**UCLA-CSD-TR#200032**

# Dynamic Textures

Gianfranco Doretto[†] Prabhakar Pundir[†] Ying Nian Wu[‡] Stefano Soatto[†∗]

∗ Department of Electrical Engineering, Washington University, St.Louis - MO 63130
† Department of Computer Science, UCLA, Los Angeles - CA 90095
‡ Department of Statistics, UCLA, Los Angeles - CA 90095

**Keywords:** textures, dynamic scene analysis, 3D textures, minimum description length, image priors, image compression, generative model, prediction error methods, ARMA regression, system identification, learning, E-M algorithm

## Abstract

*Dynamic textures are sequences of images of moving scenes that exhibit certain stationariety properties in space and time; these include sea-waves, smoke, foliage, whirlwinds etc. We present a novel characterization of dynamic texture that poses the problems of modeling, learning, recognizing and synthesizing dynamic textures on a firm analytical footing. We borrow tools from stochastic realization theory to capture the "essence" of dynamic textures; we do so by learning models that are optimal in the sense of maximum likelihood or minimum prediction error variance. Once learned, therefore, a model has predictive power and can be used for extrapolating synthetic sequences to infinite length with negligible computational cost. We present experimental evidence that, within our framework, even low dimensional models can capture very complex visual phenomena.*

## 1. Introduction

Consider a sequence of images of a moving scene. Each image is an array of positive numbers that depend upon the shape, pose and motion of the scene as well as upon its material properties (reflectance distribution) and on the light distribution of the environment. It is well known that the joint reconstruction of *photometry* and *geometry* is an intrinsically ill-posed problem: from any (finite) number of images it is not possible to uniquely recover all unknowns (shape, motion, reflectance and light distribution). Traditional approaches to scene reconstruction rely on fixing *some* of the unknowns either by

virtue of assumption or by restricting the experimental conditions, while estimating the others. [1]

However, such assumptions can never be validated from visual data, since it is always possible to construct scenes with different photometry and geometry that give rise to the same images [2]. The ill-posedness of the most general visual reconstruction problem and the remarkable consistency in the solution as performed by the human visual system reveals the importance of priors for images [42]. They are necessary to fix the arbitrary degrees of freedom and render the problem well-posed [21]. In general, one can use the extra degrees of freedom to the benefit of the application at hand: one can fix photometry and estimate geometry (e.g. in robotics), or fix geometry and estimate photometry (e.g. in image-based rendering), or recover a combination of the two that satisfies some additional optimality criterion, for instance the minimum description length of the sequence of video data [33].

Given this arbitrariness in the reconstruction and interpretation of visual scenes, it is clear that there is no notion of a true interpretation, and the criterion for *correctness* is somewhat arbitrary. In the case of humans, the interpretation that leads to a correct Euclidean reconstruction (that can be verified by other sensory modalities, such as touch) has obvious appeal, but there is no way in which the correct Euclidean interpretation can be retrieved solely from the visual signal.

In this paper we will analyze sequences of images of moving scenes solely as visual signals. "Interpreting" and "understanding" a signal amounts to inferring a stochastic model that generates it. The "goodness" of the model can be measured in terms of the total likelihood of the measurements or in terms of its predicting power: a model should be able to give accurate predictions of the future signals (akin to so-called prediction error methods, *PEM*, in system identification). Such a model will involve a combination of photometry, geometry and dynamics and will be designed for maximum likelihood or minimal prediction error variance. Notice that we will not require that the reconstructed photometry or geometry be *correct* (in the Euclidean sense), for that is intrinsically impossible without involving non-verifiable prior assumptions. But the model must be capable of predicting future measurements. In a sense, we look for an "explanation" of the image data, that allows us to recreate and extrapolate it. It can therefore be thought of as the compressed version or the "essence" of the sequence of images.

The intuitive notion of texture is an image with certain spatially invariant statistics, e.g. ground, tiles, straw etc. For moving images, we are interested in capturing scenes that contain foliage moving in wind, waterfalls, smoke, sea waves, cheering crowds and the like. We begin by a short survey of related work and the new contributions this paper makes. In the context of what we discussed in the previous paragraph, we will concentrate on a precise subset of all possible photometries and geometries, by then giving an operational definition of what we call **dynamic textures**, and inferring a dynamical model that generates the scene.

## 1.1. Prior related work

Statistical inference for analyzing and understanding general images has been extensively used for the last two decades [27]. The statistical characterization of textures was pioneered by Julesz four decades

[1] For instance, in stereo and structure from motion one assumes that (most of) the scene has Lambertian reflection properties, and exploits such an assumption to establish correspondence and estimate shape. Similarly, in shape from shading one assumes constant albedo and exploits changes in irradiance to recover shape.

[2] For example, a sequence of images of the sea at sunset could have been originated by a very complex and dynamic shape (the surface of the sea) with constant reflection properties (homogeneous material, water), and also by a very simple shape (e.g. the plane of the television monitor) with a non-homogeneous radiance (the televised spatio-temporal signal). Similarly, the appearance of a moving Lambertian cube can be mimicked by a spherical mirror projecting a light distribution to match the albedo of the cube.

back [19].

There has been extensive work in the area of 2D texture analysis, recognition and synthesis. Most of the approaches use statistical models to understand 2D textures [17, 42, 30, 31, 9, 28, 8, 16] while few others rely on deterministic structural models [12, 41]. Another distinction is that some work directly on the pixel values while others project the image intensity function onto a basis of functions[3].

