Synthesizing and Mixing Stationary Gaussian Texture Models

Gui-Song Xia, Sira Ferradans, Gabriel Peyré, Jean-François Aujol

SYNTHESIZING AND MIXING STATIONARY GAUSSIAN TEXTURE MODELS*

GUI-SONG XIA†, SIRA FERRADANS‡, GABRIEL PEYRÉ§, AND JEAN-FRANÇOIS AUJOL¶

Abstract. This paper addresses the problem of modeling textures with Gaussian processes, focusing on color stationary textures that can be either static or dynamic. We detail two classes of Gaussian processes parameterized by a small number of compactly supported linear filters, the so-called textons. The first class extends the spot noise (SN) texture model to the dynamical setting. We estimate the space-time texton to fit a translation-invariant covariance from an input exemplar. The second class is a specialization of the auto-regressive (AR) dynamic texture method to the setting of space and time stationary textures. This allows one to parameterize the covariance with only a few spatial textons. The simplicity of these models allows us to tackle a more complex problem, texture mixing which, in our case, amounts to interpolate between Gaussian models. We use optimal transport to derive geodesic paths and barycenters between the models learned from an input data set. This allows the user to navigate inside the set of texture models and perform texture synthesis from each new interpolated model. Numerical results on a library of exemplars show the ability of our method to generate arbitrary interpolations among unstructured natural textures. Moreover, experiments on a database of stationary textures show that the methods, despite their simplicity, provide state of the art results on stationary dynamical texture synthesis and mixing.

Key words. Texture analysis, texture synthesis, texture mixing, Gaussian process, dynamic textures, optimal transport.

AMS subject classifications. 15A15, 15A09, 15A23

1. Introduction.

This paper studies the analysis, synthesis and mixing of both static and dynamic textures. Textures play an important role in visual perception and image or video understanding. Static textures typically capture the surface properties of objects, while dynamic textures occur in many natural physical phenomena (such as rain, snow, or smoke).

1.1. Stationary Texture Modeling.

The modeling of textures is a longstanding and central problem in image processing, computer vision, and computer graphics. While it is difficult to give a strict mathematical definition of textures, most proposed methods tackle their analysis and synthesis using random distributions. The process of modeling a texture amounts first, to learn the underlying random process from a given exemplar image, and then, to develop algorithms to draw new samples from the learned distribution.

This paper concentrates on the modeling and mixing of both stationary static textures (SSTs) and stationary dynamic textures (SDTs). Stationary textures have been widely investigated especially after Julesz's conjecture [20], which states that humans cannot distinguish between textures with identical second-order statistics. Even though, he proved this conjecture false in the general case [21], it still hold for a

*This work has been supported by the European Research Council (ERC project SIGMA-Vision) and the French National Research Agency (project NatImages).

†State Key Lab LIESMARS, Wuhan University, 129 Luoyu Road, Wuhan, 430079, China. This author completed this work while at CNRS and CEREMADE Université Paris-Dauphine.,

‡CNRS and CEREMADE, Université Paris-Dauphine, Place du Marechal De Lattre De Tassigny, 75775 PARIS CEDEX 16, FRANCE.

§CNRS and CEREMADE, Université Paris-Dauphine, Place du Marechal De Lattre De Tassigny, 75775 PARIS CEDEX 16, FRANCE.

¶IMB, UMR 5251, Université Bordeaux 1, 351 cours de la Libération F-33405 TALENCE, France

large class of textures. Although a complex natural image is usually inhomogeneous due to the presence of objects and occlusions, it can often be considered as being locally statistically homogeneous. Furthermore, it is also reasonable to assume that the dynamical statistics of images created by natural phenomena (such as smoke, snow, waterfall, etc.) are stationary in time.

1.2. Previous Works.

Texture synthesis. Patch-based methods are adapted to complicated geometric textures, see e.g. [15]. Statistical parametric models are generally not as good in handling complex texture patterns, but they are more flexible and fast, see for instance [30]. This paper concentrates on a simple statistical texture model, namely stationary Gaussian distributions, following and extending the spot noise (SN) model initiated in [16]. For dynamic texture, we also consider Gaussian auto-regressive (AR) models as initially proposed in [10], which restricts the class of covariances but is well suited to model natural phenomena.

Modeling stationary dynamic textures. Early attempts to model SDTs include the spatio-temporal autoregressive (STAR) model [37], which creates local space-time models for individual pixels relying on their 3-D causal neighborhood. Bar-Joseph *et al.* [4] use a 3-D wavelet transform to construct multi-resolution trees for synthesizing dynamic textures. Following the idea of Efros and Leung [14], patch-based methods can be adapted to synthesize stationary dynamic textures, see for instance [42, 23]. Doretto *et al.* [12, 11] extend their initial AR models using stationary multi-scale AR models. In this paper, we consider a more direct approach to build stationary AR models using convolutive iterations, which leads to a more compact and simpler parameterization of the models. Recently, a compact Gaussian texton has been proposed for stationary 2-D textures [7], and we extend it to dynamic models.

Texture mixing. Mixing texture models requires to design algorithms to average distributions estimated from several input exemplars. The simplest approach consists in using mixture of distributions, see for instance [5]. The use of non-parametric histogram averaging has also been proposed for grayscale images [25] as well as color and wavelet features [31]. Ruiters *et. al.* [35] propose a patch-based approach for texture interpolation. Darabi *et. al.* [36] report state-of-the-art results on texture mixing using the patch match method. Mixtures of Gaussians have been used to interpolate between AR models [6], which leads to non-Gaussian sets of models. In contrast, we propose a method based on the theory of optimal transport to ensure that the texture models stay Gaussian during interpolation. We also expose how to apply this optimal transport methodology to Gaussian AR models, which resembles previous works on the geometry of AR manifolds [32, 33, 39].

Optimal transport (OT) [41] has been used intensively in computer vision as a metric between statistical features [34]. It has been much less used in computer graphics for image processing and synthesis, with the notable exception of [31], that handles statistical constraints for synthesis using a non-parametric point cloud discretization. Note that Gaussian OT has been used for color manipulation [29] but never to achieve image modeling nor synthesis.

1.3. Contributions. The first set of contributions of this paper is the development of two novel Gaussian texture models. The first one extends the spot noise (SN) model of Galerne *et. al.* [16] to the setting of dynamic textures. The second one is a specialization of the auto-regressive (AR) model of Dorreto *et. al.* [10] to the setting of spatially stationary textures. We show how both models are parameterized by a localized set of filters called textons. Our second set ofcontributions

is a novel framework to interpolate between two (geodesic computation) and several (barycenter computation) Gaussian texture models. This texture mixing method is based on displacement interpolation in the optimal transport manifold of Gaussian distributions. Finally, we show the versatility of this novel set of methods, which make explicit use of the stationarity assumption to enable fast algorithms for both analysis and synthesis. The source code to reproduce the figures of this article, as well as a database of stationary dynamic textures, synthesis and mixing results, are available online[1]

2. Stationary Stochastic Modeling of Textures.

2.1. Notations. In this paper, we denote deterministic images and videos, corresponding to samples of textures, by $f \in \mathbb{R}^{U \times d}$ ($d = 1$ for gray-scale data and $d = 3$ for color ones). We denote by $f(p) \in \mathbb{R}^d$ the value at some space or space-time pixel $p \in U$, with $U = \{0, \dots, N_1 - 1\} \times \{0, \dots, N_2 - 1\} \times \{0, \dots, N_3 - 1\}$, where $N_1 \times N_2$ is the spatial size of the data, and N_3 is the number of frames (size in time) of the data, i.e. for static textures $N_3 = 1$. We denote by $N = N_1 N_2 N_3$ the total number of voxels. Moreover, when we deal with space-time input data, we also rewrite the coordinates p as $p = (x, t)$, where $x \in U_s = \{0, \dots, N_1 - 1\} \times \{0, \dots, N_2 - 1\}$ is the 2-D indexes in space and $t \in \{0, \dots, N_3 - 1\}$ is the 1-D indexes in time.

Fourier transform. A basic tool for the analysis and synthesis of stationary texture models is the discrete Fourier transform. The Fourier transform $\hat{f} = \mathcal{F}(f) \in \mathbb{R}^{U \times d}$ of a static or dynamic texture $f \in \mathbb{R}^{U \times d}$ is

$$\forall\, \omega \in U, \quad \hat{f}(\omega) = \sum_{p \in U} f(p) \cdot e^{-2\imath\pi \langle p,\, \omega \rangle} \in \mathbb{R}^d \quad \text{where} \quad \langle p,\, \omega \rangle = \sum_{k=1}^{d} \frac{\omega_k x_k}{N_k}. \tag{2.1}$$

The transformed texture $\hat{f}$ is computed in $O(Nd \log(Nd))$ operations using the Fast Fourier Transform (FFT), and this transform is inverted with the same complexity using the inverse discrete Fourier transform written as

$$\forall\, p \in U, \quad f(p) = \frac{1}{N} \sum_{\omega \in U} \hat{f}(\omega) \cdot e^{2\imath\pi \langle p,\, \omega \rangle}.$$

Convolution. The space-time convolution of two textures $f, g \in \mathbb{R}^{U \times d}$ is defined as

$$\forall\, p \in U, \quad (f \star g)(p) = \sum_{p' \in U} f(p - p') \cdot g(p') \tag{2.2}$$

where $\cdot$ is the entry-wise multiplication of vectors in $\mathbb{C}^d$. It can be equivalently computed over the Fourier domain as

$$\forall\, \omega \in U, \quad \hat{h}(\omega) = \hat{f}(\omega) \cdot \hat{g}(\omega) \quad \text{where} \quad h = f \star g. \tag{2.3}$$

A matrix convolution kernel is $C = (C(p))_{p \in U} \in \mathbb{R}^{U \times d \times d}$ where each $C(p) \in \mathbb{R}^{d \times d}$. The convolution between such a kernel and a texture $g \in \mathbb{R}^{U \times d}$ is defined as

$$\forall\, p \in U, \quad (C \star g)(p) = \sum_{p' \in U} C(p - p') g(p') \in \mathbb{R}^d, \tag{2.4}$$

[1] http://www.ceremade.dauphine.fr/~peyre/codes/

and can be computed over the Fourier domain as

$$\forall\, \omega \in U, \quad \hat{h}(\omega) = \hat{C}(\omega)\hat{g}(\omega) \quad \text{where} \quad h = C \star g, \tag{2.5}$$

where $\hat{C}$ is obtained by applying the Fourier transform to each of the $d \times d$ entries of C.

2.2. Stationary Gaussian Processes. We model a texture using a random vector X, which is a mapping $X : \Omega \to \mathbb{R}^{U \times d}$ from $(\Omega, \mathbb{P})$ (a fixed probability space with probability $\mathbb{P}$) to $\mathbb{R}^{U \times d}$. This random vector thus maps some $\omega \in \Omega$ to a realization $X_\omega \in \mathbb{R}^{U \times d}$ which is a deterministic image or video denoted by f. Each coordinate defines a random variable $X(p) : \Omega \to \mathbb{R}^d$.

In the following, we consider a Gaussian random vector X, which follows a Gaussian distribution $\mu = \mathcal{N}(\mathfrak{m}, \mathcal{C})$ with mean $\mathfrak{m} = \mathbb{E}(X) \in \mathbb{R}^{U \times d}$ and positive semi-definite covariance $\mathcal{C} \in \mathbb{R}^{Ud \times Ud}$

$$\forall\, (p, p') \in U^2, \quad \mathcal{C}(p, p') = \mathbb{E}\big[(X(p) - \mathfrak{m}(p))(X(p') - \mathfrak{m}(p'))^*\big] \in \mathbb{R}^{d \times d}$$

where $\mathbb{E}$ is the expected value and $v^* \in \mathbb{C}^{1 \times d}$ is the transposed complex conjugate vector of $v \in \mathbb{C}^d$,. This covariance $\mathcal{C}$ can be thought of as a $(Nd) \times (Nd)$ matrix or as a collection of matrices $\mathcal{C}(p, p') \in \mathbb{R}^{d \times d}$ for $(p, p') \in U \times U$. The covariance operator is applied to a texture $f \in \mathbb{R}^{U \times d}$ as

$$\forall\, p \in U, \quad y(p) = (\mathcal{C}f)(p) = \sum_{p' \in U} \mathcal{C}(p, p') f(p'). \tag{2.6}$$

Stationarity of such a Gaussian texture model means that X has the same distribution as $X(\cdot + \tau)$ for any shift $\tau \in U$, where we assume periodic boundary conditions. It is also equivalent to imposing that the mean is a constant vector

$$\forall\, p \in U, \quad \mathfrak{m}(p) = m \in \mathbb{R}^d$$

and that the covariance matrix $\mathcal{C}$ satisfies

$$\forall\, (p, p') \in U^2, \quad \mathcal{C}(p, p') = C(p - p') \in \mathbb{R}^{d \times d}$$

where $C \in \mathbb{R}^{U \times (d \times d)}$ is the associated convolution kernel. It means that $\mathcal{C}f = C \star f$ where $\star$ is the convolution (2.4) between a kernel and a texture $f \in \mathbb{R}^{U \times d}$.

Distribution vs. realizations. In the following, if μ is a distribution (e.g. $\mu = \mathcal{N}(\mathfrak{m}, \mathcal{C})$), we denote $X \sim \mu$ when the random vector X has distribution μ. We denote $f \equiv \mu$ when the (deterministic) vector f is sampled according to a random vector having distribution μ.

3. Spot Noise Texture Models. This section introduces the first Gaussian texture model considered in the paper. It is the Spot Noise (SN) model originally considered by Galerne *et al.* [17] that we introduce and motivate differently for static textures and extend to dynamic textures. We define the canonical texton inspired that extend the one of Desolneux *et al.* [8, 9] to the dynamic setting. This texton is useful when performing model resizing, and it will also be important to perform texture mixing (see Section 6).

Given an exemplar texture $f_0 \in \mathbb{R}^{U \times d}$, the texture analysis problem corresponds to estimating the parameters $\mathfrak{m}$ and C of a stationary Gaussian model $\mu = \mathcal{N}(\mathfrak{m}, C)$

from f_0. The synthesis problem simply corresponds to sampling the distribution to obtain a realization f from μ.

