Lecture 10: 3D Shapes (OGL01Shape3D.cpp)

This example displays a 3D color-cube and a pyramid. The cube is made of of 6 quads, each having different colors. The hallow pyramid is made up of 4 triangle, with different colors on each of the vertices.

```
 1/*
 2 * OGL01Shape3D.cpp: 3D Shapes
 3 */
 4#include <windows.h>  // for MS Windows
 5#include <GL/glut.h>  // GLUT, include glu.h and gl.h
 6
 7/* Global variables */
 8char title[] = "3D Shapes";
 9
10/* Initialize OpenGL Graphics */
11void initGL() {
12   glClearColor(0.0f, 0.0f, 0.0f, 1.0f); // Set background color to black and opaque
13   glClearDepth(1.0f);                   // Set background depth to farthest
14   glEnable(GL_DEPTH_TEST);   // Enable depth testing for z-culling
15   glDepthFunc(GL_LEQUAL);    // Set the type of depth-test
16   glShadeModel(GL_SMOOTH);   // Enable smooth shading
17   glHint(GL_PERSPECTIVE_CORRECTION_HINT, GL_NICEST);  // Nice perspective corrections
18}
19
20/* Handler for window-repaint event. Called back when the window first appears and
21   whenever the window needs to be re-painted. */
22void display() {
23   glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT); // Clear color and depth buffers
24   glMatrixMode(GL_MODELVIEW);     // To operate on model-view matrix
25
26   // Render a color-cube consisting of 6 quads with different colors
27   glLoadIdentity();                 // Reset the model-view matrix
28   glTranslatef(1.5f, 0.0f, -7.0f);  // Move right and into the screen
29
30   glBegin(GL_QUADS);                // Begin drawing the color cube with 6 quads
31      // Top face (y = 1.0f)
32      // Define vertices in counter-clockwise (CCW) order with normal pointing out
33      glColor3f(0.0f, 1.0f, 0.0f);     // Green
34      glVertex3f( 1.0f, 1.0f, -1.0f);
35      glVertex3f(-1.0f, 1.0f, -1.0f);
36      glVertex3f(-1.0f, 1.0f,  1.0f);
37      glVertex3f( 1.0f, 1.0f,  1.0f);
38
39      // Bottom face (y = -1.0f)
40      glColor3f(1.0f, 0.5f, 0.0f);     // Orange
```

```
41      glVertex3f( 1.0f, -1.0f,  1.0f);
42      glVertex3f(-1.0f, -1.0f,  1.0f);
43      glVertex3f(-1.0f, -1.0f, -1.0f);
44      glVertex3f( 1.0f, -1.0f, -1.0f);
45
46      // Front face  (z = 1.0f)
47      glColor3f(1.0f, 0.0f, 0.0f);     // Red
48      glVertex3f( 1.0f,  1.0f, 1.0f);
49      glVertex3f(-1.0f,  1.0f, 1.0f);
50      glVertex3f(-1.0f, -1.0f, 1.0f);
51      glVertex3f( 1.0f, -1.0f, 1.0f);
52
53      // Back face (z = -1.0f)
54      glColor3f(1.0f, 1.0f, 0.0f);     // Yellow
55      glVertex3f( 1.0f, -1.0f, -1.0f);
56      glVertex3f(-1.0f, -1.0f, -1.0f);
57      glVertex3f(-1.0f,  1.0f, -1.0f);
58      glVertex3f( 1.0f,  1.0f, -1.0f);
59
60      // Left face (x = -1.0f)
61      glColor3f(0.0f, 0.0f, 1.0f);     // Blue
62      glVertex3f(-1.0f,  1.0f,  1.0f);
63      glVertex3f(-1.0f,  1.0f, -1.0f);
64      glVertex3f(-1.0f, -1.0f, -1.0f);
65      glVertex3f(-1.0f, -1.0f,  1.0f);
66
67      // Right face (x = 1.0f)
68      glColor3f(1.0f, 0.0f, 1.0f);     // Magenta
69      glVertex3f(1.0f,  1.0f, -1.0f);
70      glVertex3f(1.0f,  1.0f,  1.0f);
71      glVertex3f(1.0f, -1.0f,  1.0f);
72      glVertex3f(1.0f, -1.0f, -1.0f);
73  glEnd();  // End of drawing color-cube
74
75  // Render a pyramid consists of 4 triangles
