On-line Motion Retargetting

Kwang-Jin Choi and Hyeong-Seok Ko

SNU Human Animation Center
School of Electrical Engineering
Seoul National University
E-mail: {kjchoi,ko}@graphics.snu.ac.kr

Abstract

This paper presents a method to retarget the motion of a character to another in real-time. The technique is based on inverse rate control, which computes the changes in joint angles corresponding to the changes in end-effector position. While tracking the multiple end-effector trajectories of the original subject or character, our on-line motion retargetting also minimizes the joint angle differences by exploiting the kinematic redundancies of the animated model. This method can generalize a captured motion for another anthropometry to perform slightly different motion, while preserving the original motion characteristics. Because the above is done in on-line, a real-time performance can be mapped to other characters. Moreover, if the method is used interactively during motion capture session, the feedback of retargetted motion on the screen provides more chances to get satisfactory results. As a by-product, our algorithm can be used to reduce measurement errors in restoring captured motion. The data enhancement improves the accuracy in both joint angles and end-effector positions. Experiments prove that our retargetting algorithm preserves the high frequency details of the original motion quite accurately.

1. Introduction

The dream of animating complex living creatures with pure computation (such as inverse kinematics, or dynamic control) proved impractical. Even though creatures are not free from physics, their motion is not a direct consequence of physics. Dynamic control can provide solutions based on simplified assumptions about human motion. However, the result tends to look quite mechanical.

If a high quality character animation has to be produced during a short period of time, motion capture might be a most reasonable choice these days. The captured data itself is for a specific person in performing specific motion. Whenever the data needs to be reused, it has to be retargetted to account for the differences in the anthropometry and motion. Therefore motion retargetting is emerging as an important technique in recent character animation.

If the original motion characteristics are severely lost during motion retargetting, the technique loses its merit over the above pure computation approaches. The problem we try to solve in this paper can be summarized as: (1) finding in real-time the motion retargetted to a new character that has different anthropometric proportions, and (2) at the same time, preserving the features of the original motion during the retargetting. (3) As a by-product, it is possible to use the above retargetting algorithm for enhancing motion capture data so that the errors in joint angles and end-effector positions are reduced.

On-line motion retargetting presented in this paper is based on *inverse rate control* [17] (or resolved motion rate control), which is a way to implement inverse kinematics based on Jacobian. It computes the changes in joint angles corresponding to the changes in end-effector position. While tracking the multiple end-effector trajectories of the original subject or character, our on-line motion retargetting imitates the joint motion of the original character by exploiting the kinematic redundancies of the animated model. Moreover, jerky motion is prevented since the next configuration is dependent on the previous configuration in inverse rate control. As will be shown in later experiments, the high frequency details of the original motion, which carries important characteristics of the motion, are also well preserved by our algorithm.

Figure 1 shows the on-line retargetting process schematically. The input is a stream of joint angle vectors θ^{src} of the measured subject in the source motion and another stream of the reference (or desired) end-effector positions x_1 of the animated character at discrete time ticks. The output is a stream of joint angle vectors θ^{des} of the animated character during the destination motion at corresponding time ticks. The filter in the figure is causal. i.e., the output is calculated based on the current and immediately previous input values, but does not dependent on the future input. It explains why

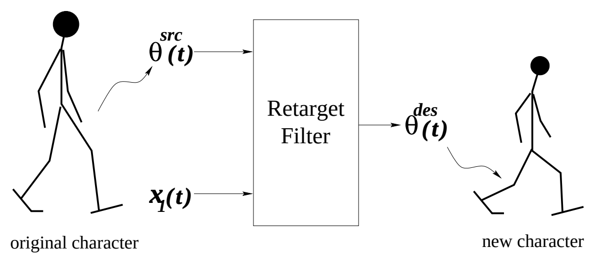

Figure 1. On-line Motion Retargetting Filter

it is called *on-line*.

If the retargetting can be done in on-line, real-time performance can be mapped to another character, or the feedback of the retargetted animation can facilitate motion capture session so that satisfactory results can be obtained with fewer trials. Since the memory required for on-line retargetting does not increase with time, our algorithm can handle an infinitely long sequence of motion.

The primary goal of our on-line motion retargetting (OMR) is to track the given reference end-effector trajectory $x_1(t)$, and the secondary goal is to imitate the pattern of joint angle trajectory $\theta^{src}(t)$ as much as possible. Therefore $\theta^{src}(t)$ carries the content to be retargetted, and $x_1(t)$ carries the variations needed during the retargetting. For example, when there is a bat-swing motion, we can obtain different swing motions aiming at different hit positions by specifying $x_1(t)$ appropriately.

As a by-product, our OMR algorithm can be used to reduce measurement errors in restoring captured motion. For this data enhancement, we use captured position data for $x_1(t)$ even though it can be calculated from $\theta^{src}(t)$ by forward kinematic positioning, to recover from possible measurement errors in joint angles. If the positioning of the pose is done from joint angles alone, the errors can accumulate as the forward kinematic positioning propagates toward the end-effector. The end-effector position data $x_1(t)$ can be utilized to limit the above error accumulation within a certain range.

In Section 2 recent work related with motion retargetting is reviewed. Section 3 discusses inverse rate control and its implementation, and Section 4 presents the formulation of motion retagetting problem with inverse rate control. Section 5 discusses how our OMR can be used to reduce measurement errors in restoring captured motion. Section 6 shows the results obtained by our technique, and finally the conclusion follows.

2. Related Work

Several techniques have been proposed for reusing or altering existing motions. Witkin et al's motion warping [19] and Bruderlin et al's motion displacement mapping [4] discuss motion editing technique based on direct manipulation of data curves. Bruderlin et al [4] and Unuma et al [11] utilized signal processing techniques for motion editing. Wiley et al [18] proposed the interpolation synthesis algorithm that chooses and combines most relevant motions from the database to produce animation with a specific positional goals.

