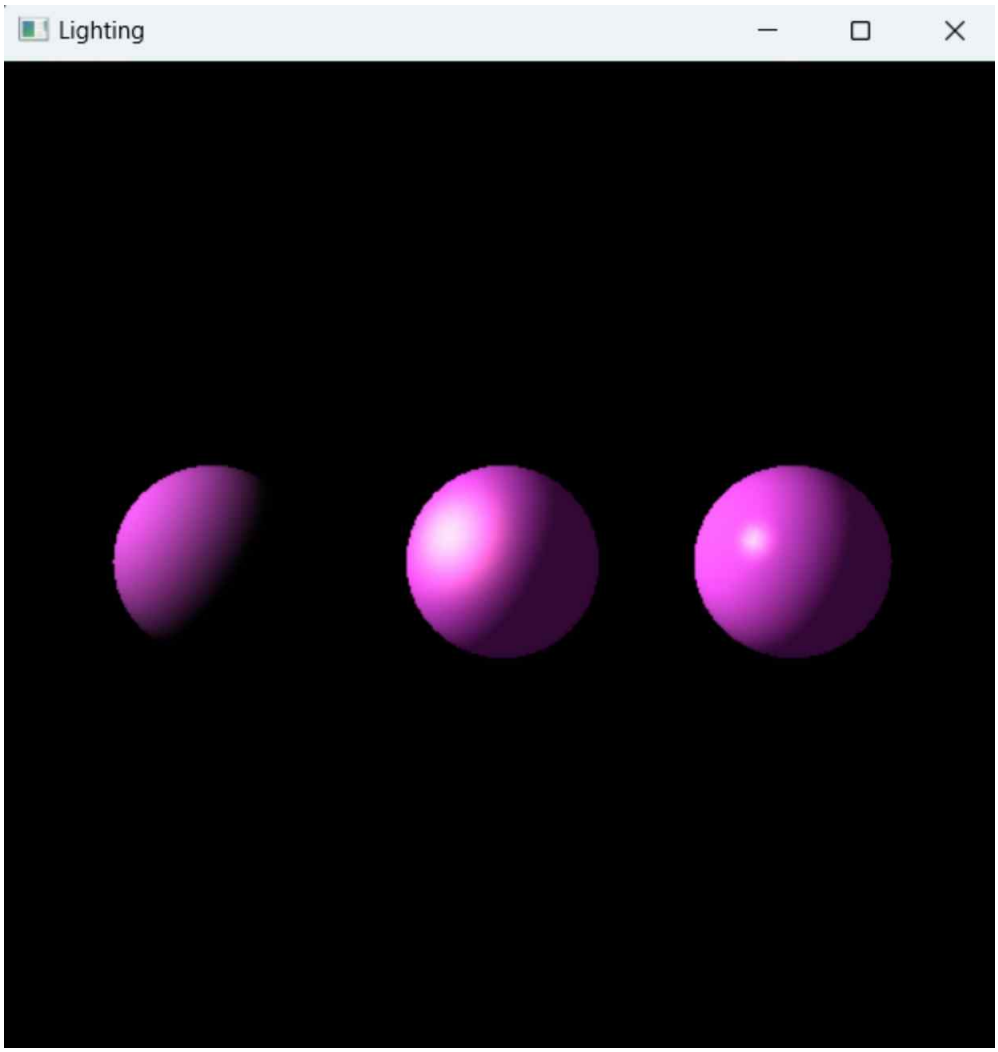
컴퓨터그래픽스 Lab07 보고서

학번	이름	분반
2312282	임다희	003

[과제] 광원의 위치, 물체, 특성을 변화시켜 3개의 원 그리기,

토끼 모델에 광원 적용하기

결과




Bunny Lighting

코드

```
import numpy as np
from OpenGL.GL import *
from OpenGL.GLU import *
from OpenGL.GLUT import *
from ObjLoader import ObjLoader

myview =0

# 광원의 특성(세기)을 나타내는 계수를 정의한다.
light_ambient = (0.2, 0.1, 0.1, 1.0)
# Ia : 광원의 주변광 세기를 조절한다. 주변광의 r,g,b값이 각각 0.2,0.1,0.1이다.
light_diffuse = (1, 0.75, 1, 1.0)
# Id : 광원의 확산광 세기를 조절한다. 확산광의 r,g,b값이 각각 1,0.75,1.0이다.
light_specular = (1.0, 0.75, 0.5, 1.0)
# Is : 광원의 경면광 세기를 조절한다. 경면광의 r,g,b값이 각각 1,0.75,0.5이다.
light_position1 = (-10.0, 5.0, -8.0, 1.0)
# 광원의 위치 첫 번째. x,y,z 값이 각각 -10.0,5.0,-8.0인 지점에 광원이 위치한다.
light_position2 = (-5.0, 10.0, 8.0, 1.0)
