Announcements

Graded:

Written Assignment – **Joel**

Michael is out this week

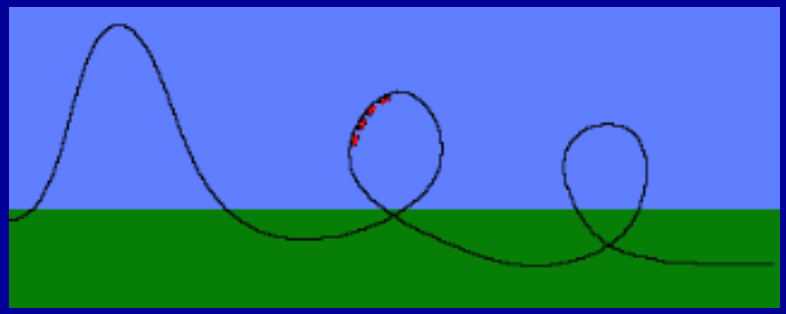
Written part of the second programming assignment is due Today before the class or Friday before 9am in Jessica's mailbox

Basics of Textures

Basics of texture mapping in OpenGL

Roller coaster

- **Model ground and sky with textures**



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

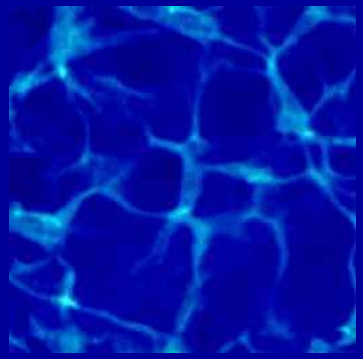
Texture Mapping

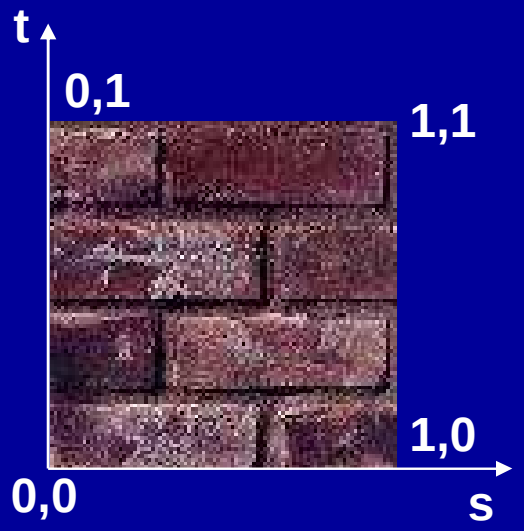
- A way of adding surface details

- 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

The texture

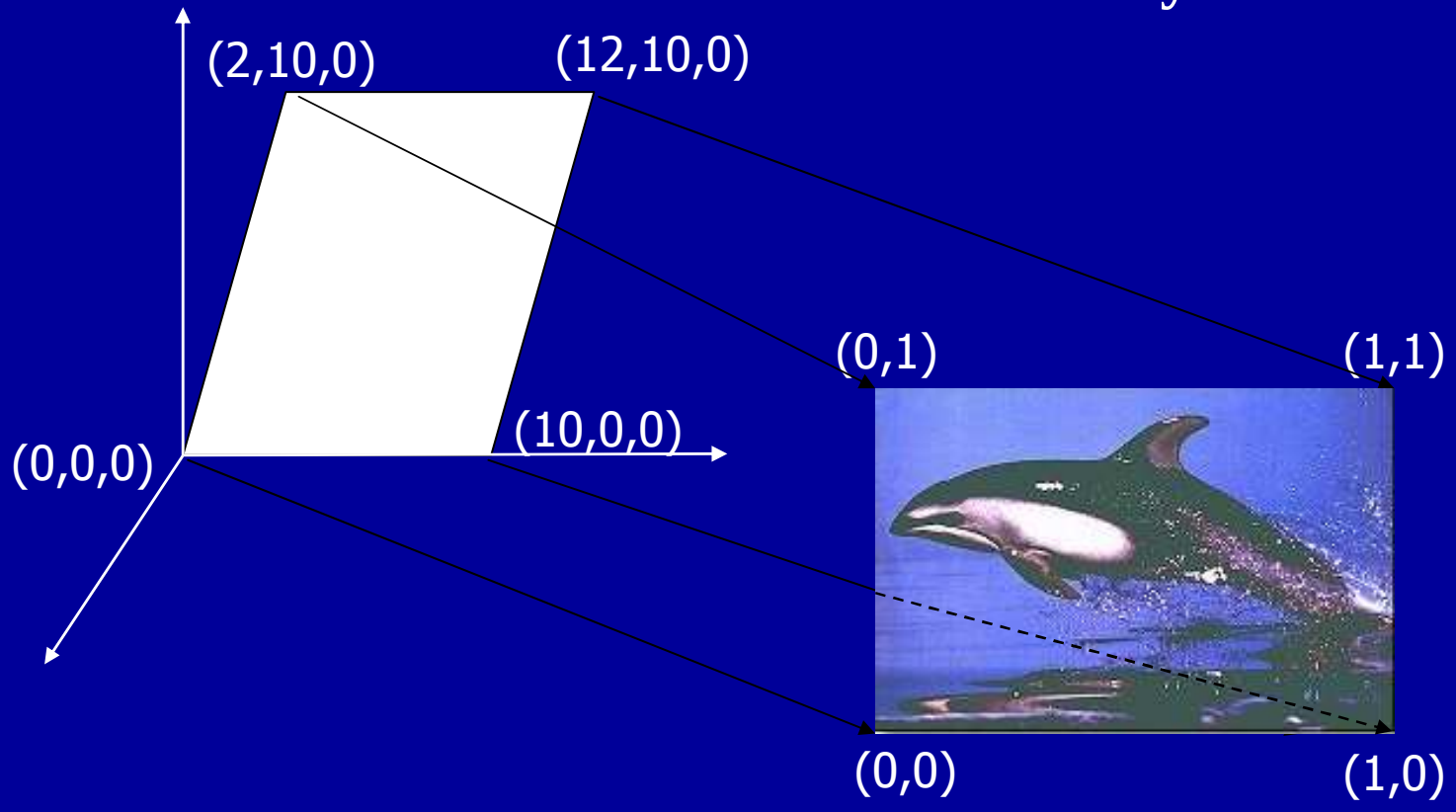
- **Texture is a bitmap image**
- **2D array: texture[height][width][4]**
- **Pixels of the texture called *texels***
- **Texel coordinates (s,t) scaled to [0,1] range**





