Proceedings of the 2nd International Symposium
on Adaptive Motion of Animals and Machines,
Kyoto, March.4-8, 2003

Optimizing Human Motion for the Control of a Humanoid Robot

Alla Safonova[1], Nancy S. Pollard[2] and Jessica K. Hodgins[3]

[1]Computer Science, Carnegie Mellon University, alla@cs.cmu.edu

[2]Computer Science, Brown University and ATR Human Information Science Laboratories, Dept 3, nsp@cs.brown.edu

[3]Computer Science, Carnegie Mellon University and ATR Human Information Science Laboratories, Dept 3, jkh@cs.cmu.edu

Abstract

Using pre-recorded human motion and trajectory tracking, we can control the motion of a humanoid robot for free-space, upper body gestures. However, the number of degrees of freedom, range of joint motion, and achievable joint velocities of today's humanoid robots are far more limited than those of the average human subject. In this paper, we explore a set of techniques for limiting human motion of upper body gestures to that achievable by a Sarcos humanoid robot located at ATR. In particular we have found that it is important to preserve the configuration of the hands and head for upper body motion. We assess the quality of the results by comparing the motion of the human actor to that of the robot, both visually and quantitatively.

1. Introduction

Humanoid robots are already common in theme parks where the investment for a new attraction is substantial. To make humanoid entertainment robots a viable alternative for smaller scale attractions such as location-based entertainment venues, museums, or restaurants, we need easier ways of programming these robots. Entertainment robots must have a natural and entertaining style of motion and often require substantial motion databases to ensure a large variety of behaviors.

For a humanoid robot, such as the Sarcos robot at ATR (DB) [1] shown in Figure 1, one obvious approach is to drive the motion of the robot with motion capture data recorded from a professional actor. Such data would contain the timing and many of the other subtle elements of the actor's performance. However, the current mechanical limitations of humanoid robots prevent the recorded motion from being directly applied, unless the human actors use only a fraction of their natural joint range and move with slower velocities than those commonly seen in human motion.

To address these limitations, the location of the markers in the motion capture data is first mapped to the degrees of freedom of the robot by inverse kinematics on individual limbs. A constrained optimization technique is then used to impose joint angles and velocity limits on the motion while avoiding self-collisions. Optimization techniques allow us to transform the motion to the capabilities of the humanoid robot by specifying an objective function and a set of constraints that preserve the salient characteristics of the original motion. The robot tracks the trajectories of the transformed data using a position and velocity tracking system with feedforward trajectory learning.

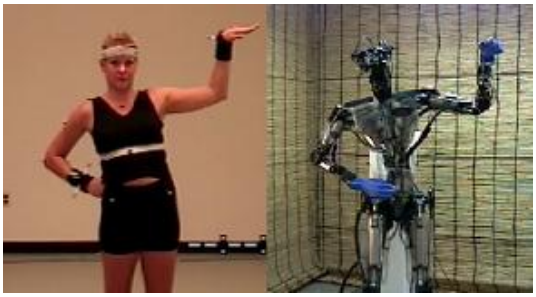

Figure 1: The Sarcos humanoid robot at ATR (DB) tracking motion capture data of a human actor.

We tested these techniques with fourteen motion sequences from seven professional actors. Each subject performed to the same audio track of the children's song, "I'm a little teapot." We chose this selection because it was familiar enough that most actors would perform the motion in a similar but not identical way. It was our hope that an individual's style would be preserved through the transformations necessary to allow the robot to perform the motion.

2. Related Work

Two bodies of work are relevant to this research: robotics researchers have explored algorithms for adapting human motion to humanoid and legged robots and researchers in computer animation have explored algorithms for adapting human motion to animated

characters.