# 광원의 위치 두 번째. x,y,z 값이 각각 -5.0, 10.0, 8.0인 지점에 광원이 위치한다.

# 물체의 특성(빛을 반사하는 정도)을 나타내는 계수를 정의한다.
no_mat = (0.0, 0.0, 0.0, 1.0)
# no_mat: 물체가 특정 빛을 반사하지 않는 경우에 계수로 넣어주는 값이다.
mat_ambient = (0.5, 0.1, 0.7, 1.0) #Ka
# Ka: 물체의 주변광 계수를 조절한다. 물체에 의해 반사되는 주변광의 r,g,b값이 각각
0.5,0.1,0.7이다.
mat_diffuse = (1.0, 0.5, 1.0, 1.0) #kd
# Kd : 물체의 확산광 계수를 조절한다.물체에 의해 반사되는 확산광의 r,g,b값이 각각
1.0,0.5,1.0이다.
mat_specular = (1.0, 0.75, 1.0, 1.0) #Ks
# Ks : 물체의 경면광 계수를 조절한다. 물체에 의해 반사되는 경면광의 r,g,b값이 각각 1.0,
0.75,1 이다.

no_shininess = 0.0  # Shininess Coefficient
# 물체의 광택 계수를 조정한다. 광택 계수가 0인 경우
low_shininess = 10.0 # Shininess Coefficient
# 물체의 광택 계수를 조정한다. 광택 계수가 10인 경우
high_shininess = 70 # Shininess Coefficient
# 물체의 광택 계수를 조정한다. 광택 계수가 70인 경우

def flatNormal(v1, v2, v3):

        cross = np.cross(v3-v2,v1-v2)
        length = np.linalg.norm(cross)
        normal = (cross[0]/length, cross[1]/length,cross[2]/length)
        return normal
# 토끼 모델 표현을 위한 노멀벡터.
```

```
def Rabbit():
    global myview
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT)

    glLoadIdentity()

    glLightfv(GL_LIGHT0, GL_AMBIENT, light_ambient)
    glLightfv(GL_LIGHT0, GL_DIFFUSE, light_diffuse)
    glLightfv(GL_LIGHT0, GL_SPECULAR, light_specular)
    glLightfv(GL_LIGHT0, GL_POSITION, light_position2)
# 토끼 모델에 사전에 정의한 계수의 주변광, 확산광, 경면광을 적용하고, 광원의 위치는 두 번째 값 (-5.0, 10.0, 8.0,1.0)으로 지정한다.

    gluLookAt(0.0, 0.0, 13.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0)
# 카메라가 바라보는 위치.

    glScalef(3, 3,3 )

# 토끼 모델에 사전에 정의한 물체의 특성을 입힌다.

  glMaterialfv(GL_FRONT, GL_AMBIENT, mat_ambient)
# 물체의 주변광 반사 계수를 mat_ambient, 즉 (0.5,0.1,0.7,1.0)으로 지정한다.
    glMaterialfv(GL_FRONT, GL_DIFFUSE, mat_diffuse)
# 물체의 확산광 반사 계수를 mat_diffuse, (1.0, 0.5,1.0,1.0)으로 지정한다.
    glMaterialfv(GL_FRONT, GL_SPECULAR, mat_specular)
# 물체의 격면광 반사 계수를 mat_specular, (1.0,0.75,1.0,1.0)으로 지정한다.
    glMaterialf(GL_FRONT, GL_SHININESS, high_shininess)
# 광택 계수를 high_shininess (70.0)으로 지정한다.

# 해당 토끼 모델은 주변광, 확산광, 격면광을 모두 반사한다. 이들은 각각 0.5, 0.1, 0.7 / 1.0, 0.5, 1.0 / 1.0, 0.75, 1.0, 1.0 의 rgb 값을 가지는 빛으로 표현된다. 물체의 광택 계수는 70으로, 표면이 강하게 반짝이는 성질을 가지고 있다.

