Dynamic Textures Synthesis as Nonlinear Manifold Learning and Traversing

Che-Bin Liu[1,2] Ruei-Sung Lin[1,3] Narendra Ahuja[1] Ming-Hsuan Yang[4]

[1]University of Illinois at Urbana-Champaign, Urbana, IL 61801, USA

[2]Epson R&D, Inc., Palo Alto, CA 94304, USA

[3]Motorola Labs, Schaumburg, IL 60196, USA

[4]Honda Research Institute, Mountain View, CA 94041, USA

http://vision.ai.uiuc.edu/~cbliu/

Abstract

We formulate the problem of dynamic texture synthesis as a nonlinear manifold learning and traversing problem. We characterize dynamic textures as the temporal changes in spectral parameters of image sequences. For continuous changes of such parameters, it is commonly assumed that all these parameters lie on or close to a low-dimensional manifold embedded in the original configuration space. For complex dynamic data, the manifolds are usually nonlinear and we propose to use a mixture of linear subspaces to model a nonlinear manifold. These locally linear subspaces are further aligned within a global coordinate system. With the nonlinear manifold being globally parameterized, we overcome motion discontinuity problems encountered in switching linear models and dynamics. We present a nonparametric method to describe the complex dynamics of data sequences on the manifold. We also apply such approach to dynamic spatial parameters such as motion capture data. The experimental results suggest that our approach is able to synthesize smooth, complex dynamic textures and human motions, and has potential applications to other dynamic data synthesis problems.

1 Introduction

In this paper, we focus on complex temporal changes in spectral or spatial parameters of objects or image sequences. There is a broad interest in modeling parameter changes of different dynamic data. For example, dynamic texture analysis concerns temporal changes in pixel intensities in an image sequence, and human motion analysis concerns temporal changes of the configurations of human body parts. By complex dynamic data, we refer to model parameters that have multi-modal distributions, which can be often seen in the real world. For instances, a flapping flag may exhibit various distinguishable shapes with changes in wind, and a person may hop, skip, jump, tiptoe, leap, etc. in a dancing sequence. In general, we are interested in developing an algorithm to model and synthesize complex dynamic data.

In dynamic textures [13], also called temporal textures [14] or video textures [2], the existing generative models have been generally limited to using a linear dynamical system (LDS). Due to the large dimensionality of images, learning of dynamic textures

BMVC 2006 doi:10.5244/C.20.88

is usually accomplished in two stages: principal component analysis (PCA) and autoregressive (AR) process, that is, finding observation matrix with noise model by PCA and estimating system matrix with noise model of an AR model. These methods are able to generate reasonable results for temporally stationary dynamic textures. Yuan et al. [20] analyze the stability of the LDS and conclude that the LDS based methods produce good synthesis results only for an oscillatory system. Although they improve results by employing a non-causal feedback-based LDS, the use of a single PCA model has its limitations in modeling temporally non-stationary dynamic textures which are common in the real world. Non-stationary dynamic textures contain different data modalities in their appearance distribution, which can not be captured by a linear dimensionality reduction scheme such as PCA. In the standard dynamic texture database [14], for instance, flapping flags in the presence of wind exhibit large variance in shapes and appearances, and therefore a more sophisticated appearance model is required for acceptable modeling and synthesis.

Note that there are existing works focusing on creating novel dynamic data without understanding underlying generative processes of the data. In the case of videos, synthesized results by such approaches [6, 11] usually have excellent image quality since they reuse original images, in the expense of storing entire input data. In this paper, while we choose to learn the intrinsic low-dimensional parameters of input data, we significantly improve synthesized image quality of existing approaches along similar directions.

