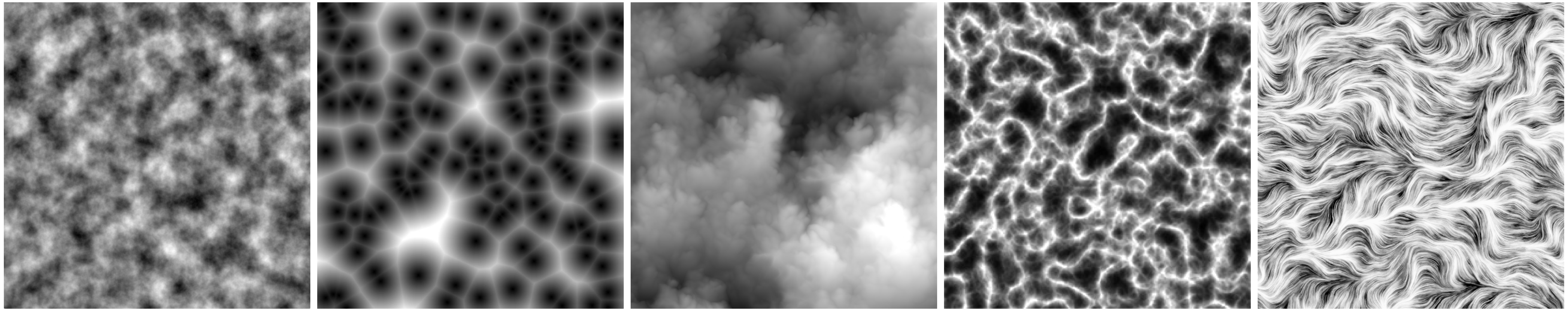
Procedural Animation

Noise function - Perlin Noise

Procedural Noise

What is a spatial/temporal procedural noise:

- Function with visible structure/pattern (limited frequency bandwith)
- Without visible periodicity
- Deterministic (Same result from a given input)



Examples of 2D procedural textures

Create procedural noise

Ex. Continous function $f(x)$ with limited frequency.

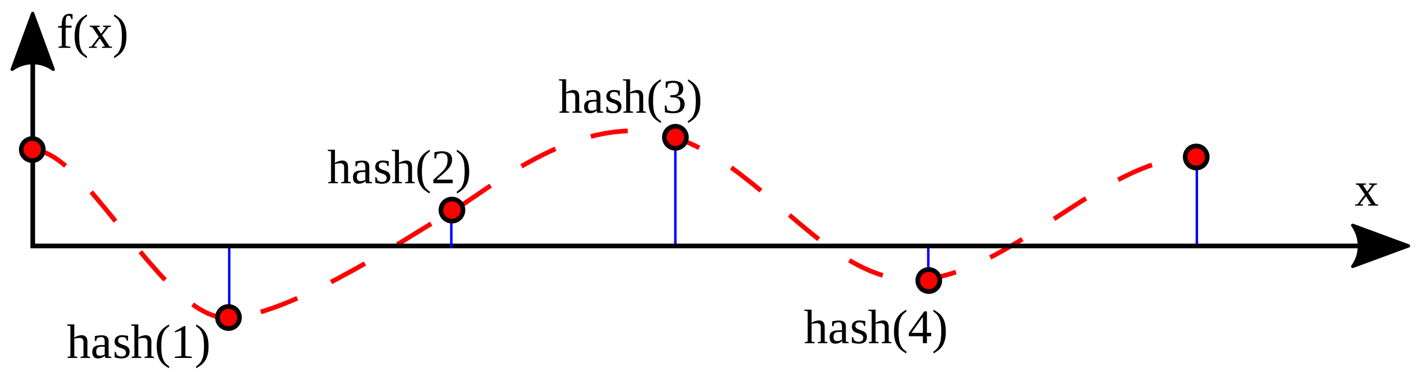
For integer value: $f(n)$ = pseudo-random, determinic, value

ex. Hash Function: float hash(float n) { return fract(sin(n) * 1e4); }

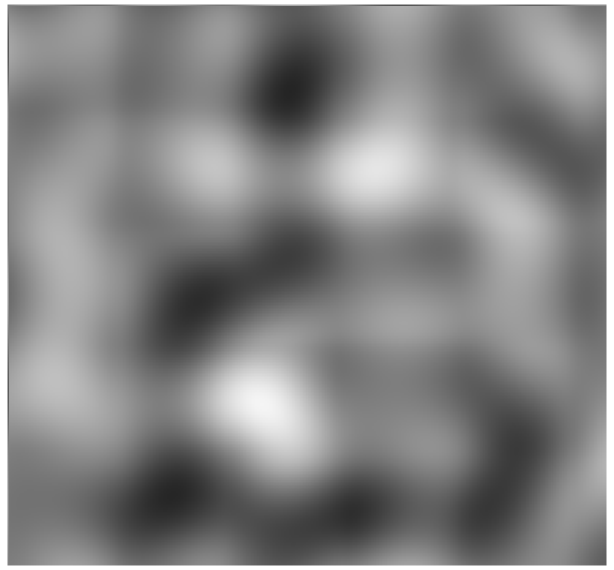
Interpolate using smooth curve between each integer value (Smooth step, cubic polynomials, etc.)

Properties:

- Continuous function
- Looks non-periodic
- Frequency limited around 1



Can be computed in 2D, 3D, etc.



Simple algorithm - ex. in 1D

For a given x, n =floor(x)

- Evaluate hash function at n, $n+1$

($(n \pm i)$ for further sampling)

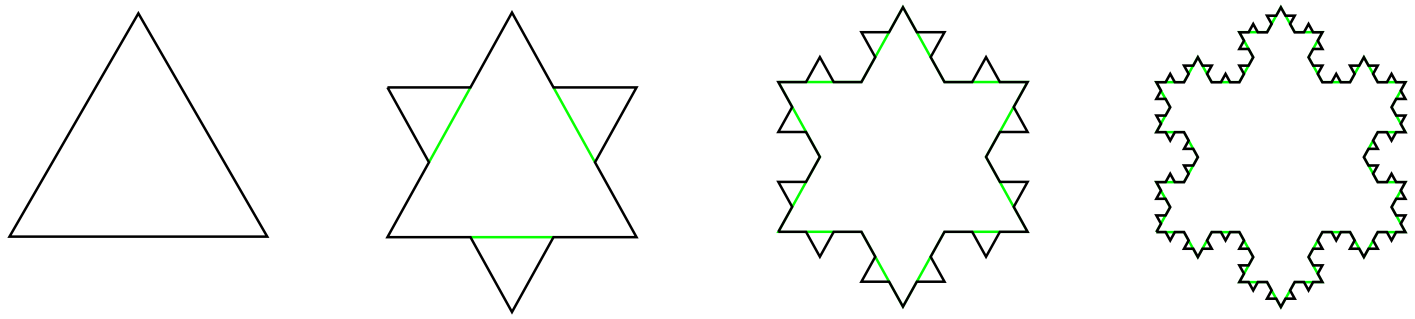
- Compute interpolated value $f(x)$ at position x