The whole pipeline (analysis and synthesis) for the SN model is summarized in Algorithm 1. The following part of the section describes in detail each step of the process. Note that some steps (e.g. model resizing and synthesis) will be re-used when performing more complicated tasks such as texture mixing.

It should be noted that the synthesis step of our algorithm is slightly different from the original one in Galerne *et al.* [17]. Indeed, only gray-scale noise is used for the sampling in [17]. In contrast we use a d-dimensional white noise w. This leads to a computational complexity overhead, that is required to be able to handle non-Spot Noise model (which corresponds to non rank-1 matrices $\hat{C}(\omega)$). They are generated when using texture mixing, or when using more complicated covariance estimators than the empirical covariance.

Algorithm 1: SN Texture Analysis and Synthesis

Input: exemplar $\tilde{f}_0 \in \mathbb{R}^{U\times d}$.

Output: sample $f \in \mathbb{R}^{\tilde{U}\times d}$ of the SN model.

1. **Pre-processing.** Compute f_0 from $\tilde{f}_0$ using (3.2).
2. **SN texture analysis.** Compute $\mathcal{C}$ from f_0 using (3.5).
3. **SN texton computation.** Compute the canonical texton $\mathcal{K}$ from $\mathcal{C}$:
   - if $\mathcal{C}$ is a SN covariance (step 2. has been used), use (3.8),
   - otherwise use (3.7).
4. **Model resizing.** Compute $\tilde{\mathcal{K}}$ from $\mathcal{K}$ using (3.9).
5. **Texture synthesis.** Compute f from $\tilde{\mathcal{K}}$:
   - for periodic time synthesis, use (3.10),
   - for infinite time synthesis (causal texton), use (3.11).

3.1. Pre-processing. For the ease of exposition and computational simplicity, we assume periodic boundary conditions. To reduce boundary artifacts during the estimation process, we use as a pre-processing step the method of Moisan [26] to extract the periodic component $f_0 = \mathrm{Per}(\tilde{f}_0)$ of an arbitrary input texture $\tilde{f}_0 \in \mathbb{R}^{U\times d}$.

For the sake of completeness, we recall that this periodic component is computed by solving a Poisson equation

$$\begin{cases} \Delta f_0 = \Delta_i \tilde{f}_0 \\ \mathrm{mean}(f_0) = \mathrm{mean}(\tilde{f}_0), \end{cases} \tag{3.1}$$

where Δ is the discrete periodic Laplacian with periodic boundary conditions while Δ_i is computed with non-periodic boundary conditions

$$\begin{aligned} \Delta f(p) &= \mathrm{Card}(\mathcal{N}) \cdot f(p) - \sum_{p'-p\in\mathcal{N}} f(p) \\ \Delta_i f(p) &= \mathrm{Card}((p+\mathcal{N})\cap U) \cdot f(p) - \sum_{p'\in(p+\mathcal{N})\cap U} f(p'). \end{aligned}$$

Here, $\mathcal{N}$ is the 4-connected (resp. 6-connected) neighborhood for static (resp. for dynamic) textures ; while $(p+\mathcal{N})\cap U$ denotes the neighborhood of p inside U. The problem in Equation (3.1) can be solved in $O(dN\log(dN))$ operations over the Fourier

domain as

$$\hat{f}_0(\omega) = \begin{cases} \frac{1}{2d - 2\sum_{i=1}^{d} \cos\left(\frac{2\pi\omega_i}{N_i}\right)} \widehat{\Delta_i f}(\omega), & \text{if} \quad \omega \neq 0, \\ N \text{mean}(\tilde{f}_0), & \text{if} \quad \omega = 0, \end{cases} \tag{3.2}$$

see [26] for more details.

3.2. Spot Noise Texture Analysis. Spot Noise (SN) models, first introduced by Wijk [40] are Gaussian distributions for which the power spectra is equal to the empirical power spectra of some input exemplar. These models have been rigorously studied and extended to color textures by Galerne *et al.* [17]. They can be equivalently described as a random convolution of each channel of the exemplar with the same gray-scale white noise.

Given some deterministic input exemplar $f_0 \in \mathbb{R}^{U \times d}$, we thus estimate the mean of the Gaussian model $\mathcal{N}(\mathfrak{m}, \mathcal{C})$ as the empirical mean

$$\forall\, p \in U, \quad \mathfrak{m}(p) = m = \frac{1}{N} \sum_{p' \in U} f_0(p'). \tag{3.3}$$

We also estimate the kernel C of the covariance matrix $\mathcal{C}$ using the empirical covariance, which is also known as the empirical periodogram

$$\forall p, p' \in U, \quad C(p) = \frac{1}{N} \sum_{p' \in U} (f_0(p) - m)(f_0(p' + p) - m)^* \in \mathbb{R}^{d \times d}, \tag{3.4}$$

where v^* is the transpose of $v \in \mathbb{R}^d$. This estimator can also be understood as the maximum likelihood estimator (MLE) of the covariance matrix from a sample f_0.

The correlation C is the auto-correlation of the input exemplar $C = f_0 \star \bar{f}_0$ where $\bar{f}_0(p) = f_0(-p)$. Its Fourier spectrum is composed of rank-1 matrices, which offers a convenient way to estimate the kernel in $O(Nd \log(Nd))$ operations

$$\forall\, \omega \neq 0, \quad \hat{C}(\omega) = \frac{1}{N} \hat{f}_0(\omega) \hat{f}_0(\omega)^*. \tag{3.5}$$

and $\hat{C}(0) = 0$. Imposing that $\hat{C}(\omega)$ is rank-1 is actually a necessary and sufficient condition to be a SN model for some image f_0.

3.3. Texton Computation.

Generic texton. Following [8], we define a texton as a basic element that parameterizes a Gaussian texture model. This texton parameterization is useful to perform texture synthesis, since it allows for a painless resizing of texture models. And it is also at the heart of the texture mixing method described in the following sections. A texton $\mathcal{K} \in \mathbb{R}^{Ud \times Ud}$ associated to a Gaussian distribution $\mathcal{N}(\mathfrak{m}, \mathcal{C})$ is any matrix such that $\mathcal{C} = \mathcal{K}\mathcal{K}^*$.

In the case of a stationary Gaussian process, the covariance C is a convolution operator (see (2.6)). A texton is thus any convolution operator $\mathcal{K}$ with convolution kernel K which Fourier transform satisfies

$$\hat{C}(\omega) = \hat{K}(\omega)\hat{K}(\omega)^*, \quad \text{where} \quad \hat{K}(\omega) \in \mathbb{C}^{d \times d}.$$

Recall that $\hat{C}$ is the Fourier transform of the correlation kernel C and that it is defined in (2.5). It is thus possible to compute a texton using for instance a Cholesky decomposition of $\hat{C}(\omega)$, see for instance [18].

Canonical texton – general case. Following [9], we define a texton which is uniquely associated to a given texture model. The canonical texton $\mathcal{K}$ associated to a Gaussian distribution $\mathcal{N}(\mathfrak{m}, \mathcal{C})$ is its unique real and symmetric positive texton.

We denote the SVD factorization of $\hat{C}(\omega)$ by

$$\hat{C}(\omega) = \hat{Q}(\omega)\,\mathrm{diag}(\lambda_1(\omega), \ldots, \lambda_d(\omega))\hat{Q}(\omega)^* \in \mathbb{C}^{d\times d} \tag{3.6}$$

where $(\lambda_1(\omega), \ldots, \lambda_d(\omega)) \in \mathbb{R}_+^d$ are the eigenvalues value of $\hat{K}(\omega)$, and $\hat{Q}(\omega) \in \mathbb{C}^{d\times d}$ is the unitary matrix of eigenvectors.

The canonical texton can be computed in $O(Nd\log(Nd))$ operations over the Fourier domain,

$$\hat{K}(\omega) = \hat{Q}(\omega)\,\mathrm{diag}(\lambda_1(\omega)^{1/2}, \ldots, \lambda_d(\omega)^{1/2})\hat{Q}(\omega)^* \in \mathbb{C}^{d\times d} \tag{3.7}$$

which only involves the diagonalization of $d \times d$ matrices.

This canonical texton $\mathcal{K}$ can thus be stored as a set $(K(p)_{i,j})_{1\leq i\leq j\leq d}$ of $d(d+1)/2$ filters, and together with m they completely characterize a Gaussian texture model. Note also that this canonical texton is in general different from the rank-1 color texton introduced in [8].

Canonical texton – SN case. When the model $\mathcal{N}(\mathfrak{m}, \mathcal{C})$ is a SN model learned from some input exemplar f_0, the covariance frequencies are rank-1 (see (3.5)). The canonical texton, that we call SN-texton, obeys the same property, and it can be computed in the Fourier domain as

$$\forall\,\omega, \quad \hat{K}(\omega) = \frac{1}{\sqrt{N}}\frac{\hat{f}(\omega)\hat{f}(\omega)^*}{|\hat{f}(\omega)|} \in \mathbb{C}^{d\times d}. \tag{3.8}$$

where $|v|$ is the modulus of v. In the special case when $d = 1$ (gray-scale images), one obtains the original texton introduced by Moisan [8]

$$\forall\,\omega, \quad \hat{K}(\omega) = \frac{1}{\sqrt{N}}|\hat{f}(\omega)| \in \mathbb{R}^+,$$

that has several interesting optimality properties in term of compactness, see [8]. It is unclear wether similar properties also hold in the vectorial case.

The obtained SN-Texton $K \in \mathbb{R}^{N\times d}$ is a set of $d(d+1)/2$ space-time 3-D filters. An example of such SN-Textons is displayed in Figure 5.1. The numerical experiments presented in Section 5 show that, in practice, the obtained SN-Texton is highly compact in space and time.

3.4. Model Resizing. An important requirement of texture synthesis methods is to be able to generate texture of an arbitrary size $(\tilde{N}_i)_{i=1}^d$. Let us denote by $\tilde{U}$ the new pixel domain of dimension $N = \prod_i \tilde{N}_i$, and its associated Gaussian distribution $\mathcal{N}(\tilde{\mathfrak{m}}, \tilde{\mathcal{C}})$ where $\tilde{\mathfrak{m}} \in \mathbb{R}^{\tilde{U}d}$ and $\tilde{\mathcal{C}} \in \mathbb{R}^{\tilde{U}d\times\tilde{U}d}$, which can either extrapolate (when $\tilde{N}_i > N_i$) or restrict (when $\tilde{N}_i < N_i$) the initial model.

Since $\mathfrak{m}(p) = m$ is constant, one defines $\tilde{\mathfrak{m}}(p) = m$. A naive approach to obtain $\tilde{\mathcal{C}}$ would be to either crop or zero-pad the original covariance C. This is not an acceptable method, since $\tilde{\mathcal{C}}$ would not be positive and symmetric, in general. Following the idea introduced in [8] for the rank-1 color-texton, we propose to crop or zero pad the kernel K associated to the canonical texton $\mathcal{K}$.

Another reason for introducing a pixel domain $\tilde{U}$ that differs from the initial domain U is to ensure causality of the covariance factorization, meaning $K(x, t) = 0$

when $t \leq 0$. Denoting $\tilde{N}_3$ the number of time frames of the cropped texton, one can simply perform a time-shift of the texton (which does not modify the covariance $\tilde{\mathcal{C}}$), which is equivalent to not centering the texton at 0.

We thus denote by $\delta = (\delta_x, \delta_t)$ the center of the target domain $\tilde{U}$, where usually $\delta_x = 0$ and, to compute a time causal texton, $\delta_t = -\tilde{N}_3/2$. The resized and translated texton is

$$\forall\, p \in \tilde{U}, \quad \tilde{K}(p) = \begin{cases} K(p-\delta) & \text{if} \quad p - \delta \in U, \\ 0 & \text{otherwise.} \end{cases} \tag{3.9}$$

3.5. Texture Synthesis. When the texton $\tilde{\mathcal{K}}$ of a model $\mu = \mathcal{N}(\mathfrak{m}, \tilde{\mathcal{C}})$ has been defined (either as the SN texton of an input texture, or from the mixing process described in the following sections), one can easily perform texture synthesis. This corresponds to computing a sample $f \in \mathbb{R}^{U \times d}$ drawn from the distribution μ. Depending on the properties of the texton (spacial extent and/or time causality), one can use either Fourier-based, convolution-based, or a mixed formulation.

Time-periodic synthesis. A time-periodic synthesis is achieved in $O(Nd \log(Nd))$ operations by sampling independently each Fourier frequency as follows

$$\forall\, \omega \in U, \quad \hat{f}(\omega) = \begin{cases} N\mathfrak{m} & \text{if} \quad \omega = 0, \\ \widehat{\tilde{K}}(\omega)\hat{w}(\omega) & \text{if} \quad \omega \neq 0 \end{cases} \quad \text{where } w \equiv \mathcal{N}(0, \mathrm{Id}_{Ud \times Ud}) \tag{3.10}$$

where $\equiv$ means "sampled from" (i.e. it is a realization, not a random variable).

Infinite time synthesis. We assume that the resized texton $\tilde{\mathcal{K}}$ has been shifted so that it is causal, i.e. $K(x,t) = 0$ for $t \leq 0$. This allows us to generate the texture $f(x,t) = f_t(x)$ in a frame-by-frame manner using, for each time step, noise blocks of reduced size

$$w_t = (w_t^{(1)}, \ldots, w_t^{(\tilde{N}_3)}) \in \mathbb{R}^{\tilde{N}_1 \times \tilde{N}_2 \times \tilde{N}_3 \times d} \quad \text{where} \quad w_t^{(k)} \equiv \mathcal{N}(0, \mathrm{Id}_{\tilde{N}_1 \times \tilde{N}_2}).$$

For each $t \geq 0$, the sampled frame is computed as

$$f_t = \mathfrak{m} + \sum_{k=1}^{\tilde{N}_3} w_t^{(k)} \star K(\cdot, 1-k) \tag{3.11}$$

where $\star$ is the 2-D space-only convolution that can be implemented using either the spatial definition of the convolution (see (2.2) in the special case $N_3 = 1$) or the Fourier domain formulation (see (2.3)).