Though some of the techniques above can be used for motion retargetting problem with user's extra efforts, they don't specifically address the motion retargetting problem. In [3], Boulic and Thalmann presented the combined direct and inverse kinematic control technique for motion editing. The concept called *coach-trainee metaphor* is very similar to the motion retargetting problem formulation. The fundamental idea is to consider the joint motion of *coach* as a reference input to *trainee* motion for the secondary task exploiting the null space of the Jacobian when solving inverse kinematics. The inverse kinematic constraint is given by half-space such as plane, cylinder, or sphere.

Although their approach shares the technique of utilizing the redundancy in inverse kinematic control with ours, the problem they solved is not the motion retargetting but is rather a motion correction technique since the end-effector constraint specified by half-spaces is not general to solve the motion retargetting problem.

A method which is devoted to the motion retargetting problem was proposed by Gleicher [6]. He used the space-time constraint method that minimizes an objective function $g(x)$ subject to the constrains of the form $f(x) = c$. The constraints can represent the ranges of parameters, or various kinds of spatial-temporal relationship among the body segments and the environment. The objective function is the time integral of the signal displacement between the source and destination motion. i.e.

$$g(x) = \int (m^{src} - m^{des})^2 dt. \qquad (1)$$

Since the whole interval has to be integrated to find the optimal solution, the method is intrinsically an off-line process.

The global method as above can correlate frames back and forth within the whole duration and thus generally produces more smooth results compared to the local method such as our OMR technique. But the look-ahead property of the global method is effective when the constraints are imposed only at sparse key frames. Our OMR takes continuous trajectories of constraints as input, so that it produces globally coherent motion in spite of local computation. The global coherence is also achieved from the effort to exploit the redundancy of the system in resembling the original motion. The local coherence of the motion comes from the fact that the adjacent frames are inter-related by the inverse rate control. Therefore, without significant degradation of quality, our algorithm provides much faster and interactive way of motion retargetting.

Bindiganavale and Badler [2] presented a method to abstract and edit motion capture data. Their algorithm detects significant events and abstracts constraints from the motion, and imposes those constraints to other character. The constraints abstracted from the motion is solved by inverse kinematics at significant frames and then those frames are interpolated. Although the constraint abstraction is an improvement compared to the other techniques, the interpolation technique might fail to preserve the high frequency details if the key frames are sparsely spaced.

3. Inverse Rate Control

In an articulated figure, the joint configuration can be related to the position and orientation of the end effector by a kinematic mapping $f : \Theta \rightarrow X$, which maps the joint space Θ to Cartesian space X. The mapping is usually a nonlinear equation given by

$$x_1 = f_1(\theta), \tag{2}$$

where x_1 is an m-dimensional vector and θ is an n-dimensional vector. $m = 3$ if we are interested only in position, or $m = 6$ if we are interested in both position and orientation of the end-effector. m can be 12 or 18 if we want to impose multiple end-effector constraints.

If we differentiate the above equation, we obtain

$$\dot{x}_1 = J_1\dot{\theta}, \tag{3}$$

where $\dot{x}_1$ and $\dot{\theta}$ denote the end-effector positional velocity and joint angle velocity, respectively. J_1 is called Jacobian and is an $m \times n$ matrix that linearly relates the end effector velocity and joint angle velocity at the moment.

Given the end effector velocity, we can get joint angle velocity by inverting the Jacobian. However, most articulated figures have kinematic redundancy and thus the inverse of Jacobian is not unique. (more specifically, $m < n$) Therefore there are an infinite number of possible solutions that satisfy Equation 3. Some criteria can be specified to pick one that best fits for our purpose. One of popular criteria is called the *minimal norm solution*

$$\dot{\theta} = J_1^+\dot{x}_1, \tag{4}$$

where $J_1^+ = J_1^T(J_1J_1^T)^{-1}$ is the pseudo inverse[1] of J_1.

Equation 4 gives a particular solution, and can be generalized to include all possible solutions by adding a term from the null space of J_1 as in

$$\dot{\theta} = J_1^+\dot{x}_1 + (I - J_1^+J_1)y, \tag{5}$$

where y is an arbitrary n-dimensional vector. $(I - J_1^+J_1)$ projects y onto the null space of J_1. This null space term corresponds to the redundant degrees of freedom, and can be utilized to perform secondary priority tasks [7, 10, 20, 21]. For example, consider the following task set.

$$\begin{aligned} \text{primary task: } & x_1 = f_1(\theta), \text{ thus } \dot{x}_1 = J_1\dot{\theta} \\ \text{secondary task: } & x_2 = f_2(\theta), \text{ thus } \dot{x}_2 = J_2\dot{\theta} \end{aligned}$$

If the equation

$$\dot{\theta} = J_1^+\dot{x}_1 + (I - J_1^+J_1)J_2^+\dot{x}_2, \tag{6}$$

is used for Equation 5, then the primary goal is accurately achieved in the case of continuous domain[2] and the secondary goal is also achieved in an optimal sense.

Another way to utilize the redundancy of the system is to set y to the gradient $-\alpha\nabla g$ of a criterion function $g(\theta)$ in Equation 5. Then integration of Equation 5 tries to reduce the value of $g(\theta)$ while the end-effector is made to track the given trajectory [9].

3.1. Closed-loop Inverse Rate Control and Its Discrete Implementation

To control the articulated figure to follow given reference end-effector trajectory $x_1(t)$, $J_1^+\dot{x}_1(t)$ should be integrated to give the value of $\theta(t)$ (Equation 4). But this open-loop fashion of integration can not eliminate the initial tracking error $e_1(t_0) = x_1(t_0) - x_1^{des}(t_0)$, where $x_1^{des}(t)$ is the resulting end-effector position at time t in the destination motion.

Balestrino et al [1], Tsai and Orin [15], and Sciavicco and Siciliano [14, 13] proposed the *closed-loop inverse kinematics* (CLIK) scheme based on Jacobian pseudo-inverse. CLIK leads to zero steady state error which means that the error is exponentially convergent to zero for a fixed target position. For CLIK, Equation 4 has to be modified to

$$\dot{\theta}(t) = J_1^+(\dot{x}_1(t) + K_1e_1(t)), \tag{7}$$

where K_1 is a positive definite matrix we can provide. It can be easily shown that as the smallest eigen value of K_1 becomes large, the convergence rate increases accordingly since the error dynamics is governed by the relation $\dot{e}_1 + K_1e_1 = 0$. In a continuous time formulation such as Equation 7, a large value of K_1 is desirable. However, as will be shown below an arbitrarily large K_1 doesn't guarantee convergence in implementing the discrete version of Equation 7.

