CSE 528: Computer Graphics

Procedural Texture Mapping

Klaus Mueller

Computer Science Department

Stony Brook University

Some material from Jian Huang, U Tennessee and Wei Shen, USC

Introduction

Introduced by Perlin and Perlin and Peachey (Siggraph 85/89)

Well described in the book by Ebert et al: “Texturing and Modeling: A Procedural Approach”

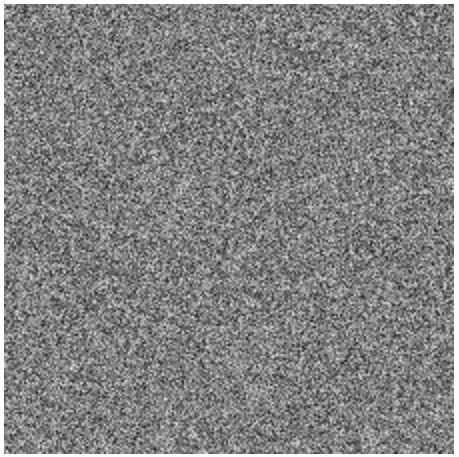
Offers

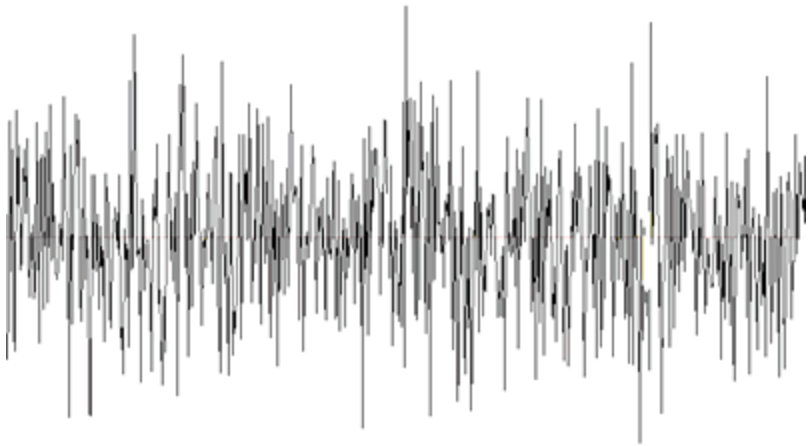
- storage savings for details (compact)
- there is no fixed resolution, a great range of detail no matter how close you look
- can cover an arbitrarily large space
- enables one to control parameters that define meaningful concepts, e.g. the roughness of a mountain

Difficulties

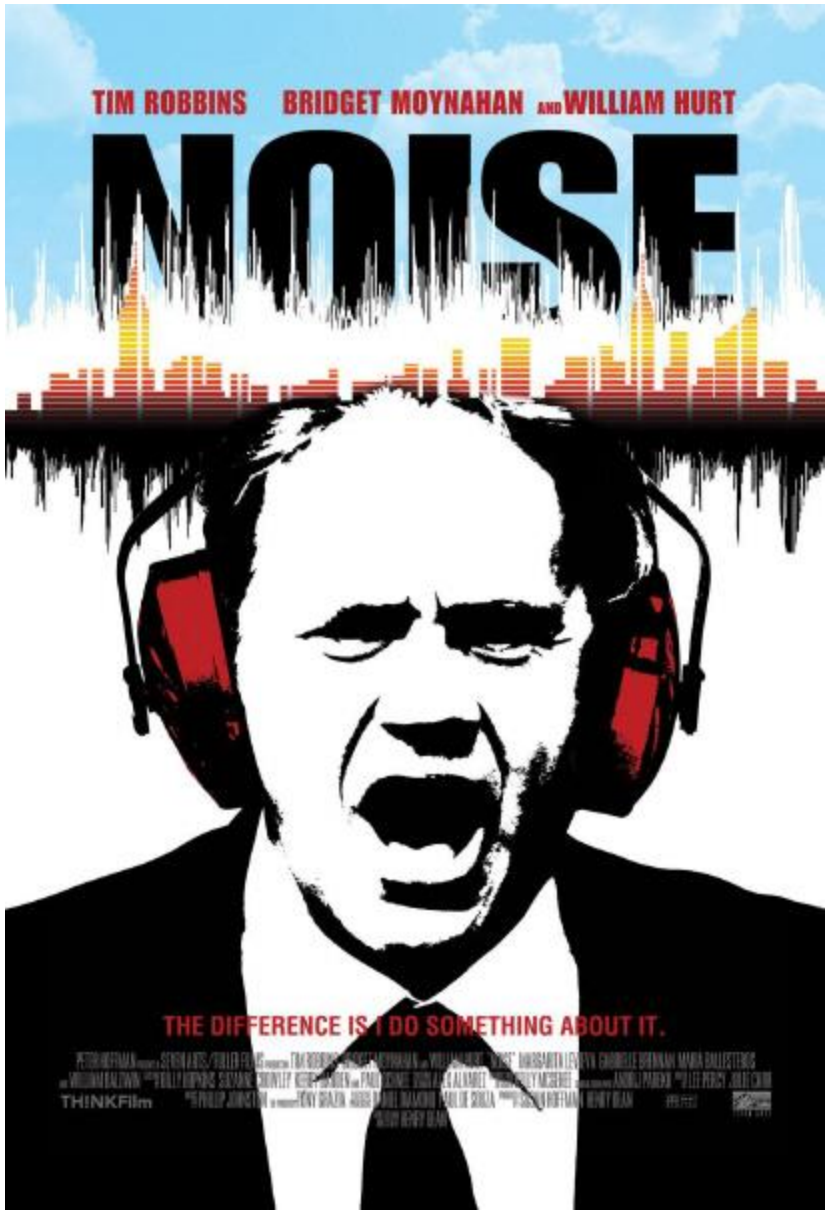
- very difficult to build and debug an implicit pattern description
- can yield surprising results
- can be slow to compute → use GPUs
- can tend to aliasing

