Realtime Human Motion Control with A Small Number of Inertial Sensors

Huajun Liu
Wuhan University
Texas A&M University

Xiaolin Wei
Jinxiang Chai
Texas A&M University

Inwoo Ha
Taehyun Rhee
Samsung Advanced Institute of Technology

Abstract

This paper introduces an approach to performance animation that employs a small number of motion sensors to create an easy-to-use system for an interactive control of a full-body human character. Our key idea is to construct a series of online local dynamic models from a prerecorded motion database and utilize them to construct full-body human motion in a maximum a posteriori framework (MAP). We have demonstrated the effectiveness of our system by controlling a variety of human actions, such as boxing, golf swinging, and table tennis, in real time. Given an appropriate motion capture database, the results are comparable in quality to those obtained from a commercial motion capture system with a full set of motion sensors (*e.g.*, XSens [2009]); however, our performance animation system is far less intrusive and expensive because it requires a small of motion sensors for full body control. We have also evaluated the performance of our system by leave-one-out-experiments and by comparing with two baseline algorithms.

Keywords: Performance animation, data-driven animation, natural user interfaces, motion capture, inertial sensors

1 Introduction

This paper introduces an approach to performance animation that employs a small number of motion sensors to create an easy-to-use system for an accurate control of a full-body character in real time (Figure 1). We choose inertial sensors for performance interfaces because they are compact, occlusion-free and relatively cheap. However, building such a performance interface is challenging because the input information from a small number of motion sensors is quite low-dimensional compared to a typical human body model, which is often represented by more than fifty degrees of freedom. The user's inputs, therefore, cannot be used to fully determine a full-body human pose because they will be consistent with many disparate solutions, some of which might correspond to unnatural poses and not be what the user intends to model.

The key idea of our paper is to construct an efficient dynamic motion model from prerecorded motion data and use it to constrain the solution space of online motion reconstruction. We represent our dynamic model as a set of mathematical functions that predict the current pose using its previous poses. Building an efficient dynamic model to predict how humans move, however, is difficult because human movement is highly nonlinear and demonstrates a wide variety of variations even for the same functional action. Instead of learning a global dynamic motion model, which would necessarily be high-dimensional and nonlinear, this paper chooses to construct a series of online local linear dynamic models to approximate a global highly nonlinear dynamic model.

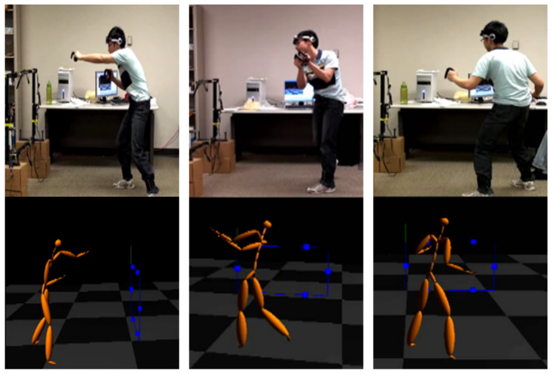


Figure 1: *The user wearing a small number of motion sensors controls an avatar in real time. From left to right: boxing, golf, and table tennis.*

At run time, we search the motion capture database for motion examples that are close to the recently synthesized motion. The K closest motion examples as well as their subsequent poses are then used as training data to learn a dynamic model that maps the previous poses to the current pose. Our dynamic model is time-varying because a new local dynamic model is created to predict a pose at each time step. Our motion model is appealing to human motion modeling and prediction because it scales up well to the size and heterogeneity of motion databases.

We formulate the online motion reconstruction problem in a maximum a posteriori framework (MAP) by combining a prior term encoded by online local dynamic motion models with a likelihood term defined by control inputs. Maximum a posteriori estimation of human poses in sequential mode produces a natural-looking motion sequence that best matches the control signals obtained from motion sensors.

We have demonstrated the effectiveness of our system by controlling a variety of human actions, such as boxing, golf swinging, and table tennis, in real time. Given an appropriate motion capture database, the results are comparable in quality to those obtained from a commercial motion capture system with a full set of motion sensors (*e.g.*, XSens [2009]); however, our performance animation system is far less intrusive and expensive because it only requires a small number of motion sensors for full body control. In addition, we have evaluated the performance of our system by leave-one-out experiments and by comparing with two baseline algorithms.

2 Background

In this section, we discuss related work in performance-based control interfaces for avatar control. Because we use a motion capture database in our system, we also briefly review research utilizing motion capture data for motion synthesis and control.

Performance-based control interfaces. One popular solution for performance animation is based on commercially available motion capture equipment. Active optical, passive optical, magnetic, and exoskeleton-based motion capture systems all now have the ability to perform real-time capture of a typical human model. However, this solution is far too expensive for common use. It is also cumbersome, requiring the user to wear 40–50 carefully positioned retro-reflective markers and skin-tight clothing, 18 magnetic motion sensors or inertial measurement units, or a full exoskeleton.

Systems that can extract meaningful information about the user's motion from only a few sensors are appealing because they dramatically reduce the intrusiveness and cost of the system. The Wiimote controllers by Nintendo, for example, are a successful commercial example of this class of interface. The system measures 3D acceleration of sensors with a three-axis accelerometer and detects 2D positions with the use of an infrared sensor; The motion information enables users to interact with virtual objects in game environments. In contrast, Sony's EyeToy [2003] is a vision-based system that requires no markers but is capable of extracting the 2D silhouettes of simple gestures such as a punch or a wave. Most recently, Sony's PlayStation Move uses the PlayStation Eye to track the position of a wand in three dimensions through a special illuminated orb at the end. None of these systems attempted to fully capture or animate the user's motion but instead focused on recognizing or locating a limited set of simple actions and showing their effect on the scene.

