On-line Motion Style Transfer

Xiaomao Wu[+], Lizhuang Ma[+], Can Zheng[+], Yanyun Chen*, and Ke-Sen Huang

[+]Department of Computer Science & Engineering, Shanghai Jiao Tong University
No. 800, Dongchuan Road, Shanghai 200240, P. R. China
wu.xiaomao@gmail.com, ma-lz@cs.sjtu.edu.cn, z_oasis@sjtu.edu.cn

*Microsoft Research Asia
No. 49, Zhichun Road, Haidian District, Beijing 100080, P. R. China.
yachen@microsoft.com

*Department of Computer Science
National Tsing Hua University
101, Kuang Fu Rd, Sec.2
HsingChu, Taiwan 300 R. O. C
kesen.huang@gmail.com

Abstract. Motion capture techniques play an important role in computer animation. Because the cost of motion capture data is relatively high and the virtual environment changes frequently in actual applications, researchers in this area focus their work on developing algorithms for editing the capture motion data, and synthesizing new motions from available motion database. Although abundant work has been done on motion editing and synthesis, few of them obviously take motion styles into consideration. Meanwhile, existing style editing algorithms either need an obvious definition of "style", or need a time-consuming training process. In this paper, we propose a fast and convenient algorithm for human-motion style editing. We define the style of motion as statistic properties of mean and standard variance of joint quaternions in 4D unit sphere space. The proposed algorithm can transfer the style of a motion to another by transferring these properties. Experiment results demonstrate that our approach has the advantages of fast execution, low memory occupation, and easy implementation. It can be widely applied to various real-time entertainment-computing applications, such as gaming and digital movie producing.

Keywords: motion editing, motion style, style transfer, quaternion mean and variance

1 Introduction

Motion capture techniques have been widely adopted in digital movie producing, gaming and other digital entertainment applications. Creating animations with motion capture techniques commonly consists of the following three steps: First, acquire 3D motion data from real actors or actress with available commercial motion capture devices. Second, map the acquired data onto virtual characters. Finally, edit and fine-tune the mapped motion until satisfactory results are obtained. Compared with traditional key-framing techniques and procedure-based methods, motion capture techniques provide a more reliable and convenient way for creating realistic character animation.

The key issue with motion capture techniques is: how to process the captured motion data so that it can satisfy the constraints of specific applications. Abundant work has been done on this issue. We can classify them into two categories: motion editing and motion synthesis. Among those works, few of them take motion style into consideration, despite that motion style is an important issue for motion editing and synthesis. Meanwhile, existing motion style editing approaches either rely on procedural definition of the motion styles [1,2,3], or explicit input of style parameters of the user [2,3], or even a model-training process [4,5].

In this paper, we propose a new on-line algorithm for motion style transfer. The proposed algorithm does not need any user's input, or expensive training process. The basic idea of the proposed algorithm is to encode motion style in a statistic distribution model, and transfer the style of one motion to another by transferring the according model parameters. As illustrated in Fig. 1, the proposed approach in this paper can transfer the style of the reference motion $\mathcal{M}_r$ to the source motion $\mathcal{M}_s$, resulting in a new motion $\mathcal{M}_t$ which preserves the motion details of $\mathcal{M}_s$ and inherits the style of $\mathcal{M}_r$. For example, we can transfer the "stride" style from a stride-working motion $\mathcal{M}_r$ to a normal running motion $\mathcal{M}_s$, resulting a new "stride running" motion $\mathcal{M}_t$.

Among the existing motion style editing approaches, Hsu et al.'s [5] work is most related to ours. Our work differs from theirs in the following aspects: First, Hsu et al. use two motions (one basic motion and one motion with style) for defining a motion style, while we use only one motion and abstract the style directly from that motion. This difference is very similar to the difference between the work of Reinhard [6]'s and of Hertzmann [7]'s in the field of image processing. Another difference between our work and Hsu's is that their method use time-consuming N4SID algorithm to train a LTI (linear time identification) model. Our approach does not need this process, thus can transfer the style from one motion to another on the fly.

2 Related work

Many techniques have been proposed for modifying captured motion data so that it can meet different requirements of actual applications. In general, these techniques can be classified into two categories: motion editing and motion synthesis. Here we only review motion editing techniques because it is more related to our work.

