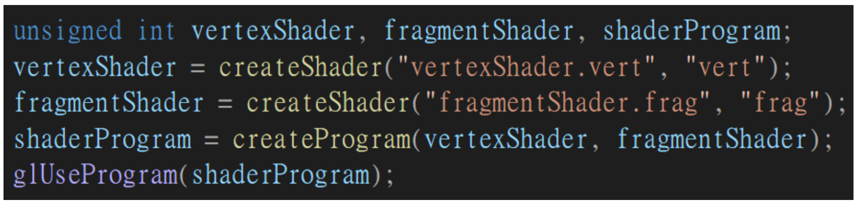
ICG HW2 Report

110550108 施柏江

1. How to use GLSL

1. Shader 設定：

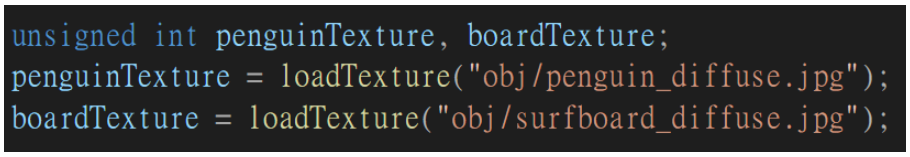
使用 createShader( )創建 vertex shader 和 fragment shader，再將 shaders 連接到 program 並使用該 program。



```
unsigned int vertexShader, fragmentShader, shaderProgram;
vertexShader = createShader("vertexShader.vert", "vert");
fragmentShader = createShader("fragmentShader.frag", "frag");
shaderProgram = createProgram(vertexShader, fragmentShader);
glUseProgram(shaderProgram);
```

2. 載入 Texture：

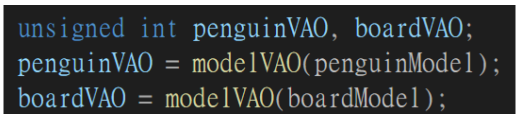
使用 loadTexture( )載入企鵝和衝浪板的 texture。



```
unsigned int penguinTexture, boardTexture;
penguinTexture = loadTexture("obj/penguin_diffuse.jpg");
boardTexture = loadTexture("obj/surfboard_diffuse.jpg");
```

3. VAO, VBO 設定：

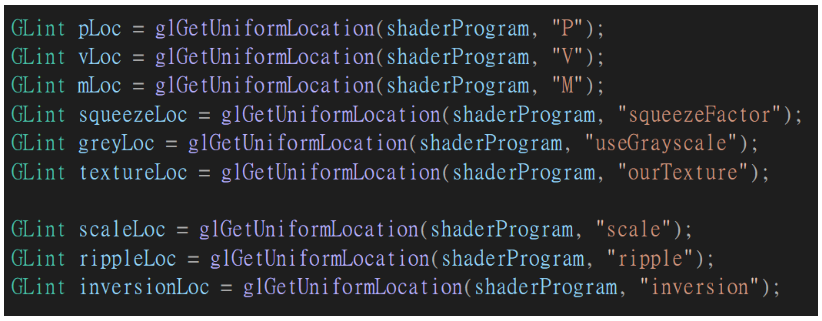
使用 modelVAO( )設定兩個 VAO，分別用於企鵝和衝浪板，並將模型的頂點數據放入 VBO。



```
unsigned int penguinVAO, boardVAO;
penguinVAO = modelVAO(penguinModel);
boardVAO = modelVAO(boardModel);
```

