
CS380: Computer Graphics

Basic OpenGL Structure

Sung-Eui Yoon
(윤성익)

Course URL:
<http://sglab.kaist.ac.kr/~sungeui/CG>

KAIST



Class Objectives

- Understand the basic OpenGL program structure and how OpenGL supports different spaces

OpenGL

- **Graphics interface**
 - **Hardware-independent**
 - **Cross-platform graphics interface for 3D rendering and 3D hardware acceleration**
- **Two main characteristics**
 - **Small, but powerful set of low-level drawing operations**
 - **Does not have any functions to interact with any device and windowing system**
- **What are problems of OpenGL, then?**

Two Additional Libraries

- **GLU (GL utility)**
 - Provide more complex rendering methods
- **GLUT (GL utility toolkit)**
 - Provide platform-independent interface to the windowing system and input devices

GLUT

- **Advantages:**
 - **Portable: Windows, Cygwin, Linux, Mac-OS**
 - **Minimal-overhead (Hides away details of opening windows, etc.)**
 - **Appeals to C-hackers (console for printf()'s, etc)**
- **Disadvantages**
 - **Limited interaction**
 - **Global variables galore**

Getting GLUT

- Web site:

Windows:

www.xmission.com/~nate/glut.html

Others:

www.opengl.org/developers/documentation/glut.html

www.sourceforge.net/projects/uncpythontools

- Overview:

Appendix D of OpenGL Programming Guide

OpenGL Tools Available

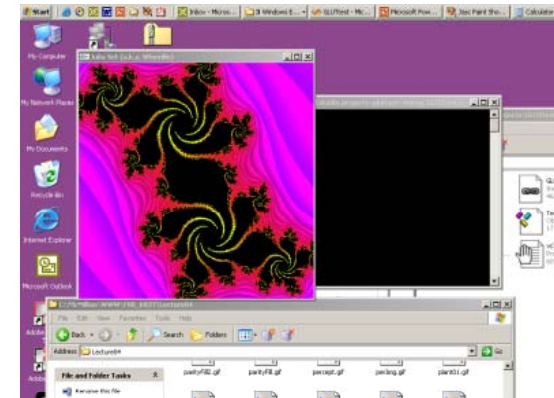
Typical OpenGL code to establish a window:

```
glutInitWindowSize(400,400);
```

```
glutInitWindowPosition(100,100);
```

Code to set up a viewport:

```
glViewport(0, 0, w, h);
```



To establish a world space coordinate system:

```
glOrtho2D(world.l, world.r, world.b, world.t);
```

