OpenGL Acceleration for Linux

LinuxWorld Tutorial HL

Goals

- **Introduce OpenGL acceleration on Linux**
- **Give direction on using OpenGL on Linux**
- **Detail existing implementations**
  - How to use
  - How they work
- **Give an idea of where things are going**

Speakers

- **David Blythe**
  - Routefree, technical lead
  - Previously at SGI; OpenGL, InfiniteReality
- **Brad Grantham**
  - Systems engineer at VA Linux Systems
- **Brian Paul**
  - VA Linux Systems Professional Services
  - Author of Mesa

Schedule

1:00 - 1:15 Introduction
1:15 - 1:45 Using OpenGL Under Linux
1:45 - 2:30 OpenGL API For Linux
2:30 - 2:45 Break
2:45 - 3:00 More OpenGL API For Linux
3:00 - 3:20 Completeness, Conformance
3:20 - 3:45 Futures
3:45 - 4:00 Conclusion

OpenGL

- **The industry standard 3D API**
  - Scientific visualization
  - Simulation, Games
  - CAD
- **Multiplatform**
  - $40 3D accelerator at CompUSA
  - $100000 InfiniteReality systems
  - Windows, MacOS, BeOS, Linux

OpenGL

- **3D polygons with surface texture**
  - 3D vertices projected onto 2D screen
  - Emperically derived lighting effects
  - Hidden surface removal with depth buffer
  - Blending allows simulation of volume effects
- **2D image operations; read and draw**
- **Various pixel operations**
  - Stencil, blending, stipple, channel mask

OpenGL

- **Example drawing a red triangle**

```
glBegin(GL_TRIANGLES);
    glColor3f(1, 0, 0);
    glVertex3f(-1, 1, 0);
    glColor3f(1, 0, 0);
    glVertex3f(-1, -1, 0);
    glColor3f(1, 0, 0);
    glVertex3f(1, -1, 0);
glEnd();
```

OpenGL

- **glLightfv for setting light parameters**
- **glMaterialfv for changing material colors**
- **glTexImage2D for setting texture data**
- **glClear to clear the color and depth buffers**

GLU, OpenGL Utility Library

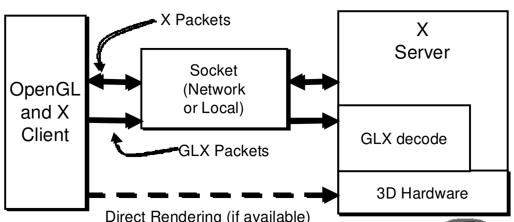
- **OpenGL kept very clean**
  - Sometimes leaves out obvious convenience
- **GLU handles some things not in OpenGL**
  - Polygon, quadric, NURBS tessellation
  - Projecting screen point back into space (gluUnproject)
  - Viewing matrix from viewpoint + target
  - Some other functions for convenience

OpenGL and the X Window System: GLX

- **OpenGL not tied to windowing API**
- **Another API glues OpenGL to X Window System: GLX**
- **Can**
  - Choose X Windows visual format with right attributes for OpenGL rendering
  - Bind OpenGL *context* to X Windows *Drawable*
  - Render into Pixmaps for off-screen rendering
    - (Pbuffers in GLX 1.3 are more flexible)

GLX

- **GLX Display**



GLX

- **Window Opening Sample Code, part 1**

```
#include <X11/xlib.h>
#include <GL/gl.h>
#include <GL/glx.h>

Display *dsp;
dsp = XOpenDisplay(":0.0");
```

GLX

- **Window Opening Sample Code, part 2**

```
int attributes[] {
  GLX_RGBA,
  GLX_DOUBLEBUFFER,
  GLX_DEPTH_SIZE = 16,
  GLX_RED_SIZE = 5,
  GLX_GREEN_SIZE = 5,
  GLX_BLUE_SIZE = 5
};
```

GLX

- **Window Opening Sample Code, part 3**

```
XVisualInfo *vi;
GLXContext cxt;
vi = glXChooseVisual(DefaultScreen(dsp),
  attributes);
cxt = glXCreateContext(dsp, vi, 0, GL_TRUE);
```

GLX

- **Window Opening Sample Code, part 4**

```
win = XCreateWindow(dsp, RootWindow(dsp, vi-
  >screen),
    0, 0, width, height,
    0, vi->depth, InputOutput, vi->visual,
    None, NULL);
XMapWindow(dsp, win);
glXMakeCurrent(dsp, win, cxt);
/* draw */
glXSwapBuffers(dsp, win);
```

GLX

- There's more to GLX
  - Synchronizing X and GL operations
  - Extensions to GLX
- Check out *Programming OpenGL for the X Window System* (by Mark Kilgard)
- GLX specification and man pages at www.opengl.org

Toolkits

- OpenGL encapsulates hardware well
  - No object system
  - No file or image loading
  - No simple geometric images
  - No window system or event handling
- Many toolkits exist to help with some or all of these areas

