**计算机图形学实验**

**姓 名：白文强**

**学 号：20191060064**

**专 业：计算机科学与技术**

**教 师：钱文华**

实验九 三维图形几何变换实验

时间：2020年12月11日

地点：信息学院2202机房

1. 实验内容

教材P222，三维图形旋转、缩放变换、平移变换、错切变换、对称变换等任意一个变换。

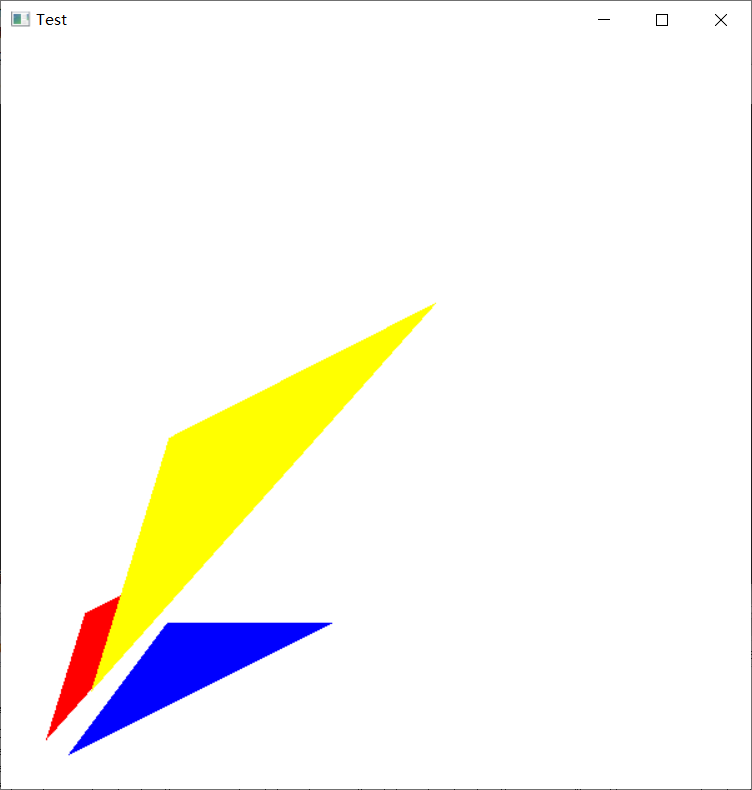
2、实验目的

调用函数完成三维图形几何变换。

3、实验代码

1. #include <windows.h>
2. #include <GL/glut.h>
3. #include <stdlib.h>
4. #include <math.h>
5. #include<iostream>
7. GLsizei winWidth = 600, winHeight = 600;
8. GLuint regHex;//显示表标识
9. /\*设置世界坐标范围\*/
10. GLfloat xwcMin = 0.0, xwcMax = 225.0;
11. GLfloat ywcMin = 0.0, ywcMax = 225.0;
13. **class** wcPt3D
14. {
15. **public**:
16. GLfloat x, y, z;
17. };
18. **typedef** GLfloat Matrix4x4[4][4];
19. Matrix4x4 matComposite;
21. **void** init(**void**)
22. {
23. glClearColor(1.0, 1.0, 1.0, 0.0);
25. }
27. **void** matrix4x4SetIdentity(Matrix4x4 matIdent4x4)
28. {
29. GLint row, col;
30. **for**(row = 0; row < 4; row++)
31. {
32. **for**(col = 0; col < 4; col++)
33. {
34. matIdent4x4[row][col] = (row == col);
35. }
36. }
37. }
38. **void** matrix4x4PreMultiply(Matrix4x4 m1, Matrix4x4 m2)
39. {
40. GLint row, col;
41. Matrix4x4 matTemp;
42. **for**(row = 0; row < 4; row++)
43. {
44. **for**(col = 0; col < 4; col++)
45. {
46. matTemp[row][col] = m1[row][0] \* m2[0][col] + m1[row][1] \* m2[1][col] + m1[row][2] \* m2[2][col] + m1[row][3] \* m2[3][col];
47. }
48. }
49. **for**(row = 0; row < 4; row++)
50. {
51. **for**(col = 0; col < 4; col++)
52. {
53. m2[row][col] = matTemp[row][col];
54. }
55. }
56. }
58. **void** translate3D(GLfloat tx, GLfloat ty, GLfloat tz)
59. {
60. Matrix4x4 matTransl3D;
61. matrix4x4SetIdentity(matTransl3D);
62. matTransl3D[0][3] = tx;
63. matTransl3D[1][3] = ty;
64. matTransl3D[2][3] = tz;
65. matrix4x4PreMultiply(matTransl3D, matComposite);
66. }
68. **void** rotate3D(wcPt3D p1, wcPt3D p2, GLfloat radianAngle)
69. {
70. Matrix4x4 matQuatRot;
71. **float** axisVectLength = sqrt((p2.x - p1.x) \* (p2.x - p1.x) + (p2.y - p1.y) \* (p2.y - p1.y) + (p2.z - p1.z) \* (p2.z - p1.z));
72. **float** cosA = cos(radianAngle);
73. **float** oneC = 1 - cosA;
74. **float** sinA = sin(radianAngle);
75. **float** ux = (p2.x - p1.x) / axisVectLength;
76. **float** uy = (p2.y - p1.y) / axisVectLength;
77. **float** uz = (p2.z - p1.z) / axisVectLength;
78. translate3D(-p1.x, -p1.y, -p1.z);
79. matrix4x4SetIdentity(matQuatRot);
80. matQuatRot[0][0] = ux \* ux \* oneC + cosA;
81. matQuatRot[0][1] = ux \* uy \* oneC - uz \* sinA;
82. matQuatRot[0][2] = ux \* uz \* oneC + uy \* sinA;
83. matQuatRot[1][0] = uy \* ux \* oneC + uz \* sinA;
84. matQuatRot[1][1] = uy \* uy \* oneC + cosA;
85. matQuatRot[1][2] = uy \* uz \* oneC - ux \* sinA;
86. matQuatRot[2][0] = ux \* uz \* oneC - uy \* sinA;
87. matQuatRot[2][1] = uz \* uy \* oneC + ux \* sinA;