There have been many physically based algorithms which target specific dynamic textures [11, 13, 29, 38]. Some simulations have also done by particle systems [32, 37]. These methods are computationally intensive, customized for particular textures and allow no parameters to control the simulation once formulated.

There has been comparatively little work in the specific area of dynamic textures or texture movies. Schödl *et al.* [34] address the problem by finding transition points in the original video sequence where the video can be looped back on itself in a minimally obtrusive way. The process involves morphing techniques to smooth out visual discontinuities. Levoy and Wei [41] have also suggested extending their approach to temporal textures by creating a repeatable sequence. The approach is clearly very restrictive and obtains a quick solution for a small subset of problems with little or no understanding of the texture.

Bar-Joseph [3, 4] uses multi resolution analysis (MRA) tree merging for the synthesis and merging of 2D textures and extends the idea to *dynamic textures*. For 2D textures new MRA trees are constructed by merging MRA trees obtained from the input; the algorithm is different from De Bonet's [9] algorithm that operates on a single texture sample. The idea is extended to dynamic textures by constructing MRA trees using a 3D wavelet transform. Impressive results were obtained for the 2D case, but only a finite length sequence is synthesized after computing the combined MRA tree. Our approach captures the essence of a dynamic texture in some parameters and an infinite length sequence can be generated in real-time using the parameters computed off-line.

Szummer and Picard's work [40, 39] on temporal texture modeling uses a similar approach towards capturing dynamic textures. They use the spatio-temporal auto-regressive model (STAR), which imposes a neighborhood causality constraint even for the spatial domain. This restricts the textures that can be captured to a large extent. The STAR model fails to capture rotation, acceleration and other simple non translational motions. It works directly on the pixel intensities rather than a smaller dimensional representation of the image. We incorporate spatial correlation without imposing causal restrictions, as would be clear in the coming sections, and can capture more complex motion. (e.g. rotational motion, on which the STAR model is ineffective, taken from the same dataset [40])

## 1.2. Contributions of this work

This work presents several novel aspects in the field of dynamic (or time-varying) textures. On the *representation*, we present a novel definition of dynamic texture that is general (it captures a wide variety of image dynamics) and precise (it allows making analytical statements and drawing from the rich literature on system identification). On *learning*, we propose two criteria: total likelihood or prediction error. The estimation algorithms we use in both cases are off-the-shelf, although more complex input distribution models call for interesting extensions. On *recognition*, we propose a rigorous notion of metric in the space of dynamic textures, which is derived from the theory of subspace identification. This shows promising results towards a general theory of recognition for dynamic textures. On *synthesis*, we show that even the simplest model (first-order ARMA with white IID Gaussian input) captures a wide range

[3] Most common methods use Gabor filters [18, 5] and steerable filters [14, 36, 15, 35, 17]. One could also infer and choose the best filters as part of the learning process [26, 43].

of textures. Our algorithm is simple to implement, efficient to learn and fast to simulate, therefore allows one to generate infinitely long sequences from short input textures movies and to control parameters that have intuitive physical meaning.

We would like to emphasize that this is *not* a paper in computer graphics. The capability to effectively synthesize dynamic textures comes as a byproduct of the model learned for the textures. What we are interested in modeling in this paper is the intrinsic spatio-temporal structure of a class of images that present some sort of spatio-temporal stationariety. The model is not physics-bases, but is designed so as to allow generalization (predictive power) in space and time.

We pose the problem of dynamic texture modeling, learning, recognition and synthesis in a firm analytic footing. Although in our experiments we only consider simple choices of input distributions, more general classes can be easily taken into account by using particle filtering techniques and more general classes of filter banks. Some of these results may be useful for image compression and for image-based rendering and synthesis of image sequences.

# 2. Representation of dynamic textures

What is a suitable definition of texture? for a single image, one can say it is a texture if it is a realization from a stationary stochastic process with spatially invariant statistics [42]. This definition captures the intuitive notion of texture discussed earlier. For a sequence of images (temporal texture), individual images are clearly not independent realizations from a stationary distribution, for there is a temporal coherence intrinsic in the process that needs to be captured. The underlying assumption, therefore, is that individual images are realizations of a dynamical system driven by a stationary distribution. Although the output of the system (measured images) is not stationary, the input is. We now make this concept precise as an operative definition of dynamic texture.

## 2.1. Definition of dynamic texture

Let $\{I(t)\}_{t=1...M}$ be a sequence of images, or a subset of it. Suppose that at each instant of time $t$ we can measure a noisy version of the image, $y(t) = I(t) + n(t)$ where $n(t)$ is an independent and identically distributed (IID) sequence drawn from a known distribution $p_n(\cdot)$ resulting in a positive measured sequence $\{y(t)\}_{t=1...M}$[4]. We say that *the sequence $\{I(t)\}$ is a (linear) dynamic texture of order $k$* if there exists a set of spatial filters $\phi_\alpha,\ \alpha = 1 \dots N$ and a stationary distribution $q(\cdot)$ with spatially invariant statistics such that, calling $x(t) \doteq \phi(I(t))$ we have $x(t) = \sum_{i=1}^{k} A_i x(t-i) + Bv(t)$, with $v(t)$ an IID realization from the density $q(\cdot)$, for some choice of matrices $A_1, \dots, A_k, B$ and initial condition $x(0) = x_0$. Therefore, a dynamic texture is associated to an auto-regressive, moving average process with unknown input (ARMAUX)

$$\begin{cases} x(t) = A_1 x(t-1) + \ldots + A_k x(t-k) + Bv(t) \\ y(t) = \phi(x(t)) + n(t) \end{cases} \tag{1}$$

with $x(0) = x_0,\ v(t) \overset{IID}{\sim} q(\cdot)$ unknown, $n(t) \overset{IID}{\sim} p_n(\cdot)$ given and $I(t) = \phi(x(t))$. Without loss of generality, we can assume $k = 1$ since we can augment the state of the above model to be $\bar{x}(t) \doteq [x(t)^T\ x(t-1)^T \dots x(t-k)^T]^T$, and we can obviously extend the definition to an arbitrary non-linear

[4]This distribution can be inferred from the physics of the imaging device. For CCD sensors, for instance, a good approximation is a Poisson distribution with intensity related to the average photon count.

