Learning Nonlinear Manifolds from Time Series

Ruei-Sung Lin[1] † Che-Bin Liu[1] ‡ Ming-Hsuan Yang[2] Narendra Ahuja[1] Stephen Levinson[1]

[1] University of Illinois at Urbana-Champaign, Urbana, IL 61801, USA
[2] Honda Research Institute, Mountain View, CA 94041, USA

Abstract. There has been growing interest in developing nonlinear dimensionality reduction algorithms for vision applications. Although progress has been made in recent years, conventional nonlinear dimensionality reduction algorithms have been designed to deal with stationary, or independent and identically distributed data. In this paper, we present a novel method that learns nonlinear mapping from time series data to their intrinsic coordinates on the underlying manifold. Our work extends the recent advances in learning nonlinear manifolds within a global coordinate system to account for temporal correlation inherent in sequential data. We formulate the problem with a dynamic Bayesian network and propose an approximate algorithm to tackle the learning and inference problems. Numerous experiments demonstrate the proposed method is able to learn nonlinear manifolds from time series data, and as a result of exploiting the temporal correlation, achieve superior results.

1 Introduction

Dimensionality reduction algorithms has been successful applied to vision problems for decades. Yet many tasks can be better approached with nonlinear methods, and recently there has been growing interests in developing nonlinear dimensionality reduction (NLDR) algorithms for vision applications. Nonlinear dimensionality reduction aims at representing high dimensional data with low dimensional intrinsic parameters. For data assumed to be distributed along a low dimensional nonlinear manifold, solving NLDR is equivalent to recovering their intrinsic coordinates. There exist two main approaches that transform data to their intrinsic parameters within a global coordinate system. Embedding methods such as Isomap [1] and LLE [2] find the intrinsic coordinates on the manifold from a set of samples. However, one limitation is that these algorithms discover the underlying embeddings rather than mapping functions from observed data. An alternative approach is to find a nonlinear mapping between the data and their intrinsic coordinates, either with a combination of local linear models [3][4][5], or a single nonlinear function [6][7][8].

All the abovementioned methods assume that the observed data samples are stationary or independent, identically (i.i.d.) distributed. However, numerous real world applications, e.g., object tracking and motion synthesis, entail analyzing continuous data sequences where strong temporal correlation inherent in samples should be taken into consideration. Consequently, it is essential to extend a conventional NLDR algorithm

† Current affiliation: Motorola Labs, Schaumburg, IL 60196
‡ Current affiliation: Epson Research & Development Inc., Palo Alto, CA 94304

to account for temporal dependence in the data, thereby discovering sample dynamics along the manifold.

Few attempts have been made to tackle the NLDR problems for time series. Examples include [9] that extends the standard generative topographic mapping to handle sequential data within the hidden Markov model framework, [10] that modifies the Isomap algorithm with heuristics to find the underlying embedding from data sequences, and [8] which applies a semi-supervised regression model to learn nonlinear mapping from temporal data. Nevertheless, these algorithms are mainly concerned with learning the nonlinear embedding or mapping functions. Less effort is made to model the dynamic process of the intrinsic coordinates on the manifold.

In this paper, we address both nonlinear dimensionality reduction with bidirectional projection and the dynamics of time series data within a single statistical framework. We propose a model that learns the nonlinear mapping from time series that is capable of performing dynamic inference. Building on the work on the global coordination model [3] which provides a generative approach for the nonlinear mapping with a mixture of factor analyzers, we extend this graphical model to a dynamic Bayesian network (DBN) by adding links among the intrinsic coordinates to account for temporal dependency. Although the exact inference of this model is intractable, we exploit unique properties of nonlinear mapping within the global coordination model and propose an efficient approximate algorithm. We show that by applying this approximate algorithm, this DBN becomes a generalized Kalman filter for nonlinear manifold where model parameters are constantly adjusted.

We take a variational learning approach to estimate model parameters. Given initial values of the parameters, we use our approximate inference algorithm to estimate the statistics of latent variables. Then based on these statistics, we update the model parameters in the DBN. With this iterative process, the learning algorithm converges to a local optimum. For concreteness, we demonstrate the merits of this DBN with applications such as object tracking and video synthesis in which it is essential to model the sample dynamics on the underlying manifold.

The rest of this paper is organized as follows. We first briefly review the global coordination model [3] in Section 2. Next, we present an extension of this model to a DBN in which temporal correlation is taken into consideration in Section 3. Based on this DBN, we propose an approximate inference method and a learning algorithm for model parameters. Experimental results on synthetic and real world applications are presented in Section 4. We conclude this paper with discussions on the proposed model and future work in Section 5.

