Champs aléatoires gaussiens pour la synthèse de textures

Bruno Galerne
bruno.galerne@univ-orleans.fr

Université d'Orléans

Modélisation :
Modèles déterministes et stochastiques pour le traitement d'images
Master de Mathématiques Approfondies

Plan du cours

Outline

Qu'est-ce qu'une texture ?

- Une définition minimale pour une **texture** est une "image contenant des motifs répétitifs".
- Pas de définition mathématique !
- Selon la nature de la texture la famille de motifs est plus ou moins **aléatoire**.
- Modélisation probabiliste : Une texture est la réalisation d'un **champ aléatoire**.



Quelques exemples de textures

Dans les images naturelles :

- Matériaux : bois, pierre, tissu, sable
- Végétaux : herbe, mousse, feuillage
- Phénomènes naturels : Surface de l'eau, nuages,

Mais aussi en imagerie (structure des matériaux, météorologie,...).

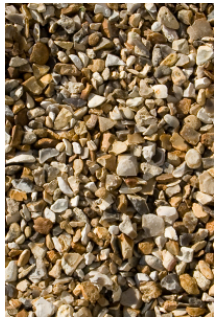
Micro-textures et macro-textures

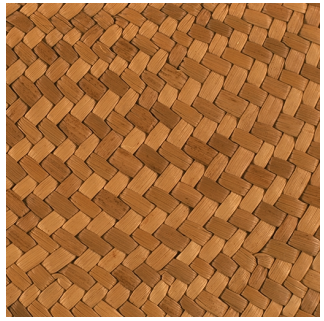
Deux sous-familles principales :

- Les ***micro-textures*** :



- Les ***macro-textures***, constituées d'objets de petites tailles mais discernable individuellement.



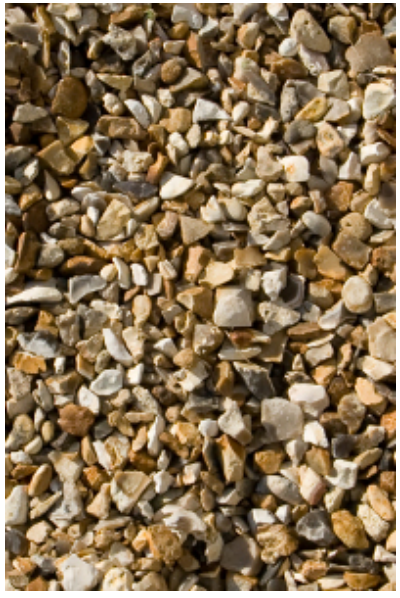


Textures et distance d'observation

Selon la distance d'observation, les même objets peuvent être perçus comme :

- une micro-texture,
- une macro-texture,
- une collection d'objets individuels.

Micro-texture



Macro-texture

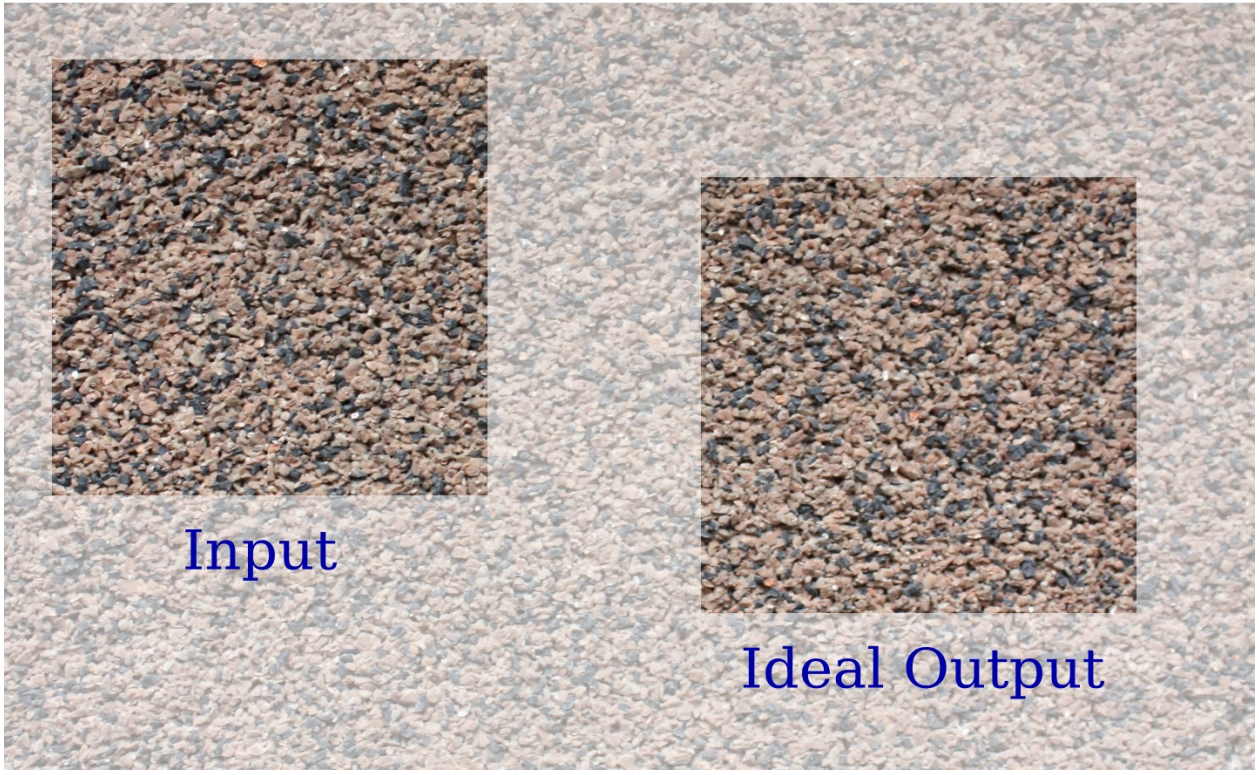
Quelques pierres

Textures en traitement d'images

- Toute scène naturelle comporte des textures.
- Les objets d'une scène se différencient par leur différence de textures.
- Discriminer différentes textures aide à la segmentation d'images.
- Synthèse de texture...

Synthèse de texture

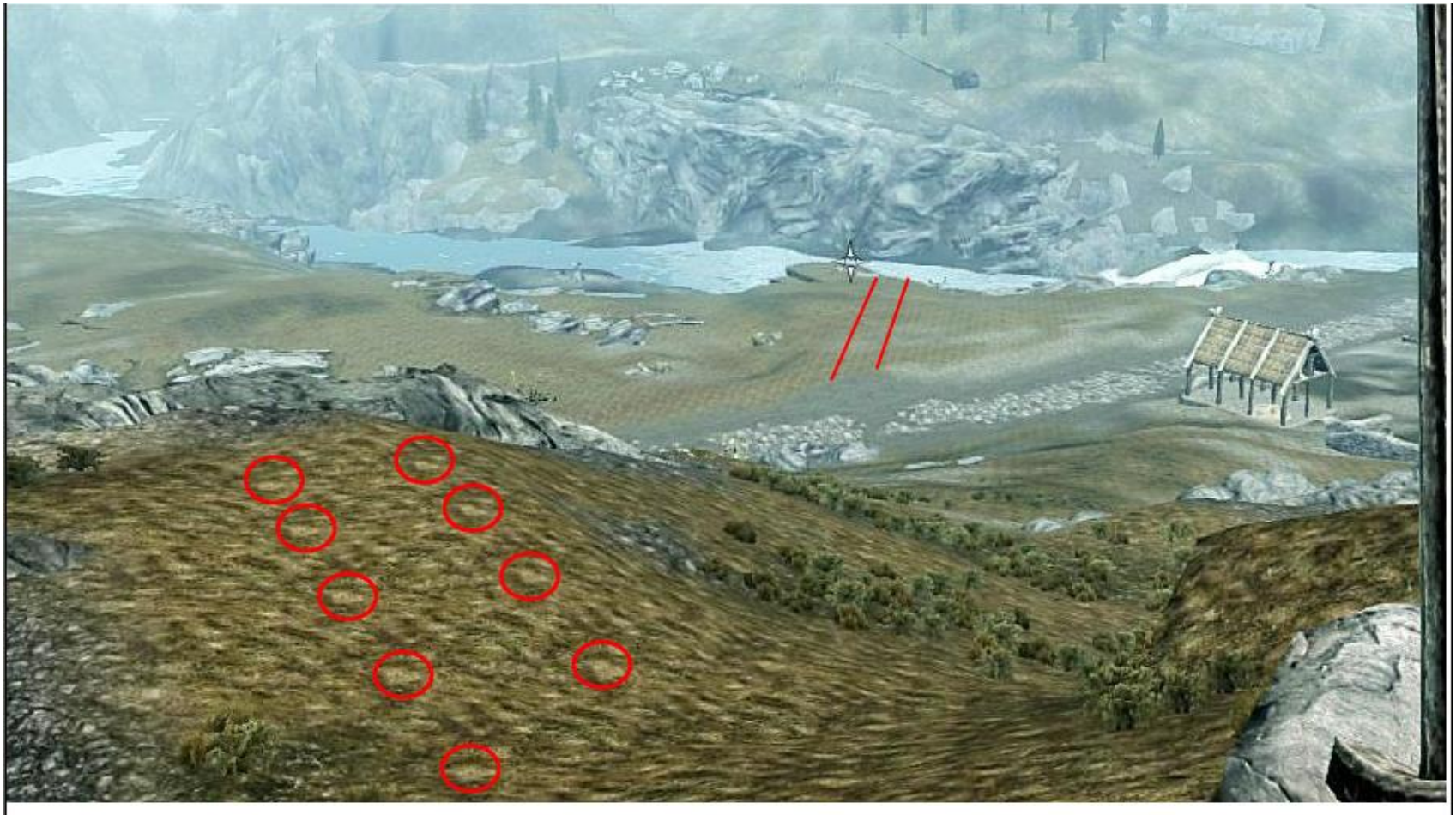
Synthèse de texture : Etant donnée en entrée une image de texture, produire une image de sortie qui soit à la fois visuellement similaire et différente pixels à pixels de l'image d'entrée.



L'image de sortie doit idéalement être perçue comme étant un autre morceau du même matériau que l'image d'entrée.

Synthèse de texture : Motivation

- Le clonage (ou répétition) n'est pas satisfaisant !



Texture synthesis algorithms

Two main kinds of algorithm:

1. Texture synthesis using statistical constraints:

   Algorithm:

   1. Extract some meaningful “statistics” from the input image (e.g. distribution of colors, of Fourier coefficients, of wavelet coefficients,...).
   2. Compute a “random” output image having the same statistics: start from a white noise and alternatively impose the “statistics” of the input.

   Properties:

   + Perceptually stable
   - Generally not good enough for macro-textures

2. Neighborhood-based synthesis algorithms (or “copy-paste” algorithms):

   Algorithm:

   - Compute sequentially an output texture such that each patch of the output corresponds to a patch of the input texture.
   - Many variations have been proposed: scanning orders, grow pixel by pixel or patch by patch, multiscale synthesis, optimization procedure,...

   Properties:

   + Synthesize well macro-textures
   - Can have some speed and stability issue, hard to set parameter...
   - See next course (March, 17) for more details.

