To appear in ACM TOG 30(4).

Filtering Solid Gabor Noise

Ares Lagae[1,2]
[1]Katholieke Universiteit Leuven

George Drettakis[2]
[2]REVES/INRIA Sophia-Antipolis

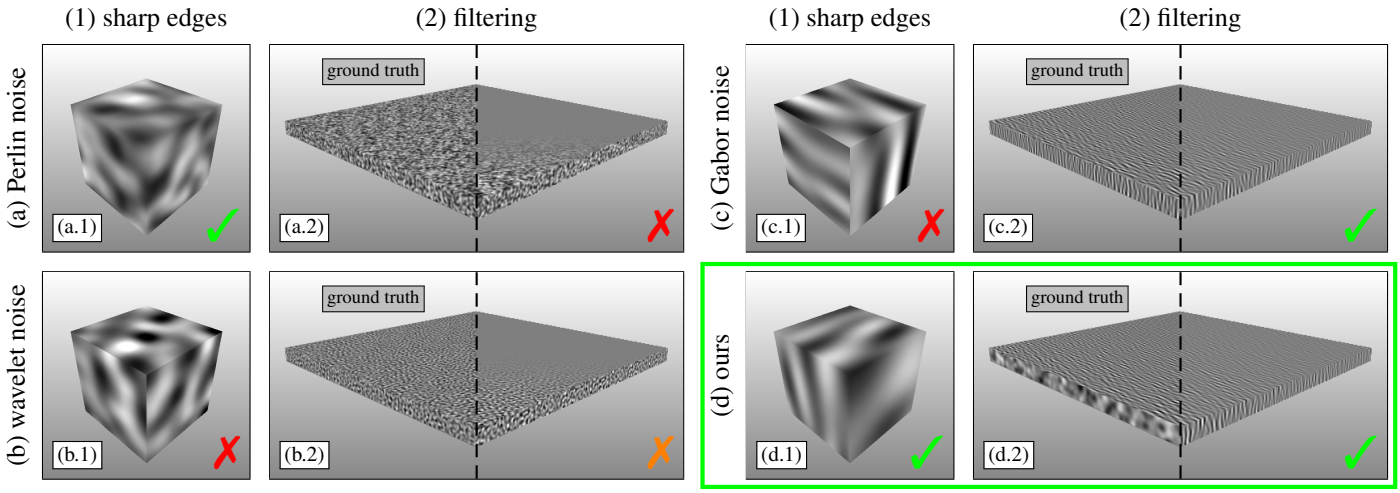

Figure 1: *Existing noise functions either introduce discontinuities of the solid noise at sharp edges, which is the case for wavelet noise (b.1) and Gabor noise (c.1), or result in detail loss when anti-aliased, which is the case for Perlin noise (a.2) and wavelet noise (b.2). We present a new noise function that preserves continuity over sharp edges (d.1) and supports high-quality anti-aliasing (d.2).*

Abstract

Solid noise is a fundamental tool in computer graphics. Surprisingly, no existing noise function supports both high-quality anti-aliasing and continuity across sharp edges. In this paper we show that a slicing approach is required to preserve continuity across sharp edges, and we present a new noise function that supports anisotropic filtering of sliced solid noise. This is made possible by individually filtering the slices of Gabor kernels, which requires the proper treatment of phase. This in turn leads to the introduction of the phase-augmented Gabor kernel and random-phase Gabor noise, our new noise function. We demonstrate that our new noise function supports both high-quality anti-aliasing and continuity across sharp edges, as well as anisotropy.

CR Categories: I.3.3 [Picture/Image Generation]: Antialiasing; I.3.7 [Three-Dimensional Graphics and Realism]: Color, shading, shadowing, and texture

Keywords: anti-aliasing, filtering, procedural texturing, rendering, shading, solid noise, solid texturing, texturing

1 Introduction

Solid texturing [Perlin 1985; Peachy 1985] is a popular method for objects that are sculpted or carved out of a solid material (e.g., a marble statue). To avoid excessive storage requirements, solid or 3D textures are typically procedural, and are often based on procedural solid noise (e.g., Perlin Noise [Perlin 1985]). To achieve high-quality rendering, solid textures must be properly anti-aliased, similarly to traditional textures.

In recent years, there has been renewed interest in the problem of anti-aliasing procedural textures. This has resulted in the introduction of several new noise functions that support filtering [Hart et al. 1999; Cook and DeRose 2005; Goldberg et al. 2008; Lagae et al. 2009]. However, despite these recent advances, no existing noise function supports both high-quality anti-aliasing and continuity across sharp edges. We illustrate this in Fig. 1. Please also refer to the videos in the supplemental material, which illustrate this more clearly. More specifically, Perlin noise [Perlin 1985] results in detail loss when filtered (Fig. 1(a.2)). Wavelet noise [Cook and DeRose 2005] integrates solid noise perpendicularly to the surface of the object, which introduces discontinuities at sharp edges (Fig. 1(b.1)), since the normal changes discontinuously. Gabor noise [Lagae et al. 2009] projects 3D points onto the surface of the object along the surface normal, which, similarly to wavelet noise, does not preserve continuity over sharp edges (Fig. 1(c.1)). For an in-depth discussion and comparison of these noise functions, please refer to the recent survey of Lagae et al. [2010a].

