## 实验题目

实验目标：使用OpenGL的三维坐标变换功能实现几何建模

实验要求：

1构建一个由4个以上基元几何体构成的复杂场景。

2场景必须是有意义的，变换越复杂得分越高。

3具有一定的动画效果，可使用鼠标/键盘控制

4加载其他模型（可选）

a. 具有动作响应（键/鼠控制的物体的添加/删除等）

b. 具有菜单管理功能(OpenGLglut)

实验环境：python3.9，安装相应库：gl glu glut

## 实验内容

通过使用相关函数，构建了一个书桌，书桌上摆放了一本书，一支笔，一个水杯和一个笔筒。实现了鼠标拖拽旋转，鼠标滚轮控制放大缩小，右击展现菜单进行喝水看书等操作，键盘控制移动，键盘增删笔，改变窗口大小等操作

### 构建画面

书桌 书本 钢笔 水杯 笔筒

基本由正方体、圆环、圆柱体经过旋转、平移、拉伸完成。其中水杯由圆环、圆柱体和圆台面构成。整体由线条组成，有部分为实体填充，显得更美观。

圆台代码：

glTranslatef(-11,4.7,-3)

    glRotatef(90,1,0,0)

    cylinder = gluNewQuadric()

    gluQuadricDrawStyle(cylinder, GLU\_LINE)   #圆台面

    gluCylinder(cylinder, 1.5, 1.3, 3.7, 20, 10)

### 鼠标

通过glut中鼠标相关函数完成：

glutMouseFunc(mouseclick)；glutMotionFunc(mousemotion)

**拖拽旋转**：实际是移动视点坐标完成。通过相关函数计算视点姿态，然后根据鼠标位移计算视点位置。相关代码：

dx = MOUSE\_X - x

        dy = y - MOUSE\_Y

        MOUSE\_X, MOUSE\_Y = x, y

        PHI += 2\*np.pi\*dy/WIN\_H

        PHI %= 2\*np.pi

        THETA += 2\*np.pi\*dx/WIN\_W

        THETA %= 2\*np.pi

        r = DIST\*np.cos(PHI)

        EYE[1] = DIST\*np.sin(PHI)

        EYE[0] = r\*np.sin(THETA)

        EYE[2] = r\*np.cos(THETA)

        if 0.5\*np.pi < PHI < 1.5\*np.pi:

            EYE\_UP[1] = -1.0

        else:

            EYE\_UP[1] = 1.0

**滚轮控制缩放**：实际通过改变缩放向量完成。每次将该向量变大或变小即可实现。相关代码：

    glScale(SCALE\_K[0], SCALE\_K[1], SCALE\_K[2])

………

if button == GLUT\_LEFT\_BUTTON:

        LEFT\_IS\_DOWNED = state==GLUT\_DOWN

    elif button == 3:

        SCALE\_K \*= 1.05

        glutPostRedisplay()

    elif button == 4:

        SCALE\_K \*= 0.95

        glutPostRedisplay()

**右击展示菜单**：通过相关函数完成。在menufunc函数中改变全局变量，通过该变量完成相关动作。代码如下：

menu = glutCreateMenu(menufunc) # 注册菜单函数menufunc

    glutAddMenuEntry('read book', 1)

    glutAddMenuEntry('close book', 0)

    glutAddMenuEntry('drink water', 2)

    glutAddMenuEntry('add water', 3)

    glutAttachMenu(GLUT\_RIGHT\_BUTTON)

### 键盘

通过glut相关函数完成：glutKeyboardFunc(keydown)

**键盘控制移动**：

1. 视点上下左右前后移动：直接改变视点坐标即可

elif key == b'a' or key == b'A': # a键，视点向左

        Dt = np.cross(EYE\_UP,EYE-LOOK\_AT)

        Dx = Dt/np.linalg.norm(Dt)

        EYE = EYE + Dx

        LOOK\_AT = LOOK\_AT + Dx

        DIST, PHI, THETA = getposture()

        glutPostRedisplay()

