Spline Motion Transitions in Linear Subspaces

Tomohiko Mukai *
Square Enix Co., Ltd.

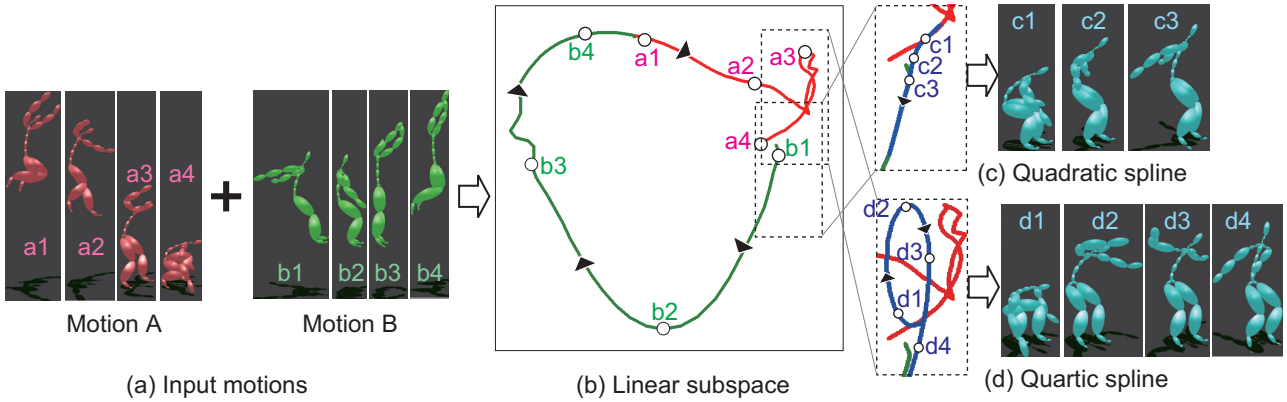

Figure 1: *(a) Synthesizing a motion sequence of successive jump by concatenating a landing motion and jumping motion. (b) Our method concatenates motion clips via a spline interpolation in a low-dimensional subspace. (c) Concatenation with short quadratic spline provides a smooth and rapid motion transition. (d) Concatenation with longer quartic spline emphasizes squatting-down and swinging arms, both of which are redundant movement in terms of smooth transition but well exaggerate the jumping.*

Abstract

We develop a novel system to concatenate motions with a smooth transition involving natural, redundant movements which cannot be obtained using traditional motion blending technique. Our basic idea is to apply a spline interpolation in a linear subspace in which each input motion is represented as a low-dimensional curve. By connecting the curves via a spline interpolation, a transition motion is synthesized according to a shape of the spline curve while preserving correlations among joints of input motions. Our system allows users to create a variety of transition motions with simple control of the spline parameters.

1 Introduction

Motion concatenation technique is widely used for generating a long stream of character animation from a collection of short motion clips. Most existing methods employ a problem formulation of energy minimization to create a smooth transition from one clip to another. A common approach creates a transition motion by blending input motions with an ease-in and ease-out interpolation. The blending-based method generates a transition motion so as to minimize dissimilarity between the transition motion and the input motions. The dissimilarity is further reduced by optimizing a transition length [Wang and Bodenheimer 2008] or by using dynamic time-warping [Kovar and Gleicher 2003], both of which reduce temporal misalignment of the input motions. Another approach uses a space-time optimization to generate a transition motion based on physical low [Rose et al. 1996]. The spacetime method minimizes joint torque required for the transition, which results in a physically realistic motion. These energy-minimization techniques successfully generate a smooth and rapid motion transition.

However, a natural transition is sometimes obtained with *redundant* movement. Jumping motion for example, a blending-based technique generates a rapid transition from landing to jumping. Moreover, we can make it more visually appealing by adding two movements; squatting down after landing and taking arms back before jumping, which correspond to a basic principle of 'follow-thorough' and that of 'anticipation', respectively. These movements are effective to exaggerate the jumping motion although they are redundant in terms of efficient transition. Existing methods, however, are not applicable to generate such redundant movement because they minimize a certain type of energy function.

The goal of our work is to develop a motion concatenation technique to generate a natural transition which involves redundant movement. In our preliminary experiment, we used a spline interpolation for concatenating motion curves of each joint. By controlling a shape of the interpolation curve with spline parameters, we could obtain redundant movements of joint rotation such as overshoot and oscillation. Such a per-joint operation, however, often broke correlations among joints and damaged original style of input motions.

Inspired by [Ye and Liu 2010], we developed a system to concatenate motions via a spline interpolation in a low-dimensional, linear subspace in which each input motion is represented as a low-dimensional curve as shown in Figure 1(b). The spline interpolation is used to connect the low-dimensional curves, and a transition motion is reconstructed from the connecting spline curve. We can create a variety of transition motions by changing a shape of the spline curve as shown in Figure 1(c) and (d). Arbitrary shape of the spline generates a possible transition because the subspace is constructed to reflect correlations among joints of input motions. Consequently, our system provides a flexible generation of natural motion transition which involves redundant movements. Users can easily control the degree of the redundancy with spline parameters while visually checking the shape of the spline in the subspace.

