Procedural Animation

Particles Trajectories

Particles Systems History

One of the first animated model in CG

[Particle systems - A Technique for Modeling a Class of Fuzzy Objects, William T. Reeves, Lucasfilm, 1983 (Star Trek II)]

[Karl Sims, Particle dreams, 1988]

Example of particle system

Free fall of spheres under gravity

- Geometrical representation of each particle: sphere
- Equation of motion $p(t) = \frac{1}{2}gt^2 + v_0 t + p_0$

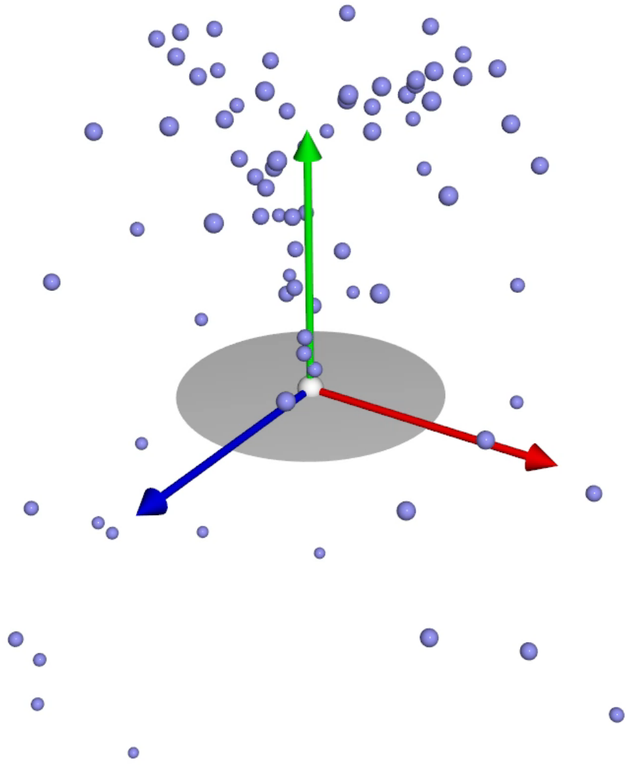
- Initial position and speed may be placed at random position
- Each particle may have a different life time