Spline fitting [8,9], signal processing [10] and constrained optimization [11,12,13] have been successfully applied to motion editing. Although these methods are useful for editing kinematic and dynamic properties of motions, they do not explicitly take motion style into consideration.

Among the motion editing techniques, motion retargetting mainly focuses on designing a mapping function which can be used to map the motion of a figure to another one who has identical skeleton topology but different segment length [14,15,16], or even has different skeleton topology [17,18] as the first one. Our work differs from them in that their work focus on processing kinematic and dynamic constraints when retargetting a motion of a character to another, while our work concentrates on transferring the style from one motion to another. Meanwhile, motion retargetting techniques treat style and motion as a whole, and do not treat the style of motion separately. Our method separates style from motion, and treats style and motion details as two different layers.

Comparing to motion editing techniques, few work have been done for motion style editing. Amaya et al. [1] introduce an algorithm which can transform the emotion of a motion to another. Unuma et al. [2] successfully use Fourier series expansions of the motion data sets to do expansion, smooth transition, interpolation, and even extrapolation between different types of motions. Brand et al. [4]'s style machine can be trained to create motions with different styles. Urtasun et al. [3] propose an algorithm which can be used to change the style of a motion by projecting it onto a PCA space generated by pre-existing motion data. More recently, Hsu et al. [5] use linear space identification analysis to obtain a style translation model between two motions.

Style editing is also explored by researchers in the filed of image processing. Two representative image style transfer approaches are respectively proposed by Reinhard et al. [6] and Hertzmann et al. [7]. Our work is inspired by Reinhard's work [6] which has obtained convincing results for transferring the color characteristics of one image onto another with an efficient statistical analysis.

3 Overview

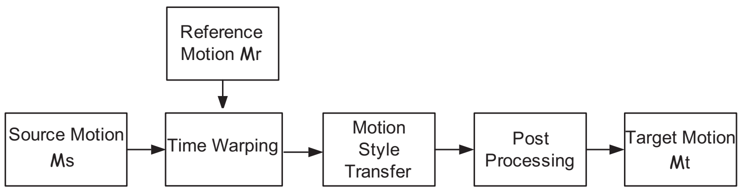

Fig. 1: Overview of our approach

The overview of our approach is illustrated in Fig. 1. The goal of our approach is to transfer the style of the reference motion $\mathcal{M}_r$ to the source motion $\mathcal{M}_s$, producing a target motion $\mathcal{M}_t$. The target motion can preserve the details of the source motion and can inherit the style of the reference motion. The style transfer pipeline works as follows: First, $\mathcal{M}_s$ and $\mathcal{M}_r$ are input directly into the time warping module to setup frame correspondence. Then, the time-warped motions are fed into the statistic style transfer module for style transferring. In the post-processing stage, we apply a reverse-time-warping to the target motion $\mathcal{M}_t$ and then remove artifacts such as footskate which is introduced during the style translation process. After that, we can obtain the target motion that inherits the style of the reference motion and preserve the details of the source motion.

4 Time Warping

The frame correspondences are vitally important for motion style transfer. Without proper frame correspondence, unnatural results may be produced. We use the IMW (iterative motion warping) algorithm proposed by Hsu et al. [5], because it performs better then other proposed DTW (dynamic time warping) algorithms when dealing with stylistically different motions.

The IMW algorithm process one DOF (degree of freedom) each time. Suppose the frame numbers of the source motion $\mathcal{M}_s$ and the reference motion $\mathcal{M}_r$ are N_s and N_r respectively. We further suppose that $N_s < N_r$. When $N_s \geq N_r$, we only need to exchange the position of $\mathbf{S}$ and $\mathbf{r}$ in equation 1. The IMW algorithm aims at minimizing the following energy equation:

$$E(\mathbf{a}, \mathbf{b}, \mathbf{W}) = \|\mathbf{W}(\mathbf{S}\mathbf{a} + \mathbf{b}) - \mathbf{r}\|^2 + \|\mathbf{F}\mathbf{a}\|^2 + \|\mathbf{F}\mathbf{b}\|^2 + \|\mathbf{G}\mathbf{b}\|^2 \tag{1}$$

Here, $\mathbf{S}$ is a diag(s), whose diagnose elements contains one DOF data of $\mathcal{M}_s$. $\mathbf{W}$ is a *warp matrix* used to perform a nonuniform time warp. The terms $\|\mathbf{F}\mathbf{a}\|^2$ and $\|\mathbf{G}\mathbf{b}\|^2$ measure the smoothness of $\mathbf{a}$ and $\mathbf{b}$. $\mathbf{F}$ and $\mathbf{G}$ provide weighted finite-difference approximations to the first derivative of $\mathbf{a}$ and $\mathbf{b}$.

