Interactive Generation of Human Animation with Deformable Motion Models

JIANYUAN MIN, YEN-LIN CHEN, and JINXIANG CHAI
Texas A&M University

This article presents a new motion model *deformable motion models* for human motion modeling and synthesis. Our key idea is to apply statistical analysis techniques to a set of precaptured human motion data and construct a low-dimensional deformable motion model of the form $\mathbf{x} = M(\vec{\alpha}, \vec{\gamma})$, where the deformable parameters $\vec{\alpha}$ and $\vec{\gamma}$ control the motion's geometric and timing variations, respectively. To generate a desired animation, we continuously adjust the deformable parameters' values to match various forms of user-specified constraints. Mathematically, we formulate the constraint-based motion synthesis problem in a Maximum A Posteriori (MAP) framework by estimating the most likely deformable parameters from the user's input. We demonstrate the power and flexibility of our approach by exploring two interactive and easy-to-use interfaces for human motion generation: direct manipulation interfaces and sketching interfaces.

Categories and Subject Descriptors: I.3.6 [**Computer Graphics**]: Methodology and Techniques—*Interaction techniques*; I.3.7 [**Computer Graphics**]: Three-Dimensional Graphics and Realism—*Animation*

General Terms: Algorithms, Design

Additional Key Words and Phrases: Character animation, data-driven animation, 3D animation interfaces, animation with constraints, statistical analysis and synthesis, optimization

ACM Reference Format:

Min, J., Chen, Y.-L., and Chai, J. 2009. Interactive generation of human animation with deformable motion models. ACM Trans. Graph. 29, 1, Article 9 (December 2009), 12 pages. DOI = 10.1145/1640443.1640452 http://doi.acm.org/10.1145/1640443.1640452

1. INTRODUCTION

A long-standing challenge in computer graphics is to build an interactive system that allows an inexperienced user to create a realistic human animation quickly and easily. However, building such a system presents two major challenges. First, the generated motions must be natural-looking. People are extremely adept at judging whether an animated motion appears realistic or not. A movement that accomplishes an intended task, for example, punching a specific point on a boxing bag, might be judged as unacceptable if it appears robotic, jerky, uncoordinated, or contains any noticeable visual artifacts such as foot sliding. The second challenge is to provide an easy-to-use interface for animation generation. To create such an interface, the system must be able to support various forms of user constraints in order to accommodate users with different skill levels. In addition, the algorithm must be fast enough that the interface appears responsive and the user remains engaged in the animation task.

One appealing solution to this problem is to construct statistical motion models from prerecorded human motion data and use the models to generalize the captured motion data for new tasks. Statistical motion models are advantageous for human motion synthesis because they are very compact, they can measure the naturalness of human motion, and they can be used to generate an *infinite* number of natural-looking motions that are not in precaptured motion data. However, current statistical motion models often lack global motion structural information (e.g., poses and timing of "left-toe-down" or "double-support"), and encode little or no information about environmental contact events (e.g., foot contact constraints). For graphics applications, this is always undesirable because it leads directly to noticeable visual artifacts (e.g., foot sliding) in output animation. In addition, this approach has not demonstrated that it can support various forms of user constraints at interactive frame rates, which significantly limits the impact and use of statistical motion models in graphics applications.

This article presents a new motion model we term *deformable motion models* for human motion analysis and synthesis. Our key idea is to apply statistical analysis techniques to a large set of *annotated* motion examples and construct a deformable motion model of the form $\mathbf{x} = M(\vec{\alpha}, \vec{\gamma})$ for particular human actions $\mathbf{x}$, where the deformable parameters $\vec{\alpha}$ and $\vec{\gamma}$ control the motion's geometric and timing variations, respectively. Deformable motion models provide a continuous, compact representation for allowable motion variations, but are specific enough not to allow arbitrary variations that are not similar to those seen in the training examples. In addition, the deformable motion models encode both global human motion structures and

Authors' address: Department of Computer Science and Engineering, Texas A & M University, College Station, TX 77843; J. Chai (corresponding author). email: jchai@cs.tamu.edu.

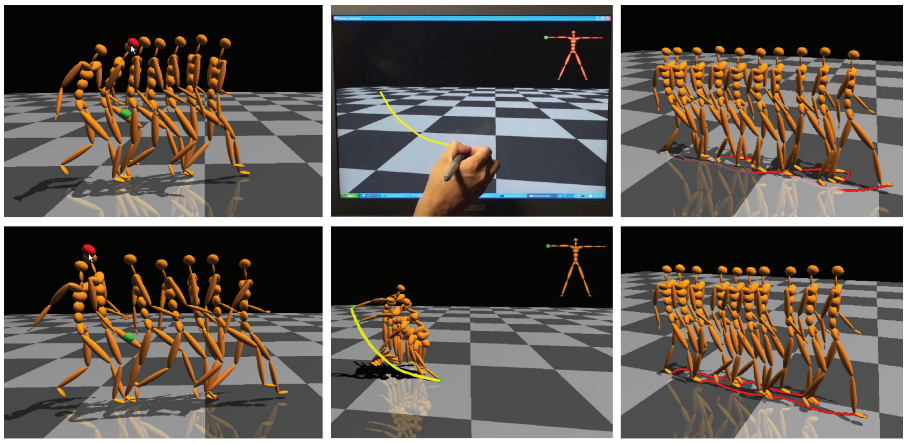


Fig. 1. Intuitive interactive motion generation with deformable motion models: (a) direct manipulation interfaces with point dragging (red point) and fixed handles (green point); (b) pen-based sketching interfaces; (c) motion filtering and foot-skating removal.

environmental contact information. With such deformable models, we could adjust the values of the deformable parameters $\vec{\alpha}$ and $\vec{\gamma}$, generating an infinite number of natural-looking motions $M(\vec{\alpha}, \vec{\gamma})$. Furthermore, we could choose values for the deformable parameters that would match various forms of user constraints derived from intuitive animation interfaces.

We have demonstrated the power and flexibility of our deformable motion models by exploring both direct manipulation interfaces and sketching interfaces for human motion generation. When using direct manipulation interfaces, one can create a desired animation in real time by dragging the 2D position of any character point at any frame, modifying the trajectory of any character point across the entire sequence, adjusting the values of high-level control knobs such as walking directions or step sizes, or specifying the timing of any frame (Figure 1(a)). In the sketching interfaces, the user can pick any point on the character and sketch a desired timed trajectory in the 2D screen space (Figure 1(b)). The system automatically generates a natural-looking human motion that best matches the positions and timing of the 2D sketches. In addition, the deformable motion models can be leveraged to improve the performance of human motion processing. We show how to apply the deformable motion models to transform a 2D animation sequence into a 3D animation sequence, and remove foot-sliding artifacts in human walking data (Figure 1(c)).

2. BACKGROUND

Our approach constructs deformable motion models from a large set of precaptured motion data and uses them to create realistic animation that satisfies various forms of user-defined constraints. Therefore, we will focus our discussion on data-driven human motion models and their application in the creation of intuitive and interactive interfaces for human motion generation.

One effective way to represent human motion is motion graphs, which represent allowable transitions between poses [Arikan and Forsyth 2002; Kovar et al. 2002; Lee et al. 2002, 2006; Safonova and Hodgins 2007] or motion segments [Treuille et al. 2007; Shum et al. 2008]. Motion graphs transform a motion synthesis problem into a discrete graph search problem, which significantly speeds up the motion generation process. However, these representations lack the capability to generalize the captured data to satisfy new kinematic constraints. Thus, motion graphs do not provide fine-grained control over human motion.

An alternative representation is weighted interpolations of motion examples [Rose et al. 1998; Kovar and Gleicher 2004; Mukai and Kuriyama 2005; Kwon and Shin 2005; Heck et al. 2007]. Motion interpolation registers a set of structurally similar but distinctive motion examples and then parameterizes them in an abstract space defined for motion control. Given the control parameters' new values, the sequences can be smoothly blended with appropriate kernel functions such as radial basis functions. This approach often runs in real time and is well suited for interactive environments where control constraints are known in advance, for example, when only the hand position will be constrained. However, motion interpolation is often based on sparse data interpolation techniques and thus does not provide a principled way to support various forms of kinematic constraints, such as key-frame constraints. In contrast, our system formulates the motion synthesis problem within a spacetime optimization framework [Witkin and Kass 1988], thereby supporting any kinematic constraints. In addition, motion interpolation is not a compact representation for human motion because it neglects spatial-temporal correlation embedded in motion examples. To address this problem, we apply dimensionality reduction analysis techniques to the registered motion examples as well as the time warping functions, constructing a compact and low-dimensional deformable model for human motion representation.

Our work builds on the success of previous statistical models used for human motion analysis and synthesis. Statistical motion models are often represented as a set of mathematical equations or functions that describe human motion using a finite number of parameters and their associated probability distributions [Molina Tanco and Hilton 2000; Brand and Hertzmann 2000; Li et al. 2002; Chai and Hodgins 2007]. Earlier research focused on discrete-state dynamic models, such as variants of Hidden Markov Models (HMMs) [Molina Tanco and Hilton 2000; Brand and Hertzmann 2000] and Switched Linear Dynamic Systems (SLDS) [Li et al. 2002]. These human motion models have been used for interpolation of key frames [Molina Tanco and Hilton 2000; Li et al. 2002] or motion styles [Brand and Hertzmann 2000].