Add high frequency details: Notion of Fractal

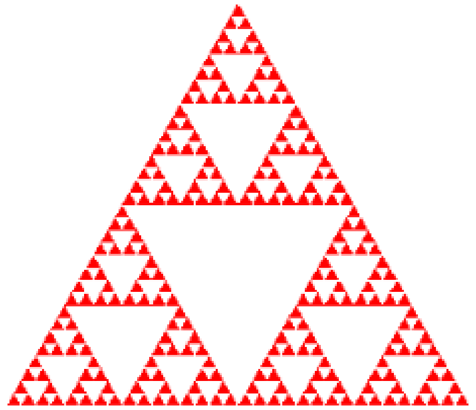
Idea: Recursively add self-similar details

Simple rule ⇒ complex shapes

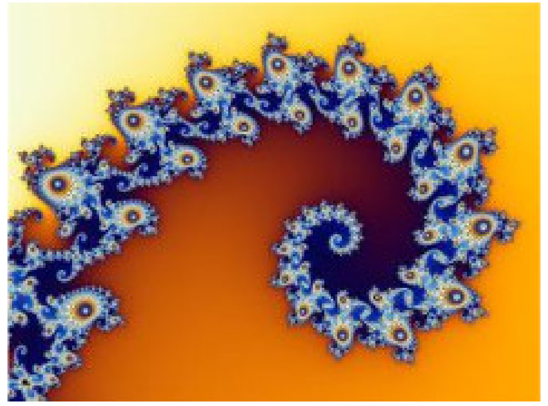
May look like complex natural details



Koch Snowflake



Sierpinski triangle, Shell:Oliva Porphyria



Standard fractals (Mandelbrots, Sierpinski, Newtons's root, etc) are very hard to control.

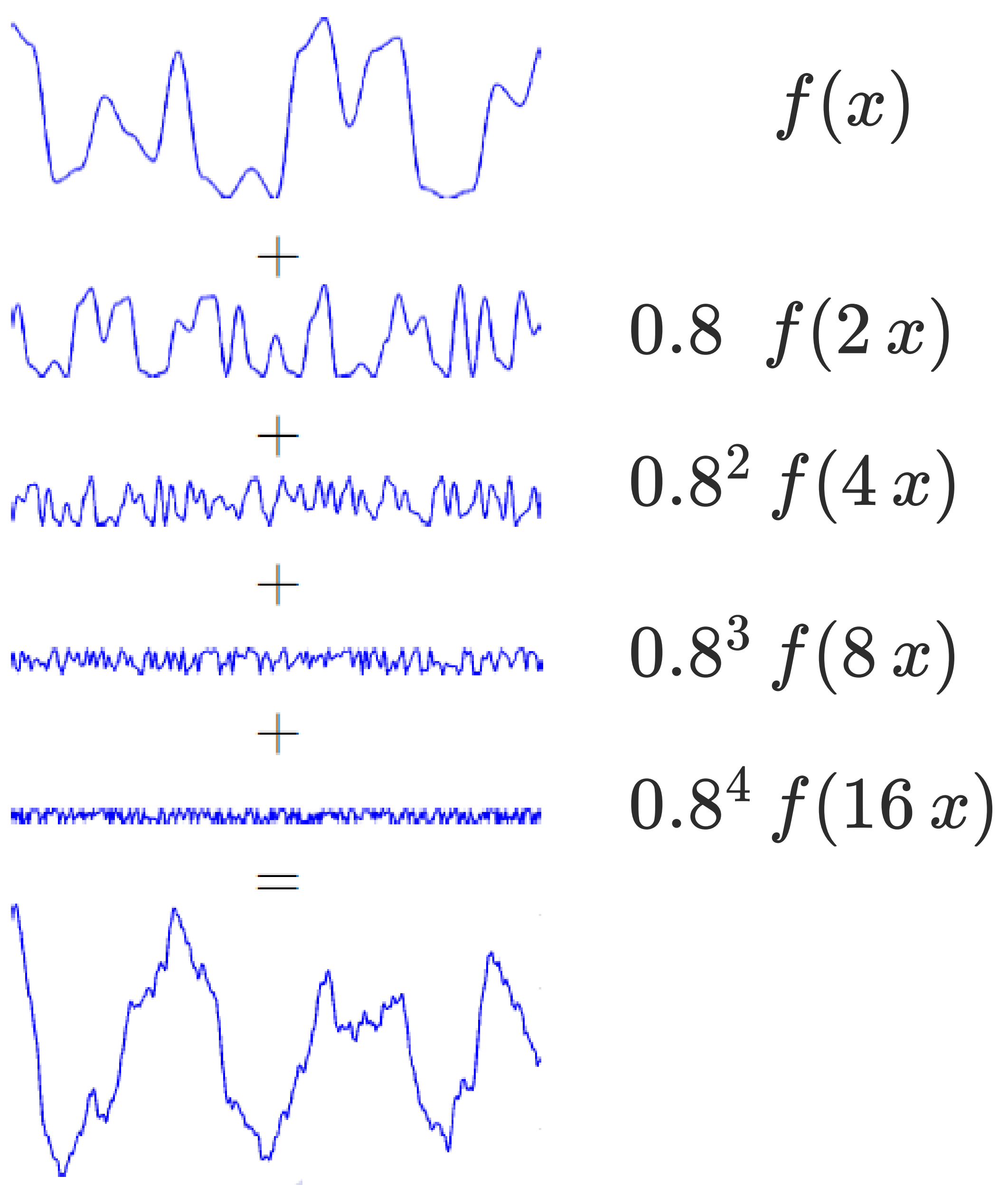
Not often used directly in CG.

