Nonstationary Shape Activities: Dynamic Models for Landmark Shape Change and Applications

Samarjit Das, *Student Member*, *IEEE*, and Namrata Vaswani, *Member*, *IEEE*

Abstract—Our goal is to develop statistical models for the shape change of a configuration of "landmark" points (key points of interest) over time and to use these models for filtering and tracking to automatically extract landmarks, synthesis, and change detection. The term "shape activity" was introduced in recent work to denote a particular stochastic model for the dynamics of landmark shapes (dynamics after global translation, scale, and rotation effects are normalized for). In that work, only models for stationary shape sequences were proposed. But most "activities" of a set of landmarks, e.g., running, jumping, or crawling, have large shape changes with respect to initial shape and hence are nonstationary. The key contribution of this work is a novel approach to define a generative model for both 2D and 3D nonstationary landmark shape sequences. Greatly improved performance using the proposed models is demonstrated for sequentially filtering noise-corrupted landmark configurations to compute Minimum Mean Procrustes Square Error (MMPSE) estimates of the true shape and for tracking human activity videos, i.e., for using the filtering to predict the locations of the landmarks (body parts) and using this prediction for faster and more accurate landmarks extraction from the current image.

Index Terms—Landmark shape sequence analysis, nonstationary shape sequences, Kendall's shape space, tangent space, tracking, particle filtering.

1 Introduction

The goal of this work is to develop statistical models for the shape change of a configuration of "landmark" points (key points of interest) over time and to use these models for filtering, tracking (to automatically extract landmarks), synthesis, and change detection applications. The "shape" of an ordered set of landmarks was defined by Kendall et al. [3] as all of the geometric information that remains when location, scale, and rotational effects are filtered out. The term "shape activity" was introduced in [4] to denote a particular stochastic model for the dynamics of "landmark shapes" (dynamics after global translation, scale, and rotation effects are normalized for). A model for shape change is invariant to camera motion under the weak perspective model (also referred to as the scaled orthographic camera) [5], which is a valid assumption when the scene depth is small compared to distance from the camera. The models studied in [4] were primarily for modeling stationary shape activities (SSA) of 2D landmarks (assume constant "mean shape"). In this work, we propose models for the dynamics of nonstationary shape sequences (referred to as "nonstationary shape activities" (NSSA)) of 2D and 3D landmarks. Most "activities" of a set of landmarks, for example, see Fig. 6, are not stationary, and hence, this more general model is needed. Even if the activity is actually stationary, it still gets tracked using our model.

Two-dimensional landmarks are usually the 2D coordinates of feature points of interest in an image sequence, e.g., these could be the joints of the different body parts of the human body and the goal could be to model and track articulated human body motion (see Figs. 6 and 7). Alternatively, these could be the locations of a set of interacting point objects and the goal could be to track their collective behavior over time and detect abnormalities [4]. Three-dimensional landmark shape sequences are often obtained from a time sequence of volume images, e.g., by manually or automatically extracting landmarks from a 3D heart MR image sequence or from a time sequence of brain MRI volumes. Two-dimensional or 3D landmarks may also be obtained from motion capture (MOCAP) [6] data where sensors are attached to various joints of the human body and their 3D coordinates measured over time. The Carnegie Mellon Motion Capture database is a common example. Modeling Mocap data have applications in biomechanics and graphics to understand the motion of human joints in various actions.

1.1 Our Contributions and Related Work

The key *contribution* of this work is a novel approach to define a generative model for 2D and 3D nonstationary landmark shape sequences.[1] The main idea is to compute the tangent space representation of the current shape in the

• *The authors are with the Department of Electrical and Computer Engineering, Iowa State University, Ames, IA 50011. E-mail: {samarjit, namrata}@iastate.edu.*

Manuscript received 7 Nov. 2008; revised 20 Mar. 2009; accepted 9 Apr. 2009; published online 24 Apr. 2009.
Recommended for acceptance by A. Srivastava, J.N. Damon, I.L. Dryden, and I.H. Jermyn.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMISI-2008-11-0774.
Digital Object Identifier no. 10.1109/TPAMI.2009.94.

1. We could have just defined the model for m-D landmark shape sequences and 2D or 3D would follow as special cases. But we model the 2D case separately since both shape computation from preshapes (compare (1) versus (16)) and Procrustes mean computation are more efficient in 2D than in general m-D (where the mean is computed using an iterative algorithm) [7].

tangent space at the previous shape. This can be referred to as the "shape velocity" vector since it quantifies the "difference" between two consecutive shapes projected into the tangent space at the first one. The coefficients of shape velocity along the orthogonal basis directions spanning the current tangent space ("shape speed") can be modeled using standard vector time-series models. An important requirement in doing this is to ensure that the basis directions of the current tangent space are *aligned* with those of the previous one. For both 2D and 3D shape sequences, we use the tangent space projections defined in [7, pages 71-77].

A second *contribution* of our work is demonstrating the use of our nonstationary model for

1. sequentially filtering noise-corrupted landmark configurations to compute Minimum Mean Procrustes Square Error (MMPSE) estimates of the true shape;
2. tracking, i.e., for using the filtering to predict the locations of the landmarks at the current time and using this prediction for faster and more accurate landmarks' extraction from the current image;
3. synthesis;
4. change detection.

Most of our experiments focus on 1 and 2. Greatly improved performance of our tracking and filtering algorithm over existing work [8], [4], [9] is demonstrated. Due to the nonlinearities in the shape dynamics model and the non-Gaussian observation model (similar to that of Condensation [10]), we use a particle filter (PF) [11] for filtering and tracking. Our tracking problem is a typical example of a large-dimensional problem with frequently multimodal observation likelihoods (due to background clutter and missing landmarks), and hence, we replace the basic PF used in previous work by the recently proposed PF with Efficient Importance Sampling (PF-EIS). We demonstrate that PF-EIS has a much better performance for landmark shape tracking than the basic PF, when the number of particles used is small.

In recent years, there has been a large amount of work on modeling sequences of landmark shapes—both in statistics [7], [12] and in computer vision and medical image analysis [8], [4], [13], [14], [10], [15], [16], [17], [18]. Active shape models (ASMs) [8] and SSAs [4] both assume stationarity of the shape sequence (single mean shape plus stationary deviations about it).

But, in most real applications, there is large shape variation over a long sequence, and therefore, a single mean shape plus an ASM or SSA model does not suffice. This is explained in more detail in Section 2.2. For example, consider a running sequence (see Fig. 6). Another example is the changes in shape within a single heart cycle. In existing work, the ASM is usually replaced by piecewise ASMs [13], for example, different ASMs are used for systolic and diastolic motions in [13] or SSA is replaced by piecewise SSA [14]. Piecewise ASMs are good for recognition problems, but not for automatic tracking or for compression since they do not model the transitions between pieces well. When piecewise SSA was used for tracking in [14], it needed to use separate change detection and shape recognition procedures to detect when and which piece to switch to. In this work, we demonstrate through extensive experiments that both filtering and tracking using our model significantly outperform either ASMs or SSAs.

Smoothing splines [12] is, to the best of our knowledge, the only other existing work that truly models nonstationary landmark shape sequences (other than the piecewise models discussed above). But it does not provide a generative model for the shape sequences, which is the key requirement in tracking, compression, or synthesis applications.

A key difference of our work from Condensation [10] is that the latter only models and tracks global affine deformation between two landmark configurations. This is a valid model for rigid or approximately rigid object motion, but not for modeling shape change of different parts of the human body performing actions such as running or jumping, where there is significant local shape deformation which is not affine.

Our modeling approach is similar in spirit to [19], which also uses piecewise geodesic priors to define a generative model but in a very different context.

Other related work includes Active Appearance Models [15] and Active Appearance Motion Models [16], which also model appearance, and hence, are not invariant to intensity changes between training and test data, and work on articulated human body tracking [17], [18], [20].

The paper is organized as follows: In Section 2, we explain the nonstationary model for 2D landmark shape sequences and its parameter estimation. We discuss the same for 3D shape sequences in Section 3. Algorithms for filtering and tracking are developed in Section 4. Experimental results are given in Section 5. We conclude the paper in Section 6.

2 Modeling 2D Shape Sequences

For modeling human motion activity or any activity involving multiple interacting objects, we represent body joints/objects as the landmark points and the corresponding activity is represented as a sequence of deforming landmark shapes over time. It is done in two steps. First, we transform the shape sequence to a vector time series using the nonstationary shape deformation model. Then, we fit standard statistical models to the time series.

2.1 2D Landmark Shape Analysis Preliminaries

The **configuration** S is an ordered set of K landmarks. In the 2D case, it can be represented as a K-dimensional complex vector [7, page 39], with the x (and y) coordinates forming the real (and imaginary) parts. The **preshape** w is obtained by translation and scale normalization, and the shape z is obtained by rotation normalization w.r.t. a given shape μ. For details, see [7, Chapter 3] or [4, Section 2]. The Procrustes distance and Procrustes mean are also defined there. The complex eigenvector solution for computing the Procrustes mean shape can be found in [7, Chap. 3] and [21].

As explained in [7, Chap. 4], the shape space $\mathcal{M}$ is a manifold in $\mathcal{C}^{K-1}$, and hence, its actual dimension is $\mathcal{C}^{K-2}$. Thus, the tangent plane at any point of the shape space is a $\mathcal{C}^{K-2}$-dimensional hyperplane in $\mathcal{C}^K$ [7]. The projection of a configuration S into the tangent space at a pole μ can be computed [7] as follows:

$$\begin{aligned} y &= C_K S, \quad C_K \triangleq I_K - 1_K 1_K^T/K, \\ w &= y/\|y\|, \\ \theta(w,\mu) &= angle(w^*\mu), \quad z(w,\mu) = we^{j\theta(w,\mu)}, \end{aligned} \tag{1}$$

$$v(z,\mu) = [I_K - \mu\mu^*]z. \tag{2}$$

Here, I_K is a $K \times K$ identity matrix and 1_K is a column vector with K rows with all entries as 1. The notation x^T denotes transpose and x^* denotes conjugate transpose. The first three equations involve translation, scale, and rotation normalization, respectively, and the last one involves projecting the shape z into the tangent space at μ.

The projection from tangent space to shape space is given by [7]:

$$z = (1 - v^*v)^{\frac{1}{2}}\,\mu + v. \tag{3}$$

2.2 Problem with SSA and ASM Models

The SSA model proposed in [4] computed a single mean shape μ for a training sequence and aligned each preshape w_t in the sequence to μ to obtain the shape sequence z_t. Tangent projections $v(z_t,\mu)$ of each z_t were computed in the tangent space at μ and their time series was modeled using an autoregressive (AR) model. The work of [9] replaced AR by ARMA models and used the models for recognition problems. The ASM of [8] assumed that z_t belongs to a vector space and replaced the tangent space projection given in (2) by its linear version $v(z_t,\mu) = z_t - \mu$ and modeled the time series of $v(z_t,\mu)$.

