Chapter21

Modeling Dynamic Scenes: An Overview of Dynamic Textures

G. Doretto and S. Soatto

Abstract

Dynamic scenes with arbitrary radiometry and geometry present a challenge in that a physical model of their motion, shape, and reflectance cannot be inferred. Therefore, the issue of representation becomes crucial, and while there is no right or wrong representation, the task at hand should guide the modeling process. For instance, if the task is three-dimensional reconstruction, one can make assumptions on reflectance and illumination in order to recover shape and motion. If the task is synthesis, or reprojection, the correct shape is unimportant, as long as the model supports the generation of a valid view of the scene. If the task is detection or recognition, a physical model is not necessary as long as one can infer a statistical model that can be used to perform classification. We concentrate our attention on the two latter cases, and describe a modeling framework for dynamic scenes for the purpose of synthesis, detection and recognition. In particular, we restrict our attention to sequences of images of moving scenes that exhibit certain statistical stationarity properties, which have been called Dynamic Textures. They include sea-waves, smoke, foliage, whirlwind etc. In this chapter we describe a characterization of dynamic textures and pose the problems of modeling, learning, recognition and segmentation of dynamic textures using tools from time series analysis, and system identification theory.

21.1 Introduction

Consider a sequence of images of a moving scene. Each image is an array of positive numbers that depend upon the shape, pose, viewpoint (*geometry*), material reflectance properties, and light distribution (*radiometry*) of the scene, as well as upon the changes of all of these factors over time, i.e. upon the *dynamics* of the scene. In principle, to fully analyze and understand the properties of a video sequence, one would want to recover the physical model of the scene that could have generated the images. Unfortunately, it is well known that the joint reconstruction of radiometry, geometry, and dynamics of the scene (*visual reconstruction problem*) is an intrinsically ill-posed problem: From any number of images it is not possible to uniquely recover all the unknowns (shape, pose, reflectance, light distribution, and viewpoint). This means that it is always possible to construct scenes with different radiometry, geometry, and dynamics that give rise to the same images. For example, a video clip of the sea at sunset could have been originated by a very complex dynamic shape (the surface of the sea) with constant reflectance properties (homogeneous material, water), but also by a very simple shape (e.g. the plane of the television monitor) with a dynamic non-homogeneous radiance (the televised spatio-temporal signal). The ill-posedness of the visual reconstruction problem can be turned into a well-posed inference problem within the context of a specific task, and one can also use the extra degrees of freedom to the benefit of the application at hand by satisfying some additional optimality criterion (e.g. the minimum description length (MDL) principle [675] for compression). This way, even though one cannot infer "the" (physically correct) model of a scene, one can infer a representation of the scene that can be sufficient to support, for instance, control, or recognition tasks.

In this chapter we survey a series of recent papers that describe very simple statistical models that can explain the measured video signal, predict new measurements, and extrapolate new image data. These models are not models of the scene, but statistical models of the video signal. In general, they fail to capture the correct radiometry, geometry, and dynamics of the scene. Instead, they capture a mixture of the three that is equivalent to the underlying physical model of the scene, once the statistical model is "visualized" as a sequence of images. Hopefully, these models will provide a representation of geometry, radiometry and dynamics that is sufficient to support recognition and segmentation tasks.

We put the emphasis on sequences of images that exhibit some form of temporal regularity[1], such as sequences of fire, smoke, water, foliage or flowers in wind, clouds, crowds of waving people, etc., and we refer to them as *dynamic textures* [278]. In statistical terms, we assume that a dynamic texture is a sequence of images, that is a realization from a stationary stochastic process[2]. In Section 21.2 we describe a representation of dynamic textures introduced in [748] that is general (it accounts for every possible decomposition of images, and every possible dynamics of sequences), and precise (it allows making analytical statements and drawing from the rich literature on system identification). In Section 21.3 we describe a technique to learn model parameters using maximum likelihood or prediction error methods. Under the hypothesis of second-order stationarity, there is a closed-form sub-optimal solution of the learning problem. In Section 21.4 the model is tested on simulation and prediction, showing that even the simplest

[1]The case of sequences that exhibit temporal and spatial regularity is treated in [280].

[2]A stochastic process is stationary (of order k) if the joint statistics (up to order k) are time-invariant. For instance a process $\{I(t)\}$ is second-order stationary if its mean $\bar{I} \doteq E[I(t)]$ is constant and its covariance $E[(I(t_1) - \bar{I})(I(t_2) - \bar{I})]$ only depends upon $t_2 - t_1$.

instance of the model captures a wide range of dynamic textures. The algorithm is simple to implement, efficient to learn and fast to simulate; it allows generating infinitely long sequences from short input sequences, and to control the parameters in the simulation [281]. In Section 21.5 we investigate the discriminative power of the models and describe a classification scheme based on the k-nearest neighbor rule, as it has been proposed in [698]. Section 21.6 addresses the problem of segmenting the image plane of a video sequence into homogeneous regions characterized by constant spatio-temporal signatures, as introduced in [279]. We illustrate a region-based segmentation framework where we model the signatures with dynamic texture models and compare them by means of the distances proposed in Section 21.5.1.