(but finite-dimensional) model of the form $x(t+1) = f(x(t), v(t))$, which brings us into the realm of *non-linear dynamic textures*.

### 2.2. Filters and dimensionality reduction

The definition of dynamic texture above entails a choice of filters $\phi_\alpha,\ \alpha = 1 \dots N$. These filters are also inferred as part of the learning process for a given dynamic texture.

There are several criteria for choosing a suitable class of filters, ranging from biological motivations to computational efficiency. In the trivial case, we can take $\phi$ to be the identity, and therefore look at the dynamics of individual pixels $x(t) = I(t)$ in (1). We view the choice of filters as a dimensionality reduction step, and seek for a decomposition of the image in the simple (linear) form

$$I(t) = \sum_{i=1}^{N} x_i(t)\theta_i \doteq Cx(t) \tag{2}$$

where $C = [\theta_1, \dots, \theta_N]$ and $\{\theta\}$ can be an ortonormal basis of $\mathcal{L}^2$, a set of principal components, or a wavelet filter bank. In this work we choose simple principal components, estimated from the given collection of images using the singular value decomposition (SVD).

An alternative non-linear choice of filters can be obtained by processing the image with a filter bank, and representing it with the collection of positions of the maximal response in the passband [25].

## 3. Learning dynamic textures

Given a sequence of noisy images $\{y(t)\}_{t=1\dots M}$, learning the dynamic texture amounts to inferring the dynamics $A_1, \dots, A_k, B$ and the distribution of the input $q(\cdot)$ in the model (1). This is a form of *stochastic realization problem* [22], where one is to infer a dynamical model from a time series. However, in the literature of dynamical systems, it is commonly assumed that the distribution of the input is known. In the context of dynamic textures, we have the additional complication of having to infer the distribution of the input along with the dynamical model. For the sake of simplicity, we will restrict our attention to first-order linear dynamic textures. The learning, or system identification, problem can then be posed as follows, for instance in the maximum likelihood sense.

### 3.1. Maximum likelihood learning

The maximum-likelihood formulation of the dynamic texture learning problem can be posed as follows:

$$\begin{aligned}
&\text{given } x(0),\ y(1), \dots, y(M),\ \text{find} \\
&\hat{A}, \hat{B}, \hat{q}(\cdot) = \arg\max_{A,B,q} \log p(y(1), \dots, y(M)) \\
&\text{such that } \begin{cases} x(t+1) = Ax(t) + Bv(t) \\ y(t) = Cx(t) + n(t) \end{cases} \\
&\text{and } v(t) \overset{IID}{\sim} q.
\end{aligned} \tag{3}$$

The inference method depends crucially upon what type of representation we choose for $q$. Note that the above inference problem involves the hidden variables $x(t)$ multiplying the unknown parameter $A$ and realizations $v(t)$ multiplying the unknown parameter $B$, and is therefore intrinsically non-linear even

if the original state model is linear. We will use, as usual, iterative techniques that alternate between estimating (sufficient statistic of) the conditional density of the state and maximizing the likelihood with respect to the unknown parameters, in a fashion similar to the expectation-maximization (EM) algorithm [10]. In order for such iterative techniques to converge to a unique minimum, *canonical realizations* need to be considered, corresponding to particular forms for the matrices $A$ and $B$. A simple example is the so-called "controllable canonical form" [20], that however results in poor numerical conditioning of the algorithms. Balanced realizations of various sorts can be considered, of the kind described in [2]. In the experimental section we will avoid this issue by considering convergence to any realization that maximizes the joint likelihood of the output.

The use of EM algorithms to learn parameters of dynamical models is standard and we therefore refer the reader to [24] for details.

## 3.2. Prediction error methods

As an alternative to maximum likelihood, one can consider estimating the model that results in the least prediction error, for instance in the sense of mean square. Let $\hat{x}(t+1|t) \doteq E[x(t+1)|y(1), \ldots, y(t)]$ be the best one-step predictor, that depends upon the unknown parameters $A, B, q$. One can then pose the problem in a causal fashion as

$$\begin{aligned} &\hat{A}, \hat{B}, \hat{q} \doteq \arg\min y(t+1) - C\hat{x}(t+1|t) \\ &\text{subject to } (3) \end{aligned} \tag{4}$$

unfortunately, explicit forms of the one-step predictors are available only under restricted assumptions, for instance linear models driven by white Gaussian noise [24]. In the experimental section we do consider one such model, however, we repeat results obtained only for the case of ML learning. For details the reader is again referred to [24].

## 3.3. Representation of the driving distribution

So far we have managed to defer addressing the fact that the unknown driving distribution belongs, in principle, to an infinite-dimensional space, and therefore something needs to be said about how this issue is dealt with algorithmically.

