Texture Mapping
Texture Mapping

- An effective method for adding surface detail, mapping texture patterns on the surface of (smooth) objects

- Real objects have details with very high frequencies
  - The surface of an orange is bumpy
  - The surface of a wooden table carries the wood’s colors
  - Mirrors and other shiny surfaces reflect the environment

- Idea: Modify the shading equation at several places
  - Surface color
  - Surface normal
  - Transparency
  - Reflectance
(Anti-)Aliasing

Antialiasing demo
(Anti-)Aliasing
Texture Coordinates

- Associates a portion of the texture with a polygon
- Assign texture coordinates to polygon vertices
texels
Interpolating Texture Coordinates

\[ c \, C_1 + d \, C_2 + e \, C_3 \]

\[(c + d + e = 1)\]

\[ a \, C_1 + (1 - a) \, C_2 \]

\[ b \, C_1 + (1 - b) \, C_3 \]
Parameterization

- We need a mapping between the model and the image

\[ F : Model \in \mathbb{R}^3 \rightarrow Image \in \mathbb{R}^2 \]

- With parametric surfaces this mapping is trivial
  - Why?

- What if we have polygonal models?
  - The mapping is not trivial at all
  - This process is called parameterization.
  - An active research topics in CG
Parameterization

- Transform the mesh into a canonical shape
  - Part of plane, sphere, cylinder
Parameterization

- Transform the mesh into a canonical shape
  - Part of plane, sphere, cylinder
- Not all parameterizations are good
  - Bijection
  - Minimize distortion
  - The trivial ones you learned about in lecture are not always good (why?)
- What can be done?
  - Conservation of (relative) distances (isometry)
    - Not always possible
  - Conservation of angles (conformal maps)
    - Not always what you want.
  - Other ideas?
Parameterization

- Flattened version of the camel on the right
- The colors encode the position in space
Computing texture coordinates

- During scan-conversion The texture is sampled
- Assign texture coordinates
  - A mapping of each vertex to the image
  - Interpolate the texture coordinates
    - Same way color is done

Texture Coordinates

(u₀, v₀) (u₁, v₁) (u₂, v₂)

(0,0) (1,0) (0,1) (1,1)
Sampling Scheme

How do we sample?

- Not every point on the surface has a pixel on the image
- Given an image and a real coordinate return a value

Common schemes

- Nearest neighbor
  - Return the pixel value that is the closest
- Bilinear interpolation
  - Given \((u,v)\) find its four neighbor pixels
  - Compute interpolation parameters
  - Compute the final color as a blend
- Higher order sampling schemes
  - Bicubic interpolation
  - Gaussian kernels
  - Might be too expensive in real-time apps
Bilinear Interpolation
Sampling Schemes

What is the parametric domain?

- $[0,1] \times [0,1]$
- So what does it mean to have texture coordinates of $(1.1, 2)$
- Clamping Vs Repeating
Texture Aliasing

many texels to one pixel

one texel to many pixels
Sampling Schemes

- **Mipmapping**
  - MIP – Multum In Parvo
  - “Much in a small space”

- **The idea:**
  - Hold a pyramid of images (a mipmap)
  - Choose a layer based on distance
  - What is it good for?

- **Trilinear interpolation**
  - Interpolate between layer as well

- **Anisotropic interpolation**
  - What?
Textures in OpenGL

- GLSL
  - Vertex shader

```glsl
in vec2 texcoord;
in vec4 vPosition;
...
out vec2 st;
...
void main()
{
  ...
  gl_Position=vPosition;
  st=texcoord;
}
```
Textures in OpenGL

- GLSL
  - Fragment shader

```glsl
in vec2 st;
uniform sampler2D texMap
out vec4 color;
void main()
{
    color = texture2D(texMap, st);
}
```
Textures in OpenGL

- **OpenGL**
  - Activate texture unit, create new texture object and bind it

```c
glActiveTexture(GL_TEXTUREi);
GLuint tex;
glGenTextures(1, &tex);
glBindTexture(GL_TEXTURE2D, tex)
```

- **Fill the texture buffer**

```c
GLubyte texels[512][512][3];
glTexImage2D(GL_TEXTURE_2D, 0, GL_RGB, 512, 512, 0, GL_RGB, GL_UNSIGNED_BYTE, texels);
```
Textures in OpenGL

- OpenGL
  - Set texture coordinates as vertex attributes
    ```cpp
    texcoord_loc = glGetUniformLocation(program, "texcoord");
    glEnableVertexAttribArray(texcoord_loc);
    glVertexAttribPointer(texcoord_loc, 2, GL_FLOAT, GL_FALSE, 0, 0);
    ```

- Set texture map
  ```cpp
  tex_loc = glGetUniformLocation(program, "texMap");
  glUniform1i(tex_loc, i);
  ```