeletons. The difference between frames 4 and 29 of Figure 6.1, in particular, can be attributed to differences between lengths of the leg segments. The transferred motion can also appear a bit stiff in certain frames, which is due to the torso having been elongated when the hips were shifted down to obtain a better match for the knee bend angles. We also chose to not change the elbow controls, which accounts for differences observed in the arm swings.

6.1.2 Motion Warping

We found that when time-warping was performed in the rig control space the algorithm is consistantly able to give us a reasonable result. This is not always the case for working in the rig joint angle space, and this is due to the nature of the rig controls and the failings of Euler angles.

Each rig control is a black box that sets joint angles with no consideration for surrounding keyframes, which can result in joint angle trajectories that jump dramatically from frame to frame even though the change in the orientation of the joint is quite small. There are post-processing steps that can correct this and create smoother trajectories by substituting different joint angle values to obtain the same orientation. However, this does not guarantee that we can get the values which would be useful for matching to the other motion. When this post-processing step fails to address both the temporal continuity of the trajectories and similarity to the other motion data, important degrees of freedom must be handpicked by comparing motion curves manually.

Using Rig Controls

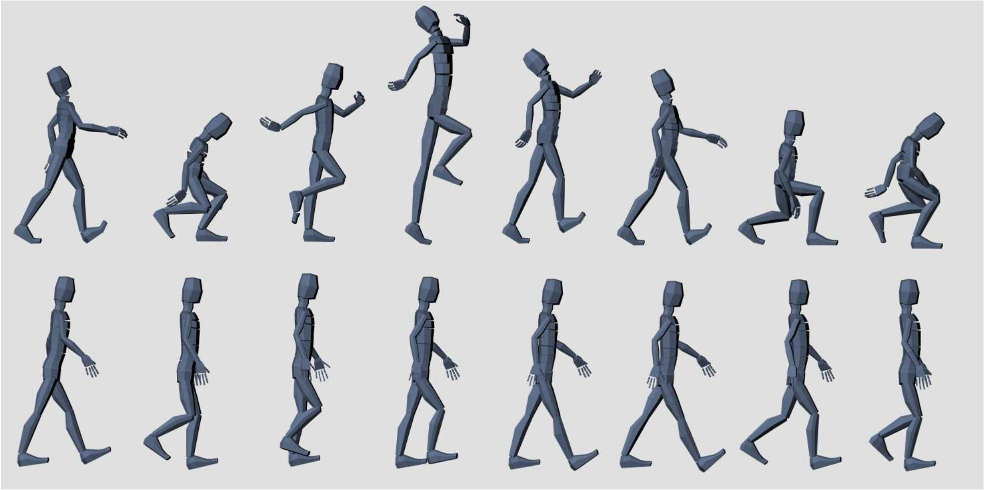


Figure 6.2: (Top row) Happy walk warped using rig controls. (Bottom row) Motion capture walk.

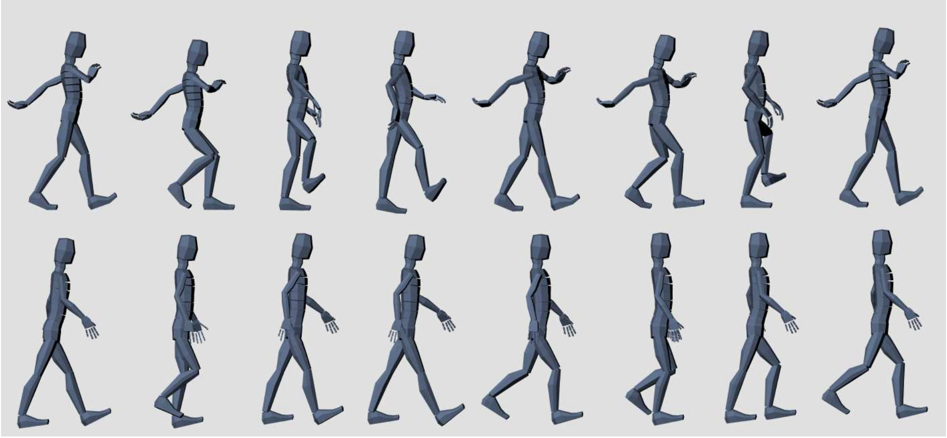


Figure 6.3: (Top row) Goofy walk warped using rig controls. (Bottom row) Motion capture walk.

Using Rig Skeleton

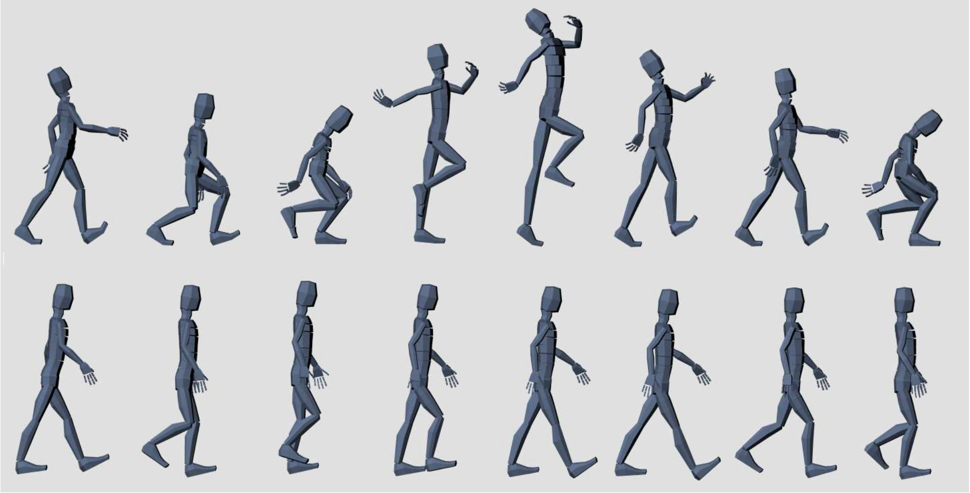


Figure 6.4: (Top row) Happy walk warped using rig joint angles. (Bottom row) Motion capture walk.

