Motion Texture: A Two-Level Statistical Model for Character Motion Synthesis

Yan Li* Tianshu Wang† Heung-Yeung Shum*

*Microsoft Research, Asia †Xi'an Jiaotong University, P.R.China

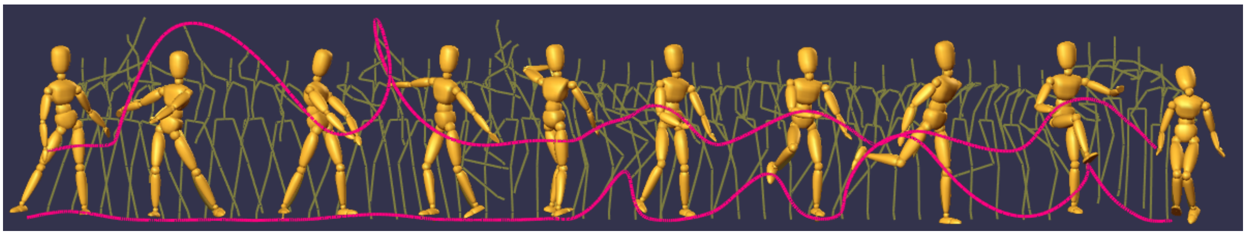


Figure 1: This 320-frame sequence of dance motion is choreographed from (1) the starting frame, (2) the ending frame and (3) the learnt motion texture from motion captured dance data. Four motion textons are generated from the motion texture and then used to synthesize all the frames in this sequence. A number of key frames are also shown in the figure to demonstrate that the synthesized motion is natural, smooth and realistic (Two red lines indicate the trajectories of the right hand and right foot).

Abstract

In this paper, we describe a novel technique, called motion texture, for synthesizing complex human-figure motion (e.g., dancing) that is statistically similar to the original motion captured data. We define motion texture as a set of motion textons and their distribution, which characterize the stochastic and dynamic nature of the captured motion. Specifically, a motion texton is modeled by a linear dynamic system (LDS) while the texton distribution is represented by a transition matrix indicating how likely each texton is switched to another. We have designed a maximum likelihood algorithm to learn the motion textons and their relationship from the captured dance motion. The learnt motion texture can then be used to generate new animations automatically and/or edit animation sequences interactively. Most interestingly, motion texture can be manipulated at different levels, either by changing the fine details of a specific motion at the texton level or by designing a new choreography at the distribution level. Our approach is demonstrated by many synthesized sequences of visually compelling dance motion.

CR Categories: I.3.7 [Computer Graphics]: Three Dimensional Graphics and Realism—Animation; G.3 [Mathematics of Computing]: Probability and Statistics—Time series analysis

Keywords: Motion Texture, Motion Synthesis, Texture Synthesis, Motion Editing, Linear Dynamic Systems.

*3F Beijing Sigma Center, No. 49 Zhichun Road, Haidian District, Beijing 100080, P.R. China. Email: {yli,hshum}@microsoft.com

†This work was done when Tianshu Wang was visiting at Microsoft Research, Asia.

1 Introduction

Synthesis of realistic character animation is an active research area and has many applications in entertainment and biomechanics. Recent advances in motion capture techniques and other motion editing software facilitate our generating of human animation with unprecedented ease and realism. By recording motion data directly from real actors and mapping them to computer characters, high quality motion can be generated very quickly. The captured motion can also be used to generate new animation sequences according to different constraints. Many techniques have been developed to tackle the difficult problem of motion editing. These techniques include motion signal processing [8], human locomotion in Fourier domain [42], motion warping [43], motion retargeting [15, 38], physically based motion transformation [33] and motion editing with a hierarchy of displacement maps [23]. More recently, several approaches have been proposed to interactively synthesize human motion by reordering the preprocessed motion capture data [21, 1, 22].

To make the edited motion "realistic", it is important to understand and incorporate the dynamics of the character motion. In physically based motion transformation, for instance, Popović and Witkin [33] obtain a physical spacetime optimization solution from the fitted motion of a simplified character model. In this paper, we present a different approach to the problem of editing captured motion by learning motion dynamics from motion captured data. We model local dynamics (of a segment of frames) by a linear dynamic system, and global dynamics (of the entire sequence) by switching between these linear systems. The motion dynamics are modeled in an analytical form which constrains the consecutive body postures. The meaning of dynamics as used in this paper is different from that in traditional animation literature, where dynamics denotes an interactive system involving force-based motion.

We call our model *motion texture* because motion sequences are analogous to 2D texture images. Similar to texture images, motion sequences can be regarded as stochastic processes. However, while texture images assume a two-dimensional spatial distribution, motion textures display a one-dimensional temporal distribution. We define motion texture by a two-level statistical model: a set of motion textons at the lower level, and the distributions of textons at the higher level. Intuitively, motion textons are those repetitive patterns in complex human motion. For instance, dance motion may

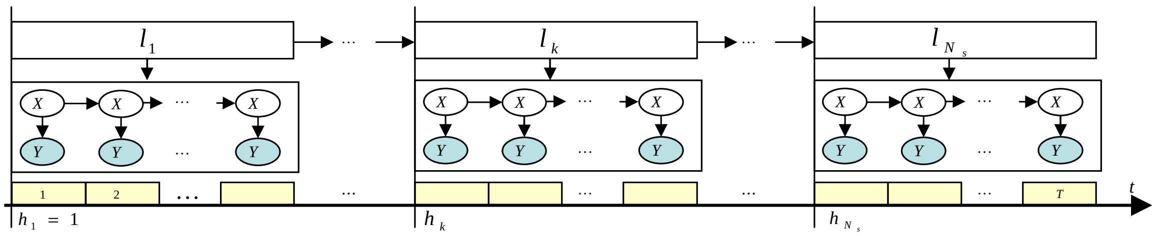


Figure 2: With the learnt motion texture, a motion sequence can be divided into multiple segments, labeled as l_k where $k = 1, ..., N_s$. Each segment may have a different length, and can be represented by one of the N_t ($N_t \leq N_s$) textons. In a texton, a local dynamic system with parameters (A, C, V, W) is used to describe the dynamics of state variables X_t and observations Y_t in a segment.

consist of repeated primitives such as spinning, hopping, kicking, and tiptoeing.

In our model, the basic element in motion texture is called a motion texton. A motion texton is represented by a linear dynamic system (LDS) that captures the dynamics shared by all instances of this texton in the motion sequence. The texton distribution, or relationship between motion textons, can be modeled by a transition matrix. Once the motion texture is learnt, it can be used for synthesizing novel motion sequences. The synthesized motion is statistically similar to, yet visually different from, the motion captured data. Our model enables users to synthesize and edit the motion at both the texton level and the distribution level.

