Gabor Noise by Example

Bruno Galerne, Ares Lagae, Sylvain Lefebvre, George Drettakis

Gabor Noise by Example

Bruno Galerne*
MAP5, Université Paris Descartes and CNRS, Sorbonne Paris Cité

Ares Lagae*
KU Leuven

Sylvain Lefebvre
ALICE/INRIA Nancy Grand-Est

George Drettakis
REVES/INRIA Sophia-Antipolis

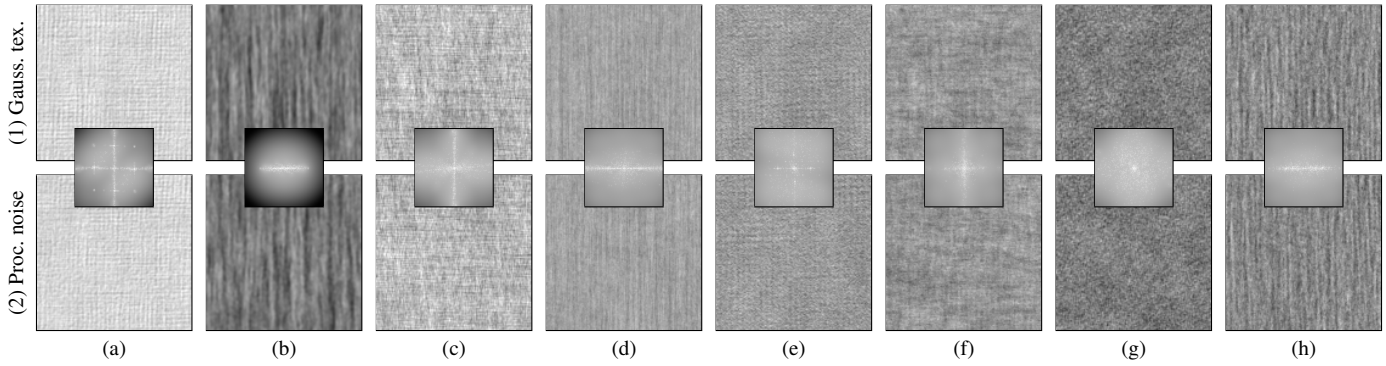

Figure 1: *We present* Gabor noise by example*, a method to estimate the parameters of bandwidth-quantized Gabor noise, a procedural noise function that can generate noise with an arbitrary power spectrum, from exemplar Gaussian textures, a class of textures that is completely characterized by their power spectrum. (row 1) Gaussian texture. (row 2) Procedural noise. (insets) Estimated power spectrum.*

Abstract

Procedural noise is a fundamental tool in Computer Graphics. However, designing noise patterns is hard. In this paper, we present *Gabor noise by example*, a method to estimate the parameters of bandwidth-quantized Gabor noise, a procedural noise function that can generate noise with an arbitrary power spectrum, from exemplar Gaussian textures, a class of textures that is completely characterized by their power spectrum. More specifically, we introduce (i) bandwidth-quantized Gabor noise, a generalization of Gabor noise to arbitrary power spectra that enables robust parameter estimation and efficient procedural evaluation; (ii) a robust parameter estimation technique for quantized-bandwidth Gabor noise, that automatically decomposes the noisy power spectrum estimate of an exemplar into a sparse sum of Gaussians using non-negative basis pursuit denoising; and (iii) an efficient procedural evaluation scheme for bandwidth-quantized Gabor noise, that uses multi-grid evaluation and importance sampling of the kernel parameters. Gabor noise by example preserves the traditional advantages of procedural noise, including a compact representation and a fast on-the-fly evaluation, and is mathematically well-founded.

Keywords: procedural noise, Gaussian texture, Gaussian random field, power spectrum estimation, non-negative basis pursuit denoising, decorrelated color space

Links: DL PDF WEB

*Bruno Galerne and Ares Lagae are joint first authors
e-mail: bruno.galerne@parisdescartes.fr, ares.lagae@cs.kuleuven.be, sylvain.lefebvre@inria.fr, george.drettakis@inria.fr

1 Introduction

Since its introduction by Perlin [1985] almost three decades ago, procedural noise has become a fundamental tool in Computer Graphics. However, designing noise patterns is still hard. Implementing procedural shaders and tweaking their parameters in order to achieve a desired visual effect requires significant effort. We believe this difficulty stems, in part, from two reasons: (i) the expressiveness of current procedural noise functions is rather limited in practice (see Sec. 2.1), and (ii) selecting the appropriate noise parameters to obtain a desired noise pattern or texture is difficult, even more so when the noise function has more parameters. The evident way to address this problem is estimating the parameters of a procedural noise function from an exemplar, i.e., *noise by example*. However, it is not immediately clear which textures are appropriate exemplars for noise by example, since noise obviously cannot reproduce any texture. We therefore introduce a new texture class, called *Gaussian textures*, that is defined, similarly to noise [Lagae et al. 2010a], as what is known in statistics as a *Gaussian random field* [Papoulis and Pillai 2002]. Both Gaussian textures and noise are completely characterized by their power spectrum. Ideally, a procedural noise function should thus be able to reproduce any Gaussian texture. However, current procedural noise functions have not demonstrated a diversity in noise patterns as rich as the Gaussian textures shown in Fig. 1.

In this paper, we present *Gabor noise by example*, a generalization of Gabor noise [Lagae et al. 2009] that (i) can generate a wide variety of Gaussian textures, and (ii) can estimate the noise parameters from a Gaussian exemplar. Our approach can reproduce the rich diversity of Gaussian textures shown in Fig. 1, is mathematically well-founded, and preserves the traditional advantages of procedural noise, including a compact representation and a fast on-the-fly evaluation. More specifically, the contributions of our work are:

- bandwidth-quantized Gabor noise, a generalization of Gabor noise to arbitrary power spectra that enables robust parameter estimation and efficient procedural evaluation (Sec. 4);
- a robust parameter estimation technique for bandwidth-quantized Gabor noise, that automatically decomposes the noisy power spectrum estimate of an exemplar into a sparse

sum of Gaussians using non-negative basis pursuit denoising [Kim et al. 2007] (Sec. 5); and

- an efficient procedural evaluation scheme for bandwidth-quantized Gabor noise, that uses multi-grid evaluation and importance sampling of the kernel parameters (Sec. 6).

Additionally, we present a maximally independent color space, a new color space for independent channel synthesis that makes the color channels of a texture maximally independent using approximate joint diagonalization [Hyvärinen et al. 2001, 11.3] (Sec. 7). We also demonstrate an interactive editor for bandwidth-quantized Gabor noise, inspired by the Wold decomposition [Francos et al. 1993] (Sec. 8).

2 Related Work

2.1 Procedural Noise Functions

Several procedural noise functions have been proposed since the seminal work of Perlin [1985]. We refer to the survey of Lagae et al. [2010a] for a recent overview. However, the expressiveness of these noise functions is rather limited in practice. Perlin noise [Perlin 1985], sparse convolution noise as proposed by Lewis [1989], and wavelet noise [Cook and DeRose 2005] are limited to isotropic power spectra. Anisotropic noise [Goldberg et al. 2008] is limited to power spectra that correspond to a coarse tiling of the frequency domain into oriented subbands. Although more complex power spectra could be obtained by using a much finer tiling, this would probably be impractical, since the method requires precomputation and storage of a noise texture for each oriented subband. Gabor noise [Lagae et al. 2009; Lagae and Drettakis 2011] has only been demonstrated on power spectra that consist of a small number of Gaussians, where the parameters of the noise corresponding to each Gaussian are set manually. We extend Gabor noise to arbitrary power spectra, where the parameters are estimated automatically from exemplars.