Perlin Noise

First procedural noise proposed in

[Ken Perlin, An Image Synthesizer. SIGGRAPH 85]

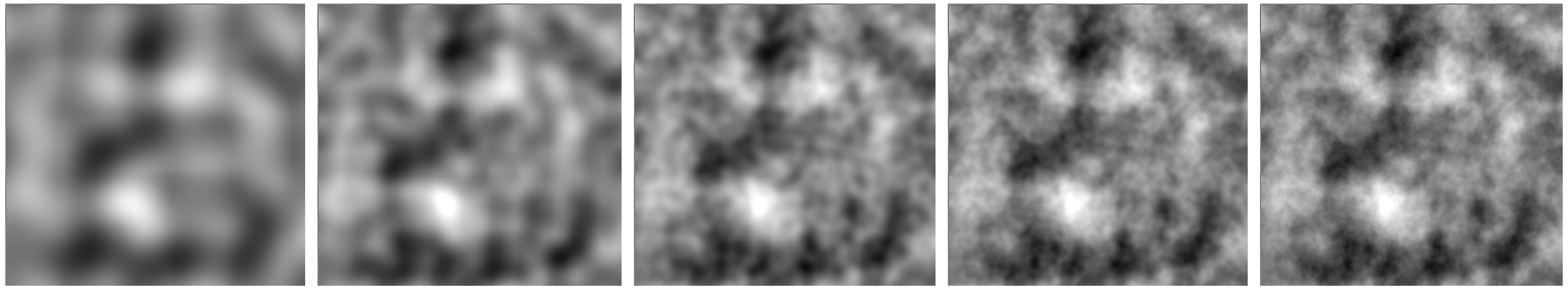
Idea: Sum over pseudo-random function with increasing frequencies and decreasing magnitude



f : Smooth pseudo-random function

$$P(x) = \sum_{k=0}^{N} \alpha^k f(\omega^k\ x)$$

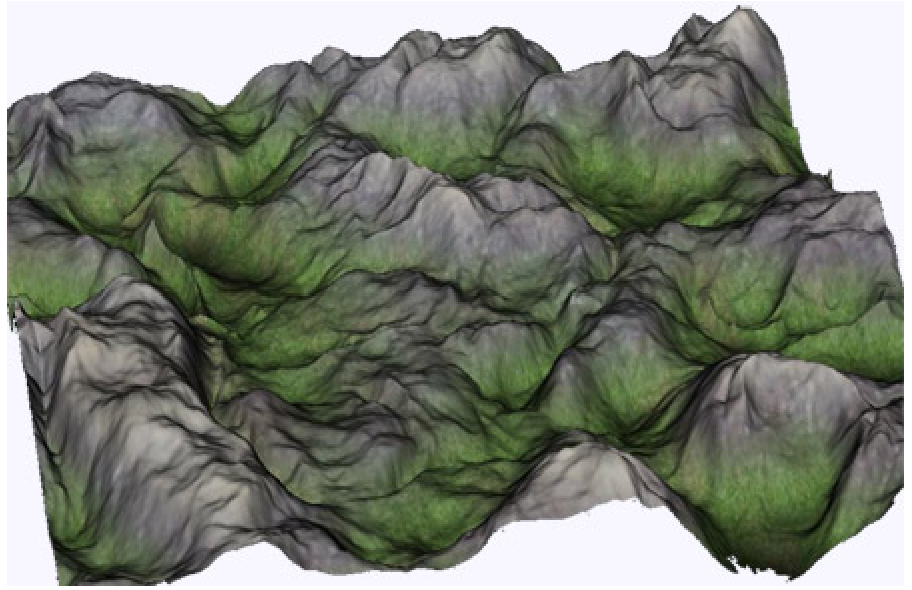
- N number of Octave
- α persistency ($1/\alpha$ attenuation)
- ω frequency gain