2 Global Coordination of Local Linear Models

The global coordination model is an extension of mixture of factor analyzers in which latent variables are aligned in a global coordinate system. Denote $y \in \mathcal{R}^D$ the observed data, s the index of the selected linear model, and $z_s \in \mathcal{R}^d$ the latent variables in the s-th local linear model. The joint probability of these parameters is:

$$P(y, z_s, s) = P(y|z_s, s)P(z_s|s)P(s) \tag{1}$$

in which $P(s)$ is the prior probability of local model s, $P(z_s|s)$ is a zero mean univariate Gaussian, i.e., $P(z_s|s) = \mathcal{N}(0, I_d)$, and $P(y|z_s, s)$ is defined by a factor analyzer:

$$P(y|z_s, s) = \frac{1}{\sqrt{(2\pi)^D |\Psi_s|}} \exp(-\frac{1}{2}(y - \Lambda_s z_s - \mu_s)^T \Psi_s^{-1} (y - \Lambda_s z_s - \mu_s)) \quad (2)$$

Since the latent variable z_s is defined within the local coordinate system of s-th local model, the global coordination algorithm transforms z_s to the corresponding intrinsic parameter within a global coordinate system. Let g denote the global coordinate of data y that is generated from s-th local linear model with z_s, the transformation is defined by

$$g(s, z_s) = A_s z_s + \kappa_s, \quad P(g|s, z_s) = \delta(g - A_s z_s - \kappa_s) \quad (3)$$

where A_s is a full ranked matrix to ensure a bidirectional mapping, and κ_s is an offset.

Given this model, the mapping from y to g is described by:

$$P(g|y) = \sum_s P(g|y, s) P(s|y) \quad (4)$$

where

$$P(g|y, s) = \int P(g|s, z_s) P(z_s|s, y) dz_s \quad (5)$$

and the mapping from g to y is defined as:

$$P(y|g) = \sum_s P(y|g, s) P(s|g). \quad (6)$$

Although $P(g|y)$ and $P(y|g)$ are in the form of mixture of Gaussians, the distributions of $P(g|y)$ and $P(y|g)$ are expected to be unimodal since ideally the mapping between g and y should be one to one. For example given two mixture components s_i and s_j, the posterior distributions for global coordinates of a data point computed by (6) should be as identical as possible since g is the global coordinate of y. That is, $P(g|y, s_i)$ should be close to $P(g|y, s_j)$ as possible, i.e., $P(g|y, s_i) \approx P(g|y, s_j)$. This unimodal constraint is imposed in learning the global coordination of local linear models by Roweis et al. [3], and we take a similar approach. For mappings between y and g, $E[P(g|y)]$ and $E[P(y|g)]$ are used in this work.

Learning the global coordination model is equivalent to estimating parameters $\{(\Lambda_s, \mu_s, A_s, \kappa_s)\}$ from a set of observed data. This is an ill-posed problem since global coordinates of the data set are unknown. A few methods have been recently been proposed to address this issue. Wang et. al. [5] apply Isomap [1] to obtain global coordinates of the data, and learn the model parameters by solving a regression problem. Roweis et. al. [3] present an algorithm in which a regularization term is introduced to enforce the alignment constraints, and model parameters are estimated using variational algorithms. Nevertheless, both approaches have limitations as the method in [5] requires a good Isomap embedding, and the algorithm in [3] might have serious local minimal problems. In addition, both methods assume observations are i.i.d. samples without taking the temporal dependence into consideration.

3 Dynamic Global Coordination Model

To account for the temporal relationship among data samples, we incorporate the global coordination method into a dynamic model. Now observations $\{y_t\}$ are a temporal sequence generated from a Markovian process $\{g_t\}$ and the mapping from g_t to y_t is based on (6). The resulting dynamic Bayesian network is depicted in Figure 1.

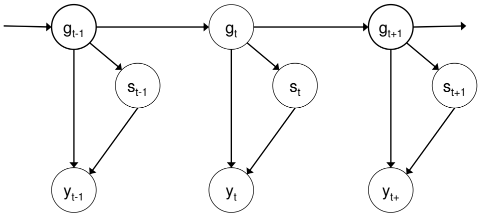

Fig. 1. Our dynamic Bayesian networks that is based on the temporal dependency among the global coordinates.