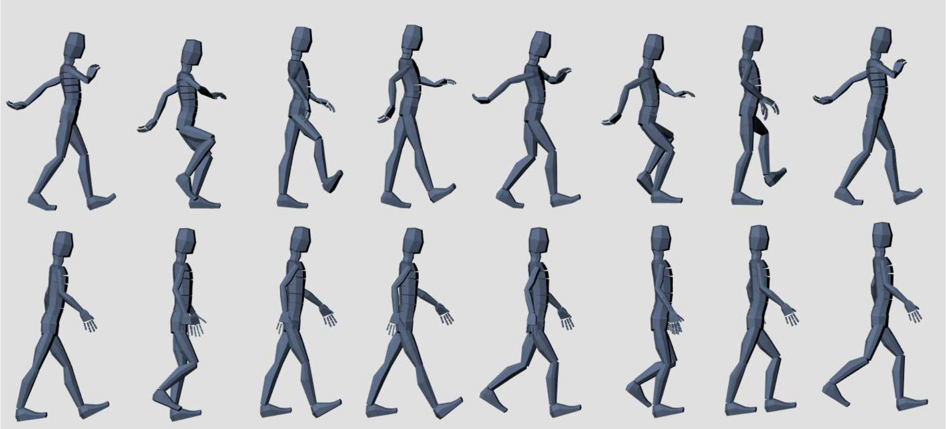


Figure 6.5: (Top row) Goofy walk warped using rig joint angles. (Bottom row) Motion capture walk.

Figures 6.6 and 6.7 show a few examples of how well the time warping aligned the two motions across the different degrees of freedom of the rig controls. Figure 6.6 is a good match. The match in Figure 6.7 seems off phase but the periodicity still matches.

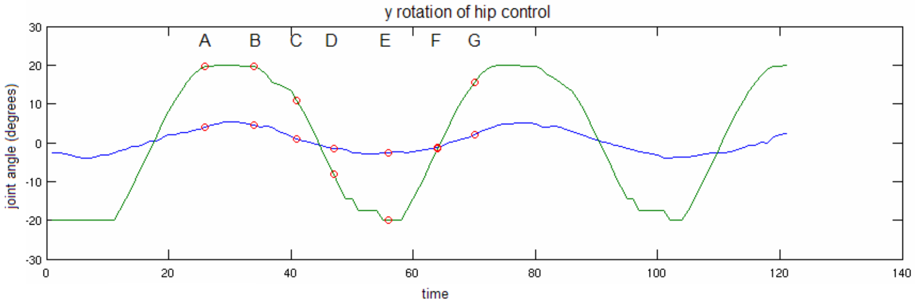


Figure 6.6: green line is warped keyframe data, blue line is motion capture data for happy walk exaggeration.

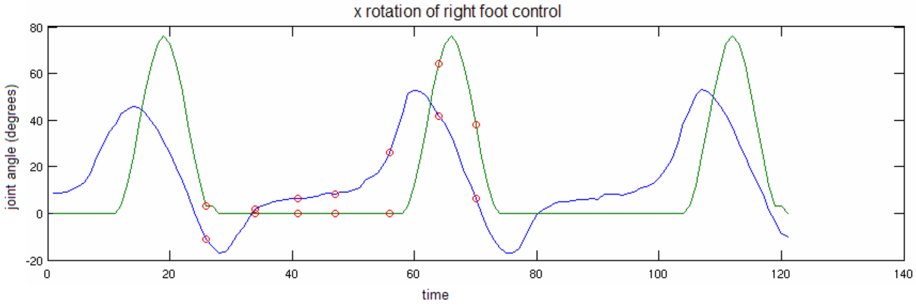


Figure 6.7: green line is warped keyframe data, blue line is motion capture data for happy walk exaggeration.

Figures 6.8 shows an example of how well the time warping aligned the two motions across the different degrees of freedom of rig joint angles. This particular degree of freedom is a good match. However, working with euler angles sometimes causes problems with some degrees of freedom as in Figures 6.9, 6.10, and 6.11, where the two motions being compared are similar, yet the euler angle trajectories over time look nothing alike. These degrees of freedom need to be given a very low weight in the warping algorithm. We had to specify the weight values manually for rig joint angles whenever we wanted to warp two walking motions together, giving high weights to the degrees of freedom where the trajectories over time look similar between the two walks, and giving zero weight to the degrees of freedom that will not help. In this way, warping in joint angle space is not nearly as robust as warping in rig control space, where the weights can be defined based on the range of motion of that degree of freedom.

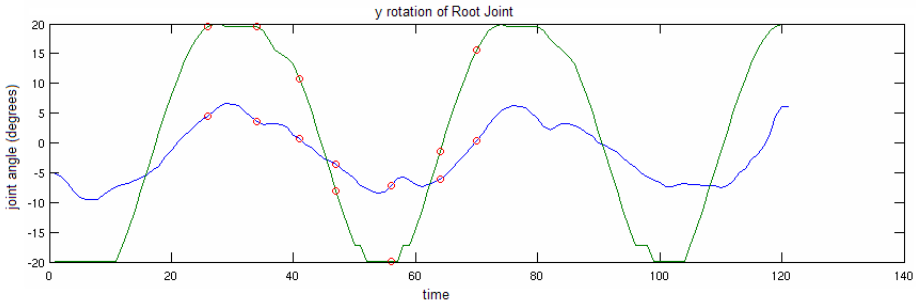


Figure 6.8: green line is warped keyframe data, blue line is motion capture data for happy walk exaggeration.