2.2 Example-Based Texture Synthesis

Many non-procedural example-based texture synthesis methods have been proposed. We refer to the survey of Wei et al. [2009] for a recent overview. However, these methods do not produce procedural textures. Nevertheless, for the class of Gaussian textures, our method is competitive with methods such as pyramid-based texture analysis/synthesis [Heeger and Bergen 1995] and parallel controllable texture synthesis [Lefebvre and Hoppe 2005] (see Sec. 9.2).

2.3 Procedural Texture Synthesis by Example

Ghazanfarpour and Dischler [1996] presented a method for the automatic generation of 3D textures using spectral analysis of two or three 2D slices of a 3D texture, based on an earlier method [Ghazanfarpour and Dischler 1995] that only takes into account a single 2D texture. They generate a discrete 3D model from the 2D slices, select a low number of high-magnitude peaks in the Fourier transform of the model, and approximate it by a cosine-like summation. However, as stated by the authors, their method is not well adapted to high-bandwidth or noisy textures such as sand, and requires the user to manually set several parameters.

Dischler and Ghazanfarpour [1997] also presented a method for automatically generating a procedural description of a geometric texture from a 1D exemplar profile. They iteratively apply spectral and histogram analysis. Their method is also applicable to isotropic color textures. However, it cannot handle anisotropic textures.

Lagae et al. [2010b] presented a method for procedural isotropic stochastic textures by example. They use a multi-octave wavelet noise model where the weights are computed from an exemplar, independent channel synthesis in a PCA color space, and histogram matching. Xue et al. [2011] presented a similar method for simulating rough appearance for stone weathering in a photograph. However, both methods cannot handle anisotropic textures.

Gilet et al. [2010] presented a method to generate procedural descriptions of anisotropic noisy textures by example. They use a Gabor noise model with a small number of noises and several cosines, independent channel synthesis in a PCA color space, and histogram matching. They decompose the 2D spectral domain into ellipses centered at the origin, by discretizing the power spectrum into a low number of values and fitting one or two ellipses to each binary area resulting from discretization, and associate a Gabor noise with every ellipse. Gilet and Dischler [2010] later used this method in an image-based approach for stochastic volumetric and procedural details. However, their method requires significant manual intervention and trial and error for each exemplar (see Sec. 9.2), while our approach is automatic.

Jeschke et al. [2011] presented a diffusion curve coloring algorithm that includes texture details based on spatially varying Gabor noise. However, their method records only a single dominant frequency, which captures only a rough impression of the texture.

Yoon et al. [2004; 2008] presented a method to edit noise values to allow user constraints to be applied to a noise function. Their method is designed to edit individual noise values, while our approach is designed to edit the overall appearance of the noise.

3 Background and Motivation

Noise is defined as what is known in statistics as a Gaussian random field, which is completely characterized by its power spectrum [Lagae et al. 2010a; Papoulis and Pillai 2002]. The problem of procedural noise by example can thus be solved by choosing a procedural noise function and setting the parameters of the noise function such that the power spectrum of the noise is similar to that of the exemplar. This raises three important questions: (i) What are the appropriate exemplars? (ii) How is the power spectrum of the exemplar obtained? (iii) What is the appropriate procedural noise function?

3.1 Gaussian Textures

We use Gaussian textures as exemplars, since these are defined, similarly to noise, as what is known in statistics as a Gaussian random field [Papoulis and Pillai 2002]. Both Gaussian textures and noise are completely characterized by their power spectrum. The class of Gaussian textures *roughly* corresponds to the class of stochastic textures. Galerne et al. [2011] recently presented methods that can be used to generate the Gaussian version of a texture. These methods explicitly destroy the information contained in the phase spectrum, only preserving the information contained in the power spectrum. We use the methods of Galerne et al. to illustrate the class of Gaussian textures, to determine how Gaussian an exemplar is, and to provide a ground truth for our approach.

3.2 Power Spectrum Estimation

We estimate the power spectrum of the exemplar using the periodogram, since this requires only a single exemplar. The periodogram, defined as the magnitude squared of the Fourier transform, is a simple estimator for the power spectrum [Press et al. 2002, 13.4]. Unfortunately, the periodogram is inherently noisy,

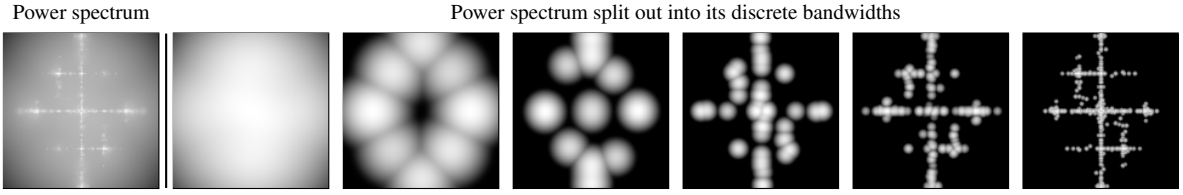

Figure 2: *Bandwidth-quantized Gabor noise. (a) A power spectrum composed of Gaussians with discrete bandwidths. (b-g) The power spectrum split out into its discrete bandwidths. The bandwidth quantization enables robust parameter estimation and efficient procedural evaluation.*

since it is actually a white noise process with the power spectrum as mean [Papoulis and Pillai 2002, 12.2]. Note that a less noisy estimate could be obtained by averaging multiple peridograms, but that would require several exemplars. The periodogram is based on the Fourier transform. However, the Fourier transform of a non-periodic textures suffers from a boundary problem, which causes a cross-shaped artifact in the estimated power spectrum. We avoid this problem by using the Fourier transform of the periodic component instead [Moisan 2011; Galerne et al. 2011].

3.3 Gabor Noise

We use Gabor noise [Lagae et al. 2009], since it has a power spectrum that can be used as a basis to approximate arbitrary power spectra. Two-dimensional anisotropic random-phase Gabor noise [Lagae and Drettakis 2011] is defined as

$$n\left(\mathbf{x}; K, a, \omega\right) = \frac{1}{\sqrt{\lambda}} \sum_i g\left(\mathbf{x} - \mathbf{x}_i; K, a, \omega, \phi_i\right), \qquad (1)$$

where K, a and ω are the amplitude, bandwidth and frequency of the noise, g is the phase-augmented Gabor kernel, the random positions $\{\mathbf{x}_i\}$ are distributed according to a Poisson process with mean λ, and the random phases $\{\phi_i\}$ are distributed according to a uniform distribution on the interval $[0, 2\pi)$. The phase-augmented Gabor kernel is defined as

$$g\left(\mathbf{x}; K, a, \omega, \phi\right) = K e^{-\pi a^2 |\mathbf{x}|^2} \cos\left(2\pi \mathbf{x} \cdot \omega + \phi\right), \qquad (2)$$

where K, a, ω and ϕ are the amplitude, bandwidth, frequency and phase of the kernel. The variance of the noise is $K^2/4a^2$. The power spectrum of the noise is

$$S_n\left(\xi; K, a, \omega\right) = \frac{K^2}{8a^2} \mathcal{G}\left(\xi; \pm\omega, \frac{a}{2\sqrt{\pi}}\right), \qquad (3)$$

where $\mathcal{G}\left(\mathbf{x}; \mu, \sigma\right)$ is the 2D normalized Gaussian function with mean μ and standard deviation σ, and $\mathcal{G}\left(\mathbf{x}; \pm\mu, \sigma\right)$ is a shorthand notation for $\mathcal{G}\left(\mathbf{x}; \mu, \sigma\right) + \mathcal{G}\left(\mathbf{x}; -\mu, \sigma\right)$. The power spectrum of Gabor noise is thus a symmetric pair of Gaussians in the frequency domain with magnitude K, frequency $\pm\omega$ and bandwidth a.

