Processing Motion Capture Data to Achieve Positional Accuracy

Kwangjin Choi, Sanghyun Park, Hyeongseok Ko
{kjchoi,shpark,ko}@graphics.snu.ac.kr
Human Animation Center
School of Electrical Engineering
Seoul National University

short running head: Processing Motion Capture Data

contact name: Hyeongseok Ko
contact E-mail: ko@graphics.snu.ac.kr
contact phone: (+82) 2–880–1780
contact fax: (+82) 2–882–4656

Abstract

In animating an articulated entity with motion capture data, if the reconstruction is based on forward kinematics, there could be a large error in the end-effector position. The inaccuracy becomes conspicuous when the entity makes interactions with the environment or other entities. The frames at which the end-effector position needs to be accurate are designated as "keyframes" (e.g. the impact moment in a punch). We present an algorithm that processes the original joint angle data to produce a new motion in which the end-effector error is reduced to zero at keyframes. The new motion shouldn't be too much different from the original motion. We formulated the problem as a constrained minimization problem so that the characteristics of the original joint angle data is optimally preserved during the enhancement steps. The algorithm was applied to several examples such as boxing, kicking, and catching motions. Experiments prove that our algorithm is a valuable tool to improve captured motion especially when the end-effector trajectory contains a special goal.

List of Symbols

All the symbols used in this paper are produced by LaTeX. In the following, we list the first instances of symbols in the sections where they appear. We do not list the second and the later instances. The right side of ":" in each row shows the LaTeX representation of the symbol.

1. Introduction

J : J
E : E
Λ : \Lambda
$J^{'}$: J∧{'}
C^2 : C∧{2}
C^1 : C∧{1}

2. Related Work

i : i

3. Overview

$(J_1, \ldots, J_M)$: (J_{1},\ldots,J_{M})
M : M
$\subset$: \subset
$\{0, \ldots, N\}$: \{0,\ldots,N\}
$J(i)$: J(i)
$E(i)$: E(i)
$(J_1(i), \ldots, J_M(i))$: (J_{1}(i),\ldots,J_{M}(i))
X : X
$J(X)$: J(X)
$\{J(i)|i \in X\}$: \{J(i)|i \in X\}
$E(X)$: E(X)
$\{E(i)|i \in X\}$: \{E(i)|i \in X\}
ϕ : \phi
ϕ^{-1} : \phi∧{-1}
$\phi(J)$: \phi(J)
$\phi^{-1}(E)$: \phi∧{-1}(E)
$(\phi_1^{-1}(E), \ldots, \phi_M^{-1}(E))$:
(\phi_{1}∧{-1}(E),\ldots, \phi_{M}∧{-1}(E))
$(J_1^{'}, \ldots, J_M^{'})$: (J∧{'}_{1}, \ldots, J∧{'}_{M})
$E(\Lambda)$: E(\Lambda)
$\phi(J^{'})$: \phi(J∧{'})
$\cap$: \cap
$J_j^{'}$: J_{j}∧{'}
J_j : J_{j}
$\approx$: \approx
$\phi^{-1}(E(\Lambda))$: \phi∧{-1}(E(\Lambda))
$\phi_j^{-1}(E(\Lambda))$: \phi_{j}∧{-1}(E(\Lambda))

4. Inverse Kinematics

$G(\theta_1, \theta_2, \ldots, \theta_M)$:
G(\theta_{1},\theta_{2},\ldots,\theta_{M})
w : w
$\|\vec{e} - \vec{e_m}\|^2$:
\| \vec{e} - \vec{e_{m}} \ |∧{2}
$\sum_{j=1}^{M}$: \sum_{j=1}∧{M}
$(\theta_i - \theta_i^{'})^2$: (\theta_{i}-\theta_{i}∧{'})∧{2}
$(\theta_1, \theta_2, \ldots, \theta_M)$:
(\theta_{1},\theta_{2},\ldots,\theta_{M})
$\vec{e_m}$: \vec{e_{m}}
$\theta_j^{'}$: \theta_{j}∧{'}

5. Finding $J^{'}$

$J_i^{'}$: J_{i}∧{'}

5.1. Cubic Spline with C^1 Continuity

C^0 : C∧{0}

5.2. Auxiliary Frames

$\vec{P}$: \vec{P}
$\vec{P'}$: \vec{P}'

5.3. Cubic Spline Regression

t_k : t_{k}
t_{k+1} : t_{k+1}
$\vec{P}_k(t)$: \vec{P}_{k}(t)
$\leq$: \leq
t : t
$<$: <
$\vec{P}_k$: \vec{P}_{k}
$\vec{P}_{k+1}$: \vec{P}_{k+1}
$\vec{P}'_k$: \vec{P}'_{k}
$\vec{P}'_{k+1}$: \vec{P}'_{k+1}
$f_{k,1}(t)$: f_{k,1}(t)
$f_{k,2}(t)$: f_{k,2}(t)
$f_{k,3}(t)$: f_{k,3}(t)
$f_{k,4}(t)$: f_{k,4}(t)
$\vec{F}_k$: \vec{F}_{k}
$\vec{V}_k$: \vec{V}_{k}
$f_{k,a}(t)$: f_{k,a}(t)
$f_{k,b}(t)$: f_{k,b}(t)
$f_{k,c}(t)$: f_{k,c}(t)
$f_{k,d}(t)$: f_{k,d}(t)
$\vec{V}_{k+1}$: \vec{V}_{k+1}
$\vec{f}_{k,fixed}(t)$: \vec{f}_{k,fixed}(t)
$\vec{P}(t)$: \vec{P}(t)
$\sum_{k=1}^{L-1}$: \sum_{k=1}∧{L-1}
t_1 : t_{1}
t_L : t_{L}
$f_1(t)$: f_{1}(t)
$f_2(t)$: f_{2}(t)
$f_L(t)$: f_{L}(t)
$\vec{V}_1$: \vec{V}_{1}
$\vec{V}_2$: \vec{V}_{2}
$\vec{V}_L$: \vec{V}_{L}
$\vec{f}_{fixed}(t)$: \vec{f}_{fixed}(t)
$f_{1,a}(t)$: f_{1,a}(t)
$f_k(t)$: f_{k}(t)
$f_{k-1,b}(t)$: f_{k-1,b}(t)
$f_{L-1,b}(t)$: f_{L-1,b}(t)
$\sum_{i=0}^{N}$: \sum_{k=1}∧{L}
$\dot{\vec{P}}[i]$: \dot{\vec{P}}[i]
$\dot{J}_j(i)$ \dot{J}_{j}(i)
X^T : X∧{T}

