# Constraint-based Motion Optimization Using A Statistical Dynamic Model

Jinxiang Chai*
Texas A&M University

Jessica K. Hodgins†
Carnegie Mellon University

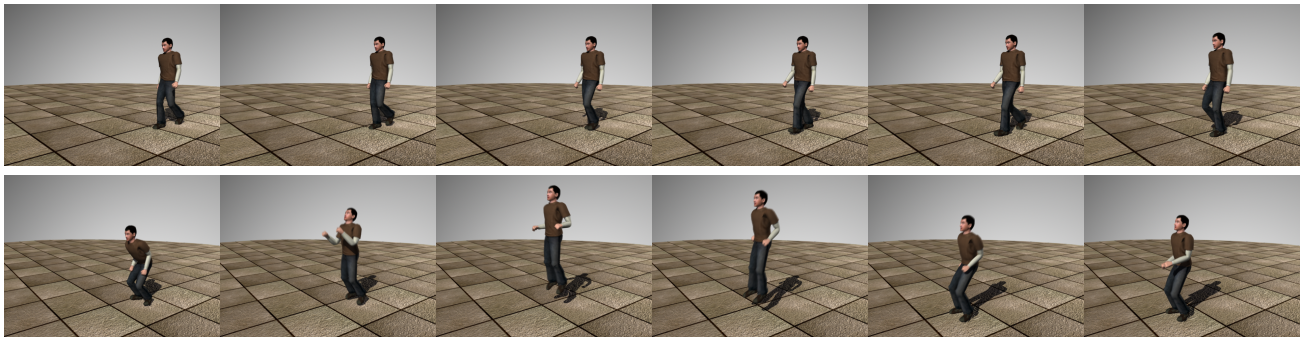

**Figure 1:** *Motions computed from spatial-temporal constraints.*

## Abstract

In this paper, we present a technique for generating animation from a variety of user-defined constraints. We pose constraint-based motion synthesis as a *maximum a posterior* (MAP) problem and develop an optimization framework that generates natural motion satisfying user constraints. The system automatically learns a statistical dynamic model from motion capture data and then enforces it as a motion prior. This motion prior, together with user-defined constraints, comprises a trajectory optimization problem. Solving this problem in the low-dimensional space yields optimal natural motion that achieves the goals specified by the user. We demonstrate the effectiveness of this approach by generating whole-body and facial motion from a variety of spatial-temporal constraints.

**CR Categories:** I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—animation; I.3.6 [Computer Graphics]: Methodology and Techniques—interaction techniques

**Keywords:** human body animation, facial animation, motion control, statistical dynamic models, spatial-temporal constraints, constraint-based motion synthesis, motion capture data

## 1 Introduction

Our objective in this paper is to design an animation system that allows users to easily create natural-looking character animation by specifying spatial-temporal constraints throughout the motion. For example, a naive user might use a performance animation system to control the trajectories of the end-positions of the limbs of a character. A more skilled user might specify a small set of poses at key time instants. The system then automatically finds a motion that best satisfies those constraints. An ideal motion synthesis system should allow users to specify a variety of constraints either at isolated points or across the entire motion in order to accommodate users with different skill levels.

One appealing solution to this problem is physically based optimization [Witkin and Kass 1988], which allows the user to specify various constraints throughout the motion and relies on optimization to compute the physically valid motion that best satisfies these constraints. Unfortunately, correct physics does not ensure that the motion will appear natural for characters with many degrees of freedom. Like physically based optimization, we formulate the problem as a trajectory optimization and consider the entire motion simultaneously. Instead of using the physical laws to generate physically correct animation, we rely on statistical models of human motion to generate a statistically plausible motion.

Our approach allows the user to generate a wide range of human body and facial animation by specifying spatial-temporal constraints throughout the motion. The system automatically learns a statistical dynamic model from motion capture data and then enforces this model as a motion prior. The statistical dynamic model plays a role similar to that played by the dynamics in physically based optimization because it constrains the motion to only part of the space of possible human motions. The statistical dynamic model, however, is usually lower dimensional than the dynamics model, making the optimization more efficient, less likely to be subject to local minima, and more likely to produce natural motion.

We demonstrate the effectiveness of this approach in two domains: human body animation and facial animation. We show that the system can generate natural-looking animation from key-frame constraints, key-trajectory constraints, and a combination of these two constraints. For example, the user can generate a walking animation from a small set of key frames and foot contact constraints (figure 1 top). The user can also specify a small set of key trajectories for the root, hands and feet positions to generate a realistic jumping motion (figure 1 bottom). The user can fine tune the animation by incrementally modifying the constraints. For example, the user can create a slightly different jumping motion by adjusting

*e-mail: jchai@cs.tamu.edu
†e-mail: jkh@cs.cmu.edu

the positions of both hands at the top of the jump. The system can generate motions for a character whose skeletal model is markedly different from those of the subjects in the database. We also show that the system can use a statistical dynamic model learned from a normal walking sequence to create new motion such as walking on a slope.

The quality of the final animation produced by our system depends on the motion priors derived from the motion capture database and the number of user-defined constraints. We, therefore, evaluate how the database influences the final motion and how increasing or decreasing the number of user-defined constraints influences the final animation. We also compare alternative techniques for generating animation from user-defined constraints such as linear interpolation, trajectory-based inverse kinematics, and inverse kinematics in a PCA subspace.

## 2 Background

In this paper, we construct statistical models from motion capture data and then combine these models with trajectory optimization to generate a motion that satisfies user-defined constraints. Consequently, we discuss related work in constraint-based trajectory optimization and data-driven animation with an emphasis on statistical models.

### 2.1 Constraint-based Trajectory Optimization

Trajectory optimization methods, which were first introduced to the graphics community by Witkin and Kass [1988], provide a powerful framework for generating character animation from user-specified constraints, physics constraints, and an objective function that measures the performance of a generated motion. Extending this approach to generate natural motion for a full human character has proved to be hard because the system is high dimensional, the physics constraints make it highly nonlinear, and defining an objective function that reliably measures the naturalness of human motion is difficult.

Much of the difficulty in solving this problem appears to result from the physics constraints because optimization without physics is effective for editing [Gleicher 1998]. Therefore, one way to make the problem tractable is to simplify the governing physical laws. Both Liu and Popović [2002] and Abe and his colleagues [2004] showed that many dynamic effects can be preserved by enforcing patterns of linear and angular momentum during the motion. Reformulating the dynamics to avoid directly computing the torques also provides a significant performance improvement [Fang and Pollard 2003].