3.4 Gabor Noise by Example

We solve the problem of Gabor noise by example by decomposing the noisy power spectrum estimate of the exemplar into a sum of Gaussians, and evaluating the corresponding sum of Gabor noises. However, a straightforward implementation of this idea is problematic. A first problem is that evaluating a potentially large number of Gabor noises is inefficient (see Sec. 6.1). A second problem is that allowing arbitrary values for the bandwidth parameter of the Gabor kernels is problematic, both for the decomposition of the power spectrum (see Sec. 5.2), as well as for the evaluation (see Sec. 6.1). To overcome these problems we introduce bandwidth-quantized Gabor noise (Sec. 4). We solve the first problem by grouping the Gabor kernels of the Gabor noises into a single Gabor noise. We solve the second problem by quantizing the bandwidths of the Gabor kernels. This, in turn, allows us to introduce a robust parameter estimation technique (Sec. 5) and an efficient procedural evaluation scheme (Sec. 6) for bandwidth-quantized Gabor noise.

4 Bandwidth-Quantized Gabor Noise

In this section, we introduce *bandwidth-quantized Gabor noise*, a generalization of Gabor noise to arbitrary power spectra, that quantizes the bandwidths of the Gabor kernels, in order to enable robust parameter estimation and efficient procedural evaluation.

4.1 Arbitrary Power Spectra

Our key insight for generalizing Gabor noise to arbitrary power spectra is that an arbitrary symmetric power spectrum $S\left(\xi\right)$ can be decomposed into a sum of G symmetric pairs of Gaussians $S\left(\xi\right) = \sum_g S_{n_g}\left(\xi; K_g, a_g, \omega_g\right)$ of the form of Eqn. 3. This implies that the corresponding noise $n\left(\mathbf{x}\right)$ can be seen as the sum of the G corresponding independent Gabor noises $n\left(\mathbf{x}\right) = \sum_g n_g\left(\mathbf{x}; K_g, a_g, \omega_g\right)$ of the form of Eqn. 1. Note that we use the subscript $_g$ to index the noise functions corresponding to the Gaussians. This decomposition is similar in spirit to how an arbitrary function can be approximated to any prescribed tolerance by a sum of Gaussians [Ferreira 1998]. However, in our case, all involved functions are symmetric, since they correspond to power spectra of real signals, and the weights of the Gaussians are positive, since power spectra are positive by definition. This is why we use the random-phase variant of Gabor noise [Lagae and Drettakis 2011] rather than the original one [Lagae et al. 2009]: The power spectrum of the original variant has an additional Gaussian centered at zero, which would make this decomposition impossible, since excess power cannot be subtracted away.

4.2 Kernel Combination

Evaluating $n\left(\mathbf{x}\right)$ by evaluating and summing the G corresponding Gabor noises is inefficient, especially when G is relatively large (see Sec. 6.1). We therefore combine all the Gabor kernels of the G noises into a single but more complex Gabor noise,

$$n\left(\mathbf{x}\right) = \frac{1}{\sqrt{\lambda}} \sum_i \frac{1}{\sqrt{p_i}} g\left(\mathbf{x} - \mathbf{x}_i; K_i, a_i, \omega_i, \phi_i\right), \qquad (4)$$

where $\lambda = \sum_g \lambda_g$ and the parameters of each Gabor kernel $\{(K_i, a_i, \omega_i)\}$ are randomly chosen from $\{(K_g, a_g, \omega_g)\}$, the parameters of the G noises, with probability $p_g = \lambda_g / \lambda$.

4.3 Bandwidth Quantization

Allowing arbitrary values for the bandwidth a_g is problematic, both for the parameter estimation (see Sec. 5.2) as well as for the procedural evaluation (see Sec. 6.1). We therefore quantize the bandwidth to a small set of B bandwidths $\mathcal{B} = \{a_b\}$, and regroup all the Gabor kernels which share the same bandwidth. We use the bandwidth discretization $a_b = 2^{-b}\sqrt{\pi/2\ln 2}$, motivated by Gabor filter bank design for texture analysis [Bovik et al. 1990], which corresponds to a set of Gaussians with a full width at half maximum (FWHM) of powers of two.

4.4 Bandwidth-Quantized Gabor Noise

We define 2D *bandwidth-quantized Gabor noise* as

$$\boxed{n(\mathbf{x}) = \sum_{b\in\mathcal{B}} \frac{1}{\sqrt{\lambda_b}} \sum_i \frac{1}{\sqrt{p_{b,i}}} g\left(\mathbf{x} - \mathbf{x}_{b,i}; K_{b,i}, a_b, \omega_{b,i}, \phi_{b,i}\right),} \tag{5}$$

where $\lambda_b = \sum_g \lambda_{b,g}$ and the parameters of each Gabor kernel $\{(K_{b,i}, \omega_{b,i})\}$ are randomly chosen from $\{(K_{b,g}, \omega_{b,g})\}$, the parameters of the G_b noises that share the same bandwidth a_b, with probability $p_{b,g} = \lambda_{b,g}/\lambda_b$. Note that $\sum_b G_b = G$, and that we use the subscript $_{b,g}$ to index the noise functions corresponding to the Gaussians in each bandwidth. The power spectrum of the noise is

$$\boxed{S_n(\xi) = \sum_{b\in\mathcal{B}} \sum_{g=0}^{G_b-1} \frac{K_{b,g}^2}{8a_b^2} \mathcal{G}\left(\xi; \pm\omega_{b,g}, \frac{a_b}{2\sqrt{\pi}}\right),} \tag{6}$$

i.e., a sum of Gaussians partitioned into a discrete set of bandwidths. This is illustrated in Fig. 2. The variance of the noise is $\sigma_n^2 = \sum_b \sum_g \frac{K_{b,g}^2}{4a_b^2}$, where $\sigma_{b,g}^2 = K_{b,g}^2/4a_b^2$ is the variance of the g-th Gaussian of bandwidth b, and $\sigma_b^2 = \sum_g \sigma_{b,g}^2$ is the variance of bandwidth b.

5 Robust Parameter Estimation

In this section, we introduce a robust parameter estimation technique for bandwidth-quantized Gabor noise, which automatically decomposes the noisy power spectrum estimate of an exemplar into a sparse sum of Gaussians, ensuring a compact procedural representation.

5.1 The Parameter Estimation Problem

The goal of the parameter estimation is to determine the parameters of a bandwidth-quantized Gabor noise such that its power spectrum is close to that of the exemplar. Intuitively, this corresponds to fitting the power spectrum of the exemplar with a sum of Gaussians.

We denote the $M \times M$ discrete power spectrum estimate of the exemplar as $S_{ex}(\xi_0, \xi_1)$ ($\xi_0, \xi_1 \in -M/2, \ldots, M/2-1$). We discretize the power spectrum of bandwidth-quantized Gabor noise (Eqn. 6) at the same resolution of the exemplar by placing for each bandwidth b at each discrete frequency (m_0, m_1) ($m_0, m_1 \in -M/2, \ldots, M/2-1$) a Gaussian with magnitude $K_{b,(m_0,m_1)}$,

$$S_n(\xi_0,\xi_1) = \sum_{b\in\mathcal{B}} \sum_{(m_0,m_1)} \frac{K_{b,(m_0,m_1)}^2}{8a_b^2} \mathcal{G}\left((\xi_0,\xi_1); \pm\left(\frac{m_0}{M}, \frac{m_1}{M}\right), \frac{a_b}{2\sqrt{\pi}}\right). \tag{7}$$

Note that this discretization restricts the values for the frequency ω to the discrete frequencies of the exemplar, but does not restrict the values for the magnitude K. We group the parameters of Eqn. 7 into an $B \times M \times M$ parameter vector $\alpha(b, m_0, m_1) = K_{b,(m_0,m_1)}^2$. We denote the discrete power spectrum of bandwidth-quantized Gabor noise with parameters α as $S_n(\alpha)$.