Perlin noise usage

Direct use: $z = P(x, y)$

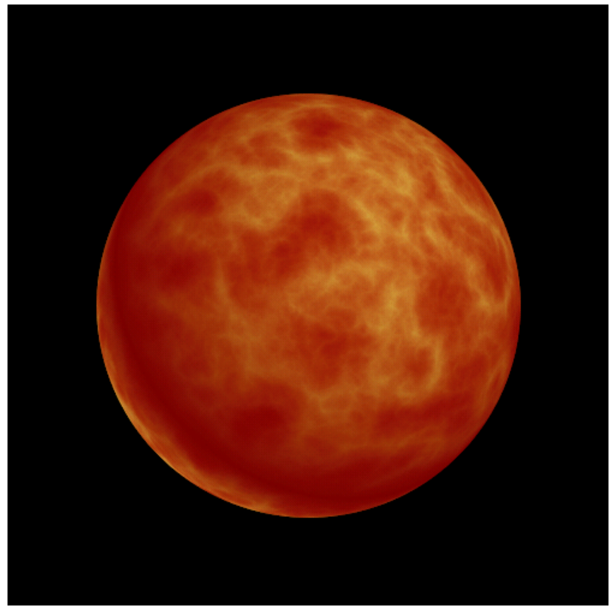
Mountain-looking terrain

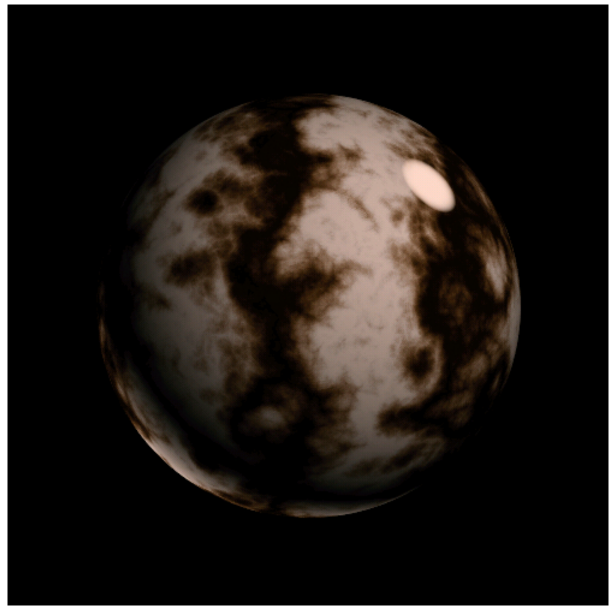


Material texture

Ridge effect: $\sum_k \alpha^k |f(\omega^k x)|$

Marble effect: $\sin\left(x + \sum_k \alpha^k |f(\omega^k x)|\right)$

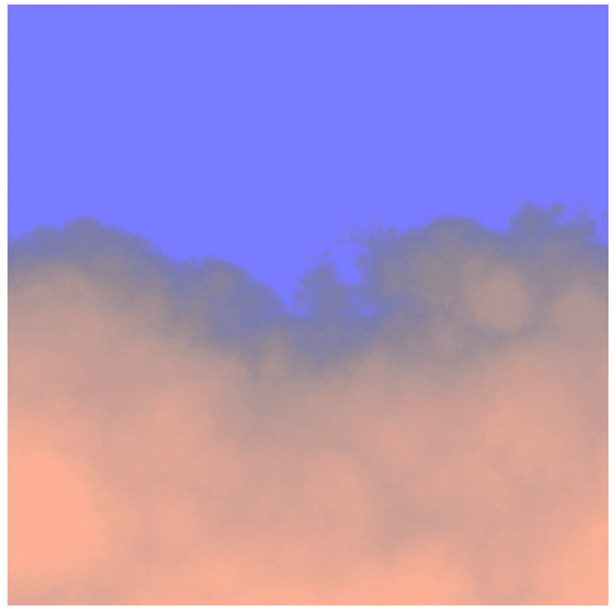




Animated textures

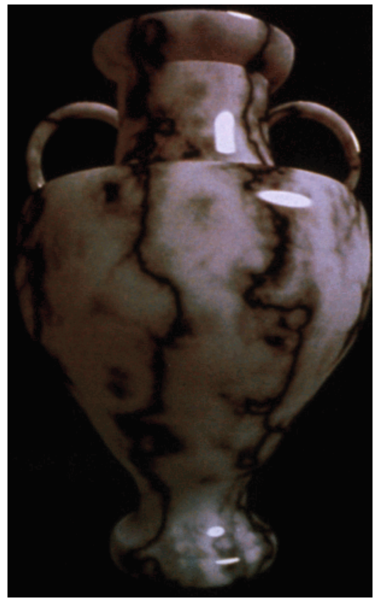
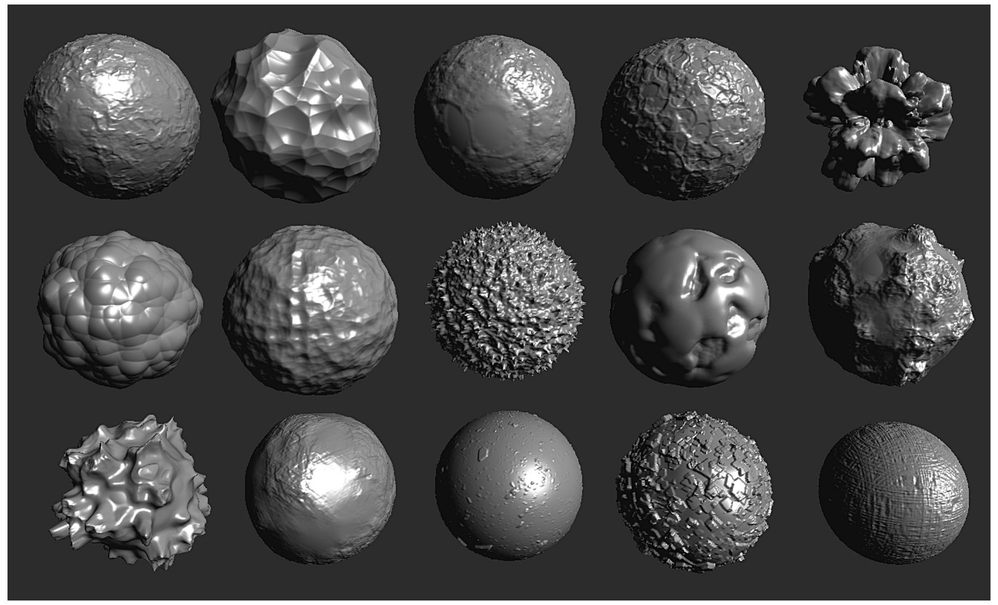
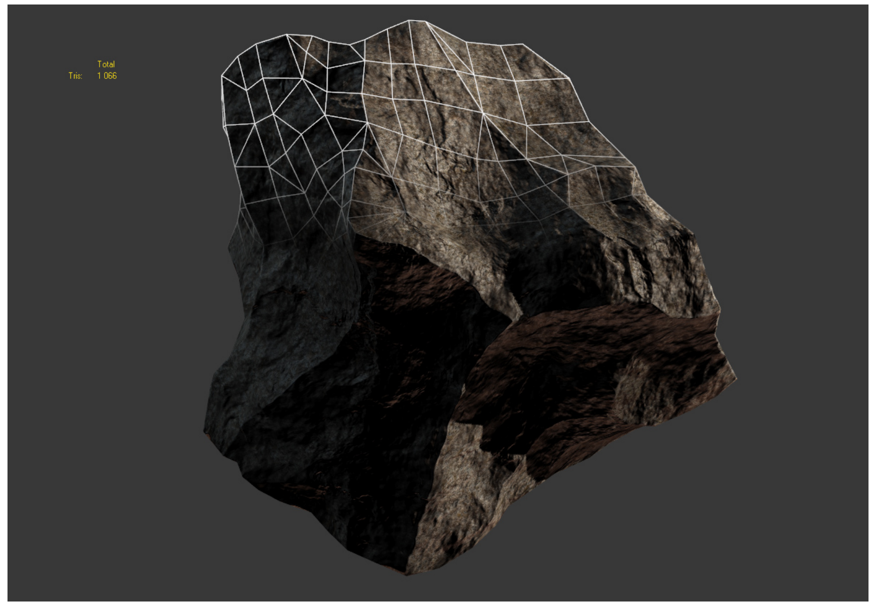
Translation: $P(x, y + t)$