Since both SSA and ASM assumed a single mean shape, they could model only small deviations from mean, which is only possible for stationary sequences. But, in many applications, this assumption may not hold, for example, a crawling or a dancing sequence or see Fig. 6. In these cases, the mean shapes for different time intervals are different. Or, in other words, considering the entire sequence, the shape activity is essentially nonstationary. Now, if we force a fixed mean shape to such a deforming shape sequence, the resulting shapes z_t would drift too far away from μ. It is important to note that a single tangent space approximation works as long as each element of $v(z_t,\mu)$ for all shapes is less than 1 (otherwise, the square root in (3) will be of a negative number). Also, a time-invariant AR or ARMA model on $v(z_t,\mu)$s is a valid one only if the magnitudes of each element of $v(z_t,\mu)$ are significantly smaller than 1 (this is because, when $v(z_t,\mu)$ is large, i.e., when z_t is far from μ, small changes in $v(z_t,\mu)$ would correspond to very large changes in z_t). But, for large shape variation, $v(z_t,\mu)$ will be large. In such a scenario, both SSA and ASM would fail to correctly model the shape dynamics.

2.3 Modeling Nonstationary Shape Sequences

To model a nonstationary shape sequence, we use $\mu = z_{t-1}$ at time t. Thus,

$$\begin{aligned} z_t &:= w_t \frac{w_t^* z_{t-1}}{|w_t^* z_{t-1}|}, \\ v_t &:= v(z_t, z_{t-1}) = [I - z_{t-1}z_{t-1}^*]z_t. \end{aligned} \tag{4}$$

The inverse map is given by

$$z_t = (1 - v_t^* v_t)^{\frac{1}{2}}\, z_{t-1} + v_t. \tag{5}$$

Since the projection of z_{t-1} in the tangent space at z_{t-1}, $T_{z_{t-1}}$, is zero, v_t can be interpreted as the difference $(z_t - z_{t-1})$ projected into $T_{z_{t-1}}$, i.e., it is the "shape velocity" at time t.

The translation, scale, and rotation normalization in 2D removes two complex dimensions (four real dimensions), and thus, the shape space is a $K-2$-dimensional manifold in $\mathcal{C}^K$ and so the tangent space is a $K-2$-dimensional hyperplane in $\mathcal{C}^K$ [7]. Thus, the shape velocity v_t has only $K-2$-independent complex dimensions, i.e., it can be rewritten as $v_t = U_t\tilde{c}_t$, where the columns of $(U_t)_{K\times K-2}$ contain the $K-2$ orthonormal basis directions spanning $T_{z_{t-1}}$ and $\tilde{c}_t \in \mathcal{C}^{K-2}$ are the basis coefficients. $\tilde{c}_t$ may be interpreted as a "shape speed" vector.

Note that, by definition, $T_{z_{t-1}}$ is perpendicular to z_{t-1} and 1_K. Also, z_{t-1} is perpendicular to 1_K (due to translation normalization). Thus, the projection matrix for $T_{z_{t-1}}$ is $[I_K - z_{t-1}z_{t-1}^*]C_K = [I_K - z_{t-1}z_{t-1}^* - 1_K 1_K^T/K]$. In other words, U_t satisfies

$$U_t U_t^* = \left[I_K - z_{t-1}z_{t-1}^* - 1_K 1_K^T/K\right]. \tag{6}$$

One way to obtain U_t is by computing the Singular Value Decomposition (SVD) of the right-hand side (RHS) of (6) and setting the columns of U_t equal to the left singular vectors with nonzero singular values. Denote this operation by $U_t = left.singular.vectors(M(z_{t-1}))$, where

$$M(z) \triangleq \left[I_K - zz^* - 1_K 1_K^T/K\right]. \tag{7}$$

This was used in [1], [4]. But, if this is done at each t, the columns of U_t and U_{t-1} may not be aligned. As an extreme example, consider the following. Let $K = 4$. It may happen that

$$U_{t-1} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

and

$$U_t = \begin{bmatrix} 0.1 & 0.995 \\ 0.995 & -0.1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}.$$

In this case, it is obvious that the first column of U_t corresponds to the second column of U_{t-1} and vice versa for the second column of U_t. Or, in other words, $\tilde{c}_{t,1}$ corresponds to $\tilde{c}_{t-1,2}$ and $\tilde{c}_{t,2}$ to $\tilde{c}_{t-1,1}$. Thus, if SVD is used to obtain U_t at each t, the $\tilde{c}_t$s cannot be assumed to be identically distributed, it is therefore incorrect to model them by an AR model, which assumes stationarity of $\tilde{c}_t$. Notice the large modeling error of this method (NSSA-unaligned) in Fig. 2a.

We fix this problem as follows (also, see Fig. 1): To obtain an aligned sequence of basis directions over time, we obtain the mth column of U_t by starting with the mth column of U_{t-1}, making it perpendicular to z_{t-1} (by subtracting $z_{t-1}z_{t-1}^*$), and then using Gram-Schmidt orthogonalization to also make the resulting vector perpendicular to the first $m-1$ columns of U_t (i.e., by further subtracting out $\sum_{j=1}^{m-1}(U_t)_j(U_t)_j^*$). This procedure can be summarized as

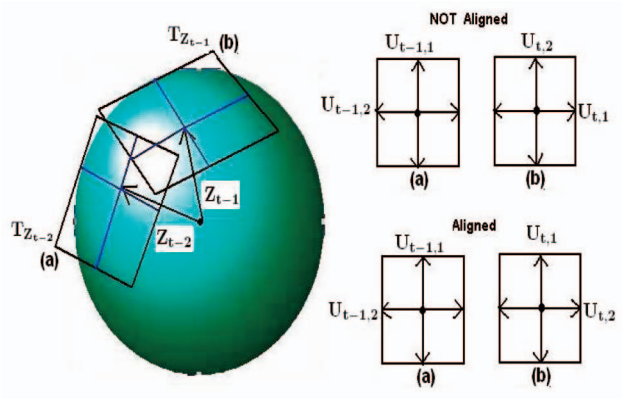

Fig. 1. This figure shows the alignment of successive tangent spaces for NSSA. When using [4], [1], the axes (here, x and y) of the consecutive tangent planes may not be aligned (top). Our method gives aligned axes (bottom).

$$U_t = g(U_{t-1}, z_{t-1}),\ where$$
$$g(.)_m \triangleq \left[I - z_{t-1}z_{t-1}^* - \sum_{j=1}^{m-1} g(.)_j g(.)_j^*\right](U_{t-1})_m, \quad (8)$$
$$\forall m = 1, \ldots (K-2).$$

Here, $g(.)_m$ denotes the mth columns of $g(U_{t-1}, z_{t-1})$. U_0 is initialized as $U_0 = left.singular.vectors(M(z_0))$.

Now, since the columns of U_t are aligned, it is fair to assume that $\tilde{c}_{t,j}$s are identically distributed for each j over time. Since they are also temporally correlated, we model them by an autoregressive model with lag 1 (AR(1) model). For simplicity of notation, we first convert $\tilde{c}_t$ into a $2K-4$-dimensional real vector. We denote this operation by

$$c_t = vec(\tilde{c}_t), \quad (9)$$

and the inverse operation (obtaining the complex vector) is denoted by $\tilde{c}_t = vec^{-1}(c_t)$. Thus, in summary, the dynamical model of the state $X_t = [U_t, z_t, c_t]$ is given by

$$\begin{aligned} c_t &= A_c c_{t-1} + \nu_{c,t},\ \nu_{c,t} \sim \mathcal{N}(0, \Sigma_c), \\ U_t &= g(U_{t-1}, z_{t-1}), \\ z_t &= \left(1 - c_t^T c_t\right)^{1/2} z_{t-1} + U_t vec^{-1}(c_t), \end{aligned} \quad (10)$$

where $\mathcal{N}(\mu, \Sigma)$ denotes Gaussian pdf with mean and covariance matrix Σ. The last equation follows from (3) and the fact that $v_t = U_t\tilde{c}_t$, $U_t^*U_t = I$, and $\tilde{c}_t^*\tilde{c}_t = c_t^T c_t$. The above model is initialized with

$$z_0 = w_0,\ U_0 = left.singular.vectors(M(z_0)),\ c_0 = 0. \quad (11)$$

2.4 Model Parameter Estimation

The above model is completely specified by $z_{init} = w_0$, A_c, Σ_c. A_c is the AR transition matrix and Σ_c is the modeling error covariance matrix in the AR model. Given a training sequence of landmark configurations, $\{S_t\}_{t=0}^{N-1}$, a maximum-likelihood (ML) estimate of the parameters can be obtained as follows:

1. Obtain the shape sequence $\{z_t\}$ by translation, scale, and rotation normalization of $\{S_t\}_{t=0}^{(N-1)}$, i.e., compute $y_t = C_K S_t$ and $w_t = \frac{y_t}{\|y_t\|}$ for each t. Set $z_0 = w_0$. Compute

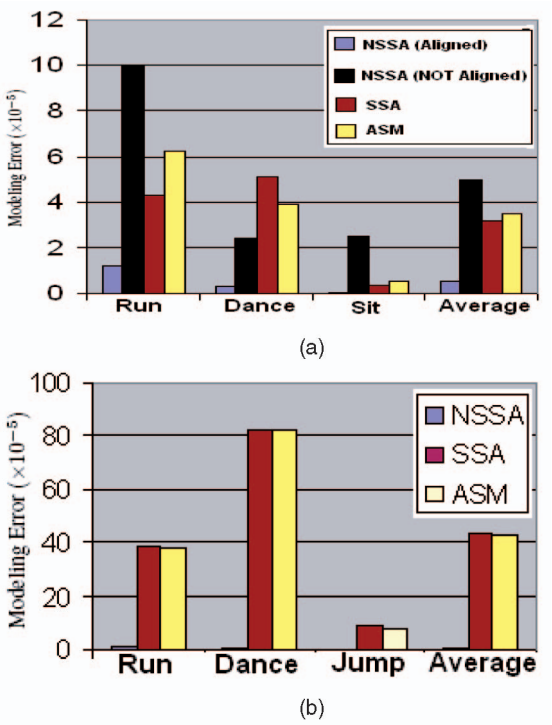

Fig. 2. (a) The ME for NSSA, ASM, and ASM for a few activities using 2D MOCAP data. It is important to note that NSSA without the basis alignment has a very large modeling error. While after the basis alignment is taken into account, NSSA has much lower ME than SSA and ASM. (b) The ME for NSSA, ASM, and ASM for a few activities using 3D MOCAP data. Again, NSSA had much lower ME compared to that SSA and ASM. That is why the corresponding bar plot for NSSA modeling error has almost disappeared.

$$z_t = w_t \frac{w_t^* z_{t-1}}{|w_t^* z_{t-1}|},\ \forall t > 0. \quad (12)$$

2. For all t, obtain the shape velocity coefficients $\{c_t\}$ from $\{z_t\}$. This involves computing

$$\begin{aligned} U_t &= g(U_{t-1}, z_{t-1}), \\ \tilde{c}_t &= U_t^* v_t = U_t^* z_t, \\ c_t &= vec(\tilde{c}_t), \end{aligned} \quad (13)$$

starting with $U_0 = left.singular.vectors(M(z_0))$. The second equation above follows because

$$\begin{aligned} U_t^* v_t &= U_t^*[I_K - z_{t-1}z_{t-1}^*]z_t \\ &= U_t^*\left[I_K - z_{t-1}z_{t-1}^* - 1_K 1_K^T/K\right]z_t \\ &= U_t^* U_t U_t^* z_t = U_t^* z_t \end{aligned}$$

(the second equality follows because z_t is translation normalized so that $1_K^T z_t = 0$ and third one follows because, by definition, $U_t U_t^* = [I_K - z_{t-1}z_{t-1}^* - 1_K 1_K^T/K]$).