Using the above notation, the goal of the parameter estimation is to find a parameter vector α satisfying three constraints: (i) $S_n(\alpha)$ is close to S_{ex}, i.e., the power spectrum of the noise is close to that of the exemplar, (ii) α is non-negative, i.e., all Gaussians are positive, and (iii) α is sparse, i.e., the representation is compact. Once such a parameter vector α is found, the parameters of the bandwidth-quantized Gabor noise are fully determined. Every non-zero entry of $\alpha(b, m_0, m_1)$ determines a Gaussian with bandwidth a_b, magnitude $K = \sqrt{\alpha(b, m_0, m_1)}$ and frequency $\omega = (m_0/M, m_1/M)$. The number of non-zero entries of α also determines the number of Gaussians G, the number of bandwidths B, and the number of Gaussians in each bandwidth G_b. We define the sparseness ratio of α as G/M^2, the number of Gaussians relative to the number of pixels of the exemplar.

5.2 Solving the Parameter Estimation Problem using Non-Negative Basis Pursuit Denoising

We solve the parameter estimation problem as a non-negative basis pursuit denoising (NNBPDN) problem (e.g., [Kim et al. 2007]),

$$\begin{cases} \text{minimize} & \|S_n(\alpha) - S_{ex}\|_2^2 + \nu\|\alpha\|_1 \\ \text{subject to} & \alpha \geq 0 \end{cases}, \tag{8}$$

where $\|\cdot\|_1$ is the ℓ^1-norm and ν is a regularization parameter. We use non-negative basis pursuit denoising because it satisfies the three constraints above, in particular, because it is designed to produce sparse solutions. This guarantees a compact procedural representation, and can deal with the noisy power spectrum estimate of the exemplar. Note that with an ℓ^2 rather than an ℓ^1-norm regularization, nearly none of the components of α would be zero.

The regularization parameter ν determines the sparseness of the solution. However, the relation between ν and the sparseness depends on the exemplar. We therefore introduce a relative regularization parameter κ, and define ν as $\kappa\nu_{\max}$, where $\nu_{\max}$ is the smallest value of ν for which $\alpha = 0$ is a solution of Eqn. 8, i.e., $\nu_{\max} = 2\left\|S_n^T(S_{ex})\right\|_\infty$ [Kim et al. 2007]. We expose the relative regularization parameter κ to the user in order to allow the user to trade sparseness for accuracy (and vice versa). Additionally, we solve Eqn. 8 following the homotopy strategy, i.e., for a sequence of kappa values $1/2^i$, initializing each step with the result of the previous one. This is more efficient, and provides the user with intermediate solutions. This is illustrated in Fig. 3.

One of the reasons why we have quantized the bandwidth in Sec. 4.3 is to guarantee a robust optimization: A fixed set of bandwidths allows us to formulate our optimization as a convex optimization, while a sufficiently spaced set of bandwidths ensures a well-conditioned optimization.

5.3 Implementation

We numerically solve Eqn. 8 using the fast iterative shrinkage-thresholding (FISTA) algorithm of Beck and Teboulle [2009], combined with the duality gap stopping criterion associated to the corresponding convex problem [Kim et al. 2007; Mairal et al. 2011]. We take symmetry into account by restricting (m_0, m_1) to the positive half-plane. For large Gaussians, we restrict the possible values for the frequency ω to the discrete frequencies of a sub-sampled

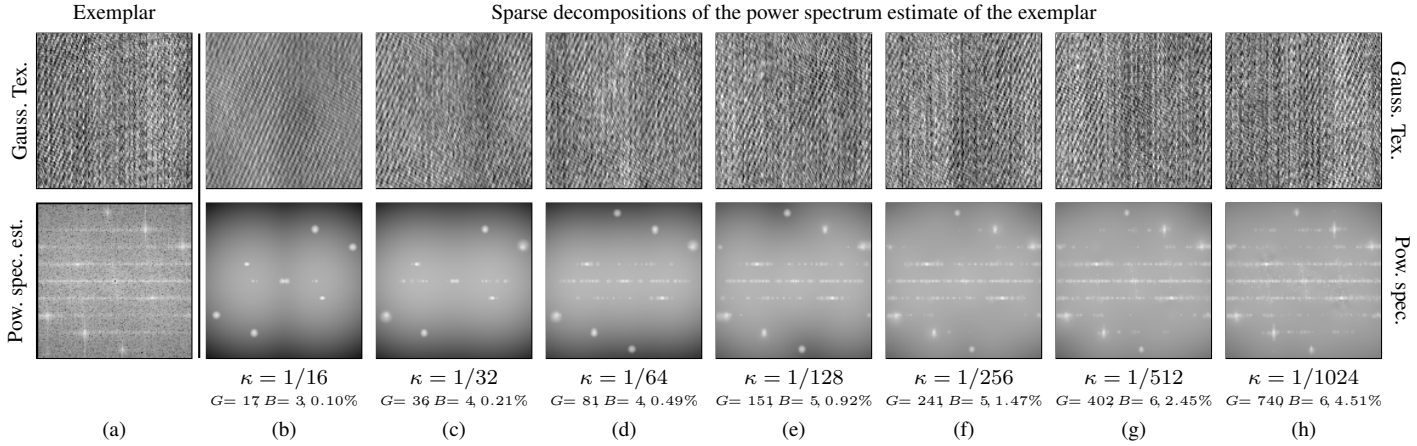

Figure 3: *Robust parameter estimation for bandwidth-quantized Gabor noise. (a) A Gaussian exemplar and its power spectrum estimate. (b-h) Sparse decompositions of the power spectrum estimate into Gaussians with discrete bandwidths for decreasing values of the relative regularization parameter κ. The sparse decomposition ensures a compact procedural representation.*

version of the exemplar, thus ensuring that Eqn. 8 does not become ill-conditioned. We implemented our solver in Matlab.

6 Efficient Procedural Evaluation

In this section, we introduce an efficient procedural evaluation for bandwidth-quantized Gabor noise, that uses multi-grid evaluation and importance sampling of the kernel parameters.

6.1 Multi-Grid Evaluation

The procedural evaluation of Gabor noise [Lagae et al. 2009] uses a grid with a cell size equal to the kernel radius r to efficiently generate the random positions of the kernels that overlap the point of evaluation. The kernel radius r, and thus the cell size of the grid, is determined by the bandwidth a of the Gabor kernels, and is defined as $r = 1/a$. However, this does not work for bandwidth-quantized Gabor noise, which consists of kernels with different bandwidths. One of the reasons why we have quantized the bandwidth in Sec. 4.3 is to be able to evaluate bandwidth-quantized Gabor noise: It allows us to interpret bandwidth-quantized Gabor noise as a sum of B independent Gabor noises that each consist only of Gabor kernels with bandwidth a_b, which, in turn, allows us to efficiently evaluate bandwidth-quantized Gabor noise using a multi-grid evaluation, where a grid with a cell size of $r_b = 1/a_b$ is associated to each of the B noises. Note that evaluating bandwidth-quantized Gabor noise by independently evaluating and summing the G noises instead would be inefficient, because each grid has an associated setup cost and G is typically much larger than B. Also note that for the multi-grid evaluation the actual value of the bandwidths $\{a_b\}$ does not matter.

6.2 Importance Sampling of Kernel Parameters

The procedural evaluation of Gabor noise [Lagae et al. 2009] allows the user to trade quality for speed (and vice versa) by exposing a parameter called the impulse budget. The impulse budget N is defined as the expected number of kernels that overlaps the point of evaluation, and determines the impulse density λ, which is in turn defined as $\lambda = N/\pi r^2$, where r is the kernel radius. However, for bandwidth-quantized Gabor noise, which can be seen as a sum of B sums of G_b Gabor noises $n_{b,g}$, the impulse density $\lambda_{b,g}$ of each of the noises must be determined.