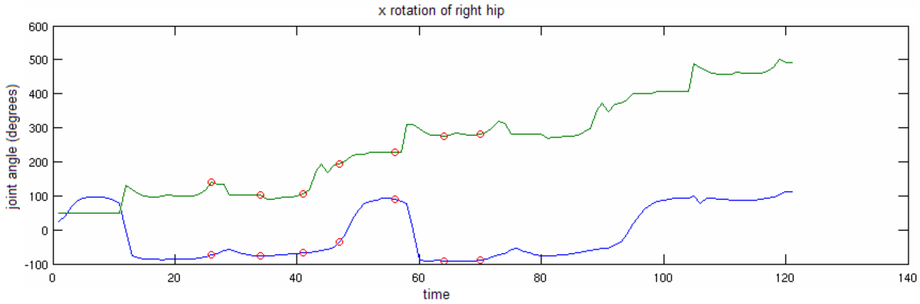


Figure 6.9: green line is warped keyframe data, blue line is motion capture data for happy walk exaggeration.

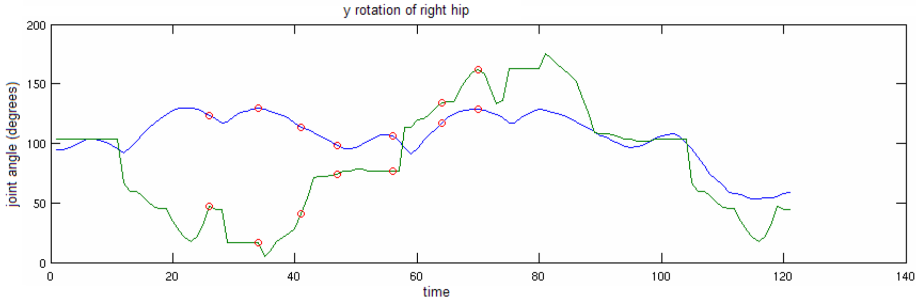


Figure 6.10: green line is warped keyframe data, blue line is motion capture data for happy walk exaggeration.

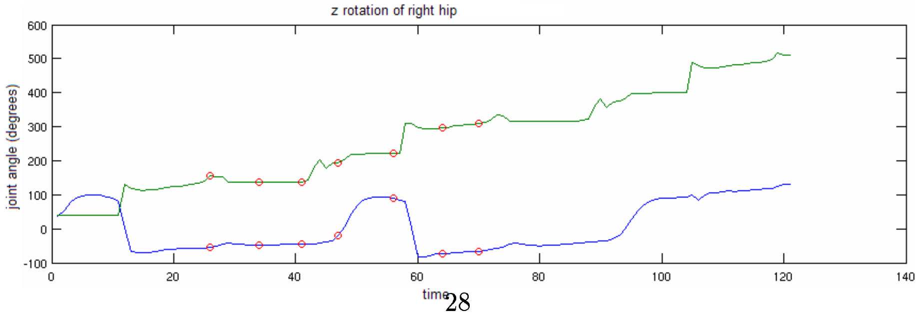


Figure 6.11: green line is warped keyframe data, blue line is motion capture data for happy walk exaggeration.

6.1.3 Comparing Motion Blending Results

Happy Exaggerations

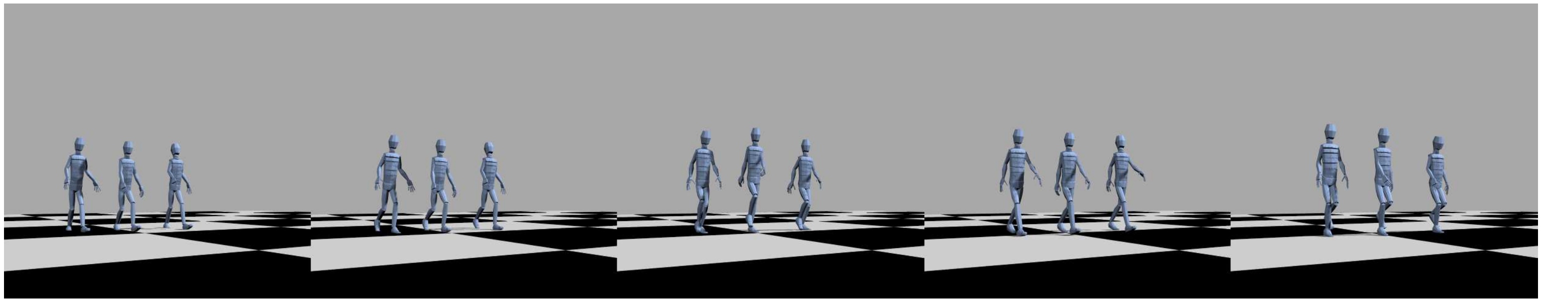


Figure 6.12: (1) Motion capture walk (2) 25% happy walk blended into motion capture walk using rig controls (3) 25% happy walk blended into motion capture walk using rig joint angles

