Random Phase Textures: Theory and Synthesis

Bruno Galerne, Yann Gousseau, Jean-Michel Morel

Random Phase Textures: Theory and Synthesis

B. Galerne[*], Y. Gousseau[†] and J.-M. Morel[*]

Abstract

This paper explores the mathematical and algorithmic properties of two sample-based micro-texture models: *random phase noise* (*RPN*) and *asymptotic discrete spot noise* (*ADSN*). These models permit to synthesize *random phase textures*. They arguably derive from linearized versions of two early Julesz texture discrimination theories. The ensuing mathematical analysis shows that, contrarily to some statements in the literature, *RPN* and *ADSN* are different stochastic processes. Nevertheless, numerous experiments also suggest that the textures obtained by these algorithms from identical samples are perceptually similar. The relevance of this study is enhanced by three technical contributions to micro-texture synthesis from samples. A solution is proposed to three obstacles that prevented the use of *RPN* or *ADSN* to emulate micro-textures. First, *RPN* and *ADSN* algorithms are adapted to color images. Second, a preprocessing is proposed to avoid artifacts due to the non-periodicity of real-world texture samples. Finally, the method is extended to synthesize textures with arbitrary size from a given sample.

Keywords: texture synthesis, random phase, shot noise, spot noise

Online resources: An online demo is available for the *RPN* algorithm discussed in this paper at http://mw.cmla.ens-cachan.fr/megawave/demo/random_phase_noise/. Also, the website http://mw.cmla.ens-cachan.fr/megawave/algo/random_phase_noise/ contains many examples and counterexamples as well as an ANSI C source code.

1 Introduction

1.1 Texture Perception Axioms and their Formalization

Oppenheim and Lim [1] state that "spectral magnitude and phase tend to play different roles" for digital images and that, in some situations, the phase contains many of the important features of images. However when it comes to textures, perception theory suggests that some of the main texture characteristics are contained in their Fourier magnitude. In his early work on texture discrimination Julesz [2] demonstrated that many texture pairs having the same second-order statistics could not be discerned by human preattentive vision. This hypothesis is referred to as the first Julesz axiom for texture perception [2]. As a consequence, textures having the same second-order statistics share a common auto-covariance and therefore a common Fourier magnitude. Even though counterexamples to the first Julesz axiom exist [2, 3, 4], it is believed that Fourier magnitude is more important than phase for the perception of textures [5]. This conclusion is still considered valid by several more recent contributions [6, 7]. For example, working on texture classification, Tang and Stewart [6] conclude that "the Fourier transform magnitudes contain enough texture information for classification" whereas "the Fourier phase information is a noise signal for texture classification."

[*]B. Galerne and J.-M. Morel are with CMLA, ENS Cachan, CNRS, UniverSud, 61 Avenue du Président Wilson, F-94230 Cachan. E-mail: galerne@cmla.ens-cachan.fr and morel@cmla.ens-cachan.fr

[†]Y. Gousseau is with LTCI CNRS, Télécom ParisTech, 46 rue Barrault, F-75013 Paris. E-mail: gousseau@telecom-paristech.fr

Thus, a weak form of the first Julesz assumption is that the perception of a texture is characterized by its Fourier modulus. Under this assumption, its perception should not vary when the texture phase is randomized. This fact turns out to be true for a large class of textures which we shall call in the sequel *micro-textures*. More generally, any two images obtained by randomizing the phase of any given sample image are perceptually very similar. As the experiments displayed here show, this is true regardless of whether the sample image is a texture or not. We shall call in the sequel *random phase texture* any image that is obtained by a phase randomization.

The second Julesz approach to texture preattentive discrimination theory introduced the notion of textons (blobs, terminators, line crossings, etc.) [3]. The texton theory assumes that the density of local indicators (the textons) is responsible for texture preattentive discrimination: images with the same texton densities should not be discriminated. A main axiom of the texton theory is that texture perception is invariant to random shifts of the textons [3]. The shift invariance of this second Julesz theory can be made into a synthesis algorithm building a texture from initial shapes by random shifts. In the Julesz toy algorithm used in his discrimination experiments, this construction was a mere juxtaposition of simple shapes on a regular grid, with random shifts avoiding overlap. This random shift principle can be used to make realistic textures provided a linear superposition is authorized, by which the colors of overlapping objects are averaged. Textures obtained by the combined use of random shifts and linear superposition will be called *random shift textures*. We shall discuss thoroughly their relation to *random phase textures*.

Random phase and random shift textures belong to a linear world where the superposition principle dominates. A sound objection is that linear superposition is not always adapted for natural image formation. Several authors prefer an occlusion principle yielding the stochastic dead leaves model [8, 9, 10]. Indeed, most visible objects do not add up visually in the image ; they hide each other.

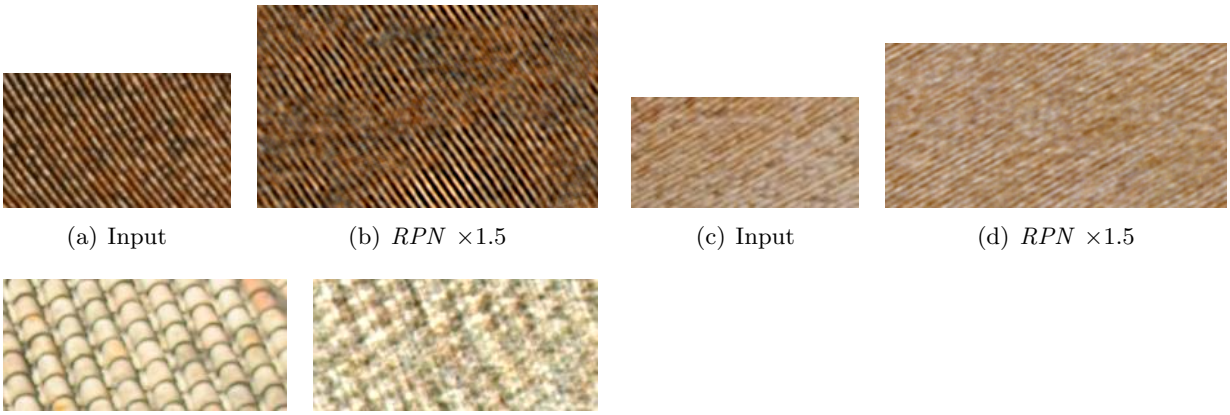
However, thin, small, or semitransparent objects obey an additive superposition principle due to the blur inherent to image formation. More generally, all homogeneous image regions made of small objects, when seen at a distance where individual shapes vanish, obey the linear superposition principle. Indeed, when individual texture constituents are close to pixel size, the camera blur linearly superposes their colors and geometric features. Thus, many homogeneous regions in any image should be *micro-textures* obeying the linear superposition principle and the random shift principle. Fig. 1 shows an example. Five rectangles belonging to various homogeneous regions were picked in a high resolution landscape (1762×1168 pixels). These textures are displayed in pairs where on the left is the original sub-image and on the right is a simulation obtained by the *RPN* algorithm elaborated in this paper. These micro-textures are reasonably well emulated by the *RPN* algorithm. This success encourages *RPN* simulation attempts on the homogeneous regions of any image. Yet, many images or image parts usually termed *textures* do not fit to the micro-texture requisites. Typically, periodic patterns with big visible elements, such as brick walls, are not micro-textures. More generally, textures whose building elements are spatially organized, such as the branches of a tree, are not micro-textures (see Fig. 13). Nonetheless, each textured object has a critical distance at which it becomes a micro-texture. For instance, as illustrated in Fig. 2, tiles at a close distance are a *macro-texture*, and are not amenable to phase randomization. The smaller tiles extracted from the roofs in Fig. 16 can instead be emulated.