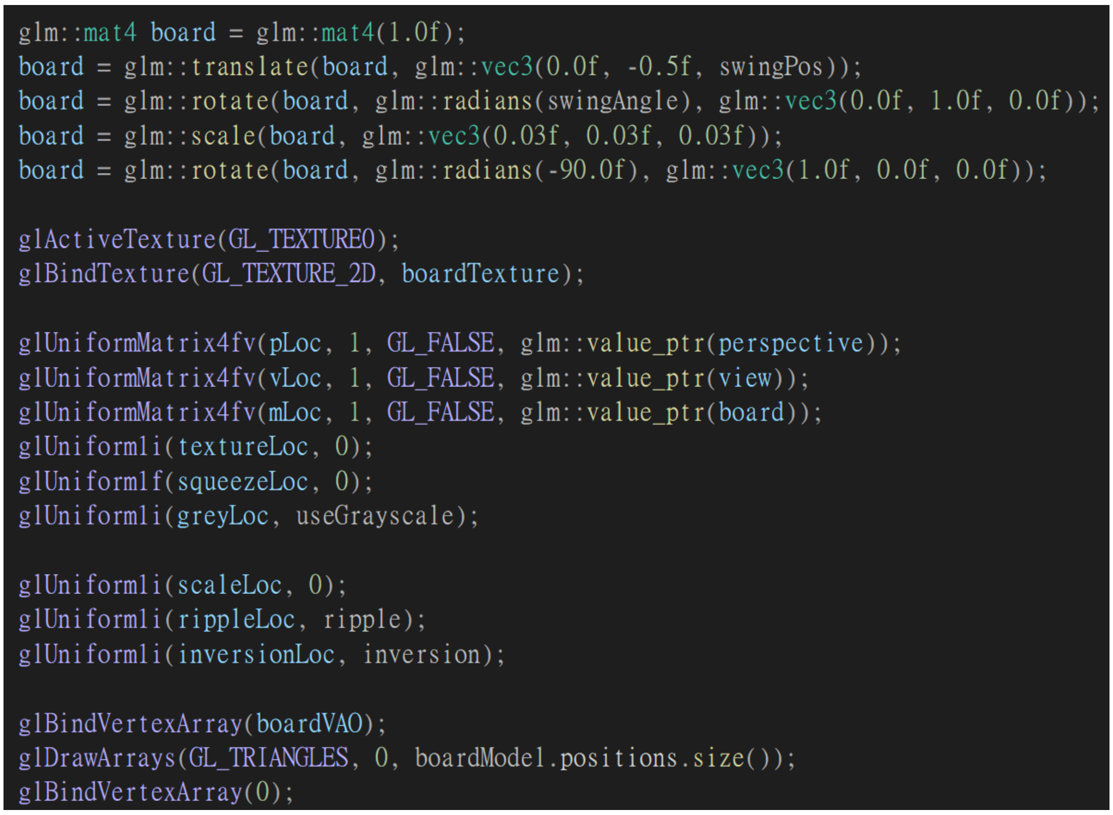
4. 獲取 Uniform 變數位置：

使用 glGetUniformLocation( )獲取 model、view、perspective matrix 和其他參數的 Uniform 變數位置。

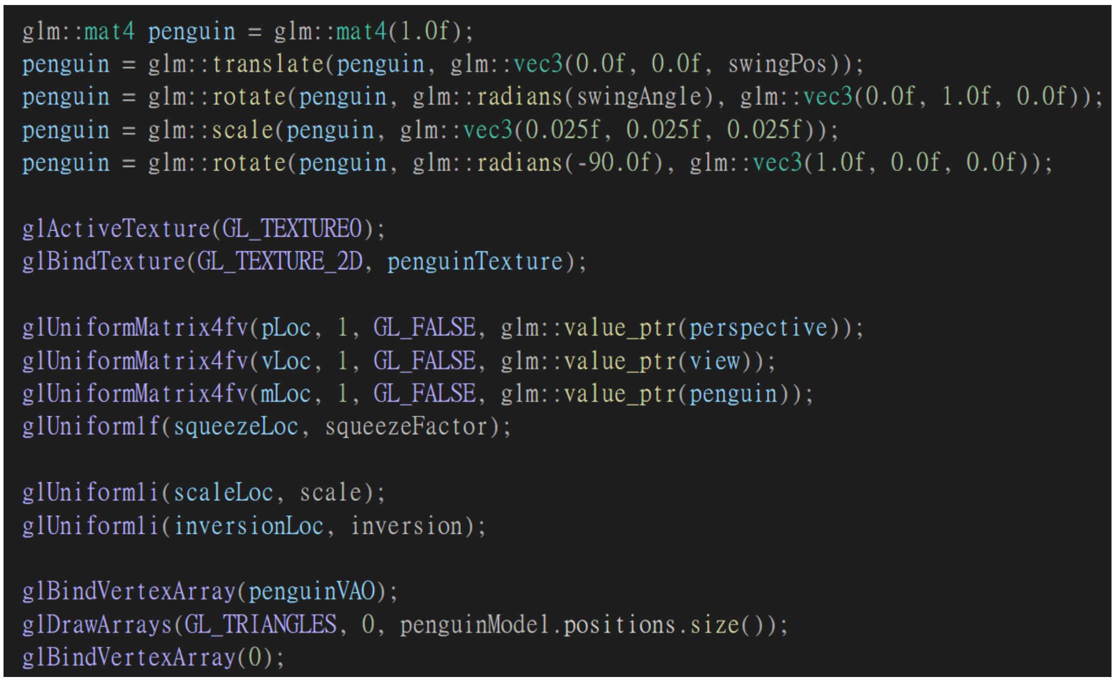


```
GLint pLoc = glGetUniformLocation(shaderProgram, "P");
GLint vLoc = glGetUniformLocation(shaderProgram, "V");
GLint mLoc = glGetUniformLocation(shaderProgram, "M");
GLint squeezeLoc = glGetUniformLocation(shaderProgram, "squeezeFactor");
GLint greyLoc = glGetUniformLocation(shaderProgram, "useGrayscale");
GLint textureLoc = glGetUniformLocation(shaderProgram, "ourTexture");

GLint scaleLoc = glGetUniformLocation(shaderProgram, "scale");
GLint rippleLoc = glGetUniformLocation(shaderProgram, "ripple");
GLint inversionLoc = glGetUniformLocation(shaderProgram, "inversion");
```

5. 渲染衝浪板和企鵝：
設定衝浪板和企鵝的 model matrix，包括平移、旋轉、縮放等操作。使用 glBindTexture( )將 texture 綁定到對應的單元。透過 glUniform( )將 model、view、perspective matrix 跟一些參數傳遞給 shader。最後使用 glBindVertexArray ( )和 glDrawArrays( )渲染物體。

