There are limits to geometric modeling

Although modern GPUs can render millions of triangles/sec, that's not enough sometimes...
Use texture mapping to increase realism through detail.

This image is just 8 polygons!

Add visual complexity.

http://www.siggraph.org/education/materials/HyperGraph/mapping/r_wolfe/r_wolfe_mapping_1.htm
No texture

With texture

[Angel and Shreiner]
Textures can be anything that you can lookup values in -- photo, procedurally generated, or even a function that computes a value on the fly.
3D solid textures
Other uses of textures...

Light maps
Shadow maps
Environment maps
Bump maps
Opacity maps
Animation
Texture mapping in the OpenGL pipeline

- Geometry and pixels have separate paths through pipeline
- meet in **fragment processing** - where textures are applied
- texture mapping applied at end of pipeline - efficient since relatively few polygons get past clipper
uv Mapping

- 2D texture is parameterized by $(u,v)$
- Assign polygon vertices texture coordinates
- Interpolate within polygon

Texture coordinates are per-vertex data – a position in the $(u,v)$ space can interpolate tex coordinates with barycentric coordinates
Texture Calibration
The major issues in texture mapping...

- What should the actual mapping be?

Teapot: Which image looks better? The image on the left uses **object coordinates** in the texture mapping – this makes more sense. The image on the right uses **world coordinates** – texture ends up changing relative to the object. **Want a nice map that doesn't look distorted**
Given a point on the object \((x,y,z)\), what point \((u,v)\) in the texture we use?
Example: planar mapping
Intermediate surfaces

First map the texture to a simpler, intermediate surface
Cylindrical mapping

\[(x, y, z) \rightarrow (\theta, h) \rightarrow (u, v)\]

- note “pie slice” phenomena
- which coordinate axis is parallel to the cylinder axis?
Spherical Mapping

\((x,y,z) \rightarrow (\text{latitude, longitude}) \rightarrow (u,v)\)

spherical map stretches squares at equator and squeezes squares at poles
- similar to planar mapping
- planar projection -- choose which plane to project onto
How do we map between intermediate and actual objects?

We associated \((x,y,z)\) on the intermediate object with the texture \((u,v)\). But which point on the actual object is this?

We choose both the intermediate shape and the mapping from the actual shape to the intermediate shape:

1. a point on the object relative to its bounding box
2. see where surface normal intersects intermediate surface
3. shoot ray from centroid through surface point to intermediate surface
4. use the reflection vector (depends on the viewer position and normal)
How do we map between intermediate and actual objects?

Can you tell what intermediate shape was used?
Planar map - in xy plane
Cylindrical  

Spherical
Parametric Surfaces

32 parametric patches
3D solid textures

can map object \((x, y, z)\) directly to texture \((u, v, w)\)
Procedural textures

e.g., Perlin noise
Triangles
Texturing triangles

- Store \((u,v)\) at each vertex
- Interpolate inside triangles using barycentric coordinates

\[\begin{array}{c}
\text{Texture Space} \\
0, 0 & | & 1, 1 \\
\hline
0, 1 & & 1, 1 \\
\end{array}\]

\[\begin{array}{c}
\text{Object Space} \\
(0.4, 0.2) & & (0.8, 0.4) \\
\hline
(0.2, 0.8) & & \\
\end{array}\]
Texturing triangles

• Store \((u,v)\) at each vertex
• Interpolate inside triangles using barycentric coordinates

\[
\begin{align*}
\mathbf{p}(\beta, \gamma) &= \mathbf{a} + \beta(\mathbf{b} - \mathbf{a}) + \gamma(\mathbf{c} - \mathbf{a}). \\
u(\beta, \gamma) &= u_a + \beta(u_b - u_a) + \gamma(u_c - u_a), \\
v(\beta, \gamma) &= v_a + \beta(v_b - v_a) + \gamma(v_c - v_a).
\end{align*}
\]
Texturing triangles

Choice of \((u,v)\) makes big difference
Texturing triangles

Choice of \((u,v)\) makes big difference

Texture extended through “tiling”
Texturing triangles

Choice of \((u,v)\) makes big difference
Textures in OpenGL

- Assign (u,v) to vertices
- OpenGL then uses interpolation for triangle interior

- Good selection of tex coordinates
- Poor selection of tex coordinates
- Texture stretched over trapezoid showing effects of bilinear interpolation
Multitexturing
Texture Sampling
• Texture coordinates: Used to identify points in the image to be mapped
• Object Coordinates: Conceptually, where the mapping takes place
• Window Coordinates: Where the final image is really produced
Point Sampling

Map back to texture image and use the **nearest texel**
Aliasing

**Point sampling** of the texture can lead to aliasing artifacts.

Point sampling in (or x,y,z) space can cause blue stripes to be missed in the texture space.

[Angel and Shreiner]
Magnification and Minification
Magnification and Minification

More than one texel can cover a pixel (*minification*) or more than one pixel can cover a texel (*magnification*).

Can use point sampling (nearest texel) or linear filtering (2x2 filter) to obtain texture values.
Aliasing artifacts

We apply filtering to reduce aliasing artifacts
Area Averaging

A better but slower option is to use **area averaging**
Use bilinear filtering

mitigate magnification artifacts

smooths out the texture – no sharp boundaries
Mipmapping

Reduce minification artifacts
Prefilter the texture to obtain reduced resolutions
Requires 1/3 more space
Get a texture hierarchy indexed by level

128x128, 64x64, 32x32, 16x16, 8x8, 4x4, 2x2, 1x1

Togikun, Wikimedia Commons
point sampling

mipmapped point sampling

linear filtering

mipmapped linear filtering

[Angel and Shreiner]