3.1 Inference

We now provide the inference algorithms for the model. Although the DBN shown in Figure 1 is structurally complex, it becomes a simple state-space model if we marginalize out s_t at each time step.

$$\begin{aligned} P(g_t|y_{1:t}) &\propto \sum_{s_t} P(y_t|g_t, s_t)P(s_t|g_t) \int P(g_t|g_{t-1})P(g_{t-1}|y_{1:t-1})dg_{t-1} \\ &= P(y_t|g_t) \int P(g_t|g_{t-1})P(g_{t-1}|y_{1:t-1})dg_{t-1} \qquad (7) \end{aligned}$$

Note that $P(y_t|g_t)$ is composed of a mixture of Gaussians. If we compute (7) directly for exact inference, the number of mixtures in the posterior distribution will grow exponentially as the time index increases, thereby making the problem intractable. As discussed earlier, the ideal mapping between y and g at any time instance should be one to one. For efficient inference, we apply the first order Generalized Pseudo Bayesian (GPB) algorithm [11] to approximate $P(y_t|g_t)$, which can be shown to be the best single Gaussian approximation in the KL sense.

In this work, we compute $P(y_t|g_t)$ with Bayes rule

$$P(y_t|g_t) = \frac{P(g_t|y_t)P(y_t)}{P(g_t)} \qquad (8)$$

and neglect the effect of $P(g_t)$ for the reason that will be explained in the next section. That is, we approximate $P(y_t|g_t)$ using the joint probability $P(y_t, g_t)$. Since $P(y_t)$ is a constant with known y_t, we carry out GPB approximation using $P(g_t|y_t)$.

Let (μ_t, Σ_t) denote the mean and the covariance matrix of the Gaussian that we use to approximate $P(g_t|y_t)$, and likewise $P(g_t|y_t, s_t) \sim \mathcal{N}(\mu_t^s, \Sigma_t^s)$. From (4), (μ_t, Σ_t) can be estimated by minimizing the weighted KL-distance:

$$(\mu_t, \Sigma_t) = \arg\min_{\mu, \Sigma} \sum_s P(s_t|y_t) KL(\mathcal{N}(\mu_t^s, \Sigma_t^s)||\mathcal{N}(\mu, \Sigma)). \tag{9}$$

and the analytic solution is

$$\mu_t = \sum_s P(s_t|y_t)\mu_t^s, \quad \Sigma_t = \sum_s P(s_t|y_t)\left(\Sigma_t^s + (\mu_t - \mu_t^s)(\mu_t - \mu_t^s)^T\right). \tag{10}$$

In our work, the dynamic model is set to be $P(g_t|g_{t-1}) = \mathcal{N}(Cg_{t-1}, \hat{Q})$ where C is the system matrix. Since $P(y_t|g_t)$ and $P(g_t|g_{t-1})$ are now both Gaussians, as a result the posterior distribution $P(g_t|y_{1:t})$ in (7) is also a Gaussian.

Let $P(g_t|y_{1:t}) \sim \mathcal{N}(g_t^t, \Sigma_t^t)$ and $P(g_t|y_{1:t-1}) \sim \mathcal{N}(g_t^{t-1}, \Sigma_t^{t-1})$. It can be shown that in our dynamic Bayesian network,

$$g_t^{t-1} = Cg_{t-1}^{t-1}, \quad \Sigma_t^{t-1} = C\Sigma_t^{t-1}C^T + \hat{Q} \tag{11}$$

, and

$$\Sigma_t^t = \left((\Sigma_t^{t-1})^{-1} + \Sigma_t^{-1}\right)^{-1} \tag{12}$$

$$g_t^t = \Sigma_t^t\left((\Sigma_t^{t-1})^{-1}g_t^{t-1} + \Sigma_t^{-1}\mu_t\right) \tag{13}$$

Likewise, it follows that for the cases of smoothing and lag-one smoothing with our model:

$$\mu_t^T = \mu_t^t + J_t(\mu_{t+1}^T - \mu_{t+1}^t) \tag{14}$$

$$\Sigma_t^T = \Sigma_t^t + J_t\left(\Sigma_{t+1}^T - \Sigma_{t+1}^t\right)J_t^T \tag{15}$$

$$J_t = \Sigma_t^t C^T[\Sigma_{t+1}^t]^{-1} \tag{16}$$

$$\Sigma_{t,t-1}^T = \Sigma_t J_{t-1}^T + J_t\left(\Sigma_{t+1,t}^T - C\Sigma_t^t\right)J_{t-1}^T \tag{17}$$

where $\Sigma_{t,t-1}^T = E\left[(g_t - \mu_t^T)(g_{t-1} - \mu_{t-1}^T)^T|y_{1:T}\right]$.