Because of the difficulty of animating complex characters, the animation community has devoted a significant effort to adapting human motion to animated characters. In his motion editing and retargeting work, Gleicher chose to perform a trajectory optimization that did not maintain physical realism in order to allow interactive response for the user [2, 3]. In performance animation or computer puppetry, in contrast, the motion is applied directly to the character without the opportunity for user intervention. Shin et al. implemented an importance-based approach in which the relative importance of joint angles for freespace movements and end effector position and orientation were evaluated based on the proximity of objects and on a priori notations in the performer's script [4]. Their system was used to animate a character in real time for broadcast. Zordan used data recorded from a magnetic motion capture setup to drive the upper body motion of dynamically simulated human figures that box and play pingpong [5].

In robotics, motion data has been proposed by Dasgupta and Nakamura as a way to modify the walking pattern of legged robots to appear more human-like [6]. Because balance is of primary concern in walking, they used the motion data to specify a trajectory for the zero moment point in the control system rather than using it explicitly as desired joint angles.

Riley and her colleagues adapted motion data recorded from an Optotrak motion capture system to the same robot used in this work [7]. If the motion was outside of the robot's joint angle limits, their system translated and scaled the joint trajectory globally. This approach kept the magnitude of the motions as large as possible but at the cost of perhaps using a part of the joint space that the human had not used.

In an earlier version of this work [8], we explored a local scaling approach designed to preserve as much as possible local variations in the motion, such as oscillations. Although this approach did produce reasonable results, we observed that large errors in hand configuration were distracting and that scaling introduced self-collisions. The current paper addresses both of these problems.

3. Techniques for Limiting Motion

For this work we used the same experimental setup and motion data as in [8]. We used a commercially available motion capture system from Vicon [9] to capture the motion of seven professionally trained actors performing twice to a pre-recorded audio track of a children's song.

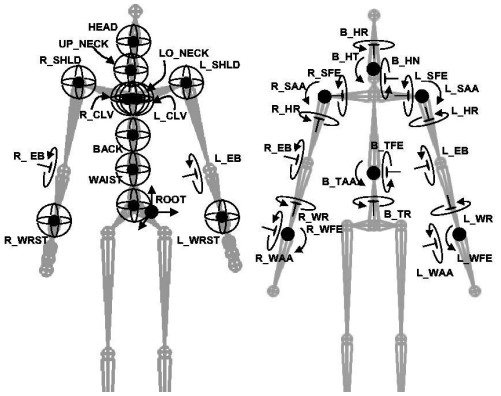

Figure 2: Degrees of freedom for the upper body of the motion capture skeleton (left) and the robot (right). For the human skeleton, all joints are three degree of freedom ball joints except the ROOT (6DOF), the elbows L_EB and R_EB (1DOF each), and the clavicles L_CLV and R_CLV (2DOF each).

The captured motion must be processed in several ways before it can be applied to the humanoid robot: (1) the motion is constrained to match the degrees of freedom of the robot; (2) joint angle and velocity limits are applied; (3) constraints are added to prevent collisions; (4) a trajectory tracking control system is used to follow the motion. The first and fourth processing steps are unchanged from [8]. We briefly summarize them in Sections 3.1 and 3.4 and concentrate on the second and third steps of the processing (Sections 3.2 and 3.3).

3.1. Constraining the motion to the robot's joints

The motion capture data is mapped onto a skeleton having 39 degrees of freedom. Because the robot may have different degrees of freedom than the ones recorded by the Vicon motion capture system, the motion data must be mapped to the robot skeleton (Figure 2). The mapping is done by setting the robot's joint angles to match the orientation of each segment of the robot with the corresponding link or set of links of the human skeleton. We used Euler angles to represent the rotation because that representation is analogous to the sequential actuators of the physical robot.

Joint angles show large variations when the robot is near gimbal lock. Regions near gimbal lock were encountered frequently in these motions because the robot shoulder has a singularity when the arms are at 90 degrees abduction, or swung out to the side of the body to a horizontal position. To address this prob-

lem, we locate regions in the data that are near gimbal lock and compute a restricted degree-of-freedom solution within those regions. See [8] for more details.