The remainder of this paper is organized as follows. After reviewing related work in Section 2, we introduce the concept of motion texture in Section 3, and show how to learn motion textons and texton distributions in Section 4. The synthesis algorithms using motion texture are explained in Section 5. Applications of motion texture including motion synthesis and motion editing are shown in Section 6. We conclude our paper in Section 7.

2 Related Work

Motion texture. Based on the observation that these repetitive patterns of life-like motion exhibit inherent randomness, Pullen and Bregler [35] proposed a multi-level sampling approach to synthesize new motions that are statistically similar to the original. Similar to multi-resolution representations of texture [10] and movie texture [2], Pullen and Bregler modeled cyclic motions by multi-resolution signals. The term *motion texture* was originally used by Pullen and Bregler (and suggested by Perlin) as their project name [34]. Our motion texture model is completely different from theirs. We explicitly model not only local dynamics of those repetitive patterns, but also global dynamics on how these patterns are linked together.

Textons. The concept of texton was first proposed by Julesz [20] some twenty years ago, although a clear definition is still in debate. Malik *et al.* [27] used oriented filters, Guo *et al.* [17] used image templates that can be transformed geometrically and photometrically, and we use LDS as the motion texton. The concept of 2D texton has been extended to 3D texton by Leung and Malik [24] to represent images with varying lighting and viewing directions. Zhu [44] proposed to represent a 2D texture image with textons and layers of texton maps. However, extracting textons from a texture has proven to be challenging, as shown by rudimentary textons in [44, 17]. Although the patches used in the patch-based texture synthesis may be regarded as textons as well, the concept of texton map was not explicitly discussed in [12, 25].

Linear dynamic system. Modeling the motion texton with LDS is related to recent work on dynamic texture analysis and synthesis for video clips (e.g., video textures [37]). Soatto *et al.* [40] proposed that a dynamic texture can be modeled by an auto-regressive, moving average (ARMA) process with unknown input distribution. A similar approach was also proposed by Fitzgibbon [13] with an autoregressive (AR) model. These approaches model the temporal behavior as samples of an underlying continuous process, and are effective for spatially coherent textures. But they break down when the underlying dynamics are beyond the scope of a simple linear dynamics system. Furthermore, the system tends to converge into the stable state at which synthesis degrades to noise-driven textures. Bregler [7] also used second order dynamical systems to represent the dynamical categories (called *movemes*) of human motion. However, the movemes are only used to recognize simple human gait with two or three joint angles. Synthesizing realistic human motion is very difficult due to the high dimensionality of human body and the variability in human motion over time.

Modeling nonlinear dynamics. Many approaches have been proposed to model complex motion with multiple linear systems. It is, however, difficult to learn these linear systems along with the transitions. For instance, North *et al.* [28] learnt multiple classes of motions by combining EM (expectation-maximization) [11] and CONDENSATION [19]. Approximate inference methods had to be devised to learn a switched linear dynamic system (SLDS) [30, 29] because exact inference cannot be found. By discretizing state variables, a hidden Markov model (HMM) can be used to describe motion dynamics as well [6, 41, 14]. With an HMM, however, the motion primitives cannot be edited explicitly because they are represented by a number of hidden states. In our work, a two-level statistical model is necessary for modeling rich dynamics of human motion. The transition matrix implicitly models the nonlinear aspect of complex human motion by piecewise linear systems.

3 Motion Texture

3.1 A Two-level Statistical Model

We propose a two-level statistical model to represent character motion. In our model, there are N_t motion textons (or "textons" for short from now on) $T = \{T_1, T_2, ..., T_{N_t}\}$, represented by respective texton parameters $\Theta = \{\theta_1, \theta_2, ..., \theta_{N_t}\}$. Our objective is to divide the motion sequence into N_s segments, such that each segment can be represented by one of the N_t textons, as shown in Figure 2. Multiple segments could be represented by the same texton. Texton distribution, or the relationship between any pair of textons can be described by counting how many times a texton is switched to another.

In Figure 2, each segment is labeled as l_k, where $k = 1, \ldots, N_s$. The length of each segment may be different. Because all N_t textons are learnt from the entire sequence of N_s segments, $N_t \leq N_s$ must hold. Segment k starts from frame h_k and has a minimum length constraint $h_{k+1} - h_k \geq T_{min}$. We also define segment labels as $L = \{l_1, l_2, ..., l_{N_s}\}$, and segmentation points as $H = \{h_1, h_2, ..., h_{N_s}\}$.

Our two-level statistical model characterizes the dynamic and stochastic nature of the figure motion. First, we use an LDS to capture the local linear dynamics and a transition matrix to model the global non-linear dynamics. Second, we use textons to describe the repeated patterns in the stochastic process.

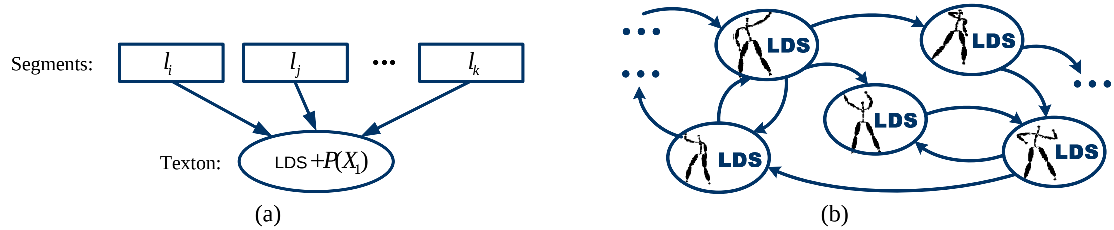


Figure 3: Motion texture is a two-level statistical model with textons and their distribution. (a) After learning, several segments may be labeled as the same texton. All these segments share the same dynamics. Each texton is represented by an LDS and the initial state distribution $P(X_1)$. (b) Texton distribution can be represented by a transition matrix.