X : X
Y : Y
$\dot{f}_1[0]$: \dot{f}_{1}[0]
$\dot{f}_2[1]$: \dot{f}_{2}[1]
$\dot{f}_L[N]$: \dot{f}_{L}[N]
$\dot{f}_1[0]$: \dot{f}_{1}[0]
$\dot{f}_2[1]$: \dot{f}_{2}[1]
$\dot{f}_L[N]$: \dot{f}_{L}[N]
$\dot{f}_1[0]$: \dot{f}_{1}[0]
$\dot{f}_2[1]$: \dot{f}_{2}[1]
$\dot{f}_L[N]$: \dot{f}_{L}[N]
$\dot{J}_j[0]$: \dot{J}_{j}[0]
$\dot{J}_j[1]$: \dot{J}_{j}[1]
$\dot{J}_j[N]$: \dot{J}_{j}[N]
$\dot{\vec{f}}_{fixed}[0]$: \dot{\vec{f}}_{fixed}[0]
$\dot{\vec{f}}_{fixed}[1]$: \dot{\vec{f}}_{fixed}[1]
$\dot{\vec{f}}_{fixed}[N]$: \dot{\vec{f}}_{fixed}[N]
$f[\cdot]$: f[\cdot]
$f(\cdot)$: f(\cdot)
$\dot{f}[i]$: \dot{f}[i]
$\dot{\vec{f}}_{fixed}[i]$: \dot{\vec{f}}_{fixed}[i]
$\dot{f}$: \dot{f}
$\dot{\vec{f}}_{fixed}$: \dot{\vec{f}}_{fixed}

6. Experiments

None

7. Conclusion

None

1 Introduction

Animating an articulated entity requires the technique to control highly redundant degrees of freedom [15, 1, 14, 10, 3, 21]. To obtain realistic motion of complex character during a short period of time, copying is more effective than computational synthesis. Therefore, motion capture is recently emerging as a powerful technique for character animation.

However, the motion capture technique by itself has poor generality and automatism. There is no general and automatic procedure to process the captured data to obtain desired motion. Therefore it is best utilized in the applications in which highly realistic motion is required but the motion is never used again, as in movies. The ideas such as motion composition, warping [20], blending [17], motion signal processing [6], motion editing with spacetime constraints [7], motion retargetting [8], and motion mapping [4] contributed in overcoming the above limitations.

Another fundamental problem of motion capturing is that the measurement errors greatly demote the value of captured data. Without an elaborate manual processing of the data the resulting animation is often inaccurate and unrealistic. This paper presents an algorithm that processes the motion capture data to produce another data set that promises more accurate motion.

Motion of an articulated entity can be reconstructed by positioning the *base* and bending the joints according to the captured data. The base is a coordinated system embedded in a segment of the articulated entity to locate it at a desired global position and orientation. The above is called *forward kinematic reconstruction*.

Whether they are directly measured or indirectly collected from positional data, joint angles contain important features of a motion, and the forward kinematic reconstruction is quite effective in restoring such features. If we reconstruct the motion from joint angle measurements only, however, the end-effector trajectory can be inaccurate due to the following reasons:

- **inaccurate joint angles:** There are several sources of errors during the joint angle measurement. The joint angle errors coming from the muscle and skin deformation or simplified joint model can be as large as a few tens of degrees. For example, the twisting component of shoulder is often measured only half of the actual bone rotation [11]. Joint angle errors at or near the base are amplified when the end-effector position is considered, and such errors are accumulated as the computation propagates to the end-effector.

- **inaccurate link lengths:** Before the reconstruction starts, first we have to build an articulated model. In the model, the link lengths should come from the direct measurement or from the data analysis [5]. In either case, the estimated link lengths contain errors, and they eventually contribute to end-effector position error.

The inaccuracy in end-effector position becomes conspicuous when the entity makes interactions with the environment or other entities. We can classify the mismatch into the following three categories:

- **intra-person:** e.g. in applauding

- **person-to-object:** e.g. in reaching the door knob

- **person-to-person:** e.g. in boxing (fist-to-fist, or fist-to-body)

Some people might think the above mismatch can be fixed by a simple method. For example, in boxing, the opponent might be translated back and forth to establish the contacts. However, it will create even more unnatural artifacts: the body will make irregular skids on the floor.

We propose a noble algorithm that processes the global part of joint angle data so that the desired end-effector goals are satisfied, while preserving the original motion pattern. The algorithm is particularly useful for the motion in which the end-effector has several positional goals.

A method for reducing the discrepancy is to apply inverse kinematics at every frame. Since inverse kinematics picks one among the multiple solutions, when the the computation is done at each frame independently, the consecutive frames may lack coherence. But even more serious problem of this method is that the measurements of joint angles, which carry most important content of the original motion, are not untilized at all.

Our approach is about halfway between the forward and inverse kinematic reconstructions. It processes the original data so that the new end-effector trajectory is correct at several keyframes where the end-effector position has to be correct, while preserving the pattern of original joint angle trajectories.

Input to the system is two kinds of measurements, joint angle data J and end-effector trajectory E, and a set Λ of *keyframes* at which the end-effector position must be correct. The output is a modified version J' of J such that the end-effector position is correct at the keyframes while changes in overall motion pattern are not noticeable.

To implement the above, we first apply inverse kinematics at the keyframes. Then the joint angles at keyframes are interplated using piecewise cubic spline curves. While the original cubic spline imposes C^2 continuity at the segment boundaries, we impose only C^1-continuity. The weaker condition leaves us several free variables for curve shape control, which are exploited to preserve the velocity pattern of the joint angle trajectories.

