Dynamic Texture Synthesis with Grouplets

Gabriel Peyré

PUO - Colloques MAPMO, Vol. 1, 2008
Editeurs: sera complété par l'éditeur
DOI: (will be inserted later)

DYNAMIC TEXTURE SYNTHESIS WITH GROUPLETS

Gabriel Peyré[1]

Abstract. This paper proposes a new method to synthesize dynamic geometric textures from a given exemplar image. It solves the Navier Stokes equations of fluid dynamics to drive the evolution of a geometric layer and a grouplet coefficients layer. The geometric layer describes the turbulent structures of the texture while the coefficient layer lays a set of strokes along the flows. These strokes corresponds to grouplet atoms, that have a wide range of width and lengths. These grouplets atoms generate a tight frame that is adapted to represent the texture to analyze and synthesize.

Résumé. Ce papier décrit une nouvelle méthode pour synthétiser des textures géométriques dynamiques à partir d'une image donnée en exemple. Les évolutions d'un flot géométrique et des coefficients de grouplets suivent les équation de Navier Stokes. Le flot géométrique dicte la structure turbulente de la texture, tandis que les coefficients de grouplets génèrent des formes le long de ce flot. Les atomes de grouplets ont un support allongé suivant une large gamme de longueurs et de largeurs. Ces atomes forment une trame ajustée qui est adaptée à la représentation compacte des textures à analyser et à synthétiser.

Natural images often contain regions composed of locally oriented structures, such as those depicted in Figure 1. These anisotropic textures are said to be locally parallel since they are composed of approximately parallel oscillations that propagate over the image plane. This paper is focussed on the animation of a static locally parallel texture according to fluid dynamics motions. This creates a turbulent texture whose patterns are similar to those of the exemplar.

[1] CNRS and Ceremade, Université Paris-Dauphine, 75775 Paris Cedex 16 France.

FIGURE 1. *Examples of locally parallel textures.*

1. INTRODUCTION

Anisotropic texture processing. The analysis of the local geometry of textures is studied extensively in computer vision through the computation of local differential estimators, see for instance [13, 15]. Section 2 uses such a local orientation descriptor to build the geometric layer of the grouplet representation.

Geometric decompositions. Texture models based on non-adaptive wavelet decompositions fail to represent in a compact manner geometric singularities. Geometrical decompositions improve over the wavelets for the approximation of edges and textures in images. Local oscillating atoms such as Gabor [24], steerable wavelets [31] or wave-atoms [5] better capture the directionality of textures. These fixed transforms are however not adaptive and are sub-optimal for texture processing. Adaptive representations such as bandlets [20, 26] and grouplets [25] are tuned for a specific image to process. This adaptivity is achieved by computing an optimized geometry that parameterizes the representation. Section 2.2 reviews the grouplet transform that computes the coefficients layer of the grouplet representation.

Static texture synthesis. Texture synthesis is performed by sampling a texture model that constrains the geometric patterns of the images. Fourier and fractals modeling of texture [22, 28] creates cloud-like texture that are suitable to model some natural phenomena. Multiscale models constrain the distribution of wavelet coefficients of textures that characterizes point-wise singularities in textures. Retaining only wavelets marginal coefficients generates textures without geometric patterns [4, 12]. Higher order statistical modeling of wavelets coefficients [30, 35] improves the visual quality of synthesis by capturing geometric singularities. The representation of local features in these methods is however implicit and thus hard to analyze or control.

Computer graphics methods generate new images of a given texture through careful and consistent copying of pixels and patches from an example image [7, 8, 34]. Despite the high visual quality of the results, these approaches do not provide a parametric model for compact charactization of texture classes.

Dynamic Textures. Fluid texture synthesis generates oriented patterns that are warped along a coherent flow. Such a turbulent flow is animated by discretizing the Navier-Stokes equations of fluid motion [10, 33] or using texture synthesis of vector fields [3]. Dynamic textures are rendered by advecting particules [27], and is used to simulate physically plausible phenomena [19, 32]. Wavelet domain texture synthesis

enhances the visual quality of turbulence simulation by adding fine scale details [14]. Dynamic textures can be generated by statistical modeling that captures the space and time homogeneity of natural dynamics [6,18]. Patch recopy methods in computer graphics can be animated according to a vector field [17, 21] and can be mapped on a fluid in motion [2, 16]. Texture mixing animates an image by interpolating between two texture models [1]. A grouplet model is introduced by Peyré [29] to perform geometric texture inpainting and synthesis. Section 4 uses the grouplet model to synthesize dynamic anisotropic textures.

