

Department of Computer Science and Engineering

Shivanirudh S G, 185001146, Semester VII

23 September 2021

UCS1712 - Graphics and Multimedia Lab

Exercise 8: 3-Dimensional Transformations in C++ using OpenGL

Aim:

Perform basic 3D Transformations on any 3D Object.

Code:

```
1  #ifndef LOPENGL_H
2  #define LOPENGL_H
3
4  #include <GL/freeglut.h>
5  #include <GL/gl.h>
6  #include <GL/glu.h>
7  #include <math.h>
8  #include <stdio.h>
9  #include <stdlib.h>
10 #include <iostream>
11 #include <vector>
12 #include <ctime>
13 #include <tuple>
14 #include <unistd.h>
15 using namespace std;
16
17 #endif
```

```

1  #ifndef LUTIL_H
2  #define LUTIL_H
3
4  #include "Headers.h"
5
6  //Screen Constants
7  const int SCREEN_WIDTH = 1362;
8  const int SCREEN_HEIGHT = 750;
9  const int SCREEN_FPS = 60;
10 const int POINT_SIZE=3;
11
12 typedef float MatrixDim [4][4];
13 MatrixDim transformation_matrix;
14
15 static GLfloat input[8][3]={
16     {40,40,-50},{90,40,-50},{90,90,-50},{40,90,-50},
17     {30,30,0},{80,30,0},{80,80,0},{30,80,0}
18 };
19 };
20
21 float output[8][3];
22 float tx,ty,tz;
23 float sx,sy,sz;
24 float angle;
25
26 int choice,choiceRot;
27
28 void init();
29
30 void render();
31
32 void setIdentityM(MatrixDim m);
33
34 void translate(int tx, int ty, int tz);
35
36 void scale(int sx, int sy, int sz);
37
38 void RotateX(float angle);
39
40 void RotateY(float angle);
41
42 void RotateZ(float angle);
43
44 void multiplyMatrices();
45
46 void Axes(void);
47
48 void draw(float a[8][3]);
49
50 #endif

```



```

1  #include "Signatures.h"
2
3  void init(){
4      glClearColor(0.0,0.0,0.0,1.0);
5      glOrtho(-454.0,454.0,-250.0,250.0,-250.0,250.0);
6      glEnable(GL_DEPTH_TEST);
7  }

```

```

8
9 void render(){
10
11
12 while(true){
13     glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT);
14     Axes();
15     glColor3f(1.0,0.0,0.0);
16     draw(input);
17     setIdentityM(transformation_matrix);
18
19     cout<<"Choose transformation: "<<endl;
20     cout<<"1 for Translation"<<endl<<"2 for Rotation"<<endl;
21     cout<<"3 for Scaling"<<endl<<"0 to Exit"<<endl;
22     cout<<"Enter your choice: ";cin>>choice;
23
24     cout<<"transformation: "<<choice<<endl;
25
26     if(choice == 1){
27         cout<<"Enter the translation factor for X, Y and Z: ";
28         cin>>tx >> ty >> tz;
29     }
30     else if(choice == 2){
31         cout<<"Enter the rotation angle: ";
32         cin>>angle;
33
34         cout<<"Choose axis to rotate around: "<<endl;
35         cout<<"1 for around X axis"<<endl<<"2 for around Y axis
36         "<<endl;
37         cout<<"3 for around Z axis"<<endl;
38         cout<<"Enter your choice: ";cin>>choiceRot;
39     }
40     else if(choice == 3){
41         cout<<"Enter the scaling factor for X, Y and Z: ";
42         cin>>sx >> sy >> sz;
43     }
44     else if(choice){
45         cout<<"Invalid option"<<endl;
46     }
47     else;
48
49     switch(choice){
50     case 1:
51         translate(tx,ty,tz);
52         break;
53     case 2:
54         switch (choiceRot) {
55             case 1:
56                 RotateX(angle);
57                 break;
58             case 2:
59                 RotateY(angle);
60                 break;
61             case 3:
62                 RotateZ(angle);
63                 break;