3. Obtain an ML estimate of the AR model parameters A_c, Σ_c from $\{c_t\}$ by using the Yule-Walker equations, i.e.,

$$A_c = R_c(1)R_c(0)^{-1},\ where$$
$$R_c(0) = \frac{1}{N}\sum_{t=0}^{N-1} c_t c_t^T,\ R_c(1) = \frac{1}{N-1}\sum_{t=1}^{N-1} c_t c_{t-1}^T, \quad (14)$$
$$\Sigma_c = \frac{1}{N-1}\sum_{t=1}^{N-1}(c_t - A_c c_{t-1})(c_t - A_c c_{t-1})^T.$$

2.4.1 Using Multiple Training Sequences

If more than one training sequence is available, one can compute a mean z_{init} (denoted by $\tilde{z}_{init}$) by aligning the initial preshapes w_0 of all the sequences. We set z_0 for each sequence as the corresponding w_0 aligned to $\tilde{z}_{init}$. These operations make sure that the initial shapes of all the training sequences are aligned. Now, starting with z_0, we can obtain the shape speed c_ts for each sequence. Say, we have a total of q training sequences for a given motion activity, each with length N. We denote c_ts corresponding to the ith sequence as $\{c_t^i\}$, where $i = 1, \ldots, q$. Now, we can estimate $R_c(0), R_c(1)$ as $R_c(0) = \frac{1}{q}\sum_{i=1}^{q}\frac{1}{N}\sum_{t=0}^{N-1} c_t^i c_t^{iT}$ and $R_c(1) = \frac{1}{q}\sum_{i=1}^{q}\frac{1}{N-1}\sum_{t=1}^{N-1} c_t^i c_{t-1}^{iT}$. Finally, we compute $A_c = R_c(1)R_c(0)^{-1}$ and $\Sigma_c = \frac{1}{q}\sum_{i=1}^{q}\Sigma_c^i$, where

$$\Sigma_c^i = \frac{1}{N-1}\sum_{t=1}^{N-1}\left(c_t^i - A_c c_{t-1}^i\right)\left(c_t^i - A_c c_{t-1}^i\right)^T.$$

The entire procedure is summarized in Algorithm 1.

Algorithm 1. 2D NSSA: Training with Multiple Training Sequences For a Given Motion Activity

Input: Preshapes corresponding to q training sequences $(\{w_t^i\}_{t=0}^{N-1},\ i = 1, \ldots, q)$

Output: Computed parameters $\tilde{z}_{init}, A_c, \Sigma_c$.

1) Compute $\tilde{z}_{init} = \mu(w_0^1, w_0^2, \ldots, w_0^q,)$ where $\mu(.)$ is the Procrustes mean shape [7], [21].
2) Compute $\tilde{U}_{init} = left.singular.vectors(M(\tilde{z}_{init}))$ where M(.) is given in (7).
3) For each i, $i = 1, 2, \ldots q$
   a) Compute $z_0^i = z(w_0^i, \tilde{z}_{init})$ using (1), compute $U_0^i = g(\tilde{U}_{init}, z_0^i)$ using (8) and set $c_0^i = 0$.
   b) For each t, $t = 1, \ldots N-1$ do
      i) Compute $z_t^i,\ U_t^i,\ c_t^i$ using (12), (13).
4) Compute $A_c = R_c(1)R_c(0)^{-1}$ where,

$$R_c(0) = \frac{1}{q}\sum_{i=1}^{q}\frac{1}{N}\sum_{t=0}^{N-1} c_t^i c_t^{iT} \text{ and}$$

$$R_c(1) = \frac{1}{q}\sum_{i=1}^{q}\frac{1}{N-1}\sum_{t=1}^{N-1} c_t^i c_{t-1}^{iT}$$

5) Compute $\Sigma_c = \frac{1}{q}\frac{1}{N-1}\sum_{i=1}^{q}\sum_{t=1}^{N-1}(c_t^i - A_c c_{t-1}^i)(c_t^i - A_c c_{t-1}^i)^T$

3 Modeling 3D Shape Sequences

A 3D configuration is represented by a set of K ordered landmarks as a $(K \times 3)$ matrix whose each row corresponds to the (x, y, z) coordinates of the corresponding landmark. In this section, we discuss the basics of 3D landmark shape analysis [7] and then develop 3D nonstationary shape activity model (3D-NSSA).

3.1 3D Landmark Shape Analysis Preliminaries

For 3D shapes, the computation of preshape $(w)_{K\times 3}$ from raw shape $(S)_{K\times 3}$ is similar to the 2D case, i.e., first get the centered shape $(y)_{K\times 3}$ and then perform size normalization:

$$y = C_K S, \quad \text{where } C_K \text{ is given in (1)}$$
$$w = \frac{y}{\|y\|_F}. \quad (15)$$

Here, $\|.\|_F$ denotes *Frobenius norm* of a matrix. The rotation aligned shape z is obtained from preshape w in the following way: Say, we want to align $(w)_{K\times 3}$ w.r.t. $(\mu)_{K\times 3}$. We do this as

$$z = w\mathcal{U}\mathcal{V}^T, \quad \text{where}$$
$$\mathcal{V}\Lambda\mathcal{U}^T = SVD(\mu^T w), \quad (16)$$

where $\mathcal{V}, \mathcal{U}$ are the left and right singular vectors of the 3×3 matrix $(\mu^T w)$. As pointed out by an anonymous reviewer, while performing 3D shape alignment, we may have reflections unlike the 2D case. This happens if $det(\mathcal{U}) = -1$ or $det(\mathcal{V}) = -1$, where $det(.)$ denotes determinant. Thus, 3D alignment is a bit different from 2D since reflections are allowed in 3D but not in 2D.

Another important thing about 3D shape analysis is the vectorization operation [7]. Say, $\mathbf{z}$ is the shape at a given instant which is a $(K \times 3)$ matrix with columns $\mathbf{z_1}, \mathbf{z_2}, \mathbf{z_3}$. We vectorize $\mathbf{z}$ to a $3K$ length vector as follows:

$$vec_{3D}(\mathbf{z}) = \left(\mathbf{z_1^T}, \mathbf{z_2^T}, \mathbf{z_3^T}\right)^T. \quad (17)$$

The inverse operation is given by $vec_{3D}^{-1}(.)$, which forms a $K \times 3$ matrix from a $3K$ length vector. The tangent space coordinate $v(z, \mu)$ of a shape z w.r.t. the shape μ is given as follows:

$$v(z, \mu) = [I_{3K} - vec_{3D}(\mu)vec_{3D}(\mu)^T]vec(z). \quad (18)$$

The inverse map (i.e., from tangent space to shape space) is given as

$$z = vec_{3D}^{-1}((1 - v^T v)^{\frac{1}{2}}\, vec_{3D}(\mu) + v). \quad (19)$$

3.2 3D Nonstationary Shape Activity (3D-NSSA)

To define an NSSA model on 3D shape data, we first obtain the translation- and scale-normalized preshape sequence $\{w_t\}$ from the 3D configuration sequence $\{S_t\}$ using (15). As in the 2D case, we use $\mu = z_{t-1}$ to compute the shape sequence followed by computing the shape velocity and shape speed vectors in an exactly analogous fashion. The final procedure can be summarized as follows:

First, we have $z_{init} = z_0 = w_0$ and then we compute the initial tangent space basis matrix as $U_{init} = U_0 = left.singular.vectors(M_{3D}(z_0))$, where

$$M_{3D}(z) \triangleq [I_{3K} - vec(z)vec(z)^T]C_{K,3D}, \quad (20)$$

where

$$C_{K,3D} = \begin{bmatrix} C_K & \mathbf{0_{K\times K}} & \mathbf{0_{K\times K}} \\ \mathbf{0_{K\times K}} & C_K & \mathbf{0_{K\times K}} \\ \mathbf{0_{K\times K}} & \mathbf{0_{K\times K}} & C_K \end{bmatrix}.$$

Here, $\mathbf{0_{K\times K}}$ is a $(K \times K)$ matrix with all zero entries and C_K is defined in (1). Now, starting with z_0 and U_0, the computation of the corresponding time sequence of *shape speed* vectors is done as follows:

$$z_t = w_t \mathcal{U}\mathcal{V}^T, \text{ where } \mathcal{V}\Lambda\mathcal{U}^T = SVD\big(z_{t-1}^T w_t\big), \tag{21}$$

$$U_t = g(U_{t-1}, vec_{3D}(z_{t-1})), \tag{22}$$

$$c_t = U_t^T v_t = U_t^T vec_{3D}(z_t), \tag{23}$$

where $g(.)$ is defined in (8). The reason why $U_t^T v_t = U_t^T vec_{3D}(z_t)$ is similar to the 2D case (see discussion below (13)).

We model c_t using a first order AR model (in general, this may be replaced by any appropriate model for c_t). Thus, the forward model for generating a 3D shape sequence is

$$\begin{aligned} c_t &= A_c c_{t-1} + n_t,\ n_t \sim \mathcal{N}(0, \Sigma_c), \\ U_t &= g(U_{t-1}, vec_{3D}(z_{t-1})), \\ z_t &= vec_{3D}^{-1}\big(\big(1 - c_t^T c_t\big)^{\frac{1}{2}} vec_{3D}(z_{t-1}) + U_t c_t\big). \end{aligned} \tag{24}$$

The last equation follows from (19) and the fact that $v_t^T v_t = (U_t c_t)^T U_t c_t = c_t^T (U_t^T U_t) c_t = c_t^T c_t$ and $U_t^T U_t = I$.

3.3 Model Parameter Estimation

The parameter estimation algorithm for the 3D case can be summarized as follows:

1. For all t, obtain $\{w_t\}$ from a given 3D landmark configuration sequence $\{S_t\}$.
2. For all t, compute $\{z_t\}$ from $\{w_t\}$ using (21).
3. For all t, compute $\{c_t\}$ from $\{z_t\}$ using (22) and (23).
4. Estimate the AR model parameters for c_t using the Yule-Walker equations given in (14).