The IMW algorithm works as following:

First, initialize $\mathbf{a} = \mathbf{1}$ and $\mathbf{b} = \mathbf{0}$.

Second, calculate matrix $\mathbf{W}$ with dynamic time warping, i. e., solve the following linear equation:

$$\min_{\mathbf{W}} \|\mathbf{W}\mathbf{p} - \mathbf{q}\|^2 = \min_{w} \left\| \begin{bmatrix} l_{w1} & & & \\ & l_{w2} & & \\ & & \cdots & \\ & & & l_{wN_s} \end{bmatrix} \mathbf{p} - \mathbf{q} \right\|^2 \tag{2}$$

where $\mathbf{p} = \mathbf{S}\mathbf{a} + \mathbf{b}$ and $\mathbf{q} = \mathbf{r}$.

Third, solve scale vector $\mathbf{a}$ and offset vector $\mathbf{b}$ with the following equation which is transformed from equation 1:

$$\begin{bmatrix} \mathbf{S}^T\mathbf{W}^T\mathbf{W}\mathbf{S} + \mathbf{F}^T\mathbf{F} & \mathbf{S}^T\mathbf{W}^T\mathbf{W} \\ \mathbf{W}^T\mathbf{W}\mathbf{S} & \mathbf{W}^T\mathbf{W} + \mathbf{G}^T\mathbf{G} \end{bmatrix} \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix} = \begin{bmatrix} \mathbf{S}^T\mathbf{W}^T\mathbf{r} \\ \mathbf{W}^T\mathbf{r} \end{bmatrix} \tag{3}$$

The above steps are performed iteratively until the change of $\mathbf{a}$ and $\mathbf{b}$ between two consecutive iteration is below a pre-defined threshold. For more detailed description of the IMW algorithm, please refer to the original paper [5].

5 Statistic style transfer

In this section, we first introduce the representation of the motion and the statistic components which will be utilized for designing the style transfer pipeline. And then, we introduce the style transfer algorithm for the root joint, followed by a detailed description of the style transfer algorithm in the Euclidean space and in the 4D quaternion space. For convenience and clarity of description, we only discuss one joint in this section. Other joints can be processed in the same way, except that the root joint should be considered separately because it contains global translation components.

5.1 Motion Representation and Style Definition

A motion can be denoted by $\mathbf{m}(t) = (\mathbf{p}(t), \mathbf{q}^1(t), ..., \mathbf{q}^n(t))^T$, where $\mathbf{p}(t) \in \mathbb{R}^3$ and $\mathbf{q}^1(t) \in \mathbb{S}^3$ represent the translation and rotation of the root joint. $\mathbf{q}^i(t) \in \mathbb{S}^3$ denotes the rotation of the i-th joint for $2 \leq i \leq n$.

In the research community of image processing, Reighar et al. [6] define the style of an image as the mean and standard variance of color components in a linearized color space $l\alpha\beta$, and has successfully transfer the style of an image to another by transferring the mean and standard variance of color components in this space. Inspired by Reighar's work, we also use mean and standard variance to represent the style of a motion. In the remaining part of this paper, we briefly describe "standard variance" as "variance". With this representation, motion style can be transferred by modifying the mean and variance of the source motion $\mathcal{M}_s$ according to that of the reference motion $\mathcal{M}_r$. However, our experiment results demonstrate that straightforwardly apply the style transfer algorithm to Euler angles produces poor results, while applying the algorithm to quaternion domain can give us smooth results. And since the mean and variance have no unified meanings for quaternions, we give our definition of quaternion mean and variance in Section 5.4.

In our implementation, we assume that each joint is independent of the others, following [10,8,5]. While doing the style transfer, we translate one joint at a time. All joints are passed through the style transfer stage one by one.

