An OpenGL Primer

Dr Nicolas Holzschuch
University of Cape Town

e-mail: holzschu@cs.uct.ac.za
What is OpenGL?

• Graphics Library
  – for Silicon Graphics
  – available on all major graphics computers

• Provides graphics primitives
  – takes care of all basic tasks (line, polygon, perspective, Gouraud shading…)

• Link with your programs as a normal C library
OpenGL availability

• Native:
  – Silicon Graphics and Windows 32

• Freeware implementation:
  – X11, Windows 16, Amiga, NeXT and Mac

• Interests:
  – source code portability
  – the only graphics library that takes all the power of the Silicon Graphics (Z-buffer, Gouraud shading...)

How to use it?

• As a normal C/C++ library:
  – include the header files:
    ```
    #include <GL/gl.h>
    #include <GL/glu.h>
    #include <aux.h>
    ```
  – link the libraries:
    ```
    -laux -ltk -lGLU -lGL -lXext -lX11 -lm
    ```
• Use a **Makefile**!!!
Use a Makefile!!!

• Sample Makefile:

LDFLAGS = -laux -ltk -lGLU -lGL -lxext -lx11 -lm
CFLAGS = -g -I$(HOME)/include
OBJE = main.o input.o output.o

all: myprog
myprog: $(OBJE)
    $(CC) -o myprog $(OBJE) $(LDFLAGS)

clean:
    rm -f *..o core

• Use of CFLAGS is automatic for object creation
Opening a window

• OpenGL independant from the window system

• Auxiliary library for simple window opening
  – takes care of the interface with the window-system
  – interaction, display, events...
  – source code portability
  – only one window

• Other libraries for more complex tasks (GLUT)
Opening a window (2)

```c
int main(int argc, char** argv)
{
    auxInitDisplayMode (AUX_SINGLE | AUX_RGBA);
    auxInitPosition (0, 0, 300, 300);
    auxInitWindow (argv[0]);
    myinit ();
    auxMainLoop(display);
}
```
A few simple tasks

• Clearing the background:
  
  ```c
  glClearColor (0.0, 0.0, 0.0, 0.0);
  glClear (GL_COLOR_BUFFER_BIT);
  ```

• Changing the color:
  
  ```c
  glColor3f (1.0, 0.0, 0.0);
  ```

• Changing line width:
  
  ```c
  glLineWidth(3.0);
  ```
Drawing graphics primitives

• Specify \texttt{type} first, then give vertices:

\begin{verbatim}
  glBegin(GL_LINE_STRIP);
  glVertex2i(20,280);
  glVertex2i(20,20);
  glVertex2i(100,20);
  glVertex2i(100,280);
  glEnd();
\end{verbatim}
Basic Graphics Primitives

GL_POINTS

GL_LINES

GL_LINE_STRIP

GL_LINE_LOOP

GL_POLYGON

GL_QUADS

GL_QUAD_STRIP

GL_TRIANGLES

GL_TRIANGLE_STRIP

GL_TRIANGLE_FAN
User interface

- **Callback scheme:** `auxMainLoop(display);`
- **Mouse events:**
  ```
  auxMouseFunc(AUX_LEFTBUTTON, AUX_MOUSEDOWN, leftDown);
  ```
- **Callback function:**
  ```
  void leftDown(AUX_EVENTREC *event)
  {
      int x, y;
      x = event->data[AUX_MOUSEX];
      y = event->data[AUX_MOUSEY];
  }
  ```
Keyboard input

• Keyboard events:
  auxKeyFunc(AUX_KEY, myKeyboardKey);

• Callback function:
  void myKeyboardKey(void)
  {
    ...
  }
Resizing the window

• Resize events:

    auxReshapeFunc(myResize);

• Callback function:

    void myResize(int sizeX, int sizeY)
    {
        Xmax = sizeX;
        Ymax = sizeY;
    }
User Interface

- Idle function (when there are no events):
  \[ \text{auxIdleFunc(advanceTheMonsters);} \]

- No need to explicitly call a redraw:
  - \text{auxMainLoop} is called after each event

- User interface sufficient for beginning

- For better UI, use \text{GLUT} (several windows, menus...
Finding out more about OpenGL

• Read The Friendly Manual (RTFM):
  – man pages
  – insight on Silicon Graphics (hypertext version of the OpenGL manual)
  – **OpenGL WWW center**

• Use The Source, Luke (UTSL):
  – look inside aux.h for options to auxFunctions
MesaGL

- Freeware implementation of OpenGL
- Runs on about everything
- Change the LDFLAGS:
  -L$(MESA_LOCATION) -lMesaaux -lMesatk -lMesaGLU
  -lMesaGL -lXext -lX11 -lm

- Windows users:
  - Visual C++ 4.0 comes with OpenGL
  - Mesa does not work well with Borland C++
Conclusion

• Graphics Library
  – simple, easy to use
  – hardware independant
  – portable
  – includes all the basic graphics primitives
• The speed of the C language
• The power of the Silicon Graphics