3.2 Motion Texton

Each motion texton is represented by an LDS with the following state-space model:

$$\begin{cases} X_{t+1} = A_t X_t + V_t \\ Y_t = C_t X_t + W_t \end{cases} \tag{1}$$

where X_t is the hidden state variable, Y_t is the observation, and V_t and W_t are independent Gaussian noises at time t. Then the parameters of an LDS can be represented by $\theta = \{A, C, V, W\}$. Each texton should have at least T_{min} frames so that local dynamics can be captured. In our system, we model a complex human figure with its global position and 19 joints. Therefore, Y_t is a $60-$ dimensional vector because each joint is represented by 3 joint angles. In Section 6.1, we will show how we reparameterize the joint rotation angles with exponential maps [16, 23]. Eventually we can represent state variables X_t by a $12 \sim 15$-dimensional vector, depending on the data variance of each segment.

3.3 Distribution of Textons

We assume that the distribution of textons satisfies the first-order Markovian dynamics, which could be represented by a transition matrix

$$M_{ij} = P(l_k = j | l_{k-1} = i). \tag{2}$$

Such a transition matrix has been commonly used in HMMs [36] to indicate the likelihood of switching from one discrete state to another. Transition matrix has also been used in video texture [37], where transition points are found such that the video can be looped back to itself in a minimally obtrusive way. Unlike conventional HMMs, however, we use hybrid discrete (L and H) and continuous (X) state variables in our model. Switched linear dynamic systems (SLDS) [30, 29] also use a hybrid state variable, but model each frame with a mixture of LDS'. For synthesis, segment-based LDS models are desirable because they better capture the stochastic and dynamic nature of motion.

4 Learning Motion Texture

Given $Y_{1:T} = \{Y_1, Y_2, \ldots, Y_T\}$, or the observation Y_t from frame 1 to frame T, our system learns the model parameters $\{\Theta, M\}$ by finding a maximum likelihood (ML) solution

$$\{\hat{\Theta}, \hat{M}\} = \arg\max_{\{\Theta, M\}} P(Y_{1:T} | \Theta, M). \tag{3}$$

By using L and H, and applying the first-order Markovian property, the above equation can be rewritten as:

$$\begin{aligned} P(Y_{1:T}|\Theta, M) &= \sum_{L,H} P(Y_{1:T}|\Theta, M, L, H) \\ &= \sum_{L,H} \left[\prod_{j=1}^{N_s} P(Y_{h_j:h_{j+1}-1}|\theta_{l_j}) M_{l_j l_{j+1}} \right] \end{aligned} \tag{4}$$

where $M_{l_{N_s} l_{N_s+1}} = 1$. In Eq. 4, the first term is the likelihood of observation given the LDS model, the second term reflects the transition between two adjacent LDS'.

Considering L and H as hidden variables, we can use the EM [11] algorithm to solve the above maximum likelihood problem. The algorithm is looped until it converges to a local optimum.

- E-step: An inference process is used to obtain segmentation points H and segment labels L. Details are in Appendix A.
- M-step: Model parameters Θ are updated by fitting LDS'. Details are provided in Appendix B.

  The transition matrix M_{ij} is set by counting the labels of segments: $M_{ij} = \sum_{k=2}^{N_s} \delta(l_{k-1} = i)\delta(l_k = j)$. The matrix M is then normalized such that $\sum_{j=1}^{N_t} M_{ij} = 1$.

We take a greedy approach to incrementally initialize our model. First, we use T_{min} frames to fit an LDS i, and incrementally label the subsequent frames to segment i until the fitting error is above a given threshold. Then all existing LDS' (from 1 to i) learnt from all preceding segments (possibly more than i) are tested on the remaining unlabeled T_{min} frames, and the best-fit LDS is chosen. If the smallest fitting error exceeds the given threshold, i.e., none of those LDS' fits the observation well, we introduce a new LDS and repeat the above process until the entire sequence is processed.

4.1 Discussion

In the learning process, the user needs to specify a threshold of model fitting error. Once the threshold is given, the number of textons N_t is automatically determined by the above initialization step. The bigger the threshold, the longer the segments, and the fewer the number of textons. Model selection methods [3] such as BIC (Bayesian Information Criteria) or MDL (Minimum Description Length) can also be used to learn N_t automatically.

Another important parameter that the user needs to determine is T_{min}. T_{min} must be long enough to capture the local dynamics of motion. In our system, we have chosen T_{min} to be approximately one second, corresponding to most beats in the disco music of the dance sequence.

What we have obtained in the learning process are segment labels, segmentation points, textons, and texton distribution. For the purpose of synthesis, a texton should also include an initial state distribution $P(X_1)$. X_1 can be regarded as the initial or key poses of a texton. Because our dynamics model is second order, we use the first two frames x_1, x_2 of each segment to represent the key poses X_1. Figure 3 shows the two-level motion texture model. Since we may have labeled several segments for an LDS, we represent $P(X_1)$ in a nonparametric way. In other words, we simply keep all the starting poses X_1 of the segments which are labeled by the same LDS.

1. Initialization:

   Generate the first two poses $\{x_1, x_2\}$ by sampling the initial state distribution $P_i(X_1)$.

2. Iterate for $t = 3, 4, \ldots$

   (a) Draw samples from the noise term v_t,

   (b) Compute x_t by the dynamics model (Eq. 10),

   (c) Synthesize y_t by projecting x_t to the motion space (Eq. 11).

Figure 4: Texton synthesis by sampling noise.

1. Initialization:

   (a) Generate the first two poses $\{x_{i1}, x_{i2}\}$ by sampling the initial state distribution $P_i(X_1)$. $x_1 = x_{i1}, x_2 = x_{i2}$.

   (b) Similarly, generate $\{x_{j1}, x_{j2}\}$ from $P_j(X_1)$. Then the end constraints (x_{l-1}, x_l) are obtained by reprojecting $\{x_{j1}, x_{j2}\}$ (Eq. 7).

   (c) For $t = 2 \ldots (l-1)$, draw samples from the noise term v_t and form vector b (Eq. 9).

2. Synthesis:

   (a) Synthesize $x_{3:l-2}$ by solving Eq. 8.

   (b) Synthesize $y_{1:l}$ by projecting $x_{1:l}$ to the motion space.

Figure 5: Texton synthesis with constrained LDS.

5 Synthesis with Motion Texture

With the learnt motion texture, new motions can be synthesized. Moreover, we can edit the motion interactively, both at the texton level and at the distribution level.

5.1 A Two-step Synthesis Algorithm

Motion texture decouples global nonlinear dynamics (transition matrix) from local linear dynamics (textons). Accordingly, we develop a two-step approach to synthesize new motions. First, a texton path needs to be generated in the state space. A straightforward approach is to randomly sample the texton distribution, so that we can obtain an infinitely long texton sequence. A more interesting way is to allow the user to edit the texton path interactively. Given two textons and their associated key poses, for instance, our algorithm can generate a most likely texton sequence that passes through those key poses (see Section 5.2).