1. 物体沿相关轴移动：将参考点反向移动即可

        if key == b'x': # 瞄准参考点 x 减小

            LOOK\_AT[0] -= 0.5

        elif key == b'X': # 瞄准参考 x 增大

            LOOK\_AT[0] += 0.5

        elif key == b'y': # 瞄准参考点 y 减小

            LOOK\_AT[1] -= 0.5

        elif key == b'Y': # 瞄准参考点 y 增大

            LOOK\_AT[1] += 0.5

        elif key == b'z': # 瞄准参考点 z 减小

            LOOK\_AT[2] -= 0.5

        elif key == b'Z': # 瞄准参考点 z 增大

            LOOK\_AT[2] += 0.5

**键盘切换投影格式**：改变全局变量即可

**键盘增删钢笔**：改变全局变量即可

### 改变窗口大小

使用相关函数glutReshapeFunc(reshape)

def reshape(width, height):

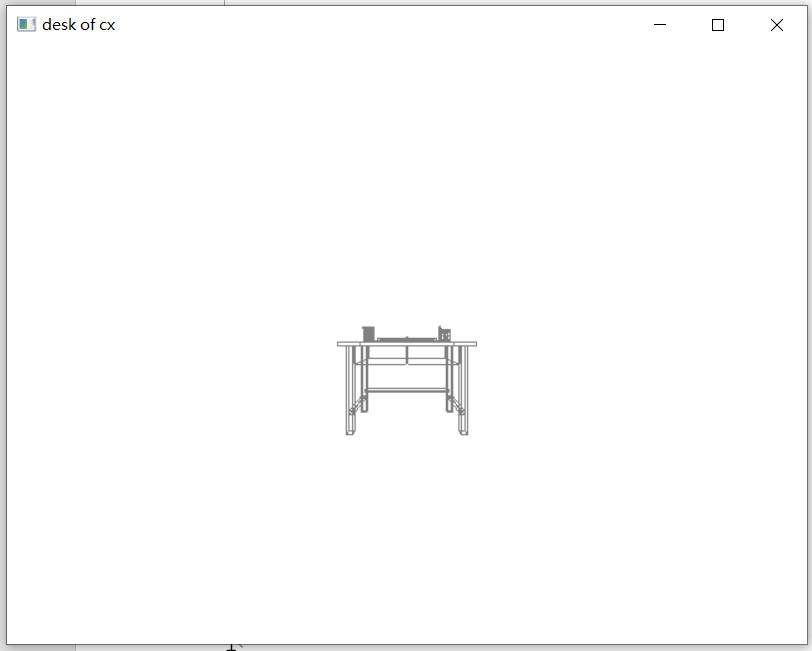
    global WIN\_W, WIN\_H

    WIN\_W, WIN\_H = width, height

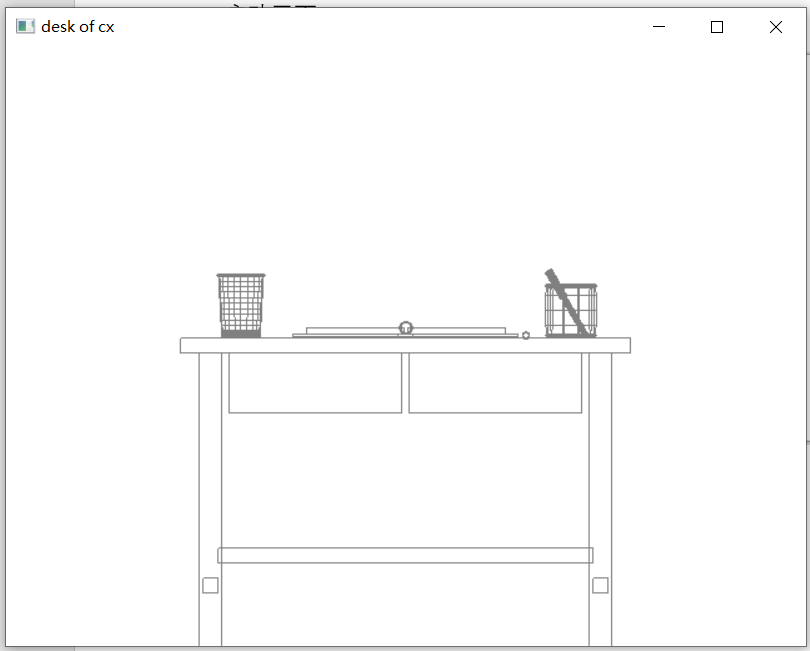
    glutPostRedisplay()