```

```

64         default:
65             break;
66     }
67     multiplyMatrices();
68     for (int i = 0; i < 8; i++){
69         for (int j = 0; j < 3; j++){
70             cout << output[i][j] << " ";
71         }
72         cout << endl;
73     }
74     break;
75     case 3:
76         scale(sx, sy, sz);
77         multiplyMatrices();
78         break;
79     }
80
81     draw(output);
82     glFlush();
83 }
84 }
85
86 void setIdentityM(MatrixDim m){
87     for(int i=0; i<4; i++){
88         for(int j=0; j<4; j++){
89             m[i][j]=(i==j);
90         }
91     }
92     void translate(int tx, int ty, int tz){
93
94         for(int i=0; i<8; i++){
95             output[i][0]=input[i][0]+tx;
96             output[i][1]=input[i][1]+ty;
97             output[i][2]=input[i][2]+tz;
98         }
99     }
100
101     void scale(int sx, int sy, int sz){
102         transformation_matrix[0][0]=sx;
103         transformation_matrix[1][1]=sy;
104         transformation_matrix[2][2]=sz;
105     }
106
107     void RotateX(float angle){
108         cout << angle << endl;
109         angle = angle * 3.1416 / 180;
110         cout << angle << endl;
111         transformation_matrix[1][1] = cos(angle);
112         transformation_matrix[1][2] = -sin(angle);
113         transformation_matrix[2][1] = sin(angle);
114         transformation_matrix[2][2] = cos(angle);
115     }
116
117     void RotateY(float angle){
118         angle = angle*3.1416/180;
119         transformation_matrix[0][0] = cos(angle);
120         transformation_matrix[0][2] = -sin(angle);

```

```

121     transformation_matrix[2][0] = sin(angle);
122     transformation_matrix[2][2] = cos(angle);
123
124 }
125
126 void RotateZ(float angle){
127     angle = angle*3.1416/180;
128     transformation_matrix[0][0] = cos(angle);
129     transformation_matrix[0][1] = sin(angle);
130     transformation_matrix[1][0] = -sin(angle);
131     transformation_matrix[1][1] = cos(angle);
132 }
133
134 void multiplyMatrices(){
135     for(int i=0;i<8;i++){
136         for(int j=0;j<3;j++){
137             output[i][j]=0;
138             for(int k=0;k<3;k++){
139                 output[i][j]+=(input[i][k]*transformation_matrix[k
140             ]
141         ]
142     ]
143     ]
144 }
145 void Axes(void){
146     glColor3f (1.0, 1.0, 1.0);
147     glBegin(GL_LINES);
148         glVertex2s(-1000 ,0);
149         glVertex2s( 1000 ,0);
150     glEnd();
151     glBegin(GL_LINES);
152         glVertex2s(0 ,-1000);
153         glVertex2s(0 , 1000);
154     glEnd();
155 }
156 void draw(float a[8][3]){
157     glBegin(GL_QUADS);
158         glColor3f(0.7,0.4,0.5); //behind
159         glVertex3fv(a[0]);
160         glVertex3fv(a[1]);
161         glVertex3fv(a[2]);
162         glVertex3fv(a[3]);
163
164         glColor3f(0.8,0.2,0.4); //bottom
165         glVertex3fv(a[0]);
166         glVertex3fv(a[1]);
167         glVertex3fv(a[5]);
168         glVertex3fv(a[4]);
169
170         glColor3f(0.3,0.6,0.7); //left
171         glVertex3fv(a[0]);
172         glVertex3fv(a[4]);
173         glVertex3fv(a[7]);
174         glVertex3fv(a[3]);
175
176         glColor3f(0.2,0.8,0.2); //right

```

```

177         glVertex3fv(a[1]);
178         glVertex3fv(a[2]);
179         glVertex3fv(a[6]);
180         glVertex3fv(a[5]);
181
182         glColor3f(0.7,0.7,0.2); //up
183         glVertex3fv(a[2]);
184         glVertex3fv(a[3]);
185         glVertex3fv(a[7]);
186         glVertex3fv(a[6]);
187
188         glColor3f(1.0,0.1,0.1);
189         glVertex3fv(a[4]);
190         glVertex3fv(a[5]);
191         glVertex3fv(a[6]);
192         glVertex3fv(a[7]);
193
194     glEnd();
195 }

1 #include "Helpers.h"
2
3 int main( int argc, char* args[] ){
4
5     glutInit(&argc,args);
6     glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB|GLUT_DEPTH);
7     glutInitWindowSize(1362,750);
8     glutInitWindowPosition(0,0);
9     glutCreateWindow( "OpenGL" );
10
11     init();
12
13     glutDisplayFunc(render);
14
15     glutMainLoop();
16
17     return 0;
18 }

