INFSI 350
Textures par le bruit en informatique graphique

Axel Schumacher
Bertrand Chazot
Samuel Mokrani

13 avril 2012

http://www-sop.inria.fr/reves/Basilic/2011/LD11/

Plan

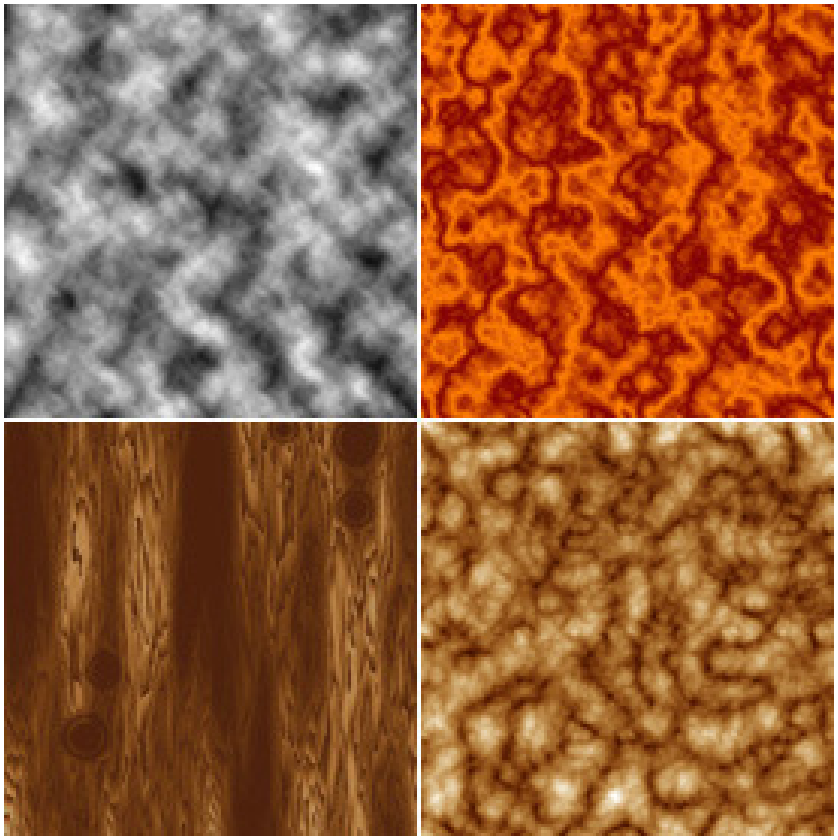
1. Généralités
2. Bruit de Perlin
3. Bruit par ondelettes (Wavelet Noise)
4. Bruit de Gabor
5. Conclusion
6. Démo

Texture par le bruit

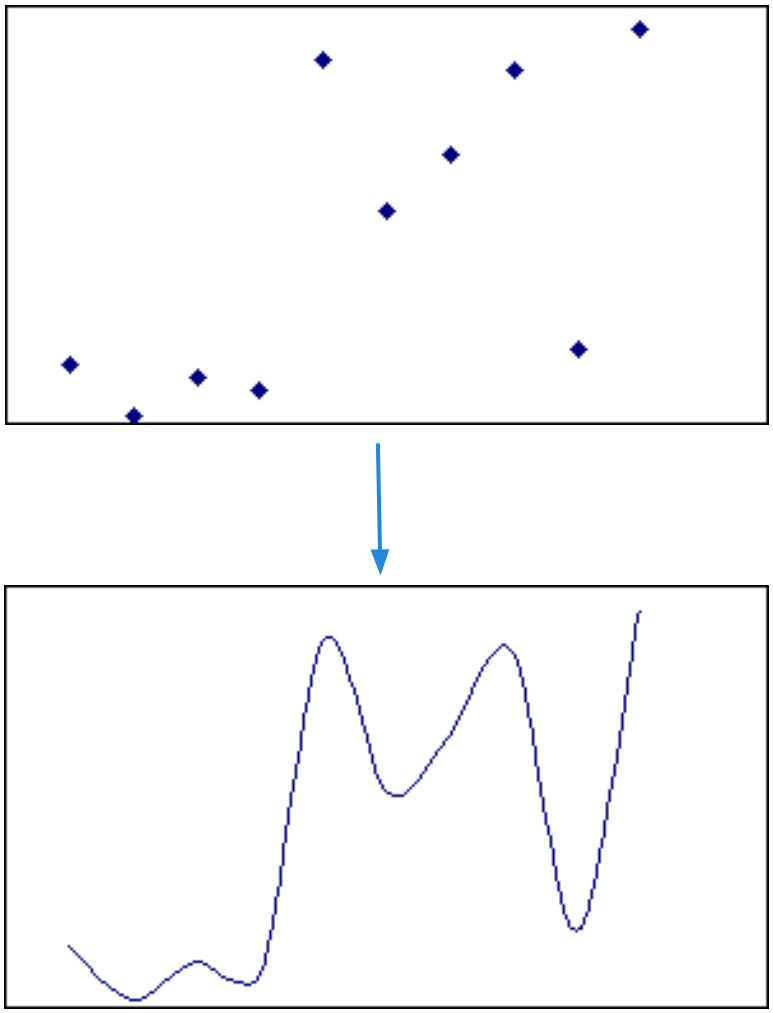
- Essentiel pour générer des textures naturelles (bois, pierre, métal, ...)