Map textures to surfaces

The polygon can have arbitrary size and shape



The drawing itself

- **Use GLTexCoord2f(s,t) to specify texture coordinates**
- **Example:**

```
glEnable(GL_TEXTURE_2D)
glBegin(GL_QUADS);
glTexCoord2f(0.0,0.0);  glVertex3f(0.0,0.0,0.0);
glTexCoord2f(0.0,1.0);  glVertex3f(2.0,10.0,0.0);
glTexCoord2f(1.0,0.0);  glVertex3f(10.0,0.0,0.0);
glTexCoord2f(1.0,1.0);  glVertex3f(12.0,10.0,0.0);
glEnd();
glDisable(GL_TEXTURE_2D)
```

- **State machine: Texture coordinates remain valid until you change them or exit texture mode via**

  glDisable (GL_TEXTURE_2D)

Color blending

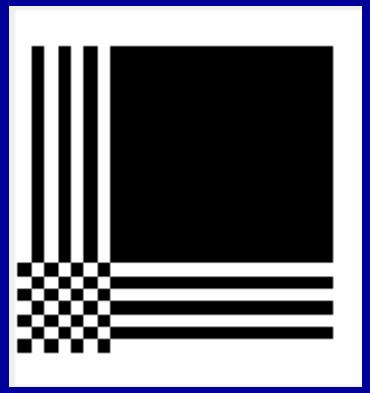
- **Final pixel color = f (texture color, object color)**
- **How to determine the color of the final pixel?**
  - **GL_REPLACE – use texture color to replace object color**
  - **GL_BLEND – linear combination of texture and object color**
  - **GL_MODULATE – multiply texture and object color**
- **Example:**
  - **glTexEnvf(GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE, GL_REPLACE);**

What happens if texture coordinates outside [0,1] ?

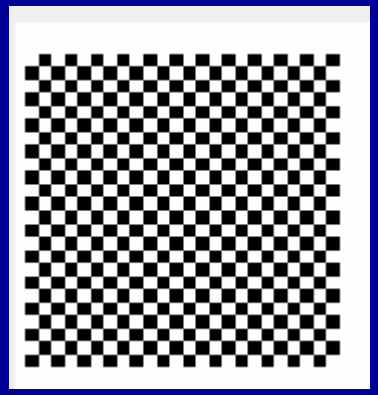
- **Two choices:**
  - **Repeat pattern (GL_REPEAT)**
  - **Clamp to maximum/minimum value (GL_CLAMP)**

- **Example:**
  - **glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_CLAMP)**
  - **glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_CLAMP)**