The block of noise is then updated by a time-shift and by adding new noise $w^{(0)}$

$$w_{t+1} = (w^{(0)}, w_t^{(1)}, \ldots, w_t^{(\tilde{N}_3 - 1)}) \quad \text{where} \quad w^{(0)} \equiv \mathcal{N}(0, \mathrm{Id}_{\tilde{N}_1 \times \tilde{N}_2}).$$

4. Auto-regressive Texture Models. We now introduce our second Gaussian texture model, which is a specialization of the auto-regressive (AR) model of Doretto *et al.* [11] to the setting of stationary random fields. A chief advantage of this specialization is that it does not require a PCA dimensionality reduction since the texture analysis is performed over the 2-D reduced frequency domain, and the synthesis is achieved using fast spatial convolutions. This setting differs significantly from the stationary model introduced in [12] which makes use of complicated tree-based spatial AR model. Our numerical tests reported in Section 5 seem to indicate that our method gives results visually similar to the ones presented by Doretto et al. in [12].

While the SN model is non-parametric (under the condition that the covariance has rank-1 frequencies), the AR model reduces drastically the number of free parameters by imposing a recursive relationship. We focus on the class of AR(1) models, although a more general recurrence (AR(p) and ARIMA) could be treated exactly in the same way.

The whole pipeline (analysis and synthesis) for the AR model is summarized in Algorithm 2. The following sections describe in detail each step of the process.

Algorithm 2: Texture synthesis by using AR-textons

Input: exemplar $\tilde{f}_0 \in \mathbb{R}^{U \times d}$.
Output: sample $f \in \mathbb{R}^{\tilde{U} \times d}$ of the AR model.

1. **Pre-processing.** Compute f_0 from $\tilde{f}_0$ using (4.7).
2. **SN texture analysis.**
   - Compute $\mathcal{A}$ from f_0 using (4.8).
   - Compute $\mathcal{B}$ from f_0 and $\mathcal{A}$ using (4.9).
3. **AR texton computation.** Compute $\mathcal{L}$ from $\mathcal{B}$ using (4.10).
4. **Model resizing.** Compute $(\tilde{\mathcal{A}}, \tilde{\mathcal{L}})$ from $(\mathcal{A}, \mathcal{L})$ using (4.11).
5. **Texture synthesis.** Compute f from $(\tilde{\mathcal{A}}, \tilde{\mathcal{L}})$ using (4.12). Remove the $|t_0|$ initial frames.

4.1. Stationary AR Processes. We consider an infinite time domain $U = U_s \times \mathbb{Z}$ where $U_s = \{1, \dots, N_1\} \times \{1, \dots, N_2\}$. A Gaussian random vector $X = (X_t)_{t \in \mathbb{Z}}$ with values in $\mathbb{R}^{U \times d}$ is distributed according to an auto-regressive AR(1) model parameterized by $(\mathcal{A}, \mathcal{B}, E)$ if it satisfies

$$\forall\, t \in \mathbb{Z}, \quad X_{t+1} = E + \mathcal{A} X_t + \bar{W}_t, \tag{4.1}$$

where $\mathcal{A}, \mathcal{B} : \mathbb{R}^{(U_s d) \times (U_s d)}$ are space-only linear operators, $E \in \mathbb{R}^{U_s}$, and $\bar{W}_t \sim \mathcal{N}(0, \mathcal{B})$. If $\mathcal{L}$ is any texton factorization of $\mathcal{B}$, i.e. $\mathcal{B} = \mathcal{L}\mathcal{L}^*$, then one can write $\bar{W}_t = \mathcal{L} W_t$ for $W_t \sim \mathcal{N}(0, \mathrm{Id}_{U_s \times d})$.

To ensure that this AR process is stationary, we impose that the operators $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{L}$ are convolutions with their respective kernels $A, B, L \in \mathbb{R}^{U_s \times d}$, and that

$$\forall\, p \in U_s, \quad E(p) = e \in \mathbb{R}^d,$$

so that

$$\forall\, t \in \mathbb{Z}, \quad X_{t+1} = E + A \star X_t + L \star W_t \in \mathbb{R}^{U_s \times d}, \tag{4.2}$$

where $\star$ is the 2-D space-only convolution (see Equation (2.2) for $N_3 = 1$). The modulus of the eigenvalues of $\mathcal{A}$ needs to be strictly smaller than one.

We denote by $\hat{X}_t$ the 2-D Fourier transform of X_t (in the special case where $N_3 = 1$, see Equation (2.1)). The AR recursion boils down to a low-dimensional $d \times d$ recursion for each frequency $\omega \in U_s$

$$\forall\, t \in \mathbb{Z}, \quad \hat{X}_{t+1}(\omega) = \hat{E}(\omega) + \hat{A}(\omega)\hat{X}_t(\omega) + \hat{L}(\omega)\hat{W}_t(\omega) \in \mathbb{C}^d \tag{4.3}$$

$$\text{where} \quad \hat{E}(\omega) = \begin{cases} N_1 N_2 e & \text{if} \quad \omega = 0, \\ 0 & \text{if} \quad \omega \neq 0. \end{cases} \tag{4.4}$$

The following proposition summarizes the main properties of AR(1) processes.

PROPOSITION 4.1. *An AR(1) process $(X_t)_t$ satisfying Equation (4.1) exists if*

$$\forall\, \omega \in U_s, \quad \hat{A}(\omega) \in \Delta_d \quad \textit{where} \quad \Delta_d = \left\{ a \in \mathbb{C}^{d\times d} \setminus \forall\, i = 1, \ldots, |\lambda_i(a)| < 1 \right\} \tag{4.5}$$

where $(\lambda_i(a))_{i=1^d} \subset \mathbb{C}$ is the set of eigenvalues of a matrix $a \in \mathbb{C}^{d\times d}$. Its distribution $\mathcal{N}(\mathfrak{m}, \mathcal{C})$ is then uniquely defined and it is space and time stationary Gaussian. Its mean satisfies

$$\forall\, p \in U, \quad \mathfrak{m}(p) = (\mathrm{Id}_d - \hat{A}(0))^{-1} e \in \mathbb{R}^d.$$

Its covariance $\mathcal{C}$ is a convolution with kernel $C = (C_t)_t \in \mathbb{R}^{U_s \times \mathbb{Z}}$. Each 2-D (space only) Fourier transform matrix $\hat{C}_t(\omega) \in \mathbb{C}^{d\times d}$ are

$$\hat{C}_t(\omega) = \begin{cases} \beta(\hat{A}(\omega), \hat{B}(\omega)), & \textit{if} \quad t = 0 \\ A(\omega)^t \hat{C}_0(\omega) & \textit{if} \quad t > 0 \\ \hat{C}_0(\omega) A(\omega)^{-t,*} & \textit{if} \quad t < 0 \end{cases}$$

where for $a \in \Delta_d$ and for a symmetric $b \in \mathbb{C}^{d\times d}$, $\beta = \beta(a, b) \in \mathbb{C}^{d\times d}$ is the unique (symmetric) matrix which is solution to the linear equation

$$\beta = a\beta a^* + b. \tag{4.6}$$

Proof. These are classical results on multichannel AR models, here we only sketch the proof. We refer for instance to [24] for more details about AR models and their properties. We consider an AR(1) process satisfying

$$\forall\, t \in \mathbb{Z}, \quad x_{t+1} = a x_t + w_t \in \mathbb{C}^d \quad \text{with} \quad w_t \sim \mathcal{N}(0, b)$$

where $a, b \in \mathbb{C}^{d\times d}$. In our case, for some ω, $x_t = \hat{X}_t(\omega)$, $a = \hat{A}(\omega)$ and $b = \hat{B}(\omega)$. If such an x_t exists and is stationary, $\mathbb{E}(x_t) = m \in \mathbb{C}^d$ and

$$\mathbb{E}((x_{t+1} - m)(x_{t+1} - m)^*) = a\mathbb{E}((x_t - m)(x_t - m)^*)a^* + \mathbb{E}(w_t w_t^*)$$

and thus the covariance β of x_t, for any t, satisfies $T_a(\beta) = b$ where $T_a(\beta) = \beta - a\beta a^*$. If one denotes $(u_i \in \mathbb{C}^d)_{i=1}^d$ and $(\lambda_i \in \mathbb{C})_{i=1}^d$ the eigenvectors and eigenvalues of a, one verifies that for all $(i, j) \in \{1, \ldots, d\}^2$, $u_i u_j^* \in \mathbb{C}^{d\times d}$ are the eigenvectors of T_a, with corresponding eigenvalues $1 - \lambda_i \lambda_j^*$. Since these eigenvalues are non-vanishing, T_a is invertible and β is uniquely defined. A recursion shows that the covariance between pairs of frames satisfies, for $\delta \geq 1$

$$\mathbb{E}((x_{t+\delta} - m)(x_t - m)^*) = a\mathbb{E}((x_{t+\delta-1} - m)(x_t - m)^*) = a^\delta \beta$$

(and similarly for $\delta \leq 0$).

The fast decay of this covariance as δ increases shows that this equation defines a covariance with bounded ℓ^2 norm on $\mathbb{C}^{d\times\mathbb{Z}}$, and that the corresponding Gaussian process satisfies by construction the recursion. □

4.2. Pre-processing. The exemplar is a video $\tilde{f}_0 = (\tilde{f}_{0,t})_{t=0}^{N_3-1} \in \mathbb{R}^{U_s \times N_3 \times d}$ of N_3 frames. If $\tilde{f}_0$ is not periodic in space, we extract its periodic component, similarly to (3.1), but only in space, since the estimation process for the AR model does not

use a Fourier transform in time. We thus solve (3.1) independently for each t, which reads

$$\forall\, t = 0, \ldots, N_3 - 1, \quad \begin{cases} \Delta f_{0,t} = \Delta_i \tilde{f}_{0,t} \\ \text{mean}(f_{0,t}) = \text{mean}(\tilde{f}_{0,t}). \end{cases} \tag{4.7}$$

Similarly to (3.2), this computation can be carried over in $O(Nd \log(Nd))$ operations over the Fourier domain.

4.3. Learning texture parameters. Without loss of generality, we assume that the mean of the AR(1) process is zero, so that $e = 0$ and the input exemplar is pre-processed by removing its empirical mean. The parameters $(\mathcal{A}, \mathcal{B})$ of the model are learned from the pre-processed exemplar $f_0 = (f_{0,t})_{t=0}^{N_3-1} \in \mathbb{R}^{U_s \times N_3 \times d}$ of N_3 sample frames. We follow [11] and we compute an estimate by solving the Yule-Walker equations [27], which can be shown to be a consistent estimator when $N_3 \to +\infty$.

Learning $\mathcal{A}$. The first step, the estimation of $\mathcal{A}$ can be understood as a least square estimate assuming $\mathcal{B} = 0$

$$\min_{\mathcal{A}} \sum_{t=0}^{N_3-2} \|f_{0,t+1} - \mathcal{A} f_{0,t}\|^2.$$

Since the model is assumed to be stationary, it has the form (4.3), and the estimation can be performed independently over each frequency $\omega \in U_s$

$$\min_{\hat{A}(\omega) \in \mathbb{C}^{d \times d}} \sum_{t=0}^{N_3-2} \|\hat{f}_{0,t+1}(\omega) - \hat{A}(\omega) \hat{f}_{0,t}(\omega)\|^2.$$

for which the solution can be computed in closed form by inverting a small $d \times d$ matrix

$$\forall\, \omega \in U_s, \quad \hat{A}(\omega) = \left(\sum_{t=0}^{N_3-2} \hat{f}_{0,t+1}(\omega) \hat{f}_{0,t}(\omega)^* \right) \left(\sum_{t=0}^{N_3-2} \hat{f}_{0,t}(\omega) \hat{f}_{0,t}(\omega)^* \right)^{-1} \tag{4.8}$$

Learning $\mathcal{B}$. Once $\mathcal{A}$ is known, the parameter $\mathcal{B}$ is estimated from the residual

$$\forall\, t = 0, \ldots, N_3 - 2, \quad R_t = f_{0,t+1} - \mathcal{A} f_{0,t}$$

which should be distributed according to $\mathcal{N}(0, \mathcal{B})$. One thus estimates $\mathcal{B}$ using the MLE estimator of a Gaussian process which is stationary in space from the time samples. This reads

$$\forall\, \omega \in U_s, \quad \hat{B}(\omega) = \frac{1}{N_3 - 1} \sum_{t=0}^{N_3-2} \hat{R}_t(\omega) \hat{R}_t(\omega)^*. \tag{4.9}$$

4.4. AR-Texton. The SVD decomposition of $\hat{B}(\omega)$ reads

$$\forall\, \omega, \quad \hat{B}(\omega) = \hat{V}(\omega)\, \text{diag}(\rho_1(\omega), \ldots, \rho_d(\omega)) \hat{V}(\omega)^* \in \mathbb{C}^{d \times d}$$

where $(\rho_1(\omega), \ldots, \rho_d(\omega)) \in \mathbb{R}_+^d$ are the eigenvalues of $\hat{B}(\omega)$, and $\hat{V}(\omega) \in \mathbb{C}^{d \times d}$ is the unitary matrix of eigenvectors.

We denote by L the canonical texton (compare to (3.6) for the SN case) associated to $\mathcal{B}$

$$\hat{L}(\omega) = \hat{V}(\omega)\, \text{diag}(\rho_1(\omega)^{1/2}, \ldots, \rho_d(\omega)^{1/2}) \hat{V}(\omega)^* \in \mathbb{C}^{d \times d} \tag{4.10}$$

We call the pair $(\mathcal{A}, \mathcal{L})$ the AR-texton associated to the AR model.

4.5. Resizing. The texton $(\mathcal{A}, \mathcal{L})$ can be resized to any new spatial domain $\tilde{U_s} = \{0, \dots, \tilde{N}_1\} \times \{0, \dots, \tilde{N}_2 - 1\}$ by cropping/zero-padding the initial kernels (A, L) into $(\tilde{\mathcal{A}}, \tilde{\mathcal{L}})$

$$\forall\, x \in \tilde{U_s}, \quad \tilde{A}(x) = \begin{cases} A(x) & \text{if} \quad p \in U_s, \\ 0 & \text{otherwise,} \end{cases} \tag{4.11}$$

(and similarly for $\mathcal{L}$).