2. Grouplet Texture Analysis

The grouplet framework is a two layers model composed of a geometric layer Γ and a coefficients layer that stores the decomposition of the texture in a wavelet-grouplet tight frame $\mathcal{B}^{\mathrm{w}}(\Gamma)$. The wavelet-grouplet tight frame $\mathcal{B}^{\mathrm{w}}(\Gamma) = \{b^{\Gamma}_{j,\ell,m}\}_{j,\ell,m}$ is composed of elongated atoms. Each $b^{\Gamma}_{j,\ell,m}$ is an anisotropic stroke of width $\sim 2^j$ and length $\sim 2^\ell$, located around a pixel m, and that follows the flow Γ.

The estimation of the model from an exemplar f_0 is performed by computing an optimized flow Γ adapted to f_0 as explained in Section 2.1, and the collection of coefficients $\{\langle f_0, b^{\Gamma}_{j,\ell,m}\rangle\}_{j,\ell,m}$ of f_0 in the wavelet-grouplet family $\mathcal{B}^{\mathrm{w}}(\Gamma)$, as explained in Section 2.3. The application of the model to perform dynamic texture synthesis corresponds to the generation of new geometric and coefficients layer, see Section 4.

2.1. Computation of the Geometric Layer

The geometric layer of our grouplet model is a vector field Γ the describes the local direction of a given input texture $f_0 \in \mathbb{R}^N$ of N pixels. It is then discretized to obtained a set of discrete association field A_ℓ that integrate the flow Γ on a distance of 2^ℓ. These discrete association fields drive the grouplet transform.

Local orientation and direction computation. The local direction of the edges in an image f_0 is captured by the vector $v(x)$ orthogonal to the gradient $\nabla_x f_0$. This vector is computed numerically on the discrete grid $\{0, \ldots, n-1\}^2$ using finite differences. While v is suitable to estimate the direction of step edges in images, it cannot be used directly to estimate the orientation of locally parallel textures such as those depicted on Figure 1. Indeed, the gradient vector vanishes at the top of ridges or at the bottom of valleys. As described in [13,15], this problem is alleviated by averaging locally the tensor field vv^{T},

$$T_{f_0}(x) = (G_\sigma \star (vv^{\mathrm{T}}))(x) = G_\sigma \star \begin{pmatrix} v_1^2 & v_1 v_2 \\ v_1 v_2 & v_2^2 \end{pmatrix}(x),$$

where G_σ is a Gaussian kernel of variance σ^2 and where the convolution is applied on each component of the tensor. This scale σ should match the width of the texture oscillations, and is set to 5 pixels in the numerical experiments.

Each symmetric tensor $T_{f_0}(x)$ is decomposed as a sum of two rank-1 tensors

$$T_{f_0}(x) = \lambda(x)(\Gamma_0(x)\Gamma_0(x)^{\mathrm{T}}) + \lambda^{\perp}(x)(\Gamma_0^{\perp}(x)\Gamma_0^{\perp}(x)^{\mathrm{T}}). \tag{1}$$

where the eigenvalues are $\lambda(x) \geqslant \lambda^{\perp}(x) \geqslant 0$ and $(\Gamma_0(x), \Gamma_0^{\perp}(x))$ are the corresponding orthonormal eigenvectors. The dominant eigenvector $\Gamma_0(x)$ indicates the local direction of the texture.

This orientation field Γ_0 is not directional since $\Gamma_0(x)$ or $-\Gamma_0(x)$ can be used indifferently in the decomposition (1). A directional vector field $\Gamma(x) = \varepsilon(x)\Gamma_0(x)$ with $\varepsilon(x) \in \{-1, +1\}$ is optimized to align the singularity of Γ with integral lines of the flow Γ_0, see [29]. This geometrical flow Γ is the geometric layer that drives the computation of the grouplet coefficients layer.

Discrete association field. A set of non-local discrete association fields $\{A_\ell\}_{\ell=0}^{L-1}$ is computed from the geometrical flow Γ. Each field A_ℓ is then used to compute the grouplet decomposition at a scale 2^ℓ. Each association field A_ℓ groups together two pixels $x \to y = A_\ell(x)$ that are separated by a distance of $\|x - y\| \approx 2^\ell$.

Our construction of the association fields A_ℓ differs from the original one of [25]. The original construction is based on a fixed directional ordering of the grid pixels, that forbids an arbitrary association field. Such a directional ordering is not well suited to process turbulent textures that might exhibit circular patterns or vortices.

The directional vector field Γ is the seed for a linear differential flow

$$\forall\, x,\ \forall\, t > 0, \quad \frac{\mathrm{d}\varphi_x}{\mathrm{d}t}(t) = \Gamma(\varphi_x(t)) \quad \text{and} \quad \varphi_x(0) = x.$$

Each integral curve $\{\varphi_x(t)\}_{t>0}$ follows the vector field Γ. The discrete association field A_ℓ at various scales ℓ is computed by sampling this integral curve at dyadic locations

$$\forall\, x, \quad A_\ell(x) = [\varphi_x(2^\ell)]_{n\times n} \in \{0, \ldots, n-1\}^2,$$

where $[\,\cdot\,]_{n\times n}$ is the rounding operator that projects the points on the sampling grid $\{0, \ldots, n-1\}^2$. Although the association fields are quantized in order to link points on the grid $\{0, \ldots, n-1\}^2$, they follow closely the structure of the input texture f_0.