The paper is organized as follows: Section 2 reviews previous work on motion capture and related topics. Section 3 gives an overview of the paper. Section 4 explains how conventional inverse kinematics is modified to fit for our purpose. Section 5 presents the procedure to find $J^{'}$. Section 6 shows the results in several examples. Finally, Section 7 concludes the paper.

2 Related Work

Methods for using motion capture data can be classified based on how the data is used. In one extreme, the whole duration of performance is reproduced by inverse kinematics [2, 18]. In [2], Badler *et al.* proposed a method to reconstruct standing postures of an articulated human figure from four 6DOF sensors by employing real-time inverse kinematics. The limit in the number of sensors restricted the reconstruction to specific (e.g. standing) postures.

Semwal *et al.* [18] developed a full-body motion capture technique that uses analytic, constant-time methods to solve inverse kinematics problem. Since the data from eight sensors was not enough to uniquely determine the posture, they utilized experimental knowledge about natural body postures. Due to the analytic nature of the computation, however, the calibration process was somewhat tricky; each sensor had to be placed at a specific position and orientation.

In the other extreme, the model is driven by pure forward kinematics. Molet *et al.* [11, 12] proposed a method to convert the sensor data to anatomical rotations. The method didn't consider end-effector constraints, but it simply analyzed the orientational displacements between adjacent segments to find the joint angles. Therefore the calibration process was simple. Since all the joint angles were captured, the pattern of original motion was quite well preserved. As pointed out earlier, however, the method could produce noticeable position errors at the end-effector.

One thing to note in Molet *et al.'s* work is that our method for extracting joint angles is essentially the same as theirs. The placement of sensor is not aligned with the reference frame embedded in each body segment. Therefore the joint angle can not be obtained by taking the simple orientation difference between the adjacent sensors, but the measurements should be calibrated to consider the offsets from the reference frame. To estimate the offsets, the subject was asked to take upright position so that the reference frames are aligned with the global frame, and then the orientation of each sensor was taken for the offset.

There is yet another approach that combines the two ones above [5, 9]. Bodenheimer *et al.* [5] proposed a method that utilizes both orientational and positional data of each joint. The two steps of the method, statistical estimation of skeletal size and inverse kinematics optimization, are relevant to our work. The statistical estimation produces an articulated model based on the measurements. In inverse kinematic optimization, the differences between the recorded data and reconstructed animation at each joint is minimized for both position and orientation. Our method in this paper also uses similar inverse kinematics (see Section 4). But two major differences can be pointed out:

- **the position error term in the objective function:** We minimize the position errors of the end-effectors only, while Bodenheimer *et al.* minimize the errors at all joints, which requires more effort in translational sensor calibration.
- **the amount of inverse kinematics computation:** We solve inverse kinematics only at sparse keyframes, while they do it for every frame.

A problem in obtaining every frame with inverse kinematic reconstruction is that joint angles can make abrupt changes since the procedure picks a solution among multiple choices. To take care of this problem they suggested two methods, both of which are related to the selection of initial guesses. The first one is to use the result of frame i for the initial guess of frame $i+1$. As they pointed out, this method can propagate the errors frame to frame. The other one is to group motions and use a similar motion as the initial guess so that it does not propagate errors. Although their method was practically applicable to many motions, there is still no guarantee of smooth motion due to local minima. In our method, the problem is not visible since (1) inverse kinematics was applied at sparse keyframes, and (2) those results were interpolated with a smooth curve.

Hirose *et al.'s* [9] spring method also used both positional and orientational data in posture reconstruction. The method was developed to avoid the shortcomings of inverse kinematics such as local minima and unnaturalness. Soft springs were used to minimize the positional and orientational discrepancies between the measurement and reconstructed animation, and the stiff springs were used to avoid dislocations between the segments. A problem that is easily predictable is that the model can often be disarticulated even when the springs reach a mechanical equilibrium. They went around this problem by introducing *discomfort potential.* But the cure is for the static postures. Therefore there is still no promise that the resulting animation doesn't make abrupt joint angle changes, especially in fast motions.

3 Overview

In this paper we propose a new method to enhance the joint angle data utilizing the measurements of end-effector positions. Input to the system are

- **joint angle data** $J = (J_1, \ldots, J_M)$ (measured). J_j is the angle trajectory at joint j, and M is the number of single-DOF joints in the model. If a joint has 3 DOFs, for example, then we can decompose it to three single-DOF joints by using Euler-angle scheme.
- **end-effector trajectory** E (measured), and
- **a set of keyframes** $\Lambda \subset \{0, \ldots, N\}$ (designated by the animator).

Let $J(i)$ and $E(i)$ denote the joint angle vector $(J_1(i), \ldots, J_M(i))$ and end-effector position, respectively, at frame i. For $X \subset \Lambda$, let $J(X) = \{J(i)|i \in X\}$, and $E(X) = \{E(i)|i \in X\}$. Let ϕ and ϕ^{-1} be the operators representing forward kinematic reconstruction and inverse kinematic reconstruction, respectively. Thus $\phi(J)$ is the end-effector trajectory corresponding to the joint angle data J, and $\phi^{-1}(E) = (\phi_1^{-1}(E), \ldots, \phi_M^{-1}(E))$ is the joint angle data obtained by performing inverse kinematics at each point in E.

The output of our algorithm is a new joint angle data $J^{'} = (J_1^{'}, \ldots, J_M^{'})$, which is the enhanced version of J. $J^{'}$ have to satisfy the following two conditions:

Condition 1. interpolation at keyframes: the end-effector trajectory of the new motion should coincide with E at keyframes.

Condition 2. perservation of joint angle pattern: $J_j^{'}$ should resemble the pattern of J_j for $j = 1, \ldots, M$. We denote the condition by $J_j^{'} \approx J_j$, $j = 1, \ldots, M$, or collectively by $J^{'} \approx J$.

We achieve Condition 1 above by computing inverse kinematics at keyframes (i.e. by computing $\phi^{-1}(E(\Lambda))$), and then connecting them with a smooth curve. Figure 1 illustrates the situation at one typical joint. The dotted points in the graph represent the angle data J_j at joint j. The points marked with X are $\phi_j^{-1}(E(\Lambda))$. We form a piecewise cubic spline that passes through those interpolation points for each $j = 1, \ldots, M$.