What happens if texture coordinates outside [0,1] ?



clamp

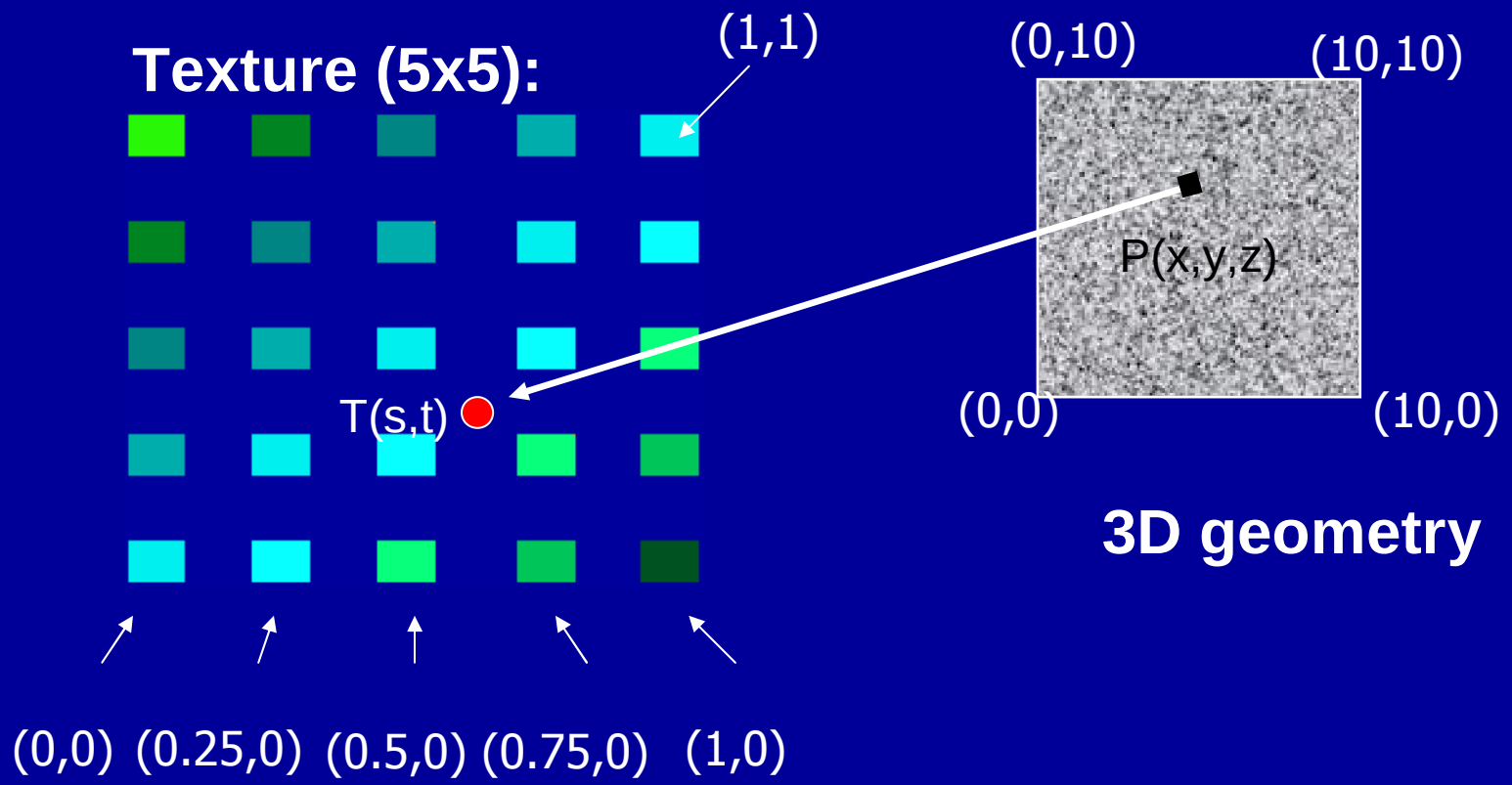


repeat

```
glTexCoord2f(0.0, 0.0);  glVertex3f(0.0,  0.0,  0.0);
glTexCoord2f(0.0, 3.0);  glVertex3f(0.0, 10.0, 0.0);
glTexCoord2f(3.0, 0.0);  glVertex3f(10.0, 0.0, 0.0);
glTexCoord2f(3.0, 3.0);  glVertex3f(10.0, 10.0, 0.0);
```