4 Filtering and Tracking

The goal of filtering is to filter out the noise and get a good estimate of the true landmark shape from noisy observed landmarks. In our algorithm, the particle filter takes noisy observed landmark data as input and outputs the MMPSE estimate of the true landmark shape. The MMPSE estimate of shape can be derived by following the Procrustes mean derivation [7] as:

$$\begin{aligned} \hat{z}_t &= \arg\min_{\mu} E[d^2(z_t, \mu)|Y_{1:t}] \\ &= \arg\min_{\mu} E[\|z_t z_t^* \mu - \mu\|^2 |Y_{1:t}] \\ &= \arg\max_{\mu} \mu^* E[z_t z_t^* | Y_{1:t}]\mu, \end{aligned} \tag{25}$$

where $E[\cdot|Y_{1:t}]$ denotes the conditional expectation given $Y_{1:t}$, d denotes the Procrustes distance [7], and $Y_{1:t}$ are the observations until t. The last equality follows because $z_t^* z_t = 1$ and $\mu^* \mu = 1$. Under a particle filtering setup, the MMPSE estimate is computed as $\hat{z}_t =$ principal eigenvector of $\sum_{i=1}^{N_{pf}} z_t^i z_t^{i*} w_t^i$, where N_{pf} denotes number of particles, i denotes the ith particle, and w_t^i represents the importance weight corresponding to the ith particle, i.e.,

$$p(z_t|Y_{1:t}) \approx \sum_{i=1}^{N_{pf}} w_t^i \delta\big(z_t - z_t^i\big).$$

The configuration parameters, i.e., scale, translation, and rotation are also estimated in the process of filtering. Apart from removing random additive noise, particle filtering can also be used to "clean up" the effects of occlusion and clutter.

Tracking is used to extract and filter out landmark configurations from a sequence of images. Filtering plays a very crucial role in the process. In fact, tracking can be considered as observation extraction coupled with filtering. It works as follows: A shape deformation model (as described in Section 2.3) predicts the shape at the current instant using the previous shape estimates. Similarly, scale, translation, and rotation models are used to predict their values as well. These, coupled with the predicted shape, give the predicted landmark locations (i.e., predicted configuration) at the current instant. Using these predicted landmark locations in the current image, the landmarks can be extracted for the current image using any technique, for example, edge detection or optical flow. Our method for doing this is described in Algorithm 4 and Section 4.5. Once the observed landmarks are obtained, they are filtered to get an MMPSE estimate of the true landmark shape and MMSE estimates of scale, rotation, and translation. These estimates are again utilized to extract the observed landmark locations at the next time instant as described above.

We describe our state transition model and observation model in Sections 4.1 and 4.2. We develop the PF algorithms for filtering in Sections 4.3 and 4.4. The PF-based tracking algorithm to extract landmarks from video sequences is described in Section 4.5.

4.1 System Model (State Transition Model)

Since the observations are landmark configurations, to extract them, we need to estimate both the shape and the "motion" (scale, translation, and rotation). Thus, our state vector is $X_t = [s_t, \theta_t, \tau_t, c_t, z_t, U_t]$, where s_t is the logarithm of global scale, θ_t is the global rotation, τ_t is the xy translation, c_t is the shape speed vector, z_t is the shape, and U_t is the basis set spanning the current tangent space. The shape dynamics model is given in (10). It is a second order model on z_t which is equivalent to a first order model on the shape speed c_t. We use a first order model on logarithm of global scale s_t, global 2D rotation θ_t (this typically models the random motion of camera), and translation τ_t:

$$\begin{aligned} s_t &= \alpha_s s_{t-1} + \nu_{s,t},\ \nu_{s,t} \sim \mathcal{N}\big(0, \sigma_s^2\big), \\ \theta_t &= \theta_{t-1} + \nu_{\theta,t},\ \nu_{\theta,t} \sim \mathcal{N}\big(0, \sigma_\theta^2\big), \\ \tau_t &= \tau_{t-1} + \nu_{\tau,t},\ \nu_{\tau,t} \sim \mathcal{N}\big(0, \sigma_\tau^2\big). \end{aligned} \tag{26}$$

Note that, in case of filtering (when landmark observations are already available), translation can be normalized for. Since it is a linear process, the form of the observation noise pdf does not change. But, in case of tracking to predict and extract landmarks from image sequences, translation does need to be tracked to predict where the configuration of landmarks translated to.

The resulting state transition prior becomes:

$$p(X_t|X_{t-1}) = \mathcal{N}(\alpha_s s_{t-1}, \sigma_s^2)\mathcal{N}(\theta_{t-1}, \sigma_\theta^2) \times \mathcal{N}(\tau_{t-1}, \sigma_\tau^2)\mathcal{N}(A_c c_{t-1}, \Sigma_c) \times \delta(U_t - g(U_{t-1}, z_{t-1}))\delta(z_t - f(z_{t-1}, U_t, c_t)),$$

where δ denotes the Dirac delta function and $f(z_{t-1}, U_t, c_t) \triangleq (1 - c_t^T c_t)^{1/2} z_{t-1} + U_t vec^{-1}(c_t)$ and $g(.)$ is defined in (8).

4.2 Observation Model

There are various ways to extract landmarks from image sequences—e.g., edge detection followed by extracting the K strongest edges closest to predicted landmark locations or using the Kanade-Lucas-Tomasi (KLT) Feature Tracker [22] (block optical flow estimation) algorithm at or around predicted landmark locations. As explained in Section 4.5 and Algorithm 4, we use a modification of KLT for this purpose.

Algorithm 2. PF-Gordon for Landmark Shape Filtering
Initialization: At time $t = 0$, sample $s_0^{(i)} \sim \mathcal{N}(s_0, \sigma_s^2)$, $\theta_0^{(i)} \sim \mathcal{N}(\theta_0, \sigma_\theta^2)$, $c_0^{(i)} = \mathbf{0}$, $z_0^{(i)} = z_0$ and $U_0^{(i)} = U_0$. Here, $i = 1, \ldots, N_{pf}$ where, N_{pf} is the number of particles.
For $t > 0$,

1) Importance sample $X_t^{(i)} \sim p(X_t|X_{t-1}^{(i)})$ as $s_t^i \sim \mathcal{N}(\alpha_s s_{t-1}^i, \sigma_s^2)$, $\theta_t^i \sim \mathcal{N}(\theta_{t-1}^i, \sigma_\theta^2)$, $c_t^i \sim \mathcal{N}(A_c c_{t-1}^i, \Sigma_c)$, $U_t^i = g(U_{t-1}^i, z_{t-1}^i)$, $z_t^i = f(z_{t-1}^i, U_t^i, c_t^i)$. $i = 1, 2, \ldots, N_{pf}$
2) Weight and Resample. Compute $w_t^i = \frac{\tilde{w}_t^i}{\sum_{j=1}^{N_{pf}} \tilde{w}_t^j}$ where, $\tilde{w}_t^i = w_{t-1}^i p(Y_t|X_t^i)$ with, $p(Y_t|X_t^i) = p(Y_t|h(s_t^i, \theta_t^i, z_t^i)) = \prod_{k=1}^K [(1-p)\mathcal{N}([h(s_t^i, \theta_t^i, z_t^i)]_k, \sigma_o^2) + p\mathcal{N}(0, 100\sigma_o^2)]$, $\forall i$
3) Compute the MMPSE estimate $\hat{z}_t$ as the principal eigenvector of $\sum_{i=1}^{N_{pf}} z_t^i z_t^{i*} w_t^i$. Estimate configuration parameters as $\hat{s}_t = \sum_{i=1}^{N_{pf}} w_t^i s_t^i$ and $\hat{\theta}_t = \sum_{i=1}^{N_{pf}} w_t^i \theta_t^i$
4) Set $t \leftarrow t + 1$ and go to step 1.

The configuration of landmarks is obtained from the shape, scale, and rotation by the transformation $h(s_t, \theta_t, z_t) = z_t e^{s_t} e^{j\theta_t}$. The simplest observation model is of the form

$$Y_t = h(s_t, \theta_t, z_t) + w_t,\ w_t \sim \mathcal{N}(0, \sigma_o^2 I), \tag{27}$$

where w_t is a complex Gaussian noise vector. This assumes that there is no background clutter: each of the K strongest edges or the K KLT-feature points are always generated by the true landmark location plus some error modeled as Gaussian noise. But this is often a simplistic model since there is always background clutter that generates false edges or false KLT-feature matches or there might be missing landmarks due to blur or occlusion. Thus, it may happen that, out of the K "observed landmark locations," some landmark at some time is actually generated by clutter (e.g., if a true landmark is blurred or occluded, while a nearby clutter point has a stronger edge). We model this as follows: with a small probability p, the kth landmark $Y_{t,k}$ is generated by a clutter point (model a clutter point location as a large variance Gaussian or by a uniform), independent of other landmarks. With probability $(1 - p)$, it is generated by a Gaussian-noise-corrupted actual landmark (independent of other landmarks), i.e.,

$$Y_{t,k} \sim (1-p)\mathcal{N}([h(s_t, \theta_t, z_t)]_k, \sigma_o^2) + p\mathcal{N}(0, 100\sigma_o^2). \tag{28}$$

The above model has been adapted from the observation model used in Condensation [10]. The resulting observation likelihood term is

$$p(Y_t|X_t) = \prod_{k=1}^K (1-p)\mathcal{N}([h(s_t, \theta_t, z_t)]_k, \sigma_o^2) + p\mathcal{N}(0, 100\sigma_o^2).$$

4.3 Particle Filter with Efficient Importance Sampling

The first PF algorithm, PF-Gordon [11], used the state transition prior (i.e., $p(X_t|X_{t-1})$) as the importance density. This assumes nothing and has very small computation burden per particle. But since it does not use knowledge of the current observation, the weights variance can be large, particularly when the observations are more reliable than the prior model. Thus, it requires more particles for a given accuracy level compared to the case when the knowledge of observations is used. The optimal importance density [23] is given by the posterior conditioned on the previous state, denoted by p^*, where

$$p^*(X_t) \triangleq p(X_t|X_{t-1}, Y_t). \tag{29}$$

But, in most problems, including ours, p^* cannot be computed analytically. When it is unimodal, PF-Doucet [23] approximates it by a Gaussian about its mode (Laplace's approximation [24]) and samples from the Gaussian. Other work that also implicitly assumes that p^* is unimodal includes [25]. But, in our case, the observation likelihood is a raised Gaussian as a function of $[h(.)]_k$ and is thus heavy tailed. If the equivalent state transition prior of $[h(.)]_k$ is broad (e.g., this will happen if STP of s_t or θ_t is broad), whenever $Y_{t,k}$ is generated from the outlier distribution (i.e., is far from the predicted landmark location), the resulting posterior given the previous state $p^*(X_t) \triangleq p(X_t|X_{t-1}, Y_t)$ will be multimodal.

