Texture Mapping

Texture Mapping • A way of adding surface details

Two ways can achieve the goal:
- Model the surface with more polygons
  » Slows down rendering speed
  » Hard to model fine features
- Map a texture to the surface
  » This lecture
  » Image complexity does not affect complexity of processing

Efficiently supported in hardware

Trompe L’Oeil (“Deceive the Eye”)

Windows and columns in the dome are painted, not a real 3D object
Similar idea with texture mapping:
Rather than modeling the intricate 3D geometry, replace it with an image!

Map textures to surfaces

The texture

Texture is a bitmap image
- Can use an image library to load image into memory
- Or can create images yourself within the program

2D array:
unsigned char texture[height][width][4]

Or unrolled into 1D array:
unsigned char texture[4*height*width]

Pixels of the texture are called texels
Texel coordinates (s,t) scaled to [0,1] range
The “st” coordinate system

Texture mapping: key slide

Specifying texture coordinates in OpenGL

What if texture coordinates are outside of [0,1]?
Solution 1: Repeat texture

\[
\begin{align*}
glTexParameteri(GL\_TEXTURE\_2D, GL\_TEXTURE\_WRAP\_S, GL\_REPEAT) \\
glTexParameteri(GL\_TEXTURE\_2D, GL\_TEXTURE\_WRAP\_T, GL\_REPEAT)
\end{align*}
\]

Solution 2: Clamp to \([0,1]\)

\[
\begin{align*}
glTexParameteri(GL\_TEXTURE\_2D, GL\_TEXTURE\_WRAP\_S, GL\_CLAMP) \\
glTexParameteri(GL\_TEXTURE\_2D, GL\_TEXTURE\_WRAP\_T, GL\_CLAMP)
\end{align*}
\]

Combining texture mapping and shading

- Final pixel color = a combination of texture color and color under standard OpenGL Phong lighting
  - GL\_MODULATE: multiply texture and Phong lighting color
  - GL\_BLEND: linear combination of texture and Phong lighting color
  - GL\_REPLACE: use texture color only (ignore Phong lighting)

Example:

\[
glTexEnvf(GL\_TEXTURE\_ENV, GL\_TEXTURE\_ENV\_MODE, GL\_REPLACE);
\]

Outline

- Introduction
- Texture mapping in OpenGL
- Filtering and Mipmaps
- Example
- Non-color texture maps

Texture mapping in OpenGL

- During your initialization:
  1. Read texture image from file into an array in memory, or generate the image using your program
  2. Specify texture mapping parameters
     » Wrapping, filtering, etc.
  3. Initialize and activate the texture

- In display():
  1. Enable OpenGL texture mapping
  2. Draw objects: Assign texture coordinates to vertices
  3. Disable OpenGL texture mapping
Initializing the texture

- Do once during initialization, for each texture image in the scene, by calling `glTexImage2D`
- The dimensions of texture images must be powers of 2
  - If not, rescale image or pad with zero
  - Or can use OpenGL extensions
- Can load textures dynamically if GPU memory is scarce

Enable/disable texture mode

- Must be done before rendering any primitives that are to be texture-mapped
- `glEnable(GL_TEXTURE_2D)`
- `glDisable(GL_TEXTURE_2D)`
- Successively enable/disable texture mode to switch between drawing textured/non-textured polygons
- Changing textures:
  - Only one texture is active at any given time (with OpenGL extensions, more than one can be used simultaneously; this is called multitexturing)
  - Use `glBindTexture` to select the active texture

Outline

- Introduction
- Texture mapping in OpenGL
- Filtering and Mipmaps
- Example
- Non-color texture maps

Texture interpolation

- `(s,t)` coordinates typically not directly at pixel in the texture, but in between
- Solutions:
  - Use the nearest neighbor to determine color
  - Faster, but worse quality
  - `glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_NEAREST)`
  - Linear interpolation
  - Incorporate colors of several neighbors to determine color
  - Slower, better quality
  - `glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR)`
Filtering

- Texture image is shrunk in distant parts of the image
- This leads to aliasing
- Can be fixed with filtering
  - bilinear in space
  - trilinear in space and level of detail (mipmapping)