We see three ways to approach this problem. One is to transform this into a finite-dimensional inference problem by choosing a parametric class of densities. This is done in the experimental section, where we have postulated that the unknown driving density belongs to a finite-dimensional parameterizations of a class of exponential densities, and therefore the inference problem is reduced to a finite-dimensional optimization. The exponential class is quite rich and it includes, in particular, multi-modal as well as skewed densities, although with experiments we show that even a single guassian models allows achieving good results.

The second alternative is to represent the density $q$ via a finite number of fair samples drawn from it; the model (1) can be used to represent the evolution of the conditional density of the state given the measurements (the discrete-time equivalent of Fokker-Planck's operator), and the density is evolved by updating the samples so that they remain fair realization of the conditional density as time evolves. Algorithms of this sort are called "particle filters" [23], and in particular the CONDENSATION filter [6] is the best known instance in the Computer Vision community. Although we believe that this avenue is very promising, we have not pursued it.

The third alternative is to treat (3) as a semi-parametric statistical problem, where one of the parameters ($q$) lives in the infinite-dimensional manifold of probability densities that satisfy certain regularity conditions, endowed with a Riemannian metric (corresponding to Fisher's Information matrix), and designing gradient descent algorithms with respect to the natural connection, as it has been done in the context of independent component analysis (ICA) by Amari and Cardoso [1]. This avenue is considerably more laborious, albeit admittedly more principled, and we are therefore not considering it in this study.

### 3.4. Recognition

According to our definition in section 2.1, each texture is characterized by an ARMAUX model. Therefore, in order to compare textures, we need to first define a base measure in the space of linear dynamical systems, and then possibly to characterize probability distributions in that space.

Defining an appropriate base measure in the space of ARMAUX models is not trivial, since each model entails a combination of an input density and state and output transition matrices. However, if we restrict our attention to models having input density in the same class, then recent results from the theory of subspace identification provide guidance on efficient ways to compute the distance between models. In particular, [7] defines subspace angles between models, and provides efficient algorithms to compute them. In the experimental section we discuss how we have computed distances between different realizations of the textures and show how similar textures cluster together in some space.

## 4. Experiments: implementation

In our experiments we show results on three sequences from the *MIT temporal texture database* (courtesy of Martin Szummer), namely `steam`, `river` and `spiraling-water`. The sequences are $140$, $120$ and $120$ frames long respectively, while the dimensions of the frames are $96 \times 176$ for the `steam` and $115 \times 170$ for the `river` and the `spiraling-water`.

In the first-order linear dynamic texture model we assume that $B$ is an identity matrix. The evolution of the model is centered around the average frame of the original sequence, $\mu = E[y(t)]$, so that each frame can be represented in the domain of the principal components $U \doteq [u_1, \ldots, u_N]$, as

$$x(t) = U^T(y(t) - \mu). \tag{5}$$

Thus, in the model (3), $x(t)$ become a zero mean random vector as well as $v(t)$. We choose $v(t)$ lo lie in the parametric class of multivariate Gaussian random vectors with zero-mean, independent components and covariance matrix $\sigma$, therefore we have $v(t) \overset{IID}{\sim} \mathcal{N}(0, \sigma)$.

After the parameter estimation, the generation process of the images is given, in accordance with (5) and (3), by

$$\begin{cases} x(t) = \hat{A}x(t-1) + v(t) \\ I(t) = Ux(t) + \hat{\mu} \end{cases} \tag{6}$$

where $v(t) \overset{IID}{\sim} \mathcal{N}(0, \hat{\sigma})$. The learning process therefore, consists of estimating $\hat{\mu}$ as the average frame of the training sequence, the principal components $U$ via SVD and the ML-estimation of the parameters $\hat{A}$ and $\hat{\sigma}$ via gradient descent.

## 4.1. Synthesis

Figure 1 shows that an "infinite length" texture sequence can be synthesized from a typically "short" input sequence by just drawing samples from $v(t)$. The initial frame, represented by $x(0)$, is chosen randomly from the original sequence, it could also be drawn from a distribution learnt from the sequence using 2D texture synthesis techniques, e.g. [42]. The frames belongs to the `spiraling-water` sequence. From a 120 frames training sequence a 1000 frames synthesized sequence[5] has been generated using $N = 50$ principal components. This sequence has been showed in [40] as a test-bed example to point out the limitations of the STAR model in capturing non-translational motion. Since our model incorporates spatial correlation without imposing any spatial causal restrictions, it can capture much more complex motion.

The first row of Figures 4 and 5 shows a few frames of the original `steam` and `river` sequence respectively. The second row shows a section of frames from the "infinitely long" synthesized texture sequence[5] (N=20 principal components).

## 4.2. Performance assessment

As explained in section 3.2 we use ML by fixing the number $N$ of principal components and learning the parameters for different lengths, $M$, of the same training sequence; in order to cross-verify our models, we test this with PEM. For each length, $M$, we predict the frame $M + 1$ and compute the prediction error per pixel in gray levels. The results are shown in Figure 2. The average error per pixel decreases and becomes stable after the $80^{th}$ frame. The standard deviation has the same shape of the average error. Thus, the predicting power of the model grows with the length of the training sequence and so does its ability to capture the spatial-temporal dynamics. Furthermore, after some "critical" length of the sequence, there is no improvement in the predicting power: the spatial-temporal dynamics has been captured and the use of a longer sequence does not provide any additional information. Figure 2 shows that this happen after the $80^{th}$ frame for the sequence `steam`.