The next job is to manipulate the above curves to achieve $J^{'} \approx J$. Now, the question is how to achieve the condition. We claim that the *angular velocity* carries the most important motion characteristics, and therefore the velocity difference should be minimized between $J_j^{'}$ and J_j for each j.

Our enhancement algorithm can be summarized into two major steps shown in Figure 2. The first step, which is further explained in Section 4, is to solve inverse kinematics at the keyframes to obtain $\phi^{-1}(E(\Lambda))$. The second step, which is further explained in Section 5, is to generate the curve that interpolates $\phi^{-1}(E(\Lambda))$ and minimizes the velocity differences.

4 Inverse Kinematics

In our algorithm, inverse kinematics is performed to reconstruct the static postures which satisfy the end-effector constraints at keyframes. Since human body model is highly redundant, there are infinitely many solutions that satisfy the given end-effector constraints. By exploiting the redundancy we can minimize the joint angle modification from the captured posture. The above goal can be achieved by minimizing the following objective function, which consists of two error terms: the end-effector position error and the sum of joint angle errors.

$$\begin{aligned} G(\theta_1, \theta_2, \ldots, \theta_M) &= w \cdot \|\vec{e} - \vec{e_m}\|^2 \\ &+ (1-w) \cdot \sum_{j=1}^{M} (\theta_j - \theta_j^{'})^2, \end{aligned} \quad (1)$$

where $\vec{e}$ is a function of the joint angles that gives the end-effector position at the current joint angles $(\theta_1, \theta_2, \ldots, \theta_M)$, $\vec{e_m}$ is the measured end-effector position, and $\theta_j^{'} (j = 1, \ldots, M)$ are measured joint angles.

Our inverse kinematics is a generalized version of Zhao and Badler's inverse kinematics [21], which does not have the second term of Eq.(1) in their objective function. (We did not consider the case when the end-effectors have multiple goals.) w in Eq.(1) is a free variable we can control. With $w = 1$, minimizing the equation becomes conventional inverse kinematics. With $w = 0$, Eq.(1) becomes forward kinematics. By having an intermediate value for w, we can transfer some of the end-effector reaching effort to the task of maintaining the measured joint angle pattern. Experimentally, we found that with $w = 0.99$ end-effector error was negligible and the original posture was well preserved.

We used the captured joint angles as the initial guess, and quasi-Newton BFGS method was used for the non-linear optimization.

5 Finding $J^{'}$

In this section, we show how to find the new joint angle curves $J_j^{'}$ $(j = 1, \ldots, M)$ that interpolate the keyframe points $\phi^{-1}(E(\Lambda))$. The curves also have to satisfy Condition 2 ($J^{'} \approx J$), for which we minimize the angular velocity difference between $J^{'}$ and J.

5.1 Cubic Spline with C^1 Continuity

A cubic spline can be represented by four parameters, the positions and velocities at the two end points. Several cubic splines can be concatenated to form a *piecewise cubic spline* [16]. When a sequence of points are given, by imposing a few constrains on the boundary velocities, we can form a piecewise C^2 cubic spline that passes through the points. In this paper we use piecewise cubic splines (or simply cubic splines afterwards) in generating the curves for J'.

By modifying the constrains on the boundary velocities, we can easily lower the continuity to C^1 or C^0. For example, to impose C^1, the left and right velocities at each boundary have to be the same. In such a case, the velocities at the boundaries become free variables for extra control. In our implementation, we decided to use cubic splines with C^1 continuity since it was enough to guarantee smooth motion.

As explained in Section 3, J' has to satisfy the interpolation condition, i.e. Condition 1. It can be easily achieved by having $\phi^{-1}(E(\Lambda))$ as the curve joints of the cubic spline. Several extra curve joints can be artificially created at *auxiliary frames* to add further control on the curve shape. Then the positions and velocities at the boundaries are adjusted to minimize the velocity difference between J' and J.

5.2 Auxiliary Frames

The curve joints come from two different kinds: keyframes and auxiliary frames. Keyframes are the frames where inverse kinematics is applied and are specified by the animator. On the other hand, auxiliary frames are introduced internally by the algorithm in order to increase the controllability.

At each curve joint there are two controllable parameters: position parameter $\vec{P}$ and velocity parameter $\vec{P}'$. At keyframes, $\vec{P}$ is already determined by inverse kinematics. Therefore only $\vec{P}'$ can be further controlled. At auxiliary frames, on the other hand, we set $\vec{P}'$ with the velocities in J, and can control $\vec{P}$ for our purpose. The situation is illustrated in Figure 3.

Since the cubic spline already passes through the interpolation points from its creation, both the velocity control at keyframes and position control at auxiliary frames can be used for minimizing the velocity difference between J' and J.

We found that the quality of the solution increases in general as more auxiliary frames are used. Therefore we picked every other frame for auxiliary frames as long as it doesn't overlap or neighboring with a keyframe. If we pick every frame as auxiliary frames, however, the velocity is determined at all frames and thus the curve is completely determined, leaving no room that can be controlled to realize the above goal.

5.3 Cubic Spline Regression

In this section, we find the solution to the problem of minimizing the velocity difference between J' and J.

Let t_k and t_{k+1} be the starting time and ending time, respectively, of the k-th curve segment. Then the k-th segment $\vec{P}_k(t)$ of the cubic spline for the duration $t_k \le t < t_{k+1}$ is represented by the following equation [16].

$$\vec{P}_k(t) = f_{k,1}(t) \cdot \vec{P}_k + f_{k,2}(t) \cdot \vec{P}_{k+1} + f_{k,3}(t) \cdot \vec{P}'_k + f_{k,4}(t) \cdot \vec{P}'_{k+1}, \tag{2}$$

where $\vec{P}_k$ and $\vec{P}'_k$ are the position and velocity parameters, respectively, at t_k. $\vec{P}_{k+1}$ and $\vec{P}'_{k+1}$ are the same parameters at t_{k+1}.