In this paper we show that a slicing approach is required to preserve continuity across sharp edges, and we present a new noise function based on Gabor noise [Lagae et al. 2009] that supports anisotropic filtering of sliced solid noise. We individually filter the slices of Gabor kernels. This requires the proper treatment of the phase of the kernel. We therefore introduce a new Gabor kernel, the phase-augmented Gabor kernel. This, in turn, leads to our new noise function, random-phase Gabor noise. We also discuss how our derivations result in a generalization of the Projection-Slice Theorem as used by Cook and DeRose [2005]. We demonstrate that our new noise function supports both high-quality anti-aliasing and continuity across sharp edges, as well as anisotropy.

2 Motivation

Texturing the surface of an object with filtered solid noise is conceptually a two-step process. In the first step, the 3D solid noise is mapped onto the 2D surface. This is done either by slicing the solid noise, i.e., evaluating the noise at the surface of the object (Fig. 2(a)) [Perlin 1985], or by projecting the solid noise, i.e., integrating the noise perpendicularly to the surface of the object (Fig. 2(b)) [Cook and DeRose 2005]. In the second step, the noise

[1,2]e-mail: ares.lagae@cs.kuleuven.be, george.drettakis@inria.fr

on the 2D surface is filtered. This is typically done using frequency clamping [Norton et al. 1982], a general approach, or using a method-specific approach.

Filtering the noise on the surface using frequency clamping works better if the power spectrum of the noise on the surface is band-pass. The relation between the power spectrum of the 3D solid noise and the 2D noise on the surface is given by the Fourier Slice Theorem [Cook and DeRose 2005], which states that projecting in the spatial domain corresponds to slicing in the frequency domain (Fig. 2(a,c)), and vice versa (Fig. 2(b,d)).

Perlin [1985] uses slicing in his noise, which is typically filtered using frequency clamping. However, filtering introduces an aliasing vs. detail loss trade-off (Fig. 1(a.2)), since the power spectrum of Perlin noise is not band-pass (Fig. 2(c)) [Cook and DeRose 2005].

Cook and DeRose [2005] use projection and frequency clamping in their wavelet noise. However, even though the power spectrum of the noise on the surface is band-pass (Fig. 2(d)), filtering does not fully solve the aliasing vs. detail loss trade-off. This is because frequency clamping works at the level of entire octaves at once (see supplemental material for the exact method we use), and because frequency clamping is limited to isotropic filtering (Fig. 1(b.2)). Moreover, projecting solid noise introduces discontinuities in the noise on the surface at sharp edges (Fig. 1(b.1)), since the normal changes discontinuously (Fig. 2(b)).

Lagae et al. [2009] use projection and a filtering approach specific to Gabor noise in their surface Gabor noise. They project Gabor kernels onto the surface and filter the resulting 2D Gabor kernels analytically. This results in high-quality anisotropic filtering (Fig. 1(c.2)), since the filtering approach works at the level of individual Gabor kernels. However, similarly to wavelet noise, projection introduces discontinuities at sharp edges (Fig. 1(c.1)).

In this paper, we present *solid random-phase Gabor noise*, a new noise function based on Gabor noise [Lagae et al. 2009]. Our new noise uses slicing, which preserves continuity at sharp edges (Fig. 1(d.1)). Since filtering is inherently a 2D operation (i.e., it depends on surface orientation) [Heckbert 1989], we have to explicitly model the slicing of the 3D Gabor kernels to be able to filter the resulting 2D Gabor kernels. This requires the introduction of a new Gabor kernel, the *phase-augmented Gabor kernel*, and a new Gabor noise, *random-phase Gabor noise*. Our new noise uses a filtering approach based on that of Lagae et al. [2009], which supports high-quality anisotropic filtering (Fig. 1(d.2)).

In Sec. 3 we discuss slicing solid Gabor noise; the principal result is Eqn. 6, which shows how to slice a 3D phase-augmented Gabor kernel. In Sec. 4 we discuss filtering sliced solid Gabor noise; the principal result is Eqn. 10, which shows how to filter the resulting 2D phase-augmented Gabor kernel.

3 Slicing Solid Gabor Noise

In this section, we introduce a solid Gabor noise that supports slicing. We first show that the Gabor kernel of Lagae et al. [2009] is not closed under slicing (Sec. 3.1). We consequently introduce the phase-augmented Gabor kernel, a new kernel that is closed under slicing (Sec. 3.2), and random-phase Gabor noise, a new noise using the phase-augmented Gabor kernel (Sec. 3.3). We finally show that solid random-phase Gabor noise supports slicing (Sec. 3.4). We also present a generalization of the Projection-Slice Theorem [Cook and DeRose 2005] (Sec. 3.5).

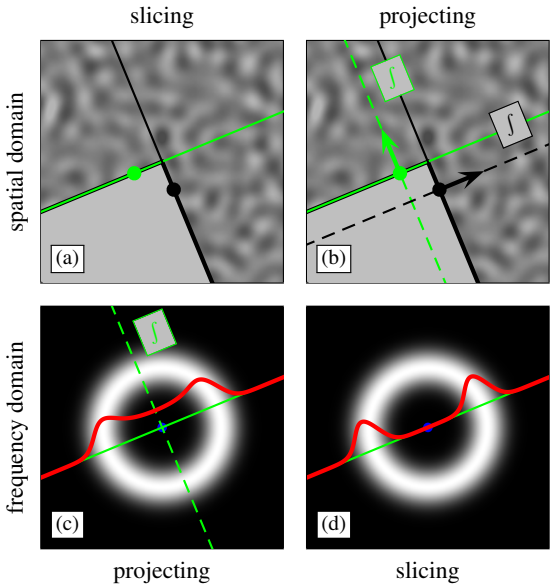

Figure 2: *Slicing and projecting a band-pass noise. Slicing in the spatial domain (solid green line) (a) corresponds to projecting in the frequency domain (integration in the direction of the dashed green line, resulting in the red curve) (c), which does not result in a band-pass power spectrum. Projecting in the spatial domain (integration in the direction of the dashed green line) (b) corresponds to slicing in the frequency domain (solid green line) (d), and introduces discontinuities at sharp edges.*

