

# Example-based Human Motion Denoising

Hui Lou and Jinxiang Chai

**Abstract**—With the proliferation of motion capture data, interest in removing noise and outliers from motion capture data has increased. In this paper, we introduce an efficient motion denoising technique for simultaneous removal of noise and outliers in corrupted human motion data. The key idea of our approach is to construct a series of filter bases from prerecorded human motion data and use them along with robust statistics to filter corrupted human motion data. Mathematically, we formulate the motion denoising in an optimization framework. The objective function measures distance between the filtered motion and noisy input as well as how well the filtered motion matches spatial-temporal patterns embedded in natural human motion. Optimizing the cost function not only filters noise and outliers in corrupted motion but also preserves spatial-temporal patterns of natural human motion. We demonstrate the effectiveness of our system by experimenting with both real and simulated motion data, and by comparing with baseline methods and existing commercial softwares such as Vicon Blade. We also show the effectiveness of our algorithm on filling in missing values of motion capture data.

**Index Terms**—motion capture data, motion data processing, statistical data analysis, filtering, optimization, robust statistics



## 1 INTRODUCTION

ONE of the most popular approaches for creating realistic human animation is to use motion capture data. A recent notable example of motion capture data is the movie *Beowulf*, where prerecorded motion data was used to animate all characters in the film. Meanwhile, in the animation community, a number of researchers have explored how to edit, transform, interpolate, retarget and recompose the motion data for new applications.

All these exciting applications and developments start with accurate capture of human motion data. However, even with high-fidelity and expensive motion capture systems, captured motion data might still contain noise and outliers that must be removed before further processing. For example, motion data recorded by optical systems such as Vicon[33] often includes outliers and missing data due to marker occlusions and mislabeling. Motion data captured by inertial or magnetic systems is often corrupted by sensor noise, output drifting, and environment disturbances. Post-processing of noisy data often requires manual user editing which is not only time-consuming but also error-prone.

Human motion denoising is challenging because human motion is highly coordinated movement and the movements between different degrees of freedom are not independent. Standard signal denoising techniques such as Gaussian low-pass filter and Kalman filter[28] often process each degree of freedom independently. Therefore, the filtered motion cannot preserve spatial-temporal characteristics embedded in natural human motion. The problem becomes more complicated when motion capture data contains a certain percentage of outliers or missing values.

In this paper, we propose an efficient data-driven technique for human motion denoising that not only filters corrupted motion data but also keeps spatial-temporal patterns in human motion data (see Fig. 1). The key idea of our approach is to construct a series of spatial-temporal patterns from prerecorded high-quality motion data and use them to filter corrupted human motion data. Mathematically, we formulate the motion denoising problem in a nonlinear optimization framework. The objective function measures residual between the filtered motion and corrupted input as well as how well the filtered motion preserves spatial-temporal human motion patterns. Optimizing the objective function generates high-quality human motion data “closest” to the input data.

We have evaluated the performance of our algorithm in terms of both real and synthetic motion data. Our experiments show our algorithm works well for both joint-angle data (.amc file) and marker position data (.c3d files). The quality of the filtering motions produced by our system depends on the percentage of outliers and scale of noise level in the motion data. We, therefore, evaluate how increasing or decreasing the number of outliers or noise scales influences the final results. Finally, we show the superior performance of our algorithm by comparing with one of the most advanced commercial motion capture softwares (Vicon Balde) and three baseline motion denoising techniques, including Gaussian filter, general Kalman filter, and data-driven Kalman filter.

## 2 BACKGROUND

This section briefly reviews related work in human motion denoising. Our algorithm is data driven—the system automatically learns spatial-temporal patterns from human motion data and uses them along with robust statistics for filtering outliers and noise in corrupted

---

• H. Lou and J. Chai are with the Department of Computer Science and Engineering, Texas A&M University, College Station, TX, 77843-3112. E-mail: wslh@cse.tamu.edu; jchai@cse.tamu.edu

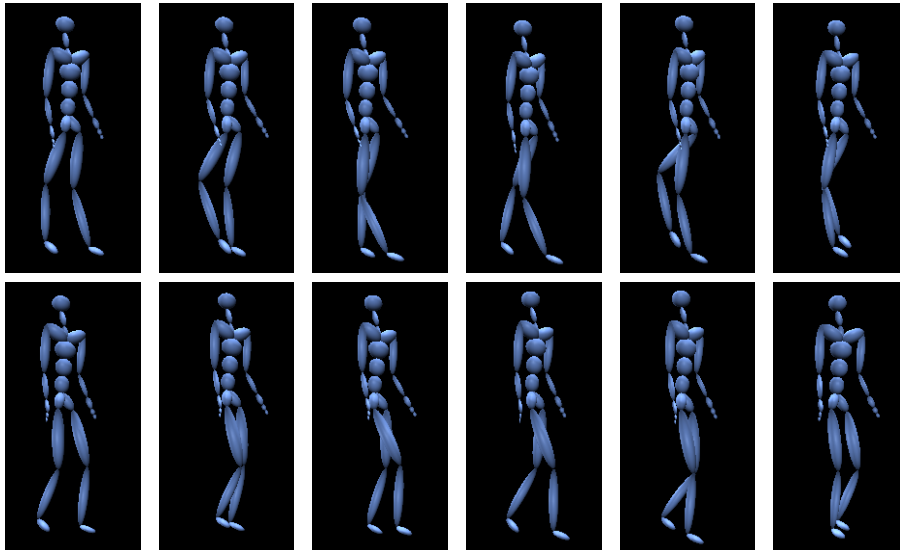


Fig. 1. Example-based human motion denoising: (top) corrupted motion data; (bottom) filtered motion data.

motion data. Consequently, we also discuss related work in spatial-temporal human motion modeling and robust statistics.

One popular approach for denoising human motion data is to apply standard low-pass filters such as Gaussian filter to noisy input data [7], [18], [32]. For example, Lee and Shin [18] presented a linear time-invariant filtering framework for filtering orientation data by transforming the orientation data into their analogues in a vector space, applying a filter mask on them, and then transforming the results back to the orientation space. Similar low-pass filters have also been implemented in commercial motion capture packages such as Vicon Blade [32].

An alternative solution is to use the Kalman filter framework [28] to sequentially filter noise present in human motion data [25], [27], [34]. In particular, Shin and his colleagues [25] applied the Kalman filter to transform the movements of a performer recorded by an online optical motion capture system to an animated character in real-time. Tak and Ko [27] adopted an unscented Kalman filter framework and used it to convert a sequence of animation motion into a physically plausible motion sequence.

Our motion denoising system transforms a sequence of corrupted motion data into high-quality human motion data. Unlike previous work, our denoising process is data-driven; the filter bases used for motion denoising are automatically constructed from prerecorded human motion data. One advantage of the data-driven denoising approach is to preserve spatial-temporal patterns embedded in natural human motion data. More importantly, it allows us to detect and remove outliers, and fill in missing values, a capability that has not been demonstrated by previous approaches.