The graph is important in analyzing the information content of the input sequence. If the sequence terminates before the graph stabilizes, it implies "insufficient data" to model the texture comprehensively. Moreover data after the critical point is "redundant". This measure or technique can be used as a benchmark to compare different temporal texture for "information content" for a particular model or to compare different texture synthesis models for a particular temporal texture. In the latter case if a model stabilizes after reading *fewer* input frames, doesn't necessarily imply that it is richer or more efficient. It may be disregarding some information in the input. Therefore, its a weigh up between *when* (i.e. the "critical point") and *where* (i.e. along the ordinate axis) the graph stabilizes that tells us, how good a texture synthesis model is.

## 4.3. Extrapolating sequences

Another important parameter to compare various texture synthesis models is the time it takes to synthesize.

It is well established that models using Gibbs sampling [42] and other sampling methods to draw sample from complex distributions are computationally intensive. Moreover, there is always an uncertainty whether the samples have converged. Deterministic methods to extend and extrapolate sequences have

[5]The training and the synthesized sequences are available online at `http://www.vision.cs.ucla.edu/dynamic-textures/movies.html`.

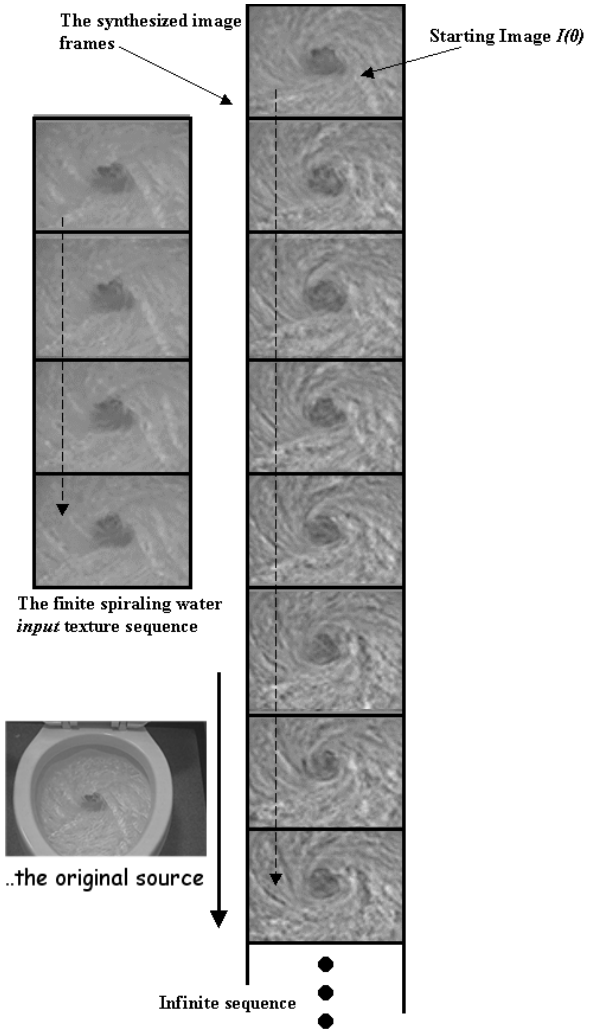


Figure 1: *The figure shows how an "infinite length" texture sequence is synthesized from a typically "short" input texture sequence by just drawing samples from $v(t)$. The data set used comes from the* MIT temporal texture database. *The particular structure of this sequence (*`spiraling-water` *synthesized using $N = 50$ principal components), is amongst the ones that cannot be captured by the STAR model [40].*

to go back and query the input texture in one way or the other to obtain information that generates the next [12, 41] [6]. In our model, once the learning phase is over, generation is much faster because it is pure simulation and there is no need to refer to the input texture and the distributions are straightforward to sample from. Moreover, we can even control the size of parameters to obtain a particular synthesis speed and change the model parameters (e.g. the eigenvalues of $\hat{A}$) to manipulate the original dynamics.

The generation problem was not as critical in the 2D texture domain, as we usually did have a size estimate (finite) to extend the input to and there were no real time constrains. In addressing the problem of generating "infinitely" long temporal sequences this is one of the key issues.

We would now discuss another facet of temporal texture, texture differentiation or recognition.

[6] In [41] for each new pixel a search is conducted for a similar neighborhood pattern in the original texture.

### 4.4. Recognition

The motivation of the recognition problem for dynamic textures was discussed in section 3.4. We use the *Kullback-Leibler* distance (K-L distance) to compute the discrepancy between different dynamic textures. The problem is formalized as follows.

Let $I(t)$, $t = 0, 1, \ldots$ be an infinite-long sequence of images. This can be modeled as a stochastic process which takes values in a subset of $\mathbb{R}^{m \times n}$ for an $m \times n$ dimensional image. Let $\bar{I}^T \doteq (I(1), I(2), \ldots, I(T))$ be a sequence of images and let $p(\bar{I}^T)$ be the corresponding probability density function (p.d.f.). The p.d.f. $p(\bar{I}^T)$ is completely determined by the parameters that define the model (1). Now, let $p_1$ and $p_2$ be two p.d.f.s that belongs to two different dynamic textures. The K-L distance, $I(p_1 \| p_2)$, between $p_1$ and $p_2$ is defined as $I(p_1 \| p_2) \doteq \lim_{T \to \infty} I^T(p_1 \| p_2)$ where $I^T(p_1 \| p_2) = \frac{1}{T} E_{p_1}[\log(p_1(\bar{I}_1^T)/p_2(\bar{I}_1^T))]$ and $E_{p_1}[\cdot]$ is the expectation taken with respect to $p_1$.