Texture synthesis algorithms

Two main kinds of algorithm:

1. Texture synthesis using statistical constraints:

   Algorithm:

   1. Extract some meaningful “statistics” from the input image (e.g. distribution of colors, of Fourier coefficients, of wavelet coefficients,...).
   2. Compute a “random” output image having the same statistics: start from a white noise and alternatively impose the “statistics” of the input.

   Properties:

   + Perceptually stable
   - Generally not good enough for macro-textures

2. Neighborhood-based synthesis algorithms (or “copy-paste” algorithms):

   Algorithm:

   - Compute sequentially an output texture such that each patch of the output corresponds to a patch of the input texture.
   - Many variations have been proposed: scanning orders, grow pixel by pixel or patch by patch, multiscale synthesis, optimization procedure,...

   Properties:

   + Synthesize well macro-textures
   - Can have some speed and stability issue, hard to set parameter...
   - See next course (March, 17) for more details.

Texture synthesis algorithms

Two main kinds of algorithm:

1. Texture synthesis using statistical constraints:

   Algorithm:

   1. Extract some meaningful “statistics” from the input image (e.g. distribution of colors, of Fourier coefficients, of wavelet coefficients,...).
   2. Compute a “random” output image having the same statistics: start from a white noise and alternatively impose the “statistics” of the input.

   Properties:

   + Perceptually stable
   - Generally not good enough for macro-textures

2. Neighborhood-based synthesis algorithms (or “copy-paste” algorithms):

   Algorithm:

   - Compute sequentially an output texture such that each patch of the output corresponds to a patch of the input texture.
   - Many variations have been proposed: scanning orders, grow pixel by pixel or patch by patch, multiscale synthesis, optimization procedure,...

   Properties:

   + Synthesize well macro-textures
   - Can have some speed and stability issue, hard to set parameter...
   - See next course (March, 17) for more details.

Heeger-Bergen algorithm [Heeger and Bergen, 1995]

Statistical constraints:

- Histogram of colors.
- Histogram of wavelet coefficients at each scale.

Algorithm: Alternating projections into the constraints starting from a white noise image.

Efros-Freeman algorithm [Heeger and Bergen, 1995]

Neighborhood-based synthesis algorithm (or “copy-paste” algorithm):

- Copy of rather large patches in a raster scan manner.
- New patches are chosen so that the overlap region matches.
- Dynamic programming is used to compute a seamless boundary in the overlap region.

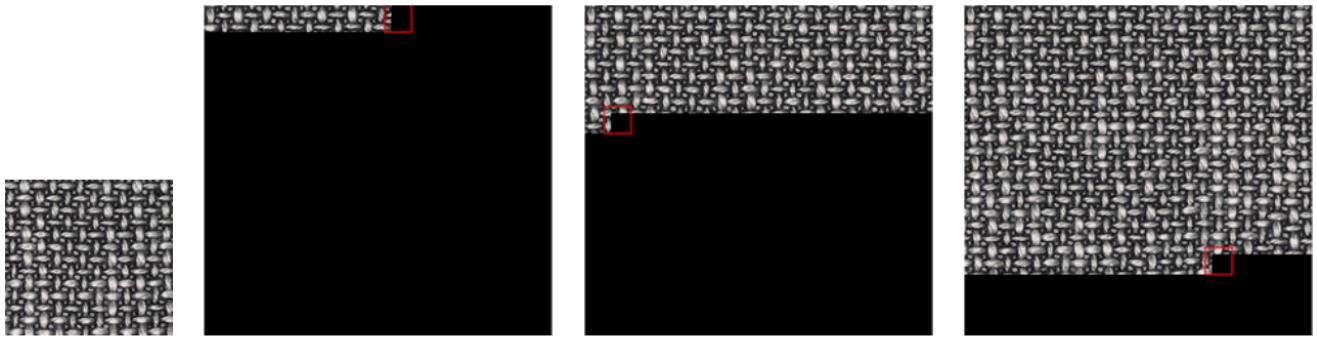
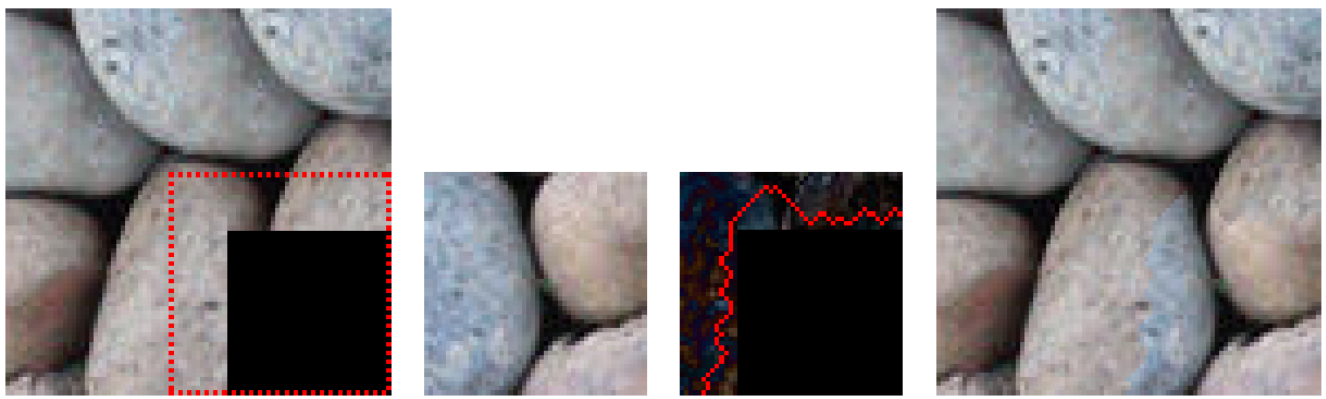


Figure 1: Three different iterations of the synthesis process are shown. At each iteration a patch is being synthesized. This patch is represented by the red square for the three cases.



Synthèse de textures gaussiennes

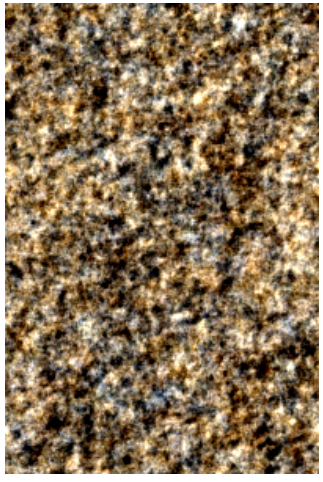
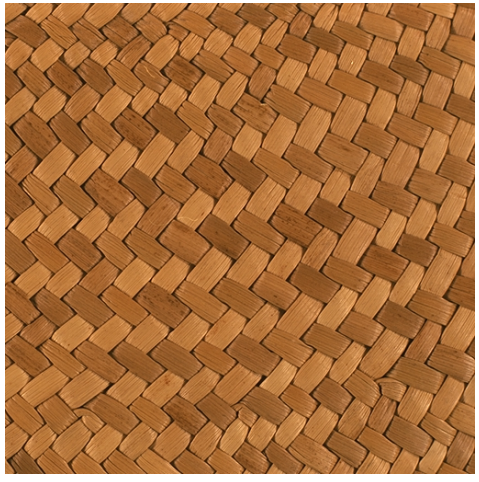
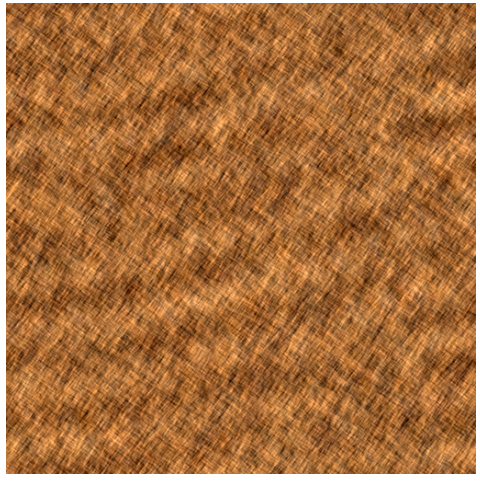
Dans ce cours on va voir comment synthétiser des textures en simulant des champs gaussiens...

Synthèse de textures gaussiennes

- Exemples de synthèses convaincantes pour des micro-textures :

Synthèse de textures gaussiennes

- Exemples d'échecs pour des macro-textures :

Outline

Champs aléatoires : Définition

- Un champ aléatoire sur un domaine discret D est une famille de variables aléatoires $(Y(x))_{x \in D}$.
- Pour les images : Une variable aléatoire par pixel.
- Les variables aléatoires (v.a.) seront à valeurs réelles, mais on pourra aussi considérer des v.a. à valeurs dans $\mathbb{R}^3$ pour les images couleurs, à valeurs dans $\mathbb{C}$ pour les transformée de Fourier, etc.
- Deux cadres :
  - Domaine fini : $D = \Omega_{M,N} = \{0, \ldots, M-1\} \times \{0, \ldots, N-1\}$
  - Domaine infini : $D = \mathbb{Z}^2$

Distribution d'un champ aléatoire

Si le domaine D est fini :

- La distribution d'un champ aléatoire $(Y(x))_{x\in D}$ est simplement la loi de probabilité de Y vu comme vecteur aléatoire.

Si le domaine D est infini :

- On considère la distribution finie-dimensionnelle de Y, à savoir la famille des distributions des vecteurs $(Y(x_1), \ldots, Y(x_k))$ obtenu en évaluant Y en un nombre fini de points.
- Plus précisément, on dit que deux champs $(Y(x))_{x\in D}$ et $(Z(x))_{x\in D}$ ont la même loi si pour tout $k \in \mathbb{N}^*$ et tout $x_1, x_2, \ldots, x_k \in D$,

$$(Y(x_1), Y(x_2), \ldots, Y(x_n)) \stackrel{\mathcal{L}}{=} (Z(x_1), Z(x_2), \ldots, Z(x_n)).$$

On note alors $Y \stackrel{\mathcal{L}}{=} Z$.

