

OpenGL API

Tom Kelliher, CS 320

Feb. 7, 2011

1 Administrivia

Announcements

Assignment

See assignment on web site.

Read over Project 1 handout.

From Last Time

Introduction to OpenGL lab.

Outline

1. Sierpinski gasket program.
2. Coordinate systems.
3. OpenGL API.

Coming Up

Project 1 and discussion.

2 Sierpinski Gasket

Let's take apart an OpenGL program.

The algorithm:

```
// Define the vertices of some 2-D convex geometric shape.

old = a random initial point within the shape;

for some number of points
    randomly choose one of the vertices of your shape;
    new = the point halfway between this vertex and old;
    plot new;
    old = new;
```

The program:

```
#include <GL/glut.h>

typedef struct { float x,y;} point;

/* A pentagon */
point vertices[5]={{50,0},{0,250},{250,500},{500,250},{450,0}};

int j;
point new, old={75,50}; /* A random point */

void clear(void)
{
glClear(GL_COLOR_BUFFER_BIT);
}

void display(void)
```

```

/* computes and plots a single new point */

{
long random();
int i;
j=random()%5; /* pick a vertex at random */

/* Compute point halfway between vertex and old point */

new.x = (old.x+vertices[j].x)/2;
new.y = (old.y+vertices[j].y)/2;

/* plot point */

glBegin(GL_POINTS);
glVertex2f(new.x, new.y);
glEnd();

/* replace old point by new */

old.x=new.x;
old.y=new.y;

glFlush();
}

void mouse(int btn, int state, int x, int y)
{
if(btn==GLUT_LEFT_BUTTON&state==GLUT_DOWN)  glutIdleFunc(display);
if(btn==GLUT_MIDDLE_BUTTON&state==GLUT_DOWN)  glutIdleFunc(NULL);
if(btn==GLUT_RIGHT_BUTTON&state==GLUT_DOWN)  exit();
}

int main(int argc, char** argv)
n{

glutInit(&argc,argv);
glutInitDisplayMode (GLUT_SINGLE | GLUT_RGB);
glutInitWindowSize(500,500);

```

```
glutInitWindowPosition(0,0);
glutCreateWindow("Sierpinski Gasket");

glutIdleFunc (display);
glutMouseFunc (mouse);
glClearColor(1.0, 1.0, 1.0, 0.0); /* white background */
glColor3f(1.0, 0.0, 0.0);
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
glOrtho(0.0, 500.0, 0.0, 500.0, -1.0, 1.0);
glMatrixMode(GL_MODELVIEW);
glutDisplayFunc(clear);

glutMainLoop();

}
```

3 Coordinate Systems

1. What units are we drawing in?
 - (a) Virtual.
 - (b) Where is the origin?
3. Window (raster) coordinates.
 - (a) Physical.
 - (b) Where is the origin?
4. Device independence via abstraction.

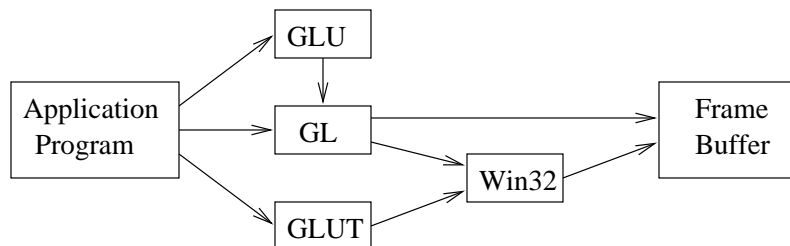
4 OpenGL API

Function classes within the OpenGL API:

1. Control: window system interactions.
2. Primitives: rendering.
3. Attribute: current color.
4. Viewing: camera settings.
5. Transformation: translation, rotation, scale.
6. Input: interaction.

Three-layered libraries: GL, GLU, GLUT.

Library organization:



5 OpenGL Primitive Types

1. Basic rendering code:

```
glBegin(primitiveType);  
    glVertex*(...);  
    // ...  
    glVertex*(...);  
glEnd();
```

2. 0-D primitives: `GL_POINTS`.

Color.

3. 1-D primitives: `GL_LINES` (pairs of points), `GL_LINE_STRIP` (continuous), `GL_LINE_LOOP` (unfilled poly).

Color. Width?

4. 2-D primitives: `GL_POLYGON`, `GL_TRIANGLES` (three-at-a-time) `GL_QUADS` (etc.), strips and fans for “intricate” surfaces.

Filled. Fill types: color, pattern.

Simple, convex polygons.

5. Text:

(a) Font, face, point size.

(b) Stroke text vs. raster text.

Stroke text: specified by vertices of the characters.

Raster text: bit maps.

Quality (or lack) of scaled text.

6. Curved objects: tessellate.