For such problems where p^* is often multimodal, a particle filter with efficient importance sampling (PF-EIS) was proposed in [26] which combines the ideas of both PF-Gordon and PF-Doucet to handle multimodal observation likelihoods. This algorithm relies on the fact that even though p^* is multimodal, for most real-life problems, it is possible to split the state vector X_t into an "effective basis" $X_{t,s}$ and "residual space" $X_{t,r}$ in such a way that p^*, conditioned on $X_{t,s}$, is unimodal, i.e.,

$$p^{**,i}(X_{t,r}) \triangleq p^*(X_t|X_{t,s}^i) = p(X_{t,r}|X_{t-1}^i, X_{t,s}^i, Y_t) \tag{30}$$

is unimodal. Here, the index i represents the sample from the ith particle. We sample the $X_{t,s}$ particle, $X_{t,s}^i$, from its state transition prior (STP) but use Laplace's approximation [24], [23] to approximate $p^{**,i}$ by a Gaussian and sample $X_{t,r}$ from it. Thus, we sample $X_{t,r}^i$ from $\mathcal{N}(m_t^i, \Sigma_{IS}^i)$, where

$$m_t^i = \arg\min_{X_{t,r}} [-\log p^{**,i}(X_{t,r})] = \arg\min_{X_{t,r}} L^i(X_{t,r}),$$
$$\Sigma_{IS}^i = \left[\nabla^2 L^i(m_t^i)\right]^{-1},$$

where

$$L^i(X_{t,r}) \triangleq \left[-\log p(Y_t|X_{t,s}^i, X_{t,r})\right] + \left[-\log p(X_{t,r}|X_{t-1}^i, X_{t,s}^i)\right],$$

and $\mathcal{N}(\mu, \Sigma)$ denotes a Gaussian pdf with mean μ and covariance matrix Σ. As shown in [26], unimodality of $p^{**,i}$ is ensured if the variance of STP of $X_{t,r}$ is small enough compared to distance between the modes of OL given $X_{t,s}$ in any direction. Even if $X_{t,s}$ is chosen so that this holds for most particles, at most times, the proposed algorithm will work.

4.4 PF-Gordon and PF-EIS for Our Problem

We summarize the basic PF (i.e., PF-Gordon [11]) for landmark shape filtering in Algorithm 2. In order to develop the PF-EIS, we follow the steps as explained in Section 4.3. The choices of $X_{t,s}$ and $X_{t,r}$ under this problem setup are justified as follows: Since the STP of s_t, θ_t is usually broad (to allow for occasional large camera motion or zoom), we use $X_{t,s} = [s_t, \theta_t]$ and $X_{t,r} = [c_t, z_t, U_t]$. Note that, for the purpose of importance sampling, only s_t, θ_t, c_t are the "importance sampling states" since z_t, U_t are deterministically computed from c_t and X_{t-1}. The particles of $X_{t,s}$ are sampled from its state transition prior, i.e., using the first two equations of (26). Conditioned on the sampled scale and rotation, $X^i_{t,s}$, it is much more likely that p^* is unimodal, i.e., $p^{**,i}(c_t, U_t, z_t)$ defined below is unimodal:

$$p^{**,i}(c_t, z_t, U_t) = \zeta\, p\big(Y_t \mid h\big(s^i_t, \theta^i_t, z_t\big)\big)\, \mathcal{N}\big(c_t; A_c c^i_{t-1}, \Sigma_c\big) \times \delta\big(U_t - g\big(U^i_{t-1}, z^i_{t-1}\big)\big)\delta\big(z_t - f\big(z^i_{t-1}, U_t, c_t\big)\big),$$

where $\mathcal{N}(x; \mu, \Sigma)$ denotes the value of a Gaussian pdf with mean μ and variance Σ computed at the point x and ζ is a proportionality constant. Since the pdfs of U_t, z_t, conditioned on c_t, X_{t-1}, are Dirac delta functions, the above simplifies to

$$\begin{aligned} p^{**,i} &= \zeta\, p\big(Y_t \mid h\big(s^i_t, \theta^i_t, f\big(z^i_{t-1}, g^i, c_t\big)\big)\big)\, \mathcal{N}\big(c_t; A_c c^i_{t-1}, \Sigma_c\big) \\ &\quad \times \delta(U_t - g^i)\, \delta\big(z_t - f\big(z^i_{t-1}, g^i, c_t\big)\big) \\ &\triangleq p^{**,i}(c_t)\, \delta(U_t - g^i)\, \delta\big(z_t - f\big(z^i_{t-1}, g^i, c_t\big)\big), \end{aligned} \tag{31}$$

where $g^i \triangleq g(U^i_{t-1}, z^i_{t-1})$. The importance sampling part of $X_{t,r}$ is only c_t. We compute the importance density for c_t by approximating $p^{**,i}(c_t)$ by a Gaussian at its unique mode. The mode is computed by minimizing $L^i(c_t) = -\log p^{**,i}(c_t)$ defined below:

$$\begin{aligned} L^i(c_t) &= \big[-\log p\big(Y_t \mid h\big(s^i_t, \theta^i_t, f\big(z^i_{t-1}, g^i, c_t\big)\big)\big)\big] \\ &\quad + \big[-\log \mathcal{N}\big(c_t; A_c c^i_{t-1}, \Sigma_c\big)\big]. \end{aligned} \tag{32}$$

The PF-EIS algorithm for landmark shape tracking is summarized in Algorithm 3.

Algorithm 3. PF-EIS for Landmark Shape Filtering
Initialization: At time $t = 0$, sample $s^{(i)}_0 \sim \mathcal{N}(s_0, \sigma^2_s)$, $\theta^{(i)}_0 \sim \mathcal{N}(\theta_0, \sigma^2_\theta)$, $c^{(i)}_0 = \mathbf{0}$, $z^{(i)}_0 = z_0$ and $U^{(i)}_0 = U_0$. Here, $i = 1, \ldots, N_{pf}$ where, N_{pf} is the number of particles.
For $t > 0$,
1) Importance sample $s^i_t \sim \mathcal{N}(\alpha_s s^i_{t-1}, \sigma^2_s)$, $\theta^i_t \sim \mathcal{N}(\theta^i_{t-1}, \sigma^2_\theta)$. $i = 1, 2, \ldots, N_{pf}$
2) Compute $m^i_t = \arg\min_{c_t} L^i(c_t)$ and $\Sigma^i_{IS} = [\nabla^2 L^i(m^i_t)]^{-1}$ where L^i is defined in (32).
3) Importance sample $c^i_t \sim \mathcal{N}(m^i_t, \Sigma^i_{IS})$. Compute $U^i_t = g(U^i_{t-1}, z^i_{t-1})$ and $z^i_t = f(z^i_{t-1}, U^i_t, c^i_t)$.
4) Compute Importance weights as, $w^i_t = \frac{\tilde{w}^i_t}{\sum_{j=1}^{N_{pf}} \tilde{w}^j_t}$ where $\tilde{w}^i_t = w^i_{t-1} \frac{p(Y_t \mid h(s^i_t, \theta^i_t, z^i_t))\mathcal{N}(c^i_t; A_c c^i_{t-1}, \Sigma_c)}{\mathcal{N}(c^i_t; m^i_t, \Sigma^i_{IS})}$.
5) Compute the MMPSE estimate $\hat{z}_t$ as the principal eigenvector of $\sum_{i=1}^{N_{pf}} z^i_t z^{i*}_t w^i_t$. Resample.
6) Set $t \leftarrow t + 1$ and go to step 1.

4.5 Tracking to Automatically Extract Landmarks

In this section, we describe out technique to track and automatically extract landmark configurations over a sequence of images or a video. The system comprises of a optical flow (OF) tracker coupled with filtering. We compute optical flow at a cluster of points around each currently estimated landmark location $[\hat{S}_t]_k$ (k denotes the kth landmark) and use this to move the cluster of points into the next frame (frame $t + 1$). The centroid of the moved cluster serves as the new observation for the kth landmark at $t + 1$. The same thing is done for all landmark points to get Y_{t+1} (the vector of observed landmark locations at $t + 1$). This observation is fed into the NSSA-based PF which outputs the estimated landmark locations (and estimated shape) at $t + 1$. The entire procedure is summarized in Algorithm 4. For computing the optical flow, we used the code/method developed by [27].

Algorithm 4. Automatic Landmark Extraction over a Sequence of Images
Input: image(t-1), image(t), $\hat{S}_{t-1}$ (estimated landmark configuration at $t - 1$)
Output: $\{X^i_t, w^i_t\}, i = 1, 2, \ldots,$ $\hat{S}_t = \sum_{i=1}^{N_{pf}} (z^i_t e^{s^i_t + j\theta^i_t} + \tau^i_t) w^i_t$ (estimated landmark configuration at time t) where, z_t is the shape and e^{s_t}, θ_t, τ_t are the global scale, rotation and translation respectively.
1) For each estimated landmark $[\hat{S}_{t-1}]_k, k = 1, 2, \ldots K$, compute optical flow at a cluster of points around $[\hat{S}_{t-1}]_k$ and use this to move the points into image(t). Use the centroid of the moved cluster as the kth observed landmark at time t, $[Y_t]_k$. Do this for all landmark points to get Y_t
2) Run PF-Gordon using Y_t, i.e., implement steps 1 and 2 of Algorithm 2. But this time, we include the global translation in the state-space as well.
3) Display the estimated landmarks' location, $\hat{S}_t = \sum_{i=1}^{N_{pf}} (z^i_t e^{s^i_t + j\theta^i_t} + \tau^i_t) w^i_t$ (estimated landmark configuration at time t), set $t \leftarrow t + 1$, and go to step 1.

5 Experimental Results

We began by comparing the ability of NSSA to model real-life landmark shape sequences with that of SSA, ASM, and the wrong NSSA model (NSSA-unaligned) [1]. This is discussed in Section 5.1. It was observed that NSSA had much smaller modeling error than all three. This comparison gave us the first indication that NSSA would provide a much more accurate prior dynamic model for Bayesian filtering or tracking applications, as well as also for synthesis/extrapolation applications.

Next, we simulated multiple realizations of a "nonstationary shape activity" along with scale and rotation variations and attempted to track it using three different PF

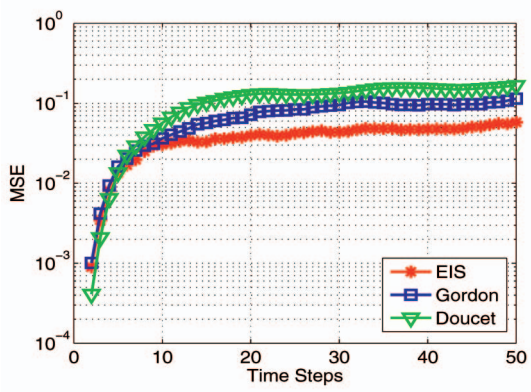

Fig. 3. Comparison of the MSE of estimated configurations computed using PF-EIS, PF-Doucet, and PF-Gordon for simulated shape sequence. EIS has the smallest MSE (discussed in Section 5.2).

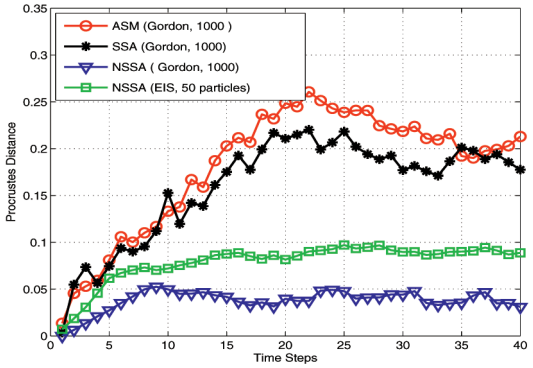

Fig. 4. Filtering noise and clutter corrupted MOCAP data: comparing NSSA-, SSA-, and ASM-based PF-Gordon and also NSSA-based PF-EIS. NSSA significantly outperforms SSA and ASM. NSSA-based PF-EIS with just 50 particles has comparable performance to that of 1,000 particle NSSA-based PF-Gordon (discussed in Section 5.3).

algorithms: the original PF (PF-Gordon) was compared with PF-Doucet [23] and PF-EIS [26] (described in Section 4.3). These comparisons are discussed in Section 5.2. It was observed that, when the number of available particles is small, PF-EIS has the best performance.