A number of researchers have also explored filtering techniques for transforming human motion data into

physically correct motion [31], [24] or cartoon animation [29]. For example, Yamane and Nakamura [31] introduced a dynamics filter to convert a physically infeasible source motion sequence into a feasible one. Recently, Wang and his colleagues [29] presented the cartoon animation filter that takes an arbitrary input motion signal and modulates it in such a way that the output motion is more “alive” or “animated”. But none of these systems attempted to process human motion data corrupted with outliers, noise and missing values.

Our motion denoising algorithm models spatial-temporal correlations between degrees of freedom in human motion. Other researchers have also constructed spatial-temporal models from human motion data and used them in such applications as motion synthesis [20], [10], [3], [19], [21], [5], inverse kinematics [11], [4], motion compression [1], [17], motion quantification [23], motion registration [6], and footskate detection [14]. Our work is different because we construct a series of spatial-temporal filters with multi-channel singular spectrum analysis (M-SSA) [26], [9] and use them for a new application—human motion denoising. M-SSA has also been successfully applied to many practical problems in geophysics [9]. In this work, we extend the M-SSA to model high-dimensional human motion data and use it for reducing noise, removing outliers and filling in missing data.

To deal with outliers, we apply robust estimators [13], [12] to measure residual between the filtered motion and noisy input. The robust statistics has also been applied to deal with outliers in many problems of graphics and vision, such as 3D mesh smoothing [15], surface reconstruction [8], optical flow estimation [2], and image matching [30].

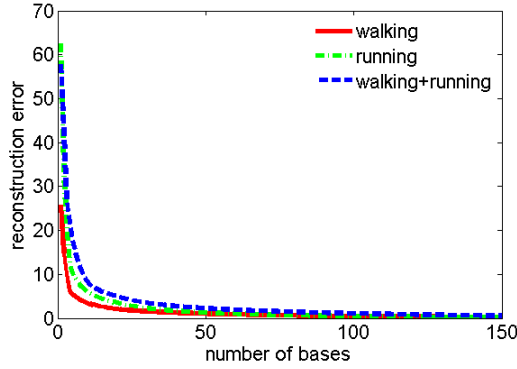


Fig. 2. Reconstruction errors vs. number of filter bases for human motion data with window size 20. The reconstruction error is evaluated via cross-validation techniques.

### 3 HUMAN MOTION DENOISING

The goal of this paper is to develop a human motion denoising algorithm that can reduce noise, remove outliers, and fill in missing data simultaneously. Our solution is to construct a series of data-driven filter bases from prerecorded human motion data and use them along with robust statistics for human motion denoising. This section is organized as follows: we explain how to construct filter bases from prerecorded motion data (Section 3.1), and discuss how to deal with outliers in corrupted human motion data (Section 3.2). We then formulate the denoising problem in an optimization framework and introduce an object function for human motion denoising (section 3.3). We develop an efficient, iterative algorithm for optimization of the object function in Section 3.4 and extend the framework for filling in missing values in Section 3.5.

#### 3.1 Construction of Filter Bases

The key idea of our approach is to construct a number of filter bases that resemble spatial-temporal patterns in natural human motion. We model a series of filter bases using multi-channel singular spectrum analysis (M-SSA) [26], [9]. Applying the M-SSA to human motion data allows us to identify a set of orthogonal spatial-temporal patterns embedded in natural human motion data. For simplicity, we focus our description on constructing M-SSA from one motion sequence.

Let  $\{x_l(t) : l = 1, \dots, L; t = 1, \dots, N\}$  be a sequence of prerecorded high-quality motion capture data, where  $N$  is the number of total frames and  $L$  is the total number of degrees of freedom for a character pose. Let  $\mathbf{x}_t = (x_1(t), \dots, x_L(t))^T$  denote a character pose at time  $t$ . Similarly, let  $\{y_l(t) : l = 1, \dots, L; t = 1, \dots, T\}$  be a sequence of corrupted motion data and let  $\mathbf{y}_t = (y_1(t), \dots, y_L(t))^T$  denote its character pose at frame  $t, t = 1, \dots, T$ .

We form a channel-specific trajectory matrix  $X_l$  by augmenting each channel  $\{x_l(t) : t = 1, \dots, N\}$  of the

data with  $S_N$  lagged copies of itself:

$$X_l = \begin{pmatrix} x_l(1) & x_l(2) & \dots & x_l(M) \\ x_l(2) & x_l(3) & \dots & x_l(M+1) \\ \vdots & \vdots & \dots & \vdots \\ x_l(S_N) & x_l(S_N+1) & \dots & x_l(N) \end{pmatrix}, \quad (1)$$

where  $M$  is the size of the lagged windows and  $S_N$  is the total number of the lagged windows:  $N - M + 1$ .

We then form a fully augmented trajectory matrix:

$$D = (X_1 \ X_2 \ \dots \ X_L). \quad (2)$$

To find the spatial-temporal correlation in degrees of freedom of human motion data, we can compute a grand lag covariance matrix  $C_D$  as follows:

$$C_D = \frac{D^T D}{S_N} = \begin{pmatrix} C_{1,1} & C_{1,2} & \dots & C_{1,L} \\ \vdots & C_{2,2} & \dots & \vdots \\ \vdots & \vdots & \dots & \vdots \\ C_{L,1} & C_{L,2} & \dots & C_{L,L} \end{pmatrix}, \quad (3)$$

where the blocks of  $C_D$  are given by

$$C_{l,l'} = \frac{X_l^T X_{l'}}{S_N}, \quad l, l' = 1, \dots, L, \quad (4)$$

with entries

$$(C_{l,l'})_{j,j'} = \frac{\sum_{n=1}^{S_N} x_l(n+j-1)x_{l'}(n+j'-1)}{S_N}, \quad j, j' = 1, \dots, M. \quad (5)$$

The covariance matrix  $C_{l,l'}$  computes the covariance between trajectories of the  $l$ -th dof and the  $l'$ -th dof within a window of size  $M$ . The grand lagged covariance matrix  $C_D$  is a symmetric  $ML$  by  $ML$  matrix, and it encodes spatial-temporal correlation of all degrees of freedom within a window of  $M$  frames. For example,  $(C_{3,4'})_{1,2'}$  encodes the correlation between the 3-th dof of the first frame and the 4'-th dof of the second frame within a window of  $M$  frames. M-SSA applies singular value decomposition to the grand lag covariance matrix and extracts most frequent spatial-temporal patterns in human motion data.