Secret of the Game





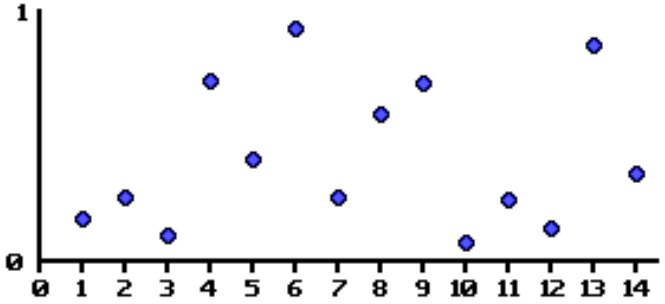
but done the right way



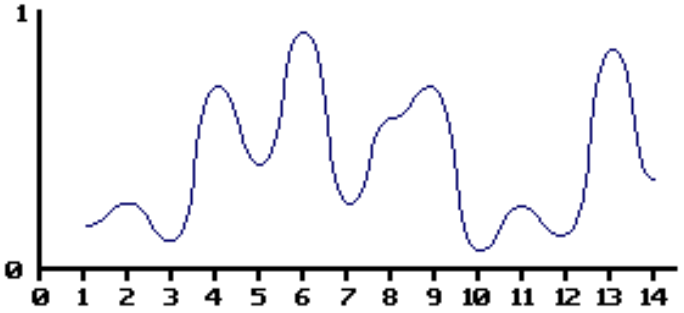
Noise Function: Noise(x)

According to Ken Perlin [Siggraph, 1995]

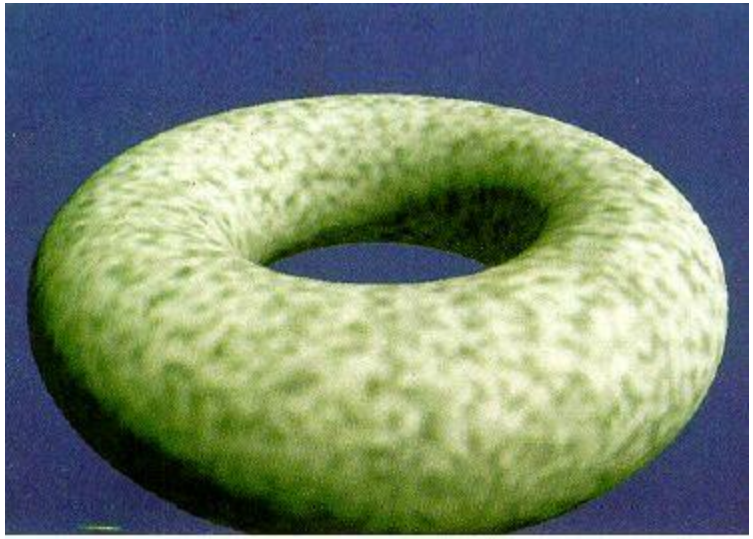
- Academy Award-winning well seeded random number generator at grid points (1D, 2D, 3D, … , n-D)



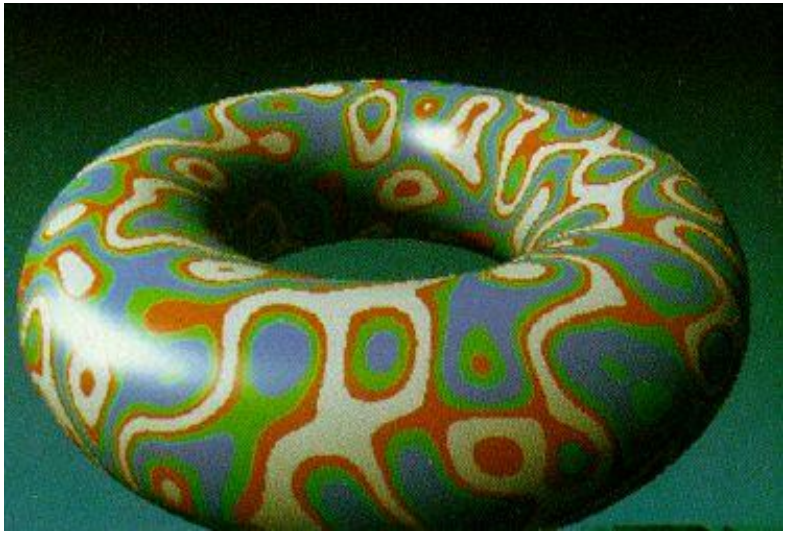
- paired with Hermite spline interpolation function for off-grid value estimation