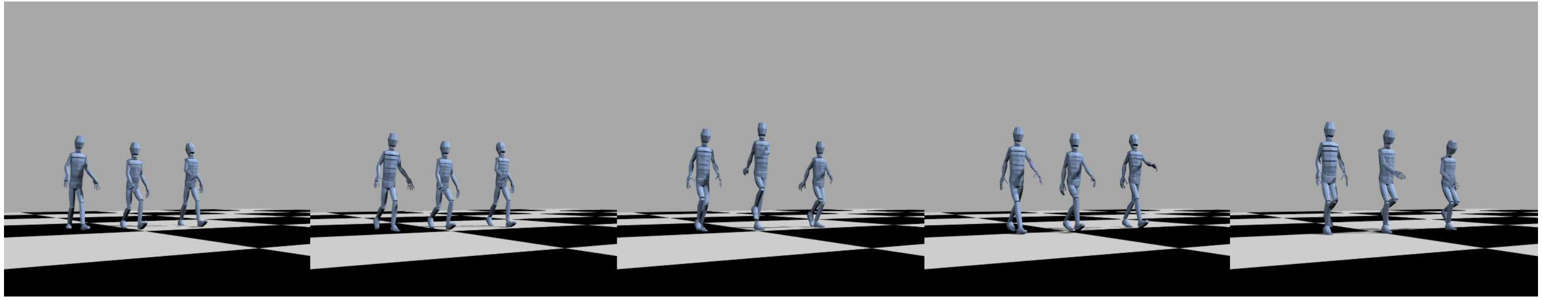


Figure 6.13: (1) Motion capture walk (2) 50% happy walk blended into motion capture walk using rig controls (3) 50% happy walk blended into motion capture walk using rig joint angles

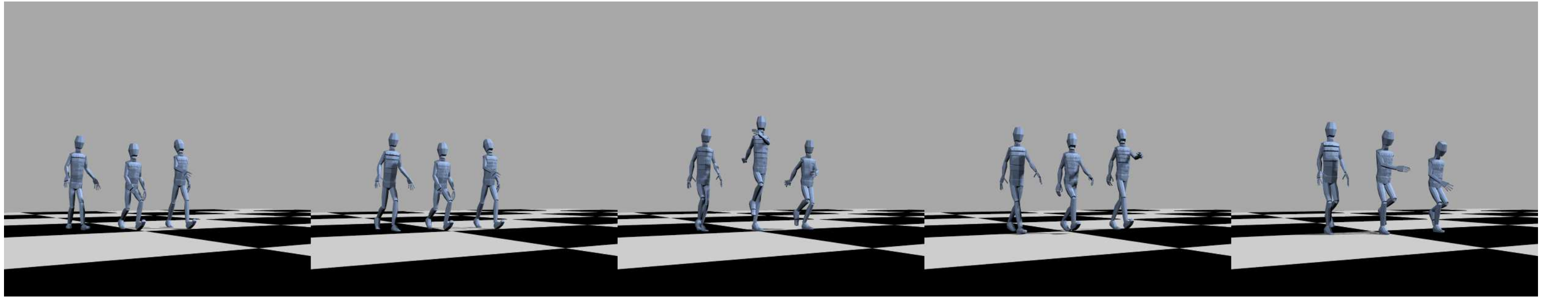


Figure 6.14: (1) Motion capture walk (2) 75% happy walk blended into motion capture walk using rig controls (3) 75% happy walk blended into motion capture walk using rig joint angles

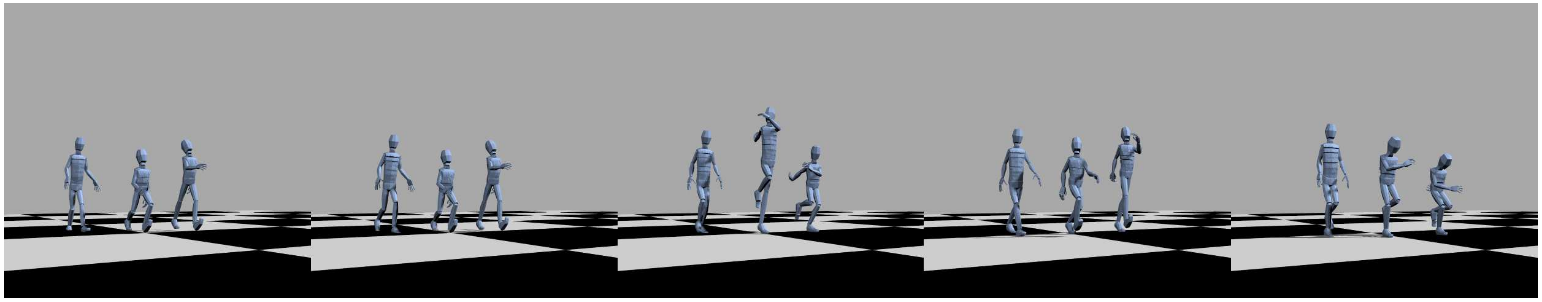


Figure 6.15: (1) Motion capture walk (2) 125% happy walk blended into motion capture walk using rig controls (3) 125% happy walk blended into motion capture walk using rig joint angles