[1]We actually used damped least squares solution, $\dot{\theta} = J_1^T(J_1J_1^T + \lambda^2I)^{-1}\dot{x}_1$, to get consistent motion near the singularities [12, 16].

[2]In the discrete domain, since the Jacobian is a linear estimation of a non-linear function, the integration can produce errors, especially when the step size is large.

The difference equation corresponding to Equation 7 is given by

$$\Delta\theta[n] = J_1^+[n-1](\Delta x_1[n] + K_1 e_1[n]), \tag{8}$$

where

$$\Delta\theta[n] = \theta[n] - \theta[n-1],$$

and

$$\begin{aligned} \Delta x_1[n] &= x_1[n] - x_1[n-1], \\ e_1[n] &= x_1[n] - x_1^{des}[n]. \end{aligned} \tag{9}$$

Here $u[i]$ is the value of the function $u(t)$ at the discrete time t_i.

Equation 8 is implicit: to compute $\Delta\theta[n]$ we need to know the value of $e_1[n]$. But computing $e_1[n]$ by Equation 9 in turn requires the value of $x_1^{des}[n]$, which is not available until $\Delta\theta[n]$ is known. Therefore, $e_1[n]$ should be estimated. Below we show that any estimation based on the old values (at $n-1, n-2, \ldots$) requires K_1 to be I for the best tracking performance.

Suppose that we estimated $e_1[n]$ simply with $e_1[n-1]$. Then Equation 8 becomes

$$\Delta\theta[n] = J_1^+[n-1](\Delta x_1[n] + K_1 e_1[n-1]). \tag{10}$$

To obtain the error equation, multiply $J_1[n-1]$ at both sides of Equation 10 and we obtain

$$J_1[n-1]\Delta\theta[n] = \Delta x_1[n] + K_1 e_1[n-1]. \tag{11}$$

Assuming that the step size is small enough, we can rewrite the above equation as

$$\Delta x_1^{des}[n] = \Delta x_1[n] + K_1 e_1[n-1]. \tag{12}$$

With the relations $e_1[n] = x_1[n] - x_1^{des}[n]$ and $e_1[n-1] = x_1[n-1] - x_1^{des}[n-1]$, Equation 12 can be rewritten into

$$e_1[n] = (I - K_1)e_1[n-1]. \tag{13}$$

Equation 13 reveals that the eigen values of $I - K_1$ should be within the interval $(-1, 1)$ to prevent the error from growing indefinitely. Even with $K_1 = I$ the stability is not guaranteed due to the nonlinearity of f_1. (Note that $J_1[n-1]\Delta\theta[n]$ is approximated as $\Delta x_1^{des}[n]$ in Equation 12.) But in practice, we found that instability rarely occurs at a usual sampling rate (30 $\sim$ 60Hz) in dealing with human motion.

If we include the secondary task $x_2 = f_2(\theta)$, the open-loop control law takes the form

$$\dot\theta = J_1^+(\dot x_1 + K_1 e_1) + (I - J_1^+ J_1)J_2^+ \dot x_2 \tag{14}$$

To prevent from possible divergence due to errors, however, a closed-loop version needs to be considered again. The CLIK scheme including the secondary task based on Jacobian transpose is given by

$$\dot\theta = J_1^+(\dot x_1 + K_1 e_1) + (I - J_1^+ J_1)J_2^T K_2 e_2 \tag{15}$$

where $e_2 = x_2 - x_2^{des}$. x_2^{des} is the actual result of the secondary task that tries to realize the given goal x_2.

It is proven that e_2 is ultimately bounded within a certain range and the tracking error for the primary task is not affected by the second term of Equation 15 [13]. But again, arbitrarily large K_2 is not allowed in discrete implementation. With the estimation of $e_2[n]$ based on old values, a reasonable choice for K_2 is I.

The final CLIK scheme in discrete domain with $K_1 = K_2 = I$ is given by

$$\begin{aligned} \Delta\theta[n] =& J_1^+[n-1](\Delta x_1[n] + \tilde e_1[n]) + \\ & (I - J_1^+[n-1]J_1[n-1])J_2^T \tilde e_2[n]. \end{aligned} \tag{16}$$

$\tilde e_1[n]$ and $\tilde e_2[n]$ in Equation 16 are the estimations of $e_1[n]$ and $e_2[n]$, respectively. Although any estimation scheme based on the old values can not completely eliminate the error caused by the nonlinearity of f_1, it can be reduced with a higher order estimation for $e_1[n]$ than $e_1[n-1]$, and consequently can give better tracking performance. We found that the estimation rule described below gives satisfactory results.

$$\begin{aligned} \text{Step 1}: \ & \Delta\tilde\theta[n] = J_1^+[n-1]\Delta x_1[n] \\ \text{Step 2}: \ & \tilde x_1^{des}[n] = f_1(\theta[n-1] + \Delta\tilde\theta[n]) \\ & \tilde x_2^{des}[n] = f_2(\theta[n-1] + \Delta\tilde\theta[n]) \\ \text{Step 3}: \ & \tilde e_1[n] = x_1[n] - \tilde x_1^{des}[n] \\ & \tilde e_2[n] = x_2[n] - \tilde x_2^{des}[n] \end{aligned}$$

The above procedures complete the discrete implementation of the CLIK algorithm with a secondary task.

3.2. Inverse Rate Control with Multiple End-effector Trajectories

In this section, we discuss how to extend inverse rate control to track multiple end-effector trajectories.

The serial chain is not suitable for modeling creatures since underlying articulated structures contain branches. An illustrative example is taken from human upper body, and is shown in Figure 2. The model consists of spine and two arms. The waist is the root of the kinematic tree structure, and the two arms are branching at the top of the spine. If both hands have their own goals to reach, and if inverse kinematics is solved for these cases separately, then the spine angles will differ in the solutions.