Examples

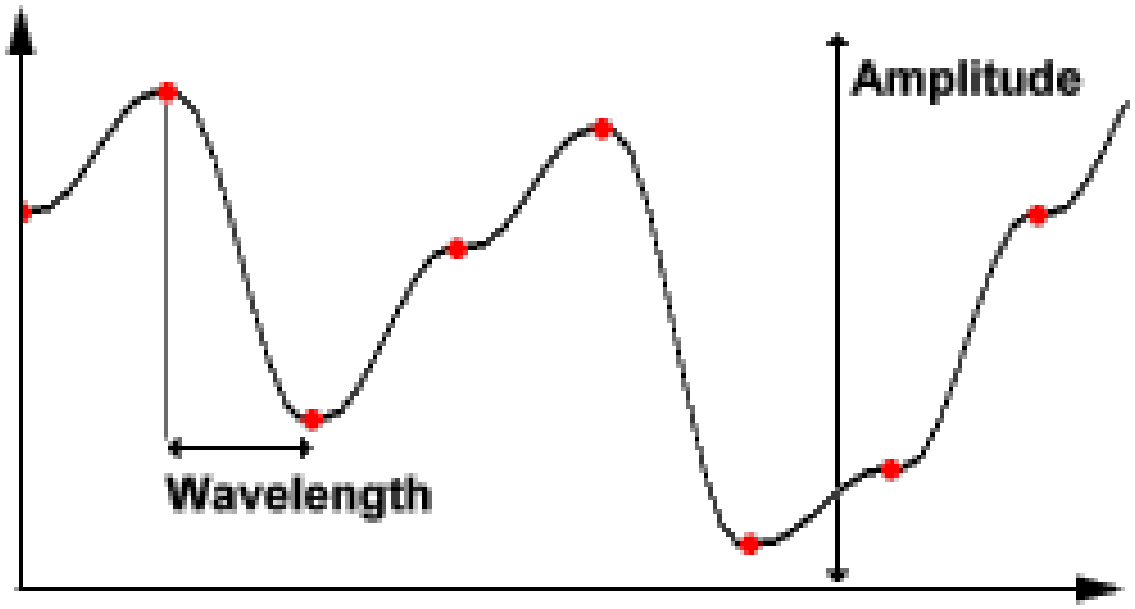


spotted donut
color = white * noise(**x**)



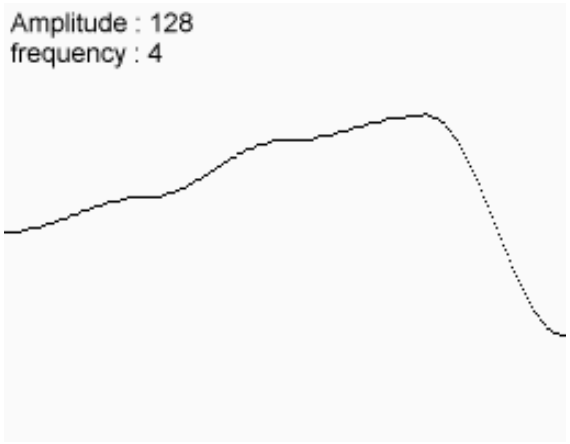
Bozo's donut
color = Colorful(noise(k***x**))