Q. *What are the parameters used for* p_0 *and* v_0 *in this example ?*

$p_0 = \ldots$

$v_0 = \ldots$

Note: Coordinate system (x, y, z) *=(red, green, blue).*



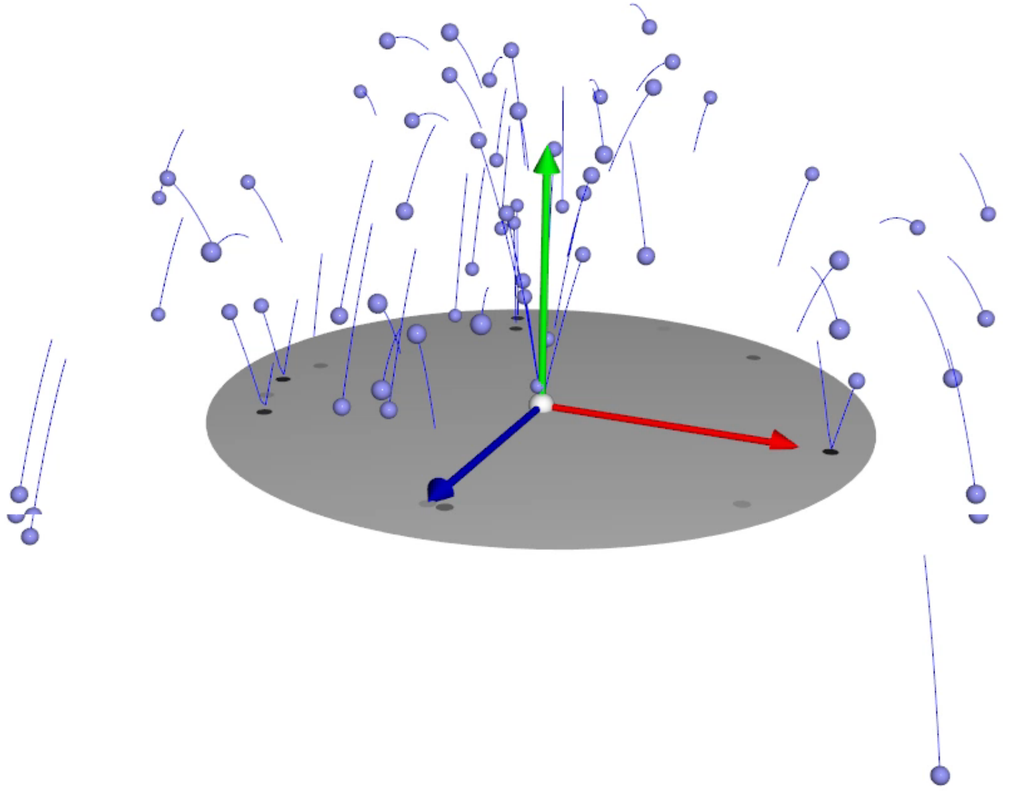
Bouncing spheres

Q. What is the equation of motion ?
(taking into account the bouncing)
$p(t) = \ldots$

Help:

- *Consider a particle emited at time* $t = 0$
- *At what time* t_i*, the particle touches the ground ?*
- *What is the new speed after impact ?*
- *Express the complete equation of trajectory*

(piecewise definition)



Bonus: How the "impact shadow" effect is computed ?

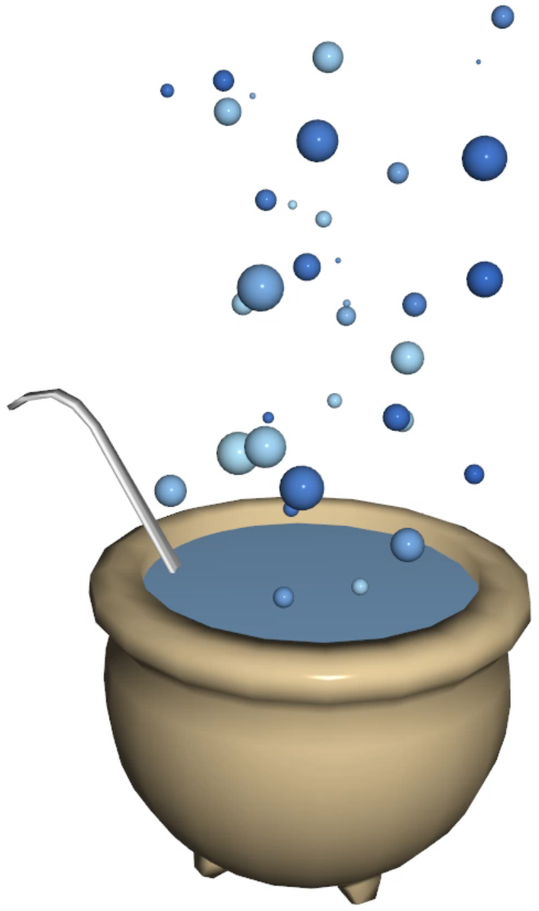
Arbitrary motions

Trajectories are not restricted to physics-based equations

Q. *What are the parameters (and initial value) associated to each particle ?*

- position = ...
- radius = rand([0,0.1])
- ...

- *What is the equation of motion of a particle ?*
  - $p(t) = \ldots$



Billboards, Impostors, Sprites

Particles can be displayed as small images/thumbnails (instead of simple sphere)