3.1 Slicing The Gabor Kernel

The Gabor kernel g of Lagae et al. [2009] (Fig. 3(a), image) is defined as

$$g\left(\mathbf{x};a,\omega\right)=e^{-\pi a^2|\mathbf{x}|^2}\cos\left(2\pi\omega\cdot\mathbf{x}\right), \tag{1}$$

the multiplication of a Gaussian (Fig. 3(b), image) and a harmonic (Fig. 3(c), image), where $a\in\mathbb{R}^+$ is the bandwidth and $\omega\in\mathbb{R}^n$ is the frequency. Note that we wrote the kernel in vector notation and extended it to n dimensions. Slicing is denoted as the operator $\mathcal{S}$ subscripted by the hyperplane along which is sliced. We slice the kernel with a hyperplane Π (Fig. 3, solid green line), which results in

$$\mathcal{S}_\Pi\left[g\left(\mathbf{x};a,\omega\right)\right]=\mathcal{S}_\Pi\left[e^{-\pi a^2|\mathbf{x}|^2}\right]\mathcal{S}_\Pi\left[\cos\left(2\pi\mathbf{x}\cdot\omega\right)\right], \tag{2}$$

the product of a sliced Gaussian and a sliced harmonic. This is because $\mathcal{S}$ is a *multiplicative* linear operator (i.e., for every pair of functions f and g and scalar t, $\mathcal{S}\left[fg\right]=\mathcal{S}\left[f\right]\mathcal{S}\left[g\right]$ and $\mathcal{S}\left[tf\right]=t\mathcal{S}\left[f\right]$). The hyperplane Π is specified by its unit normal $\mathbf{n}$ and its signed distance to the origin d (Fig. 3, dashed green line). Quantities in Π, such as the $(n-1)$-dimensional vector $\mathbf{x}'$, are expressed relative to an arbitrary coordinate system, whose origin is the perpendicular projection of the origin of the n-dimensional space. We continue by slicing the n-dimensional Gaussian (Fig. 3(b), image), which results in

$$\mathcal{S}_\Pi\left[e^{-\pi a^2|\mathbf{x}|^2}\right]\left(\mathbf{x}'\right)=e^{-\pi a^2 d^2}e^{-\pi a^2|\mathbf{x}'|^2}, \tag{3}$$

a weighted $(n-1)$-dimensional Gaussian (Fig. 3(b), red curve), where the weight $e^{-\pi a^2 d^2}$ depends on the distance from the hyperplane to the center of the Gaussian. We also slice the n-dimensional harmonic (Fig. 3(c), image), which results in

$$\mathcal{S}_\Pi\left[\cos\left(2\pi\mathbf{x}\cdot\omega\right)\right]\left(\mathbf{x}'\right)=\cos\left(2\pi\mathrm{proj}_\Pi\omega\cdot\mathbf{x}'-2\pi d\mathbf{n}\cdot\omega\right), \tag{4}$$

a *phase-shifted* $(n-1)$-dimensional harmonic (Fig. 3(c), red curve), where $\mathrm{proj}_\Pi\omega$ is the perpendicular projection of ω onto Π, and the

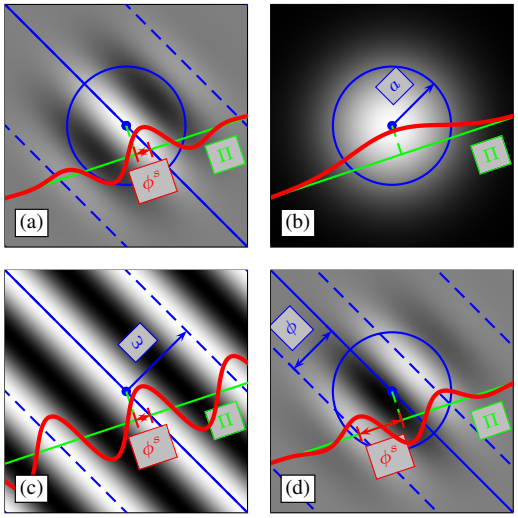

Figure 3: *Slicing the Gabor kernel and the phase-augmented Gabor kernel: (a) Slicing the 2D Gabor kernel; (b) Slicing the 2D Gaussian; (c) Slicing the 2D harmonic; (d) Slicing the 2D phase-augmented Gabor kernel. Note the phase shift ϕ^s introduced by slicing.*

phase shift $2\pi d\mathbf{n}\cdot\omega$ depends on both the distance from Π to the origin and the relative orientation of $\mathbf{n}$ and ω. Note that $\mathrm{proj}_\Pi\omega$ is the component of ω parallel to Π, and $\mathbf{n}\cdot\omega$ is the length of the component of ω perpendicular to Π. We now observe that, because of the phase shift (Fig. 3(a,c), ϕ^s), the product of the sliced Gaussian (Eqn. 3) and the sliced harmonic (Eqn. 4) cannot be expressed as a Gabor kernel (Eqn. 1). In other words, the Gabor kernel of Lagae et al. [2009] is not closed under slicing.

