OpenGL Notes [a]

Stu Pomerantz
smp@psc.edu
http://www.psc.edu/∼ smp
August 30, 2004

[a] All material is adapted from: OpenGL ARB, et. al, "The OpenGL Programming Guide", Third Ed., Reading: Addison-Wesley, 1999

What is OpenGL ?

- A software interface to graphics hardware.
- The interface consists of about 250 commands used to specify the objects and operations needed to produce interactive 3D graphics.
  - OpenGL *geometric primatives* include points, lines, & polygons. There is specific support for triangle and quadrilateral polys.
  - Quadric (defined by quadratic equation) and NURBS (spline) surface support.
  - Texture mapping support.
  - As of OpenGL 2.0: Programmable shader support.

What is OpenGL ?

- Hardware *and* Operating System independant.
  - No commands for performing windowing tasks.
  - No commands for obtaining input.
- A state machine
  - You put OpenGL into a state (or mode) and that state remains in effect until you change it.
  - State is encapusated in *contexts*. Each OpenGL window has its own, separate, state.
  - Think of each context as a `struct` with fields for each OpenGL state variable.

What is OpenGL ?

- An Open Standard
  - The OpenGL language is not controlled by a single company but rather steered by the Architecture Review Board (ARB).
- Extensible
  - OpenGL language has an extension mechansim defined in it.
  - Venders can expose non-standard OpenGL features this way.
  - Features migrate (typically) from non-standard extensions to ARB approved extensions to first class language features.

What is OpenGL ?

- Evolving
  - OpenGL changes slowly (the members of the ARB vote on changes).
  - Backwards compatibility is important. The current version of OpenGL is 2.0. Nevertheless, valid 1.0 programs will run on a 2.0 machine.
- **Doom 3** is written in OpenGL.

OpenGL Syntax

- All OpenGL are prefixed by the letters **gl** .
- Defined constants begin with **GL_** . Underscores separate words.
- OpenGL commands may be postfixed by one or more of:
  - A number {2,3,4} indicating the number of arguments.
  - A letter {i,f,d,ub} indicating the data type of the arguments.
  - A letter **v** indicating the argument is a vector (array).

OpenGL Syntax

- OpenGL defines its own types which correspond to C types:
  - GLbyte → signed char
  - GLint → int or long
  - GLfloat → float
  - The C++ compiler may generate a warning/error and need a cast.

OpenGL Syntax Examples

Example: Setting the current color using `glColor` .

- Colors may have 3 components, RGB or 4 components, RGBA. Think of A (or alpha) as opacity.
- Floating point - color component values range from 0 to 1
  - `glColor3f( 0.0, 0.5, 1.0 ) ;`
  - This is: 0 % Red, 50% Green, 100% Blue
  - `glColor4f( 0.0, 0.5, 1.0, 0.3 ) ;`
  - This is: 0 % Red, 50% Green, 100% Blue, 30% Opacity
  - `GLfloat color[4] = { 0.0, 0.5, 1.0, 0.3 } ;`
    `glColor4fv( color ) ;`
  - Again, 0 % Red, 50% Green, 100% Blue, 30% Opacity

OpenGL Syntax Examples

- Unsigned byte - color component values range from 0 to 255 (same as C's unsigned char).
  - `glColor3ub( 0, 127, 255 ) ;`
  - This is: 0 % Red, 50% Green, 100% Blue
  - `glColor4f( 0, 127, 255, 76 ) ;`
  - This is: 0 % Red, 50% Green, 100% Blue, 30% Opacity
  - ```
    GLubyte color[4] = { 0, 127, 255, 76 } ;
    glColor4ubv( color ) ;
    ```
  - Again, 0 % Red, 50% Green, 100% Blue, 30% Opacity
    ```

OpenGL Libraries

- GLU – The OpenGL Utility Library.
  - Utility functions built from OpenGL functions.
  - Most OpenGL programs use some GLU calls.
- OpenGL Libraries for specific Windowing Systems
  - GLX – OpenGL support in the X Window System.
  - WGL – OpenGL support in Microsft Windows.
  - AGL – OpenGL support in Apple Macintosh.

The GLUT Library – The OpenGL Utility Toolkit

- NOT a part of OpenGL
- Library written by Mark Kilgard formerly of SGI, now at nVidia.
- Abstracts away the details of GLX, WGL, & AGL
  - limited access to some native OpenGL OS functionality.
- Abstracts away the details of opening a window and an OpenGL context across operating systems.
- Abstracts away the details of keyboard & mouse input.
- Widely used in demonstration programs and literature.

GLUT – Hello World

Opening a window:

```
int main(int argc, char **argv) {
 // step 1, initialize glut and pass glut argc & argv
 glutInit(&argc, argv) ;
 // set parameters of the opengl window we're about to open
 glutInitDisplayMode(GLUT_RGBA|GLUT_DEPTH|GLUT_DOUBLE) ;
 // set the size of the window
 glutInitWindowSize(800,600) ;
 // open the window
 glutCreateWindow("Hello World Glut") ;
 // The OpenGL context is now available
 // tell glut the name of the function inside which will do the drawing
 glutDisplayFunc( display ) ;
 // Enter the glut event loop, this function never returns
 glutMainLoop() ;
 return(0) ;
}
```

GLUT – Hello World

After opening a window:

- Initialize OpenGL *after*
  `glutCreateWindow("Hello World Glut") ;`
- *Do not* assume that OpenGL commands can be called anywhere.
  - Most must be called inside your `display` function to have the desired effect.
  - There are some exceptions to this rule.
- The display function is `void (*func)(void)`. For example
  `void display(void) { } ;`

GLUT – Nuts & Bolts

Input:

- Request keyboard input using: `glutKeyboardFunc(`*keyboard*`)`. For example:

  ```
  keyboard( unsigned char key, int x, int y ) ;
  ```

- Request mouse input using: `glutMouseFunc(`*mouse*`)`. For example:

  ```
  mouse(int button, int state,
                  int mousex, int mousey) ;
  ```

Error Checking:

- For debugging, call: `glutReportErrors() ;` once per frame.

GLUT – Compilation

- Compiling
  - `#include <GL/glut.h>` automatically includes `<GL/gl.h>`
  - If you use glu functions you'll need `#include <GL/glu.h>`
  - Windows: Choose **Win32 Console Application**
- Linking
  - UNIX/Linux: `gcc asn00.c -lglut -lGLU -lGL -lXmu -lXi -lX11 -lm -L/usr/X11R6/lib`
  - Windows: Add `glut32.lib glu32.lib opengl32.lib`
  - MacOS X: Add `-framework GLUT -framework OpenGL -framework Foundation -framework CoreServices -framework ApplicationServices -framework Carbon`

Resources

- "The Red Book" a.k.a The OpenGL Programming Guide
- "The Blue Book" a.k.a The OpenGL Reference Manual
- `http://www.opengl.org`