21.1.1 Related work

Statistical inference for analyzing and understanding general images has been extensively used for the last two decades. There has been a considerable amount of work in the area of 2D texture analysis, starting with the pioneering work of Julesz [444], until the more recent statistical models (see [658] and references therein).

There has been comparatively little work in the specific area of dynamic (or time-varying) textures. The problem has been first addressed by Nelson and Polana [596], who classify regional activities of a scene characterized by complex, non-rigid motion. Szummer and Picard's work [783] on temporal texture modeling uses the spatio-temporal auto-regressive model, which imposes a neighborhood causality constraint for both spatial and temporal domain. This restricts the range of processes that can be modeled, and does not allow to capture rotation, acceleration and other simple non translational motions. Bar-Joseph et al. [50] uses multi-resolution analysis and tree-merging for the synthesis of 2D textures and extends the idea to dynamic textures by constructing trees using a 3D wavelet transform.

Other related work [318] is used to register nowhere-static sequences of images, and synthesize new sequences. Parallel to these approaches there is the work of Wang and Zhu [855, 856] where images are decomposed by using a dictionary of Gabor or Fourier bases to represent image elements called "movetons," or by computing their primal sketch, and the model captures the temporal variability of movetons, or the graph describing the sketches. Finally, in [913] feedback control is used to improve the rendering performance of the dynamic texture model we describe in this chapter.

The problem of modeling dynamic textures for the purposes of synthesis has been tackled also by computer graphics researchers. The typical approach is to synthesize new video sequences using procedural techniques that essentially entail clever concatenation or repetition of training image data. The reader is referred to [716, 869, 499, 83] and references therein.

21.2 Representation of dynamic textures

The intuitive notion of a dynamic texture is that of a sequence of images that exhibits temporal regularity. 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. Therefore, the underlying assumption is that the temporal correlation of sequences can be modeled by the integration of independent and identically distributed (IID) samples from a stationary distribution. In other words, a sequence of images can be modeled as the output of a dynamical system. We follow [278] and now make this concept precise.

Let $\{I(t)\}_{t=1\ldots\tau}$, $I(t) \in \mathbb{R}^m$, be a sequence of τ images. Suppose that at each instant of time t we can measure a noisy version of the image, $y(t) = I(t) + w(t)$, where $w(t) \in \mathbb{R}^m$ is an IID sequence drawn from a known distribution $p_w(\cdot)$ (that can be inferred from the physics of the imaging device), resulting in a positive measured sequence $\{y(t)\}_{t=1\ldots\tau}$. We say that *the sequence* $\{I(t)\}$ *is a (linear) dynamic texture* if there exists a set of n spatial filters $\phi_\alpha : \mathbb{R} \rightarrow \mathbb{R}^m$, $\alpha = 1\ldots n$ and a stationary distribution $q(\cdot)$ such that, defining $x(t) \in \mathbb{R}^n$ such that $I(t) = \phi(x(t))$ (where $\phi(\cdot)$ indicates the combination of the output of the n filters $\{\phi_\alpha\}$ respectively applied to each of the n state components) we have $x(t) = \sum_{i=1}^{k} A_i x(t-i) + v(t)$, with $v(t) \in \mathbb{R}^n$ an IID realization from the density $q(\cdot)$, for some choice of matrices, $A_i \in \mathbb{R}^{n\times n}$, $i = 1,\ldots,k$, and initial condition $x(0) = x_0$. Without loss of generality, we can assume $k = 1$ since we can augment the state of the above model to be $\bar{x}(t) \doteq [x(t)^T\, x(t-1)^T \ldots x(t-k)^T]^T$. Therefore, a linear dynamic texture is associated to the dynamical system

$$\begin{cases} x(t+1) = Ax(t) + v(t) \\ y(t) = \phi(x(t)) + w(t) \end{cases} \tag{21.1}$$

with $x(0) = x_0,\ v(t) \overset{IID}{\sim} q(\cdot)$ unknown, $w(t) \overset{IID}{\sim} p_w(\cdot)$ given, such that $I(t) = \phi(x(t))$. One can easily generalize the definition to an arbitrary non-linear model of the form $x(t+1) = f(x(t), v(t))$, leading to the concept of a *non-linear dynamic texture*.

21.3 Learning dynamic textures[3]

Given a sequence of noisy images $\{y(t)\}_{t=1\ldots\tau}$, learning the dynamic texture model (21.1) amounts to identifying the model parameter A, the filters $\phi(\cdot)$, and the distribution of the input $q(\cdot)$. This is a form of *system identification problem* [524], where one has to infer a dynamical model from a time series. The maximum-likelihood formulation of the dynamic texture learning problem can be

[3]This section is based on material coauthored with A. Chiuso and Y. N. Wu [278].

posed as follows:

$$\begin{aligned}&\text{given } y(1),\ldots,y(\tau),\ \text{find}\\ &\hat{A},\hat{\phi}(\cdot),\hat{q}(\cdot) = \arg\max_{A,\phi,q} \log p(y(1),\ldots,y(\tau))\\ &\text{subject to (21.1) and } v(t) \overset{IID}{\sim} q.\end{aligned} \tag{21.2}$$

While we refer the reader to [278] for a more complete discussion about how to solve problem (21.2), and how to set out the learning via prediction error methods, here we summarize a number of simplifications that lead us to a simple closed-form procedure.