2.2. **Grouplet Transform**

Once the geometric layer Γ is known, the coefficients layer is computed with a fast redundant grouplet transform. A redundant grouplet transform is introduced by Mallat in [25]. It corresponds to the decomposition of an image on a redundant family of vectors $\mathcal{B}(\Gamma)$. This family is a tight frame of $\mathbb{R}^N$ chosen adaptively to process in an optimized way some given input image of $N = n^2$ pixels. This section describes a new construction of grouplets frame parameterized by a local geometric flow Γ. The grouplet atoms follow closely the flow lines of Γ. These grouplets are thus efficient to represent textures whose geometry is locally parallel to the orientation field Γ.

A grouplet tight frame

$$\mathcal{B}(\Gamma) = \left\{ b_{\ell,m}^{\Gamma} \setminus m \in \{0, \ldots, n-1\}^2, \ell = 0, \ldots, L \right\}$$

is a redundant family of $(L+1)N$ vectors $b^{\Gamma}_{\ell,m} \in \mathbb{R}^N$ parameterized by the local direction flow Γ. Each grouplet vector $b^{\Gamma}_{\ell,m}$ is localized around the pixel $m \in \{0,\dots,n-1\}^2$ and follows the flow Γ on a length of 2^ℓ pixels. This family of vectors is thus efficient to compress a texture whose structures follow closely the direction of Γ. This is in particular the case for the image f_0 that has been used to extract the flow Γ as explained in section 2, but the tight frame $\mathcal{B}(\Gamma)$ can be used to analyze any image f of N pixels.

The forward grouplet transform [25] corresponds to the computation of the decomposition of f onto the vectors of $\mathcal{B}(\Gamma) = \{b_{\ell,m}\}_{\ell,m}$

$$\forall\, \ell, \quad \forall\, m, \quad d_\ell(m) = \langle f, b^{\Gamma}_{\ell,m} \rangle. \tag{2}$$

The grouplet vectors $b^{\Gamma}_{\ell,m}$ are defined implicitly through a decomposition algorithm that computes directly the coefficients $d_\ell(m)$.

This decomposition is computed with a fast algorithm similar to the Haar wavelet transform. At a given scale ℓ, it processes the pixels m sequentially according to an ordering computed from the association field A_ℓ, see [29]. It maintains a filtered residual $\tilde{f}$ initialized to f. Each pixel m is linked to another pixel $\tilde{m} = A_\ell(m)$. The grouplet detail coefficient $d_\ell(m)$ is proportional to the difference $\tilde{f}(m) - \tilde{f}(\tilde{m})$ and the value of the low pass residual $\tilde{f}(\tilde{m})$ is updated to be proportional to the average $(\tilde{f}(m) + \tilde{f}(\tilde{m}))/2$. After processing the final scale $\ell = L-1$, the low pass coefficients d_L are set proportional to the low pass residual $\tilde{f}$. The proportionality factors at each step of the transform are computed to ensure energy conservation even in the situation where several associations are converging from different points $A_\ell(m_1) = A_\ell(m_2)$. The complexity of this fast grouplet transform is $O(N(L+1))$ operations for an image of N pixels and L grouplet scales.

The backward grouplet transform retrieves an image f from the coefficients $\{d_\ell(m)\}_{m,\ell}$ of the decomposition on the tight frame $\mathcal{B}(\Gamma)$. A stable inversion is obtained using the pseudo inverse that implements the reconstruction formula

$$f = \sum_{0 \leqslant \ell \leqslant L} \tau_\ell \sum_m d_\ell(m) b^{\Gamma}_{\ell,m},$$

where $\tau_\ell = 2^{-\ell}$ for $\ell < L$ and $\tau_L = 2^{-L+1}$. The fast backward grouplet transform performs this reconstruction and has a complexity of $O(N(L+1))$ operations, see [29].

2.3. **Wavelet-Grouplet Transform**

A grouplet basis vector $b_{\ell,m}$ has an arbitrary length 2^ℓ but a fixed width of 1 pixel. This is problematic to represent textures with patterns of arbitrary width. Applying the grouplet transform over the wavelet coefficients of an image defines a wavelet-grouplet transform. The corresponding wavelet-grouplet atoms have an arbitrary width 2^j that corresponds to the scale of the wavelet coefficients.

