

Motion Fairing

Jehee Lee

Sung Yong Shin

Computer Science Department

KAIST

Taejeon 305-701, Korea

Abstract

Motion capturing is widely used for generating realistic motions. A 3D input device captures a sequence of 6 DOF rigid motion samples. The sampled data consist of two components, translation and orientation; the former is represented by a vector in R^3 and the latter by a rotation matrix in the rotation group, $SO(3)$. Since the sequence of data contain sampling noises, the captured motion is not smooth and wiggles along the moving path. There are well-known fairing algorithms in Euclidean space based on difference geometry. Extending these for the motion data, we present a new fairing algorithm. The new algorithm iteratively minimizes the energy function reflecting the forces and torques, exerted on a moving object, to perform motion fairing.

Keywords: Quaternion, rotation, $SO(3)$, S^3 , fairing, signal processing

1 INTRODUCTION

Motion control of a human-like articulated figure is a fundamental problem in computer graphics. Traditional keyframing techniques provide editing and manipulating tools for motions based on spline interpolation. Recent progress in orientation interpolation makes it possible to generate rigid motions consisting of smoothly varying positions and orientations. However, the manipulation of an articulated figure by means of keyframes and spline curves is a very complex task that requires highly developed human skills.

Recently, researches on motion control aim to automatically synthesize realistic motions. Dynamic simulation techniques have been used for generating the motion of an articulated figure without a self-actuating force [9, 13]. Specialized motion controllers have been constructed for walking, hopping, and swinging

through the careful analysis of each motion [1, 14]. These approaches often suffer from lack of interactivity. Since motions are governed by a controller, animators can not explicitly edit the motions. The spacetime control is devised to overcome this problem [18, 3]. Through numerical optimization, user-specified boundary postures are interpolated. However, it requires heavy computation to solve constrained dynamic equations.

An alternative is to capture data from a real actor in motion instead of automatically generating them. Since 3D motion capture devices have recently become commercially available, they have widely spread for obtaining realistic and complex motions. However, the captured data are not smooth enough to give an intended motion sequence. For virtual reality applications, a virtual agency moves according to the signal received from a 3D input device such as a data globe. Since the control signal contains sampling noises, the controlled agency unwillingly wiggles. Therefore, a sequence of sampled data must be faired to give a smoothly varying motion.

A posture of a rigid body in R^3 can be represented by a Cartesian product of two components: one in Euclidean space R^3 and the other in the rotation group $SO(3)$ [4, 11]. The former represents the center position of the rigid body, and the latter does its orientation. Considering an articulated figure, a posture of a freely rotating limb can be represented by the rotational component. The fairing in R^3 is well-established in computer-aided geometric design [7, 6, 16]. However, there are few results on fairing a sequence of orientation samples. In this paper, we present a new fairing algorithm that iteratively smoothes 6 DOF rigid motion data.

Section 2 discusses an underlying idea for our ap-

proach in motion fairing. In Section 3 we introduce a unit quaternion to compactly represent an orientation, and discretize an energy function using forward difference operators. Section 4 provides a physical interpretation of the energy function. Section 5 presents our fairing algorithm, and Section 6 gives experimental results. Finally, Section 7 concludes this paper.

2 BACKGROUND

In geometric modeling, fairing plays an important role for designing high quality curves and surfaces. There are rich results on fairing to get an aesthetically pleasing and functionally useful curve [7, 6, 16]. In most of literatures, the resulting curve minimizes the weighted sum of derivatives of the curve, which is given by

$$E(C) = \int_C \sum_{m=1}^n \alpha_m \|C^{(m)}(s)\|^2 ds, \quad (1)$$

where $C^{(m)}(s)$ is a m -th order derivative for a curve $C(s)$ which is parameterized by arc length. If we choose weights α_m such that

$$\alpha_m = \begin{cases} 1 & \text{if } m = 2 \\ 0 & \text{if } m \neq 2, \end{cases}$$

then the function is reduced to the strain energy of a spline curve, given by

$$E_s(C) = \int_C |\kappa(s)|^2 ds \quad (2)$$

which is frequently used to fair a curve.

Free-form curves and surfaces are produced by digitizing a clay model or a prototype, which gives a piecewise linear curve (PLC) passing through a sequence of digitized points [5, 8, 15]. Eck and Jaspert suggested a PLC fairing algorithm which minimizes the estimated strain energy function.