Goofy Exaggerations

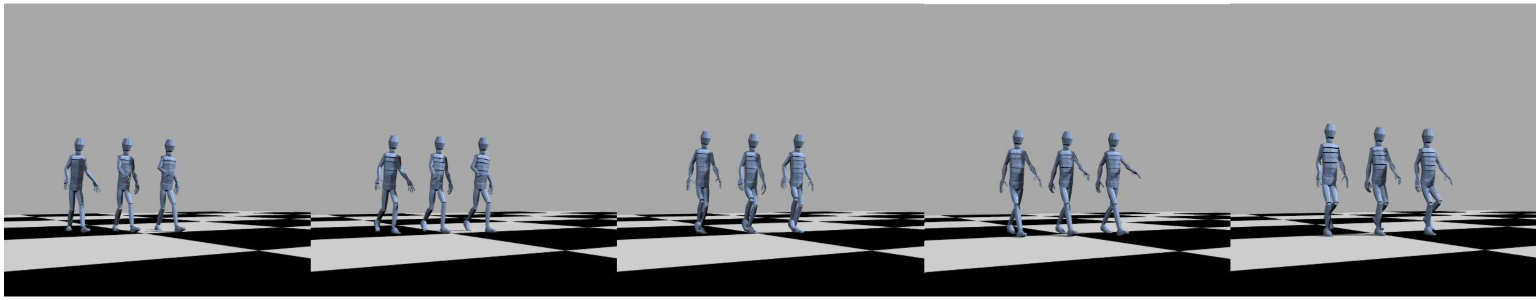


Figure 6.16: (1) Motion capture walk (2) 25% goofy walk blended into motion capture walk using rig controls (3) 25% goofy walk blended into motion capture walk using rig joint angles

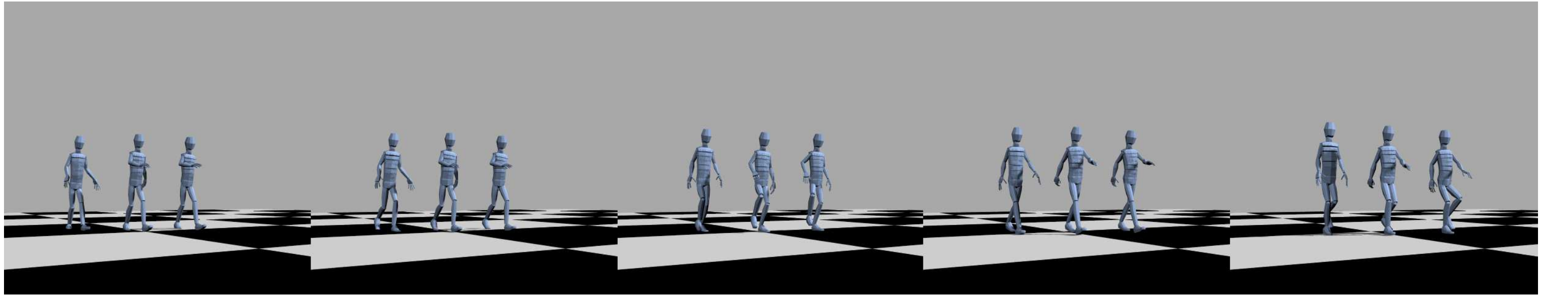


Figure 6.17: (1) Motion capture walk (2) 50% goofy walk blended into motion capture walk using rig controls (3) 50% goofy walk blended into motion capture walk using rig joint angles

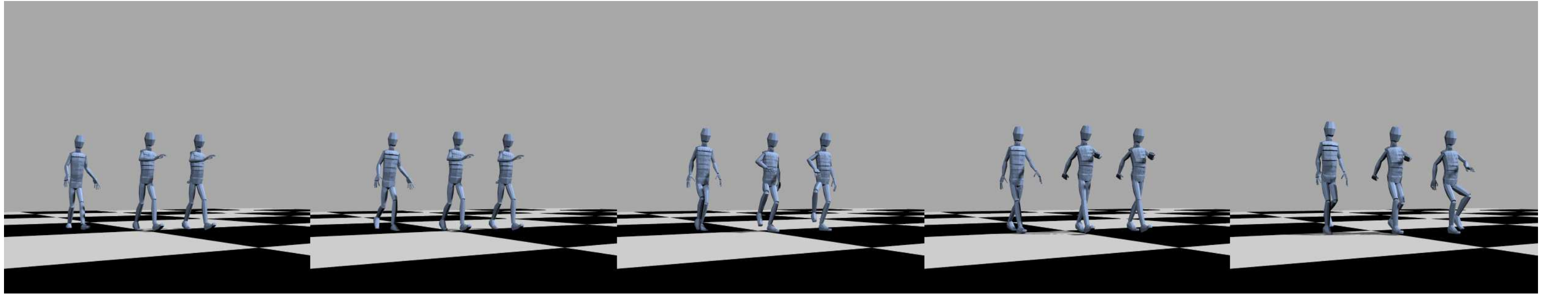


Figure 6.18: (1) Motion capture walk (2) 75% goofy walk blended into motion capture walk using rig controls (3) 75% goofy walk blended into motion capture walk using rig joint angles

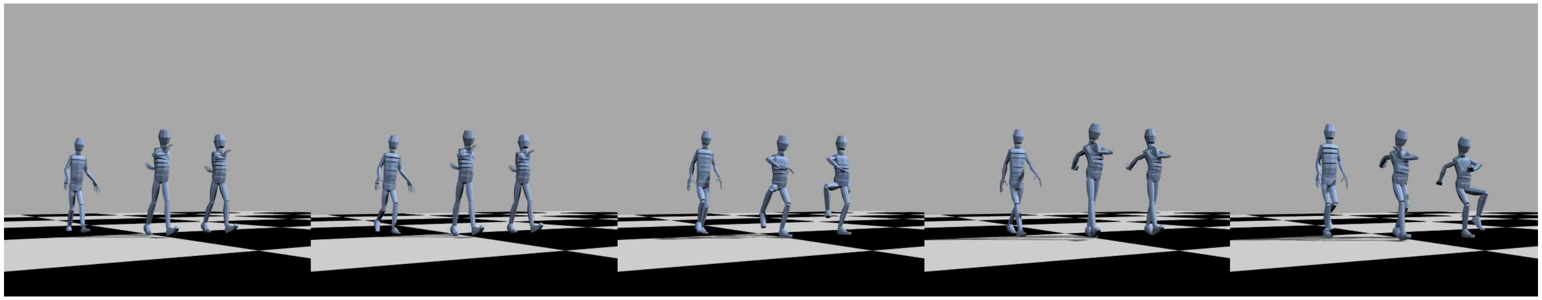


Figure 6.19: (1) Motion capture walk (2) 125% goofy walk blended into motion capture walk using rig controls (3) 125% goofy walk blended into motion capture walk using rig joint angles