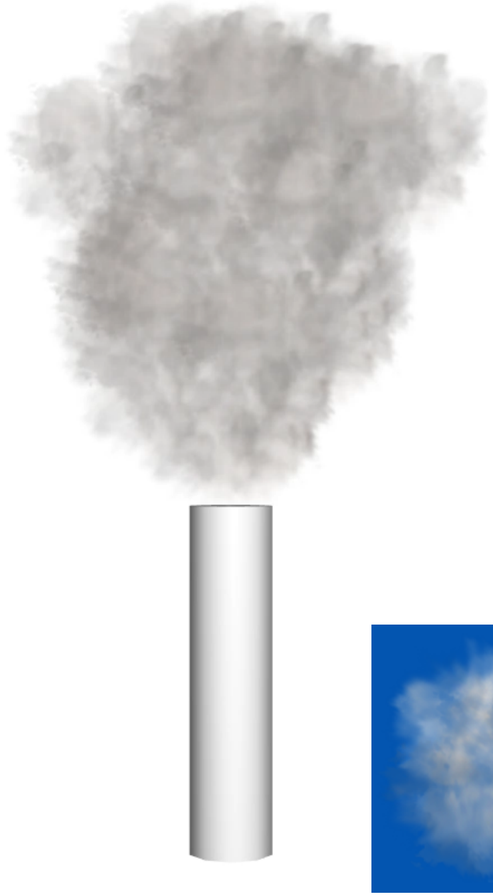
In practice

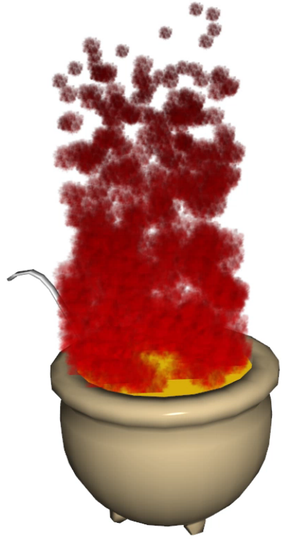
- Each particle is displayed as a quadrangle
- A texture is mapped on the quad

- The texture can contain transparency : quad geometry is invisible

  Fake a more complex geometry

- The quad can be facing the camera at all time (billboard)
- The texture can be animated (sprite)
- Texture can be adapted to the point of view (impostors)

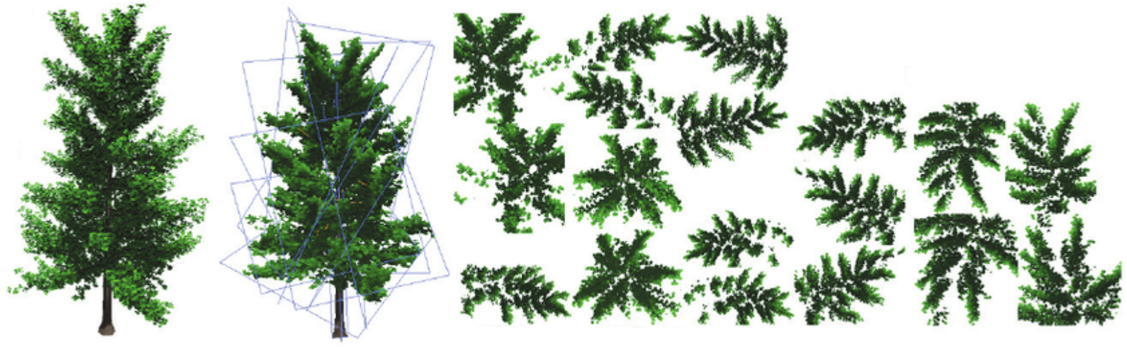


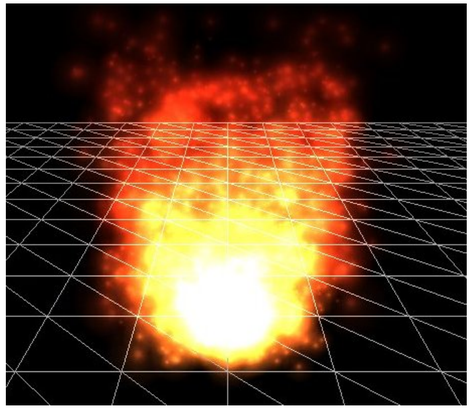


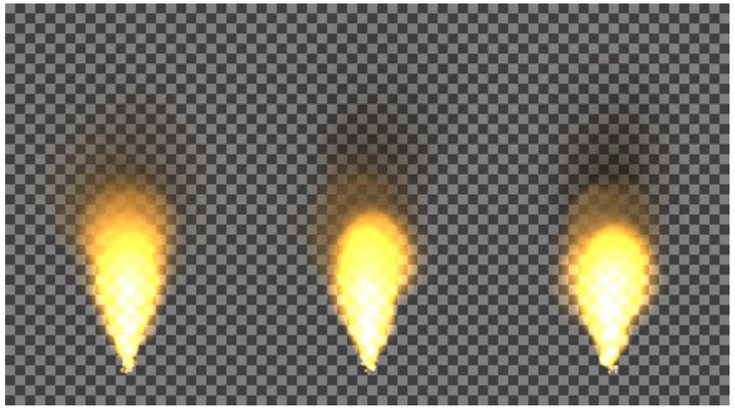
Usage of billboards

Large use of billboards for complex models

Vegetation, fire, smoke, etc.