Researchers have also explored techniques for using a few sensors to reconstruct full-body motion. Badler and colleagues [1993] used four magnetic sensors and real-time inverse kinematics algorithms to control a standing figure in a virtual environment. Their system adopted a heuristic approach to handling the kinematic redundancy while we use a data-driven approach. Semwal and colleagues [Semwal et al. 1998] provided an analytic solution to the inverse kinematics algorithm based on eight magnetic sensors. Yin and Pai [2003] used a foot pressure sensor to develop an interface that extracts full-body motion from a database. Their system was successful at reproducing full body motion for a limited range of behaviors. However, foot pressure patterns may be insufficient to accurately reconstruct a motion with detailed upper body motions.

An alternative to performance interfaces is to act out the motion in front of vision sensors [Lee et al. 2002; Ren et al. 2004; Chai and Hodgins 2005; Ishigaki et al. 2009]. Most recently, with a single depth camera, Kinect by Microsoft enables users to control and interact with the Xbox 360 without the need to touch a game controller through a natural user interface using body gestures. Our system is different because our performance-based control interfaces are based on a small number of inertial motion sensors. Unlike vision sensors, inertial motion sensors do not suffer from occlusion problems.

Slyper and Hodgins [2008] creates a similar performance animation system that leverages the power of low-cost inertial sensors and prerecorded motion data. Their system enables the user to control upper-body movement with five inertial sensors. In contrast, our system allows full-body motion control with six inertial sensors. Similar to their system, we also use prior knowledge embedded in prerecorded motion data to reduce ambiguity of motion reconstruction. However, our work is different because we learn a series of online local dynamic models for motion interpolation rather than play the best match. Statistical interpolation of example motions is advantageous to our application because it can generalize motion examples to give the user more precise control over the character's motion.

Animation with motion capture data. A number of researchers have developed data-driven models for motion interpolation and synthesis. Three distinct approaches have been used: constructing models of human motion [Brand and Hertzmann 2000; Li et al. 2002; Grochow et al. 2004; Chai and Hodgins 2005; Chai and Hodgins 2007; Lau et al. 2009; Min et al. 2009], reordering motion clips employing a motion graph [Lee et al. 2002; Arikan and Forsyth 2002; Kovar et al. 2002; Lee et al. 2006; Safonova and Hodgins 2007] and interpolating motion to create new sequences [Rose et al. 1998; Kovar and Gleicher 2004; Mukai and Kuriyama 2005; Kwon and Shin 2005; Heck and Gleicher 2007]. In our work, we search the motion examples that are close to recently synthesized motion and use them to build a statistical dynamic model for motion reconstruction. We therefore discuss statistical motion modeling in more detail.

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. Thus far, statistical motion models constructed from prerecorded motion data have been used for interpolation of key frames [Li et al. 2002] or motion styles [Brand and Hertzmann 2000], inverse kinematics [Grochow et al. 2004], interactive control of human motion with performance interfaces [Chai and Hodgins 2005], synthesis of human animation that satisfies the user-defined constraints [Chai and Hodgins 2007], editing of human motion with direct manipulation interfaces and sketching interfaces [Min et al. 2009], perturbations of natural-looking human motion [Lau et al. 2009], and so forth.

Among all the statistical models aforementioned, our work is most similar to local subspace models used for online motion control [Chai and Hodgins 2005] because both are constructed at run time and are based on training examples relevant to current poses. However, there is an important distinction. We consider spatial-temporal correlation embedded in a prerecorded motion database and construct a statistical dynamic model which maps the previous synthesized poses to the current pose. Our experiment in Section 7 shows that our local dynamic models can produce more accurate results for performance animation than the previous online pose models [Chai and Hodgins 2005].

Our work is motivated by the success of statistical dynamic models used for space-time motion optimization [Chai and Hodgins 2007], which constructs a single global linear dynamic model to create an animation that satisfies sparse constraints specified by the user. We significantly extend the idea of statistical dynamic models by constructing a series of local linear dynamic models on fly to approximate highly nonlinear dynamic behaviors of human movement. The online local dynamic models avoid the problem of finding an appropriate structure for a global dynamic model, which would necessarily be high-dimensional and nonlinear.

3 Overview

Our system transforms control signals obtained from a small number of motion sensors into full-body motion by constructing a series of online local dynamic models from a prerecorded motion database and using those models to interpret probable values for the information about the user's motion not captured by motion sensors. The entire system consists of three major components:

Skeleton and sensor calibration. We first introduce a novel calibration process for the actor to ensure that the performance interface is robust to users of different skeletal sizes and to variations in sensor placement. Our calibration process simultaneously estimates the actor's skeletal lengths and local coordinates for each motion sensor using a small number of "calibration" poses.

Online modeling of dynamic behaviors. We present a new statistical dynamic model for online human motion reconstruction. Our system models nonlinear dynamic behaviors of human movement using a series of online local linear dynamic models learned from prerecorded motion data. The proposed local dynamic models pre-

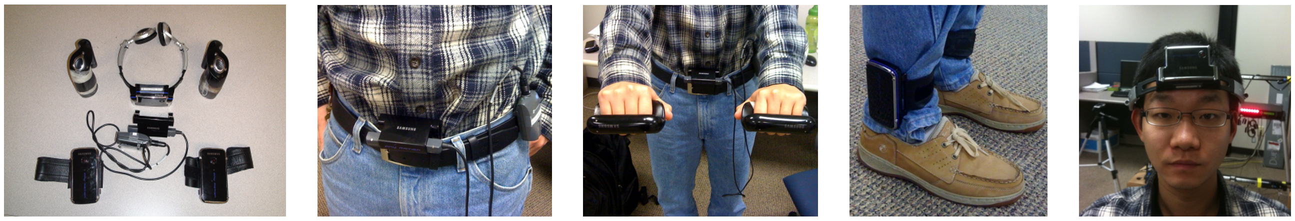


Figure 2: *A close-up view of motion sensors used for performance-based control interfaces.*

dict how humans move in local regions of the current poses and are used to constrain the reconstructed motion to lie in the space of natural-looking human motion.