6.2 Perception Study Results

In order to evaluated the aesthetic quality of our results, we ran a perception test on 24 subjects. We showed the subjects pairs of motions, one of which was warped and blended using rig controls, and the other using joint angles. Each pair of motions had the same exaggeration style parameter (happy or goofy) and the same exaggeration ratio (25%, 50%, 75%, 100%, 125%). For each of the 10 pairs of motion, we asked the subjects two questions. The first question is "Which walk looks happier?" or "Which walk looks more goofy?" depending on which exaggeration parameter was used. The second question is "Which walk do you like more?"

Figures 6.14, 6.15, and 6.16 show the responses obtained.

When the exaggeration ratio is below 100% for the happy walk, most subjects picked the joint angle method as appearing happier. However, when the exaggeration ratio is 100% or above, most thought the rig control method yielded happier results. When asked to pick the goofier motion, most subjects picked the rig control method as producing goofier results for the center values of the exaggeration ratio, and the rig joint angle method for the 25% exaggeration trial and the extrapolated motion.

When the subject were asked to choose which exaggerated walk they preferred, most subjects picked the rig control method for generating happy walks in 3 of the 5 examples. Conversely, most perfered the rig joint angle method for generating goofy walks in 3 of the 5 examples.

Overall, neither method appears to be strongly favored in any category.

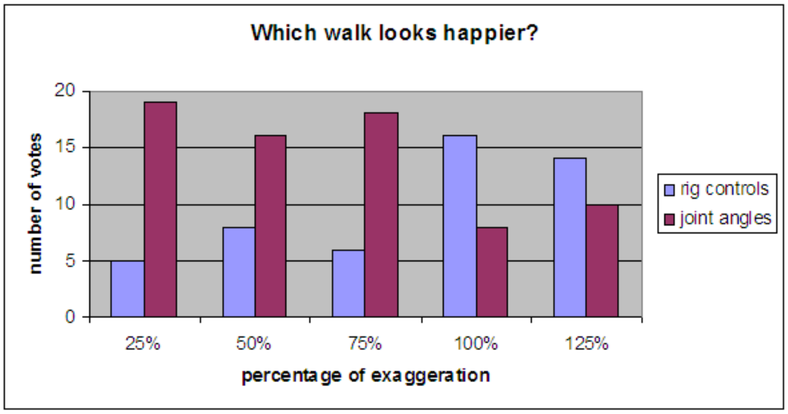


Figure 6.20: Responses to the question “Which walk looks happier?”

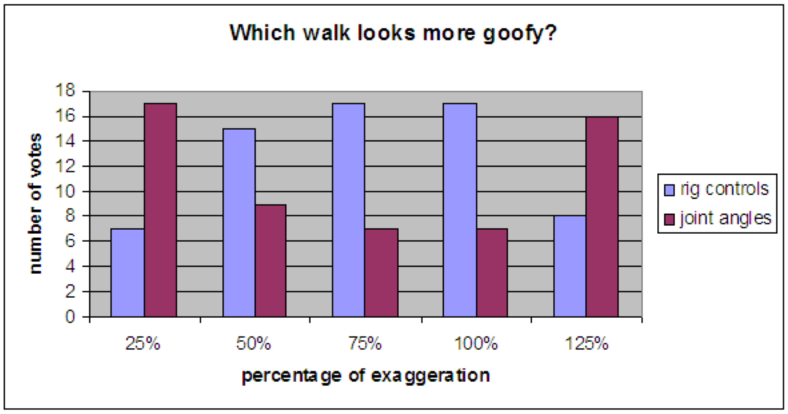


Figure 6.21: Responses to the question “Which walk looks more goofy?”

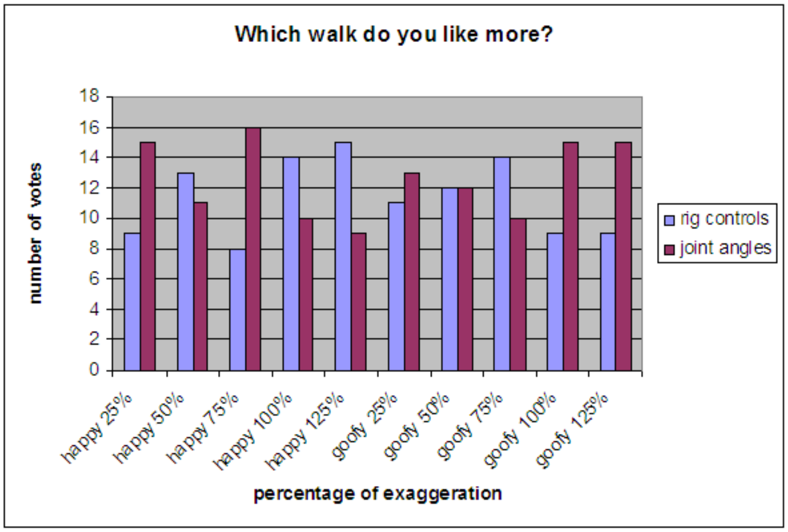


Figure 6.22: Responses to the question “Which walk did you like more?”

