**Agenda**

- Finish last Monday’s slides
  - VBO Layout
  - OpenGL Multithreading
- Today’s slides
  - OpenGL Textures and Multitexturing
  - OpenGL Framebuffers and Deferred Shading
  - Ambient Occlusion

**Textures**

```c
unsigned char *pixels = // ...
GLuint id;
glGenTextures(1, &id);
glBindTexture(GL_TEXTURE_2D, id);
glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR);
glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);
glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_REPEAT);
glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_REPEAT);
glTexImage2D(GL_TEXTURE_2D, 0, GL_RGBA, width, height, 0, GL_RGBA, GL_UNSIGNED_BYTE, pixels);
// ...
glDeleteTextures(1, &id);
```
Textures

unsigned char *pixels = // ...
GLuint id;
glGenTextures(1, &id);

GLuint id;
glGenTextures(1, &id);
gTexParameter(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR);
gTexParameter(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);
gTexParameter(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_REPEAT);
gTexParameter(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_REPEAT);
gTexImage2D(GL_TEXTURE_2D, 0, GL_RGBA, width, height, 0, GL_BGRA,
GL_UNSIGNED_BYTE, pixels);
// ...
gDeleteTextures(1, id);

gTexImage2D(GL_TEXTURE_2D, 0, GL_RGBA, width, height, 0, GL_BGRA,
GL_UNSIGNED_BYTE, pixels);
// ...
gDeleteTextures(1, id);

I hate global state. You should too. What is the alternative design?

Images for an image in system memory.

Standard business.

Sampler state. More info to follow.

unsigned char *pixels = // ...
GLuint id;
glGenTextures(1, &id);

GLuint id;
glGenTextures(1, &id);
gTexParameter(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR);
gTexParameter(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);
gTexParameter(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_REPEAT);
gTexParameter(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_REPEAT);
gTexImage2D(GL_TEXTURE_2D, 0, GL_RGBA, width, height, 0, GL_BGRA,
GL_UNSIGNED_BYTE, pixels);
// ...
gDeleteTextures(1, id);
Textures

unsigned char *pixels = // ...
GLuint id;
glGenTextures(1, &id);
glBindTexture(GL_TEXTURE_2D, id);
gTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR);
gTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);
gTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_REPEAT);
gTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_REPEAT);
gTexImage2D(GL_TEXTURE_2D, 0, GL_RGBA, width, height, 0, GL_BGRA, GL_UNSIGNED_BYTE, pixels);
// ...
gDeleteTextures(1, &id);

Transfer from system memory to driver-controlled (likely, GPU) memory. Does it need to block?

Pixel data format and datatype

Textures

unsigned char *pixels = // ...
GLuint id;
glGenTextures(1, &id);
glBindTexture(GL_TEXTURE_2D, id);
gTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR);
gTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);
gTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_REPEAT);
gTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_REPEAT);
gTexImage2D(GL_TEXTURE_2D, 0, GL_RGBA, width, height, 0, GL_BGRA, GL_UNSIGNED_BYTE, pixels);
// ...
gDeleteTextures(1, &id);

Internal (GPU) texture format

Texture Wrap Parameters

GL_CLAMP
GL_REPEAT
GL_MIRRORED_REPEAT

Multitexturing

- Using multiple textures in the same rendering pass
- Each is bound to a different texture unit and accessed with a different sampler uniform in GLSL

Multitexturing: Light Map

- Recall our Light Map example:

```glsl
uniform sampler2D lightMap;
uniform sampler2D surfaceMap;
in vec2 fs_TxCoord;
in vec3 out_Color;
void main()
{
    float intensity = texture(lightMap, fs_TxCoord).r;
    vec3 color = texture(surfaceMap, fs_TxCoord).rgb;
    out_Color = intensity * color;
}
```

Each texture is accessed with a different sampler.

Pass the sampler to `texture()` to read from a particular texture.
Multitexturing: Terrain

- How was this rendered?