3.2 The Phase-Augmented Gabor Kernel

In order to overcome the problem with the phase shift, we now introduce the phase-augmented Gabor kernel, a new kernel that can accommodate the phase shift introduced by slicing.

Following the derivation of the previous section, we define the phase-augmented Gabor kernel g as

$$g\left(\mathbf{x};a,\omega,\phi\right)=e^{-\pi a^2|\mathbf{x}|^2}\cos\left(2\pi\mathbf{x}\cdot\omega+\phi\right) \tag{5}$$

(Fig. 3(d), image), where $\phi\in[0,2\pi)$ is the phase of the harmonic. Note that the Gabor kernel of Lagae et al. [2009] is a special case of our kernel with zero phase ($\phi=0$), and that, in contrast to the kernel of Lagae et al., Gabor filters used in image processing typically have a phase parameter as well. We now observe that slicing the n-dimensional phase-augmented Gabor kernel g results in

$$\boxed{\mathcal{S}_\Pi\left[g\left(\mathbf{x};a,\omega,\phi\right)\right]\left(\mathbf{x}'\right)=w^s g\left(\mathbf{x}';a,\omega^{\mathbf{s}},\phi^s\right),} \tag{6}$$

a weighted $(n-1)$-dimensional phase-augmented Gabor kernel (Fig. 3(d), red curve), where $w^s=e^{-\pi a^2 d^2}$, $\omega^{\mathbf{s}}=\mathrm{proj}_\Pi\omega$ and $\phi^s=\phi-2\pi d\mathbf{n}\cdot\omega$. Note that all quantities introduced by or affected by slicing are indicated with a superscript ^{s}. In other words, in contrast to the Gabor kernel of Lagae et al. [2009], the phase-augmented Gabor kernel is closed under slicing.

3.3 Random-Phase Gabor Noise

With the phase-augmented Gabor kernel, we can now introduce random-phase Gabor noise, a new noise that supports slicing.

The most important aspect of random-phase Gabor noise is how the phase parameter ϕ is handled. Since ϕ was introduced only to accommodate slicing, it does not make sense to expose it as a noise parameter to the user, nor to fix it to a specific value, since it will be affected by subsequent slicing. We instead assign a random value to ϕ, i.e., the phases $\{\phi_i\}$ of the kernels are distributed according to a uniform distribution in the interval $[0,2\pi)$. The full motivation for this choice is provided in the next subsection. The phase-augmented Gabor kernel with a random phase is called the *random-phase Gabor kernel*. The adjective *random-phase* is used when the phases are random, otherwise the adjective *phase-augmented* is used. We do not use the random weights $\{w_i\}$ used by Lagae et al. [2009], since the noise has a zero mean due to the random phases. We thus define n-dimensional random-phase Gabor noise n as

$$n\left(\mathbf{x};a,\omega\right)=\sum_i g\left(\mathbf{x}-\mathbf{x_i};a,\omega,\phi_i\right), \tag{7}$$

where the random positions $\{\mathbf{x_i}\}$ are distributed according to a n-dimensional Poisson process with mean λ, and $\{\phi_i\}$ are the random phases.

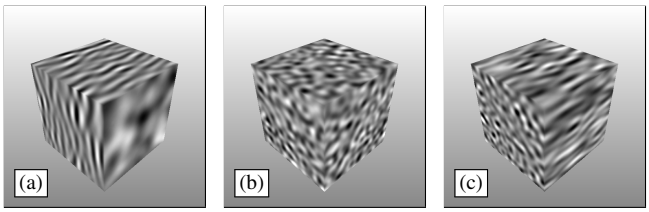

Figure 4: *Different kinds of solid random-phase Gabor noise: (a) anisotropic; (b) isotropic; (c) hybrid anisotropic/isotropic.*

In its most basic form (Eqn. 7), solid random-phase Gabor noise is an anisotropic solid noise parameterized by a (3D) frequency ω and a bandwidth a (Fig. 4(a)). Other kinds of solid noise are obtained depending on how the parameters $\{a_i\}$ and $\{\omega_\mathbf{i}\}$ vary for different kernels. We obtain an isotropic solid noise parameterized by a (1D) radial frequency f and a bandwidth a (Fig. 4(b)) using $\omega_{i,r}=f$, $\omega_{i,\theta}\sim U\left[0,2\pi\right]$ and $\omega_{i,\phi}\sim\cos^{-1}\left\{U\left[-1,1\right]\right\}$. We obtain an interesting hybrid anisotropic/isotropic solid noise (Fig. 4(c)) similarly but with $\omega_{i,\phi}=0$. A greater variety of solid noise patterns could be obtained by compositing solid noise patterns using *widgets*, as in [Lagae et al. 2009], or by using spatially varying solid noise parameters, as in [Lagae et al. 2011].

The statistical properties of a noise, i.e., its variance and power spectrum, determine its aspect, and are needed for several texturing operations, such as color mapping and filtering. We provide a complete set of analytical expressions for random-phase Gabor noise in the supplemental material. Note that our new noise results in a more elegant overall formulation in comparison with the noise of Lagae et al. [2009] since many expressions are simplified.

3.4 Slicing Random-Phase Gabor Noise

In addition to the statistical properties of the 3D solid noise, texturing an object with solid noise also requires the statistical properties of the 2D noise on the surface, i.e., the sliced noise.