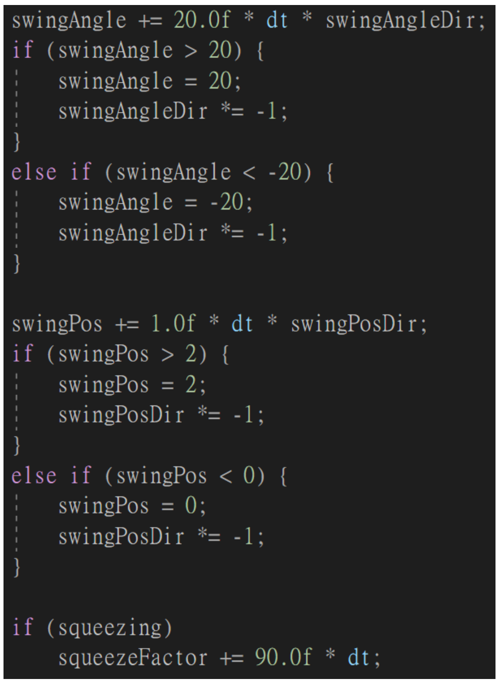


```
glm::mat4 board = glm::mat4(1.0f);
board = glm::translate(board, glm::vec3(0.0f, -0.5f, swingPos));
board = glm::rotate(board, glm::radians(swingAngle), glm::vec3(0.0f, 1.0f, 0.0f));
board = glm::scale(board, glm::vec3(0.03f, 0.03f, 0.03f));
board = glm::rotate(board, glm::radians(-90.0f), glm::vec3(1.0f, 0.0f, 0.0f));

glActiveTexture(GL_TEXTURE0);
glBindTexture(GL_TEXTURE_2D, boardTexture);

glUniformMatrix4fv(pLoc, 1, GL_FALSE, glm::value_ptr(perspective));
glUniformMatrix4fv(vLoc, 1, GL_FALSE, glm::value_ptr(view));
glUniformMatrix4fv(mLoc, 1, GL_FALSE, glm::value_ptr(board));
glUniform1i(textureLoc, 0);
glUniform1f(squeezeLoc, 0);
glUniform1i(greyLoc, useGrayscale);

glUniform1i(scaleLoc, 0);
glUniform1i(rippleLoc, ripple);
glUniform1i(inversionLoc, inversion);

glBindVertexArray(boardVAO);
glDrawArrays(GL_TRIANGLES, 0, boardModel.positions.size());
glBindVertexArray(0);
```



```
glm::mat4 penguin = glm::mat4(1.0f);
penguin = glm::translate(penguin, glm::vec3(0.0f, 0.0f, swingPos));
penguin = glm::rotate(penguin, glm::radians(swingAngle), glm::vec3(0.0f, 1.0f, 0.0f));
penguin = glm::scale(penguin, glm::vec3(0.025f, 0.025f, 0.025f));
penguin = glm::rotate(penguin, glm::radians(-90.0f), glm::vec3(1.0f, 0.0f, 0.0f));

glActiveTexture(GL_TEXTURE0);
glBindTexture(GL_TEXTURE_2D, penguinTexture);

glUniformMatrix4fv(pLoc, 1, GL_FALSE, glm::value_ptr(perspective));
glUniformMatrix4fv(vLoc, 1, GL_FALSE, glm::value_ptr(view));
glUniformMatrix4fv(mLoc, 1, GL_FALSE, glm::value_ptr(penguin));
glUniform1f(squeezeLoc, squeezeFactor);

glUniform1i(scaleLoc, scale);
glUniform1i(inversionLoc, inversion);

glBindVertexArray(penguinVAO);
glDrawArrays(GL_TRIANGLES, 0, penguinModel.positions.size());
glBindVertexArray(0);
```

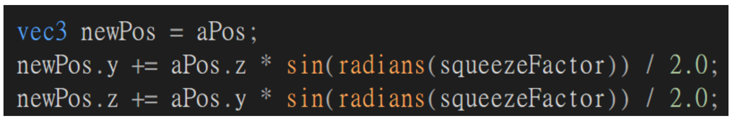
6. 更新動畫參數：
更新擺動的角度、位置以及 squeeze factor，以實現動畫效果。



```
swingAngle += 20.0f * dt * swingAngleDir;
if (swingAngle > 20) {
    swingAngle = 20;
    swingAngleDir *= -1;
}
else if (swingAngle < -20) {
    swingAngle = -20;
    swingAngleDir *= -1;
}

swingPos += 1.0f * dt * swingPosDir;
if (swingPos > 2) {
    swingPos = 2;
    swingPosDir *= -1;
}
else if (swingPos < 0) {
    swingPos = 0;
    swingPosDir *= -1;
}

if (squeezing)
    squeezeFactor += 90.0f * dt;
```

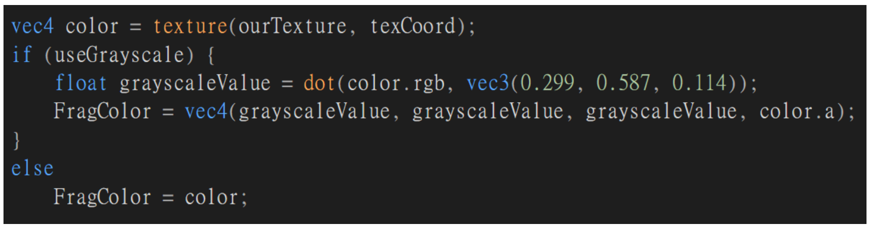
2. Key Callback Implementation

1. Key S: Start / Stop penguin squeezing



```
vec3 newPos = aPos;
newPos.y += aPos.z * sin(radians(squeezeFactor)) / 2.0;
newPos.z += aPos.y * sin(radians(squeezeFactor)) / 2.0;
```

