ICG HW3

410785018 資工四 邱信瑋

一、 程式說明

- **TriangleMesh**
  1. 在此完成讀檔部分：首先先打開 Obj 檔讀取，直到讀到 mtllib 後，則轉向先去讀取 Mtl 裡的檔案內容並透過 SubMesh 做存取(這裡相較 Hw2 只是增加了 if else 的判斷式，增加判斷 map_Kd)，完成後，再回到 Obj 檔中繼續完成剩餘的存取，這邊值得注意的是在某些檔案中並不會按照順序 Vp,Vn,Vt,f，有些檔案會穿插因此在存取上需要多加注意。
  2. 透過 CreateBuffers()和 Draw()函式完成繪圖：讀檔完成後，使用 CreateBuffers()函式將所有出現過的 Vertex 丟進 Vertex Buffer；透過 SubMesh，在不同材質中存取各自的 Vertex Index 並且丟進各自的 Index Buffer。Draw()函式則是將 Buffer 裡 Vertex 的內容，傳入 GPU 裡處理。
  3. 在 VertexPTN 我有稍微修改一下存入 Buffer 的順序，原先老師給的是依序是 PNT，而我將其改成 PTN(因為寫程式當下比較好記，腦袋不用一直轉 QAQ)
  4. 在讀檔的時候我有發現好像是瓦斯彈還有車子的 mtl 檔，他們的檔尾是在最後一行的文字；另外四個的 mtl 檔，其檔尾是在最後一行的下一行，導致我 De 這隻蟲有夠久，哭阿 QQ。
- **Texture Map**
  1. 實作細節：原本 Hw2 中，Ka、Kd、Ks、Ns 的值是固定的，然而這次作業中 Kd 是會變動的，變動的值是依據在 shader 中透過 texture mapping 的方式找出 vertex 各自對應的 Kd 值，然而有某些 vertex 並不會使用到 texture 這點需要特別注意，針對這點，我有另外設定一個判斷有沒有使用 texture 的 flag，並且傳至 shader，給其判斷是否要透過 texture mapping 找尋 Kd 或者直接使用 mtl 檔案上的 Kd 就好，如下:

```
// check whether using texture or not.
vec3 new_Kd = Kd;
if(textureFile_Flag > 0.0){
    new_Kd = texture2D(mapKd, iTexCoord).rgb;
}
```

  2. 計算光所使用到的公式：ambient = Ka * ambientLight; diffuse = Kd * I * max(0, dot(N, lightDir)); Specular = Ks * I * pow(max(0, dot(viewDir, lightReflectDir)), Ns);
- **Other**

1. 實作動態載入檔案(Model and SkyBox)

```
void CreatePopUpMenu()
{
    int menu = glutCreateMenu(ProcessMenuEvents);
    glutAddMenuEntry("Load 3D Model", LOAD_MODEL_EVENT);
    glutAddMenuEntry("Load 3D SkyBox", LOAD_SKYBOX_EVENT);
    glutAttachMenu(GLUT_RIGHT_BUTTON);
}
// Callback function for glutCreateMenu.
void ProcessMenuEvents(int option)
{
    switch (option) {
    case LOAD_MODEL_EVENT: {
        string filePath = GetOpenModelFilePath();
        LoadObjects(filePath);
        break;
    }
    case LOAD_SKYBOX_EVENT:
        string filePath = GetOpenSkyBoxFilePath();
        CreateSkybox(filePath);
        break;
    }

}

string GetOpenModelFilePath() { ... }

string GetOpenSkyBoxFilePath() { ... }
```