At each curve joint, either the position or velocity parameter is fixed and the other is variable, depending on whether it is a keyframe or auxiliary frame. Let's denote the fixed parameter at t_k as $\vec{F}_k$, and variable parameter at t_k as $\vec{V}_k$, for $k = 1, \ldots, L$, where L is the total number of curve joints including the two extreme points. For example, if k is a keyframe, $\vec{F}_k = \vec{P}_k$ and $\vec{V}_k = \vec{P}'_k$. With the new notations, Eq.(2) becomes

$$\vec{P}_k(t) = f_{k,a}(t) \cdot \vec{V}_k + f_{k,b}(t) \cdot \vec{V}_{k+1} + f_{k,c}(t) \cdot \vec{F}_k + f_{k,d}(t) \cdot \vec{F}_{k+1}. \tag{3}$$

In Eq.(3), $(a, c) = (1, 3)$ if k-th curve joint is an auxiliary frame, but $(a, c) = (3, 1)$ if k-th curve joint is a keyframe. b and d are determined in a similar way except that it is for the $(k + 1)$-th frame.

Our interest is in determining the variable parameters, i.e. $\vec{V}_k$ and $\vec{V}_{k+1}$. So, we rearrange Eq.(3) into the following form,

$$\vec{P}_k(t) = f_{k,a}(t) \cdot \vec{V}_k + f_{k,b}(t) \cdot \vec{V}_{k+1} + \vec{f}_{k,fixed}(t), \tag{4}$$

where $\vec{f}_{k,fixed}(t) = f_{k,c}(t) \cdot \vec{F}_k + f_{k,d}(t) \cdot \vec{F}_{k+1}$ in Eq.(3).

If we define $f_{k,a}(t)$, $f_{k,b}(t)$, and $\vec{f}_{k,fixed}(t)$ to be 0 outside $t_k \le t < t_{k+1}$, the whole curve can be represented as a sum of all the curve pieces as in the following equation

$$\vec{P}(t) = \sum_{k=1}^{L-1} \vec{P}_k(t) \quad (t_1 \le t \le t_L). \tag{5}$$

Since the regression have to be done for the whole curve, we expand the above equation to obtain

$$\vec{P}(t) = f_1(t) \cdot \vec{V}_1 + f_2(t) \cdot \vec{V}_2 + \ldots + f_L(t) \cdot \vec{V}_L + \vec{f}_{fixed}(t), \tag{6}$$

where

$$\vec{f}_{fixed}(t) = \sum_{k=1}^{L-1} \vec{f}_{k,fixed}(t) \tag{7}$$

$$f_1(t) = f_{1,a}(t) \tag{8}$$

$$f_k(t) = f_{k,a}(t) + f_{k-1,b}(t) \quad (1 < k < L-1) \tag{9}$$

$$f_L(t) = f_{L-1,b}(t) \tag{10}$$

Now we can describe our objective as

$$\texttt{minimize}\left[\sum_{i=0}^{N}(\dot{\vec{P}}[i]-\dot{J}_j(i))^2\right] \tag{11}$$

where $\dot{\vec{P}}[i]$ is the angular velocity of $\vec{P}(t)$ at frame i, and $\dot{J}_j(i)$ is the angular velocity of the original joint angle data J_j at frame i. The solution is given by the following equation.

$$[\vec{V}_1\ \vec{V}_2 \ldots \vec{V}_L]^T = (X^TX)^{-1}X^TY \tag{12}$$

where

$$X = \begin{bmatrix} \dot{f}_1[0] & \dot{f}_2[0] & \ldots & \dot{f}_L[0] \\ \dot{f}_1[1] & \dot{f}_2[1] & \ldots & \dot{f}_L[1] \\ \vdots & \vdots & \vdots & \vdots \\ \dot{f}_1[N] & \dot{f}_2[N] & \ldots & \dot{f}_L[N] \end{bmatrix} \tag{13}$$

$$Y = \begin{bmatrix} \dot{J}_j(0) - \dot{\vec{f}}_{fixed}[0] \\ \dot{J}_j(1) - \dot{\vec{f}}_{fixed}[1] \\ \vdots \\ \dot{J}_j(N) - \dot{\vec{f}}_{fixed}[N] \end{bmatrix} \tag{14}$$

$f[\cdot]$ is the discrete versions of $f(\cdot)$, i.e. $\dot{f}[i]$ and $\dot{\vec{f}}_{fixed}[i]$ are the values of $\dot{f}$ and $\dot{\vec{f}}_{fixed}$, respectively, at frame i.

By substituting the above $\vec{V}_1$, $\vec{V}_2$, $\ldots$, $\vec{V}_L$ into Eq.(6), we obtain a curve passing through the interpolation points that preserves the original joint angle pattern optimally.

The above procedure is for joint j. Therefore it has to be repeated for the joints $j = 1, \ldots, M$.

6 Experiments

We applied the above procedure to a boxing motion. The motion capture session involved two people: one was Boxer and the other was Trainer (Figure 4). Boxer was asked to punch the palms of Trainer. We captured Boxer's upper body motion by attaching sensors on his head, chest, pelvis, upper arms, lower arms, and hands. For Trainer, we captured only the palm motion. The session was recorded by a video camera and the clip is available at `http://graphics.snu.ac.kr/demo/pmcd/boxing_capture_session.mov.zip` in QuickTime movie format.

6.1 Forward Kinematic Reconstruction

The above motion was reconstructed with forward kinematics. The result is available at `http://graphics.snu.ac.kr/demo/pmcd/boxing_fk.mov.zip`. In Boxer's motion, pelvis was the base frame and the other sensors were used to extract the joint angles. As pointed out in the earlier discussion, the reconstruction produced large offsets between the fists and palms. Figure 5 (a) shows the frame (frame 13) at which the left fist was supposed to hit Trainer's left palm. Figure 5 (b) shows the frame (frame 29) at which the right fist was supposed to hit Trainer's left palm. The offsets are clearly recognizable.