1.2 Random Phase and Random Shift Algorithms

The two texture models under study have been used either to create new textures from initial spots, or to analyze texture perception. Emulating real texture samples by these algorithms requires the solution of several technical obstacles which will be treated in Section 5. Here, we first sketch the mathematical and algorithmic discussion.

Random phase textures are produced by *random phase noise (RPN)* which is a very simple

Figure 1: Some examples of micro-textures taken from a single image (water with sand, clouds, sand, waves with water ground, pebbles). The emplacements of the original textures are displayed with red rectangles. Each micro-texture is displayed together with an outcome of the *RPN* algorithm to its right. These micro-textures are reasonably well emulated by *RPN*. Homogeneous regions that have lost their geometric details due to distance are often well simulated by *RPN*



(a) Input (b) *RPN* ×1.5 (c) Input (d) *RPN* ×1.5

(e) Input (f) *RPN* ×1

Figure 2: The first two inputs are rectangles taken from the tiled roofs in Fig. 16. The third input is again a piece of tiled roof taken at a shorter distance. *RPN* does well on tiles viewed at a distance at which they make a micro-texture. *RPN* fails instead on the third sample, which is a macro-texture

Julesz theory	Texture model	Algorithm(s)
1^{st}	Random phase	*RPN*
2^{nd} (textons)	Random shift	*(A)DSN*

Figure 3: Julesz theories, corresponding linear texture types, and the algorithms generating them. *RPN* stands for random phase noise and *(A)DSN* for (asymptotic) discrete spot noise

algorithm performing phase randomization. Random shift textures correspond to a classical model in signal processing called *shot noise* [11]. *Spot noise*, the two-dimensional *shot noise*, was introduced by van Wijk [12, 13] to create new textures from simple spot images (Fig. 4). In this paper we call *discrete spot noise (DSN)* the corresponding discrete model.

Van Wijk [12] claimed that the asymptotics of *discrete spot noise (DSN)* is obtained by uniformly randomizing the phases of all Fourier coefficients. In short, it is claimed that the *DSN* asymptotic process is the *random phase noise* (*RPN*). Our first result here is that the limit of *DSN* is not *RPN* but is another process, which we shall call *asymptotic discrete spot noise (ADSN)*. The difference between the two models lies in the modulus of the Fourier transform of their outcomes. For *RPN*, it is given by the Fourier magnitude of the spot whereas for *ADSN*, it is subject to pointwise multiplication by a Rayleigh noise.

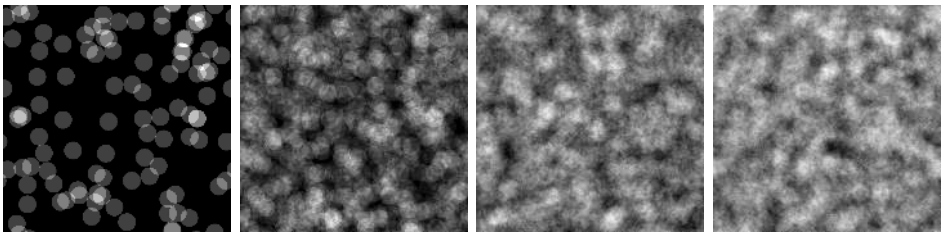
The two investigated algorithms and the corresponding Julesz theories and texture models are summarized in the table of Fig. 3.

It will be shown that *ADSN* and *RPN*, in spite of their theoretical differences, give perceptually similar results and therefore justify van Wijk's approach [12] (see Fig. 6 and 11). These experiments show that the perception of random phase textures is actually robust to the pointwise multiplication of the Fourier magnitude by a Rayleigh noise. By contrast, natural images containing macroscopic structures are in no way robust to this same perturbation (Fig. 16). Hence the perceptual invariance of random phase textures to a multiplicative noise on their magnitude possibly characterizes this kind of texture.

In short, mathematical arguments clarify the asymptotics of *DSN* and also establish a link between Julesz's first and second texture perception theories: their linearized versions give perceptually equivalent textures.

This mathematical and experimental study is completed by three important improvements of the texture synthesis algorithms that stem from both considered randomization processes. The *ADSN* and *RPN* algorithms are first extended to color images by preserving the phase coherence between color channels (Fig. 7). Second, artifacts induced by the non periodicity of the input sample are avoided by replacing the input sample with its periodic component [14]. Eventually, a natural extension of the method permits to synthesize *RPN* and *ADSN* textures with arbitrary size.

The resulting algorithms are fast algorithms based on fast Fourier transform (FFT). They can be used as powerful texture synthesizers starting from any input image. As explained above, the algorithms under consideration do not reproduce all classes of textures: they are restricted to the so-called micro-textures. Exemplar-based methods like those of [15] and the numerous variants that have followed, see *e.g.* [16], successfully reproduce a wide range of textures, including many micro- and macro-textures. However, these methods are also known to be slow, highly unstable, and to often produce garbage results or verbatim copy of the input (see Fig. 14). In contrast, *RPN* and *ADSN* are limited to a class of textures, but are fast, non iterative and parameter free. They are also robust, in the sense that all the textures synthesized from the same original sample are perceptually similar. Speed and stability are especially important in computer graphics, where the classical *Perlin noise* model [17] has been massively used for almost 25 years. Similarly to *ADSN*, the Perlin noise model (as well as its numerous very recent variants [18, 19, 20]) relies on stable and fast noise filters.



(a) $n = 10^2$ (b) $n = 10^3$ (c) $n = 10^4$ (d) $n = 10^5$

Figure 4: Outcomes of the *DSN* associated with the binary image of a small disk for different values of n. As n increases the random images f_n converge towards a stationary texture

The plan of the paper is as follows. The two discrete mathematical models corresponding to Julesz first and second axioms are presented in Sections 2 and 3. The mathematical difference between these two processes is emphasized in Section 4. The corresponding micro-texture synthesis algorithms are introduced in Section 5, and their performance illustrated in Section 6.