Online motion synthesis. At run time, the user wears a small number of motion sensors to act out the motion. The performance interfaces record the global locations and orientations of each motion sensor, $[\mathbf{c}_1, ..., \mathbf{c}_t]$, in real time. The trajectories of the motion sensors specify the desired trajectories of certain points or vectors on the animated character. We formulate the online motion reconstruction in a MAP framework by combining a prior term encoded by online local dynamic motion models with a likelihood term defined by control inputs. The local dynamic models are used to reconstruct the user's pose $\mathbf{q}_t$ based on the current control signals $\mathbf{c}_t$ obtained from the performance-based interface as well as previous synthesized poses $[\tilde{\mathbf{q}}_{t-1}, ..., \tilde{\mathbf{q}}_{t-m}]$. Maximum a posteriori estimation of sequential poses produces a natural-looking motion that matches the control signals obtained from motion sensors. The system runs at interactive frame rates (about 35 fps).

The calibration process are done in offline mode, while both motion modeling and motion synthesis steps are performed online based on the live performance from the user. We describe each component in detail in the next three sections.

4 Skeleton and Sensor Calibration

Our system requires the user to wear a small number of motion sensors on the left hand, right hand, left ankle, right ankle, head and torso for realtime full-body motion control (see Figure2). The performance interfaces use a InterSense IS-900 system to record 3D position and orientation data of each motion sensor in real time (40 fps). IS-900 processor controls the entire IS-900 system and processes motion signals from tracking device to compute 6-DOF tracking data, including 3D orientation data integrated from gyroscopes, accelerometers and magnetometers and 3D position data obtained from ultrasonic sensors. Our calibration process ensures that the performance interfaces are robust to users of different skeletal sizes and to variations in sensor placement. In particular, the *skeleton calibration* process estimates the actor's skeletal lengths of each bone segment and the *sensor calibration* process calculates the local coordinates of each motion sensor, *i.e.*, the positions and orientations of each sensor with respect to their local reference frames.

We propose to use a small set of "calibration" poses to simultaneously estimate the skeletal lengths of the actor and local coordinates of each motion sensor. We choose eight "calibration" poses for skeleton and sensor calibrations. We instruct the actor to take the same poses as visualized on the screen and record the global locations and orientations of motion sensors under each calibration pose (Figure 3).

To reduce ambiguity for human skeleton modeling, we construct a principle subspace for skeletal lengths using a large set of prerecorded human skeleton data. Our human skeletal data is downloaded from the online CMU mocap library [1] and represented by Acclaim Skeleton File (ASF) format. Each skeleton example records length of individual bones. Our human skeletal model contains 24 bones, including head, thorax, upper neck, lower neck, upper back, lower back, and left and right clavicle, humerus, radius, wrist, hand, hip, femur, tibia, and metatarsal. Figure 4 shows the top five eigen modes of human skeleton variations.

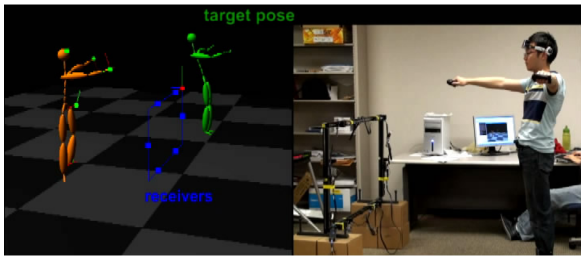


Figure 3: *Skeleton and sensor calibration with a small set of "calibration" poses. The user is instructed to perform the same pose as a target pose shown on the screen. The "green" character shows the target pose and the "orange" character shows the actor's pose.*

We represent the skeletal size of a human figure using a long vector $\mathbf{s}$, which stacks the lengths of each bone segment. We also represent the positions and orientations of the j-th motion sensor with respect to its local coordinate systems using the vectors $\mathbf{p}_j$ and $\mathbf{o}_j$, respectively. Let the vectors $\mathbf{q}_i, i = 1, ..., 8$, represent the i-th calibration pose, which is known in our calibration process. We formulate the calibration process as the following nonlinear optimization problem:

$$\begin{aligned}&\arg\min_{\mathbf{s},\{\mathbf{p}_j\},\{\mathbf{o}_j\},\{\lambda_b\}} \sum_i \sum_j \|\mathbf{f}(\mathbf{s},\mathbf{p}_j;\mathbf{q}_i) - \mathbf{t}_i^j\|^2 \\ &+\alpha_1\|\mathbf{f}(\mathbf{s},\mathbf{o}_j;\mathbf{q}_i) - \mathbf{r}_i^j\|^2 + \alpha_2\|\mathbf{s} - \mathbf{e}_0 - \textstyle\sum_{b=1}^{B} \lambda_b \mathbf{e}_b\|^2\end{aligned} \quad (1)$$

where the vector-valued function $\mathbf{f}$ is the forward kinematics function that computes from the joint angles of the calibration poses, $\mathbf{q}_i$, given the user's skeleton model, $\mathbf{s}$, and the locations or orientations of the j-th motion sensors, $\mathbf{p}_j$ or $\mathbf{o}_j$, relative to the inboard joint. The vectors $\mathbf{t}_i^j$ and $\mathbf{r}_i^j$ represent the recorded global positions and orientations of the j-th motion sensor under the i-calibration pose, respectively. The vectors $\mathbf{e}_0$ and $\mathbf{e}_b, b = 1, ..., B$ represent the mean and eigen modes of human skeletal models, respectively. The weights α_1 and α_2 balance the importance of each term.

We minimize the cost function using the Levenberg-Marquardt (LM) algorithm [Bazaraa et al. 1993]. The optimization simultaneously computes sensor locations and orientations $\tilde{\mathbf{p}}_j$ and $\tilde{\mathbf{o}}_j$ and skeletal lengths of the actor $\tilde{\mathbf{s}}$.