Stationnarité

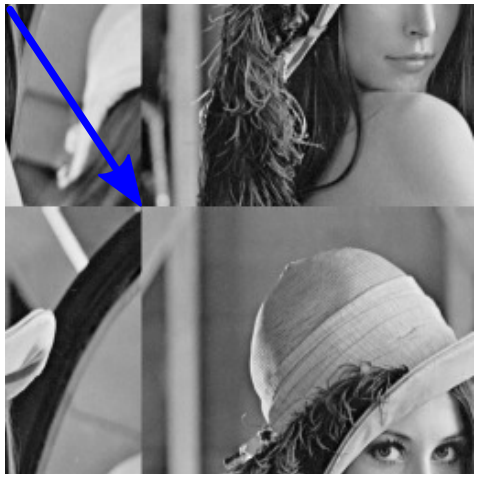
- Un champ aléatoire $(Y(x))_{x\in D}$ est dit **stationnaire** si sa loi est invariante par translation :

$$\forall \tau \in \mathbb{Z}^2, \quad (Y(x))_{x\in D} \overset{\mathcal{L}}{=} (Y(x-\tau))_{x\in D}.$$

- Hypothèse naturelle pour modéliser des textures.
- La translation par le vecteur τ s'interprète différemment selon le cadre.
- Si $D = \mathbb{Z}^2$, translation naturelle sur $\mathbb{Z}^2$.
- Si $D = \Omega_{M,N} = \{0,\dots,M-1\} \times \{0,\dots,N-1\}$ est un rectangle fini, la translation s'effectue par périodicité :

$$u(m-\tau, n-\tau) = u(m-\tau \mod M, n-\tau \mod N).$$

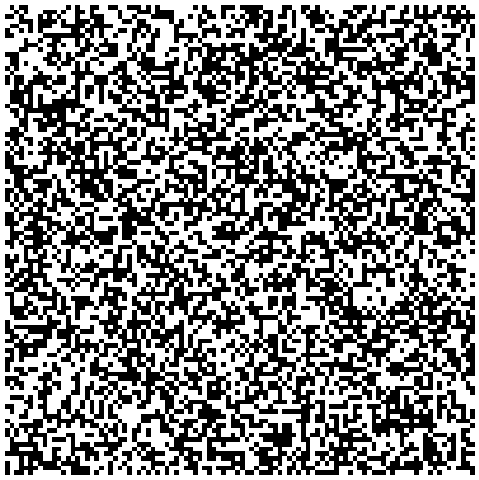
u



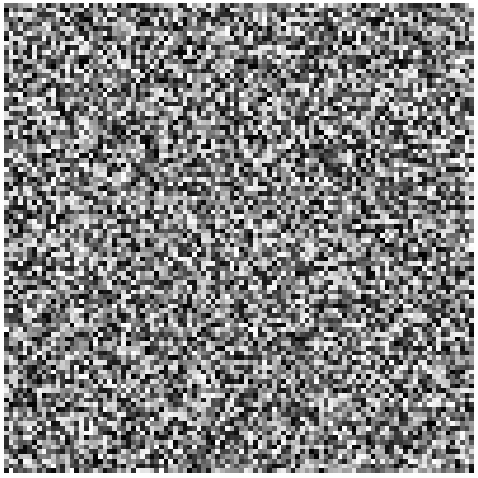
$\theta_\tau u$

Exemples de champs aléatoires

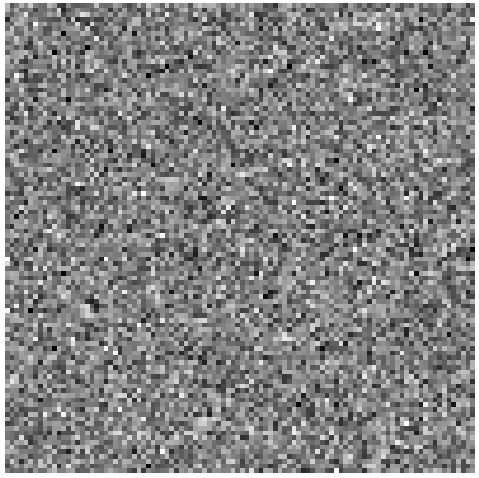
- **Bruit blanc iid** (indépendant identiquement distribué) : Chaque pixel est la réalisation d'une loi fixée sur $\mathbb{R}$ (e.g. Bernouilli, uniforme, gaussienne,...).



Bernouilli



Uniforme



Gaussien ($\mu = 128$, $\sigma = 40$)

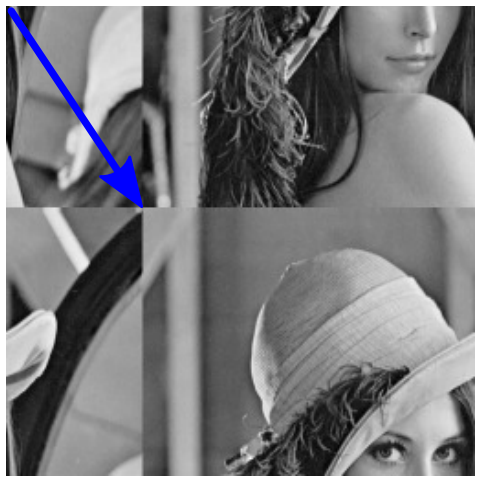
- Les bruits blancs sont stationnaires.
- Le bruit blanc gaussien est le modèle de bruit standard pour le débruitage d'images.

Exemples de champs aléatoires

- Une image h déterministe est un champ aléatoire (non aléatoire...).
- Ce n'est pas un champ stationnaire, sauf si l'image est constante.
- Cadre D fini : On obtient un champ stationnaire en considérant les translations uniforme aléatoire d'une image donnée h :

$$Y(x) = h(x - \tau), \quad \text{où } \tau \text{ est uniforme dans } D = \Omega_{M,N}.$$

h



Une réalisation de Y

- Remarque : Pas d'équivalent sur $\mathbb{Z}^2$!

Espérance

Espérance :

- L'espérance d'un champ aléatoire $(Y(x))_{x \in D}$ est l'image déterministe $(\mathbb{E}(Y(x)))_{x \in D}$.
- Si $(Y(x))_{x \in D}$ est stationnaire, alors $\mathbb{E}(Y)$ est une image constante à $m \in \mathbb{R}$.

Covariance

Covariance :

- La covariance d'un champ aléatoire $(Y(x))_{x\in D}$ est la covariance entre les valeurs des pixels :

$$\begin{array}{rcl} C : D \times D & \to & \mathbb{R} \\ (x, y) & \mapsto & \mathrm{Cov}(Y(x), Y(y)) = \mathbb{E}((Y(x) - \mathbb{E}Y(x))(Y(y) - \mathbb{E}Y(y))) \end{array}$$

- La covariance est symétrique : $C(x, y) = C(y, x)$.
- La covariance est une fonction positive : Pour tout $k \in \mathbb{N}^*$, $x_1, \ldots, x_k \in D$, $\alpha_1, \ldots, \alpha_k \in \mathbb{R}$,

$$\sum_{i,j=1}^{k} \alpha_i \alpha_j C(x_i, x_j) \geqslant 0.$$

- Si $(Y(x))_{x\in D}$ est stationnaire, alors $C(x, y)$ ne dépend que de la différence $x - y$:
  - En dimension 1, C est une matrice **circulante**.
  - En dimension 2, en parcourant les pixels lignes par lignes, on a une matrice circulantes par blocs avec des blocs circulants.

Exercice : Matrices circulantes

Une matrice est circulante si elle est de la forme

$$C = \begin{pmatrix} c_0 & c_1 & c_2 & \dots & c_{n-1} \\ c_{n-1} & c_0 & c_1 & & c_{n-2} \\ c_{n-2} & c_{n-1} & c_0 & & c_{n-3} \\ \vdots & & & \ddots & \vdots \\ c_1 & c_2 & c_3 & \dots & c_0 \end{pmatrix}$$

avec $c \in \mathbb{R}^N$.

Réduction par approche classique :

1. On introduit J la matrice circulante associée au deuxième vecteur de la base canonique ${}^t(0, 1, 0 \dots, 0)$.
   1. Déterminer les puissances de J.
   2. En déduire que J est diagonalisable et préciser une base de diagonalisation dont la première coordonnée de chaque vecteur vaut 1.
2. En déduire que tout matrice circulante est diagonalisable et préciser ses valeurs propres et vecteurs propres en fonction de c.

Exercice : Matrices circulantes

Une matrice est circulante si elle est de la forme

$$C = \begin{pmatrix} c_0 & c_1 & c_2 & \dots & c_{n-1} \\ c_{n-1} & c_0 & c_1 & & c_{n-2} \\ c_{n-2} & c_{n-1} & c_0 & & c_{n-3} \\ \vdots & & & \ddots & \vdots \\ c_1 & c_2 & c_3 & \dots & c_0 \end{pmatrix}$$

avec $c \in \mathbb{R}^N$.

Réduction par lien avec le produit de convolution :

Rappel: Le produit de convolution entre deux vecteurs u et v de $\mathbb{C}^N$ est le vecteur $u * v \in \mathbb{C}^N$ défini pour tout $n \in \Omega_N$ par

$$(u * v)_n = \sum_{m=0}^{N-1} u_m v_{n-m}$$

1. Montrer que l'application $v \mapsto Cv$ est une convolution par rapport à vecteur que l'on précisera.
2. En déduire que C est diagonalisable par la matrice de la transformée de Fourier discrète et préciser les valeurs propres et vecteurs propres de C en fonction de c.

Outline

Champs gaussiens : Définition

Définition

- Un champ aléatoire $(Y(x))_{x\in D}$ est un champ aléatoire gaussien si toute combinaison linéaire finie des ses coordonnées suit une loi normale sur $\mathbb{R}$.
- Cette définition généralise celle des vecteurs gaussiens.

Proposition

- La distribution d'un champ aléatoire gaussien est déterminée par son espérance et sa covariance.

Exemple

- Un bruit blanc gaussien est un champ aléatoire gaussien (car la somme de deux v.a. gaussiennes indépendantes est encore une v.a. gaussienne). Si la loi des pixels est (μ, σ^2), alors l'espérance du bruit blanc est constante à μ et sa covariance est

$$C(x,y) = \begin{cases} \sigma^2 & \text{si } x = y, \\ 0 & \text{sinon.} \end{cases}$$

Vecteurs gaussiens

Faire l'**Exercice 4** de la feuille téléchargeable ici (Merci à Arthur Leclaire) : `https://www.idpoisson.fr/galerne/m2_tours/td_vecteurs_gaussiens.pdf`

Exercice 4. Densité des vecteurs gaussiens

Soit $X \sim \mathcal{N}(m, \Gamma)$ avec $m \in \mathbb{R}^d$ et $\Gamma \in \mathbb{R}^{d\times d}$. Soit $A \in \mathbb{R}^{d\times d}$ telle que $AA^T = \Gamma$.

1. Soit $Z \sim \mathcal{N}(0, I_d)$. Montrer que X et $m + AZ$ ont même loi.

2. Montrer que toute matrice symétrique positive Γ est la covariance d'un vecteur gaussien.

3. Supposons Γ inversible. En utilisant la question 1, montrer que X admet une densité par rapport à la mesure de Lebesgue, et l'expliciter.

4. Supposons Γ non inversible. Montrer qu'alors il existe un hyperplan H de $\mathbb{R}^d$ tel que $X \in H$ p.s.
En déduire que X n'est pas à densité.

Simulation de vecteurs gaussiens

- **Question:** Comment simuler un vecteur gaussien $X \in \mathbb{R}^N$ d'espérance $m \in \mathbb{R}^N$ et de matrice de covariance Γ ?
- Quelles méthodes numériques pour calculer A telle que $AA^T = \Gamma$?

Simulation de vecteurs gaussiens

- **Question:** Comment simuler un vecteur gaussien $X \in \mathbb{R}^N$ d'espérance $m \in \mathbb{R}^N$ et de matrice de covariance Γ ?
- Quelles méthodes numériques pour calculer A telle que $AA^T = \Gamma$?

Théorème central limite vectoriel

- On considère le cas D fini.
- Soit (Y_n) une suite iid de champs aléatoires sur D d'espérance $\mathbb{E}(Y)$ et de covariance C.
- On pose $S_n = \dfrac{Y_1 + \cdots + Y_n}{n}$.

Loi des grands nombres :

- $\dfrac{Y_1 + \cdots + Y_n}{n}$ converge presque sûrement vers $\mathbb{E}(Y)$.

Théorème central limite :

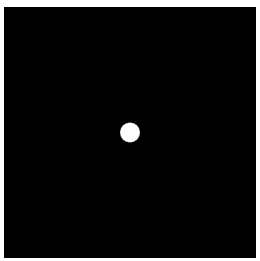
- $\dfrac{Y_1 + \cdots + Y_n - n\mathbb{E}(Y)}{\sqrt{n}}$ converge en loi vers le champ gaussien de moyenne nulle et de covariance C.