2 *Asymptotic Discrete Spot Noise*

2.1 Discrete Spot Noise

We consider the space $\mathbb{R}^{M\times N}$ of discrete, real-valued and periodic rectangular images. The components of an image $f \in \mathbb{R}^{M\times N}$ are indexed on the set $\Omega = \{0,\dots,M-1\}\times\{0,\dots,N-1\}$, and by periodicity $f(x) = f(x_1 \mod M, x_2 \mod N)$ for all $x = (x_1, x_2) \in \mathbb{Z}^2$.

Let h be a real-valued image and let X_p, $p = 1, 2, \dots$ be independent identically distributed (i.i.d.) random variables (RV), uniformly distributed on the image domain Ω. We define the *discrete spot noise (DSN)* of order n associated with the *spot* h as the random image

$$f_n(x) = \sum_{p=1}^{n} h(x - X_p),\ x \in \Omega.$$

Fig. 4 shows several realizations of the *DSN* for several values of n where the spot h is the binary image of a small disk. It appears clearly that as n increases the random images f_n converge towards a texture-like image. Our purpose is to rigorously define this limit random texture and determine an efficient synthesis algorithm.

2.2 Definition of the *Asymptotic Discrete Spot Noise*

In order to define an interesting limit to the *DSN* sequence we need to normalize the images f_n. As f_n is the sum of n i.i.d. random images, the normalization is given by the central limit theorem. This limit will be called the *asymptotic discrete spot noise (ADSN)* associated with h.

Let X be a uniform RV on Ω and let $H(x) = h(x - X)$. A direct computation shows that the expected value of H is $\mathbb{E}(H) = m\mathbf{1}$ where m denotes the arithmetic mean of h and $\mathbf{1}$ is the image whose components are all equal to 1. Similarly the covariance of the random image H is shown to be equal to the autocorrelation of h, that is for all $(x, y) \in \Omega^2$

$$\mathrm{Cov}\left(H(x), H(y)\right) = C_h(x, y),$$

where

$$C_h(x, y) = \frac{1}{MN}\sum_{u\in\Omega}\left(h(x-u) - m\right)\left(h(y-u) - m\right). \tag{1}$$

The central limit theorem ensures that the random sequence $n^{-1/2}(f_n - nm\mathbf{1})$ converges in distribution towards the MN-dimensional normal distribution $\mathcal{N}(0, C_h)$, yielding the following definition.

Definition 1. *(Asymptotic discrete spot noise)*
With the above notations, the asymptotic discrete spot noise (ADSN) associated with h *is the normal distribution* $\mathcal{N}(0, C_h)$.

2.3 Simulation of the *ADSN*

It is well known that applying a spatial filter to a noise image results in synthesizing a stochastic texture, the characteristic features of which are inherited from the filter and from the original noise [21, 22]. In this section we show that the *ADSN* associated with h can be simulated as a convolution product between a normalized zero-mean copy of h and a Gaussian white noise. We recall that the convolution of two (periodic) images f, g of $\mathbb{R}^{M\times N}$ is defined by

$$(f * g)(x) = \sum_{u\in\Omega} f(x-u)g(u),\ x \in \Omega.$$

Theorem 2. *(Simulation of* ADSN*)*
Let $Y \in \mathbb{R}^{M\times N}$ be a Gaussian white noise, that is, a random image whose components are i.d.d. with distribution $\mathcal{N}(0,1)$. Let h be an image and m be its arithmetic mean. Then the random image

$$\frac{1}{\sqrt{MN}}(h - m\mathbf{1}) * Y \tag{2}$$

is the ADSN *associated with h.*

Proof. Denote

$$\tilde{h} := \frac{1}{\sqrt{MN}}(h - m\mathbf{1})$$

and $Z := \tilde{h} * Y$ the random image defined by Equation (2). Since the convolution product is a linear operator, Z is Gaussian and $\mathbb{E}(Z) = \tilde{h} * \mathbb{E}(Y) = 0$. Besides, for all $(x, y) \in \Omega$,

$$\begin{aligned}
\mathrm{Cov}\,(Z(x), Z(y)) &= \mathbb{E}\,(Z(x)Z(y)) \\
&= \mathbb{E}\left[\left(\sum_{u\in\Omega} \tilde{h}(x-u)Y(u)\right)\left(\sum_{v\in\Omega} \tilde{h}(y-v)Y(v)\right)\right] \\
&= \sum_{u\in\Omega} \tilde{h}(x-u)\tilde{h}(y-u),
\end{aligned}$$

since $\mathbb{E}\,(Y(u)Y(v)) = 1$ if $u = v$ and 0 otherwise. Using Equation (1) and the definition of $\tilde{h}$, we obtain $\mathrm{Cov}\,(Z(x), Z(y)) = C_h(x, y)$. Hence Z is Gaussian with distribution $\mathcal{N}(0, C_h)$. □

3 Random Phase Noise

In this section we analyze a stochastic process, the *random phase noise (RPN)* which was used by van Wijk and his co-workers as a technique to synthesize stationary textures [12, 23]. Using a random phase to obtain a texture from a given Fourier spectrum was first evoked by Lewis in [24].

The *RPN* associated with a discrete image h is a random real image that has the same Fourier modulus as h but has a random phase. We first define a uniform random phase, which is a uniform random image constrained to be the phase of a real-valued image.

Definition 3. *(Uniform random phase)*
We say that a random image $\theta \in \mathbb{R}^{M\times N}$ *is a* uniform random phase *if*

1. θ *is odd:* $\forall x \in \Omega$, $\theta(-x) = -\theta(x)$,
2. *each component* $\theta(x)$ *is either uniform on the interval* $]-\pi, \pi]$ *if* $x \notin \left\{(0,0), \left(\frac{M}{2}, 0\right), \left(0, \frac{N}{2}\right), \left(\frac{M}{2}, \frac{N}{2}\right)\right\}$, *or uniform on the set* $\{0, \pi\}$ *otherwise,*
3. *for every subset* $\mathcal{S}$ *of the Fourier domain which does not contain distinct symmetric points, the family of RV* $\{\theta(x) | x \in \mathcal{S}\}$ *is independent.*

Definition 4. *(*Random phase noise*)*
Let $h \in \mathbb{R}^{M\times N}$ *be an image. A random image* Z *is a* random phase noise (RPN) *associated with* h *if there exists a uniform random phase* θ *such that*

$$\hat{Z}(\xi) = \hat{h}(\xi) e^{i\theta(\xi)}, \ \xi \in \Omega.$$

It is equivalent to define *RPN* as the random image Z such that $\hat{Z}(\xi) = \left|\hat{h}(\xi)\right| e^{i\theta(\xi)}$, where θ is a uniform random phase. This is because if ϕ is the phase of a real-valued image and θ is a uniform random phase then the random image $(\theta + \phi) \mod 2\pi$ is also a uniform random phase. One of the assets of this second definition is to emphasize that the *RPN* associated with an image h only depends on the Fourier modulus of this image. However, as developed in Section 5.1, the first definition permits to extend *RPN* to color images.