We distribute the impulse budget N over the G noises proportionally to their variance $\sigma^2_{b,g}$, since the contribution of each of these noises can differ significantly. This is done by setting $N_{b,g}$ to $\left(\sigma^2_{b,g}/\sigma^2\right) N$. This implies that $N_b = \left(\sigma^2_b/\sigma^2\right) N$, the impulse budget for each bandwidth, is proportional to the variance of that bandwidth, and that $p_{b,g} = \sigma^2_{b,g}/\sigma^2_b$, the kernel parameters, are selected with a probability proportional to the variance of the corresponding Gaussian.

In contrast to Gabor noise, bandwidth-quantized Gabor noise thus requires sampling discrete probability distributions to determine the parameters of each kernel. The efficiency of the sampling procedure is essential for performance. Additionally, the efficiency of the preprocess that is required for the sampling procedure is essential as well, since these distributions change whenever the noise parameters change (e.g., during editing).

We therefore use the alias method [Walker 1977], which features a constant-time sampling procedure and a linear-time preprocess. To sample a discrete probability distribution, the alias method builds two tables, the alias table A and the probability table F, after which sampling is done by generating a random index i, and returning the i-th value with probability $F[i]$, or the $A[i]$-th value with probability $1 - F[i]$. Several linear-time algorithms for building the tables required by the alias method are available [Walker 1977; Vose 1991]. However, these algorithms require linear auxiliary storage (typically two linked lists), which can be problematic (e.g., when using CUDA or GLSL) and inefficient (due to dynamic memory allocations). Following a hint in the work of Vose [1991, Sec. VI], we have implemented an in-place linear-time algorithm that does not require auxiliary storage.

6.3 Implementation

In contrast to Gabor noise [Lagae et al. 2009], bandwidth-quantized Gabor noise operates at much higher impulse densities. We therefore generate random numbers distributed according to a Poisson distribution with mean λ using the Gaussian approximation $P = \lfloor \lambda + \sqrt{\lambda} X + 1/2 \rfloor$, where X is distributed according to the standard normal distribution. We generate random numbers distributed according to the standard normal distribution using the basic form of Box-Muller transform $X = \sqrt{-2 \ln (U_1)} \cos (2\pi U_2)$, where U_1 and U_2 are distributed according to the standard uniform distribution. We seed the random number generator of each cell using an approach based on random number tables, $\text{seed}(b, x, y) = P_b[b\%L] \oplus P_x[x\%L] \oplus P_y[y\%L]$, where P_b, P_x and P_y are random number tables of size L (typically 256), and $\%$ and $\oplus$ denote

modulo and XOR. Please see the supplemental material for a more detailed discussion. We implemented our procedural evaluation in CUDA.

7 Maximally Independent Color Space

In this section, we introduce the maximally independent color space, a new color space for independent channel synthesis, which we use to synthesize color Gaussian textures with our method. Note that this is only one specific way to obtain colored noise-based textures. Our noise patterns can be used in any way conventional noise patterns are used.

Previous work, inspired by the work of Heeger and Bergen [1995], typically performs independent channels synthesis in a PCA color space, obtained by diagonalizing the covariance matrix using principal component analysis (PCA). However, the use of a PCA color space in this context is somewhat contrived, both in theory, since color channels with zero covariance are not necessarily independent, and in practice, since some previous work reports it works well (e.g., [Qin and Yang 2007]), while some reports the contrary (e.g., [Kopf et al. 2007]). We therefore introduce the maximally independent color space, a new color space for independent channel synthesis where the color channels are maximally independent.

We explain our new maximally independent color space, as well as its relation to the PCA color space used in previous work, using techniques from independent component analysis (ICA) [Hyvärinen et al. 2001].

We define the *lagged correlation matrices* [Hyvärinen et al. 2001, 18.1] $\mathbf{C_I}(\xi)$ of the $M \times N$ RGB image $\mathbf{I}(\mathbf{x})$ as

$$\begin{aligned}\mathbf{C_I}(\xi) &= \frac{1}{M^2}\sum_{\mathbf{x}}\left(\mathbf{I}(\mathbf{x})-\mu_{\mathbf{I}}\right)\left(\mathbf{I}(\mathbf{x}+\xi)-\mu_{\mathbf{I}}\right)^T \\ &= \begin{pmatrix} C_{\mathbf{I}_R,\mathbf{I}_R}(\xi) & C_{\mathbf{I}_R,\mathbf{I}_G}(\xi) & C_{\mathbf{I}_R,\mathbf{I}_B}(\xi) \\ C_{\mathbf{I}_G,\mathbf{I}_R}(\xi) & C_{\mathbf{I}_G,\mathbf{I}_G}(\xi) & C_{\mathbf{I}_G,\mathbf{I}_B}(\xi) \\ C_{\mathbf{I}_B,\mathbf{I}_R}(\xi) & C_{\mathbf{I}_B,\mathbf{I}_G}(\xi) & C_{\mathbf{I}_B,\mathbf{I}_B}(\xi) \end{pmatrix}, \end{aligned} \quad (9)$$

where ξ is the lag and $\mu_{\mathbf{I}}$ is the mean of $\mathbf{I}$, and $C_{\mathbf{I}_i,\mathbf{I}_j}(\xi)$ is the cross-correlation of i-th channel with the j-th channel shifted over ξ. The lagged correlation matrices of the image $\mathbf{I}$ after applying a color transform $\mathbf{T}$ are $\mathbf{T}\mathbf{C_I}(\xi)\mathbf{T}^T$.

Using this notation, the PCA color space used in previous work is obtained by finding a matrix $\mathbf{T}_{PCA}$ that diagonalizes the correlation matrix $\mathbf{C_I}(\mathbf{0})$ using PCA. Although this implies that the color channels have zero covariance, the matrix $\mathbf{T}_{PCA}$ does not diagonalize the lagged correlation matrices $\mathbf{C_I}(\xi)$ for $\xi \neq \mathbf{0}$, which implies that the channels are not independent.

We instead obtain our new maximally independent color space by finding the matrix $\mathbf{T}_{AJD}$ that approximately diagonalizes all lagged correlation matrices $\mathbf{C_I}(\xi)$, and not just $\mathbf{C_I}(\mathbf{0})$. This is motivated by the fact that diagonalizing the lagged correlation matrices transforms the mixed channels into independent ones in the ICA framework [Hyvärinen et al. 2001, 18.1]. We find the matrix $\mathbf{T}_{AJD}$ using approximate joint diagonalization (AJD) [Hyvärinen et al. 2001, 11.3]. More specifically, we minimize the objective function $\mathcal{J}$,

$$\mathcal{J}(\mathbf{T}) = \sum_{\xi}\mathbf{off}\left(\mathbf{T}\mathbf{C_I}(\xi)\mathbf{T}^T\right), \quad (10)$$

where $\mathbf{off}(\mathbf{A}) = \sum_{i\neq j} a_{ij}^2$ is a measure for the diagonality of a matrix, using the algorithm of Cardoso and Souloumiac [1996].

The value of $\mathcal{J}(\mathbf{T})$ is a measure for the independence of the color channels in the color space determined by $\mathbf{T}$. This allows us to compare the effectiveness of the RGB, PCA and maximally independent color space by comparing the values of $\mathcal{J}(\mathbf{T}_{RGB} = \mathbf{I})$, $\mathcal{J}(\mathbf{T}_{PCA})$ and $\mathcal{J}(\mathbf{T}_{AJD})$. Please refer to the supplemental materials for more details.

8 Interactive Noise Editing

In this section, we present an interactive editor for bandwidth-quantized Gabor noise, inspired by the Wold decomposition [Francos et al. 1993]. Although our approach is exemplar-based, it still allows editing for maximum flexibility.