# 토끼 모델을 로드한다.
    global obj
    obj=ObjLoader()
    obj.load_model("res/bunny.obj")
    index_count = len(obj.vertex_index)
    normal = None
    glBegin(GL_POLYGON)
    for i, vi in enumerate(obj.vertex_index):
        if i % 3 == 0:
            v1 = np.array((obj.model[i * 3]
                           , obj.model[i * 3 + 1]
                           , obj.model[i * 3 + 2]))
            v2 = np.array((obj.model[(i + 1) * 3]
                           , obj.model[(i + 1) * 3 + 1]
                           , obj.model[(i + 1) * 3 + 2]))
            v3 = np.array((obj.model[(i + 2) * 3]
                           , obj.model[(i + 2) * 3 + 1]
```

```
                    , obj.model[(i + 2) * 3 + 2]))

        normal = flatNormal(v1, v2, v3)

        if i > 0:
            glEnd()
            glBegin(GL_POLYGON)

        glNormal3fv((normal[0], normal[1], normal[2]))

        glVertex3fv((obj.model[i * 3]
                , obj.model[i * 3 + 1]
                , obj.model[i * 3 + 2]))

    glEnd()

    glutSwapBuffers()

# 화면에 토끼 모델을 그리는 함수 loadRabbit.
def loadRabbit():
    glutInit(sys.argv)
    glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB | GLUT_DEPTH)
    glutInitWindowSize(500, 500)
    glutCreateWindow('Bunny Lighting')

    glEnable(GL_DEPTH_TEST)
    glEnable(GL_LIGHTING)
    glEnable(GL_LIGHT0)
    glEnable(GL_NORMALIZE)

    glShadeModel(GL_SMOOTH)
    glutReshapeFunc(myReshape)
    glutDisplayFunc(Rabbit)
# 토끼 모델을 그리는 Rabbit을 화면에 나타낸다.

    glutMainLoop()

def MyDisplay():
    global myview
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT)

    glLoadIdentity()

    glLightfv(GL_LIGHT0, GL_AMBIENT, light_ambient)
    glLightfv(GL_LIGHT0, GL_DIFFUSE, light_diffuse)
    glLightfv(GL_LIGHT0, GL_SPECULAR, light_specular)
    glLightfv(GL_LIGHT0, GL_POSITION, light_position1)
# 구체에 사전에 정의한 계수의 주변광, 확산광, 경면광을 적용하고, 광원의 위치는 첫 번째 값 (-
10.0, 5.0,-8.0,1.0)으로 지정한다.

    gluLookAt(0.0, 0.0, 13.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0)
# 카메라가 바라보는 위치.
```

```
    #object1
    glPushMatrix()
    glTranslatef(-3, 0.0, 0.0)

# 첫 번째 구체 object1에 사전에 정의한 물체의 특성을 입힌다.
    glMaterialfv(GL_FRONT, GL_AMBIENT, no_mat)
# 물체의 주변광 반사 계수를 no_mat, 즉 (0,0,0,1)으로 지정한다.
    glMaterialfv(GL_FRONT, GL_DIFFUSE, mat_diffuse)
# 물체의 확산광 반사 계수를 mat_diffuse, (1.0, 0.5,1.0,1.0)으로 지정한다.
    glMaterialfv(GL_FRONT, GL_SPECULAR, no_mat)
# 물체의 격면광 반사 계수를 no_mat, 즉 (0,0,0,1)으로 지정한다.
    glMaterialf(GL_FRONT, GL_SHININESS, no_shininess)
# 광택 계수를 no_shininess (0.0)으로 지정한다.

#첫 번째 구체의 경우 Matrerial에서 주변광, 격면광, 광택 계수가 0으로 물체는 확산광만을 반사하고 이는 rgb값이 각각 1.0, 0.5, 1.0인 빛으로 표현된다.