Since most existing work that used SSA or ASM for tracking used PF-Gordon, we retained this PF for most of our comparisons between NSSA, SSA, and ASM. The following four sets of experiments were done. In Section 5.3, we demonstrate the superior ability of NSSA-based PF (PF-Gordon with $N_{pf} = 1{,}000$ and PF-EIS with $N_{pf} = 50$) to filter out the landmark shapes from heavily noise-corrupted and cluttered observed landmark configurations. In Section 5.4, we compare the tracking ability of NSSA-based PF with SSA-based PF and ASM-based PF, i.e., their ability to accurately extract out landmarks from image sequences. Once again NSSA is found to be significantly superior to SSA and ASM and also to direct landmark extraction (without any model-based filtering). The use of 3D-NSSA to accurately synthesize new human activity sequences is discussed in Section 5.5. In Section 5.6, we give a preliminary experiment that demonstrates that NSSA is able to remain in track even when a model change occurs.

5.1 Modeling Error Comparison

We used the Carnegie Mellon Motion Capture (CMU Mocap) database [6] for our experiments. Each file in the database had the coordinates of the markers placed at the body landmark locations (especially the body joints) for successive frames of a specific human motion activity, e.g., running, jumping, etc. An example is shown in Fig. 6. The corresponding 2D and 3D landmark shape sequences were used as the training data for learning the NSSA (or SSA or ASM) model parameters.

We define modeling error (ME) as the trace of the noise covariance matrix of the AR modeling error, i.e., $\mathrm{ME} = trace(\Sigma_c)$, where $\Sigma_c = E[(c_t - A_c c_{t-1})(c_t - A_c c_{t-1})^T]$ and c_t are the "aligned" coefficients of shape velocity. We also compute the modeling error when the c_ts are not aligned, i.e., when U_t is computed using SVD at each t (as in [1]). When computing error for SSA, c_ts are the tangent space coefficients of shape (not of shape velocity), i.e., all shapes z_t are projected into a single tangent space at the common mean shape μ. For ASM, modeling error is still the same but now $c_t = z_t - \mu$, i.e., the shape space is assumed to be euclidean.

We computed the modeling error of SSA, ASM, and NSSA (unaligned) for various human actions and compared with that of NSSA. It was found that NSSA has much lower modeling error than all three. The modeling error bar plots of 2D and 3D shape sequences have been shown in Figs. 2a and 2b for a few motion activities.

Modeling error tells us how well we are able to capture the shape deformation dynamics using the given model. It thus quantifies how well we can predict the shape at a time instant given the information from the previous time instant. Thus, lower modeling error will result in better tracking ability and also a better ability to synthesize a new sequence or to extrapolate an existing sequence (graphics problems).

5.2 Comparing PF Algorithms

We simulated landmark shape change of a set of $K = 5$ landmarks (a deforming pentagon) and tracked it using PF-EIS (Algorithm 3), PF-Gordon (Algorithm 2) [11], and PF-Doucet [23] with $N_{pf} = 50$ particles. The initial shape z_0 was a regular pentagon. The shape and global motion change of the configuration followed (10), (26) with $\Sigma_c = 0.0025 I_6$, $\sigma_s^2 = 0.0001$, $\sigma_\theta^2 = 0.25$, $A_c = 0.6 I_6$, and $\alpha_s = 0.9$. The observations followed (28) with $\sigma_o^2 = 0.04$ and $p = 0.2$. I_6 denotes a 6×6 identity matrix. It is to be noted that the STP of scale (e^{s_t}) is a log-normal, and hence, even $\sigma_s^2 = 0.0001$ results in a fairly broad distribution.

Whenever one or more landmarks are generated by clutter, the observation likelihood (OL) of log-scale (s_t) is either heavy-tailed with the wrong (outlier) mode or is multimodal. When many landmarks are generated by clutter, the same happens for θ_t. This combined with a broad prior of s_t, θ_t results in multimodal $p^*(s_t, \theta_t)$. Whenever this happens, most particles of PF-Doucet end up sampling from a Gaussian about the wrong mode of $p^*(s_t, \theta_t)$ or $p^*(s_t)$, resulting in loss of track. But, PF-EIS does not suffer from this problem since it samples from the prior of s_t, θ_t. Also, since the prior of c_t is narrow compared to the distance between likelihood modes, $p^{**,i}(c_t)$ is usually unimodal, and thus, sampling from its Laplace's

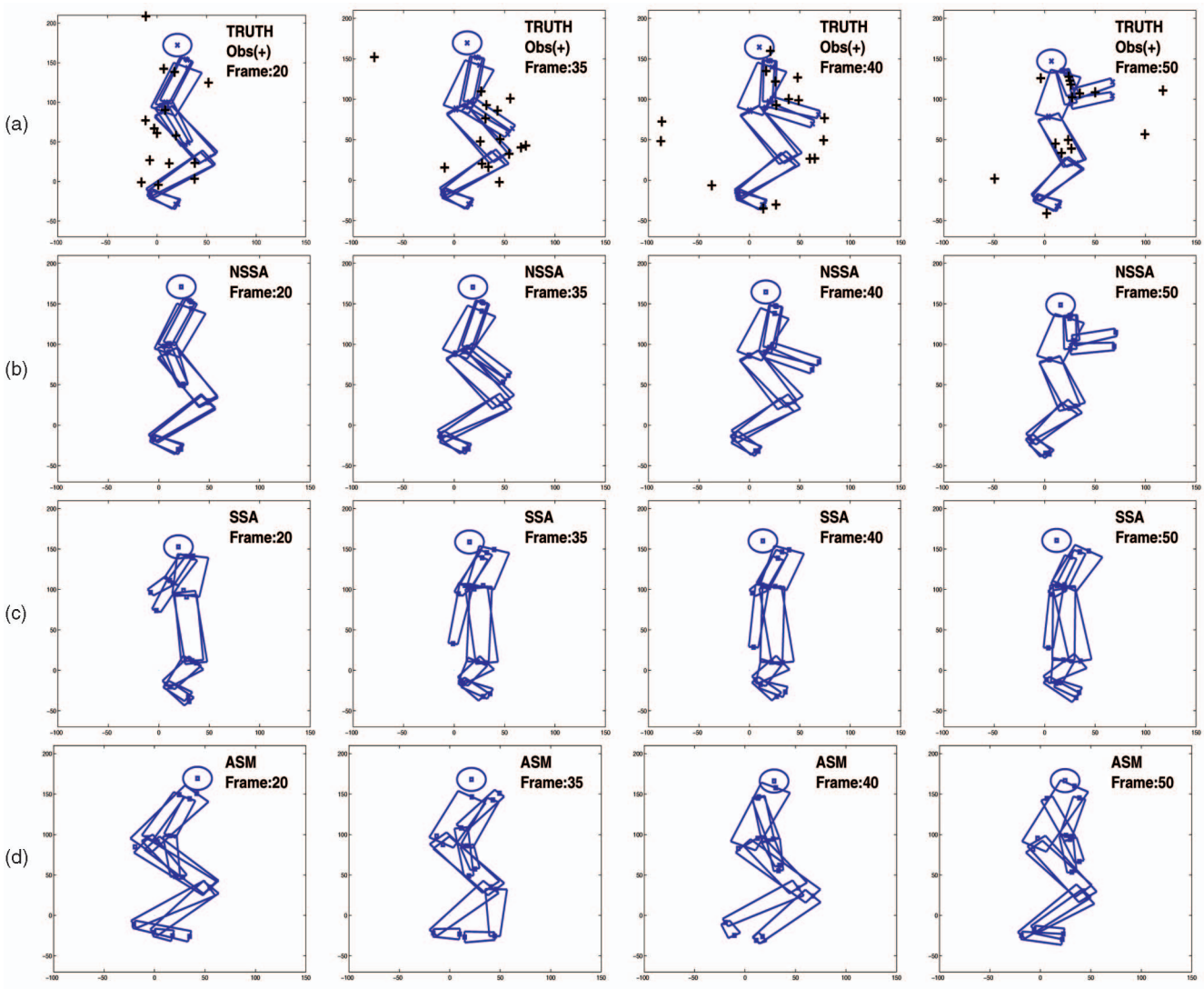

Fig. 5. Filtering out true shape data from noisy/cluttered observations using PF-Gordon (motion activity: *jump*). The landmark points (denoted by $\times$ for ground truth and $\square$ for the filter output) are fitted with rectangular patches to visualize the body posture at a given time instant. The tracking performances of NSSA, SSA, and ASM are shown over four frames. (a) Ground truth with observations ($+$), (b) tracked with NSSA, (c) tracked with SSA, and (d) tracked with ASM. It can be clearly seen that NSSA way outperforms SSA and ASM.

approximation is valid. On the other hand, for small N_{pf}, PF-Gordon often loses track because it samples all states from the prior, thus resulting in small effective particle size. In Fig. 3, the mean squared error (MSE) plot averaged over 80 Monte Carlo runs is shown. Also, see Fig. 4 for MOCAP data, where PF-EIS with $N_{pf} = 50$ particles is able to achieve almost the same tracking accuracy as PF-Gordon with $N_{pf} = 1{,}000$.

5.3 Filtering from Noisy and Cluttered Observations

In this experiment, 2D landmark shape sequences from CMU Mocap database (refer Section 5.1) were used as the ground truth (as shown in Fig. 5a). We incorporated random scale variations, additive noise, and clutter to the ground truth. The observations were simulated using (28). This experiment helped us to quantitatively compare performance since ground truth was available.

We used the PF-Gordon (i.e., the basic PF) with $N_{pf} = 1{,}000$ particles for filtering. Our primary objective was to compare the performance of NSSA-based system model with that of SSA and ASM. PF-Gordon solely depends on the STP for importance sampling, and thus, its performance heavily relies on the accuracy of the system model. Also, all past works on SSA- or ASM-based tracking used PF-Gordon.