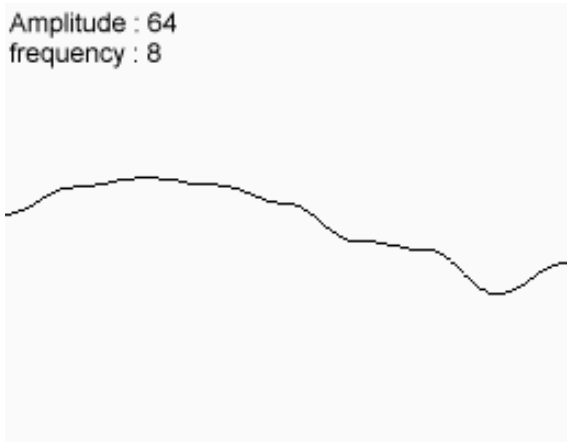
Wave Properties

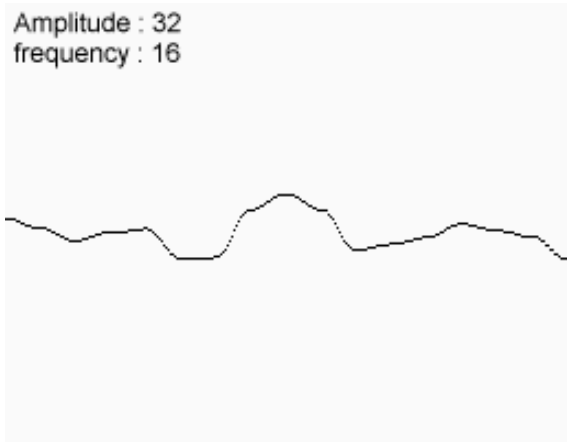


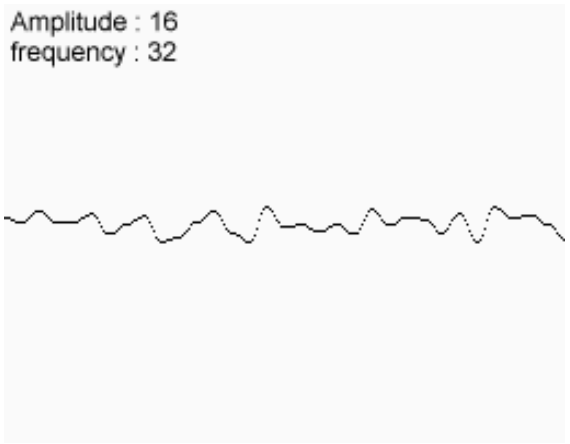
Frequency = 1/Wavelength

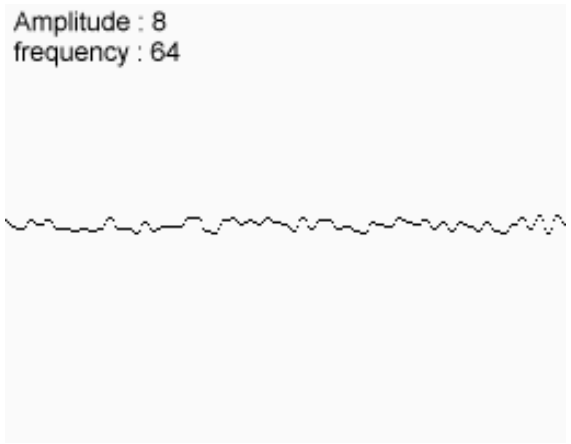
Noise Functions: Varied Frequency and Amplitude