It should be emphasized that although our filtering and smoothing procedures are similar to the ones used in standard Kalman filter, our model is a generalized filter. While Kalman filter performs dynamic inferences on a linear manifold, our model extends this framework and performs dynamic inference on a nonlinear manifold. Therefore, unlike a standard Kalman filter which uses a fixed Gaussian for the measurement function $P(y_t|g_t)$, in our model μ_t and Σ_t are adaptively updated according to y_t to account for the nonlinearity on the manifold as in shown in (10).

3.2 Learning

We take a variational approach to learn the model parameters. Let $\theta = \{(\varLambda_s, \mu_s, A_s, \kappa_s, \varPsi_s\ ), C, \hat{Q}\}$ denote the set of model parameters. Using Jensen's inequality,

$$\log P(y_{1:T}|\theta) \geq \varPhi = \sum_{s_{1:T}} \int Q(g_{1:T}, s_{1:T}|\theta) \log\left(\frac{P(y_{1:T}, g_{1:T}, s_{1:T}|\theta)}{Q(g_{1:T}, s_{1:T}|\theta)}\right) dg_{1:T} \tag{18}$$

We first define a proper function Q and then learn the model parameters using an EM algorithm. Starting with the initial value $\theta^{(0)}$, in the E-step we maximize $\varPhi$ with respect to $Q(g_{1:T}, s_{1:T}|\theta^{(0)})$. In the M-step we fix Q and update the model parameters θ to maximize $\varPhi$. This iterative procedure continues until it reaches convergence.

In this work, we factorize $Q(g_{1:T}, s_{1:T}|\theta)$ into two components:

$$Q(g_{1:T}, s_{1:T}|\theta) = Q(s_{1:T}|\theta)Q(g_{1:T}|\theta) \tag{19}$$

For $Q(g_{1:T}|\theta)$, we want it to be close to $P(g_{1:T}|y_{1:T}, \theta)$ as possible. Let $\tilde{P}(g_{1:T}|y_{1:T}, \theta)$ denote the approximation of $P(g_{1:T}|y_{1:T}, \theta)$ computed by our inference algorithm discussed in Section 3.1, and set $Q(g_{1:T}|\theta) = \tilde{P}(g_{1:T}|y_{1:T}, \theta)$. For $Q(s_{1:T}|\theta)$, we further factorize it to $Q(s_{1:T}|\theta) = \prod_{t=1}^{T} Q(s_t|\theta)$, and define $Q(s_t|\theta) = q_{s,t}$ where $q_{s,t}$ is a scalar.

It follows that,

$$\begin{aligned}\varPhi = &\sum_{t=1}^{T}\sum_{s=1}^{S} q_{s,t} \int \tilde{P}(g_t|y_{1:T}, \theta) \log P(y_t, g_t, s_t|\theta) dg_t \\ &+ \sum_{t=2}^{T} \int \tilde{P}(g_t, g_{t-1}|y_{1:T}, \theta) \log P(g_t|g_{t-1}) dg_t dg_{t-1} \\ &- \sum_{t=1}^{T}\sum_{s=1}^{S} q_{s,t} \log q_{s,t} - \int \tilde{P}(g_{1:T}|y_{1:T}, \theta) \log \tilde{P}(g_{1:T}|y_{1:T}, \theta) dg_{1:T}\end{aligned} \tag{20}$$

Notice that in the E-step we do not compute $\tilde{P}(g_{1:T}|y_{1:T}, \theta)$, but rather $\tilde{P}(g_t|y_{1:T}, \theta)$ and $\tilde{P}(g_t, g_{t-1}|y_{1:T}, \theta)$ for all t. With known $\tilde{P}(g_t|y_{1:T}, \theta)$, the dynamic model is factorized into T global coordination models at each time instance, and $q_{s,t}$ is:

$$q_{s,t} = \frac{\exp(-\mathcal{E}_{s,t})}{\sum_s \exp(-\mathcal{E}_{s,t})}, \quad \mathcal{E}_{s,t} = \int \tilde{P}(g_t|y_{1:T}, \theta) \log P(y_t, g_t, s_t|\theta) dg_t \tag{21}$$