2 Related Work

Although using nonlinear component analysis is an intuitive extension of PCA for manifold representation, modeling dynamics in nonlinear subspace is not straightforward. There exist several methods for nonlinear dimensionality reduction. Methods of global nonlinear projection (e.g. LLE [9], Isomap [16]) aim at preserving spatial relationships among given samples, and map them into a global coordinate system of the intrinsic low-dimensional subspace. These methods typically find the low-dimensional embeddings from the observed data in the high-dimensional input space, and the mappings are not reversible. Hence, even if a dynamics model can be learned and simulated in the low-dimensional subspace, these algorithms cannot be applied to infer data in the original space based on the low-dimensional representations. Methods of locally linear projection (e.g. [3, 17]) characterize a nonlinear manifold by fitting multiple locally linear models. Their mappings preserve information in the original space, so data can be reconstructed given low-dimensional representations. However, each local model has its own mapping between data and its corresponding linear subspace, resulting in different coordinate systems. For the lack of a single and coherent coordinate system, these methods are not appropriate for continuous video or motion synthesis.

A general switching linear dynamical system (SLDS) [4] contains multiple linear systems and has a transition matrix indicating the likelihood of switching from one LDS to another. There is no continuity constraint in the observed variable, which is a main problem for continuous dynamic data synthesis. It is also difficult to learn a SLDS for high-dimensional image data. Li et al. [8] modify SLDS by setting end constraints for each LDS to ensure smooth transitions between local models for human motion capture data. These end constraints represent transition points that connect different linear subspaces. Synthesized motions have to go through these pre-selected transition points (i.e.

fixed key frames) in order to correctly convert coordinate systems between subspaces, which limits its descriptive capability for motion transitions.

Most recently, Gaussian process latent variable models (GPLVM) [7] and its extensions have been applied to solve inverse kinematics in pose estimation [5] and human tracking [18]. Notwithstanding the demonstrated success in these tasks, these methods utilize gradient-based optimization techniques and therefore rely on smoothness constraints for good performance. That is, GPLVM-based methods tend to oversmooth the discontinuities in the latent space where non-stationary temporal or spatial changes occur, and in turn the synthesized results may be less satisfactory. Another side effect is that the kernel matrix grows as number of training samples increases, although this problem is alleviated by the use of greedy approximation algorithms. An extension to Gaussian process dynamical models (GPDM), however, has even more difficulty to deal with large data sets [19].

Algorithms such as GPLVM, GPDM, SLDS, variational methods and our method are guaranteed to reach only local optimum as they all use gradient decent or EM algorithms to learn complex data sets. However, our model can be learned more efficiently as we adopt a two-stage learning approach using [17] and [15]. On the other hand, other methods, such as GPLVM using a kernel function to model nonlinearity, have to solve highly complicated optimization problems. Also, methods like GPDM have to save all original input data, while our algorithm saves only low-dimensional representations of input data.

3 Our Approach

Our approach is conceptually illustrated in Figure 1. In the learning stage, we learn a mixture of PCA models that best represents the entire input data sequence. These PCA subspaces are then aligned into a global subspace within a maximum likelihood framework. The input data are projected onto this global subspace to form a continuous trajectory, and their projected coefficients are stored, not raw data. In the synthesis stage, we pick an initial point, usually a projection of given data point, in the global subspace. We synthesize data sequences by traversing in the global subspace, according to the local dynamics of projected input data. For each point g in the subspace, we find the most probable PCA model associated with the point, and compute its mapping z in the selected PCA subspace. This PCA model then generates a data point using the mapping z.

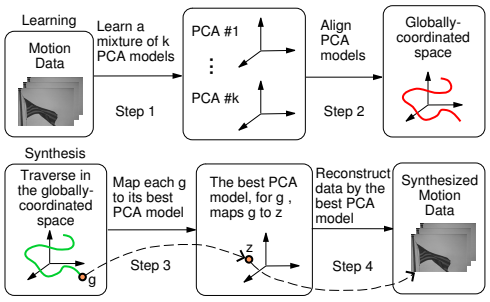


Figure 1: Overview of our approach to learning and synthesis of a given motion sequence.

3.1 Globally-Coordinated Nonlinear Manifold Representation

We characterize the nonlinear manifold by a mixture of linear subspaces. Earlier, we have proposed an algorithm that learns a globally-coordinated nonlinear manifold from a temporal data sequence, and apply it to tracking applications. In that paper, we have shown superior manifold learning results to those without exploiting temporal relationships between data points. However, for the synthesis problem we are solving here, we find that the subspace alignment algorithm proposed by Teh and Roweis [15] gives satisfactory results as well. Due to the anonymous review policy and the limited space allowed for this paper, we will use the subspace alignment algorithm in the following.