In (21.2) we have not made any assumption on the class of filters $\phi(\cdot)$, and there are many ways in which one can choose them. However, in texture analysis the dimension of the signal is huge (tens of thousands components) and there is a lot of redundancy. Therefore, we view the choice of filters as a dimensionality reduction step and seek for a decomposition of the image in the simple (linear) form $I(t) = \sum_{i=1}^{n} x_i(t)\theta_i \doteq Cx(t)$, where $C = [\theta_1,\ldots,\theta_n] \in \mathbb{R}^{m\times n}$ and $\{\theta_i\}$ can be an orthonormal basis of L^2, a set of principal components, or a wavelet filter bank, and $m \gg n$. Note that the inference method depends also upon what type of representation we choose for q. In principle, the unknown driving distribution belongs to an infinite-dimensional space. The simplest parametric class of densities is the Gaussian $v(t) \overset{IID}{\sim} \mathcal{N}(0,Q)$, and $Q \in \mathbb{R}^{n\times n}$ is a symmetric positive-definite matrix. We assume a similar distribution for the measurement noise $w(t) \overset{IID}{\sim} \mathcal{N}(0,R)$, $R \in \mathbb{R}^{m\times m}$. Under these hypotheses the more general model (21.1) reduces to the following linear Gauss-Markov model

$$\begin{cases} x(t+1) = Ax(t) + v(t)\ , & v(t) \sim \mathcal{N}(0,Q)\ ,\quad x(0) = x_0\ ,\\ y(t) = Cx(t) + w(t)\ , & w(t) \sim \mathcal{N}(0,R)\ ,\end{cases} \tag{21.3}$$

and the system identification problem consists in estimating the parameters A, C, Q, R from the measurements $y(1),\ldots,y(\tau)$. It is well known that this model can capture the second-order properties of a generic stationary stochastic process [524].

The first observation concerning model (21.3) is that the choice of matrices A, C, Q is not unique, in the sense that there are infinitely many such matrices that give rise to exactly the same sample paths $y(t)$ starting from suitable initial conditions. This is immediately seen by substituting A with TAT^{-1}, C with CT^{-1} and Q with TQT^T, and choosing the initial condition Tx_0, where $T \in GL(n)$ is any invertible $n\times n$ matrix. In other words, the basis of the state-space is arbitrary, and any given process has *not* a unique model, but an *equivalence class* of models $\mathcal{R} \doteq \{[A] = TAT^{-1}, [C] = CT^{-1}, [Q] = TQT^T,\ \mid T \in GL(n)\}$. In order to identify a unique model of the type (21.3) from a sample path $y(t)$, it is necessary to choose a representative of each equivalence class: such a representative is called a *canonical model realization*, in the sense that it does not depend on the choice of basis of the state space (because it has been fixed).

While there are many possible choices of canonical models (see for instance [446]), we will make the assumption that $\mathrm{rank}(C) = n$ and choose the canonical model that makes the columns of C orthonormal: $C^T C = I_n$, where I_n is the identity matrix of dimension $n \times n$. As we will see shortly, this assumption results in a unique model that is tailored to the data in the sense of defining a basis of the state space such that its covariance $P \doteq \lim_{t\to\infty} E[x(t)x^T(t)]$ is asymptotically diagonal (see Equation (21.7)).

With the above simplifications one may use *subspace identification* techniques [524] to learn model parameters in closed-form in the maximum-likelihood sense, for instance with the well known N4SID algorithm [832]. Unfortunately this is not possible. In fact, given the dimensionality of our data, the requirements in terms of computation and memory storage of standard system identification techniques are far beyond the capabilities of the current state of the art workstations. For this reason, following [278], we describe a closed-form sub-optimal solution of the learning problem, that takes few seconds to run on a current low-end PC when $m = 170 \times 110$ and $\tau = 120$.

21.3.1 Closed-form solution

Let $Y_1^\tau \doteq [y(1), \dots, y(\tau)] \in \mathbb{R}^{m\times\tau}$ with $\tau > n$, and similarly for $X_1^\tau \doteq [x(1), \dots, x(\tau)] \in \mathbb{R}^{n\times\tau}$ and $W_1^\tau \doteq [w(1), \dots, w(\tau)] \in \mathbb{R}^{m\times\tau}$, and notice that

$$Y_1^\tau = C X_1^\tau + W_1^\tau \ . \tag{21.4}$$

Now let $Y_1^\tau = U\Sigma V^T$; $U \in \mathbb{R}^{m\times n}$; $U^T U = I$; $V \in \mathbb{R}^{\tau\times n}$, $V^T V = I$ be the singular value decomposition (SVD) [352] with $\Sigma = \mathrm{diag}\{\sigma_1, \dots, \sigma_n\}$, and $\{\sigma_i\}$ be the singular values, and consider the problem of finding the best estimate of C in the sense of Frobenius: $\hat{C}(\tau), \hat{X}(\tau) = \arg\min_{C, X_1^\tau} \|W_1^\tau\|_F$ subject to (21.4). It follows immediately from the fixed rank approximation property of the SVD [352] that the unique solution is given by