## 实验结果

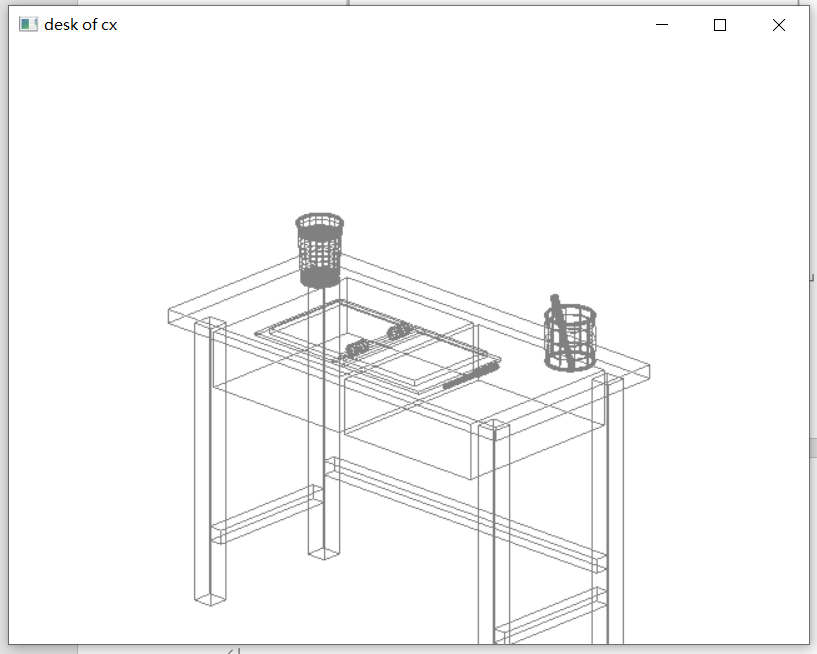
启动画面：

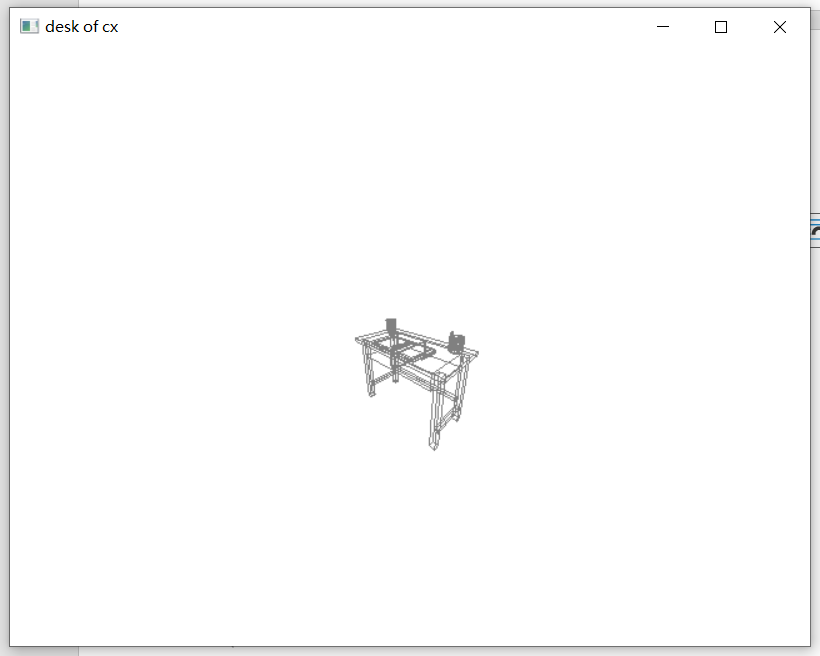


改变投影模式（空格键）：

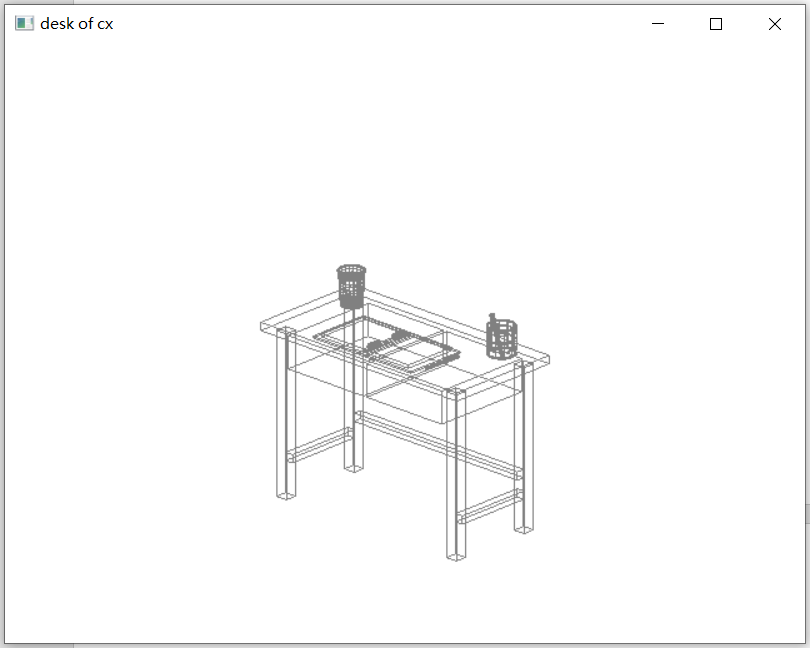


鼠标拖拽旋转：