Use case in production

How to model the *horse heads made of water* ?

The Lord of the Rings

Use case in production

How to model the *horse heads made of water* ?

The Lord of the Rings

Use case in production

Full Making-Of

The Lord of the Rings

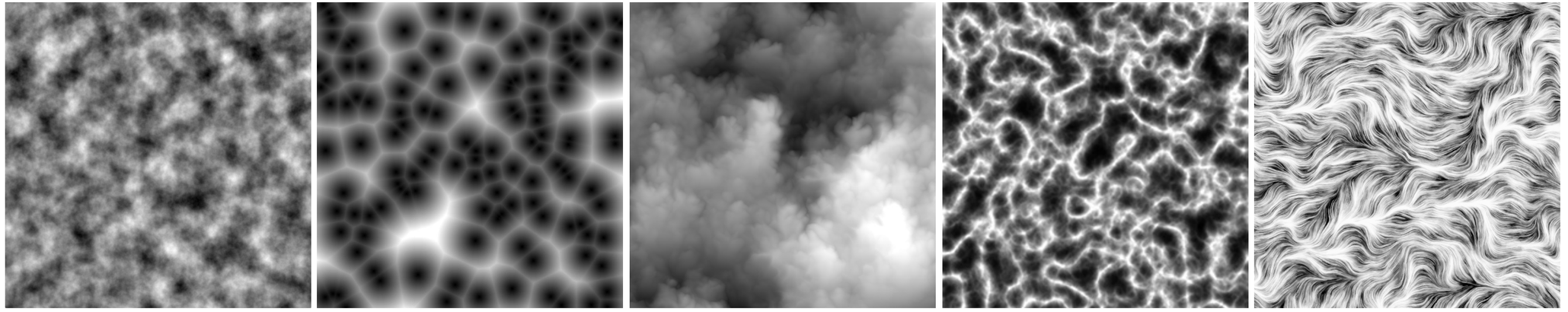
Procedural Animation

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.

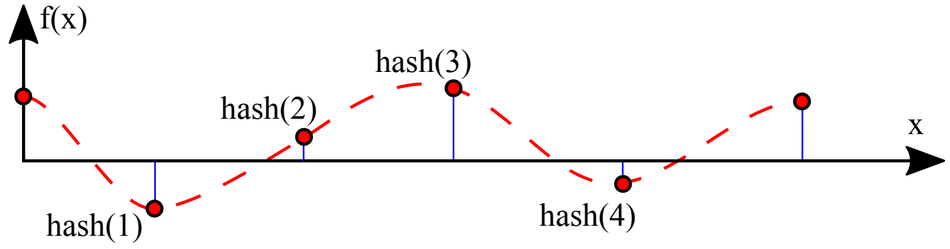
For integer value: $f(n) =$ pseudo-random, determinic, value

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

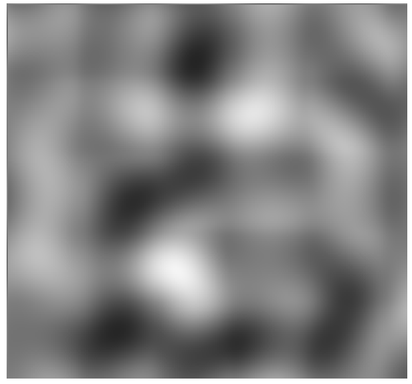
Interpolate using smooth curve between each integer value (Smooth step, cubic polynomials, etc.)