For a unit quaternion curve $Q(t)$ which describes a rotational motion, the energy function in Equation (1) does not work. For example, let $Q_s(t)$ be a rotational curve with a constant angular velocity. Then, it moves along a great arc of the unit hyper-sphere, $x^2 + y^2 + z^2 + w^2 = 1$. The curve is optimally smooth, but $\|Q_s^{(2)}\|$ is 1 instead of zero. Moreover, there exist non-geodesic curves with the same second derivative. This shows that $Q^{(j)}(t)$ may not be a good criterion for measuring the fairness of $Q(t)$. Barr, et al. [2] addressed this problem for spline interpolation of quaternion keyframes. He adopted the tangential component of $Q^{(2)}(t)$ as a fairness measure.

Our approach focuses on the derivatives of angular velocity rather than those of $Q(t)$. The angular velocity $\omega(t)$ is very useful for analyzing the infinitesimal behavior of a rotational motion [12]. Accommodating rotation and simplifying Equation (1), we get an objective function for a rigid motion $R(t) = (P(t), Q(t)) \in R^3 \times S^3$:

$$E(P, Q) = \int_0^t \alpha \|P^{(m)}(s)\|^2 + \beta \|\omega^{(m-1)}(s)\|^2 ds, \quad (3)$$

where α and β are weighting factors. This formulation generalizes the strain energy given in Equation (2). Moreover, it is reduced to the fairness measure of Barr, et al. when $\alpha = 0$, $\beta = 4$ and $m = 2$ [10].

3 PRELIMINARIES

3.1 Orientation Representation

From the fundamental theorem presented by Euler, any orientation of a rigid body can be represented as a rotation about a fixed axis $\hat{v} \in R^3$ through an angle $\theta \in [0, 2\pi)$. Using a unit quaternion $q = (\cos \frac{\theta}{2}, \hat{v} \sin \frac{\theta}{2}) \in S^3$, the orientation is described by a 3D rotation $R_q \in SO(3)$ in a global coordinate system as follows:

$$R_q(p) = qpq^{-1}, \quad \text{for } p \in R^3, \quad (4)$$

where a point $p = (x, y, z)$ in the global coordinate system is interpreted as a quaternion $(0, x, y, z)$. Note that $R_q = R_{-q}$; that is, two antipodal points, q and $-q$ represent the same rotation in $SO(3)$. Therefore, the mapping between S^3 and $SO(3)$ is 2-to-1.

Given a vector $v = \theta \hat{v} \in R^3$, the corresponding unit quaternion defines the exponential map:

$$\exp(v) = (\cos \theta, \hat{v} \sin \theta) \in S^3. \quad (5)$$

This map is surjective, but not one-to-one. To define its inverse function, we limit the domain so that $\|\theta\| < \pi$. Then, the $\exp(v)$ becomes one-to-one, and the inverse map $\log(q) : S^3 / (-1, 0, 0, 0) \rightarrow R^3$ are well-defined.

Given a unit quaternion curve $Q(t)$, its angular velocity $\omega(t) \in R^3$ is :

$$\omega(t) = 2\dot{Q}(t)Q^{-1}(t), \quad (6)$$

which is measured in the global coordinate system. $\frac{\omega(t)}{\|\omega(t)\|}$ is an instantaneous axis of the rotation, and $\|\omega(t)\|$ is the rate of change of a rotation angle about the axis.

To interpolate two quaternion points q_1 and q_2 , Shoemake suggested the *slerp* (spherical linear interpolation) which joins the two points along a geodesic curve on S^3 [17]:

$$\begin{aligned} \text{slerp}(t; q_1, q_2) &= q_1 \exp(t \cdot \log(q_1^{-1} q_2)) \\ &= q_1 (q_1^{-1} q_2)^t. \end{aligned} \quad (7)$$

3.2 Difference Operators

Given ordered points $P = (p_0, p_1, \dots, p_{n-1})$, where $p_i \in R^3$, the difference operator is defined recursively as follows:

$$\begin{aligned} D^0 p_i &= p_i, \\ D^j p_i &= D^{j-1} p_{i+1} - D^{j-1} p_i \quad \text{if } j > 0. \end{aligned} \quad (8)$$

Suppose that p_i 's are a sequence of digitized points of an unknown curve $C(t)$ and that Δ is the time interval between each pair of consecutive points. It is well-known that $\frac{D^j p_i}{\Delta^j}$ is a good approximation of $C^{(j)}(t)$ at p_i . $\frac{D p_i}{\Delta}$ is called a *discrete tangent*, and the magnitude of $\frac{D^2 p_i}{\Delta^2}$ is called a *discrete curvature* at p_i . $\|D^2 P\|$ has frequently been used as a measure of fairness in many previous works [6]. As $\|D^2 P\|$ decreases, P gradually straighten. Eventually, the points are on a straight line segment when $\|D^2 P\| = 0$.