Mipmapping

- Pre-calculate how the texture should look at various distances, then use the appropriate texture at each distance
- Reduces / fixes the aliasing problem

Mipmapping in OpenGL

- `gluBuild2DMipmaps(GL_TEXTURE_2D, components, width, height, format, type, data)`
  - This will generate all the mipmaps automatically
- `glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_NEAREST_MIPMAP_NEAREST)`
  - This will tell GL to use the mipmaps for the texture

Outline

- Introduction
- Texture mapping in OpenGL
- Filtering and Mipmaps
- Example
- Non-color texture maps

Complete example

```c
void initTexture()
{
    load image into memory; // can use libjpeg, libtiff, or other image library
    // image should be stored as a sequence of bytes, usually 3 bytes per pixel (RGB), or 4 bytes (RGBA); image size is 4 * 256 * 256 bytes in this example
    // we assume that the image data location is stored in pointer "pointerToImage"

    // create placeholder for texture
    glGenTextures(1, &texName); // must declare a global variable in program header: GLuint texName
    glBindTexture(GL_TEXTURE_2D, texName); // make texture "texName" the currently active texture

    // (continues on next page)
}```
Complete example (part 2)

// specify texture parameters (they affect whatever texture is active)
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_REPEAT); // repeat pattern in s
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_REPEAT); // repeat pattern in t

// use linear filter both for magnification and minification
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR);

// load image data stored at pointer "pointerToImage" into the currently active texture ("texName")
gTexImage2D(GL_TEXTURE_2D, 0, GL_RGBA, 256, 256, 0, GL_RGBA, GL_UNSIGNED_BYTE, pointerToImage);

} // end init()

Complete example (part 3)

void display()
{
  ...
  // no modulation of texture color with lighting; use texture color directly
  glTexEnvf(GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE, GL_REPLACE);

  // turn on texture mapping (this disables standard OpenGL lighting, unless in GL_MODULATE mode)
glEnable(GL_TEXTURE_2D);

  (continues on next page)

}

(continues on next page)

Complete example (part 4)

glBegin(GL_QUADS); // draw a textured quad
  glTexCoord2f(0.0,0.0); glVertex3f(-2.0,-1.0,0.0);
gTexCoord2f(0.0,1.0); glVertex3f(-2.0,1.0,0.0);
gTexCoord2f(1.0,0.0); glVertex3f(0.0,1.0,0.0);
gTexCoord2f(1.0,1.0); glVertex3f(0.0,-1.0,0.0);
gEnd();

// turn off texture mapping
glDisable(GL_TEXTURE_2D);

// draw some non-texture mapped objects (standard OpenGL lighting will be used if it is enabled)
  ...
  // switch back to texture mode, etc.
  ...
} // end display()

Outline

• Introduction
• Texture mapping in OpenGL
• Filtering and Mipmaps
• Example
• Non-color texture maps

Textures do not have to represent color

• Specularity (patches of shininess)
• Transparency (patches of clearness)
• Normal vector changes (bump maps)
• Reflected light (environment maps)
• Shadows
• Changes in surface height (displacement maps)

Bump mapping

• How do you make a surface look rough?
  – Option 1: model the surface with many small polygons
  – Option 2: perturb the normal vectors before the shading calculation
    » Fakes small displacements above or below the true surface
    » The surface doesn’t actually change, but shading makes it look like there are irregularities!
    » A texture stores information about the “fake” height of the surface

Real Bump  Fake Bump
Bump mapping

- We can perturb the normal vector without having to make any actual change to the shape.
- This illusion can be seen through—how?

![Original model](original.png) ![Simplified](simplified.png) ![Simplified with bump map](simplified_bump.png)

Light Mapping

- Quake uses light maps in addition to texture maps. Texture maps are used to add detail to surfaces, and light maps are used to store pre-computed illumination. The two are multiplied together at runtime, and cached for efficiency.

![Texture Map Only](texture_only.png) ![Texture + Light Map](texture_light.png)

Summary

- Introduction
- Texture mapping in OpenGL
- Filtering and Mipmaps
- Example
- Non-color texture maps