Texture Mapping

Daniel Leventhal
Adapted from Brian Curless
CSE 457
Autumn 2011

Texture mapping

Angel, 8.6, 8.7, 8.9, 8.10, 9.13-9.13.2
Paul S. Heckbert. Survey of texture mapping.
IEEE Computer Graphics and Applications

Optional
Woo, Neider, & Davis, Chapter 9
James F. Blinn and Martin E. Newell. Texture
and reflection in computer generated images.
Communications of the ACM 19(10): 542–
547, October 1976.

Texture mapping allows you to take a simple polygon
and give it the appearance of something much more
complex.

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
paramaterized 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/defoms, the texture goes
with it.
Mapping to texture image coords

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

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

to texture image coordinates:

\((stex,ttex)\)

\((w_{tex},h_{tex})\)

Point on triangle mesh

Mapping to abstract texture coords

Mapping to texture pixel coords

\(Q\) lands between texture pixels?

Texture resampling

We need to resample the texture:

\[
T(x,y) = T(i + \Delta_x, j + \Delta_y)
\]

\(\Delta_x, \Delta_y\) are the displacement amounts.

Thus, we seek to solve for:

\[
T(i + \Delta_x, j + \Delta_y) = \frac{i}{w_{tex}} T(i, j) \quad \frac{j}{h_{tex}} T(i, j)
\]

A common choice is bilinear interpolation

\[
T(i + \Delta_x, j + \Delta_y) = \frac{i}{w_{tex}} T(i, j) + \frac{j}{h_{tex}} T(i, j)
\]

\[
= \frac{i}{w_{tex}} T(i + \Delta_x, j) + \frac{j}{h_{tex}} T(i, j + \Delta_y)
\]

\[
= \frac{i}{w_{tex}} T(i + \Delta_x, j + \Delta_y) + \frac{j}{h_{tex}} T(i, j + \Delta_y)
\]

\[
= \frac{i}{w_{tex}} T(i + \Delta_x, j + \Delta_y) + \frac{j}{h_{tex}} T(i, j + \Delta_y)
\]

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** the surface geometry itself. Here’s the idea in 2D:

\[
Q_0(i) = \text{normal} (Q(i))
\]

\[
Q(i) = Q(i) + d(i) N(i)
\]

- 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 \(Q^d\)?
Bump mapping

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

- Use the original, simpler geometry, $\rho$, for hidden surfaces
- Use the normal from the displacement map for shading:

$$\hat{N} = \text{normal}(\rho)$$

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

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](image)

Solid marble texture by Ken Perlin, (Foley, IV-21)

Solid textures (cont’d)

\[\text{shift}(x,y,z) \xrightarrow{K \text{ noise}(\cdot)} \text{stripes}(x)\]

\[\text{out}(x,y,z) \xrightarrow{\text{stripes}(x+\text{shift})} \]

Increasing K

Environment mapping

In environment mapping (also 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).

HDR Capture
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.