$$\hat{C}(\tau) = U \ , \quad \hat{X}(\tau) = \Sigma V^T \ , \tag{21.5}$$

$\hat{A}$ can be determined uniquely, again in the sense of Frobenius, by solving the following linear problem: $\hat{A}(\tau) = \arg\min_A \|X_1^\tau - A X_0^{\tau-1}\|_F$, where $X_0^{\tau-1} \doteq [x(0), \dots, x(\tau-1)] \in \mathbb{R}^{n\times\tau}$ which is trivially done in closed-form using the state estimated from (21.5):

$$\hat{A}(\tau) = \Sigma V^T D_1 V (V^T D_2 V)^{-1} \Sigma^{-1} \ , \tag{21.6}$$

where $D_1 = \begin{bmatrix} 0 & 0 \\ I_{\tau-1} & 0 \end{bmatrix}$ and $D_2 = \begin{bmatrix} I_{\tau-1} & 0 \\ 0 & 0 \end{bmatrix}$. Notice that $\hat{C}(\tau)$ is uniquely determined up to a change of sign of the components of C and x. Also note that

$$E[\hat{x}(t)\hat{x}^T(t)] \equiv \lim_{\tau\to\infty} \frac{1}{\tau} \sum_{k=1}^{\tau} \hat{x}(t+k)\hat{x}^T(t+k) = \Sigma V^T V \Sigma = \Sigma^2 \ , \tag{21.7}$$

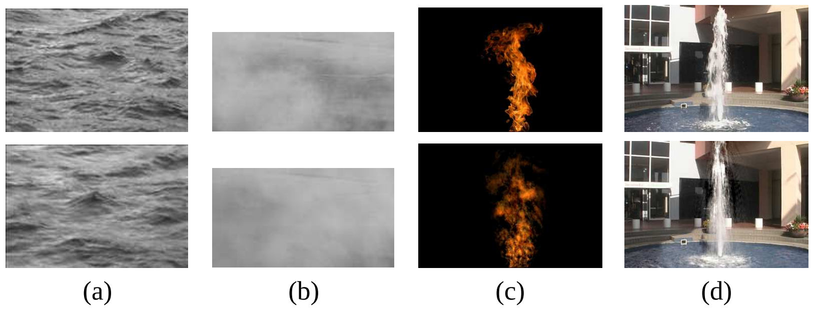

Figure 21.1. Samples of four training sequences (top row) and four synthesized frames with the corresponding models (bottom row). (a) River sequence ($\tau = 120$ training images of $m = 170 \times 115$ pixels). Simulation is performed with a model of state dimension $n = 50$. (b) Steam sequence ($\tau = 120$, $m = 176 \times 96$), $n = 30$. (c) Fire sequence ($\tau = 150$, $m = 360 \times 243$), $n = 50$. (d) Fountain sequence ($\tau = 150$, $m = 320 \times 220$), $n = 50$. The river and stem sequences have been borrowed from the MIT Temporal Texture database, whereas the fire sequence comes from the Artbeats Digital Film Library. In all these cases the state dimension n was given as input parameter. The movies are available on-line at `http://www.cs.ucla.edu/~doretto/projects/dynamic-textures.html`.

which is diagonal as mentioned in the first part of Section 21.3. Finally, the sample input noise covariance Q can be estimated from

$$\hat{Q}(\tau) = \frac{1}{\tau} \sum_{i=1}^{\tau} \hat{v}(i)\hat{v}^T(i) \; , \tag{21.8}$$

where $\hat{v}(t) \doteq \hat{x}(t+1) - \hat{A}(\tau)\hat{x}(t)$. Should $\hat{Q}$ not be full rank, its dimensionality can be further reduced by computing the SVD $\hat{Q} = U_Q \Sigma_Q U_Q^T$ where $\Sigma_Q = \operatorname{diag}\{\sigma_{Q,1}, \ldots, \sigma_{Q,n_v}\}$ with $n_v \leq n$, and one can set $v(t) \doteq B\eta(t)$, with $\eta(t) \sim \mathcal{N}(0, I_{n_v})$, and $\hat{B}$ such that $\hat{B}\hat{B}^T = \hat{Q}$.

In the algorithm above we have assumed that the order of the model n was given. In practice, this needs to be inferred from the data. Following [278], one can determine the model order empirically from the singular values $\sigma_1, \sigma_2, \ldots,$ by choosing n as the cutoff where the singular values drop below a threshold. A threshold can also be imposed on the difference between adjacent singular values.

21.4 Model validation

One of the most compelling validations for a dynamic texture model is to simulate it to evaluate to what extent the synthesis captures the essential perceptual features of the original data. Given a typical training sequence of about one hundred frames, using the procedure described in Section 21.3.1 one can learn model parameters in a few seconds, and then synthesize a potentially infinite number of new images by simulating (21.3). To generate a new image one needs to draw a sample $v(t)$ from a Gaussian distribution with covariance Q, update the state

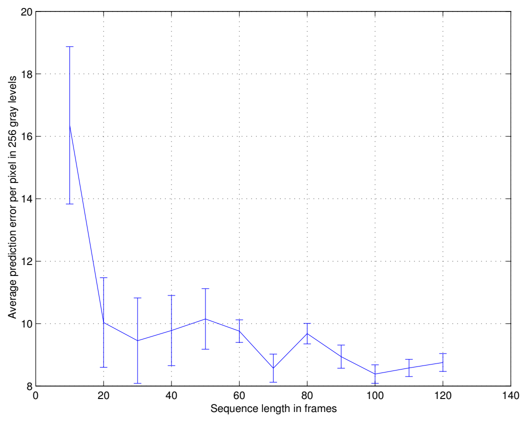

Figure 21.2. Error-bar plot of the average prediction error and standard deviation (for 100 trials), per pixel as a function of the length of the steam training sequence, expressed in gray levels (the range of pixel values is [0,255]). The state dimension is set to $n = 20$.