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

Perlin Noise

First procedural noise proposed in

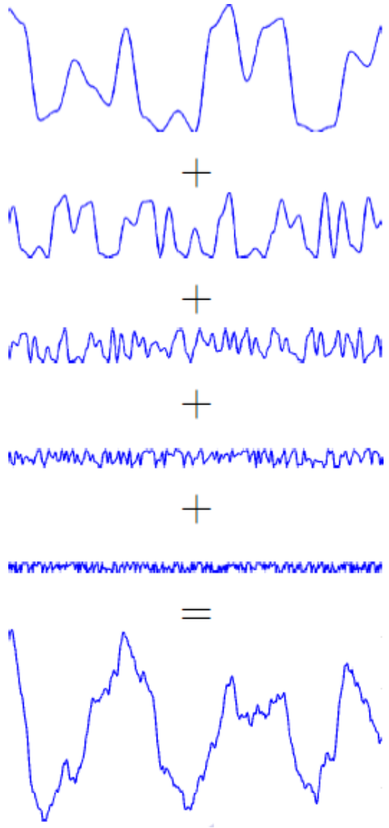
[Ken Perlin, An Image Synthesizer. SIGGRAPH 85]

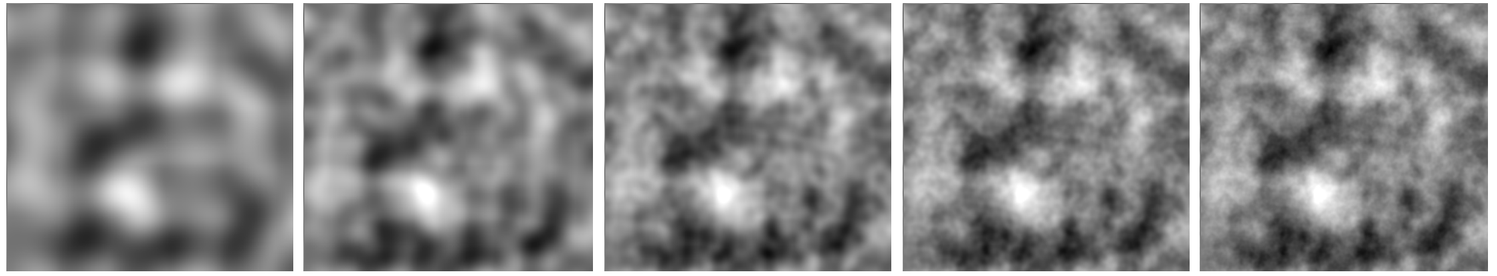
Idea: Sum over pseudo-random function with increasing frequencies and decreasing magnitude