Chapter 7

Discussion

7.1 Motion Transfering

Overall, the algorithm for transfering the joint angle motion to our particular character rig is robust and produces good results for walking sequences. We have also begun implementing the algorithm on a second character rig to show its versatility. As mentioned before, the quality of the tranferred motion depends on the target rig and how well the target skeleton matches to the motion capture skeleton. The process is unlikely to work for very incomparable skeletons.

Our algorithm encountered trouble attempting to manipulate arms of our rig since the upper arm controls and elbow controls interacted in strange ways. This could be a potential problem with controls in other rigs as well.

Given the positive results of the motion transfer method, we can expand it slightly such that a user who wants to transfer motion from one skeleton to another skeleton can attach a few rig handles to the target skeleton, and run this algorithm. Our algorithm will match the joints by manipulating the rig handles, and the user can simply export joint angles in the end.

7.2 Motion Warping and Blending

While we obtained good results from time-warping walking motions in both representations, we found that the rig-representation is more robust to the algorithm. The key advantage is that the motion curves in the rig control space are more consistant across similiar motions and vary less wildly. The joint angle data, as we stated previously, cover a larger range as a result of the multiple existing representations of any particular angle, thus visually similar motions can have entirely different motion curves. This forces us to disregard certain degrees of freedom since they would be unhelpful, but the property does not consistantly appear in certain degrees of freedom and hence it is necessary to manually pick the degrees of freedom of interest for each motion pair we wish to warp.

A disadvantage of time-warping in the rig control space, as mentioned previously, is that interpolating between the original set keyframes can sometimes produce undesirable effects. But we found that using a threshold method to detect large changes in trajectories can significantly reduce such artifacts.

In the end, the perception study results did not support either method as generally superior. A majority of the subjects preferred the goofy motion that was warped using rig controls, and on the other hand, a majority perferred the happy motion that was warped using joint angles. Which leads us to believe that the two methods are comparable and the advantage shifts depending on the target motion.

Similarly, we also did not find strong evidence from the perception study that either method yielded better blended motion.

7.3 Future Work

While neither of the two representations of motion yielded results that were clearly stronger for warping or blending, the rig representation is clearly convenient to have and useful for editing motion data. For example, step sizes of walks can be modified by simply scaling the translational values of the feet IK handles, and the root by the same factor, and no foot-sliding artifacts would be introduced. Doing something comparable in the joint angle space becomes messy and complicated.

We have observed some additional advantages of the character rig representation that may be a sign that it is a more convenient way to represent motion. There are a number of experiments that can be performed to explore other potential benefits of using the rig representation. Animators are also familiar with rigs and would presumably prefer to work with them.

We have also noticed that the trajectories of motion capture data in the rig representation tend to have simpler shapes in the important degrees of freedom, and we believe this will lend the data better to curve simplification algorithms than traditional joint angles. If the motion data can be simplified to be represented by a sparse set of keyframes in the rig representation, then this would make motion capture data easier to edit and more flexible for use.

We would also like to convolve motion capture data in the rig representation with the Cartoon Animation Filter [13]. Foot sliding is a very obvious artifact that is introduced when this filter is run on joint angle data. We believe this can be remedied by the use of IK handles, and we would like to compare the results of filtering in the rig control space versus filtering in the joint angle space.

We can also expand the motion exaggeration study to include classes of motions like runs and sneaks because the exaggerations of these motions are also well studied in traditional animation references. Since the keyframed exaggeration is specific to certain classes of motions, it would be desirable to obtain sets of exaggerated data for other classes of motions that users can choose to blend into motion capture data.

Another matter to explore is whether users would really prefer to work with motion capture data in the rig representation. We can explore this option in two ways. First, we can provide online some of the data from the Carnegie Mellon motion capture library in rig representation and monitor if there is an interest from users. Secondly, since our program is already configured to work with the data from Carnegie Mellon's motion library, we are considering providing our motion transfer Maya plugin tool and an accompanying character rig for users to try directly.