- Texture procédurale:
  - gain en place mémoire
  - pas besoin de plaquer

Bruit de Perlin (1985)



Génération d'un bruit et interpolation



- Bruit sur un nombre fini de points
- Interpolation:
  - linéaire
  - cosinusoïdale
  - cubique
- Se généralise en 2D, 3D, ...

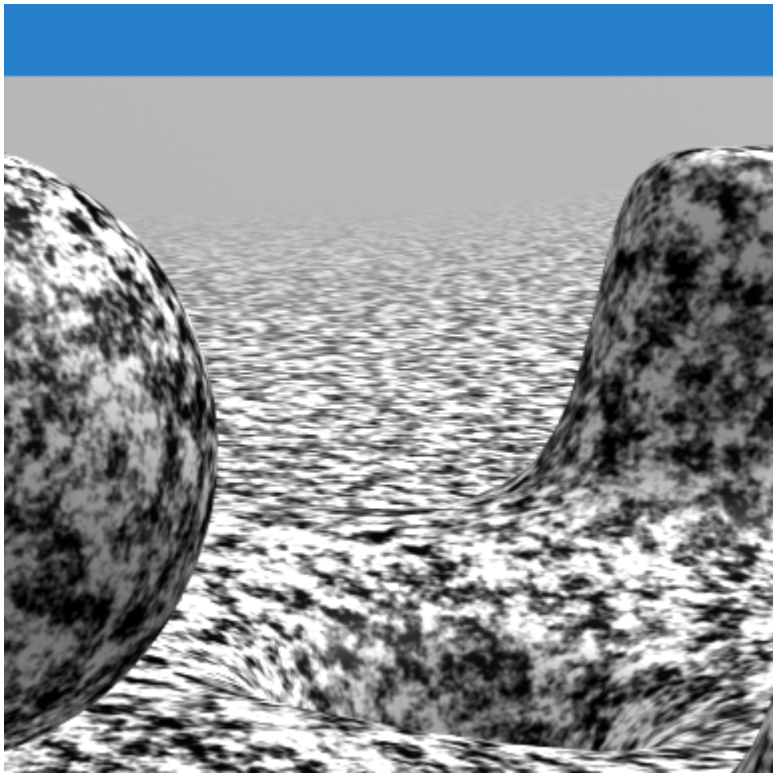
Construction du bruit de Perlin

- Sommer différentes fonctions de bruit:
  - Variation du "pas"
  - Variation de l'amplitude