76  glLoadIdentity();                  // Reset the model-view matrix
77  glTranslatef(-1.5f, 0.0f, -6.0f);  // Move left and into the screen
78
79  glBegin(GL_TRIANGLES);           // Begin drawing the pyramid with 4 triangles
80     // Front
81     glColor3f(1.0f, 0.0f, 0.0f);     // Red
82     glVertex3f( 0.0f, 1.0f, 0.0f);
83     glColor3f(0.0f, 1.0f, 0.0f);     // Green
84     glVertex3f(-1.0f, -1.0f, 1.0f);
85     glColor3f(0.0f, 0.0f, 1.0f);     // Blue
```

```
 86      glVertex3f(1.0f, -1.0f, 1.0f);
 87
 88      // Right
 89      glColor3f(1.0f, 0.0f, 0.0f);     // Red
 90      glVertex3f(0.0f, 1.0f, 0.0f);
 91      glColor3f(0.0f, 0.0f, 1.0f);     // Blue
 92      glVertex3f(1.0f, -1.0f, 1.0f);
 93      glColor3f(0.0f, 1.0f, 0.0f);     // Green
 94      glVertex3f(1.0f, -1.0f, -1.0f);
 95
 96      // Back
 97      glColor3f(1.0f, 0.0f, 0.0f);     // Red
 98      glVertex3f(0.0f, 1.0f, 0.0f);
 99      glColor3f(0.0f, 1.0f, 0.0f);     // Green
100      glVertex3f(1.0f, -1.0f, -1.0f);
101      glColor3f(0.0f, 0.0f, 1.0f);     // Blue
102      glVertex3f(-1.0f, -1.0f, -1.0f);
103
104      // Left
105      glColor3f(1.0f,0.0f,0.0f);       // Red
106      glVertex3f( 0.0f, 1.0f, 0.0f);
107      glColor3f(0.0f,0.0f,1.0f);       // Blue
108      glVertex3f(-1.0f,-1.0f,-1.0f);
109      glColor3f(0.0f,1.0f,0.0f);       // Green
110      glVertex3f(-1.0f,-1.0f, 1.0f);
111   glEnd();   // Done drawing the pyramid
112
113   glutSwapBuffers();  // Swap the front and back frame buffers (double buffering)
114}
115
116/* Handler for window re-size event. Called back when the window first appears and
117   whenever the window is re-sized with its new width and height */
118void reshape(GLsizei width, GLsizei height) {  // GLsizei for non-negative integer
119   // Compute aspect ratio of the new window
120   if (height == 0) height = 1;                // To prevent divide by 0
121   GLfloat aspect = (GLfloat)width / (GLfloat)height;
122
123   // Set the viewport to cover the new window
124   glViewport(0, 0, width, height);
125
126   // Set the aspect ratio of the clipping volume to match the viewport
127   glMatrixMode(GL_PROJECTION);  // To operate on the Projection matrix
128   glLoadIdentity();             // Reset
129   // Enable perspective projection with fovy, aspect, zNear and zFar
130   gluPerspective(45.0f, aspect, 0.1f, 100.0f);
```

```
131}
132
133/* Main function: GLUT runs as a console application starting at main() */
134int main(int argc, char** argv) {
135   glutInit(&argc, argv);            // Initialize GLUT
136   glutInitDisplayMode(GLUT_DOUBLE); // Enable double buffered mode
137   glutInitWindowSize(640, 480);   // Set the window's initial width & height
138   glutInitWindowPosition(50, 50); // Position the window's initial top-left corner
139   glutCreateWindow(title);          // Create window with the given title
140   glutDisplayFunc(display);       // Register callback handler for window re-paint ever
141   glutReshapeFunc(reshape);       // Register callback handler for window re-size event
142   initGL();                       // Our own OpenGL initialization
143   glutMainLoop();                 // Enter the infinite event-processing loop
144   return 0;
145}
```