Overview

Level of detail hierarchy
Texture maps
Procedural shading and texturing
Texture synthesis and noise

Hierarchy

Physics

Geometrical optics
- Macro-structures
  - Transport
- Micro-structures
  - Microfacets
Physical optics
  - Kirchoff approx.
  - Quantum optics

Computer Graphics

Geometry
Displacement (P) maps
Bump (N) maps
Texels (BRDF)
Texture
Texture Maps

How is texture mapped to the surface?
- Dimensionality: 1D, 2D, 3D
- Texture coordinates (s,t)
  - Surface parameters (u,v)
  - Direction vectors: reflection R, normal N, halfway H
  - Projection: cylinder
  - Developable surface: polyhedral net
  - Reparameterize a surface: old-fashion model decal

What does texture control?
- Surface color and opacity
- Illumination functions: environment maps, shadow maps
- Reflection functions: reflectance maps
- Perturb geometry: bump and displacement maps

History

Catmull/Williams 1974 - basic idea
Blinn and Newell 1976 - basic idea, reflection maps
Blinn 1978 - bump mapping
Williams 1978, Reeves et al. 1987 - shadow maps
Smith 1980, Heckbert 1983 - texture mapped polygons
Williams 1983 - mipmap
Miller and Hoffman 1984 - illumination and reflectance
Perlin 1985, Peachey 1985 - solid textures
Greene 1986 - environment maps/world projections
Akeley 1993 - Reality Engine
Direction Maps

Many ways to map directions to images...

Methods:
- Gazing Ball (N)
  Create by photographing a reflective sphere
- Fisheye Lens
  Standard camera lens
- Cubical Environment Map (R)
  Create with a rendering program, photography...
- Latitude-Longitude (Map Projections)
  Create by painting

Issues:
- Non-linear mapping - expensive, curved lines
- Area distortion - spatially varying resolution
- Convert between maps using image warp
Cubical Environment Map

Note:

- Easy to produce with rendering system
- Possible to produce from photographs
- “Uniform” resolution
- Simple texture coordinates calculation

Gazing Ball

Note:

- Photograph of reflective ball
- Reflection indexed by normal
- Maps entire field of view to circle
- Resolution function of orientation; maximum head-on
- Alternatives: Fish eye, map projections
Reflectance Maps

Integrate over a hemisphere: BRDF * L
Very low resolution often sufficient: At NYIT 49x49

Reflectance Maps

Reflectance map
  For a given viewing direction
    For each normal direction
      For each incoming direction (hemispherical integral)
        Evaluate reflection equation

Reflection functions
  ■ Diffuse: Irradiance map
  ■ Glossy: Radiance map
  ■ Anisotropic: for each tangent direction
  ■ Mirror: Reflection map related to environment map

Illumination functions
  ■ Environment maps
  ■ Procedural light sources
Quake Light Maps

Illumination Maps

reflectance  irradiance  radiosity
Shadow Maps

May incorporate shadow maps into lighting calculations

Correct Shadow Maps

Step 1:
Create z-buffer of scene as seen from light source

Step 2.
Render scene as seen from the eye
For each light
Transform point into light coordinates
return (zl < zbuffer[xl][yl]) ? 1 : 0
Displacement/Bump Mapping

Offset surface position

- Displacement
  \[ P'(u, v) = P(u, v) + h(u, v)N(u, v) \]
- Perturb normal
  \[ N(u, v) = \frac{\partial P(u, v)}{\partial u} \times \frac{\partial P(u, v)}{\partial v} \]

From Blinn 1976

Shading and Texturing Language

Flexibility

- Create procedural texture models
- Modulate multiple parameters
- Control over mapping

Texture access

float/color texture( "image", s, t, ...)
point bump("heights", N, Ps, Pt, s, t, ...)
float/color environment("cubefaces", D, ...)
float shadow("depths", P, ...)
Abstract Shading Model

Light Shader State

light bulb(
    float intensity = 1;
    color filament = TUNGSTEN )
{
    illuminate( P )
    Cl = intensity * filament / (L.L);
}
Barzel’s UberLight.sl

Example of a complex shader

UberLight( )
{
    // Clip to near/far planes
    Clip to shape boundary
    foreach superelliptical blocker
        atten *= ...
    foreach cookie texture
        atten *= ...
    foreach slide texture
        color *= ...
    foreach noise texture
        atten, color *= ...
    foreach shadow map
        atten, color *= ...
    Calculate intensity fall-off
    Calculate beam distribution
}

Inconsistent Shadows

Projected Shadow Matte

Projected Texture

Surface Shader

surface diffuse()
{
    color Ci = 0;
    illuminance( P, N, Pi/2 )
        Ci += Cs * Cl * N.L;
}

Pat Hanrahan, Spring 2000
RenderMan Surface Shader

surface corrode(float Ks=0.4, Ka=0.1, rough=0.25)
{
    float i, freq=1, turb=0;

    // compute fractal texture
    for( i=0; i<6; i++ ) {
        turb += 1/freq*noise(freq*P);
        freq *= 2;
    }

    // perturb surface
    P -= turb * normalize(N);
    N = faceforward(normalize(calculatenormal(P)));

    // compute reflection and final color
    Ci = Cs*(Ka*ambient()+Ks*specular(N,I,rough));
}

Perlin’s Noise Function

1. Generate a table of random numbers
2. Hash a 3D lattice into a table entry
3. Use random values as the gradient
4. Perform cubic interpolation
Turbulence

fBm

// compute fractal texture
for( i=0; i<6; i++ ) {
    turb += 1/freq*noise(freq*P);
    freq *= 2;
}

Images from http://freespace.virgin.net/hugo.elias/models/m_perlin.htm

Examples

Wood and Stone - RenderMan Companion

Marble - Ken Perlin
Examples (continued)

Cloud -
David Ebert