4.6. Texture Synthesis. Once the texton $(\tilde{\mathcal{A}}, \tilde{\mathcal{L}})$ has been learned, a texture f is obtained by sampling the model (4.3) from an initial time t_0

$$\forall\, t \geq t_0, \quad f_{t+1} = A \star f_t + L \star w_t \quad \text{where} \quad w_t \equiv \mathcal{N}(0, \mathrm{Id}_{U_s \times d}) \tag{4.12}$$

where f_{t_0} is drawn from any distribution (for instance $f_{t_0} = 0$).

We should note that a resulting texture is sampled from a time-stationary process only in the limit $t_0 \to -\infty$. In practice, only a few time steps $t_0 \leq t < 0$ are needed to reach almost stationarity. We thus initialize the sampling at some initial time t_0, we generate $\tilde{N}_3 + t_0$ frames, and we then discard the first t_0 samples, and we consider $f = (f_t)_{t=0}^{\tilde{N}_3 - 1}$ where $\tilde{N}_3$ is the number of desired frames.

5. Numerical Results for Synthesis. In this section, following a brief introduction of the experimental setup, we discuss the performance of Spot Noise textons, AR textons and compare them with the results obtained by Doretto *et al.* in [12].

Experimental setup. Though there are several dynamic texture datasets, such as for instance DynTex [28] and DynTex++ [19], none is available both for the analysis and synthesis of SDTs. In order to test the synthesis algorithm, we compiled a dataset of SDTs containing 35 different color dynamic textures.Each sequence is of spatial size 64×64 pixels and with 100 frames.

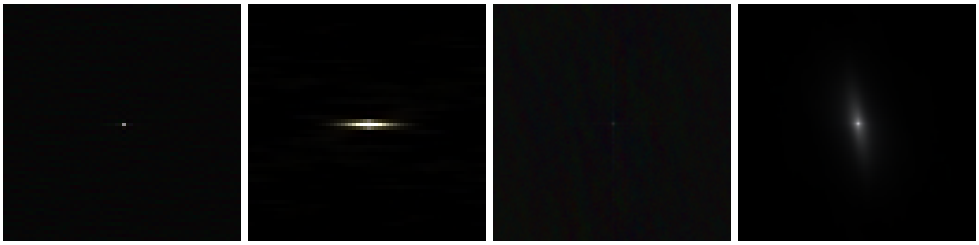
Comparing SN-textons with AR-textons. Figure 5.1 presents the SN-textons and AR-textons learned from two exemplar textures *moving goldenlines* and *waterfall*, respectively. It shows the fast decay in space and time of the learned textons. Observe that the SN-textons are 6 3-D space-time filters, while the AR-textons are 6 2-D spatial filters. Instead of showing all the 6 filters, (b) and (c) only display the diagonal elements (r, g, and b for the red, green and blue channels, which results in a color image) of the textons. Note that the synthesized results of these two Gaussian models are visually comparable.

Comparing AR-textons with LDS [12]. We now compare our method with the one proposed in [12], which uses multiscale autoregressive models. Figure 5.2 shows the synthesized results for two dynamic sequences: *waterfall* and *fire* (courtesy of Doretto et al. [12]). Both models capture the temporal and spatial stationarity of the exemplar. They produce quite similar results on these sequences. Both exhibit artifacts when the input texture is not stationary, which can be observed on the results of the fire sequence, Figure 5.2 (right).

6. Geodesic Mixing Two Gaussian Models. In this section, we discuss the problem of mixing two textures, which corresponds to computing the geodesics between two Gaussian models according to some distance between distributions. We give the explicit solution of this problem in the case of the L^2 optimal transport (OT) distance for SN models. Moreover, we give an approximate solution for AR models by approximating the full distribution of the space-time video by the distribution of two consecutive frames.

(a) 2 consecutive frames of the exemplar textures video f_0: *goldenlines* and *waterfall*

(b) 2 frames of the diagonal elements ($K_{r,r}$, $K_{g,g}$, $K_{b,b}$) of the learned 3D canonical SN-Texton.



(c) The diagonal elements ($(A, L)_{r,r}$, $(A, L)_{g,g}$, $(A, L)_{b,b}$) of the learned 2D AR-Texton.

Figure 5.1: Learned textons from two stationary dynamic textures. (a) 2 consecutive frames of the original texture videos. (b) 2 consecutive frames of the diagonal elements ($K_{r,r}$, $K_{g,g}$, $K_{b,b}$) of the learned canonical SN-texton; (c) diagonal elements ($A_{r,r}, A_{g,g}, A_{b,b}$) and ($L_{r,r}, L_{g,g}, L_{b,b}$) of the learned canonical AR-texton.

6.1. Optimal transport distance. We consider two distributions μ_0, μ_1 defined on $\mathbb{R}^P$, for instance $P = Nd$ when dealing with dynamic textures. The square of the L^2 optimal transport (OT) distance between two distributions is defined as

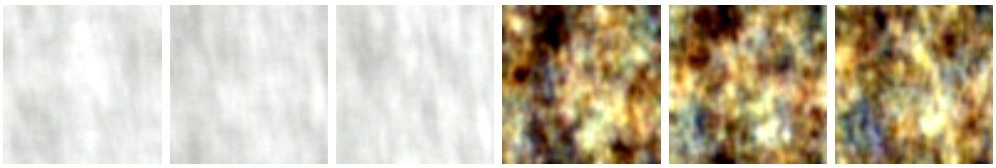
$$d_{\mathrm{OT}}(\mu_0, \mu_1)^2 = \min_{Z \in C(\mu_0, \mu_1)} \int_{\mathbb{R}^P \times \mathbb{R}^P} \|y_0 - y_1\|^2 d\mathbb{P}_Z(y_0, y_1),$$

where $Z \in C(\mu_0, \mu_1)$ is a joint distribution having marginal distributions μ_0, μ_1, i.e.

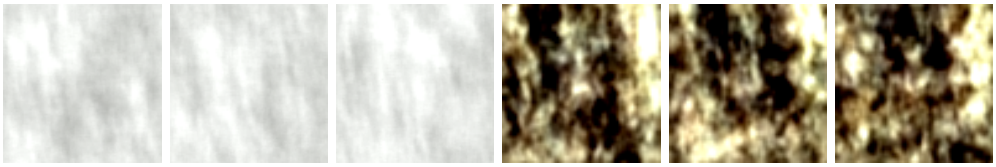
$$\int_{U \times \mathbb{R}^P} d\mathbb{P}_Z(y_0, y_1) = \mu_0(U), \quad \text{and} \quad \int_{\mathbb{R}^P \times V} d\mathbb{P}_Z(y_0, y_1) = \mu_1(V)$$

for all measurable sets $U, V \subset \mathbb{R}^P$. Intuitively, $d_{\mathrm{OT}}(\mu_0, \mu_1)$ measures the amount of work (supposed to be proportional to the squared L^2 distance) needed to displace the mass of the distribution μ_0 to the mass of μ_1. More details about OT can be found for instance in [41].

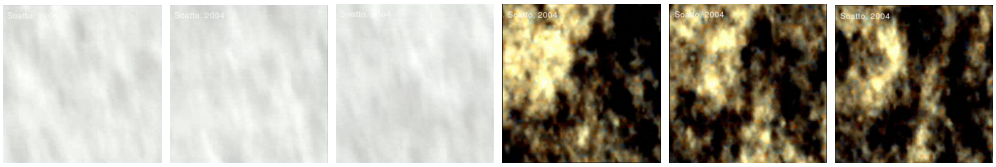
(a) 3 consecutive frames of the input videos.



(b) 3 frames of SN-texton synthesis



(c) 3 frames of AR-texton synthesis



(d) 3 frames of synthesis with [12].

Figure 5.2: Results on stationary dynamic texture synthesis using SN-textons, AR-textons and the method of [12].

The OT distance between Gaussian distributions $\mu_i = \mathcal{N}(\mathfrak{m}_i, \mathcal{C}_i)$, $i = 0, 1$ can be computed explicitly as

$$d_{\mathrm{OT}}(\mu_0, \mu_1)^2 = \mathrm{tr}(\mathcal{C}_0 + \mathcal{C}_1 - 2\mathcal{C}_{0,1}) + \|\mathfrak{m}_0 - \mathfrak{m}_1\|^2, \tag{6.1}$$

$$\text{where} \quad \mathcal{C}_{0,1} = (\mathcal{C}_1^{1/2}\mathcal{C}_0\mathcal{C}_1^{1/2})^{1/2}, \tag{6.2}$$

where $A^{1/2}$ is the unique semi-definite positive square root of a symmetric semi-definite positive matrix A. See for instance [13] for a proof of this result.

When $(\mathcal{C}_i)_{i=0,1}$ diagonalize in the same orthogonal basis, with eigenvalues $(\lambda_i^k)_{k=1}^P$, then the OT distance between $(\mu_i = \mathcal{N}(\mathfrak{m}_i, \mathcal{C}_i))_{i=0,1}$ is

$$d_{\mathrm{OT}}(\mu_0, \mu_1)^2 = \|\mathfrak{m}_0 - \mathfrak{m}_1\|^2 + \sum_{k=1}^{P} \left(\sqrt{\lambda_0^k} - \sqrt{\lambda_1^k}\right)^2.$$

6.2. Geodesic Interpolation.

Barycentric interpolation between two distributions. We consider a distance d between distributions (for instance $d = d_{\mathrm{OT}}$). A barycentric path linking μ_0 to μ_1 is defined as

$$\forall\, \rho \in [0, 1], \quad \mu_\rho = \operatorname*{argmin}_{\mu} (1 - \rho)\, d(\mu_0, \mu)^2 + \rho\, d(\mu_1, \mu)^2, \tag{6.3}$$

where $\rho \in [0,1] \mapsto \mu_\rho$ parameterizes the path (we assume existence and uniqueness of such minimizer).

Connection with geodesics. The OT distance is a geodesic distance. This means that $d_{\mathrm{OT}}(\mu_0,\mu_1)$ is equal to the length of the shortest path (the so-called geodesic path) between μ_0 and μ_1, i.e.

$$d_{\mathrm{OT}}(\mu_0,\mu_1) = \operatorname*{argmin}_{\mu_\rho} \mathcal{L}_{d_{\mathrm{OT}}}(\mu_\rho)$$

where the length of a path for a metric d is

$$\mathcal{L}_d(\mu) = \inf_{K\in\mathbb{N}^*, 0=t_0<t_1<\ldots<t_K=1} \sum_{k=1}^{K} d(\mu_{t_{k-1}},\mu_{t_k}).$$

In this case, $d_{\mathrm{OT}}(\mu_0,\mu_1) = \mathcal{L}_{d_{\mathrm{OT}}}(\mu_\rho)$ where μ_ρ solves (6.3) and is a geodesic path.

6.3. OT Geodesics of Gaussian Distributions. The following proposition shows that the set of Gaussian distributions is geodesically convex for the OT-distance, see [38].

PROPOSITION 6.1. *If* $\ker(\mathcal{C}_0) \not\subset \ker(\mathcal{C}_1)^\perp$ *and* $\operatorname{rank}(\mathcal{C}_0) \geq \operatorname{rank}(\mathcal{C}_1)$*, we define a Gaussian distribution* $\mu_\rho = \mathcal{N}(\mathfrak{m}_\rho, \mathcal{C}_\rho)$ *with*

$$\forall\, \rho \in [0,1], \quad \begin{cases} \mathfrak{m}_\rho = (1-\rho)\,\mathfrak{m}_0 + \rho\,\mathfrak{m}_1, \\ \mathcal{C}_\rho = [(1-\rho)\operatorname{Id} + \rho\Pi]\mathcal{C}_0[(1-\rho)\operatorname{Id} + \rho\Pi], \end{cases} \tag{6.4}$$

$$\text{where} \quad \Pi = \mathcal{C}_1^{1/2}\mathcal{C}_{0,1}^{+}\mathcal{C}_1^{1/2} \tag{6.5}$$

with $\mathcal{C}_{0,1}$ *defined by Equation* (6.2)*,* A^+ *is the Moore-Penrose pseudo-inverse of a matrix* A*. It is the unique Gaussian OT-geodesic between* $\mu_i = \mathcal{N}(\mathfrak{m}_i, \mathcal{C}_i)$ *for* $i = 0, 1$*.*

Proof. The linear interpolation of the mean is a well known result, see [41]. We thus assume that $\mathfrak{m}_0 = \mathfrak{m}_1 = 0$. The proof follows the one in [38] taking extra care of the rank-deficient matrix case. Classical results (see e.g. [41]) ensures that a map $T : \mathbb{R}^P \to \mathbb{R}^P$ is an L^2 OT between μ_0 and μ_1 if and only if T is the gradient of a convex function and $T\sharp\mu_0 = \mu_1$ (where $\sharp$ is the push-forward operator). An OT geodesic between μ_0 and μ_1 is then defined by $\mu_\rho = ((1-\rho)\operatorname{Id} + \rho T)\sharp\mu_0$. In our case, if the transport is linear, this means that T is symmetric and $T\mathcal{C}_0T = \mathcal{C}_1$. We first prove that Π defined in (6.5) satisfies this equality. Indeed, condition $\ker(\mathcal{C}_0) \not\subset \ker(\mathcal{C}_1)^\perp$ and $\operatorname{rank}(\mathcal{C}_0) \geq \operatorname{rank}(\mathcal{C}_1)$ ensures that

$$\Pi\mathcal{C}_0\Pi = \mathcal{C}_1^{1/2}(\mathcal{C}_{0,1}^{1/2})^+\mathcal{C}_{0,1}^{1/2}(\mathcal{C}_{0,1}^{1/2})^+\mathcal{C}_1^{1/2} = \mathcal{C}_1^{1/2}\mathrm{Proj}_{\mathrm{Im}(\mathcal{C}_1)}\mathcal{C}_1^{1/2} = \mathcal{C}_1$$

where $\mathrm{Proj}_{\mathrm{Im}(\mathcal{C}_1)}$ is the orthogonal projector on $\mathrm{Im}(\mathcal{C}_1)$. A Gaussian barycenter necessarily corresponds to a transport $\tilde{\Pi}$ which is linear from $\mathrm{Im}(\mathcal{C}_0)$ to $\mathrm{Im}(\mathcal{C}_1)$. It satisfies $\tilde{\Pi}\mathcal{C}_0\tilde{\Pi} = \mathcal{C}_1 = \Pi\mathcal{C}_0\Pi$, and hence, by uniqueness of the matrix square root, $\mathcal{C}_0^{1/2}\tilde{\Pi}\mathcal{C}_0^{1/2} = \mathcal{C}_0^{1/2}\Pi\mathcal{C}_0^{1/2}$. Then, $\tilde{\Pi}$ and Π are equal on $\mathrm{Im}(\mathcal{C}_0)$ and thus changing Π by $\tilde{\Pi}$ in (6.4) defines the same geodesic $\mathcal{C}_\rho$. □

When $\mathcal{C}_0$ or $\mathcal{C}_1$ are full rank, the geodesic μ_ρ is known to be unique, and it is thus the one defined in (6.4). Note also that if the hypothesis $\operatorname{rank}(\mathcal{C}_0) \geq \operatorname{rank}(\mathcal{C}_1)$ is not satisfied, one can exchange $\mathcal{C}_0$ and $\mathcal{C}_1$ and compute the geodesic $\mathcal{C}_\rho$ after exchanging ρ by $1-\rho$.

