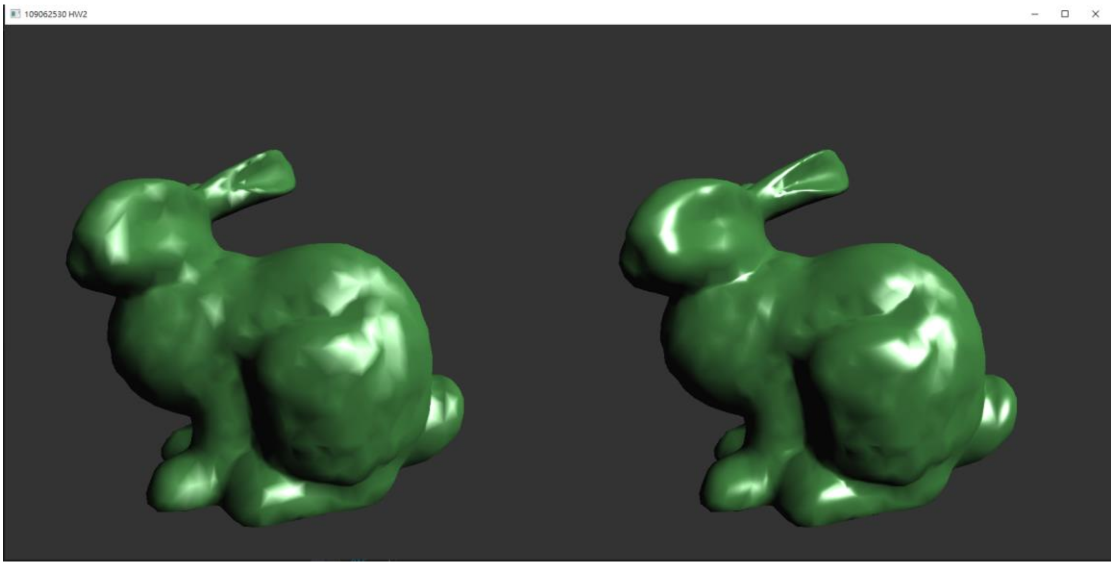
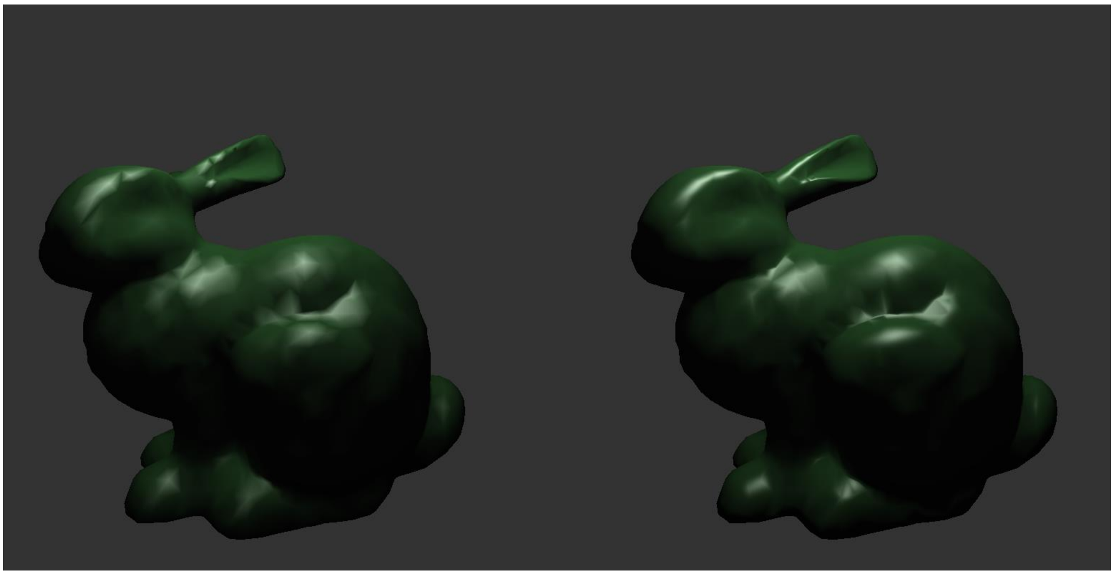
計算機圖學 HW2　109062530