Reducing the number of degrees of freedom to be optimized can also create tractable problems. For example, Popović and Witkin [1999] showed that significant changes to motion capture data can be made by manually reducing the degrees of freedom to those most important for the task. Safonova and her colleagues [2004] demonstrated that an efficient optimization can be achieved in a behavior-specific, low-dimensional space without simplifying the dynamics. More recently, Liu and her colleagues [2005] introduced a novel optimization framework—Nonlinear Inverse Optimization—for optimizing appropriate parameters of the objective function from a small set of motion examples and then used the estimated parameters to synthesize a new locomotion.

Our work also uses a trajectory optimization framework but replaces the physical dynamic model with a statistical dynamic model computed from a motion capture database.

### 2.2 Data-driven Motion Synthesis

Our approach is also part of an alternative set of techniques that relies on motion data to constrain the search to natural looking motions. For example, motion graphs can be used to resequence whole-body or facial motions (see, for example, [Arikan and Forsyth 2002; Kovar et al. 2002; Lee et al. 2002; Zhang et al. 2004]. These systems cannot match poses or satisfy such kinematic constraints as end effector constraints unless the motion database happens to contain a motion that satisfies those constraints. Motion interpolation, on the other hand, does allow isolated constraints to be satisfied (for example, [Rose et al. 1998; Kovar and Gleicher 2004; Mukai and Kuriyama 2005]). However, interpolation across a complete behavior does not have enough degrees of freedom to allow the specification of full pose constraints or end effector constraints across multiple frames. Recently, interpolation and motion graphs have been combined to obtain some of the advantages of each approach [Safonova and Hodgins 2007].

Statistical models of human motion have also been used for motion synthesis. A number of researchers have used variants of Hidden Markov Models (HMMs) to statistically represent human motion: either full-body movements [Molina Tanco and Hilton 2000; Brand and Hertzmann 2000; Galata et al. 2001] or speech-driven facial expressions [Bregler et al. 1997; Brand 1999]. HMMs learned from human motion data have been used to interpolate key frames [Molina Tanco and Hilton 2000; Galata et al. 2001], synthesize a new style of motion [Brand and Hertzmann 2000], and generate facial expressions from speech signals [Bregler et al. 1997; Brand 1999]. Grzeszczuk and his colleagues[1998] developed a neural network approximation of dynamics based on simulated data and use it to animate dynamic models such as fish and lunar landers. Urtasun and her colleagues[2006] learned linear motion models from pre-aligned motion data via Principal Component Analysis (PCA) and used them to track 3D human body movements from video by performing nonlinear optimization over a small sliding temporal window. Switching linear dynamic system (SLDS) have also been used to model human motion. Pavlović and his colleagues [2000] present results for human motion synthesis, classification, and visual tracking using learned SLDS models. Li and his colleagues [2002] used SLDS to synthesize and edit disco dancing motion.

Our approach is also to learn a statistical dynamic model from human motion capture data; however, the dynamic behavior of our model is controlled by a continuous control state rather than a discrete hidden state as in HMMs and SLDS. This property led us to formulate the motion synthesis problem as a trajectory optimization problem. More importantly, our system allows the user to specify a variety of spatial-temporal constraints such as end effector constraints throughout the motion, a capability that has not been demonstrated by previous approaches.

A number of researchers have developed statistical models for human poses and used them to solve the inverse kinematics problem. Grochow and colleagues [2004] applied a global nonlinear dimensionality reduction technique, Gaussian Process Latent Variable Model, to human motion data and then used the learned statistical pose model to compute poses from a small set of user-defined constraints. Another solution for data-driven inverse kinematics is to interpolate a small set of preexisting examples using constraints. This idea has been used to compute human body poses [Rose et al. 2001] and facial expressions [Zhang et al. 2004] from kinematic constraints at a single frame. These models lack temporal information and therefore cannot be used to generate an animation from sparse constraints such as key frames.

Local statistical models are sufficient if the user provides continuous

control signals (the performance animation problem). Chai and colleagues [2003] presented a real-time vision-based performance animation system that transforms a small set of automatically tracked facial features into facial animation by interpolating examples in a database at run time. They also used a series of local statistical pose models constructed at run time to reconstruct full-body motion from continuous, low-dimensional control signals obtained from video cameras [Chai and Hodgins 2005].

The statistical dynamic model used in this paper was motivated by the dynamic model used for video textures by Soatto and his colleagues [2001]. They showed that a sequence of images of such moving scenes as sea-waves, smoke, and whirlwinds can be modeled by second-order linear dynamic systems. They applied the learned dynamic systems to synthesize an "infinite length" texture sequence by sampling noise from a known Gaussian distribution. We extend the model to learn an efficient and low-dimensional representation of human motion and use it to generate an animation that achieves the goal specified by the user.

## 3 Overview

The key idea behind our approach is that motion priors learned from prerecorded motion data can be used to create natural human motion that matches constraints specified by the user. The combination of the motion prior and the user's constraints provides sufficient information to produce motion with a natural appearance.

The human body motion capture database (about 15 minutes) includes data of locomotion (jumping, running, walking, and hopping) and interacting with the environment (standing up/sitting down, reaching/picking up/placing an object). The facial expression database (about 9 minutes) includes six basic facial expressions (happiness, surprise, disgust, fear, anger, sadness) and three facial movements related to everyday life (speaking, eating, and snoring). The motion was captured with a Vicon motion capture system of 12 MX-40 cameras [Vicon Systems 2004] with 41 markers for full-body movements and 92 markers for facial expressions. The motion was captured at 120Hz and then downsampled to 30Hz.

We denote the set of motion capture data in the database as $\mathbf{y}_{1:N} = [\mathbf{y}_1, ..., \mathbf{y}_N]$, where $\mathbf{y}_n, n = 1, ..., N$, is the measurement of the character's configuration in the $n$th frame. In facial animation, $\mathbf{y}_n$ is the 3D positions of all vertices on the face model. In human body animation, $\mathbf{y}_n$ is the position and orientation of the root and the joint angles.

We preprocess the motion capture data by applying Principal Component Analysis (PCA) [Bishop 1996] to the motion capture data and obtain a reduced subspace representation for $\mathbf{y}_n$:

$$\mathbf{y}_n = C \cdot \mathbf{x}_n + D \tag{1}$$

where the vector $\mathbf{x}_n \in R^{d_x}$ is a low-dimensional representation of the character configuration $\mathbf{y}_n \in R^{d_y}$. The matrix $C$ is constructed from the eigenvectors corresponding to the largest eigenvalues of the covariance matrix of the data, and D is the mean of all example data, $D = (\Sigma_{n=1}^{N} \mathbf{y}_n)/N$. The dimensionality of the system state, $d_x$, can be automatically determined by choosing the $d_x$ for which the singular values drop below a threshold.

The constraints defined by the user are represented by $E = \{\mathbf{e}_j | j = 1, ..., J\}$. The goal of our constraint-based motion synthesis problem is to create an animation, $H$, based on the constraints, $E$.