Outline

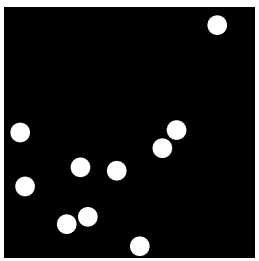
Spot noise discret

- Soit $h \in \mathbb{R}^{M \times N}$ une image discrète appelée *spot*.
- Soit (X_k) une suite i.i.d. de vecteurs de translations aléatoires uniformément distribués dans $\Omega_{M,N}$.
- Le **spot noise discret d'ordre** n **associé à** h est l'image aléatoire

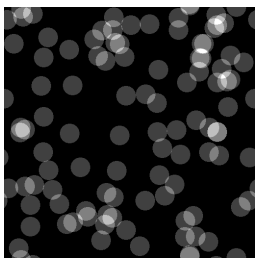
$$f_n(x) = \sum_{k=1}^{n} \theta_{X_k} h(x) = \sum_{k=1}^{n} h(x - X_k).$$



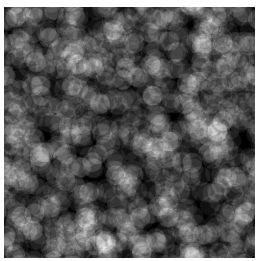
Spot h



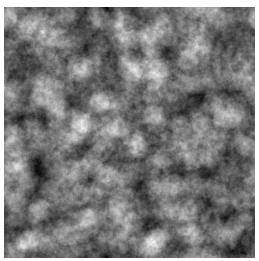
$n = 10$



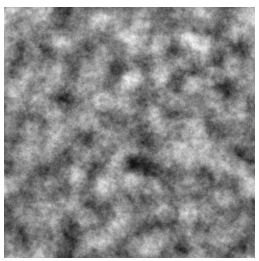
$n = 10^2$



$n = 10^3$



$n = 10^4$



$n = 10^5$

Spot noise discret asymptotique (SNDA)

- Pour la synthèse de texture on est intéressé par la limite de la séquence de spot noise quand n tend vers $+\infty$. C'est le **Spot noise discret asymptotique (SNDA)**.
- Quelle est cette limite ?
- Le *SND* d'ordre n, $f_n(x) = \sum_k h(x - X_k)$, est la somme des n images aléatoires i.i.d. $h(\cdot - X_k)$.
- D'après le **théorème central limite vectoriel** : $\frac{f_n - n\mathbb{E}(h(\cdot - X_1))}{\sqrt{n}}$ converge en loi vers le vecteur gaussien $Y = (Y(x))_{x \in \Omega_{M,N}}$ d'indice $\Omega_{M,N}$ de moyenne nulle et de covariance $\mathrm{Cov}(h(\cdot - X_1))$.

Spot noise discret asymptotique (SNDA)

- Pour la synthèse de texture on est intéressé par la limite de la séquence de spot noise quand n tend vers $+\infty$. C'est le **Spot noise discret asymptotique (SNDA)**.
- Quelle est cette limite ?
- Le *SND* d'ordre n, $f_n(x) = \sum_k h(x - X_k)$, est la somme des n images aléatoires i.i.d. $h(\cdot - X_k)$.
- D'après le **théorème central limite vectoriel** : $\frac{f_n - n\mathbb{E}(h(\cdot - X_1))}{\sqrt{n}}$ converge en loi vers le vecteur gaussien $Y = (Y(x))_{x \in \Omega_{M,N}}$ d'indice $\Omega_{M,N}$ de moyenne nulle et de covariance $\mathrm{Cov}(h(\cdot - X_1))$.

Spot noise discret asymptotique (SNDA)

- Pour la synthèse de texture on est intéressé par la limite de la séquence de spot noise quand n tend vers $+\infty$. C'est le **Spot noise discret asymptotique (SNDA)**.
- Quelle est cette limite ?
- Le *SND* d'ordre n, $f_n(x) = \sum_k h(x - X_k)$, est la somme des n images aléatoires i.i.d. $h(\cdot - X_k)$.
- D'après le **théorème central limite vectoriel** : $\frac{f_n - n\mathbb{E}(h(\cdot - X_1))}{\sqrt{n}}$ converge en loi vers le vecteur gaussien $Y = (Y(x))_{x \in \Omega_{M,N}}$ d'indice $\Omega_{M,N}$ de moyenne nulle et de covariance $\mathrm{Cov}(h(\cdot - X_1))$.

Spot noise discret asymptotique (SNDA)

- Pour la synthèse de texture on est intéressé par la limite de la séquence de spot noise quand n tend vers $+\infty$. C'est le **Spot noise discret asymptotique (SNDA)**.
- Quelle est cette limite ?
- Le *SND* d'ordre n, $f_n(x) = \sum_k h(x - X_k)$, est la somme des n images aléatoires i.i.d. $h(\cdot - X_k)$.
- D'après le **théorème central limite vectoriel** :
  $\frac{f_n - n\mathbb{E}(h(\cdot - X_1))}{\sqrt{n}}$ converge en loi vers le vecteur gaussien $Y = (Y(x))_{x \in \Omega_{M,N}}$ d'indice $\Omega_{M,N}$ de moyenne nulle et de covariance $\mathrm{Cov}(h(\cdot - X_1))$.

Explicitons les espérances et covariances

Exercice : Soit $H = h(\cdot - X)$ la translation aléatoire uniforme de h, i.e. X est un vecteur de loi uniforme sur $\Omega_{M,N}$.

1. Justifier que H est un champ stationnaire.
2. Calculer l'espérance de H, i.e. $\mathbb{E}(h(x - X))$ pour tout $x \in \Omega_{M,N}$.
3. Calculer la covariance de H, i.e. $\mathrm{Cov}(h(x - X), h(y - X))$ pour tous $x, y \in \Omega_{M,N}$.

Spot noise discret asymptotique (SNDA)

Espérance des translations aléatoires :

- X_1 suit une loi uniforme sur la grille de pixels $\Omega_{M,N}$.

$$\begin{aligned}\mathbb{E}(h(x-X_1)) &= \sum_{y\in\Omega_{M,N}} h(x-y)\mathbb{P}(X_1=y)\\ &= \sum_{y\in\Omega_{M,N}} h(x-y)\frac{1}{MN}\\ &= \frac{1}{MN}\sum_{z\in\Omega_{M,N}} h(z)\\ &= \text{moyenne de } h.\end{aligned}$$

- $\mathbb{E}(h(x-X_1)) = m$, où m est la moyenne arithmétique de h.

Spot noise discret asymptotique (SNDA)

Espérance des translations aléatoires :

- X_1 suit une loi uniforme sur la grille de pixels $\Omega_{M,N}$.

$$\begin{aligned}\mathbb{E}(h(x-X_1)) &= \sum_{y\in\Omega_{M,N}} h(x-y)\mathbb{P}(X_1=y)\\ &= \sum_{y\in\Omega_{M,N}} h(x-y)\frac{1}{MN}\\ &= \frac{1}{MN}\sum_{z\in\Omega_{M,N}} h(z)\\ &= \text{moyenne de } h.\end{aligned}$$

- $\mathbb{E}(h(x-X_1)) = m$, où m est la moyenne arithmétique de h.

Spot noise discret asymptotique (SNDA)

Espérance des translations aléatoires :

- X_1 suit une loi uniforme sur la grille de pixels $\Omega_{M,N}$.

$$\begin{aligned}\mathbb{E}(h(x-X_1)) &= \sum_{y\in\Omega_{M,N}} h(x-y)\mathbb{P}(X_1=y)\\ &= \sum_{y\in\Omega_{M,N}} h(x-y)\frac{1}{MN}\\ &= \frac{1}{MN}\sum_{z\in\Omega_{M,N}} h(z)\\ &= \text{moyenne de } h.\end{aligned}$$

- $\mathbb{E}(h(x-X_1)) = m$, où m est la moyenne arithmétique de h.

Spot noise discret asymptotique (SNDA)

Espérance des translations aléatoires :

- X_1 suit une loi uniforme sur la grille de pixels $\Omega_{M,N}$.

$$\begin{aligned}\mathbb{E}(h(x-X_1)) &= \sum_{y\in\Omega_{M,N}} h(x-y)\mathbb{P}(X_1=y)\\ &= \sum_{y\in\Omega_{M,N}} h(x-y)\frac{1}{MN}\\ &= \frac{1}{MN}\sum_{z\in\Omega_{M,N}} h(z)\\ &= \text{moyenne de } h.\end{aligned}$$

- $\mathbb{E}(h(x-X_1)) = m$, où m est la moyenne arithmétique de h.

Spot noise discret asymptotique (SNDA)

Espérance des translations aléatoires :

- X_1 suit une loi uniforme sur la grille de pixels $\Omega_{M,N}$.

$$\begin{aligned}\mathbb{E}(h(x-X_1)) &= \sum_{y\in\Omega_{M,N}} h(x-y)\mathbb{P}(X_1=y)\\ &= \sum_{y\in\Omega_{M,N}} h(x-y)\frac{1}{MN}\\ &= \frac{1}{MN}\sum_{z\in\Omega_{M,N}} h(z)\\ &= \text{moyenne de } h.\end{aligned}$$

- $\mathbb{E}(h(x-X_1)) = m$, où m est la moyenne arithmétique de h.

Spot noise discret asymptotique (SNDA)

Covariance des translations aléatoires : Soient $x, y \in \Omega_{M,N}$,

$$\begin{aligned}\mathrm{Cov}(h(x-X_1), h(y-X_1)) &= \mathbb{E}((h(x-X_1)-m)(h(y-X_1)-m)) \\ &= \sum_{z\in\Omega_{M,N}} (h(x-z)-m)(h(y-z)-m)\mathbb{P}(X_1=z) \\ &= \frac{1}{MN}\sum_{z\in\Omega_{M,N}} (h(x-z)-m)(h(y-z)-m) \\ &= C_h(x,y).\end{aligned}$$

- $\mathrm{Cov}(h(x-X_1), h(y-X_1)) = C_h(x,y)$ où C_h est l'**autocorrélation** de h :

$$C_h(x,y) = \frac{1}{MN}\sum_{t\in\Omega_{M,N}} (h(x-t)-m)\,(h(y-t)-m)\,, \quad (x,y)\in\Omega_{M,N}.$$

Spot noise discret asymptotique (SNDA)

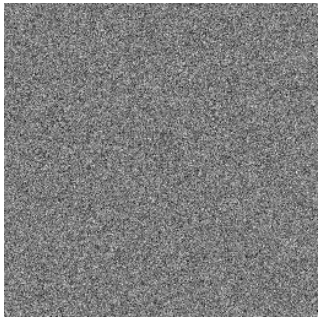
- $\mathbb{E}(h(x - X_1)) = m$, où m est la moyenne arithmétique de h.
- $\mathrm{Cov}(h(x - X_1), h(y - X_1)) = C_h(x, y)$ où C_h est l'autocorrélation de h.

Définition du *SNDA*:

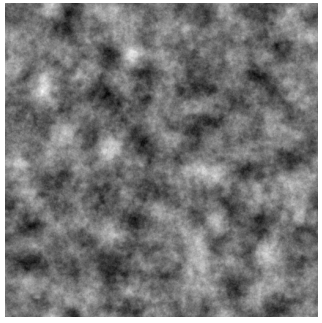
- Le spot noise discret asymptotique (SNDA) associé à h est le vecteur gaussien Y de distribution $\mathcal{N}(0, C_h)$.