In this wavelet-grouplet setting, one first computes the decomposition of the image f on a set of wavelet vectors

$$\forall\, j = 0, \ldots, J,\, \forall\, p \in \{0, \ldots, n-1\}^2, \quad f_j(p) = \langle f,\, \psi_{j,p} \rangle,$$

where $\psi_{j,p}$ is a wavelet vector scale 2^j and position p. A Laplacian pyramid tight frame is used in the numerical experiments, which has a redundancy of $J+1$, see [24]. This multiscale transform has the advantage of being non-oriented, which leaves the processing of orientation to the grouplet transform alone. The reconstruction from the coefficients in such a tight frame reads

$$f = \sum_j \kappa_j \sum_p f_j(p) \psi_{j,p},$$

where $\kappa_j = 2^{-2j}$ for $j < J$ and $\kappa_J = 2^{-2(J-1)}$. The set of coefficients f_j is re-transformed using the grouplet transform explained in section 2.2.

The projection of each f_j on a grouplet frame $\mathcal{B}(\Gamma)$ generates the following set of coefficients

$$\forall\, j, \ell, m, \quad d_{j,\ell}(m) = \langle f_j,\, b^{\Gamma}_{\ell,m} \rangle = \langle f,\, b^{\Gamma}_{j,\ell,m} \rangle$$

where $b^{\Gamma}_{j,\ell,m}$ is the wavelet-grouplet basis vector defined by

$$b^{\Gamma}_{j,\ell,m} = \sum_p b^{\Gamma}_{\ell,m}(p) \psi_{j,p} \in \mathbb{R}^N.$$

A wavelet-grouplet atom $b^{\Gamma}_{j,\ell,m}$ is an elongated stroke of width $\sim 2^j$ and length $\sim 2^\ell$, that follows the geometrical flow Γ.

This set of vectors $\{b^{\Gamma}_{j,\ell,m}\}_{j,\ell,m}$ is a wavelet-grouplet tight frame $\mathcal{B}^{\mathrm{w}}(\Gamma)$ of $\mathbb{R}^N$ composed of $(J+1)(L+1)N$ vectors. The pseudo-inverse reconstruction from the wavelet-grouplet coefficients reads

$$f = \sum_{j,\ell} \tau_{j,\ell} \sum_m d_{j,\ell}(m) b^{\Gamma}_{j,\ell,m} \tag{3}$$

where $\tau_{j,\ell} = \kappa_j \tau_\ell$.

The wavelet-grouplet forward transform corresponds to the computation of the coefficients $\{d_{j,\ell}(m)\}_{j,\ell,m}$ of a given image f in the tight frame $\mathcal{B}^{\mathrm{w}}(\Gamma)$. This algorithm first computes the f_j using the fast translation invariant wavelet transform, see [24] and then applies the fast grouplet transform, to each of these f_j. The fast backward wavelet-grouplet transform first applies the backward grouplet transform, and then the backward wavelet transform. Both forward and backward algorithms have a complexity of $O((J+1)(L+1)N)$ operations.

3. Texture Synthesis with Grouplets

Grouplet texture synthesis creates a new geometric texture f visually similar to a given input exemplar f_0. This requires a statistical modeling of both the geometric layer and the wavelet-grouplet coefficients layer. The grouplet model for locally parallel textures is based on non parameteric statistics of the wavelet-grouplet coefficients layer. Similarly to the work of Heeger and Bergen [12], we retain only marginal statistics of the decomposition. In contrast to texture modeling over multiscale decompositions [12, 30, 35], our texture model also constrains the geometric layer of the texture.

3.1. Geometric and Coefficients Layers Learning

Grouplet layer learning. The geometric layer of the grouplet model is learned by computing the flow Γ_0 of the exemplar texture f_0 as detailed in Section 2.1.

Coefficients layer learning. The coefficients layer D_0 learned from f_0 is described through a the set of coefficients in the frame $\mathcal{B}^{\mathrm{w}}(\Gamma_0)$

$$\forall\,(j,\ell),\quad D_{j,\ell}(f_0,\Gamma_0)=\{\langle f_0,\, b^{\Gamma_0}_{j,\ell,m}\rangle\}_m,$$

together with the set of pixel values $D^{\mathrm{sp}}(f_0)=\{f_0(x)\}_x$. Each $D_{j,\ell}(f_0,\Gamma_0)$ is computed using the fast transform algorithm.

The layer D_0 is composed of several independent sets (one per scale (j,ℓ) and the pixel values) that empirical marginals of the statistical distribution of coefficients. They are used by the synthesis algorithm to enforce the synthesized texture to share the same empirical marginals as f_0.