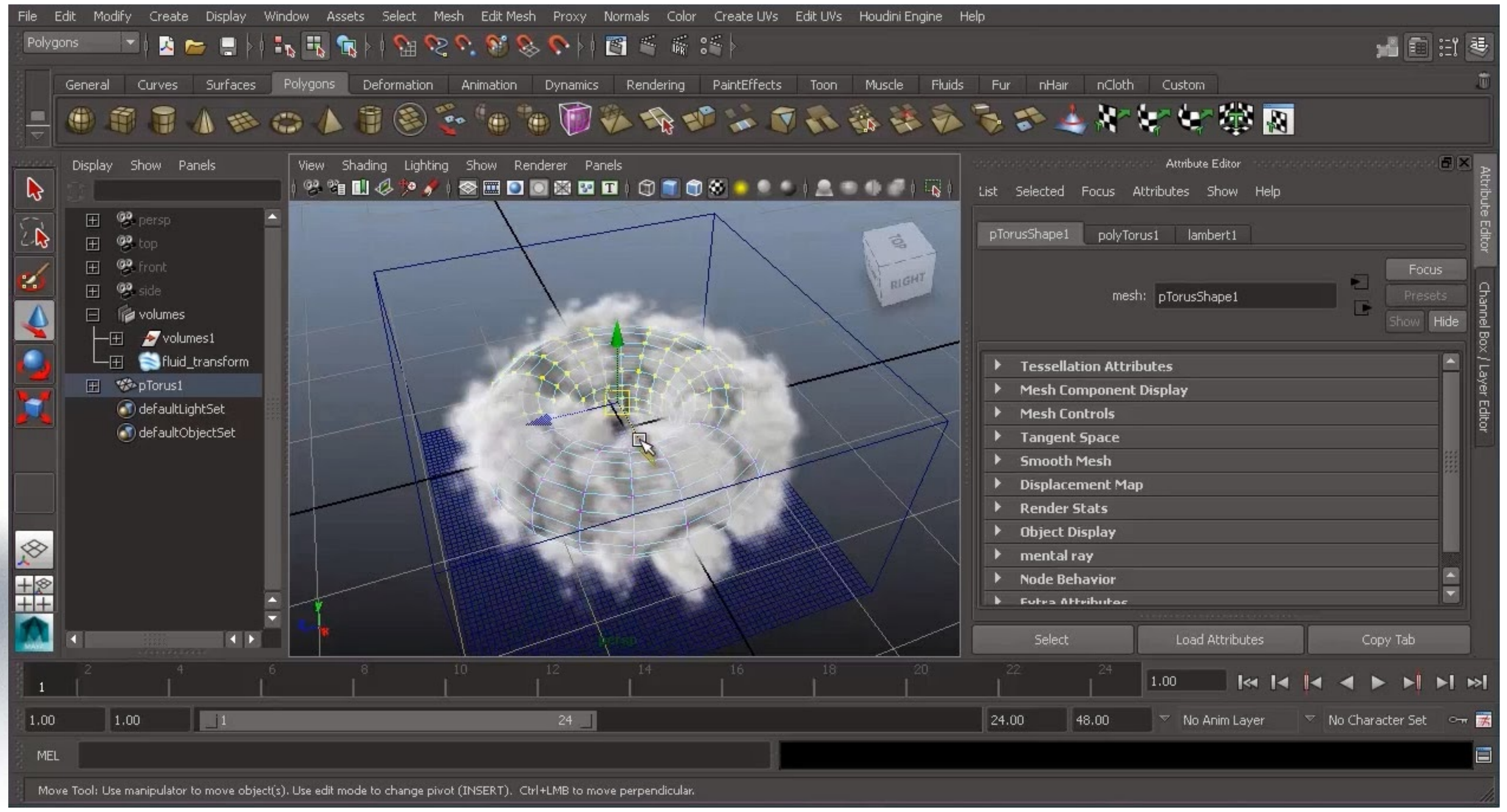
Smooth evolution: $P(x, y, t)$



Perlin noise applications

In almost every complex shapes ...

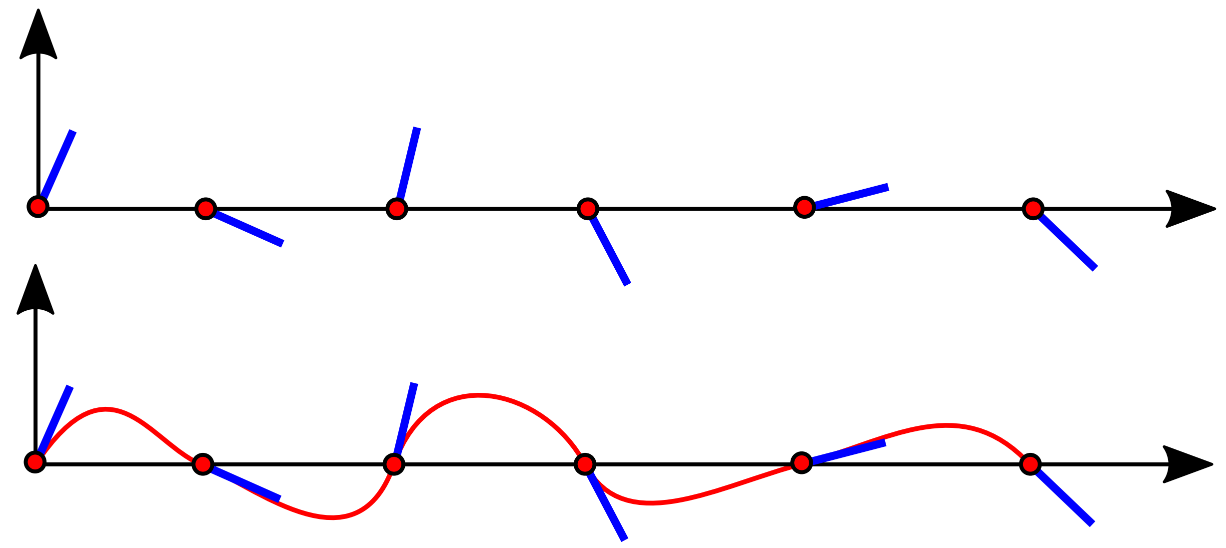
Look at Shader Toy + Noise example

Perlin Noise - Improvement

Gradient Noise

Use pseudo-random Gradients instead of Positions

Improved frequency control: Oscillate at period 1



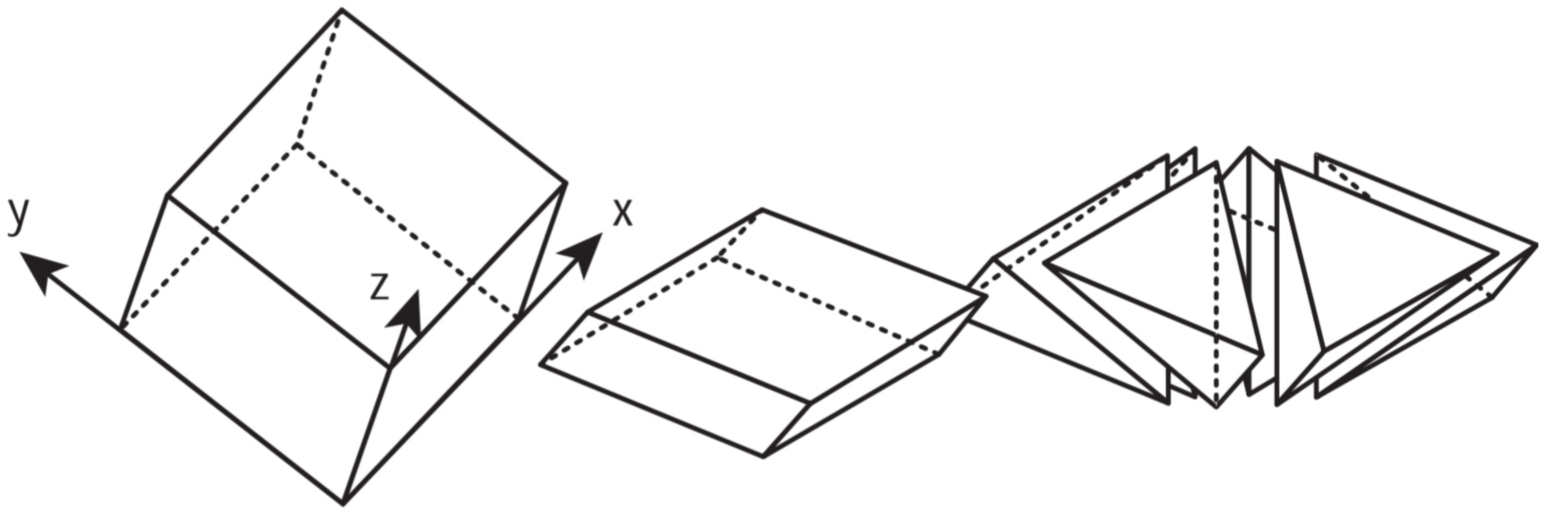
Simplex Noise

Simplex-based interpolation instead of grid interpolation

Avoids grid-direction artifact

Faster for high dimensions

[Stefan Gustavson Simplex noise demystified]



