Texturing

Outline

• Applications
  – Bump mapping
  – Environment mapping
Bump Mapping

- So far we’ve been thinking of textures modulating color only
  - Decal
  - Lightmap
- But any other per-pixel properties are fair game...
- Pixel normals usually smoothly varying
  - Computed at vertices for Gouraud shading; color interpolated
  - Interpolated from vertices for Phong shading
- Textures allow setting per-pixel normal with a bump map

Bump mapping: Why?

- Can get a lot more surface detail without expense of more object vertices to light, transform

[Images of a planet with and without bump mapping]
Bump Mapping: How?

- Idea: Perturb pixel normals \( \mathbf{n}(u, v) \) derived from object geometry to get additional detail for shading
- Compute lighting per pixel (like Phong)

Bump mapping: Representations

- 3-D vector \( \mathbf{m}(u, v) \) added directly to normal \( \mathbf{n} \)
- Or: 2-D vector of coefficients \( (b_u, b_v) \) that scale \( \mathbf{u}, \mathbf{v} \) vectors tangent to surface
Bump representation: Height map $f(u, v)$

- Store scalar “altitude” at each pixel
- Get $b_u$, $b_v$ from partial derivatives:
  \[
  b_u = \frac{\partial f}{\partial u} \quad b_v = \frac{\partial f}{\partial v}
  \]

Example: Converting height maps to normal displacements

Z coordinate set to some constant scale factor; (X, Y) normalized to [0, 1] range.
Right image is mostly blue because “straight up” vector is (0, 0, 1)
Bump mapping: Example

Height map  Bump texture applied to teapot
Environment/Reflection Mapping

- Problem: To render pixel on mirrored surface correctly, we need to follow reflection of eye vector back to first intersection with another surface and get its color
- This is an expensive procedure
- Idea: Approximate with texture mapping

Projecting in non-standard directions

- Don’t have to project ray from object center through position \((x, y, z)\)—can use any attribute of that position. For example:
  - Ray comes from another location
  - Ray is surface normal \(n\) at \((x, y, z)\)
  - Ray is reflection-from-eye vector \(r\) at \((x, y, z)\)
  - Etc.
Projecting in non-standard directions

• This can lead to interesting or informative effects

Environment mapping: Details

• Key idea: Render 360 degree view of environment from center of object with sphere or box as intermediate surface
• Intersection of eye reflection vector with intermediate surface provides texture coordinates for reflection/environment mapping
Making environment textures: Cube

• Cube map straightforward to make: Render/photograph six rotated views of environment
  – 4 side views at compass points
  – 1 straight-up view, 1 straight-down view

Making environment textures: Sphere

• Most often constructed with two photographs of mirrored sphere taken 90 degrees apart
Environment mapping: Issues

• Not physically correct to the extent that object shape differs from intermediate surface shape
• Looks better when environment objects are farther from object

Environment mapping: Example

courtesy of G. Miller
Environment mapping: Example

From “Terminator II”