3.2. Coefficients Layer Synthesis

Texture synthesis with the grouplet model corresponds to synthesizing both a new geometric layer Γ and a new set of coefficients. This section assumes that the geometric layer Γ is known. The synthesized image is drawn at random from a texture ensemble parameterized by Γ. Section 4 describes how a dynamic layer can be computed using fluid dynamics evolutions.

Grouplet texture ensemble. Once both the geometric layer Γ_0 and the coefficients layers D_0 have been learned, a grouplet texture ensemble $\mathcal{T}(\Gamma,D_0)$ is defined for any orientation field Γ. It is the set of textures having the same wavelet-grouplet and pixels marginals as f_0

$$\begin{aligned}\mathcal{T}(\Gamma,D_0)=\big\{f\in\mathbb{R}^N\setminus\forall\,(j,\ell),\ D_{j,\ell}(f,\Gamma)=D_{j,\ell}(f_0,\Gamma_0)\big\}\\ \cap\big\{f\in\mathbb{R}^N\setminus D^{\mathrm{sp}}(f)=D^{\mathrm{sp}}(f_0)\big\}\,,\end{aligned}$$

This texture ensemble is thus parameterized by the geometric layer Γ of the new texture and by the coefficients layer D_0 of the exemplar. The geometry Γ that defines

the model might be different from the learned geometry Γ_0. This allows one to synthesize new textures with a different geometry.

Marginal constraint. Our texture model enforces synthesized coefficients to have the same coefficients layer as the one of the exemplar D_0. Each marginal $D_{j,\ell,0} = D_{j,\ell}(f_0, \Gamma_0)$ generates a constraint on the texture to synthesize. The synthesis requires to compute the orthogonal projection $\mathcal{P}(d_{j,\ell}, D_{j,\ell,0})$ of wavelet-groulpet coefficients $d_{j,\ell} \in \mathbb{R}^N$ on

$$\left\{ d \in \mathbb{R}^N \setminus \{d(m)\}_m = D_{j,\ell,0} \right\}.$$

The same projection algorithm applies to the pixel values constraints generated by $D^{\mathrm{sp}}(f_0)$.

This projection corresponds to an histogram equalization of $d_{j,\ell}$ with the values of $D_{j,\ell,0}$, see [11]. We denote by $d^r_{j,\ell}(i)$ the values of $D_{j,\ell,0}$ ordered by increasing values, and by $d_{j,\ell}(r(i))$ the ordered values of $d_{j,\ell}$, where r is a permutation of the indices. The projection is

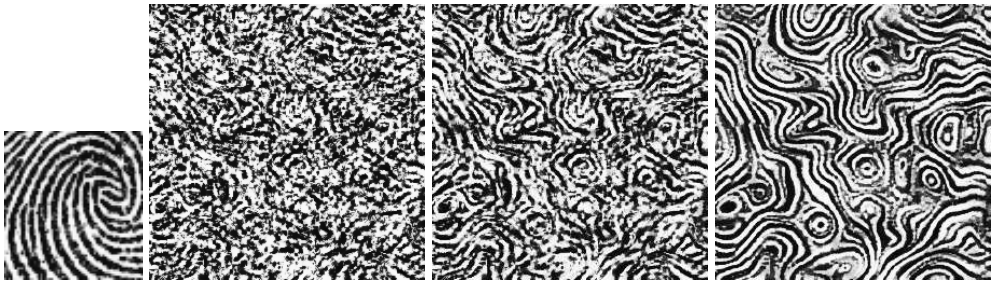
$$\mathcal{P}(d_{j,\ell}, D_{j,\ell,0}) = \tilde{d}_{j,\ell} \quad \text{with} \quad \tilde{d}_{j,\ell}(r(i)) = d^r_{j,\ell}(i). \tag{4}$$

If $d_{j,\ell}$ and $D_{j,\ell,0}$ do not have the same number of coefficients, this formula requires interpolation.

Sampling the grouplet texture ensemble. As explained in the FRAME model [35], the uniform distribution on $\mathcal{T}(\Gamma, D_0)$ has maximal entropy, thus leading to the least synthesis bias. Performing an un-biased synthesis is difficult, and Portilla and Simoncelli [30] replace this uniform sampling by a sampling with high entropy. This is achieved by iteratively projecting a white noise image on the set of constraints that define the texture ensemble.

The grouplet coefficients synthesis algorithm detailed in Table 1 uses a similar iterative projection strategy. The initial image $f^{(0)}$ is a random Gaussian white noise image, and the synthesis algorithm converges to a synthesized image $\mathcal{S}(f^{(0)}\,; \Gamma, D_0) \in \mathcal{T}(\Gamma, D_0)$. The procedure iteratively decomposes the image into the wavelet-grouplet frame $\mathcal{B}^{\mathrm{w}}(\Gamma)$, forces the wavelet-grouplet marginal $D_{j,\ell}(f, \Gamma)$ to match the coefficients layer $D_{j,\ell,0}$ of the exemplar, reconstructs an image from the grouplet coefficients and then enforces consistency of the pixel marginal $D^{\mathrm{sp}}(f)$ with the one of the exemplar.