We show in the supplemental material that slicing an n-dimensional random-phase Gabor noise n with a hyperplane Π results in

$$\mathcal{S}_\Pi\left[n\left(\mathbf{x};a,\omega\right)\right]\left(\mathbf{x}'\right)=\sqrt{w^s}n\left(\mathbf{x}';a,\omega^{\mathbf{s}}\right), \tag{8}$$

an $(n-1)$-dimensional random-phase Gabor noise, i.e., random-phase Gabor noise is closed under slicing. This means that the statistical properties of a sliced solid random-phase Gabor noise are obtained using the analytical expressions for the statistical properties of a 2D random-phase Gabor noise.

We were able to perform this derivation only when using random

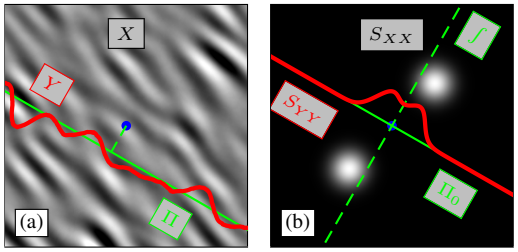

Figure 5: *The Slice-Projection Theorem for Stochastic Processes: The power spectrum of X is S_{XX}; therefore, the power spectrum of Y is S_{YY}.*

phases. Intuitively, this is because the random phases ensure invariance w.r.t. slicing: since the phases of the Gabor kernels are random, the phase-shifted phases of the sliced kernels are random as well. In other words, the random phases allow us to both strengthen and simplify the Gabor noise model of Lagae et al. [2009].

3.5 The Slice-Projection Theorem for Stochastic Processes

By combining insight gained from Eqn. 8 with the definition of noise as a stochastic process [Lagae et al. 2010a, Sec. 2.2], we were able to generalize the Fourier Slice Theorem as used by Cook and DeRose [2005]. More specifically, we generalize from a single noise instance and its Fourier transform to the stochastic process corresponding to the noise and its power spectrum.

Slice-Projection Theorem for Stochastic Processes. *Let X be an n-dimensional stationary stochastic process with power spectrum S_{XX}. Then the $(n-1)$-dimensional stochastic process $Y = \mathcal{S}_\Pi[X]$, which is the slice of X along the hyperplane Π, has the power spectrum $S_{YY} = \mathcal{P}_\Pi[S_{XX}]$, which is the projection of S_{XX} onto Π.*

For example, in Fig. 5, the 2D noise X (a, image) has the power spectrum S_{XX} (b, image); therefore, the 1D noise Y (a, red curve), the slice of X along the line Π (a, green line), has the power spectrum S_{YY} (b, red curve).

Proof. The proof follows from three elements. (i) The fact that R_{YY} (the auto-correlation of Y) is $S_{\Pi_0}[R_{XX}]$, which is the slice of R_{XX} (the auto-correlation of X) along the hyperplane Π_0 through the origin and parallel to Π. This is because X is stationary, and independent of whether or not Π goes through the origin. (ii) The Wiener-Khinchin Theorem, which states that the auto-correlation and the power spectrum of a stochastic process are a Fourier transform pair [Papoulis and Pillai 2002, Ch. 9]. (iii) The Projection-Slice Theorem [Bracewell 1999, Ch. 16]. □

This theorem allows us to validate Eqn. 8, and to explain the aspect of a 2D slice of noise from the power spectrum of the 3D solid noise (e.g., in Fig. 4). Note that, although the Fourier Slice Theorem as used by Cook and DeRose [2005] provides the basic intuition, this theorem is more general and more complete. For example, the proof of this theorem explains why the slicing plane Π does not have to go through the origin, which is not explained by the Fourier Slice Theorem alone. We believe this theorem might have applications beyond noise (e.g., 3D solid texture extrapolation from 2D exemplar slices).

4 Filtering Sliced Solid Gabor Noise

Now that we have a noise that supports slicing, we introduce a filtering approach for our new noise.

Filtering a sliced solid random-phase Gabor noise n with an anisotropic filtering Gaussian $\mathcal{G}_f$ is equivalent to filtering the sliced phase-augmented Gabor kernels,

$$\mathcal{S}_\Pi\left[n\left(\mathbf{x};a,\omega\right)\right]\left(\mathbf{x}'\right) * \mathcal{G}_f\left(\mathbf{x}';\mathbf{0},\Sigma_f\right) = \sum_i w_i^s\, g\left(\mathbf{x}'-\mathbf{x}'_\mathbf{i};a,\omega^\mathbf{s},\phi_i^s\right) * \mathcal{G}_f\left(\mathbf{x}';\mathbf{0},\Sigma_f\right), \quad (9)$$

where $*$ denotes convolution, and $\mathcal{G}\left(\mathbf{x};\mu,\Sigma\right)$ is a Gaussian with mean μ and variance matrix Σ. This is because $\mathcal{S}$ and $*$ are linear operators, and because the phase-augmented Gabor kernel is closed under slicing (Eqn. 6). Note that n is a 3D function, and $\mathcal{S}[n]$, $\mathcal{G}_f$ and g are 2D functions on the plane Π. We thus derive an analytical expression for filtering a 2D phase-augmented Gabor kernel g with an anisotropic filtering Gaussian $\mathcal{G}_f$,

$$\boxed{g\left(\mathbf{x};a,\omega,\phi\right) * \mathcal{G}_f\left(\mathbf{x};\mathbf{0},\Sigma_f\right) \approx w^f\, g\left(\mathbf{x};a^f,\omega^\mathbf{f},\phi\right),} \quad (10)$$