Simulation du SNDA

Définition du *SNDA* associé à h : Vecteur gaussien Y de distribution $\mathcal{N}(0, C_h)$.



Bruit blanc gaussien : les pixels sont i.i.d. de distribution gaussienne

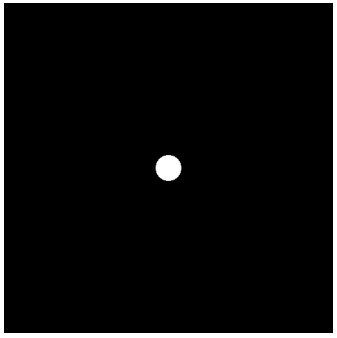


Vecteur gaussien : pixels ont une distribution gaussienne et sont corrélés

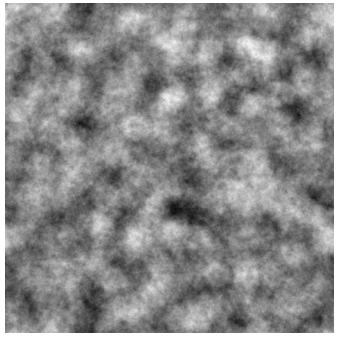
Produit de convolution : $(f * g)(x) = \sum_{y \in \Omega_{M,N}} f(x-y)g(y),\ x \in \Omega.$

Simulation du *SNDA* :

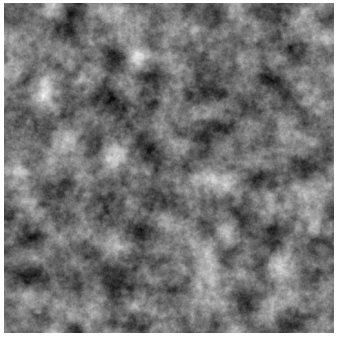
- Soit $h \in \mathbb{R}^{M \times N}$ un spot, m la moyenne de h et X un bruit blanc gaussien.
- Alors l'image aléatoire $\frac{1}{\sqrt{MN}}(h-m) * X$ est le SNDA associé à h.



Spot h



SND avec $n = 10^5$



SNDA

Exercice : Simulation du SNDA

Exercice : Montrer que $Y = \frac{1}{\sqrt{MN}} (h - m) * X \simeq \mathcal{N}(0, C_h)$ où

- $h \in \mathbb{R}^{M \times N}$ un spot,
- X un bruit blanc gaussien,
- m la moyenne de h,
- C_h est l'autocorrélation de h:

$$C_h(x, y) = \frac{1}{MN} \sum_{t \in \Omega_{M,N}} (h(x - t) - m)(h(y - t) - m), \quad (x, y) \in \Omega_{M,N}.$$

Exercice : Simulation du SNDA

Exercice : Montrer que $Y = \frac{1}{\sqrt{MN}} (h - m) * X \simeq \mathcal{N}(0, C_h)$.

- Y est l'image de X par une application linéaire. Comme X est un vecteur gaussien, Y est également un vecteur gaussien.
- Il suffit donc de vérifier que $\mathbb{E}(Y(x)) = 0$ et $\mathrm{Cov}(Y(x), Y(y)) = C_h(x, y)$.
- Par linéarité, $\mathbb{E}(Y(x)) = \frac{1}{\sqrt{MN}} (h - m) * \mathbb{E}(X)(x) = 0$.
- Soient $x, y \in \Omega_{M;N}$,

$$\begin{aligned}
\mathrm{Cov}(Y(x), Y(y)) &= \mathbb{E}(Y(x)Y(y)) \\
&= \frac{1}{MN} \mathbb{E}\left(\sum_{s \in \Omega_{M,N}} (h(s-x) - m) X(s) \sum_{t \in \Omega_{M,N}} (h(t-y) - m) X(t) \right) \\
&= \frac{1}{MN} \sum_{s,t \in \Omega_{M,N}} (h(s-x) - m)(h(t-y) - m) \underbrace{\mathbb{E}(X(s)X(t))}_{= 1 \text{ si } s = t \text{ et } 0 \text{ sinon}} \\
&= \frac{1}{MN} \sum_{s \in \Omega_{M,N}} (h(s-x) - m)(h(t-y) - m) \\
&= C_h(x, y)
\end{aligned}$$

Exercice : Simulation du SNDA

Exercice : Montrer que $Y = \frac{1}{\sqrt{MN}} (h - m) * X \simeq \mathcal{N}(0, C_h)$.

- Y est l'image de X par une application linéaire. Comme X est un vecteur gaussien, Y est également un vecteur gaussien.
- Il suffit donc de vérifier que $\mathbb{E}(Y(x)) = 0$ et $\mathrm{Cov}(Y(x), Y(y)) = C_h(x, y)$.
- Par linéarité, $\mathbb{E}(Y(x)) = \frac{1}{\sqrt{MN}} (h - m) * \mathbb{E}(X)(x) = 0$.
- Soient $x, y \in \Omega_{M;N}$,

$$\mathrm{Cov}(Y(x), Y(y)) = \mathbb{E}(Y(x)Y(y))$$

$$= \frac{1}{MN} \mathbb{E}\left(\sum_{s \in \Omega_{M,N}} (h(s - x) - m) X(s) \sum_{t \in \Omega_{M,N}} (h(t - y) - m) X(t) \right)$$

$$= \frac{1}{MN} \sum_{s,t \in \Omega_{M,N}} (h(s - x) - m)(h(t - y) - m) \underbrace{\mathbb{E}(X(s)X(t))}_{= 1 \text{ si } s = t \text{ et } 0 \text{ sinon}}$$

$$= \frac{1}{MN} \sum_{s \in \Omega_{M,N}} (h(s - x) - m)(h(t - y) - m)$$

$$= C_h(x, y)$$

Exercice : Simulation du SNDA

Exercice : Montrer que $Y = \frac{1}{\sqrt{MN}} (h - m) * X \simeq \mathcal{N}(0, C_h)$.

- Y est l'image de X par une application linéaire. Comme X est un vecteur gaussien, Y est également un vecteur gaussien.
- Il suffit donc de vérifier que $\mathbb{E}(Y(x)) = 0$ et $\mathrm{Cov}(Y(x), Y(y)) = C_h(x, y)$.
- Par linéarité, $\mathbb{E}(Y(x)) = \frac{1}{\sqrt{MN}} (h - m) * \mathbb{E}(X)(x) = 0$.
- Soient $x, y \in \Omega_{M;N}$,

$$\mathrm{Cov}(Y(x), Y(y)) = \mathbb{E}(Y(x)Y(y))$$

$$= \frac{1}{MN} \mathbb{E}\left(\sum_{s \in \Omega_{M,N}} (h(s-x) - m) X(s) \sum_{t \in \Omega_{M,N}} (h(t-y) - m) X(t) \right)$$

$$= \frac{1}{MN} \sum_{s,t \in \Omega_{M,N}} (h(s-x) - m)(h(t-y) - m) \underbrace{\mathbb{E}(X(s)X(t))}_{= 1 \text{ si } s = t \text{ et } 0 \text{ sinon}}$$

$$= \frac{1}{MN} \sum_{s \in \Omega_{M,N}} (h(s-x) - m)(h(t-y) - m)$$

$$= C_h(x, y)$$

Exercice : Simulation du SNDA

Exercice : Montrer que $Y = \frac{1}{\sqrt{MN}} (h - m) * X \simeq \mathcal{N}(0, C_h)$.

- Y est l'image de X par une application linéaire. Comme X est un vecteur gaussien, Y est également un vecteur gaussien.
- Il suffit donc de vérifier que $\mathbb{E}(Y(x)) = 0$ et $\mathrm{Cov}(Y(x), Y(y)) = C_h(x, y)$.
- Par linéarité, $\mathbb{E}(Y(x)) = \frac{1}{\sqrt{MN}} (h - m) * \mathbb{E}(X)(x) = 0$.
- Soient $x, y \in \Omega_{M;N}$,

$$\begin{aligned}
&\mathrm{Cov}(Y(x), Y(y)) = \mathbb{E}(Y(x)Y(y)) \\
&= \frac{1}{MN} \mathbb{E}\left(\sum_{s \in \Omega_{M,N}} (h(s-x) - m) X(s) \sum_{t \in \Omega_{M,N}} (h(t-y) - m) X(t) \right) \\
&= \frac{1}{MN} \sum_{s,t \in \Omega_{M,N}} (h(s-x) - m)(h(t-y) - m) \underbrace{\mathbb{E}(X(s)X(t))}_{= 1 \text{ si } s = t \text{ et } 0 \text{ sinon}} \\
&= \frac{1}{MN} \sum_{s \in \Omega_{M,N}} (h(s-x) - m)(h(t-y) - m) \\
&= C_h(x, y)
\end{aligned}$$

Simulation du SNDA

Bien sûr le produit de convolution $Y = \frac{1}{\sqrt{MN}} (h - m) * X$ **est calculé par TFD !**

Outline

Phase aléatoire

- Définition: Une image aléatoire $\theta : \hat{\Omega} \to \mathbb{R}$ est une ***phase aléatoire*** si
  1. Symétrie: θ est impaire:
  $$\forall (k,l) \in \hat{\Omega}_{M,N}, \theta(-k,-l) = -\theta(k,l).$$
  2. Distribution: Chaque coordonnée $\theta(k,l)$ est
     - uniforme sur l'intervalle $]-\pi,\pi]$ si $(k,l) \notin \{(0,0), (\frac{M}{2},0), (0,\frac{N}{2}), (\frac{M}{2},\frac{N}{2})\}$,
     - uniforme sur $\{0,\pi\}$ sinon.
  3. Indépendance: Pour chaque sous-ensemble $\mathcal{S} \subset \hat{\Omega}_{M,N}$ qui ne contient pas des points distincts symétriques, les variables aléatoires $\{\theta(k,l)|(k,l) \in \mathcal{S}\}$ sont indépendantes.

Bruit à phase aléatoire (BPA)

- Algorithme de synthèse de textures: **Bruit à phase aléatoire (BPA)** = randomization de phase :

1. Calculer la TFD $\hat{h}$ de l'input h.
2. Calculer une phase aléatoire θ à l'aide d'un générateur de nombre pseudo-aléatoire.
3. Calculer $\hat{Z} = \left|\hat{h}\right| e^{i\theta}$ (or $\hat{Z} = \hat{h}e^{i\theta}$).
4. Retourner Z la TFD inverse de $\hat{Z}$.