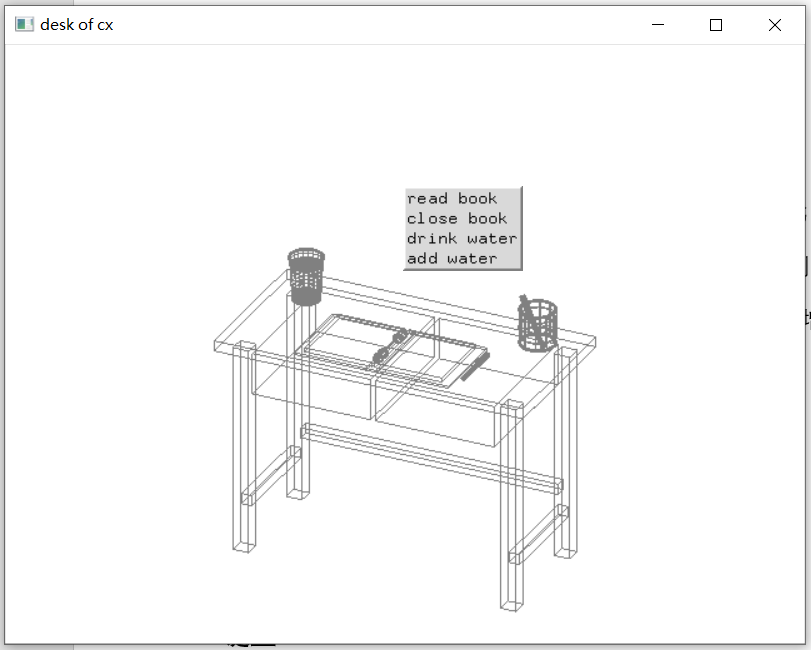


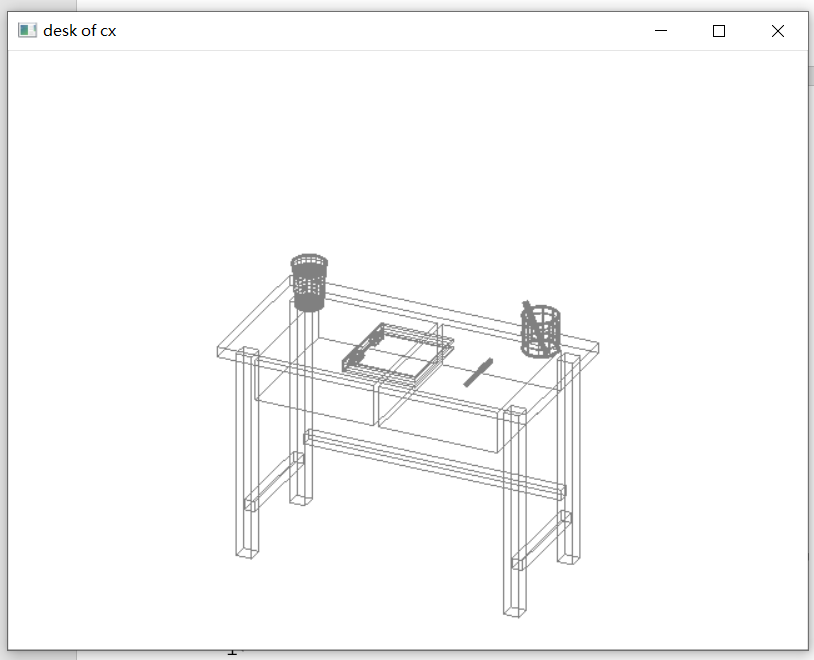


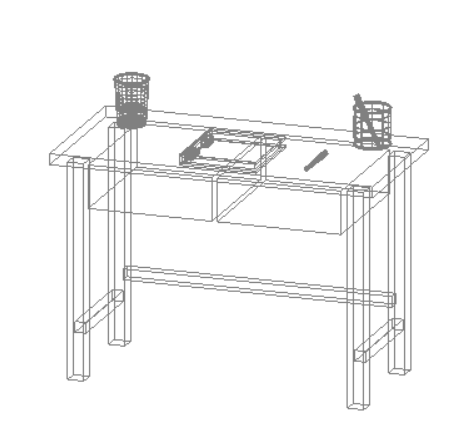
鼠标滚轮缩放（透视投影图像不变）：



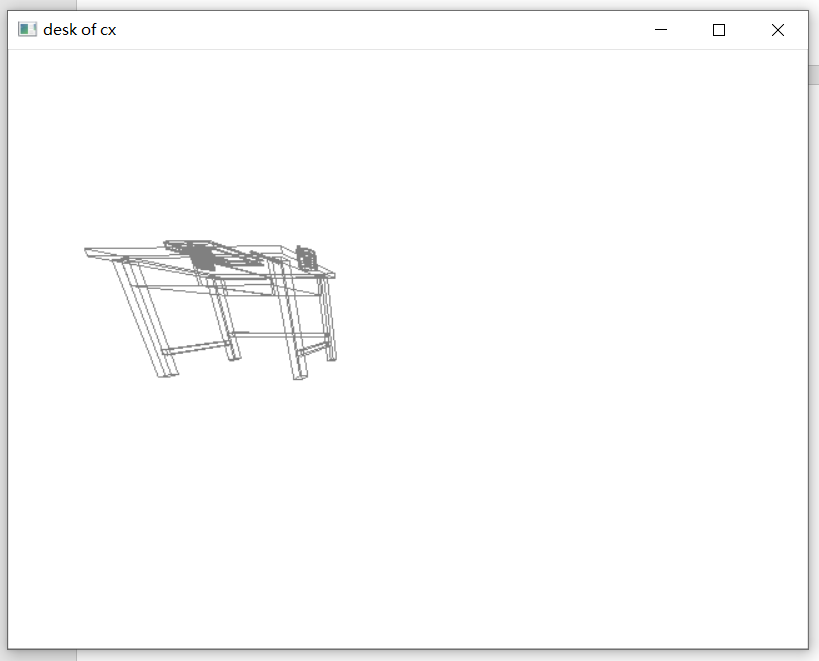