$x(t+1) = Ax(t) + v(t)$, and compute the image $I(t) = Cx(t)$. This can be done in real time. Even though the result is best shown with movies, Figure 21.1 provides some examples of the kind of output that one can get (see [278] for more results). The simple model (21.3), that captures only the second-order temporal statistics of a video sequence, is able to capture most of the perceptual features of sequences of images of natural phenomena, such as fire, smoke, water, flowers or foliage in wind, etc., and even dynamic textures that are periodic signals in time [277].

An important question is how long should the input sequence be in order to capture the dynamics of the process. To answer this question experimentally, for a fixed state dimension, we consider the prediction error as a function of the length τ, of the input (training) sequence. This means that for each length τ, we predict the frame $\tau + 1$ (not part of the training set) and compute the prediction error per pixel in gray levels. We do so many times in order to infer the statistics of the prediction error, i.e. mean and variance at each τ. Using one criterion for learning (the procedure in Section 21.3.1), and another one for validation (prediction error) is informative for challenging the model. Figure 21.2 shows an error-bar plot including mean and standard deviation of the prediction error per pixel for the steam sequence. The average error decreases and becomes stable after approximately 70 frames. The plot of Figure 21.2 validates *a-posteriori* the model (21.3) inferred with the procedure described in Section 21.3.1. Other dynamic textures have similar prediction error plots [278].

21.5 Recognition[4]

According to the model (21.3) a dynamic texture is characterized by a linear dynamic system with multiple-input and multiple-output (MIMO) driven by white noise (which is also a vector autoregressive moving average (ARMA) model). Therefore, following [698], in order to build a recognition system able to categorize dynamic textures, one needs to first define a base measure in the space of vector ARMA models, and then to characterize probability distributions in that space. Defining an appropriate base measure in the space of vector ARMA models is not trivial, since each model entails a combination of an input density and state and output transition matrices that have a very particular Riemannian structure (they are *not* a linear space), and this problem remains unsolved to this day.

From a pattern recognition viewpoint [288], constructing a probability density is not necessary to solve problems such as classification, clustering or grouping. For instance, the k-nearest neighbor algorithm only requires a distance to be implemented. This approach can be applied to the space of models, that will be endowed by a probability structure induced by the notion of distance that we have defined. More precisely, suppose a set of model samples M_1, $\cdots$, M_N, is given, where each model is labeled with λ_j, which is one out of c classes. Given a new model sample M, the label λ_m is chosen by taking a vote among the k nearest model samples. That is, λ_m is selected if the majority of the k nearest neighbors have label λ_m. For $c = 2$ this happens with probability $\sum_{i=(k+1)/2}^{k} \begin{pmatrix} k \\ i \end{pmatrix} P(\lambda_m|M)^i(1 - P(\lambda_m|M))^{k-i}$. It can be shown [288] that if k is odd, $N \gg c$, and $c = 2$, the error rate is bounded above by the smallest concave function of P^* (the optimal error rate) greater than $\sum_{i=0}^{(k+1)/2} \begin{pmatrix} k \\ i \end{pmatrix} ({P^*}^{i+1}(1-P^*)^{ki} + {P^*}^{k-i}(1-P^*)^{i+1})$. Note that the analysis holds for k fixed as $N \to \infty$, and that the rule approaches the minimum error rate for $k \to \infty$.

We assume that a model M is given by the couple (A, C). That is, we do not consider the covariance of the measurement noise R, since that carries no information on the underlying process. Moreover, we consider processes with different input noise covariance as equivalent, which means that we ignore Q.

21.5.1 Distances between dynamic texture models

One of the difficulties in defining a distance between ARMA models is that each model M is described not only by the parameters (A, C), but by an equivalence class of such parameters, as pointed out in Section 21.3. Therefore, a suitable discrepancy measure has to compare *not* the parameters directly, but their equivalence classes. One technique for doing so has been proposed in [251]. It consists

[4]This section is based on material coauthored with P. Saisan and Y. N. Wu [698].

of building infinite observability matrices, whose columns span the space generated by the measurements $y(t)$ of the model (21.3), which is an n-dimensional subspace of the infinite-dimensional space of all possible measurements. Then one can compute the geometric angles between such subspaces through their embedding.