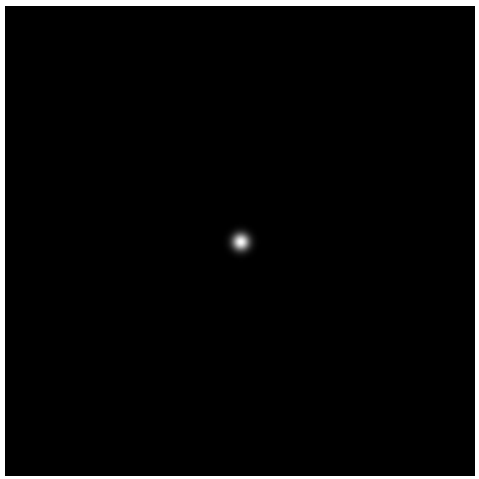
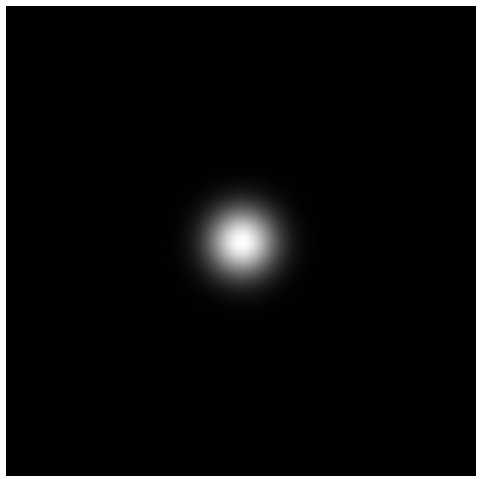
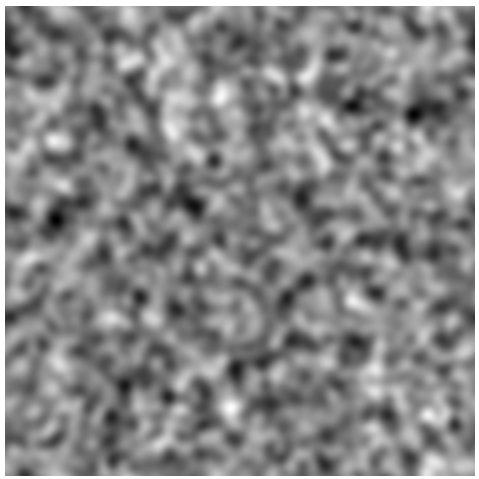


Image originale h



Module $\left|\hat{h}\right|$

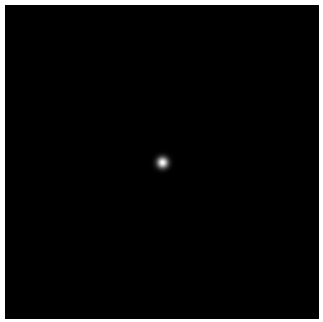


BPA associé à h

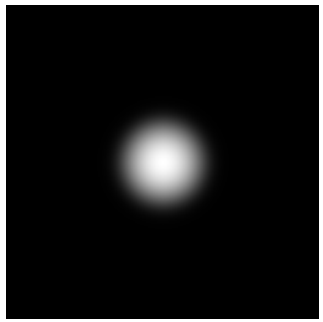
Différences entre le BPA et l'SNDA

Proposition:

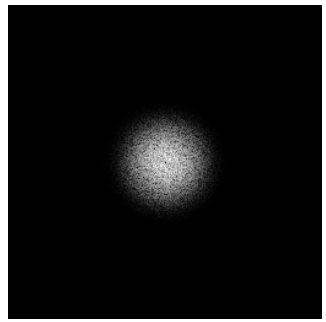
- BPA et SNDA ont tous les deux une phase aléatoire.
- Le module de Fourier du BPA est celui de h.
- Le module de Fourier du SNDA est la multiplication entre $\left|\hat{h}\right|$ et un bruit de Rayleigh.



Spot h

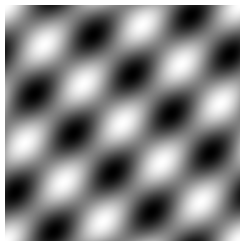


Module du BPA

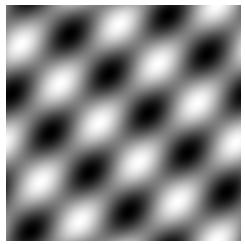


Module du SNDA

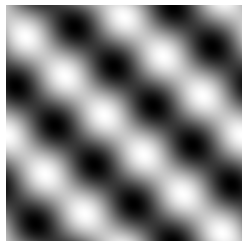
- **Le BPA et le SNDA sont deux vecteurs aléatoires différents.**



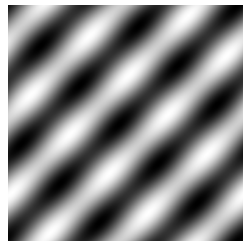
Spot h



BPA



Une réalisation du SNDA



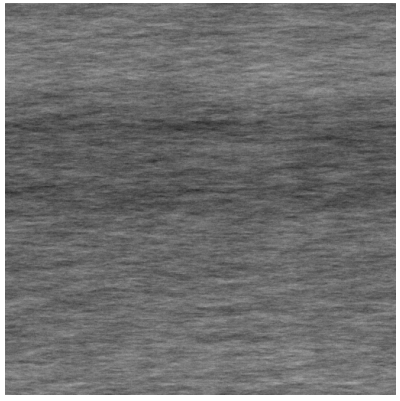
Une autre réalisation du SNDA

Outline

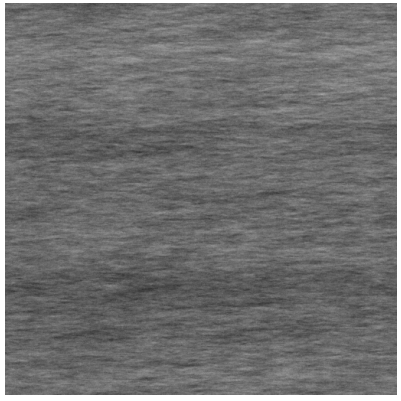
BPA and *SNDA* associés à des images de textures

- Certaines textures sont relativement bien reproduites par BPA et SNDA.

Image originale



BPA



SNDA

- ... Mais plusieurs amélioration sont nécessaires : Images couleurs, artefacts basses fréquences,...

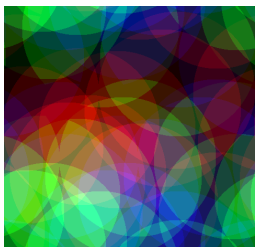
Extension aux images couleurs

- On utilise la représentation RGB des images couleurs.
- ***SND* couleur:** La définition du SND s'étend aux images couleurs $h = (h_r, h_g, h_b)$.
- Le SNDA couleur Y est la limite gaussienne obtenue en faisant tendre le nombre de spots vers $+\infty$. On le simule par :

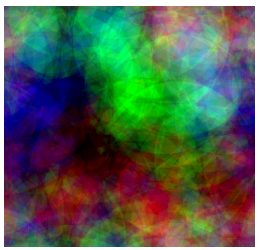
$$Y = \frac{1}{\sqrt{MN}} \begin{pmatrix} (h_r - m_r) * X \\ (h_g - m_g) * X \\ (h_b - m_b) * X \end{pmatrix}, \quad X \text{ un bruit blanc gaussien}$$

- On convole chaque canal couleur par le **même** bruit blanc gaussien X.

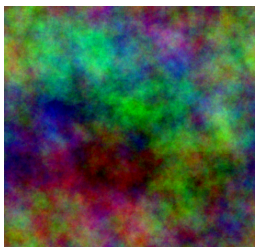
Spot h



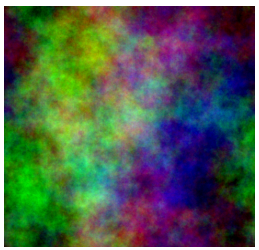
$n = 10$



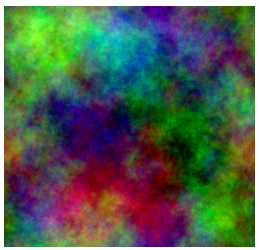
$n = 10^2$



$n = 10^3$



$n = 10^4$



SNDA couleur

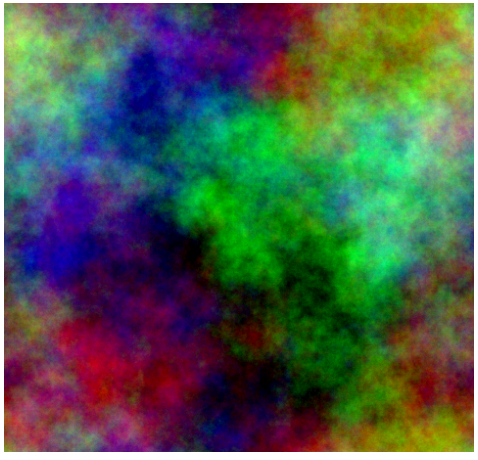
- Phase du SNDA couleur *SNDA*: La même phase aléatoire est ajoutée à la TFD de chaque canal couleur.

Extension aux images couleurs

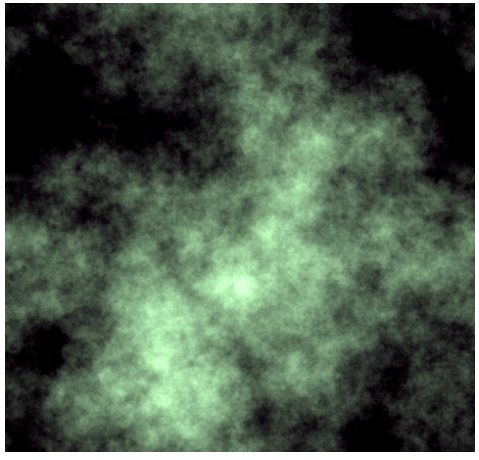
- **BPA couleur :** Par analogie, le *BPA* associé à l'image couleur $h = (h_r, h_g, h_b)$ est l'image couleur obtenue en **ajoutant la même phase aléatoire** à la TFD de chaque canal couleur.

Image originale h

BPA couleur



"Mauvais BPA couleur" : chaque canal à la même phase aléatoire

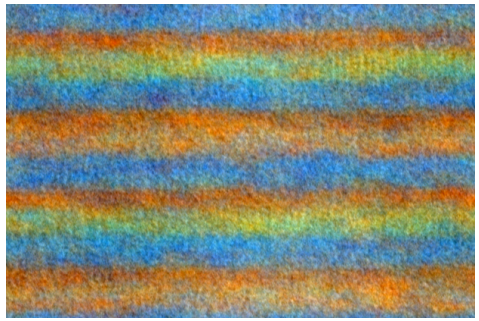


$$\hat{h} = \begin{pmatrix} |\hat{h}_R| e^{i\varphi_R} \\ |\hat{h}_G| e^{i\varphi_G} \\ |\hat{h}_B| e^{i\varphi_B} \end{pmatrix} \qquad \hat{Z} = \begin{pmatrix} |\hat{h}_R| e^{i(\varphi_R+\theta)} \\ |\hat{h}_G| e^{i(\varphi_G+\theta)} \\ |\hat{h}_B| e^{i(\varphi_B+\theta)} \end{pmatrix} \qquad \hat{Z}_W = \begin{pmatrix} |\hat{h}_R| e^{i\theta} \\ |\hat{h}_G| e^{i\theta} \\ |\hat{h}_B| e^{i\theta} \end{pmatrix}$$

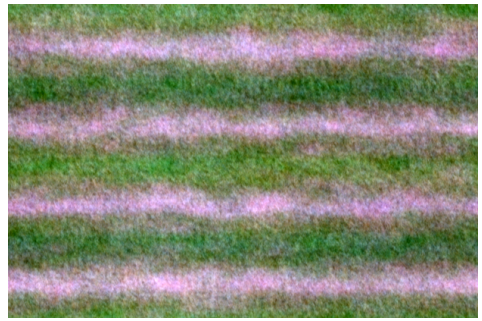
Extension aux images couleurs

- Un autre exemple avec une photo.