Once we have the texton sequence, the second step in synthesis is conceptually straightforward. In principle, given a texton and its key poses (first two frames to be exact), a motion sequence can be synthesized frame by frame with the learnt LDS and sampled noise. However, the prediction power of LDS decreases after some critical length of the sequence as LDS approaches its steady state. This is why we propose in Section 5.4 a constrained texton synthesis algorithm that preserves the same dynamics of the given texton, with two additional frames at the end of the synthesized segment. Because we can use the key poses of the texton next to the one we synthesize, a smooth motion transition between two neighboring textons can be achieved.

5.2 Texton Path Planning

Given two motion textons T_u and T_v, the goal of path planning is to find a single best path, $\bar{\Pi} = \{\bar{S}_1 \bar{S}_2 \ldots \bar{S}_n\}$ $(n \leq N_t)$, which starts at $\bar{S}_1 = T_u$ and ends at $\bar{S}_n = T_v$. Depending on the application, we propose two different approaches.

5.2.1 Finding the Lowest Cost Path

In this approach, we favor multiple "good" transitions from T_u to T_v. Since the transition matrix is defined on a first-order Markov chain, the best path is equivalent to

$$\begin{aligned}\bar{\Pi} &= \arg\max_{\Pi} P(S_1 S_2 \ldots S_n | S_1 = T_u, S_n = T_v, M) \\ &= \arg\max_{\Pi} P(T_u S_2) P(S_2 S_3) \cdots P(S_{n-1} T_v) \\ &= -\arg\min_{\Pi} (\log P(T_u S_2) + \log P(S_2 S_3) + \\ &\quad \ldots + \log P(S_{n-1} T_v)). \end{aligned} \tag{5}$$

If we consider each texton as a vertex, and the negative Log probability as the weight associated with the edge between two vertices, the transition matrix forms a weighted, directed graph G. Then the shortest path problem in Eq. 5 can be efficiently solved in $O(N^2)$ time by *Dijkstra*'s algorithm [9].

Since each texton is represented by a single LDS, the self-connecting vertices in graph G will appear at most once when we seek for the optimal texton path by *Dijkstra*'s algorithm. In order to enrich the synthesized motion, we further repeat each texton on the path according to its self-transition probability. Specifically, we randomly sample the transition probability $P(S_j|S_i)$ and repeat texton i until a different texton is sampled.

5.2.2 Specifying the Path Length

Due to limited training data, interpolation between two motion textons may result in a long path. An alternative way of texton path planning is to specify the path length. In this case, we need to trade-off cost and length. The best path between T_u and T_v with length L ($L < \left|\bar{\Pi}\right|$) can be found by a dynamic programming algorithm [36] in $O(LN_s^2)$ time.

5.3 Texton Synthesis by Sampling Noise

Once we have the texton path, we can generate a new motion sequence by synthesizing motion for all the textons. The first task is to synthesize motion for a single texton i. A simple but effective approach is to draw samples from the white noise v_t (see Appendix B) frame by frame. The key poses of texton i are the two starting poses, represented by x_1 and x_2. The final synthesis result can be generated by projecting x_t from the state space to motion space y_t. The algorithm is summarized in Figure 4.

In theory, an infinitely long sequence can be synthesized from the given texton after initialization and by sampling noise. However, the synthesized motion will inevitably depart from the original motion as time progresses, as shown by the difference between Figures 7(a) and 7(b). This behavior is due to the fact that LDS learns only locally consistent motion patterns. Moreover, the synthesis errors will accumulate (Figure 7(b)) as the hidden variable x_t propagates.

Noise-driven synthesis has been widely used in animation. For example, Perlin-noise [31] has been used for procedural animation [32]. Noises are also used for animating cyclic running motion [4], dynamic simulation [18] and animating by multi-level sampling [35]. Our approach is similar to Soatto's ARMA model [40] that can generate dynamic texture by sampling noise, when video frames instead of figure joints are considered.

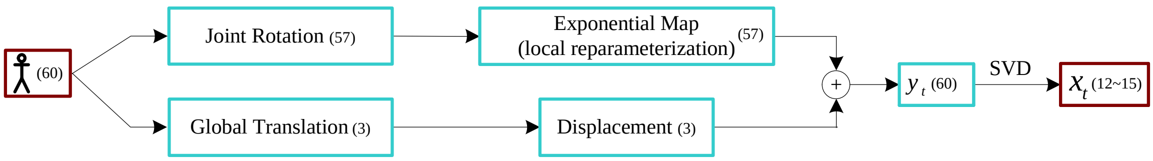
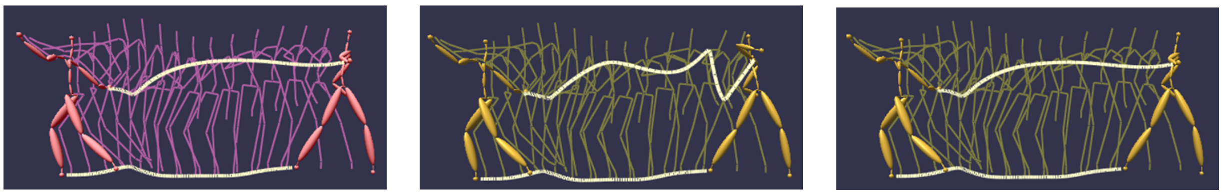


Figure 6: Representation of observation and state variables. We model the human figure with 19 joint angles and a global translation. After re-parameterizing the joint angles with exponential maps, we construct the observation variable Y_t with a 60-dimensional vector. The state variable X_t is only 12 $\sim$ 15-dimensional because it represents the subspace of Y_t after SVD.



(a) (b) (c)

Figure 7: Comparison between constrained and unconstrained synthesis with a single texton. (a) Original motion. (b) Synthesized motion without end constraints. The dynamics deviate from the original one as time progresses. (c) Synthesized motion with end constraints. The original dynamics are kept. The last two frames in (a) are chosen as the end constraints.