Chapter 8

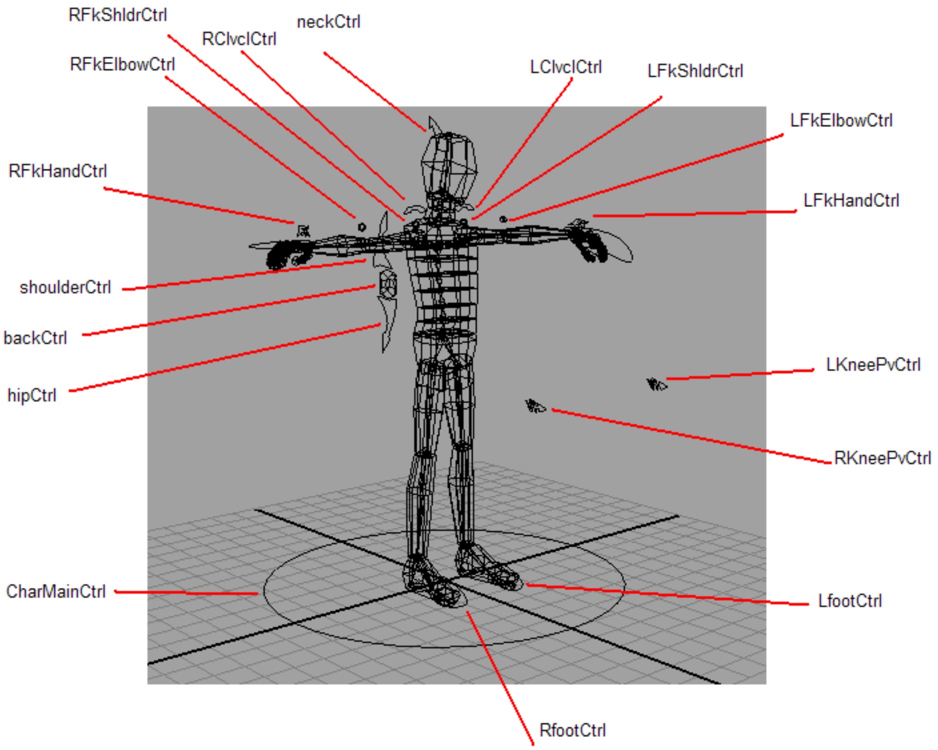
Conclusion

The goal of this thesis is to add expressiveness to motion capture data by blending in hand keyframed exaggerations. However, motion capture data and keyframe animation are traditionally expressed in different formats - rig control parameters versus joint angles. We explored the advantages and disadvantages of applying warping and blending algorithms with these two different formats. We also introduced a method to transfer motion capture data to character rigs using an iterative approach on IK and rotational handles, and a direction projecting approach on directional handles. We warped and blended a happy exaggerated walk as well as a goofy exaggerated walk to a motion captured walk.

We obtained nice results from blending exaggerations into the motion capture data. And while the motions looked different depending on the representation we used, the perception study did not show a clear advantage for either method. However, character rigs are convenient in a large number of ways for human users, and the idea of using character rigs to represent motion in animation research has not really been explored before. We believe that there are other potential benefits, not the least of which is making motion capture data more flexible to use and expanding the variety of already existing data through exaggeration and stylization.

Appendix A

Appendix A



Bibliography

[1] Watt Alan, and M. Watt, *Advanced Animation and Rendering Techniques*, Addison-Wesley, 1992.

[2] Johnston, Ollie, and Frank Thomas. *The Illusion of Life: Disney Animation* 1981: Disney Editions.

[3] Williams, Richard. *The Animator's Survival Kit: A Manual of Methods, Principles, and Formulas for Classical, Computer, Games, Stop Motion, and Internet Animators* 2001: Faber and Faber Limited.

[4] Choi, Jae-jin. *Maya Character Animation* 2004: Sybex Inc.

[5] Lasseter, John. 1987. "Principles of traditional animation applied to 3D computer animation." In *Computer Graphics: SIGGRAPH '87 Conference Proceedings*, ACM SIGGRAPH, p.35-44.

[6] Chenney, S., Pingel,M., Iverson, R., Szymanski, M., "Simulating Cartoon Style Animation" in *Proceedings of Non-Photorealistic Animation and Rendering (NPAR) 2002*, p.133-138.

[7] Li, Y., Gleicher, M., Xu, Y.Q., Shum, H. "Stylizing Motion with Drawings" in *Proceedings of 2003 Symposium on Computer Animation*, Eurographics Association, 2003, p. 309-319.

[8] Bregler, C., Loeb, L., Chuang, E., Deshpande, H. "Turning to the Masters: Motion Capturing Cartoons" in *Computer Graphics: SIGGRAPH '02 Conference Proceedings*, ACM SIGGRAPH, Vol. 21:3, p. 399-407.

[9] Hsu, E., Pulli, K., Popovic, J. "Style Translation for Human Motion" in *Computer Graphics: SIGGRAPH '05 Conference Proceedings*, ACM SIGGRAPH, Vol. 24, pp.910-914.

[10] Safonova, A., Hodgins, J.K., Pollard, N. "Synthesizing physically realistic human motion in low-dimensional, behavior-specific spaces" in *Computer Graphics: SIGGRAPH '04 Conference Proceedings*, ACM SIGGRAPH, Vol. 23, 3.

[11] Safonova, A., Hodgins, J.K. "Analyzing the physical correctness of interpolated human motion" in *Eurographics Symposium on Computer Animation*, ACM SIGGRAPH, 2005.

[12] Kovar, L., Schreiner, J., Gleicher, M. "Footskate Cleanup for Motion Capture Editing" in *Computer Graphics: SIGGRAPH '02 Conference Proceedings*, ACM SIGGRAPH, Vol. 21.

[13] Wang, J., Drucker, S., Agrawala, M. "The Cartoon Animation Filter" in *Computer Graphics: SIGGRAPH '06 Conference Proceedings*, ACM SIGGRAPH, Vol. 25, 3, pp. 1169-1173.

[14] Gleicher, Michael "Retargeting Motion to New Characters" in *Computer Graphics: SIGGRAPH '98 Conference Proceedings*, ACM SIGGRAPH, pp. 33-42.

[15] Shin, H.J., Lee, J., Gleicher, M., and Shin, S.Y. "Computer Puppetry: An Importance-Based Approach." in *ACM Transaction on Graphics.* 2001, pp. 67-94.

[16] Rose, C., Cohen, M.F., Bodenheimer, B. "Verbs and Adverbs: Multidimensional Motion Interpolation" in *IEEE COmputer Graphics and Applications* , IEEE Computer Society Press, 1998, pp. 32-40.

[17] Liu, C.K., Hertzmann, A., Popovic, Z. "Learning Physics-based Motion Style with Nonlinear Inverse Optimization" in *Computer Graphics: SIGGRAPH '05 Conference Proceedings*, ACM SIGGRAPH, Vol. 24, pp. 1071-1081.