We considered two motion activities, namely, *run* and *jump*, with $K = 16$ landmarks. Different data sequences were used for training and testing. Our results are shown in Appendix I (run), which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPAMI.2009.94, and Fig. 5 (jump), and the Procrustes distance comparison plots are given in Fig. 4. The landmark locations were fitted with rectangular patches to visualize the body posture at a given instant. The first row shows the ground truth and also the observation. It can be seen that the observed landmark locations (represented as $+$ on top of the ground truth) were severely corrupted with clutter and random noise. Thus, visually, the observed configurations hardly conform to the human body. But, as shown in Fig. 5b, NSSA has been able to filter out the true shape from those observations with very good accuracy. SSA (Fig. 5c) and ASM

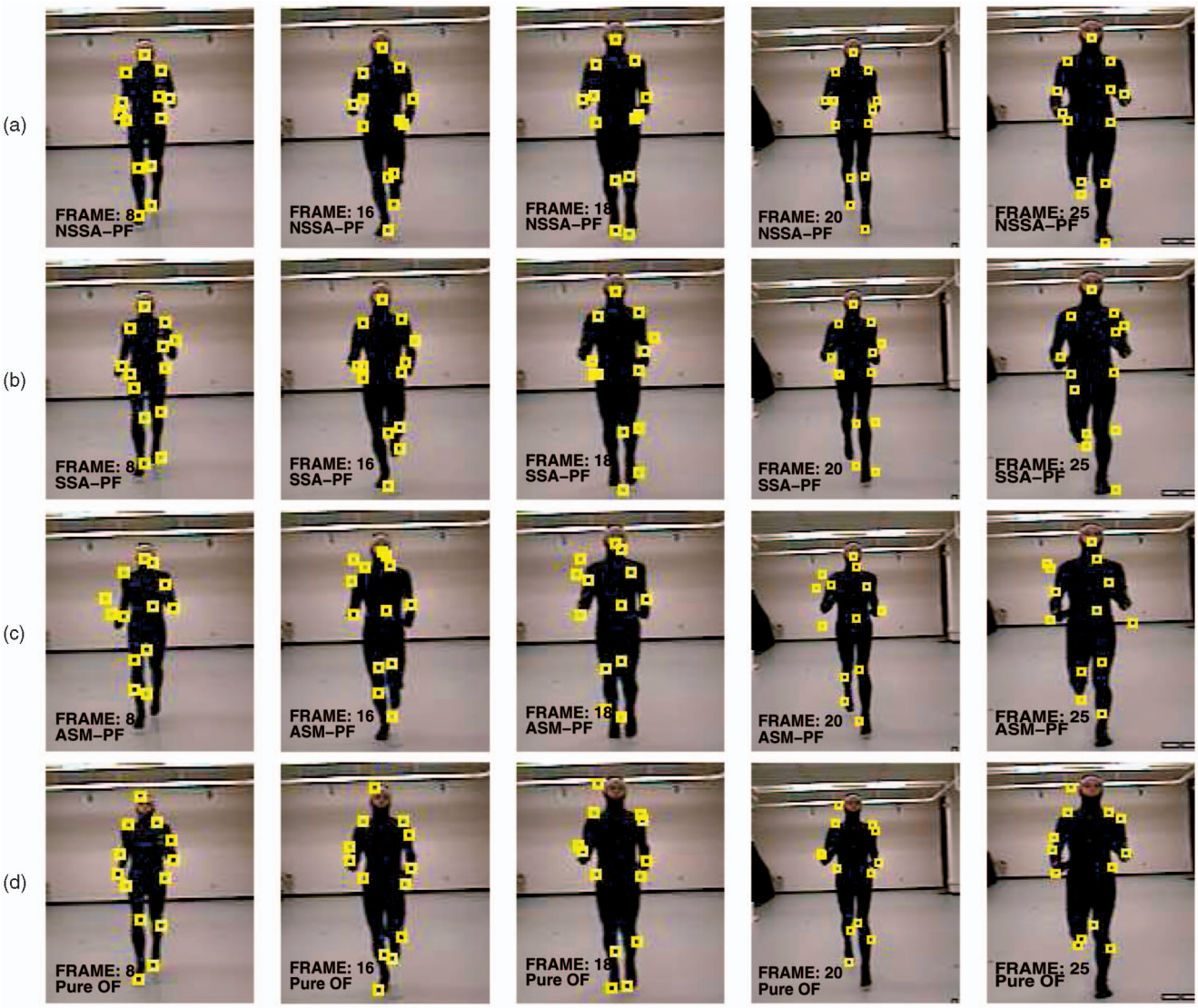

Fig. 6. Tracking landmark configurations over the video of a running sequence (discussed in Section 4.5). (a) NSSA, (b) SSA, and (c) ASM. It can be clearly seen that NSSA way outperforms SSA and ASM. (d) Landmark observations extracted using purely optical-flow-based approach. It can be seen that the observed landmark loses posture information gradually over time. Such observation leads to poor filtering/tracking performance of the system.

(Fig. 5d), however, perform poorly in this job. Similar results were found for motion activity *run* (see Appendix I, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPAMI.2009.94). In Fig. 4, we plot the Procrustes distance between the estimated shape and the true shape at each instant as the quantitative measure of filtering accuracy. Also, note that PF-EIS (with NSSA model) is able to achieve almost the same accuracy as PF-Gordon (with NSSA model) with as few as $N_{pf} = 50$ particles.

In case of SSA and ASM, filtering performed around the fixed mean shape ends up generating shape samples far away from the desired shape space where the original set of deforming shapes lies. This, in turn, led to their poor performances. But NSSA, on the other hand, does an excellent job in filtering because of its time varying mean shape assumption. Of course, we assume that the shape deviations over successive time frames are small enough to make sure the current shape is in the neighborhood of the current mean shape (i.e., the previous shape) for our mathematical treatments to be valid. This is a reasonable assumption for most of the real-life motion activities.

5.4 Tracking and Automatic Landmark Extraction

For automatic landmark extraction and tracking, we used 50 frames of real-life videos of human activities (as shown in Fig. 6 (run) and Fig. 7 (jump)). A total of 13 body landmarks were considered as shown in Fig. 7. The body landmarks were: right shoulder, right elbow, right hand, right hip, right knee, right foot, head, left shoulder, left elbow, left hand, left hip, left knee, and left foot.

For run sequences, the system model was learned from the hand-marked ground truth data corresponding to the training database. In case of jump, however, we used the mocap data itself. We appropriately subsampled the mocap data frames in order to roughly synchronize them to the video frames. Implementations of Algorithm 4 using system models based on NSSA, SSA, and ASM were compared. Their performances over multiple time frames are shown in

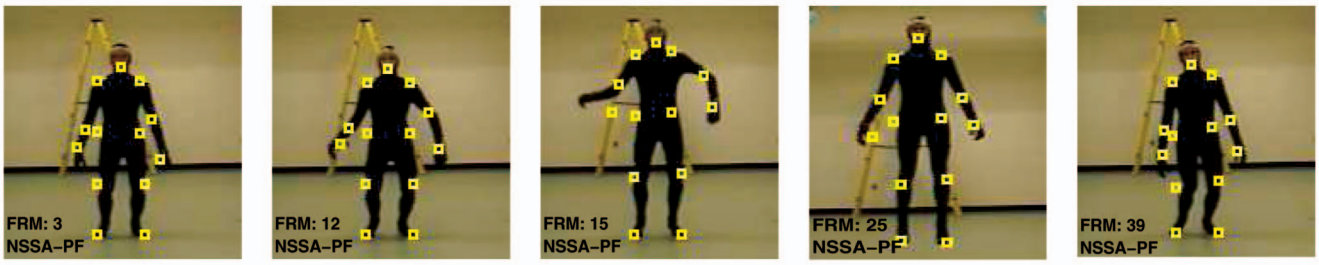

Fig. 7. Tracking landmark configurations over the video of a jump sequence with NSSA-based system model (discussed in Section 4.5). It can be seen that at frame 15, NSSA lost track of one of the landmarks (right hand). But after that, it regains track of the landmark.

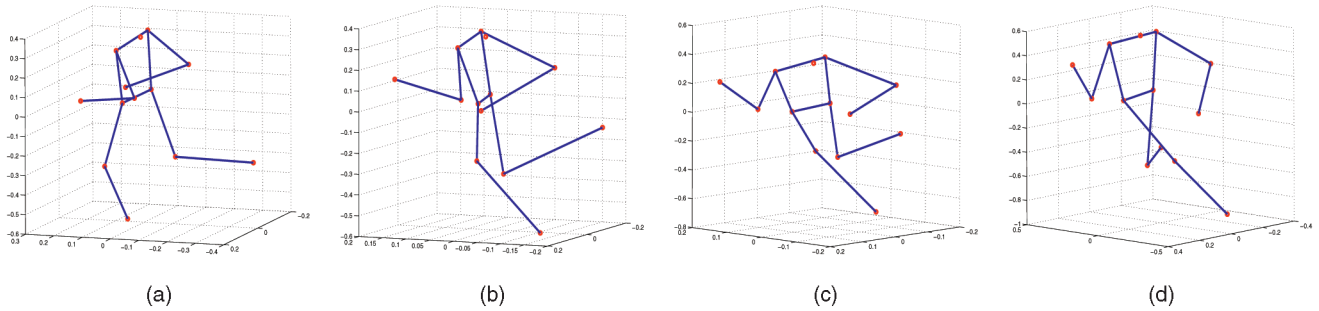

Fig. 8. Synthesis of a 3D shape sequence corresponding to motion activity *run*. Four frames are shown. The system model was learned using 3D-NSSA on 3D landmark shape sequences for running. The sequence shown above visually resembles a running sequence.

Fig. 6 (run). It can be clearly seen that NSSA (Fig. 6a) performs much better than SSA (Fig. 6b) or ASM (Fig. 6c) in terms of *keeping track* of the body landmarks. It is very important to note that filtering and prediction play a very crucial role in the process of extracting landmark observations. To verify this, we extracted the landmark observations purely based on optical flow, i.e., starting with the initial locations, we used the optical flow between successive frames to drift the points, and thus, getting observations over time. This procedure does not use the knowledge of the state estimates at each instant to decide where the expected landmark might be while computing the optical flow. As can be seen from Fig. 6d, quite a few of the observed landmarks in this case were found to get stuck at wrong locations (due to background interference, occlusion, or clutter) and from that point on they were never in sync with the body movements. In fact, the point of using the state estimates is to correct the locations of such landmarks and making sure that we do not end up computing the OF at the wrong regions for getting the landmark location for the next frame.

Next, we test the system with NSSA-based system model on the video of a person *jumping*. It can be seen in Fig. 7 that NSSA did a very good job in terms of tracking the landmarks over various time frames. It is to be noticed that, at frame 15, it loses track of the landmark corresponding to the right hand. But later, it regains the track (see frame 39, Fig. 7).

5.5 3D-NSSA Synthesis

Motivated by the drastically small modeling error of 3D-NSSA for human actions (see Fig. 2b), we attempted to use 3D NSSA for a motion synthesis application. We learned the deformation model for 3D body shape of the human body while running from MOCAP data. The synthesized run sequence using this model is shown in Fig. 8. Since there is no ground truth in this case, we visually verified the systems ability to synthesize *run* and the system performance was found to be promising.

5.6 Change Detection with NSSA

We did a simple experiment to test the ability of the NSSA-based tracker to detect changes in activity while still not completely losing track. We used a sequence where a person begins by running, and then, leaps (http://mocap.cs.cmu.edu:8080/subjects/49/49_04.avi). Notice that this is a fairly sudden change. The ELL-based change detection statistic [28] was able to detect the change to leap and, for some time after the change also, our tracker did not lose track (see Fig. 9—tracking error does not increase until later). More results and comparison with SSA are shown in [1]. Detailed results and all MATLAB codes can be found at http://www.public.iastate.edu/~samarjit/pami.html.