5.4 Texton Synthesis with Constrained LDS

We preserve texton dynamics by setting the end constraints of a synthesized segment. Since we have kept key poses (starting two poses) for each texton, we can incorporate those of the following texton (e.g., the right next one in the texton path) as hard constraints into the synthesis process. Let S_i and S_j be the two adjacent textons in the motion path, and $\{x_{i1}, x_{i2}\}$ and $\{x_{j1}, x_{j2}\}$ be the corresponding key poses. Rearranging the dynamic equation in Eq. 10 (Appendix B), we have

$$\begin{bmatrix} -A_2 & -A_1 & I \end{bmatrix} \begin{bmatrix} x_{t-1} \\ x_t \\ x_{t+1} \end{bmatrix} = D + Bn_t \tag{6}$$

where I is the identity matrix. In order to achieve a smooth transition from S_i to S_j, we set the following hard constraints

$$\begin{cases} x_1 = x_{i1}, \quad x_2 = x_{i2} \\ x_{l-1} = C_i^T C_j x_{j1}, \quad x_l = C_i^T C_j x_{j2} \end{cases} \tag{7}$$

Note that we need to re-project the end constraints since the observation model C_t (Eq. 1) is switched between motion textons. Then the in-between frames $x_{3:l-2} = [x_3, x_4, \cdots, x_{l-2}]^T$ (l is the length of the synthesized texton i) can be synthesized by solving a block-banded system of linear equations:

$$AX = b \tag{8}$$

where

$$A = \begin{bmatrix} I & & & & \\ -A_{i1} & I & & 0 & \\ -A_{i2} & -A_{i1} & I & & \\ & \ddots & \ddots & \ddots & \\ & & -A_{i2} & -A_{i1} & I \\ & 0 & & -A_{i2} & -A_{i1} \\ & & & & -A_{i2} \end{bmatrix}$$

$$b = \begin{bmatrix} A_{i1}x_{i2} + A_{i2}x_{i1} + D_i + B_i v_2 \\ A_{i2}x_2 + D_i + B_i v_3 \\ D_i + B_i v_4 \\ \vdots \\ D_i + B_i v_{l-3} \\ -x_{l-1} + D_i + B_i v_{l-2} \\ A_{i1}x_{l-1} - x_l + D_i + B_i v_{l-1} \end{bmatrix} \tag{9}$$

The algorithm is summarized in Figure 5.

A by-product of the constrained synthesis is smooth transition between two textons because the two starting poses of the second texton are guaranteed from the synthesis processes of both the first and the second textons.

6 Experimental Results

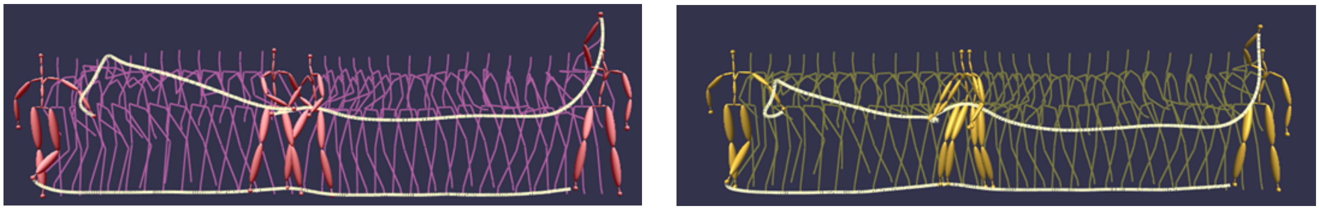
We have captured more than 20 minutes of dance motion of a professional dancer (performing mostly disco) at high frequency (60Hz) as our training data. We chose dance motion in our study because dancing is representative, complex and exhibits stochastic behavior with repeated patterns. It took approximately 4 hours to learn the motion texture of 49800 frames on an Intel Pentium IV 1.4GHz computer with 1G memory. A total of 246 textons are found after learning. The length of the textons ranges from 60 to 172 frames. Synthesizing a texton takes only 25ms to 35ms because it only involves solving a block-banded system of linear equations. Therefore, we can synthesize the character motion in real-time.

6.1 Dealing with High-dimensional Motion Data

To deal with the high dimensionality of human motion, one must simplify characters, as shown in [33]. K-means clustering and PCA are also used in [5, 41] to model the complex deformation of human body. However, all these methods failed to find the intrinsically low-dimensional subspace of the motion patterns embedded in the high-dimensional nonlinear data space.

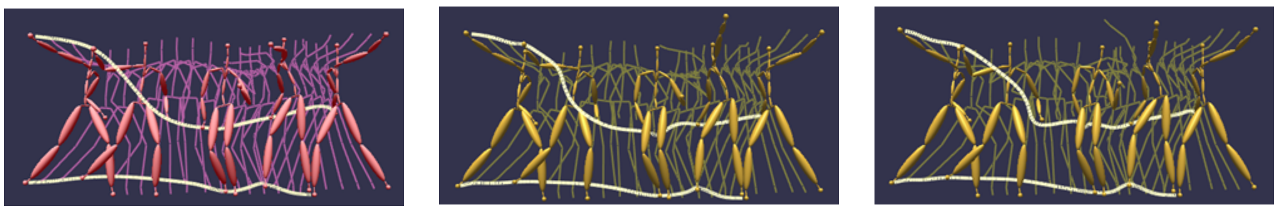
In our system, we model a complex human figure with 19 joints (57 rotation angles) plus its global position (3 translations). For global translation, we compute the displacement because we need to accumulate the subsequent translation to the previous frame for synthesis. For 3D rotation, we use exponential maps [16, 23] instead of Euler angles, unit quaternions [39], or rotation matrices. Using exponential maps [16, 23] for representing joint rotation angles is essential because they are locally linear. Singularities in exponential maps are avoided in our system since the rotation change at each step in a texton is small (obviously less than π).

The observation Y_t is thus a 60-dimensional vector, with a 57-dimensional locally reparameterized exponential map and a 3-dimensional translation displacement. The state variables X_t should be chosen to be of low dimensionality and highly correlated with the observation Y_t. Using a locally linear exponential map and the translation displacement, we are able to apply SVD on Y_t and represent X_t by 12 $\sim$ 15 (depending on the specific texton) most significant principal vectors of Y_t.



(a) (b)

Figure 8: Synthesis with two adjacent textons. (a) Synthesizing two adjacent textons independently results in a jump at the transition. (b) By setting the starting poses of the second texton as the end constraints of the first texton, we achieve smooth motion at the transition. Pay attention to the difference between the ending frame of the first texton and the starting frame of the second texton in both (a) and (b), shown as key frames in the middle.



(a) (b) (c)

Figure 9: (a) Original texton. (b) and (c) are synthesized textons perturbed by noise. Notice there are differences in the intermediate key poses of (b) and (c). The synthesized textons have the same content as the original texton, but different fine details.

6.2 Examples

We demonstrate our approach with a number of synthesis and editing examples, also shown in the accompanying videotape.