Diagonalizing the grand lagged covariance matrix yields  $ML$  eigen-vectors  $\{\mathbf{e}^k : k = 1, \dots, ML\}$ . The  $ML$  eigen-vectors provide a set of orthogonal filter bases, which can be used to reconstruct any segments of human motion data with a window of size  $M$ :

$$\mathbf{y}_{1:M} = \sum_{k=1}^{ML} \langle \mathbf{e}^k, \mathbf{y}_{1:M} \rangle \mathbf{e}^k, \quad (6)$$

where the vector  $\mathbf{y}_M$  stacks all data points in the  $M$ -frame long window and the operator  $\langle \cdot \rangle$  represents the dot product between two vectors. In practice, spatial-temporal redundancy of human movement allows us to reconstruct a segment of natural human motion data  $\mathbf{y}_{1:M}$  with a very small number of spatial-temporal patterns. Fig. 2 shows that for a particular action such as walking, 50 filter bases might be sufficient to model data variation within a window of size 20.

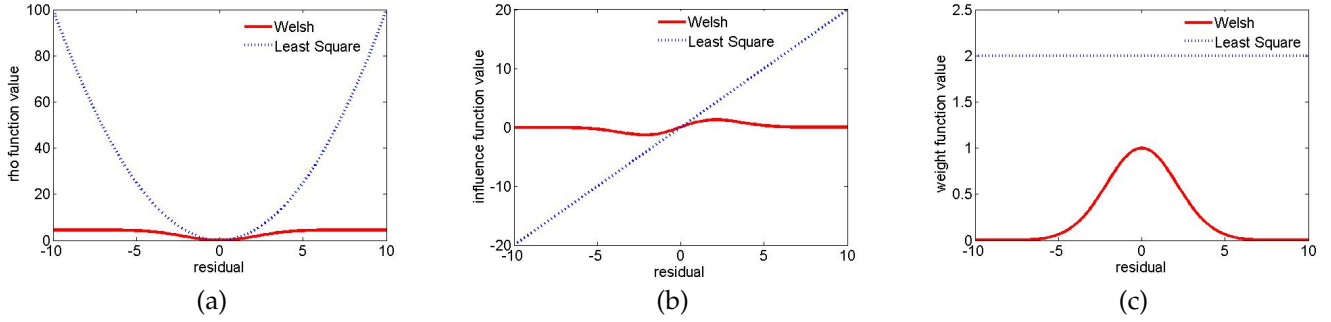


Fig. 3. The differences between the robust estimator (Welsch function) and least square: (a) the function for measuring the residual distance ( $r$ ) between the reconstructed motion and noisy measurement ( $\rho(r)$ ); (b) the influence function for each data point ( $\phi(r) = \frac{\partial \rho(r)}{\partial r}$ ); (c) the weights for each data point ( $\frac{\phi(r)}{r}$ ). Note that the weights for outliers become zero for robust estimators.

Conceptually, the orthogonal bases are similar to “sine” and “cosine” waves in Fourier analysis. We therefore can design a similar “low-pass” filter with constructed orthogonal bases  $\mathbf{e}^k$  and a user-defined cutoff threshold  $K$ . For example, we can pass frequent motion patterns but attenuate (reduce the amplitude of) improbable motion patterns. This motivates us to design the following “low-pass” filter:

$$\mathbf{z}_{1:M} = \sum_{k=1}^K \langle \mathbf{e}^k, \mathbf{y}_{1:M} \rangle \mathbf{e}^k, \quad (7)$$

where the vector  $\mathbf{z}_{1:M}$  is the filtered motion which stacks all data points in the  $M$ -frame long window, and the cutoff threshold  $K$  is usually chosen by keeping 99% of the original motion energy.

### 3.2 Dealing with Outliers

Real motion capture data often contains outliers due to marker occlusions and mislabeling. The “low-pass” filter described in Equation (7) will not work well for noisy data corrupted with outliers because least-square solutions (*i.e.* the dot product between the input data and eigen patterns) give too much influence to outliers, and a single degree of freedom with a large error (an outlier) will deteriorate the solution dramatically (see Fig. 3). We therefore introduce robust estimators to remove the influences of outliers.

Robust estimation measures the distance between the reconstructed data  $z_l(t)$  and noisy data  $y_l(t)$  with a robust estimator  $\rho$ :

$$E_{data}(\mathbf{z}_{1:T}, \mathbf{y}_{1:T}) = \sum_t \sum_l \rho(z_l(t) - y_l(t)), \quad (8)$$

where the robust estimator  $\rho$  is a function of residual between the noisy data and reconstructed data. The derivative of this function characterizes the bias that a particular measurement has on the solution. In the least-square case, the influence of data points increases linearly and is unbounded.

To increase robustness we only consider estimators for which the influences of outliers tend to zero (see Fig. 3). We choose the Welsch estimator but the treatment here could be equally applied to a wide variety of other estimators. A discussion of various estimators can be found in [13], [12].

Mathematically, the Welsch robust function is defined as follows:

$$\rho(r) = \frac{p^2}{2} \left(1 - \exp\left(-\frac{r^2}{p^2}\right)\right), \quad (9)$$

where the scalar  $p$  is a parameter for the robust estimator and  $r$  is the residual between the measured data and reconstructed data, which equals to  $z_l(t) - y_l(t)$  from equation (8).

### 3.3 Objective Function

We now combine the “low-pass” filter (Equation 7) with the robust estimator (Equation 8) for robust denoising of corrupted motion data. We formulate this as an optimization problem by reconstructing the original motion  $\mathbf{z}_{1:M} = \{z_l(t) : l = 1, \dots, L; t = 1, \dots, M\}$  as well as the reconstruction coefficients  $\mathbf{c}$  directly from the corrupted motion data  $\mathbf{y}_{1:M} = \{y_l(t) : l = 1, \dots, L; t = 1, \dots, M\}$ :

$$\hat{\mathbf{c}}, \hat{\mathbf{z}}_{1:M} = \arg \min_{\mathbf{c}, \mathbf{z}_{1:M}} \sum_{i=1}^M \sum_{l=1}^L \rho(z_l(i) - y_l(i)) + \lambda_1 \|\mathbf{z}_{1:M} - E\mathbf{c}\|^2, \quad (10)$$

where the matrix  $E = [\mathbf{e}^1 \dots \mathbf{e}^K]$  stacks the spatial-temporal patterns reconstructed from prerecorded human motion data. The first term is the robust estimation term defined in Equation (9), and it makes sure the reconstructed data stays as “close” as possible to the original data (while discarding outliers at the mean time). The second term is reformulated from the “low-pass” filter described in Equation (7), and it measures how well the reconstructed motion preserves the discovered spatial-temporal human motion patterns. The weight  $\lambda_1$  controls the importance of two terms.