Our editor is inspired by the *Gabor noise widgets* of Lagae et al. [2009, 3.3], which allow the user to manipulate the parameters of a Gabor noise by directly manipulating the corresponding Gaussians in the spectral domain. However, in contrast to Lagae et al., we have to deal with tens or even hundreds of Gaussians. We therefore group the Gaussians, and allow the user to manipulate the groups rather than the individual Gaussians. We group the Gaussians automatically inspired by the Wold decomposition [Francos et al. 1993], which states that a Gaussian random field can be decomposed into a stochastic component, a set of strongly anisotropic components, and a set of periodic components. We use guidelines from the Wold decomposition [Liu 1997] to perform a basic decomposition of the sparse set of Gaussians. Please refer to the supplemental materials for more details. Note that, in contrast to previous work [Francos et al. 1993; Liu 1997] our goal is not to fully decompose a discrete power spectrum, but rather to provide meaningful groups of Gaussians to the user, which is less challenging. We allow the user to manipulate groups by rotating, scaling and translating groups, which affects the frequency ω of the Gaussians in the group, and by changing the magnitude and bandwidth of groups, which affects the magnitude K and bandwidth a of the Gaussians in the group. We also allow the user to manually group Gaussians, and to cut, copy and paste groups.

9 Results, Comparisons and Discussion

9.1 Results

In Fig. 4, we show detailed results of our approach. Col. 1 shows the exemplar. Col. 2 shows the Gaussian version of the exemplar. Differences between col. 1 and col. 2, if any, are due to the the fact that the exemplar is not necessarily Gaussian. Col. 2 also serves as ground truth for our approach. Col. 3 shows the Gaussian texture corresponding to the sparse power spectrum estimated using our parameter estimation. Differences between col. 2 and col. 3, if any, are due to the approximate nature of the parameter estimation. Col. 4 shows the noise generated by our procedural evaluation. Differences between col. 3 and col. 4, if any, are due to the various approximations in the procedural evaluation.

In Fig. 1, we show more results of our approach for the grayscale case. This figure shows the Gaussian version of the exemplar (row 1) and the procedural noise (row 2), and illustrates that our approach works very well for Gaussian textures. In Fig. 5 (and in Fig. 7(col. 1-2)), we show more results of our approach for the color case. This figure shows the exemplar (row 1) and the procedural noise (row 2), and illustrates that our approach works well for many nearly-Gaussian textures (a-e), but less well for non-Gaussian texures (f-h), including structured textures and textures with non-Gaussian color distributions. Please refer to the supplemental material for detailed results for a set of 44 textures, including all examples in the paper.

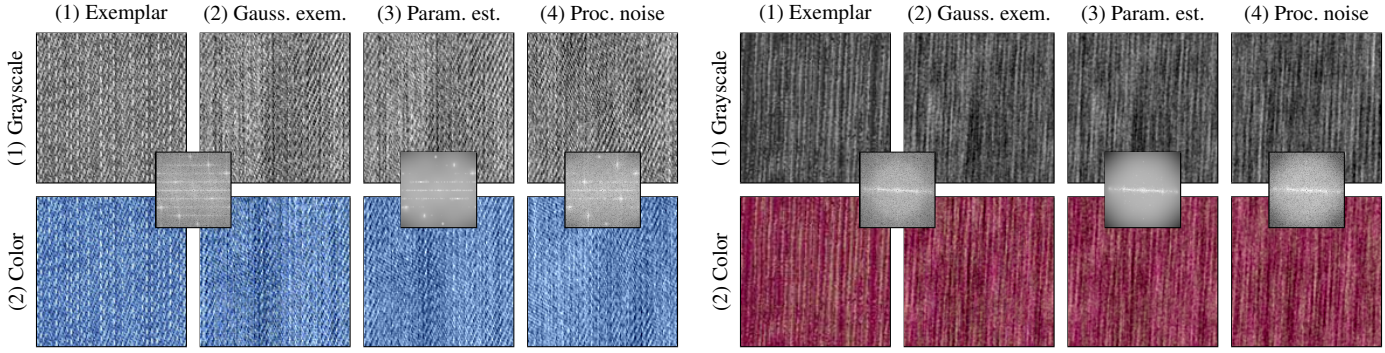

Figure 4: *Detailed results. (col. 1) Exemplar. (col. 2) Gaussian version of the exemplar. (col. 3) Gaussian texture corresponding to the sparse power spectrum estimated using our robust parameter estimation. (col. 4) Procedural noise obtained using our efficient procedural evaluation. (insets) The power spectrum estimate or power spectrum. (row 1) Grayscale results. (row 2) Color results obtained using our maximally independent color space.*

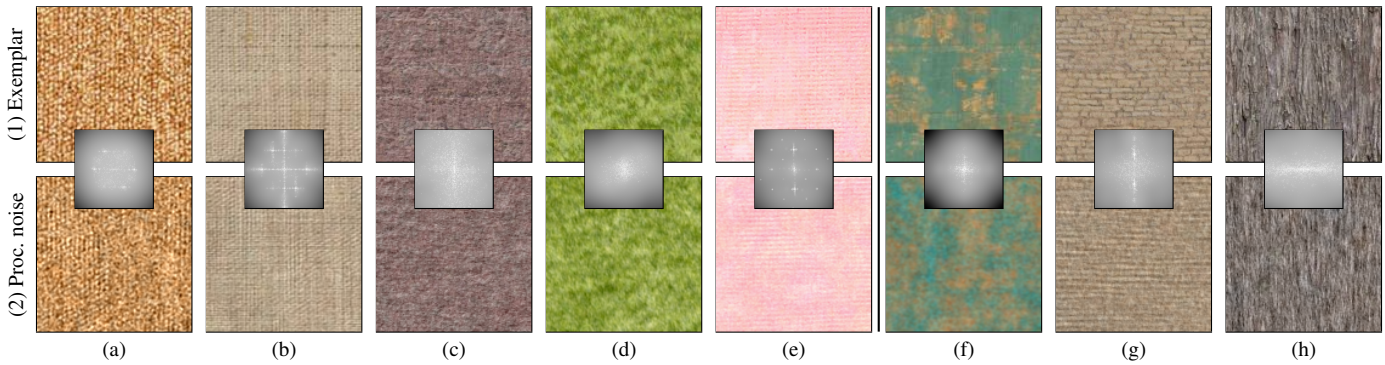

Figure 5: *More results. (row 1) Exemplar. (row 2) Procedural noise. (insets) Sparse power spectrum representation obtained using our robust parameter estimation. (a-e) Nearly-Gaussian textures. (f-h) Non-Gaussian textures.*

The two parameters of our approach are (i) the relative regularization parameter κ, which trades sparseness for accuracy in the parameter estimation, and (ii) the impulse budget N, which trades quality for speed in the procedural evaluation. We used the default parameters $\kappa = 1/256$ and $N = 1024$ for all examples, which in practice makes our method automatic. Note that for many exemplars, better results could be obtained by tweaking κ and N.

The performance of our parameter estimation is roughly two minutes per texture (unparallelized Matlab, 2.67GHz Intel Xeon X5650). However, we believe this could be significantly improved if needed. The performance of our procedural evaluation is roughly 30 FPS (CUDA, NVIDIA Quadro 5000). Both numbers are for the grayscale case, an image resolution of 128×128, and the default parameters. The performance is roughly linear in the image resolution and N. The performance is only marginally lower in the color case, since typically one channel dominates in the maximally independent color space. Please refer to the supplementary materials for exact values.

The sparseness of our representation of the estimated power spectrum is on the average roughly 3.5%, for the default value of κ. Please refer to the supplementary materials for exact values. These results suggest that the power spectrum of Gaussian textures can be sparsely represented.