In Figure 3 we display the distance, the quantity $I^T(p_1 \| p_2)$, between different dynamic textures plotted against the length $T$. We have taken different realizations of the textures `river` and `steam` and have computed the distance of the former realizations against themselves and the latter[7]. It is evident that alike textures tend to cluster together. Therefore in principle a comprehensive database of parameters learned from commonly occurring dynamic textures can be maintained and a new temporal sequence can be categorized after learning its parameters and computing the distance. An extensive assessment of the recognition capabilities of our system is premature at this point due to the lack of extensive databases of dynamic textures.

### 4.5. Sequence compression

In this section we present a subjective comparison between storage requirements for the estimated parameters w.r.t. the original space requirement of the texture sequences, to get an estimate of the sequence comparison capabilities of our model. Note again that the parameters ($\mu$, $U$, $A$, $\sigma$) are self sufficient and we don't need to refer to the original texture for the synthesis of new sequences.

To make the comparison we only consider the optimal number of frames or the frames which are informative in the input sequence although there may be many more redundant frames. Consider the `steam` sequence. The critical number of frames in the `steam` sequence as given by Figure 3 is 80. Therefore the storage requirement amounts to $80 \times 96 \times 176$ or $1.35 \times 10^6$. We used N=20 principal components, with one average image I, the space requirement for the estimated parameters equals $96 * 176$ ($I$) $+20 * 96 * 176$ (*principal components*) $+20 * 20$ ($A$) $+20$ ($\sigma$) or $4.5 \times 10^5$. This gives us a worst case compression of $1 : 3$ for a typical example. It should be understood however that the the number of input frames are often far more than the critical length and our space complexity is indifferent to this value. For instance, if the requirement is to generate an infinite sequence from a given 10,000 frame movie the compression would be $1 : 400$.

## 5. Discussion

We have introduced a novel representation of dynamic texture and associated algorithms to perform learning, recognition and synthesis of sequences from training data. We have demonstrated experimentally that even the simplest choices in the model (linear stochastic systems driven by Gaussian white

[7]We obtain a similar plot if we compute the distance from the latter w.r.t. the former although the K-L distance by definition is not commutative.

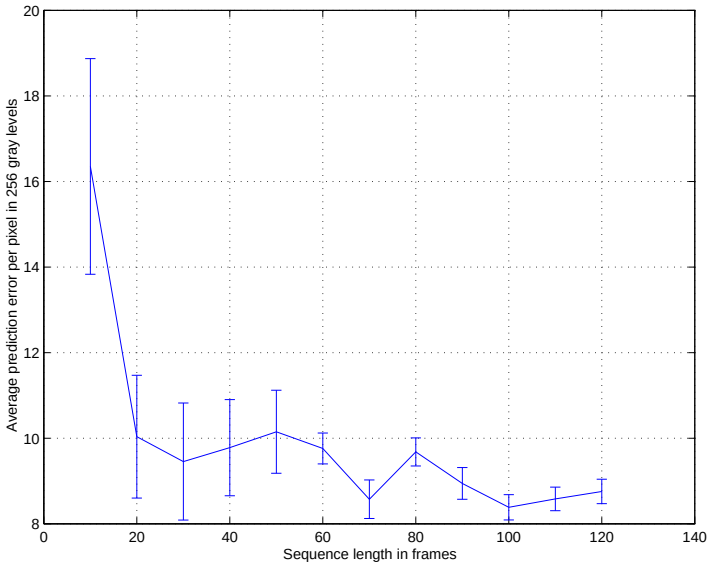


Figure 2: **Model verification** : *to verify the quality of the model learned, we have used a fixed number of principal components in the representation (20) and considered sub-sequences of the original data set of length varying from* 10 *to* 120*. We have used such sub-sequences to learn the parameters of the model in the Maximum-Likelihood sense, and then used the model to predict the next image. Using one criterion for learning (ML) and another one for validation (PEM) is informative, for it challenges the model. The average prediction error per pixel is shown as a function of the length of the training sequence (for the* `steam` *sequence), expressed in gray scale within a range of* 256 *levels. The average error per pixel decreases and becomes stable after some critical length (in this case* 80 *frames).*

noise) can capture complex visual phenomena. The algorithm is simple to implement, efficient to learn and fast to simulate. Some of these results may be useful for image compression and for image-based rendering and synthesis of image sequences.

Our framework can be extended to account for higher-order and nonlinear dynamics, and arbitrary input distributions, although we do not pursue this avenue in this paper.

### Acknowledgements

This research is supported in part by NSF grant IIS-9876145.

# References

[1] S. Amari and J. Cardoso. Blind source separation - semiparametric statistical approach. *IEEE Transactions on Signal Processing*, 45:692–700, nov 1997.

[2] K. Arun and S. Kung. Balanced approximation of stochastic systems. *SIAM Matrix Analysis and Applications, 11*, pages 42–68, 1990.

[3] Z. Bar-Joseph. Statistical learning of multi-dimensional textures. Master's thesis, The Hebrew University of Jerusalem, June 1999.

[4] Z. Bar-Joseph, R. El-Yaniv, D. Lischinski, and M. Werman. Texture mixing and texture movie synthesis using statistical learning. To appear in IEEE Transactions on Visualization and Computer Graphics.

