Advanced Shading II: Procedural Texture and Noise
Recall: Texture Mapping
Procedural Texture

Main idea: determine color at \((u,v)\) using mathematical function
Procedural Texture

Main idea: determine color at \((u,v)\) using mathematical function

- no need for art assets
- computed on the fly: no memory cost
- can generate infinite amounts of data
Checkerboard

\[ I(u, v) = (\lfloor 2u \rfloor + \lfloor 2v \rfloor) \mod 2 \]
Stripes
Stripes

\[ I(u, v) = \sin u \]
Stripes

\[ I(u, v) = \sin u \quad \text{and} \quad I(u, v) = \sin(u + v) \]
Stripes
Stripes

\[ I(u, v) = \sin \sqrt{(u - 0.5)^2 + (v - 0.5)^2} \]
Problem with Procedural Noise

“Looks fake” – too regular

Real texture has noise
White Noise

\[ I(u, v) = \text{rand}(() \]
White Noise

White noise problems:
• isn’t smooth
• isn’t correlated

\[ I(u, v) = \text{rand}() \]
Upsampling Noise

Bilinear interpolation

\[ (u, v) \]
Upsampling Noise

Bilinear interpolation

\[(1 - u)d + uc\]

\[(1 - u)a + ub\]
Upsampling Noise

Bilinear interpolation

\[(u, v) \leftarrow (1 - u)(d + uc) + (1 - v)((1 - u)a + ub) + v((1 - u)d + uc)\]
Upsampling Noise

Bilinear interpolation
Upsampling Noise

Special case: power of 2 grid
Mountain Analogy

At coarse scale, has sparse peaks
Mountain Analogy

At coarse scale, has sparse peaks
Look closer, see false peaks
Mountain Analogy

At coarse scale, has sparse peaks
Look closer, see false peaks
Look even closer, see boulders, etc…
Procedural Noise

Pick random values on a coarse grid
Procedural Noise

Pick random values on a coarse grid

Interpolate to get coarse smooth texture
Procedural Noise

Apply mountain analogy to grid size:

\[ + \frac{1}{2} + \frac{1}{4} + \frac{1}{8} = n(u, v) \]
Procedural Noise

Previous scheme called value noise

Popular alternative: gradient (Perlin) noise
Perlin Noise

Sample unit vectors instead of values
Perlin Noise

Interpolate dot product with vec to corners

\[^{\vec{d}}\hat{a} \rightarrow \hat{c} \rightarrow \vec{c} (u, v) \rightarrow \hat{b}\]
Perlin Noise

Interpolate dot product with vec to corners

\[ (1 - v) \left[ (1 - u)\hat{a} \cdot \tilde{a} + u\hat{b} \cdot \tilde{b} \right] + v \left[ (1 - u)\hat{d} \cdot \tilde{d} + u\hat{c} \cdot \tilde{c} \right] \]
Procedural Noise Applications

\[ I(u, v) = \sin(u + v) \]
Procedural Noise Applications

\[ I(u, v) = \sin(u + v) \quad I(u, v) = \sin[u + v + \alpha n(\beta u, \beta v)] \]
Procedural Noise Applications

\[ I(u, v) = \sin[u + v + \alpha n(\beta u, \beta v)] \]
Procedural Noise Applications

\[ I(u, v) = \sin[u + v + \alpha n(\beta u, \beta v)] \]
Procedural Noise Examples
Minecraft Landscape
Minecraft Landscape

Key features:
• discrete (made of blocks / voxels)
• random – no repeated features
• extends indefinitely
• persistent
Minecraft Landscape

Key features:
• discrete (made of blocks / voxels)
• random – no repeated features
• extends indefinitely
• persistent
  • walk 5 miles north, then 5 miles south, should see the same landscape features
Minecraft Landscape

Making a small piece of landscape easy:
1. Generate procedural noise patch
   • one pixel per block in xy directions
Minecraft Landscape

Making a small piece of landscape easy:
1. Generate procedural noise patch
   • one pixel per block in xy directions
2. Clamp heights to discrete steps
Minecraft Landscape

Problems
1. not persistent
2. has seams between patches
Guaranteeing Persistence

Could precompute entire world...
Guaranteeing Persistence

Could precompute entire world...

...intractable and unnecessary

Cut up world into tiles
Guaranteeing Persistence

Could precompute entire world...
...intractable and unnecessary

Cut up world into tiles

Use deterministic seed for each tile
Building World on the Fly

Swap tiles in and out as needed

Keep $n \times n$ buffer of tiles loaded around the player

• “Zelda Algorithm”

Will still have some popping…
Building World on the Fly

Swap tiles in and out as needed

Keep $n \times n$ buffer of tiles loaded around the player

- “Zelda Algorithm”

Will still have some popping…
…could show only coarse levels far away
Eliminating Seams

Interpolate with neighbor tiles at each level