[1]http://mocap.cs.cmu.edu

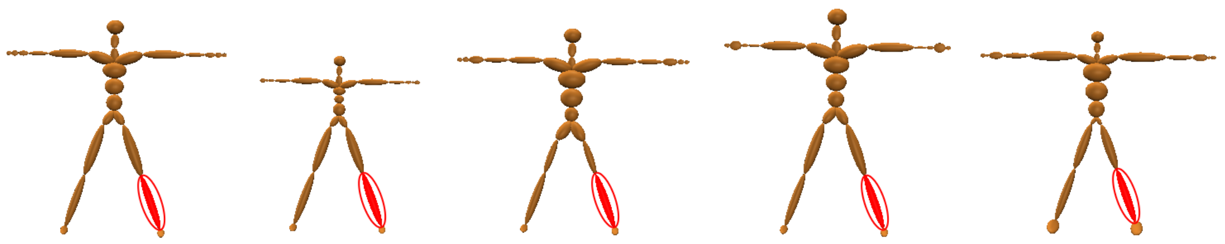


Figure 4: *The top five modes of human skeleton variations keeps about 99% of skeleton variations from a large set of prerecorded human skeletal models. Note that we normalize each bone segment using left tibia (red) for visualization.*

5 Online Motion Modeling

Reconstructing human motion from a small number of motion sensors is difficult because the control information from performance interfaces does not adequately constrain the joint angles of a full-body human model. The key idea of our approach is to automatically construct a series of online local dynamic models to resolve reconstruction ambiguity. We assume dynamic behaviors of human movement can be described as a m-order Markov chain. In other words, we assume the current pose $\mathbf{q}_t$ only depends on previous m poses: $Pr(\mathbf{q}_t|\mathbf{q}_{t-1},...,\mathbf{q}_1) = Pr(\mathbf{q}_t|\mathbf{q}_{t-1},...,\mathbf{q}_{t-m})$.

However, building a dynamic motion model for human motion prediction (*i.e.*, predicting the current pose from previous poses) is challenging because human movement is highly nonlinear and demonstrates a wide variety of variations even for the same functional action. To address this challenge, we propose to construct a series of online local dynamic models to approximate a global highly nonlinear dynamic model. The local dynamic models avoid the problem of finding an appropriate structure for a global dynamic model, which would necessarily be high-dimensional and nonlinear.

To predict the current pose $\mathbf{q}_t$ from previous synthesized motion, we first search the motion capture database for motion examples that are close to the recently synthesized poses $\tilde{Q} = [\tilde{\mathbf{q}}_{t-1},...,\tilde{\mathbf{q}}_{t-m}]$. The K closest examples $Q^k = [\mathbf{q}_{t_k-1},...,\mathbf{q}_{t_k-m}], k = 1,...,K$ as well as their subsequent poses $\mathbf{q}_{t_k}, k = 1,...,K$ are then used as training data to learn a prediction function that maps the previous m poses $\mathbf{q}_{t-1},...,\mathbf{q}_{t-m}$ to the current pose $\mathbf{q}_t$.

We assume a linear relationship of an input vector $\mathbf{x} = [\mathbf{q}_{t-1},...,\mathbf{q}_{t-m}]$ and an output vector $\mathbf{y} = \mathbf{q}_t$. Here, for simplicity, we consider only the case of one output variable, since we can learn prediction functions for each degree of freedom in output $\mathbf{q}_t$ separately. We also assume, without loss of generality, that the mean values of **x** and y are zeros. This can be achieved by subtracting the mean from input and output training data. Given this simplification, the model function in linear regression is

$$y = \beta^T\mathbf{x} + \epsilon_y \tag{2}$$

where the vector **x** is a $m \times D$-dimensional input vector, where D is the number of degrees of freedom of the configuration space for a human figure and y is the output value. The vector β are the regression coefficients and ϵ_y is a homoscedastic (independent of **x**) noise variable.

Furthermore, given the K closest examples $\{(\mathbf{x}_k, y_k)\}, k = 1,...,K$, the coefficients β can be estimated by minimizing the expected squared error E,

$$E = \sum_{k=1}^{K} \|y_k - \beta^T\mathbf{x}_k\|^2 \tag{3}$$

which results in the ordinary-least-squares solution

$$\beta = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} \tag{4}$$

The matrix **X** contains the input vectors $\mathbf{x}_k$ in its rows, and the vector **y** stacks the K output values $y_k, k = 1,...,K$.

In our implementation, we first apply principle component analysis to the input data $\mathbf{x}_k, k = 1,...K$ and then compute ordinary least squares regression in the principal subspace. Thus, we minimize $\|\mathbf{y} - \mathbf{X}\mathbf{U}\gamma\|^2$ with respect to the reduced coefficients γ. Therefore, the final regression coefficients are

$$\begin{aligned} \beta &= \mathbf{U}\gamma \\ &= \mathbf{U}(\mathbf{U}^T\mathbf{X}^T\mathbf{X}\mathbf{U})^{-1}\mathbf{U}^T\mathbf{X}^T\mathbf{y} \\ &= \mathbf{U}\Lambda^{-1}\mathbf{U}^T\mathbf{X}^T\mathbf{y} \end{aligned} \tag{5}$$

where the matrix Λ is a diagonal matrix containing the eigenvalues of the covariance matrix $\mathbf{C} = \mathbf{X}^T\mathbf{X}$.

We choose the weight for each data point $\mathbf{x}_k$ based on their relative distance from the recently synthesized motion $\tilde{Q} = [\tilde{\mathbf{q}}_{t-1},...,\tilde{\mathbf{q}}_{t-m}]$ using Gaussian functions:

$$w_k = \exp(-\frac{1}{2}(\mathbf{x}_k - \tilde{Q})^T D(\mathbf{x}_k - \tilde{Q})) \tag{6}$$

where the diagonal matrix D specifies the weights for each degree of freedom from the recently synthesized motion. For simplicity, we choose D as an identity matrix in our experiment.