To filter a motion sequence  $\mathbf{y}_{1:T}$  of the length  $T$ , we slide a window of the size  $M$  throughout the entire

sequence. We filter the entire input sequence in a batch mode. After summing over the optimization functions of all the sliding windows, we have

$$\{\hat{\mathbf{c}}_s, \hat{\mathbf{z}}_t\} = \arg \min_{\{\mathbf{c}_s\}, \{\mathbf{z}_t\}} \sum_{t=1}^T \sum_{l=1}^L \rho(z_l(t) - y_l(t)) + \lambda_1 \sum_{s=1}^S \|\mathbf{z}_{s,1:M} - E\mathbf{c}_s\|^2, \quad (11)$$

where the vectors  $\mathbf{c}_s$  and  $\mathbf{z}_{s,1:M}$  are the reconstruction coefficients and motion for the sliding window starting at frame  $s$  respectively. The parameter  $S = T - M + 1$  is the total number of sliding windows used for data filtering. The first term evaluates the distance between the measured motion data  $\mathbf{y}_{1:T}$ , and filtered motion data  $\mathbf{z}_{1:T}$  across the entire sequence. The second term makes sure that in the constructed motion  $\mathbf{z}_{1:T}$ , any  $M$ -frame motion segments in the filtered motion preserve spatial-temporal patterns embedded in human motion data.

### 3.4 Iterative Optimization

The total energy function (Equation 11) has two groups of unknowns: the reconstruction coefficients  $\mathbf{c}_s, s = 1, \dots, T - M + 1$  for each sliding window, and the filtered motion data  $\mathbf{z}_{1:T}$  across the entire sequence. The total number of the optimization parameters is  $K * (T - M + 1) + T * L$ .

Direct optimization of the energy function might not be desirable. In particular, when  $T$  is large, the system might run out of memory. The optimization might also become very slow and be prone to fall into local minima. To address these issues, we introduce an iterative optimization algorithm to decompose the above large optimization problem into a series of small optimization problems that can be solved efficiently.

We perform an iterative procedure to optimize the object function. In each iteration, we keep one group of the unknowns constant and search for the optimal update for the second group. More specifically, we initialize the filtered motion  $\hat{\mathbf{z}}_{1:T}$  with the noisy data  $\mathbf{y}_{1:T}$ . We then iteratively update the filtered motion and reconstruction coefficients as follows: (a). keep the filtered motion  $\hat{\mathbf{z}}_{1:T}$  constant and seek the optimal coefficients  $\mathbf{c}_s$ ; (b). keep the reconstruction coefficients  $\mathbf{c}_s$  constant and update the filtered motion  $\hat{\mathbf{z}}_{1:T}$ .

#### 3.4.1 Weight Update

In this step, we keep the filtered motion constant and seek the optimal coefficients  $\mathbf{c}_s$ . With the fixed motion  $\hat{\mathbf{z}}_{1:T}$ , we can estimate the optimal coefficients  $\mathbf{c}_s$  by decomposing the whole optimization function into  $S$  independent quadratic functions:

$$\hat{\mathbf{c}}_s = \arg \min_{\mathbf{c}_s} \|\hat{\mathbf{z}}_{s,1:M} - E\mathbf{c}_s\|^2, \quad s = 1, \dots, S. \quad (12)$$

This simple quadratic objective function has a closed-form solution:

$$\{\hat{\mathbf{c}}_s\} = E^T \hat{\mathbf{z}}_{s,1:M}, \quad s = 1, \dots, S. \quad (13)$$

#### 3.4.2 Motion Update

After we update  $\hat{\mathbf{c}}_s$ , we keep it temporarily constant and use it to update  $\hat{\mathbf{z}}_{1:T}$ . This allows us to decompose the whole optimization function into the  $T$  smaller independent optimization functions:

$$\hat{\mathbf{z}}_t = \arg \min_{\mathbf{z}_t} \sum_l \rho(z_l(t) - y_l(t)) + \lambda_1 \sum_{m=\max\{1, t-T+M\}}^{m=\min\{M, t\}} \|\mathbf{z}_t - E_m \hat{\mathbf{c}}_{t-m+1}^k\|^2, \quad t = 1, \dots, T, \quad (14)$$

where the matrix  $E_m$  is a  $L \times K$  matrix that stacks rows  $M(l-1) + m, l = 1, \dots, L$  of the matrix  $E$ . The min and max functions are used to deal with the first and the last  $M-1$  frames respectively.

The above optimization can be formulated as an iteratively re-weighted least square (IRLS) problem [12], [22]. After applying IRLS to the objective function in Equation (14), we can iteratively solve the following weighted least-square problem:

$$\hat{\mathbf{z}}_t = \arg \min_{\mathbf{z}_t} \sum_l w_l(t) (z_l(t) - y_l(t))^2 + \lambda_1 \sum_{m=\max\{1, t-T+M\}}^{m=\min\{M, t\}} \|\mathbf{z}_t - E_m \hat{\mathbf{c}}_{t-m+1}^k\|^2, \quad t = 1, \dots, T, \quad (15)$$

where  $w_l(t) = \frac{\varphi(z_l(t) - y_l(t))}{z_l(t) - y_l(t)}$  and  $\varphi(\cdot)$  is the derivative of the Welsch function  $\rho(\cdot)$ .

#### 3.4.3 Iterative Optimization Procedure

The final motion denoising algorithm iteratively updates the motion across the entire sequence. The iterative procedure is outlined as follows:

1. Initialize  $\hat{\mathbf{z}}_{1:T}^0 = \mathbf{y}_{1:T}$
2. Update the coefficients  $\hat{\mathbf{c}}_s$  and motion  $\hat{\mathbf{z}}_{1:T}$ 
  - 2.1. Update  $\hat{\mathbf{c}}_s = E^T \hat{\mathbf{z}}_{s,1:M} \quad s = 1, \dots, S$
  - 2.2. Update motion  $\hat{\mathbf{z}}_{1:T}$  with IRLS techniques.
3. Repeat step 2 until the change of energy function value is smaller than a user-defined threshold.

The algorithm converges fast and it typically runs about 10 to 20 iterations in our experiments.

#### 3.4.4 Post Processing

Given an appropriate set of training data, the data-driven denoising algorithm transforms corrupted motion data into high-quality motion. However, the filtered motion may violate kinematic constraints imposed by the environment, particularly when the input motion is corrupted by large scale noise or high percentages of outliers. The most visible artifact is footskate. The system uses the method in [16] to correct the footskate artifact.

