OpenGL 2.0 Update

Cass Everitt
cass@nvidia.com
Overview

- What’s probably in…
- What’s probably *not* in…
- review of functionality

*Caveat: The ARB has not reached a final decision on the exact 2.0 feature set, so things could change.*
What’s probably in…

• shading lanugage
• floating point pipeline
• pixel buffer object (PBO)
• sprites
• texture dimensions
• accum enhancements
• multiple draw buffers
• two-sided stencil
• blend func/equation separate
• extension query
What’s probably *not* in…

- uber buffers
- texture mirror clamp
- assembly vertex programs
- assembly fragment programs
Shading Language

- Bill Licea-Kane covered these details in his presentation.
Floating Point Pipeline

- Allow general-purpose rendering to and texturing from fp buffers and textures
- Incorporates several extensions:
  - ARB_pixel_format_float
  - ARB_texture_float
  - ARB_half_float_pixel
  - ARB_color_clamp_control
ARB_pixel_format_float

- Allow specification of float components for PFD
- unclamped clear value
ARB_texture_float

- floating point internal formats
  - RGBA32F
  - RGB32F
  - ALPHA32F
  - INTENSITY32F
  - LUMINANCE32F
  - LUMINANCE_ALPHA32F
  - RGBA16F
  - RGB16F
  - ALPHA16F
  - INTENSITY16F
  - LUMINANCE16F
  - LUMINANCE_ALPHA16F

- Also supports queries to determine component type
ARB_half_float_pixel

- “external format” for fp16 pixel data from the CPU
- s1e5m10
  - 1 sign bit
  - 5 exponent bits (bias of 15)
  - 10 mantissa bits
  - special numbers undefined
  - denorms defined
ARB_color_clamp_control

- for fp rendering, clamps usually unwanted
- but OpenGL specifies clamping due to its fixed-point heritage
- this extension allows the app to explicitly disable those clamps
Pixel Buffer Object (PBO)

• VBO “buffer objects” are just arrays of bytes managed by the driver
• PBO uses same API, but has binding points for
  - PIXEL_PACK_BUFFER
    • glReadPixels(), glGetTexImage(), etc.
  - PIXEL_UNPACK_BUFFER
    • glTexImage*, glDrawPixels(), etc.
• **Caveat**: Currently only an EXT extension!
Sprites

• Points with varying texture coordinates
• Nothing especially new here, though the question of t coordinate still being discussed
  – most hardware supports upper-left origin
  – import from Direct3D
Texture Dimensions

- (EXT,NV)_texture_rectangle
  - no power-of-two constraints
  - but, no mipmapping, no repeat, no borders
  - coordinates are not normalized
    - (0..w, 0..h)
  - lots of hardware support
Texture Dimensions

• ARB_texture_non_power_of_two
  – just conventional texture targets without the requirement that dimensions be power-of-two
  – mip level dimensions use “floor” convention
    • $w_i = \text{floor}( w_0 / 2^i )$
  – some concern about support
    • may be limited to 2D targets in OpenGL 2.0
Accum Enhancements

• Modern consumer hardware finally accelerates accum! 😊
• Some additional functionality
  – SUN_slice_accum
    • dst alpha lerp into accumulation buffer
  – ARB_accum_composite
ARB_accum_composite

- Augments the accum operations
  - OVER
  - UNDER
  - PREMULT_OVER
  - PREMULT_UNDER

- analogous to glBlendFuncSeparate for
  accum
Multiple Draw Buffers

• From ATI_drawBuffers

• In the C code:
  GLenum buffers[] = { GL_AUX0, GL_AUX1 };  
glDrawBuffers( 2, buffers );

• In the shader:
  OP result.color[0], src0, src1, ...;  
  OP result.color[1], src0, src1, ...;
Two-Sided Stencil

• Some unified form of the functionality in
  – EXT_stencil_two_side or
  – ATI_separate_stencil

• Helpful for single-pass stenciled shadow volume rendering.
  – front faces incr, back faces decr happens simultaneously
Blend Equation Separate

- **EXT_blend_func_separate**
  - separate srcFactor and dstFactor for RGB and A

- **EXT_blend_equation_separate**
  - separate blend equations for RGB and A
Extension Query

- **ARB_extension_query**
  - boolean IsExtensionSupportedARB(const GLubyte *name);
  - Nothing fancy, just a clearer way for apps to ask about extension support
  - Seems we always have apps that copy the extensions string into a fixed-size buffer!
Stuff that’s not in…

• uber buffers
  – Rob Mace will be talking about the status of the superbuffers work group

• texture mirror clamp
  – ARB not convinced that this is important enough to be “core”
Stuff that’s not in…(2)

• ARB_vertex_program & ARB_fragment_program
  – general lack of interest in making the ASM interfaces core
Parting Queries

• Where does the ARB need to focus efforts?
  – demos, whitepapers?
  – more rapid spec revs?
  – developer conference?

• How do ISVs provide feedback?
  – Is something less formal than the participant undertaking desirable?
Questions?

• cass@nvidia.com