Introduction to the OpenGL Shading Language (GLSL)

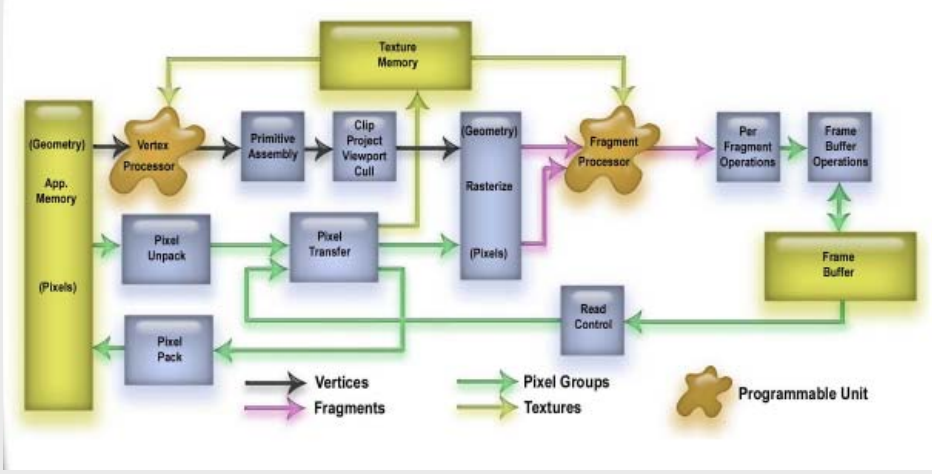
Mike Bailey

Oregon State University

Fundamental Differences Between RenderMan Shaders and OpenGL Shaders

Topic	RenderMan	GLSL
Goals	1. Image quality, 2. Speed	1. Speed, 2. Image quality
Shader Types	Surface, Displacement (+3 others)	Vertex, Fragment, Geometry
Surface Preprocessing	Microfacets	None
Recompute Normals	CalculateNormal	None
Get Rid of Pixels	Oi = 0.;	discard;
Surface/Fragment shader sets	R, G, B, αr, αg, αb	R, G, B, A, Z
Shader Variables	Uniform, Varying	Attribute, Uniform, Varying
Coordinate Systems	Shader, World, Object	Model (=OC), Eye (≈WC)
Noise	Built-in	Eventually built-in, Texture for now
Compile Shaders	Must do yourself	Driver does it for you
Compiler messages	Cryptic	Cryptic

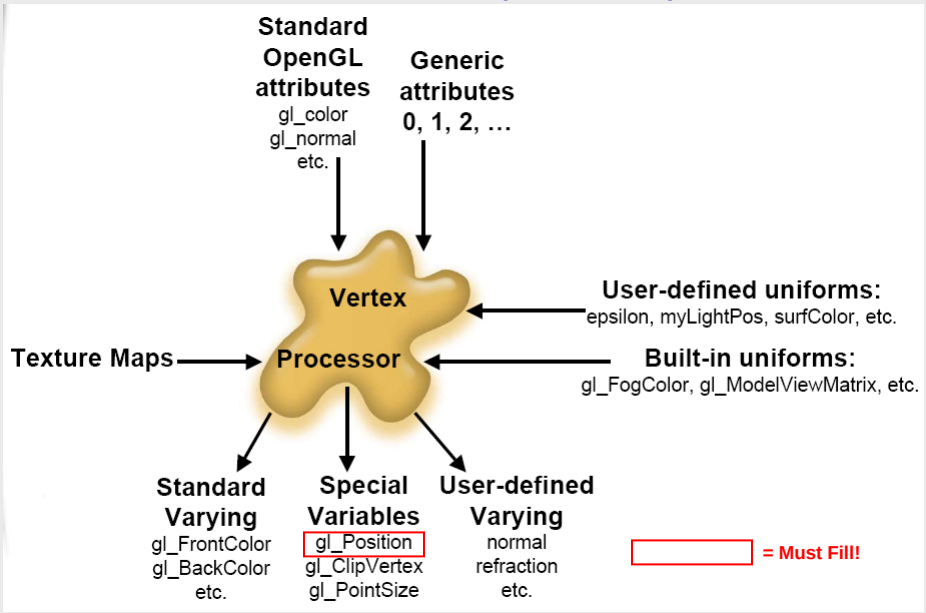
OpenGL Graphics Pipeline



Courtesy of Randi Rost

mjb – April 14, 2007

GLSL Vertex Shader Inputs and Outputs



Courtesy of Randi Rost

mjb – April 14, 2007

A GLSL Vertex Shader Replaces These Operations:

- Vertex transformations
- Normal transformations
- Normal normalization
- Handling of per-vertex lighting
- Handling of texture coordinates

A GLSL Vertex Shader Does Not Replace These Operations:

- Frustum clipping
- Homogeneous division
- Viewport mapping
- Backface culling
- Polygon mode
- Polygon offset

Built-in Vertex Shader Variables You Will Use *a Lot*:

vec4 gl_Vertex

vec3 gl_Normal

vec4 gl_Color

vec4 gl_MultiTexCoordi (i=0, 1, 2, …)