Constrained versus unconstrained texton synthesis. Figure 7 shows the results of synthesized motion from a single texton. In this 68-frame motion, the dancer changes her orientation from left to right, which represents a swing dance motion (Figure 7(a)). Given the learnt LDS and the starting two poses, we can synthesize a motion sequence by simply sampling noise (Figure 7(b)). The synthesized motion looks similar to the original dynamics, but gradually deviates as time progresses. By adding constraints of ending frames, we can ensure that the synthesized sequence (Figure 7(c)) has similar dynamics to the original. The least-squares solver took approximately 27ms to synthesize this sequence.

Synthesis with two adjacent textons. Figure 8 illustrates that a smooth transition can be achieved between two adjacent textons. Because we represent $P(X_1)$ in a non-parametric way, the starting pose of the second texton may be rather different from the ending pose of the first texton. Synthesizing these two textons separately results in a jump in the sequence (Figure 8(a)). We solve the problem of preserving smooth transitions by applying the starting poses of the second texton as the end constraints of the first texton (Figure 8(b)).

Noise-driven synthesis with different fine details. Figure 9 shows that by varying noise slightly, we can generate two different motions from the same texton. Yet the dynamics of two sequences look perceptually similar.

Extrapolating new dynamics. The learnt motion texton can be generalized to synthesize novel motion. In the motion shown in Figure 10(a), the dancer waves her left arm once. From the texton of 127 frames, we can synthesize a longer sequence of 188 frames that contains new motion (the dancer waves her arm twice, as shown in Figure 10(d)). Comparison to simple linear interpolation with the same constraints is shown in Figure 10(c).

Editing a texton. Motion texture can be edited at the texton level with precise pose changes. Figure 11 shows that after an intermediate frame in a texton sequence is edited, we obtain a new sequence similar to the original one, without the need for modifying any other frames.

Virtual Choreographer. Figure 1 shows that a sequence of 320 frames can be automatically generated given the starting and ending frames, and the learnt motion texture. This is similar to what a choreographer does. The synthesis has two steps: texton path generation, and motion synthesis from textons.

Ballroom demo. By randomly sampling motion texture, we can synthesize an infinitely long dance sequence. We present in the videotape a character dancing to the music.

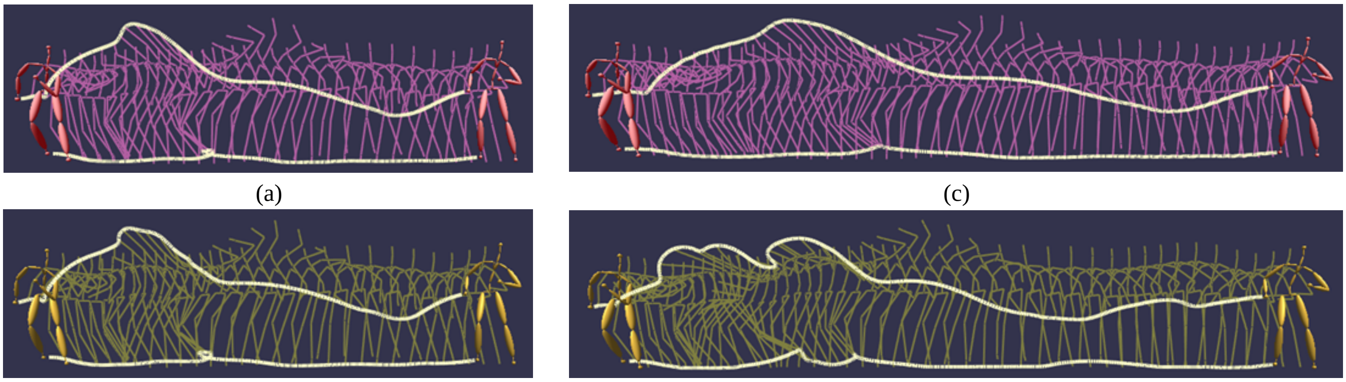
7 Discussion

Summary. In this paper, we have proposed a two-level statistical model, called motion texture, to capture complex motion dynamics. Motion texture is represented by a set of motion textons and their distribution. Local dynamics are captured by motion textons using linear dynamic systems, while global dynamics are modeled by switching between the textons. With the learnt motion texture, we can synthesize and edit motion easily.

Limitations. Although our approach has been effective on generating realistic and dynamic motion, there remain several areas for improvement. First, because we calculate the texton distribution by counting how many times a texton is switched to another, our approach is best suited for motions consisting of frequently repeated patterns such as disco dance. The synthesized motion may lack global variations when the training data is limited.

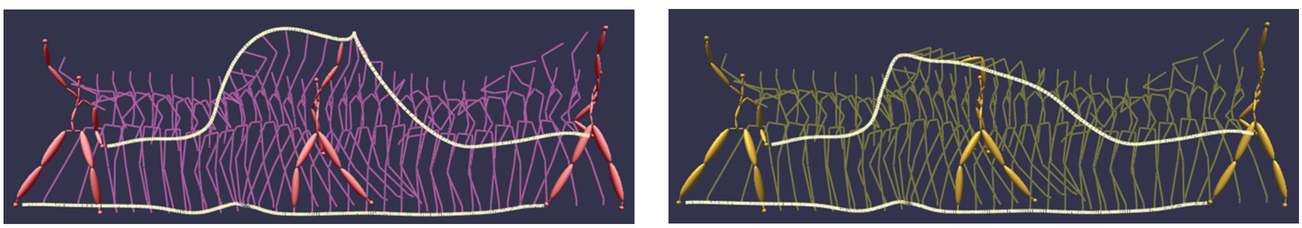
In order to enrich the synthesis variation, we have perturbed the dynamics model by Gaussian noise. We did not incorporate any physical model into the synthesis algorithm. So there is no guarantee that the synthesized motion is physically realistic in the absolute sense. However, since the LDS' preserve the motion dynamics very well, our algorithm captures the essential properties of the original motion.

Although our algorithm allows users to edit the motion at the texton level, the edited pose can not deviate from the original one too much (as shown in Fig 11). Otherwise the additional hard constraint may contaminate the synthesized texton. Another shortcoming of our algorithm is that interacting with environment objects is not taken into consideration. Nevertheless, our algorithm provides a good initialization of an animation sequence, which can be further improved by other animation tools.



(b) (d)

Figure 10: (a) Original texton (127 frames). (b) Synthesized texton by LDS (127 frames). (c) Linear time warping of the original texton (188 frames) has a slow motion effect. (d) Synthesized motion by LDS (188 frames). Our synthesis algorithm generalizes the learnt dynamics by extrapolating new motions. Notice the difference between the trajectories in (c) and (d). See the accompanying video for comparison.



