学号：E51841014 姓名：吴振龙 专业：数字媒体技术

实验5 创建和观察三维物体

【实验目的】

1. 学习创建3D物体。

2. 学习3D几何变换方法。

3. 掌握投影变换和照相机位置及方向的设置方法，进行三维观察。

【实验原理】

1.对比以下函数的区别；

glOrtho(GLdouble left，GLdouble right，GLdouble bottom，GLdouble top，

GLdouble near，GLdouble far)；

和gluOrtho2D(GLdouble left，GLdouble right，GLdouble bottom，

GLdouble top)；

2.设置照相机的位置和朝向：gluLookAt(eyex, eyey, eyez, atx, aty, atz, upx, upy, upz);

3. 根据索引列表绘制多边形的原理；

4. 隐藏面消除的函数及原理。

5. 用户交互与三维物体观察的结合，主要用到的函数有

glMatrixMode设置当前矩阵模式:

GL\_MODELVIEW,对模型视景矩阵堆栈应用随后的矩阵操作.

GL\_PROJECTION,对投影矩阵应用随后的矩阵操作.

gluPerspective(fovy, aspect, near, far); 透视投影

glOrtho(left, right, bottom, top, near, far); 正投影

gluLookAt(eyex, eyey, eyez, atx, aty, atz, upx, upy, upz);

glRotated (GLdouble angle, GLdouble x, GLdouble y, GLdouble z);

通过这些函数的调用可以分别在正投影和透视投影下从不同的角度来观察三维物体

【实验内容】

1. 阅读arraycube.c，掌握彩色立方体的建模方法，为程序加注释。

答：加注释后的python版代码见附录一

2. 在arraycube.c的基础上编写一个交互式程序，实现立方体的旋转。具体要求如下：

立方体的旋转方式由鼠标和键盘按键来控制：按下鼠标左键，立方体绕x轴连续旋转；按下鼠标左键+ctrl键，立方体绕y轴连续旋转；按下鼠标右键，立方体绕z轴连续旋转。（注意：旋转的不动点在原点，正好是立方体的中心。）

答：Python版源代码见附录二

3. 修改arraycube.c，实现交互式地移动照相机来观察已经建模好的彩色立方体。即用鼠标或键盘来改变gluLookAt(eyex, eyey, eyez, atx, aty, atz, upx, upy, upz)函数的9个参数，以此来观察立方体。要求：

（1）交互时采用的鼠标和键盘按键自定。例如：每次按下‘q’，eyex增加；每次按下‘Q’，eyex减小。

（2）分别在正投影和透视投影下实现题目中的功能。

（3）（选做）可增加菜单功能，将正投影和透视投影下的观察功能融合到一个程序中。可参考交互式教程Projection.c的功能。

答：正投影和透视投影的区别在于一个使用前者glOrtho函数，后者则使用gluPerspective函数，gluPerspective的参数设置为(50, 1, 3, 5)，具体源代码见附录三。

附录一：

*from* OpenGL.GLUT *import* \*

*from* OpenGL.GL *import* \*

*from* OpenGL.GLU *import* \*

*import* numpy as np

*# 顶点数组*

vertices = np.array([[-1.0, -1.0, 1.0],[-1.0,1.0,1.0],

            [1.0,1.0,1.0], [1.0,-1.0,1.0],[-1.0,-1.0,-1.0],

            [-1.0,1.0,-1.0], [1.0,1.0,-1.0], [1.0,-1.0,-1.0]])

*# 颜色数组*

colors = np.array([[0.0,0.0,0.0], [1.0,0.0,0.0],

    [1.0,1.0,0.0],[0.0,1.0,0.0], [0.0,0.0,1.0],

    [1.0,0.0,1.0], [1.0,1.0,1.0],[0.0,1.0,1.0]])

def polygon(a, b, c , d):

    ''' 绘制正方体的一个面 '''

    glBegin(GL\_QUADS)

    glColor3dv(colors[a])

    glVertex3dv(vertices[a])

    glColor3dv(colors[b])

    glVertex3dv(vertices[b])

    glColor3dv(colors[c])

    glVertex3dv(vertices[c])

    glColor3dv(colors[d])

    glVertex3dv(vertices[d])

    glEnd()

def colorcube():

    ''' 绘制正方体的六个面 '''

    polygon(0,3,2,1)

    polygon(2,3,7,6)

    polygon(0,4,7,3)

    polygon(1,2,6,5)

    polygon(4,5,6,7)

    polygon(0,1,5,4)

def display():

    glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT)

    glMatrixMode(GL\_MODELVIEW)

    glLoadIdentity()

*# 定义摄像机位置*

    gluLookAt(1.0, 1.0, 1.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0)

*# 绘图*

    colorcube()

*# 交换缓存区内容*

    glutSwapBuffers()

def myReshape(w, h):

    glViewport(0, 0, w, h)

    glMatrixMode(GL\_PROJECTION)

    glLoadIdentity()