Sample Codes of Visualization of a Fractal



Libraries, Header Files, etc

```
#pragma comment(lib,"opengl32.lib")
#pragma comment(lib,"glu32.lib")
#pragma comment(lib,"glut32.lib")

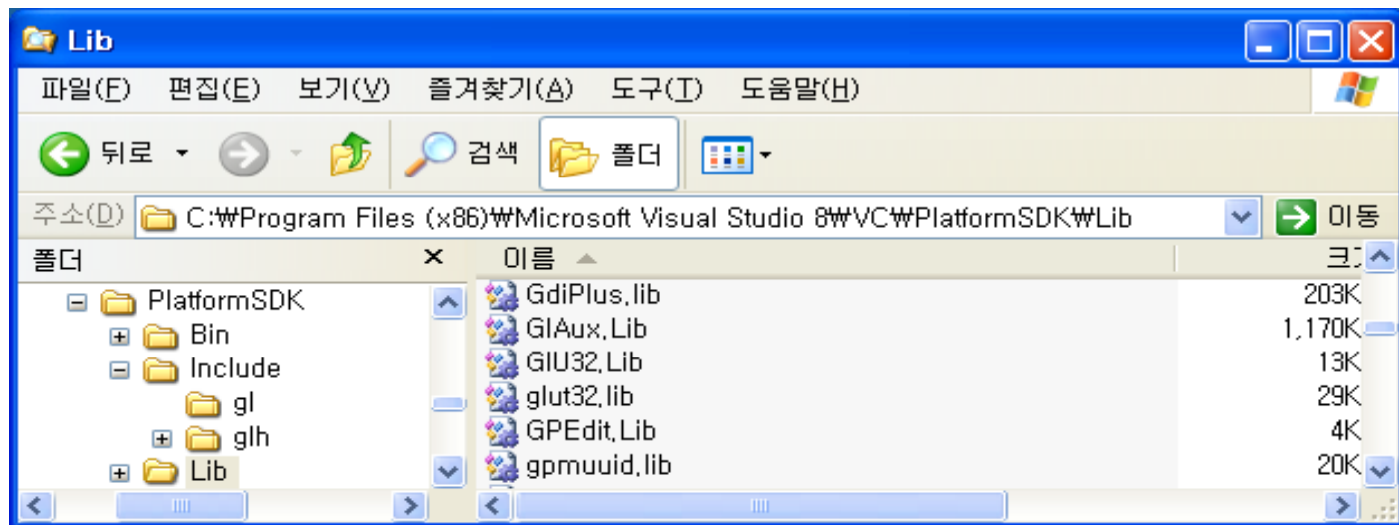
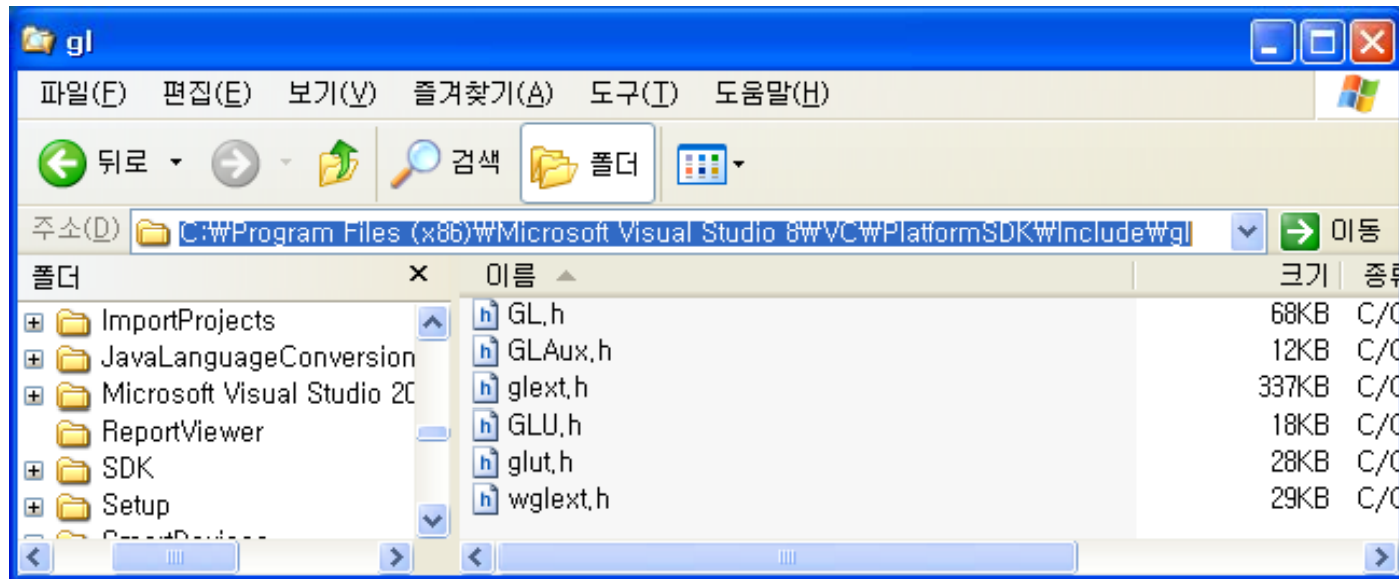
#include <GL/glut.h>
#include <GL/glu.h>
#include <math.h>

// glut callbacks
void display();
void onKeyPress(unsigned char k, int x, int y);
void onMouse( int button, int state, int x, int y);
void onReshape( int w, int h );
void idle();
```