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

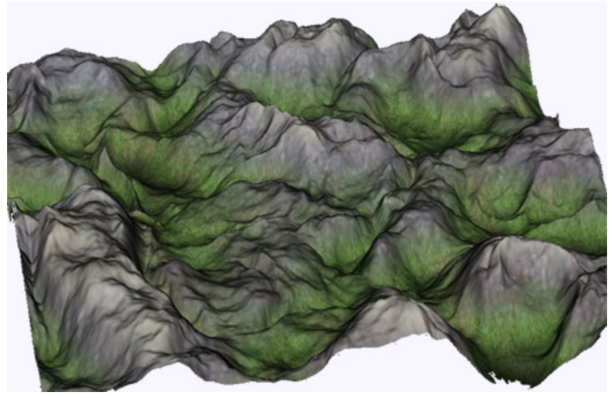




Perlin noise usage

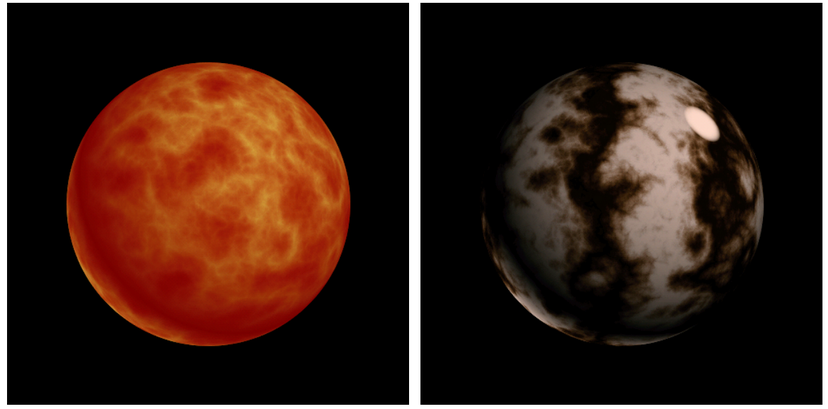
Direct use: $z = P(x, y)$

- Mountain-looking terrain



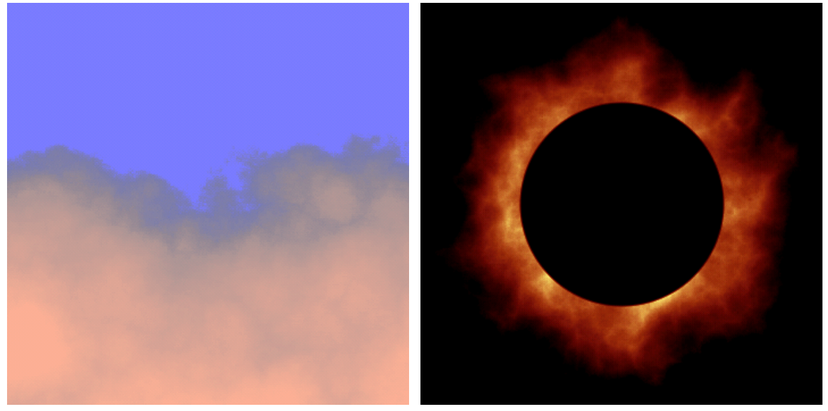
Material texture

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



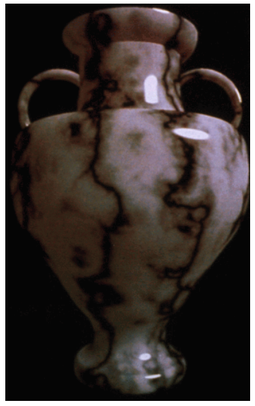
Animated textures

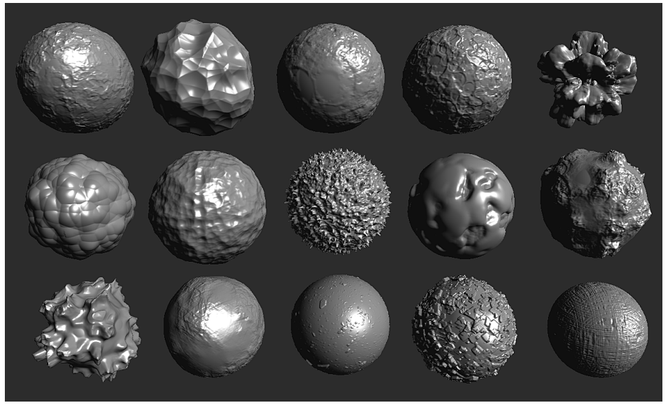
- Translation: $P(x, y + t)$
- Smooth evolution: $P(x, y, t)$