*if* w <= h:

        glOrtho(-2.0, 2.0, -2.0 \* h / w,

            2.0 \* h / w, -10.0, 10.0)

*else*:

        glOrtho(-2.0 \*  w / h,

            2.0 \*  w / h, -2.0, 2.0, -10.0, 10.0)

    glMatrixMode(GL\_MODELVIEW)

def main():

    glutInit()

    glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB | GLUT\_DEPTH)

    glutInitWindowSize(500, 500)

    glutCreateWindow("colorcube")

    glClearColor(0.0,0.0,0.0,0.0)

*# 设置明暗处理模式*

    glShadeModel(GL\_FLAT)

*# 注册当前窗口的形状变化回调函数*

    glutReshapeFunc(myReshape)

    glutDisplayFunc(display)

    glutMainLoop()

main()

附录二：

*from* OpenGL.GLUT *import* \*

*from* OpenGL.GL *import* \*

*from* OpenGL.GLU *import* \*

*import* numpy as np

*import* time

angle = 0.0

*# axis1为x轴，2为y轴。3为z轴*

axis = 0

*# status为1时代表按下ctrl键*

status = 0

vertices = np.array([[-1.0, -1.0, 1.0],[-1.0,1.0,1.0],

            [1.0,1.0,1.0], [1.0,-1.0,1.0],[-1.0,-1.0,-1.0],

            [-1.0,1.0,-1.0], [1.0,1.0,-1.0], [1.0,-1.0,-1.0]])

*# 颜色数组*

colors = np.array([[0.0,0.0,0.0], [1.0,0.0,0.0],

    [1.0,1.0,0.0],[0.0,1.0,0.0], [0.0,0.0,1.0],

    [1.0,0.0,1.0], [1.0,1.0,1.0],[0.0,1.0,1.0]])

def polygon(a, b, c , d):

    ''' 绘制正方体的一个面 '''

    glBegin(GL\_QUADS)

    glColor3dv(colors[a])

    glVertex3dv(vertices[a])

    glColor3dv(colors[b])

    glVertex3dv(vertices[b])

    glColor3dv(colors[c])

    glVertex3dv(vertices[c])

    glColor3dv(colors[d])

    glVertex3dv(vertices[d])

    glEnd()

def colorcube():

    ''' 绘制正方体的六个面 '''

    polygon(0,3,2,1)

    polygon(2,3,7,6)

    polygon(0,4,7,3)

    polygon(1,2,6,5)

    polygon(4,5,6,7)

    polygon(0,1,5,4)

def display():

    glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT)

    glMatrixMode(GL\_MODELVIEW)

    glLoadIdentity()

*# 定义摄像机位置*

    gluLookAt(1.0, 1.0, 1.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0)

*if* axis == 1:

        glRotatef(angle, 1.0, 0.0, 0.0)

*elif* axis == 2:

        glRotatef(angle, 0.0, 1.0, 0.0)

*elif* axis == 3:

        glRotatef(angle, 0.0, 0.0, 1.0)

*# 绘图*

    colorcube()

*# 交换缓存区内容*

    glutSwapBuffers()

def myReshape(w, h):

    glViewport(0, 0, w, h)

    glMatrixMode(GL\_PROJECTION)

    glLoadIdentity()

*if* w <= h:

        glOrtho(-2.0, 2.0, -2.0 \* h / w,

            2.0 \* h / w, -10.0, 10.0)

*else*:

        glOrtho(-2.0 \*  w / h,

            2.0 \*  w / h, -2.0, 2.0, -10.0, 10.0)

    glMatrixMode(GL\_MODELVIEW)

def rotate():

    global angle

    angle += 10.0

*if* angle > 360:

        angle = angle -360

    time.sleep(0.3)

    glutPostRedisplay()

def mouse(button, state, x, y):

    global axis

    global status

*if* button == GLUT\_LEFT\_BUTTON and state == GLUT\_DOWN:

        axis = 1

*if* glutGetModifiers() == GLUT\_ACTIVE\_ALT and button == GLUT\_LEFT\_BUTTON and state == GLUT\_DOWN:

        axis = 2

*if* button == GLUT\_RIGHT\_BUTTON and state == GLUT\_DOWN:

        axis = 3

    glutIdleFunc(rotate)

*if* (button == GLUT\_LEFT\_BUTTON or button == GLUT\_RIGHT\_BUTTON) and state == GLUT\_UP:

        glutIdleFunc(None)

def main():

    glutInit()

    glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB | GLUT\_DEPTH)

    glutInitWindowSize(500, 500)

    glutCreateWindow("colorcube")

    glClearColor(0.0,0.0,0.0,0.0)

*# 设置明暗处理模式*

    glShadeModel(GL\_SMOOTH)

*# 注册当前窗口的形状变化回调函数*

    glutReshapeFunc(myReshape)

    glutDisplayFunc(display)

    glutMouseFunc(mouse)

    glutMainLoop()

main()