First, we use a mixture of probabilistic PCA model (MPPCA) [17] to capture complex multi-modal data. MPPCA is formulated within a maximum likelihood framework and its parameters can be estimated by an EM algorithm. Although MPPCA is guaranteed to reach only local optimum, empirically, in our dynamic texture experiments, this learning algorithm converges to a good estimate of model parameters within 15 iterations with coarse data clustering. Since each local linear model has an independent coordinate system, we globally parameterize all linear models by incorporating some topological constraints to align them into a single coordinate system. These topological constraints are similar to LLE [9]: preserving the same neighborhood structure between the high-dimensional input space and the low-dimensional embedding.

For each high-dimensional data point y_n, we denote its nearest neighbors as y_m ($m \in \mathcal{N}_n$) and minimize

$$\mathcal{E}(Y,W) = \sum_n \| y_n - \sum_{m \in \mathcal{N}_n} w_{nm} y_m \|^2 \tag{1}$$

with respect to W subject to $\sum_{m \in \mathcal{N}_n} w_{nm} = 1$. The weights w_{nm} are unique and can be estimated by constrained least squares. These weights represent the locally linear relationships between y_n and its neighbors. Accordingly, we define the same cost function in the low-dimensional global subspace

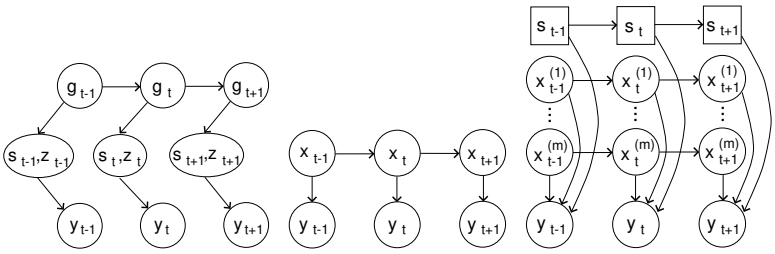
$$\mathcal{E}(G,W) = \sum_n \| g_n - \sum_{m \in \mathcal{N}_n} w_{nm} g_m \|^2 \tag{2}$$

with respect to G, where g_n is the intrinsic parameter of y_n in the global coordinate system. By minimizing $\mathcal{E}(G,W)$, with additional translation constraint $\frac{1}{N}\sum_n g_n = 0$ and rotation and scale constraint $\frac{1}{N}\sum_n g_n g_n^T = I$, the mappings between local linear models and the global coordinate system can be optimally determined without local minima problems by solving a generalized eigenvalue system [15].

Alternative algorithms for nonlinear mapping exist, for instance, the global coordination method [10] and the manifold charting algorithm [1]. However, it is very difficult to obtain satisfactory mappings and model parameters using [10] as a variational approximation method is employed which suffers from serious local minima problems in practice. Further, it requires good initialization to obtain good results. On the other hand, the algorithm of manifold charting [1] is not ideal in the sense that the mapping between data and its globally-coordinated subspace is pseudo-invertible and may suffer from numerical instability. Consequently, we adopt a two-stage learning strategy described above.

3.2 Nonparametric Dynamic Model

Once we learn the global representations for the input data, we derive a dynamical model as shown in Figure 2(a). Compared to the LDS (Figure 2(b)), we replace the hidden state variable x with global coordinate g, and the inference from g to y is nonlinear through the mixture model $\{s,z\}$. Contrasted to a general SLDS (Figure 2(c)), where it defines a transition probability $p(s_t|s_{t-1})$ to select the local model, our method implicitly switches models depending on state g so the synthesized data is continuous. Different from the constrained SLDS [8], where it switches model only at transition points (pre-determined states) to avoid random switches like SLDS, our state variable g has no such constraints.



(a) Our model (b) LDS (c) SLDS

Figure 2: The proposed dynamic model, LDS and SLDS.