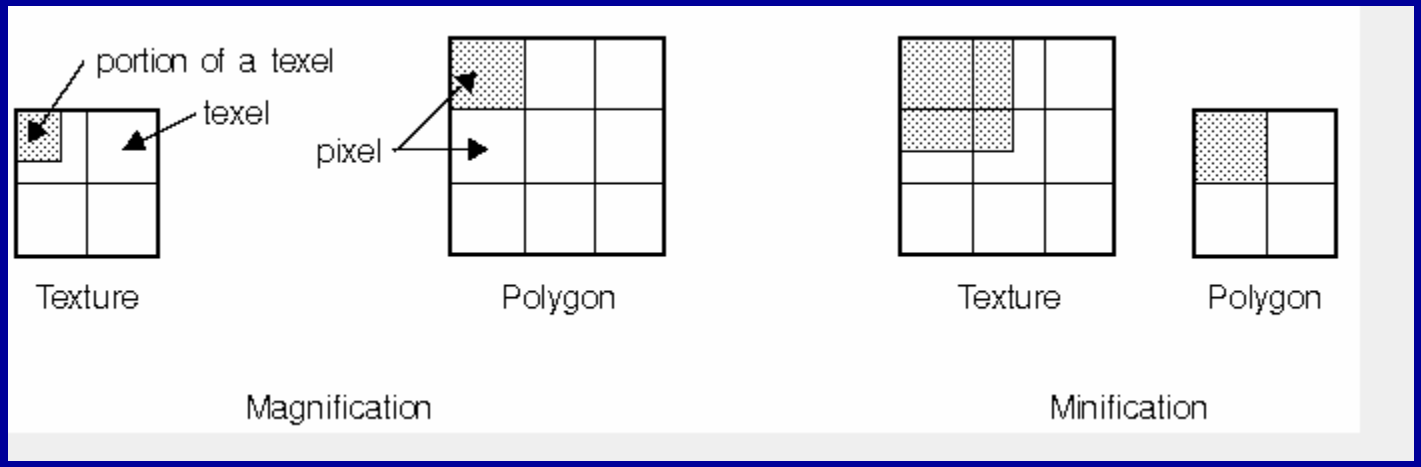
Texture Value Lookup

- For given texture coordinates (s,t), we can find a unique image value, corresponding to the texture image at that location



Interpolating colors

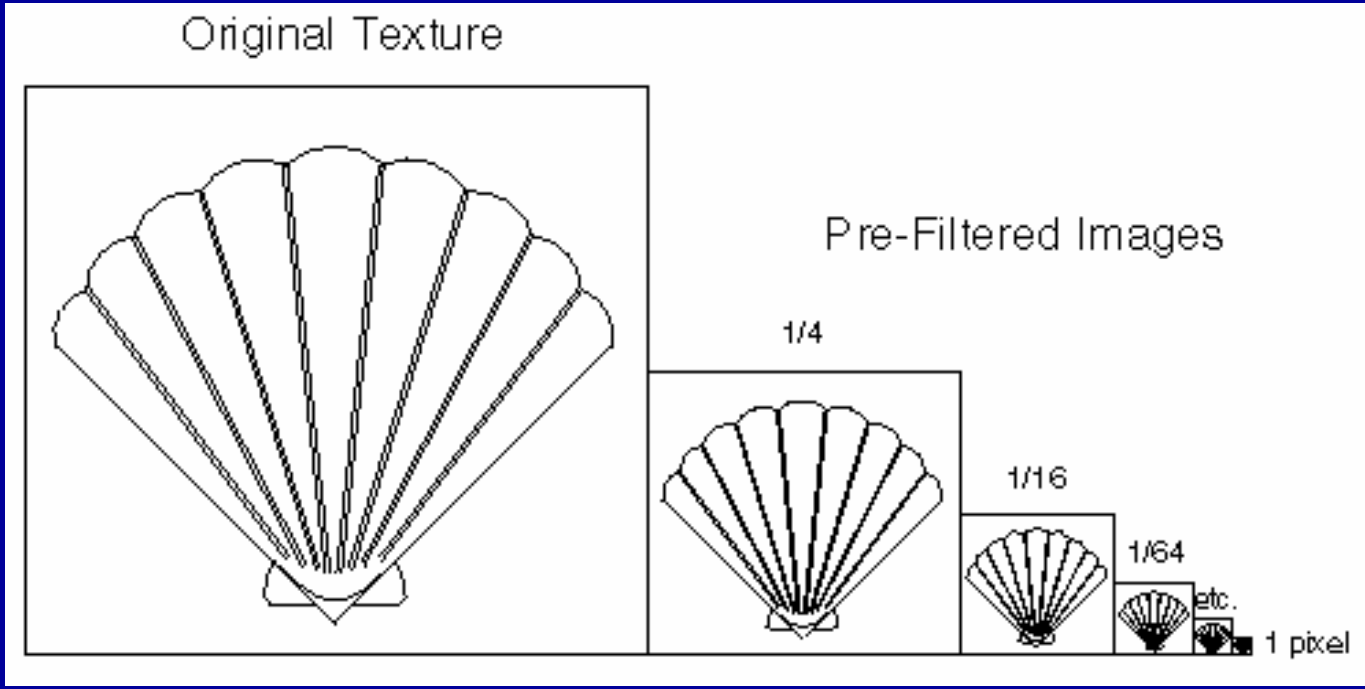
- **Some (s,t) coordinates not directly at pixel in the texture, but in between**