One drawback of discrete-state dynamic models is that they usually do not provide a principled way to support continuous kinematic constraints such as key-trajectories or foot contact constraints. Chai and Hodgins [2007] recently addressed this limitation by constructing continuous dynamic models from human motion data and using them to generate realistic animation in a spacetime optimization framework. However, their system often produces poor results when a sparse set of constraints, such as step sizes for walking, are used for motion synthesis. Furthermore, their system does not run at interactive frame rates; the synthesis process often takes about half a minute. More importantly, both *continuous* and *discrete* statistical models encode little or no information about environmental contact events. As a result, unless the user explicitly specifies contact constraints throughout the motion, the motions they generate often violate the undefined contact constraints and thereby display noticeable visual artifacts, such as foot sliding.

The proposed deformable motion models combine the advantages of motion interpolation and statistical motion models while avoiding their disadvantages at the same time. First, they can be used to generate realistic animation from *any* kinematic constraints at interactive frame rates. Second, our algorithm can generate high-quality animation without specifying *any* environmental contact constraints. Third, deformable motion models can be used to produce a natural-looking animation from both fine-grained constraints and sparse constraints.

In spirit, deformable motion models are similar to low-dimensional statistical human pose models for inverse kinematics [Grochow et al. 2004; Chai and Hodgins 2005]. However, these models lack the temporal information they need to generate animations unless the user provides continuous control signals (the performance animation problem). We significantly extend the idea by building a low-dimensional deformable model for an entire motion sequence and using it for interactive motion design.

Statistical motion models are also used for reconstructing 3D human motion from video [Howe et al. 1999; Pavlović et al. 2000; Ormoneit et al. 2001; Sidenbladh et al. 2002]. This approach is related to our own in that statistical models generated from motion capture data are used to reduce the ambiguity of video-based human motion tracking. However, the statistical models used for human motion tracking have limitations that are similar to the models used for human motion synthesis. They are often based on discrete-state dynamic models and lack environmental contact information. In addition, the process of video-based motion tracking is often performed in a sequential framework, which initializes a tracker in one frame and then performs tracking forward recursively in time. Instead, our motion synthesis algorithm simultaneously computes an entire motion sequence based on the constraints the user defines throughout the motion.

3. OVERVIEW

The goal of this research is to develop an interactive system that allows novices to create realistic human animation quickly and easily. For this purpose, we construct a low-dimensional deformable motion model from a large set of preregistered human motion data and use it to create a natural-looking animation ($\mathbf{x}$) that achieves the goal ($\mathbf{y}$) specified by the user. Our deformable motion models are constructed from a large set of structurally similar but distinctive motion examples.

We represent the set of motion examples in the database as $\{\mathbf{x}_n(t')|n = 1, \ldots, N, t' = 1, \ldots, T'_n\}$, where N is the number of motion examples and T'_n is the total number of frames in the nth motion example. Let the vector $\mathbf{x}_n(t')$ represent the root position and orientation as well as the joint angle values of a full-body pose at frame t'. Let $\mathbf{x}_n$ be the motion vector sequentially concatenating all poses of the nth motion example.

Our system consists of the following major components.

Motion decomposition. First, we use a semi-automatic process to register all motion examples $\mathbf{x}_n, n = 1, \ldots, N$ to a reference motion sequence with appropriate time warping functions $\mathbf{w}_n, n = 1, \ldots, N$. Next, we apply the computed time warping functions $\mathbf{w}_n, n = 1, \ldots, N$ to warp each motion sequence $\mathbf{x}_n$ to a new motion sequence $\mathbf{s}_n$ in a canonical timeline defined by the reference motion. Therefore, we can decompose the original motion data $\mathbf{x}_n, n = 1, \ldots, N$ into two datasets: the warped motion examples $\mathbf{s}_n, n = 1, \ldots, N$ and the corresponding time warping functions $\mathbf{w}_n, n = 1, \ldots, N$. Both datasets are defined in the canonical timeline, which are suitable for statistical analysis.

Motion analysis. We apply statistical analysis techniques to the registered motion data $\mathbf{s}_n, n = 1, \ldots, N$ and construct a deformable *geometric* model of the form $\mathbf{s} = P(\vec{\alpha})$, where $\vec{\alpha}$ is a deformable vector of the model. Similarly, we can apply statistical analysis techniques to the time warping functions $\mathbf{w}_n, n = 1, \ldots, N$ to construct a deformable *timing* model of the form $\mathbf{w} = H(\vec{\gamma})$, where $\vec{\gamma}$ is a deformable vector of the model. The deformable parameters $\vec{\alpha}$ and $\vec{\gamma}$ describe the motion's geometric and timing variations, respectively. Furthermore, we learn a joint probability density function $pr(\vec{\alpha}, \vec{\gamma})$ to model the correlation between the geometric and timing variations. The deformable models and the joint probability distribution define a generative model for human motion synthesis.

Motion synthesis and control. The goal here is to use the deformable motion models to generate a natural-looking animation $\mathbf{x}$ that achieves the goal $\mathbf{y}$ specified by the user. Mathematically, we formulate the motion synthesis problem in a Maximum A Posteriori (MAP) framework by estimating the most plausible parameters $\vec{\alpha}$ and $\vec{\gamma}$ from the user's input $\mathbf{y}$. We explore two interfaces for interactive intuitive human motion generation: direct manipulation interfaces and sketching interfaces. We also show how to extend the framework for filtering noisy motion data and removing foot-sliding artifacts in input motion.

We describe these components in detail in the next three sections.

4. MOTION DATA DECOMPOSITION

We have constructed deformable motion models for a wide variety of human actions, including walking, jumping, bowling, and golf swing. To build a deformable motion model for a particular human action such as walking, we record a database from an actor performing walking with different styles, such as speeds, step sizes, directions, and emotions. We preprocess all motion examples in the database by registering them against each other. More specifically, we pick one example motion as a reference motion (e.g., normal walking) and use it to register the rest of database examples with appropriate time warping functions.

We require all motion examples to be structurally similar so that the captured motion examples can be semantically registered against each other. In addition, we require motion examples have consistent foot contact events in order to model the locations and timing of foot contact events in walking data. A set of walking examples, for instance, must all start out on the same foot, take the same number of steps, and have the same arm-swing phase. However, we do not include any "spurious" motions into the training data. We define "spurious" motions as motions that cannot be semantically registered with the reference motion and/or have different contact events from the reference motion. For example, "walking

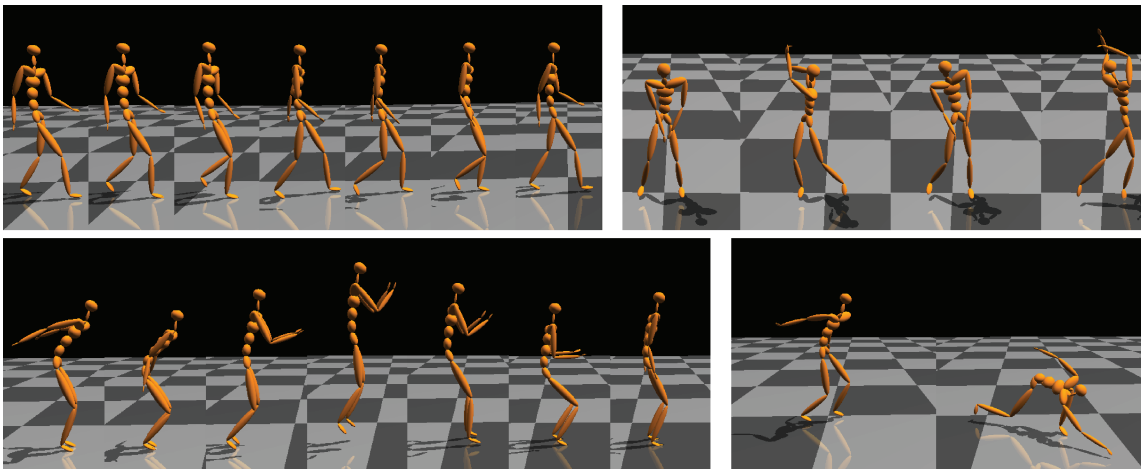


Fig. 2. Key frames used for motion data registration, including walking, golf swing, jumping, and bowling.

with head scratching" is deemed to be "spurious" for our current walking database because the "head scratching" pattern cannot be semantically registered with a normal "arm swing" pattern in the reference motion.

We register motion examples in a translation- and rotation-invariant way by decoupling each pose from its translation in the ground plane and the rotation of its hips about the up axis [Kovar and Gleicher 2003]. To ensure the quality of registration results, we take a "semi-automatic" procedure to register database examples. First, we manually select a small set of key frames for both the reference motion and the motion examples. The key frames often include the starting and ending frames as well as the instants when contact state transitions occur. The key frames allow us to divide the whole motion sequence into multiple subsequences. Next, we use dynamic time warping techniques [Myers and Rabiner 1981] to automatically register each subsequence. If the registration results are not "perceptually" good, we can select more key frames, which often correspond to instants with highest visual content change or frames with poorest registration results, and add them to the registration procedure. We repeat this procedure until satisfactory registration results are achieved.

Figure 2 visualizes key frames used for registering walking, golf swing, jumping, and bowling data, respectively. For walking, the key frames include the starting and ending frames as well as five contact state transition frames. Golf swing needs four key frames, including the starting and ending frames, one contact state transition frame, and one frame with the highest visual content changes. The key frames for jumping include the starting and ending frames, two contact state transition frames, and three frames with the highest visual content changes. Bowling only needs two key frames (the starting and ending frames).