Image originale h



BPA couleur



“Mauvais BPA couleur”

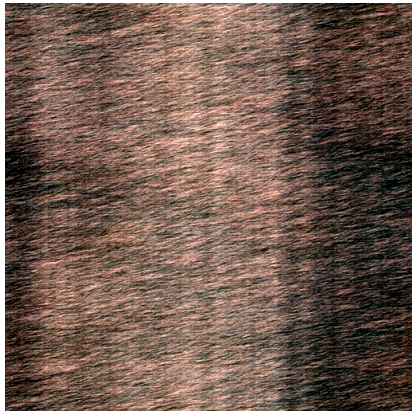
- Préserver les décalages de phase entre les canaux couleurs est important pour conserver le contenu de couleur.
- ...toutefois pour de nombreuses textures monochromatic, il n’y a pas de différence !

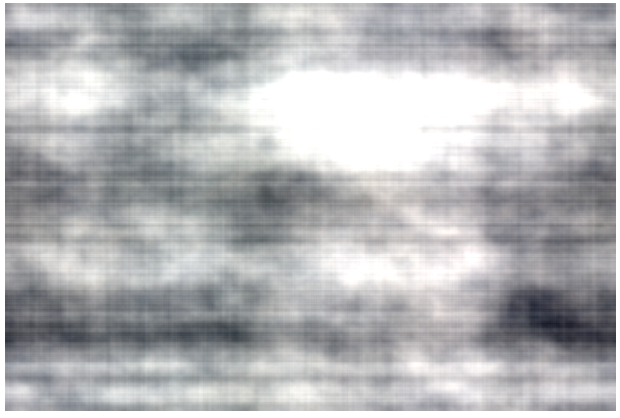
Eviter les artefacts dus à la non périodicité

- SNDA et BPA utilisent la TFD.
  $\Longrightarrow$ Hypothèse implicite de périodicité
- Utiliser des images non périodiques entraîne des artefacts importants.

Spot *h*

SNDA





Eviter les artefacts dus à la non périodicité

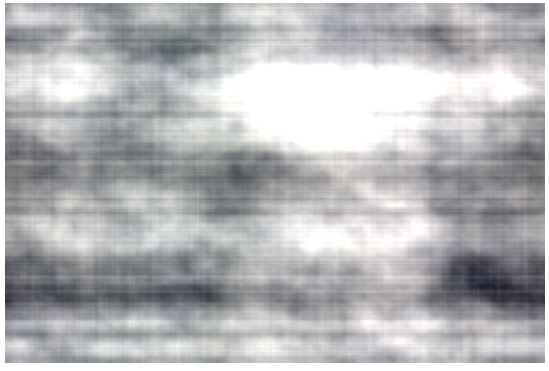
- Solution: Forcer la périodicité de l'image d'entrée.
- L'image h est remplacée par sa **composante périodique** $p = \mathrm{per}(h)$.

[Moisan, 2011]

Eviter les artefacts dus à la non périodicité

Spot h

SNDA(h)

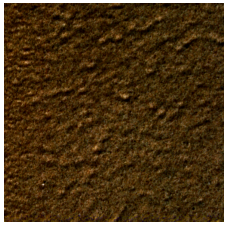


SNDA(p)

Synthétiser des textures de plus grande taille

Pour synthétiser des textures de plus grande taille que celle de h, on calcule un “spot équivalent” $\tilde{h}$:

- Copier $p = \mathrm{per}(h)$ au centre d’une image constante à la moyenne h.
- Normaliser la variance.
- Atténuer la transition sur la frontière intérieure.



Spot h

Spot équivalent $\tilde{h}$

BPA(h)

BPA$\left(\tilde{h}\right)$

Propriétés des algorithmes

- Les deux algorithmes sont rapides, avec la complexité de la FFT [$\mathcal{O}(MN \log(MN)$
- Stabilité visuelle : Toutes les réalisations sont visuellement similaires.

Spot h

BPA 1

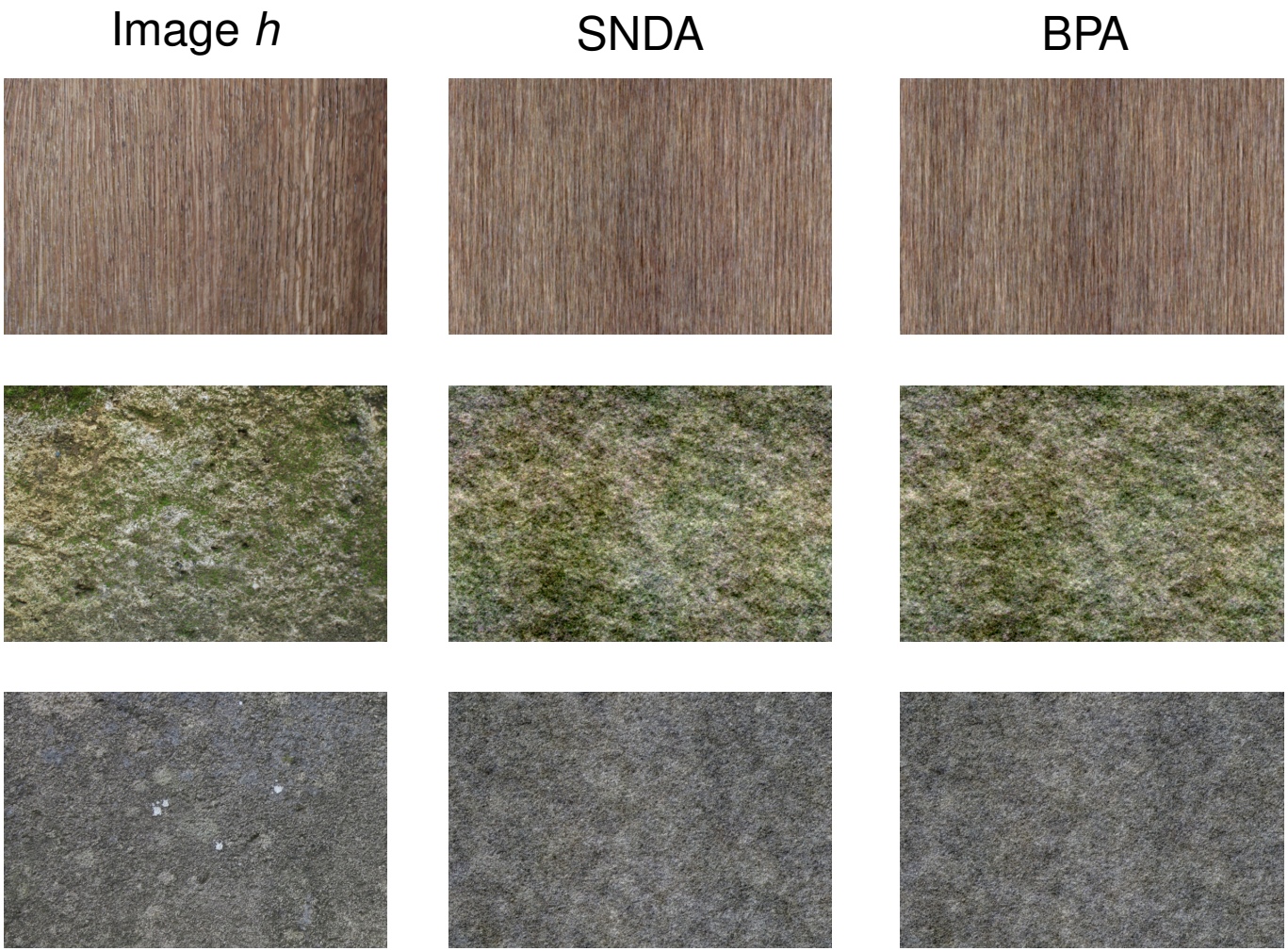
BPA 2

BPA 3

Outline

Résultats numériques : Similarité des textures

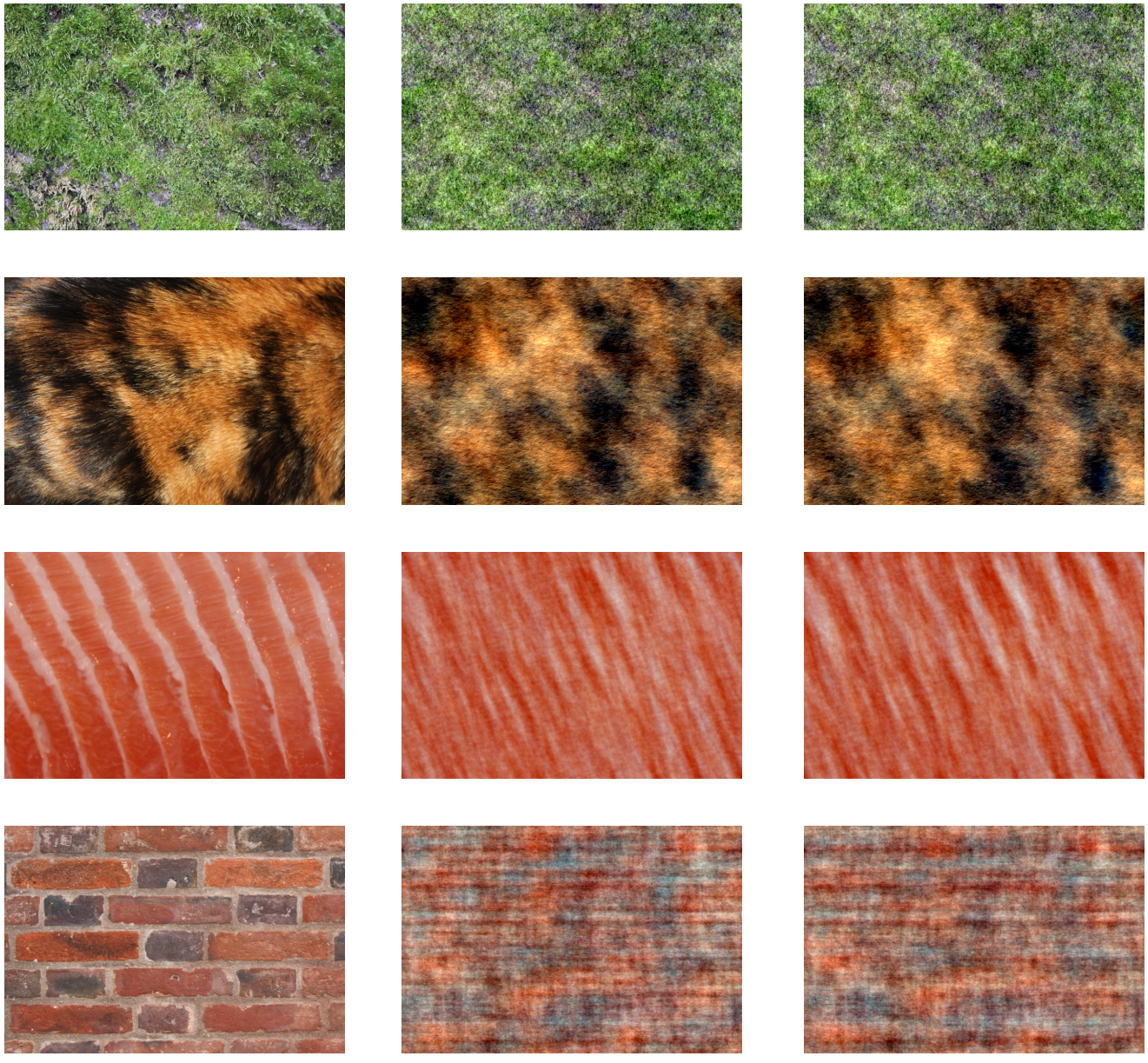
- Pour comparer les deux algorithmes, on utilise la même phase aléatoire pour SNDA et BPA.



