Texture Management

Michael I. Gold

NVIDIA Corporation

Overview

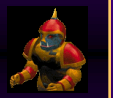
- Texture Objects
- Texture Internal Formats
- The EXT_packed_pixels extension
- Dynamic Texture Updates

glTexImage2D

Overview

- Texture image specification

```
glTexImage2D(
        target, lod, components,
        width, height, border,
        type, format, data);
```



Internal Formats

OpenGL 1.0 - components

- 1, 2, 3, 4

OpenGL 1.1 - internalformat

- ALPHA, LUMINANCE, INTENSITY, LUMINANCE_ALPHA, RGB, RGBA
- Two new single-component formats
  - ALPHA - alpha channel only
  - INTENSITY - RGBA use same value

Sized Formats

- Driver chooses default storage format
  - " What were they thinking?"
- You can give the driver a hint
- Driver uses closest supported format
  - RGB5 vs. RGB8
  - RGBA4 vs. RGB5_A1 vs. RGBA8
  - see TexImage2D reference for complete list
- Greater control over performance/quality tradeoffs

glGetTexLevelParameter

What format did I get?

- Component sizes may be queried

```
glGetTexLevelParameteriv(
      GL_TEXTURE_2D, lod,
      GL_RED_SIZE, &redSize);
```

EXT_packed_pixels

Faster texture downloads

- Textures are often stored in 16-bit formats
- OpenGL 1.1 textures must be expanded
- Driver often repacks textures internally
- EXT_packed_pixels allows packed data to be sent to the driver
- Supported in SGI ICD kit

EXT_packed_pixels (cont'd)

New data types

- GL_UNSIGNED_SHORT_4_4_4_4_EXT
- GL_UNSIGNED_SHORT_5_5_5_1_EXT
- see extension spec for complete set
- Use with corresponding internal format for best results

EXT_packed_pixels (cont'd)

Example - 5551 packed data

```
glTexImage2D(GL_TEXTURE_2D, lod,
  GL_RGB5_A1, w, h, 0,
  GL_UNSIGNED_SHORT_5_5_5_1_EXT,
  GL_RGBA, data);
```

Texture Objects - Old Way

OpenGL 1.0 - Display Lists

- difficult to optimize

```
GLuint tex = glGenLists(1);
glNewList(tex, GL_COMPILE);
glTexImage2D(GL_TEXTURE_2D, …);
glEndList();

glCallList(tex);
```

Texture Objects - New Way

OpenGL 1.1 - Texture Objects

- more elegant
- higher performance

```
GLuint tex;
glGenTextures(1, &tex);
glBindTexture(GL_TEXTURE_2D, tex);
glTexImage2D(GL_TEXTURE_2D, …);

glBindTexture(GL_TEXTURE_2D, tex);
```

Dynamic Texture Updates

- Divide static and dynamic textures
- Let OpenGL cache static textures
- Only update dynamic textures
- Use glTexSubImage if possible
- Re-use same “shape” textures if possible
  - Sort textures by size

Dynamic Texture Updates

- Don’t change one texture multiple times per frame
- Ping-pong between two “dynamic” textures, alternating frames
- Utilize texture matrix to avoid downloads if possible
- Minimize state changes - changing textures is expensive