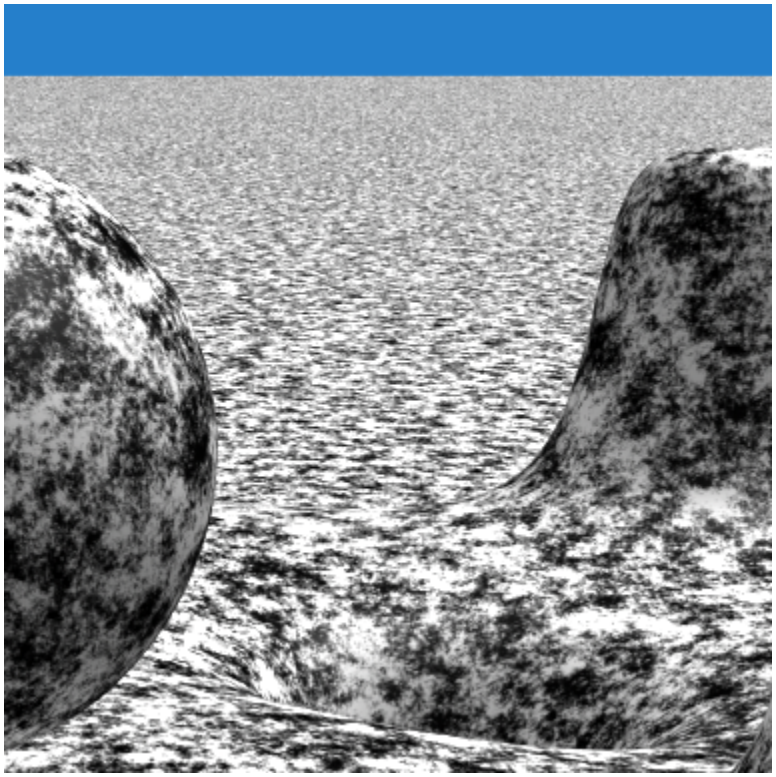
$$Perlin(x) = \sum_i p^i * interpolatedNoise(2^i * x)$$

- Le pas est multiplié par deux
  - bruit de moins en moins cohérent
  - nécessite de diminuer la persistance 0 < p < 1

Wavelet (2005)