Thus, the solution in Equation (5) is replaced by

$$\beta_w = \mathbf{U}_w\Lambda_w^{-1}\mathbf{U}_w^T\mathbf{X}^T\mathbf{W}\mathbf{y} \tag{7}$$

where the vector **W** is a diagonal matrix, containing the w_k's along its diagonal. The vectors $\mathbf{U}_w$ and Λ_w are extracted from the weighted covariance matrix $\mathbf{C}_w = \mathbf{X}^T\mathbf{W}\mathbf{X}$.

Furthermore, by assuming a Gaussian distributed noise variable ϵ_y, we can estimate its standard deviation σ from the residues $y_k - \beta_w\mathbf{x}_k, k = 1,...,K$. Because we learn a prediction function for each degree of freedom of the current pose, the local dynamic model to predict the d-th degree of freedom of the current pose can be described as

$$q_{t,d} = \beta_{w,d}^T\tilde{Q} + \mathcal{N}(0, \sigma_d) \tag{8}$$

where the scalar $q_{t,d}$ is the d-th degree of freedom of the current pose $\mathbf{q}_t$. The vectors $\beta_{w,d}$ are the regression coefficients for d-th degree of freedom of the current pose $\mathbf{q}_t$. The vector $\tilde{Q}$ represents the recently synthesized poses. The scalar σ_d is the standard deviation for the d-th prediction function.

Equation (8) can be thought as a time-varying dynamic model because it uses the recently synthesized poses to predict the current pose and all the model parameters, including $\beta_{w,d}$ and σ_d, are dependent on time. A new local dynamic model is created to predict each pose. The local dynamic model avoids the problem of finding an appropriate structure for a global dynamic model, which would necessarily be high-dimensional and nonlinear. Instead, we assume that a series of local linear dynamic models are sufficient to approximate the global highly nonlinear dynamic model.

selected for reconstruction.

6 Online Motion Reconstruction

This section focuses on how to reconstruct a sequence of joint angle poses, $[\tilde{\mathbf{q}}_1, ..., \tilde{\mathbf{q}}_t]$, from the input control signals, $[\mathbf{c}_1, ..., \mathbf{c}_t]$ obtained from a small number of motion sensors in sequential mode. We formulate the online motion reconstruction problem in a maximum a posteriori (MAP) framework by estimating the most likely pose $\mathbf{q}_t$ from the current control input $\mathbf{c}_t$ as well as previous synthesized poses $\tilde{\mathbf{q}}_{t-1}, ..., \tilde{\mathbf{q}}_{t-m}$:

$$\begin{array}{ll} & \arg\max_{\mathbf{q}_t} pr(\mathbf{q}_t|\mathbf{c}_t, \tilde{\mathbf{q}}_{t-1}, ..., \tilde{\mathbf{q}}_{t-m}) \\ \propto & \arg\max_{\mathbf{q}_t} pr(\mathbf{c}_t|\mathbf{q}_t) \cdot pr(\mathbf{q}_t|\tilde{\mathbf{q}}_{t-1}, ..., \tilde{\mathbf{q}}_{t-m}). \end{array} \quad (9)$$

In our implementation, we minimize the negative logarithm of the posteriori probability density function, yielding the following energy minimization problem:

$$\arg\min_{\mathbf{q}_t} \underbrace{-\ln pr(\mathbf{c}_t|\mathbf{q}_t)}_{E_{control}} + \underbrace{-\ln pr(\mathbf{q}_t|\tilde{\mathbf{q}}_{t-1}, ..., \tilde{\mathbf{q}}_{t-m})}_{E_{prior}}, \quad (10)$$

where the first term $E_{control}$ is the *control* term that measures how well the reconstructed pose $\mathbf{q}_t$ matches the current control inputs $\mathbf{c}_t$, and the second term is the *prior* term E_{prior} that constrains the reconstructed motion to stay close to the training examples. Optimal estimation of the current pose sequentially produces a natural-looking motion sequence that matches the control signals obtained from motion sensors. We discuss each term in more detail in the following sections.

6.1 Control Consistency

The control input $\mathbf{c}_t$ obtained from motion sensors might vary due to noise. Assuming gaussian noise with a standard deviation of σ_{data}, we can define the control term as follows:

$$\begin{array}{lll} E_{control} & = & -\ln pr(\mathbf{c}_t|\mathbf{q}_t) \\ & \propto & \frac{\|\mathbf{f}(\mathbf{q}_t;\tilde{\mathbf{s}},\tilde{\mathbf{z}})-\mathbf{c}_t\|^2}{2\sigma^2_{data}}, \end{array} \quad (11)$$

where the vector $\mathbf{q}_t$ represents the reconstructed pose at frame t, the vector $\tilde{\mathbf{s}}$ is the character's skeleton model, and the vector $\tilde{\mathbf{z}}$ represents the local coordinates of motion sensors (*i.e.*, $\tilde{\mathbf{p}}_j$ and $\tilde{\mathbf{o}}_j$). The vector $\mathbf{c}_t$ is the observed data from motion sensors. The vector-valued function $\mathbf{f}$ is the forward kinematics function which maps the current pose $\mathbf{q}_t$ to the global coordinates, given the estimated skeletal lengths and local coordinates from the calibration process.

The control signals from motion sensors are often very noisy and occasionally contains outliers. For example, the position data from ultrasonic sensors are often corrupted by outliers due to sensor noise. To deal with noise and outliers, we measure the distance between the predicted data $\mathbf{f}(\mathbf{q}_t; \tilde{\mathbf{s}}, \tilde{\mathbf{z}})$ and the observed data $\mathbf{c}_t$ with robust estimators. In our experiment, we choose the Lorentzian robust estimator to define the matching cost term:

$$\rho(e) = \log(1 + \frac{e^2}{2\sigma^2}) \quad (12)$$

where e is the residual distance between the predicted data and the observed data and the scalar σ is a parameter for the robust estimator.