(a) (b)

Figure 11: Editing a texton. (a) Original texton. (b) Synthesized texton by modifying an intermediate frame, but without changing any other frames. The specified frame is used as an additional hard constraint to synthesize the texton.

Acknowledgement

We would like to thank Ying-Qing Xu for the fruitful discussion on animation and help on motion data capture. We are grateful for the extremely valuable help on Siggraph video preparation provided by Yin Li and Gang Chen. Timely proofreading by Steve Lin is highly appreciated. Finally, the authors thank Siggraph reviewers for their constructive comments.

References

[1] O. Arikan and D. Forsyth. Interactive motion generation from examples. In *Proceedings of ACM SIGGRAPH 02*, 2002.

[2] Z. Bar-Joseph, R. El-Yaniv, D. Lischiniski, and M. Werman. Texture movies: Statistical learning of time-varying textures. *IEEE Transactions on Visualization and Computer Graphics*, 7(1):120–135, 2001.

[3] C. M. Bishop. *Neural Networks for Pattern Recognition*. Clarendon Press, Oxford, 1995.

[4] B. Bodenheimer, A. Shleyfman, and J. Hodgins. The effects of noise on the perception of animated human running. In *Computer Animation and Simulation '99, Eurographics Animation Workshop*, pages 53–63, 1999.

[5] R. Bowden. Learning statistical models of human motion. In *IEEE Workshop on Human Modeling, Analysis and Synthesis, CVPR*, 2000.

[6] M. Brand and A. Hertzmann. Style machines. In *Proceedings of ACM SIGGRAPH 00*, pages 183–192, 2000.

[7] C. Bregler. Learning and recognizing human dynamics in video sequences. In *Int. Conf. on Computer Vision and Pattern Rocognition*, pages 568–574, 1997.

[8] A. Bruderlin and L. Williams. Motion signal processing. In *Proceedings of ACM SIGGRAPH 95*, pages 97–104, 1995.

[9] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. The MIT Press, 1997.

[10] J. de Bonet. Multiresolution sampling procedure for analysis and synthesis of texture images. In *Proceedings of ACM SIGGRAPH 97*, pages 361–368, 1997.

[11] N. M. Dempster, A. P. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *J. R. Statist. Soc. B*, 39:185–197, 1977.

[12] A. Efros and W. Freeman. Image quilting for texture synthesis and transfer. In *Proceedings of ACM SIGGRAPH 01*, pages 341–346, 2001.

[13] A. W. Fitzgibbon. Stochastic rigidity: Image registration for nowhere-static scenes. In *IEEE Int. Conf. on Computer Vision*, pages 662–669, 2001.

[14] A. Galata, N. Johnson, and D. Hogg. Learning variable length Markov models of behaviour. *Computer Vision and Image Understanding*, 81(3):398–413, 2001.

[15] M. Gleicher. Retargetting motion to new characters. In *Proceedings of ACM SIGGRAPH 98*, pages 33–42, 1998.

[16] F. S. Grassia. Practical parameterization of rotations using the exponential map. *Journal of Graphics Tools*, 3(3):29–48, 1998.

[17] C. E. Guo, S. C. Zhu, and Y. N. Wu. Visual learning by integrating descriptive and generative methods. In *Int. Conf. Computer Vision*, 2001.

[18] J. K. Hodgins, W. L. Wooten, D. C. Brogan, and J. F. O'Brien. Animating human athletics. In *Proceedings of ACM SIGGRAPH 95*, pages 71–78, 1995.

[19] M. Isard and A. Blake. Condensation - conditional density propagation for visual tracking. *Int'l J. Computer Vision*, 28(1):5–28, 1998.

[20] B. Julesz. Textons, the elements of texture perception and their interactions. *Nature*, 290:91–97, 1981.

[21] L. Kovar, M. Gleicher, and F. Pighin. Motion graphs. In *Proceedings of ACM SIGGRAPH 02*, 2002.

[22] J. Lee, J. Chai, P. Reisma, and J. Hodgins. Interactive control of avartas animated with human motion data. In *Proceedings of ACM SIGGRAPH 02*, 2002.

[23] J. Lee and S. Y. Shin. A hierarchical approach to interactive motion editing for human-like figures. In *Proceedings of ACM SIGGRAPH 99*, pages 39–48, 1999.

[24] T. Leung and J. Malik. Recognizing surfaces using three-dimensional textons. In *Int. Conf. on Computer Vision*, 1999.

[25] L. Liang, C. Liu, Y. Xu, B. Guo, and H.-Y. Shum. Real-time texture synthesis by patch-based sampling. Technical Report MSR-TR-2001-40, Microsoft Research, 2001.

[26] L. Ljung. *System Identification - Theory for the User*. Prentice Hall, Upper Saddle River, N.J., 2nd edition, 1999.

[27] J. Malik, S. Belongie, J. Shi, and T. Leung. Textons, contours and regions: Cue integration in image segmentation. In *Int. Conf. Computer Vision*, pages

918–925, 1999.
[28] B. North, A. Blake, M. Isard, and J. Rittscher. Learning and classification of complex dynamics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(9):1016–1034, Sep. 2000.
[29] V. Pavlović, J. M. Rehg, T. J. Cham, and K. P. Murphy. A dynamic Bayesian network approach to figure tracking using learned dynamic models. In *IEEE International Conference on Computer Vision*, 1999.
[30] V. Pavlović, J. M. Rehg, and J. MacCormick. Impact of dynamic model learning on classification of human motion. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2000.
[31] K. Perlin. An image synthesizer. In *Computer Graphics (Proceedings of ACM SIGGRAPH 85)*, pages 287–296, 1985.
[32] K. Perlin and A. Goldberg. Improv: A system for scripting interactive actors in virtual worlds. In *Proceedings of ACM SIGGRAPH 96*, pages 205–216, 1996.
[33] Z. Popović and A. Witkin. Physically based motion transformation. In *Proceedings of ACM SIGGRAPH 99*, pages 11–20, 1999.
[34] K. Pullen and C. Bregler. http://graphics.stanford.edu/∼pullen/motion_texture.
[35] K. Pullen and C. Bregler. Motion capture assisted animation: Texturing and synthesis. In *Proceedings of ACM SIGGRAPH 02*, 2002.
[36] L. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–285, February 1989.
[37] A. Schödl, R. Szeliski, D. H. Salesin, and I. Essa. Video textures. In *Proceedings of ACM SIGGRAPH 00*, pages 489–498, 2000.
[38] H. J. Shin, J. Lee, M. Gleicher, and S. Y. Shin. Computer puppetry: An importance-based approach. *ACM Transactions on Graphics*, 20(2):67–94, April 2001.
[39] K. Shoemake. Animating rotation with quaternion curves. In *Computer Graphics (Proceedings of ACM SIGGRAPH 85)*, pages 245–254, 1985.
[40] S. Soatto, G. Doretto, and Y. N. Wu. Dynamic textures. In *IEEE International Conference on Computer Vision*, pages 439–446, 2001.
[41] L. M. Tanco and A. Hilton. Realistic synthesis of novel human movements from a database of motion capture examples. In *IEEE Workshop on Human Motion*, 2000.
[42] M. Unuma, K. Anjyo, and R. Takeuchi. Fourier principles for emotion-based human figure animation. In *Proceedings of ACM SIGGRAPH 95*, pages 91–96, 1995.
[43] A. Witkin and Z. Popović. Motion warping. In *Proceedings of ACM SIGGRAPH 95*, pages 105–108, 1995.
[44] S. C. Zhu, C. E. Guo, Y. Wu, and Y. Wang. What are textons. In *Proc. of European Conf. on Computer Vision (ECCV)*, 2002.