where $w^f = c_{GF}$, $a^f \approx \sqrt{2\pi\sqrt{|\Sigma_{GF}|}}$, $\omega^\mathbf{f} = \mu_{GF}$; $\Sigma_{GF} = \left(\Sigma_G^{-1}+\Sigma_F^{-1}\right)^{-1}$, $\mu_{GF} = \Sigma_{GF}\Sigma_G^{-1}\mu_G$, $c_{GF} = c_F G\left(\mu_\mathbf{G};\mathbf{0},\Sigma_G+\Sigma_F\right)$; $\Sigma_F = \frac{1}{4\pi^2}\Sigma_f^{-1}$, $c_F = \frac{1}{2\pi\sqrt{|\Sigma_f|}}$; and $\Sigma_G = \frac{a^2}{2\pi}I$, $\mu_\mathbf{G} = \omega$. Note that all quantities introduced by or affected by filtering are indicated with a superscript ^{f}. Our filtering expression is a generalization of the one of Lagae et al. [2009, Eqn. 22, Eqn. 23]. Note that the approximate nature of Eqn. 10 (and Eqn. 11 below) is due to the approximation of the anisotropic filtered Gabor kernel with an isotropic kernel [Lagae et al. 2009, Eqn. 22, Eqn. 23 and Fig. 6], which works very well in practice.

We now have all the pieces in place to provide the expression for our new solid random-phase Gabor noise,

$$\boxed{\mathcal{S}_\Pi\left[n\left(\mathbf{x};a,\omega\right)\right]\left(\mathbf{x}'\right) * \mathcal{G}_f\left(\mathbf{x}';\mathbf{0},\Sigma_f\right) \approx \sum_i w_i^s w_i^f\, g\left(\mathbf{x}'-\mathbf{x}'_\mathbf{i};a^f,\omega^{\mathbf{s},\mathbf{f}},\phi_i^s\right),} \quad (11)$$

where $\omega^{\mathbf{s},\mathbf{f}}$ is obtained by applying Eqn 6 and then Eqn 10 to ω. This means that our new noise is evaluated by summing a contribution for each 3D kernel close to the point of evaluation. This contribution is obtained by slicing the 3D kernel (Eqn. 6), filtering the resulting 2D kernel (Eqn. 10), and evaluating the filtered 2D kernel.

5 Implementation

The implementation of our new solid random-phase Gabor noise follows the general structure of that of sparse convolution noise [Lewis 1989; Lagae et al. 2009]. We provide the annotated source code of the GPU implementation of our new noise in the supplemental material. In this section, we only highlight some important aspects of our implementation.

Coordinate Systems To evaluate our filtered sliced solid Gabor noise (Eqn. 11), a point $\mathbf{x}$, a unit normal $\mathbf{n}$, and a filtering Gaussian $\mathcal{G}_f$ are required. The point $\mathbf{x}$ and normal $\mathbf{n}$ are provided in (solid or 3D) texture space (tex), i.e., the space in which $n\left(\mathbf{x};a,\omega\right)$ is defined. The filtering Gaussian $\mathcal{G}_f$ is provided to Eqn. 11 in tangent space (tan), i.e., the space determined by the 2D plane Π, but is defined in screen space (scr). The 2D plane Π is determined by $\mathbf{x_{tex}}$ and $\mathbf{n_{tex}}$, which, together with two additional orthonormal vectors in Π, $\mathbf{t_{tex}}$ and $\mathbf{b_{tex}}$, determine a coordinate system for tangent space, and the mapping from texture space to tangent space $M_{tex\to tan}$. The point $\mathbf{x_{tex}}$ and normal $\mathbf{n_{tex}}$ are obtained from their counterparts in object space (obj) and a (solid or 3D) texture placement matrix $M_{obj\to tex}$.

Continuity of the Local Coordinate System We note that the local coordinate system formed by $\mathbf{t}$, $\mathbf{b}$ and $\mathbf{n}$ does *not* have to be continuous over the surface. The tangents $\mathbf{t}$ and $\mathbf{b}$ do not affect the final filtered noise value, and therefore they can be chosen arbitrarily. We use an arbitrary normalized vector perpendicular to $\mathbf{n_{tex}}$ for $\mathbf{t_{tex}}$, or the normalized version of $\partial \mathbf{x_{tex}} / \partial x_{scr,x}$, which lies in Π and is needed to compute $M_{scr \to tex}$ anyway.

Definition of the Filtering Gaussian We define the 2D filtering Gaussian $\mathcal{G}_f$ in a manner similar to Heckbert [1989]. The filtering Gaussian is defined by its variance matrix in screen space, $\Sigma_{f,scr}$, which typically is $\sigma_{f,scr}^2 I$ with $\sigma_{f,scr} = 1/2$, and it is propagated to tangent space, in which $\mathcal{G}_f$ should integrate to one, using the local affine approximation of the mapping from screen space to tangent space $M_{scr \to tan}$. The filtering Gaussian in tangent space is thus $\mathcal{G}_{f,tan}\left(\mathbf{x_{tan}}; \mathbf{0}, \Sigma_{f,tan}\right)$ with $\Sigma_{f,tan} = M_{scr \to tan} \Sigma_{f,scr} M_{scr \to tan}^T$. We obtain $M_{scr \to tan}$ on the GPU using the local affine approximation of the mapping of screen space to texture space, $M_{scr \to tex}$, which is the matrix determined by the vectors $\partial \mathbf{x_{tex}} / \partial x_{scr,x}$ and $\partial \mathbf{x_{tex}} / \partial x_{scr,y}$. These are easily obtained using built-in derivative functions (e.g., `dFdx()` and `dFdy()` in GLSL). Note that this definition also supports anisotropic screen-space filtering Gaussians (e.g., for motion blur).