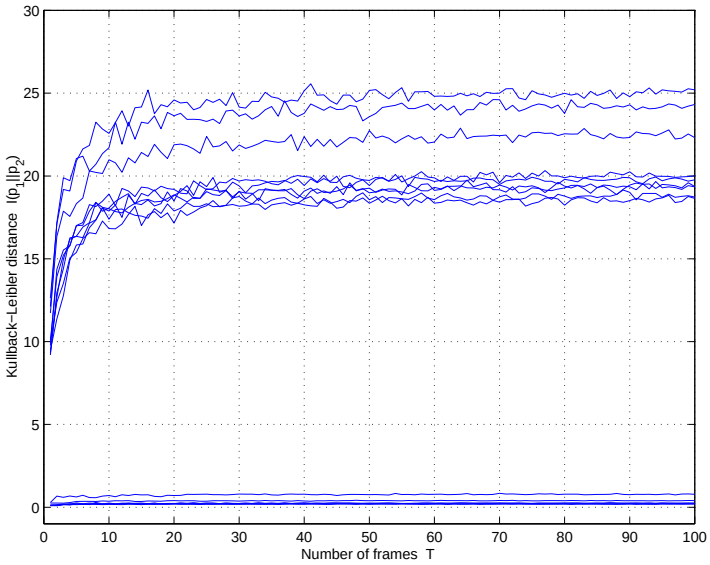


Figure 3: *The figure demonstrates that textures belonging to the same class tend to cluster together in the K-L distance space. In particular for this figure distances are computed amongst three realizations of the* `river` *sequence and three of the* `steam` *sequence w.r.t. the former. The cluster of graphs on top refer to (steam w.r.t. river) type of distances and the ones below refer to the (river w.r.t. river) type. The K-L distances are computed using Monte-Carlo methods.*

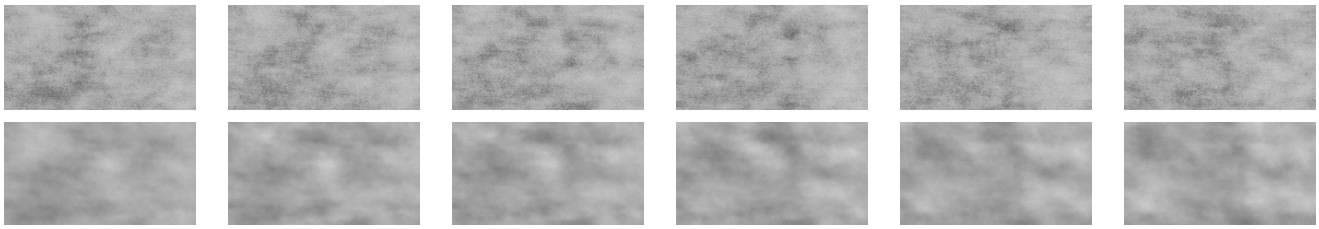

Figure 4: *A few frames of the original* `steam` *sequence (140 frames-long) in the first row and a section of frames from the "infinitely long" synthesized texture sequence (1000 frames-long; synthesized using* $N = 20$ *principal components) in the second row. Spatial smoothing in the synthesized patterns obtained is due to the dimensionality reduction step, moreover the effect is accentuated because a low number of principal components has been used for the movie generation. However, this work emphasize the characterization of the dynamic textures not only in space but even in time and this is hard to show with a paper. At this purpose the reader can refer to the movies, about the training and the synthesized sequences, that are available online at* `http://www.vision.cs.ucla.edu/dynamic-textures/movies.html`.


[5] J. Bigun and J. M. du Buf. N-folded symmetries by complex moments in gabor space and their application to unsupervised texture segmentation. In *IEEE transactions on Pattern Analysis and Machine Intelligence,16(1)*, pages 80–87, january 1994.

[6] A. Blake and M. Isard. Active contours. *SpringerVerlag*, 1998.

[7] K. D. Cock and B. De Moor. subspace angles and distances between arma models. In *Proceedings of the 14th international symposium of mathematical theory of networks and systems, Perpignan, France*, June 2000.

[8] G. Cross and A. Jain. Markov random field texture models. In *IEEE transactions on Pattern Analysis and Machine Intelligence*, volume 5, pages 25–40, 1983.

[9] J. de Bonet and P. Viola. Multiresolution sampling procedure for analysis and synthesis of texture images. In *Proceedings IEEE Conf. On Computer Vision and Pattern Recognition*, 1998.

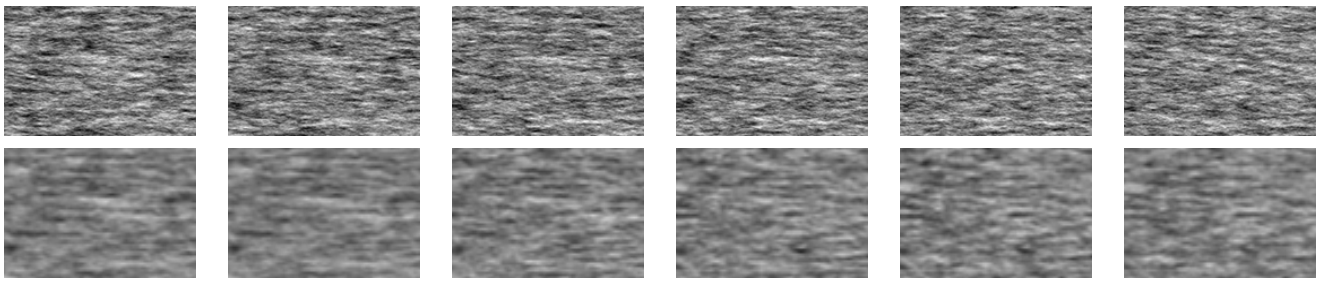

Figure 5: *A similar representation as in Figure 4 for the* `river` *sequence.*


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

[11] D. Ebert, W. Carlson, and R. Parent. Solid spaces and inverse particle systems for controlling the animation of gases and fluids. *The Visual Computer,10(4)*, pages 179–190, September 1994.