GLUT, the OpenGL Utility Toolkit

- Originally by Mark Kilgard
- Based on libaux, libtk in original OpenGL
- Simple geometric shapes
- Window opening and event callbacks
- Bitmapped text
- Pop-up menus

GLUT

- **Creating a window**

```
glutInit(&argc, argv);
glutInitWindowSize(512,512);
glutInitDisplayString("rgb double depth");
glutCreateWindow("window title");
```

GLUT

- **Example Button Callback**

```
glutMouseFunc(button); /* in main() or init() */

void button(int b, int state, int x, int y)
{
    if(button == 0) {
        if(state == 0)
           doLeftButtonDownAction();
        else
           doLeftButtonUpAction();
    }
}
```

GLUT

- **Let GLUT take control**

```
glutDisplayFunc(display);
glutMainLoop(); /* in main() or init() */

void display()
{
    glClear(GL_COLOR_BUFFER_BIT |
        GL_DEPTH_BUFFER_BIT);
    glutSolidSphere(1.0, 20, 20)
    glutSwapBuffers();
}
```

Simple DirectMedia Layer (SDL)

- Sam Lantinga's SDL; portable media services
- Opening framebuffer window, blitting
- Playing 8, 16-bit audio samples
- Threading model, thread safe functions
- Window system event handling
- Attempt to make system-independent media layer (like DirectX but portable)

SDL

- Opening a window

```
SDL_Init(SDL_INIT_VIDEO);

SDL_SetVideoMode(640, 480, 0, SDL_OPENGL);

SDL_WM_SetCaption("SDL Window");

/* Not checking errors returned, though */
```

SDL

- Checking events

```
while(!done) {
   SDL_Event event;
   while(SDL_PollEvent(&event)) {
      if((event.type == SDL_QUIT) ||
         (event.type == SDL_KEYDOWN &&
          event.key.keysym.sym == SDLK_ESCAPE))
         { done = 1; }
   }
}
SDL_Quit();
```

SDL

`Http://www.devolution.com/~slouken/SDL`

PLIB - Portable Game Library (by Steve Baker)

- PUI - Picoscopic User Interface, layered on GLUT
- SL - Sound Library with MOD loader/player
- SGL - Simple Geometry Library
- SSG - Simple Scene Graph Library
- JS - Joystick support (better than GLUT's)
- FNT - Fonts 'n Text Library

PLIB - Portable Game Library

- C++ based.
- Works on Linux, BSD, Irix, etc, MacOS and all Win32 variations.
- If developing a full-screen app or game PLIB can help you get up and running fast.

Scene graphs

- **3D scenes represented by directed graph representing attribute inheritance**
  - Geometry transformations
  - Maybe Appearance
- **Usually have other bits and pieces**
  - File loaders
  - Geometry utlities
  - Window and event handling

Open Inventor

- **Silicon Graphics**
- **Object-oriented C++ toolkit**
- **Easy to program**
- **Easy graphical interaction**
- **Powerful extension mechanism**

Open Inventor

- **Creating a scene**

```
root = new SoSeparator;
root->ref();
SoPerspectiveCamera *camera = new
  SoPerspectiveCamera;
root->addChild(camera);
root->addChild(new SoDirectionalLight);
```

Open Inventor

- Populating a scene

```
SoMaterial *material = new SoMaterial;
material->diffuseColor.setValue(1, 0, 0);
root->addChild(material);
root->addChild(new SoCone);
```

Open Inventor

- Using a window

```
Widget window = SoXt::init(argv[0]);
SoXtRenderArea renderArea = new
  SoXtRenderArea(window);
camera->viewAll(root, renderArea-
  >getViewportRegion());
renderArea->setSceneGraph(root);
renderArea->show()
SoXt::show(window);
SoXt::mainloop();
```

Open Inventor

- Template Graphics (www.tgs.com) sells Linux Inventor port
- COIN (www.coin3d.org) sells library compatible with Inventor
- Apprentice is open source Inventor clone

IRIS Performer

- **Silicon Graphics**
- **Object oriented, C++**
- **Geared toward simulation & performance**
  - Optimized rendering loops
  - Multiprocess cull-to-view, sort by material change
- **Scene graph builder library**
- **Render performance utilities**

Iris Performer

- **Creating a scene**

```
pfScene *scene = new pfScene;
pfGeoState *gstate = new pfGeoState;
gstate->setMode(PFSTATE_ENLIGHTING, PF_ON);
scene->setGState(gstate);
scene->addChild(new pfLightSource);
```

Iris Performer

- **Populating a scene**

```
pfString *str = new pfString;
pfFont *fnt = pfdLoadFont_type1("Times-Elfin",
  PFDFONT_EXTRUDED);
str->setfont(fnt);
str->setMode(PFSTR_JUSTIFY, PFSTR_MIDDLE);
str->setString("Hello World!\n");
pfText *text = new pfText;
text->addString(str);
scene->AddChild(text);
```

IRIS Performer

- Performer for Linux available from www.sgi.com

Other Toolkits

GLTT	ClanLib
CrystalSpace	FreeWRL
Maverik	MAM/VRS
VTK	GLOW
GLTT	bluevoid.com