2. 實作 Visualize 光源 Z 軸方向的移動

```
void MoveLeft (const float moveSpeed) { position += moveSpeed * glm::vec3(-0.1f,  0.0f, 0.0f); }
void MoveRight(const float moveSpeed) { position += moveSpeed * glm::vec3( 0.1f,  0.0f, 0.0f); }
void MoveUp   (const float moveSpeed) { position += moveSpeed * glm::vec3( 0.0f,  0.1f, 0.0f); }
void MoveDown (const float moveSpeed) { position += moveSpeed * glm::vec3( 0.0f, -0.1f, 0.0f); }
void MoveForward(const float moveSpeed) { position += moveSpeed * glm::vec3(0.0f, 0.0f, 0.1f); }
void MoveBackward(const float moveSpeed) { position += moveSpeed * glm::vec3(0.0f, 0.0f, -0.1f); }
```

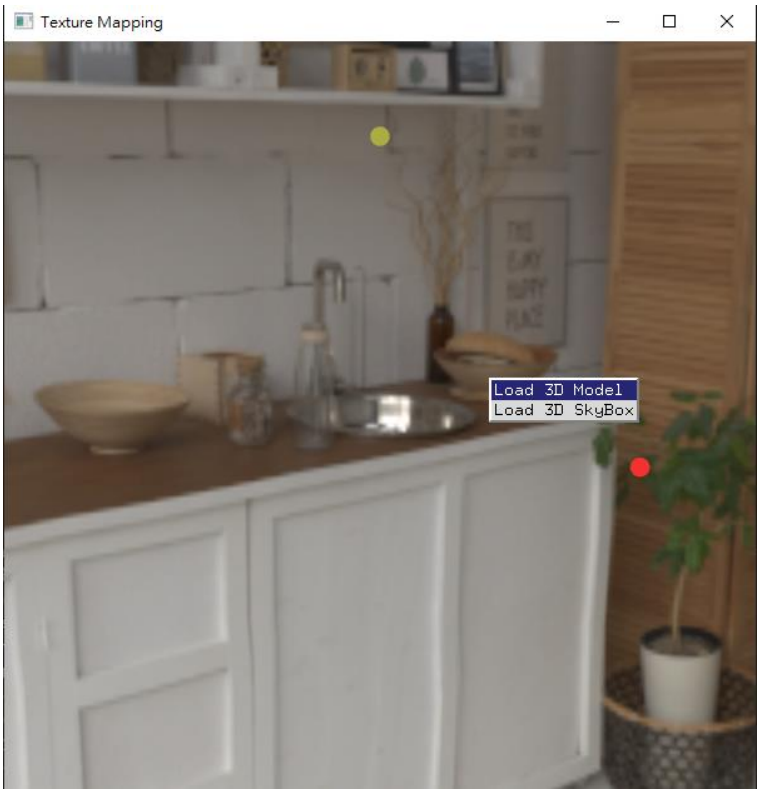
```
if (spotLight != nullptr) {
    if (key == 'a')
        spotLight->MoveLeft(lightMoveSpeed);
    if (key == 'd')
        spotLight->MoveRight(lightMoveSpeed);
    if (key == 'w')
        spotLight->MoveUp(lightMoveSpeed);
    if (key == 's')
        spotLight->MoveDown(lightMoveSpeed);
    if (key == 'z')
        spotLight->MoveForward(lightMoveSpeed);
    if (key == 'x')
        spotLight->MoveBackward(lightMoveSpeed);
}
```