When $\mathcal{C}_0$ and $\mathcal{C}_1$ diagonalize in the same ortho-basis of $\mathbb{R}^P$, $\mathcal{C}_\rho$ diagonalizes in the same basis and its eigenvalues $(\lambda_\rho^k)_{k=1}^P$ are

$$\forall\, k = 1,\ldots,P, \quad \sqrt{\lambda_\rho^k} = (1-\rho)\sqrt{\lambda_0^k} + \rho\sqrt{\lambda_1^k}.$$

6.4. Comparison with Other Metrics. While we focus our attention to the OT interpolation of Gaussian models, many other distances could be used as well. Let us however stress several key features that makes the use of OT appealing:

- The set of Gaussian distributions are geodesically convex.
- The geodesic path μ_ρ can be computed in closed form, see (6.4).
- The distance is defined and finite even for rank-deficient covariances (such as the SN covariance for color texture, see (3.5)).
- The geodesic between rank-r covariance is rank-r, which is exploited in Section 6.5 when computing the geodesic of SN models in the case $r = 1$ (rank-1 covariances).

To gain a better insight about this interpolation, let us focus in the special case of 1-D Gaussian ($P = 1$), and compare OT with other interpolations. In this case, $\mu_i = \mathcal{N}(\mathfrak{m}_i, \mathcal{C}_i)$ for $i = 0, 1$ where $\mathfrak{m}_i \in \mathbb{R}$ and $\mathcal{C}_i = \sigma_i^2$ where $\sigma_i \in \mathbb{R}^+$ is the standard deviation.

Optimal Transport interpolation. The OT interpolation $\mathcal{N}(\mathfrak{m}_\rho, \mathcal{C}_\rho = \sigma_\rho^2)$ defines a linear segment in the half plane $(\mathfrak{m}, \sigma) \in \mathbb{R} \times \mathbb{R}^+$

$$\forall\, \rho \in [0,1], \quad (\mathfrak{m}_\rho, \sigma_\rho) = (1-\rho)(\mathfrak{m}_0, \sigma_0) + \rho(\mathfrak{m}_1, \sigma_1).$$

Figure 6.1, left, shows the resulting interpolation, which clearly shows the translation of the mean of the Gaussian.

Linear interpolation. A naive interpolation consists in performing a linear averaging of the densities. This results in a Gaussian mixture (with two mixtures) that is no longer Gaussian. This is illustrated on Figure 6.1, middle.

Fisher-Rao interpolation. As detailed in [3], the Fisher-Rao (FR) geodesic of 1-D Gaussian satisfies

$$\forall\, \rho \in [0,1], \quad (\mathfrak{m}_\rho^{\mathrm{FR}}, \sigma_\rho^{\mathrm{FR}}) = (\mathfrak{m}^{\mathrm{FR}} - \lambda\cos(\varphi_\rho), \lambda\sin(\varphi_\rho)) \tag{6.6}$$

$$\text{where} \quad \forall\, \rho \in (0,1), \quad \varphi_\rho = (1-\rho)\varphi_0 + \rho\varphi_1 \tag{6.7}$$

$$\lambda^2 = \sigma_0^2 + \frac{((\mathfrak{m}_0 - \mathfrak{m}_1)^2 - (\sigma_0^2 - \sigma_1^2))^2}{4(\mathfrak{m}_0 - \mathfrak{m}_1)^2} \tag{6.8}$$

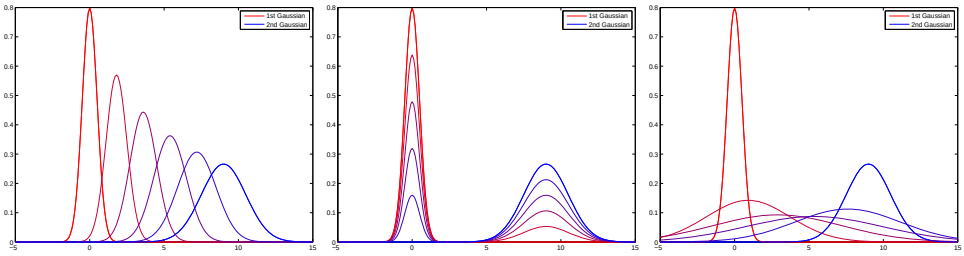
$$\mathfrak{m}^{\mathrm{FR}} = \frac{\mathfrak{m}_0 + \mathfrak{m}_1}{2} + \frac{\sigma_0^2 - \sigma_1^2}{2(\mathfrak{m}_0 - \mathfrak{m}_1)} \tag{6.9}$$

$$\forall\, i = 0, 1, \quad \varphi_i = \sin^{-1}(\sigma_i/\lambda). \tag{6.10}$$

Here, φ_i is taken to be in the interval $]0, \frac{\pi}{2}]$ if $\mathfrak{m}_i \geq \mathfrak{m}^{\mathrm{FR}}$ and in $]\frac{\pi}{2}, \pi]$, otherwise. Geometrically, $(\mathfrak{m}_\rho^{\mathrm{FR}}, \sigma_\rho^{\mathrm{FR}})$ draws an arc jointing μ_0 and μ_1 with radius λ and centered on $(\mathfrak{m}^{\mathrm{FR}}, 0)$. It corresponds to a geodesic in a space with negative curvature (see [2]), as opposed to the OT geometry which has positive curvature (and which is actually Euclidean in the 1-D $(\mathfrak{m}, \sigma)$ parameters).

Moreover, it is worth noticing that no explicit formula is known for the FR geodesic in dimension greater that 1 when the distributions do not have the same mean. Figure 6.1, right, shows the FR interpolation of the densities.

6.5. SN Model Geodesics. We denote $\mathcal{N}(\mathfrak{m}, \mathcal{C}) = \mathcal{S}(f_0)$ a SN texture learned from an input exemplar $f_0 \in \mathbb{R}^{U \times d}$. Its mean and covariances are thus defined by (3.3) and (3.5).



OT interpolation (6.4) Linear interpolation Rao interpolation (6.6)

Figure 6.1: Comparisons of geodesic interpolation for $(\mathfrak{m}_0, \sigma_0) = (0, 0.5)$ and $(\mathfrak{m}_1, \sigma_1) = (9, 1.5)$.

SN models are geodesically convex. The following theorem shows that if the input models μ_0, μ_1 are spot noises, then the geodesic interpolation is also a spot noise. This means that the SN texture model is geodesically convex.

THEOREM 6.2. *For $i = 0, 1$, let $\mu_i \sim \mathcal{S}(f_i)$ be spot noise distributions. We suppose that $\forall \omega, \hat{f}_1(\omega)^* \hat{f}_0(\omega) \neq 0$. The OT geodesic path μ_ρ defined in (6.4) is a spot noise model $\mu_\rho = \mathcal{S}(f_\rho)$ where f_ρ is defined as*

$$\forall\, \rho \in [0,1], \quad f_\rho = (1-\rho)\, f_0 + \rho\, g_1, \tag{6.11}$$

where g_1 is computed from f_1 as

$$\forall\, \omega, \quad \hat{g}_1(\omega) = \hat{f}_1(\omega) \frac{\hat{f}_1(\omega)^* \hat{f}_0(\omega)}{|\hat{f}_1(\omega)^* \hat{f}_0(\omega)|}. \tag{6.12}$$

Proof. The covariance operator $\mathcal{C}_i$ of μ_i is a matrix-convolution operator, and its associated kernel C_i has the following Fourier transform

$$\hat{C}_i(\omega) = \hat{f}_i(\omega) \hat{f}_i(\omega)^* \in \mathbb{C}^{d \times d}. \tag{6.13}$$

The symmetric operator Π defined in Equation (6.5) is thus also a matrix convolution $\Pi g = \pi \star g$ with kernel whose Fourier transform is

$$\hat{\pi}(\omega) = \hat{C}_1^{1/2}(\omega) \left[(\hat{C}_1^{1/2}(\omega) \hat{C}_0(\omega) \hat{C}_1^{1/2}(\omega))^{1/2} \right]^+ \hat{C}_1^{1/2}(\omega).$$

Note that the square root of a rank-1 matrix can be easily computed as

$$\forall\, u \in \mathbb{C}^d, \quad (uu^*)^{1/2} = \frac{1}{|u|} uu^* \in \mathbb{C}^{d \times d}.$$

Using this property, together with Equation (6.13), denoting $u_i = \hat{f}_i(\omega)$ one proves that

$$\hat{\pi}(\omega) = \frac{1}{|u_1^* u_0|} u_1 u_1^* (u_1 u_1^*)^+ u_1 u_1^* = \frac{u_1 u_1^*}{|u_1^* u_0|}. \tag{6.14}$$

Observe that although the matrix $u_1^* u_0$ is non invertible, the above expression is correct because the mapping $\pi(\omega)$ is zero on the orthogonal of u_1.

The expression in Equation (6.4) of the covariance $\mathcal{C}_\rho$ implies that it is also a matrix-convolution operator with kernel C_ρ defined over the Fourier domain as

$$\hat{C}_\rho(\omega) = \hat{f}_\rho(\omega)\hat{f}_\rho(\omega)^* \in \mathbb{C}^{d\times d},$$

where

$$\hat{f}_\rho(\omega) = [(1-\rho)\,\mathrm{Id} + \rho\hat{\pi}(\omega)]u_0 \in \mathbb{C}^d.$$

Using the expression (6.14) for $\hat{\pi}(\omega)$, one thus has that $\mu_\rho = \mathcal{S}(f_\rho)$ is a Spot Noise model where f_ρ is defined as

$$\hat{f}_\rho(\omega) = (1-\rho)\hat{f}_0(\omega) + \rho \underbrace{\frac{\hat{f}_1(\omega)^*\hat{f}_0(\omega)}{|\hat{f}_1(\omega)^*\hat{f}_0(\omega)|}\hat{f}_1(\omega)}_{\hat{g}_1(\omega)} \in \mathbb{C}^d.$$

□

Grayscale SN model geodesic. For grayscale texture, specializing (6.12) to the case $d = 1$ leads to

$$\mu_\rho = \mathcal{S}(\tilde{f}_\rho) \quad \text{where} \quad \tilde{f}_\rho = (1-\rho)\tilde{f}_0 + \rho\tilde{f}_1,$$

where $\tilde{f}_i$ is the grayscale texton associated to f_i, as defined in [8]

$$\forall\,\omega, \quad \widehat{\tilde{f}}_i(\omega) = |\hat{f}_i(\omega)|.$$

SN geodesics and synthesis. Theorem 6.2 details how to compute f_ρ, which parameterizes the the geodesic path as a spot noise $\mu_\rho = \mathcal{S}(f_\rho)$. One can thus perform a synthesis from the geodesic model by essentially replacing f_0 by f_ρ in the framework described in Section 3. This is detailed in Algorithm 3.

Algorithm 3: SN Geodesic Path Synthesis

Input: exemplars $(\tilde{f}_0, \tilde{f}_1) \in \mathbb{R}^{U\times d}$, weight $\rho \in [0,1]$.

Output: sample $f \in \mathbb{R}^{\tilde{U}\times d}$ of the geodesic SN model.

1. **Pre-processing.** Compute (f_0, f_1) from $(\tilde{f}_0, \tilde{f}_1)$ using (3.2).
2. **SN mixing.** Compute f_ρ from (f_0, f_1) using (6.11).
3. **Texture synthesis.** Apply step 3, 4 and 5 of Algorithm 1 with f_ρ instead of f_0.

6.6. AR Model Geodesics. Recall that according to (4.3), a stationary AR(1) process follows, for each frequency $\omega \in U_s$, the AR(1) recursion in $\mathbb{C}^d$

$$\forall\, t \in \mathbb{Z}, \quad x_{t+1} = ax_t + \bar{w}_t \in \mathbb{C}^d.$$

where $a = \hat{A}(\omega) \in \mathbb{C}^{d\times d}$, $x_t = \hat{X}_t(\omega)$, $\bar{w}_t \sim \mathcal{N}(0, \hat{B}(\omega))$. Since the model is separable across frequencies, we will thus focus on computing the geodesic of a AR(1) model in small dimension $\mathbb{C}^d$ parameterized by (a, b).

While the set of SN models is geodesically convex for the OT distance, it is not the case for the set of AR(1) distributions. A brute force way to impose the geodesic to be an AR(1) model is to restrict the optimization (6.3) to be an AR(1) process. This leads to an intractable highly non-convex problem. We propose here to use a simple approximation that works well in our numerical experiments.

Low-dimensional covariance embedding. The basic idea is that the set of AR(1) model $(x_t)_t$ on $\mathbb{C}^d \times \mathbb{Z}$ is in bijection with a subset of Gaussian processes on $\mathbb{C}^{2d}$, and an explicit bijection is given by the selection of two consecutive frames $(x_t)_{t\in\mathbb{Z}} \mapsto (x_0, x_1)$, where we have arbitrarily chosen the frames indexed by $t = 0$ and $t = 1$. Note that this is formally a map between random vectors. The following proposition shows how to compute the covariance of these pair of frames and how to invert the corresponding embedding.