We define time warping functions $t' = w(t)$ in the canonical timeline t, where $t = 1, \ldots, T$. This allows us to describe the time warping functions as a T-dimensional vector $\mathbf{w} = [w(1), \ldots, w(T)]^T$. We also use the time warping functions to warp the original motion examples $\mathbf{x}_n(t'), t' = 1, \ldots, T'_n$ into the registered motion examples $\mathbf{s}_n(t), t = 1, \ldots, T$, where t represents the canonical timeline specified by the reference motion. Figures 3(a), 3(b), and 3(d) visualize the joint angle values of the right femur of the original walking examples $\mathbf{x}_n, n = 1, \ldots, N$, the registered walking examples $\mathbf{s}_n, n = 1, \ldots, N$, and the time warping functions $\mathbf{w}_n, n = 1, \ldots, N$, respectively. The vectors $\mathbf{s}_n$ and $\mathbf{w}_n$ encode the motion's geometric and timing variations, respectively.

One advantage of the new representation is that the vectors $\mathbf{s}_n$ and $\mathbf{w}_n$ incorporate expert knowledge in the annotation of the training data. Note that key frames in the original motion examples annotate important motion structures as well as environmental contact information. As a result, the new vectors capture structural elements and environmental contact knowledge of human actions. For example, we could easily identify which components in the new vectors control the poses of important events such as "toe-down" and "heel-up", or the timing of contact transitions. In addition, the new representation enables us to describe the motion examples with two point clouds $\mathbf{s}_n, n = 1, \ldots, N$ and $\mathbf{w}_n, n = 1, \ldots, N$, which are suitable for statistical data analysis.

5. HUMAN MOTION ANALYSIS

In this section, we apply statistical analysis techniques to both the registered motion examples $\mathbf{s}_n, n = 1, \ldots, N$ and the time warping functions $\mathbf{w}_n, n = 1, \ldots, N$ to construct a deformable motion model of the form $\mathbf{x} = M(\vec{\alpha}, \vec{\gamma})$, where the deformable parameters $\vec{\alpha}$ and $\vec{\gamma}$ control the motion's geometric and timing variations, respectively.

5.1 Modeling Geometric Variations

Our first goal is to construct a deformable *geometric* model of the form $\mathbf{s} = P(\vec{\alpha})$ from the registered motion examples $\mathbf{s}_n, n = 1, \ldots, N$. The registered motion data $\mathbf{s}_n \in R^{DT}, n = 1, \ldots, N$ form a distribution in a high-dimensional motion space. If we can model this distribution, we can generate an infinite number of new motion sequences $\mathbf{s}$, similar to those in the original training set. We can also examine new motions to decide whether they are plausible motions or not.

We apply Principle Component Analysis (PCA) to the registered motion examples $\mathbf{s}_n, n = 1, \ldots, N$. Note that $\mathbf{s}_n$ is a long vector that sequentially concatenates all poses of the nth motion example. As a result, we can construct a deformable geometric model for the entire motion sequence using a mean motion $\mathbf{p}_0$ and a weighted

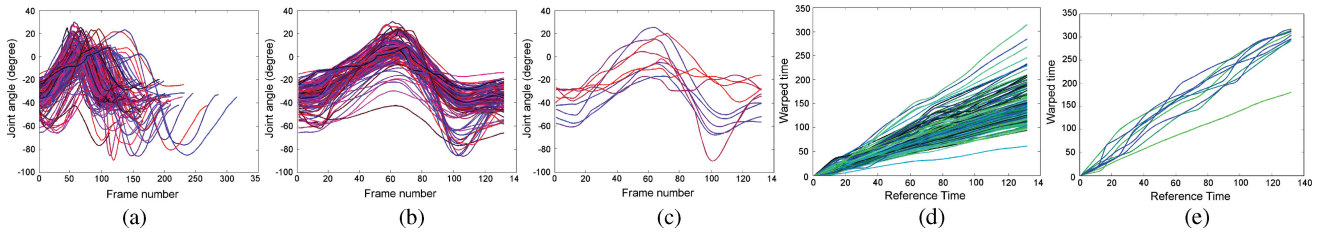


Fig. 3. Analysis and visualization of the 200 walking examples (right femur's rotational angle about the x axis). (a) The unregistered data $\mathbf{x}_n$, $n = 1, \ldots, 200$. (b) The registered data $\mathbf{s}_n$, $n = 1, \ldots, 200$. (c) The top ten deformable modes of the geometric variations $\mathbf{p}_m$, $m = 1, \ldots, 10$. (d) The time warping functions $\mathbf{w}_n$, $n = 1, \ldots, 200$. (e) The top ten deformable modes of the timing variations $\mathbf{b}_k$, $k = 1, \ldots, 10$.

combination of eigen-motions $\mathbf{p}_m$, $m = 1, \ldots, M$:

$$\begin{aligned} \mathbf{s} &= P(\vec{\alpha}) \\ &= \mathbf{p}_0 + [\mathbf{p}_1 \ldots \mathbf{p}_M]\vec{\alpha}, \end{aligned} \tag{1}$$

where the vector $\vec{\alpha} = [\alpha_1, \alpha_2, \ldots, \alpha_M]^T$ represents the eigen weights and the vectors $\mathbf{p}_m$, $m = 1, \ldots, M$ are a set of orthogonal modes to model geometric variations across the entire motion sequence. In addition, the tth pose of the deformable geometric model is given by

$$\begin{aligned} \mathbf{s}(t) &= P(t; \vec{\alpha}) \\ &= \mathbf{p}_0(t) + [\mathbf{p}_1(t) \ldots \mathbf{p}_M(t)]\vec{\alpha}, \quad t = 1, \ldots, T, \end{aligned} \tag{2}$$

where $\mathbf{s}(t)$, $\mathbf{p}_0(t)$, and $\mathbf{p}_m(t)$ represent the poses of the new motion, mean motion, and orthogonal modes at frame t, respectively.

What remains is to determine how many modes (M) to retain. This leads to a trade-off between the accuracy and the compactness of the motion model. However, it is safe to consider small-scale variations as noise. In our experiments, we automatically determine the number of modes by keeping 99% of the original variations. For our walking database containing 200 examples, we can represent an entire walking sequence using a 30-dimensional vector. Figure 4(c) visualizes the first ten modes of the walking database.

5.2 Modeling Timing Variations

A deformable geometric model $\mathbf{s} = P(\vec{\alpha})$ does not consider any timing variations embedded in the motion examples because it is constructed from the registered motion examples $\mathbf{s}_n$. A good extension is to warp the deformable geometric model $P(\vec{\alpha})$ with an appropriate time warping function of the form $\mathbf{w} = H(\vec{\gamma})$, where $\vec{\gamma}$ is a vector of the deformable timing model H.

Direct application of statistical analysis techniques to time warping functions, however, could be problematic because time warping functions are constrained functions; they should be positive and strictly monotonic everywhere. For example, if we directly apply the PCA to the time warping functions $\mathbf{w}_n$, $n = 1, \ldots, N$, the resulting deformable timing model might produce invalid time warping functions that violate the monotonicity (Figure 4). Therefore, we need to extend statistical analysis techniques for time warping functions.

Our idea is to transform the constrained time warping functions $\mathbf{w}_n$ into unconstrained functions $\mathbf{z}_n$ and apply statistical analysis to $\mathbf{z}_n$, $n = 1, \ldots, N$ to construct a deformable timing model of the form $\mathbf{z} = Z(\vec{\gamma})$. Transforming the deformable timing model $\mathbf{z} = Z(\vec{\gamma})$ from the new space $\mathbf{z}$ to the original space $\mathbf{w}$ results in our final deformable timing model $\mathbf{w} = H(\vec{\gamma})$.

More specifically, we choose to transform a time warping function $w(t)$ into a new space $z(t)$ as follows.

$$z(t) = \ln(w(t) - w(t-1)), \quad t = 1, \ldots, T \tag{3}$$

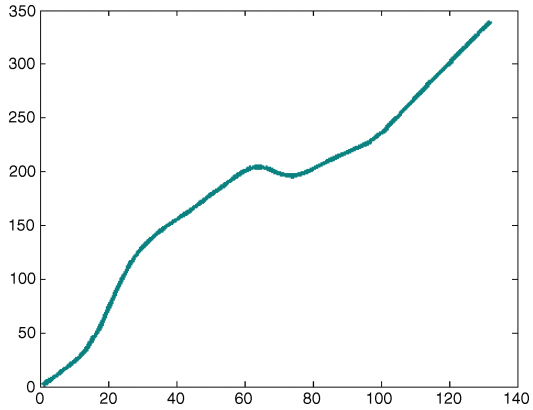


Fig. 4. Monotonic condition of time warping functions: a linear combination of eigen curves in the original space produces a time warping curve that violates the monotonic condition.

After choosing $w(0)$ to be zero, we can easily transform the function from the new space $\mathbf{z}$ back to the original space $\mathbf{w}$.

$$w(t) = \sum_{i=1}^{t} \exp[z(i)], \quad t = 1, \ldots, T \tag{4}$$

Eq. (4) ensures that both positive and monotonic constraints will be automatically satisfied if we conduct statistical analysis in the new space $\mathbf{z}$ and transform the deformable timing models back to the original space.