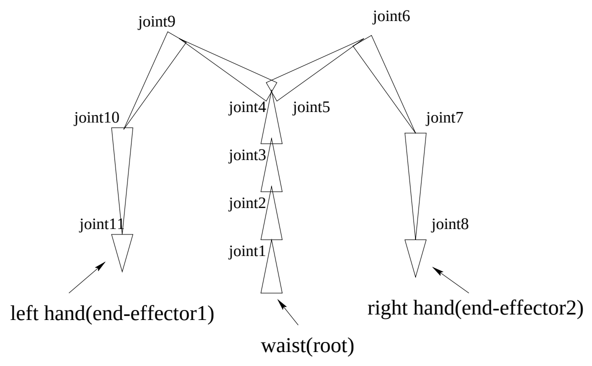

Figure 2. Kinematic structure of human upper body

In [22], Zhao and Balder solved this problem by a weighted sum of independently obtained gradients, each of which directs its corresponding end-effector to a goal position. However, the effects of different weights are not easily predictable. Depending on the weight assignment, their algorithm can fail to find an inverse kinematic solution even if all the constraints can be actually met.

Intrinsically, the problem of finding inverse kinematic solution of multiple constraints doesn't require any weight or priority assignment: if all the end-effector constraints can be met, then it should be possible without considering weights or assigning priorities to each end-effector constraint.

Compared with Zhao and Badler's algorithm, Jacobian based inverse rate control gives a quite simple and intuitive solution to the problem. The only thing we have to do in order to incorporate multiple end-effector constraints is concatenating the end-effector vectors and composing the Jacobian appropriately. In the above example, the end effector vector x_1 should be 12-dimensional vector and the Jacobian J_1 becomes 12×33 matrix (two end-effector constraints with six DOFs for each end-effector, and eleven joints with three DOFs for each joint). Of course, the Jacobian will have many zeroes where the joint angle and the end-effector have no relation such as left elbow joint and right hand. In inverse rate control, the above conflict of the spine angles is resolved during the computation of the pseudo inverse of the Jacobian.

4. Motion Retargetting with Task Priority Strategy

In general, we can formulate the motion retargetting problem with the following task set, and can solve for θ^{des}.

$$\text{primary task:} \quad x_1 = f_1(\theta^{des}) \tag{17}$$

$$\text{secondary task:} \quad x_2 = f_2(\theta^{des}) \tag{18}$$

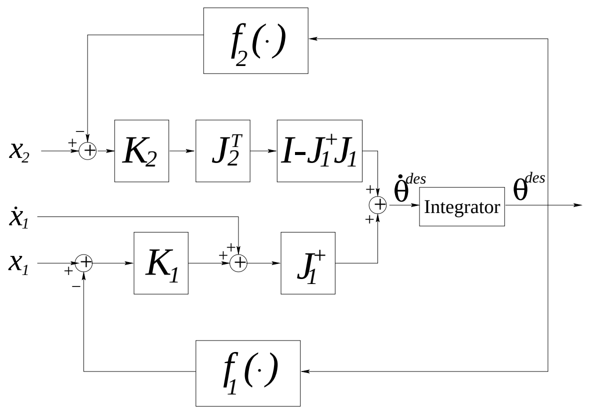

Figure 3. Closed-loop control scheme with a secondary task

x_1 in the above is the desired end-effector trajectories which can be taken from the source character (and then be modified for necessary variations) or can be provided by the user.

According to Equation 6, the open-loop control law for these tasks is given by

$$\dot{\theta}^{des} = J_1^+\dot{x}_1 + (I - J_1^+J_1)J_2^+\dot{x}_2, \tag{19}$$

and the block diagram of its closed-loop version is shown in Figure 3.

Since joint angle trajectories contain important characteristics of a motion, and since the end-effector movements are already tracked by the primary task, an obvious and useful choice for the secondary task might be to imitate the joint motion of the source character. i.e.

$$\text{secondary task: } \theta^{src} = \theta^{des}, \tag{20}$$

which is simply the case when θ^{src} and the identity function are used for x_2 and f_2, respectively, in Equation 18. Then Equation 19 becomes

$$\dot{\theta}^{des} = J_1^+\dot{x}_1 + (I - J_1^+J_1)\dot{\theta}^{src}. \tag{21}$$

The block diagram of the closed-loop control scheme with the secondary task of joint motion imitation is shown in Figure 4.

Reasonable choices for K_1 and K_2 are I's in discrete implementation as stated before. But K_1, K_2 can be adjusted based on the *dexterity measure* to get consistent motion near the kinematic singularities. A popular dexterity measure is $\sigma_{min}/\sigma_{max}$, where σ_{min} and σ_{max} are the minimum and maximum, respectively, among the singular values of the Jacobian. In this case, smaller K_1 and K_2 should be used if the dexterity measure turns out to be small.

The adaptive scheme can be also useful if we apply the OMR algorithm to motion transition. Smaller gain (e.g.

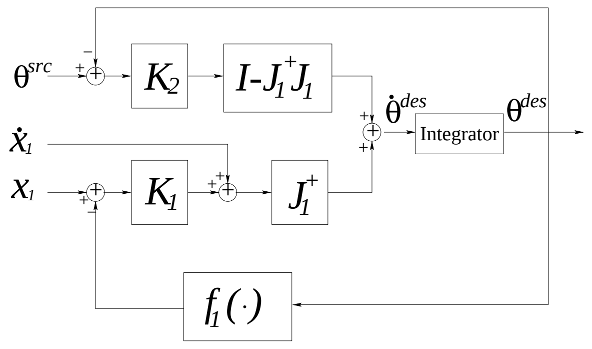

Figure 4. Closed-loop control scheme with the secondary task of joint motion imitation

$K_1 = K_2 = 0.1I$) will produce sluggish tracking, but produces smooth motion. Therefore, if the animated model switches to another motion and there exists a large discrepancy at the motion boundary, smooth transition can be obtained by adjusting the gain matrix K_1 and K_2 appropriately.