The essential idea of our nonparametric dynamic model is to traverse along the learned trajectory in the global subspace without drifting far away from it. Therefore, we sample and learn motions captured in the input data sequence with *spatial locality* and *temporal similarity* constraints. Figure 3 helps to visualize our motion prediction process. We denote the projections of the input data sequence as g with subscripts indicating temporal indices. We also denote the current position in the globally-coordinated subspace as x_t, conventionally representing the current state in dynamic models.

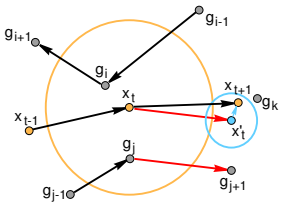


Figure 3: An illustration of our nonparametric dynamics in the nonlinear manifold.

To advance x_t to x_{t+1}, we first find the nearest neighbors of x_t among $\{g\}$. We prefer using motions of these neighbors (spatial locality) while encouraging temporal smoothness (temporal similarity). In Figure 3, although g_i is closer to x_t, we prefer using the motion from g_j to g_{j+1} because the motion used to advance g_{j-1} is more similar to that of x_{t-1}. In other words, our dynamic model takes the temporal and spatial similarity into account. After advancing to x'_t, we perturb it with noise for motion variations. However,

Algorithm 1 The motion synthesis algorithm for globally-coordinated subspace.

1. *Sampling neighbors*: At x_t, find its K nearest neighbors $\{g_i\}$, $i \in \mathcal{N}_t$, with weights $W_i^{(1)} \sim \mathcal{N}(x_t, \sigma_h^2)$.
2. *Temporal smoothness*: Compute the motion similarity between x_t and each g_i:
$$\cos(\theta_i) = \frac{\langle dx_t, dg_i \rangle}{\sqrt{\langle dx_t, dx_t \rangle \langle dg_i, dg_i \rangle}}, \tag{3}$$
where $dx_t = x_t - x_{t-1}$, $dg_i = g_i - g_{i-1}$, and $\langle \cdot, \cdot \rangle$ denotes an inner product. Scale the motion similarity to $W_i^{(2)} = \exp(\alpha(\cos(\theta_i) - 1))$ where α is a constant.
3. *Noise perturbation*: Sample noise $\{v_j\} \sim \mathcal{N}(0, \sigma_p^2)$. For each (i, j) pair, we form position candidates at time $t+1$:
$$x_{t+1}^{(i,j)} = x_t + dg_{i+1} + v_j. \tag{4}$$
4. *Drift prevention*: Weigh each position candidate using Parzen window
$$p(x_{t+1}^{(i,j)}) = W_i^{(1)} W_i^{(2)} \sum_k \varphi\left(\frac{x_{t+1}^{(i,j)} - g_k}{h}\right), \tag{5}$$
where h is the window width and φ is the window function.
5. *Normalization*: Normalize weights so that $\sum_{i,j} p(x_{t+1}^{(i,j)}) = 1$.
6. *Prediction*: Sample x_{t+1} with the weight $p(x_{t+1}^{(i,j)})$.

to avoid drifting away from the given projected trajectory, we favor sample noise values that pull x_t' toward given data (spatial locality). In Figure 3, we eventually obtain the next position x_{t+1} as the predicted state because it is close to g_k. This synthesis algorithm is depicted in Algorithm 1. By using this algorithm, if we set the initial condition as $x_1 = g_1$ and $x_2 = g_2$, and let $\sigma_h = \sigma_p = 0$, we are able to reconstruct the input motion data.

Hundreds or even thousands of data points are still a sparse data set in the global subspace, normally no larger than 20 dimensions in our experiments. As a result, we speed up the sampling algorithm by ignoring outliers. For instance, if the current position x_t is closest to the example g_i, then we can take the nearest neighbors of g_i as the nearest neighbors of x_t, which can be computed beforehand. In addition, in (5), we can drop the summation operator and compute the window function using only the nearest neighbor g_k because when h is small, the probability of the occurrence of additional examples is very small. Hence, the synthesis can be performed in real time. On the other hand, the closed-loop LDS (CLDS) method [20] is not an online synthesis algorithm.

Note that Markov Chain Monte Carlo (MCMC) or other sampling methods may be substituted in our method but they are less efficient due to slow convergence problems.