Similarly, we apply the Principle Component Analysis (PCA) to the time warping functions $\mathbf{z}_n$, $n = 1, \ldots, N$ in the new space. We can represent the deformable timing model $\mathbf{z} = Z(\vec{\gamma})$ in the new space as a mean warping function $\mathbf{b}_0$ plus a linear combination of eigen-vectors $\mathbf{b}_k$, $k = 1, \ldots, K$. We have

$$\begin{aligned} \mathbf{z} &= Z(\vec{\gamma}) \\ &= \mathbf{b}_0 + [\mathbf{b}_1 \ldots \mathbf{b}_K]\vec{\gamma}, \end{aligned} \tag{5}$$

where the vector $\vec{\gamma} = [\gamma_1, \gamma_2, \ldots, \gamma_K]^T$ is the deformable timing vector of the model. Combining Eq. (5) with Eq. (4), we obtain the following deformable timing model in the original space:

$$\begin{aligned} w(t) &= H(t; \vec{\gamma}) \\ &= \textstyle\sum_{i=1}^{t} \exp(b_0(i) + [b_1(i) \ldots b_K(i)]\vec{\gamma}), \ t = 1, \ldots, T, \end{aligned} \tag{6}$$

where the scalar $b_k(i)$ represents the ith component of the kth eigenvector $\mathbf{b}_k$. Note that the time warping functions generated by the deformable timing model $H(\vec{\gamma}) = [H(1; \vec{\gamma}), \ldots, H(T; \vec{\gamma})]^T$ are always positive and monotonic no matter how we adjust the values

of the deformable parameters $\vec{\gamma}$. Figure 3(e) visualizes the first ten modes of the time warping functions in the original space $\mathbf{w}$.

5.3 Deformable Motion Models

We now combine the deformable geometric models $P(\vec{\alpha})$ with the deformable timing models $H(\vec{\gamma})$ to construct a complete deformable motion model

$$\begin{aligned} \mathbf{x} &= M(\vec{\alpha}, \vec{\gamma}) \\ &= P(\vec{\alpha}) \otimes H(\vec{\gamma}), \end{aligned} \tag{7}$$

where the operator $\otimes$ warps the motion sequence $P(\vec{\alpha})$ with the time warping function $H(\vec{\gamma})$.

The deformable motion model $M(\vec{\alpha}, \vec{\gamma})$ can be applied to generate a new motion sequence $\mathbf{x}$. In particular, given the deformable geometric parameters $\vec{\alpha}$, we can use Eq. (2) to generate a new motion sequence $\mathbf{s} = P(\vec{\alpha})$ in the canonical timeline. Similarly, a new time warping function $\mathbf{w} = H(\vec{\gamma})$ can be generated by choosing appropriate values for the deformable timing parameters $\vec{\gamma}$ based on Eq. (6). A new motion sequence $\mathbf{x}$ can then be obtained by warping the new motion $\mathbf{s} = P(\vec{\alpha})$ in the canonical timeline with the time warping function $\mathbf{w} = H(\vec{\gamma})$.

Mathematically, we can generate a new motion instance $\mathbf{x}(t')$, $t' = 1, \ldots, T'$ with the following equation.

$$\begin{aligned} \mathbf{x}(t') &= P(w(t); \vec{\alpha}) \\ &= P(H(t; \vec{\gamma}); \vec{\alpha}) \\ &= \mathbf{p}_0(H(t; \vec{\gamma})) + [\mathbf{p}_1(H(t; \vec{\gamma})) \ldots \mathbf{p}_M(H(t; \vec{\gamma}))]\vec{\alpha} \end{aligned} \tag{8}$$

The generated motion $\mathbf{x}$ is sampled at the timeline t', $t' = 1, \ldots, T'$; typically it is linearly interpolated from $\mathbf{x}(w(t))$, $t = 1, \ldots, T$. Note that the deformable motion model is a nonlinear human motion generative model because it is nonlinear w.r.t (with respect to) the timing parameters $\vec{\gamma}$ although linear w.r.t the geometric parameters $\vec{\alpha}$.

In addition, we learn a joint probability distribution function $pr(\vec{\alpha}, \vec{\gamma})$ based on the registered motion examples $\mathbf{s}_n$, $n = 1, \ldots, N$ and the time warping functions $\mathbf{w}_n$, $n = 1, \ldots, N$. The joint probability distribution, which models the correlation between the geometric and timing variations, can be used to constrain the resulting motions to stay close to the training examples.

For each motion example $\mathbf{s}_n$ and $\mathbf{w}_n$ (thus also $\mathbf{z}_n$) in the database, we can compute the deformable parameters $\vec{\alpha}_n$ and $\vec{\gamma}_n$ with Eq. (2) and Eq. (5) respectively.

$$\begin{aligned} \vec{\alpha}_n &= [\mathbf{p}_1 \ldots \mathbf{p}_M]^T(\mathbf{s}_n - \mathbf{p}_0) \\ \vec{\gamma}_n &= [\mathbf{b}_1 \ldots \mathbf{b}_K]^T(\mathbf{z}_n - \mathbf{b}_0) \end{aligned} \tag{9}$$

We then form a concatenated vector $\mathbf{u}_n$ as

$$\mathbf{u}_n = \begin{pmatrix} \vec{\alpha}_n \\ \mathbf{W}\vec{\gamma}_n \end{pmatrix} = \begin{pmatrix} [\mathbf{p}_1 \ldots \mathbf{p}_M]^T(\mathbf{s}_n - \mathbf{p}_0) \\ \mathbf{W}[\mathbf{b}_1 \ldots \mathbf{b}_K]^T(\mathbf{z}_n - \mathbf{b}_0) \end{pmatrix}, \quad n = 1, \ldots, N, \tag{10}$$

where the diagonal matrix $\mathbf{W}$ controls the weights of each timing parameter γ_k, $k = 1 \ldots, K$, allowing for the difference in units between the geometric and timing parameters. We model the probability distribution $pr(\vec{\alpha}, \vec{\gamma})$ with a Gaussian Mixture Model (GMM). The parameters of the Gaussian mixture model are automatically estimated using an expectation-maximization algorithm [Bishop 1996].

We now describe how to compute the diagonal weight matrix $\mathbf{W}$. The elements of $\vec{\alpha}$ and $\vec{\gamma}$ have different units so they cannot be compared directly. Because $\mathbf{p}_m$, $m = 1, \ldots, M$ are orthogonal columns, varying $\vec{\alpha}$ by one unit moves $\mathbf{s} = P(\vec{\alpha})$ by one unit. To make $\vec{\alpha}$ and $\vec{\gamma}$ commensurate, we must estimate the effect of varying $\vec{\gamma}$ on the motion $\mathbf{s}$. For each training example $\mathbf{s}_n$, $n = 1, \ldots, N$, we systematically displace each element of $\vec{\gamma}$ from its default value $\vec{\gamma}_n$ and warp the default motion $\mathbf{s}_n$ with the displaced time warping functions. The RMS (Root Mean Square) change in $\mathbf{s}_n$, $n = 1, \ldots, N$ per unit change in timing parameters gives the weight to be applied to that parameter in Eq. (10).

The deformable motion models $M(\vec{\alpha}, \vec{\gamma})$ and the joint distribution function $pr(\vec{\alpha}, \vec{\gamma})$ provide a compact generative model for human motion synthesis. With such a generative model, we can generate an infinite number of motion instances by sampling the joint probability distribution $pr(\vec{\alpha}, \vec{\gamma})$ and warping the synthesized motion $P(\vec{\alpha})$ in the canonical timeline with the synthesized time warping function $H(\vec{\gamma})$. More importantly, we can use them to create a natural-looking animation that achieves the goal specified by the user.

6. INTERACTIVE HUMAN MOTION SYNTHESIS

This section discusses the application of deformable motion models to intuitive interactive human motion generation. The key idea of our motion synthesis process is sampling the prior distribution function $pr(\vec{\alpha}, \vec{\gamma})$ to generate a motion instance $\mathbf{x} = M(\vec{\alpha}, \vec{\gamma})$ that best matches the user's input $\mathbf{y}$.

We will formulate the motion synthesis problem in a Maximum A Posteriori (MAP) framework by estimating the most likely deformable parameters $\vec{\alpha}$ and $\vec{\gamma}$ from the user's input $\mathbf{y}$.

$$\begin{aligned} \arg\max_{\vec{\alpha},\vec{\gamma}} pr(\vec{\alpha}, \vec{\gamma}|\mathbf{y}) &= \arg\max_{\vec{\alpha},\vec{\gamma}} \frac{pr(\mathbf{y}|\vec{\alpha},\vec{\gamma})pr(\vec{\alpha},\vec{\gamma})}{pr(\mathbf{y})} \\ &\propto \arg\max_{\vec{\alpha},\vec{\gamma}} pr(\mathbf{y}|\vec{\alpha}, \vec{\gamma})pr(\vec{\alpha}, \vec{\gamma}). \end{aligned} \tag{11}$$

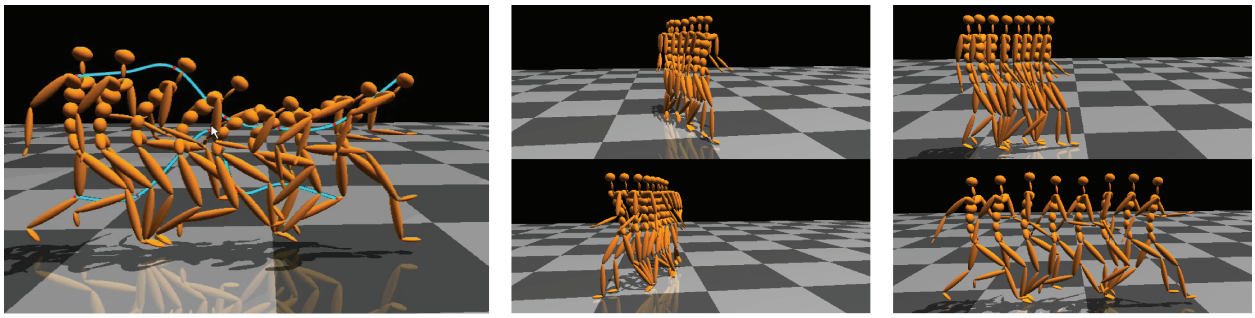
In our implementation, we minimize the negative logarithm of the posteriori probability density function $pr(\vec{\alpha}, \vec{\gamma}|\mathbf{y})$, yielding the following energy minimization problem:

$$\arg\min_{\vec{\alpha},\vec{\gamma}} \underbrace{-\ln pr(\mathbf{y}|\vec{\alpha}, \vec{\gamma})}_{E_{likelihood}} + \underbrace{-\ln pr(\vec{\alpha}, \vec{\gamma})}_{E_{prior}}, \tag{12}$$

where the first term $E_{likelihood}$ is the *likelihood* term that measures how well the generated motion $\mathbf{x} = M(\vec{\alpha}, \vec{\gamma})$ matches the user's input $\mathbf{y}$, and the second term is the *prior* term E_{prior} that constrains the generated motion to stay close to the training examples. Optimal estimation of the deformable parameters from the user's input produces a natural-looking motion that achieves the goal specified by the user.