Transforming the Phase-Augmented Gabor Kernel Slicing a 3D Gabor kernel (Eqn. 6) involves a transformation between texture space and tangent space, since the 3D kernel is defined in texture space but $\mathcal{G}_f$ and the 2D sliced kernel are defined in tangent space. We note that transforming a phase-augmented Gabor kernel $g\left(\mathbf{x}; a, \omega, \phi\right)$ with a special orthogonal matrix M (i.e., a rotation) results in the phase-augmented Gabor kernel $g\left(\mathbf{x}'; a, \omega^{\mathbf{t}}, \phi\right)$ with $\mathbf{x}' = M\mathbf{x}$ and $\omega^{\mathbf{t}} = M\omega$.

Avoiding Matrix Inversions when Filtering Filtering a Gabor kernel (Eqn. 10) involves several matrix inversions. We reduce this number by exploiting the structure of the matrices (e.g., the covariance matrices Σ are symmetric and positive definite, and Σ_G is a scaled identity matrix), and by simplifying the matrix expressions (see supplemental material for details).

Performance Our new random-phase solid Gabor noise runs at 43.74 and 28.83 megapixels per second for Fig. 4(a) and (b) (NVIDIA GeForce GTX 580 GPU, 512×512, 98,076 shaded pixels, 446 and 294 FPS). We use a simple scene, since the performance figures are optimization-, implementation- and scene-dependent. Despite the fact that the other noise functions have a reduced feature set, we also provide an indicative performance comparison. Our new noise is $1.8 - 1.4\times$ faster than Gabor surface noise (247 and 209 FPS), $2.4 - 1.6\times$ faster than wavelet noise (184 FPS), but up to an order of magnitude slower than Perlin noise (3050 FPS). Note that, although Perlin noise is faster than our noise, this extra performance cannot make up for the filtering quality: even with 256× stratified super-sampling, some of our "ground-truth" video's still exhibit aliasing (see e.g., supplemental material, Video `fig1_c2.avi`). Also note that the projection step in wavelet noise incurs a large performance penalty.

6 Results, Comparisons and Discussion

In Fig. 1, we compare Perlin noise [Perlin 1985], wavelet noise [Cook and DeRose 2005], surface Gabor noise [Lagae et al. 2009], and our new solid random-phase Gabor noise. We filter Perlin noise and wavelet noise using frequency clamping [Norton et al. 1982] (see supplemental material for details). We obtain ground-truth filtering using 256× stratified super-sampling. Our new noise function is the only one that supports high-quality filtering and does not introduce discontinuities at sharp edges. Please also refer to the videos in the supplemental material, which illustrate this more clearly. Note that we do not compare with Goldberg et al. [2008] since they do not support solid noise.

In Fig. 6, we compare a procedural solid texture generated with Perlin noise, wavelet noise, surface Gabor noise, and our new random-phase solid Gabor noise. We use a single scene (Fig. 6(a), a partial model of the Parthenon), a single texture (Fig. 6(b), a granite-like material), and two viewpoints (Fig. 6(a), red and green viewpoint). We compare the continuity at sharp edges (Fig. 6(c), view from the green viewpoint), and the filtering quality (Fig. 6(d), close-up from the view from the red viewpoint). We use the procedural texture model of Lagae et al. [2010b] (without the optional histogram matching), with parameters derived from exemplars and/or edited manually, and anisotropic noise octaves for the anisotropic examples. We disable lighting in order to avoid the introduction of lighting discontinuities. This comparison confirms the result of the previous one: Only our new noise function both supports high-quality filtering and does not introduce discontinuities at sharp edges. Please also refer to the images in the supplemental material, which illustrate this more clearly.

In Fig. 7, we show several solid procedural textures generated with our new random-phase solid Gabor noise. Our new noise function enables a variety of isotropic and anisotropic solid noise textures. Note the continuity across sharp edges (e.g., in Fig. 7(a), at the sharp edges of the chess pieces), and the high-quality anisotropic filtering (e.g., in Fig. 7(c), at the far end of the chess board). Perlin noise and wavelet noise do not support anisotropy. Surface Gabor noise does support anisotropy, but models a surface texture rather than a solid texture, which has a different appearance. Also note that, in contrast to surface Gabor noise, our noise does not require a continuous vector field over the surface.

Lagae et al. [2009] briefly mention solid Gabor noise as well. However, their solid noise does not support filtering; they simply evaluate 3D Gabor kernels at the surface.

It may seem that it would be possible to slice *zero-phase* rather than *random-phase* Gabor noise, and avoid the use of (random) phases altogether. However, this is not true, since slicing zero-phase Gabor noise introduces phase shifts in the sliced noise as well, and thus requires derivations similar to those in Sec. 3. Moreover, as pointed out in Sec. 3.4, the zero-phase-case is actually more complex than the random-phase case.

7 Conclusion

We have presented a new procedural noise function, random-phase Gabor noise, that supports continuity across sharp edges, high-quality anisotropic filtering, and anisotropy. We have achieved this by augmenting Gabor noise with phase, which allowed us to explicitly model the slicing of the 3D Gabor kernels, which in turn allowed us to filter the resulting 2D Gabor kernels. We believe that Gabor noise should always be used with random phases, since it both strengthens and simplifies the model.

Interesting opportunities for future work include exploring volumetric filtering of solid Gabor noise (e.g., for hyper-texture, when ray marching through solid noise), further exploring anisotropy in the context of solid noise, and designing user interfaces for interacting with anisotropic solid noise.