### 3.5 Missing Data Fill-in

Motion capture data from optical motion capture systems often contains missing values due to marker occlusions. Our framework can easily be extended for filling in missing values in motion capture data. We assume that the index to missing data points is known in advance (this is often the case for optical motion capture systems). We use binary weights  $\alpha_{l,t} \in \{0, 1\}, l =$

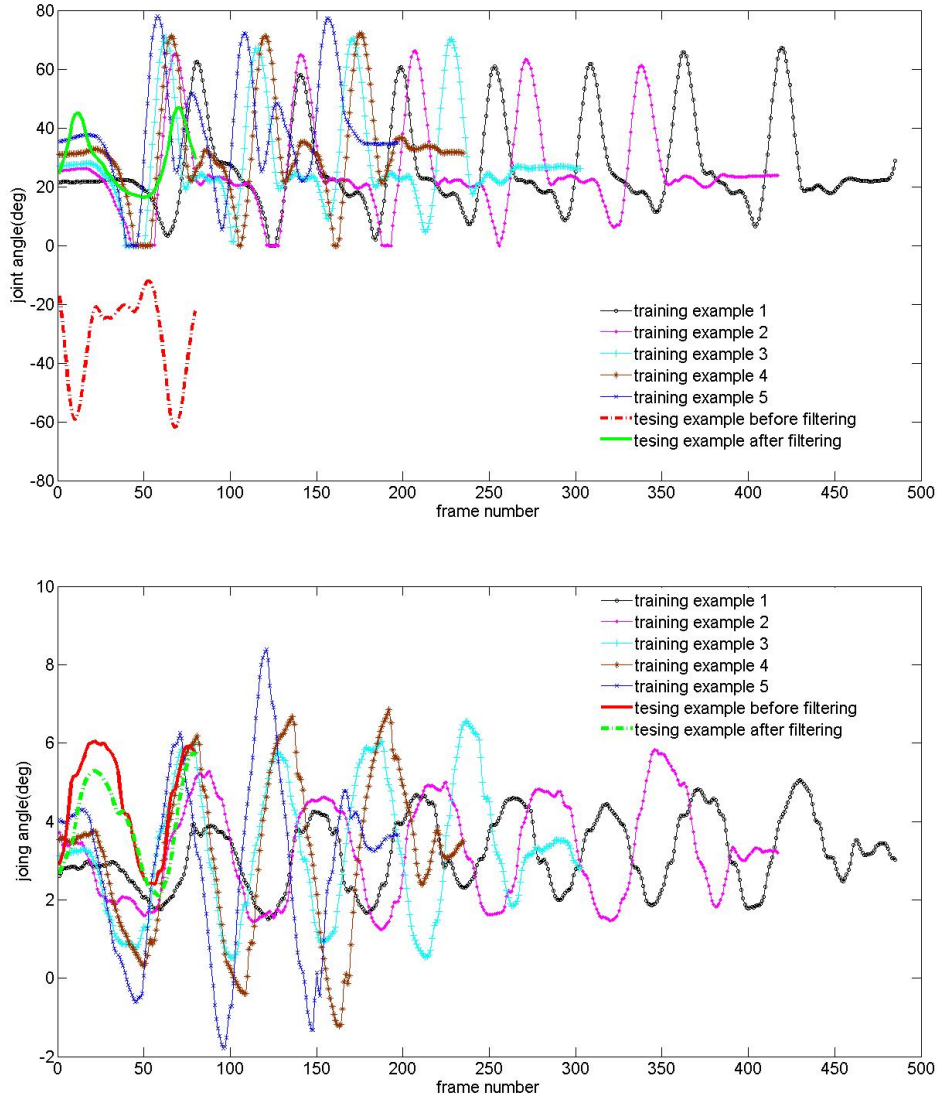


Fig. 4. Motion generalization: (top) training data set, filtered motion, and input noisy motion for knee joint; (bottom) training data set, filtered motion, and input noisy motion for thorax joint. Note that knee joint is corrupted by outliers across the entire motion while the thorax joint is not corrupted by any outliers.

$1, \dots, L, t = 1, \dots, T$  to indicate whether or not the observed data point  $y_l(t)$  contains missing data.

To reduce noise and remove outliers as well as fill in missing data, we can optimize the following objective function:

$$\{\hat{\mathbf{c}}_s, \hat{\mathbf{z}}_t\} = \arg \min_{\{\mathbf{c}_s\}, \{\mathbf{z}_t\}} \sum_{t=1}^T \sum_{l=1}^L \alpha_{l,t} \rho(z_l(t) - y_l(t)) + \lambda_1 \sum_{s=1}^S \|\mathbf{z}_{s,1:M} - E\mathbf{c}_s\|^2. \quad (16)$$

Similarly, we use the iterative procedure described in Section 3.4 to efficiently optimize the above objective function.

## 4 RESULTS

We have evaluated the performance of our algorithm on both real and simulated data. Our algorithm works

well for both joint angle data (.amc) and marker position data (.c3d). We also compare with three baseline algorithms and a commercial motion capture software (Vicon Blade). Our results are best viewed in the accompanying video although we show sample frames of a few results here.

In our experiments, all the data was originally captured at 120 fps and then downsampled to 30 fps. We set the window size ( $M$ ) to 20 frames. We automatically determine the number of bases ( $K$ ) by keeping the energy to be 99%, that is, the energy of the first  $K$  largest eigen values is around 99% of the total energy. We also experimentally set the parameter of Welsch estimator ( $p$ ) and weight ( $\lambda_1$ ) to 3 and 0.1 respectively. Our training database is behavior-specific and typically contains a small number of motion examples with different style

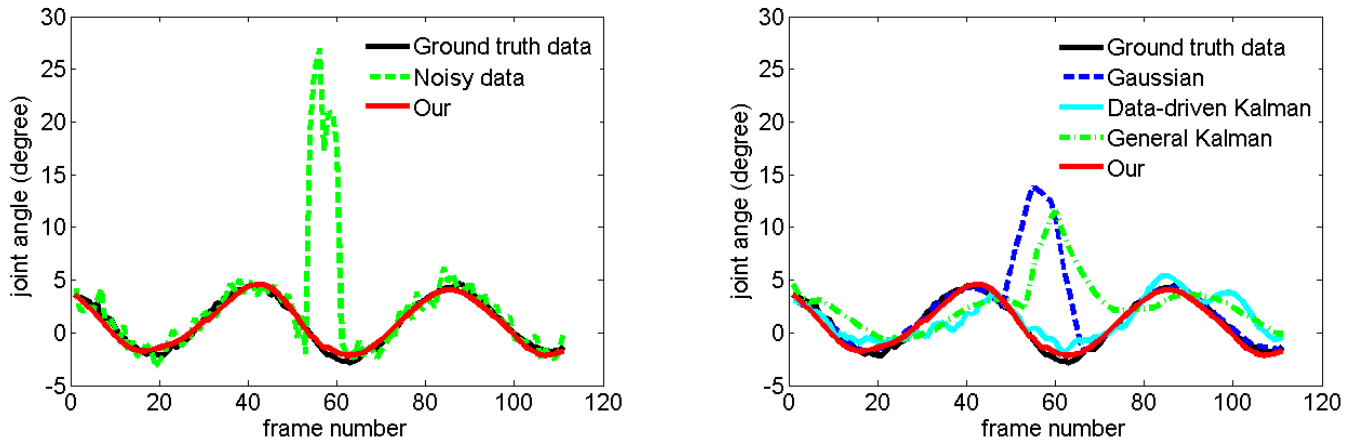


Fig. 5. Comparisons of our method with ground truth data and other signal denoising techniques for one joint angle of thorax: (left) our results vs. ground truth data; (right) our result vs. results obtained by other methods.