We formulate the constraint-based motion synthesis in a maximum a posterior (MAP) framework and consider the entire motion simultaneously. From Bayes' theorem, the goal of MAP is to infer the most likely motion, $H$, given the user-defined constraints, $E$:

$$\begin{aligned} \arg\max_H \ p(H|E) &= \arg\max_H \ \frac{p(E|H)p(H)}{p(E)} \\ &\propto \arg\max_H \ p(E|H)p(H) \end{aligned} \tag{2}$$

where $p(E)$is the normalizing constant that ensures that the posterior distribution on the left-hand side is a valid probability density and integrates to one.

In our implementation, we minimize the negative log of $p(H|E)$, yielding the following optimization for motion $\hat{H}$:

$$\hat{H} = \arg\min_H \ -\ln p(E|H) \ -\ln p(H) \tag{3}$$

where the first term measures how well a motion sequence matches the user-specified constraints and the second term measures a priori likelihood of the motion sequence using the knowledge embedded in human motion data.

The system contains three major components:

**Motion prior.** The system first automatically learns a statistical dynamic model from motion capture data. This model is then used to compute the motion prior, $-\ln p(H)$.

**User-defined Constraints.** The user defines various forms of constraints, $E$, throughout the motion, which are then used to compute the likelihood term, $-\ln p(E|H)$. The constraints could be any kinematic constraints such as position, orientation, or the distance between two points on the character. They could be specified either at isolated points (key frames) or across the whole motion (key trajectories).

**Motion optimization.** The system uses trajectory optimization to automatically find an animation $\hat{H}$ that best satisfies the user-specified constraints while matching the statistical properties of the motion capture data: $\hat{H} = \arg\min_H \ -\ln p(E|H) \ -\ln p(H)$.

We describe the statistical model in the next section and then present the three components in detail in section 5.

## 4 Motion Analysis

We use an $m$-order linear time-invariant system to describe the dynamical behavior of the captured motion in the low-dimensional space [Ljung 1999]:

$$\mathbf{x}_n = \sum_{i=1}^{m} A_i \mathbf{x}_{n-i} + B\mathbf{u}_n \tag{4}$$

where $m$ is the order of the linear dynamic model. $\mathbf{x}_n \in R^{d_x}$ and $\mathbf{u}_n \in R^{d_u}$ are the system state and control input, and $d_u$ is the dimensionality of the control input $\mathbf{u}_n$. This formulation is similar to the linear time-invariant control system commonly adopted in the control community [Palm 1999].

However, the matrix $B$ is not unique because the control input $\mathbf{u}_t$ is unknown. Therefore, any non-singular transformation of the matrix $B$ represents the motion because $BT$ and $T^{-1}\mathbf{u}_n$ are also consistent with the dynamic model. To remove this ambiguity, we assume that the matrix $B$ is an orthogonal matrix.

Given the low-dimensional representation of the original motion capture data, $\mathbf{x}_{1:N} = [\mathbf{x}_1, ..., \mathbf{x}_N]$, we want to identify the state-space model, including system matrices $\{A_i | i = 1, ..., m\}$, $B$, and the corresponding control input $\mathbf{u}_{m+1:N} = [\mathbf{u}_{m+1}, ..., \mathbf{u}_N]$.

The matrices $\{A_i | i = 1, ..., m\}$ are dependent on the distribution of $\mathbf{u}_n$. To eliminate the ambiguity of the matrices $A_i$, we seek to

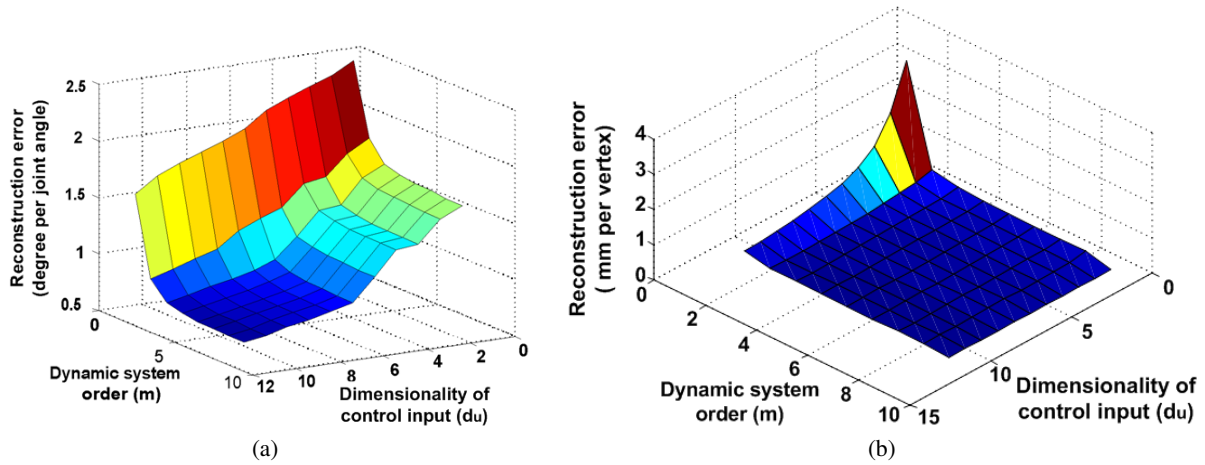


**Figure 2:** *The average reconstruction error of the linear time-invariant system computed by cross-validation techniques: (a) The average per frame reconstruction error for the walking test data as a function of the order of the dynamic system ($m$) and the number of dimensions of the control input ($d_u$); (b) The average per frame reconstruction error of the facial test data as a function of the order of the dynamic system ($m$) and the number of dimensions of the control input ($d_u$).*

find the $\{A_i | i = 1, ..., m\}$ that minimize the sum of the squared control input $\mathbf{u}_n$:

$$\hat{A}_1, ..., \hat{A}_m \quad = \quad \arg\min_{A_1,...,A_m} \sum_n \|\mathbf{u}_n\|^2 \qquad (5)$$

The matrices $A_i$ can thus be uniquely found by computing the least-square solution:

$$\hat{A}_1, ..., \hat{A}_m = \arg\min_{A_1,...,Am} \sum_{n=m+1}^{N} \|\mathbf{x}_n - \sum_{i=1}^{m} A_i \mathbf{x}_{n-i}\|^2 \qquad (6)$$

We use the estimated matrices $\{A_i | i = 1, ..., m\}$ to compute the control input term:

$$\hat{\mathbf{z}}_n = \mathbf{x}_n - \sum_{i=1}^{m} \hat{A}_i \mathbf{x}_{n-i}, \quad n = m+1, ..., N \qquad (7)$$

We form a $d_x \times (N - m)$ matrix by stacking the estimated control inputs $\hat{\mathbf{z}}_n$:

$$\underbrace{\left( \hat{\mathbf{z}}_{m+1} \mid ... \mid \hat{\mathbf{z}}_N \right)}_{Z} \quad = \quad B \cdot \underbrace{\left( \mathbf{u}_{m+1} \mid ... \mid \mathbf{u}_N \right)}_{U} \qquad (8)$$

Equation (8) shows that without noise, the rank of the matrix $Z$ is $d_u$. Therefore, we can automatically determine the dimensionality of the control input $\mathbf{u}_n$ by computing the rank of matrix $Z$.