As briefly mentioned before, $K_1 = I$ does not guarantee stability since the non-linear function $\dot{\theta}$ is linearly approximated by $J_1^+ \dot{x}_1$. The system can become unstable when $\dot{x}_1$ gets very large, or the sampling rate is very low. Therefore another provision for enforcing stability might be to clamp the value that goes into the box of J_1^+ in Figure 4 whenever it is over a certain threshold. The provision might be effective when there is an excessively large acceleration, or when the model is fully stretched and almost no manipulative redundancy is left in the system. (In dealing with the human motion, however, the above provision was almost never needed.)

5. Motion Capture Data Enhancement

When we capture a motion, we often measure the joint angles and use forward kinematics to reconstruct the motion. But the method can introduce large end-effector position errors since the joint angle error near the base is amplified when it comes to the end-effector, and joint angle errors are accumulated as the forward kinematic positioning propagates toward the end-effector.

Choi et al's interpolation/regression method [5], applies inverse kinematics at sparse keyframes and the resulting joint angles are interpolated with cubic spline curves. The interpolation is combined with least square fitting so that the characteristics of the original joint angle data is preserved in the resulting motion.

The OMR algorithm described in the previous section can be used to reduce measurement errors in restoring the captured motion. The new method is an improvement over the above interpolation/regression method in three aspects: (1) inverse kinematics is done at every frame, which promises much closer end-effector tracking, (2) the joint angle imitation is done by exploiting redundant degrees of freedom rather than depending on the least square fit, and (3) the high frequency component of the original motion is preserved much better in the new method.

For the enhancement, we measure both joint angle and end-effector trajectories during the motion capture session. The measured trajectories are supplied to our motion retargetting algorithm: the end-effector trajectories are supplied for x_1, and the joint angle trajectories are supplied for θ^{src}. Of course, the destination character has to be same with the source character, if pure data enhancement needs to be done. As the retargetting progresses, θ^{src} will be adjusted to θ^{des} so that the end-effector constraint x_1 is met while maintaining the joint angle pattern of θ^{src}.

Compared to the forward kinematic motion reconstruction our OMR algorithm reduces end-effector errors remarkably. In general, our algorithm also reduces the errors in joint angle measurements. While the joint angle errors can accumulate in forward kinematic reconstruction, once it is processed by our OMR, the total amount of accumulated error is limited by the amount of end-effector position error. Moreover, the joint angle error due to the end-effector position error is distributed among all the joints. Therefore unless the amount of end-effector position error is excessively larger than that of joint angle errors, our OMR produces more accurate result than the unprocessed data.

Note that the above does not mean the retargetting and data enhancement should be done separately. If a different destination character is used, the two things are actually achieved at the same time. This is especially useful when a real-time performance is retargetted.

6. Experiments

This section describes the results of two experiments. In the first experiment, we show a retargetting example in which our OMR is applied to retarget a walking motion, to demonstrate that our OMR based on inverse rate control is not inferior in the quality to the retargetting based on spacetime constraints. Major error analysis of the algorithm is given in this example. In the second experiment, we show the retargetting of bat-swing motion.

The motion clips mentioned below are available at `http://graphics.snu.ac.kr/demo/omr/omr.mov`.

6.1. Retargetting of Walking Motion

In this experiment, the source motion (refer to the video clip #1) is a curved path walking motion which was procedurally generated by Ko's locomotion algorithm [8]. The

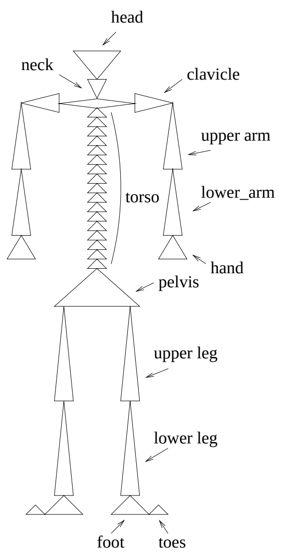

Figure 5. Kinematic model used for walking motion

walker took 13 steps and produced a total of 390 frames. The kinematic structure of the characters used for walking motion is shown in Figure 5.

Since the lower body motion is far more important than the upper body motion in walking example, we retargetted only the lower body motion. As shown in Figure 5 the lower body consists of pelvis, upper leg, lower leg, foot, and toes, and they are connected at the hip, knee, ankle, and ball joints. The total degree of freedom of the lower body is $8 \times 3 + 6 = 30$. (All the joints were modeled by 3-DOF joints, and the base has extra 6 DOFs.)

The destination character was about 60% scaled down from the source character with non-uniform proportions. They are shown in Figure 6, and the lower body dimensions are compared in Table 1.

	src character	des character	ratio
pelvis (width)	30.0	30.0	1.00
upper leg	46.0	26.0	0.57
lower leg	46.0	34.0	0.74
foot	16.0	16.0	1.00
toes	8.0	8.0	1.00

Table 1. Comparison of lower body dimensions

In the retargetting, the secondary task was set to $\theta^{src} = \theta^{des}$. To specify the primary task, we set the toe-tip of the stance leg as the base and the toe-tip of the swing leg as

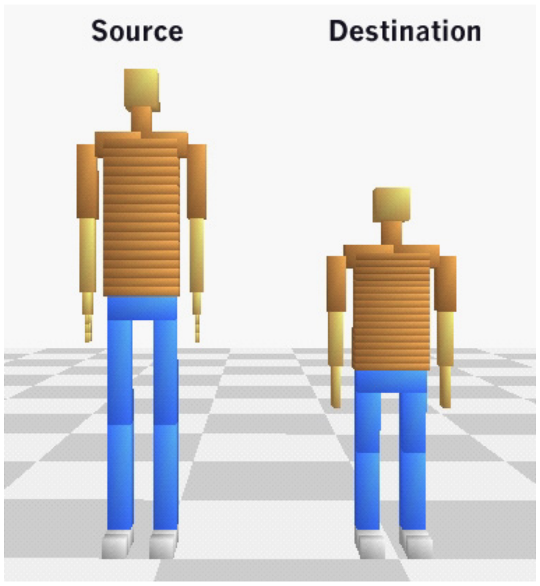

Figure 6. Characters used for walking motion (left: source character, right: destination character)

the end-effector. The source character's toe-tip trajectory was used for x_1 without any modification. Therefore the destination character had to take relatively bigger steps considering his body size. At the boundaries of steps the base and end-effector were switched. It implies that there can be discontinuities at the boundary if the tracking error is large. The retargetted motion with the above task set is shown in the video clip #2. The tracking error of the swing foot was negligible and thus the produced motion was smooth at the step boundaries.