6.2 Dynamic Motion Priors

The motion prior term measures the a-priori likelihood of the current pose using the knowledge embedded in the prerecorded motion database, given the recently synthesized motion. The prior term is used to constrain the reconstructed motion to satisfy the probabilistic distribution determined by the training examples in the local region of the recently synthesized motion.

In our system, we maximize the conditional probability of the current pose $\mathbf{q}_t$ given the recently synthesized poses $\tilde{\mathbf{q}}_{t-1}, ..., \tilde{\mathbf{q}}_{t-m}$:

$$Pr(\mathbf{q}_t|\tilde{\mathbf{q}}_{t-1}, ..., \tilde{\mathbf{q}}_{t-m}) \propto \prod_{d=1}^{D} \exp(-\frac{(q_{t,d} - \beta^T_{w,d}\tilde{Q})^2}{2\sigma^2_d}) \quad (13)$$

where $q_{t,d}, d = 1, ..., D$, is the d-th degree of freedom of the current pose $\mathbf{q}_t$. The vector $\beta_{w,d}$ and the scalar σ_d are the regression coefficients and standard deviation of the d-th prediction model. The long vector $\tilde{Q}$ sequentially stacks the recently synthesized poses $[\tilde{\mathbf{q}}_{t-1}, ..., \tilde{\mathbf{q}}_{t-m}]$.

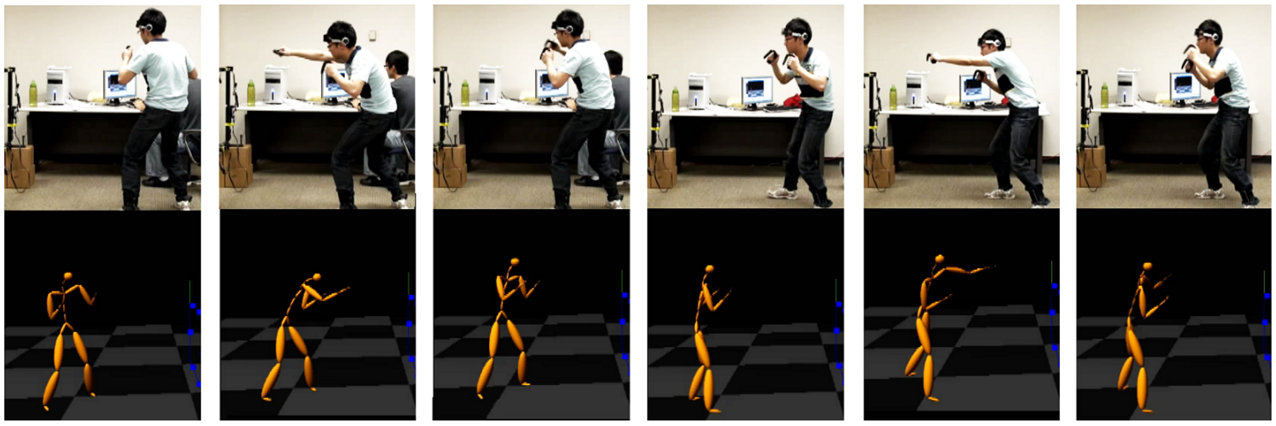
We minimize the negative log of $Pr(\mathbf{q}_t|\tilde{\mathbf{q}}_{t-1}, ..., \tilde{\mathbf{q}}_{t-m})$, yielding the energy formulation

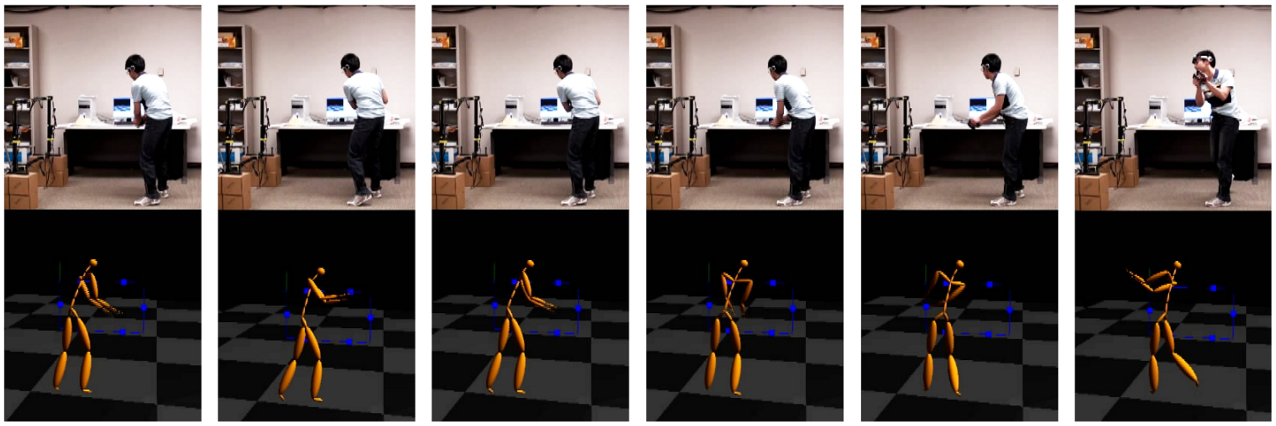
$$E_{prior} = \sum_d \frac{(q_{t,d} - \beta^T_{w,d}\tilde{Q})^2}{2\sigma^2_d} \quad (14)$$

6.3 Implementation Details

This section briefly describes implementation details of our system as well as details of our motion capture databases. We analytically evaluate the Jacobian terms of the objective function defined in Equation 10, initialize the optimization with the closest example in the database, and optimize the objective function using the Levenberg-Marquardt programming method [Lourakis 2009]. The solution converges rapidly because of a good starting point. The computational efficiency of our system highly depends on the size of prerecorded motion databases. Our system speeds up the K nearest neighbor search process with a strategy similar to a precomputed neighbor graph reported in [Chai and Hodgins 2005], which accelerates the runtime query by utilizing temporal coherence of query signals. Our system runs in real time with an average frame rate of 35 fps; and the latency of our current system is 0.025s.