For a unit quaternion curve $Q(t)$ describing a rotational movement, the angular velocity of $Q(t)$ can be defined:

$$\omega(t) = \lim_{\Delta t \rightarrow 0} \frac{\log(Q(t)^{-1} Q(t + \Delta t))}{\Delta t}.$$

Let $q_i = Q(t_i)$, $i = 0, \dots, n-1$. From $Q = \{q_0, q_1, \dots, q_{n-1}\}$, we approximate $\omega(t)$ at t_i :

$$\omega(t_i) = \frac{\log(q_i^{-1} q_{i+1})}{\Delta}. \quad (9)$$

Let $Q_i(t) : [t_i, t_{i+1}] \rightarrow SO(3)$ be a function which varies from q_i to q_{i+1} with a constant angular velocity. Then, $\omega(t_i)$ is the angular velocity of $Q_i(t)$ at t_i .

An analogy of the recursive definition in Equation (8) gives the following formula:

$$\begin{aligned} W^1 q_i &= \log(q_i^{-1} q_{i+1}), \\ W^j q_i &= W^{j-1} q_{i+1} - W^{j-1} q_i, \quad \text{if } j > 1. \end{aligned} \quad (10)$$

$\frac{W^{j+1} q_i}{\Delta^{j+1}}$ is a finite difference approximation of $\omega^{(j)}(t_i)$. The behavior of W^j is analogous to that of D^j for points in R^3 . As $\|W^2 Q\|$ declines to zero, q_i 's approach to a great arc on the unit hyper-sphere S^3 .

4 Energy functions

4.1 Discretization

Let $P = (p_0, \dots, p_{n-1})$ and $Q = (q_0, \dots, q_{n-1})$ be an input sequence of rigid postures, where $p_i \in R^3$ and $q_i \in S^3$ for all i . Approximating $\|P^{(m)}\|$ and $\|\omega^{(m-1)}\|$ by $\|D^m p_i\|$ and $\|W^m q_i\|$, respectively, the objective function in Equation (3) is discretized:

$$E_m(P, Q) = \alpha \sum_{i=0}^{n-m} \|D^m p_i\|^2 + \beta \sum_{i=0}^{n-m} \|W^m q_i\|^2, \quad (11)$$

which is called an *m-th order energy function* for a rigid motion. Minimizing this function finds the sequences, P and Q . This problem can be divided into two independent subproblems which minimize following two functions, respectively:

$$\begin{aligned} E_m^t(P) &= \sum_{i=0}^{n-m} \|D^m p_i\|^2, \quad \text{and} \\ E_m^r(Q) &= \sum_{i=0}^{n-m} \|W^m q_i\|^2. \end{aligned} \quad (12)$$

4.2 Physical Analogy

Let f_i and τ_i be the force and torque at t_i , respectively, for $i = 0, \dots, n-1$. They are exerted to make a rigid body move along a path $R(t_i) = (P(t_i), Q(t_i))$, where $p_i = P(t_i)$ and $q_i = Q(t_i)$. Since the force and torque are proportional to $\|D^{(2)} p_i\|$ and $\|W^{(2)} q_i\|$, their minimum values are achieved when the second order energy function vanishes.

The magnitude of the second order difference terms are approximations of translational and angular accelerations, respectively. Similarly, the third order difference terms approximate rates of changes of the accelerations. Hence, a motion with zero third order difference terms has constant accelerations.

Let \mathcal{C}_m be a class of motion data whose m -th order energy function is zero. If $\|D^{(m-1)}\| = 0$ and $\|W^{(m-1)}\| = 0$, then $\|D^{(m)}\| = 0$ and $\|W^{(m)}\| = 0$. Therefore, a motion $R \in \mathcal{C}_{(m-1)}$ is also contained in \mathcal{C}_m . The converse does not always hold true. Hence, we can conclude that a higher order energy function gives a more broad class of motions. For examples, circular translation is contained in \mathcal{C}_3 but have non-zero second order energy functions. Regular precession in \mathcal{C}_4 is a typical movement of a rotating top. The axis of the top rotates uniformly about a fixed direction, describing a circular cone, and at the same time it rotates about its own axis.