右击进行操作（操作后可以看出书本或水杯水位变化）：

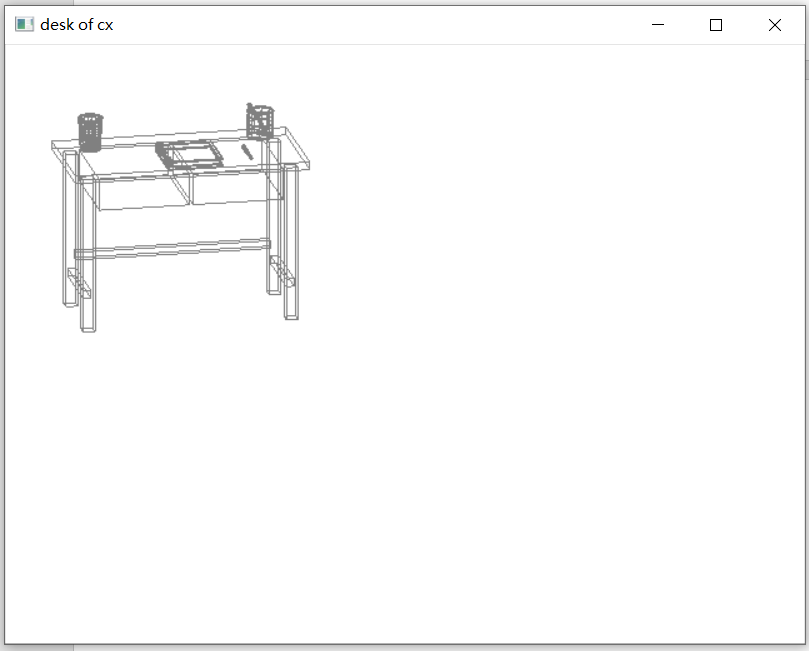




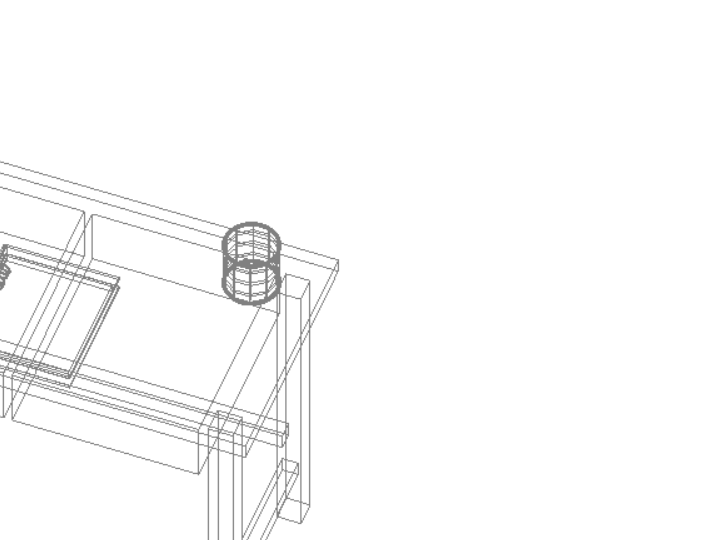


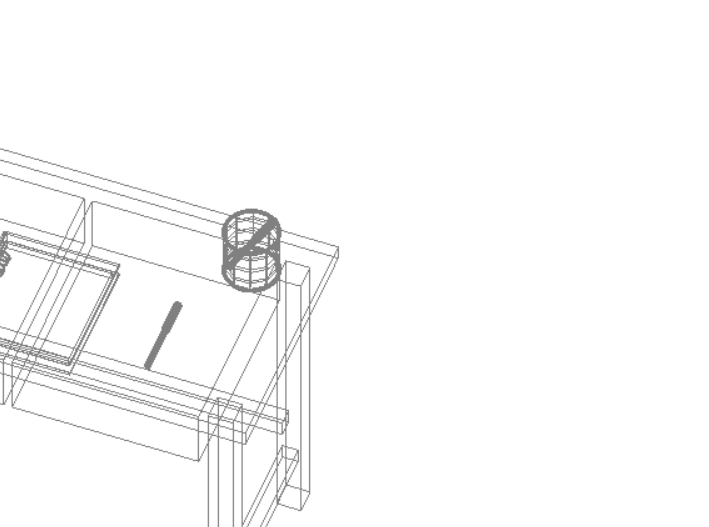
键盘移动位置（前后不改变平行投影图像）（wasdxyz大小写均可）：

