Texture Mapping

CSE 457
Winter 2015
Reading

Required

- Angel, 7.4-7.10

Recommended


Optional

- Woo, Neider, & Davis, Chapter 9
Texture mapping

Due to Ed Catmull, PhD thesis, 1974
Refined by Blinn & Newell, 1976

A texture can modulate just about any parameter – diffuse color, specular color, specular exponent, …
Implementing texture mapping

A texture lives in its own abstract image coordinates parameterized by \((s,t)\) in the range \(([0..1],[0..1])\):

It can be wrapped around many different surfaces:

With a ray caster, we can do the sphere and cylinder mappings directly (as we will see later). For z-buffers, everything gets converted to a triangle mesh with associated \((s,t)\) coordinates.

Note: if the surface moves/deforms, the texture goes with it.
Texture coordinates on a surface of revolution

\[ t = \frac{\sum_{i=0}^{n-1} d_i}{\sum_{i=0}^{n-1} d_i} \]

\[ s = \frac{\Theta}{2\pi} \]

\[ t = \frac{n}{N} \]

Use this \( t \)
Mapping to texture image coords

The texture is usually stored as an image. Thus, we need to convert from abstract texture coordinate:

\[(s, t) \text{ in the range } ([0..1], [0..1])\]

to texture image coordinates:

\[(s_{\text{tex}}, t_{\text{tex}}) \text{ in the range } ([0.. w_{\text{tex}}], [0.. h_{\text{tex}}])\]

**Q:** What do you do when the texture sample you need lands between texture pixels?
Texture resampling

We need to resample the texture:

\[ t_{\text{tex}} = t \cdot h_{\text{tex}} \]

Thus, we seek to solve for:

\[ T(a, b) = T(i + \Delta_x, j + \Delta_y) \]

A common choice is \textit{bilinear interpolation}:

\[
T(i + \Delta_x, j) = (1 - \Delta_x) T[i, j] + \Delta_x T[i + 1, j]
\]

\[
T(i + \Delta_x, j + 1) = (1 - \Delta_x) T[i, j + 1] + \Delta_x T[i + 1, j + 1]
\]

\[
T(i + \Delta_x, j + \Delta_y) = (1 - \Delta_y) T(i + \Delta_x, j) + \Delta_y T(i + \Delta_x, j + 1)
\]

\[
= (1 - \Delta_x) (1 - \Delta_y) T[i, j] + \Delta_x (1 - \Delta_y) T[i + 1, j] + (1 - \Delta_x) \Delta_y T[i, j + 1] + \Delta_x \Delta_y T[i + 1, j + 1]
\]
Texture mapping and the z-buffer

Texture-mapping can also be handled in z-buffer algorithms.

**Method:**

- Scan conversion is done in screen space, as usual
- Each pixel is colored according to the texture
- Texture coordinates are found by Gouraud-style interpolation

**Note:** Mapping is more complicated to handle perspective correctly!
Displacement mapping

Textures can be used for more than just color.

In displacement mapping, a texture is used to perturb the surface geometry itself. Here’s the idea in 2D:

- These displacements “animate” with the surface
- In 3D, you would of course have (s,t) parameters instead of just s.

Suppose $Q$ is a simple surface, like a cube. Will it take more work to render the modified surface $\tilde{Q}$?
Bump mapping

In **bump mapping**, a texture is used to perturb the normal:

- Use the original, simpler geometry, \( Q(s) \), for hidden surfaces
- Use the normal from the displacement map for shading:

\[
\tilde{N} = \text{normal}[\tilde{Q}(s)]
\]

What artifacts in the images would reveal that bump mapping is a fake?
Displacement vs. bump mapping

Input texture

Rendered as displacement map over a rectangular surface
Displacement vs. bump mapping (cont'd)

Original rendering  Rendering with bump map wrapped around a cylinder

*Bump map and rendering by Wyvern Aldinger*
Solid textures

Q: What kinds of artifacts might you see from using a marble veneer instead of real marble?

One solution is to use **solid textures**:

- Use model-space coordinates to index into a 3D texture
- Like “carving” the object from the material

One difficulty of solid texturing is coming up with the textures.
Solid textures (cont'd)

Here's an example for a vase cut from a solid marble texture:

![Solid marble texture by Ken Perlin, (Foley, IV-21)](image-url)
Solid textures (cont'd)

\[
in(x,y,z) = \text{stripes}(x) \\
\text{shift}(x,y,z) = K \cdot \text{noise}(x,y,z) \\
out(x,y,z) = \text{stripes}(x + \text{shift}(x,y,z))
\]
Environment mapping

In environment mapping (also known as reflection mapping), a texture is used to model an object's environment:

- Rays are bounced off objects into environment
- Color of the environment used to determine color of the illumination
- Environment mapping works well when there is just a single object – or in conjunction with ray tracing

This can be readily implemented (without interreflection) using a fragment shader, where the texture is stored in a “cube map” instead of a sphere.

With a ray tracer, the concept is easily extended to handle refraction as well as reflection (and interreflection).
Summary

What to take home from this lecture:

1. The meaning of the boldfaced terms.

2. Familiarity with the various kinds of texture mapping, including their strengths and limitations.