But the pelvis motion showed non-uniform speed along the direction of progression (anterior-posterior), which wasn't observable in the source motion. So we constrained the transverse plane motion of the pelvis. i.e. the pelvis was designated as another end-effector, and the (x, z) component of the source character's pelvis movement was tracked in the destination motion. (Note that the pelvis motion along y-axis should be adapted to account for the height difference). After adding the constraint, we could obtain a satisfactory result as shown in the video clips #3 and #4. Even with the extra constraints, the end-effector trajectories of the source and the destination made an accurate match. The comparison is shown in Figure 7. The dotted curves for the source motion are not visible because they overlap exactly with the solid curves, the end-effector trajectories of the destination motion.

To show the tracking error microscopically, the area indicated with a small box near the 150th frame in Figure 7

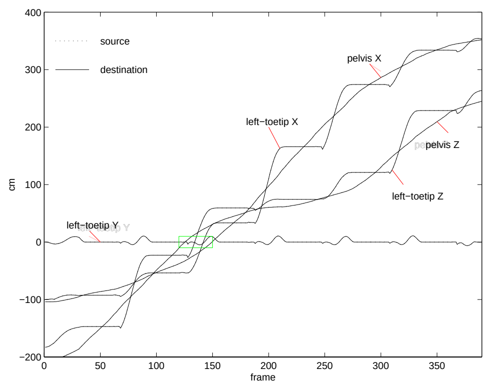

Figure 7. Comparison of end-effector trajectories

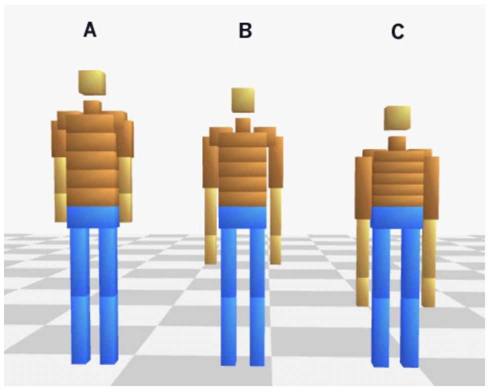

Figure 9. Three characters used for retargetting the bat swing motion

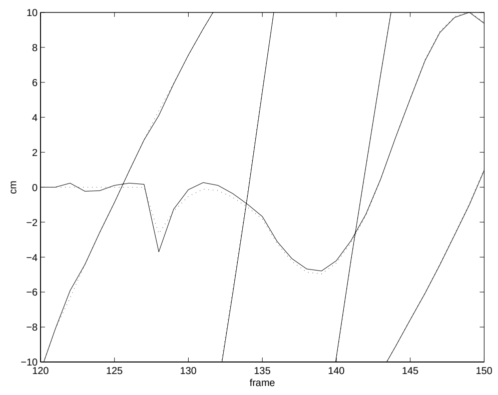

Figure 8. Comparison of end-effector trajectories (a magnified view)

was magnified in Figure 8. The trajectories in the figure show that the tracking error is kept small where the velocity is nearly constant, but the error increases when the velocity makes sudden changes. The maximum error (1.0464 cm) occurred at the 128th frame where the y-coordinate (height) of the toe-tip reached its peak acceleration and this error was reduced to a negligible level at around the 135th frame as the acceleration decreased. The step boundary was taken from low-acceleration points so that the base to end-effector switch makes a smooth transition.

The joint angle trajectories of the left leg during the original and retargetted motion are plotted in Figure 10. Only the angles around the sideways direction (medial-lateral) axes are presented in the graphs. The comparison shows that the amplitude of the hip angle is increased in the destination motion to cover the given step length with the relatively smaller body. Other than that the original joint angle pattern was quite well preserved.[3]

At the end of every step, the ball joint of the source character showed an abrupt change from a large negative value to zero. It corresponds to the toe-off moment when the toes take off the ground. After the retargetting, the sharp corner of the trajectory was well preserved. In general, our OMR preserves the high frequency content of the motion quite well, since inverse rate control is directed by Jacobian values. Big mountains or valleys are never missed. To recover tiny fluctuations as well, however, a high sampling rate is needed to avoid aliasing.

If the sharp corners are undesirable, they can be prevented by adjusting the gain matrix K_2 or clamping some of the control input as stated in Section 4. The adjustment of K_2 does not affect the end-effector tracking performance.

6.2. Retargetting of Bat Swing Motion

In this experiment, actual performance of a bat swing motion was processed by our OMR to produce the destination motion of three different characters shown in Figure 9. The anthropometry of Character B is about the average. Character A has a longer torso but shorter limbs than average, and Character C has a shorter torso but longer limbs. Their kinematic structures are same as Figure 5 except that the torso is segmented to 5 parts and the feet are excluded.

To set the primary task, the base and end-effectors should be specified as before. In this experiment, the pelvis was chosen as the base and two hands were chosen as the end-effectors. Three 6-DOF sensors were used to capture those positions and orientations. The end-effector motion was directly supplied for $x_1(t)$ without any modification. Therefore Character A, for example, had to make a relatively

[3]Note that zero error in tracking the joint angle trajectories is unachievable due to the anthropometric difference.