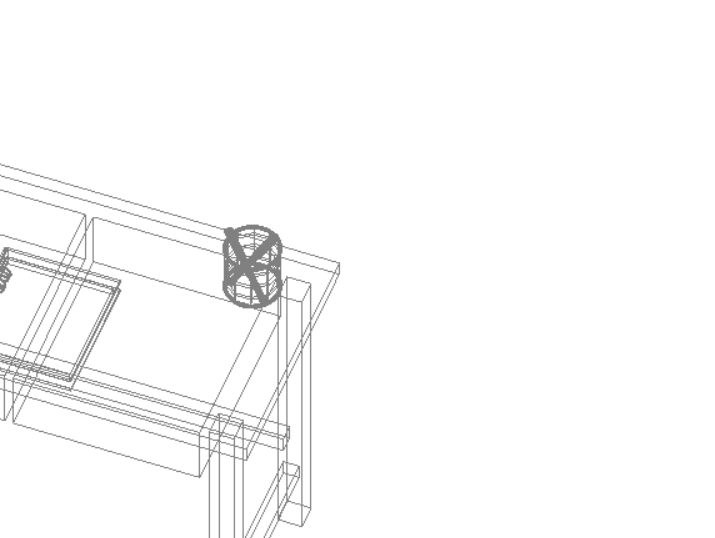




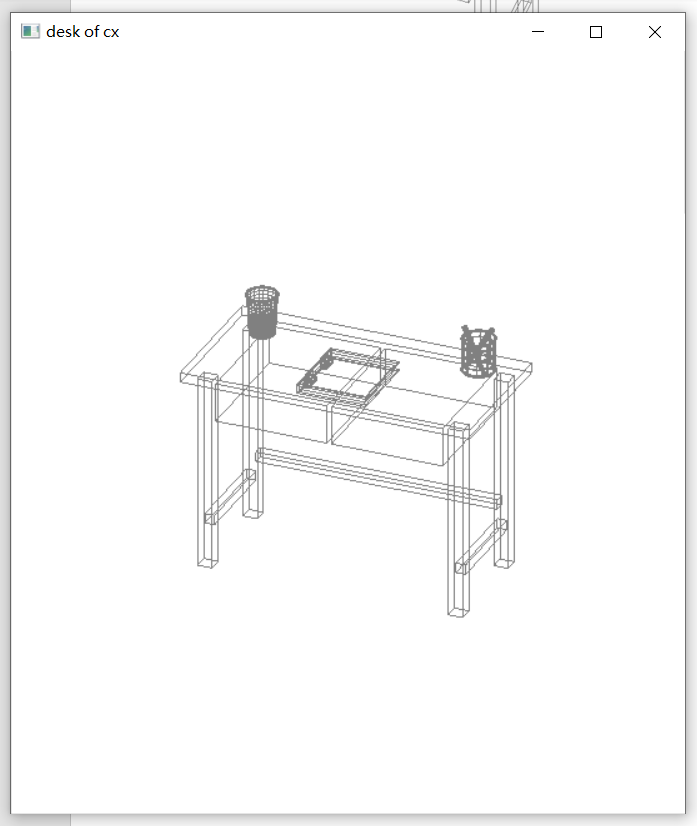
键盘改变钢笔数量（123+键均可）：

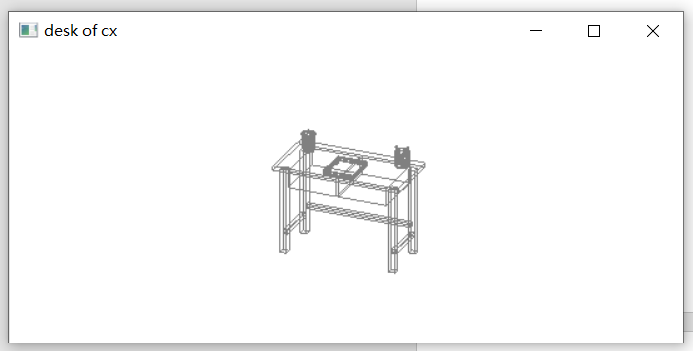






改变窗口大小：





## 实验总结

学习了OpenGL的相关操作，加深了对旋转、平移、缩放矩阵的理解，理解了矩阵栈的操作过程，学习了glut相关函数及其操作。

实验过程中发现长方体没有相关函数，发现可以利用缩放函数进行操作；发现水杯圆台难以操作，发现二次曲面函数；发现液体难以表示，发现利用薄圆柱可以表示水面。

实验参考：https://cloud.tencent.com/developer/article/1546505