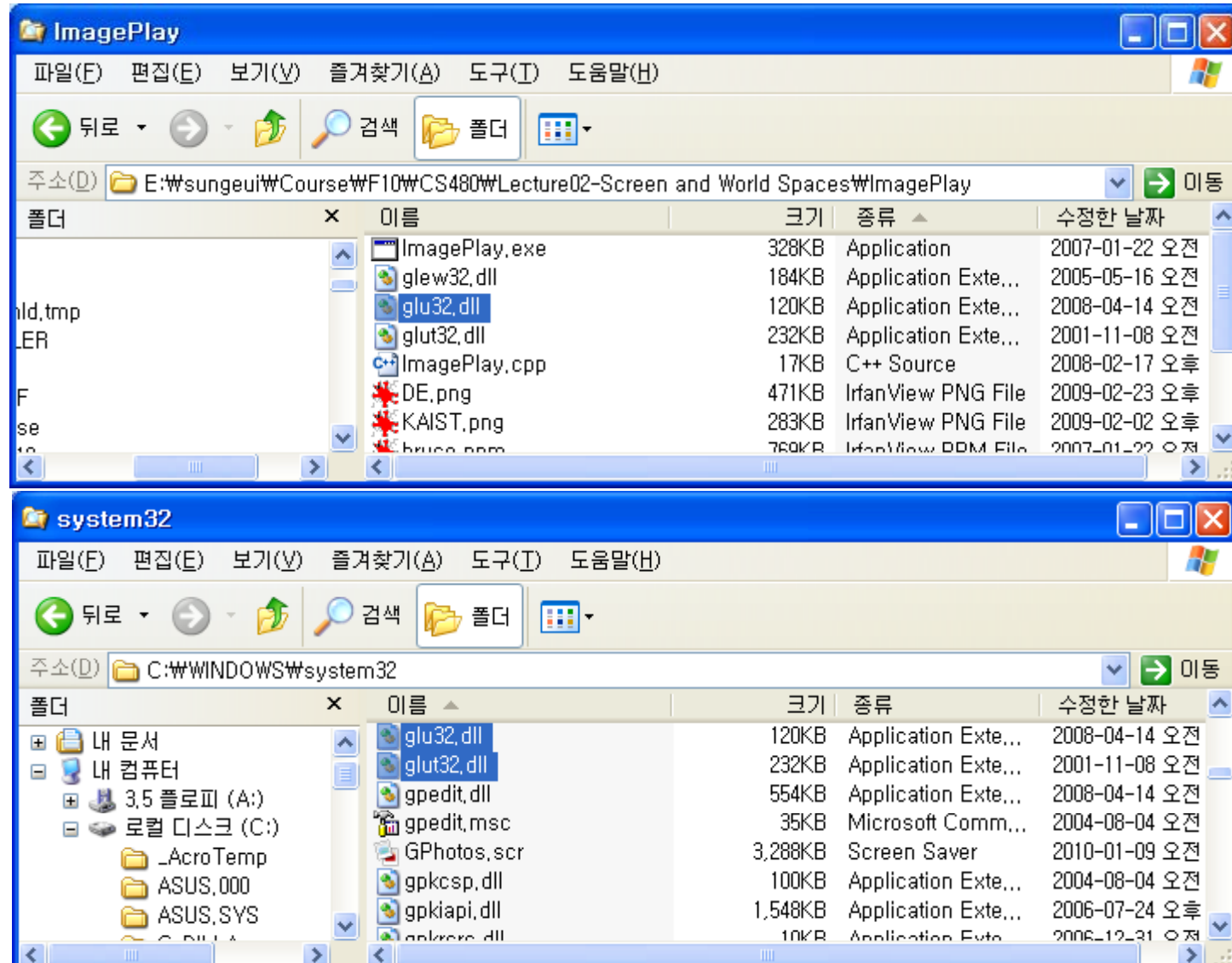
```
class Complex {
    float re, im;
};

Complex c(0.109, 0.603);
int width = 512, height = 512;
```

Example: Header/Lib. Directories with Visual Studio 2005



Example: DLLs for OpenGL



Initializing GLUT

```
void main (int argc, char * argv []) {  
    glutInit(& argc, argv);  
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);  
  
    glutInitWindowSize(width, height);  
    glutInitWindowPosition(100, 100);  
    glutCreateWindow("Julia Set");  
  
    glutDisplayFunc(display);  
    glutMouseFunc(onMouseButton);  
    glutKeyboardFunc(onKeyPress);  
    glutReshapeFunc(onReshape);  
  
    Initialize ();  
    glutMainLoop();  
}
```

Initialize

- Executed at the beginning of display():

```
void initialize()
{
    // Clear the screen
    glClearColor(0,0,1,0);
    glClear(GL_COLOR_BUFFER_BIT);

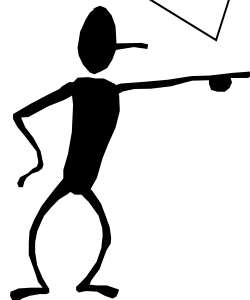
    glMatrixMode(GL_PROJECTION); // related to a camera setting
    glLoadIdentity();
    gluOrtho2D(world.l, world.r, world.b, world.t);

    glMatrixMode(GL_MODELVIEW); // related to model transformation
    glLoadIdentity();
}
```

