Thoughts and Experiences with the OpenGL Software Rasterizer

Because using *real* Graphics Hardware is BAD for teaching

Matthias Hopf
Professor for Applied Computer Science

NIT - Nuernberg Institute of Technology
Department of Electrical Engineering,
Precision Engineering, Information Technology
Obsolescence
f (plural obsolescences)

• (uncountable) The state of being obsolete — no longer in use; gone into disuse; disused or neglected.
• (countable) The process of becoming obsolete, outmoded or out of date.
Obsolescence in OpenGL

- OpenGL 3.1 (+ GLSL 1.40)
  - Current available max version in Mesa
  - Spec finished for more than 4 years now
  - Current version: (almost) 4.4...
  - Oldest(!) OpenGL quick reference docs: OpenGL 3.2
  - Oldest(!) GLSL quick reference docs: GLSL 3.30

- Differences in GLSL hurting more than in OpenGL
  - Most of functionality up to 4.0 already present
  - Attribute location binding

- Supported version also driver dependent...
Graphics Drivers? Nah...

• In real Life?
  Yes, everybody needs them

• For teaching?
  • Version differences between graphics stack vendors
  • In Mesa: even within one "vendor" for different drivers
  • Behavioral differences between different graphics hw vendors
  • Behavioral differences even between different graphics cards from the same vendor
  • Different bugs...

• Better use the Software Rasterizer...
Speed? Not an issue

- In teaching, the most complex Geometry usually is:

- Software rasterization is *almost always* fast enough
**Speed: Amazing...**

- Early times: flat shaded triangles was too slow

- Beginning of this century: flat shaded triangles in software faster than on most Gfx hardware

- Smooth shaded triangles reasonably fast for teaching

- Recently: textured triangles reasonably fast, even w/ interpolation

- Now: even Shaders reasonably fast, thanks LLVM!
Feature-wise: Only Recently an Option

• Before Mesa 9.2:
  • Hardware: GL 3.0 GLSL 1.30
  • Software: GL 2.1 GLSL 1.20

• Mesa 9.2:
  • GLSL version supported in HW and SW: 1.30

• Good idea to use MESA_GL_VERSION_OVERRIDE?!?
  • Which parts are just not announced, which are actually absent?
  • Feature matrices: current?
    • docs/GL3.txt
    • http://dri.freedesktop.org/wiki/MissingFunctionality/
OpenGL 4.4 not required soon-ish

But OpenGL 3.3 (rather: GLSL 3.30) is:

- `layout (location = 3) in vec3 normal;`
  so much nicer and semantically more reasonable than
  `in vec3 normal; [...]`
  `glBindAttribLocation (program, 3, "normal");`
- Documentation on opengl.org

- Already pretty much implemented (at least layout stuff)

- Good idea to use
  `MESA_GLSL_VERSION_OVERRIDE`?!?
  - Probably better wait for 10.0...
OpenGL x.y with x.y > something

• Goal: create newest possible context within a certain range (e.g. at least 2.1, at most 4.0)
  • Typically wanted (decide about render paths afterwards)
  • Annoyingly much code required to do that

• Any best practice about that?
Results

• Ilvm software driver is in a reasonable state to use for teaching shaders

• Support for GLSL 3.30 out-of-the-box would be awesome for ease of attribute binding

• Really required to not expose GLSL levels that aren't typically available in the announced OpenGL level?
  • E.g. Announce OpenGL 3.0 but GLSL 3.30...
  • It's only about syntactic sugar...