In the M-step with known $\tilde{P}(g_t|y_{1:T}, \theta)$ and $q_{s,t}$, the model parameters are updated as follows. Let $q_s = \sum_t q_{s,t}$,

$$P(s) = q_s / \sum_s q_s \tag{22}$$

$$\kappa_s = q_s^{-1} \sum_t q_{s,t} \mu_t^T \tag{23}$$

$$\mu_s = q_s^{-1} \sum_t q_{s,t} y_t \tag{24}$$

Also denote $y_{s,t} = y_t - \mu_s$, $g_{s,t} = \mu_t^T - \kappa_s$, $M_s = \sum_t q_{s,t} y_{s,t} g_{s,t}^T$ and $N_s = \sum_t q_{s,t}[\Sigma_t^T + g_{s,t} g_{s,t}^T]$, we obtain the remaining model parameters in θ:

$$\Lambda_s = M_s N_s^{-1} A_s \tag{25}$$

$$[\Psi_s]_i = q_s^{-1} \sum_t q_{s,t} \left\{ \left[y_{s,t} - \Lambda_s A_s^{-1} g_{s,t} \right]_i^2 + \left[\Lambda_s A_s^{-1} \Sigma_t^T A_s^{-T} \Lambda_s^T \right]_i \right\} \tag{26}$$

$$A_s^{-1} = (I + \Lambda_s^T \Psi_s^{-1} \Lambda_s)^{-1} \{ A_s^T q_s + \Lambda_s^T \Psi_s^{-1} M_s \} N_s^{-1} \tag{27}$$

As for the dynamic model, denote $D_{t,t-1} = \Sigma_{t,t-1}^T + (\mu_t^T)(\mu_{t-1}^T)^T$ and $D_t^T = \Sigma_t^T + (\mu_t^T)(\mu_t^T)^T$:

$$C_{new} = \left[\sum_{t=2}^{T} D_{t,t-1}^T \right] \left[\sum_{t=2}^{T} D_{t-1}^T \right]^{-1} \tag{28}$$

$$\hat{Q}_{new} = \frac{1}{T-1} \sum_{t=2}^{T} (D_t^T - C_{new} D_{t,t-1}) \tag{29}$$

These equations bear similarities to the work by Roweis et al. [3], but at its core they are rather different by design. In our model, the estimation of g_t is conditioned on the whole observation sequence $y_{1:T}$, i.e., $\tilde{P}(g_t | y_{1:T}, \theta)$, whereas in [3] the estimation of g_t is conditioned on a single, i.i.d. sample y_t. That is, our model is developed within a dynamic context in which temporal correlation is taken into consideration.

Note that in our algorithm, when factorizing $P(y_{1:T}, g_{1:T}, s_{1:T})$,

$$P(y_{1:T}, g_{1:T}, s_{1:T}) = P(g_1) \prod_{t=2}^{T} P(g_t | g_{t-1}) \prod_{t=1}^{T} P(y_t | s_t, g_t) P(s_t | g_t) \tag{30}$$

we use joint probability $P(s_t, g_t)$ instead of $P(s_t | g_t)$. Neglecting $P(g_t)$ here makes the model consistent with our inference procedure described in the previous section. As a matter of fact, $P(g_t)$ has little effect on computing $\log \Big(P(y_{1:T}, g_{1:T}, s_{1:T}|) / \tilde{P}(g_{1:T} | y_{1:T}) \Big)$ since $P(g_t)$ in $P(y_{1:T}, g_{1:T}, s_{1:T})$ and $\tilde{P}(g_{1:T} | y_{1:T})$ can be canceled out.

4 Experiments

We apply the proposed algorithm to learn nonlinear manifolds and sample dynamics from time series for a few applications. Comparative studies are carried out to show the merits of the proposed method that takes temporal dependence into design, thereby better recovering the underlying manifold from time series data. More experimental results are available on our web site (`http://www.ifp.uiuc.edu/˜rlin1/dgcm.html`).

4.1 Synthetic Data

We first test our algorithm with a synthetic data set generated from a 2D manifold and embedded in a 3D space as shown in Figure 2. The data points are generated by a 2D random walk, similar to the data set tested in [8], in a rectangle area $[0, 5] \times [-3, 3]$, and then embedded in 3D by a mapping function $f(x, y) = (x, |y|, \sin(\pi y)(y^2 + 1)^{-2} + 0.3y)$. Notice that this data set is challenging as it is difficult to estimate the neighborhood structure around the neck where the manifold is folded.