3.2. Joint angle and velocity limits

The range of motion and achievable joint velocities of humanoid robots are significantly less than that commonly seen in human motion. Therefore, we need to modify human motion data to bring it within the joint angle and velocity limits of the robot while trying to preserve important characteristics of the original human motion as much as possible. In [8] we used a non-uniform, local scaling to modify the trajectory of each joint angle separately to lie within the robot's joint angle limits while retaining most of the individual oscillations seen in the original motion. To bring the motion within the velocity limits of the robot, each joint angle curve is then scaled by averaging the results of an ideal, simulated, velocity-limited tracking controller run both forward and backward in time.

The above procedure nicely preserves the individual oscillations seen in the original motion, but because it performs scaling of each joint angle independently it does not preserve the overall configuration of the robot's limbs. To fix this problem, we used an optimization procedure that scales all angles at the same time.

Trajectory optimization minimizes some objective function subject to satisfying a given set of constraints. A constrained optimization problem has four main components: the objective function, a set of constraints, an initial guess and the optimization technique.

3.2.1. Objective function

A good objective function should preserve desirable properties in the motion and avoid undesirable artifacts. Let M represent the output motion and $M(t)$ represent a particular frame of that motion at time t. The objective function, $G(M)$, is a weighted sum of three components:

$$G(M) = w_a G_a(M) + w_c G_c(M) + w_l G_l(M)$$

where, G_a is the component that preserves oscillations seen in the original motion; G_c is the component that preserves the configuration of the robot's body segments; G_l is the component that prevents the motion from approaching the angle limits too closely; w_a, w_c and w_l are the corresponding weights. We discuss each of the components and show the relative importance of each in the next four Sections.

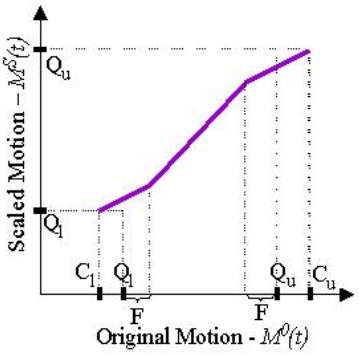

Figure 3: Scaling human data to be within robot angle limits.

Preserving oscillations seen in the original motion
Preserving oscillations in the motion is often important for maintaining the characteristics of the motion. Therefore our objective function includes a term that tries to preserve them. This term minimizes the differences between the output motion, M, and the motion M^S which was computed by scaling each joint angle independently [8].

$$G_a(M) = ||M - M^S||_2$$

We compute M^S in two steps by first, scaling the motion to be within the angle limits, and second, scaling it to be within the velocity limits. Velocity limits are implemented as in [8]. For the angle limits, a simple piecewise linear scaling proved to be sufficient because the motion is modified further during the optimization. We look at each joint angle independently. Let Q_u and Q_l be the desired upper and lower angle limits respectively. Let C_u and C_l be the current angle limits. We compute locally scaled motion $M^S(t)$ given the original motion $M^o(t)$ as follows (Figure 3):

$M^S(t) = aM^o(t) + b$, where

$a = 1$ and $b = 0$,	if $(Q_l+F) < M^o(t) < (Q_u-F)$
$a = \frac{F}{C_u - Q_u + F}$ and $b = (Q_u-F) - a(Q_u-F)$,	if $M^o(t) \geq (Q_u-F)$
$a = \frac{F}{C_l - Q_l - F}$ and $b = (Q_l+F) - a(Q_l+F)$,	if $M^o(t) \leq (Q_l+F)$

Preserving configuration
Because the configuration of the robot's body segments is important for the overall appearance of the motion, we pick a set of points, P, on the robot's skeleton and add a term into the objective function that minimizes the sum of squared distances between these points in the original and output motions.