The MAP framework naturally balances the trade-off between the user constraint term and the motion prior term. For our application, this is particularly important because constraints from intuitive interfaces are often noisy, and we usually cannot find a natural-looking human motion that exactly matches the noisy constraints defined by novices. Furthermore, this allows us to formulate the motion synthesis process in a spacetime optimization framework. This framework is extremely flexible; *any* form of user constraint can be integrated into the framework as long as we can numerically evaluate the corresponding likelihood terms, that is, how well the generated motion $M(\vec{\alpha}, \vec{\gamma})$ matches the user's input $\mathbf{y}$. Another advantage of this framework is that the total number of deformable parameters is often very low (typically less than 50). Optimizing human motions in such a low-dimensional space not only improves the speed of the synthesis, it also reduces the synthesis ambiguity.

In the following subsections, we will explore two interactive interfaces for human motion generation: direct manipulation interfaces and sketching interfaces. In addition, we will discuss how to transform noisy motion data into high-quality human motion data by applying the framework.



(a) (b) (c)

Fig. 5. Our system allows an intuitive design of human motion in real time: (a) direct manipulation with multiple trajectory constraints; (b) direct manipulation of walking directions; (c) direct manipulation of step sizes.

6.1 Direct Manipulation Interfaces

Direct manipulation interfaces start with a default motion sequence: $M(\vec{\mathbf{0}}, \vec{\mathbf{0}})$. The user can adjust the motion in real time by dragging *any* character point at *any* frame, modifying the trajectory of *any* character point across the entire sequence, adjusting the values of high-level control knobs such as walking directions or step sizes, or specifying timing constraints for *any* frame. All the constraints can be specified on a 2D screen space. Thus we avoid the need for complex and cumbersome 3D interactions that are common in animation software. In particular, the current direct manipulation interfaces support the following real-time interactions.

Point dragging. The user can select any character point at any frame and drag its position with a mouse in screen space (Figure 1(a)). The system automatically adjusts the deformable parameters to satisfy the point dragging constraints.

Trajectory editing. The user can select any character point and edit its trajectory across the entire sequence in screen space (Figure 5(a)). First, the system fits the trajectory using Catmull-Rom splines [Catmull and Rom 1974]. Then, the user can incrementally edit the trajectory by adjusting the control points located on the spline. We choose Catmull-Rom splines because they are easy to compute and guarantee that control points lie on the curve. Moreover, the tangents of the generated curve are continuous over multiple segments.

Fixed handles. Because natural human motion is a coordinated movement, the movement between different joints might not be independent. If the user controls the motion with one of the aforesaid handles, unselected regions of human motion might still change. With fixed handles, the user can select any character point, key pose, or trajectory that should remain unchanged. This handle must be used with other constraints, which allows for local control of any human motions that are being changed. In Figure 1(a), the user drags a point on the character's head by clicking a fixed handle pinned on its left leg.

Other handles. The user can achieve a high-level motion control by combining fixed handles with point dragging operations. For example, the user can adjust the walking direction or step size by fixing the root position of the starting frame and dragging the root position of the ending frame in different directions, such as away or towards the fixed points. Figures 5(b) and 5(c) illustrate the direct manipulation interface for adjusting the walking direction and step size. In addition, the user can adjust a small set of key poses and use the system to automatically interpolate the intermediate poses.

Timing control. For novice users, timing an animation is one of the most difficult processes. Our system allows the user to select any frame throughout the motion and specify a desired timing. In addition, the user can select any two frames and control the time duration between the two frames.

In general, the constraints here can be categorized into two groups: *kinematic* constraints $\mathbf{y}_{kine}$ and *timing* constraints $\mathbf{y}_{time}$. The kinematic constraints control the motion's geometric variations while the timing constraints specify the generated motion's desired timing. One nice property of the direct manipulation interfaces is to allow the user to specify the kinematic constraints w.r.t. particular frames in the canonical timeline. In other words, the specified kinematic constraints $\mathbf{y}_{kine}$ are independent of the deformable timing parameters $\vec{\gamma}$. In addition, because the timing constraints $\mathbf{y}_{time}$ are fully determined by time warping functions, they are independent of the geometric deformable parameters $\vec{\alpha}$. Therefore, we can decompose the likelihood term for the direct manipulation interfaces into two terms

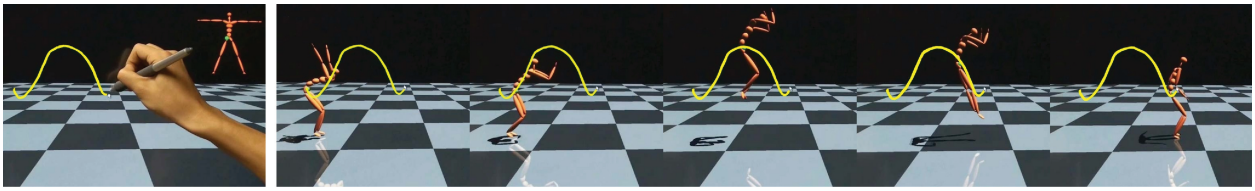
$$\begin{aligned} E_{likelihood} &= -\ln pr(\mathbf{y}_{kine}, \mathbf{y}_{time}|\vec{\alpha}, \vec{\gamma}) \\ &= -\ln pr(\mathbf{y}_{kine}|\vec{\alpha}, \vec{\gamma}) + -\ln pr(\mathbf{y}_{time}|\vec{\alpha}, \vec{\gamma}) \\ &= \underbrace{-\ln pr(\mathbf{y}_{kine}|\vec{\alpha})}_{E_{kine}} + \underbrace{-\ln pr(\mathbf{y}_{time}|\vec{\gamma})}_{E_{time}}, \end{aligned} \tag{13}$$

where E_{kine} and E_{time} represent the likelihood terms for kinematic and timing constraints, respectively.

While the deformable parameters $\vec{\alpha}$ and $\vec{\gamma}$ completely determine the generated motion $M(\vec{\alpha}, \vec{\gamma})$, the input $\mathbf{y}_{kine}$ and $\mathbf{y}_{time}$ specified by the user might vary due to noise. Assuming Gaussian noise with a standard deviation of σ_{kine} for kinematic input at frame t_k ($\mathbf{y}^k_{kine}$), we can define the likelihood term for kinematic constraints as

$$\begin{aligned} E_{kine} &= -\ln pr(\mathbf{y}_{kine}|\vec{\alpha}) \\ &= -\ln \textstyle\prod_k \frac{1}{\sqrt{2\pi}} \exp \frac{-\|\mathbf{y}^k_{kine} - \mathbf{f}(P(t_k;\vec{\alpha}))\|^2}{\sigma^2_{kine}} \\ &\propto \textstyle\sum_k \frac{\|\mathbf{y}^k_{kine} - \mathbf{f}(P(t_k;\vec{\alpha}))\|^2}{\sigma^2_{kine}}, \end{aligned} \tag{14}$$

where the function $\mathbf{f}$ maps the t_k-th pose of the synthesized motion sequence in the joint angle space to the positions and/or orientations of end-effectors in screen space. Note that we drop off the constant $-\ln \frac{1}{\sqrt{2\pi}}$ in the equation. A good match between the synthesized motion and the user's input results in a low value for the likelihood term.



(a) (b)

Fig. 6. Sketching interfaces for interactive motion generation. (a) The user picks a point on the character and sketches a timed trajectory in 2D screen space. (b) The generated animation automatically matches the timing and trajectory of the 2D sketch.

Similarly for timing constraints $\mathbf{y}_{time}$, we assume Gaussian noise with a standard deviation of σ_{time}. We can define the likelihood term for the timing constraints as

$$E_{time} \propto \frac{\|\mathbf{y}_{time}-\mathbf{g}(H(\vec{\gamma}))\|^2}{\sigma_{time}^2}, \tag{15}$$

where the function $\mathbf{g}$ maps the generated time warping function $H(\vec{\gamma})$ to the timings of particular frames or the time duration of two selected frames.

