Computer Graphics

spring, 2013

Chapter 4
Shaders and Programs

What to Discuss…

- What are shaders & programs?
- How to create & compile a shader?
- How to create & link a program?
- How to get & set uniform variables?
- How to get & set (vertex) attributes?
- Shader compiler & shader binaries

Shaders & Programs

- Shader object
  - ≈ C source code (*.c)
  - Compiling a shader ≈ a.c --> a.obj
- Program object
  - ≈ Executable (*.exe)
  - Linking a program ≈ v.o+f.o --> a.exe
- Compilation & linking are done by the OpenGL ES driver

Creating & Compiling a Shader

- glCreateShader -- creates a shader
- glDeleteShader -- deletes a shader
  - What if the shader is attached to a program object?
- glShaderSource -- provides the shader source (the source strings are copied to the shader object)
- glCompileShader -- compiles the shader (compiler is optional --> offline compilation required)
- glGetShaderiv -- gathers shader information (including compile status)
- glGetShaderInfoLog -- retrieves the info log

Creating & Linking a Program

- glCreateProgram -- creates a program object
- glDeleteProgram -- deletes a program object
- glAttachShader -- attaches a shader object (one vert shader + one frag shader)
- glDetachShader -- detaches a shader object
- glLinkProgram -- links a program object
- glGetProgramiv -- gathers program info including link status
- glGetProgramInfoLog -- retrieves the info log
- glValidateProgram -- validates the program (Will it execute with the current state?) --> for debugging only
- glUseProgram -- uses the program for rendering

Linker

- What does the linker check while linking?
  - Are all the varying variables consumed by the frag shader written by the vert shader?
  - Do all the uniform variables declared in both shader have matching types?
  - Does the final program fit within the limit of the implementation? (# of attributes, uniforms, varyings, instructions, etc.)
  - ...And more

Uniforms

- Read-only constants for shaders
- One set of uniforms for a program
- The linker assign uniform locations to each of the active uniforms during linking
- The list of active uniforms can be queried by glGetProgramiv
- glGetActiveUniform -- details on a uniform
- glGetUniformLocation
- glUniform* -- loads a uniform --> acts on the currently bound program object

Shader Binaries

- Compiler is not mandatory for any OpenGL ES 2.0 implementations --> Can be checked by glBooleannv with GL_SHADER_COMPILER
- The vendor needs to provide offline tools to build executable on the CPU
- A shader binary can be loaded by glShaderBinary
- No standard binary format --> less portability --> Can be queried by glGetIntegerv with GL_SHADER_BINARY_FORMATS
- glReleaseShaderCompiler