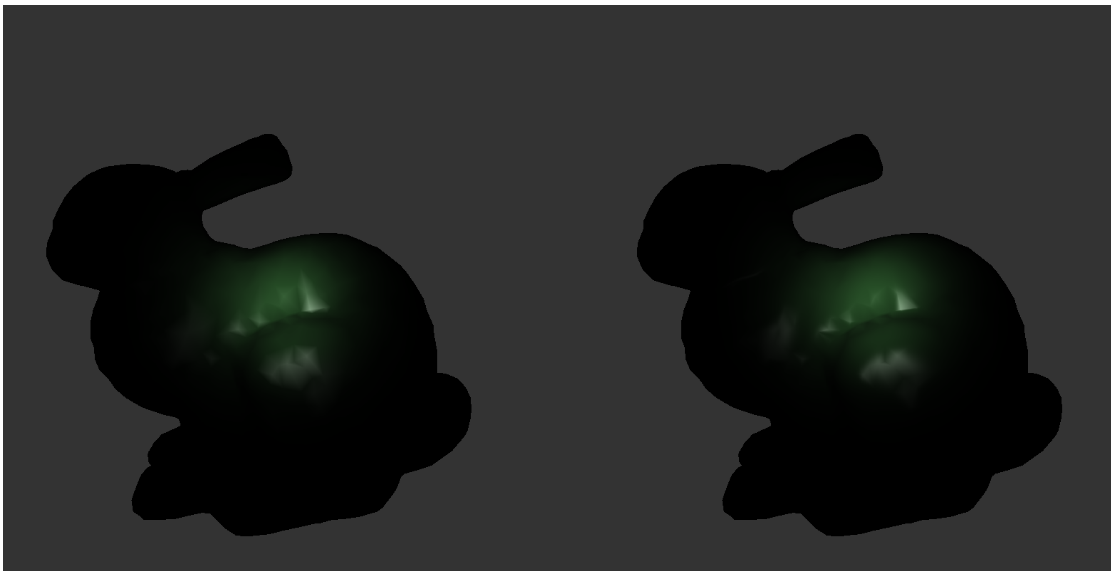
Directionallight:



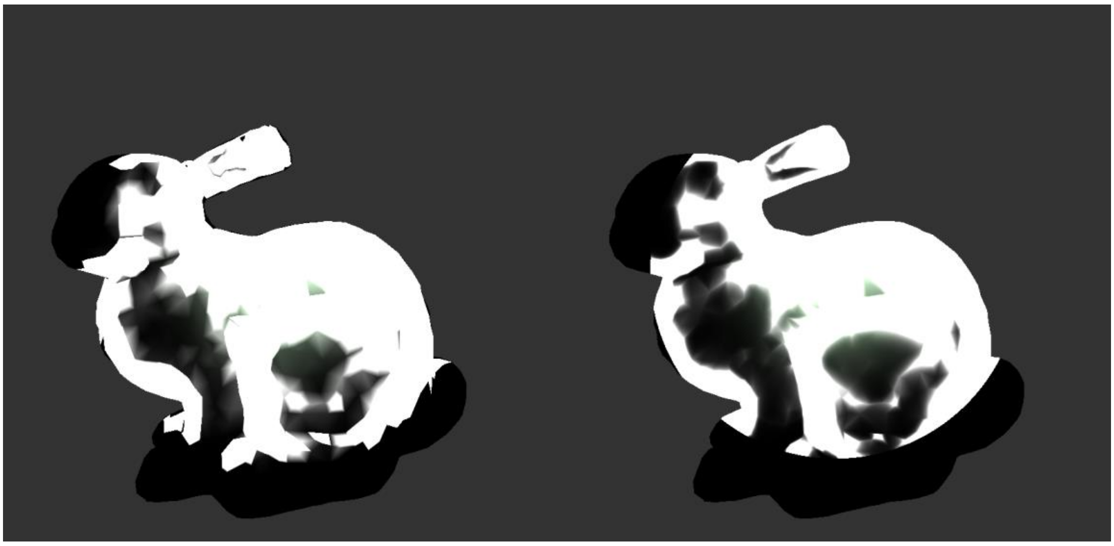
Position(point)light



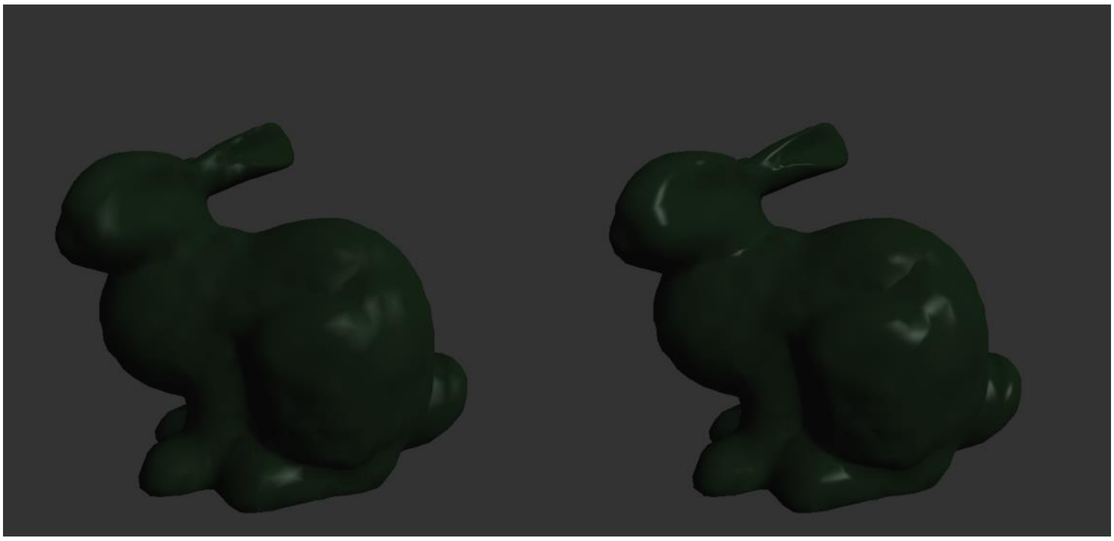
Spotlight



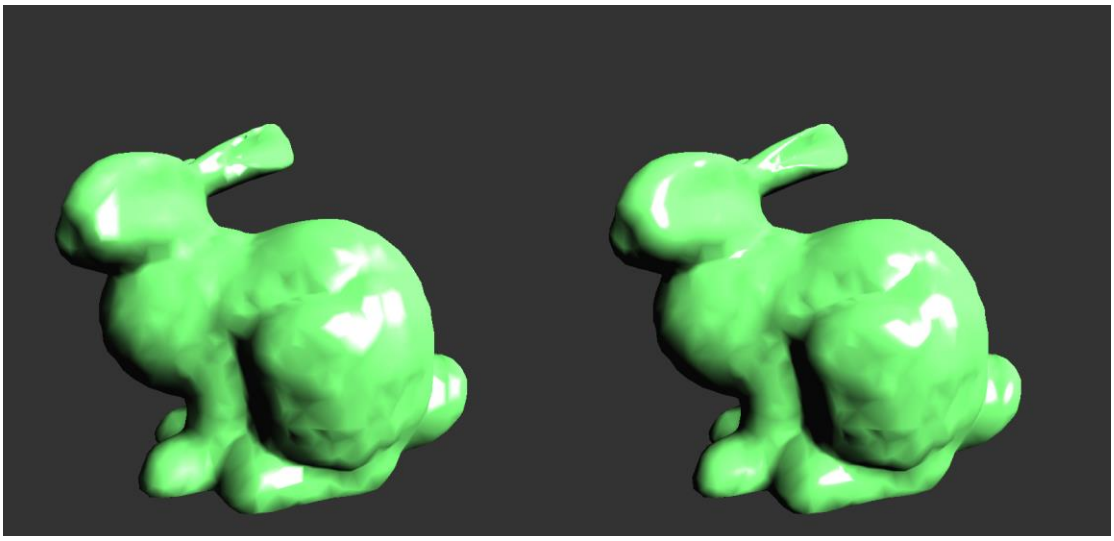
Shininess ctrl:



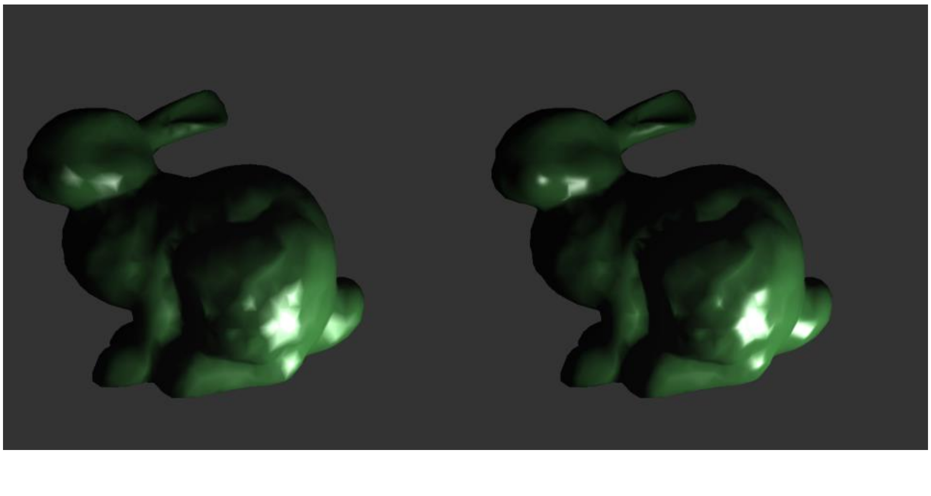
Diffuse intensity low:

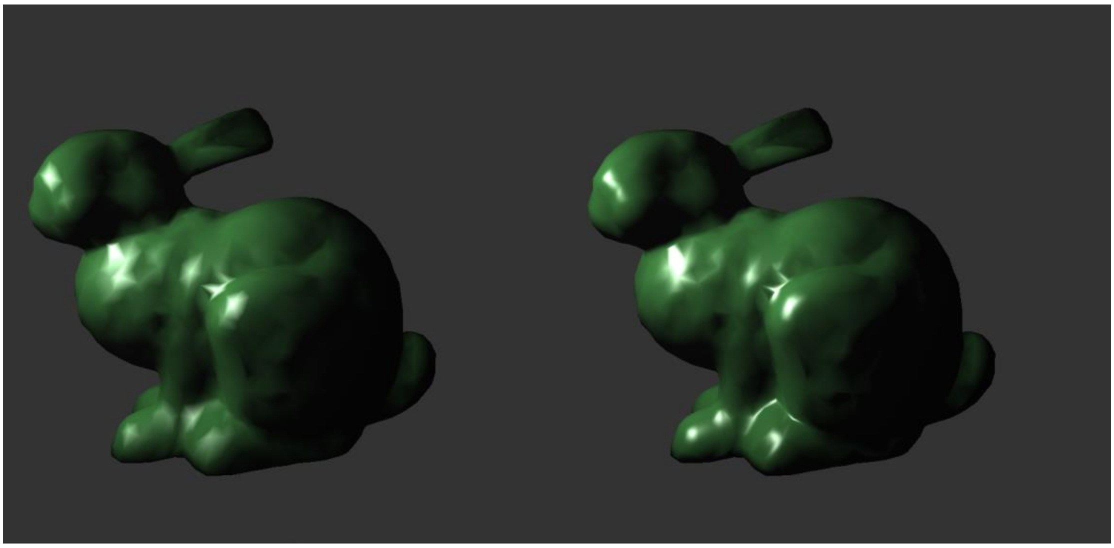


Diffuse intensity high:

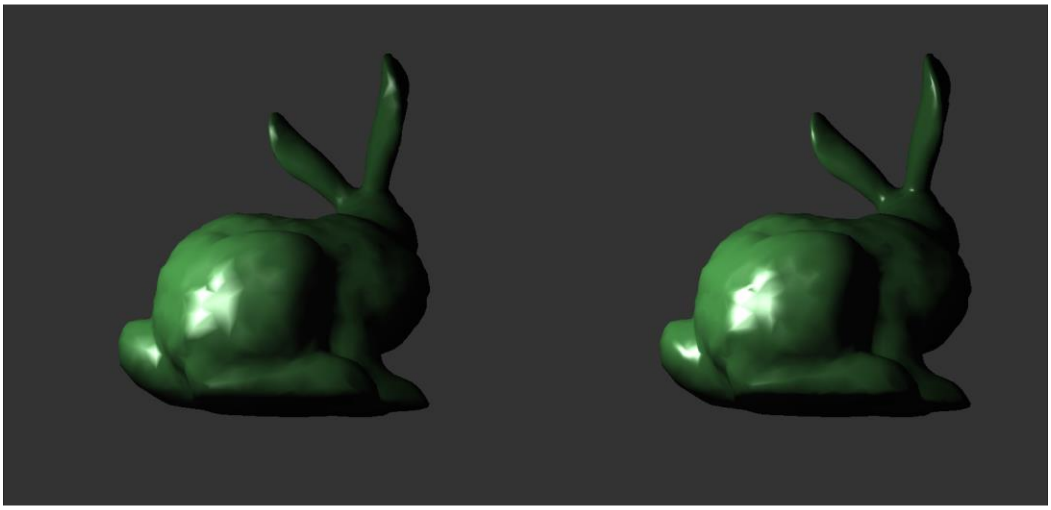


Position(point)light position

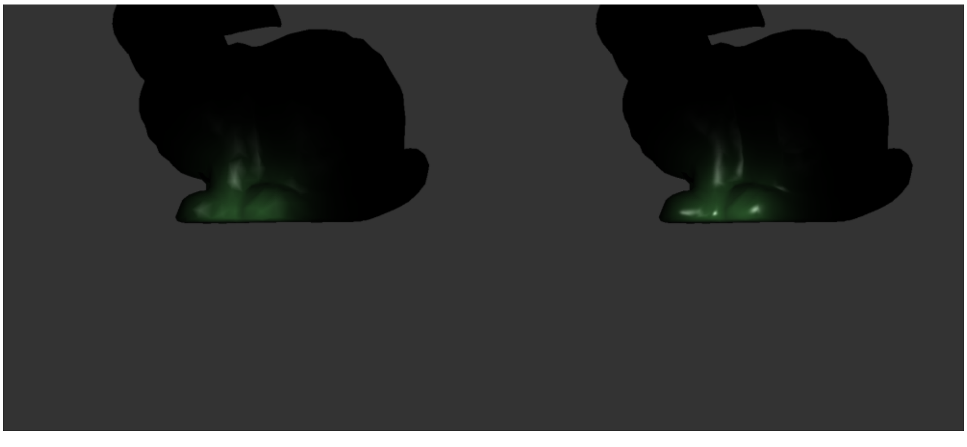