[12] A. Efros and T. Leung. Texture synthesis by non-parametric sampling. In *Seventh International Conference on Computer Vision, Corfu, Greece*, 1999.

[13] A. Fournier and W. Reeves. A simple model of ocean waves. In *ACM SIGGRAPH Proceedings*, pages 75–84, 1986.

[14] W. Freeman and E. Adelson. The design and use of steerable filters. In *IEEE transactions on Pattern Analysis and Machine Intelligence, 13(9)*, pages 891–906, 1991.

[15] H. Greenspan, S. Belongie, P. Perona, R. Goodman, S. Rakshit, and C. Anderson. Overcomplete steerable pyramid filters and rotation invariance. In *Proceedings IEEE Conf. On Computer Vision and Pattern Recognition*, pages 222–228, June 1994.

[16] M. Hassner and J. Sklansky. The use of markov random fields as models of texture. *Image Modeling. Academic Press Inc.*, 1981.

[17] D. Heeger and J. Bergen. Pyramid-based texture analysis /synthesis. In *ACM SIGGRAPH Conference Proceedings*, August 1995.

[18] A. K. Jain and F. Farrokhnia. Unsupervised texture segmentation using gabor filters. *Pattern Recognition*, 24:1167–1186, 1991.

[19] B. Julesz. Visual pattern discrimination. *IRE Trans info theory, IT-8*, 1962.

[20] R. Kailath. *Linear Systems*. Prentice Hall,Englewood Cliffs, NJ., 1980.

[21] A. Kirsch. An introduction to the mathematical theory of inverse problems. *Springer-Verlag, New York*, 1996.

[22] A. Lindquist and G. Picci. the stochastic realization problem. *SIAM J. Control Optim. 17*, pages 365–389, 1979.

[23] J. Liu, R. Chen, and T. Logvinenko. A theoretical framework for sequential importance sampling and resampling. Technical report, Stanford University, Department of Statistics, January 2000.

[24] L. Ljung. *System Identification -Theory for the User*. Prentice Hall, Englewood Cliffs, NJ, 1987.

[25] S. Mallat. A theory of multiresolution signal decomposition: the wavelet representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11:674–693, 1989.

[26] R. Manduchi and J. Portilla. Independent component analysis of textures. In *Proceedings IEEE Conf. On Computer Vision and Pattern Recognition, Corfu, Greece*, 1999.

[27] D. Mumford and B. Gidas. Stochastic models for generic images. In *Technical report, Division of Applied Mathematics, Brown University*, 1998.

[28] R. Paget and D. Longstaff. A nonparametric multiscale markov random field model for synthesising natural textures. In *Fourth International Symposium on Signal Processing and its Applications*, volume 2, pages 744–747, August 1996.

[29] D. Peachey. Modeling waves and surf. In *SIGGRAPH Conference Proceedings*, pages 65–74, 1986.

[30] K. Popat and R. Picard. Novel cluster-based probability model for texture synthesis, classification, and compression. In *Proceedings SPIE visual Communications and Image Processing, Boston*, 1993.

[31] J. Portilla and E. Simoncelli. Texture representation and synthesis using correlation of complex wavelet coefficient magnitudes. In *CSIC,Madrid*, April 1999.

[32] W. Reeves. Particle systems: a technique for modeling a class of fuzzy objects. In *ACM Trans. Graphics*, volume 2, pages 91–108, 1983.

[33] J. Rissanen. Modeling by shortest data description. *Automatica*, 14:465–471, 1978.

[34] A. Schodl, R. Szeliski, D. Salesin, and I. Essa. Video textures. In *Proceedings of ACM SIGGRAPH Conference, New Orleans, LA*, January 2000.

[35] D. Shy and P. Perona. separable pyramid steerable scalable kernels. In *Proceedings IEEE Conf. On Computer Vision and Pattern Recognition,Seattle*, pages 237–244, June 1994.

[36] E. Simoncelli, W. Freeman, E. Adelson, and D. Heeger. Shiftable multi-scale transforms. In *IEEE Trans Information Theory, 38(2)*, pages 587–607, March 1992.

[37] K. Sims. Particle animation and rendering using data parallel computation. *Computer Graphics*, 24(4):405–413, 1990.

[38] J. Stam and E. Fiume. Depicting fire and other gaseous phenomena using diffusion processes. In *SIGGRAPH Conference Proceedings*, pages 129–136, 1995.

[39] M. Szummer. Temporal texture modeling. Master's thesis, Massachusetts Institute Of Technology, 1995.

[40] M. Szummer and R. W. Picard. Temporal texture modeling. In *IEEE International Conference on Image Processing, Lausanne, Switzerland*, volume 3, Sept 1996.

[41] L. Y. Wei and M. Levoy. Fast teture synthesis using tree structured vector quantization. In *SIGGRAPH Conference Proceedings*, 2000.

[42] S. Zhu, Y. Wu, and D. Mumford. Filters, random fields and maximum entropy (frame):towards the unified theory for texture modeling. In *Proceedings IEEE Conf. On Computer Vision and Pattern Recognition*, June 1996.

[43] S. Zhu, Y. Wu, and D. Mumford. Minimax entropy principle and its application to texture modeling. *Neural Computation*, 9:1627–1660, 1997.