When noise corrupts the motion capture data, the data matrix $Z$ will not be exactly of rank $d_u$. However, we can perform singular value decomposition (SVD) on the data matrix $Z$ such that $Z = WSV^T$, and then get the best possible rank $d_u$ approximation of the data matrix, factoring it into two matrices: $\hat{B} = W$ and $\hat{U} = SV^T$, where $\hat{B}$ is a $d_x \times d_u$ matrix and $\hat{U}$ is a $d_u \times (T - m)$ matrix. The dimensionality of the control input ($d_u$) can be automatically determined by choosing the $d_u$ for which the singular values drop below a threshold.

Functionally, a statistical dynamic model is similar to a physical dynamic model. For example, given initial values of the system state ($\mathbf{x}_{1:m} = [\mathbf{x}_1, ..., \mathbf{x}_m]$), the linear dynamic model in Equation (4) can be used to generate an animation ($\mathbf{x}_{m+1:T} = [\mathbf{x}_{m+1}, ..., \mathbf{x}_T]$) by sequentially choosing an appropriate value for the control input ($\mathbf{u}_{m+1:T} = [\mathbf{u}_{m+1}, ..., \mathbf{u}_T]$), just as joint torques would be used to advance a physical model through time. The main advantage of using a statistical dynamic model for animation is that the dimensionality of the control input in a statistical dynamic model is usually much lower than a physical dynamic model. Therefore, the statistical dynamic model might achieve faster convergence and be less subject to local minima.

The number of dimensions of the control input, $d_u$, characterizes the complexity of our dynamic model. Figure 2 plots the reconstruction error of a walking test data set and a facial test data set as a function of the order of the dynamic system ($m$) and the number of dimensions of the control input, $d_u$. The walk data set is from multiple subjects and contains different styles. The facial expression data are from the same subject and contain a variety of facial expressions such as "happy" and "sad." The average reconstruction error is the $L_2$ distance between the original test motion and the motion reconstructed from the linear time-invariant system and computed by cross-validation techniques. We observe that the reconstruction error of the statistical model decreases as both the order of dynamic system and the number of dimensions of the control input increases. If we choose $d_u$ as "zero" (simply dropping off the control term), our model becomes the linear dynamic model used by Soatto and colleagues [2001] and has the largest reconstruction error. If $d_u$ is equal to the number of dimensions of the system state $d_x$, the model can be used to represent an arbitrary motion sequence with zero error.

In practice, human motion is highly coordinated, and the dimensionality of the control input for accurate motion representation, $d_u$, is often much lower than the dimensionality of the system state, $d_x$. For the examples reported here, we set the dynamic order to three and the dimensionality of control input to four for human body animation (the reconstruction error is about 0.7 degrees/joint per frame); we set the dynamic order to two and the dimensionality of control input to one for facial movement (the reconstruction error is about 0.1 mm/vertex per frame).

## 5 Constraint-based Motion Synthesis

Constraint-based motion synthesis provides the user with intuitive control over the resulting motion: the user specifies a desired motion with various forms of constraints, such as key frames, end effector target positions, or joint angle values; the system then auto-

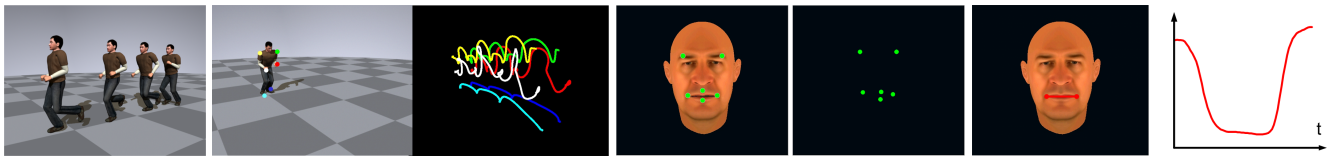


**Figure 3:** *Various form of spatial-temporal constraints: (a) key-frame constraints for creating full-body animation; (b) key-trajectory constraints where the user selects six points on the character and then specifies their 3D trajectories across the motion (from a performance animation interface); (c) the user picks six points on the face and their screen-space position constraints at some moment in time; (d) the user defines a distance between two facial points (the width of the mouth) and controls the distance throughout the motion.*

matically finds the animation that best satisfies the user-specified constraints while matching the spatial-temporal properties of the motion capture data. This section first derives the likelihood term, $-\ln p(E|H)$, based on the user-defined constraints, $E$. We then model the motion prior term, $-\ln p(H)$, using the learned statistical dynamic model. Finally, we discuss how to optimize motion by combining both terms: $\hat{H} = \arg\min_H -\ln p(E|H) - \ln p(H)$.

Like physically based optimization [Witkin and Kass 1988], we represent the system state $\mathbf{x}_t$ and the control signal $\mathbf{u}_t$ independently. The motion to be synthesized is therefore represented as a sequence of system states and control inputs $H = (\mathbf{x}_1, ..., \mathbf{x}_T, ..., \mathbf{u}_{m+1}, ..., \mathbf{u}_T)$.

### 5.1 User-defined Constraints

The system allows the user to specify various forms of kinematic constraints $E = \{\mathbf{e}_j | j = 1, ..., J\}$ throughout the motion or at isolated points in the motion. For facial animation, the user can specify the positions or orientations of any points on the face, or the distance between any two points. For whole-body animation, the user can specify the positions or orientations of any points on the body, or joint angle values for any joints. Rather than requiring that constraints be specified in 3D, it is often more intuitive to specify where the projection of a point on the character should be located. Therefore, the system also allows the user to specify the 2D projections of any 3D point on a user-defined screen space. This approach could be used for rotoscoping a video, or for a single camera performance animation.

The system allows the user to sketch out the motion in greater or lesser detail. For example, a novice user might want to control the paths of specific joints or paths over a period of time using a performance animation system while a more skilled user might prefer using key frame constraints. Spatially, the constraints could provide either an exact configuration such as a full-body pose or a small subset of the joint angles or end-positions. Temporally, the constraints could be instantaneous constraints for a particular frame, multiple-frame constraints, or continuous constraints over a period of time.