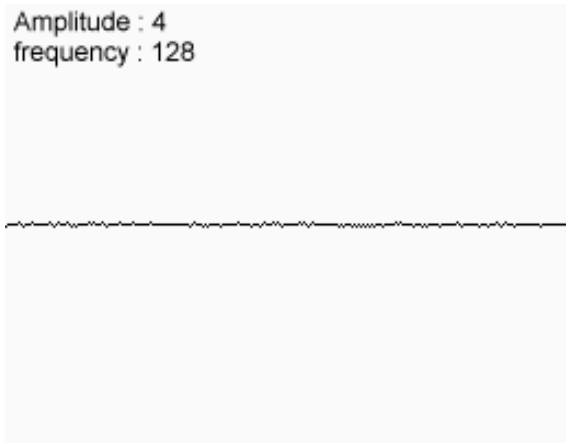




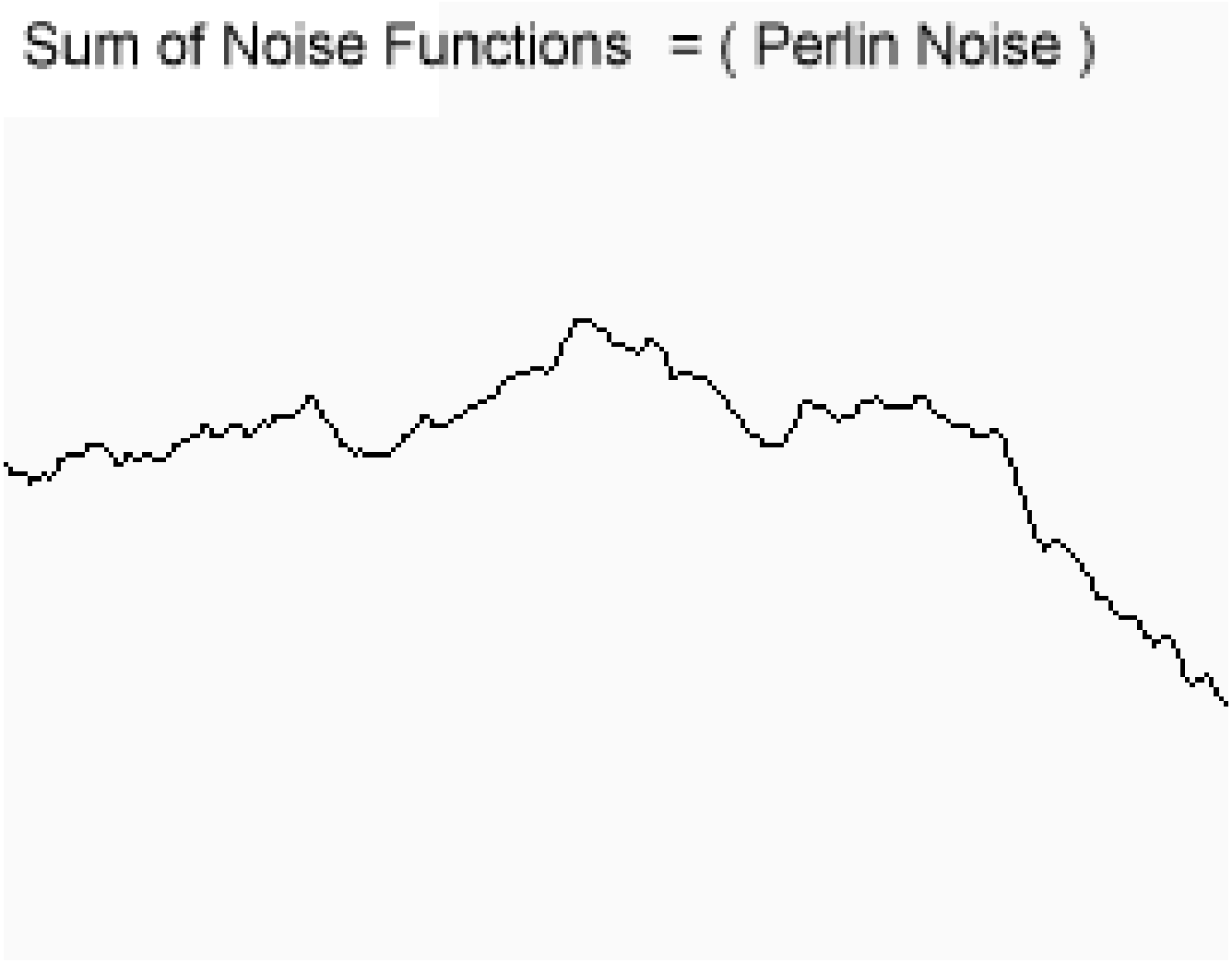








Add Them Together, Then

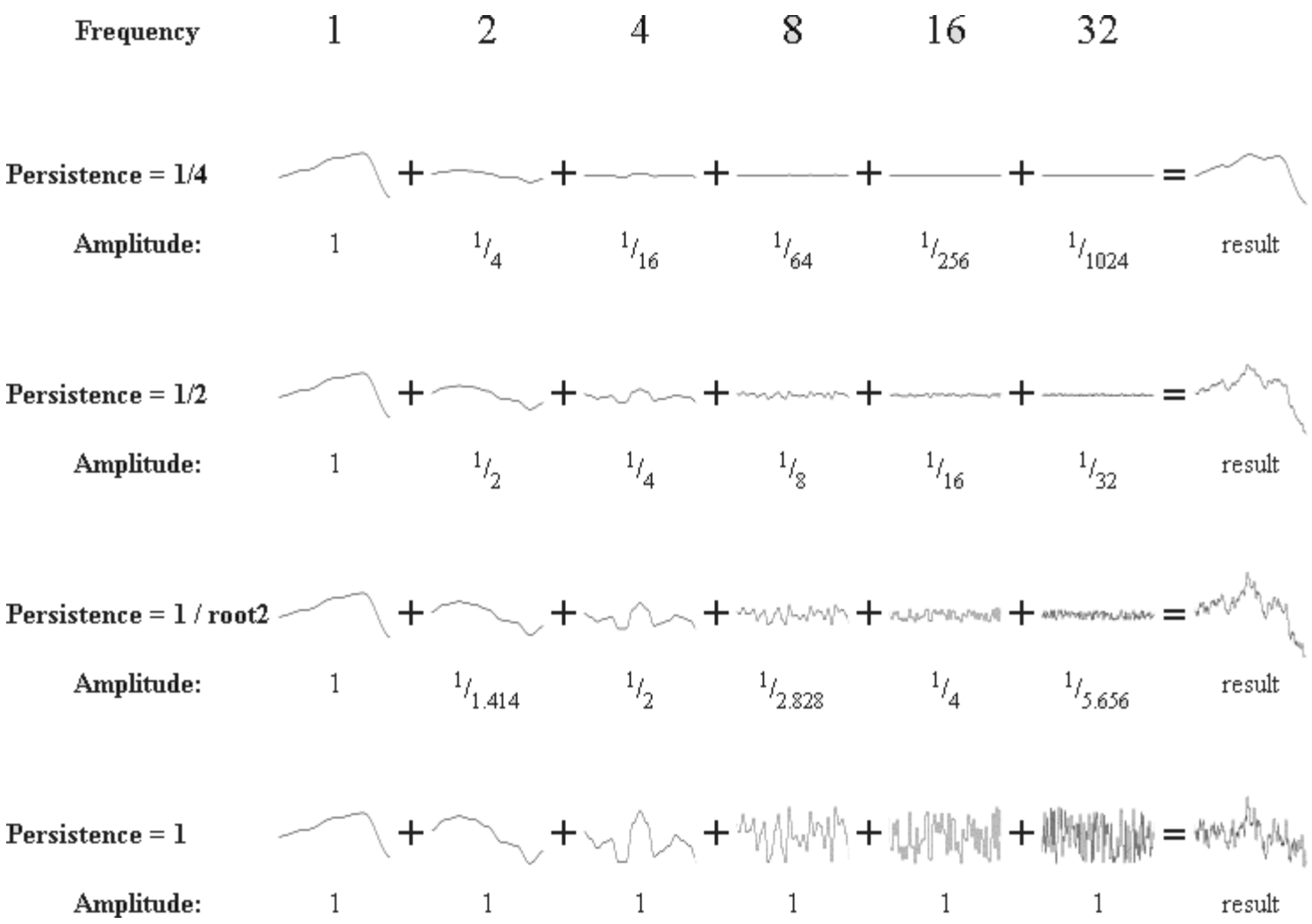


Persistence

A single number used to specify the amplitude of each frequency

- frequency = 2^i
- amplitude = $persistence^i$ → the higher the persistence the noisier the function
- i is the i^{th} noise function being added
- octave – each successive noise function

Persistence



Turbulence

Uses the persistence concept

Noise of higher frequency, similar to fractal brownian motion:

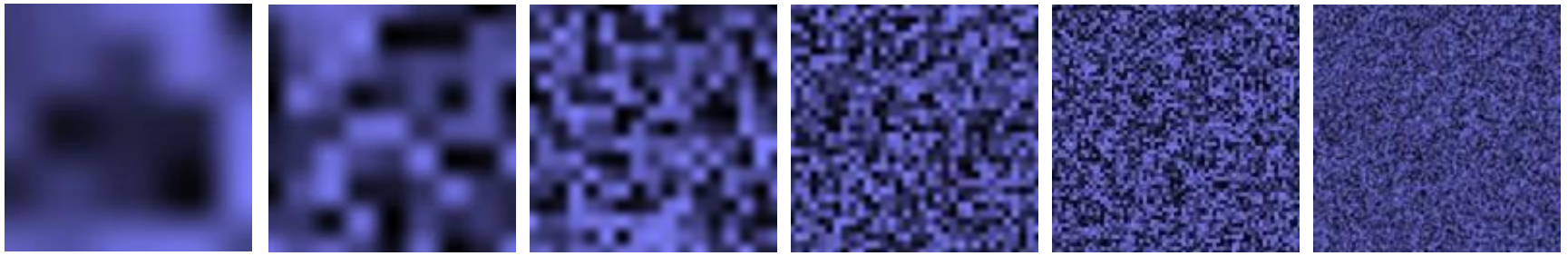
$$\sum_i abs(\frac{1}{2^i} noise(2^i \, \boldsymbol{x}))$$

- **x** is the coordinate vector of the grid point
- as frequency increases, the amplitude decreases

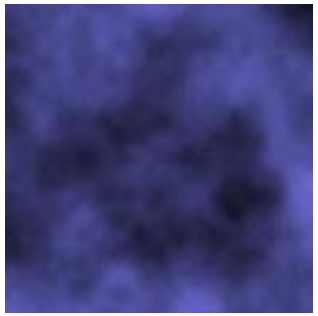
The abs() adds discontinuities in the first derivative

- gives a visual impression of discontinuous flow
- this will be interpreted by the viewer as turbulent

Some Noise Functions Created in 2D



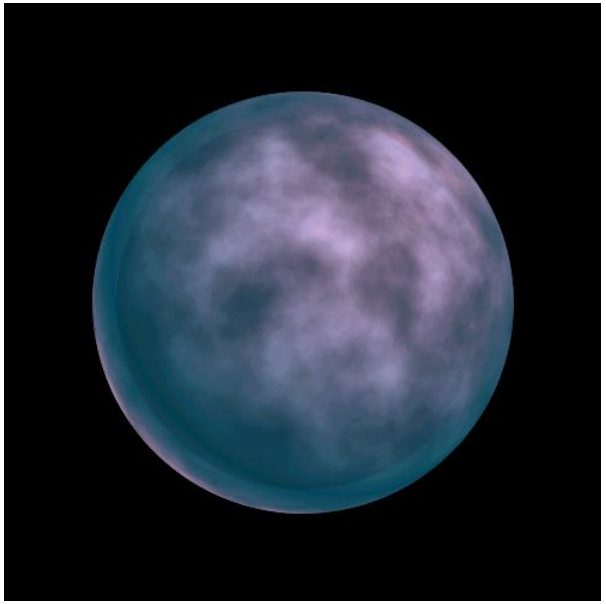
Adding all these functions together produces a noisy pattern



Limit the sum to the Nyquist limit of the current pixel sampling

- or run risk of aliasing due to insufficient patterns sampling

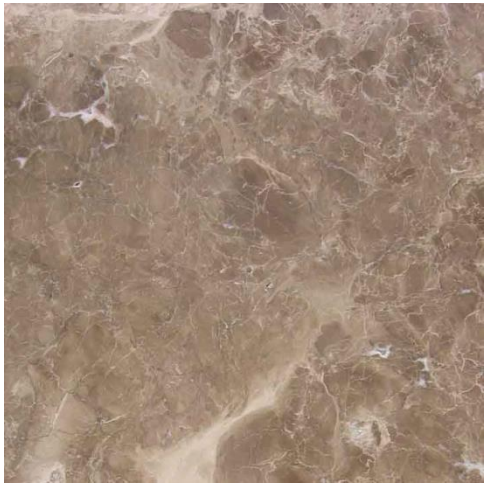
Some Noise Functions Created in 3D



Turbulence Example: Marble

We observe that marble consists of heterogeneous layers

- the "marble" look derives from turbulent forces which create deformations before these layers solidify