The final objective function for direct manipulation interfaces combines the two likelihood terms with the prior term.

$$\arg\min_{\vec{\alpha},\vec{\gamma}} E_{kine}(\vec{\alpha}) + E_{time}(\vec{\gamma}) - \ln pr(\vec{\alpha},\vec{\gamma}) \tag{16}$$

In practice, we decompose the entire optimization for the direct manipulation interfaces into two separate processes. For kinematic motion manipulation, the system keeps the timing parameters $\vec{\gamma}$ constant and optimizes the geometric parameters $\vec{\alpha}$ iteratively. The timing manipulating process fixes the geometric parameters $\vec{\alpha}$ and updates the timing parameters $\vec{\gamma}$ based on the timing constraints.

6.2 Sketching Interfaces

When using a sketching interface, the user can pick *any* point on the character and sketch a timed trajectory in 2D screen space. The system automatically generates a natural-looking animation that best matches the path and timing of the 2D sketch (Figure 1(b)). Unlike direct manipulation interfaces, the sketches drawn by the user are defined on their own timelines $t', t' = 1,\ldots,T'$ rather than the canonical timeline $t, t = 1,\ldots,T$.

Let $\mathbf{y}(t'), t' = 1,\ldots,T'$ denote the sketched 2D position at frame t'. We assume that the observed sketches $\mathbf{y}$ vary with a standard deviation of σ_{sketch}, due to Gaussian noise. We can define the likelihood term for sketching interfaces E_{sketch} as

$$\begin{aligned} E_{sketch} &= -\ln pr(\mathbf{y}|\vec{\alpha},\vec{\gamma}) \\ &\propto \textstyle\sum_{t'=1}^{T'} \|\mathbf{y}(t') - \mathbf{f}(P(w(t);\vec{\alpha})\|^2/\sigma_{sketch}^2 \\ &\propto \textstyle\sum_{t'=1}^{T'} \|\mathbf{y}(t') - \mathbf{f}(P(T(t;\vec{\gamma});\vec{\alpha}))\|^2/\sigma_{sketch}^2, \end{aligned} \tag{17}$$

where the function $\mathbf{f}$ maps a synthesized full-body pose at a particular frame to the positions and/or orientations of end effectors. For simplicity, we measure the consistency between the input sketches and the synthesized motion at the timeline of the input sketch $t' = 1,\ldots,T'$. The temporal correspondences between the two motions are described by the time warping function $t' = w(t)$. Usually, an evaluation of the likelihood term needs to interpolate the synthesized motion at the timeline $t' = 1,\ldots,T'$ based on the values at $w(1),\ldots,w(T)$.

Combining the likelihood term with the prior term, we can formulate the sketch-based animation process as the following optimization problem. We have

$$\arg\min_{\vec{\alpha},\vec{\gamma}} E_{sketch} - \ln pr(\vec{\alpha},\vec{\gamma}), \tag{18}$$

where the first term E_{sketch} is the likelihood term for the sketch-based animation, which measures how well the generated animation matches the path and timing of a 2D sketch.

The current system allows the user to sketch out a motion using a pen-based interface (Figure 6(a)). However, the sketching interface can also be used with other types of input devices. For example, the user could wear a small number of accelerometers, such as wiimotes and act out a desired motion with their own performance.

6.3 Motion Filtering

One nice property of deformable motion models is that they encode environmental contact information. With such a model, we could filter the noisy motion that violates environmental contact constraints. For example, foot-skating artifacts often appear during various motion data processing stages, such as motion editing, blending, warping, or synthesis. Our algorithm can be used to automatically detect and remove foot-skating artifacts present in an input walking sequence.

Let $\mathbf{y}(t'), t' = 1,\ldots,T'$ represent noisy motion data, we can formulate the motion filtering problem as the MAP problem

$$\arg\min_{\vec{\gamma},\vec{\alpha}} \sum_{t'=1}^{T'} \|\mathbf{y}(t') - P(T(t;\vec{\gamma});\vec{\alpha})\|^2/\sigma_{filter}^2 - \ln pr(\vec{\alpha},\vec{\gamma}), \tag{19}$$

where $P(T(t;\vec{\gamma});\vec{\alpha}))$ represents the filtered motion.

Figure 1(c) shows the result for filtering a noisy walking sequence. The original walking sequence contains a significant number of foot-skating artifacts, in which, instead of remaining firmly in place, a character's foot slides on the ground after the character plants it .

6.4 Real-time-Motion Optimization

For all applications, we first analytically evaluate the Jacobian terms of the objective function. Next, we run a gradient-based optimization with the Levenberg-Marquardt algorithm [Lourakis 2009]. In the following section, we briefly discuss the initialization and convergence speed for each application.

The direct manipulation interface for animation synthesis runs in real time. We initialize the motion by using the mean motion and time warping function. When the user employs the direct manipulation interfaces to edit the motion on-the-fly, we initialize the motion in the current timestep t with the motion generated in the previous timestep $t-1$. This significantly improves the optimization efficiency because the initialized motion is already close to the final motion. We have completed hundreds of tests with various forms of

Table I. Details of the Data We Used

	Walking	Jumping	Bowling	Golf Swing
N	200	41	40	45
M	30	18	20	23
K	20	12	11	15
T	133	145	69	306
F	120	120	120	120

N is the number of motion examples. M is the number of the deformable parameters for geometric variations. K is the number of the deformable parameters for timing variations. T is total number of frames used for reference motions. F is the frame rate for motion databases (frame per second).

direct manipulation constraints and have never found a local minimum problem. Besides optimization in a very low-dimensional space and very good initial guesses, we believe there are two other factors involved in fast convergence. First, we optimize α and γ separately in the direct manipulation interface. Second, the direct manipulation constraints are often very sparse.

In the sketching interfaces, we initialize the motion with the mean motion. The solution converges rapidly due to a low-dimensional optimization space and a prior term constraining the solution space. Usually, the system runs at interactive frame rates and does not have local minimum problems. However, we have observed that the system often slows down a little bit when the timing of the sketched trajectories (e.g., an extremely slow sketch of a jumping motion) is unreasonable.

7. RESULTS

We demonstrate the performance of our motion generation system in a variety of human actions, including walking, jumping, bowling, and golf swing as well as transition motions, such as walking to jumping. Table I shows the details of our training data. The motion was captured with a Vicon motion capture system consisting of 12 MXF20 cameras with 41 markers, running at 120Hz. Our results are best seen in the accompanying video although we show sample frames of a few motions in the article.

Direct manipulation interfaces. Figures 1(a) and 5 show sample images for the direct manipulation interfaces, including direct manipulation with point constraints and fixed constraints, direct manipulation with multiple trajectory constraints, direct manipulation of walking directions, and direct manipulation of a walking step size. The direct manipulation interfaces run in real time.

Sketching interfaces. Figure 1(b) and Figure 6 show sample images of the sketching interfaces. The user selects one point on the character and draws a timed trajectory in the screen space. The system automatically generates a motion sequence that matches the speeds and trajectories defined by the input sketches.

Other applications. The motion synthesis framework for sketching interfaces is highly flexible. It can easily be extended to transform various forms of user constraints into a high-quality 3D motion sequence. For example, it can be used to generate 3D animation sequences from 2D animation sequences, reconstruct 3D human motion from a small set of 2D trajectories tracked from video, or interpolate intermediate frames between key frames. Figure 7 shows that we can take a 2D biped walking sequence generated by physically-based simulation [Yin et al. 2007] as an input and lift the 2D motion into a high-quality 3D animation sequence. More specifically, we use the 2D foot positions and the 2D joint angle values of the biped motion data to reconstruct the deformable motion parameters. Figure 8 shows that we can transform a small set of 2D tracking features from a single-camera video stream into a high quality 3D motion sequence.

7.1 Synthesis of Long Motions

Our current system can create a long motion by seamlessly stitching cyclic motions of the same class. For example, in the "lifting 2D animation into 3D animation" and "motion filtering" applications, both systems produce a two-cycle walking sequence. To create such results, we optimize the deformable parameters of the two cycles simultaneously. To ensure a smooth transition from one cycle to another, we enforce smoothness constraints at transition frames.

If we cannot smoothly transition from one action to another, we need to construct deformable models for transition motions as well. We record various styles of transition motions and use the motion registration and analysis procedure to construct a deformable model for the transition motions. We synthesize a long motion by simultaneously estimating the deformable parameters of multiple segments (including atomic motions and transition motions) from user constraints. Similarly, we enforce smoothness constraints at all transition frames. Figure 9 shows a long motion generated by deformable motion models of three atomic motions (walking, jumping, and bowling) and one transition motion (walking to jumping). Note that the system directly transitions from jumping to bowling with smoothness constraints.

7.2 Comparisons

The deformable motion models combine the advantages of both the motion interpolation and statistical modeling approaches while avoiding their disadvantages at the same time. We evaluate the effectiveness of our algorithm by comparing it with motion interpolation [Kovar and Gleicher 2004] and statistical dynamic models [Chai and Hodgins 2007].

Comparison with motion interpolation. Unlike deformable motion models, motion interpolation does not support timed trajectory constraints such as sketched trajectories, timed key frames, 2D animation sequences, or noisy motion sequences. For the simplicity of comparison, we assume that the input trajectory has already been registered with the reference motion. We use the same set of constraints (one single trajectory) for motion synthesis and interpolation. Figures 10(a) and 10(b) show that our algorithm can satisfy the constraints much better than motion interpolation.

