Computer Graphics

spring, 2013

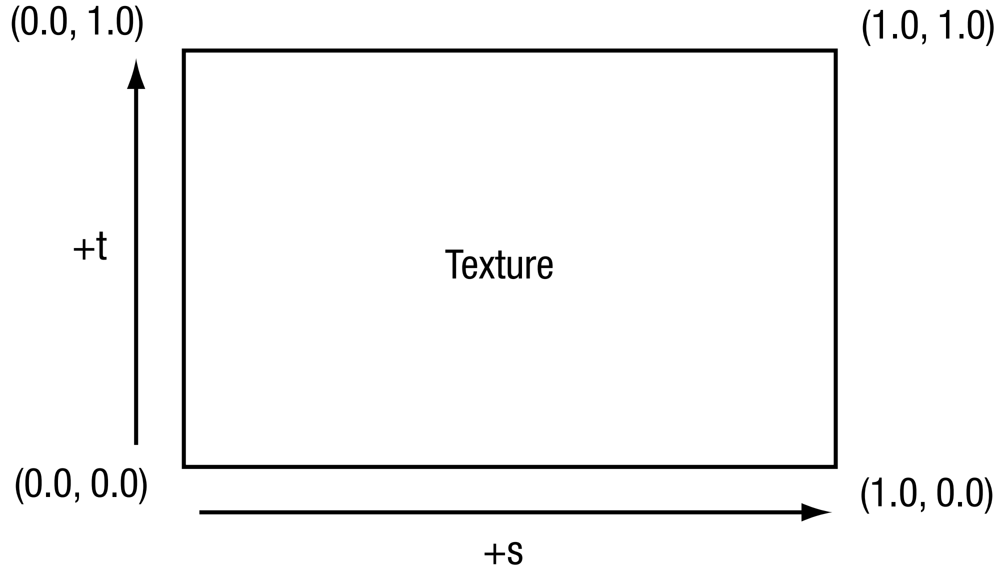
Chapter 9
Texturing

Topics

- Texturing basics.
- Loading textures and mipmapping.
- Texture filtering and wrapping.
- Using textures in the fragment shader.
- Texture subimage specification.
- Copying texture data from the color buffer.
- Optional texturing extensions.

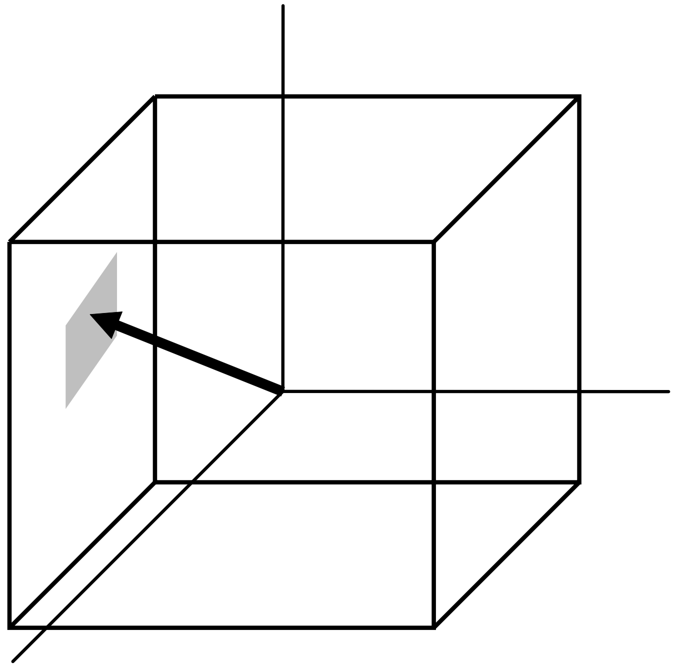
2D Textures

- 2D array composed of texels
- Formats -- RGB, RGBA, L, LA, Alpha
- Texture image is indexed by a per-vertex texcoord (s,t)
- Texcoord is normalized to [0,1]



Cubemap Textures

- For environment mapping
- Indexed by a texcoord (s,t,r) (vector)
- Usually reflection vector is used



Initialization

- glGenTextures - creates a texture obj
- glBindTexture - binds a texture obj
- glTexImage2D - loads texture data
  - No border supported
  - format should be the same with internalFormat
  - Data should be aligned as specified by glPixelStorei

```
// Texture object handle
GLuint textureId;

// 2 x 2 Image, 3 bytes per pixel(R, G, B)
GLubyte pixels[4 * 3] =
{
   255,   0,   0, // Red
     0, 255,   0, // Green
     0,   0, 255, // Blue
   255, 255,   0  // Yellow
};

// Use tightly packed data
glPixelStorei(GL_UNPACK_ALIGNMENT, 1);

// Generate a texture object
glGenTextures(1, &textureId);

// Bind the texture object
glBindTexture(GL_TEXTURE_2D, textureId);

// Load the texture
glTexImage2D(GL_TEXTURE_2D, 0, GL_RGB, 2, 2, 0, GL_RGB,
             GL_UNSIGNED_BYTE, pixels);

// Set the filtering mode
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_NEAREST);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_NEAREST);
```