5.2 Transfer for the Root Joint

We treat the root joint separately because it encodes a transformation with respect to the global coordinate system. The translation part of the root joint is translated separately for each dimension. Because motion style is invariant to ground plane translation, we encode the translation of the root joint of each frame in its previous frame [5,19]. We first calculate the mean and variance of the translations of the root joints of the source motion $\mathcal{M}_s$ and the reference motion $\mathcal{M}_r$ respectively, and then transfer each of the three translation DOFs as:

$$\mathbf{p}_t^i = \sigma_r/\sigma_s \cdot (\mathbf{p}_s^i - \bar{\mathbf{p}}_s) + \bar{\mathbf{p}}_r, 1 \leq i \leq N_s \tag{4}$$

where N_s is the frame number of $\mathcal{M}_s$ after time warping. $\mathbf{p}_s^i$ and $\mathbf{p}_t^i$ denote the i-th DOF of the root joint of $\mathcal{M}_s$ and $\mathcal{M}_t$. $\bar{\mathbf{p}}_s$ and $\bar{\mathbf{p}}_r$ denote the mean of the root joint angle of $\mathcal{M}_s$ and $\mathcal{M}_t$. σ_s and σ_r denote the variance of the root joint of $\mathcal{M}_s$ and $\mathcal{M}_t$, respectively. The vertical rotation is preserved by encoding the root orientation in the previous frame, as suggested by Kovar [19] and Hsu [5].

5.3 Transfer with Euler Angles

Transfer with Euler angles is straightforward. Our algorithm transfers each DOF individually. Specifically, let $\bar{\theta}_s$ and σ_s denote the mean and variance of the source motion $\mathcal{M}_s$, and $\bar{\theta}_r$ and σ_r denote the mean and variance of the reference motion $\mathcal{M}_r$. The transferred angle is calculated as

$$\theta_t = \sigma_r/\sigma_s \cdot (\theta_s - \bar{\theta}_s) + \bar{\theta}_r \tag{5}$$

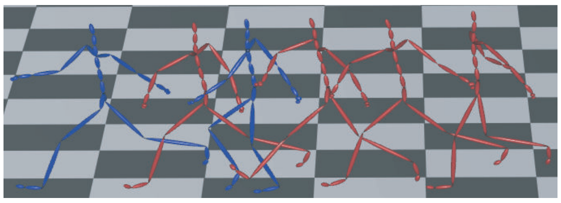

Fig. 2: Style translation with Euler angles produces unnatural results, shown in red.

where θ_t represents the transferred angle of the target motion $\mathcal{M}_t$.

Our experiments demonstrate that style transfer with Euler angles does not produce smooth results in most cases (Fig. 2), because interpolation of Euler angles may result in poor rotations [20].

5.4 Transfer with Quaternions

Representing rotations and orientations by quaternions is widely adopted in computer animation as it is free of gimbal lock and has good interpolation behavior [21,22,23,9]. Quaternion is also an ideal representation for our style transfer operation. However, we should carefully choose the meaning of “mean” and “variance” in quaternion space. Although quaternions can be linearly interpolated and we can define the mean with linear interpolation. This definition will result in inconsistent velocities [24]. By using a recursive definition based on the sphere linear interpolation ($Slerp$) [21], we can obtain a well defined centroid, or the mean of a sequence of quaternions. Meanwhile, we define the distance between two quaternions on the 4-D hypersphere, and use this definition to derive the variance of quaternions.

Mean and Variance of Quaternions Let $\mathbf{q}_i$ for $0 \leq i \leq n$ denotes a sequence of n quaternions. The mean of $\mathbf{q}_i$s can be recursively defined as:

$$\bar{\mathbf{q}}_n = \begin{cases} \mathbf{q}_1 \text{ }, for\, n = 1 \\ Slerp(\frac{1}{2}, \mathbf{q}_1, \mathbf{q}_2), for\ n = 2 \\ Slerp(\frac{n-1}{n}, \bar{\mathbf{q}}_{n-1}, \mathbf{q}_n), for\ n \geq 3 \end{cases} \tag{6}$$