4 Experiments

We apply the proposed method to synthesize dynamic textures and human motions, thereby demonstrating it is able to model temporal changes in spectral or spatial parameters of objects and image sequences. The synthesized videos are available on our web site.

4.1 Dynamic Texture Synthesis

For dynamic texture synthesis, the input data are raw image vectors. The image sequences used in our experiment are taken from a commonly used temporal texture database [14]. Most image sequences in the database have resolutions of 170 by 115 and contain 120 to 150 frames. They include both temporally stationary and non-stationary dynamic textures. In the following, we compare synthesized videos using our method and the LDS based method which is implemented by PCA with AR approach. The order of AR model is automatically determined by Schwarz's Bayesian criterion [12]. In the literature of dynamic texture synthesis, the-state-of-the-art approach [20] also uses a PCA model, but with a more sophisticated dynamic model. We use up to three PPCAs for each mixture model because the data sequences from the temporal texture database are short.

For stationary dynamic textures, we take a river sequence of 120 frames as an example. We use a 20-dimensional PCA for the LDS method, and align two 20-dimensional PPCA models into a 20-dimensional globally-coordinated subspace for our method. The synthesis results in Figure 4 show that our method is able to produce high-quality images, while the LDS method produces images with decreasing visual quality over time. Although CLDS [20] can fix this problem for stationary dynamic textures, for non-stationary dynamic textures, using a single PCA model results in significant loss of image quality after dimensionality reduction and reconstruction, while a mixture model alleviates such problem. The artifact is more serious when shape variations are significant (see Figure 5).

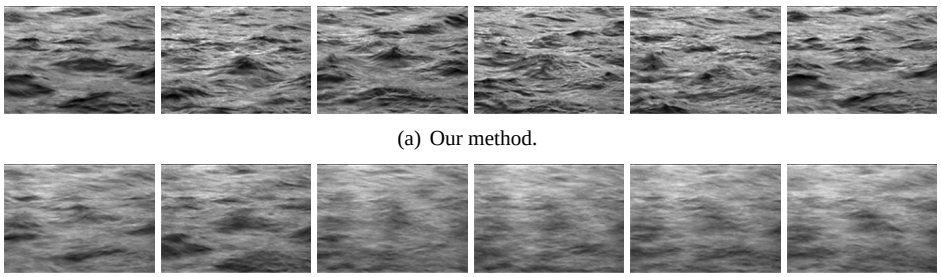
Synthesis results of two flag sequences using our method and the LDS approach are compared in Figure 6. Due to fast motion and dramatic shape variations, there is still some small artifact around the flag by our method (Figure 6(c)), although when being displayed as a movie, this artifact looks like motion blur and much less noticeable than the LDS approach. It all thanks to our local linear models that confine the artifact to be local, in contrast to the videos synthesized by the LDS method and other PCA based approaches.

Figure 7 shows the projected trajectories of the three dynamic texture sequences in the respective globally-coordinated subspace. Note that the input sequence has to form loops so that our synthesized sequence can be longer than the original video sequence, which holds true for most approaches in video or motion synthesis. Additional images can be interpolated and inserted between two similar but non-consecutive input image frames to create new image paths and enrich synthesis results.

Several sequences are synthesized using the proposed approach and existing methods. Notice that the synthesized sequences by our methods are smooth and crisp whereas the results using [13] or AR models have significant blurry or jittery artifacts. These results would further confirm that our method outperforms the existing methods.

4.2 Human Motion Synthesis

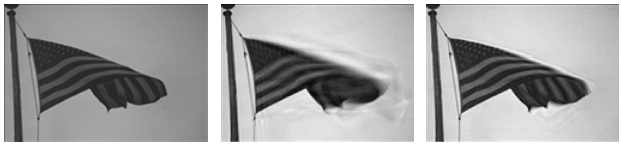
The inputs for human motion synthesis are 60-dimensional motion capture data for 20 body parts of a hierarchical kinematic model. All test sequences are between 200 and 500 frames. We use a mixture of up to three 12-dimensional PPCA models for each sequence. Other than the *bow* sequence that has no repeated motion, we synthesize 1,000 frames for all other input data. A few frames of each rendered motion sequence are shown in the Figure 8. More synthesized videos are available on our web site. All these sequences contain easily distinguishable poses. These experimental results show that our method is able to produce smooth and complex human motions.