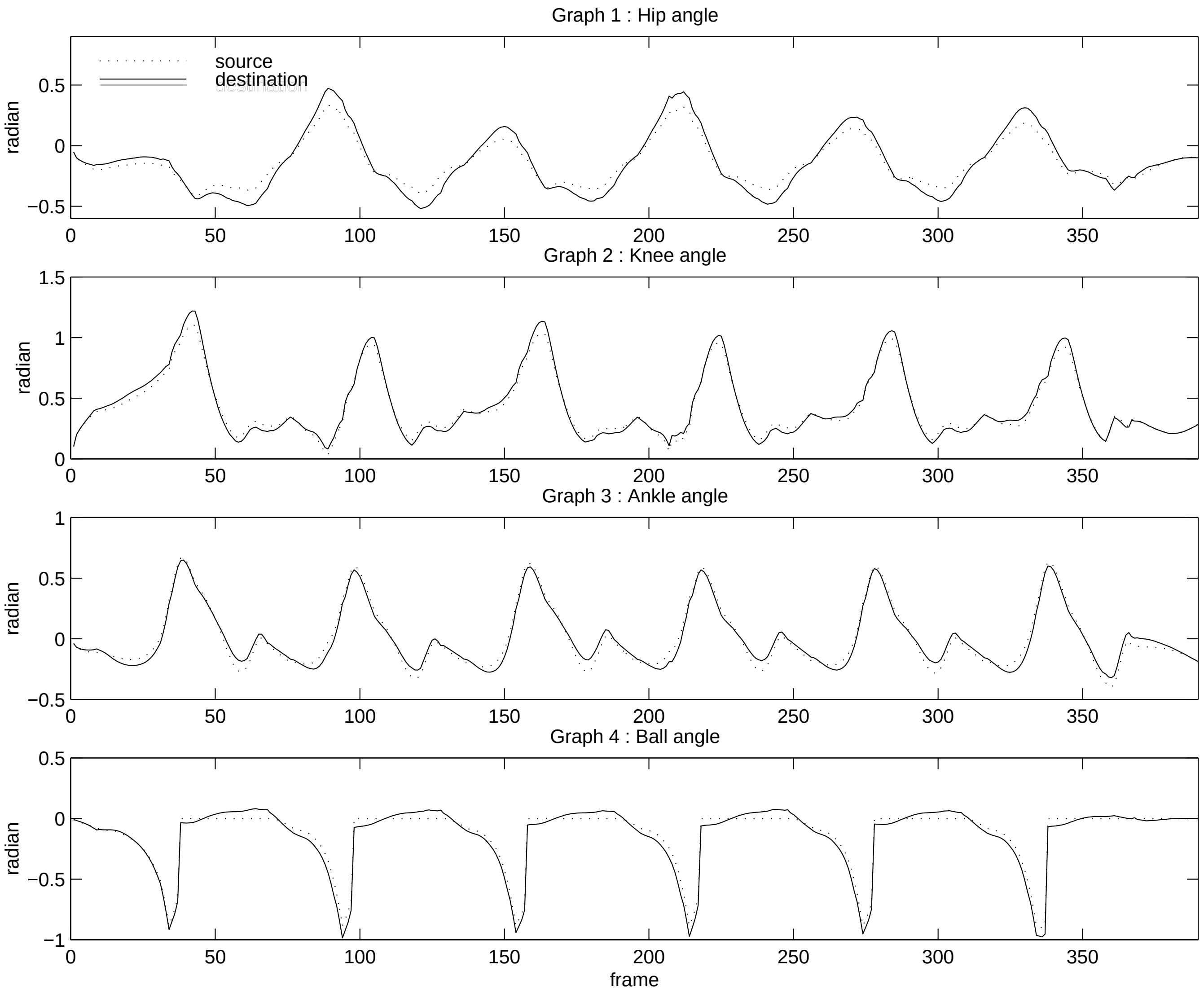

Figure 10. Comparison of joint angle trajectories

lower swing.

Since all the torso segments can not be measured due to the limited number of sensors, we measured only the top-most segment. Therefore, the five joints from the waist to the top-most torso segment had to be generated from the orientation difference between the pelvis and the top-most torso segment. For this, the measured orientation of the top-most torso segment was added to the primary task, and zero angles for the unmeasured joints were added to the secondary task, expecting the primary task can be met with minimal joint angles along the torso.

The other sensors were used to measure the joint angles. (Because we had only 13 sensors available, we had to give up capturing the foot motion.) In this experiment only the upper body motion was adapted by OMR. The lower body motion was reconstructed by applying the measured joint angles directly.

The retargetting of the source motion to Characters A, B, C are shown in the video clips #5˜6, #7˜8, and #9˜10, respectively. The small green boxes in the video indicate the position of the end-effectors and base. In the video we can observe that end-effector positions are accurately tracked.

Since the body dimensions of Character B and the real performer are similar, the retargetted motion does not contain any noticeable difference from the source motion. In the case of Character A, however, we can see the waist is bent to lower the hit position, and the torso is shifted forward to account for the shorter arms. In the case of Character C, the torso is bent backward and makes a bigger twist to account for the longer arms and shorter torso. Snap shots

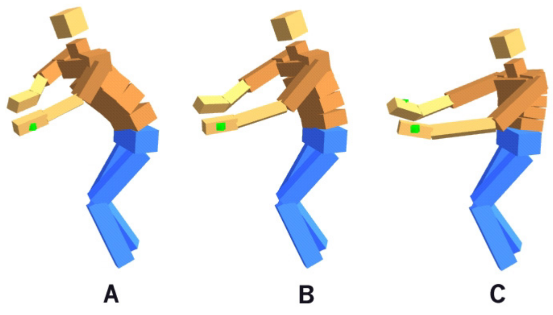

Figure 11. Snap shots taken from the retargetted motion. Observe different adaptations to compensate the anthropometric differences

	walking	bat swing
number of frames	390	136
total DOFs	30	42
constraint DOFs	8	9
elapsed time (sec)	1.219	0.984
frame rate* (Hz)	300.7	138.2

Table 2. Computational time spent for retargetting (*The frame rate does not include the time for visualization)

were taken during the retargetted motions to clearly demonstrate the above adaptation for the anthropometric differences and shown in Figure 11.

6.3. Computational time

Since we had no privilege to install our program to the platform equipped with a motion capture system, we had to emulate the real-time motion capture. That is, the motion data captured at 30Hz was fed to the OMR system with the same sampling rate using a timer. At this sampling rate, not a single frame was lost even with the visualization included. We used an Intergraph GX1 system (dual P-III 550, wildcat 4000) for the experiments. Table 2 describes the computational time purely spent for the retargetting procedure in each motion. The code was not fully optimized and so the performance can be potentially improved further.

The slower rate of the bat swing motion is due to the bigger size of the Jacobian matrix compared to the walking motion (8×30 vs. 9×42). As shown in the table, the OMR is fast enough to process motion capture data collected at a usual sampling rate (30$\sim$90Hz) in real-time for the models of reasonable complexity.