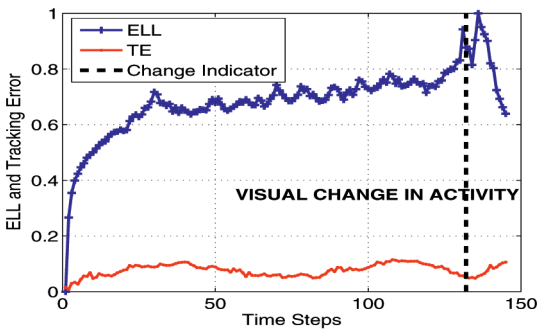

Fig. 9. The ELL and tracking error (TE) plot for the shape sequence with *run* followed by *leap*. PF-Gordon was used with NSSA-based system model. The actual activity transition occurred at $t = 132$. It can be clearly seen that ELL detected the change. There is distinct spike of ELL around $t = 132$. However, the tracker still keeps tracking even after the change has occurred.

5.7 Shortcomings of NSSA: Classification

Despite very good performances while tracking/filtering, in its current form, NSSA does not perform as well for classification. We tried to perform a model-based maximum-likelihood classification among various motion activities (e.g., run, jump, sit, etc.). The input to the classifier was the time sequence of shape velocity coefficients for NSSA, tangent space coefficients for SSA, and shape deviation vectors for ASM. The output was the most likely activity to have generated the sequence. In our preliminary experiments with run, sit, jump, and dance activity sequences, NSSA had a 4 percent misclassification rate, while SSA and ASM had 2.5 and 2 percent, respectively. The reason NSSA does not perform as well as the rest is the same as the reason it significantly outperforms SSA and ASM for tracking—it is a more generic model for shape change. It consists of a zero mean random walk model on shape and a zero mean AR model on shape velocities. The effect of initial shape is lost in a long sequence.

To use NSSA for classification, we should modify the current model and define a nonzero-mean shape velocity change model. Alternatively, a good idea would be to use NSSA for tracking, i.e., for extracting landmark shape sequences from video, and then, feeding these into a piecewise SSA [14] or piecewise ASM [13] based classifier.

6 Conclusions and Future Work

The key contribution of this work is a novel approach to define a generative model for both 2D and 3D nonstationary landmark shape sequences, which we call NSSA. The main idea is to compute the tangent space representation of the current shape in the tangent space at the previous shape. This can be referred to as the shape velocity vector since it quantifies the difference between two consecutive shapes projected into the tangent space at the first one. The "aligned" shape velocity coefficients (shape speed vector) are modeled using vector time-series methods.

Applications in filtering, tracking, synthesis (using 3D-NSSA models), and change detection are demonstrated. Filtering and tracking are studied in detail and significantly improved performance over related work is demonstrated (see Figs. 2, 3, 4, 5, 6, 7, 8, and 9). With the exception of smoothing splines [29], [12], most other existing work does not model nonstationary shape sequences. In other work [30], we have also successfully demonstrated the use of the NSSA for model-based compression of landmark shape sequence data.

Future work includes developing the 3D synthesis, change detection, and classification applications (see Sections 5.5, 5.6, and 5.7 for details). Another possible future research direction is combining our models with GPVLM-based observation extraction techniques [17], [31]. Also, our optical-flow-based landmark extractor could be improved by using ideas from [32].

Acknowledgments

Part of this paper appeared in [1], [2]. This work was partially funded by US National Science Foundation (NSF) grant ECCS 0725849.

References

[1] N. Vaswani and R. Chellappa, "Nonstationary Shape Activities," *Proc. IEEE Conf. Decision and Control,* Dec. 2005.

[2] N. Vaswani and S. Das, "Particle Filter with Efficient Importance Sampling and Mode Tracking (PF-EIS-MT) and Its Application to Landmark Shape Tracking," *Proc. Asilomar Conf. Signals, Systems and Computers,* 2007.

[3] D. Kendall, D. Barden, T. Carne, and H. Le, *Shape and Shape Theory.* John Wiley and Sons, 1999.

[4] N. Vaswani, A. RoyChowdhury, and R. Chellappa, "'Shape Activity': A Continuous State HMM for Moving/Deforming Shapes with Application to Abnormal Activity Detection," *IEEE Trans. Image Processing,* vol. 14, no. 10, pp. 1603-1616, Oct. 2005.

[5] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision.* Cambridge Univ. Press, 2000.

[6] "CMU Motion Capture Data," http://mocap.cs.cmu.edu/, 2009.

[7] I. Dryden and K. Mardia, *Statistical Shape Analysis.* John Wiley and Sons, 1998.

[8] T. Cootes, C. Taylor, D. Cooper, and J. Graham, "Active Shape Models: Their Training and Application," *Computer Vision and Image Understanding,* vol. 61, pp. 38-59, Jan. 1995.

[9] A. Veeraraghavan, A. RoyChowdhury, and R. Chellappa, "Matching Shape Sequences in Video with an Application to Human Movement Analysis," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 27, no. 12, pp. 1896-1909, Dec. 2005.

[10] M. Isard and A. Blake, "Condensation: Conditional Density Propagation for Visual Tracking," *Int'l J. Computer Vision,* vol. 29, pp. 5-28, 1998.

[11] N.J. Gordon, D.J. Salmond, and A.F.M. Smith, "Novel Approach to Nonlinear/Nongaussian Bayesian State Estimation," *IEE Proc.-F (Radar and Signal Processing),* vol. 140, no. 2, pp. 107-113, 1993.

[12] A. Kume, I. Dryden, and H. Le, "Shape Space Smoothing Splines for Planar Landmark Data," *Biometrika,* vol. 94, pp. 513-528, 2007.

[13] N. Paragios, M. Jolly, M. Taron, and R. Ramaraj, "Active Shape Models and Segmentation of the Left Ventricle in Echocardiography," *Proc. Fifth Int'l Conf. Scale Space and PDE Methods in Computer Vision,* Apr. 2005.

[14] B. Song, N. Vaswani, and A.K. Roy-Chowdhury, "Closedloop Tracking and Change Detection in Multi-Activity Sequences," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2007.

[15] T.F. Cootes, G. Edwards, and C.J. Taylor, "Active Appearance Models," *Proc. European Conf. Computer Vision,* vol. 2, pp. 484-498, 1998.

[16] B.P. Lelieveldt, R.J. van der Geest, J.H.C. Reiber, J.G. Bosch, S.C. Mitchel, and M. Sonka, "Time-Continuous Segmentation of Cardiac Image Sequences Using Active Appearance Motion Models," *Proc. Int'l Conf. Information Processing in Medical Imaging,* pp. 446-452, Jan. 2001.

[17] T. Tai-Peng, L. Rui, and S. Sclaroff, "Tracking Human Body on a Learned Smooth Space," Technical Report No. 2005-029, Boston Univ. Computer Science, 2005.

[18] S. Hou, A. Gatala, F. Caillette, N. Thacker, and P. Bromiley, "Real-Time Body Tracking Using a Gaussian Process Latent Variable Model," *Proc. IEEE Int'l Conf. Computer Vision,* Oct. 2007.

[19] A. Srivastava and E. Klassen, "Bayesian and Geometric Subspace Tracking," *Advances in Applied Probability,* vol. 36, pp. 43-56, Mar. 2004.

[20] S.X. Ju, M.J. Black, and Y. Yacoob, "Cardboard People: A Parameterized Model of Articulated Image Motion," *Proc. Second Int'l Conf. Automatic Face and Gesture Recognition,* 1996.

[21] J. Kent, "The Complex Bingham Distribution and Shape Analysis," *J. Royal Statistical Soc., Series B,* vol. 56, pp. 285-299, 1994.

[22] J. Shi and C. Tomasi, "Good Features to Track," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 593-600, 1994.

[23] A. Doucet, "On Sequential Monte Carlo Sampling Methods for Bayesian Filtering," Technical Report CUED/F INFENG/TR. 310, Dept. of Eng., Cambridge Univ., 1998.

[24] L. Tierney and J.B. Kadane, "Accurate Approximations for Posterior Moments and Marginal Densities," *J. Am. Statistical Assoc.,* vol. 81, no. 393, pp. 82-86, Mar. 1986.

[25] S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A Tutorial on Particle Filters for On-Line Non-Linear/Non-Gaussian Bayesian Tracking," *IEEE Trans. Signal Processing,* vol. 50, no. 2, pp. 174-188, Feb. 2002.

[26] N. Vaswani, "Particle Filtering for Large Dimensional State Spaces with Multimodal Observation Likelihoods," *IEEE Trans. Signal Processing,* vol. 56, no. 10, pp 4583-4597, Oct. 2008.
[27] S. Khan, "Matlab Code for Optical Flow Using Lucas Kanade Method," www.cs.ucf.edu/khan/, 2008.
[28] N. Vaswani, "Additive Change Detection in Nonlinear Systems with Unknown Change Parameters," *IEEE Trans. Signal Processing,* vol. 55, no. 3, pp. 859-872, Mar. 2007.
[29] A. Kume, I. Dryden, H.L. Le, and A. Wood, "Fitting Cubic Splines to Data in Shape Spaces of Planar Configurations," *Proc. Statistics of Large Datasets,* 2002.
[30] S. Das and N. Vaswani, "Model-Based Compression of Nonstationary Landmark Shape Sequences," *Proc. IEEE Int'l Conf. Image Processing,* 2008.
[31] N. Lawrence, "Probabilistic Non-Linear Principal Component Analysis with Gaussian Process Latent Variable Models," *J. Machine Learning Research,* vol. 6, pp. 1783-1816, Nov. 2005.
[32] A. Srivastava and I.H. Jermyn, "Looking for Shapes in Two-Dimensional Cluttered Point Clouds," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 31, no. 9, pp. 1616-1629, Sept. 2009.

Samarjit Das received the BTech degree in electronics and communications engineering from the Indian Institute of Technology (IIT), Guwahati, in May 2006. He is currently working toward the doctoral degree in the Department of Electrical and Computer Engineering at Iowa State University. His research interests are in computer vision and image processing, statistical signal processing, and machine learning. He is a student member of the IEEE.

Namrata Vaswani received the BTech degree in electrical engineering from the Indian Institute of Technology (IIT), Delhi, in August 1999, and the PhD degree in electrical and computer engineering from the University of Maryland, College Park, in August 2004. Her PhD thesis was on change detection in stochastic shape dynamical models and applications to activity modeling and abnormal activity detection. She was a postdoctoral fellow and research scientist at the Georgia Institute of Technology during 2004-2005. She is currently an assistant professor in the Department of Electrical and Computer Engineering at Iowa State University. Her research interests are in estimation and detection problems in sequential signal processing, biomedical image sequence analysis, and computer vision. Her current research focus is on sequential compressed sensing and large-dimensional particle filtering. She is a member of the IEEE.