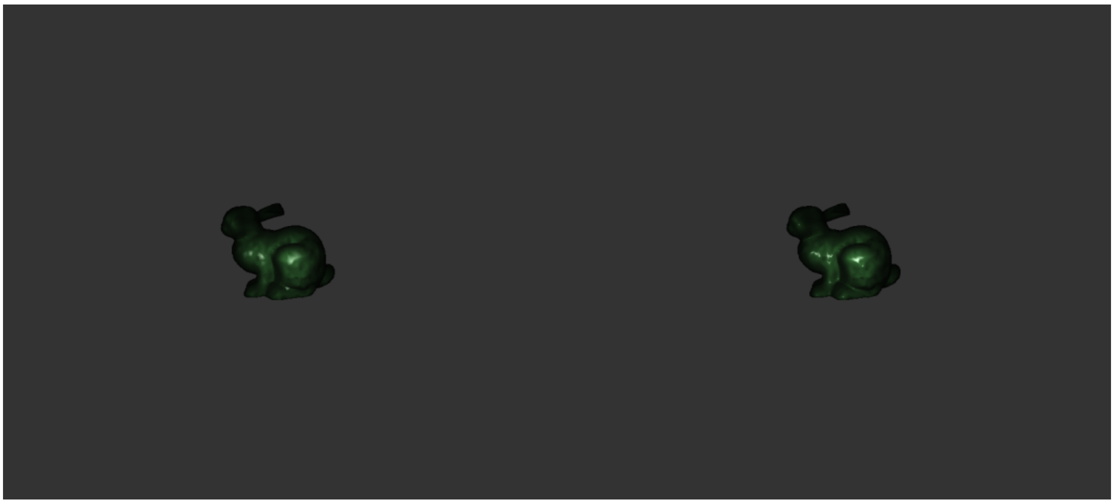
Rotation (point light)



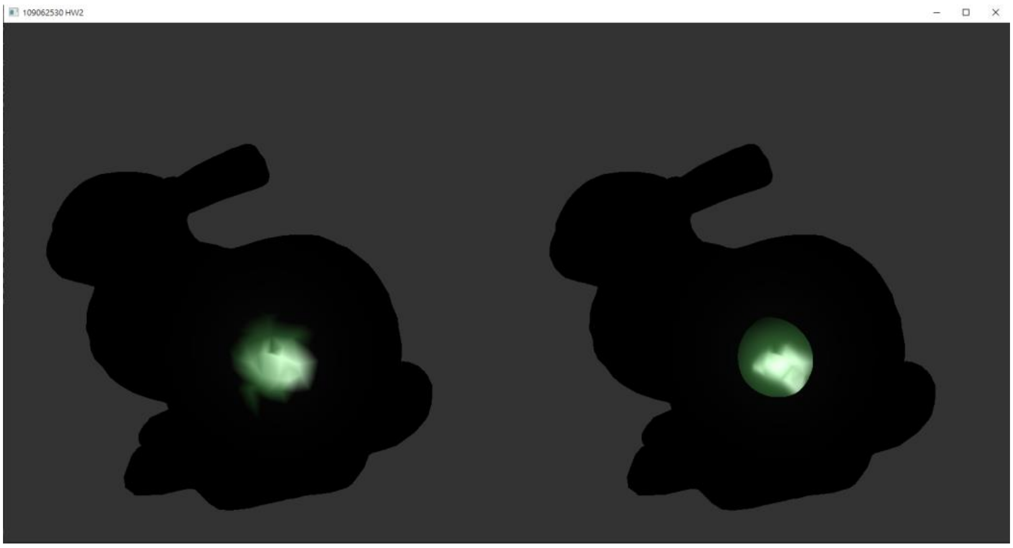
Translation (spotlight)