7. Discussion & Conclusion

This paper presented the on-line motion retargetting technique based on inverse rate control. The method is an improvement over the off-line retargetting based on spacetime constraints since real-time performances can be retargetted without degradation of retargetting quality. The OMR technique greatly helps to get more satisfactory results in motion capturing with fewer trials by giving the real-time feedback to the performer. Furthermore, the captured data are enhanced in both end-effector positions and joint angles by going through our OMR filter.

One minor unsolved problem is that there is no easy way to guarantee full-proof stability of the system due to the non-linearity. We observed that at a very low sampling rate, and if the model goes near the kinematic singularity and thus very little manipulative redundancy is left, then the system can became unstable. However, experiments proved that the system never become unstable at 30Hz or higher sampling rate. If the source motion is available only at a low sampling rate, two remedies are recommended: (1) by interpolating the source motion curves, first produce more samples, and then use them as the input to the OMR filter, or (2) scale down the end-effector trajectory to avoid the singular configuration, or use both of (1) and (2).

The above remedies are for an excessively bad situation. Our on-line motion retargetting produces satisfactory results in retargetting most human or creature motion. If the technique is well utilized, it can be very useful to people in character animation and game industries.

Acknowledgment

This work was supported by Creative Research Initiatives of the Korean Ministry of Science and Technology. This work was partially supported by ASRI (Automation and Systems Research Institution), Seoul National University.

References

[1] G. D. M. A. Balestrino and L. Sciavicco. Robust control of robotic manipulators. In *Preprints of the 9th IFAC World Congress*, volume 6, pages 80–85, 1984.

[2] R. Bindiganavale and N. I. Badler. Motion abstraction and mapping with spatial constraints. In *Modelling and Motion Capture Techniques for Virtual Environments, International Workshop, CAPTECH'98*, pages 70–82, Nov. 1998.

[3] R. Boulic and D. Thalmann. Combined direct and inverse kinematic control for articulated figure motion editing. *Computer Graphics Forum*, 11(4):189–202, 1992.

[4] A. Bruderlin and L. Williams. Motion signal processing. In R. Cook, editor, *Computer Graphics (SIGGRAPH '95 Pro-*

ceedings), pages 97–104, August 1995. ACM-0-89791-701-4.

[5] K. Choi, S. Park, and H. Ko. Processing motion capture data to achieve positional accuracy. Submitted to *Graphical Models and Image Processing* for review.

[6] M. Gleicher. Retargeting motion to new characters. In *SIGGRAPH 98 Conference Proceedings*, Annual Conference Series, pages 33–42. ACM SIGGRAPH, Addison Wesley, July 1998. ISBN 0-89791-999-8.

[7] J. M. Hollerbach and K. C. Suh. Redundancy resolution of manipulators through torque optimization. *IEEE J. Robotics Automat.*, RA-3(4):308–316, Aug. 1987.

[8] H. Ko. *Kinematic and Dynamic Techniques for Analyzing, Predicting, and Animating Human Locomotion*. PhD thesis, University of Pennsylvania, Department of Computer and Information Science, Philadelphia, PA 19104-6389, May 1994. MS-CIS-94-31.

[9] A. Liegeois. Automatic supervisory control of the configuration and behavior of multibody mechanisms. *IEEE Trans. Syst., Man., Cybern.*, SMC-12(12):868–871, Dec. 1977.

[10] A. A. Maciejewski and C. A. Klein. Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments. *Int. J. Robotics Res.*, 4(3):109–117, 1985.

[11] K. A. Munetoshi Unuma and R. Takeuchi. Fourier principles for emotion-based human figure animation. In R. Cook, editor, *Computer Graphics (SIGGRAPH '95 Proceedings)*, pages 91–96, August 1995. ACM-0-89791-701-4.

[12] Y. Nakamura and H. Hanafusa. Inverse kinematic solutions with singularity robustness for robot manipulator control. *ASME J. Dynam. Sys. Measurement Control*, 108(3):163–171, 1986.

[13] L. S. P. Chiacchio, S. Chiaverini and B.Siciliano. Closed-loop inverse kinematics schemes for constrained redundant manipulators with task space augmentation and task priority strategy. *International Journal of Robotics Research*, 10(4):410–425, 1991.

[14] L. Sciavicco and B. Siciliano. A dynamic solution to the inverse kinematic problem for redundant manipulators. In *Proceedings of the 1987 IEEE International Conference on Robotics and Automation*, pages 1081–1087. IEEE Computer Society Press, 1987.

[15] Y. T. Tsai and D. E. Orin. A strictly convergent real-time solution for inverse kinematics of robot manipulators. *J. Robot. Sys.*, 4(4):477–501, 1987.

[16] C. W. Wampler. Manipulator inverse kinematic solutions based on damped least-squares solutions. *IEEE Trans. Sys. Man Cybernet.*, SMC-16(1):93–101, 1986.

[17] D. E. Whitney. Resolved motion rate control of manipulators and human prostheses. *IEEE Trans. Man-Machine Sys.*, MMS-10:47–53, 1969.

[18] D. J. Wiley and J. K. Hahn. Interpolation synthesis of articulated figure motion. *IEEE Computer Graphics and Applications*, pages 39–45, November/December 1997.

[19] A. Witkin and Z. Popovic. Motion warping. In R. Cook, editor, *Computer Graphics (SIGGRAPH '95 Proceedings)*, pages 105–108, August 1995. ACM-089791-701-4.

[20] H. H. Y. Nakamura and T. Yoshikawa. Task-priority based redundancy control of robot manipulators. *Int. J. Robotics Res.*, 6(2):3–15, 1987.

[21] T. Yoshikawa. Analysis and control of robot manipulators with redundancy. In M. Brady and E. R. Paul, editors, *Robotics Research: The First International Symposium*, pages 735–747. MIT Press, 1984.

[22] J. Zhao and N. I. Badler. Inverse kinematics positioning using nonlinear programming for highly articulated figures. *ACM Transactions on Graphics*, 13(4):313–336, October 1994.