4 Spectral Representation of *ADSN* and *RPN*

The *ADSN* associated with an image h is a convolution of a normalized zero-mean copy of h with a Gaussian white noise whereas the *RPN* is obtained by multiplying each Fourier coefficient of h by a uniform random phase.

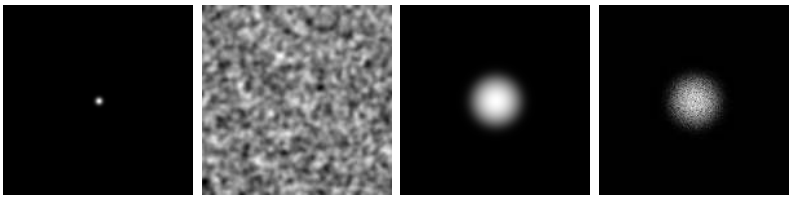
ADSN is easily described in the Fourier domain. A white Gaussian noise image has a uniform random phase, its Fourier modulus is a white Rayleigh noise and its Fourier phase and modulus are independent [25, Chapter 6]. Consequently, the convolution theorem ensures that the phase of the *ADSN* is a uniform random phase whereas its Fourier modulus is the pointwise multiplication of the Fourier modulus of h by a Rayleigh noise.

Thus the phases of *ADSN* and *RPN* are both uniform random phases. However, the Fourier modulus of the two processes have different distributions: The Fourier modulus of the *RPN* is by definition equal to the Fourier modulus of the original image h whereas the Fourier modulus of the *ADSN* is the modulus of h degraded by a pointwise multiplication by a white Rayleigh noise (see Fig. 5). This characteristic of the limit process is clearly visible on Fig. 3 and 4 of the recent paper [20] where the noisy Fourier spectra of some *spot noise* textures are displayed.

To highlight the differences between *ADSN* and *RPN*, consider the effect of both processes on a single oscillation $h(x) = \sin(\lambda x_1 + \mu x_2)$. Because of the phase shift, the *RPN* associated with h is a random uniform translation $\sin(\theta + \lambda x_1 + \mu x_2)$ of h whereas the *ADSN* associated with h is a random uniform translation of h multiplied by a random Rayleigh amplitude R that is $R\sin(\theta + \lambda x_1 + \mu x_2)$. In the same way, if h is the sum of two oscillations then the *RPN* is still a translation of h whereas the *ADSN* may favor one of the two frequencies as illustrated by Fig. 6.

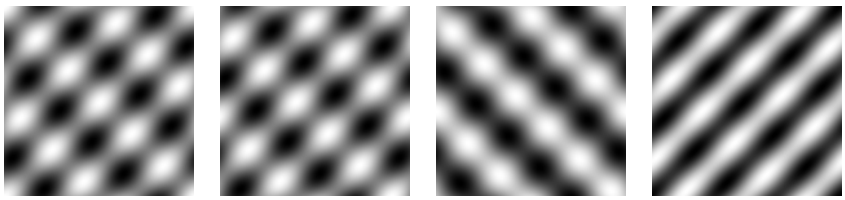
5 Texture Synthesis Algorithms

Theorem 2 and Definition 4 yield two fast synthesis algorithms based on FFT. To preserve the mean of outcomes, the mean value m of the input image is added to the *ADSN* outcomes, while for *RPN*,



(a) Gaussian spot (b) *ADSN* (c) (d)

Figure 5: *ADSN* 5(b) associated with a Gaussian spot 5(a) and their respective Fourier modulus 5(d) and 5(c) represented on logarithmic scale. The modulus of the *ADSN* is the pointwise multiplication of the modulus of h by a white Rayleigh noise



(a) Input (b) *RPN* (c) *ADSN* (d) *ADSN*

Figure 6: Differences between the outcomes of the *RPN* and the *ADSN* associated with the bisinusoidal image6(a). The *RPN* 6(b) is always a translation of the original image 6(a) whereas the *ADSN* randomly favors one of the two frequencies (two different realizations are displayed in 6(c) and 6(d))

the condition $\theta(x) = 0$ is enforced if $x = 0$. Observe that both processes can produce values outside the initial range. These values are usually very few and are simply cut off. An alternative yielding visually similar results is to stretch the histogram of outcomes.

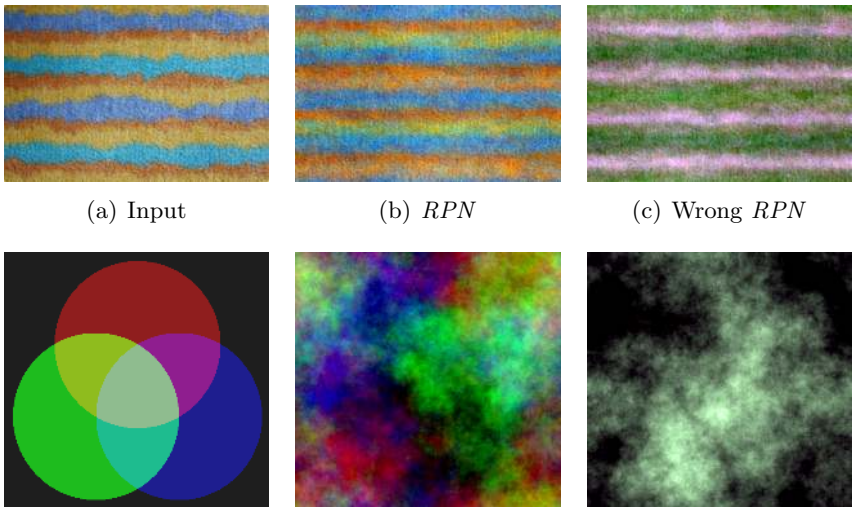
Two important practical points for the simulation of *ADSN* and *RPN* associated with non periodic color images are addressed next, and then we consider the issue of synthesizing *ADSN* and *RPN* textures having larger size than their initial sample.

5.1 Extension to Color Images

Color *RPN*: As we saw in Section 3, the *RPN* process for gray level images is synthesized either by adding a uniform random phase to the phase of the input image or by replacing the phase of this input image by a uniform random phase. The first option allows one to simply use an RGB representation of the image to perform color synthesis. Indeed, adding the same random phase to the original phases of each color channel preserves the phase displacement between channels. This is important as it permits to create new textures without creating false colors, as Fig. 7 shows.

Note that this procedure is much simpler than a classical approach to color texture synthesis relying on a PCA transform of the color space [26, 22], which for some images leads to color mixing.

Color *ADSN*: The mathematical analysis of Section 2 is easily extended to color images using the linearity of the RGB representation. The *ADSN* associated with a color image is thus obtained by convolving each color channel with the same realization of a white Gaussian noise. As for the color *RPN*, the phase of each color channel of the color *ADSN* is random whereas the initial phase displacement between color channels is preserved.