real marble

Perlin marble

Can use Perlin turbulence to create marble solid textures

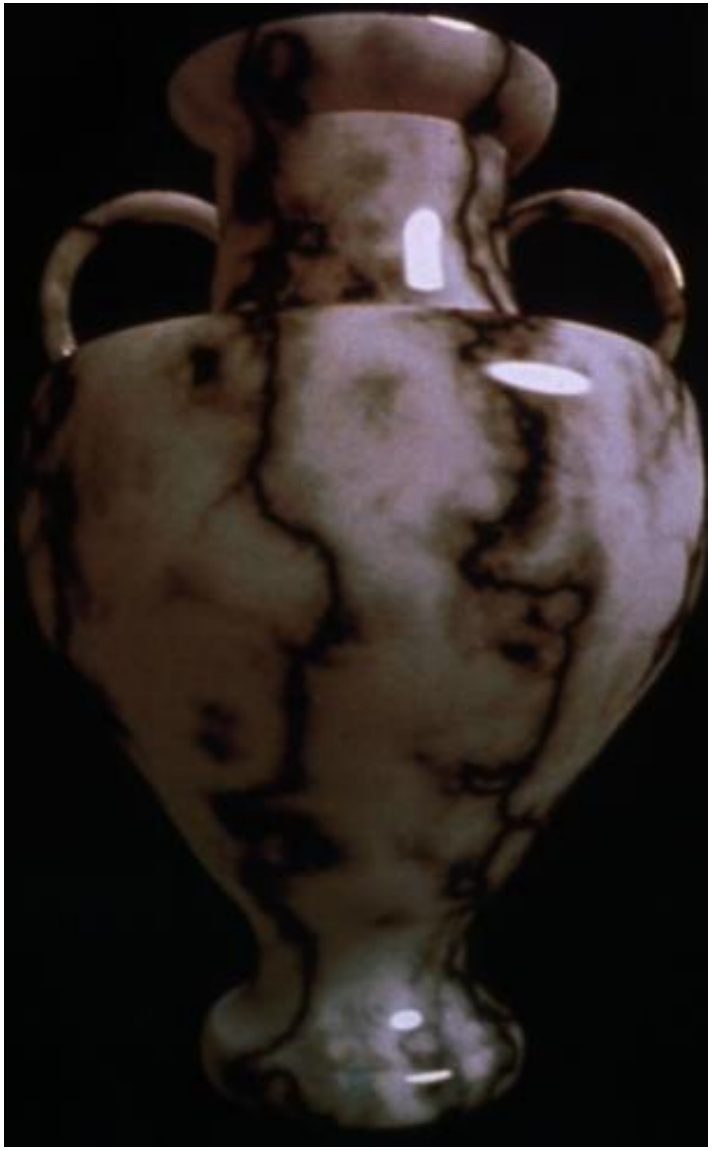
- boring_marble (**x**) = marble_color (sin(**x**[1])
  x[1] is the x-component of **x**
- good_marble (**x**) = marble_color (sin(**x**[1] + turbulence(**x**))