6.2 Applying Our Enhancement Algorithm

When we animate the interaction of Boxer with Trainer, the fact whether the punch makes actual impact is quite important. We analyzed the original recording and identified 12 frames – frames 13, 29, 43, 58, 94, 102, 117, 130, 144, 182, 192, and 208 – at which the fists were supposed to hit the palms. To ensure the impacts, we designated those frames as keyframes. The final animation after our enhancement algorithm was applied is available in QuickTime format at `http://graphics.snu.ac.kr/demo/pmcd/boxing_enhanced.mov.zip`.

At the keyframes, first we performed inverse kinematics computation. Figure 6 shows the resulting joint angle trajectory around x-rotation axis at the waist joint. The keyframes are marked by circles. The graph is basically the original joint angle curve except for the keyframes, at which the result of inverse kinematics was plotted instead. That explains the discontinuities at the keyframes. Figure 7 shows the corrected postures by inverse kinematic reconstruction at the same frames as in Figure 5.

We proceeded further and minimized the velocity difference between J' and J according to the procedure described in Section 5, to obtain the smooth cubic spline curve shown in Figure 8. In the graph the curve was compared with the result in Figure 6. The solid curve is the new spline obtained by our algorithm, and the dotted curve is the one from Figure 6. Observe two things:

1. The curve *smoothly* interpolates the interpolation points.

2. The curve shape is quite *similar* to the original data curve.

The resemblance of the curves in Figure 8 indicates that our velocity difference minimization is quite effective in achieving $J' \approx J$.

The two velocities $\dot{J}'$ and $\dot{J}$ at the waist joint are compared in Figure 9. The solid and dotted curves represent $\dot{J}'$ and $\dot{J}$, respectively. If a joint angle value has to be increased (for example, to realize the keyframe constraint), the displacement has to be made by raising the velocity at previous frames. More specifically, the velocity adjustment should be done at intermediate non-auxiliary frames since the velocity is fixed at auxiliary frames. The above adjustments can produce the velocity ripples shown in Figure 9.

If we increase the number of curve joints the ripple frequency also increases proportionally. Although the marring impact of the ripple was not noticeable in the animation, we consider it as an unwanted side-effect. On the other hand, there are also benefits in having more curve joints (see Section 5.3). Our current choice – every other frame – produced nice results. But the trade-off remains to be studied further.

6.3 Other Results

We applied our algorithm to two other examples, catching and kicking motions, in which the human figure makes person-to-object interaction with a ball. The results are available at `http://graphics.snu.ac.kr/demo/pmcd/other_results.mov.zip` in QuickTime movie format.

7 Conclusion

In this paper we proposed an algorithm for processing motion capture data to achieve positional accuracy. When a set of keyframes are selected by the animator, the algorithm processes the original data to achieve the accuracy in the end-effector position, while preserving the original motion pattern. We formulated the problem as a constrained minimization of velocity difference between $J^{'}$ and J. Experiments proved that the formulation was in fact quite effective in preserving the motion pattern.

In the implementation, the frequency of curve joints were about 15 Hz - every other frame - since the sampling rate of the motion capture equipment was controlled to 30 Hz. This implies that our algorithm can effectively handle motion signals below 7.5 Hz according to the sampling theory [13]. 7 Hz is enough to cover the most important features of human motions [19]. But, if a motion is sampled at a higher frequency, our algorithm will be able to handle motion features of wider range of frequencies.

Our method can be useful to people in character animation. It will save animators from laborious manual correction of motion capture data, especially when the end-effector trajectory has to satisfy several positional goals.

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. Norman I. Badler, Brian A. Barsky, and David Zeltzer. *Making Them Move: Mechanics, Control, and Animation of Articulated Figures.* Morgan Kaufmann Publishers, 1990.

2. Norman I. Badler, Michael J. Hollick, and John P. Granieri. Real-time control of a virtual human using minimal sensors. *Presence*, 2(1):82–86, 1993.

3. D. Baker and Charles W. Wampler. On the inverse kinematics of redundant manipulators. *The International Journal of Robotics Research*, 7(2):3–21, March/April 1988.

4. Rama Bindiganavale and Norman 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, November 1998.

5. Bobby Bodenheimer, Charles F. Rose, S. Rosenthal, and J. Pella. The process of motion capture: Dealing with the data. In *Computer Animation and Simulation '97*, pages 3–18, 1997.

6. Armin Bruderlin and Lance Williams. Motion signal processing. In Robert Cook, editor, *Computer Graphics (SIGGRAPH '95 Proceedings)*, pages 97–104, August 1995. ACM-0-89791-701-4.

7. Michael Gleicher. Motion editing with spacetime constraints. In *1997 Symposium on Interactive 3D Graphics*, 1997.

8. Michael Gleicher. Retargetting 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.

9. M. Hirose, G. Deffaux, and Y. Nakagaki. Development of an effective motion capture system based on data fusion and minimal use of sensors. In *VRST '96*, pages 117–123, 1996.

10. Hyeongseok Ko and Norman I. Badler. Animating human locomotion with inverse dynamics. *IEEE Computer Graphics and Applications*, 16(2):50–59, March 1996.

11. T. Molet, , R. Boulic, and D. Thalmann. A real time anatomical converter for human motion capture. In *Eurographics workshop on Computer Animation and Simulation '96*, pages 79–94. Springer-Verlag Wien, 1996.

12. T. Molet, Z. Huang, R. Boulic, and D. Thalmann. An animation interface designed for motion capture. In *Proc. of Computer Animation '97*, pages 77–85. IEEE Press, 1997.

13. Alan V. Oppenheim, Alan S. Willsky, and S. Hamid Nawab. Signals & systems. Prentice-Hall, second edition, 1997.

14. Cary B. Phillips and Norman I. Badler. Interactive behaviors for bipedal articulated figures. In Thomas W. Sederberg, editor, *Computer Graphics (SIGGRAPH '91 Proceedings)*, volume 25, pages 359–362, July 1991.

15. David R. Pratt, Paul T. Barham, John Locke, Micheal J. Zyda, Naval Postgraduate School; Bryant Eastman, SARCOS Inc; Micheal Hollick, John Granieri, Hyeongseok Ko, Norman I. Badler, University of Pennsylvania. Insertion of an articulated human into a networked virtual environment. In *Distributed Interactive Simulation Environments Conference*, University of Florida, December 1994.