(a) Input (b) *RPN* (c) Wrong *RPN*

(d) Input (e) *RPN* (f) Wrong *RPN*

Figure 7: *RPN* associated with color images: The *RPN* 7(b) and 7(e) of the RGB color images 7(a) and 7(d) are obtained by adding the same random phase to each phase of the three color channels. If one imposes the same random phase to each color channel one obtains images 7(c) and 7(f), the colors of which are not consistent with the original images

5.2 Avoiding Artifacts Due to Non Periodicity

Since both *ADSN* and *RPN* are based on FFT the periodicity of the input sample is a critical requirement. A digital image can always be considered as a periodic image but this results in creating artificial discontinuities at its boundary. In our case it is not possible to avoid this problem using a symmetrization of the image because this would change the features of the output, for example by creating new characteristic directions.

The goal here is to slightly change the input sample h to enforce its periodicity. This is done by replacing h with its periodic component $p = \mathrm{per}(h)$ as introduced in [14]. In the original paper [14], p is defined as the solution of a variational problem. One can in fact show that p is the unique solution of the Poisson problem

$$\begin{cases} \Delta p = \Delta_i h \\ \mathrm{mean}(p) = \mathrm{mean}(h) \end{cases} \tag{3}$$

where Δ is the usual discrete periodic Laplacian and Δ_i is the discrete Laplacian in the interior of the domain. For a periodic image f, these discrete operators are defined by

$$\Delta f(x) = 4f(x) - \sum_{y \in N_x} f(y)$$

and

$$\Delta_i f(x) = |N_x \cap \Omega| \, f(x) - \sum_{y \in N_x \cap \Omega} f(y),$$

where $N_x \subset \mathbb{Z}^2$ denotes the 4-connected neighborhood of x and $|N_x \cap \Omega|$ the number of those neighbors that are in Ω. Note that Δf and $\Delta_i f$ only differ at the boundary of the image domain. As a consequence (3) ensures that p and h have a similar behavior inside the image domain. In particular if h is constant at its boundary we have $p = h$.

(a) Input h (b) p (c) $s + \text{mean}(h)$

(d) $ADSN(h)$ (e) $ADSN(p)$ (f) $s+ADSN(p)$

Figure 8: First row: periodic and smooth component of the input sample $h = p + s$ [14]. The mean of h is added to the smooth component s for visualization. Second row: $ADSN$ associated with the original texture sample h and $ADSN$ associated to its periodic component p. In 8(d) the vertical stripes are due to the change of lighting between the left and the right sides of the input sample 8(a). When using the preprocessed decomposition (8(e)) this artifact due to the non periodicity of the input sample does not appear (results are similar for the RPN algorithm). Note that for rendering purpose one can add back the smooth component s to the $ADSN$ associated with p

In the general case p is computed directly by the classical FFT-based Poisson solver [27] since in the Fourier domain (3) becomes

$$\begin{cases} \left(4 - 2\cos\left(\frac{2\xi_1\pi}{M}\right) - 2\cos\left(\frac{2\xi_2\pi}{N}\right)\right)\hat{p}(\xi) = \widehat{\Delta_i h}(\xi), \ \xi \in \Omega \\ \hat{p}(0) = \hat{h}(0). \end{cases}$$

The definition of p given by (3) is preferable, in the context of this paper, to the original one of [14]. Indeed it enables the direct computation of the Fourier transform of p, which is useful in view of the synthesis algorithm.

Using the periodic component p in place of the initial image h permits to avoid strong artifacts due to the non periodicity of the input samples, as illustrated by Fig. 8. From now on, this preprocessing will be used in all numerical experiments.

Observe that other solutions exist in the literature for finding a "good" periodic representative of h, especially for solving the periodic tiling problem (see [28] for example). Nevertheless the periodic component p is particularly adapted to our problem since it has been defined to eliminate the "cross structure" present in the Fourier transform [14].

5.3 Synthesizing Textures With Arbitrary Sizes

So far both discussed algorithms synthesize output textures which have the same size as the original input sample. However, an important issue in texture synthesis is to synthesize textures with arbitrary large size from a given sample. In this section we propose a practical method which solves this problem for $ADSN$ and RPN textures (see Fig. 10).

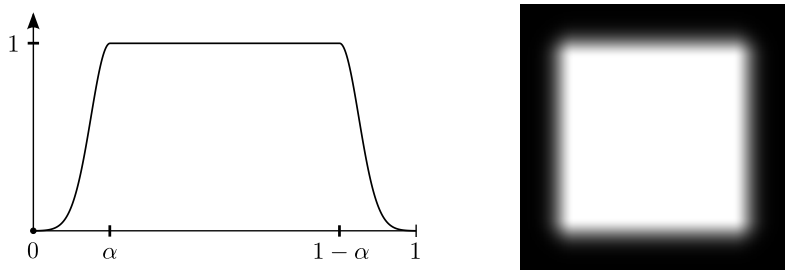


Figure 9: Cross section and gray-level representation of the smooth transition function φ_α used to attenuate the spot along the border of the image. On the intervals $[0, \alpha]$ and $[1-\alpha, 0]$ the function decreases as a Gaussian function with standard deviation $\sigma = \alpha/4$ (here $\alpha = 0.2$). To preserve the variance of the spot, φ_α is normalized so that its L^2-norm equals 1

Given a spot h of size $M_1 \times N_1$ and an output size $M_2 \times N_2$, with $M_2 > N1$ and $N_2 > N_1$, we synthesize *ADSN* (resp. *RPN*) textures of size $M_2 \times N_2$ by computing the *ADSN* (resp. *RPN*) associated with an extended spot $\tilde{h} \in \mathbb{R}^{M_2 \times N_2}$ which represents suitably the original spot $h \in \mathbb{R}^{M_1 \times N_1}$. The extended spot $\tilde{h} \in \mathbb{R}^{M_2 \times N_2}$ is constructed by pasting a normalized copy of the periodic component p of h in the center of an image constant to m. More precisely:

$$\tilde{h}(x) = m + \sqrt{\frac{M_2 N_2}{M_1 N_1}} \, (p(x) - m) \, \mathbb{1}_{R_1}(x), \tag{4}$$

where $\mathbb{1}_{R_1}$ denotes the indicator function of R_1, the centered rectangle of size $M_1 \times N_1$ included in $\Omega_2 = \{0, \dots, M_2 - 1\} \times \{0, \dots, N_2 - 1\}$. As defined by Equation (4), $\tilde{h}$ has the same mean and variance as p, and the autocorrelation of both spots is close for small distances. However $\tilde{h}$ has discontinuities along the border of R_1 which is undesirable since, as illustrated by Fig. 8, those discontinuities can lead to artifacts after randomization.