Marble

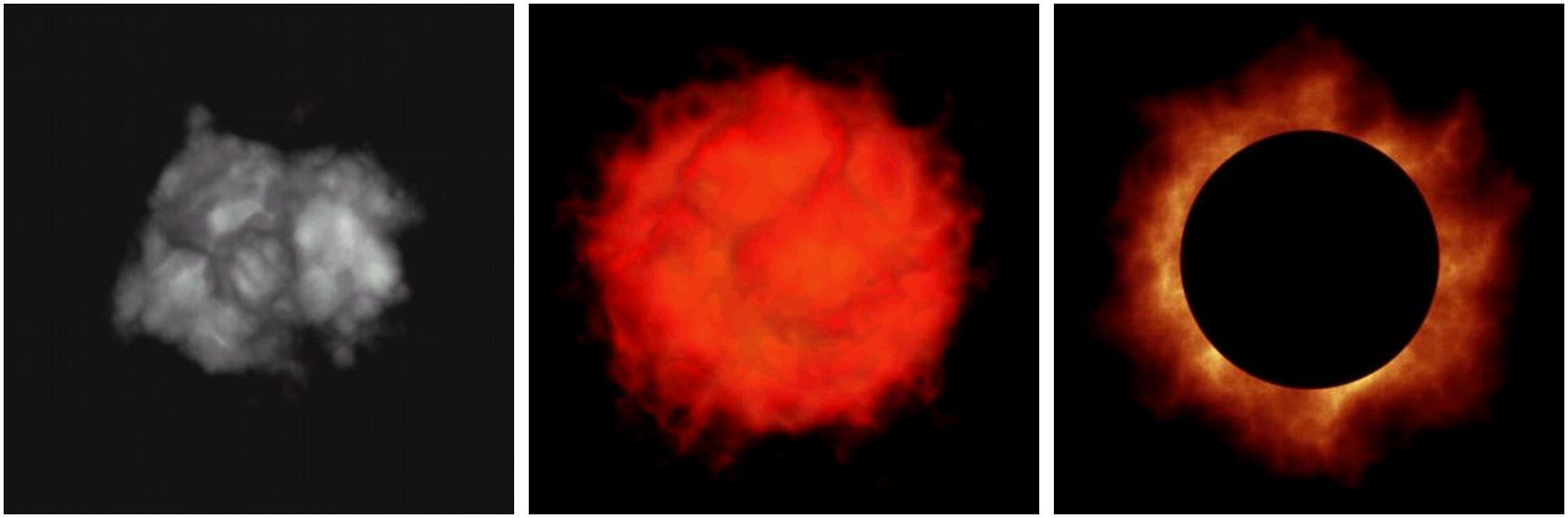
Boring marble

Good marble

Perlin's Marble Vase



Fire

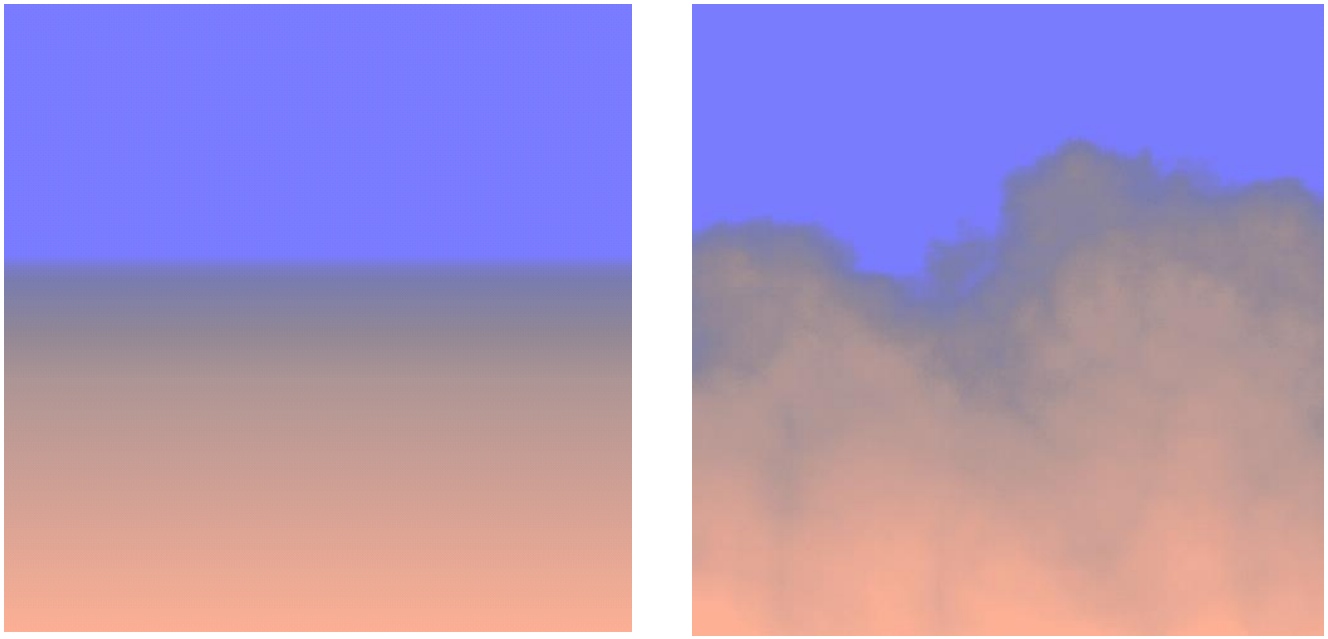


$$sphere(x(1+turbulence(x)))$$

Noise Effects: Translation

Effects:

- clouds
- noise translates in x,y



Many other effects:

- wood, fur, etc.

Rendering

Solid texture

- keeps original surface
- map (x,y,z) to (u,v,w) as usual

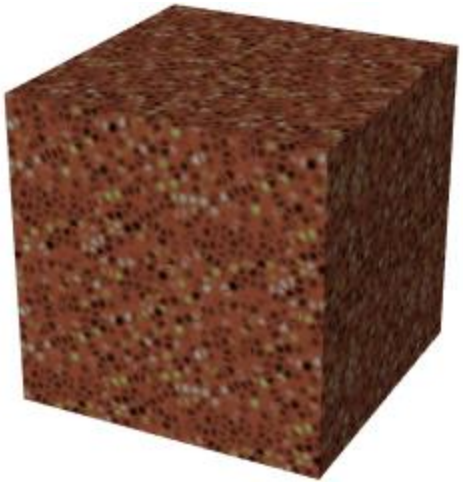
Hypertexture

- changes surface as well (density function)
- volume rendering approach → discrete ray caster

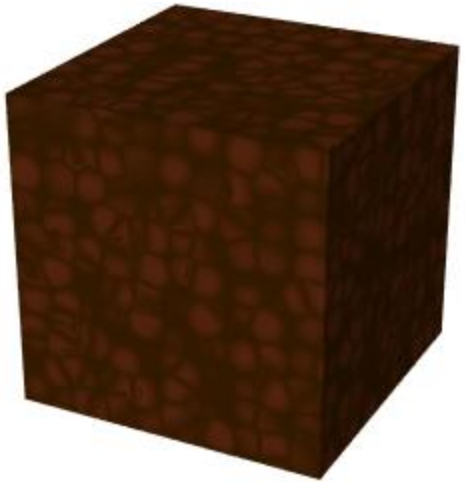
More Solid Textures

autodesk page

Wood

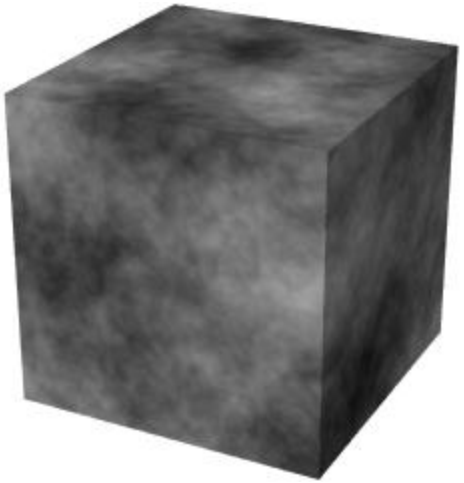


Granite

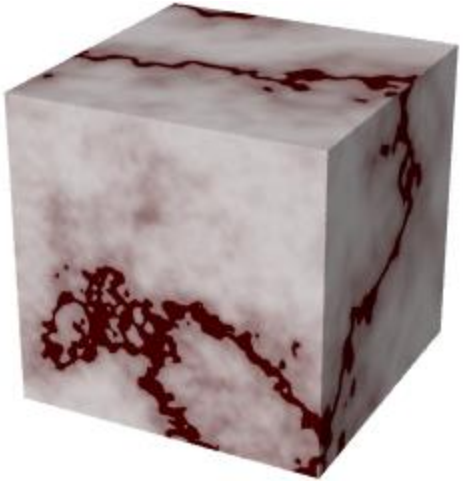


Leather

Rock



Fractal



Marble