3. 實作 SkyBox 的旋轉

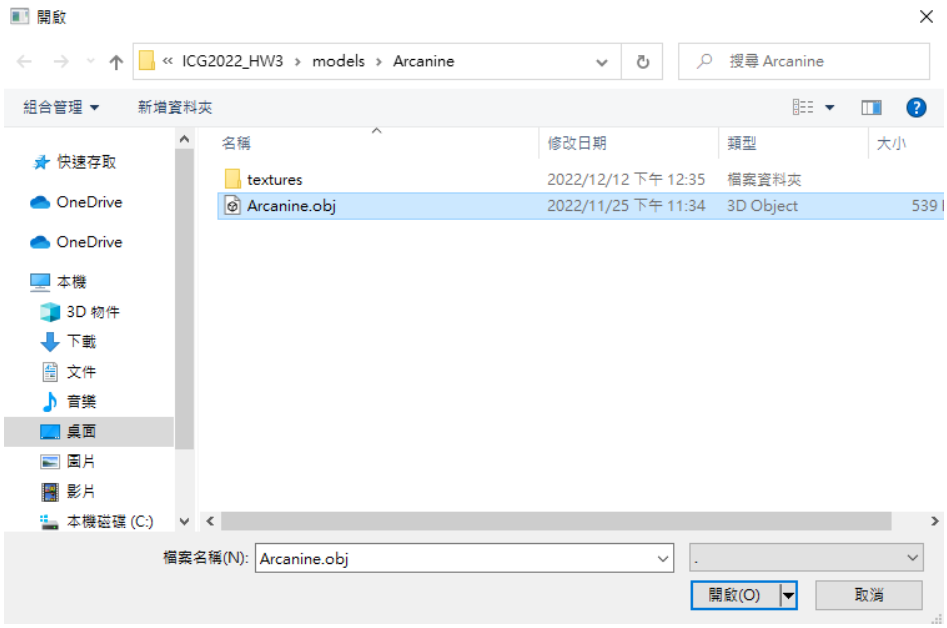
```
// TODO: modify code here to rotate the skybox.
glm::mat4x4 S = glm::scale(glm::mat4x4(1.0f), glm::vec3(1.5f, 1.5f, 1.5f));
glm::mat4x4 R = glm::rotate(glm::mat4x4(1.0f), glm::radians(GetRotation()), glm::vec3(0, 1, 0));
glm::mat4x4 MVP = camera->GetProjMatrix() * camera->GetViewMatrix() * S * R;
```

```
// Render skybox. ------------------------------------------------------------------------------------------
if (skybox != nullptr) {
    // ------------------------------------------------------------
    // Add your code to rotate the skybox.
    // ------------------------------------------------------------
    skybox->SetRotation(curSkyObjRotationY += rotStep_sky);
    skybox->Render(camera, skyboxShader);
}
```