In order to wear off this brusque transition the inner spot $p - m$ is progressively attenuated at its border. This is done by multiplying $p - m$ by a smooth transition function φ_α. The function φ_α, which is precisely described in Fig. 9, is constant at the center of the domain and decreases smoothly to zero at the border, similarly to a Gaussian function. All experiments in this paper are performed using $\alpha = 0.2$.

The resulting spot extension technique is illustrated in Fig. 10. In this example $\alpha = 0.2$. Experiments show that the value of this parameter is not critical and that for most images $\alpha = 0.2$ seems a good compromise between wave artifact correction and information loss.

Summary of the synthesis method: To conclude this section we summarize the whole procedure for both *ADSN* and *RPN* texture synthesis algorithms. The input of both algorithms is a color input sample h of size $M_1 \times N_1$, the size $M_2 \times N_2$ of the output texture and (optionally) a value for the parameter α involved in the smooth transition function φ_α.

1. Compute the periodic component p of h.
2. Extends p into $\tilde{h}$ using Equation (4) and the pointwise multiplication by the smooth transition function φ_α.
3. - ***ADSN***: Simulate a Gaussian white noise Y and return $Z = m + \frac{1}{\sqrt{M_2 N_2}} \left(\tilde{h} - m \right) * Y$, the convolution being applied to each color channel of $\tilde{h} - m$.

(a) Spot h (b) Extended Spot $\tilde{h}$ (c) $RPN(h)$ (d) $RPN\left(\tilde{h}\right)$

Figure 10: Spot extension technique: the original spot h 10(a) is extended into the spot 10(b) by copying its periodic component p in the center, normalizing its variance (see Equation (4)), and smoothing the transition at the border of the inner frame by multiplying by the function φ_α (here $\alpha = 0.2$). The *RPN* associated to the extended spot is visually similar to the *RPN* associated with the original spot and has an higher size. Results are similar for the *ADSN* algorithm

- ***RPN***: Simulate a random phase θ with $\theta(0) = 0$ and compute Z by adding θ to the phase of each color channel of $\tilde{h}$.

Note that step 1) and step 3) are based on FFT whereas step 2) has linear complexity. Eventually both algorithms have a complexity of $\mathcal{O}\left(M_2 N_2 \log\left(M_2 N_2\right)\right)$. The slight advantage of *RPN* is that it only necessitates the generation of about $\frac{M_2 N_2}{2}$ uniform variables versus the $M_2 N_2$ Gaussian variables necessary for the *ADSN*. Moreover, with *RPN* the Fourier modulus of the original sample is conserved.

6 Numerical Results

6.1 Perceptual Similarity of *ADSN* and *RPN*

Even though the two processes *ADSN* and *RPN* have different Fourier modulus distributions (see Section 4), they produce visually similar results when applied to natural images as shown by Fig. 11. In order to better illustrate this perceptual similarity, we display for each input image the corresponding *ADSN* and *RPN* to which the *same* uniform random phase was imposed. Recall that it was shown in Fig. 6 that perceptual similarity does not hold in the case of images having a sparse Fourier spectrum.

6.2 *RPN* and *ADSN* as Micro-Texture Synthesizers

This section investigates the synthesis of real-world textures using *RPN*. As mentioned above, *ADSN* produces visually similar results.

Fig. 12 and Fig. 11 show that the *RPN* algorithm can be used to synthesize micro-textures similar to a given original sample, whereas Fig. 13 illustrates that it gives poor results with macro-textures. For this kind of texture, resampling algorithms (*e.g.* [15] or [29]) can give impressive results if the parameters (window or patch size, initialization, scanning order, ...) are well-chosen for each input image. However, as said in the introduction, these algorithms are also known for their tendency to sometimes produce erratic results or to excessively use verbatim copying (see Fig. 14), not to mention their computational cost. Recent inpainting algorithms [30, 16] partially solve these issues, but instabilities remain in the case of texture synthesis.

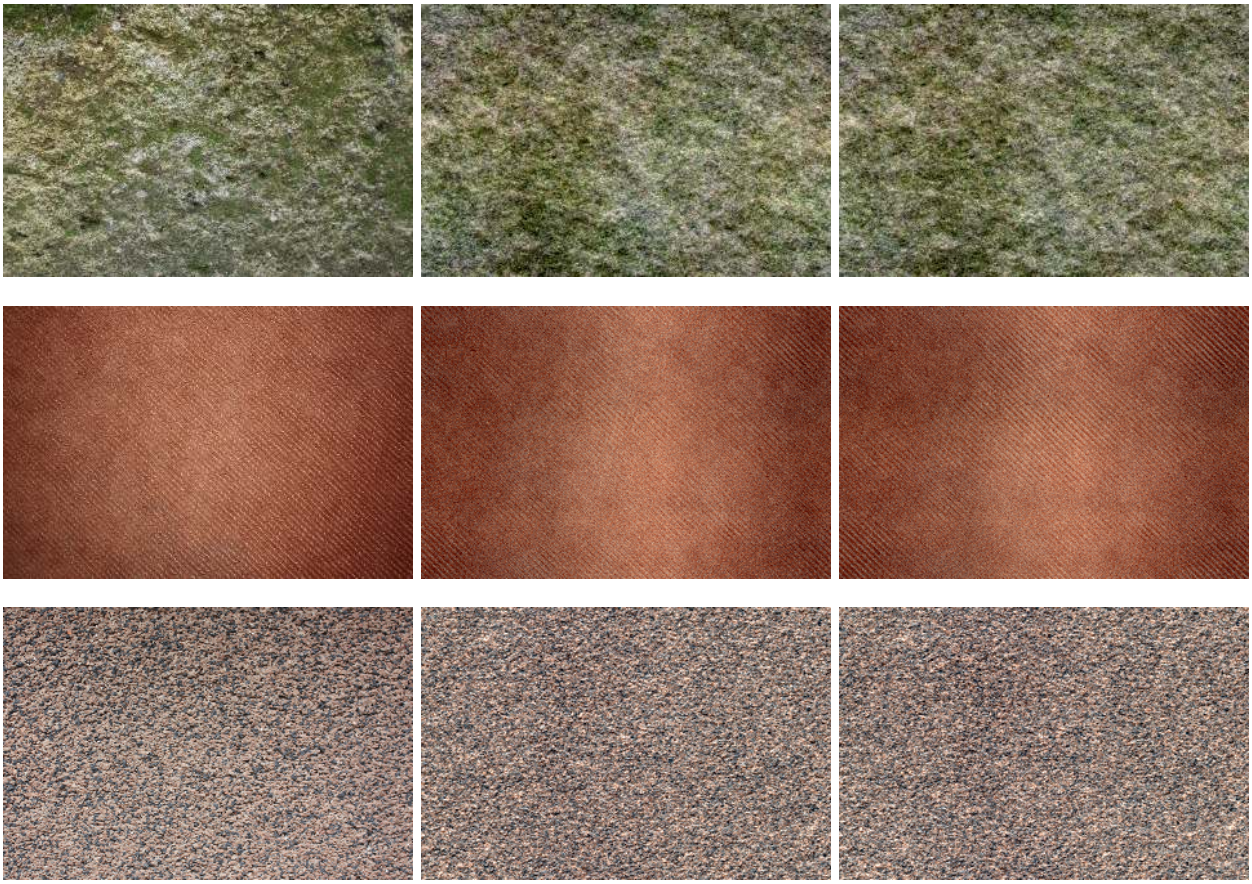


Figure 11: *ADSN* (middle) and *RPN* (right) associated with several input textures (left): stone, carpet, pink concrete. In order to compare the results the same uniform random phase is imposed to both *ADSN* and *RPN*. Observe that there is nearly no perceptual difference between the outcome of both algorithms. This perceptual similarity has been observed for every *ADSN* and *RPN* outcomes associated with natural textures, showing that random phase and random shift textures are perceptually the same class of texture