User-defined constraints can be linear or nonlinear. Linear constraints can be used to define joint angle constraints in human body animation and positions in facial animation. The most common nonlinear constraints in human body animation might be end effector constraints, for example, foot contact constraints. In facial animation, nonlinear constraints can be used to specify the distance between two points on the face or 2D projections of 3D facial points. Figure 3 illustrates the user-defined constraints that were used to generate human body animation and facial animation.

Mathematically, we can model the likelihood term, $-\ln p(E|H)$, as follows:

$$\begin{aligned} E_{constraints} &= -\ln p(E|H) \\ &\sim \sum_{j=1}^{J} \beta \|\mathbf{e}_j - \mathbf{f}_j(\mathbf{y}_1, ..., \mathbf{y}_T)\|^2 \\ &\sim \sum_{j=1}^{J} \beta \|\mathbf{e}_j - \mathbf{f}_j(C\mathbf{x}_1 + D, ..., C\mathbf{x}_T + D)\|^2 \end{aligned} \tag{9}$$

where the function $\mathbf{f}_j$ is usually a forward kinematics function and the parameter $\beta$ is a constant specifying the importance of the constraints. The likelihood term evaluates how well the synthesized motion matches the constraints specified by the user. A good match between the motion and the user-defined constraints results in a low energy solution.

### 5.2 Motion Priors

Many motions might satisfy the user-defined constraints. For example, when the user specifies a small set of key frames or key trajectories, the number of constraints is not sufficient to completely determine the whole motion sequence, $\mathbf{x}_{1:T}$.

To remove ambiguities, we would like to constrain the generated motion to lie in the space of natural human motions by imposing a prior on the generated motion:

$$\begin{aligned} E_{prior} &= -\ln p(H) \\ &= -\ln p(\mathbf{x}_{1:T}, \mathbf{u}_{m+1:T}) \end{aligned} \tag{10}$$

Based on the statistical dynamic equation (Equation 4), the current system state $\mathbf{x}_t$ only depends on the previous system states $\mathbf{x}_{t-m:t-1}$ and the current control input $\mathbf{u}_t$. We have

$$\begin{aligned} p(H) &= p(\mathbf{x}_{1:T}, \mathbf{u}_{m+1:T}) \\ &= \prod_{t=m+1}^{T} p(\mathbf{x}_t | \mathbf{x}_{t-1:t-m}, \mathbf{u}_t) \cdot p(\mathbf{x}_{1:m}, \mathbf{u}_{m+1:T}) \end{aligned} \tag{11}$$

We assume that the likelihood of the first term on the right side of Equation 11 is measured by the deviation of the statistical dynamic equation (Equation 4). We have the corresponding energy term

$$\begin{aligned} E_{prior}^{dynamic} &= -\ln \prod_{t=m+1}^{T} p(\mathbf{x}_t | \mathbf{x}_{t-1:t-m}, \mathbf{u}_t) \\ &\sim -\alpha \sum_{t=m+1}^{T} \|\mathbf{x}_t - \sum_{i=1}^{m} A_i \mathbf{x}_{t-i} - B\mathbf{u}_t\|^2 \end{aligned} \tag{12}$$

where $\alpha$ is a tuning parameter. Conceptually, the dynamic prior can be thought as dimensionality reduction of the motion in a spatial-temporal domain. It significantly reduces the dimensionality of the motion from the space of $\mathbf{x}_{1:T}$ to the space of the initial state $\mathbf{x}_{1:m}$ and the control input $\mathbf{u}_{m+1:T}$.

The second term on the right side of Equation 11 computes the prior for the initial state, $\mathbf{x}_{1:m}$, and control input, $\mathbf{u}_{m+1:T}$. We assume that both the initial state, $\mathbf{x}_{1:m}$, and the control input, $\mathbf{u}_t$, are independent and identically distributed. The energy term for the second

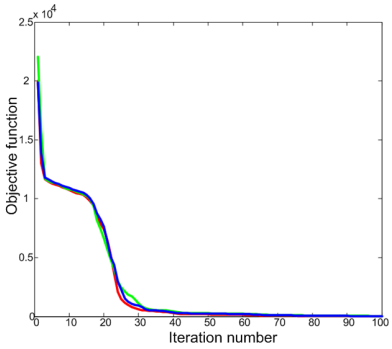


**Figure 4:** *The horizontal axis shows the iteration number and the vertical axis shows the value of the objective function for whole-body optimization The three colored curves show the evolution of the objective function values with three different initial guesses. The optimization converges within 100 iterations.*

term on the right side of Equation 11 can be simplified as follows:

$$\begin{aligned} E_{prior}^{control} &= -\ln p(\mathbf{x}_{1:m}, \mathbf{u}_{m+1:T}) \\ &= -\sum_{t=1}^{m} \ln p(\mathbf{x}_t) - \sum_{t=m+1}^{T} \ln p(\mathbf{u}_t) \end{aligned} \quad (13)$$

We model the control input $(\mathbf{u}_t)$ as a mixture with $K$ component Gaussian densities [Bishop 1996]:

$$p(\mathbf{u}_t) = \Sigma_{k=1}^{K} \pi_k N(\mathbf{u}_t; \phi_k, \Lambda_k) \quad (14)$$

where $K$ is the number of Gaussian density models and $\pi_k$ is a mixing parameter that corresponds to the prior probability that $\mathbf{u}_t$ was generated by the $k$th component. The function $N(\mathbf{u}_t; \phi_j, \Lambda_j)$ denotes the multivariate normal density function with mean $\phi_j$ and covariance matrix $\Lambda_j$. The parameters of the Gaussian mixture models $(\pi_k, \phi_k, \Lambda_k)$ are automatically estimated using an Expectation-Maximization (EM) algorithm [Bishop 1996]. The training data are the values of control inputs $\{\hat{\mathbf{u}}_n\}$ computed from the original motion capture data $(\{\mathbf{y}_n | n = 1, ..., N\})$ (see section 4). The density function of the initial states, $p(\mathbf{x}_t)$, $t = 1, ..., m$, is also modeled as a mixture of multivariate Gaussian distributions whose parameters are learned from motion data, $\mathbf{x}_{1:N}$, using the EM algorithm.

Note that we choose weak priors (static models) to model the priors for both initial states and control inputs so as not to restrict the type of motions the algorithm can generate.

### 5.3 Motion Optimization

After combining the user-defined constraints and the motion prior, the constraint-based motion synthesis problem becomes the following unconstrained motion optimization problem:

$$\arg\min_{\mathbf{x},\mathbf{u}} \quad E_{constraint} + E_{prior}^{dynamic} + E_{prior}^{control} \quad (15)$$

where $\mathbf{x}$ and $\mathbf{u}$ are the concatenation of the system states $\mathbf{x}_t$ and the concatenation of the control signals $\mathbf{u}_t$ over the entire motion.