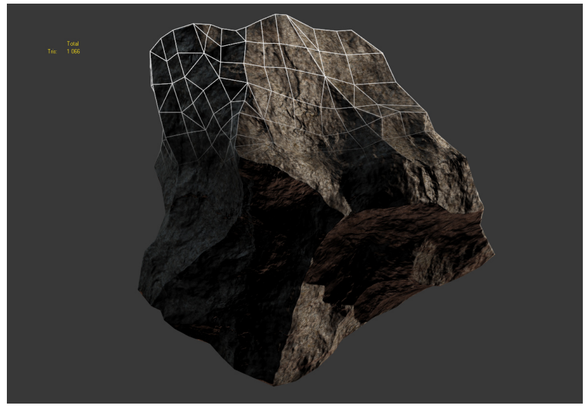


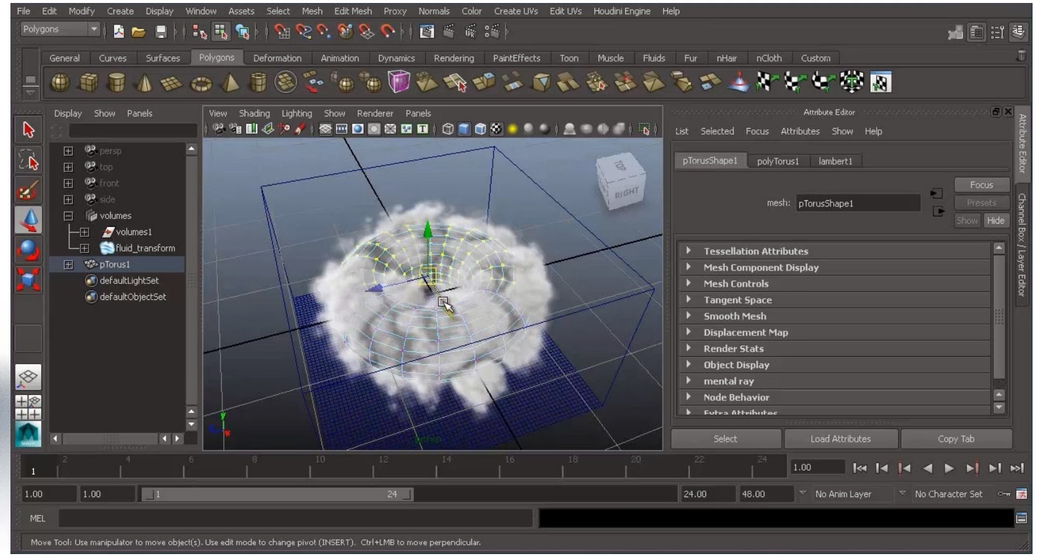
Perlin noise applications

In almost every complex shapes ...