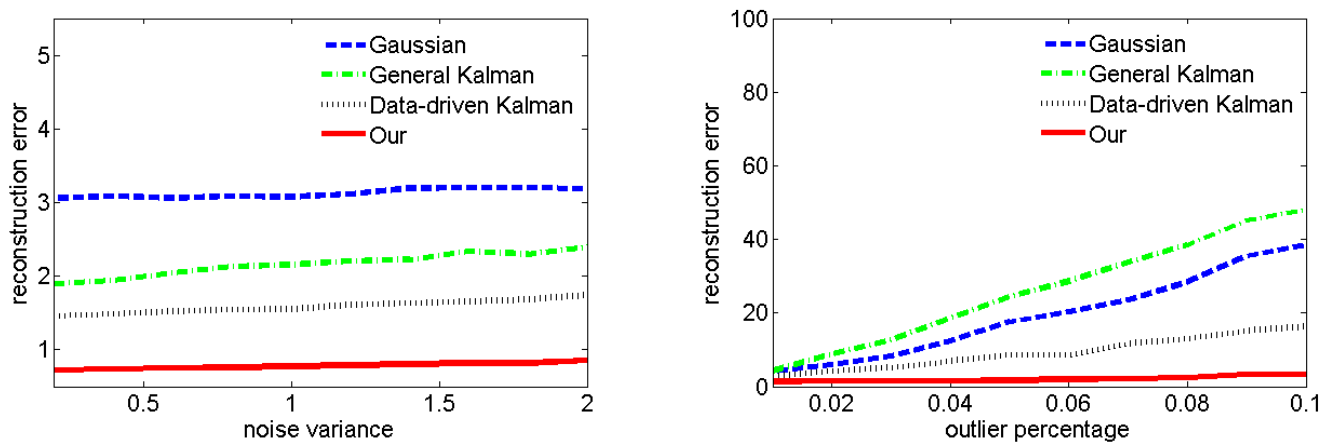


Fig. 6. Comparisons of our method with other signal denoising techniques: (left) reconstruction errors vs. different noise levels; (right) reconstruction errors vs. different percentages of outliers.

variations. For example, the training data for walking contains five walking examples with different speeds and step sizes. The accompanying video and Fig. 4 show the training examples and filtered motion as well as the corrupted motion for results shown in Fig. 1.

#### 4.1 Testing on Real Data

We have evaluated the performance of our algorithm on real motion data, which is recorded by optical motion capture systems (Vicon). Due to occlusions and marker mislabeling, real mocap data from the Vicon system often contains a certain percentage of missing values and outliers. We tested our algorithm for denoising corrupted motion in both original marker position space (.c3d files) and joint-angle space (.amc). Fig. 1 and Fig. 7 show the results in joint angle space and marker position space respectively. Most corrupted data reported here are from the CMU online mocap library. For example, we constructed filter bases from two online motion

files “83-04.c3d” and “83-55.c3d” and then use them to denoise one corrupted motion sequence “83-68.c3d”. All three files are from the same subject (83). Our experiments show that our algorithm can filter motions whose variations are not in the database. For example, Fig. 4 visualizes joint angle data (knee and thorax) of the training examples, the filtered motion and input motion. Obviously the reconstructed motion has a different phase and scale from the training data set.

We also tested our algorithm on filling in missing data in both joint angle and marker position space. Fig. 8 shows sample frames of our results, where all the markers on both arms are completely missing. The accompanying video shows comparison with one of the most advanced motion capture softwares *Vicon Blade*. Our method produces much better results than *Vicon Blade*. For example when markers on both arms are missing throughout the motion, *Vicon Blade* fails to handle this case. In contrast, our algorithm successfully fills in missing data on both arms across the entire

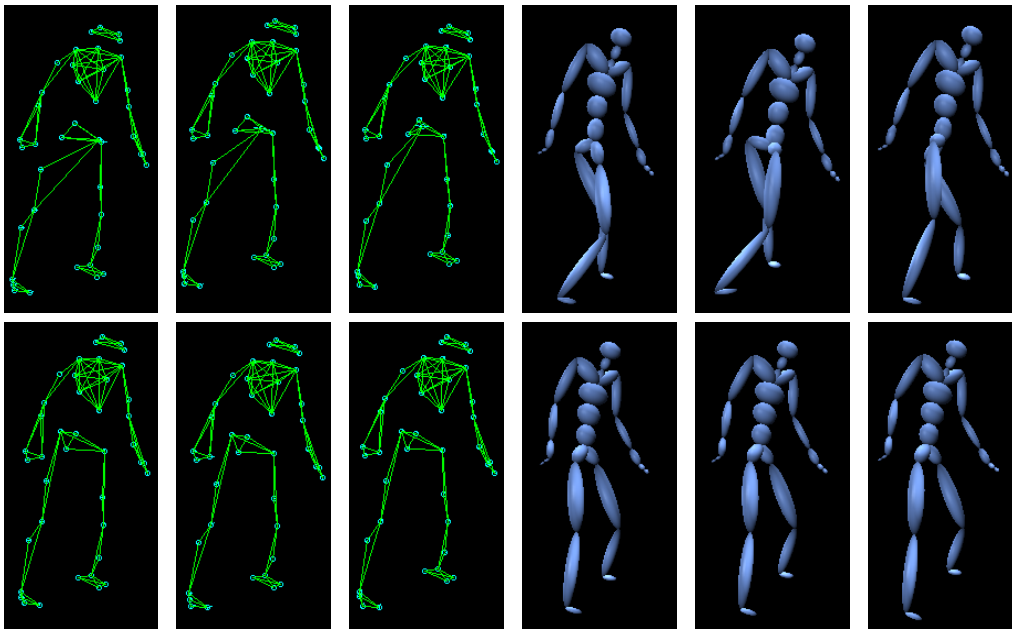


Fig. 7. Filtering real noisy motion data in the 3D position space: (top left) the corrupted motion data in the position space; (bottom left) the filtered motion data in the position space; (top right) the corrupted motion data in the joint-angle space; (bottom right) the filtered motion data in the joint-angle space.

sequence (see Fig. 8).

#### 4.2 Evaluation on Simulated Data

We have evaluated the performance of our algorithm on a variety of simulated noisy motion data, which is corrupted with different percentages of outliers and different levels of noise. The accompanying video demonstrates the effectiveness of our algorithm for filtering a number of individual behaviors, including walking, running and jumping.