PROPOSITION 6.3. *The covariance $\Gamma(a, b)$ of (x_0, x_1) defines the map*

$$\Gamma : (a, b) \in \Delta_d \times \mathcal{S}_d^+ \mapsto \begin{bmatrix} \beta(a,b) & a\beta(a,b) \\ \beta(a,b)a^* & \beta(a,b) \end{bmatrix} \in \mathcal{U}_d \subset \mathbb{C}^{2d\times 2d}$$

where Δ_d is defined in (4.5) and $\mathcal{S}_d^+$ is the set of symmetric positive definite matrices, and

$$\mathcal{U}_d = \left\{ \begin{bmatrix} \beta & c \\ c^* & \beta \end{bmatrix} \in \mathbb{C}^{2d\times 2d} \setminus \beta \in \mathcal{S}_d^+, c \in \mathbb{C}^{d\times d} \right\}.$$

We define

$$\Gamma^{-1} : \begin{bmatrix} \beta & c \\ c^* & \beta \end{bmatrix} \in \mathcal{U}_d \mapsto (a = c\beta^{-1}, b = \beta - a\beta a^* = \beta - c\beta^{-1}c^*).$$

This map satisfies $\Gamma^{-1} \circ \Gamma = \mathrm{Id}$.

Proof. The expression of the covariance $\Gamma(a, b)$ is a direct consequence of Proposition 4.1. For $b \in \mathcal{S}_d^+$, $\beta = a\beta a^* + b$ is the sum of a semi-definite positive matrix and a positive definite matrix, and it is hence positive definite. One thus has $\Gamma(a, b) \in \mathcal{U}_d$. The inversion formula for Γ^{-1} follows from this, using the fact that β is invertible. □

Note that we restrict our attention to noise covariances $b \in \mathcal{S}_d^+$ that are full rank. It is certainly possible to weaken this assumption and still be able to define an inverse map Γ^{-1}.

Approximate AR geodesic through embedding. For any covariances $(\mathcal{C}_i)_{i=0,1}$, let us denote

$$\mathcal{C}_\rho = \mathrm{Bar}_\rho(\mathcal{C}_0, \mathcal{C}_1)$$

where $\mathcal{C}_\rho$ is the covariance of the geodesic of $(\mu_i = \mathcal{N}(0, \mathcal{C}_i))_{i=0,1}$ with weight ρ, that minimizes (6.3) and that can be computed in closed form as exposed in (6.4).

We compute an approximate geodesic of AR models parameterized by $(a_i, b_i)_{i=0,1}$ by computing the barycenter over the embedding domain after the application of Γ, and then lifting back to the set of AR models using Γ^{-1}. Formally, this reads

$$\forall\, \rho \in (0, 1), \quad (a_\rho, b_\rho) = \Gamma^{-1}\left(\mathrm{Bar}_\rho(\Gamma(a_0, b_0), \Gamma(a_1, b_1))\right). \tag{6.15}$$

The property $\Gamma^{-1} \circ \Gamma = \mathrm{Id}$ guarantees that $\rho \in (0, 1) \mapsto (a_\rho, b_\rho)$ interpolates between (a_i, b_i) for $i = 0, 1$.

Denoting

$$\forall\, i \in \{0, 1\}, \quad \Gamma(a_i, b_i) = \begin{bmatrix} \beta_i & c_i \\ c_i^* & \beta_i \end{bmatrix} \in \mathcal{U}_d,$$

one easily sees that the barycenter is in $\mathcal{U}_d$,

$$\forall\, \rho \in [0, 1], \quad \mathrm{Bar}_\rho(\Gamma(a_0, b_0), \Gamma(a_1, b_1)) = \begin{bmatrix} \beta_\rho & c_\rho \\ c_\rho^* & \beta_\rho \end{bmatrix}.$$

AR barycenters and synthesis. Algorithm 4 describes the steps of the texture mixing and synthesis with the AR model. It operates by applying Equation (6.15) to each spacial frequency, and then using the synthesis framework described in Section 4.

Algorithm 4: AR Geodesic Path Synthesis

Input: exemplars $(\tilde{f}_0, \tilde{f}_1) \in \mathbb{R}^{U\times d}$, weight $\rho \in [0,1]$.

Output: sample $f \in \mathbb{R}^{\tilde{U}\times d}$ of the geodesic AR model.

1. **Pre-processing.** Compute (f_0, f_1) from $(\tilde{f}_0, \tilde{f}_1)$ using (3.2).
2. **AR textons learning.** For $i = 0, 1$, learn $(\mathcal{A}_i, \mathcal{B}_i)$ using (4.8) and (4.9).
3. **AR mixing.** Learn the mixed parameter $(\mathcal{A}_\rho, \mathcal{B}_\rho)$ by applying (6.15) to each $\omega \in U_s$, with $(a_\rho, b_\rho) = (\hat{A}_\rho(\omega), \hat{B}_\rho(\omega))$.
4. **Texture synthesis.** Apply step 4 and 5 of Algorithm 2 with $(\mathcal{A}_\rho, \mathcal{B}_\rho)$ instead of $(\mathcal{A}, \mathcal{B})$.

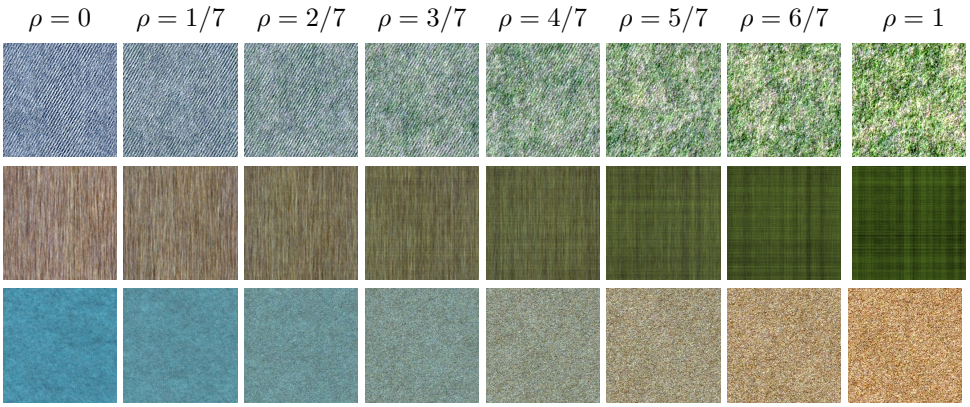
6.7. Numerical Results for Geodesics. In this section, we show some results obtained with the OT-geodesic mixing method. Note that we experiment both on static and dynamic textures.

Mixing static SN models. Each row of Figure 6.2 corresponds to a single experiment. Given two input textures $(\tilde{f}_0, \tilde{f}_1)$, the texture models $\mu(f_i)_{i=0,1}$ are learned for both, and then several models $\mu(f_\rho)$ where interpolated along the path between model $\mu(f_0)$ and $\mu(f_1)$. Detailed numerical procedures are given in Algorithm 3. In this experiment, we take $\rho \in \{\frac{k}{8}, k = 0, 1, 2, \dots, 8\}$. The resulting images are shown on the first row of Figure 6.2. Note that the images on the extreme left and right of each row correspond to $\rho = 0, 1$ respectively, which are instances of the original models. We would like to point out how these instances are perceptually similar to the original input textures.

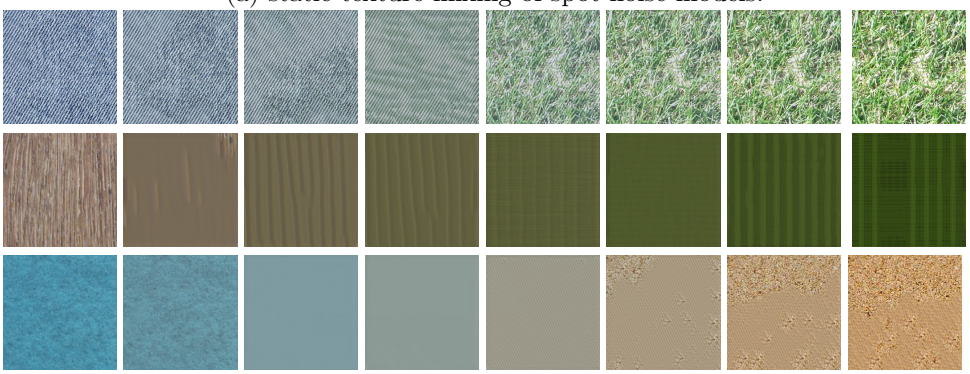
With respect to the interpolated models and their instances, there are two effects that we would like the reader to note. First of all, the color changes gradually as we move along the geodesic path, see Figure 6.2 for examples. Secondly, there is a continuous morphing between the spacial patterns of the exemplar due to a change in the covariance matrix. As we move along the geodesic path, the models mix in a different proportion the spatial patterns of the original textures. This is specially visible in the first rows, where diagonal features progressively replaces the isotropic structures of the grass.

We compare our results with those achieved by a state-of-the-art method, named *Image Melding*, proposed in [36], which mixes two textures by a patch-based approach. More details about the implementations can be found in [36], and we use the same ρ values as mentioned before. Figure 6.2(b) displays the texture mixing results by Image Melding. Comparing with the results in Figure 6.2(a), we can see that the interpolation of either color or spatial patterns are worse than those obtained by the proposed OT-geodesic method.

An example of dynamic texture mixing can be observed in Figure 6.3. Each row corresponds to a single video, where every image is a single frame, ordered from left to right. The first and last rows are the input videos and the two middle ones are instances of interpolated models obtained with the geodesic mix method. In this experiment, we take $\rho = \{\frac{k}{7}, k = 0, 1, \dots, 7\}$. Note how the colors, spatial patterns, and movements are interpolated.



(a) static texture mixing of spot noise models.



(b) static texture mixing via Image Melding.

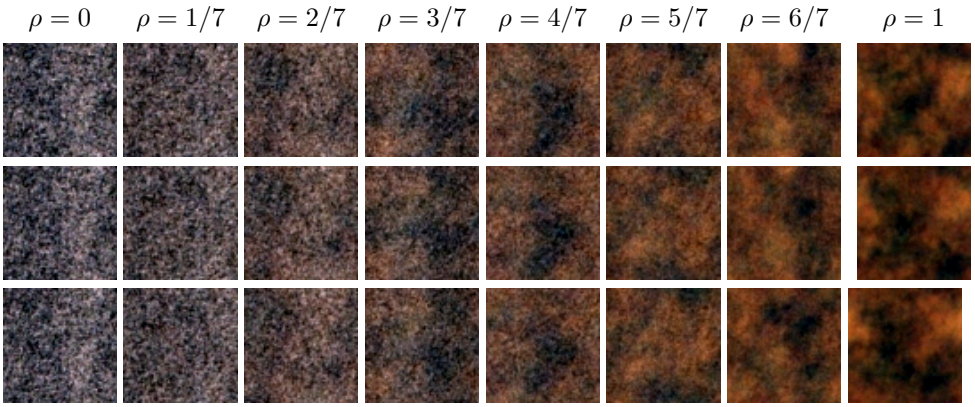
Figure 6.2: Results obtained with OT texture mixing. In (a), each texture was synthesized from the SN models along the OT-geodesic. Comparison results of texture mixing using Image Melding [36] are displayed in (b).

Mixing dynamic AR and SN models. Figure 6.3 shows a similar experiment for dynamic textures. (a) shows results obtained using the SN models. (b) shows results obtained with AR models. Given two input textures $(\tilde{f}_0, \tilde{f}_1)$, both AR texture models $(a_i, b_i)_{i=0,1}$ are learned, and then several (a_ρ, b_ρ) are interpolated along the path between these two models. Detailed numerical procedures are given in Algorithm 4. One can observe that both methods produces visually similar results.

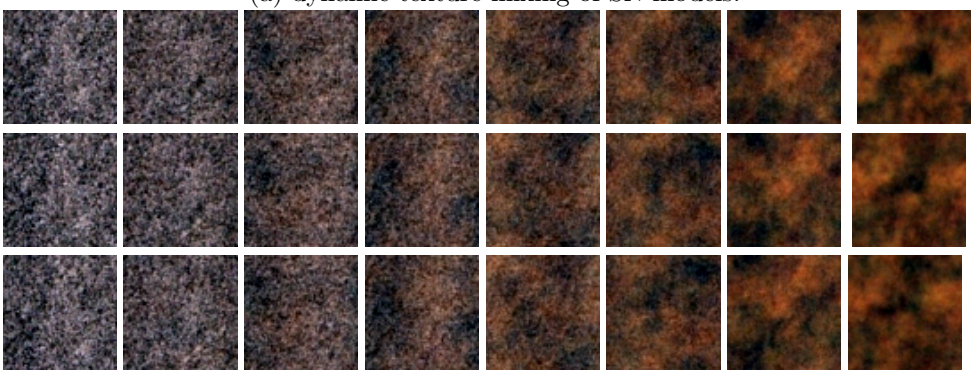
7. Barycenter Mixing of Several Gaussian Models. This section extends the construction of Section 6 to the case of mixing an arbitrary number of Gaussian models. The computation is however more involved since no closed form is known for the OT barycenter.

7.1. Optimal Transport Barycenter of Gaussian Distributions. Given a family of Gaussian distributions $(\mu_i)_{i\in I}$ and weights $\rho = (\rho_i)_{i\in I}$ with $\rho_i \geq 0, \sum_i \rho_i = 1$, the barycenter according to some distance d is defined as

$$\mu_\rho = \operatorname*{argmin}_{\mu} \sum_{i\in I} \rho_i d(\mu_i, \mu)^2. \tag{7.1}$$



(a) dynamic texture mixing of SN models.



(b) dynamic texture mixing of AR models.

Figure 6.3: Example of dynamic texture mixing of SN and AR models. From left to right, the weight ρ takes values in $\{\frac{k}{7}, k = 0, 1, \ldots, 7\}$. From top to bottom, 3 frames of each synthesized dynamic texture are displayed.

Note that when $|I| = 2$, one recovers the definition (6.3). Note that $\rho \in (\mathbb{R}^+)^I$ is now a vector of barycentric coordinates (which differs from Section 6 where ρ was a single scalar). We now recall a result proved in [1].

