Computer Graphics

Tom Ellman
Lecture 16
Texture Mapping

• Rendered surfaces are often too smooth and uniform to appear realistic.

• Real surfaces often have texture:
  – The wood grain on a desk top.
  – The checkered tiling on a floor.
  – The stripes on a shirt.

• Texture mapping adds features like these to otherwise smooth, uniform and unrealistic surfaces.
Texture Maps

• A rectangular array of texels.
• The array is indexed by its own (u,v) coordinate space.
• A texel may be a single intensity level, or a three (r,g,b) vector, or a four (r,g,b,α) vector.
• Texture maps may be generated from:
  – Mathematical formulae. (E.g., A checkerboard.)
  – Algorithmic procedures. (E.g, A random number generator.)
  – A scanned photographs. (E.g., A human face.)
  – A drawing tool. (E.g., A product label.)
Checkerboard Texture Map

color = (row + column) % 2
Procedural Wave Pattern Texture Map on a Plane
Ketchup Label from Photograph
Mustard Label from Drawing Tool
Assigning Texture Coordinates to a Geometric Surface

- **Parametric surface:**
  - Use \((u,v)\) surface parameters as texture coordinates.
  - Use \((u',v')\) as texture coordinates, where \((u',v')\) are related to \((u,v)\) by an affine (linear + translation) transformation.

- **Polygon Mesh:**
  - Use interactive texture coordinate editing tools.
  - Project polygon vertices onto a sphere, cylinder, plane or box.
Texture Mapping a Parametric Surface with an Affine Transformation
Projection Mapping a Polygon Mesh onto a Cylinder

Point on Texture Map

Vertex of Polygon Mesh

Center of Projection
Two Issues of Texture Mapping

• Issue 1: For a given pixel in the rendered image, which texels in the texture map contribute to the value of pixel?

• Issue 2: How should the selected texels be combined with each other and with the untextured value of the pixel?
Association between Pixel and Texels

Texture Map to Surface Transformation

Viewing and Perspective Transformations

Texture Map
Surface of Object
Pixel on Screen
Association between Pixel and Texels

Inverse of Texture Map to Surface Transformation

Inverse of Perspective and Viewing Transformations
Association between Pixels and Texels

- **Magnification**: One pixel covers only a small part of a single texel.
- **Minification**: One pixel covers all or part of many texels.
Interpolated Texture Mapping

• Represent each surface as a polygonal mesh.
• Assign texture map coordinates \((u,v)\) to each vertex.
• Use linear interpolation to associate texels with a given pixel.
When to Carry out Interpolation?

- Perspective projection is a non-linear transformation.
- It can make a big difference whether we interpolate the texture map before or after perspective projection.
- For proper foreshortening of features in the texture map, we must interpolate before the perspective projection.
Interpolated Texture Mapping

1. Pixel on Screen

2. Preimage of pixel on polygon.
Interpolated Texture Mapping

2. Preimage of pixel on polygon.

3. Texmap coordinates assigned to polygon vertices.
Interpolated Texture Mapping

3. Texmap coordinates assigned to polygon vertices.

4. Texmap coordinates interpolated to pixel preimage vertices.
Interpolated Texture Mapping

4. Texmap coordinates interpolated to pixel preimage vertices.

5. Pixel preimage in texture map.
Association between Pixel and Texels

Texture Map with Preimage of Pixel

Outlined Texels May Contribute to Pixel
Combining Multiple Texels

- Select only the texel that is nearest to the preimage of center of pixel.
- Weight each texel according to its area of overlap with preimage of pixel.
- Weight each texel according to its distance from the preimage of center of pixel.
- Use a filtering function to weight texels.
Combining Texels with Untextured Value of Pixel

• Decal Mode: Texture overwrites untextured value of the pixel.

• Scaling Mode: Texture values are used to scale the untextured value of the pixel.

• Blending Mode: Texture values and untextured pixel value are combined in a weighted average.
Problems with Texture Mapping

- Consider a textured polygon that gradually recedes from the foreground to the background.
- A pixel near the camera may cover only a small part of one texel.
- A pixel far from the camera may cover a large number of texels.
- Shimmering, flashing and scintillation may occur in between.
Multiple Levels of Detail

• OpenGL allows user to specify a family of texture maps at levels of resolution.
• The texture maps in a family will have sizes $2^k \times 2^k$ for $k = 0 \ldots n$.
• OpenGL automatically chooses the best level of resolution for a given pixel.
• OpenGL provides options for combining several texture maps at different levels of resolution in the course of rendering each pixel.
Multi-Level Texture Maps

\[ 2^0 \times 2^0 \text{ Map} \]

\[ \vdots \]

\[ 2^{k-1} \times 2^{k-1} \text{ Map} \]

\[ 2^k \times 2^k \text{ Map} \]
Procedural Fractal Noise Bump Map on a Sphere
Bump Mapping