We also compare with three baseline human motion denoising techniques: Gaussian filter, the simple general Kalman filter [28], and the data-driven Kalman filter. We applied a standard Gaussian filter to every degree of freedom independently. We experimentally set a window size to 11. We also implemented two types of Kalman filter. For the first one, we set its system matrices to identity matrices by simply choosing prediction model as follows:  $z_t(t+1) = z_t(t) + N(0, \sigma)$ . The second Kalman filter is a simple data-driven algorithm, which learns the system matrices from the same set of training data as used in our algorithm.

We tested a number of trials on each setting and computed the average reconstruction errors measured by degrees per DoF (Degree of Freedom) per frame. The errors were evaluated via cross validation techniques and they were measured by the squared distance between ground truth data and filtered data. Fig. 6 shows the comparison of the performances of four algorithms under various outlier percentages and noise levels for walking motion. Our algorithm produces the best results for all testing scenarios. Fig. 5 (right) shows both Gaussian filter and

general Kalman filters fail to detect and remove outliers in the corrupted motion data. Data-driven Kalman filter cannot preserve spatial-temporal patterns in the input motion. Only our algorithm can remove outliers while keeping spatial-temporal patterns in the input motion.

## 5 DISCUSSION

We have presented an efficient data-driven denoising algorithm for human motion data denoising, which simultaneously reduces noise, removes outliers, and fills in missing values. More importantly, the denoised motion data can keep spatial-temporal patterns embedded in natural human motion. The key idea of our approach is to construct a series of filter bases from high-quality motion capture data and then employ them along with robust statistics for human motion data denoising.

The constructed filter bases are in spirit similar to other filter bases used in signal processing community, such as “sine” waves, “cosine” waves, and “wavelet” bases. They are normalized and orthogonal against each other; a complete set of the filter bases can be used to uniquely reconstruct a segment of human motion data with a specific window size. One unique property of our filtering process is that it keeps important human motion patterns while throwing away impossible patterns, which cannot be interpreted by training data. The data-driven denoising filter, therefore, could still keep high-frequency components of the input motion because frequent patterns might still contain high-frequency components. Another benefit of data-driven filter bases is to allow the system to detect outliers and fill in missing

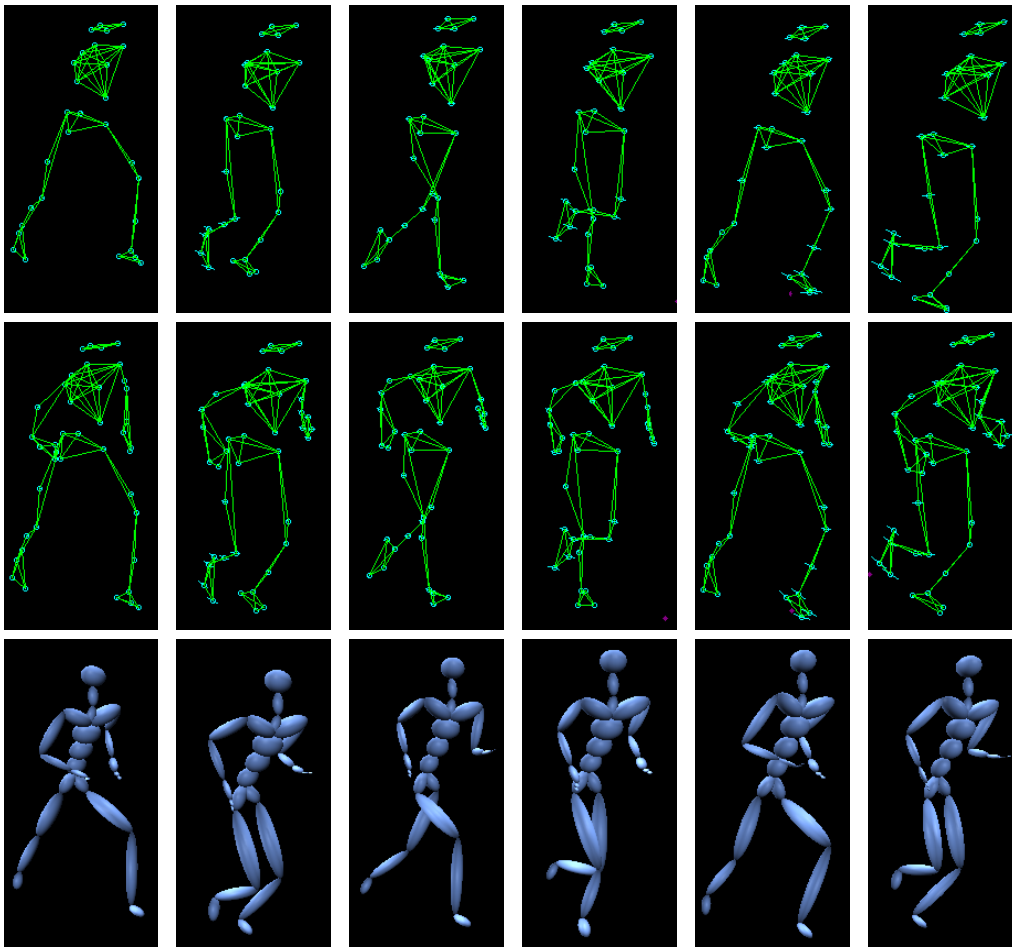


Fig. 8. Filling in missing data in the 3D position space: (top) missing motion data in the 3D position space; (middle) completed motion data in the 3D position space; (bottom) completed motion data in the joint-angle space.

values, a capability that has not been demonstrated by any previous human motion denoising approaches.

Just like any other data-driven approaches, one limitation of our approach is that an appropriate database must be available. We require the training data be “clean” (perceptually high-quality) and contain similar motion patterns of the input motion. Fig. 4 shows that our algorithm can generate motion variations that are not in the database. However, filtering the noisy input with different motion patterns (denoising walking data with running patterns, for instance) is not likely to yield reasonable results.

Our system has generated good results on filtering motion data with different levels of noise and different percentages of outliers. We have observed that filtering results deteriorate rapidly when the percentage of outliers is larger than 15%. But we have not rigorously assessed when our system will break down.

Just like other filters in the signal processing community, the performance of our algorithm depends on the size of the filtering window. A large window might better capture spatial-temporal characteristics embedded in human motion data but the filtering process might

become slower. We now experimentally set the window size ( $M$ ) to 20. Though, we believe an optimal window size should be behavior-specific. An immediate future work is thus to study how changing the window size influences the performance of our filtering algorithm for particular actions.

In future, we would like to explore how to learn spatial-temporal filter bases from noisy data itself. It might be possible to reconstruct the spatial-temporal filter bases  $\mathbf{e}_k, k = 1, \dots, K$ , filtered motion  $\hat{\mathbf{z}}_{1:T}$ , and reconstruction coefficients  $\hat{\mathbf{c}}_s, s = 1, \dots, S$  from the noisy input data  $\hat{\mathbf{y}}_{1:T}$  simultaneously using the optimization framework described in this paper. We would also like to extend our framework to transform a small set of noisy 3D trajectories constraints into high-quality joint angle data. We also plan to add joint angle limit constraints into the motion denoising framework in our future work.