Our motion capture database contains five full-body behaviors: walking (18160 frames), boxing (31358), jumping (6483), golf swing (2339), and table tennis (7582). The motions were captured with a Vicon motion capture system consisting of 12 MXF20 cameras with 41 markers, running at 120Hz. Note that we downsampled the original motion capture data to match the frame rate of motion sensors.


(a)


(b)

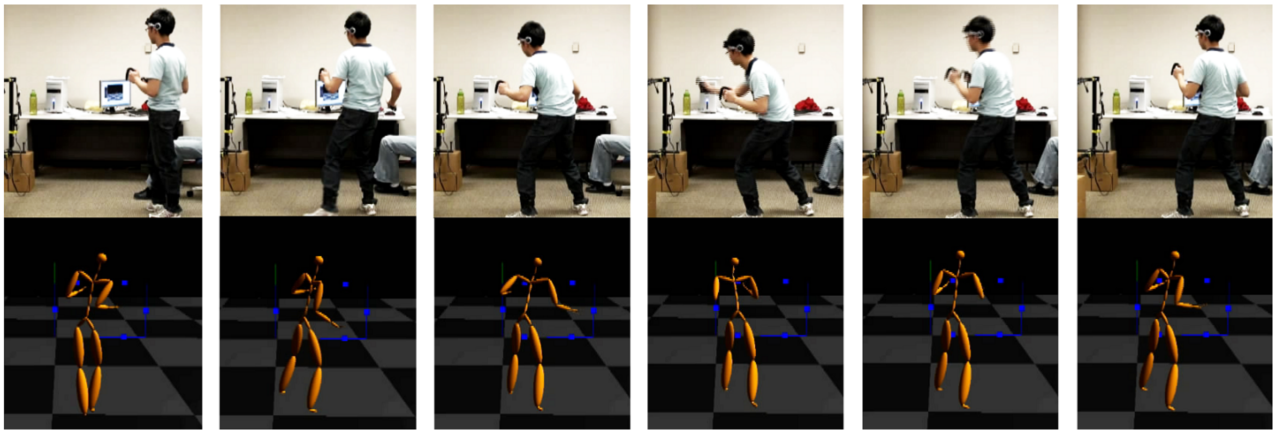

(c)

Figure 5: *Performance interfaces for realtime human motion control: (a) boxing; (b) golf swinging; (c) table tennis.*

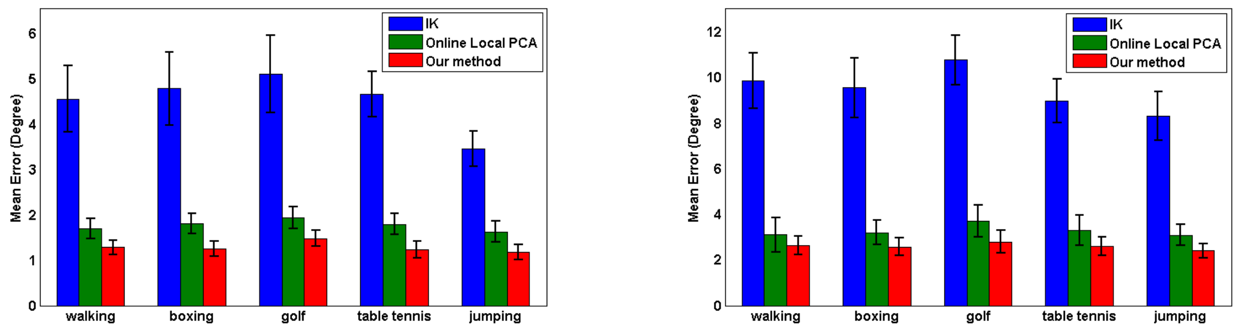


Figure 6: *Comparison with two baseline algorithm (IK and online local PCA [Chai and Hodgins 2005]) for five different human actions: (left) motion reconstruction with position and orientation data from six motion sensors; (right) motion reconstruction with position from six motion sensors.*

7 Results

We tested the effectiveness of our algorithm on different behaviors and evaluated the quality of the reconstructed motions by comparing with motion capture data recorded with a full marker set. Our results are best seen in the accompanying video although we show several frames of a few motions in Figure 5.

Testing on real data. We tested our system by controlling and animating a virtual character using real data captured by a small number of motion sensors. Figure 5 shows sample frames of the results for boxing, table tennis, and golf swing. The accompanying video demonstrates realtime motion control for a full-body avatar with six sensors (left ankle, right ankle, left hand, right hand, torso, and head). In addition, a side-by-side comparison between inverse kinematics techniques and our method is also shown in the video.

Leave-one-out evaluation. We evaluated the quality of the reconstructed motions by leaving out one sequence of motion capture data from each database as the testing sequence. We tested on different human actions and computed the average reconstruction errors measured by degrees per joint angle per frame. More specifically, we pulled testing motion sequences out of the training database and simulated the 3D positions and/or orientations for each motion sensor. We then employed our online motion reconstruction system to synthesize an animation. The reconstruction error was computed by the average squared distance between the reconstructed motion data and ground truth data. Figure 6 shows the means and standard deviations of the reconstruction errors for five different actions.

Comparisons with baseline algorithms. We compared the performance of our algorithm against two baseline algorithms, including inverse kinematics techniques and motion reconstruction with online local PCA models [Chai and Hodgins 2005], with leave-one-out evaluation. Figure 6 reported our evaluation results, including mean errors and standard deviations, for all three techniques. We reported the reconstruction errors for two types of control inputs: motion reconstruction using both 3D position and orientation data from six motion sensors and motion reconstruction using only 3D position data from six motion sensors. Figure 7 shows a detailed frame-by-frame comparison for one testing sequence. The evaluation results show that our method creates more accurate results than either inverse kinematics techniques or online local PCA models.