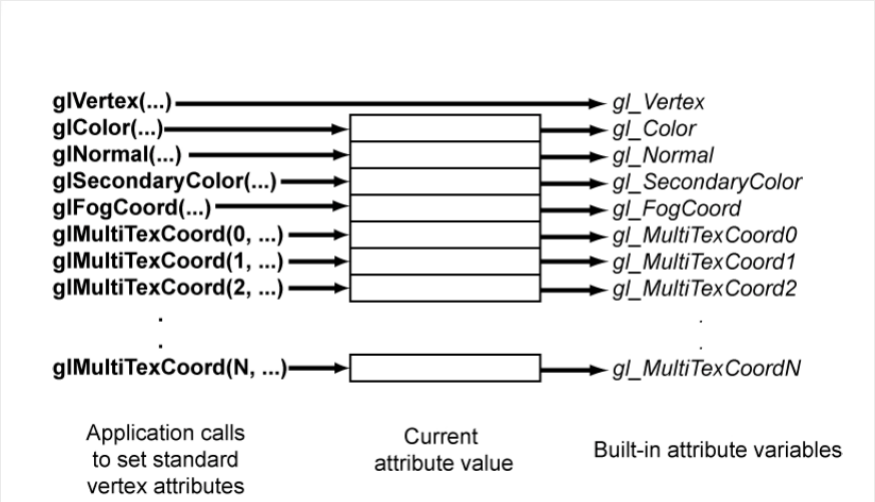
mat4 gl_ModelViewMatrix

mat4 gl_ProjectionMatrix

mat4 gl_ModelViewProjectionMatrix

mat4 gl_NormalMatrix

GLSL Vertex Shader Internal Names



Courtesy of Randi Rost

GLSL Shaders Are Like C With Extensions for Graphics:

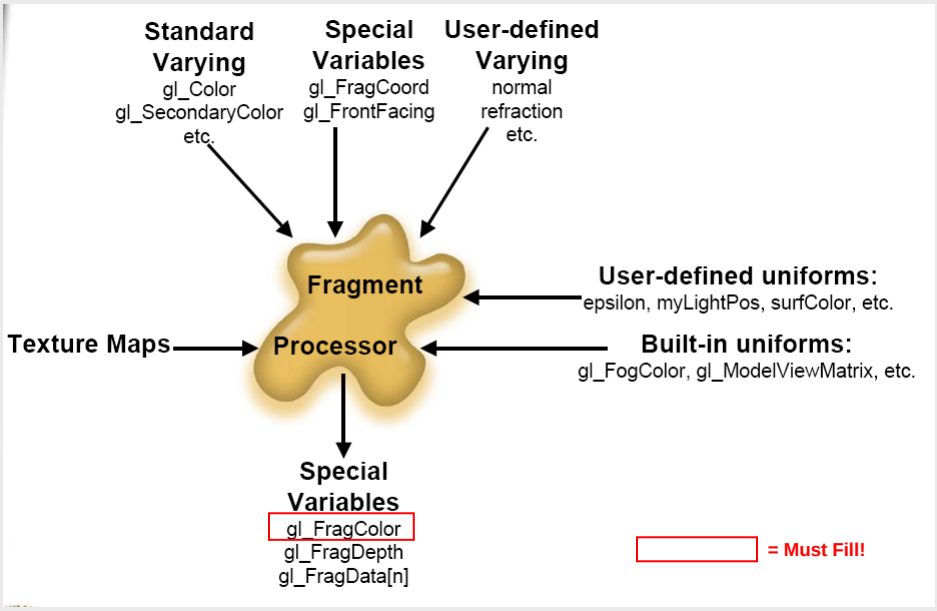
- Types include int, ivec2, ivec3, ivec4
- Types include float, vec2, vec3, vec4
- Types include mat2, mat3, mat4
- Types include bool, bvec2, bvec3, bvec4
- Types include sampler to access textures
- Vector components are accessed with [index], .rgba, .xyzw, and .stpq

- Vector components can be "swizzled" (c1.rgba = c2.abgr)
- *discard* operator used in frag shaders to discard fragments
- Type qualifiers: const, attribute, uniform, varying
- Procedure type qualifiers: in, out, inout

GLSL Shaders Are Missing Some C-isms:

- No type casts (use constructors instead)
- No automatic promotion
- No switch statement
- No pointers
- No strings
- No bitwise operators (this is changing)
- No enums
- Can only use 1-D arrays (no bounds checking)
- Array indices must be compiler-time constants
- No *file-based* pre-processor directives

GLSL Fragment Shader Inputs and Outputs



Courtesy of Randi Rost

A GLSL Fragment Shader Replaces These Operations:

- Color computation
- Texturing
- Color arithmetic
- Handling of per-pixel lighting
- Fog
- Blending
- Discarding fragments

A GLSL Fragment Shader Does Not Replace These Operations:

- Stencil test
- Z-buffer test
- Stippling

Built-in Fragment Shader Variables You Will Use *a Lot*:

vec4 gl_FragColor

float gl_FragDepth