(a) Our method.

(b) LDS approach.

Figure 4: Selected images from synthesized river sequences during extrapolation using (a) a mixture of two PPCA models with global coordination, and (b) LDS approach.



(a) Original image. (b) PCA. (c) Our method.

Figure 5: An image from a flag sequence reconstructed by PCA and our method.

5 Conclusions

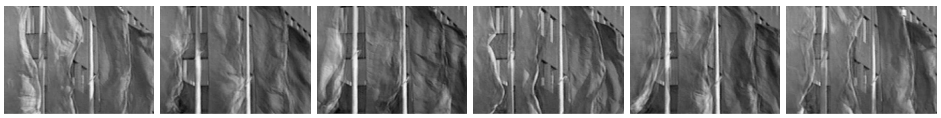
In this paper, we propose a new approach to learning and synthesizing multi-modal, complex dynamic data. We have shown that smooth and realistic synthesis can be accomplished by mixtures of linear subspace models with global coordination. Although both using mixtures of linear models, unlike SLDS, our model ensures continuous motion and does not require constraints for local model transitions. We have demonstrated our approach in two classes of motions: holistic textured motions and articulated human motions. In particular, compared to the existing works in dynamic texture analysis, our method enhances the visual quality in synthesis of temporally stationary dynamic textures, and is able to model and synthesize non-stationary dynamic textures with fast and large shape variations which have not been accomplished in the literature.

References

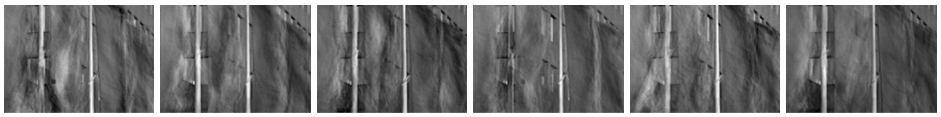
[1] M. Brand. Charting a manifold. In *Advances in Neural Information Processing Systems*, volume 15, pages 961–968, 2002.

[2] N. W. Campbell, C. Dalton, D. Gibson, and B. Thomas. Practical generation of video textures using the auto-regressive process. In *Proceedings of British Machine Vision Conference*, pages 434–443, 2002.

[3] Z. Ghahramani and G. E. Hinton. The EM algorithm for mixtures of factor analyzers. Technical Report CRG-TR-96-1, Department of Computer Science, University of Toronto, 1996.

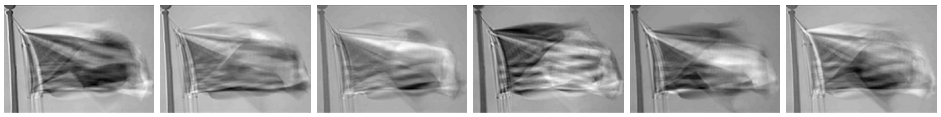


(a) Our method.



(b) LDS approach.

(c) Our method.



(d) LDS approach.

Figure 6: Corresponding image frames of synthesized dynamic textures by our and existing methods.

[4] Z. Ghahramani and G. E. Hinton. Variational learning for switching state-space models. *Neural Computation*, 12:831–864, 2000.

[5] K. Grochow, S. Martin, A. Hertzmann, and Z. Popovic. Style-based inverse kinematics. In *Proceedings of ACM SIGGRAPH*, 2004.

[6] V. Kwatra, A. Schödl, I. Essa, G. Turk, and A. Bobick. Graphcut textures: Image and video synthesis using graph cuts. *Proceedings of ACM SIGGRAPH*, pages 277–286, 2003.

[7] N. Lawrence. Gaussian process latent variable models for visualization of high dimensional data. In *Advances in Neural Information Processing Systems*, volume 16, pages 585–591, 2003.