Different combinations of sensor data. The companying video compared the reconstructed motions using different combinations of sensor data obtained from six motion sensors, which include (1) position and orientation data from all six sensors; (2) position data from all six sensors plus orientation data from torso; (3) position data from all six sensors; (4) position data from torso and both ankles plus orientation data from torso, head, and both hands. For boxing, the reconstruction errors for each combination are 1.25 degrees, 2.25 degrees, 2.52 degrees, and 3.27 degrees, respectively. Such results are expected because the reconstruction errors usually increase as the number of constraints is reduced.

8 Conclusion

We have presented an approach for performance-based control interfaces using a small number of motion sensors. Our key idea is to construct a series of online local dynamic models from a prerecorded motion database and utilize them to construct full-body human motion in real time. We have demonstrated the effectiveness of our system by controlling a variety of human actions, including boxing, golf swinging, and table tennis, in real time. Given an appropriate database, the results are comparable in quality to those obtained from a commercial motion capture system with a full set of motion sensors (*e.g.*, XSens [2009]); however, our system is far less intrusive and expensive because it only requires a small number of sensors for full body control.

References

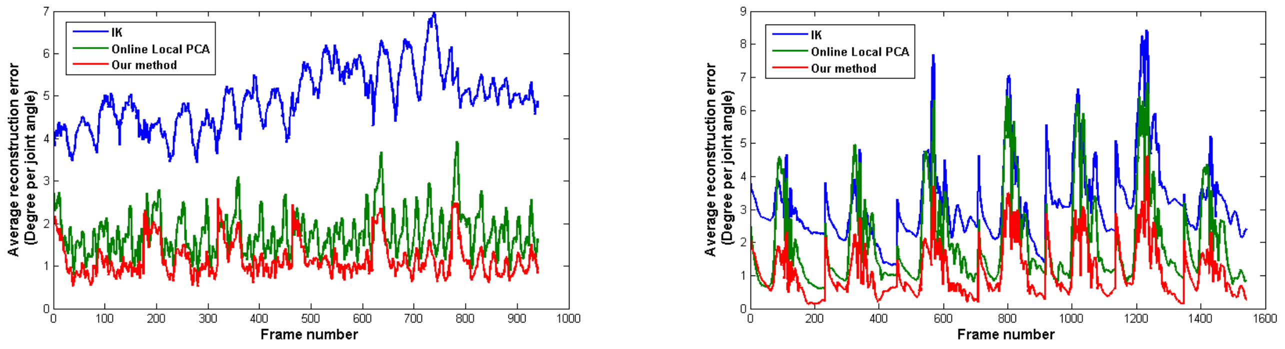
ARIKAN, O., AND FORSYTH, D. A. 2002. Interactive Motion Generation from Examples. In *ACM Transactions on Graphics*. 21(3):483–490.

BADLER, N. I., HOLLICK, M., AND GRANIERI, J. 1993. Real-time Control of A Virtual Human using Minimal Sensors. In *Presence*. 2(1):82–86.

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

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

CHAI, J., AND HODGINS, J. 2005. Performance Animation from Low-dimensional Control Signals. In *ACM Transactions on Graphics*. 24(3):686–696.



(a) walking (b) jumping

Figure 7: *Comparison of methods for synthesizing motions from a small number of motion sensors for one testing sequence. We compared our method with inverse kinematics techniques and motion reconstruction using online local PCA models in [Chai and Hodgins 2005].*

CHAI, J., AND HODGINS, J. 2007. Constraint-based Motion Optimization Using A Statistical Dynamic Model. In *ACM Transactions on Graphics*. 26(3):Article No.8.

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

HECK, R., AND GLEICHER, M. 2007. Parametric Motion Graphs. In *Proceedings of the 2007 symposium on Interactive 3D graphics and games*. 129-136.

ISHIGAKI, S., WHITE, T., ZORDAN, V. B., AND LIU, C. K. 2009. Performance-based control interface for character animation. 1–8. 28(3).

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.

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

LAU, M., CHAI, J., XU, Y.-Q., AND SHUM, H. 2009. Face Poser: Interactive Modeling of 3D Facial Expressions Using Facial Priors. In *ACM Transactions on Graphics*, Article No. 3. 29(1): article No. 3.

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.

LEE, K. H., CHOI, M. G., AND LEE, J. 2006. Motion patches:building blocks for virtual environments annotated with motion data. In *ACM Transactions on Graphics*. 25(3):898–906.

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.

LOURAKIS, M. I. A., 2009. levmar: Levenberg-Marquardt nonlinear least squares algorithms in {C}/{C}++.

MIN, J., CHEN, Y.-L., AND CHAI, J. 2009. Interactive Generation of Human Animation with Deformable Motion Models. *ACM Transactions on Graphics*. 29(1): article No. 9.

MUKAI, T., AND KURIYAMA, S. 2005. Geostatistical Motion Interpolation. In *ACM Transactions on Graphics*. 24(3):1062–1070.

REN, L., SHAKHNAROVICH, G., HODGINS, J. K., PFISTER, H., AND VIOLA, P. A. 2004. Learning Silhouette Features for Control of Human Motion. In *Computer Science Technical Reports 2004, Carnegie Mellon University*. CMU-CS-04-165.

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.

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

SEMWAL, S., HIGHTOWER, R., AND STANSFIELD, S. 1998. Mapping Algorithms for Real-time Control of An Avatar using Eight Sensors. In *Presence*. 7(1):1–21.

SLYPER, R., AND HODGINS, J. 2008. Action capture with accelerometers. In *2008 ACM SIGGRAPH / Eurographics Symposium on Computer Animation*.

SYSTEMS, S. E. T., 2003. http://www.eyetoy.com.

XSENS, 2009. http://www.xsens.com.

YIN, K., AND PAI, D. K. 2003. FootSee:An Interactive Animation System. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 329–338.