Comparison with statistical dynamic models. We also conducted comparison against statistical dynamic models [Chai and Hodgins 2007]. Once again, we used the same set of constraints for comparison, and chose the key-frame constraints in order to evaluate their performances. The accompanying video shows that our algorithm creates much better results than statistical dynamic models. In particular, the animation generated by the statistical dynamic models contains significant foot-sliding artifacts. This is because statistical dynamic models do not encode any foot contact information. In contrast, our algorithm can generate a natural-looking animation without any foot-sliding artifacts. Figures 10(c) and 10(d) show sample frames from the comparison results. In addition, our system runs at interactive frame rates, and the computational time of our system is approximately 100 times faster than the synthesis system using statistical dynamic models.

7.3 Limitations

The quality of the generated animation highly depends on the quality of the user constraints and prior knowledge embedded in training

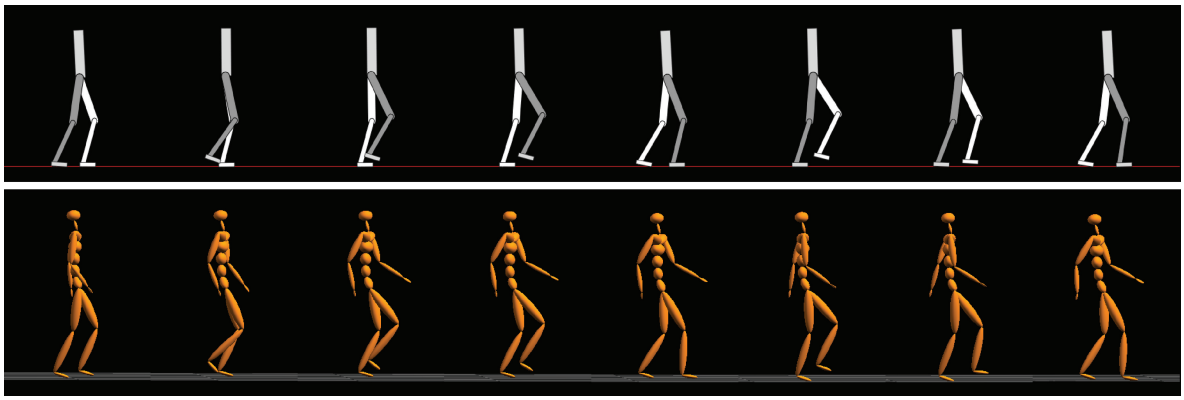


Fig. 7. Motion transformation using deformable motion models: (top) a 2D biped walking sequence generated by physically-based simulation; (bottom) the transformed 3D animation.

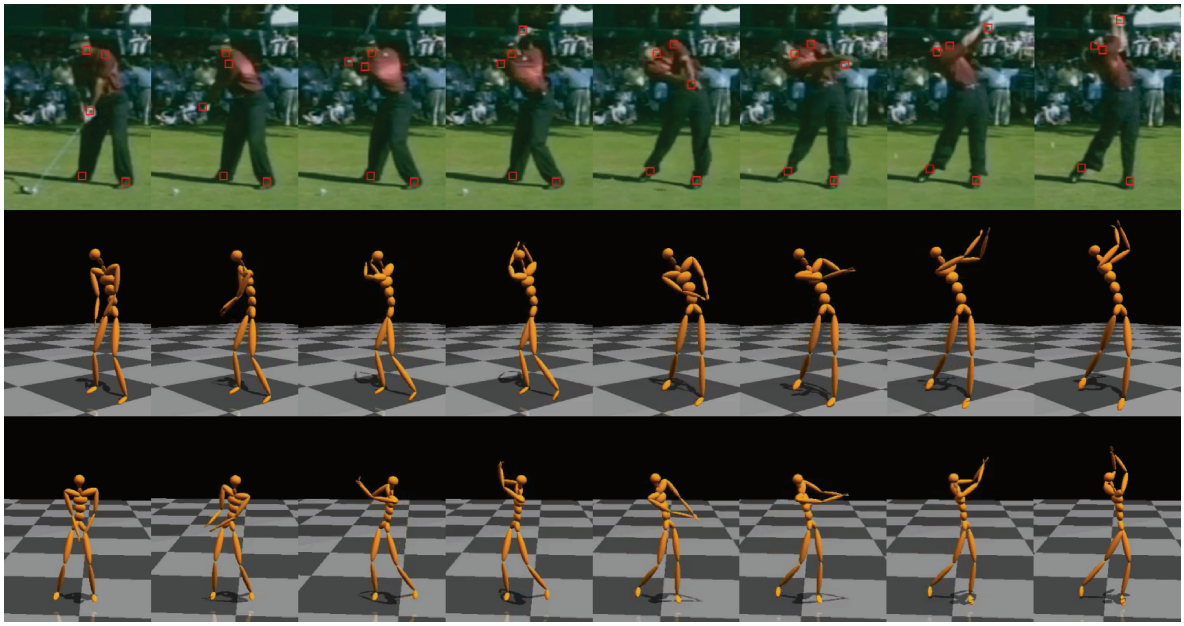


Fig. 8. Interactive generation of a golf swing motion from an input video: (top) the input video and tracked image features; (middle) the reconstructed 3D motion from the same viewpoint; (bottom) the reconstructed 3D motion from a different viewpoint.

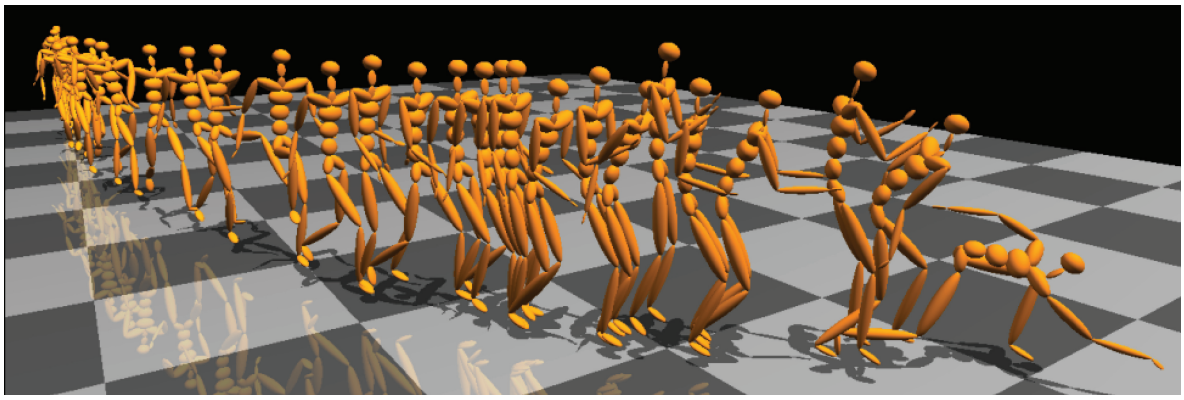
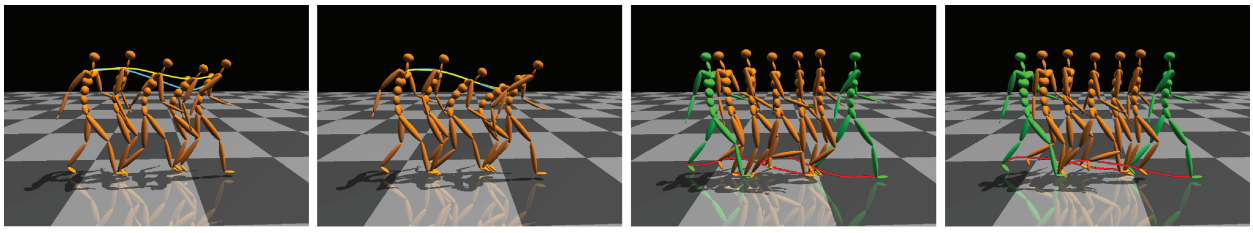
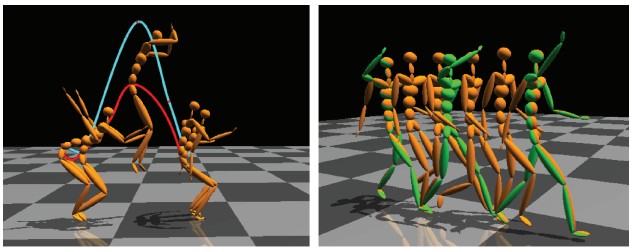


Fig. 9. A long animation generated by deformable motion models of three atomic actions (walking, jumping, and bowling) and one transition motion (walking to jumping).



(a) (b) (c) (d)

Fig. 10. Comparisons with motion interpolation and statistical dynamic models. (a) The result from motion interpolation does not satisfy the constraints. (b) Our result with the same set of trajectory constraints. (c) The result from statistical dynamic models contains significant foot-sliding artifacts and appears very jerky. (d) Our result with the same set of key-frame constraints. Note that the green poses are the key frames specified by the user.



(a) (b)

Fig. 11. Failures and questionable examples. (a) Generating a jumping motion with an impossible jumping height in the sketching interfaces, where the cyan curve is the sketched root trajectory and the red curve is the synthesized root trajectory. (b) Generating walking with hand waving by using three key frames in the direct manipulation interfaces, where the green frames are key-frame constraints, and the gold frames are synthesized motions.

data. The system often fails to generate a natural-looking motion that accurately satisfies the user's input in the following two cases.

—The user constraints are not natural or self-conflicting. In other words, there does not exist a natural-looking motion that can accurately satisfy the constraints. For example, if the root trajectory sketched by the user is not a perfect ballistic trajectory, the generated jumping animation will not precisely follow the user's sketch because there are no natural jumping motions that accurately satisfy the sketching constraints. Figure 11(a) shows that the system cannot generate a jumping motion with an impossible height.