To go beyond:

[A Lagae et al., STAR in Procedural Noise Functions. EG 2010]

Perlin Noise Terrain

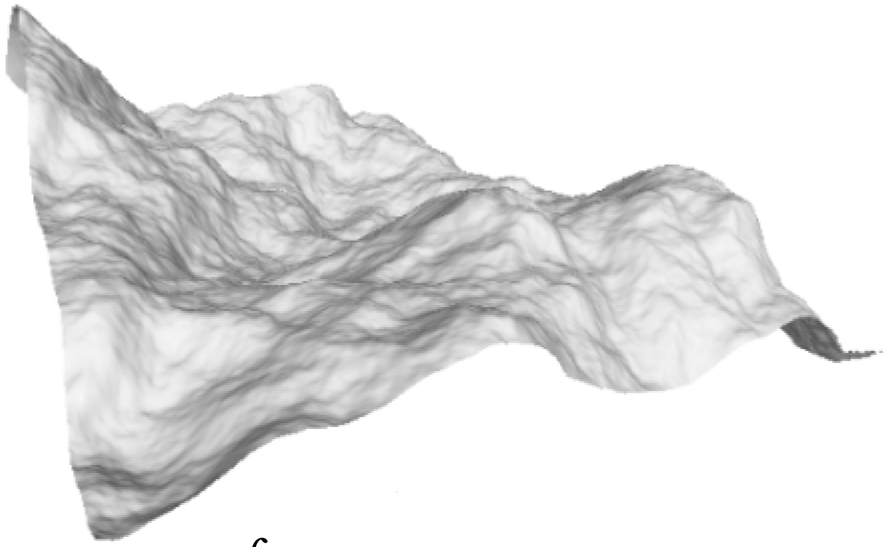
Consider the surface S defined as

$$S(u,v) = \begin{cases} x(u,v) = u \\ y(u,v) = v \\ z(u,v) = h\,P(s(u+o), s(v+o)) \end{cases}$$

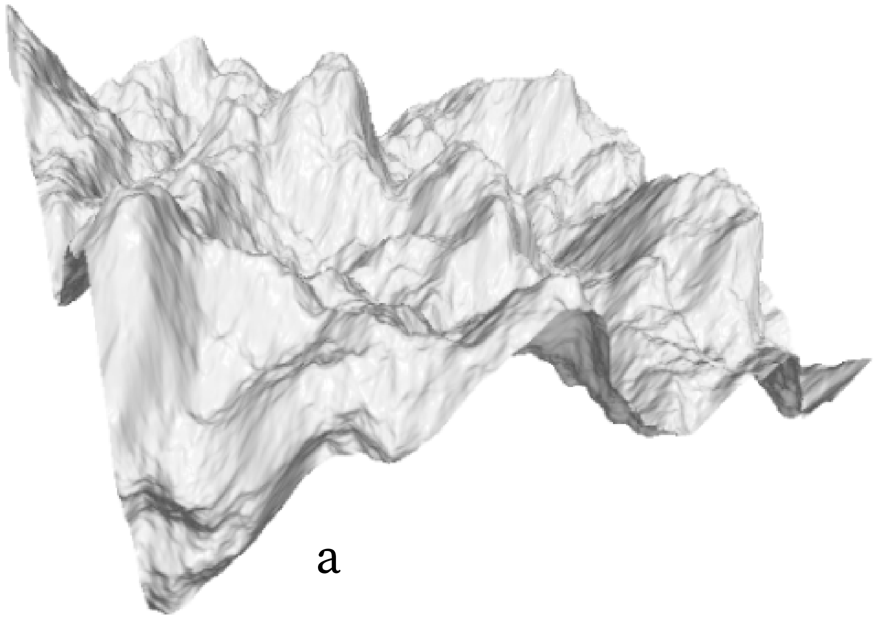
The perlin noise

$$P(u,v) = \sum_{k=0}^{N} \alpha^k \, f(2^k u, 2^k v)$$

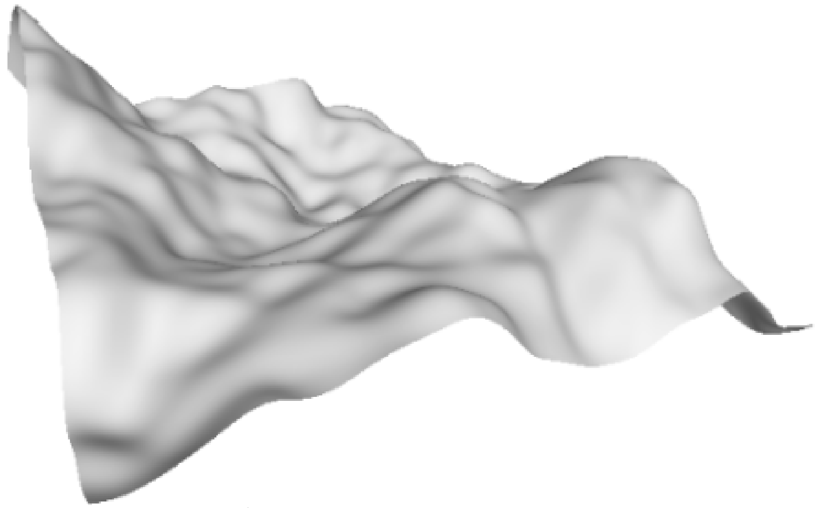
$N = 9$
$\alpha = 0.4$
$h = 0.3$
$s = 1$
$o = 0$



