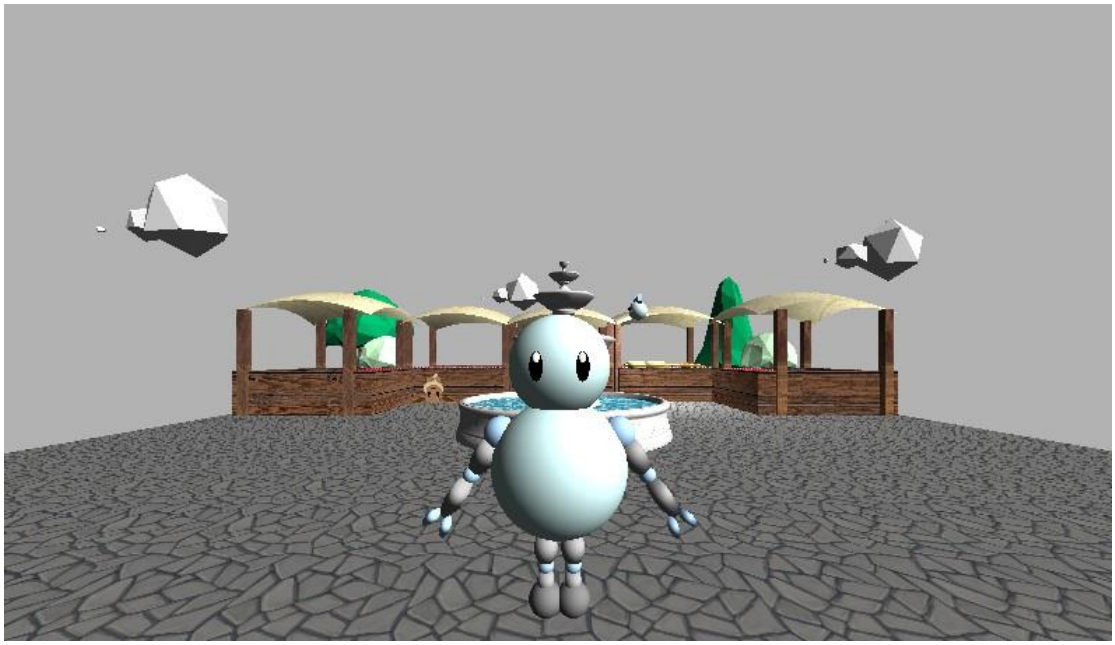
2023Computer Game Engine, Project #1, the shaders

一、 介紹:

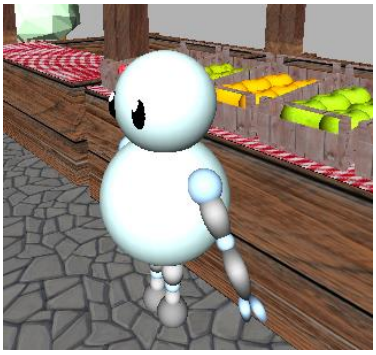
這次作業的內容是要自己寫 shader 並且創作場景，我選擇小攤販市場作為我的主題，其實我原本的想法是中世紀貿易市場(就是異世界動漫常常出現的那種市場)，但是還不太擅長建模，所以先使用簡單的小攤販當作這學期的主題。



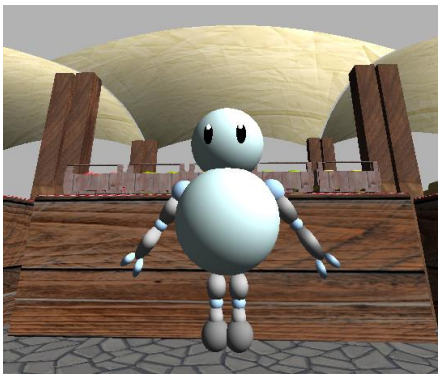
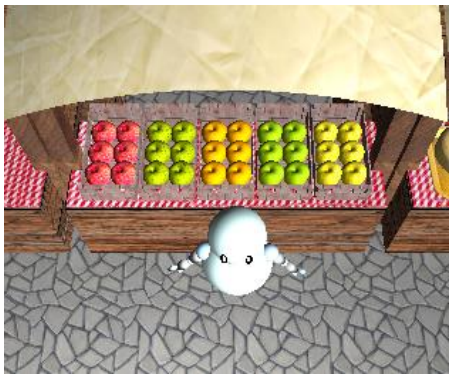
二、 使用方法:

- 機器人走路: 機器人的前進方向比較特別，我是根據目前的視野去做方向的判斷(類似 rpg 遊戲)，連按 2 下會切換成跑步。
  - W/w -> 遠離使用者
  - A/a -> 往左方
  - S/s -> 靠近使用者
  - D/d -> 往右
  - R/r -> 原地轉圈圈
  - 空白鍵 -> 跳躍
- 視野:
  - 按著右鍵且往右滑 -> 視野向右
  - 按著右鍵且往左滑 -> 視野向左
  - 按著右鍵且往上滑 -> 轉為由上往下看
  - 按著右鍵且往下滑 -> 轉為由下往上看
  - 滾輪往上 -> zoom in

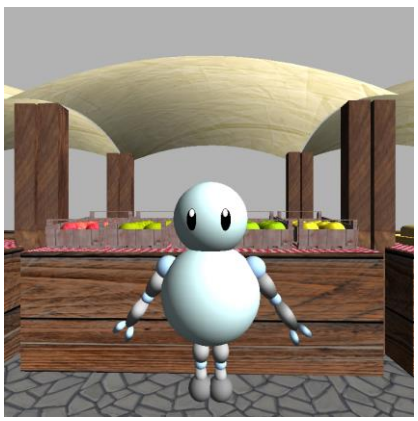
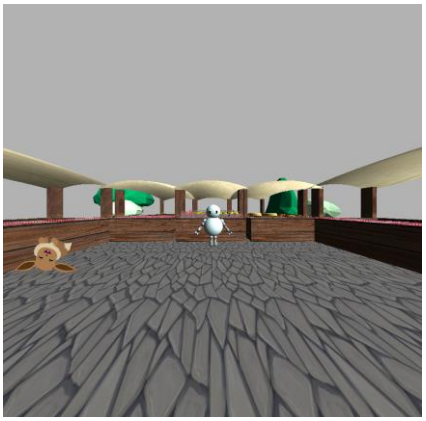
- 滾輪往下 -> zoom out
- Ctrl + back -> 初始化為預設視野

