Perlin Noise

CS334 Fall 2017

Daniel G. Aliaga
Department of Computer Science
Purdue University

(slides with help from Jyun-Ming Chen, homepage.ttu.edu.tw/jmchen
and
Harriet Fell, http://www.ccs.neu.edu/home/fell)
To Ken Perlin for the development of Perlin Noise, a technique used to produce natural appearing textures on computer generated surfaces for motion picture visual effects.
The Movies

• James Cameron Movies (Abyss, Titanic,...)
• Animated Movies (Lion King, Moses,...)
• Arnold Movies (T2, True Lies, ...)
• Star Wars Episode I
• Star Trek Movies
• Batman Movies
• and lots of others

In fact, after around 1990 or so, every Hollywood effects film has used it.
What is Noise?

- Noise is a mapping from $\mathbb{R}^n$ to $\mathbb{R}$ - you input an $n$-dimensional point with real coordinates, and it returns a real value.
- $n=1$ for animation
- $n=2$ cheap texture hacks
- $n=3$ less-cheap texture hacks
- $n=4$ time-varying solid textures
Noise is Smooth Randomness
Example Images
Perlin’s Clouds and Corona
Perlin Noise Function

• Take lots of such smooth functions, with various frequencies and amplitudes
  – Idea similar to fractal, Fourier series, ...

• Add them all together to create a nice noisy function.
Fourier Analysis

Diagram showing a composite waveform made up of multiple partials.
Example

Amplitude: 128
frequency: 4

Amplitude: 64
frequency: 8

Amplitude: 32
frequency: 16

Amplitude: 16
frequency: 32

Amplitude: 8
frequency: 64

Amplitude: 4
frequency: 128
Example (cont)

- Function has large, medium and small variations.

![Graph](Sum of Noise Functions = ( Perlin Noise ))

Similar to the ideas of fractal
$\text{NOISE}(\mathbf{x}) = \sum_{i=0}^{N-1} \frac{\text{Noise}(b^i \mathbf{x})}{a^i}$
Some noise functions are created in 2D.

Adding all these functions together produces a noisy pattern.
You can create Perlin noise functions with different characteristics by using other frequencies and amplitudes at each step

- Is a multiplier that determines how quickly the amplitudes diminish for each successive octave in a Perlin-noise function.

\[
\text{frequency} = 2^i \\
\text{amplitude} = \text{persistence}^i
\]
<table>
<thead>
<tr>
<th>Frequency</th>
<th>1</th>
<th>2</th>
<th>4</th>
<th>8</th>
<th>16</th>
<th>32</th>
</tr>
</thead>
<tbody>
<tr>
<td>Persistence = 1/4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Amplitude:</td>
<td>1</td>
<td>1/4</td>
<td>1/16</td>
<td>1/64</td>
<td>1/256</td>
<td>1/1024</td>
</tr>
<tr>
<td>Persistence = 1/2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Amplitude:</td>
<td>1</td>
<td>1/2</td>
<td>1/4</td>
<td>1/8</td>
<td>1/16</td>
<td>1/32</td>
</tr>
<tr>
<td>Persistence = 1 / \sqrt{2}</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Amplitude:</td>
<td>1</td>
<td>1/\sqrt{2}</td>
<td>1/2</td>
<td>1/\sqrt{4}</td>
<td>1/\sqrt{8}</td>
<td>1/\sqrt{16}</td>
</tr>
<tr>
<td>Persistence = 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Amplitude:</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Interpolation

- Linear Interpolation
  
  ```
  function Linear_Interpolate(a, b, x)
  return a*(1-x) + b*x
  ```

- Cosine Interpolation
  
  ```
  function Cosine_Interpolate(a, b, x)
  ft = x * 3.1415927
  f = (1 - cos(ft)) * .5
  return a*(1-f) + b*f
  ```
Interpolation

• Cubic Interpolation

Etc…
function \texttt{Noise}(x)
  \ldots
end function

def \texttt{SmoothNoise\_1D}(x):
    return \texttt{Noise}(x)/2 + \texttt{Noise}(x-1)/4 + \texttt{Noise}(x+1)/4
end function

function \texttt{SmoothNoise\_2D}(x, y):
    corners = ( \texttt{Noise}(x-1, y-1)+\texttt{Noise}(x+1, y-1)+\texttt{Noise}(x-1, y+1)+\texttt{Noise}(x+1, y+1) ) / 16
    sides = ( \texttt{Noise}(x-1, y) +\texttt{Noise}(x+1, y) +\texttt{Noise}(x, y-1) +\texttt{Noise}(x, y+1) ) / 8
    center = \texttt{Noise}(x, y) / 4
    return corners + sides + center
end function
Example Code
http://mrl.nyu.edu/~perlin/