• Surfaces rendered with texture mapping will often appear to be smooth.
• Bump mapping is a technique that simulates small scale variations on a surface.
• Uses a “bump map” to perturb the surface normal vector according to a pattern that varies over the surface.
• Uses the perturbed normal vector to calculate illumination of points on the surface.
Perturbing the Surface Normal

Unperturbed

Perturbed
Parametric Surface Description

\[ \mathbf{P}(s, t) \quad \text{Parametric description of a surface.} \]

\[ \mathbf{P}_s = \frac{\partial \mathbf{P}(s, t)}{\partial s} \quad \text{Surface tangent in s direction.} \]

\[ \mathbf{P}_t = \frac{\partial \mathbf{P}(s, t)}{\partial t} \quad \text{Surface tangent in t direction.} \]

\[ \mathbf{N} = \mathbf{P}_s \times \mathbf{P}_t \quad \text{Surface normal (not normalized).} \]

\[ \mathbf{n} = \mathbf{N} / |\mathbf{N}| \quad \text{Surface normal (normalized).} \]
The “Bump Function”

• Let $b(s,t)$ be a “bump function”.
• The bump function represents a distribution of bumps and/or dents over the surface.
• Each dent or bump pushes the surface in or out in the direction of the local normal.
• The bump function is usually computed by table lookup and interpolation.
Perturbed Surface Normal

\[ P'(s,t) = P(s,t) + b(s,t) \mathbf{n}(s,t) \]

\[ P'_s = \frac{\partial P'}{\partial s} = P_s + b_s \mathbf{n} + b \mathbf{n}_s \approx P_s + b_s \mathbf{n} \]

\[ P'_t = \frac{\partial P'}{\partial t} = P_t + b_t \mathbf{n} + b \mathbf{n}_t \approx P_t + b_t \mathbf{n} \]

\[ \mathbf{N}' = P'_s \times P'_t = (P_s + b_s \mathbf{n}) \times (P_t + b_t \mathbf{n}) \]

\[ = P_s \times P_t + P_s \times b_t \mathbf{n} + b_s \mathbf{n} \times P_t + b_s \mathbf{n} \times b_t \mathbf{n} \]

\[ = P_s \times P_t + P_s \times b_t \mathbf{n} + b_s \mathbf{n} \times P_t \]

\[ = \mathbf{N} + b_t (P_s \times \mathbf{n}) + b_s (\mathbf{n} \times P_t) \]

\[ \mathbf{n}' = \mathbf{N}' / |\mathbf{N}'| \]
Procedural Fractal Noise Texture and Bump Map on a Sphere
Fractal Brownian Motion

1. Start with a flat \((x,y)\) grid.
2. Construct a low frequency noise function \(N(x,y)\).
3. Add \(N(x,y)\) to each \((x,y)\) point on the grid.
4. Increase the frequency and lower the amplitude of \(N(x,y)\).
5. If texture is rough enough, then stop, else go back to step three.
Frequency Limited Noise

- Noise is a function $N(t)$ of a variable $t$.
- $N(t)$ appears to be random, when sampled at widely separated $t$ values.
- $N(t)$ appears to be a continuous, smooth function of $t$, when sampled at closely spaced $t$ values.
Constructing Frequency Limited Noise

• Generate an array $A[i]$ ($0 < i < n$) of random numbers:
  – Value noise: Each $A[i]$ is a number in the range $[0, 1]$ and represents the value of $N(t)$ at $t = i$.
  – Gradient noise: Each $A[i]$ is a number in the range $(-\infty, \infty)$ and represents the value of $N'(t)$ at $t = i$.

• Use an interpolation technique to construct a continuous and smooth function $N(t)$. 
Additive FBM Noise

\[
\text{Value}(t) = \sum_{k=0}^{n-1} \frac{N(L^k t)}{L^{kH}}
\]

\[
\text{Value}(t) = N(t) + \frac{N(L t)}{L^H} + \frac{N(L^2 t)}{L^{2H}} + \ldots + \frac{N(L^{(n-1)} t)}{L^{(n-1)H}}
\]

H is the fractal increment parameter.

L is the lacunarity.

n is the number of octaves.
Multiplicative FBM Noise

\[
\text{Value}(t) = \prod_{k=0}^{n-1} 1 + \frac{N(L^{kt})}{L^{kH}}
\]

\[
\text{Value}(t) = \left\{1+N(t)\right\} * \left\{1 + \frac{N(L^t)}{L^H}\right\} * ... * \left\{1 + \frac{N(L^{(n-1)t})}{L^{(n-1)H}}\right\}
\]

H is the fractal increment parameter.