We follow a standard approach of representing $\mathbf{x}_t$ and $\mathbf{u}_t$ using cubic B-splines. We solve the optimization problem using sequential quadratic programming (SQP) [Bazaraa et al. 1993], where each iteration solves a quadratic programming subproblem. The Jacobian matrix and the Hessian matrix of the energy function are symbolically evaluated at each iteration.

We choose all initial values using random values between 0 and 1 except that a linear interpolation of the user-specified keyframe constraints is used for initialization. We found that the optimization procedure always converges quickly (usually less than 100 iterations and less than 30 seconds). Typically, the objective function values decrease rapidly in the early iterations and then level off as they approach the optimal value. Figure 4 shows the objective function values for three different initial guesses.

Our optimization framework can also be applied to the problem of generating human body motions for a character whose skeletal model is markedly different from the subjects in the database. User-defined constraints for motion retargeting can either be directly computed from the source motion or specified by the user. In our experiment, we extract foot positions from a source walking motion and then use it to generate a walking sequence for a new character. We also add one term in the objective function that measures the difference between the source motion and retargeted motion:

$$E_{diff} = \sum_{t=1}^{T} \|\mathbf{y}_t^{source} - C\mathbf{x}_t - D\|^2 \quad (16)$$

where $\mathbf{y}_t^{source}$ is the source pose at frame $t$.

## 6 Results

We test our system by generating both human body animation and facial animation from various forms of user-defined constraints. We also evaluate the performance of our algorithm in terms of the motion priors and user-defined constraints.

### 6.1 Full-body Animation

We learn the statistical model for each individual behavior and use it to generate individual behavior based on user-defined constraints. Two kinds of constraints were used to generate most of the examples in this paper: key-frame constraints and key-trajectory constraints. We can also combine these two constraints. For example, a jumping motion can be created by specifying a start pose and the positions of both feet and root throughout the motion. The accompanying video demonstrates the effectiveness of our system for generating a number of individual behaviors, including walking, running, and jumping. Our behavior-specific statistical motion model is capable of generating a rich variety of actions. For example, we can use a small set of key frames and foot contacts to generate normal walking, climbing over an obstacle, a baby walking, and mickey-mouse style walking. Figure 5 shows sample frames of the results.

Our system can also synthesize motion that transitions from one behavior to another by using the statistical model learned from transition data. In the accompanying video, we demonstrate that the user can generate a transition from walking to jumping, from walking to sitting down, and from walking to picking up an object (figure 6). The accompanying video also shows that the system can generate motions for characters with skeletal dimensions different from those in the database. Figure 7 shows sample frames of the results. We also show that we can use motion priors learned from a small sequence of a normal walking motion (about 100 frames) to create walking on a slope and walking with small steps.

The user can refine the animation by incrementally modifying the constraints. For example, the user can create a slightly different jumping motion by adjusting the positions of both hands at the top of the jump. Figure 8 shows sample frames of the results.

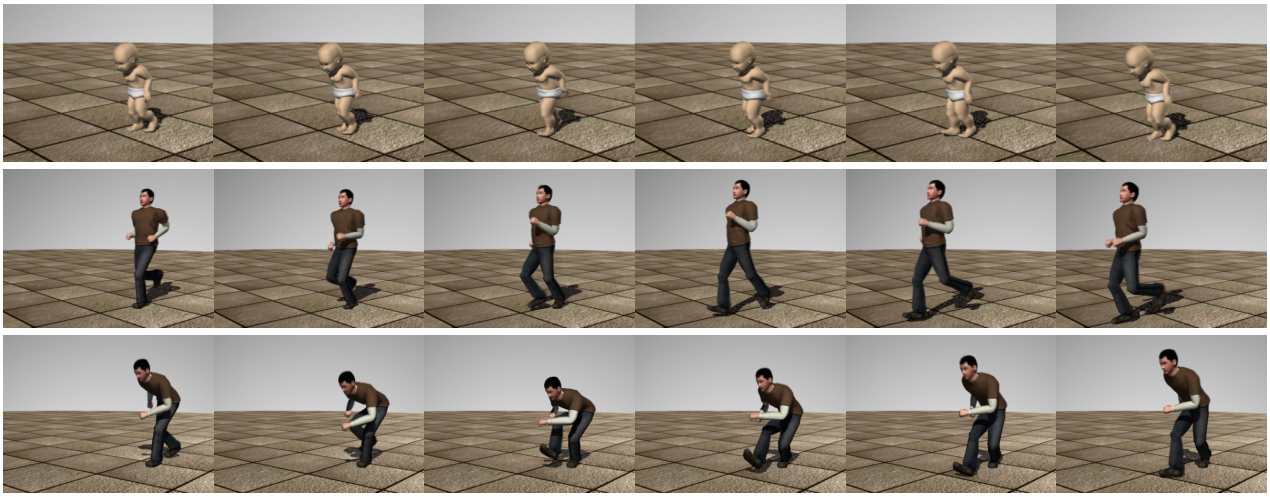

**Figure 5:** *Animation generated by a small set of key frames: (top) baby walking; (middle) running; (bottom) stylized walking.*

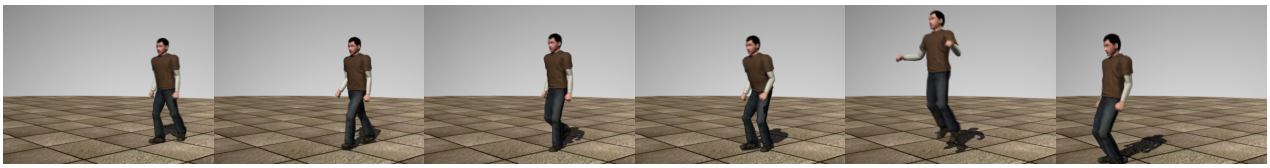

**Figure 6:** *Our system can generate a transition from walking to jumping using a small set of key frames.*

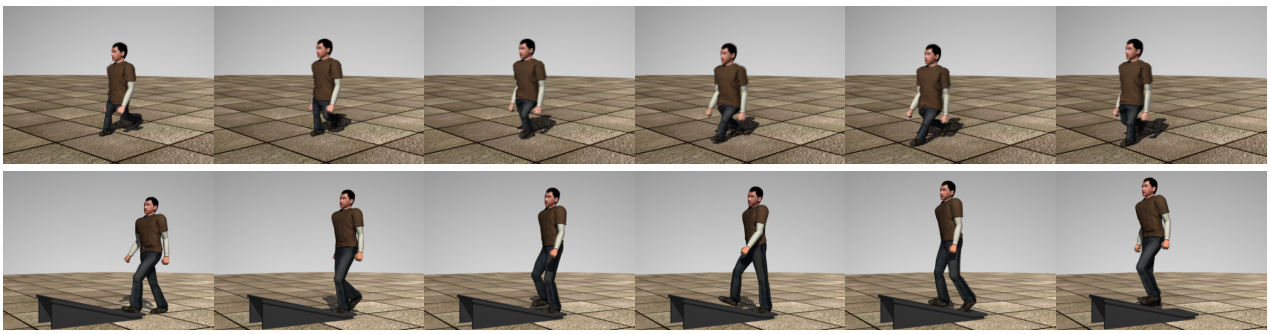

**Figure 7:** *Motion generality: (top) The system generates motion for a character whose skeletal dimensions are different from the subjects in the database; (bottom) The system modifies a normal walking motion to create a new motion–walking on a slope.*