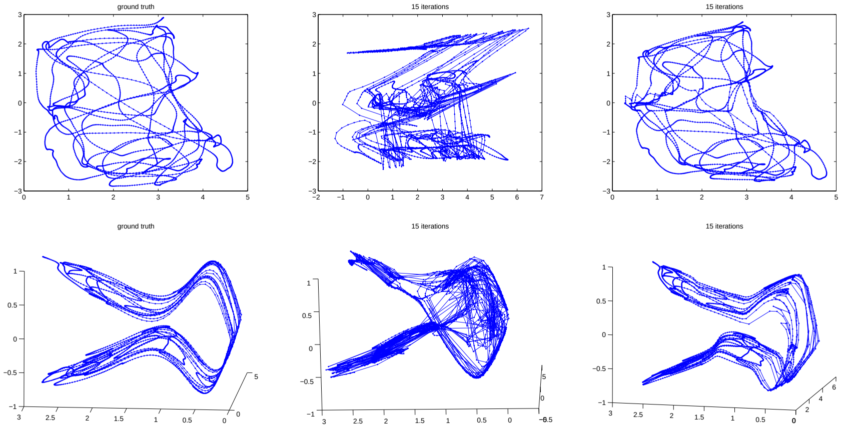

Fig. 2. Synthetic data: (left column) ground truth data points generated from a random walk path in 2D and its embedding in 3D space. (middle column) recovered 2D manifold and its 3D lifting using the method by Roweis et al. after 15 iterations [3]. (right column) recovered 2D manifold and its 3D lifting using parameters after 15 iterations.

The second and third columns of Figure 2 show the results using the method by Roweis et al [3] and our algorithm. Notice that without taking the temporal information into consideration, the random walk path on the 2D manifold cannot be recovered correctly and thereby the 3D lifted points near the neck region are tangled together. Compared to the ground truth on the first column, our method recovers the 2D manifold better than the unsupervised nonlinear manifold learning algorithm without taking temporal dependence into consideration. In contrast to the semi-supervised method presented in [8], our algorithm is able to discover the underlying 2D manifold from 3D time series as the temporal correlation is exploited in estimating local neighborhood structures without any supervision.

4.2 Object Tracking

We apply the proposed dynamic model to an object tracking problem based on appearance. Images of an object appearance are known to be embedded on a nonlinear manifold, and a sequence of observations is expected to form a smooth trajectory on the manifold. Exploiting this strong temporal dependency, we can better track an object by exploring the trajectory of the mapped global coordinates on the appearance manifold from observed images. The graphical model for object tracking is shown in Figure 3 where x_t is the video frame at time t, location parameters l_t specifies the location of the tracked object in x_t, and g_t is the global coordinates of the object's appearance in x_t.

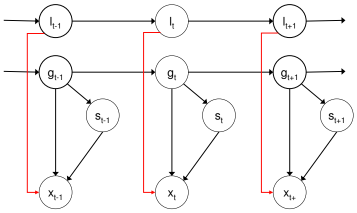

Fig. 3. Extension of our dynamic global coordination model for object tracking. Based on this model, we apply Rao-Blackwellized particle filter for efficient tracking.

The state vector includes the location parameters and the global coordinates of the observed image, thereby making it ineffective to employ a simple particle filter for tracking. However, we can factorize the posterior as:

$$P(l_t, g_t|y_{1:t}) = P(g_t|x_{1:t}, l_t)P(l_t|x_{1:t}) \tag{31}$$

According to our inference algorithm in Section 3.1, $P(g_t|x_{1:t}, l_t)$ is approximated as an Gaussian distribution. Therefore, our tracker can sample particles only on l_t and model $P(g_t|x_{1:t}, l_t)$ using an analytical distribution. That is, our tracker can use Rao-Blackwellized particle filter (RBPF) [12] for efficient tracking.

We test our model on a face tracking experiment which undergoes large pose variations. In our tracking video, there are other faces around the target object. We first test the video using a baseline tracker that tracks location parameters l_t only, and use a mixture of factor analyzers as the measurement function. The result shows that this tracker might track the wrong target when the two faces are close. On the other hand, our tracker is able to track the target well even though several similar objects appear in close proximity because we exploit the temporal dependency in the appearance images of the target (i.e., global coordinates). Figure 4 shows the tracking results using the proposed method. More detail on incorporating a RBPF into our dynamic model and experimental results are available on our web page.