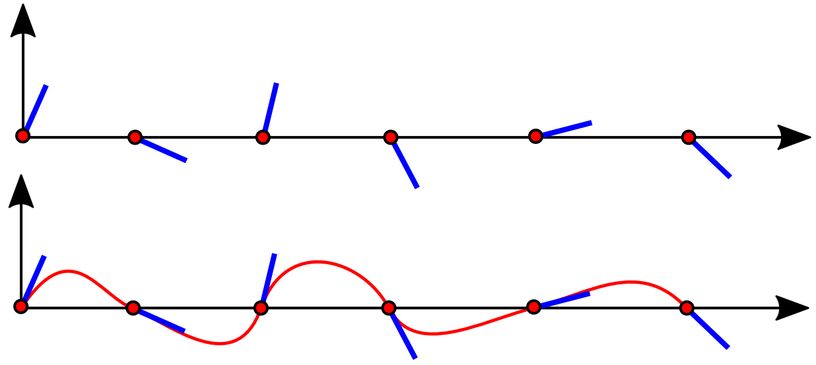
Look at Shader Toy + Noise example

Perlin Noise - Improvement

Gradient Noise

Use pseudo-random Gradients instead of Positions

Improved frequency control: Oscillate at period 1



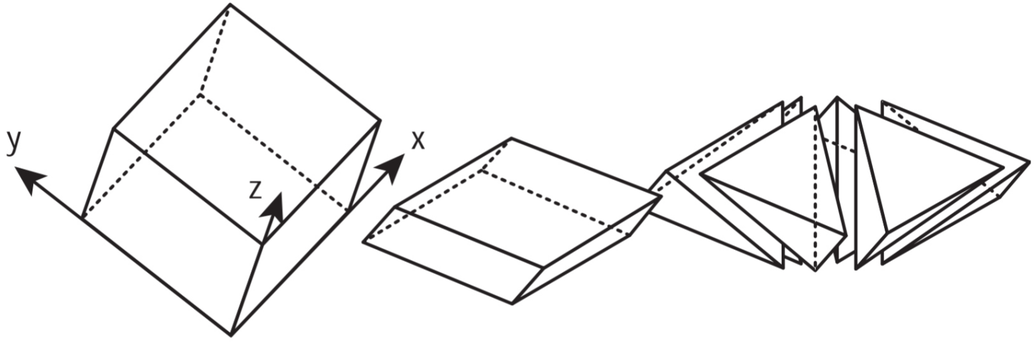
Simplex Noise

Simplex-based interpolation instead of grid interpolation

Avoids grid-direction artifact

Faster for high dimensions

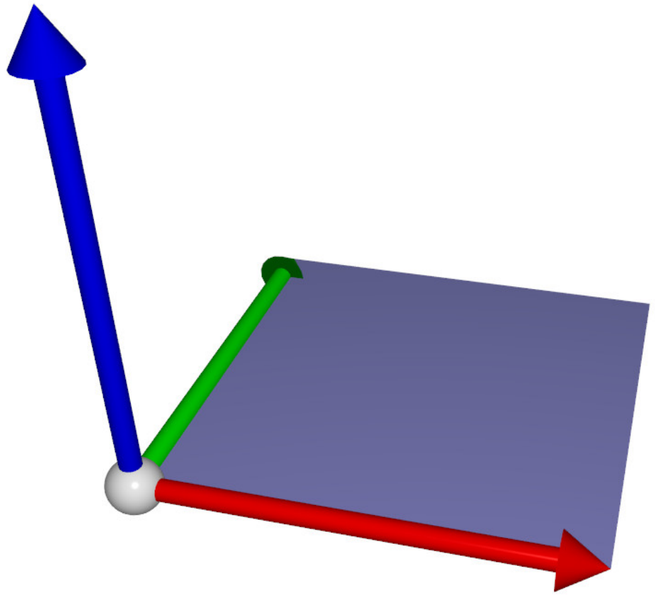
[Stefan Gustavson Simplex noise demystified]



To go beyond:

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

Perlin Noise - Animation



Reference surface function

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

Perlin noise can depends on time

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

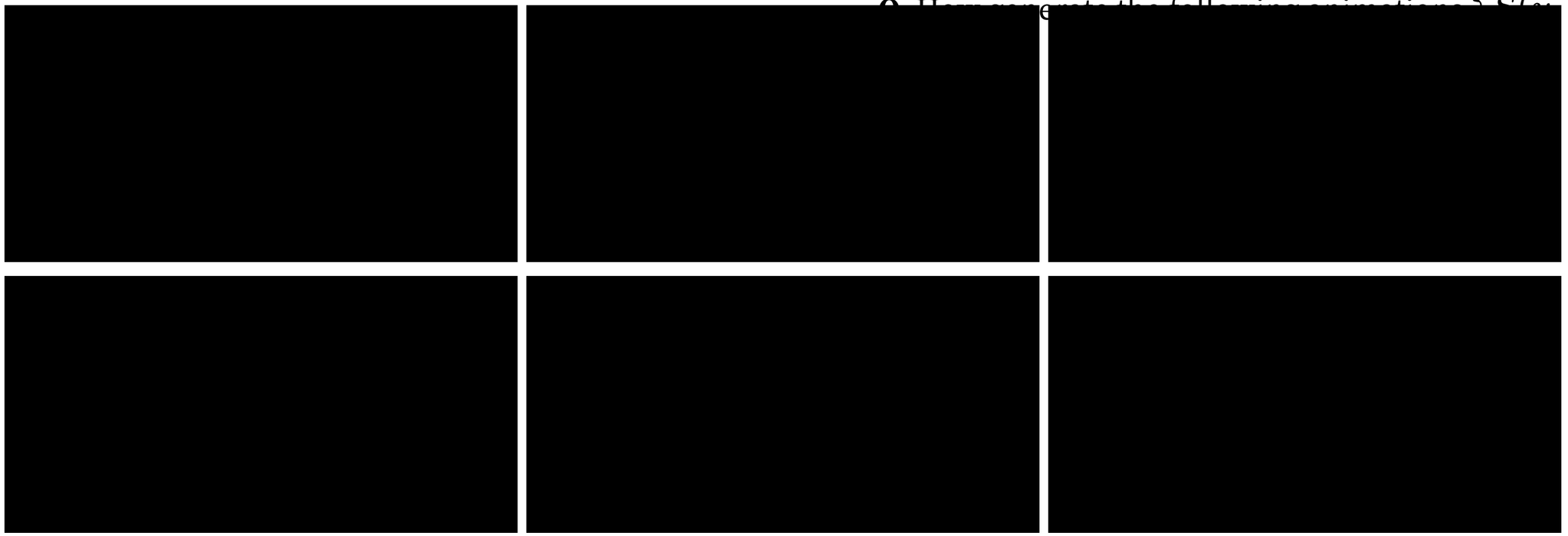
Depends on space and time

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

Perlin Noise - Animation

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

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



d e f