Figure 2 shows the iterations of the synthesis with a user defined geometric layer Γ. The iterations progressively filter the noise along the geometric flow to create texture patterns. The exemplar image f_0 is twice smaller than the synthesized texture f.



f_0 Iter #1 Iter #3 Iter #20

FIGURE 2. *Iterations of the synthesis algorithm 1.*

Input: orientation flow Γ, initial image $f^{(0)}$, coefficients layer $D_0 = \{D_{j,\ell,0}\}_{j,\ell} \cup D_0^{\mathrm{sp}}$.
Output: synthesized texture f.

(1) *Initialization:* set $i \leftarrow 0$.
(2) *Forward transform:* compute the wavelet-grouplet coefficients $\{d_{j,\ell}^{(i)}(m)\}_{j,\ell,m}$ of $f^{(i)}$ in $\mathcal{B}^{\mathrm{w}}(\Gamma)$.
(3) *Grouplet constraints:* for each (j,ℓ), perform the histogram equalization (4): $\bar{d}_{j,\ell}^{(i)} \leftarrow \mathcal{P}(d_{j,\ell}^{(i)}, D_{j,\ell,0})$.
(4) *Backward transform:* compute $\bar{f}^{(i+1)}$ from the coefficients $\{d_{j,\ell}^{(i)}(m)\}_{j,\ell,m}$.
(5) *Projection on gray-level constraint:* equalize the pixel histogram: $f^{(i+1)} \leftarrow \mathcal{P}(\bar{f}^{(i+1)}, D_0^{\mathrm{sp}})$.
(6) *Stop:* while not converged, set $i \leftarrow i+1$ and go back to 3.

Table 1: Synthesis algorithm to compute $\mathcal{S}(f^{(0)}\,;\Gamma, D_0)$.

4. Dynamic Texture Synthesis

A dynamic fluid texture $\{\tilde{f}_t\}_{t\geqslant 0}$ is created from a single exemplar image f_0 by evolving in time the direction field Γ_t a of the geometric layer according to the incompressible Navier Stokes equation

$$\frac{\partial \Gamma_t}{\partial t} = \mathcal{P}_{\mathrm{inc}}\left(-(\Gamma_t \cdot \nabla)\Gamma_t + \nu \Delta \Gamma_t\right),$$

with Neumann or periodic boundary conditions, where ν is the viscosity of the fluid and Δ is the Laplacian operator. The projector $\tilde{v} = \mathcal{P}_{\mathrm{inc}}(v)$ on the set of divergence free vector fields, computed by solving the Poisson equation for a scalar potential V

$$\tilde{v} = v - \nabla V \quad \text{where} \quad \Delta V = \nabla \cdot v. \tag{5}$$

The synthesized texture $\tilde{f}_t$ for $t > 0$ is obtained by solving an advection equation projected on the grouplet texture model $\mathcal{T}(\Gamma_t, D_0)$ defined in Section (3.2) by the

current geometry Γ_t and the coefficients layer D_0 of f_0

$$\frac{\partial \tilde{f}_t}{\partial t} = \mathcal{S}\left(-(\Gamma_t \cdot \nabla)\tilde{f}_t + \tilde{\nu}\Delta\tilde{f}_t \;;\; \Gamma_t/\|\Gamma_t\|, D_0\right), \tag{6}$$

where $\tilde{\nu}$ is the viscosity of the texture, that might be different from ν. The projection $\mathcal{S}(\cdot\,;\Gamma, D_0)$ on the grouplet model is computed with a few steps of the iterative algorithm detailed in Table 1.

The solution of both (5) and (6) are computed numerically using the implicit solver of Stam [33]. It makes use of the following flow-warping operator, that is defined for scalar function and vector fields

$$\mathcal{W}_\Gamma(f) = \tilde{f} \quad \text{where} \quad \tilde{f}(x) = f(x - \Gamma(x)). \tag{7}$$

Table 2 details the steps of the fluid texture synthesis algorithm[1]. Figure 3 shows two examples of dynamic texture synthesis.

Input: initial texture f_0, time step size $\tau > 0$.
Output: synthesized texture $\tilde{f}_i$ for times $0 \leqslant \tau i \leqslant T$.