—The system will not produce a desired motion if the training data does not contain any desired motion patterns. For example, the current deformable walking model fails to generate walking with "hand waving" because the current database does not include any "hand waving" patterns (Figure 11(b)).

When a failure or questionable case happens, the system prefers to generate a natural-looking motion that "best" matches the user's input rather than generating the "best"possible motion that "precisely" matches the user's input.

8. DISCUSSION AND CONCLUSION

We have presented deformable motion models for human motion analysis and synthesis. The proposed deformable motion models can represent human actions in a low-dimensional space, which not only speeds up the synthesis process but also reduces the synthesis ambiguity. Another advantage of the deformable motion models is that they encode global motion structures and environmental contact information. For example, we could easily identify which frames in the deformable walking models correspond to "left-toe-down" or "double-support." This ensures synthesized motions have correct global motion structures and environmental contact states, thereby significantly reducing visual artifacts that are often present in statistically-based motion synthesis systems.

We have developed two interactive and easy-to-use interfaces for human motion generation: direct manipulation interfaces and sketching interfaces. The direct manipulation interface allows users to directly manipulate an entire motion sequence presented to them. The interfaces are easy to learn and use; rapid, incremental feedback allows users to make fewer errors and complete tasks in less time, because they can see the results of an action before wasting the time it takes to complete the action. The interfaces work well for novice users because the user can generate realistic animation even with a very sparse set of constraints, for example, point dragging at key frames, foot step sizes, or directions. The user could also fine-tune the motion with fixed handles and incremental constraints. But the interfaces might become time consuming when a large number of constraints are needed to generate a fine-grained motion.

Sketching interfaces, which allow for placing a series of kinematic and timing constraints in a single step, probably provide the fastest and most convenient way to generate a high-quality animation. The system, however, may fail to generate good results if the user does not have the skills to control the positions and timing of an end-effector. One way to address this problem is to use accelerometers (e.g., wiimotes) to sketch out the motion with full-body performances. However, sketching interfaces do not provide the flexibility to fine tune the motion. One possible solution is to combine them with the direct manipulation interfaces. For example, the user can quickly produce a rough motion with the sketching interfaces and then rely on the direct manipulation interfaces to fine tune the motion.

We formulate the constraint-based motion synthesis problem in a Maximum A Posteriori (MAP) framework. The MAP framework provides a principled way to balance the trade-off between user constraints and motion priors. In our experiments, the constraints from intuitive interfaces are often noisy and could even be unnatural, or conflict with each other. When this happens, the system prefers to generate a "natural-looking" motion that "best" matches the user constraints rather than generating the "best" possible motion that "exactly" matches the user constraints.

Deformable motion models have demonstrated a strong generalization ability because they can interpolate/extrapolate an infinite

number of motions that are not in the database. For example, our current deformable walking model can transform a physically simulated 2D biped walking sequence [Yin et al. 2007] into a high-quality 3D motion sequence. However, the system will not generate new motions that cannot be represented by the deformable motion models. For instance, our deformable walking model cannot synthesize a motion "walking with head scratching" because the current walking database does not include similar patterns. One possibility to address this limitation is to capture or keyframe a motion "walking with head scratching", register the motion with the synthesized motion using the low-body motion data, and transfer the upper-body pattern "head scratching" to the synthesized motion [Heck et al. 2007].

Another drawback to the approach is that our motion preprocessing step is not fully automatic. For a very large dataset, the registration process could be very time consuming. However, a "bootstrap" approach can be adopted. We can first use the semi-automatic algorithm to register a small set of representative motion examples and build a deformable motion model from them. We then match the model to a new motion example. Again, we can use the semi-automatic algorithm to correct the registration result if they do not match. We can update the deformable motion model with the new motion example and use it to register another new motion example. This process is repeated, incrementally building a model until the deformable motion model can match a new motion example sufficiently accurately every time, and therefore needs no more training.

We show the deformable motion models can be applied for transforming 2D animation to 3D animation, reconstructing high-quality 3D motion from video, and detecting and removing foot-sliding artifacts of input walking sequences. We believe the deformable motion models could also be leveraged for many other applications in human motion processing, such as motion reconstruction, registration, editing, compression, and completion. For example, encoding human motion data with deformable parameters could be used to compress human motion data. One of the immediate directions for future work is, therefore, to investigate the application of the deformable motion models to human motion processing.

REFERENCES

ARIKAN, O. AND FORSYTH, D. A. 2002. Interactive motion generation from examples. *ACM Trans. Graph. 21*, 3, 483–490.

BISHOP, C. 1996. *Neural Network for Pattern Recognition*. Cambridge University Press.

BRAND, M. AND HERTZMANN, A. 2000. Style machines. In *Proceedings of the ACM SIGGRAPH International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'00)*. 183–192.

CATMULL, E. AND ROM, R. 1974. A class of local interpolating splines. In *Computer Aided Geometric Design*, Academic Press.

CHAI, J. AND HODGINS, J 2005. Performance animation from low-dimensional control signals. *ACM Trans. Graph. 24*, 3, 686–696.

CHAI, J. AND HODGINS, J. 2007. Constraint-Based motion optimization using a statistical dynamic model. *ACM Trans. Graph. 26*, 3, Article no.8.

GROCHOW, K., MARTIN, S. L., HERTZMANN, A., AND POPOVIĆ, Z. 2004. Style-Based inverse kinematics. *ACM Trans. Graph. 23*, 3, 522–531.

HECK, R., KOVAR, L., AND GLEICHER, M 2007. Splicing upper-body actions with locomotion. *Comput. Graph. Forum 25*, 3, 459–466.

HOWE, N., LEVENTON, M., AND FREEMAN, W 1999. Bayesian reconstruction of 3D human motion from single-camera video. *Adv. Neural Inform. Process. Syst. 12*, 820– 826.

KOVAR, L. AND GLEICHER, M. 2003. Registration curves. In *Proceedings of the ACM SIGGRAPH/EUROGRAPHICS Symposium on Computer Animation*. 214–224.

KOVAR, L. AND GLEICHER, M. 2004. Automated extraction and parameterization of motions in large data sets. *ACM Trans. Graph. 23*, 3, 559–568.

KOVAR, L., GLEICHER, M., AND PIGHIN, F. 2002. Motion graphs. *ACM Trans. Graph. 21*, 3, 473–482.

KWON, T. AND SHIN, S. Y. 2005. Motion modeling for on-line locomotion synthesis. In *Proceedings of the ACM SIGGRAPH Symposium on Computer Animation*. 29–38.

LEE, J., CHAI, J., REITSMA, P., HODGINS, J., AND POLLARD, N 2002. Interactive control of avatars animated with human motion data. *ACM Trans. Graph. 21*, 3, 491–500.

LEE, K. H., CHOI, M. G., AND LEE, J 2006. Motion patches: Building blocks for virtual environments annotated with motion data. *ACM Trans. Graph. 25*, 3, 898–906.

LI, Y., WANG, T., AND SHUM, H.-Y. 2002. Motion texture: A two-level statistical model for character synthesis. *ACM Trans. Graph. 21*, 3, 465–472.

LOURAKIS , M. 2009. Levmar: Nonlinear least squares algorithms in C/C++. http://www.ics.forth.gr/ lourakis/levmar/.

MOLINA TANCO, L. AND HILTON, A. 2000. Realistic synthesis of novel human movements from a database of motion capture examples. In *Proceedings of the Workshop on Human Motion*. 137–142.

MUKAI, T. AND KURIYAMA, S. 2005. Geostatistical motion interpolation. *ACM Trans. Graph. 24*, 3, 1062–1070.

MYERS, C. S. AND RABINER, L. R. 1981. A comparative study of several dynamic time-warping algorithms for connected word recognition. *The Bell Syst.Tech. J. 60*, 7, 1389–1409.

ORMONEIT, D., SIDENBLADH, H., BLACK, M., AND HASTIE, T. 2001. Learning and tracking cyclic human motion. *Adv. Neural Inform. Process. Syst. 13*, 894–900.

PAVLOVIĆ, V., REHG, J. M., AND MACCORMICK, J. 2000. Learning switching linear models of human motion. *Adv. Neural Inform. Process. Syst. 12*, 981–987.

ROSE, C., COHEN, M. F., AND BODENHEIMER, B. 1998. Verbs and adverbs: Multidimensional motion interpolation. *IEEE Comput. Graph. Appl. 18*, 5, 32–40.

SAFONOVA, A. AND HODGINS, J. K. 2007. Construction and optimal search of interpolated motion graphs. *ACM Trans. Graph. 26*, 3, Article no. 106.

SHUM, H. P. H., KOMURA, T., SHIRAISHI, M., AND YAMAZAKI, S. 2008. Interaction patches for multi-character animation. *ACM Trans. Graph. 27*, 5, Article no. 114.

SIDENBLADH, H., BLACK, M. J., AND SIGAL, L. 2002. Implicit probabilistic models of human motion for synthesis and tracking. In *Proceedings of the European Conference on Computer Vision*. 784–800.

TREUILLE, A., LEE, Y., AND POPOVIĆ, Z. 2007. Near-Optimal character animation with continuous control. *ACM Trans. Graph. 26*, 3, Article no. 7.

WITKIN, A. AND KASS, M. 1988. Spacetime constraints. In *Proceedings of the ACM SIGGRAPH International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'88)*. 159–168.

YIN, K., LOKEN, K., AND VAN DE PANNE, M. 2007. Simbicon: Simple biped locomotion control *ACM Trans. Graph. 26*, 3, Article no. 105.

Received May 2009; accepted August 2009