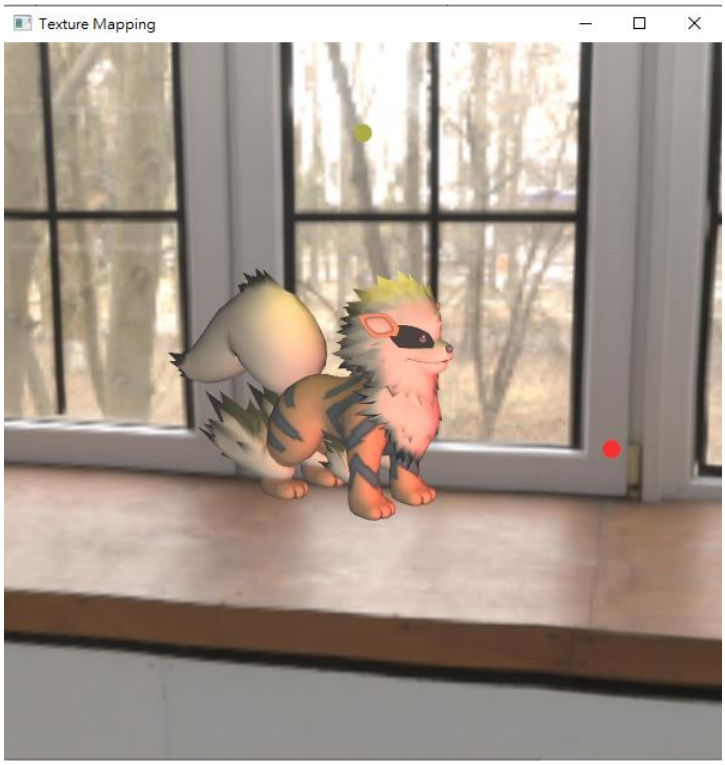
二、 Demo 結果



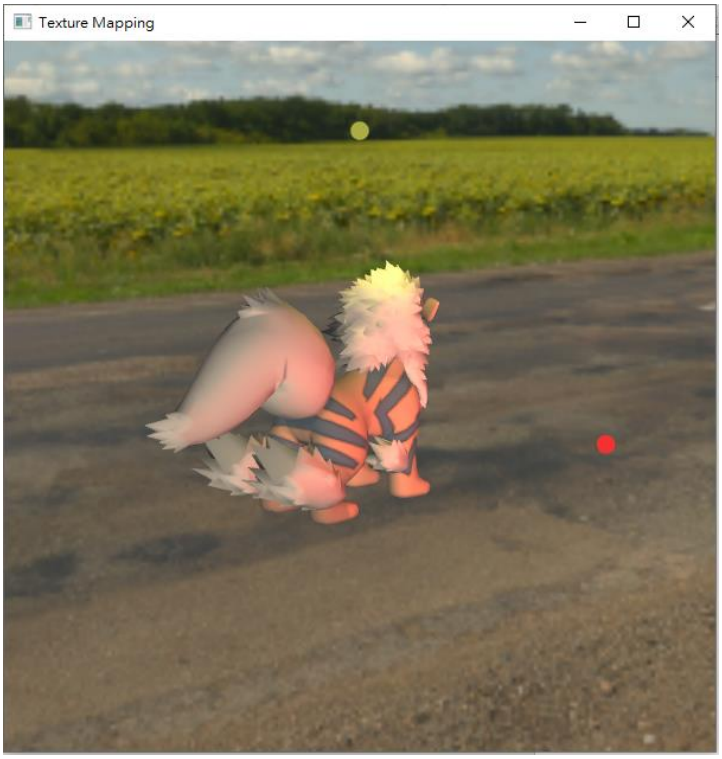
圖一 初始狀態(可選擇響載入的 Model 或 SkyBox)



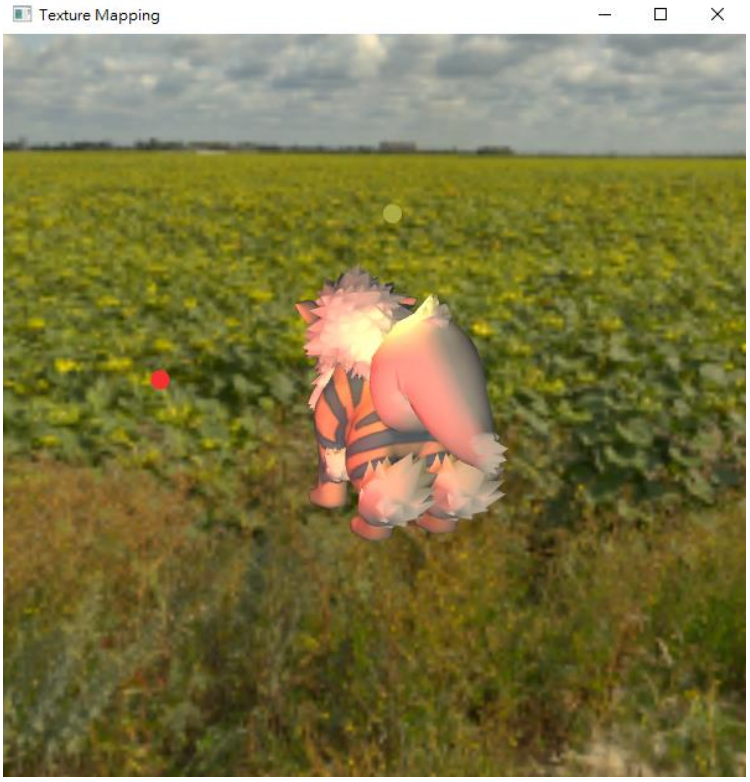
圖二 選擇載入模型



圖三 載入模型



圖四 更換場景



圖五 移動光源