5 FAIRING ALGORITHM

We treat the problem of fairing as energy minimization. Unfortunately, the energy function to be minimized in Equation (11) is non-linear. There is no known direct method to find a solution. For a small angular displacement, the function can be approximated by a linear function. Therefore, we adopt an iterative numerical method to solve the problem.

5.1 Fairing Operators

Every existing fairing algorithm tries to strictly reduce an energy function at each step of iterations. Formally stating, given a sequence of rigid postures $R = (P, Q)$, an m -th order fairing operator ρ_m is defined as follows:

$$R^* = \rho_m(R),$$

such that

$$E_m(R^*) < E_m(R). \quad (13)$$

Consider the second order energy function:

$$\begin{aligned} E_2(R) &= E_2^t(P) + E_2^r(Q) \\ &= \sum_{i=0}^{n-m} \|D^2 p_i\|^2 + \sum_{i=0}^{n-m} \|W^2 q_i\|^2, \end{aligned} \quad (14)$$

where

$$\begin{aligned} D^2 p_i &= p_{i+2} - 2p_{i+1} + p_i, \\ W^2 q_i &= \log(q_{i+2}^{-1} q_{i+1}) - \log(q_{i+1}^{-1} q_i). \end{aligned} \quad (15)$$

This function gives a minimum value when $D^2 p_i = 0$ and $W^2 q_i = 0$. From these conditions, the equation can be rearranged for p_i 's and q_i 's:

$$\begin{aligned} p_i &= 2p_{i+1} - p_{i+2}, \\ p_{i+1} &= \frac{1}{2}(p_i + p_{i+2}), \\ p_{i+2} &= 2p_{i+1} - p_i, \\ q_i &= q_{i+1} q_{i+2}^{-1} q_{i+1}, \\ q_{i+1} &= q_{i+2} q_i^{-1} q_i, \\ q_{i+2} &= q_{i+1} q_i^{-1} q_{i+1}. \end{aligned}$$

Shifting the indices of these equations, a fixed point iteration is obtained:

$$R^* = \rho_2(R) = (\rho_2^t(P), \rho_2^r(Q)), \quad (16)$$

where

$$\begin{aligned} p_i^* &= \begin{cases} 2p_1 - p_2, & \text{if } i = 0, \\ \frac{1}{2}(p_{i-1} + p_{i+1}), & \text{if } 0 < i < n-1, \\ 2p_{n-2} - p_{n-3}, & \text{if } i = n-1, \end{cases} \\ q_i^* &= \begin{cases} q_1 q_2^{-1} q_1, & \text{if } i = 0, \\ q_{i+1} q_i^{-1} q_{i-1}, & \text{if } 0 < i < n-1, \\ q_{n-2} q_{n-3}^{-1} q_{n-2}, & \text{if } i = n-1. \end{cases} \end{aligned}$$

In order to avoid unexpected oscillation, we accommodate damping terms $0 < \lambda_t, \lambda_r < 1$, which give:

$$\hat{\rho}_2(R) = (\hat{\rho}_2^t(P), \hat{\rho}_2^r(Q)), \quad (17)$$

where

$$\begin{aligned} \hat{\rho}_2^t(P) &= \lambda_t \rho_2^t(P) + (1 - \lambda_t)P, \quad \text{and} \\ \hat{\rho}_2^r(Q) &= \text{slerp}(\lambda_r; Q, \rho_2^r(Q)). \end{aligned}$$

Since operator $D^{(2)}$ is diagonal dominant, the fixed point iteration converges. Due to the non-linearity of $W^{(2)}$, the convergency of iteration of $\hat{\rho}_2^r$ is not guaranteed. However, it can be regarded as a linear function for a small angular displacement. Therefore, a sufficiently small λ_r bounds the displacement at each step, and the iteration is expected to converge.