Reshape

- Reshape gets called when the window size changes

Keep center of world in the center of the screen



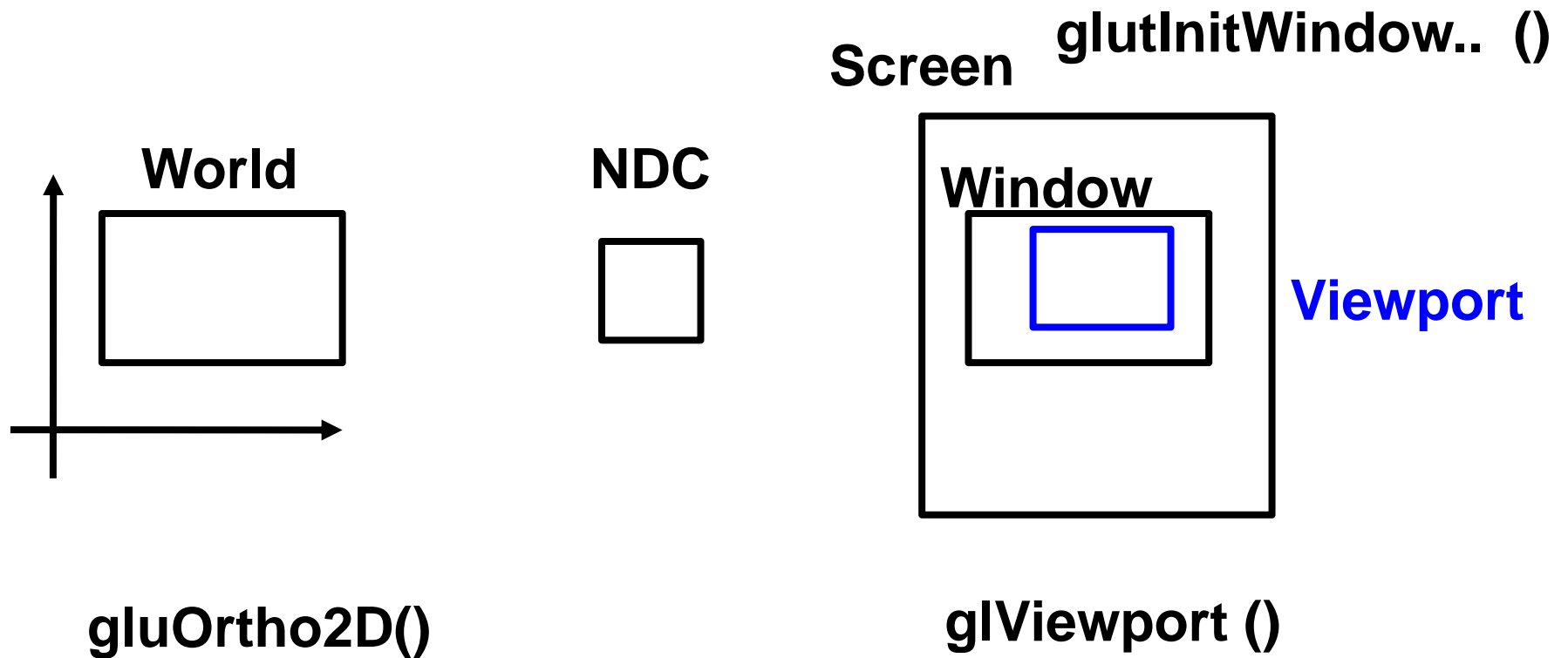
```
void onReshape (int w, int h)
{
    width = w;
    height = h;

    glViewport (0, 0, w, h);

    float cx = 0.5*(world.r + world.l);
    float dy = world.t - world.b;;

    world.l = cx - 0.5*dy * w/h;
    world.r = cx + 0.5*dy * w/h;
}
```