(1) *Initialization:* compute Γ_0 and D_0 from f_0, set $i \leftarrow 0$.
(2) *Advect:* compute

$$\Gamma_i^{(1)} = \mathcal{W}_{\tau\Gamma_i}(\Gamma_i) \quad \text{and} \quad \tilde{f}_i^{(1)} = \mathcal{W}_{\tau\Gamma_i}(\tilde{f}_i).$$

(3) *Diffuse:* compute

$$\Gamma_i^{(2)} = \Gamma_i^{(1)} + \tau\Delta\Gamma_i^{(1)} \quad \text{and} \quad \tilde{f}_i^{(2)} = \tilde{f}_i^{(1)} + \tau\Delta\tilde{f}_i^{(1)}.$$

(4) *Project:* compute $\Gamma_{i+1} = \mathcal{P}_{\text{inc}}(\Gamma_i^{(2)})$ and

$$\tilde{f}_{i+1} = \mathcal{S}(\tilde{f}_i^{(2)}\,;\, \Gamma_{i+1}/\|\Gamma_{i+1}\|, D_0)$$

(5) *Stop:* while $\tau i < T$, go back to 2.

Table 2: Grouplet fluid texture synthesis.

5. Conclusion

This paper proposed a new algorithm to perform fluid texture synthesis. The analysis of the exemplar texture is carried over by computing a geometric flow that drives the wavelet-grouplet transform. This defines a texture model using statistics of the wavelet-grouplet coefficients of the exemplar. A dynamic texture is obtained by

[1]The turbulence image is obtained by the resolution of fluid dynamics equations computed by M. Farge et al. [9]. The second texture is obtained from [23].

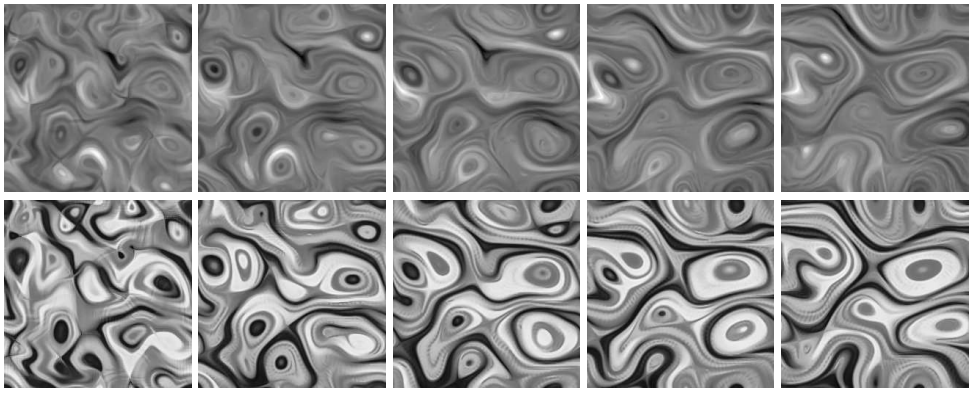


FIGURE 3. *Example of dynamic texture synthesis. The exemplar images used are the first two textures shown on Figure 1.*

synthesizing the geometry using fluid dynamics and by evolving in time the texture inside the texture model. This opens many questions regarding the dependancy of geometry and texture in natural images, which is only partially addressed by our model.

Acknowledgements. I would like to thank Stéphane Mallat and Eero Simoncelli for fruitful discussions, and Marie Farge for providing the turbulence image.

REFERENCES

[1] Z. Bar-Joseph, R. El-Yaniv, D. Lischinski, and M. Werman. Texture mixing and texture movie synthesis using statistical learning. *IEEE Transactions on Visualization and Computer Graphics*, 7(2):120–135, April/June 2001.

[2] A. W. Bargteil, F. Sin, J. E. Michaels, T. G. Goktekin, and J. F. O'Brien. A texture synthesis method for liquid animations. In *Proc. of SCA '06*, pages 345–351, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association.

[3] R. Bridson, J. Houriham, and M. Nordenstam. Curl-noise for procedural fluid flow. *ACM Trans. Graph.*, 26(3):46, 2007.

[4] R.L. Cook and T. DeRose. Wavelet noise. *ACM Trans. Graph.*, 24(3):803–811, 2005.

[5] L. Demanet and L. Ying. Wave atoms and sparsity of oscillatory patterns. *Applied and Computational Harmonic Analysis*, 23(3):368–387, 2007.

[6] G. Doretto, A. Chiuso, Y. N. Wu, and S. Soatto. Dynamic textures. *International Journal of Computer Vision*, 51(2):91–109, February 2003.

[7] A. A. Efros and T. K. Leung. Texture synthesis by non-parametric sampling. In *ICCV '99: Proceedings of the International Conference on Computer Vision-Volume 2*, page 1033. IEEE Computer Society, 1999.

[8] A.A. Efros and W.T. Freeman. Image quilting for texture synthesis and transfer. *Proc. Siggraph '01*, pages 341–346, August 2001.

[9] M. Farge, N. Kevlahan, V. Perrier, and U. Goirand. Wavelets and turbulence. *Proceedings of the IEEE*, 84(4):639–669, 1996.