Similarly, the fixed point iteration for a higher order fairing operator can be derived. In particular, the third order fairing operator ρ_3 gives the following iteration:

$$R^* = \rho_3(R) = (\rho_3^t(P), \rho_3^r(Q)), \quad (18)$$

where

$$\begin{aligned} p_i^* &= \begin{cases} p_3 - 3p_2 + 3p_1, & \text{if } i = 0, \\ p_{i+1} + \frac{1}{3}(p_{i-1} - p_{i+2}), & \text{if } 0 < i < n-2, \\ p_{n-3} + \frac{1}{3}(p_{n-1} - p_{n-4}), & \text{if } i = n-2, \\ p_{n-4} - 3p_{n-3} + 3p_{n-2}, & \text{if } i = n-1, \end{cases} \\ q_i^* &= \begin{cases} q_1 \cdot \exp(2 \log(q_2^{-1} q_1) - \log(q_3^{-1} q_2)), & \text{if } i = 0, \\ q_{i+1} \cdot \exp(\frac{1}{2}(\log(q_{i+2}^{-1} q_{i+1}) + \log(q_i^{-1} q_{i-1}))), & \text{if } 0 < i < n-2, \\ q_{n-3} \cdot \exp(-\frac{1}{2}(\log(q_{n-1}^{-1} q_{n-2}) + \log(q_{n-3}^{-1} q_{n-4}))), & \text{if } i = n-2, \\ q_{n-2} \cdot \exp(\log(q_{n-3}^{-1} q_{n-4}) - 2 \log(q_{n-2}^{-1} q_{n-3})), & \text{if } i = n-1. \end{cases} \end{aligned}$$

5.2 Termination Criteria

In general, a fixed point iteration terminates when an error between $R^{(k)}$ and $R^{(\infty)}$ is within a tolerance. However, $R^{(\infty)}$ is not known. Our algorithm,

```

function FAIRING $m, \lambda_t, \lambda_r$ ( $P^{(0)}, Q^{(0)}; \epsilon_t, \epsilon_r$ )
begin
   $P = P^{(0)}$ 
   $Q = Q^{(0)}$ 
  while ( $\max_i \|p_i^{(0)} - p_i\| < \epsilon_t$ ) and
        ( $\max_i(\text{dist}(q_i^{(0)}, q_i)) < \epsilon_r$ )
  begin
     $P = \hat{\rho}_m^t(P)$ 
     $Q = \hat{\rho}_m^r(Q)$ 
  end
  return ( $P, Q$ )
end.

```

Figure 1: The fairing algorithm

described in Figure (1), bounds the error between $R^{(0)}$ and $R^{(k)}$ instead of that between $R^{(k)}$ and $R^{(\infty)}$. It is reasonable to measure the error of P by Euclidean distance between the initial position $p_i^{(0)}$ and the final position $p_i^{(k)}$. Since quaternion points $q_i^{(0)}$ and $q_i^{(k)}$ are on the unit hyper-sphere S^3 , the distance should be the length of a geodesic curve joining the two points, and the length is equivalent to the angle between them. Since $q_i^{(0)}$ and $-q_i^{(0)}$ represent the same orientation, we define the distance between two unit quaternions as follows:

$$\begin{aligned}
& \text{dist}(q_i^{(0)}, q_i^{(k)}) \\
&= \min(\text{angle}(q_i^{(0)}, q_i^{(k)}), \text{angle}(-q_i^{(0)}, q_i^{(k)})), \\
&= \min(\cos^{-1}(q_i^{(0)} \cdot q_i^{(k)}), \cos^{-1}(-q_i^{(0)} \cdot q_i^{(k)})).
\end{aligned} \tag{19}$$

Given user-specified error bounds ϵ_t and ϵ_r , the algorithm terminates at the k -th iteration when the following conditions are not satisfied:

$$\begin{aligned}
& \max_i \|p_i^{(0)} - p_i^{(k)}\| < \epsilon_t, \quad \text{and} \\
& \max_i(\text{dist}(q_i^{(0)}, q_i^{(k)})) < \epsilon_r.
\end{aligned} \tag{20}$$

These terminating conditions guarantee that every resulting data point is within a given bound from its initial position.