Here, $Slerp$ is a sphere linear interpolation function [21] which can be defined by: $Slerp(t, \mathbf{q}_1, \mathbf{q}_2) = \frac{\sin(1-t)\theta}{\sin\theta}\mathbf{q}_1 + \frac{\sin u\theta}{\sin\theta}\mathbf{q}_2$, $\theta = \cos^{-1}(\mathbf{q}_1 \cdot \mathbf{q}_2)$. When doing $Slerp$ between two quaternions $\mathbf{q}_1$ and $\mathbf{q}_2$ during the mean calculation process, one should make sure that the angle between $\mathbf{q}_1$ and $\mathbf{q}_2$ are not larger than $\frac{\pi}{2}$ for consistent representation. This can be done by checking the sign of the dot product of two quaternions $d = \mathbf{q}_1 \cdot \mathbf{q}_2$. If $d < 0$, we reverse the sign of $\mathbf{q}_1$, because $\mathbf{q}_1$ and $-\mathbf{q}_1$ represent the same rotation.

We should mention that besides this definition, other definitions can also be adopted for calculating the mean of quaternions, because mean is a special case of weighted sum. We use the above definition because it is simple and efficient, and can produce reasonable results. Other more rigorous methods exist, such as global linearization [23] and functional optimization [25]. Those algorithms follow more rigorous definition of mean of quaternions, but are more computationally expensive. We have found that equation 6 works well in all of our experiments.

The variance of quaternions is defined similar to its definition for real numbers, except that the distance is defined in $\mathbb{S}^3$ as: $dist(\mathbf{q}_1, \mathbf{q}_2) = ||log(\mathbf{q}_1^{-1}, \mathbf{q}_2)||$. The variances of quaternion sequence of the joint of the source and reference motion, denoted by σ_s and σ_t, are calculated under this definition.

Transfer Algorithm With the definition of mean and variance of quaternion, we can calculate the mean and variance of the source motion, denoted by $\bar{\mathbf{q}}_s$ and σ_s, and that of the reference motion, denoted by $\bar{\mathbf{q}}_r$ and σ_r.

The first step of motion style transfer is to transfer the mean of the reference motion to the source motion. This can be implemented by applying a $\mathbf{q}^T$ transformation (Fig. 3). $\mathbf{q}^T$ is defined by: $\mathbf{q}^T = (\bar{\mathbf{q}}_s)^{-1} * \bar{\mathbf{q}}_r$.

By applying transformation $\mathbf{q}^\mathbf{T}$ to all the quaternion sequence of the source motion$\mathcal{M}_s$, we can obtain the aligned version of quaternions whose mean is equal to that of the reference motion $\mathcal{M}_r$.

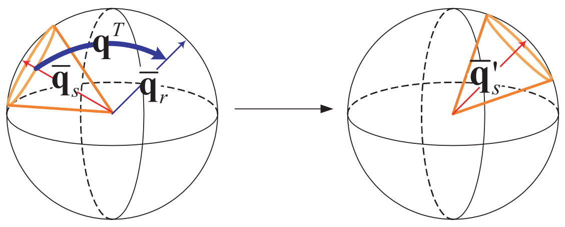

Fig. 3: The mean of the joint quaternions of the source motion, denoted by $\bar{\mathbf{q}}_\mathbf{s}$, is aligned with that of the reference motion $\bar{\mathbf{q}}_\mathbf{r}$ by applying a transform $\mathbf{q}^T$.

Finally, we scale the variance of the joint of the source motion according to that of the reference motion by the following equation:

$$\mathbf{q}_t^i = Slerp(\frac{\sigma_r}{\sigma_s}, \bar{\mathbf{q}}_r, \mathbf{q}_s^{i'}) \tag{7}$$

where $\mathbf{q}_s^{i'}$ and $\mathbf{q}_t^i$ are the i-th quaternion of the joint of the source motion $\mathcal{M}_s$ and the target motion $\mathcal{M}_t$.

6 Post-Processing

The purpose of post-process stage is to reverse the time warping effect and process kinematic constraints.

In the reverse-time-warping step, we apply a $Slerp$ for the adjusted frames, in order to bring back the normal timing.

In the kinematic constraints processing step, we apply the footskate cleaup algorithm proposed by Kovar et al. [26] to $\mathcal{M}_t$ in order to obtain correct and smooth results. The footskate cleanup algorithm works as follows: First, the position of each footplant constraint is calculated. For each constrained frame, compute the ankle's global positions and orientations. Then we determine where the root should be placed. For each constrained ankle, adjust the leg so the ankle meets the configurations found previously. Finally, we filter the motion in order to obtain smooth results. Readers are referred to the original paper [26] for detailed description.