Acknowledgements

We would like to thank the anonymous reviewers; Carles Bosch, Fredo Durand, Bruno Galerne, Ian Jermyn, Sylvain Lefebvre, Javier Taibo (maya2osg) and Celine Vens; and Autodesk (software donation) and

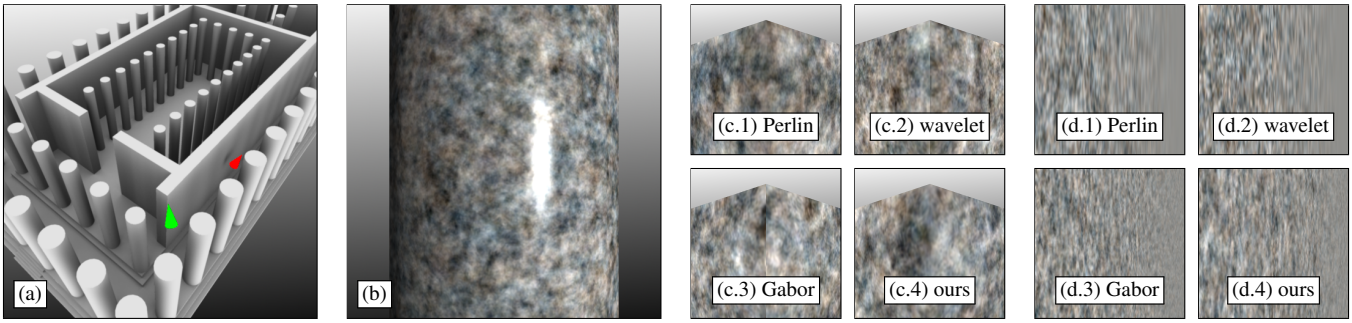

Figure 6: *Comparison of a procedural solid texture generated with Perlin noise, wavelet noise, surface Gabor noise and our new random-phase solid Gabor noise, using the procedural texture model of Lagae et al. [2010b]: (a) Scene, partial model of the Parthenon; (b) Texture, granite-like material; (c) Continuity across sharp edges, green viewpoint (no discontinuity is better); (d) Filtering quality, red viewpoint (more detail towards the right is better). Our new noise function is the only one that supports high-quality filtering and does not introduce discontinuities at sharp edges.*

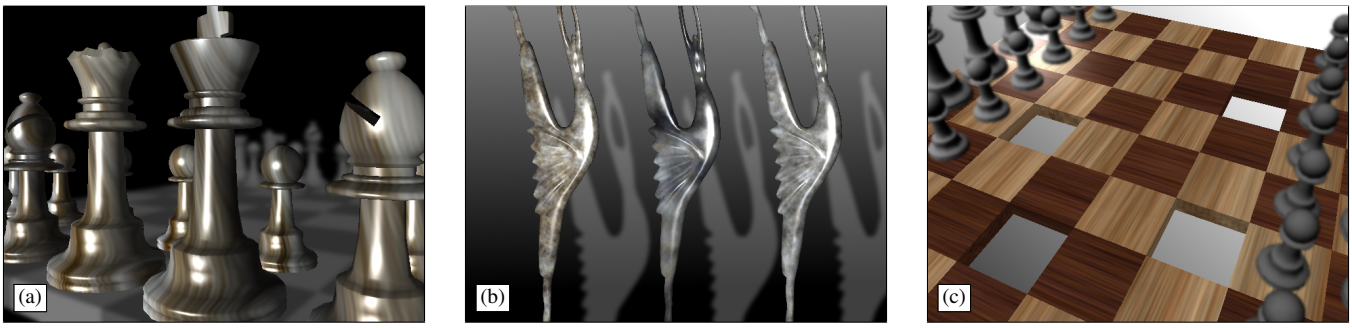

Figure 7: *Solid procedural textures generated with our new random-phase solid Gabor noise, using the procedural texture model of Lagae et al. [2010b]: (a) Granite chess pieces; (b) Granite dancer statuettes; (c) Wooden chess board. Our new noise function enables a variety of isotropic and anisotropic solid noise textures.*

NVIDIA (Academic Partnership Program). Ares Lagae is a Postdoctoral Fellow of the Research Foundation - Flanders (FWO), and acknowledges K.U.Leuven CREA funding (CREA/08/017).

References

BRACEWELL, R. N. 1999. *The Fourier Transform and its Applications*, 3rd ed. McGraw-Hill.

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

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

HART, J. C., CARR, N., KAMEYA, M., TIBBITTS, S. A., AND COLEMAN, T. J. 1999. Antialiased parameterized solid texturing simplified for consumer-level hardware implementation. In *Proc. Graph. hardware*, 45–53.

HECKBERT, P. S. 1989. *Fundamentals of Texture Mapping and Image Warping*. Master's thesis.

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. 2010a. A survey of procedural noise functions. *Comp. Graph. Forum 29*, 8, 2579–2600.

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

LAGAE, A., LEFEBVRE, S., AND DUTRÉ, P. 2011. Improving Gabor noise. *IEEE Trans. Vis. Comp. Graph.*. to appear.

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

NORTON, A., ROCKWOOD, A. P., AND SKOLMOSKI, P. T. 1982. Clamping: A method of antialiasing textured surfaces by bandwidth limiting in object space. In *Comp. Graph. (Proc. ACM SIGGRAPH 82)*, vol. 16, 1–8.

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

PEACHY, D. R. 1985. Solid texturing of complex surfaces. In *Comp. Graph. (Proc. ACM SIGGRAPH 85)*, vol. 19, 279–286.

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