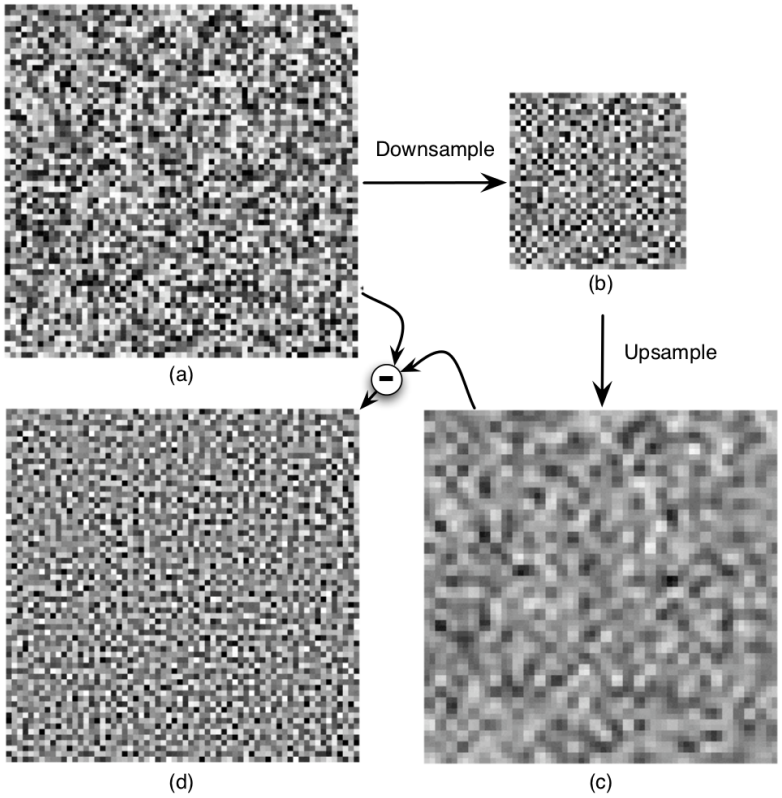


Bruit de Perlin



Wavelet

Création du bruit



$$M(x) = \sum_{b=b_{min}}^{b_{max}} w_b N(2^b x))$$

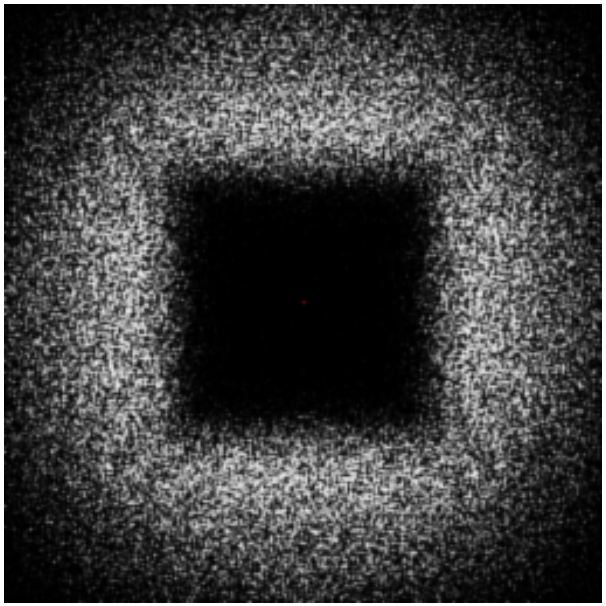
$$Pixel(i) = \int M(2^s(x-k))K(x-i)dx$$

$$Pixel(i) = \sum_{j=b_{min}+s}^{b_{max}+s} w_{j-s} \int N(2^j x - l)K(x-i)dx$$

- Pour j ≥ 0, et pour tout l

$$\int N(2^j x - l)K(x-i)dx = 0$$

Propriétés du bruit



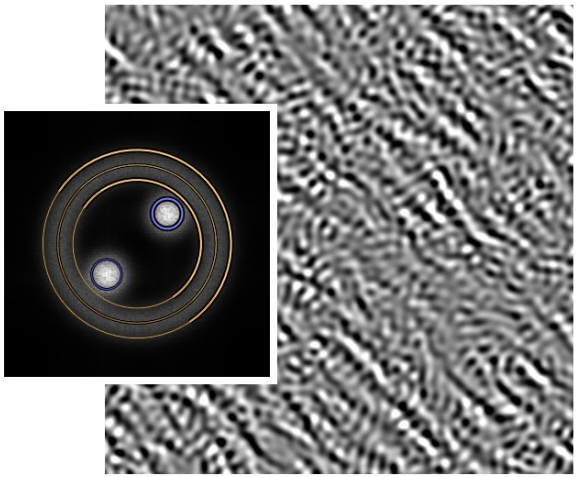
- Élimination des hautes fréquences créant de l'aliasing (orthogonalité)
- Élimination des basses fréquences pour éliminer les motifs

Surface 2D et bruit 3D

- Coupe 2D dans bruit 3D limité en bande : pas limité en bande (Lewis 1989)
- On intègre sur la normale du vertex

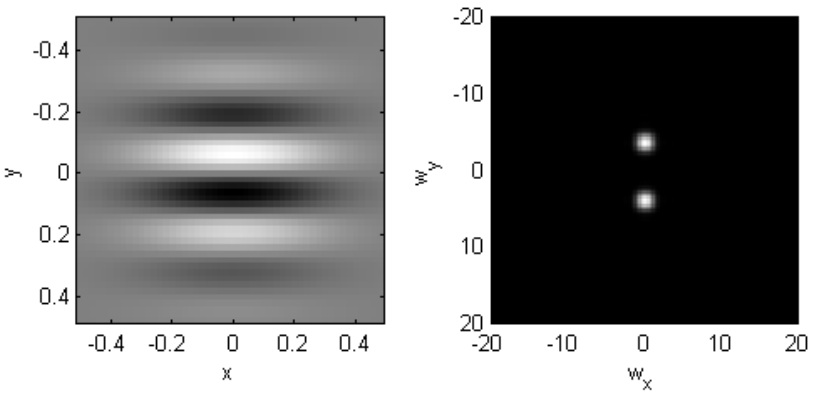
- Bruit non évalué dans tout l'espace (coûteux)
  - pavage avec une/des tuile(s)

Bruit de Gabor (2009)



Le noyau de Gabor

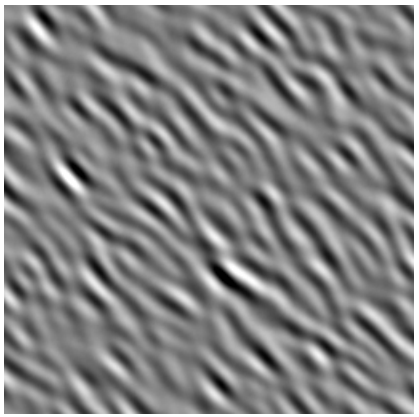
$$g(x,y) = Ke^{-\pi a^2(x^2+y^2)}cos(2\pi F_0(xcos\omega_0 + ysin\omega_0))$$