[8] Y. Li, T. Wang, and H.-Y. Shum. Motion texture: A two-level statistical model for character motion synthesis. In *Proceedings of ACM SIGGRAPH*, pages 465–472, 2002.

[9] S. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, Dec 2000.

[10] S. Roweis, L. Saul, and G. E. Hinton. Global coordination of local linear models. In *Advances in Neural Information Processing Systems*, volume 14, pages 889–896, 2001.

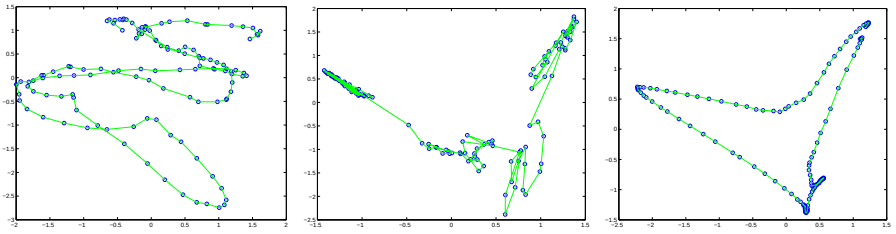
[11] A. Schödl, R. Szeliski, D. H. Salesin, and I. Essa. Video textures. In *Proceedings of ACM SIGGRAPH*, pages 489–498, July 2000.

[12] G. Schwarz. Estimating the dimension of a model. *Annals of Statistics*, 6:461–464, 1978.

[13] S. Soatto, G. Doretto, and Y. Wu. Dynamic textures. In *Proceedings of IEEE International Conference on Computer Vision*, volume 3, pages 439–446, 2001.

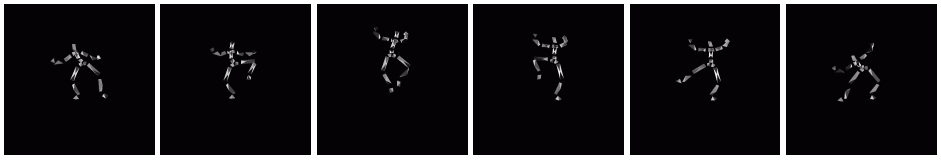
[14] M. Szummer and R. W. Picard. Temporal texture modeling. In *Proceedings of IEEE International Conference on Image Processing*, volume 3, pages 823–826, 1996.

[15] Y. W. Teh and S. Roweis. Automatic alignment of local representations. In *Advances in Neural Information Processing Systems*, volume 15, pages 841–848, 2002.

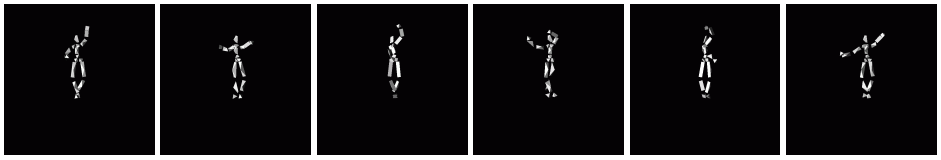


(a) River sequence. (b) Four-flag sequence. (c) Single flag sequence

Figure 7: The 20D trajectories of three dynamic texture sequences projected onto the first 2D of the respective globally-coordinated subspace.



(a) Selected images from our synthesized *bow* sequence.



(b) Selected images from our synthesized *ballet* sequence.

Figure 8: Synthesized human motion sequences by our method.

[16] J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

[17] M. E. Tipping and C. M. Bishop. Mixtures of probabilistic principal component analysers. *Neural Computation*, 11(2):443–482, 1999.

[18] R. Urtasun, D. J. Fleet, A. Hertzmann, and P. Fua. Priors for people tracking from small training sets. In *Proceedings of IEEE International Conference on Computer Vision*, pages 403–410, 2005.

[19] J. Wang, D. J. Fleet, and A. Hertzmann. Gaussian process dynamical models. In *Advances in Neural Information Processing Systems*, volume 18, 2005.

[20] L. Yuan, F. Wen, C. Liu, and H.-Y. Shum. Synthesizing dynamic texture with closed-loop linear dynamic system. In *Proceedings of European Conference on Computer Vision*, volume 2, pages 603–616, 2004.