Environment Mapping

Computer Graphics
CSE 167
Lecture 13
CSE 167: Computer graphics

• Environment mapping
  – An image-based lighting method
  – Approximates the appearance of a surface using a precomputed texture image (or images)
  – In general, the fastest method of rendering a surface
More realistic illumination

• In the real world, at each point in a scene, light arrives from all directions (not just from a few point light sources)
  – Global illumination is a solution, but is computationally expensive
  – An alternative to global illumination is an environment map
    • Store “omni-directional” illumination as images
    • Each pixel corresponds to light from a certain direction
    • Sky boxes make for great environment maps

Based on slides courtesy of Jurgen Schulze
Reflection mapping

• Early (earliest?) non-decal use of textures
• Appearance of shiny objects
  – Phong highlights produce blurry highlights for glossy surfaces
  – A polished (shiny) object reflects a sharp image of its environment
• The whole key to a shiny-looking material is providing something for it to reflect

Figure 2. (a) A shiny sphere rendered under photographically acquired real-world illumination. (b) The same sphere rendered under illumination by a point light source.
Reflection mapping

- A function from the sphere to colors, stored as a texture

[Blinn & Newell 1976]
Reflection mapping

- Interface (Lance Williams, 1985)

Video
Reflection mapping

• Flight of the Navigator (1986)
Reflection mapping

Reflection mapping

• Star Wars: Episode I *The Phantom Menace* (1999)
Creating environment maps

• An environment map is a spherical panoramic image

• Capturing a spherical panoramic image
  – Use a 360 degree camera
  – Take a picture of a mirrored ball (called a light probe)

Light Probes by Paul Debevec http://www.debevec.org/Probes/
Environment maps as light sources

• Key assumption: light captured by an environment map is emitted from infinitely far away

• As such, an environment map consists of directional light sources
  – An environment map value is defined for each direction, independent of position in scene
  – The same environment map is used at each point in the scene
Environment maps

Global illumination with pre-computed radiance transfer [Sloan et al. 2002]

Reflection mapping [Georg-Simon Ohm University of Applied Sciences]
Environment maps
Cube environment map

• Store incident light on six faces of a cube instead of on sphere
Cube environment map

Spherical map

Cube map
Cube vs. spherical maps

• Advantages of cube maps
  – More even texel sample density causes less distortion, allowing for lower resolution maps
  – Easier to dynamically generate cube maps for real-time simulated reflections
Cube environment map

• Cube map look-up
  – Given light direction \((x, y, z)\)
  – Largest coordinate component determines cube map face
  – Dividing by magnitude of largest component yields coordinates within face
  – In GLSL:
    • Use \((x, y, z)\) direction as texture coordinates to samplerCube
Environment mapping

Source: http://antongerdelan.net/opengl/cubemaps.html
Reflection maps

- Simulates mirror reflection
- Computes reflection vector at each pixel
- Use reflection vector to look up cube map
- Rendering cube map itself is optional (application dependent)
Reflection maps

Images from *Illumination and Reflection Maps: Simulated Objects in Simulated and Real Environments*
Gene Miller and C. Robert Hoffman
SIGGRAPH 1984 “Advanced Computer Graphics Animation” Course Notes
Reflection mapping in GLSL

• Application setup
  – Load and bind a cube environment map
    ```
    glBindTexture(GL_TEXTURE_CUBE_MAP, ...);
    glTexImage2D(GL_TEXTURE_CUBE_MAP_POSITIVE_X,...);
    glTexImage2D(GL_TEXTURE_CUBE_MAP_NEGATIVE_X,...);
    glTexImage2D(GL_TEXTURE_CUBE_MAP_POSITIVE_Y,...);
    ... 
    glEnable(GL_TEXTURE_CUBE_MAP);
    ```
#version 400

in vec3 vp; // positions from mesh
in vec3 vn; // normals from mesh
uniform mat4 P, V, M; // proj, view, model matrices
out vec3 pos_eye;
out vec3 n_eye;

void main()
{
    pos_eye = vec3(V * M * vec4(vp, 1.0));
    n_eye = vec3(V * M * vec4(vn, 0.0));
    gl_Position = P * V * M * vec4(vp, 1.0);
}
Environment mapping: fragment shader

#version 400

in vec3 pos_eye;
in vec3 n_eye;
uniform samplerCube cube_texture;
uniform mat4 V; // view matrix
out vec4 frag_colour;

void main()
{
    // reflect ray around normal from eye to surface
    vec3 incident_eye = normalize(pos_eye);
    vec3 normal = normalize(n_eye);

    vec3 reflected = reflect(incident_eye, normal);
    // convert from eye to world space
    reflected = vec3(inverse(V) * vec4(reflected, 0.0));

    frag_colour = texture(cube_texture, reflected);
}
Diffuse irradiance environment map

• Given a scene with $k$ directional lights, light directions $d_1, ..., d_k$, and intensities $i_1, ..., i_k$ illuminating a diffuse surface with normal $n$ and color $c$

• Pixel intensity $B$ is computed as $B = c \sum_{j=1 \ldots k} \max(0, d_j \cdot n)i_j$

• The cost of computing $B$ is proportional to the number of texels in environment map

• Observations
  – All surfaces with normal direction $n$ will return the same value for the sum
  – The sum is dependent on just the lights in the scene and the surface normal
Diffuse irradiance environment map

• Precompute diffuse reflections
  – Precompute sum for any normal $n$ and store result in a second environment map, indexed by surface normal
• Second environment map is called diffuse irradiance environment map
• Allows to illuminate objects with arbitrarily complex lighting environments with single texture lookup
Diffuse irradiance environment map

- Two cubic environment maps
  - Reflection map
  - Diffuse map

Diffuse shading vs. shading with diffuse map

Environment mapping

• Rendering with Natural Light (Paul Debevec, 1998)