88. matQuatRot[2][2] = uz \* uz \* oneC + cosA;
89. matrix4x4PreMultiply(matQuatRot, matComposite);
90. translate3D(p1.x, p1.y, p1.z);
91. }
93. **void** scale3D(GLfloat sx, GLfloat sy, GLfloat sz, wcPt3D fixedPt)
94. {
95. Matrix4x4 matScale3D;
96. matrix4x4SetIdentity(matScale3D);
97. matScale3D[0][0] = sx;
98. matScale3D[0][3] = (1 - sx) \* fixedPt.x;
99. matScale3D[1][1] = sy;
100. matScale3D[1][3] = (1 - sy) \* fixedPt.y;
101. matScale3D[2][2] = sz;
102. matScale3D[2][3] = (1 - sz) \* fixedPt.z;
103. matrix4x4PreMultiply(matScale3D, matComposite);
104. }
106. **void** translate3d(GLint nverts, wcPt3D \*verts)
107. {
108. //对象顶点与复合变换矩阵的乘法，得到坐标变换后的点
109. GLint k;
110. **for**(k = 0; k < nverts; ++k)
111. {
112. verts[k].x = verts[k].x \* matComposite[0][0] + verts[k].y \* matComposite[0][1] + matComposite[0][2];
113. verts[k].y = verts[k].x \* matComposite[1][0] + verts[k].y \* matComposite[1][1] + matComposite[1][2];
114. }
115. }
117. **void** triangle(wcPt3D \*verts)
118. {
119. //输出一个空间三角形
120. GLint k;
121. glBegin(GL\_TRIANGLES);
122. **for**(k = 0; k < 3; ++k)
123. {
124. glVertex3f(verts[k].x, verts[k].y, verts[k].z);
125. }
126. glEnd();
127. }
129. **void** displayFcn()
130. {
131. GLint nverts = 3;
132. wcPt3D verts[3] = { { 20, 10.0, 0 }, { 100.0, 50.0, 50}, { 50.0, 50.0, 10 } }; //定义一个三角形
133. GLdouble theta = 3.14159 / 4.0;//旋转45°
134. wcPt3D p1, p2;
135. p1.x = 0; p1.y = 0; p1.z = 0;
136. p2.x = 1; p2.y = 1; p2.z = 1;//定义旋转轴为(0,0,1)
137. glClear(GL\_COLOR\_BUFFER\_BIT);
138. glColor3f(0.0, 0.0, 1.0);
139. triangle(verts);//蓝色三角形为原始三角形
140. matrix4x4SetIdentity(matComposite);
142. rotate3D(p1, p2, theta); //旋转
143. translate3d(nverts, verts);//对三角形坐标按照矩阵变换
144. glColor3f(1.0, 0.0, 0.0);//红色为移动后的三角形
145. triangle(verts);
146. matrix4x4SetIdentity(matComposite);
148. scale3D( 2, 2, 2 , p1);//缩放
149. translate3d(nverts, verts);//对三角形坐标按照矩阵变换
150. glColor3f(1.0, 1.0, 0.0);//红色为移动后的三角形
151. triangle(verts);
152. matrix4x4SetIdentity(matComposite);
154. glFlush();
155. }
157. **void** winReshapeFcn(**int** newWidth, **int** newHeight) {
158. //窗口重定形函数
159. glMatrixMode(GL\_PROJECTION);
160. glLoadIdentity();//将当前的用户坐标系的原点移到了屏幕中心：类似于一个复位操作
161. glOrtho(xwcMin, xwcMax, ywcMin, ywcMax,-55,0);
162. glClear(GL\_COLOR\_BUFFER\_BIT);
163. }
165. **int** main(**int** argc, **char** \*\*argv)
166. {
167. glutInit(&argc, argv);
168. glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);
169. glutInitWindowPosition(50, 50);
170. glutInitWindowSize(600, 600);
171. glutCreateWindow("Test");
172. init();
173. glutDisplayFunc(displayFcn);
174. glutReshapeFunc(winReshapeFcn);
175. glutMainLoop();
176. **return** 0;
177. }

4、实验结果



原图

旋转后

旋转后缩放

5、实验总结

利用每种变换时对应的齐次矩阵，将三维坐标按照齐次矩阵进行变换从而实现图形在三维坐标上的变换。