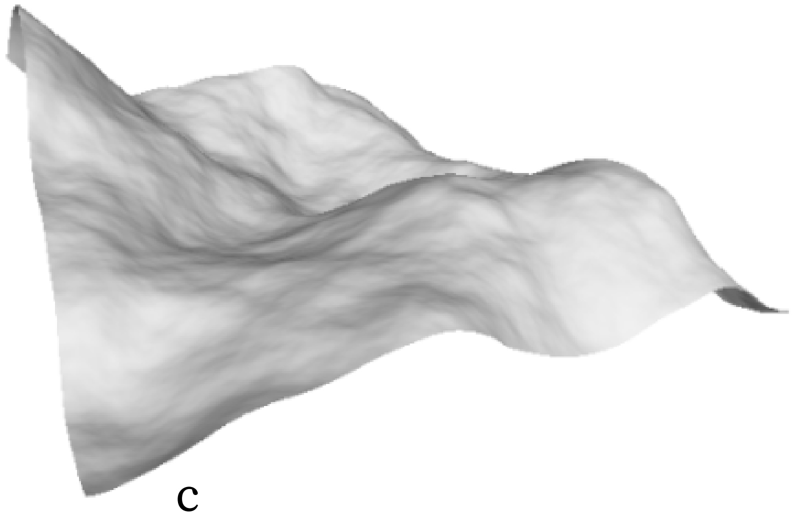
reference



a

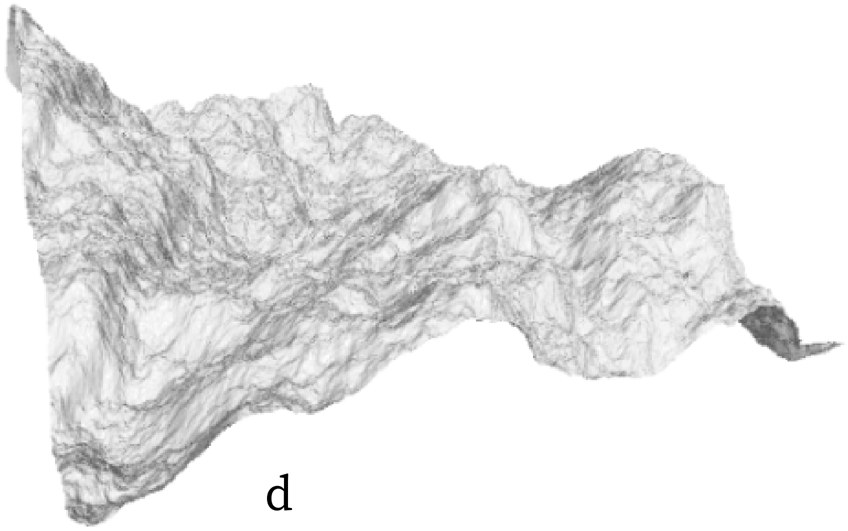


b

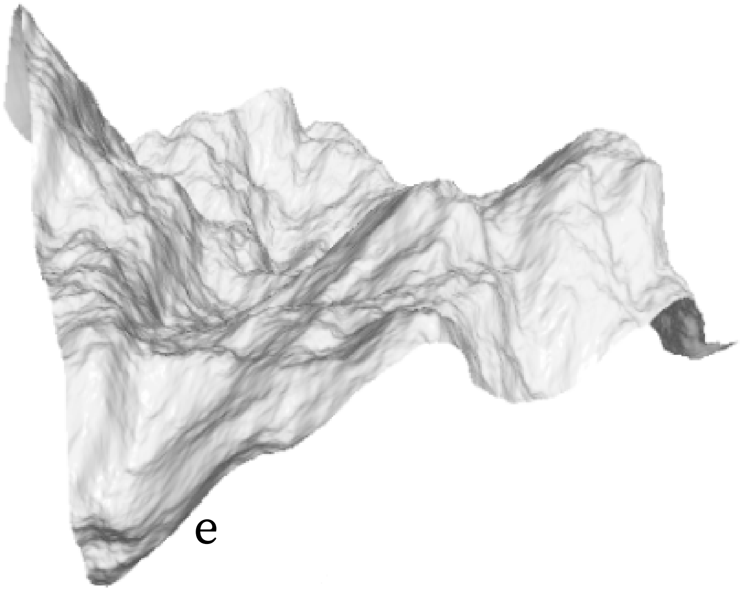


c

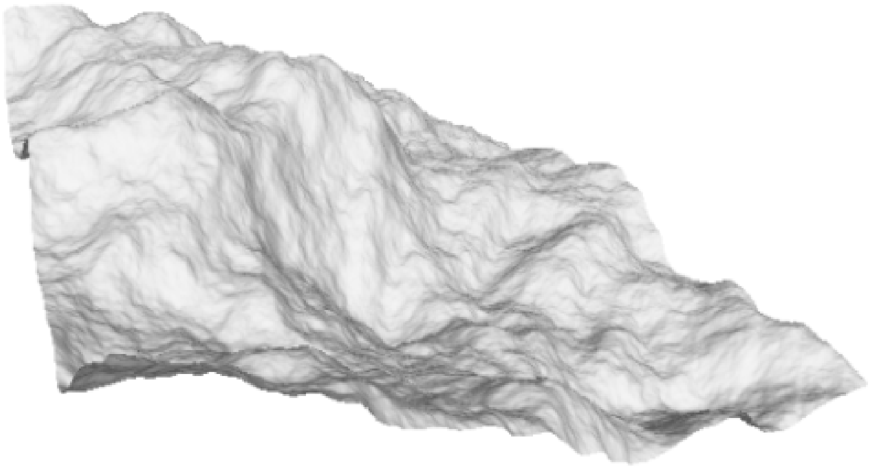
Q. Which parameters correspond to (a,b,c,d,e,f) terrains?