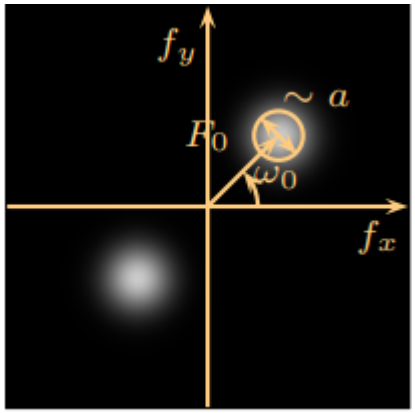
- Gaussienne modulée par un cosinus
- Paramètres intuitifs

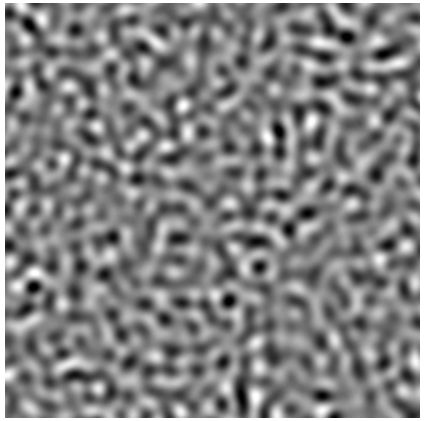
$$N(x,y) = \left[g * \sum_i w_i \delta_i(x_i, y_i) \right] (x,y)$$