We included the end-points of all robot body segments in the set of points P. Additional points could

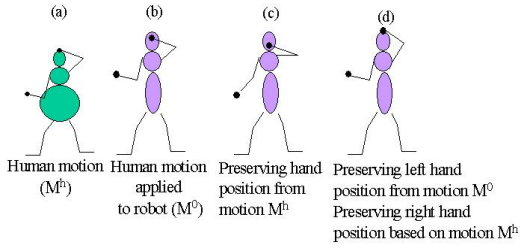

Figure 4: Preserving positions of the end-effectors of the robot.

be added to preserve the orientation of the hands and the head of the robot but the G_a component of the optimization function also preserves the orientation. Let $Pos(M^0)$ and $Pos(M)$ represent the positions of all points in P for the original motion M^0 and the output motion M:

$$G_c(M) = ||Pos(M) - Pos(M^0)||_2$$

The positions are from M^0 (the original motion that was mapped to the robot's degrees of freedom but not joint angle or velocity limited) rather than from the human motion M^h. In our experiments preserving the positions from the motion M^h did not produce good results due to the differences in proportions of the human and the robot (Figures 4a and 4c). By preserving the position of points from the motion M^0, we are preserving the configuration of the robot in the human motion M^h (Figure 4b). When the position of the end-effector of the human arm in motion M^h is particularly important (left arm in Figure 4a) constraints can be added to the optimization (see Section 3.2.2).

Effect of G_c ***and*** G_a

We found that using both G_c and G_a produces better results than using either one of these components separately. Figure 5 demonstrates the importance of the G_c component of the optimization function. M^0 is the original motion. M^1 is the optimized motion where the G_c component is given low weight. M^2 is the optimized motion where the G_a and G_c components are given sufficient weight. At the frame shown, the two angles of the upper arm of the robot in the original motion, M^0, are outside of their angle limits (Figures 5d and 5e). As can be seen from the Figure 5b, the change made to the trajectories of these two angles in motion M^1, results in a significantly different configuration of the arm. The G_c component in motion M^2 (Figure 5c), compensates for the large changes made to the two angle components of the upper arm by modifying the angles of the forearm (Figure 5f), which produces a visually far more similar result.

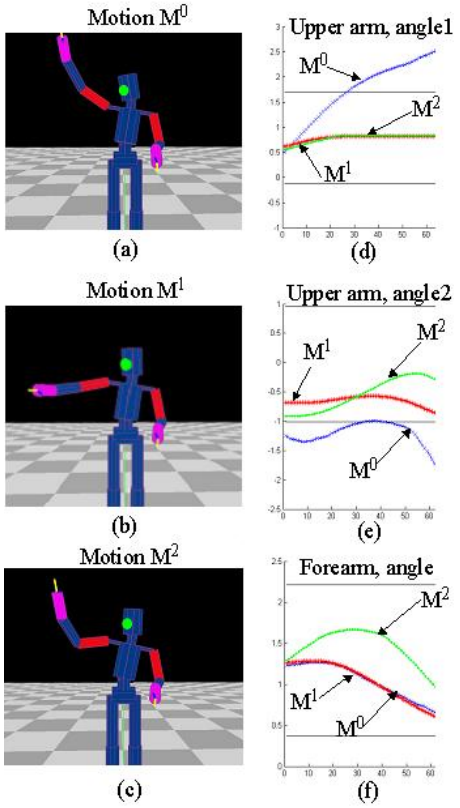

Figure 5: Importance of the G_c component of the optimization function. Figures 5a, 5b, and 5c show the configuration of the robot at a time frame 63, extracted from motions M^0, M^1, and M^2 respectively. Figures 5d- 5j show the trajectories of various angles of the upper arm of the robot. Horizontal lines show the angle limits.

Figure 6 shows the importance of the G_a component with a graph of one angle of the upper arm. M^0 is the original motion. M^1 is the optimized motion where G_a component is given low weight. M^2 is the optimized motion where the G_a and G_c components are given sufficient weight. The original motion represents a human moving his hand up and down quickly, and therefore has large oscillations in the angles and is significantly outside of the angle limits. The motion M^1 lacks these oscillations because the G_c component minimizes the error in the displacement of the points in P by keeping the joint at the joint limit. However, if we also include the angle component into the optimization function, we get a motion M^2, which is visually much better.