Interpolating colors

- **Solutions:**
  - **Nearest neighbor**
    - » **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)**

Texture Levels



Texture mapping in OpenGL

- **In init():**
  - **Specify texture**
    - » **Read image from file into an array in memory or generate the image using the program**
  - **Specify texture mapping parameters**
    - » **Wrapping, filtering, etc.**
  - **Define (activate) the texture**
- **In display():**
  - **Enable GL texture mapping**
  - **Draw objects: Assign texture coordinates to vertices**
  - **Disable GL texture mapping**

Specifying texture mapping parameters

- **Use glTexParameteri**
- **Example:**

```
// texture wrapping on

glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S,
GL_REPEAT); // repeat pattern in s texture coordinate

glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T,
GL_REPEAT); // repeat pattern in t texture coordinate


// use nearest neighbor for both minification and magnification

glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER,
GL_NEAREST);

glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER,
GL_NEAREST);
```

Defining (activating) texture

- Do once in init() to set up initial pattern
- To use another texture, make further calls in display() to glTexImage2D, specifying another image
  - But this is slow: use Texture Objects itself
- The dimensions of texture images must be powers of 2
  - if not, rescale image or pad with zeros
- glTexImage2D(Glenum target, Glint level,
  Glint internalFormat, int width, int height, Glint border,
  Glenum format, Glenum type, Glvoid* img)
- Example:
  - glTexImage2D(GL_TEXTURE_2D, 0,
    GL_RGBA, 256, 256, 0,
    GL_RGBA, GL_UNSIGNED_BYTE, pointerToImage)

Enable/disable texture mode

- Can do in init() or successively in display()
- 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 active at any given time
  - make another call to glTexImage2D to make another pattern active

The drawing itself

- Use GLTexCoord2f(s,t) to specify texture coordinates
- State machine: Texture coordinates remain valid until you change them or exit texture mode via

  glDisable (GL_TEXTURE_2D)
- Example:

```
glEnable(GL_TEXTURE_2D)
glBegin(GL_QUADS);
glTexCoord2f(0.0,0.0); glVertex3f(-2.0,-1.0,0.0);
glTexCoord2f(0.0,1.0); glVertex3f(-2.0,1.0,0.0);
glTexCoord2f(1.0,0.0); glVertex3f(0.0,1.0,0.0);
glTexCoord2f(1.0,1.0); glVertex3f(0.0,-1.0,0.0);
…
glEnd();
glDisable(GL_TEXTURE_2D)
```

Everything together

```
void init(void):
{
…
put image into 2D memory array; // can use libpicio library

// specify texture parameters
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 nearest neighbor for both minification and magnification
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_NEAREST);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_NEAREST);

// make the pattern at location pointerToImage the active pattern
glTexImage2D(GL_TEXTURE_2D, 0, GL_RGBA, 256, 256, 0,
             GL_RGBA, GL_UNSIGNED_BYTE, pointerToImage)

…
}
```

Everything together (contd.)

```
void display(void):
{
…
// no blending, use texture color directly
glTexEnvf(GL_TEXTURE_ENV,GL_TEXTURE_ENV_MODE, GL_REPLACE);
// turn on texture mode
glEnable(GL_TEXTURE_2D);

glBegin(GL_QUADS); // draw a quad
glTexCoord2f(0.0,0.0); glVertex3f(-2.0,-1.0,0.0);
glTexCoord2f(0.0,1.0); glVertex3f(-2.0,1.0,0.0);
glTexCoord2f(1.0,0.0); glVertex3f(0.0,1.0,0.0);
glTexCoord2f(1.0,1.0); glVertex3f(0.0,-1.0,0.0);
…
glEnd();

// turn off texture mode
glDisable(GL_TEXTURE_2D);

// draw some non-texture mapped objects
…
// switch back to texture mode, etc.
…
}
```