附录三：

*from* OpenGL.GLUT *import* \*

*from* OpenGL.GL *import* \*

*from* OpenGL.GLU *import* \*

*import* numpy as np

*# 摄像机9个参数的增量*

indent = [0, 0, 0, 0, 0, 0, 0, 0, 0]

*# 顶点数组*

vertices = np.array([[-1.0, -1.0, 1.0],[-1.0,1.0,1.0],

            [1.0,1.0,1.0], [1.0,-1.0,1.0],[-1.0,-1.0,-1.0],

            [-1.0,1.0,-1.0], [1.0,1.0,-1.0], [1.0,-1.0,-1.0]])

*# 颜色数组*

colors = np.array([[0.0,0.0,0.0], [1.0,0.0,0.0],

    [1.0,1.0,0.0],[0.0,1.0,0.0], [0.0,0.0,1.0],

    [1.0,0.0,1.0], [1.0,1.0,1.0],[0.0,1.0,1.0]])

def polygon(a, b, c , d):

    ''' 绘制正方体的一个面 '''

    glBegin(GL\_QUADS)

    glColor3dv(colors[a])

    glVertex3dv(vertices[a])

    glColor3dv(colors[b])

    glVertex3dv(vertices[b])

    glColor3dv(colors[c])

    glVertex3dv(vertices[c])

    glColor3dv(colors[d])

    glVertex3dv(vertices[d])

    glEnd()

def colorcube():

    ''' 绘制正方体的六个面 '''

    polygon(0,3,2,1)

    polygon(2,3,7,6)

    polygon(0,4,7,3)

    polygon(1,2,6,5)

    polygon(4,5,6,7)

    polygon(0,1,5,4)

def display():

    glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT)

    glMatrixMode(GL\_MODELVIEW)

    glLoadIdentity()

*# 定义摄像机位置*

    gluLookAt(1.0 + indent[0], 1.0 + indent[1], 1.0 + indent[2], 0.0 + indent[3], 0.0 + indent[4],

           0.0 + indent[5], 0.0 + indent[6], 1.0 + indent[7], 0.0 + indent[8])

*# 绘图*

    colorcube()

*# 交换缓存区内容*

    glutSwapBuffers()

def myReshape(w, h):

    glViewport(0, 0, w, h)

    glMatrixMode(GL\_PROJECTION)

    glLoadIdentity()

*# if w <= h:*

*#   glOrtho(-2.0, 2.0, -2.0 \* h / w,*

*#       2.0 \* h / w, -10.0, 10.0)*

*# else:*

*#   glOrtho(-2.0 \*  w / h,*

*#       2.0 \*  w / h, -2.0, 2.0, -10.0, 10.0)*

    gluPerspective(50, 1, 3, 5)

    glMatrixMode(GL\_MODELVIEW)

def keyboard(key, x, y):

    global indent

*if* ord(key) == ord('q'):

        indent[0] += 0.1

*elif* ord(key) == ord('Q'):

        indent[0] -= 0.1

*elif* ord(key) == ord('w'):

        indent[1] += 0.1

*elif* ord(key) == ord('W'):

        indent[1] -= 0.1

*elif* ord(key) == ord('e'):

        indent[2] += 0.1

*elif* ord(key) == ord('E'):

        indent[2] -= 0.1

*elif* ord(key) == ord('r'):

        indent[3] += 0.1

*elif* ord(key) == ord('R'):

        indent[3] -= 0.1

*elif* ord(key) == ord('t'):

        indent[4] += 0.1

*elif* ord(key) == ord('T'):

        indent[4] -= 0.1

*elif* ord(key) == ord('y'):

        indent[5] += 0.1

*elif* ord(key) == ord('Y'):

        indent[5] -= 0.1

*elif* ord(key) == ord('u'):

        indent[6] += 0.1

*elif* ord(key) == ord('U'):

        indent[6] -= 0.1

*elif* ord(key) == ord('i'):

        indent[7] += 0.1

*elif* ord(key) == ord('I'):

        indent[7] -= 0.1

*elif* ord(key) == ord('o'):

        indent[8] += 0.1

*elif* ord(key) == ord('O'):

        indent[8] -= 0.1

*elif* ord(key) == ord('p'):

        indent = [0, 0, 0, 0, 0, 0, 0, 0, 0]

    glutPostRedisplay()

def main():

    glutInit()

    glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB | GLUT\_DEPTH)

    glutInitWindowSize(500, 500)

    glutCreateWindow("colorcube")

    glClearColor(0.0,0.0,0.0,0.0)

*# 设置明暗处理模式*

    glShadeModel(GL\_FLAT)

*# 注册当前窗口的形状变化回调函数*

    glutReshapeFunc(myReshape)

    glutDisplayFunc(display)

    glutKeyboardFunc(keyboard)

    glutMainLoop()

main()