7 Results

Table 1: The running time of the main steps for four examples in our experiment.

Process (source-reference)	Frame number (source-reference)	Time warping (milliseconds)	Transfer time (milliseconds)	Post-processing time (milliseconds)
Normal walk - Hobble walk	427-400	900	21	80
Normal walk - Stealthy walk	277-410	600	17	61
Normal running - Stride walk	135-302	300	10	22
Normal running - Jaunty walk	135-450	400	13	25

We have tested our algorithm on several captured motion pairs in order to demonstrate its validity and efficiency. All the motion clips are obtained from CMU motion capture library [27].

In the first example, we transfer the "hobble" style of a walk motion to a normal walk motion (Fig. 4), producing a new "hobble walk" motion. In the second example, we transfer the "stealthy" style of a walk motion to a normal walk motion (Fig. 5), producing a new "stealthy walk" motion. In the third example, we transfer "stride" style of a walk motion to a running motion, producing a "stride running" motion (Fig. 6). In the fourth example, we transfer the "jaunty" style of a walk motion to a running motion, producing a "jaunty running" motion (Fig. 7).

All the experiments were done on a Pentium 4 2.8G PC with 1.5GB memory. The computation time are listed in Table 1. The memory usage is negligible, because we only need to additionally store the time-warping parameters and the mean and variance of the source and reference motion during the style transfer process.

According to the experimental results, our approach runs very fast, occupies negligible memory, which makes it suitable for real-time motion editing applications.

8 Conclusion and Discussion

In this paper, we propose a fast and memory efficient algorithm for transferring the style of a motion to another motion. Comparing to existing style transfer approaches, our approach is simple to implement, runs very fast and occupies negligible memory, making it suitable for interactive applications.

Compared with the state-of-the-art work of Hsu's [5], our approach is more suitable for fast prototyping. One may argue that when the training process has been carried out with N4SID, Hsu's LTI model can achieve very fast transfer speed. But the training the LTI model is a time-consuming time, and if the training data is not enough, i. e., the frame number is relatively small, one may not obtain satisfactory results. Meanwhile, the N4SID model is sensitive to input parameters. This could be a problem with inexperienced users. Our approach do not depends on any user input, and can run automatically, with rather fast speed.

Currently, our approach can not be applied to transfer style between figures that do not have identical structures. Existing motion retargetting algorithms [18,28] may be helpful for solving this problem. Meanwhile, when the reference has several different styles, e. g. hobble at the beginning then stride in the middle and stealthily run at the end, then the reference motion should be segmented into several segments so that each segment has a single distinguishable style.

Acknowledgements

We would like to thank the Graphics Lab. of Carnegie Mellon University for generously providing their motion capture data on their web. This project was supported by Supported by National Science Fund for Creative Research Groups (grant No. 60521002) and National Natural Science Foundation of China (grant No. 60373070 and 60573147).

References

1. Kenji Amaya, Armin Bruderlin, and Tom Calvert. Emotion from motion. In *Graphics Interface '96*, pages 222–229, 1996.
2. Munetoshi Unuma, Ken Anjyo, and Ryozo Takeuchi. Fourier principles for emotion-based human figure animation. In *SIGGRAPH '95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 91–96, 1995.
3. Raquel Urtasun, Pascal Glardon, Ronan Boulic, Daniel Thalmann, and Pascal Fua. Style-based motion synthesis. *Computer Graphics Forum*, 23(4):799–812, 2004.
4. Matthew Brand and Aaron Hertzmann. Style machines. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 183–192, 2000.
5. Eugene Hsu, Kari Pulli, and Jovan Popović. Style translation for human motion. *ACM Trans. Graph.*, 24(3):1082–1089, July 2005.
6. Erik Reinhard, Michael Ashikhmin, Bruce Gooch, and Peter Shirley. Color transfer between images. *IEEE Computer Graphics and Applications*, 21(5):34–41, Sep.-Oct. 2001.
7. Aaron Hertzmann, Charles E. Jacobs, Nuria Oliver, Brian Curless, and David H. Salesin. Image analogies. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 327–340, 2001.
8. Andrew Witkin and Zoran Popović. Motion warping. In *SIGGRAPH '95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 105–108, 1995.
9. Jehee Lee and Sung Yong Shin. A hierarchical approach to interactive motion editing for human-like figures. In *SIGGRAPH '99: Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 39–48, 1999.