The main advantage of the G_c component is the ability to compensate for a large change in the trajectory of a particular angle by adjusting the trajectories

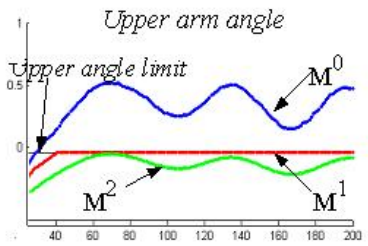

Figure 6: Importance of the G_a component.

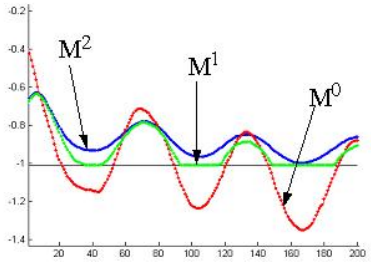

Figure 7: Importance of the G_l component.

of other angles. The main benefit of the G_a component, on the other hand, is to preserve oscillations in the motion.

Avoiding joint limits

If we use an optimization function consisting only of the components described above, the resulting motion will have a tendency to approach the angle limits too closely (Figure 7) because the range of motion of the robot is significantly less than that commonly seen in human motion. The robot does not perform well near the angle limits, and we do not want C^1 discontinuities in the data, so we add an exponential penalty term to our optimization function that prevents the motion from getting too close to the limits:

$$G_l(M) = exp(\frac{-k(M(t)-Q_l)}{Q_u-Q_l}) + exp(\frac{-k(Q_u-M(t))}{Q_u-Q_l})$$

where k controls the shape of the exponential function.

3.2.2. Constraints

We use angle and velocity constraints to restrict the motion to the robot's capabilities. Constraints can also be used to preserve characteristics of the motion. For example, the position of the hands may be important in some motions, particularly when they are in contact with the body or with an object. However, the motions that we used in this work did not include contact

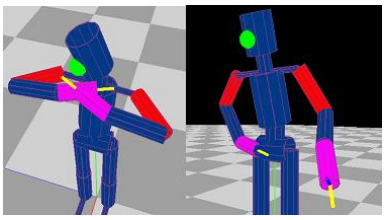

Figure 8: Image on the left shows collision of robot's arms. Image on the right shows collision of the right arm with the hip.

and the absolute positions of the hands was influenced only by that term in the objective function, not by a constraint.

Constraints were used to prevent interpenetrations of body segments that might be introduced by the processing of the motion (Figure 8). These constraints are described in detail (Section 3.3).

3.2.3. Trajectory optimization problem

The configuration of the robot is defined by the angles of its joints, q^i, and position and orientation of the root segment, v, at each moment of time:

$$M(t) = \{v, q^0, ..., q^n\}$$

where n is the number of degrees of freedom. The displacement map is defined as $d(t) = M^1(t) - M^2(t)$ and describes the difference between any two motions M^1 and M^2 at time t (see [10, 11, 3, 12]). The output motion that we compute can then be represented as $M(t) = M^0(t) + d(t)$ where $M^0(t)$ is the original motion and $d(t)$ is a displacement map computed by our optimization routine.

Therefore, we need to compute a smooth displacement map, $d(t)$, such that the output motion, $M(t)$, satisfies all the constraints and minimizes the objective function. Gleicher [3] formulates and solves a single mathematical problem for the entire motion to solve such problem. On the other hand, Lee [12] solves a separate optimization problem for each frame of the motion and then filters the resulting displacement map to reduce discontinuities. Filtering may cause constraint violations so the process is repeated several times with less filtering on each iteration. In our work we use Lee's approach because it is much simpler to implement and provides more predictable solution times.

We solve the optimization problem for each frame of the motion. Because the constraints also incorporate velocity limits, the solution for each frame of the motion is not independent of the solution for the previous

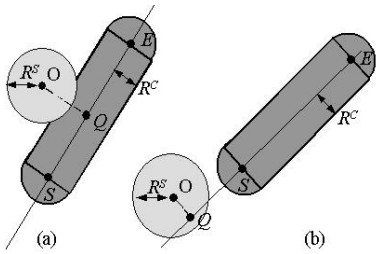

Figure 9: Collision of a sphere with center O and radius R^s and a cylinder parallel to line $L(t) = S + t * (E - S)$, containing points S and E and of radius R^C. Q is the point on line $L(t)$ which is closest to O. (a) Q belongs to line segment (S, E) and therefore we need to add a constraint that the distance between Q and O is larger than $(R^s + R^c)$. (b) Q does not belong to the line segment (S, E) and we do not want the constraint that would require the distance between O and Q to be larger than $(R^s + R^c)$.