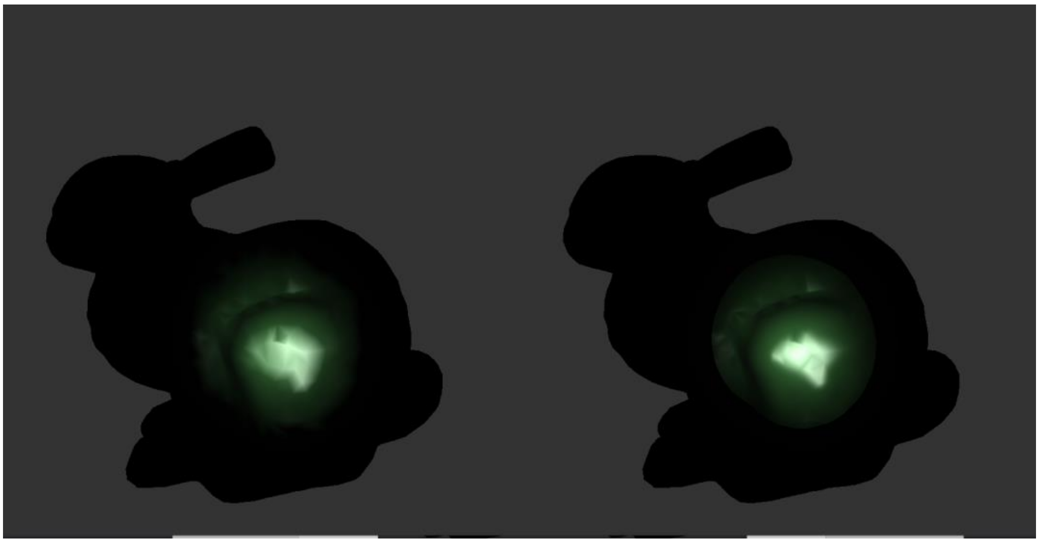


Scale (spotlight)