▲ 視野向右/視野向左

▲ 由下往上看/由上往下看

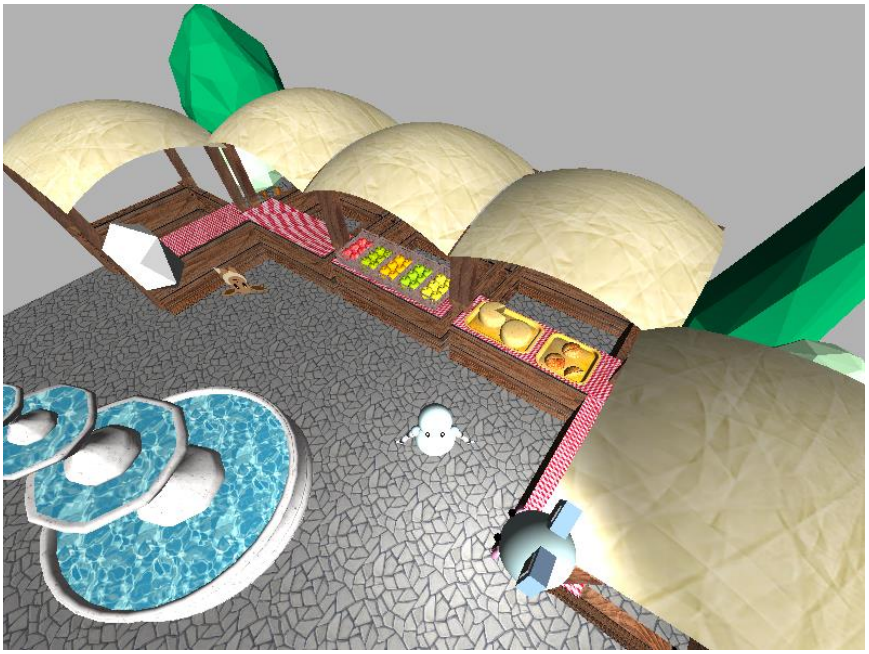
▲ (皆為整個 view port 截圖)zoom in/zoom out

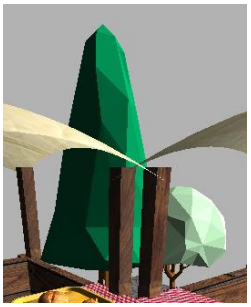
- 燈光開關: 按下 0 可以對平行光做開關

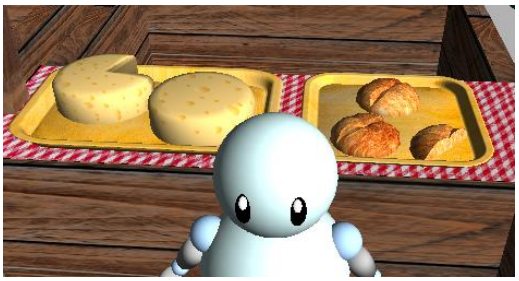

▲關平行光/在關平行光下被 spot light 照到

三、 實作要點 & 程式內容:

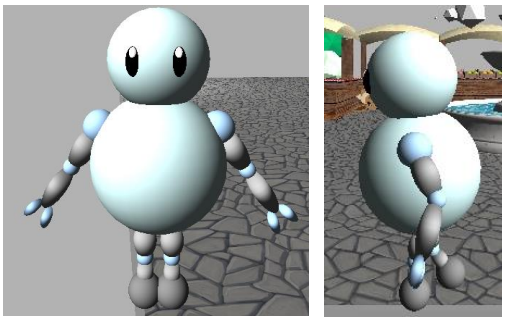
- 場景:
  - 場景內有五個攤販(水果 & 麵包)、四棵樹、噴水池、三朵雲、spot light 小精靈還有一隻伊布(他是動態 billboard，由 55 張圖片組成)。







  - 小精靈 spot light，會在天上移動，碰到障礙物會隨機朝某個方向轉彎。
  - 平行光是沿著(1,1,1)的方向射來。
  - 機器人不得超過邊界，也不得穿越物體(有做障礙物偵測效果)
  - 幾乎都有做 texture mapping(攤販、水果、麵包和噴水池)

- 機器人:
  - 機器人是之前圖學的作業，他在做運動時(跑走跳)，關節都會一起動。 
  - 比較不一樣的是，我之前圖學都是把東西全部放到 main.cpp 裡，但這次我嘗試拆成很多檔案，有物件導向的感覺(?不然之前寫到最後好痛苦，一個 main 有 4000 多行，變數名稱到最後都改不定了。

```
⊞class foot { ... };
⊞class hand { ... };
⊞class robot { ... };
```

◂ 寫成 class

- View & projection:
  - 視野的部分我是參考一款叫托蘭異世錄的視野移動方式。他的往前是往場景的前方，然後可以用滑鼠配合 W/A/S/D 去做同步的移動，不需要先調到好位置在移動。

  - 實作的原理是，攝影機固定看著機器人的 xz 座標，攝影機本身是在機器人**後方 30 單位**處。因為畫面轉動後，機器人的**前方就會改變**，所以抓攝影機位置時必須*上旋轉矩陣。底下的 eye 是攝影機位置，pos 是機器人位置，eyeAngY 是畫面旋轉的角度。

```
eye[0] = pos[0] + eyeDis * cos(eyeAngy * PI / 180.0);
eye[2] = pos[2] + eyeDis * sin(eyeAngy * PI / 180.0);
```

  - 有了這些資訊，就可以透過**機器人位置 - 眼睛位置**去算往前方的向量，再透過往前方的向量外積(0,1,0)得到左方和右方的向量。
  - 機器人移動時，就根據這 4 個向量的單位向量*移動篇幅就可以了!

```
float w[3] = { pos[0] - eye[0] ,pos[1] - eye[1], pos[2] - eye[2] };
float s[3] = { -pos[0] + eye[0] , -pos[1] + eye[1], -pos[2] + eye[2] };
float d[3] = { -w[2] , 0 , w[0] };
float a[3] = { w[2] , 0 , -w[0] };
```