frame. In particular, if we perform the optimization forward (starting with the first frame of the motion) then the solution for the first frame imposes restrictions for the subsequent frames. Running the optimization forward and backward (starting from the last frame) and averaging the results of both produces much better results in practice.

In our implementation we used donlp [13]: a commercially available library that solves general nonlinear constrained optimization problems. We also used a modeling language ampl [14] that allows the user easily to formulate linear and nonlinear optimization problems in mathematical terms and automatically generates code appropriate for various existing solvers.

We used the scaled motion, M^S, as an initial guess for the optimization problem because it is usually not far from the solution.

3.3. Collisions

When the motion is processed to fit the robot, interpenetrations (collisions) of body segments can occur (Figure 8) and we add constraints to prevent this problem. For collision detection, we approximate the robot body segments with spheres and cylinders. We consider a motion to be free of self-collisions if no two shapes that approximate the robot's body segments intersect. We now give a mathematical formulation of the constraints that will prevent sphere-sphere, sphere-cylinder and cylinder-cylinder intersections.

To prevent two spheres with centers O^1 and O^2 and radiuses R^1 and R^2 from intersecting we set a constraint that the distance between the centers of two spheres must be larger than the sum of their radiuses.

Now consider a sphere S with center O and radius R^S and a cylinder C parallel to line $L(t) = S + t * (E - S)$, containing points S and E and of radius R^C (Figure 9a). Sphere S and a cylinder C do not intersect, if the distance between any point on the line segment (S, E) and center of a sphere O is larger than the sum of radiuses of the sphere and the cylinder, $(R^s + R^c)$. Let point Q be the point on line $L(t)$ which is closest to O. Then, Q is the projection of vector $(O - S)$ to the line $L(t)$, $Q = S + t_{min}(E - S)$, where

$$t_{min} = \frac{(O - S) \circ (E - S)}{(E - S) \circ (E - S)} \tag{1}$$

If Q belongs to the line segment (S, E) then Q is the point on (S, E) that is closest to the center of the sphere O. Otherwise the closest point is one of the endpoints of the line segment. We can easily specify two constraints that will ensure that the distance between O and the endpoints S and E is larger than $(R^S + R^C)$:

$$(O - S) \circ (O - S) - (R^s + R^c)^2 \geq 0 \tag{2}$$

$$(O - E) \circ (O - E) - (R^s + R^c)^2 \geq 0 \tag{3}$$

We could specify a constraint that the distance between O and Q is larger than $(R^S + R^C)$ in a similar way. However, we only want that constraint to be active if Q belongs to the line segment (S, E). If Q does not belong to (S, E) then even if the distance between O and Q is less than $(R^S + R^C)$ the sphere and the cylinder may not collide (Figure 9b). We accomplish this with a variable W (Equation 5) which is equal to zero if Q does not belong to line segment (S, E) and is negative if Q belongs to line segment (S, E) (variable T in Equation 4 is negative if Q does belong to (S, E) and positive otherwise).

$$T = t_{min} * (t_{min} - 1); \tag{4}$$

$$W = T - sqrt(T * T) \tag{5}$$

Thus, the following constraint will be trivially equal to zero (and thus satisfied) if Q does not belong to line segment (S, E) and will be violated only if Q belongs to (S, E) and $|O - Q| < (R^S + R^C)$:

$$W * ((O - Q) \circ (O - Q) - (R^1 + R^2)^2) \leq 0 \tag{6}$$

Constraints for preventing cylinder-cylinder intersection are derived using similar reasoning.

3.4. Trajectory Tracking and Learning

The joint angle and velocity limited trajectory is used as input to a trajectory tracking control system that computes joint torques for the robot using a PD servo. The joint angles produced by the trajectory tracking servo necessarily lag behind the desired joint angles.