Our maximally independent color space improves results over the PCA color space, but in practice this difference is barely noticeable. For nearly-Gaussian exemplars, $\mathcal{J}(\mathbf{T}_{RGB}) >> \mathcal{J}(\mathbf{T}_{PCA}) > \mathcal{J}(\mathbf{T}_{AJD})$, but $\mathcal{J}(\mathbf{T}_{AJD})$ is relatively close to $\mathcal{J}(\mathbf{T}_{PCA})$, e.g., for Fig. 4(a), the values are respectively 0.9962, 0.0067 and 0.0028. Please refer to the supplementary materials for all values. Nevertheless, the fact that $\mathcal{J}(\mathbf{T}_{PCA})$ is relatively close to $\mathcal{J}(\mathbf{T}_{AJD})$, the optimal value, explains why the PCA color space has been successful for stochastic textures in previous work. We have not found any Gaussian texture where independent channel synthesis in the maximally independent color space fails, which seems to suggest that the color channels of Gaussian textures are largely independent. For non-Gaussian exemplars, such as [Kopf et al. 2007, Fig. 2], the maximally independent channels might still be too correlated to allow successful independent channel synthesis. Interestingly, for this example, both $\mathcal{J}(\mathbf{T}_{RGB})$ and $\mathcal{J}(\mathbf{T}_{PCA})$ are relatively large, which seems to suggest that neither color space will allow successful independent channel synthesis.

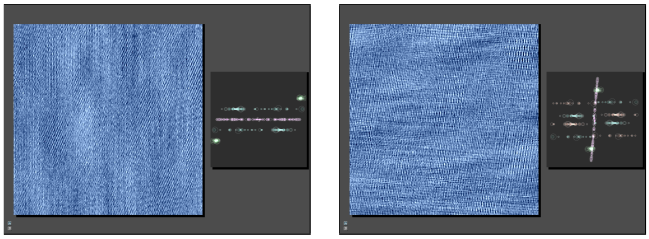

Figure 6: *Interactive noise editing. Two screen captures of an interactive editing session. Please refer to the supplemental materials for the corresponding video sequence.*

In Fig. 6 and in the video sequences in the supplemental materials we show results of our interactive noise editor. These demonstrate several editing operations. Note that none of the other examples

has been edited, and that the editor adapts the impulse budget to guarantee interactive performance.

Our approach inherits all advantages of Gabor noise, including surface mapping without a texture parameterization (surface Gabor noise) and analytic texture filtering [Lagae et al. 2009].

9.2 Comparisons

In Fig. 7, we compare our approach to the methods of Heeger and Bergen [1995], Lefebvre and Hoppe [2005], Lagae et al. [2010b], and Gilet et al. [2010]. Please refer to the supplemental material for a more detailed version of this comparison for a set of 19 textures. Col. 1 shows the exemplar. Col. 2 shows our results. Col. 3 shows the results of Gilet et al. These results were generated by the authors of [Gilet et al. 2010] themselves, who reported that significant manual intervention and trial and error were needed to achieve these results, while our approach is automatic. Additionally, their results typically exhibit a lack of high-frequency detail, increased regularity, and/or subtle color shifts. Col. 4 shows the results of Lagae et al. Their method cannot handle anisotropy. Col. 5 shows the results of Heeger and Bergen. Their method cannot handle anisotropy well. Note that similar results would probably be obtained with anisotropic noise [Goldberg et al. 2008], which uses the same steerable filters. Col. 6 shows the results of Lefebvre and Hoppe. Their method is not procedural, and cannot preserve the anisotropic detail well in all cases, although this could be alleviated by manually constraining the jitter.

The method of Ghazanfarpour and Dischler [1995] cannot handle noisy textures such as sand, while our method can, and requires the user to manually set parameters (i.e., Err_a and Err_ϕ), while our method is automatic. The method of Dischler et al. [1997] cannot handle anisotropic textures, while our method can.

9.3 Discussion

A simpler alternative to our parameter estimation would be to obtain the K and ω parameter of each Gabor kernel by importance sampling the power spectrum of the exemplar during the procedural evaluation. However, it is not clear how to obtain the a parameter in this case, and fixing it to a large or small bandwidth would respectively be inaccurate or inefficient. Additionally, this alternative is not compact.

An alternative to non-negative basis pursuit denoising are methods based on Gaussian mixture models (GMM) and expectation maximization (EM), such as the one by Papas et al. [2011]. However, non-negative basis pursuit denoising has several advantages over these methods for our problem: (i) it allows the user to specify the desired accuracy of the approximation rather than the desired number of Gaussians; (ii) it does not require an initialization; and (iii) it provides a solution in terms of quantized rather than continuous bandwidths.

Compared to our approach, Gilet et al. [2010] additionally use cosines and histogram matching, which can improve the results in some cases. Although our approach is compatible with these additions, we have chosen not use them, since the usage of cosines and histogram matching is not compatible with surface Gabor noise and analytic filtering of Gabor noise [Lagae et al. 2009].

10 Conclusion

We have presented Gabor noise by example, a generalization of Gabor noise that can generate a wide variety of Gaussian textures, and can estimate the noise parameters from an exemplar Gaussian texture. Gabor noise by example preserves the traditional advantages of procedural noise, including a compact representation and fast on-the-fly evaluation, and is mathematically well-founded.

For all of our examples, the Gaussian version of the exemplar and the procedural noise are virtually indistinguishable. This means that all remaining information that makes the exemplar different from its Gaussian version is in the phase spectrum of the exemplar. We therefore believe that understanding the phase spectrum of textures is an important direction for future research.

Acknowledgements

We would like to thank the anonymous reviewers and Jean-Michel Dischler, Fredo Durand, Guillaume Gilet and Gabriel Peyré. Ares Lagae is a Postdoctoral Fellow of the Research Foundation - Flanders (FWO). This work was partly supported by ANR project MATAIM (ANR-09-BLAN-0029-01), KU Leuven project CREA/08/017, ANR project SIMILAR-CITIES (2008-COORD-021-01), EU project VERVE (`http://www.verveconsortium.eu/`), Autodesk (research and software donations), NVIDIA (Academic Partnership Program) and Adobe (research donation).

References

BECK, A., AND TEBOULLE, M. 2009. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. Img. Sci. 2*, 183–202.

BOVIK, A. C., CLARK, M., AND GEISLER, W. S. 1990. Multichannel texture analysis using localized spatial filters. *IEEE Trans. Pattern Anal. Mach. Intell. 12*, 1, 55–73.

CARDOSO, J.-F., AND SOULOUMIAC, A. 1996. Jacobi angles for simultaneous diagonalization. *SIAM J. Matrix Anal. Appl. 17*, 161–164.

COOK, R. L., AND DEROSE, T. 2005. Wavelet noise. *ACM Trans. Graph. 24*, 3, 803–811.

DISCHLER, J.-M., AND GHAZANFARPOUR, D. 1997. A procedural description of geometric textures by spectral and spatial analysis of profiles. *Comp. Graph. Forum 16*, 3, 129–139.

FERREIRA, P. 1998. A comment on the approximation of signals by gaussian functions. *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process. 45*, 2, 250–251.

FRANCOS, J. M., MEIRI, A. Z., AND PORAT, B. 1993. A unified texture model based on a 2-D Wold-like decomposition. *IEEE Trans. Signal Process. 41*, 8, 2665–2678.

GALERNE, B., GOUSSEAU, Y., AND MOREL, J. 2011. Random phase textures: Theory and synthesis. *IEEE Trans. Image Process. 20*, 1, 257–267.

GHAZANFARPOUR, D., AND DISCHLER, J.-M. 1995. Spectral analysis for automatic 3-d texture generation. *Comp. & Graph. 19*, 3, 413–422.