16. David F. Rogers and J. Alan Adams. *Mathematical Elements for Computer Graphics.* McGraw-HILL International Editions, second edition, 1990.

17. Charles Rose, Brian Guenter, Bobby Bodenheimer, and Michael F. Cohen. Efficient generation of motion transitions using spacetime constraints. In *Computer Graphics (SIGGRAPH '96 Proceedings)*, pages 147–154, August 1996. ACM-0-89791-746-4.

18. S. K. Semwal, R. Hightower, and S. Stansfield. Mappingg algorithms for real-time control of an avatar using eight sensors. *Presence*, 7(1):1–21, 1998.

19. David A. Winter. *Biomechanics and Motor Control of Human Movement.* Wiley, New York, second edition, 1990.

20. Andrew Witkin and Zoran Popovic. Motion warping. In Robert Cook, editor, *Computer Graphics (SIGGRAPH '95 Proceedings)*, pages 105–108, August 1995. ACM-089791-701-4.

21. Jianmin Zhao and Norman I. Badler. Inverse kinematics positioning using nonlinear programming for highly articulated figures. *ACM Transactions on Graphics*, 13(4):313–336, October 1994.

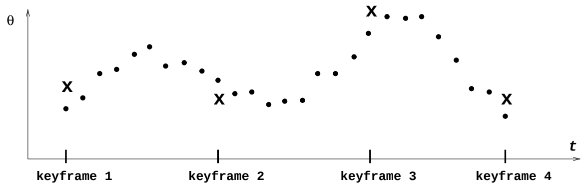

Figure 1:

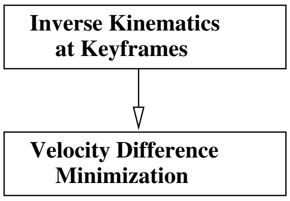

Figure 2:

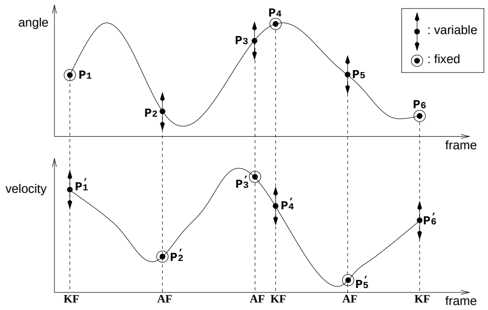

Figure 3:

Figure 4:

(a) (b)

Figure 5:

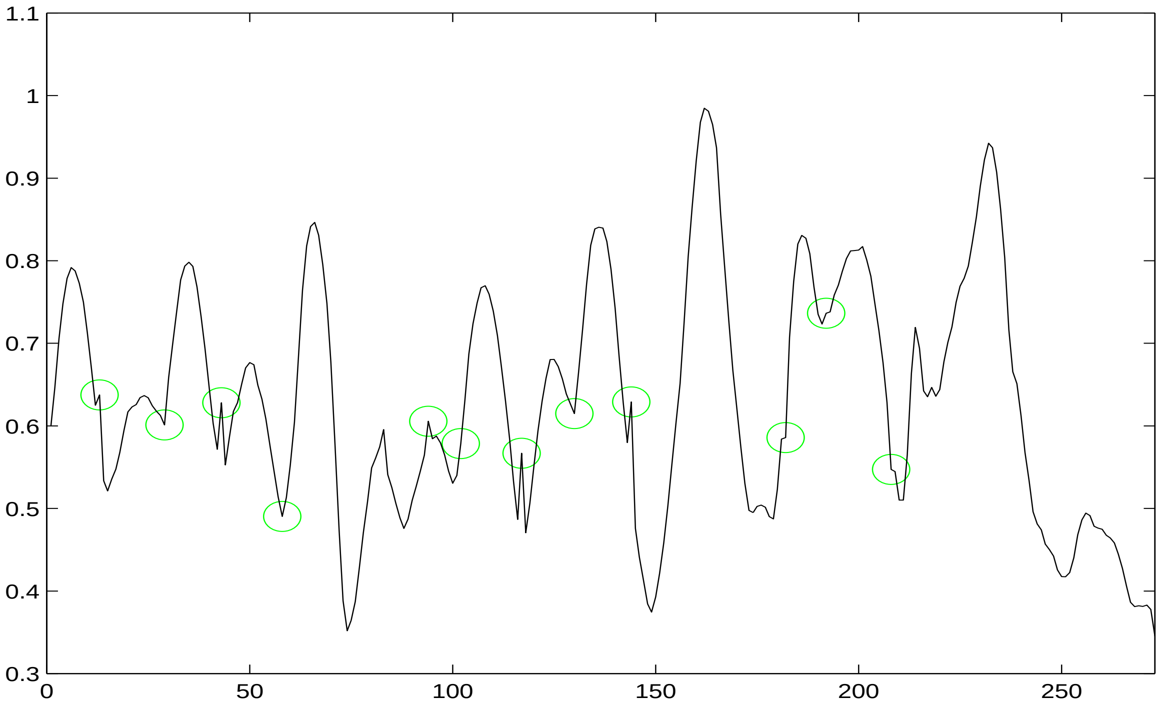

Figure 6:

(a)

(b)

Figure 7:

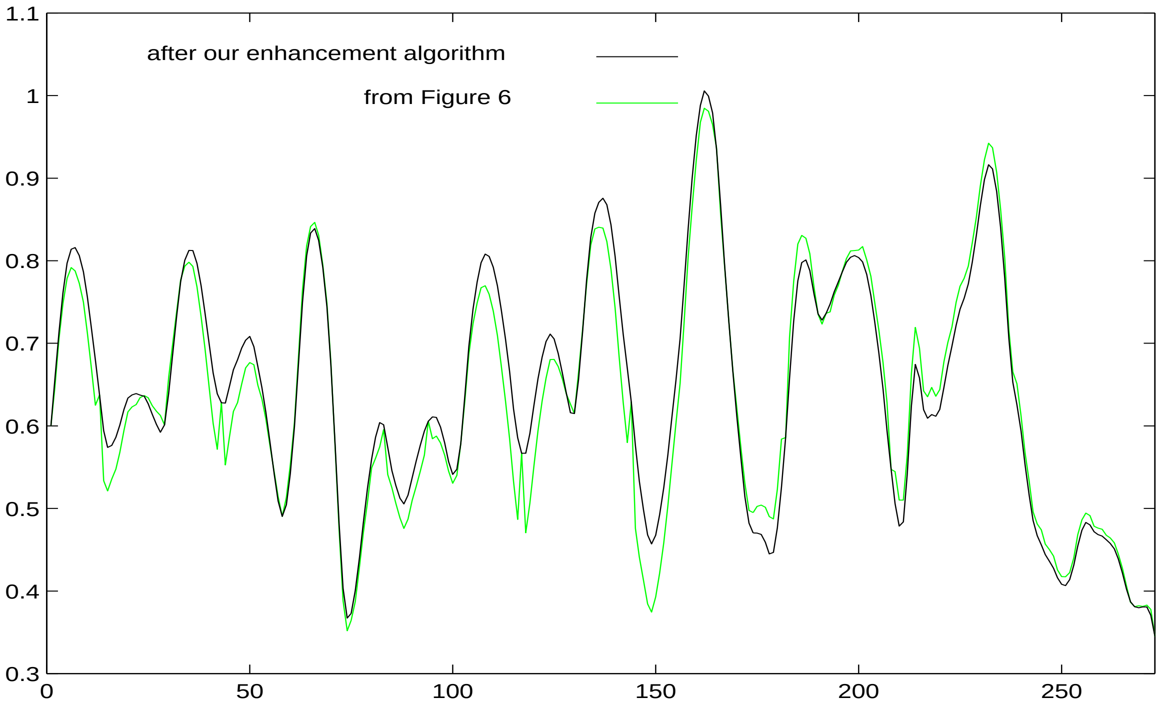

Figure 8:

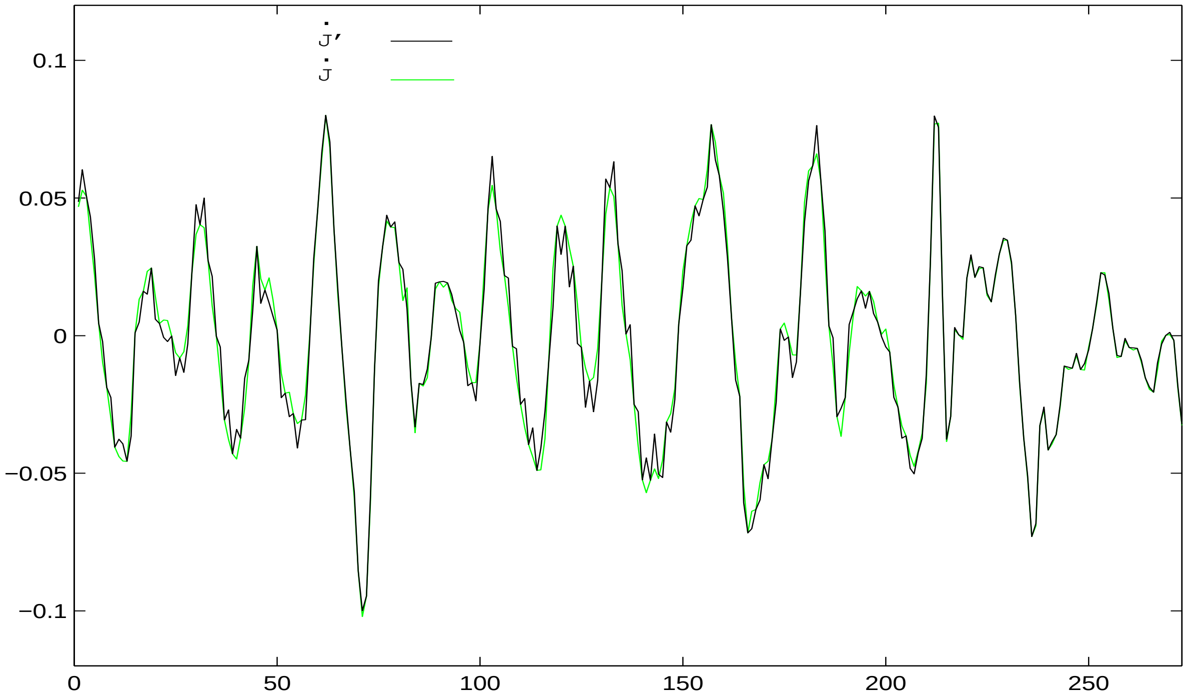

Figure 9:

Figure 1 : Joint angle data J_j at a typical joint j (dotted points), and $\phi^{-1}(E(\Lambda))$ (marked with X)

Figure 2 : Two major steps of our algorithm

Figure 3 : A joint angle trajectory plotted in the angle space (above) and velocity space (below). KF and AF stand for keyframe and auxiliary frame, respectively. Vertical arrows mean the point can be adjusted up or down for our purpose. At each curve joint, if the position is fixed then the velocity is variable, and vice versa.

Figure 4 : The motion capture session: Boxer (right), Trainer (left)

Figure 5 : End-effector offsets at impact moments. (a) forward kinematic reconstruction of frame 13. (b) forward kinematic reconstruction of frame 29.

Figure 6 : The joint angle data at the waist joint. At the keyframes (13, 29, 43, 58, 94, 102, 117, 130, 144, 182, 192, and 208), the result of inverse kinematics was plotted instead. The keyframes are marked with circles.

Figure 7 : Inverse kinematic reconstruction of impact moments. (a) the reconstruction of frame 13, (b) the reconstruction of frame 29.

Figure 8 : The new joint angle trajectory.
solid curve: after our enhancement algorithm
dotted curve: copied from Figure 6.

Figure 9 : Comparison of velocity curves:
solid curve - from our algorithm ($\dot{J}'$)
dotted curve - from the original data ($\dot{J}$)