Bruit anisotrope et isotrope

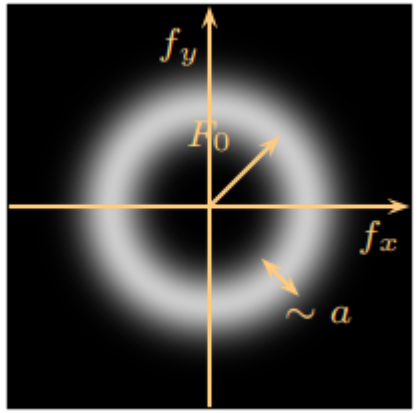


$$N(x,y) = \sum_i \omega_i g(x - x_i, y - y_i)$$

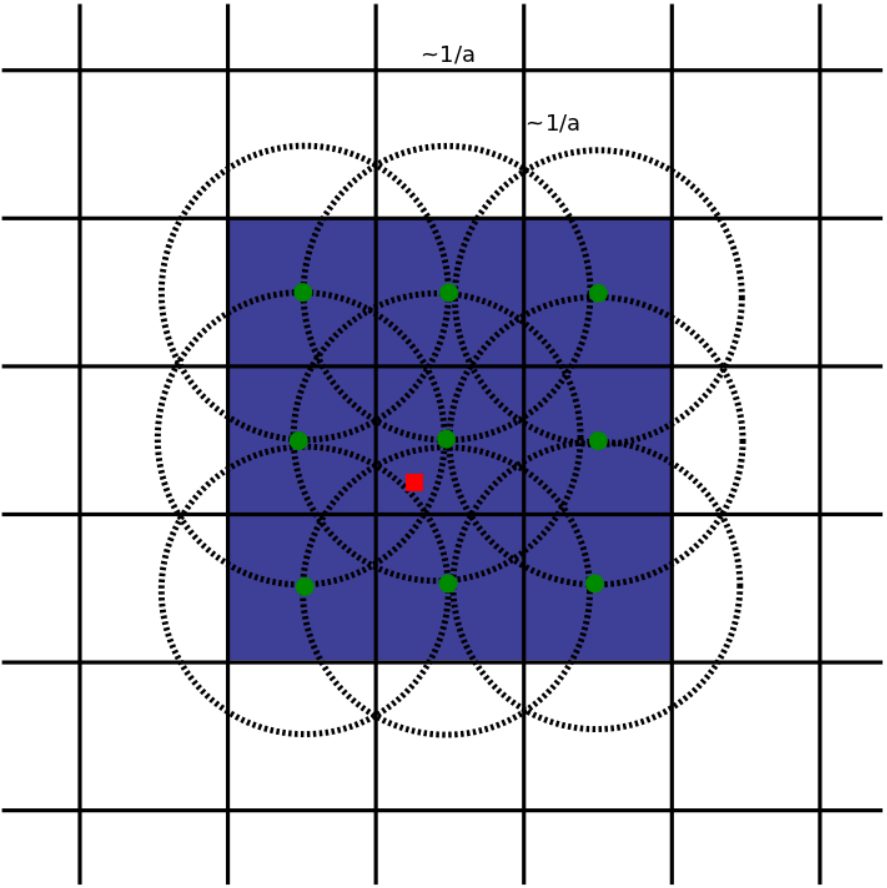




$$N(x,y) = \sum_i \omega_i g(x - x_i, y - y_i : \omega_{0,i})$$

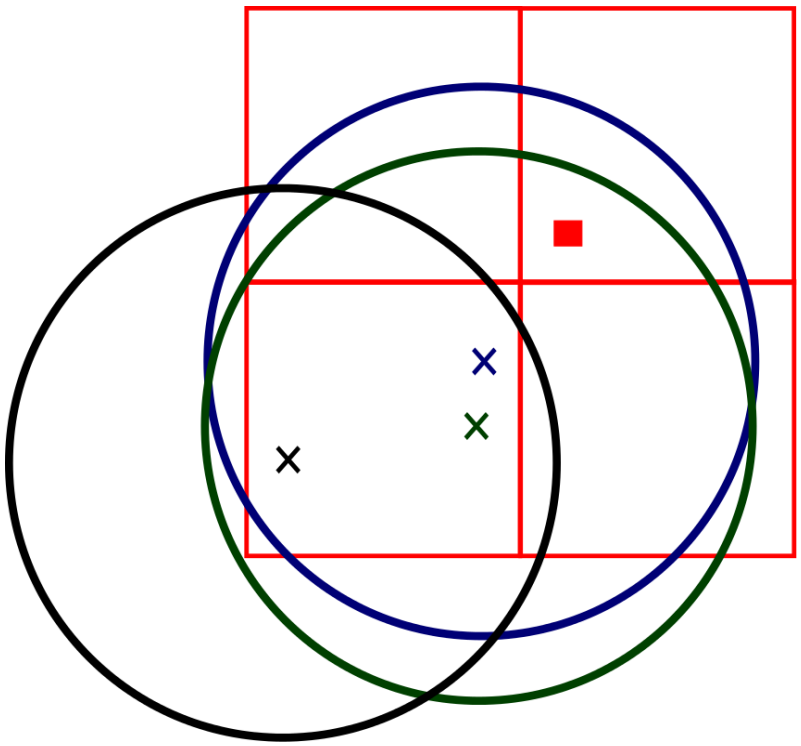


Division en cellules



- taille d'une cellule:
  - ~1/a: enveloppe gaussienne atteint 5% de sa valeur max
  - seulement évaluer l'influence des cellules voisines

Plusieurs impulsions par cellule



- Par cellule :
  - calcul nombre impulsions (suit une loi de Poisson)
  - pour chaque impulsion : calcul d'un nouveau centre (suit loi uniforme)
    - si le pixel d'origine est dans le cercle : Gabor dans le repère du cercle

Conclusion

- éviter l'aliasing
- continuité
- facilement paramétrable
- pré-calcul contre totalement procédural