Figure 12: Examples of well-reproduced textures: *RPN* (right) associated with different input textures (left): wood, indoor wall, stone wall, wallpaper, pinewood, dirt, water. All these textures are satisfyingly reproduced by the *RPN* algorithm, which indicates that they are random phase textures

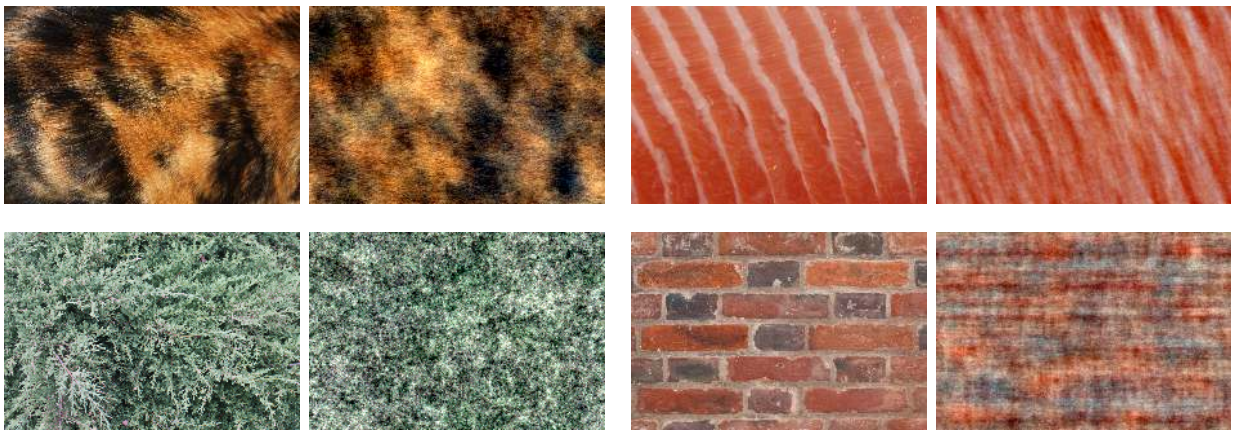


Figure 13: Examples of failures: *RPN* (right) associated with different input textures (left): cat fur, salmon, thuja, bricks. All input textures are, to some extent, not well reproduced by the *RPN* algorithm and therefore are not random phase textures. On the second line are displayed two highly structured textures to which the algorithm is clearly not adapted

(a) Input image (b) $w = 9$ (c) $w = 15$ (d) $w = 21$ (e) *RPN*

Figure 14: Illustrations of the limitations of exemplar-based algorithms. The pinewood texture 14(a) of size 256×256 pixels was used to synthesize twice larger textures using the Efros-Leung algorithm [15] with several values for the window width w (14(b), 14(c), and 14(d)). These algorithms are prone to grow garbage at times, as well as to produce verbatim copies of the input textures. In contrast, as illustrated by 14(e), the *RPN* algorithm is stable

In contrast *RPN* (as well as *ADSN*), despite being limited to the synthesis of specific textures, is parameter-free and non iterative. Besides, it is very fast with a complexity of $\mathcal{O}(MN \log(MN))$. Last but not least, *RPN* (as well as *ADSN*) produces visually stable results: for any given image it always produces perceptually similar results, as illustrated by Fig. 15. As said in the introduction, this property is important in view of an automatic use in the context of computer graphic applications and explain why older and very simple synthesis procedures such as Perlin noise [17], also relying on noise filtering, are still popular today [18, 19, 20].

6.3 A Perceptual Robustness of Phase Invariant Textures

In Section 4 we showed that the *ADSN* associated to a spot can be obtained from its *RPN* by a pointwise product of the Fourier modulus with a Rayleigh noise. Hence the observed visual similarity of the outcomes of the *ADSN* and the *RPN* (see Fig. 11) leads us to claim that the perception of random phase or random shift textures is actually robust to pointwise multiplication of the Fourier modulus by a Rayleigh noise.

One can wonder wether this robustness is also observed for every image. The answer is no and Fig. 16 illustrates that non random phase images are damaged by this multiplication. Thus, the perceptual invariance of random phase textures to a multiplicative noise on their magnitude may be a characteristic of this kind of texture.

7 Conclusion

This article presented a mathematical analysis of *spot noise* texture models and synthesis methods. Two texture perception models stemming from Julesz's theories were recalled. The first one is the random phase model, which leads exactly to the *RPN* algorithm. The second one is the shift invariant texton model. When applied in conjunction with the superposition principle, we have seen that this last model yields a stationary texture model which we called *ADSN*. Experimental evidence has shown that random phase textures and random shift textures generated from the same sample are indistinguishable. Consequently, an unexpected additional perceptual invariance property of random phase textures was uncovered: random phase textures are perceptually invariant under a multiplicative noise on the Fourier modulus. To the best of our knowledge, this surprising fact had never been pointed out.