6 EXPERIMENTAL RESULTS

To apply our algorithm, we need to determine the order of a fairing operator and damping factors. The order is related to the quality of a produced motion. Let \mathcal{C}_m be a class of motions which make the m -th order energy function zero. Then, fairing motion data is

interpreted as a projection of the data onto \mathcal{C}_m . Since $\mathcal{C}_n \subset \mathcal{C}_m$ for $n < m$, a higher order fairing operator projects the data within a smaller error bound. We perform experiments with the second order and the third order fairing algorithms. The initial data q_i 's are sampled from a quaternion spline interpolation curve with random perturbations $\|\delta_i^t\| < 10(mm)$ and $\|\delta_i^r\| < 0.1(\text{radians})$, assuming that p_i 's are placed in a cube whose size is $400 \times 400 \times 400 (mm^3)$. Figure (2) shows the average errors $\sum_{i=0}^n (\cos^{-1}(q_i^{(0)} \cdot q_i^{(k)}))/n$ of the second and the third order algorithms. The average error of the third order algorithm converges within the range of perturbation. However, the second order algorithm deforms the data too much.

An iteration converges when $\hat{\rho}_m$ satisfies the condition of fairing in Equation (13). Unfortunately, it is hard to find such an operator $\hat{\rho}_m$ because the energy function is non-linear. In practice, we can make the iteration converge by properly choosing a reasonably small damping factor. Figure (3) plots the average errors for variants of λ_r . The large damping factor causes the iteration to converge at a local minimum.

Figure (6)–(5) shows an example with 100 data points for $m = 3$, $\lambda_t = 0.5$, and $\lambda_r = 0.1$. The initial motion, depicted in Figure (6), is generated by perturbing sample data from a spline curve. The perturbation is performed within 0.2(radians) for rotation and 20(mm) for translation. Fairing in R^3 with 100 iterations gives smooth translation as shown in Figure (7) where the trajectory is well smoothed but its rotation wiggles. Figure (8) demonstrates the final motion where its orientation is also fairied with 100 iterations. Figure (4) and (5) show acceleration and angular acceleration plots, respectively. The acceleration and the angular acceleration are rapidly reduced at first 10 steps. The experiments are performed on an SGI Indigo 2 with 133 MHz CPU clock. It takes only a second to fair 300 data points containing both translation and orientation with 100 iterations. Our algorithm is sufficiently fast for real-time applications.

7 CONCLUSIONS

Given captured motion data, the proposed algorithm fairs both translational and rotational components. We formulate the fairing in terms of non-linear optimization. The m -th order energy function which is a generalization of the strain energy function is adopted as an objective function to be minimized. We discretized the energy function with the recursive dif-