    glutSolidSphere(1.0, 40, 40)
    glPopMatrix()

    #object2
# 두 번째 구체 object2에 사전에 정의한 물체의 특성을 입힌다.
    glPushMatrix()
    glMaterialfv(GL_FRONT, GL_AMBIENT, mat_ambient)
# 물체의 주변광 반사 계수를 mat_ambient, 즉 (0.5,0.1,0.7,1.0)으로 지정한다.
    glMaterialfv(GL_FRONT, GL_DIFFUSE, mat_diffuse)
# 물체의 확산광 반사 계수를 mat_diffuse, (1.0, 0.5,1.0,1.0)으로 지정한다.
    glMaterialfv(GL_FRONT, GL_SPECULAR, mat_specular)
# 물체의 격면광 반사 계수를 mat_specular, (1.0,0.75,1.0,1.0)으로 지정한다.
    glMaterialf(GL_FRONT, GL_SHININESS, low_shininess)
# 광택 계수를 low_shininess (10.0)으로 지정한다.
    glutSolidSphere(1.0, 40, 40)
    glPopMatrix()

#두 번째 구체의 경우 물체는 주변광, 확산광, 격면광을 모두 반사한다. 이들은 각각 0.5, 0.1, 0.7 / 1.0, 0.5, 1.0 / 1.0, 0.75, 1.0, 1.0 의 rgb 값을 가지는 빛으로 표현된다. 물체의 광택 계수는 10으로, 광택 계수가 0이었던 첫 번째 구체에 비해 표면이 반짝이는 성질을 가지고 있다.

    #object3
# 세 번째 구체 object3에 사전에 정의한 물체의 특성을 입힌다.
    glPushMatrix()
    glTranslatef(3, 0.0, 0.0)
    glMaterialfv(GL_FRONT, GL_AMBIENT, mat_ambient)
# 물체의 주변광 반사 계수를 mat_ambient, 즉 (0.5,0.1,0.7,1.0)으로 지정한다.
    glMaterialfv(GL_FRONT, GL_DIFFUSE, mat_diffuse)
# 물체의 확산광 반사 계수를 mat_diffuse, (1.0, 0.5,1.0,1.0)으로 지정한다.
    glMaterialfv(GL_FRONT, GL_SPECULAR, mat_specular)
# 물체의 격면광 반사 계수를 mat_specular, (1.0,0.75,1.0,1.0)으로 지정한다.
    glMaterialf(GL_FRONT, GL_SHININESS, high_shininess)
```

```
# 광택 계수를 high_shininess (70.0)으로 지정한다.
    glutSolidSphere(1.0, 40, 40)
    glPopMatrix()

#세 번째 구체의 경우 물체는 두 번째 구체와 동일하게 주변광, 확산광, 격면광을 모두 반사한다. 이들은 각각 0.5, 0.1, 0.7 / 1.0, 0.5, 1.0 / 1.0, 0.75, 1.0, 1.0 의 rgb 값을 가지는 빛으로 표현된다. 물체의 광택 계수는 70으로, 광택 계수가 10이었던 두 번째 구체보다 더욱 표면이 반짝이는 성질을 가지고 있다.

    glutSwapBuffers()

def myReshape(w, h):
    glViewport(0, 0, w, h)
    glMatrixMode(GL_PROJECTION)
    glLoadIdentity()
    # glFrustum (left, right, bottom, top, near distance, far distance)
    #if w <= h:
    #glFrustum(-2.0, 2.0, -2.0 * float(h)/ float(w), 2.0* float(h) / float(w), 2.0, 20.0)
    #else:
    glFrustum(-2.0, 2.0, -2.0 * float(w)/ float(h), 2.0* float(w) / float(h), 5.0, 30.0)
    glMatrixMode(GL_MODELVIEW)

# 화면에 구체 3개를 그리는 함수 main.
def main():

    glutInit(sys.argv)
    glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB | GLUT_DEPTH)
    glutInitWindowSize(500, 500)
    glutCreateWindow('Lighting')

    glEnable(GL_DEPTH_TEST)
    glEnable(GL_LIGHTING)
    glEnable(GL_LIGHT0)
    glEnable(GL_NORMALIZE)

    glShadeModel(GL_SMOOTH)
    glutReshapeFunc(myReshape)
    glutDisplayFunc(MyDisplay)
# 구체 3개 모델을 그리는 MyDisplay를 화면에 나타낸다.

    glutMainLoop()

if __name__ == "__main__":
    main()
    #loadRabbit()
# loadRabbit()을 주석처리하면 지정한 광원 및 물체 특성이 적용된 구체 모델 3개가 나타나고, main()을 주석처리하면 지정한 광원 및 물체 특성이 적용된  토끼 모델이 나타난다.
```