First hint: two textures

Grass  Stone

Second hint: terrain slope

• 0 is flat
• 1 is steep

Images courtesy of A K Peters, Ltd. www.virtualglobebook.com
Multitexturing: Terrain

- Third and final hint: a blend ramp

```glsl
uniform sampler2D grass;
uniform sampler2D stone;
uniform sampler2D blendRamp;

in vec3 out_Color;

void main()
{
    // ...
    out_Color = intensity * mix(
        texture(grass, repeatTextureCoordinate).rgb,
        texture(stone, repeatTextureCoordinate).rgb,
        texture(u_blendRamp, vec2(0.5, slope).r));
}
```

- Three samplers
- `blendRamp` could be 1D; it is just 1xn

Use terrain slope to look up a blend value in the range [0, 1].
Multitexturing: Terrain

```glsl
uniform sampler2D grass;
uniform sampler2D stone;
uniform sampler2D blendRamp;
in vec3 out_color;
void main(void)
{
    // ...
    out_color = intensity * mix(
        texture(grass, repeatTextureCoordinate).rgb,
        texture(stone, repeatTextureCoordinate).rgb,
        texture(blendRamp, vec2(0.5, slope)).r);
}
```

Linearly blend between grass and stone

Multitexturing: Globe

- How will you render this?

[Day texture](http://planetpixelemporium.com)
[Day Night](http://planetpixelemporium.com)
Multitexturing: Globe

- Videos
  - Night and Day
  - Clouds
  - Specularity

Framebuffer Objects (FBOs)

- Framebuffer Objects (FBOs)
  - Allow fragment shader to write to one or more off-screen buffers
  - Can then use the off-screen buffer as a texture in a later rendering pass
  - Allows render to texture
  - Don’t worry about the OpenGL API; we’ve already coded it for you

Framebuffer Objects (FBOs)

- FBOs are lightweight containers of textures

![FBO Diagram]

Framebuffer Objects (FBOs)

- FBO use case: post processing effects
  - Render scene to FBO with depth and color attachment
  - Render a viewport-aligned quad with texture that was the color attachment and apply effect
  - How would you design a post processing framework?
Deferred Shading

- FBO use case: *deferred shading*
  - Render scene in two passes
    - 1st pass: Visibility tests
    - 2nd pass: Shading

Deferred Shading

- 1st Pass: Render geometry into **G-Buffers**
  - Fragment Colors
  - Normals
  - Depth
  - Edge Weight

Deferred Shading

- 2nd pass: shading == post processing effects
- Render viewport-aligned quads that read from G-Buffers
- Objects are no longer needed

Deferred Shading

- Light accumulation result
Deferred Shading

- What are the benefits:
  - Shading and depth complexity?
  - Memory requirements?
  - Memory bandwidth?
  - Material and light decoupling?

Ambient Occlusion

- Ambient Occlusion (AO)
  - "shadowing of ambient light"
  - "darkening of the ambient shading contribution"

Ambient Occlusion

- "the crevices of the model are realistically darkened, and the exposed parts of the model realistically receive more light and are thus brighter"
- "the soft shadow generated by a sphere light of uniform intensity surrounding the scene"
**Ambient Occlusion**

Evenly lit from all directions

Global Illumination


**Ambient Occlusion**

- "the integral of the occlusion contributed from inside a hemisphere of a given radius $R$, centered at the current surface point $P$ and oriented towards the normal $n$ at $P$"

Figure 2. Hemisphere $\Omega$ around a surface point $P$.

**Object Space Ambient Occlusion**

- AO does not depend on light direction
- Precompute AO for static objects using *ray casting*
  - How many rays?
  - How far do they go?
  - Local objects? Or all objects?

• Cosine weight rays
  * or use *importance sampling*: cosine distribute number of rays

Object Space Ambient Occlusion

- Depends on scene complexity
- Stored in textures or vertices
- How can we
  - Support dynamic scenes
  - Be independent of scene complexity

Screen Space Ambient Occlusion

- Apply AO as a post processing effect using a combination of depth, normal, and position buffers
- Not physically correct but plausible
- Visual quality depends on
  - Screen resolution
  - Number of buffers
  - Number of samples

Depth Buffer

Normal Buffer
View Space Eye Position Buffer

Screen Space Ambient Occlusion

Screen Space Ambient Occlusion

Screen Space Ambient Occlusion

- Blur depth buffer
- Subtract it from original depth buffer
- Scale and clamp image, then subtract from original
- Superficially resembles AO but fast

Image from Mike Pan, http://mikepan.com