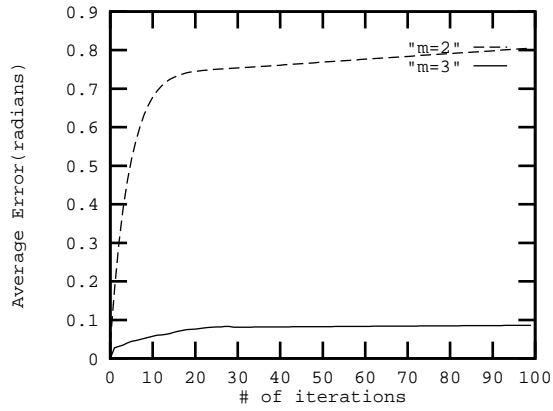


Figure 2: Error plots for $m = 2$ and $m = 3$; $\lambda_r = 0.1$

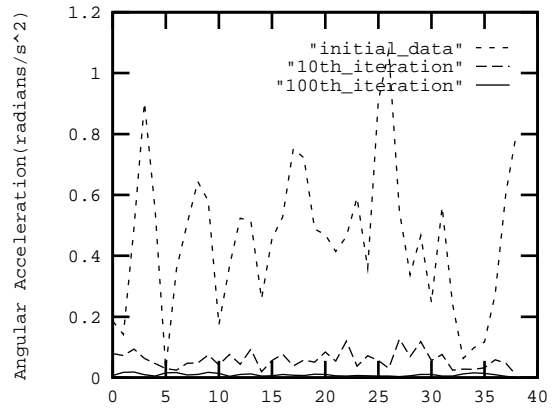


Figure 5: Angular acceleration plots

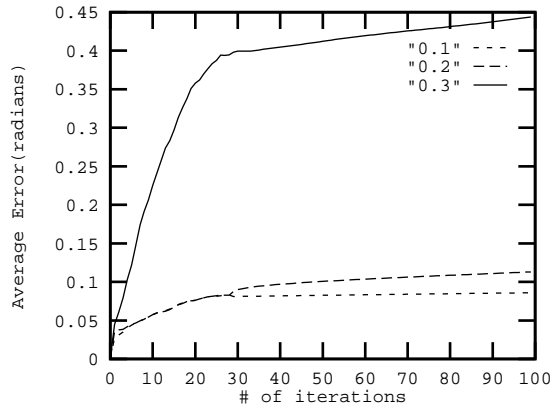


Figure 3: Error plots for $\lambda_r = 0.1, 0.2,$ and 0.3 ; $m = 3$

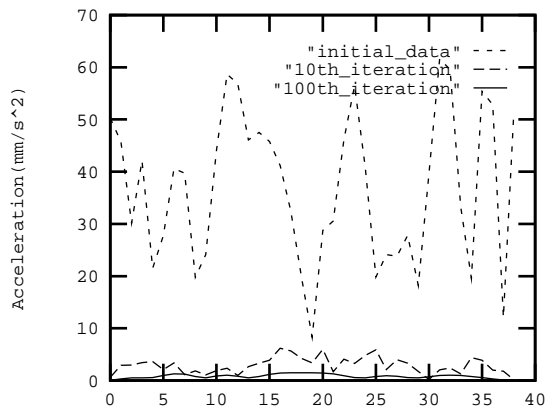


Figure 4: Acceleration plots

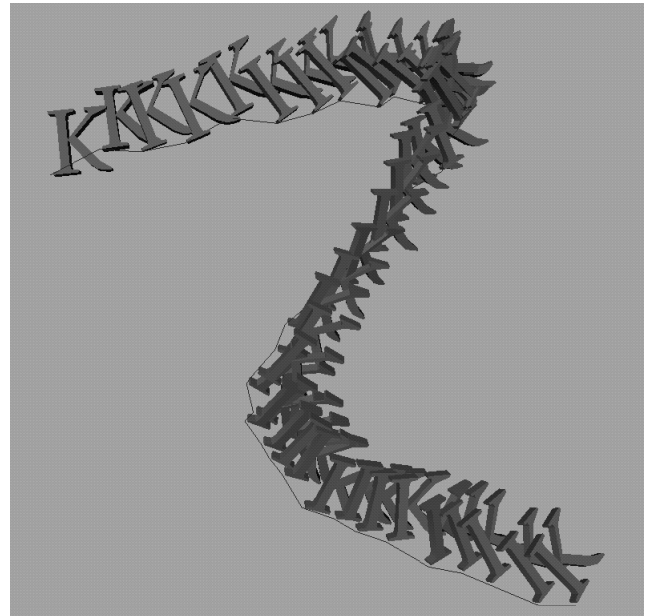


Figure 6: The initial noisy motion

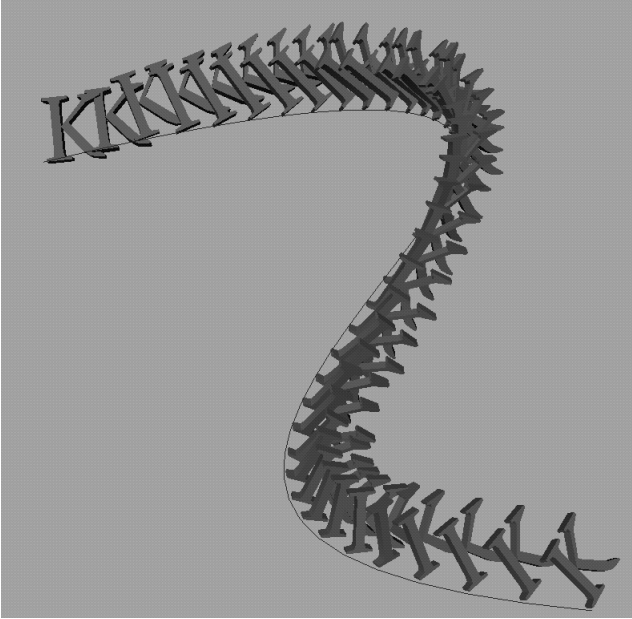


Figure 7: After translational fairing

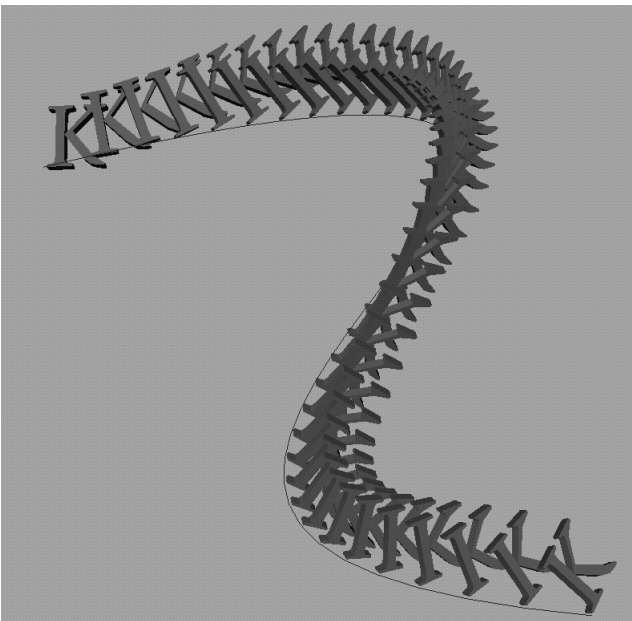


Figure 8: The final faired motion

ference operators, which approximate derivative terms in the energy function. To minimize the discretized energy function, we have derived fairing operators which iteratively reduce the function. The convergency of the algorithm is achieved by properly adjusting the damping factors.

Acknowledgements

This research was partially supported by Korean Ministry of Science and Technology through Software Technique Enhancement Program 2000.

References

- [1] N. Badler, B. Barsky, and D. Zelzer(Eds.). *Making Them Move*. Morgan Kaufmann, 1991.
- [2] A. Barr, B. Currin, S. Gabriel, and J. Hughes. Smooth interpolation of orientations with angular velocity constraints. *Computer Graphics(Proceedings of SIGGRAPH '92)*, 26:313–320, July 1992.
- [3] M. Cohen. Interactive spacetime control for animation. *Computer Graphics(Proceedings of SIGGRAPH '92)*, 26(2):293–302, July 1992.
- [4] M. Curtis. *Matrix Groups*. Springer-Verlag, 1972.
- [5] M. Eck and R. Jaspers. Automatic fairing of point sets. in *Designing Fair Curves and Surfaces*, Edited by N. S. Sapidis, SIAM, pages 45–60, 1994.