4.3 Video Synthesis

We demonstrate merits of the proposed algorithm on a video synthesis problem. The image sequences are taken from a database of textured motion [13] where most videos have 170 by 115 pixel resolution and contain 120 to 150 frames. Such problem has been referred to a dynamic texture problem where scene appearance is modeled in a linear subspace [14]. However, scene

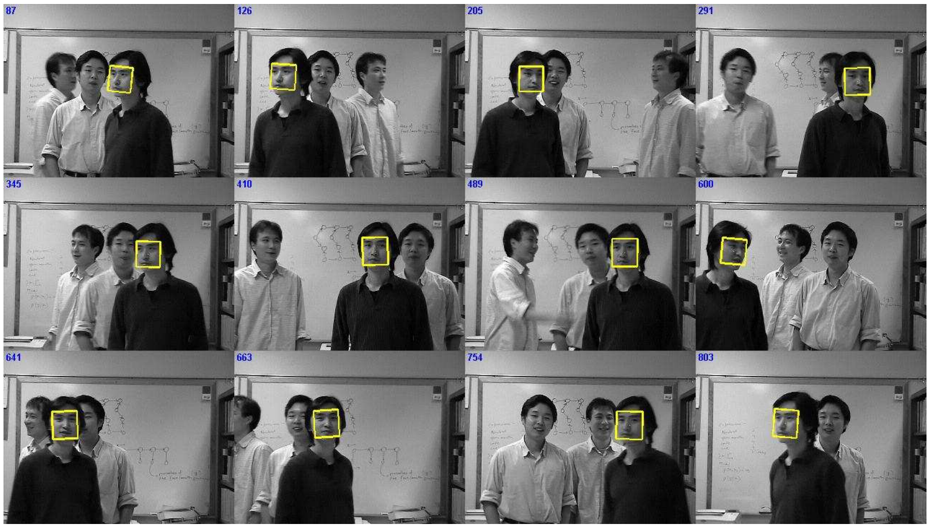

Fig. 4. Tracking results (left to right on each row): a target with large pose variation and moving in close proximity of similar faces. Our algorithm is able to track the target person in different pose, without confusing with other people.

appearance is usually complex and rarely linear. In addition, for a short video, thus a sparse data set, temporal correlations between image frames offer additional information to robustly learn its underlying low-dimensional manifold.

In our experiment, we learn the nonlinear manifold of scene appearance using our proposed algorithm by setting the system matrix C in our dynamic model to be an identity matrix, i.e., $P(g_t|g_{t-1}) = \mathcal{N}(g_{t-1}, \hat{Q})$. For each sequence, we model the underlying scene dynamics as a continuous low-dimensional trajectory along a globally coordinated manifold using a mixture of 20-dimensional factor analyzers. From each learned trajectory, we then generate synthesized videos by drawing samples and mapping them back to the image space. Note that care needs to be taken in sampling points along the learned trajectory to prevent drifts. Otherwise the synthesized images may not look realistic. The details of our sampling algorithm can be found on our web site.

Figure 5 shows the synthesized results of our method (a mixture of two factor analyzers for river sequence and a mixture of three factor analyzers for flag sequence) and the dynamic texture approach [14]. More videos are available at our web page.

Clearly the images synthesized by our method (first and third rows) are significantly crisper than the ones generated by the dynamic texture algorithm (second and fourth rows). The results are not surprising as complex scene dynamics inherent in videos can be better modeled on a globally coordinated nonlinear manifold rather than a linear dynamic system (LDS). Although the closed-loop LDS approach [15] improves results by [14], it also models scene appearance in a linear subspace and therefore cannot synthesize high-quality videos of complex scenes such as our flag example.