2 Technical approach

Our system first learns a low-dimensional subspace of input motions. We use classical multidimensional scaling (CMDS) to learn a linear subspace which reflects essential differences among the input motions [Shin and Lee 2006]. The low-dimensional input motions are then concatenated using a spline interpolation, where the degree and length of the spline are manually specified. A transition

*e-mail: tmki@acm.org

motion is finally synthesized by an inverse mapping of the spline curve from the subspace to high-dimensional motion space.

2.1 Preliminaries

We here denote a motion sequence as $\mathbf{q}(t), t \in \{1, \cdots, T\}$ where T denotes the number of frames of the motion. A pose vector $\mathbf{q}(t)$ consists of root position $\mathbf{p}(t)$, root orientation $\mathbf{o}(t)$ and joint rotations $\mathbf{r}_j(t)$, $j \in J$ where J is a set of joint nodes. The root orientation $\mathbf{o}(t)$ is decomposed into two components as $\mathbf{o}(t) = \mathbf{o}_{\psi+\theta}(t)\mathbf{o}_{\phi}(t)$ where ψ, θ and ϕ represents roll, pitch and yaw angle, respectively.

Given two input motions $\mathbf{q}_A(t_a), t_a \in \{1, \cdots, T_A\}$ and $\mathbf{q}_B(t), t_b \in \{1, \cdots, T_B\}$, a transition segment is determined by manually specifying at which frame a transition begins in $\mathbf{q}_A(t_a)$, denoted by τ_A, and at which frame the transition ends in $\mathbf{q}_B(t_b)$, denoted by τ_B. A transition motion $\mathbf{q}_C(t_c), t_c \in \{1, \cdots, T_c\}$ is generated so as to smoothly transit from $\mathbf{q}_A(\tau_A)$ to $\mathbf{q}_B(\tau_B)$. The transition length T_c is automatically determined according to the shape of the interpolation curve as described in §2.3.

2.2 Dimensionality reduction

We use CMDS to project input motions into a linear subspace in which dissimilarity between two poses is approximated by Euclidean distance between projected points. We define a dissimilarity measure based on position and velocity of all joints so that more similar poses are projected nearer locations in the subspace. As the result, input motions $\mathbf{q}_A(t_a)$ and $\mathbf{q}_B(t_b)$ are approximated by low-dimensional curves $\mathbf{x}_A(t_a)$ and $\mathbf{x}_B(t_b)$, respectively. All vertices of two curves $\mathbf{x}_A(t_a)$ and $\mathbf{x}_B(t_b)$ are merged into a collection of sample points $\mathbf{x}_s, s \in \{1, \cdots, S\}$ where $S = T_A + T_B$.

Each sample point $\mathbf{x}_s$ has several embedded data required to reconstruct a transition motion from the low-dimensional curve. A set of embedded data consists of joint rotation $\mathbf{r}_{j,s}$, root orientation $\mathbf{o}_{\psi+\theta,s}$, root velocity $\dot{\mathbf{p}}_s$, angular velocity of yaw angle $\dot{\phi}_s$, and temporal derivative of the low-dimensional curve $\|\dot{\mathbf{x}}_s\|$. The last component is added to reconstruct a transition motion with appropriate motion speed as described in the following subsection.

2.3 Spline interpolation and motion synthesis

We use a Bézier spline to connect the low-dimensional curves. Given degree and length of the spline, a smooth curve $\mathbf{h}(\alpha), 0 \leq \alpha \leq 1.0$ is automatically generated under boundary conditions $\mathbf{h}(0) = \mathbf{x}_A(\tau_A)$, $\dot{\mathbf{h}}(0) = \dot{\mathbf{x}}_A(\tau_A)$, $\mathbf{h}(1.0) = \mathbf{x}_B(\tau_B)$ and $\dot{\mathbf{h}}(1.0) = \dot{\mathbf{x}}_B(\tau_B)$.

A transition motion $\mathbf{q}_C(t_c)$ is synthesized from the spline curve $\mathbf{h}(\alpha)$. We use an incremental algorithm to trace the spline curve with a proper step length. The transition length T_c is therefore automatically determined as the result of the incremental tracing. We also use a linear regression technique to reconstruct a pose vector at each point on the spline.

