Texture Mapping

Brian Curless
CSE 457
Autumn 2010

Reading

Required

- Angel, 8.6, 8.7, 8.9, 8.10, 9.13-9.13.2

Recommended


Optional

- Woo, Neider, & Davis, Chapter 9

Implementing texture mapping

A texture lives in it's own abstract image coordinates parameterized by $st$ in the range $[0,1]$: $[0,1]$:

It can be wrapped around many different surfaces:

With a ray tracer, 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 $st$ coordinates.

Note: if the surface moves/transforms the texture goes with it.

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, ...
Mapping to texture image coords

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

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

to texture image coordinates:

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

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

Texture resampling

We need to resample the texture:

\[T[i, j] + \Delta x \Delta y \in [0, 1]\]

Thus, we seek to solve for: \(T(i + \Delta x, j + \Delta y)\)

A common choice is **bilinear interpolation**:

\[T(i + \Delta x, j) = \frac{\Delta y}{\Delta y} T(i, j) + \frac{\Delta x}{\Delta x} T(i + 1, j)\]

\[T(i + \Delta x, j + 1) = \frac{\Delta y}{\Delta y} T(i, j + 1) + \frac{\Delta x}{\Delta x} T(i + 1, j + 1)\]

\[T(i + \Delta x, j + \Delta y) = \frac{\Delta y}{\Delta y} T(i + \Delta x, j) + \frac{\Delta x}{\Delta x} (T(i, j + \Delta y)) + \frac{\Delta x}{\Delta x} \Delta y T(i + 1, j + \Delta y)\]

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.

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 more parameters instead of just s.

Suppose \(Q\) is a simple surface, like a sphere. 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.

$$N = \text{normal}(Q(s))$$

What artifacts in the images would reveal that bump mapping is a failure?
- Perspective difference
- Occasions wrong
- Silhouette wrong
- Shadows wrong

**Displacement vs. bump mapping**

**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 of a vase cut from a solid marble texture:

![Solid marble texture](image)

---

Solid textures (cont'd)

\[
\begin{align*}
\text{input:} & (x,y,z) \\
\text{shift:} & (x,y,z) \\
\text{output:} & (x,y,z)
\end{align*}
\]

Increasing K

---

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 the environment.
- Color of the environment is 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 interpolation) 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 reflection as well as reflection (and refraction).

---

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.