Figure 10: Frames from motion of actor #2.

We used the trajectory learning approach that was originally proposed by Kawato et al. [15, 16] to improve the performance of the tracking.

4. Assessment of the Motion Quality

Figures 10 and 13 show frames from the motion of actors #2 and #3 respectively. In many of the frames the robot motion is a good match for the actor motion. Figures 11 and 12 show the differences between the recorded human motion mapped to the robot's degrees of freedom but not joint angle or velocity limited and the actual motion executed by the robot. In Figure 11 each number is the average of the sum squared error over all joint angles and over the entire motion. In Figure 12 each number is the average of the sum squared error over the positions of points in P which we picked on the robot's body and over the entire motion. If the robot trajectory is a good match for the human motion from which it was derived, we would expect smaller numbers on the diagonal and that is indeed the case.

	1a	1b	2a	2b	3a	3b	4a	4b	5a	5b	6a	6b	7a	7b
1a	206	349	1085	1226	922	850	984	991	806	868	953	889	1066	941
1b	350	192	983	1117	904	848	856	849	677	739	865	767	986	816
2a	1177	1089	242	726	1038	1050	630	717	900	911	962	949	1082	1070
2b	1204	1097	563	218	938	1013	699	746	987	956	796	788	910	933
3a	1140	1161	1266	1278	436	1020	1228	1210	1260	1095	1214	1187	1285	1217
3b	1145	1176	1363	1395	1128	454	1377	1360	1485	1340	1391	1389	1339	1335
4a	925	839	547	723	897	957	149	226	627	655	788	692	894	719
4b	893	797	542	678	861	914	183	141	578	653	735	611	800	646
5a	829	722	1073	1287	1074	1144	782	778	263	540	971	853	1042	780
5b	972	914	1139	1247	1040	1076	942	983	734	305	1137	1157	1166	1046
6a	1170	1083	1042	929	1167	1247	1016	997	1141	1217	357	700	934	879
6b	1829	1677	1584	1536	2011	2137	1447	1403	1643	1862	1433	995	1662	1497
7a	1270	1196	1263	1172	1330	1158	1190	1143	1154	1190	1048	1084	398	688
7b	985	878	1110	1106	1046	1055	812	781	742	885	793	739	626	224

Figure 11: The squared error of the joint angles for each of the fourteen trials. If the robot trajectory is a good match for the input error of a particular trial, the error should be smaller. If the other trial from that individual also has a low number that indicates that the two trials were similar (and is perhaps evidence of a particular style).

	1a	1b	2a	2b	3a	3b	4a	4b	5a	5b	6a	6b	7a	7b
1a	3	10	29	38	30	28	29	29	30	28	33	37	33	30
1b	1	3	10	13	10	9	10	10	10	9	11	12	11	10
2a	10	9	1	6	10	10	8	9	9	8	6	9	11	9
2b	12	10	6	1	12	13	9	10	10	8	9	8	13	11
3a	11	11	11	13	2	12	13	14	13	9	13	14	16	12
3b	10	11	10	14	11	2	12	13	13	11	14	16	13	12
4a	8	7	6	8	10	10	1	1	4	6	7	7	7	5
4b	9	8	8	9	12	11	2	1	5	7	8	7	8	5
5a	10	8	9	10	12	13	6	6	1	8	11	9	10	6
5b	9	8	8	8	8	11	7	8	7	1	10	11	9	7
6a	12	10	7	9	13	14	10	11	12	11	1	7	13	11
6b	13	10	10	8	13	16	9	10	9	12	7	1	14	10
7a	11	12	13	14	15	13	9	11	12	10	13	15	2	8
7b	9	7	8	10	10	10	5	6	5	7	9	8	6	1

Figure 12: The squared error of the points P which we picked on the robot's body for each of the fourteen trials.