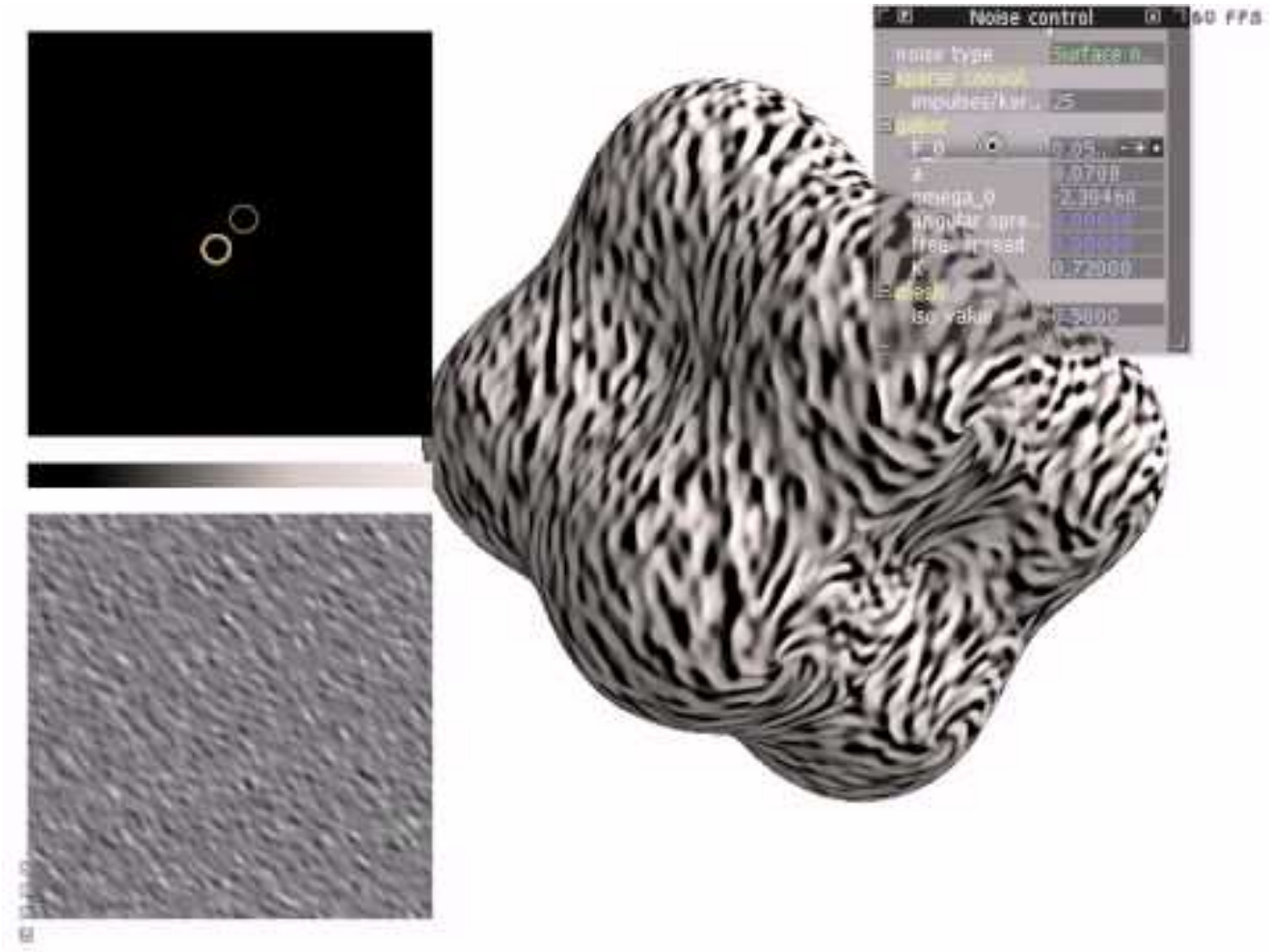
Démo

Bibliographie

- *An image synthetizer*: Ken Perlin 1985
  - dl.acm.org/citation.cfm?id=325247
- *Improving Noise*: Ken Perlin 1985
  - mrl.nyu.edu/~perlin/doc/oscar.html
- *Wavelet Noise*: Robert L. Cook, Tony DeRose 2005
  - graphics.pixar.com/library/WaveletNoise/paper.pdf
- *Procedural Noise using Sparse Gabor Convolution*: Ares Lagae, Sylvain Lefebvre, Georges Drettrakis, Philip Dutré 2009
  - graphics.cs.kuleuven.be/publications/LLDD09PNSGC/

- Code glsl : github.com/Rekamux/Noise-as-textures
- Code 2D : github.com/B-C/gminis/

Bonus : Gabor


Noise control
noise type
impulses/ker...
omega_0
angular spre...
freq. spread
K
iso value
60 FPS