- Les deux algorithmes produisent des textures visuellement similaires.

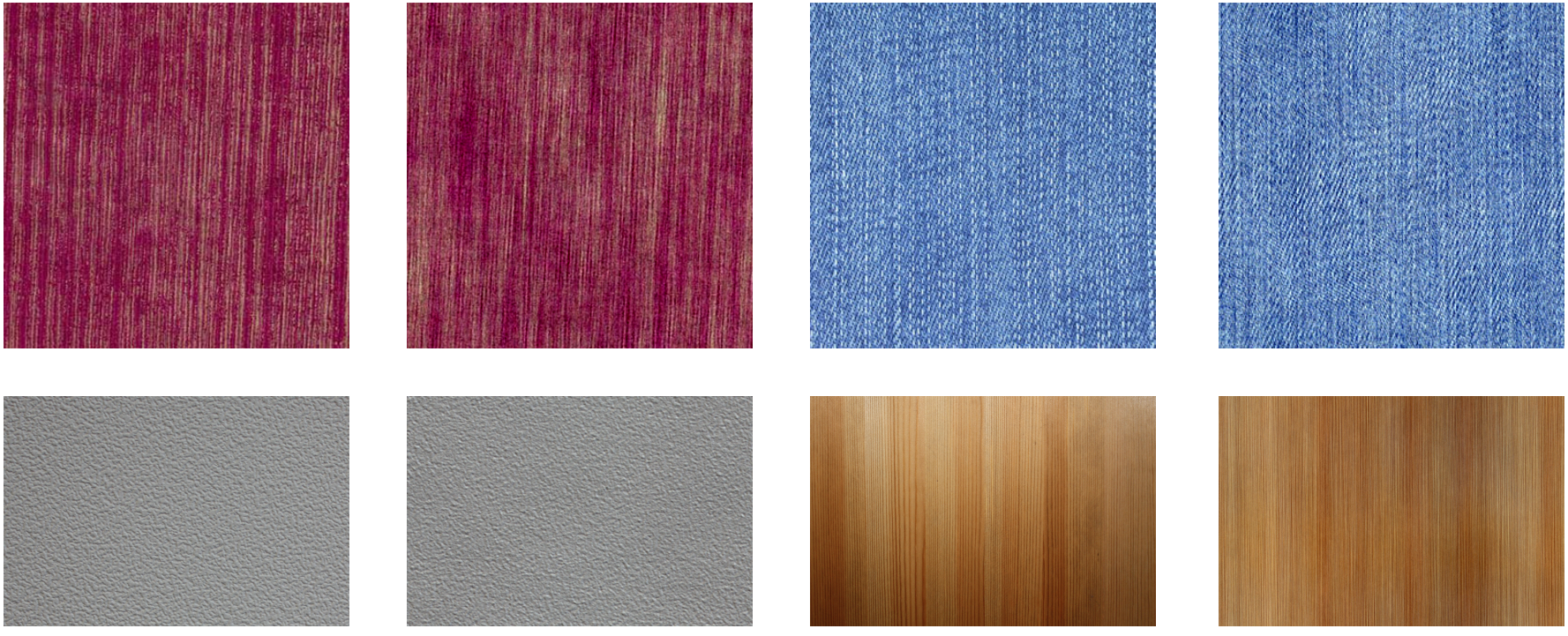
Résultats numériques : Textures à phase non aléatoire

Image h | SNDA | BPA



D'autres exemples...

Image *h* | BPA | Image *h* | BPA



Bibliographic references I


A. A. Efros and W. T. Freeman, *Image quilting for texture synthesis and transfer*, SIGGRAPH '01, 2001


A. A. Efros and T. K. Leung, *Texture synthesis by non-parametric sampling*, ICIP 1999, 1999


B. Galerne, Y. Gousseau, and J.-M. Morel, *Random phase textures: Theory and synthesis*, IEEE Trans. Image Process., 2011


B. Galerne, Y. Gousseau, J.-M. Morel, *Micro-Texture Synthesis by Phase Randomization*, Image Processing On Line, 2011

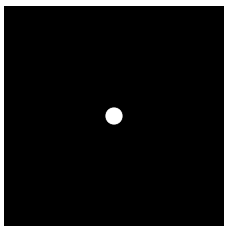

D. J. Heeger and J. R. Bergen, *Pyramid-based texture analysis/synthesis*, SIGGRAPH '95, 1995

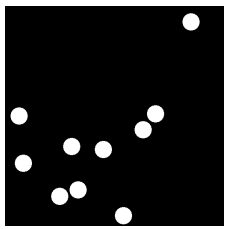

L. Moisan, *Periodic plus smooth image decomposition*, J. Math. Imag. Vis., 2011

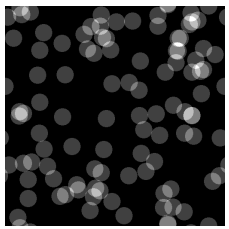
Outline

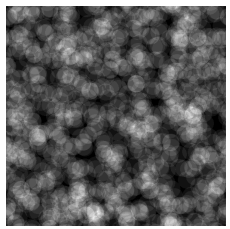
TP Python

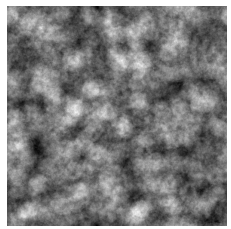
- Définir une fonction `circshift(u,t1,t2)` qui applique une translation circulaire de vecteur $t = (t_1, t_2)$ à l'image u.
- Ecrire une fonction `snd(u,n)` qui simule un SND v d'ordre n associé à u.
- Ecrire une fonction `snda(u)` qui simule un SNDA v associé à u.
- Reproduire une expérience de ce type (avec la forme géométrique de votre choix) :

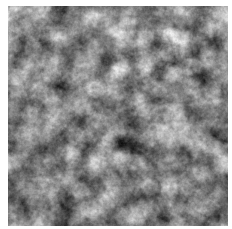

Spot h

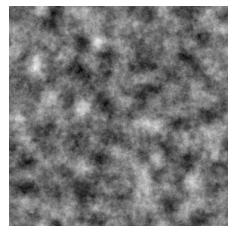

$n = 10$


$n = 10^2$


$n = 10^3$


$n = 10^4$


$n = 10^5$


SNDA

TP Python

- Ecrire une fonction `sndargb(u)` qui simule un SNDA associé à une image RGB u. On prendra soin d'ajouter la couleur moyenne de u à l'image v (pour ne pas avoir une image couleur de moyenne nulle).
- Faire plusieurs expériences pour montrer que :
  - La synthèse de micro-textures est satisfaisante.
  - La synthèse de macro-textures n'est pas satisfaisante.
  - Des problèmes de bords sont visibles dans les textures synthétisées.
- Remplacer les images d'entrée par leur composante périodique (appliquée à chaque canal couleur).

Periodic component

- **Periodic component** $p = \mathrm{per}(h)$, as proposed by [Moisan, 2011].
- Definition of the periodic component p of h: p unique solution of

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

where, noting N_x the neighborhood of $x \in \Omega$ for 4-connexity:

$$\Delta f(x) = -4f(x) + \sum_{y \in N_x} f(y) \quad \text{and} \quad \Delta_i f(x) = -|N_x \cap \Omega| f(x) + \sum_{y \in N_x \cap \Omega} f(y).$$

These two Laplacians only differ at the border:

- Δ: discrete Laplacian with periodic boundary conditions
- Δ_i: discrete Laplacian without periodic boundary conditions (index i for interior)

- p is “visually close” to h (same Laplacian).
- p is fastly computed using the FFT...

FFT-based Poisson Solver

Periodic Poisson problem: Find the image p such that

$$\begin{cases} \Delta p = \Delta_i h \\ \text{mean}(p) = \text{mean}(h) \end{cases}$$

In the Fourier domain, this system becomes:

$$\begin{cases} \left(-4 + 2\cos\left(\frac{2s\pi}{M}\right) + 2\cos\left(\frac{2t\pi}{N}\right)\right)\hat{p}(s,t) = \widehat{\Delta_i h}(s,t), \ (s,t) \in \hat{\Omega} \setminus \{(0,0)\}, \\ \hat{p}(0,0) = \text{mean}(h). \end{cases}$$

Algorithm to compute the periodic component:

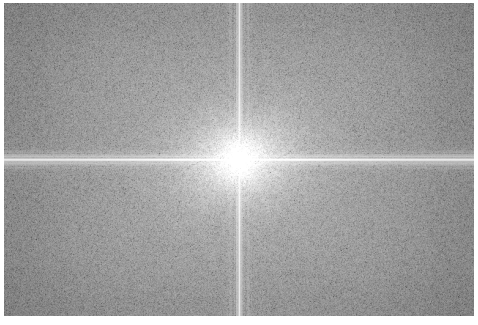
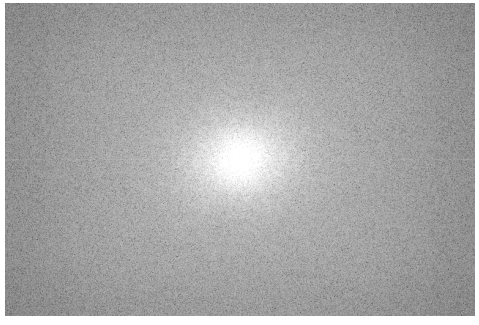
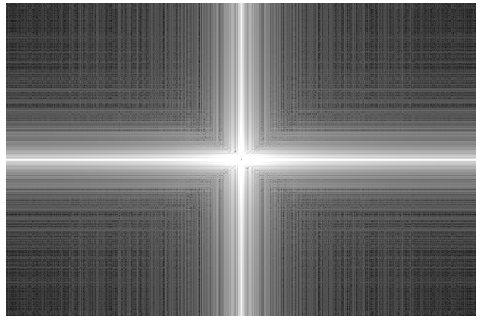
1. Compute $\Delta_i h$ the discrete Laplacian of h.
2. Compute $m = \text{mean}(h)$.
3. Compute $\widehat{\Delta_i h}$ the DFT of $\Delta_i h$ using the forward FFT.
4. Compute the DFT $\hat{p}$ of p defined by

$$\begin{cases} \hat{p}(s,t) = \frac{\widehat{\Delta_i h}((s,t))}{-4+2\cos\left(\frac{2s\pi}{M}\right)+2\cos\left(\frac{2t\pi}{N}\right)} & \text{for } (s,t) \in \hat{\Omega} \setminus \{(0,0)\} \\ \hat{p}(0,0) = m \end{cases}$$

5. Compute p using the backward FFT (if necessary).

Periodic component: effects on the Fourier modulus

- p is "visually close" to h (same Laplacian).

	Image h	Periodic component $p = \mathrm{per}(h)$	Smooth component $s = h - p$ $(+m)$
Images			
Fourier modulus			

- The application $\mathrm{per} : h \mapsto p$ filters out the "cross structure" of the spectrum.