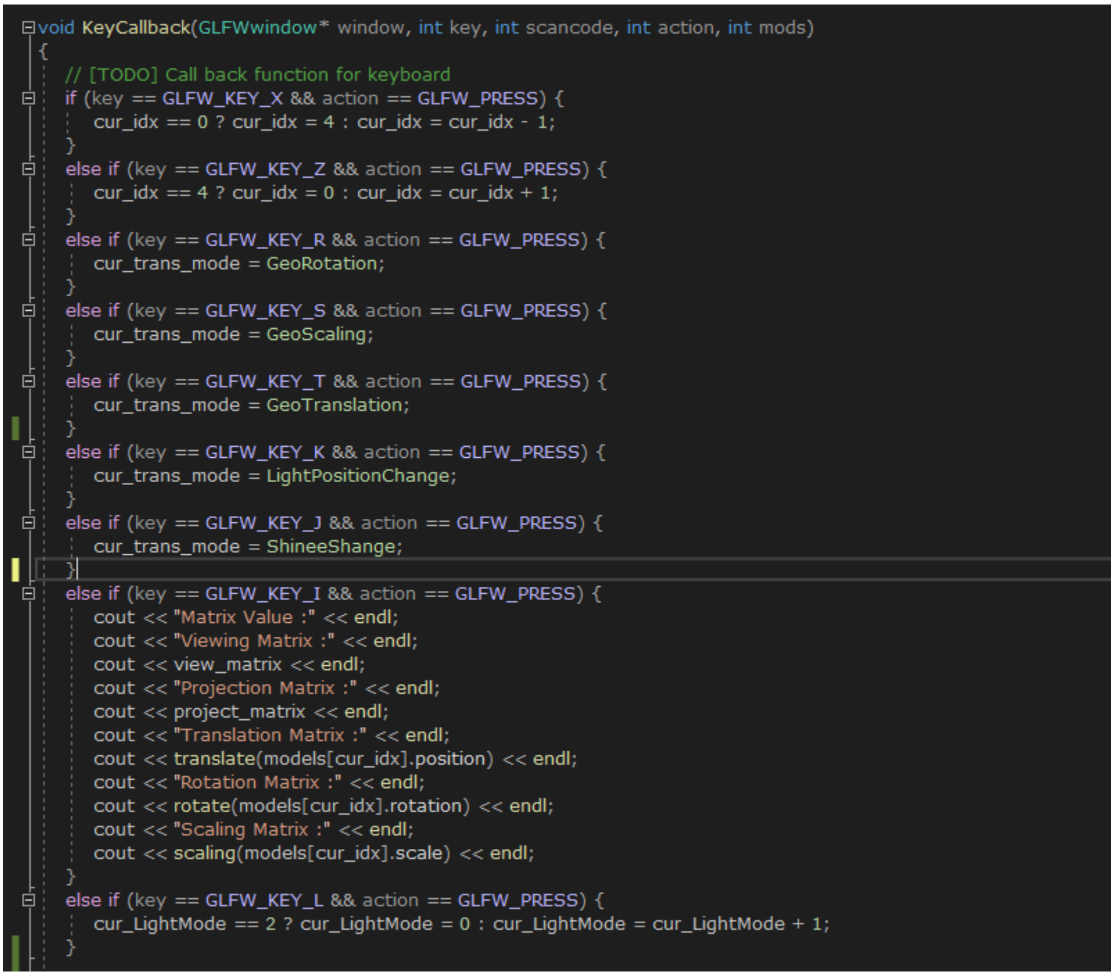


Spotlight Cutoff angle





這次作業的控制，我依照作業說明完成



```
void KeyCallback(GLFWwindow* window, int key, int scancode, int action, int mods)
{
    // [TODO] Call back function for keyboard
    if (key == GLFW_KEY_X && action == GLFW_PRESS) {
        cur_idx == 0 ? cur_idx = 4 : cur_idx = cur_idx - 1;
    }
    else if (key == GLFW_KEY_Z && action == GLFW_PRESS) {
        cur_idx == 4 ? cur_idx = 0 : cur_idx = cur_idx + 1;
    }
    else if (key == GLFW_KEY_R && action == GLFW_PRESS) {
        cur_trans_mode = GeoRotation;
    }
    else if (key == GLFW_KEY_S && action == GLFW_PRESS) {
        cur_trans_mode = GeoScaling;
    }
    else if (key == GLFW_KEY_T && action == GLFW_PRESS) {
        cur_trans_mode = GeoTranslation;
    }
    else if (key == GLFW_KEY_K && action == GLFW_PRESS) {
        cur_trans_mode = LightPositionChange;
    }
    else if (key == GLFW_KEY_J && action == GLFW_PRESS) {
        cur_trans_mode = ShineeShange;
    }
    else if (key == GLFW_KEY_I && action == GLFW_PRESS) {
        cout << "Matrix Value :" << endl;
        cout << "Viewing Matrix :" << endl;
        cout << view_matrix << endl;
        cout << "Projection Matrix :" << endl;
        cout << project_matrix << endl;
        cout << "Translation Matrix :" << endl;
        cout << translate(models[cur_idx].position) << endl;
        cout << "Rotation Matrix :" << endl;
        cout << rotate(models[cur_idx].rotation) << endl;
        cout << "Scaling Matrix :" << endl;
        cout << scaling(models[cur_idx].scale) << endl;
    }
    else if (key == GLFW_KEY_L && action == GLFW_PRESS) {
        cur_LightMode == 2 ? cur_LightMode = 0 : cur_LightMode = cur_LightMode + 1;
    }
```

Z/X：更換模型

R: 旋轉模型

S: 放大縮小

T: 改變模型位置

L: 切換 平面光(Directional)、點光源(point)、探照燈(Spot)

K: 滑鼠移動改變 光源位置、滾輪改變 Diffuse intensity 或 Spot 的 cutoff

J: 改變 Shininess

在 lighting shader 的實作上 主要參考 OPENGL 的教學網站:
https://learnopengl.com/

並且有實現 view space 觀察模式 燈光與物體不會黏在一起