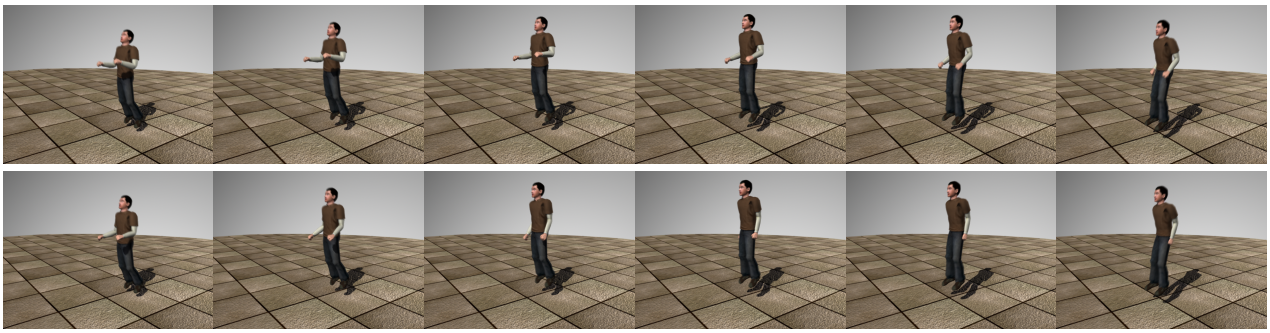

**Figure 8:** *The user can fine tune an animation by incrementally adding constraints: (top) jumping generated by the user using five key trajectories (both hands, both feet, and root); (bottom) a slightly different jumping motion generated after adjusting the positions of the hands at the top of the jump.*

### 6.2 Facial Animation

The system learns a single statistical model from the whole facial motion capture database and then uses it to create facial animation with a variety of spatial-temporal constraints. The following examples are illustrated in the accompanying video:

**Combination of keyframe and trajectory constraints.** The user can generate realistic facial animation by combining sparse keyframe constraints (three key frames) and sparse trajectory constraints (one trajectory).

**Sparse screen constraints.** The user selects six points on the face and specifies the 2D projections on the screen space at three key instants. This type of constraint could be extracted by rotoscoping.

**Trajectory constraints.** The user can achieve detailed control over facial movement by specifying the trajectories of a small set of 3D facial points. The user can also use trajectories of a small set of high-level facial features (the mouth width and height and the openness of the eyes) to generate facial animation.

### 6.3 Evaluation

The quality of the final animation depends on the motion priors and the user-defined constraints. We, therefore, have designed a number of experiments to evaluate the performance of our algorithm:

**The importance of the motion priors.** We evaluate the importance of motion priors by comparing our method against alternative constraint-based motion synthesis methods. The first method is a simple linear interpolation of key frames. The second method is trajectory-based inverse kinematics that minimizes the velocity changes of the motion in the original configuration space, $\mathbf{y}_t$, without any priors. The third method is a simple data-driven inverse kinematics algorithm that minimizes the velocity changes of the motion in a reduced PCA space, $\mathbf{x}_t$. We compare the methods using key-frame constraints and key-trajectory constraints. We keep the constraints constant and use a cubic spline to represent the motion. The results of this comparison are shown in the video. Without the use of the statistical dynamic model, the system can not generate natural motions unless the user specifies a very detailed set of constraints across the entire motion.

**Motion priors from different databases.** We evaluate how the database influences the final motion by keeping the user-defined constraints constant. We have experimented with both key-frame and key-trajectory constraints. For key-frame constraints, the user defined a sparse set of walking constraints and used them to generate walking motion from the priors learned from a number of different databases. We compare the results for a database of general locomotion, running, hopping, jumping and walking. The accompanying video shows that we can generate a good walking motion with a walking database. The quality of the animation becomes worse when we use a large and general locomotion database to generate walking. As would be expected, the system fails to generate a good walking motion if the motion prior is learned from running, hopping, or jumping data. We have tested the creation of jumping motion from key-trajectory jumping constraints when the prior is learned from a database of jumping, general locomotion, or walking. Similarly, the prior from a walking database fails to generate a good jumping motion because of the mismatch between the prior and the user-defined constraints.

**Different numbers of constraints.** With an appropriate database, we compare the quality of motions generated by different numbers of constraints. More specifically, we take one motion sequence out of the database and use it as a testing sequence. We then compare the animations created by key frames that are spaced increasingly far apart in time. We also compare the results by decreasing the number of key trajectories. The accompanying video shows that results become worse when we decrease the number of the user-defined constraints. For example, the numerical error increases steadily (0.94, 1.06, 1.81 degrees per joint per frame) when the number of constraints is decreased (6, 4, 2 key frames). We observe a noticeable foot sliding artifact on one foot when two key trajectories (root and one foot) are used to create a walking motion.

## 7 Discussion

We have presented an approach for generating both full-body movement and facial expression from spatial-temporal constraints while matching the statistical properties of a database of captured motion. The system automatically learns a low-dimensional linear dynamic model from motion capture data and then enforces this as spatial-temporal priors to generate the motion. The statistical dynamic equations, together with an automatically derived objective function and user-defined constraints, comprise a trajectory optimization problem. Solving this optimization problem in the low-dimensional space yields optimal, natural motion that achieves the goals specified by the user.

The system achieves a degree of generality beyond the motion capture data. For example, we have generated a motion using constraints that cannot be satisfied directly by any motion in the database and found that the quality of the reconstructed motion was acceptable. Our video also demonstrates that the system can generate motion for characters whose skeletal models differ significantly from those in the database. However, we have not yet attempted to assess how far the user's constraints can stray from the motions in the database before the quality of the resulting animation declines to an unacceptable level.

This statistically based optimization approach complements a physically based optimization approach and offers a few potential advantages. First, using a low-dimensional statistical dynamic model for the constrained optimization might achieve faster convergence and be less subject to local minima. Second, our approach can generate slow and even stylized motions that have proven particularly difficult for physically based optimization. Third, the optimization does not require physical models. Building anatomically accurate physical models for facial animation or whole-body motion remains challenging. There are two limitations of our approach: an appropriate database must be available and the user cannot specify such dynamic constraints as ground reaction forces or character mass.