- [6] G. Farin. *Curves and Surfaces for Computer Aided Geometric Design*. Academic Press, 1988.
- [7] G. Farin, G. Rein, N. Sapidis, and A. J. Worsey. Fairing cubic b-spline curves. *Computer Aided Geometric Design*, 4:91–103, 1987.
- [8] M. Feldman. Tight string method to fair piecewise linear curves. in *Designing Fair Curves and Surfaces*, Edited by N. S. Sapidis, SIAM, pages 61–72, 1994.
- [9] J. K. Hahn. Realistic animation of rigid bodies. *Computer Graphics(Proceedings of SIGGRAPH '88)*, 22(4):299–308, August 1988.
- [10] Hyung jin Ha. A new camera control method preserving view-up vectors. Master's thesis, KAIST, 1995.

- [11] Myoung-Jun Kim, Myung-Soo Kim, and Sung Yong Shin. A general construction scheme for unit quaternion curves with simple high order derivatives. *Computer Graphics(Proceedings of SIGGRAPH '95)*, 29:369–376, August 1995.
- [12] L. D. Landau and E. M. Lifshitz. *Mechanics(Course of Theoretical Physics Vol.1)*. Robert Maxwell, M. C, 1976.
- [13] M. Moore and J. Wilhelms. Collision detection and response for computer animation. *Computer Graphics(Proceedings of SIGGRAPH '88)*, 22(4):289–298, August 1988.
- [14] M. H. Raibert and J. K. Hodgins. Animation of dynamic legged locomotion. *Computer Graphics(Proceedings of SIGGRAPH '91)*, 25:349–358, July 1991.
- [15] W. Renz. Interactive smoothing of digitized point data. *Computer Aided Design*, 14:267–269, 1982.
- [16] N. Sapidis and G. Farin. Automatic fairing algorithm for b-spline curves. *Computer Aided Design*, 22:121–129, 1990.
- [17] K. Shoemake. Animating rotation with quaternion curves. *Computer Graphics(Proceedings of SIGGRAPH '85)*, 19:245–254, 1985.
- [18] A. Witkin and M. Kass. Spacetime constraints. *Computer Graphics(Proceedings of SIGGRAPH '88)*, 22(4):159–168, August 1988.