▲w/a/s/d 方向向量的算法(d/a 是 w 和(0,1,0)做外積)

```
tpPos[0] += myRobot->moveOffset * s[0];
tpPos[2] += myRobot->moveOffset * s[2];
```

▲移動的時候，加上位移量*方向向量就行了

- 至於方向移動鏡頭的部分，我是判斷滑鼠移動的方向，去決定要往左/右/上/下，xy 為當前滑鼠的位置，mouseXY 是之前滑鼠的位置。每次移動都去改 eyeAngY 的角度，或是眼睛位置的高低。

```
if (mouseBtn == GLUT_RIGHT_BUTTON) {
    if (x > mouseX) eyeAngy += 1.5;
    else eyeAngy -= 1.5;

    if (eyeAngy >= 360) eyeAngy -= 360;
    if (eyeAngy <= 0) eyeAngy += 360;

    if (y > mouseY)  eye[1] = fmin(eye[1] + 0.5, 80.0);
    else eye[1] = fmax(eye[1] - 0.5, 10.0);
}
```

- shader:
  - 在寫 shader 架構時，我想了很久，因為每次畫東西，都需傳入點座標，所以我把它寫成一個叫 mesh.cpp 的檔案，裡面的函式會讀入點座標，然後要畫圖時呼叫 mesh 類別的物件 -> draw()，即可渲染畫面，但這樣的話我每次要畫都需要重新傳入會很麻煩，所以我又使用一個 obj.cpp 去存入各種圖案，然後要用時再呼叫，有點像原生 glut 的感覺。Textute 的部分也是一樣，所以我在使用的時候需要先選 texture，在選 model。

```
glGetFloatv(GL_MODELVIEW_MATRIX, objMtx);
glUniformMatrix4fv(2, 1, GL_FALSE, objMtx);
myTex->grass_dark->use(programID);
myObj->tree_round_up->draw(programID);
myTex->coffee_dark->use(programID);
myObj->tree_round_btn->draw(programID);
```

  - 不過缺點時，每次要加入新的圖案，我都必須去改 mytex.cpp 或是 myobj.cpp 的程式碼。到最後就覺得好煩 q。mytex.cpp 提供四種函式，use 是使用那個圖案，可以傳入光罩係數改他的