10. Armin Bruderlin and Lance Williams. Motion signal processing. In *SIGGRAPH '95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 97–104, 1995.
11. Charles Rose, Brian Guenter, Bobby Bodenheimer, and Michael F. Cohen. Efficient generation of motion transitions using spacetime constraints. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 147–154, 1996.
12. Michael Gleicher. Motion editing with spacetime constraints. In *SI3D '97: Proceedings of the 1997 symposium on Interactive 3D graphics*, pages 139–148, 1997.
13. Anthony C. Fang and Nancy S. Pollard. Efficient synthesis of physically valid human motion. *ACM Trans. Graph.*, 22(3):417–426, 2003.
14. Michael Gleicher. Retargetting motion to new characters. In *SIGGRAPH '98: Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 33–42, 1998.
15. Kwang-Jin Choi and Hyeong-Seok Ko. Online motion retargetting. *Journal of Visualization and Computer Animation*, 11(5):223–235, 2000.
16. Hyun Joon Shin, Jehee Lee, Sung Yong Shin, and Michael Gleicher. Computer puppetry: An importance-based approach. *ACM Trans. Graph.*, 20(2):67–94, 2001.
17. Jean-Sébastien Monzani, Paolo Baerlocher, Ronan Boulic, and Daniel Thalmann. Using an intermediate skeleton and inverse kinematics for motion retargeting. *Compuuter Graphics Forum (Eurographics 2000)*, 19(3), 2000.
18. Min Je Park and Sung Yong Shin. Example-based motion cloning. *Journal of Computer Animation and Virtual Worlds*, 15(3-4):245–257, 2004.
19. Lucas Kovar and Michael Gleicher. Flexible automatic motion blending with registration curves. In *SCA '03: Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 214–224, 2003.
20. F. Sebastian Grassia. Practical parameterization of rotations using the exponential map. *Journal of graphics tools*, 3(3):29–48, 1998.
21. Ken Shoemake. Animating rotation with quaternion curves. In *SIGGRAPH '85: Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, pages 245–254, 1985.
22. Jehee Lee and Sung Yong Shin. A coordinate-invariant approach to multiresolution motion analysis. *Graphical Models*, 63(2):87–105, March 2001.
23. Sang Il Park, Hyun Joon Shin, and Sung Yong Shin. On-line locomotion generation based on motion blending. In *SCA '02: Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 105–111, 2002.
24. Erik B. Dam, Martin Koch, and Martin Lillholm. Quaternions, interpolation and animation. *DIKU technical report 98/5*, 1998.
25. Samuel R. Buss and Jay P. Fillmore. Spherical averages and applications to spherical splines and interpolation. *ACM Trans. Graph.*, 20(2):95–126, 2001.
26. Lucas Kovar, John Schreiner, and Michael Gleicher. Footskate cleanup for motion capture editing. In *SCA '02: Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 97–104, 2002.
27. CMU graphics lab motion capture database. *http://mocap.cs.cmu.edu/*.
28. Maher Moakher. Means and averaging in the group of rotations. *SIAM Journal on Matrix Analysis and Applications*, 24(1):1–16, 2002.

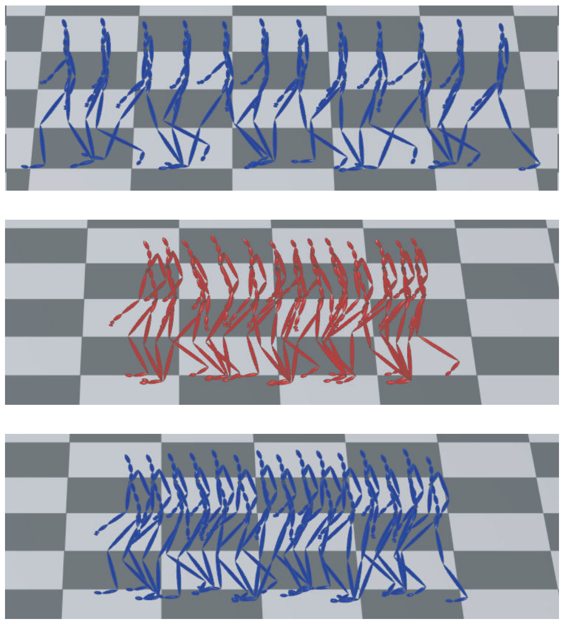

Fig. 4: Transfer the “hobble” style of a walking motion (middle) to another walking motion (top) produces a new “hobble walking” motion (bottom).