More formally, let $S \in \mathbb{R}^{m\times n}$ and $T \in \mathbb{R}^{m\times n}$ be two matrices with full column rank. The n *principal angles* $\theta_k \in \left[0, \frac{\pi}{2}\right]$ between range(S) and range(T) are recursively defined for $k = 1, \dots, n$ as

$$\begin{cases} \cos\theta_1 = \max\limits_{x,y\in\mathbb{R}^n} \frac{|x^T S^T T y|}{\|Sx\|_2\|Ty\|_2} = \frac{|x_1^T S^T T y_1|}{\|Sx_1\|_2\|Ty_1\|_2} \ , \\ \cos\theta_k = \max\limits_{x,y\in\mathbb{R}^n} \frac{|x^T S^T T y|}{\|Sx\|_2\|Ty\|_2} = \frac{|x_k^T S^T T y_k|}{\|Sx_k\|_2\|Ty_k\|_2} \ , \text{ for } k = 2, \dots, n \\ \text{subject to } x_i^T S^T S x = 0 \text{ and } y_i^T T^T T y = 0, \text{ for } i = 1, 2, \dots, k-1 \ . \end{cases}$$

Now, let $M_1 \doteq (A_1, C_1)$ and $M_2 \doteq (A_2, C_2)$ be two models with the same output dimensionality. Their *infinite observability matrices* $\mathcal{O}_i$, for $i = 1, 2$, are defined as $\mathcal{O}_i \doteq \begin{bmatrix} C_i^T & A_i^T C_i^T & \dots & (A_i^T)^n C_i^T & \dots \end{bmatrix}^T \in \mathbb{R}^{\infty\times n}$, and we refer to the principal angles between the ranges of $\mathcal{O}_1$, and $\mathcal{O}_2$ as *subspace angles*. They can be computed in closed-form with a procedure described in [251].

For the case of minimum-phase single-input single-output (SISO) state space models that correspond to autoregressive (AR) models, one can use the subspace angles to define the so-called *Martin distance*:

$$d_M(M_1, M_2)^2 = \ln \prod_{i=1}^{n} \frac{1}{\cos^2\theta_i} \ , \tag{21.9}$$

which was originally proposed in [548] as function of the cepstrum coefficients[5] of the model, whereas the expression of the Martin distance as function of the subspace angles was introduced in [251]. It is also possible to define another distance, that uses only the biggest subspace angle, i.e. $d_F = \theta_n$. Geometrically d_F is the *Finsler distance* between two subspaces viewed as two elements in the Grassman manifold $G(\infty, n)$ [874]. Roughly speaking, the difference between the Martin and Finsler distance is that d_M^2 is an L^2-norm but d_F is an L^∞-norm between linear systems.

Unfortunately, the generalization of d_M^2 and d_F to the case of MIMO linear dynamic systems is not possible. For instance, it is not even guaranteed that the Martin distance be non-negative. Nevertheless, we used the idea of comparing two models by computing their subspace angles, and tested the ability of the Martin, and Finsler distance, and the naïve Frobenius norm between model parameters, to classify dynamic textures within a k-nearest neighbor scheme.

[5]The cepstrum of a discrete-time process is the inverse Fourier transform of the logarithm of the power spectrum.

21.5.2 Performance of the nearest neighbor classifier

We illustrate tests of the distances proposed in the previous section against a database of 50 categories of dynamic textures, each of which represented by four models, following [698]. The models have state dimension $n = 20$, and have been extracted from sequences of length $\tau = 75$ frames of $m = 48 \times 48$ pixels, using the procedure illustrated in Section 21.3.1. The sequences capture natural phenomena like ocean waves, smoke, steam, fire, and plants. Included in the database are similar sequences with different dynamics. For example, there are water streams recorded from different angles, with flows moving in different orientations and at different speeds.

Between each pair of models of the database we computed the Frobenius norm, and the Martin and Finsler distance. Figure 21.3 shows a gray-level representation of the confusion matrix for a subset of 10 categories (40 sequences of the entire database), for the case of Frobenius norm (top) and Martin distance (bottom). Moving along the horizontal axis, we marked the first (with an "o") and second (with an "x") nearest neighbors. For example, with reference to the results using Martin distance (bottom), the closest dynamic texture to Smoke1 (in the vertical axis) is Smoke2 (in the horizontal axis). Similarly, the second closest dynamic texture to Smoke1 is Water-Fall-b1.

If we define a hit when the first nearest neighbor of a sequence is one of the other three sequences in the same category, Figure 21.3 already highlights the differences between the Frobenius norm and the Martin distance. In the latter case most of the first nearest neighbors lie on the diagonal blocks (meaning that there is a hit), whereas in the former they are almost randomly spread all over the blocks. In particular, if we count the number of correct hits for the whole database, in the case of Frobenius norm we obtain a hit ratio of 5.5%. This poor result is not unexpected since, even though ARMA models are linear, the space of model parameters is nonlinear and the Frobenius norm assumes linearity. On the other hand, the hit ratio of the Martin distance is 89.5%, whereas the Finsler distance is less efficient with a hit ratio of 24.5%.

The encouraging results obtained using the Martin distance suggest that, in principle, a comprehensive database of models of commonly occurring dynamic textures can be maintained, and a new sequence could be categorized, after learning its parameters, using the k-nearest neighbor rule.

21.6 Segmentation[6]

Modeling the (global) spatio-temporal statistics of the entire video sequence can be a daunting task due to the complexity of natural scenes. An alternative consists of choosing a simple class of models, and then partition the scene into

[6]This section is based on material coauthored with D. Cremers and P. Favaro [279].

