Better Faster Noise with the GPU

Wyvill, Geoff; Frisvad, Jeppe Revall

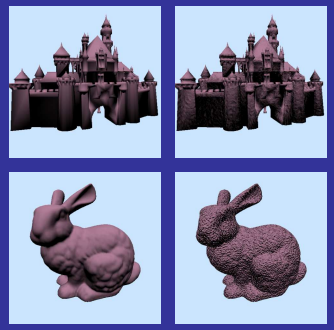
Publication date:
2007

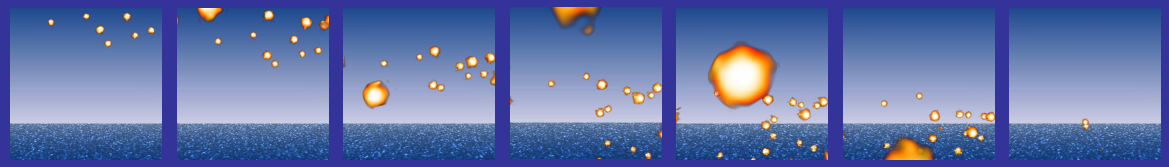
Document Version
Publisher's PDF, also known as Version of record

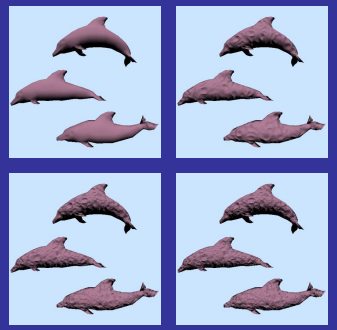
Better Faster Noise with the GPU

Geoff Wyvill
University of Otago
geoff@otago.ac.nz

Jeppe Revall Frisvad
Technical Univerity of Denmark
jrf@imm.dtu.dk



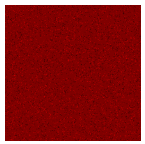


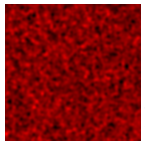


Sparse convolution noise [1]

Reference noise

A Gaussian filter over pseudo-randomly placed sources of random value.





Time: 342.6 sec.

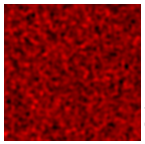
Smooth artitact-free noise

Lossless simplification

Choose a similar filter kernel with compact support.

Gaussian never zero

cubic compact support



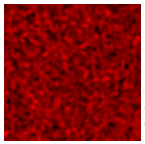
Time: 0.70 sec.

No loss of quality

GPU implementation

Point rendering

Render each source as a point with width corresponding to kernel size, and use blending. (Similar to spot noise [2].)



Time: 0.0056 sec.

Same as before

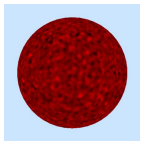
Increasing dimension

Problem: Too many sources (points).

CPU → Find volumes in space where we need sources

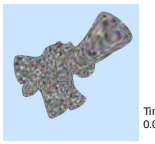
vertex shader → Use depth and perspective to find the size of each point.

fragment shader → Evaluate filter kernel



Time: 0.079 sec.

Solid noise



Time: 0.052 sec.

Elephant slice

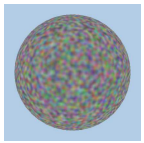
Optimizations

Hints:

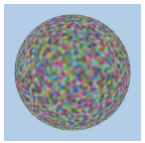
- Use a grid.
- Assign an index to each grid cell.
- Use the index to seed pseudo-random numbers for the sources in a cell.
- Render all sources in a cell by sending only cell index to the GPU. Create the sources (points) in a geometry shader.

Comparison

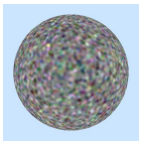
Classic Perlin | Perlin's simplex | Our noise



Time: 0.016 sec.
Regularity problems / Texture data needed



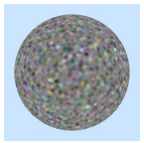
Time: 0.011 sec.



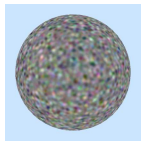
Time: 0.080 sec.
Artifact-free / Computed on the fly

Quality/quantity trade-off

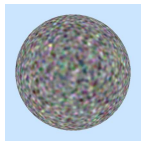
By adjusting the number N of sources per unit volume.



N = 10
Time: 0.030 sec.



N = 20
Time: 0.055 sec.



N = 30
Time: 0.080 sec.

References:

[1] LEWIS, J. P. 1989. Algorithms for solid noise synthesis. In *Computer Graphics (Proceedings of ACM SIGGRAPH '89)*, 23, 3, ACM, 263-270.

[2] VAN WIJK, J. J. 1991. Texture synthesis for data visualization. In *Computer Graphics (Proceedings of ACM SIGGRAPH '91)*, 25, 4, ACM, 309-318.

Timings done on an Centrino 2.13 GHz laptop with a Nvidea Quadro FX Go1400 GPU