PROPOSITION 7.1. *If at least one of the $\mathcal{C}_i$ has full rank, then the OT barycenter μ_ρ of $(\mu_i = \mathcal{N}(\mathfrak{m}_i, \mathcal{C}_i))_{i \in I}$ exists, is unique, and is a Gaussian process $\mathcal{N}(\mathfrak{m}_\rho, \mathcal{C}_\rho)$ where*

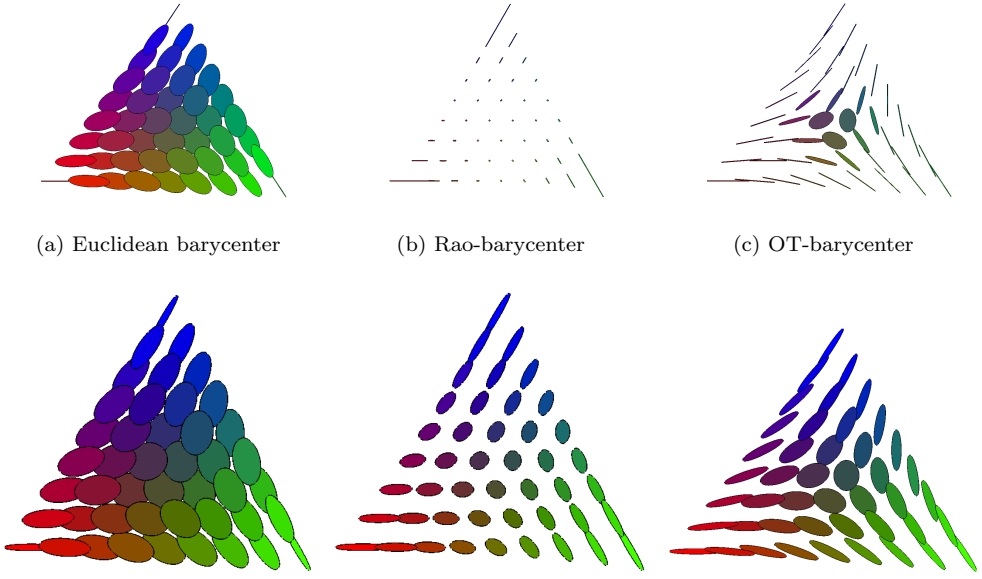
$$\mathfrak{m}_\rho = \sum_{i \in I} \rho_i \mathfrak{m}_i$$

and $\mathcal{C}_\rho$ is the unique solution of the following fixed point equation

$$\Phi(\mathcal{C}_\rho) = \mathcal{C}_\rho \quad \textit{where} \quad \Phi(\mathcal{C}) = \sum_{i \in I} \rho_i \left(\mathcal{C}^{1/2} \mathcal{C}_i \mathcal{C}^{1/2} \right)^{1/2}. \tag{7.2}$$

An open problem is to determine under which hypothesis the barycenter is still unique when all the covariances $\mathcal{C}_i$ are rank-deficient, which is the case when averaging SN models. In the case $|I| = 2$, Proposition 6.1 ensures that this is the case under a restriction on the kernel of the covariances.

In Figure 7.1, we illustrate and compare the barycenters of three 2-D Gaussians (displayed at the vertices of a triangle) for the Euclidean-distance, Rao-distance, and OT-distance respectively. Note that each edge of the triangle corresponds to a geodesic between two 2-D Gaussian distributions. A covariance $\mathcal{C}$ is represented using its associated unit ball (an ellipse) $\left\{x \in \mathbb{R}^2 \setminus x^* \mathcal{C} x \leq 1\right\}$. The Euclidean barycenter does not maintain the rank-1 property, and ellipses obtained as barycenters are more "round". The Rao barycenter is not defined for rank-1 covariances, and tends to become ill-posed as the covariance becomes rank deficient. In contrast, the OT barycenter maintains the rank-1 property along the edges of the triangle, and become full rank only near the center of the triangle.



(a) Euclidean barycenter (b) Rao-barycenter (c) OT-barycenter

(d) Euclidean barycenter (e) Rao-barycenter (f) OT-barycenter

Figure 7.1: Comparisons of different barycenters of three 2-dimensional Gaussian distributions with the same mean vector but different covariance matrix, $\mu_i = \mathcal{N}(\mathfrak{m}, \mathcal{C}_i), i = 0, 1, 2$. (a, b, c) Barycenters of three Gaussian distributions with almost rank-1 covariance matrices. (d, e, f) Barycenters of three full rank covariances.

7.2. Barycenter of SN Models.

Fixed point equation. The following proposition, which is a direct consequence of Proposition 7.1, describes the barycenter of SN models using a series of fixed point equation over the Fourier domain.

PROPOSITION 7.2. *If $(\mu_i = \mathcal{N}(\mathfrak{m}_i, \mathcal{C}_i))_{i \in I}$ are stationary Gaussian processes, a barycenter satisfies*

$$\forall \omega \in U, \quad \hat{C}_\rho(\omega) = \Phi_\omega(\hat{C}_\rho(\omega)) \tag{7.3}$$

where

$$\Phi_\omega : c \in \mathbb{C}^{d \times d} \mapsto \sum_{i \in I} \rho_i \left(c^{1/2} \hat{C}_i(\omega) c^{1/2} \right)^{1/2} \in \mathbb{C}^{d \times d}.$$

We note that in general, μ_ρ is not a spot noise because $\hat{C}_\rho(\omega)$ is not necessarily rank one.

Numerical scheme. In general, put aside the case where the covariances $\hat{C}_i(\omega)$ diagonalize in the same bases, or the case $|I| = 2$ (see Theorem 6.2), it is not possible to solve the fixed point equation (7.3) in closed form.

Following [22], we propose to approximate $\hat{C}_\rho(\omega)$ by iterating the mapping Φ_ω. Formally, for each $\omega \in U$, we initialize $\hat{C}_\rho^{(0)}(\omega)$ (for instance using $\hat{C}_i(\omega)$ for some $i \in I$) and then iterate

$$\forall\, \ell \geq 0, \quad \hat{C}_\rho^{(\ell+1)}(\omega) = \Phi_\omega(\hat{C}_\rho^{(\ell)}(\omega)). \tag{7.4}$$

We observe numerically the convergence

$$\hat{C}_\rho^{(\ell)}(\omega) \overset{\ell \to +\infty}{\longrightarrow} \hat{C}_\rho(\omega)$$

where $\hat{C}_\rho(\omega)$ is the Fourier transform of the kernel of a barycenter covariance $\mathcal{C}_\rho$.

Note that the mapping Φ_ω is not strictly contracting, but we conjecture that some iterate $\Phi_\omega \circ \ldots \circ \Phi_\omega$ of this mapping is contracting (as it is the case in dimension $d = 1$), although we were not able to prove it in arbitrary dimension $d > 1$.

The numerical computation of Φ_ω in the case $d = 3$ requires the computation of the square root of 3×3 matrices, which is performed explicitly by computing the eigenvalues of the symmetric matrix as the root of a third order polynomial. Algorithm 5 details the method.

Algorithm 5: SN Barycenter Synthesis

Input: exemplars $(\tilde{f}_i \in \mathbb{R}^{U \times d})_{i \in I}$, weight $(\rho_i)_{i \in I} \in (\mathbb{R}^+)^I$ with $\sum_i \rho_i = 1$.

Output: sample $f \in \mathbb{R}^{\tilde{U} \times d}$ of the barycentric SN model.

1. **Pre-processing.** Compute $(f_i)_{i \in I}$ from $(\tilde{f}_i)_{i \in I}$ using (3.2).
2. **SN mixing.** Compute $\mathcal{C}_\rho$ from $(\mathcal{C}_i)_{i \in I}$ using the iterations (7.4).
3. **Texture synthesis.** Apply step 3, 4 and 5 of Algorithm 1 with $\mathcal{C}_\rho$ instead of $\mathcal{C}$.

7.3. Barycenter of AR Models. For any covariance $(\mathcal{C}_i)_{i \in I}$, let us denote

$$\mathcal{C}_\rho = \mathrm{Bar}_\rho(\mathcal{C}_i)_{i \in I}$$

where $\mathcal{C}_\rho$ is the covariance of the barycenter $\mu_\rho = \mathcal{N}(0, \mathcal{C}_\rho)$ of $(\mu_i = \mathcal{N}(0, \mathcal{C}_i))_{i \in I}$ with weight ρ, that minimizes (7.1) and can be computed using the iterations of the mapping Φ defined in (7.3).

We compute an approximate OT barycenter of the AR models using the low-dimensional mapping exposed in Section 6.6. This corresponds to computing the AR textons $(\mathcal{A}_\rho, \mathcal{B}_\rho)$ of the barycenter as

$$\forall\, \omega \in U_s,\ \forall\, \rho \in [0,1], \quad (\hat{A}_\rho(\omega), \hat{B}_\rho(\omega)) = \Gamma^{-1}\left(\mathrm{Bar}_\rho(\Gamma(\hat{A}_i(\omega), \hat{B}_i(\omega)))_{i \in I}\right). \tag{7.5}$$

Algorithm 6 details the whole process.

Algorithm 6: AR Barycenter Synthesis

Input: exemplars $(\tilde{f}_i \in \mathbb{R}^{U \times d})_{i \in I}$, weight $(\rho_i)_{i \in I} \in (\mathbb{R}^+)^I$ with $\sum_i \rho_i = 1$.

Output: sample $f \in \mathbb{R}^{\tilde{U} \times d}$ of the barycentric AR model.

1. **Pre-processing.** Compute $(f_i)_{i \in I}$ from $(\tilde{f}_i)_{i \in I}$ using (3.2).
2. **AR texton learning.** For $i \in I$, learn $(\mathcal{A}_i, \mathcal{B}_i)$ using (4.8) and (4.9).
3. **AR mixing.** Compute the mixed parameter $(\mathcal{A}_\rho, \mathcal{B}_\rho)$ by applying (7.5) for each $\omega \in U_s$.
4. **Texture synthesis.** Apply step 4 and 5 of Algorithm 2 with $(\mathcal{A}_\rho, \mathcal{B}_\rho)$ instead of $(\mathcal{A}, \mathcal{B})$.

7.4. Numerical Results for Barycenter Mixing. This section shows some numerical results obtained with the barycenter mixing method with three input textures. Given the input textures (f_0, f_1, f_2), we first learn their Gaussian models (SN or AR), and then compute the weighted barycenters with Algorithm 5 (for SN) and Algorithm 6 (for AR). Figure 7.2 displays the barycentric coordinates $\rho = (\rho_0, \rho_1, \rho_2) \in \mathbb{R}^3$ on a triangle, whose vertices correspond to $\rho = (1, 0, 0)$, $\rho = (0, 1, 0)$, and $\rho = (0, 0, 1)$, respectively. We illustrate the method by navigating along two straight paths $s \mapsto \rho(s)$ that are displayed on the same figure.

Figure 7.3 displays the results obtained on static textures for the AR barycenters, along the two paths. Note how the synthesized textures obtained from the models that are not located at the vertices have mixed colors and texture patterns.

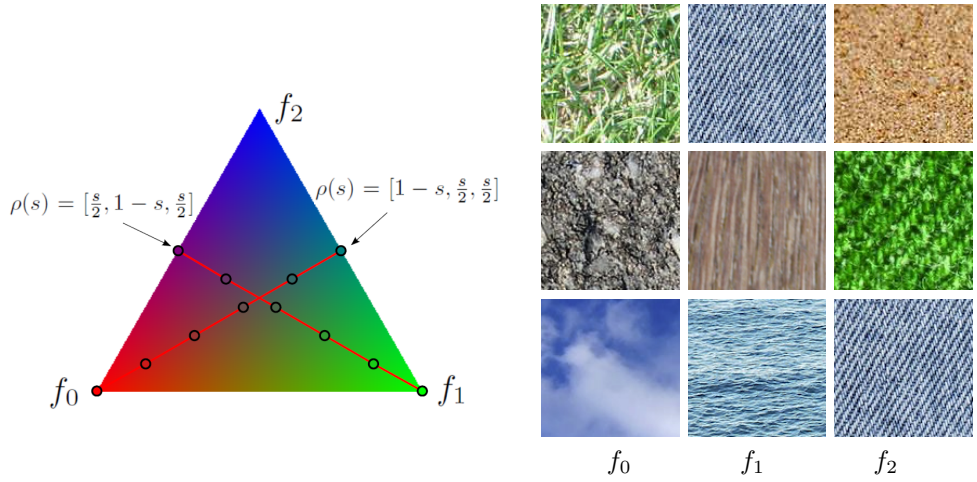
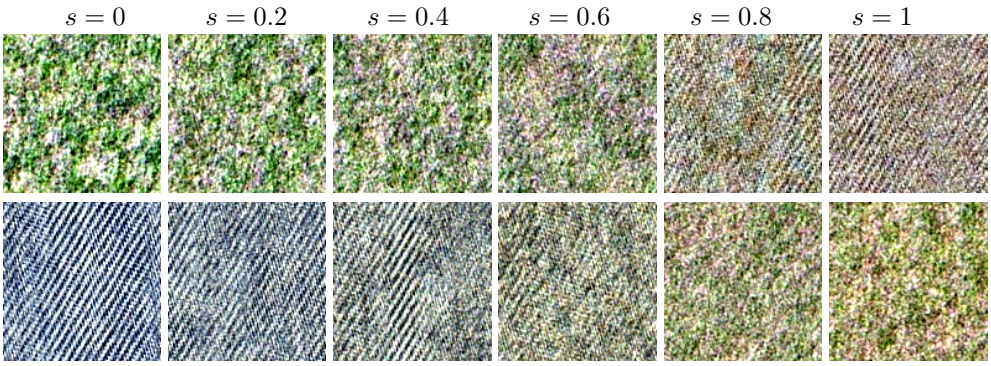


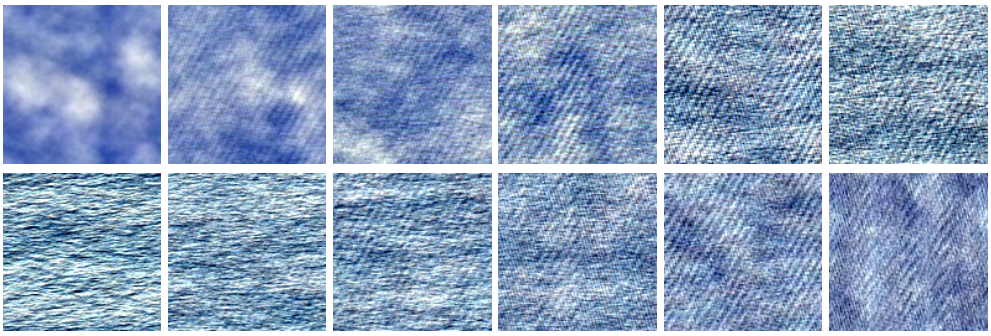
Figure 7.2: Left: (ρ_0, ρ_1, ρ_2) barycentric coordinates together with the two paths $s \mapsto \rho(s)$ used in our experiments. Right: input textures (f_0, f_1, f_2) used for the experiments showed on Figure 7.3. Each column corresponds to a single experiment, where (f_0, f_1, f_2) are the exemplar textures located at the vertices of the triangle.