As for the texture synthesis algorithms, three significant technical points have been developed

Figure 15: Several outcomes of the *RPN* associated with the same input image (top left). *RPN* (as well as *ADSN*) is a visually stable algorithm: indeed even though the output images are locally quite different they are always visually similar

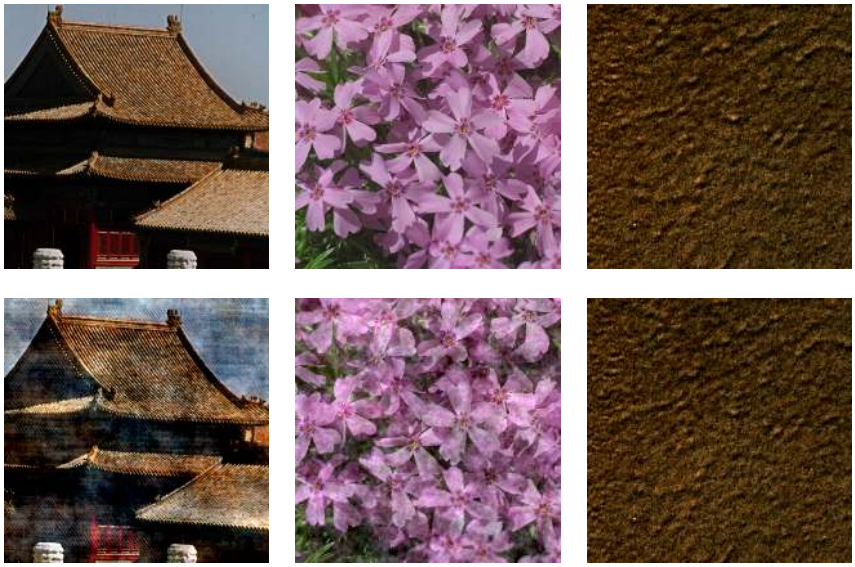


Figure 16: Effect of the pointwise multiplication of the Fourier modulus by a Rayleigh noise. The non random phase images (left and middle) are damaged whereas the random phase texture (right) is perceptually robust to this transformation

permitting the synthesis of textures from real-world texture samples. Numerical results have shown that *ADSN* and *RPN* reproduce satisfyingly well a relatively large range of textures, namely the micro-textures. The algorithms are ideally fast and produce visually stable results, two properties which are crucial for computer graphics applications.

Several new perspectives open up. First, a similar study should be conducted on perceptually based texture synthesis methods relying on wavelet decompositions, following the seminal work of [26]. Second, a strong limitation of the models discussed here is the exclusive use of a linear superposition principle, and it is therefore of interest to investigate asymptotic properties of more elaborate generative texture models involving an occlusion principle or random transparent templates.

References

[1] A. V. Oppenheim and J. S. Lim. The importance of phase in signals. In *Proceedings of the IEEE*, volume 69, pages 529–541, May 1981.

[2] B. Julesz. Visual pattern discrimination. *IRE transactions on information theory*, 8(2):84–92, February 1962.

[3] B. Julesz. A theory of preattentive texture discrimination based on first-order statistics of textons. *Biological Cybernetics*, 41(2):131–138, August 1981.

[4] J. I. Yellott. Implications of triple correlation uniqueness for texture statistics and the julesz conjecture. *J. Opt. Soc. Am. A*, 10:777–793, May 1993.

[5] B. Julesz. Spatial nonlinearities in the instantaneous perception of textures with identical power spectra. *Phil. Trans. R. Soc. London, Ser. B*, 290(1038):83–94, July 1980.

[6] X. Tang and W. K. Stewart. Optical and sonar image classification: wavelet packet transform vs Fourier transform. *Comput. Vis. Image Underst.*, 79(1):25–46, July 2000.

[7] Mingshi Wang and André Knoesen. Rotation- and scale-invariant texture features based on spectral moment invariants. *J. Opt. Soc. Am. A*, 24(9):2550–2557, 2007.

[8] G. Matheron. Schéma booléen séquentiel de partition aléatoire. Technical Report 89, CMM, 1968.

[9] J. Serra, editor. *Image Analysis and Mathematical Morphology*, volume 1. Academic press, London, 1982.

[10] C. Bordenave, Y. Gousseau, and F. Roueff. The dead leaves model: a general tessellation modeling occlusion. *Adv. in Appl. Probab.*, 38(1):31–46, 2006.

[11] W. B. Davenport Jr. and W. L. Root. *An Introduction to the Theory of Random Signals and Noise*. McGraw-Hill, 1958.

[12] J. J. van Wijk. Spot noise texture synthesis for data visualization. In *SIGGRAPH '91*, pages 309–318, New York, NY, USA, 1991. ACM.

[13] W. C. de Leeuw and J. J. van Wijk. Enhanced spot noise for vector field visualization. In *IEEE Visualization*, pages 233–239, 1995.

[14] L. Moisan. Periodic plus smooth image decomposition. preprint, MAP5, Université Paris Descartes, 2009.

[15] A. A. Efros and T. K. Leung. Texture synthesis by non-parametric sampling. In *IEEE International Conference on Computer Vision*, pages 1033–1038, Corfu, Greece, September 1999.

[16] L.-Y. Wei, S. Lefebvre, V. Kwatra, and G. Turk. State of the art in example-based texture synthesis. In *Eurographics 2009, State of the Art Report, EG-STAR*. Eurographics Association, 2009.

[17] K. Perlin. An image synthesizer. In *SIGGRAPH '85*, pages 287–296, New York, NY, USA, 1985. ACM.

[18] R. L. Cook and T. DeRose. Wavelet noise. In *SIGGRAPH '05*, pages 803–811, New York, NY, USA, 2005. ACM.

[19] A. Goldberg, M. Zwicker, and F. Durand. Anisotropic noise. In *SIGGRAPH '08*, pages 1–8, New York, NY, USA, 2008. ACM.

[20] A. Lagae, S. Lefebvre, G. Drettakis, and P. Dutré. Procedural noise using sparse Gabor convolution. *SIGGRAPH '09*, 28(3), August 2009.

[21] L. P. Yaroslavsky. Pseudo-random numbers, evolutionary models in image processing and biology and nonlinear dynamic systems. In *Int. Symposium on Optical Science, Engineering, and Instrumentation*, volume 2824, Denver, CO, USA, July 1996.

[22] K. B. Eom. Synthesis of color textures for multimedia applications. *Multimedia Tools Appl.*, 12(1):81–98, 2000.

[23] D. Holten, J.J. van Wijk, and J.-B. Martens. A perceptually based spectral model for isotropic textures. *ACM Trans. Appl. Percept.*, 3(4):376–398, 2006.

[24] J.-P. Lewis. Texture synthesis for digital painting. In *SIGGRAPH '84*, pages 245–252, New York, NY, USA, 1984. ACM.

[25] A. Papoulis and S. U. Pillai. *Probability, Random Variables and Stochastic Processes*. McGraw-Hill, fourth edition, 2002.

[26] D. J. Heeger and J. R. Bergen. Pyramid-based texture analysis/synthesis. In *SIGGRAPH '95*, pages 229–238, New York, NY, USA, 1995. ACM.

[27] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes 3rd Edition: The Art of Scientific Computing*. Cambridge University Press, New York, NY, USA, 2007.

[28] P. Pérez, M. Gangnet, and A. Blake. Poisson image editing. In *SIGGRAPH '03*, pages 313–318, New York, NY, USA, 2003. ACM.

[29] A. A. Efros and W. T. Freeman. Image quilting for texture synthesis and transfer. In *SIGGRAPH '01*, pages 341–346, New York, NY, USA, 2001. ACM.

[30] P. Pérez, M. Gangnet, and A. Blake. Patchworks: Example-based region tiling for image editing. Technical Report MSR-TR-2004-04, Microsoft Research, 2004.