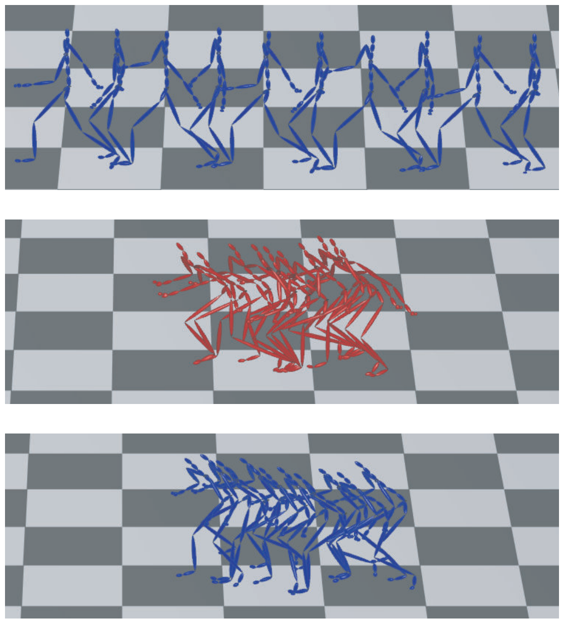

Fig. 5: Transfer the “stealthy” style of a walking motion (middle) to another walking motion (top) produces a new “stealthy walking” motion (bottom).

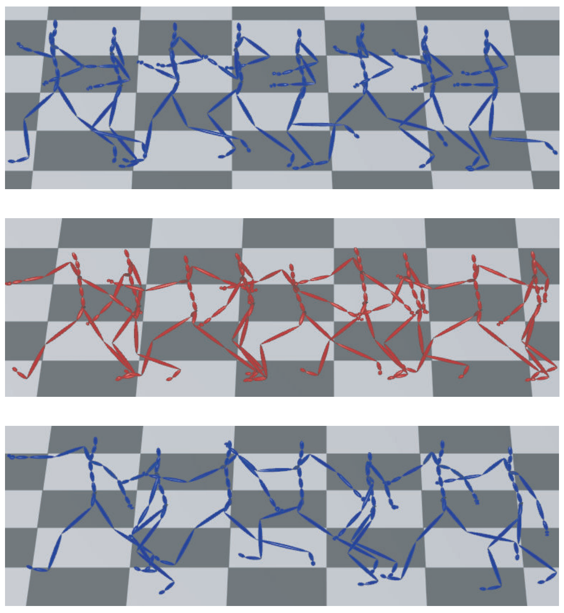

Fig. 6: Transfer a “stride” style of a walking motion (middle) to a running motion (top) produces a “stride running” motion (bottom).

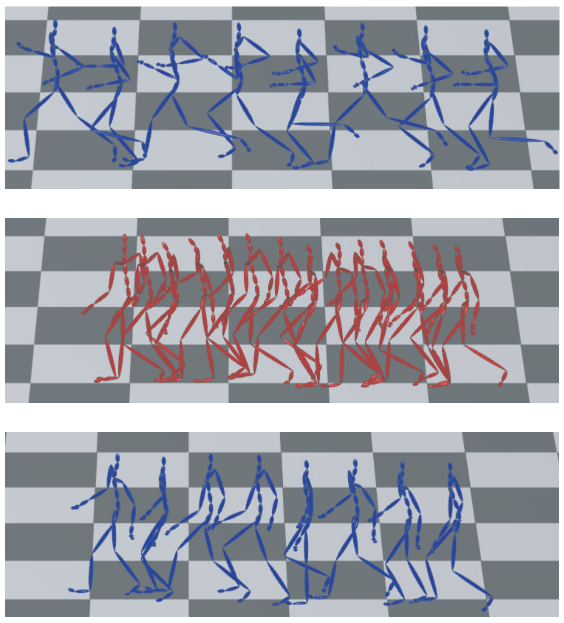

Fig. 7: Transfer the “jaunty” style of a walking motion (middle) to a running motion (top) produces a “jaunty running” motion (bottom).