d

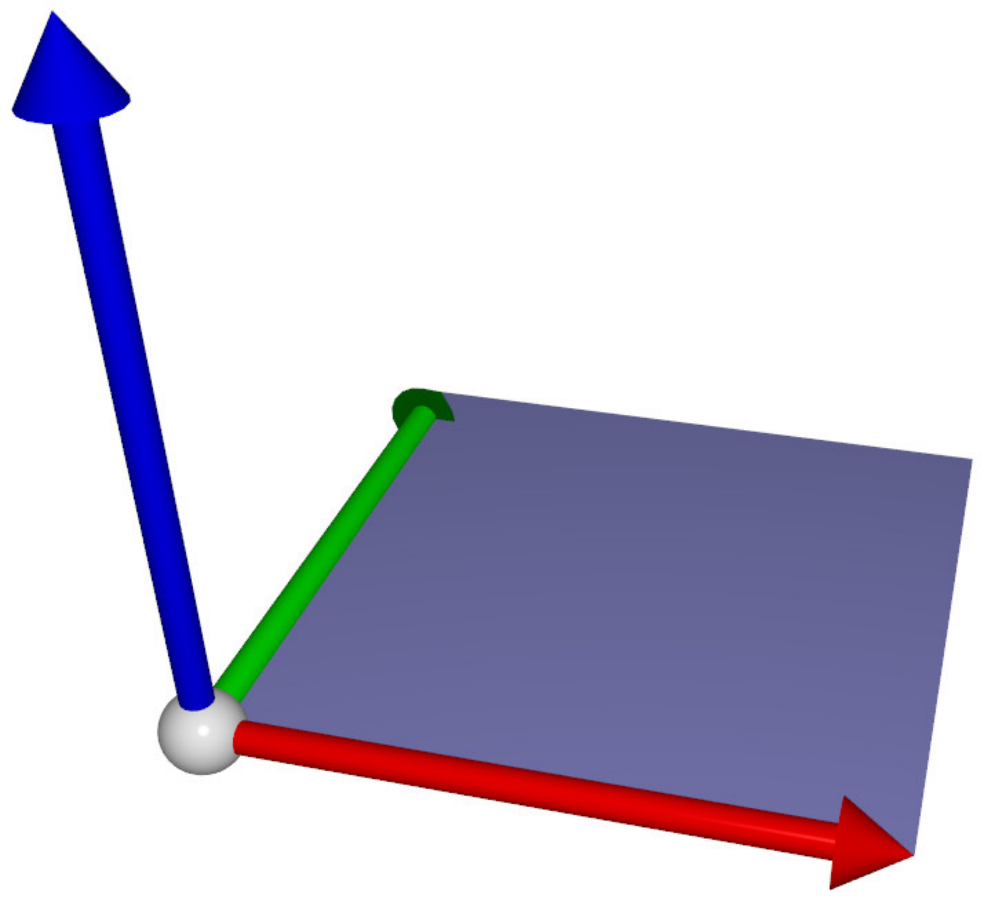


e



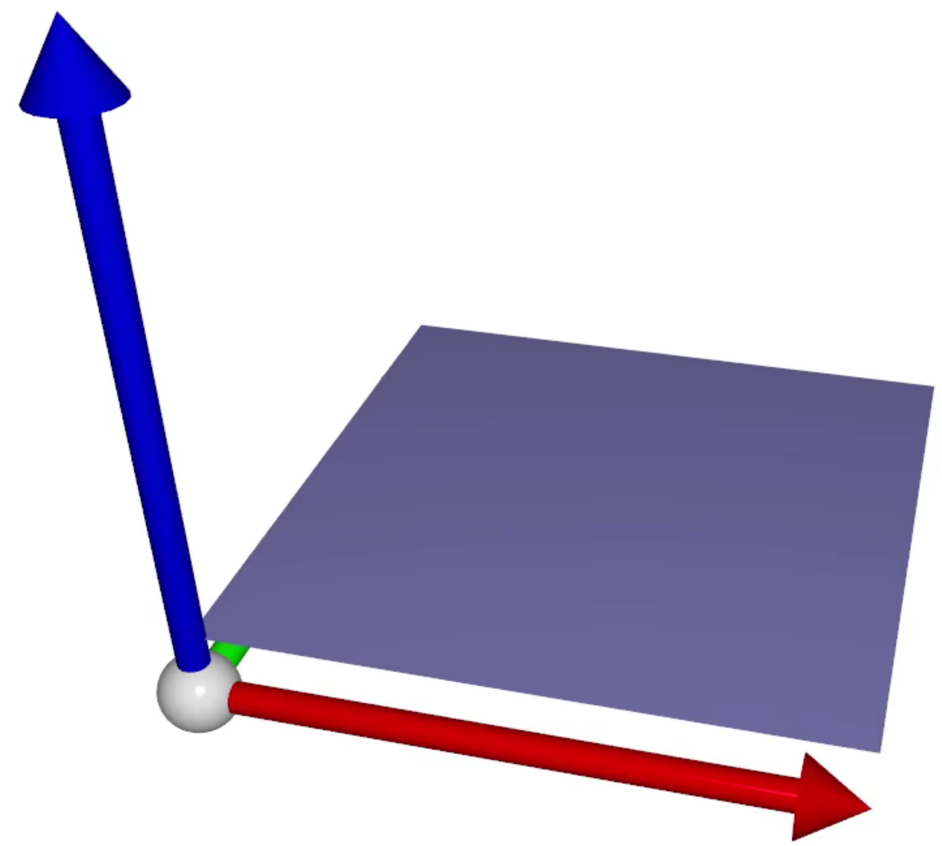
f

Perlin Noise - Animation



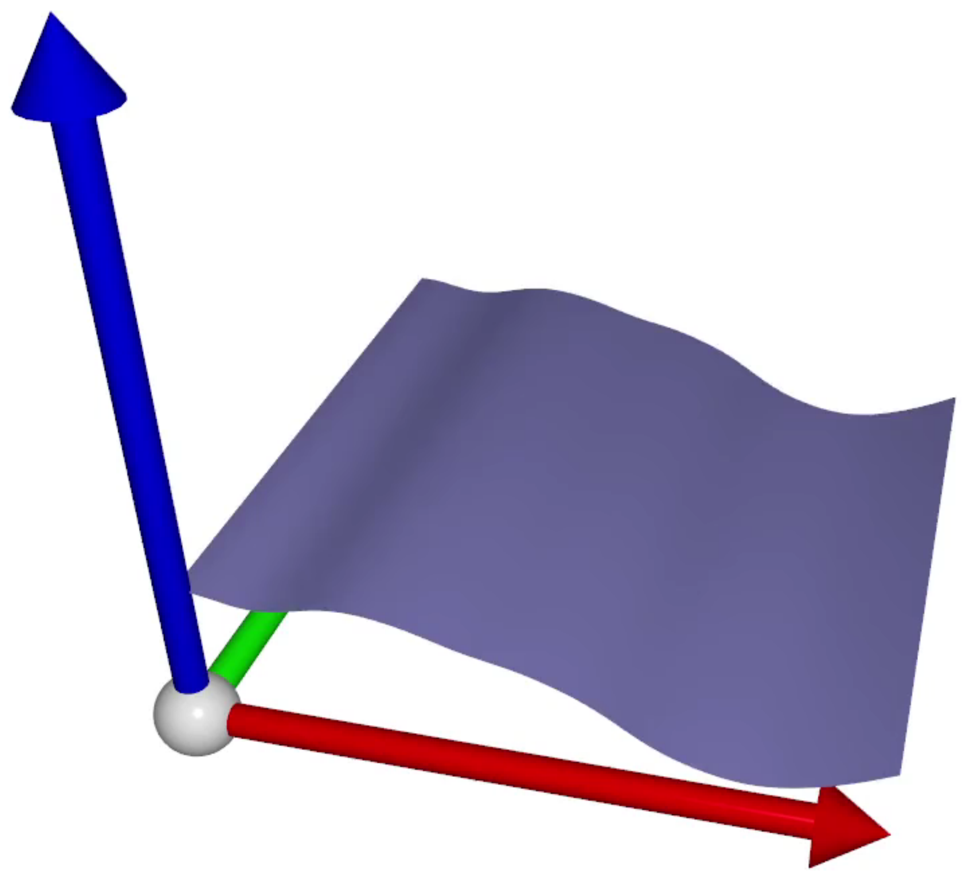
Reference surface function

$$(u, v) \in [0, 1]^2 \,,\ S(u, v) = (u, v, 0)$$



Perlin noise can depends on time

$$S(u, v, t) = (u, v, P(t))$$



Depends on space and time

$$S(u, v, t) = (u, v, P(u, t))$$

Perlin Noise - Animation

Reference surface function: $(u,v) \in [0,1]^2$, $f(u,v) = (u,v,0)$

Q. How generate the following animations ? $S(u,v,t) = ...$