The main focus of this paper has been an exploration of the use of prior knowledge in motion capture data to generate natural motion that best satisfies user-defined constraints. Another important issue for building any interactive animation system is to design an intuitive interface to specify the desired motion. In our experiments, most of keyframe constraints were modified from example poses in the database. Foot contact constraints were specified by the user directly. Key trajectory constraints were extracted from a performance interface using two video cameras [Chai and Hodgins 2005]. Alternatively, the user could rely on commercial animation software such as Maya to specify constraints. This process is time-consuming even for a professional artist; it is more difficult for a naive user to specify such constraints. One of immediate directions for future work is, therefore, to design intuitive interfaces that allow the user to specify spatial-temporal constraints quickly and easily.

## Acknowledgements

The authors would like to thank Moshe Mahler for his help in modeling and rendering the images for this paper and Autodesk for the

donation of *Maya* software. Partial support for this research was provided by NSF IIS-0326322.

## References

ABE, Y., LIU, C. K., AND POPOVIĆ, Z. 2004. Momentum-based parameterization of dynamic character motion. In *Proceedings of the 2004 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 173–182.

ARIKAN, O., AND FORSYTH, D. A. 2002. Interactive motion generation from examples. In *ACM Transactions on Graphics*. 21(3):483–490.

BAZARAA, M. S., SHERALI, H. D., AND SHETTY, C. 1993. *Nonlinear Programming: Theory and Algorithms*. John Wiley and Sons Ltd. 2nd Edition.

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

BRAND, M., AND HERTZMANN, A. 2000. Style machines. In *Proceedings of ACM ACM SIGGRAPH 2000*. 183–192.

BRAND, M. E. 1999. Voice puppetry. In *Proceedings of ACM SIGGRAPH 1999*. 21–28.

BREGLER, C., COVELL, M., AND SLANEY, M. 1997. Video rewrite: Driving visual speech with audio. In *Proceedings of ACM SIGGRAPH 1997*. 353-360.

CHAI, J., AND HODGINS, J. 2005. Performance animation from low-dimensional control signals. In *ACM Transactions on Graphics*. 24(3):686–696.

CHAI, J., XIAO, J., AND HODGINS, J. 2003. Vision-based control of 3d facial animation. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 193-206.

FANG, A., AND POLLARD, N. S. 2003. Efficient synthesis of physically valid human motion. In *ACM Transactions on Graphics*. 22(3):417–426.

GALATA, A., JOHNSON, N., AND HOGG, D. 2001. Learning variable length markov models of behavior. In *Computer Vision and Image Understanding (CVIU) Journal*. 81(3):398-413.

GLEICHER, M. 1998. Retargeting motion to new characters. In *Proceedings of ACM SIGGRAPH 1998*. 33-42.

GROCHOW, K., MARTIN, S. L., HERTZMANN, A., AND POPOVIĆ, Z. 2004. Style-based inverse kinematics. In *ACM Transactions on Graphics*. 23(3):522–531.

GRZESZCZUK, R., TERZOPOULOS, D., AND HINTON, G. 1998. Neuroanimator: Fast neural network emulation and control of physics-based models. In *Proceedings of ACM SIGGRAPH 1998*. 9-20.

KOVAR, L., AND GLEICHER, M. 2004. Automated extraction and parameterization of motions in large data sets. In *ACM Transactions on Graphics*. 23(3):559–568.

KOVAR, L., GLEICHER, M., AND PIGHIN, F. 2002. Motion graphs. In *ACM Transactions on Graphics*. 21(3):473–482.

LEE, J., CHAI, J., REITSMA, P., HODGINS, J., AND POLLARD, N. 2002. Interactive control of avatars animated with human motion data. In *ACM Transactions on Graphics*. 21(3):491–500.

LI, Y., WANG, T., AND SHUM, H.-Y. 2002. Motion texture: A two-level statistical model for character synthesis. In *ACM Transactions on Graphics*. 21(3):465–472.

LIU, C. K., AND POPOVIĆ, Z. 2002. Synthesis of complex dynamic character motion from simple animations. In *ACM Transactions on Graphics*. 21(3):408–416.

LIU, K., HERTZMANN, A., AND POPOVIĆ, Z. 2005. Learning physics-based motion style with nonlinear inverse optimization. In *ACM Transactions on Graphics*. 23(3):1071–1081.

LJUNG, L. 1999. System identification: Theory for the user. Prentice Hall PTR. 2nd Edition.

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. In *ACM Transactions on Graphics*. 24(3):1062–1070.

PALM, W. J. 1999. Modeling, analysis, and control of dynamic systems. Wiley Publishers. 2nd Edition.

PAVLOVIĆ, V., REHG, J. M., AND MACCORMICK, J. 2000. Learning switching linear models of human motion. In *Advances in Neural Information Processing Systems 13*, 981–987.

POPOVIĆ, Z., AND WITKIN, A. P. 1999. Physically based motion transformation. In *Proceedings of ACM SIGGRAPH 1999*. 11-20.

ROSE, C., COHEN, M. F., AND BODENHEIMER, B. 1998. Verbs and adverbs: Multidimensional motion interpolation. In *IEEE Computer Graphics and Applications*. 18(5):32–40.

ROSE, C. F., SLOAN, P.-P. J., AND COHEN, M. F. 2001. Artist-directed inverse-kinematics using radial basis function interpolation. In *Computer Graphics Forum*. 20(3):239-250.

SAFONOVA, A., AND HODGINS, J. K. 2007. Construction and optimal search of interpolated motion graphs. In *ACM Transactions on Graphics*. 26(3).

SAFONOVA, A., HODGINS, J., AND POLLARD, N. 2004. Synthesizing physically realistic human motion in low-dimensional, behavior-specific spaces. In *ACM Transactions on Graphics*. 23(3):514–521.

SOATTO, S., DORETTO, G., AND WU, Y. N. 2001. Dynamic textures. In *Proceedings of International Conference on Computer Vision (ICCV'01)*. 2:439–446.

URTASUN, R., FLEET, D. J., AND FUA, P. 2006. Temporal motion models for monocular and multiview 3d human body tracking. In *Computer Vision and Image Understanding (CVIU)*. 104(2):157-177.

VICON SYSTEMS, 2004. http://www.vicon.com.

WITKIN, A., AND KASS, M. 1988. Spacetime constraints. In *Proceedings of ACM SIGGRAPH 1998*. 159–168.

ZHANG, L., SNAVELY, N., CURLESS, B., AND SEITZ, S. M. 2004. Spacetime faces: high resolution capture for modeling and animation. In *ACM Transactions on Graphics*. 23(3):548–558.