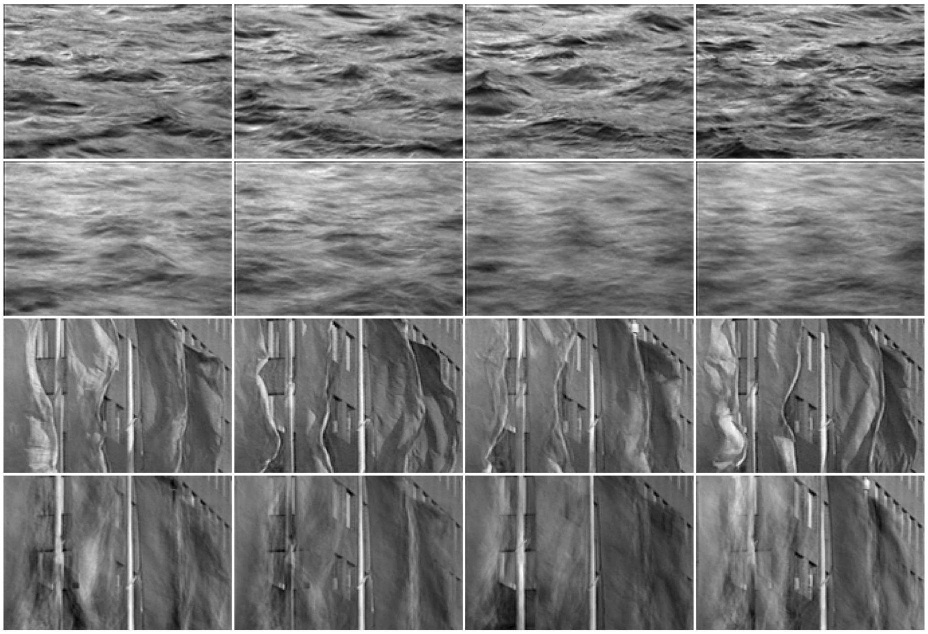

Fig. 5. Synthesized results by our method (first and third rows) and the dynamic texture algorithm (second and fourth rows). Clearly the images synthesized by our method are significantly crisper than the ones generated by the dynamic texture algorithm.

5 Concluding Remarks

Numerous vision problems entail analyzing time series where the underlying nonlinear manifold as well as strong temporal correlation among the data should be learned and exploited. In this paper, we extend the global coordination model within a dynamic context to learn the nonlinear manifolds and the dynamics inherent in time series data. Positing this problem within a Bayesian framework, we present an approximate algorithm for efficient inference and parameter learning. The proposed algorithm finds numerous applications from which the merits are demonstrated. Our future work includes finding better initialization methods in learning model parameters, and applying the proposed algorithm to other problem domains.

Acknowledgments

We thank David Forsyth and the anonymous reviewers for their valuable comments. The support of the Office of Naval Research under grant N00014-03-1-0107 is gratefully acknowledged.

References

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

2. Roweis, S., Saul, L.: Nonlinear dimensionality reduction by locally linear embedding. Science **290** (2000) 2323–2326
3. Roweis, S., Saul, L., Hinton, G.E.: Global coordination of local linear models. In: Advances in Neural Information Processing Systems. Volume 14. (2001) 889–896
4. Brand, M.: Charting a manifold. In: Advances in Neural Information Processing Systems. Volume 15. (2002) 961–968
5. Wang, Q., Xu, G., Ai, H.: Learning object intrinsic structure for robust visual tracking. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition. Volume 2. (2003) 227–234
6. Elgammal, A., Lee, C.S.: Inferring 3d body pose from silhouettes using activity manifold learning. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition. Volume 2. (2004) 681–688
7. Urtasun, R., Fleet, D.J., Hertzmann, A., Fua, P.: Priors for people tracking from small training sets. In: Proceedings of IEEE International Conference on Computer Vision. (2005) 403–410
8. Rahimi, A., Recht, B., Darrell, T.: Learning appearance manifolds from video. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition. Volume 1. (2005) 868–875
9. Bishop, C.M., Hinton, G.E., Strachan, I.G.: GTM through time. In: Proceedings of IEE Fifth International Conference on Artificial Neural Networks. (1997) 11–116
10. Jenkins, O.C., Mataric, M.: A spatio-temporal extension to Isomap nonlinear dimension reduction. In: Proceedings of International Conference on Machine Learning. (2004) 441–448
11. Bar-Shalom, Y., Li, X.: Estimation and Tracking: Principles, Techniques, and Software. Artech House (1993)
12. Khan, Z., Balch, T., Dellaert, F.: A Rao-Blackwellized particle filter for eigentracking. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition. Volume 2. (2004) 980–986
13. Szummer, M.: MIT temporal texture database. (ftp://whitechapel.media.mit.edu/pub/szummer/temporal-texture/)
14. Soatto, S., Doretto, G., Wu, Y.: Dynamic textures. In: Proceedings of IEEE International Conference on Computer Vision. Volume 2. (2001) 439–446
15. Yuan, L., Wen, F., Liu, C., Shum, H.Y.: Synthesizing dynamic texture with closed-loop linear dynamic system. In: Proceedings of European Conference on Computer Vision. Volume 2. (2004) 603–616