Mipmap

- Collection of hierarchical tex images
- Increases rendering speed
- Reduces aliasing artifacts
- Can be generated from the original image using box filtering (See source codes)
- Can be generated automatically by glGenerateMipmap -- Useful for render-to-texture technique

Texture Filtering

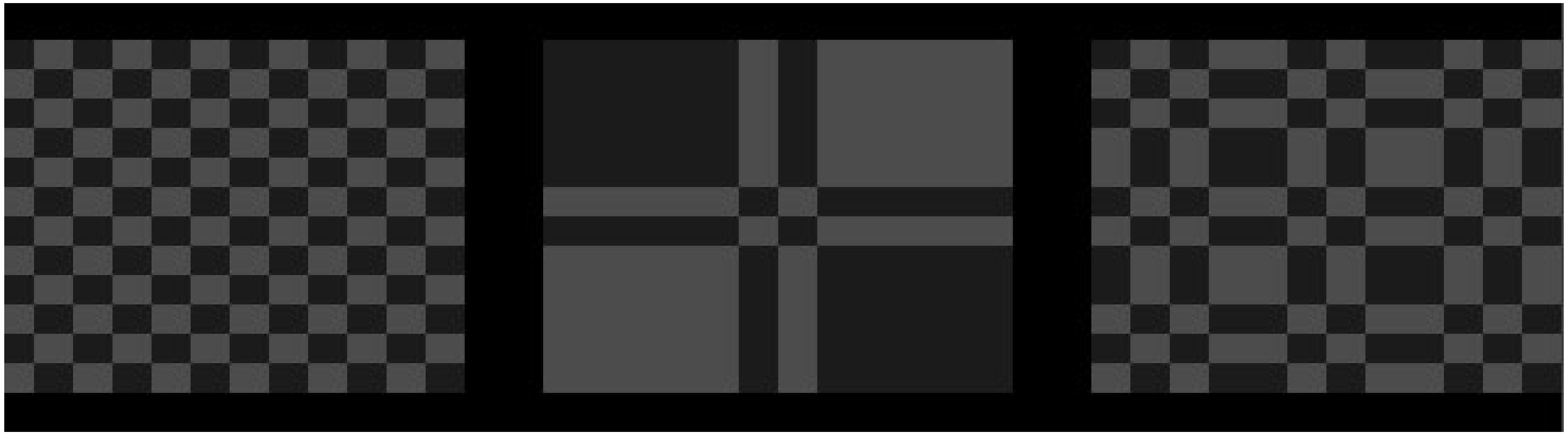
- Minification
  - Polygon projected to small area of texture --> sampling mode (interpolation)
  - Mipmap used
- Magnification
  - Polygon projected to large area of texture --> averaging
  - Mipmap not used --> largest image used
- Set by glTexParameter

Performance Issues

- Minification --> using a mipmap filtering results in high cache utilization
- Higher filtering mode --> high performance cost (Some HW supports HW filtering)

TexCoord Wrapping

- How to map for texcoords outside [0,1]
- GL_REPEAT, GL_CLAMP_TO_EDGE, GL_MIRRORED_REPEAT



Using Texture in Shaders

```
GLbyte vShaderStr[] =
   "attribute vec4 a_position;   \n"
   "attribute vec2 a_texCoord;   \n"
   "varying vec2 v_texCoord;     \n"
   "void main()                  \n"
   "{                            \n"
   "   gl_Position = a_position; \n"
   "   v_texCoord = a_texCoord;  \n"
   "}                            \n";

GLbyte fShaderStr[] =
   "precision mediump float;                           \n"
   "varying vec2 v_texCoord;                           \n"
   "uniform sampler2D s_texture;                       \n"
   "void main()                                        \n"
   "{                                                  \n"
   "  gl_FragColor = texture2D(s_texture, v_texCoord); \n"
   "}                                                  \n";
```

Using Texture in Shaders

- Sampler is not the texture id, but the texture unit to which the texture is bound.
- Texture unit is set by glActiveTexture
- Texture value is fetched in the shader by the built-in texture2D/textureCube (currently texture)
- Six cube images need to be loaded for cubemap

Compressed Textures

- Why?
  - To reduce memory footprint
  - To save memory bandwidth when fetching textures in a shader
  - To reduce app size
- Optional
- No format defined --> HW-dependent
- glCompressedTexImage2D

Texture Subimage Specification

- glTexSubImage2D
- glCompressedTexSubImage2D

Copying Tex Data from the Color Buffer

- To use rendered image as a texture
- RTT (Render-to-texture) using FBO is faster
- Copying from back buffer/pbuffer/FBO
- glCopyTexImage2D, glCopyTexSubImage2D
- Copying to fewer channel is allowed, but not to more channel

Optional Extensions

- 3D textures
- floating-point textures
- Ericsson texture compression (ETC)
- non-power-of-2 (npot) textures -- restrictions on wrapping & filtering