A. Inference algorithm for H and L

In inference we use current parameters of textons to recognize the motion sequence. More specifically, we divide the motion sequence into a sequence of concatenated segments and label each segment to a texton. The optimal solution could be derived by maximizing likelihood in Eq. 4. We can efficiently compute globally optimal segmentation points $H = \{h_2, ..., h_{N_s}\}$ and segment labels $L = \{l_1, ..., l_{N_s}\}$ by a dynamic programming algorithm. The details of the algorithm are given by the following.

We use $G_n(t)$ to represent the maximum value of the likelihood derived from dividing the motion sequence ending at frame t into a concatenated sequence of n segments. $E_n(t)$ and $F_n(t)$ are used to represent the label and the beginning point of the last segment of the sequence to achieve $G_n(t)$.

1. Initialization

$$G_1(t) = \max_{1 \le i \le N_t} P(Y_{1:t}|\theta_i),$$
$$E_1(t) = \arg\max_i P(Y_{1:t}|\theta_i), \qquad T_{min} \le t \le T.$$

2. Loop while $2 \le n \le \dfrac{T}{T_{min}}, \qquad n \cdot T_{min} \le t \le T$

$$G_n(t) = \max_{\substack{1 \le i \le N_t \\ (n-1)\cdot T_{min} < b \le (t - T_{min})}} [G_{n-1}(b-1) P(Y_{b:t}|\theta_i) M_{li}]$$
$$E_n(t), F_n(t) = \arg\max_{i,b} [G_{n-1}(b-1) P(Y_{b:t}|\theta_i) M_{li}]$$
$$\text{where} \quad l = E_{n-1}(b-1).$$

3. Final solution

$$G(T) = \max_{1 \le n \le \frac{T}{T_{min}}} G_n(T).$$
$$N_s = \arg\max_n G_n(T).$$

4. Backtrack the segment points and labels

$$h_{N_s+1} = T+1, \quad l_{N_s} = E_{N_s}(T), \quad h_1 = 1$$
$$h_n = F_n(h_{n+1}-1),\ l_{n-1} = E_{n-1}(h_n - 1),\ \ N_s \ge n > 1.$$

For T frames and N_t textons, the complexity is $O(N_t T^2)$.

B. Fitting an LDS

Given a segment of an observation sequence, we can learn the model parameters of a linear dynamic system (LDS). In order to capture richer dynamics (velocity and acceleration), we use a second-order linear dynamic system:

$$\textit{Dynamics Model:} \quad x_{t+1} = A_1 x_t + A_2 x_{t-1} + D + B v_t \tag{10}$$
$$\textit{Observation Model:} \quad y_t = C_t x_t + W_t \tag{11}$$

where $v_t \sim N(0,1)$, $W_t \sim N(0,R)$, and R is the covariance matrix. Note that this model could also be written in the standard form of Eq. 1 by setting $X_t = \begin{bmatrix} x_t \\ x_{t-1} \end{bmatrix}$, $A_t = \begin{bmatrix} A_1 & A_2 \\ I & 0 \end{bmatrix}$, $V_t \sim N(\begin{bmatrix} D \\ 0 \end{bmatrix}, \begin{bmatrix} B^T B & 0 \\ 0 & 0 \end{bmatrix})$.

For any linear dynamic system, the choice of its model parameters is not unique. For instance, by choosing $\hat{X} = TX$, $\hat{A} = TAT^{-1}$, $\hat{C} = CT^{-1}$, in which T is an arbitrary full rank square matrix, we will obtain another linear dynamic system which generates exactly the same observation. In order to find a unique solution, canonical model realizations need to be considered, as suggested in [40]. And a closed form approximated estimation of the model parameters could be derived as follows [26]:

1. *Observation Model* [1]. We calculate the SVD of the observation sequence $Y_{1:T}$, $[U, S, V] = SVD(Y_{1:T})$, and set:

$$C = U, \quad X_{1:T} = SV^T. \tag{12}$$

2. *Dynamics Model.* The maximum likelihood estimation of A_1, A_2, D, B is given by:

$$\begin{aligned} [A_1, A_2] &= [R_{0,0}, R_{0,1}] \cdot \begin{bmatrix} R_{1,1} & R_{2,1} \\ R_{1,2} & R_{2,2} \end{bmatrix}^{-1} \\ D &= \frac{1}{T-2}\left(Q_0 - \sum_{i=1}^{2} A_i Q_i\right) \\ BB^T &= \frac{1}{T-2}\left(R_{0,0} - \sum_{i=1}^{2} A_i R_{i,0}\right) \end{aligned} \tag{13}$$

where $Q_i, R_{i,j}$ is:

$$Q_i = \sum_{t=3}^{T} X_{t-i}$$
$$R_{i,j} = \sum_{t=3}^{T} X_{t-i}(X_{t-j})^T - \frac{1}{T-2} Q_i Q_j^T.$$

In learning, we may need to fit an LDS on several segments. For this purpose, we can concatenate such segments into a sequence and apply the same algorithm except drop some terms on boundaries of the segments when Q_i and R_{ij} are calculated.

[1] We do not incorporate W_t in the learning process because the observation noise is not used for synthesis.