Mapping from World to Screen in OpenGL



Main Display Code

```
void display () {
    initialize();

    float delta = (world.r - world.l)/float(width);
    for( int j=0; j < height; j++ ) {
        for( int i=0; i < width; i++ ) {
            float x = world.l + i*delta;           // convert pixel location to world coordinates
            float y = world.b + j*delta;

            int its; float R; Complex p(x,y);
            julia( p, c, its, R );
            if (its == 255)                         // set a color
                glColor3d(0,0,0);
            else {
                float r = R/float(3); float g = its/float(128); float b = R/float(its+1);
                glColor3d(r,g,b);
            }

            glBegin(GL_POLYGON)                    // Draw pixel
            glVertex2d(x, y);
            glVertex2d(x, y+delta);
            glVertex2d(x+delta, y+delta);
            glVertex2d(x+delta, y);
            glEnd();
        }
    }
    glFlush();
}
```



Now the GUI Stuff

```
void mouse( int button, int state, int mx, int my )
{
    float x = xScreenToWorld(mx);
    float y = yScreenToWorld(my);

    float dx = (world.r - world.l);
    float dy = (world.t - world.b);

    if( (button == GLUT_LEFT_BUTTON) && (state == GLUT_DOWN) ) {
        world.l = x - dx/4;    world.r = x + dx/4;
        world.b = y - dy/4;    world.t = y + dy/4;
    }
    else if( (button == GLUT_RIGHT_BUTTON) && (state == GLUT_DOWN) ) {
        world.l = x - dx;    world.r = x + dx;
        world.b = y - dy;    world.t = y + dy;
    }

    glutPostRedisplay ();
}
```

Screen-to-World Mapping

```
float xScreenToWorld(float scrX)
{
    return ((world.r - world.l) * scrX / float(width)) + world.l;
}
```

```
float yScreenToWorld(float scrY)
{
    return ((world.t - world.b) * (1 - scrY / float(height))) + world.b;
}
```

No OpenGL function for this!

Keyboard Handling

```
void keyboard (unsigned char key, int x, int y)
{
    if ((key == 'r') || (key == 'R'))
    {
        // return to initial position
        c = Complex(0.109, 0.603);
        world.l = -1;    world.r = 1;
        world.b = -1;    world.t = 1;
    }

    glutPostRedisplay ();
}
```

Source Code

- C code is available at the course homepage

Class Objectives were:

- Understand the basic OpenGL program structure and how OpenGL supports different spaces

Homework

- Download the code, compile the code, and play it



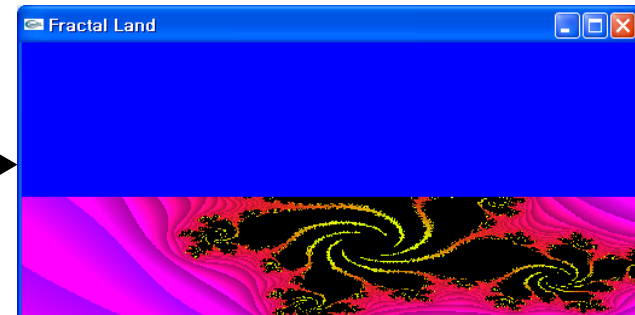
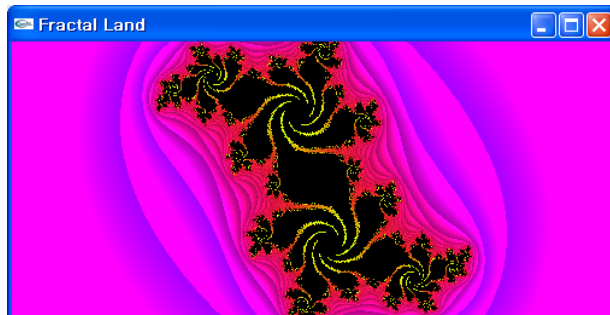
Homework

- Make it work even if using the following code:

```
void reshape( int w, int h)
{
    width = w;  height = h;
    glViewport(0, 0, w, h );

    float cx = 0.5*(world.r + world.l);
    float dy = world.t - world.b;;
    world.l = cx - 0.5*dy * w/h;
    world.r = cx + 0.5*dy * w/h;
}
```

```
void reshape( int w, int h)
{
    width = w;
    height = h;
    glViewport(0, 0, w, h );
}
```



Homework

- Read:
 - Sec. 5: Transformation Matrices

Any Questions?

- **Write down question on what we have discussed in the class**
 - 1 for already answered questions
 - 2 for typical questions
 - 3 for questions with thoughts
 - 4 for questions that surprised me

Homework

- **Go over the next lecture slides before the class**
- **Watch 2 SIGGRAPH Videos and submit their abstract every Wed. class**

Next Time

- Transformations