[10] N. Foster and D. N. Metaxas. Realistic animation of liquids. *Graphical Models and Image Processing*, 58(5):471–483, September 1996.

[11] R. C. Gonzales and R. E. Woods. *Digital Image Processing*. Addison-Wesley Publishing Company, 1993.

[12] D. J. Heeger and J. R. Bergen. Pyramid-Based texture analysis/synthesis. In Robert Cook, editor, *Proc. Siggraph '95*, Annual Conference Series, pages 229–238. ACM SIGGRAPH, Addison Wesley, August 1995.

[13] M. Kass and A. Witkin. Analyzing oriented patterns. *Comput. Vision Graph. Image Process.*, 37(3):362–385, 1987.

[14] T. Kim, N. Thürey, D. James, and M. Gross. Wavelet turbulence for fluid simulation. In *Proc. of SIGGRAPH '08*, pages 1–6, New York, NY, USA, 2008. ACM.

[15] U. Kothe. Edge and junction detection with an improved structure tensor. In *Proc. DAGM03*, pages 25–32, 2003.

[16] V. Kwatra, D. Adalsteinsson, T. Kim, N. Kwatra, M. Carlson, and M. C. Lin. Texturing fluids. *IEEE Trans. Vis. Comput. Graph*, 13(5):939–952, 2007.

[17] V. Kwatra, I. Essa, A. Bobick, and N. Kwatra. Texture optimization for example-based synthesis. *ACM Transactions on Graphics*, 24(3):795–802, July 2005.

[18] V. Kwatra, A. Schödl, I. Essa, G. Turk, and A. Bobick. Graphcut textures: image and video synthesis using graph cuts. *ACM Transactions on Graphics*, 22(3):277–286, July 2003.

[19] A. Lamorlette and N. Foster. Structural modeling of natural flames. In Stephen Spencer, editor, *Proc. of Siggraph'02*, volume 21, 3 of *ACM Transactions on Graphics*, pages 729–735, New York, July 21–25 2002. ACM Press.

[20] E. Le Pennec and S. Mallat. Bandelet Image Approximation and Compression. *SIAM Multiscale Modeling and Simulation*, 4(3):992–1039, 2005.

[21] S. Lefebvre and H. Hoppe. Appearance-space texture synthesis. *ACM Transactions on Graphics*, 25(3):541–548, July 2006.

[22] J-P. Lewis. Texture synthesis for digital painting. *SIGGRAPH Comput. Graph.*, 18(3):245–252, 1984.

[23] L. Liang, C. Liu, Y-Q. Xu, B. Guo, and H-Y. Shum. Real-time texture synthesis by patch-based sampling. *ACM Trans. Graph.*, 20(3):127–150, 2001.

[24] S. Mallat. *A Wavelet Tour of Signal Processing*. Academic Press, San Diego, 1998.

[25] S. Mallat. Geometrical grouplets. *to Appear in Applied and Computational Harmonic Analysis*, 2008.

[26] S. Mallat and G. Peyré. Orthogonal bandlets for geometric image approximation. *Communication on Pure and Applied Mathematics*, 61(9):1173–1212, 2008.

[27] F. Neyret. Advected textures. In D. Breen and M. Lin, editors, *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 147–153, Aire-la-Ville, July 26–27 2003. Eurographics Association.

[28] K. Perlin. An image synthesizer. In *Proc. Siggraph '85*, pages 287–296, New York, NY, USA, 1985. ACM Press.

[29] G. Peyré. Texture synthesis with grouplets. *Preprint*, 2008.

[30] J. Portilla and E. P. Simoncelli. A parametric texture model based on joint statistics of complex wavelet coefficients. *Int. J. Comput. Vision*, 40(1):49–70, 2000.

[31] E. P. Simoncelli, W. T. Freeman, E. H. Adelson, and D. J. Heeger. Shiftable multi-scale transforms. *IEEE Trans Information Theory*, 38(2):587–607, March 1992.

[32] J. Stam and E. Fiume. Depicting Fire and Other Gaseous Phenomena Using Diffusion Processes. In *Proc. of Siggraph'95*, pages 129–136, 1995.

[33] Jos Stam. Stable fluids. In Alyn Rockwood, editor, *Proc. of Siggraph'99*, pages 121–128, N.Y., August 1999. ACM Press.

[34] L-Y. Wei and M. Levoy. Fast texture synthesis using tree-structured vector quantization. In *Proc. Siggraph '00*, pages 479–488. ACM Press/Addison-Wesley Publishing Co., 2000.

[35] S. C. Zhu, Y. Wu, and D. Mumford. Filters, random fields and maximum entropy (FRAME): Towards a unified theory for texture modeling. *Int. J. Comput. Vision*, 27(2):107–126, 1998.