GHAZANFARPOUR, D., AND DISCHLER, J.-M. 1996. Generation of 3d texture using multiple 2d models analysis. *Comp. Graph. Forum 15*, 3, 311–323.

GILET, G., AND DISCHLER, J.-M. 2010. An image-based approach for stochastic volumetric and procedural details. *Comp. Graph. Forum 29*, 4, 1411–1419.

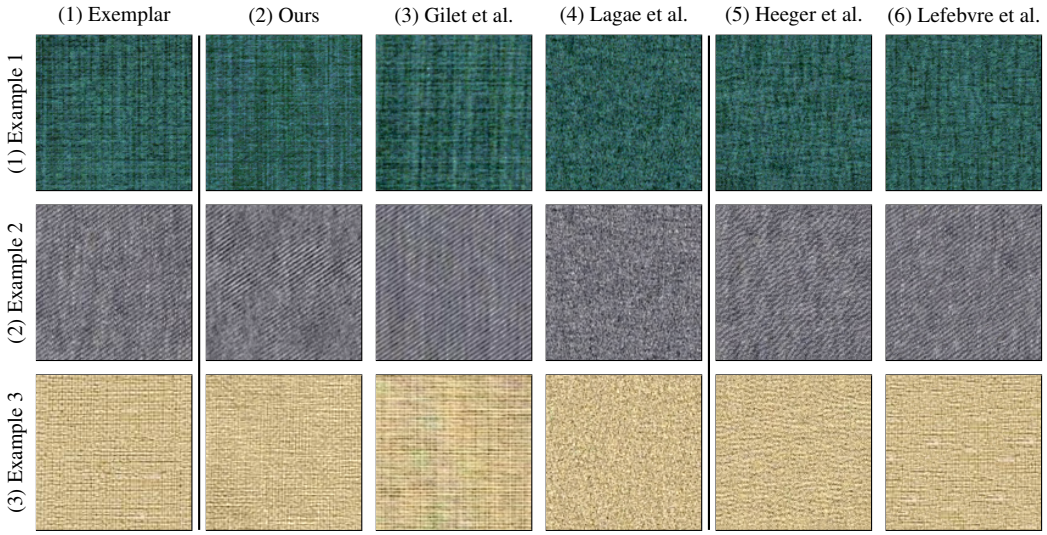

Figure 7: *Comparison. (col. 1) Exemplar. (col. 2) Our result. (col. 3) Result of Gilet et al. (col. 4) Result of Lagae et al. (col. 5) Result of Heeger and Bergen. (col. 6) Result of Lefebvre and Hoppe. Note that the method of Gilet et al. requires significant manual intervention and trial and error, and that the method of Lefebvre and Hoppe is not procedural. Please see Sec. 9.2 for a detailed discussion.*

GILET, G., DISCHLER, J.-M., AND SOLER, L. 2010. Procedural descriptions of anisotropic noisy textures by example. In *EG 2010 - Short papers*, 77–80.

GOLDBERG, A., ZWICKER, M., AND DURAND, F. 2008. Anisotropic noise. *ACM Trans. Graph. 27*, 3, 54:1–54:8.

HEEGER, D. J., AND BERGEN, J. R. 1995. Pyramid-based texture analysis/synthesis. In *Proc. ACM SIGGRAPH 1995*, 229–238.

HYVÄRINEN, A., KARHUNEN, J., AND OJA, E. 2001. *Independent Component Analysis*. John Wiley & Sons.

JESCHKE, S., CLINE, D., AND WONKA, P. 2011. Estimating color and texture parameters for vector graphics. *Comp. Graph. Forum 30*, 2, 523–532.

KIM, S.-J., KOH, K., LUSTIG, M., BOYD, S., AND GORINEVSKY, D. 2007. An interior-point method for large-scale l_1-regularized least squares. *IEEE J. Sel. Topics Signal Process. 1*, 4, III–117–III–120.

KOPF, J., FU, C.-W., COHEN-OR, D., DEUSSEN, O., LISCHINSKI, D., AND WONG, T.-T. 2007. Solid texture synthesis from 2D exemplars. *ACM Trans. Graph. 26*, 3, 2:1–2:9.

LAGAE, A., AND DRETTAKIS, G. 2011. Filtering solid Gabor noise. *ACM Trans. Graph. 30*, 4, 51:1–51:6.

LAGAE, A., LEFEBVRE, S., DRETTAKIS, G., AND DUTRÉ, P. 2009. Procedural noise using sparse Gabor convolution. *ACM Trans. Graph. 28*, 3, 54:1–54:10.

LAGAE, A., LEFEBVRE, S., COOK, R., DEROSE, T., DRETTAKIS, G., EBERT, D. S., LEWIS, J. P., PERLIN, K., AND ZWICKER, M. 2010. A survey of procedural noise functions. *Comp. Graph. Forum 29*, 8, 2579–2600.

LAGAE, A., VANGORP, P., LENAERTS, T., AND DUTRÉ, P. 2010. Procedural isotropic stochastic textures by example. *Comp. & Graph. 34*, 4, 312–321.

LEFEBVRE, S., AND HOPPE, H. 2005. Parallel controllable texture synthesis. *ACM Trans. Graph. 24*, 3, 777–786.

LEWIS, J. P. 1989. Algorithms for solid noise synthesis. In *Comp. Graph. (Proc. ACM SIGGRAPH 89)*, vol. 23, 263–270.

LIU, F. 1997. *Modeling spatial and temporal textures*. PhD thesis, Massachusetts Institute of Technology.

MAIRAL, J., JENATTON, R., OBOZINSKI, G., AND BACH, F. 2011. Convex and network flow optimization for structured sparsity. *J. Mach. Learn. Res. 12*, 2681–2720.

MOISAN, L. 2011. Periodic plus smooth image decomposition. *J. Math. Imag. Vis. 39*, 161–179.

PAPAS, M., JAROSZ, W., JAKOB, W., RUSINKIEWICZ, S., MATUSIK, W., AND WEYRICH, T. 2011. Goal-based caustics. *Comp. Graph. Forum 30*, 2, 503–511.

PAPOULIS, A., AND PILLAI, U. 2002. *Probability, Random Variables and Stochastic Processes*, 4rd ed. McGraw-Hill.

PERLIN, K. 1985. An image synthesizer. In *Comp. Graph. (Proc. ACM SIGGRAPH 85)*, vol. 19, 287–296.

PRESS, W. H., VETTERLING, W. T., TEUKOLSKY, S. A., AND FLANNERY, B. P. 2002. *Numerical Recipes in C++: the art of scientific computing*, 2nd ed. Cambridge University Press.

QIN, X., AND YANG, Y.-H. 2007. Aura 3d textures. *Visualization and Computer Graphics, IEEE Transactions on 13*, 2, 379–389.

VOSE, M. D. 1991. A linear algorithm for generating random numbers with a given distribution. *IEEE Trans. Softw. Eng. 17*, 972–975.

WALKER, A. J. 1977. An efficient method for generating discrete random variables with general distributions. *ACM Trans. Math. Softw. 3*, 3, 253–256.

WEI, L.-Y., LEFEBVRE, S., KWATRA, V., AND TURK, G. 2009. State of the art in example-based texture synthesis. In *Eurographics 2009 State of the Art Reports*, 93–117.

XUE, S., DORSEY, J., AND RUSHMEIER, H. 2011. Stone weathering in a photograph. *Comp. Graph. Forum 30*, 4, 1189–1196.

YOON, J.-C., AND LEE, I.-K. 2008. Stable and controllable noise. *Graph. Models 70*, 5, 105–115.

YOON, J.-C., LEE, I.-K., AND CHOI, J.-J. 2004. Editing noise. *Comp. Anim. Virtual Worlds 15*, 3-4, 277–287.