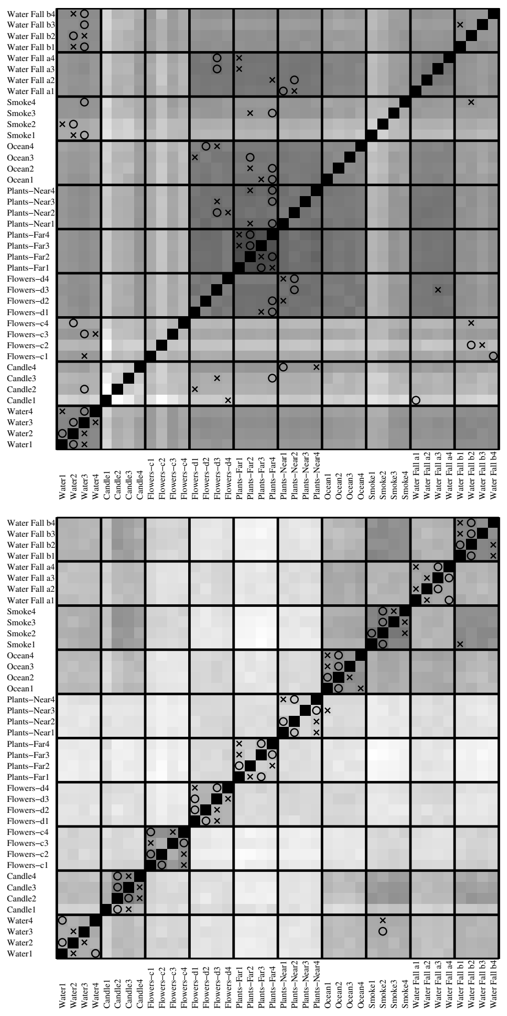

Figure 21.3. Gray-level representation of the confusion matrices for a subset of 10 dynamic texture categories (40 sequences out of 200 of the entire database), computed using the Frobenius norm (top) and the Martin distance (bottom).

regions where the model fits the data within a specific accuracy. In this section, which follows [279], we discuss a simple model for partitioning the scene into regions where the spatio-temporal statistics, represented by a dynamic texture model, is constant. To perform this *segmentation* task we use a region-based approach pioneered in [591]. In particular, we revert to a level set framework of the Mumford-Shah functional introduced in [170, 809].

Let $\Omega \subset \mathbb{R}^2$ be the domain of an image and $\{\Omega_i\}_{i=1,\ldots,N}$ be a partition of Ω into N (unknown) regions[7]. We assume that the intensities of the pixels $y_i(t)$, contained in the region Ω_i, are a Gauss-Markov process that can be modeled with a dynamic texture model (21.3), with (unknown) parameters $A_i \in \mathbb{R}^{n_i \times n_i}$, $C_i \in \mathbb{R}^{m_i \times n_i}$, and $Q_i \in \mathbb{R}^{n_i \times n_i}$. Note that we allow the number of pixels m_i to be different in each region, as long as $\sum_{i=1}^{N} m_i = m$, the size of the entire image, and that we require that neither the regions nor the parameters change over time, $\Omega_i, A_i, C_i, Q_i = \text{const}$. With this generative model, the segmentation problem can be formalized as follows: *Given a sequence of images $\{y(t)\}_{t=1,\ldots,\tau}$, $y(t) \in \mathbb{R}^m$, with two or more distinct regions $\Omega_i,\ i = 1, \ldots, N \geq 2$ that satisfy model (21.3), estimate both the regions Ω_i, and model parameters of each region, namely the matrices A_i, C_i, and Q_i.*

If the regions $\Omega_i,\ i = 1, \ldots, N$ were known, one would just be left with two problems. The first one is the learning of model parameters. This problem has already been solved in Section 21.3.1. Assuming that the parameters A_i, C_i, Q_i have been inferred for each region, in order to set the stage for a segmentation procedure, one has to define a discrepancy measure among regions, i.e. between dynamic texture models. This problem have been approached in Section 21.5, and one can measure the discrepancy between different models by comparing either the subspace angles or their combination via the Martin distance (21.9).

On the other hand, if the dynamic texture associated with each pixel were known, then one could easily determine the regions by thresholding or other grouping or segmentation techniques. However, a dynamic texture associated with a certain pixel $\mathbf{x} \in \Omega$, as defined in Equation (21.3), depends on the whole region Ω_i containing $\mathbf{x}$. Therefore, we have a classic “chicken-and-egg” problem: If we knew the regions, we could easily identify the dynamical models, and if we knew the dynamical models we could easily segment the regions. Unfortunately, we know neither.

Since one can always explain the image with a few high-order models with large support regions (the entire image in the limit), or with many low-order models with small support regions (individual pixels in the limit), in order to render the chicken-and-egg problem well posed, a model complexity cost needs to be added, for instance the description length of model parameters and the boundaries of each region [675]. This significantly complicates the algorithms and the derivation. Following [279], we simplify the problem, and first associate a local *signature* $s(\mathbf{x})$ to each pixel $\mathbf{x} \in \Omega$, by integrating visual information on a fixed spatial neighborhood of that pixel $B(\mathbf{x}) \subset \Omega$; then we group together pixels with similar signatures in a region-based segmentation approach.