Figure 7.4 displays the results obtained on dynamic textures mixing with SN barycenters. Each column corresponds to a single video, where every image is a single frame, ordered from top to bottom. Result obtained with SN barycenter are visually similar.



(a) Barycenter texture mixing of the three texture models in the 1^{st} set.

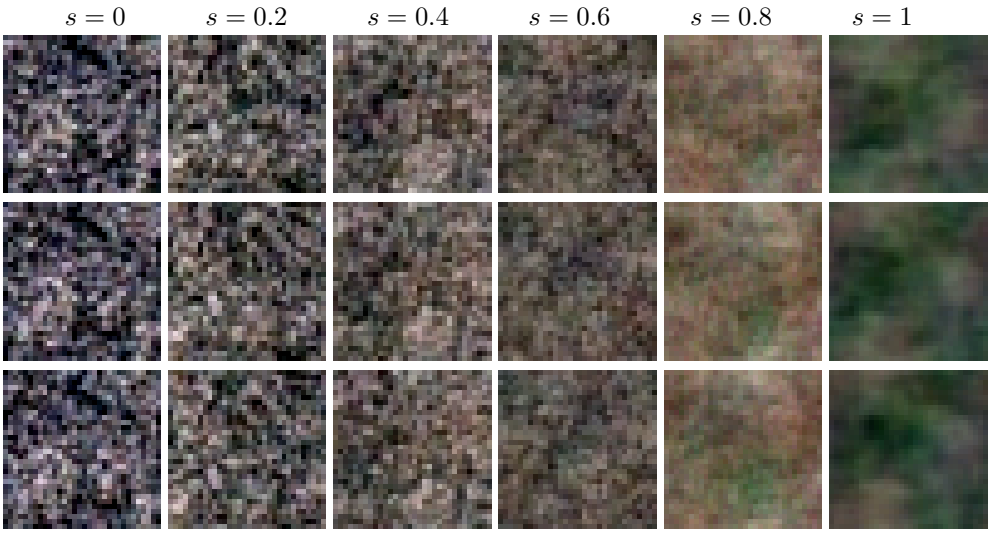
(b) Barycenter texture mixing of the three texture models in the 2^{nd} set.



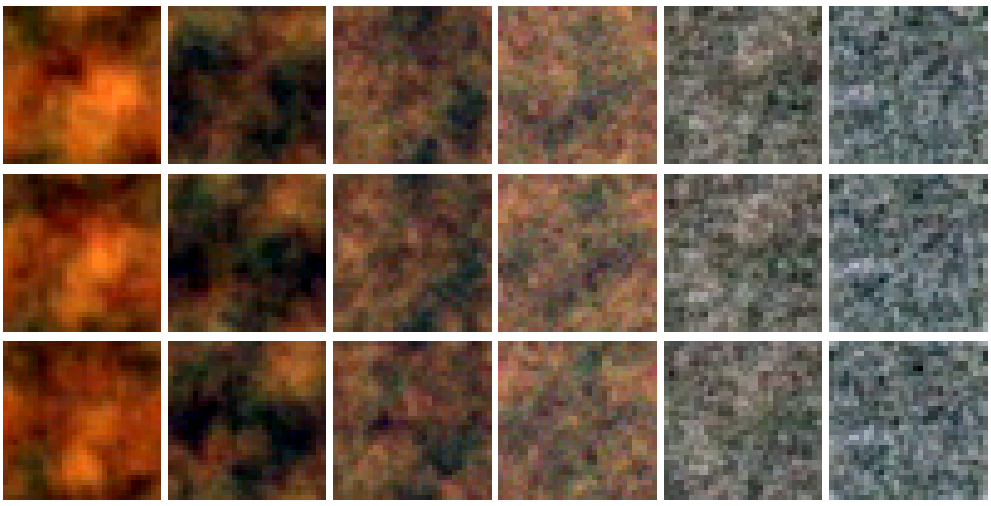
(c) Barycenter texture mixing of the three texture models in the 3^{rd} set.

Figure 7.3: SN barycenter texture mixing of three texture models, using different input exemplar sets (f_0, f_1, f_2) shown in Figure 7.2, right. In each subfigure, the top row displays the mixing along the path $\rho(s) = (1 - s, s/2, s/2)$, and the bottom row displays the mixing on the path $\rho(s) = (s/2, 1 - s, s/2)$, where $s \in [0, 1]$ parameterizes the path.

8. Conclusion. This paper has introduced two compact representations of stationary static and dynamic textures. These representations correspond to two different parameterizations of Gaussian processes. Experimental results demonstrate that the proposed methods are quite effective at describing sequences which exhibit temporal and spatial regularity. Moreover, the computational complexity of the proposed



(a) mixing dynamic textures on the path $\rho(s) = (1 - s, s/2, s/2)$.



(b) mixing dynamic textures on the path $\rho = (s/2, 1 - s, s/2)$.

Figure 7.4: Barycenter dynamic texture mixing of three input texture videos, using AR models. The top three rows show 3 consecutive frames of the mixing on the path $\rho(s) = (1 - s, s/2, s/2)$, and the bottom three rows display those on the path $\rho(s) = (s/2, 1 - s, s/2)$, where $s \in [0, 1]$ parameterizes the path.

algorithms for texture synthesis is low. While both methods tend to produce visually similar results on our numerical examples, their parameterization, and thus their typical usage, are different. Only the SN model can be used for static textures. The AR model is probably the most appropriate for dynamic textures with simple temporal patterns, since it offers the most compact representation with only 2-D textons. On contrast, the SN model requires the computation of a full 3-D texton, but is able to capture arbitrary Gaussian models.

The second main contribution of this paper is a new method for texture mixing that enables the creation of new textures from a set of exemplars. It is based on optimal transport, which provides a mathematically sound way to interpolate between distributions. A major feature of this method is that it is robust to rank-deficient covariances. This is crucial to deal with rank-deficient models that often occurs when learning from a a small number of exemplars. The numerical results show how the method successfully merges the visual features of the original images into new complex patterns.

These contributions open the door to several potential future works beside texture synthesis and mixing. The computation of compact texture representations could be used for dynamic textures recognition and video compression. The ability to perform models averaging is a key ingredient for un-supervised learning algorithms (e.g. for clustering tasks), such as for instance in the celebrated k-means algorithm.

REFERENCES

[1] M. Agueh and G. Carlier, *Barycenters in the wasserstein space*, SIAM J. on Mathematical Analysis, 43 (2011), pp. 904–924.

[2] S. Amari, *Differential-geometrical methods in statistics*, Lecture notes in statistics, Springer-Verlag, 1985.

[3] C. Atkinson and A.F.S. Mitchell, *Rao's distance measure*, Sankhya, The Indian J. of Stat., Ser. A, 43 (1981), pp. 345–365.

[4] Z. Bar-Joseph, R. El-Yaniv, D. Lischinski, and M. Werman, *Texture mixing and texture movie synthesis using statistical learning*, IEEE Trans. Vis. Comput. Graphics, 7 (2001), pp. 120–135.

[5] ———, *Texture mixing and texture movie synthesis using statistical learning*, IEEE Tr. on Vis. and Comp. Graph., 7 (2001), pp. 120–135.

[6] A. B. Chan and N. Vasconcelos, *Mixtures of dynamic textures*, in ICCV, 2005, pp. I: 641–647.

[7] A. Desolneux, L. Moisan, and S. Ronsin, *Vers un texton pour les micro-textures*, in GRETSI, 2011.

[8] A. Desolneux, L. Moisan, and Samuel Ronsin, *A compact representation of random phase and gaussian textures*, in Proc. the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), 2012, pp. 1381–1384.

[9] A. Desolneux, L. Moisan, and S. Ronsin, *A texton for random phase and gaussian textures*, in preparation, (2013).

[10] G. Doretto, A. Chiuso, Y. Wu, and S. Soatto, *Dynamic textures*, International Journal of Computer Vision, 51 (2003), pp. 91–109.

[11] G. Doretto, A. Chiuso, Y. N. Wu, and S. Soatto, *Dynamic textures*, Int. J. Comput. Vision, 51 (2003), pp. 91–109.

[12] G. Doretto, E. Jones, and S. Soatto, *Spatially homogeneous dynamic textures*, in Proc. European Conf. Computer Vision, 2004, pp. 591–602.

[13] D. C. Dowson and B. V. Landau, *The fréchet distance between multivariate normal distributions*, J. Multivariate Anal., 3 (1982), pp. 450–455.

[14] A. Efros and T. Leung, *Texture synthesis by non-parametric sampling*, in Proc. Int. Conf. Computer Vision, 1999, pp. 1033–1038.

[15] A. A. Efros and T. K. Leung, *Texture synthesis by non-parametric sampling*, in Proc. of ICCV '99, 1999, p. 1033.

[16] B. Galerne, Y. Gousseau, and J-M. Morel, *Random phase textures: Theory and synthesis*, IEEE Trans. on Image Processing, 20 (2011), pp. 257–267.

[17] ———, *Random phase textures: Theory and synthesis*, IEEE Trans. Image Processing, 20 (2011), pp. 257–267.

[18] J. E. Gentle, *Numerical Linear Algebra for Applications in Statistics*, Berlin: Springer-Verlag, 1998, ch. Cholesky Factorization, pp. 93–95.

[19] B. Ghanem and N. Ahuja, *Phase based modelling of dynamic textures*, in Proc. Int. Conf. Computer Vision, 2007, pp. 1–8.

[20] B. Julesz, *Visual pattern discrimination*, IEEE Trans. Information Theory, 8 (1962), pp. 84–92.

[21] B. Julesz, E. N. Gilbert, L. A. Shepp, and H. L. Frisch, *Inability of humans to discrimi-*

nate between visual textures that agree in second-order statistics – revisited, Perception, 2 (1973), pp. 391–405.

[22] M. Knott and C. S. Smith, *On a generalization of cyclic monotonicity and distances among random vectors*, Linear Algebra and its Applications, 199 (1994), pp. 363–371.

[23] V. Kwatra, A. Schödl, I. Essa, G. Turk, and A. Bobick, *Graphcut textures: image and video synthesis using graph cuts*, in Proc. ACM SIGGRAPH, 2003, pp. 277–286.

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

[25] W. Matusik, M. Zwicker, and F. Durand, *Texture design using a simplicial complex of morphable textures*, ACM Transactions on Graphics, 24 (2005), pp. 787–794.

[26] L. Moisan, *Periodic plus smooth image decomposition*, J. Math. Imag. Vis., 39 (2011), pp. 161–179.

[27] S. M. Pandit and S-M. Wu, *Time Series and System Analysis with Applications*, John Wiley & Sons, 1983.

[28] R. Péteri, S. Fazekas, and M. J. Huiskes, *DynTex: a comprehensive database of dynamic textures*, Pattern Recognition Letters, 31 (2010), pp. 1627–1632.

[29] F. Pitié, A. Kokaram, and R. Dahyot, *Automated colour grading using colour distribution transfer*, Computer Vision and Image Understanding, (2007).

[30] J. Portilla and E. P. Simoncelli, *A parametric texture model based on joint statistics of complex wavelet coefficients*, Int. J. Comput. Vision, 40 (2000), pp. 49–70.

[31] J. Rabin, G. Peyré, J. Delon, and M. Bernot, *Wasserstein barycenter and its application to texture mixing*, Proc. SSVM'11, (2011).

[32] N. Ravishanker, *Differential geometry of arfima processes*, Communications in Statistics, Theory and Methods, 30 (2001), pp. 1889–1902.

[33] N. Ravishanker, E.L. Melnick, and C-L. Tsai, *Differential geometry of arma models*, Journal of Time Series Analysis, 11 (1990), pp. 259–274.

[34] Y. Rubner, C. Tomasi, and L. J. Guibas, *The earth mover's distance as a metric for image retrieval*, International Journal of Computer Vision, 40 (2000), pp. 99–121.

[35] R. Ruiters, R. Schnabel, and R. Klein, *Patch-based texture interpolation*, Computer Graphics Forum (Proc. of EGSR), 29 (2010), pp. 1421–1429.

[36] D. Soheil, E. Shechtman, C. Barnes, D. B Goldman, and P. Sen, *Image Melding: Combining Inconsistent Images using Patch-based Synthesis*, ACM Transactions on Graphics (TOG) (Proceedings of SIGGRAPH 2012), 31 (2012).

[37] M. Szummer and R. W. Picard, *Temporal texture modeling*, in Proc. Int. Conf. Image Processing, vol. 3, Sep. 1996, pp. 823–826.

[38] A. Takatsu, *Wasserstein geometry of gaussian measures*, Osaka J. Math., (2011).

[39] K. J. van Garderen, *Exact geometry of first order autoregressive models*, Journal of Econometrics, 95 (2000), pp. 285–331.

[40] J. J. van Wijk, *Spot noise texture synthesis for data visualization*, SIGGRAPH Comput. Graph., 25 (1991), pp. 309–318.

[41] C. Villani, *Topics in Optimal Transportation*, American Mathematical Society, 2003.

[42] L-Y. Wei and M. Levoy, *Texture synthesis over arbitrary manifold surfaces*, in Proc. ACM SIGGRAPH, 2001, pp. 355–360.