5. Discussion

The range of motion and the achievable joint velocities of humanoid robots are significantly less than that commonly seen in human motion. In this paper we have presented a set of constraints and an objective function that make the output motion suitable for the humanoid robot. Our approach considers all joint angles for a particular frame of the motion together and thus does not assume their independence. We have added collision constraints into the optimization problem, thus preventing interpenetration of robot body

Figure 13: Frames from motion of actor #3.

segments and allowing the motion to run safely on the robot. The experimental evaluation shows that this results in a robot motion that closely resembles the original human motion.

The approach that we have used to solve the optimization problem solves a separate problem for each frame of the motion. Because of the interdependency between solutions for the consecutive frames such an approach may not necessarily find the best solution over the entire motion. Solving one large optimization problem over the entire motion may result in a more optimal solution because such a method considers the relationships between frames. A global approach might also provide a better solution to the gimbal lock problem (Section 3.1). The cost of this approach, however, is very large optimization problem that might be difficult to solve.

In the longer term, we would like to more rigorously answer the question of whether the style of an individual actor is retained despite the transformations performed on the data to fit it to the robot. We plan to show paired videos of motions (one robot, one human) to subjects and ask them whether the two videos are based on the same motion sequence. If the results of this experiment show that the subjects are able to make this distinction then we will have demonstrated that the style of a particular performance was retained. A second, more difficult, test would assess whether an individual's style is recognizable. In this experiment, subjects would view several instances of a particular actor performing and then decide if a given robot motion had been performed by that actor. If subjects can make this judgment successfully, then the actor's style has been retained in the motion.

References

[1] DB, "http://www.his.atr.co.jp/cyh/," .

[2] M. Gleicher, "Motion editing with spacetime constraints," *1997 Symposium on Interactive 3D Graphics*, pp. 139–148, April 1997.

[3] M. Gleicher, "Retargeting motion to new characters," *Proceedings of SIGGRAPH 1998*, pp. 33–42, July 1998.

[4] H. J. Shin, J. Lee, M. Gleicher, and S. Y. Shin, "Computer puppetry: An importance-based approach," *ACM Transactions on Graphics*, April 2001.

[5] V. B. Zordan and J. K. Hodgins, "Motion capture-driven simulations that hit and react," in *Symposium on Computer Animation*, 2002.

[6] A. Dasgupta and Y. Nakamura, "Making feasible walking motion of humanoid robots from human motion capture data," *Proceedings for the 1999 IEEE International Conference on Robotics and Automation*, pp. 1044 – 1049, May 1999.

[7] M. Riley, A. Ude, and C. G. Atkeson, "Methods for motion generation and interaction with a humanoid robot: Case studies of dancing and catching," *AAAI and CMU Workshop on Interactive Robotics and Entertainment 2000*, April 2000.

[8] N.S. Pollard, J.K. Hodgins, M.J. Riley, and C.G. Atkeson, "Adapting human motion for the control of a humanoid robot," *Proceedings of the IEEE International Conference on Robotics and Automation*, May 2002.

[9] Vicon Motion Systems, "http://www.vicon.com/," .

[10] A. Witkin and Z. Popović, "Motion warping," in *Proceedings of SIGGRAPH 1995*, Aug. 1995.

[11] A. Bruderlin and L. Williams, "Motion signal processing," *In Proceedings of SIGGRAPH 95*, pp. 97 – 104, August 1995.

[12] J. Lee and S.Y. Shin, "A hierarchical approach to interactive motion editing for human-like figures," *In Proceedings of SIGGRAPH 99*, pp. 39–48, 1999.

[13] DONLP, "http://plato.asu.edu/donlp2.html/," .

[14] AMPL, "http://www.ampl.com/cm/cs/what/ampl/," .

[15] M. Kawato, K. Furukawa, and R. Suzuki, "A hierarchical neural-network model for control and learning of voluntary movement," *Biological Cybernetics*, vol. 57, pp. 169–185, 1987.

[16] M. Kawato, "Feedback-error-learning neural network for supervised motor learning," in *Advanced Neural Computers*, R. Eckmiller, Ed., pp. 365–472. Elsevier Science Publishers, 1990.