Each signature contains the cosines of the subspace angles between the local dynamic texture model corresponding to $\{y(\tilde{\mathbf{x}}, t)\}_{\tilde{\mathbf{x}} \in B(\mathbf{x}), t=1,\ldots,\tau}$, and a reference dynamic texture model, for instance the one corresponding to a preselected spatio-temporal neighborhood centered at $x_0 \in \Omega$, i.e. $\{y(\tilde{\mathbf{x}}, t)\}_{\tilde{\mathbf{x}} \in B(\mathbf{x}_0), t=1,\ldots,\tau}$. With

[7]That is, $\Omega = \cup_{i=1}^{N} \Omega_i$ and $\Omega_i \cap \Omega_j = \emptyset, i \neq j$.

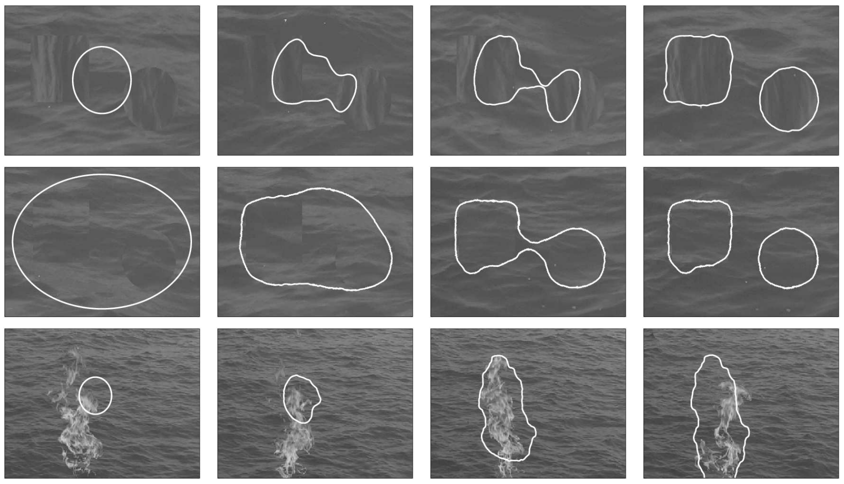

Figure 21.4. Top row: segmentation of two dynamic textures that differ only for the "orientation" of their spatial statistics, but that share the same dynamics. Middle row: segmentation of two dynamic textures that are identical in appearance, and differ only in the dynamics. Bottom row: segmentation when the region boundaries (in particular, the flame texture), are changing in time. In all the experiments, the local dynamic textures were defined on neighborhood of 11×11 pixels, whereas the state dimension was set to $n = 10$. The contour evolutions are available on-line at `http://www.cs.ucla.edu/~doretto/projects/dynamic-segmentation.html`.

this representation, a segmentation of the image plane Ω into a set of pairwise disjoint regions Ω_i of constant signature $s_i \in \mathbb{R}^n$ is obtained by minimizing the Mumford-Shah cost functional [591]:

$$E(\Gamma, \{s_i\}) = \sum_i \int_{\Omega_i} \big(s(\mathbf{x}) - s_i\big)^2 \,\mathrm{d}\mathbf{x} + \nu\,|\Gamma|\,, \qquad (21.10)$$

simultaneously with respect to the region descriptors $\{s_i\}$ modeling the average signature of each region, and with respect to the boundary Γ separating these regions. The first term in the functional (21.10) aims at maximizing the homogeneity with respect to the signatures in each region Ω_i, whereas the second term aims at minimizing the length $|\Gamma|$ of the separating boundary.

Figure 21.4 demonstrates some aspects of dynamic texture segmentation, the reader is referred to [279] for a more complete account. The first row has a few snapshots of a sequence with ocean waves, where the portion of every frame within the square and the circle have been rotated by 90 degrees. The superimposed contour evolution shows that the segmentation system can partition the image plane based only on the spatial statistics of the images. On the other hand, the sequence in the second row has the square and circle filled with the same ocean waves of the background, but running at different speeds. In this case the algorithm segments based only on the dynamics of the regions. This ability is the most important aspect of this approach. The last row shows a sequence of fire combined with the ocean waves. Here the region boundaries are moving, against the initial hypothesis. The algorithm manage to estimates the “average” region where the flame is mostly present.

21.7 Discussion

This chapter, which draws on a series of works published recently [748, 278, 698, 279], illustrates that very simple statistical models can capture the phenomenology of very complex physical processes, such as water, smoke, fire etc. The fact that the synthesis from a very simple linear Gauss-Markov model is perceptually indistinguishable from the simulation of non-linear Navier-Stokes partial differential equations, such as those that govern fluid motion, is indication that such models may be sufficient to support detection and recognition tasks and, to a certain extent, even synthesis and animation [281]. This work shows that modeling image motion, i.e. deformations of the domain of the image, can be done through modeling image values, i.e. the range of the image. Depending on the statistical properties of the scene, this can be more or less efficient. Joint modeling of the variation in domain and range of the image can result in more efficient models, as it has been recently explored [282], pointing at a direction of new development.

Acknowledgements

This work is supported by NSF ECS-0200511/IIS-0208197, ONR N00014-03-1-0850/N00014-02-1-0720, and AFOSR F49620-03-1-0095/E-16-V91-G2.