## REFERENCES

- [1] O. Arıkan, “Compression of motion capture databases,” *Proc. ACM SIGGRAPH '06*, vol. 25, no. 3, pp. 890-897, 2006.
- [2] M. J. Black and P. Anandan, “The robust estimation of multiple motions: parametric and piecewise-smooth flow fields,” *Computer Vision and Image Understanding '96*, vol. 63, no. 1, pp. 75-104, 1996.

- [3] M. Brand and A. Hertzmann, "Style machines," *Proc. ACM SIGGRAPH '00* pp. 183-192, 2000.
- [4] J. Chai, J. Hodgins, "Performance animation from low-dimensional control signals," *ACM Transactions on Graphics '05*, vol. 24, no. 3, pp. 686-696, 2005.
- [5] J. Chai, J. Hodgins, "Constraint-based motion optimization using a statistical dynamic model," *ACM Transactions on Graphics '07* vol. 26, no. 3, article no. 8, 2007.
- [6] Y.-L. Chen, J. Min, J. Chai, "Flexible registration of human motion data with parameterized motion models," *ACM Symposium on Interactive 3D Graphics and Games (i3D 2009)*, 2009
- [7] Y. Fang, C. C. Hsieh, M. J. Kim, J. J. Chang and T. C. Woo, "Real time motion fairing with unit quaternions," *Computer-Aided Design '98*, vol. 30, no. 3, pp. 191-198, 1998
- [8] S. Fleishman, D. Cohen-or and C. T. Silva, "Robust moving least-squares fitting with sharp features," *ACM Transactions on Graphics '05*, vol. 24, no. 3, pp. 554-552, 2005.
- [9] M. Ghil, M. R. Allen, M. D. Dettinger, K. Ide, D. Kondrashov, M. E. Mann, A. W. Robertson, A. Saunders, Y. Tian, F. Varadi and P. Piou, "Advanced spectral methods for climatic time series," *Reviews of Geophysics '02*, vol. 40, no. 1, 2002.
- [10] A. Galata, N. Johnson and D. Hogg, "Learning variable length markov models of behavior," *Computer Vision and Image Understanding '01*, vol. 81, no. 3, pp. 398-413, 2001.
- [11] K. Grochow, S. L. Martin, A. Hertzmann and Z. Popović, "Style-based inverse kinematics," *ACM Transactions on Graphics '04*, vol. 23, no. 3, pp. 522-531, 2004.
- [12] F. R. Hampel, E. M. Ronchetti, P. J. Rousseeuw and W. A. Stahel, *Robust statistics: The approach based on influence functions*, Wiley.
- [13] P. Huber, *Robust statistics*, Wiley.
- [14] L. Ikemoto, O. Arikan and D. Forsyth, "Knowing when to put your foot down," *Proc. the 2006 Symposium on Interactive 3D graphics and games*, pp. 49-53, 2006.
- [15] T. R. Jones, F. Durand and M. Desbrun, "Noniterative, feature-preserving mesh smoothing," *ACM Transactions on Graphics '03*, vol. 22, no. 3, pp. 943-949, 2003.
- [16] L. Kovar, J. Schreiner and M. Gleicher, "Footskate cleanup for motion capture editing," *ACM SIGGRAPH/Eurographics Symposium on Computer Animation '02*, pp. 97-104, 2002.
- [17] G. Liu and L. McMillan, "Segment-based human motion compression," *Proc. the 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation '06*, pp. 127-135, 2006.
- [18] J. Lee and S. Y. Shin, "General construction of timedomain filters for orientation data," *IEEE Transactions on Visualization and Computer Graphics '02*, vol. 8, no. 2, pp. 119-128, 2002.
- [19] Y. Li, T. Wang and H.-Y. Shum, "Motion texture: A two-level statistical model for character synthesis," *ACM Transactions on Graphics '02*, vol. 21, no. 3, pp. 465-472, 2002.
- [20] L. M. Tanco and A. Hilton, "Realistic synthesis of novel human movements from a database of motion capture examples," *Proc. of the Workshop on Human Motion '00*, pp. 137-142, 2000.
- [21] K. Pullen and C. Bregler, "Motion capture assisted animation: Texturing and synthesis," *ACM Transactions on Graphics '02*, vol. 21, no. 3, pp. 501-508, 2002.
- [22] W. H. Press, B. P. Flannery, S. A. Teukolsky and W. T. Vetterling, *Numerical recipes in c: The art of scientific computing*, second ed. Cambridge University Press, New York, 1992.
- [23] L. Ren, A. Patrick, A. A. Efros, J. K. Hodgins and J. M. Rehg, "A data-driven approach to quantifying natural human motion," *ACM Transactions on Graphics '05*, vol. 24, pp. 1090-1097, 2005.
- [24] K. W. Sok, M. Kim and J. Lee, "Simulating biped behaviors from human motion data," *ACM Transactions on Graphics '07*, vol. 26, no. 3, article no. 3, 2007.
- [25] H. J. Shin, J. Lee, M. Gleicher and S. Y. Shin, "Computer puppetry: An importance-based approach," *ACM Transactions on Graphics '01*, vol. 20, no. 2, pp. 67-94, 2001.
- [26] H. V. Storch, F. W. Zwiers, *Statistical analysis in climate research*, Cambridge University Press, 1999.
- [27] S. Tak and H.-S. Ko, "A physically-based motion retargeting filter," *ACM Transactions on Graphics '05*, vol. 24, no. 1, pp. 98-117, 2005.
- [28] G. Welch and G. Bishop, "An introduction to the kalman filter," *ACM SIGGRAPH 2001 Course Notes*, 2001.
- [29] J. Wang, S. Drucker, M. Agrawala and M. Cohen, "The cartoon animation filter," *ACM Transactions on Graphics '06*, vol. 25, no. 3, pp. 1169-1173, 2006.
- [30] G. Xu and Z. Zhang, *Epipolar Geometry in Stereo, Motion, and Object Recognition: A Unified Approach*, Kluwer, 1996.
- [31] K. Yamane and Y. Nakamura, "Dynamics filter - concept and implementation of online motion generator for human figures," *IEEE Transactions on Robotics and Automation '03*, vol. 19, no. 3, pp. 421-432, 2003.
- [32] Vicon blade, <http://www.vicon.com/products/blade.html>, 2008.
- [33] Vicon Systems, <http://www.vicon.com>, 2008.
- [34] D. Vlastic, R. Adelsberger, G. Vannucci, J. Barnwell, M. Gross, W. Matusik, J. Popović, "Practical motion capture in everyday surroundings," *ACM Transactions on Graphics '07*, vol. 26, no. 3, article no. 35, 2007.