Texture Mapping

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

Texture mapping ensures that “all the right things” happen as a textured polygon is transformed and rendered.

Non-parametric texture mapping

With “non-parametric texture mapping”:

- Texture size and orientation are fixed
- They are unrelated to size and orientation of polygon
- Gives cookie-cutter effect

Reading

Required

- Angel, 7.6-7.8.

Recommended


Optional

- Woo, Neider, & Davis, Chapter 9
**Parametric texture mapping**

With “parametric texture mapping,” texture size and orientation are tied to the polygon.

**Idea:**
- Separate “texture space” and “screen space”
- Texture the polygon as before, but in texture space
- Deform (render) the textured polygon into screen space

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 \((u,v)\) in the range \([0..1], [0..1]\):

It can be wrapped around many different surfaces:

Computing \((u,v)\) texture coordinates in a ray tracer is fairly straightforward.

Note: if the surface moves/deforms, 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:

\((u,v)\) in the range \([0..1], [0..1]\)

to texture image coordinates:

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

**Texture resampling**

We need to resample the texture:

A common choice is **bilinear interpolation**:

\[
T(a,b) = T(i + \Delta_x, j + \Delta_y) = T[i, j] + T[i+1, j] + T[i, j+1] + T[i+1, j+1]
\]
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 if you want to do perspective right!

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)

Displacement mapping

Textures can be used for more than just color.

In displacement mapping, a texture is used to perturb the surface geometry itself:

- These displacements “animate” with the surface

Q: Do you have to do hidden surface calculations on \( \tilde{Q} \)?
Bump mapping

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

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

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

**Q**: 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*

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
- Really, a simplified form of ray tracing
- Environment mapping works well when there is just a single object – or in conjunction with ray tracing

Under simplifying assumptions, environment mapping can be implemented in hardware.

With a ray tracer, the concept is easily extended to handle refraction as well as reflection.
Combining texture maps

Using texture maps in combination gives even better effects, as Young Sherlock Holmes demonstrated …

Construction of the glass knight, (Foley, IV-24)

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.

3. Understanding of the various approaches to antialiased texture mapping:
   - Brute force
   - Mip maps
   - Summed area tables