The spline tracing starts with $\alpha = 0$ and $t_c = 1$. Embedded data at $\mathbf{h}(\alpha)$, which are $\mathbf{r}_j(t_c)$, $\mathbf{o}_{\psi+\theta}(t_c)$, $\|\dot{\mathbf{p}}(t_c)\|$, $\dot{\phi}(t_c)$ and $\|\dot{\mathbf{x}}(t_c)\|$, are synthesized using a linear interpolation of all samples. For instance, $\mathbf{r}_{C,j}(t_c)$ is obtained by $\mathbf{r}_{C,j}(t_c) = \sum_{s=1}^{S} w_s(\mathbf{h}(\alpha))\mathbf{r}_{j,s}$. The interpolation weight $w_s(\mathbf{h}(\alpha))$ is optimized using a simplified version of kriging [Mukai and Kuriyama 2005] as follows:

$$\begin{bmatrix} \mathbf{w} \\ \lambda \end{bmatrix} = \begin{bmatrix} \mathbf{X} & \mathbf{1}_S, \\ {}^t\mathbf{1}_S & 0 \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{\chi} \\ 1 \end{bmatrix}, \tag{1}$$

where $\mathbf{w} = {}^t[w_1 \ \cdots \ w_S \ \lambda]$ is weight vector, $\mathbf{X}$ is a $S \times S$ distance matrix in which $\mathbf{X}_{i,j} = \|\mathbf{x}_i - \mathbf{x}_j\|$, $\boldsymbol{\chi}$ is a S dimensional distance vector in which $\boldsymbol{\chi}_s = \|\mathbf{x}_s - \mathbf{h}(\alpha)\|$, $\mathbf{1}_S$ is a column vector of S ones, λ is a Lagrange multiplier, and the last element of the vector in the right-hand side constrains the weights w_s sum up to one. The optimized weight is used for interpolating each embedded data, and a pose vector $\mathbf{q}_C(t_c)$ is composed of all embedded data, where $\mathbf{p}(t_c)$ and $\phi(t_c)$ are obtained by time integration. The spline tracing is then advanced with $\alpha' \leftarrow \alpha + \triangle\alpha$ and $t'_c \leftarrow t_c + 1$ until reaching the end of the spline. The step length $\triangle\alpha$ is optimized to satisfy $\int_{\alpha}^{\alpha+\triangle\alpha} \|\dot{\mathbf{h}}(u)\| du = \|\dot{\mathbf{x}}(t_c)\|$ for reflecting the speed of input motions.

After finishing the tracing, undesired motion artifacts such as footskating are reduced using a stock IK solver in a post process.

3 Experimental results

We create a motion sequence of successive jumping by concatenating a clip of landing and that of jumping. Our system takes a few seconds for the dimensionality reduction, and all other processes are executed at interactive rates. A short quadratic spline results in a smooth and rapid transition as a blending-based method provides (Figure 1(c)). In contrast, a longer quartic spline emphasizes a swing of arms (Figure 1(d)). Although this result is hard to be predicted just by checking the shape of the low-dimensional curves, we can easily achieve a desirable result through the interactive control of spline parameters and visual check of the corresponding curve shapes. Please see the accompanying video for further examples of swinging sword.

4 Conclusion

This paper presents a new system to concatenate motions using spline interpolation in low-dimensional subspaces. We can generate a variety of transition motion involving redundant movements, where the redundancy can be controlled with spline parameters. Another advantage of our system is that there is no need for an overlap between input motions whereas a blending-based method requires that both input motions have a sufficient margin to be blended. A major limitation is that our data-driven method cannot generate a pose that does not appear in the input motions.

Our system provides side-by-side visualization of shapes of low-dimensional curves and a resulting animation. This helps user to see a relation between the shape of the spline and the synthesized motion. However, our system does not provide a function to directly sketch or manipulate the curve on the screen because we experimentally confirmed that such direct editing is not intuitive.

Our future work includes an investigation of more efficient subspace algorithm applicable to real-time applications such as games. User testing is also important for evaluating the usability of the interactive system.

Acknowledgement

We thank Yutaka Miyauchi and Motion Capture Group at Square Enix for their assistance in the motion capture session.

References

Kovar, L., and Gleicher, M. 2003. Flexible automatic motion blending with registration curves. In *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation 2003*, 214–224.

Mukai, T., and Kuriyama, S. 2005. Geostatistical motion interpolation. *ACM Transactions on Graphics 24*, 3, 1062–1070.

Rose, C., Guenter, B., Bodenheimer, B., and Cohen, M. F. 1996. Efficient generation of motion transitions using spacetime constraints. In *Proc. of SIGGRAPH 96*, 147–154.

Shin, H. J., and Lee, J. 2006. Motion synthesis and editing in low-dimensional spaces. *Computer Animation and Virtual Worlds 17*, 3-4, 219–227.

Wang, J., and Bodenheimer, B. 2008. Synthesis and evaluation of linear motion transitions. *ACM Trans. on Graphics 27*, 1, 1.

Ye, Y., and Liu, C. K. 2010. Synthesis of responsive motion using a dynamic model. *Computer Graphics Forum 29*, 2, 555–562.