L is the lacunarity.

n is the number of octaves.
Generating High Frequency Noise

\[ N(t) \]

\[ N(L \cdot t) \]
fBm Noise
$H = 4.0$
fBm Noise
H = 3.0
N(t)

fBm Noise
H = 2.5
fBm Noise
$H = 2.0$
N(t) vs t

fBm Noise
H = 1.75
$N(t)$

fBm Noise

$H = 1.5$
N(t)  

fBm Noise
H = 1.25
fBm Noise

$H = 1.0$

$N(t)$
N(t)

fBm Noise
H = 0.8
fBm Noise
$H = 0.6$
fBm Noise
$H = 0.4$
fBm Noise

H = 0.2

N(t)
fBm Noise
H = 0.0
FBM Texture

Gradient noise, fractal increment parameter $H = 4.0$, lacunarity $L = 2.0$ and number of octaves $n = 24$. 

Project: Fractal Noise
Gradient noise, fractal increment parameter $H = 1.5$, lacunarity $L = 2.0$ and number of octaves $n = 24$. 
Gradient noise, fractal increment parameter $H = 0.25$, lacunarity $L = 2.0$ and number of octaves $n = 24$. 

FBM Texture
Bias and Gain Functions

- Bias(b,x) and Gain(g,x) are continuous, smooth, monotonically increasing functions from [0,1] onto [0,1].
- Bias and gain functions can be used to control the brightness and contrast of a texture.
- They may also be used to control variations in shading of a texture.
Bias and Gain Functions

\[ \text{Bias}(b,x) = x^{\ln(b)/\ln(1/2)} \]

\[ \text{Gain}(g,x) = \begin{cases} 
\text{Bias}(1-g,2x)/2 & \text{x } \leq \text{1}/2 \\
1-\text{Bias}(1-g,2-2x)/2 & \text{x } > \text{1}/2
\end{cases} \]
Bias(b,x)

- b=0.25
- b=0.5
- b=0.75
Gradient noise, bias $b = 0.5$, gain $g = 0.5$, fractal increment parameter $H = 1.0$, lacunarity $L = 2.0$ and number of octaves $n = 24$. 

FBM Texture
FBM Texture

Gradient noise, bias $b = 0.75$, gain $g = 0.5$, fractal increment parameter $H = 1.0$, lacunarity $L = 2.0$ and number of octaves $n = 24$. 
FBM Texture

Gradient noise, bias $b = 0.25$, gain $g = 0.5$, fractal increment parameter $H = 1.0$, lacunarity $L = 2.0$ and number of octaves $n = 24$. 
Gradient noise, bias $b = 0.5$, gain $g = 0.5$, fractal increment parameter $H = 1.0$, lacunarity $L = 2.0$ and number of octaves $n = 24$. 
FBM Texture

Gradient noise, bias $b = 0.5$, gain $g = 0.85$, fractal increment parameter $H = 1.0$, lacunarity $L = 2.0$ and number of octaves $n = 24$. 
FBM Texture

Gradient noise, bias $b = 0.5$, gain $g = 0.15$, fractal increment parameter $H = 1.0$, lacunarity $L = 2.0$ and number of octaves $n = 24$. 
Solid Texture

- Design a function $S(x,y,z)$ that gives the shading as a function of coordinates in 3D space.
- Evaluate $S(x,y,z)$ at vertices of a polygonal mesh that represents an object’s surface.
- Interpolate to get shading at points on the interior of the polygon.
Wood

\[ S(r, \phi, z) = A \sin(\omega r + \phi) + \text{Noise} \]

- Making \( S(r, \phi, z) \) a periodic function of radius represents growth rings of a tree.

- Evaluation of \( S(r, \phi, z) \) on a flat surface will result in wood grain.
Wood Texture
Hypertexture

- An approach to describing amorphous 3D shapes.
- (E.g., Fluffy clouds, billowing smoke, etc.)
- Start with a shape defined implicitly with a function: \( f(x,y,z) \leq 0 \).
- Add FBM noise to the function to define a shape by \( S(x,y,z) = f(x,y,z) + \text{FBM}(x,y,z) \leq 0 \).
Hypertexture Example

\[ S(r, \theta, \phi) \equiv R - r + \text{FBM}(\theta, \phi) \]

• The expression \( f(r, \theta, \phi) \equiv R - r \leq 0 \) defines a spherical region of radius \( R \).

• The function \( \text{FBM}(\theta, \phi) \) describes how the sphere is perturbed differently in each direction.
Cloud
Jonathan Merritt (BMRT)