```

Output:

Translation:

Choose transformation:

1 for Translation

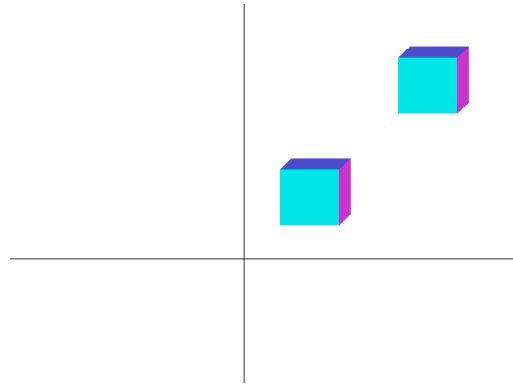
2 for Rotation

3 for Scaling

0 to Exit

Enter your choice: 1

Enter the translation factor for X, Y and Z: 100 100 100



Rotation - X axis:

Choose transformation:

1 for Translation

2 for Rotation

3 for Scaling

0 to Exit

Enter your choice: 2

Enter the rotation angle: 45

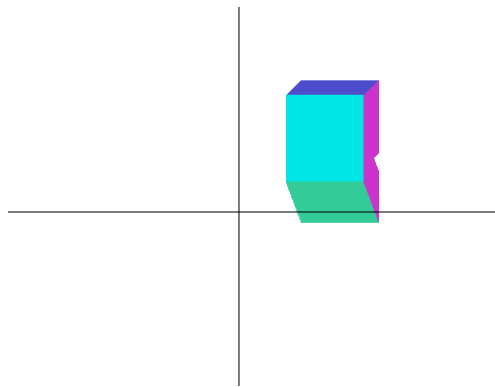
Choose axis to rotate around:

1 for around X axis

2 for around Y axis

3 for around Z axis

Enter your choice: 1



Rotation - Y axis:

Choose transformation:

1 for Translation

2 for Rotation

3 for Scaling

0 to Exit

Enter your choice: 2

Enter the rotation angle: 45

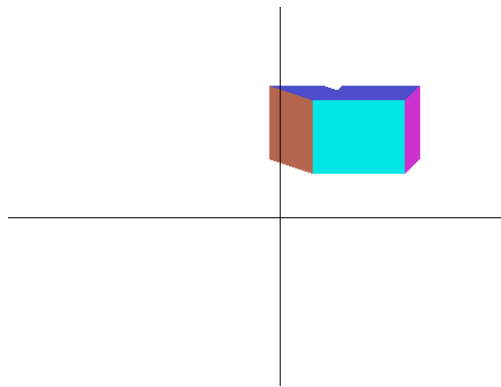
Choose axis to rotate around:

1 for around X axis

2 for around Y axis

3 for around Z axis

Enter your choice: 2



Rotation - Z axis:

Choose transformation:

1 for Translation

2 for Rotation

3 for Scaling

0 to Exit

Enter your choice: 2

Enter the rotation angle: 45

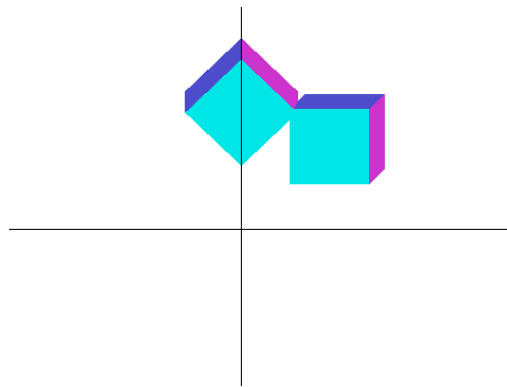
Choose axis to rotate around:

1 for around X axis

2 for around Y axis

3 for around Z axis

Enter your choice: 3



Scaling:

Choose transformation:

1 for Translation

2 for Rotation

3 for Scaling

0 to Exit

Enter your choice: 3

Enter the translation factor for X, Y and Z: 2 2 2