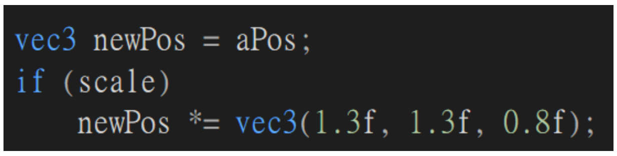
2. Key G: Enable / Disable grayscale



```
vec4 color = texture(ourTexture, texCoord);
if (useGrayscale) {
    float grayscaleValue = dot(color.rgb, vec3(0.299, 0.587, 0.114));
    FragColor = vec4(grayscaleValue, grayscaleValue, grayscaleValue, color.a);
}
else
    FragColor = color;
```

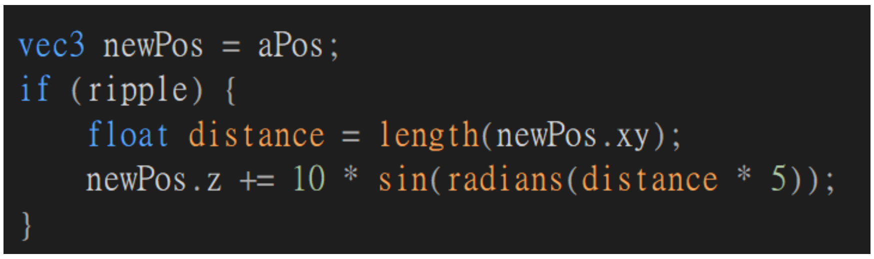
(bonus)

3. Key 1: Start / Stop penguin scaling
   按下 1 後企鵝會變得比較矮胖，再按一次會復原



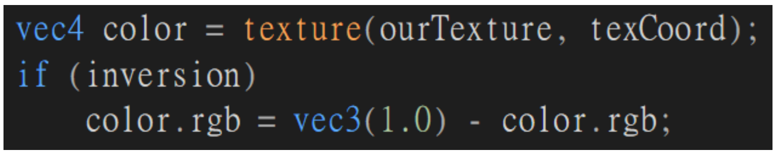
```
vec3 newPos = aPos;
if (scale)
    newPos *= vec3(1.3f, 1.3f, 0.8f);
```

4. Key 2: Start / Stop board rippling
   按下 2 後衝浪板會變成波浪形，再按一次會復原



```
vec3 newPos = aPos;
if (ripple) {
    float distance = length(newPos.xy);
    newPos.z += 10 * sin(radians(distance * 5));
}
```

5. Key 3: Enable / Disable color inversion
   按下 3 後顏色會變成互補色，再按一次會復原



```
vec4 color = texture(ourTexture, texCoord);
if (inversion)
    color.rgb = vec3(1.0) - color.rgb;
```

3. Problems

1. 在寫 createShader 時，以為直接將檔名傳入 glShaderSource( )就可以了，導致出現了 shader compilation error，上網查了之後才發現要先把 shader file 讀入，修改之後就成功創建 shader 了。
2. 在寫 modelVAO 時，發現範例裡用了 3 個 VBO，一開始不知道除了 positions 還要什麼要設定，於是跑到 Object.h 中查了一下，看到了 vertex 還有 normals 和 texcoords 的資訊，而且 VBO 要一個 bind 完並且把 glBufferData( ) 和 glVertexAttribPointer( )設置好才能再 bind 另一個，不然會把原本的覆蓋掉。