material，或是不傳入，那就是 default = 0。至於創造一個 texture 有 2 種方法，一種是給 rgba，他會生成一塊 1*1 的顏色方塊，或是給檔案名，他會去讀圖片 。

```
void texture::use(unsigned int programID,float sper, float speg, float speb,float shine) { ... }
void texture::use(unsigned int programID) { ... }
texture::texture(const string& fname, unsigned int programID) { ... }
texture::texture(int r, int g, int b, int a, unsigned int programID) { ... }
```

```
mytex::mytex(unsigned int programID)
{
    //red = new texture(".\\texture\\a.png", programID);
    red = new texture(255,0,0,255, programID);
    //red = new texture(255,0,0,255, programID);
    black = new texture(0, 0, 0, 255, programID);
    white = new texture(255, 255, 255, 255, programID);
    robot_blue_main = new texture(219, 255, 255, 255, programID);
    robot_blue_sub = new texture(173, 214, 255, 255, programID);
    robot_blue_eye = new texture(128, 128, 255, 255, programID);
    robot_pink_eye = new texture(255, 122, 189, 255, programID);
    robot_gray = new texture(167, 167, 167, 255, programID);
    robot_gray_dark = new texture(127, 127, 127, 255, programID);
    elf_red = new texture(255,168,212,255, programID);
    for (int i = 0; i <= 55; i++) { ... }
    magic_wand_wood = new texture(158, 79, 0, 255, programID);
    fountain_base = new texture(".\\texture\\fountain_base.jpg", programID);
    fountain_water = new texture(".\\texture\\fountain_water.jpg", programID);
    grass_dark = new texture(1, 152, 89,255, programID);
    grass_light = new texture(204, 255, 204,255, programID);
    coffee_dark = new texture(141,84,28,255, programID);
    coffee_light = new texture(162, 92, 21, 255, programID);
    flower = new texture(".\\texture\\flower.png", programID);
    yellow_light = new texture(255, 255, 204, 255, programID);
    yellow_dark = new texture(173, 142, 0, 255, programID);
    red_dark = new texture(117, 0, 0, 255, programID);
    stone_floor = new texture(".\\texture\\stone_floor.jpg", programID);
    wood2 = new texture(".\\texture\\wood2.jpg", programID);
    wood3 = new texture(".\\texture\\wood3.jpg", programID);
    wood4 = new texture(".\\texture\\wood4.jpg", programID);
    wood5 = new texture(".\\texture\\wood5.jpg", programID);
    tablecloth = new texture(".\\texture\\tablecloth.jpg", programID);
    tent = new texture(".\\texture\\tent.jpg", programID);
    orange_dark = new texture(255, 165, 0, 255, programID);
    bread = new texture(".\\texture\\bread.jpg", programID);
    cheese = new texture(".\\texture\\cheese.jpg", programID);
    red_apple = new texture(".\\texture\\red_apple.jpg", programID);
    green_apple = new texture(".\\texture\\green_apple.jpg", programID);
    pear = new texture(".\\texture\\pear.jpg", programID);
    lemon = new texture(".\\texture\\lemon.jpg", programID);
```

```
myobj::myobj(unsigned int programID)
{
    solidsphere = getMesh(".\\model\\solidsphere.obj", programID);
    cube = getMesh(".\\model\\cube.obj", programID);
    cylinder = getMesh(".\\model\\cylinder.obj", programID);
    solidtorus_5_025 = getMesh(".\\model\\solidtorus_0.5_0.025.obj", programID);
    fountain_base = getMesh(".\\model\\fountain_base.obj", programID);
    fountain_water = getMesh(".\\model\\fountain_water.obj", programID);
    blacksmith = getMesh(".\\model\\Blacksmith.obj", programID);
    square = getMesh(".\\model\\square.obj", programID);
    vendor_base = getMesh(".\\model\\vendor_base.obj", programID);
    vendor_col = getMesh(".\\model\\vendor_col.obj", programID);
    vendor_roof = getMesh(".\\model\\vendor_roof.obj", programID);
    vendor_desk = getMesh(".\\model\\vendor_desk.obj", programID);
    tree_round_up = getMesh(".\\model\\tree_round_up.obj", programID);
    tree_round_btn = getMesh(".\\model\\tree_round_btn.obj", programID);
    tree_conical_up = getMesh(".\\model\\tree_conical_up.obj", programID);
    tree_conical_btn = getMesh(".\\model\\tree_conical_btn.obj", programID);
    cloud1 = getMesh(".\\model\\cloud1.obj", programID);
    vendor_crate = getMesh(".\\model\\vendor_crate.obj", programID);
    apple = getMesh(".\\model\\apple.obj", programID);
    cheese = getMesh(".\\model\\cheese.obj", programID);
    bread2 = getMesh(".\\model\\bread2.obj", programID);
    plate = getMesh(".\\model\\plate.obj", programID);
}
```

▲ 每次要用都需要去加入 q

四、心得:

經過這一次作業更加了解老師上學期圖學課介紹的內容了，不管是 Phong 模型，還是物體轉換的 pipeline 流程。我程式碼的設計是，將要畫的物體切成很多三角形，然後讀入三角形座標。因為這個做法的關係，所以我有使用 blender 輔助我建物體，blender 可以直接將物體輸出成三角形的座標，讓我讀入。不過，因為是第一次用，所以非常不擅長。希望可以開一堂專門教建模的課(x。然後這次作業要特別感謝冠宏，沒有他的幫忙我可能一行也寫不出來，謝謝冠宏。

冠宏除了教我怎麼寫 shader 外，還教我怎麼寫物件導向和使用 blender，冠宏根本是大神。再次謝謝冠宏!

五、未來展望:

還有三個店鋪沒有賣商品，而且也沒有商人，可能下一次會把他補完。