計算機圖學HW2 Report

0611031 資工10 謝至恆

create program

依照example的code做createShader

```
void shaderInit() {
    //// TODO: ////
    //
    // Hint:
    // 1. createShader
    //-----------------------------------------------------------------
    GLuint vert = createShader("Shaders/vertexShader.vert", "vertex");
    GLuint frag = createShader("Shaders/fragmentShader.frag", "fragment");
    // 2. createProgram
    program = createProgram(vert, frag);
    //-----------------------------------------------------------------
```

create VAO and VBO and Bind

- glGenBuffers產生buffer
- glBindVertexArray綁定VAO
- 生成3個VBO的buffer
  以綁定VBO[0]為例子

```
void bindbufferInit() {
    //Generate a new buffer object
    // 1. Setup VAO
    glGenBuffers(1, &VAO);
    glBindVertexArray(VAO);

    // 2. Setup VBO of vertex positions, normals, and texcoords

    glGenBuffers(3, VBO);
    glBindBuffer(GL_ARRAY_BUFFER, VBO[0]);
```

Copy data to the buffer object

- glBufferData設置要傳入shader的資料形式　(target , size , * data , usage)
- glVertexAttribute告訴shader應該怎麼讀　(index , size , type , normalized , stride , * pointer)
- glEnableVertexAttrribArray告訴vertex shader此筆資料是哪個layout

```
//Copy vertex data to the buffer object
//position
VertexAttribute* vertices;
vertices = drawVertex();
glBufferData(GL_ARRAY_BUFFER, sizeof(VertexAttribute) * 4 * model->fNum, vertices, GL_STATIC_DRAW);
glVertexAttribPointer(0, 3, GL_FLOAT, GL_FALSE, sizeof(VertexAttribute), (void*)(offsetof(VertexAttribute, position)));
glEnableVertexAttribArray(0);

//normal
VertexAttribute* normals;
normals = drawNormal();
glBindBuffer(GL_ARRAY_BUFFER, VBO[1]);
glBufferData(GL_ARRAY_BUFFER, sizeof(VertexAttribute) * 4 * model->fNum, normals, GL_STATIC_DRAW);
glVertexAttribPointer(1, 3, GL_FLOAT, GL_FALSE, sizeof(VertexAttribute), (void*)(offsetof(VertexAttribute, position)));
glEnableVertexAttribArray(1);

//texcoord
VertexAttribute* texcoords;
texcoords = drawTexcoord();
glBindBuffer(GL_ARRAY_BUFFER, VBO[2]);
glBufferData(GL_ARRAY_BUFFER, sizeof(VertexAttribute) * 4 * model->fNum, texcoords, GL_STATIC_DRAW);
//bug position -> texcord
glVertexAttribPointer(2, 2, GL_FLOAT, GL_FALSE, sizeof(VertexAttribute), (void*)(offsetof(VertexAttribute, texcoord)));
glEnableVertexAttribArray(2);
//-------------------------------------------------------------------------------------------
```

VertexAttribute
把model中的點
給shader的函數
使用自訂的函數
把每個vertex所要
傳的點壓成一個
vertices並回傳

```
VertexAttribute* drawVertex() {
    VertexAttribute* vertices;
    vertices = new VertexAttribute[4*model->fNum];
    for (int i = 0; i < 4 * model->fNum ; i++) {
        vertices[i].setPosition(model->positions[3*i], model->positions[3*i + 1], model->positions[3*i + 2]);
    }
    return vertices;
}

VertexAttribute* drawNormal() {
    VertexAttribute* vertices;
    vertices = new VertexAttribute[4 * model->fNum];
    for (int i = 0; i < 4 * model->fNum; i++) {
        vertices[i].setPosition(model->normals[3 * i], model->normals[3 * i + 1], model->normals[3 * i + 2]);
    }
    return vertices;
}

VertexAttribute* drawTexcoord() {
    VertexAttribute* vertices;
    vertices = new VertexAttribute[4 * model->fNum];
    for (int i = 0; i < 4 * model->fNum; i++) {
        vertices[i].setTexcoord(model->texcoords[2 * i], model->texcoords[2 * i + 1]);
    }
    return vertices;
}
```

VertexAttribute

- setPosition 給一個3個GLfloat的資料
- setTexcoord 給一個2個GLfloat的資料

```
class VertexAttribute {
public:
    GLfloat position[3];
    void setPosition(float x, float y, float z) {
        position[0] = x;
        position[1] = y;
        position[2] = z;
    };
    GLfloat texcoord[2];
    void setTexcoord(float x, float y) {
        texcoord[0] = x;
        texcoord[1] = y;
    };

};
```

Vertex shader

```
#version 430

layout(location = 0) in vec3 position;
layout(location = 1) in vec3 normal;
layout(location = 2) in vec2 texcoord;

uniform mat4 M;
uniform mat4 Projection;
uniform mat4 View;

out vec3 frag_normal;
out vec2 frag_texcoord;

void main() {
  gl_Position =  Projection * View * M * vec4(position, 1.0);
  frag_normal = normal;
  frag_texcoord = texcoord;
}
```

fragment shader

```
#version 430

layout(binding = 0) uniform sampler2D Texture;

in vec3 frag_normal;
in vec2 frag_texcoord;

out vec4 frag_Color;


void main() {
    frag_Color = texture2D(Texture, frag_texcoord);
}
```

problems I met

1. 一開始vertex shader寫好後發現Projection , View , M matrix 都沒有作用，但shader有把點畫出來，還在檢查是不是matrix 的data type給錯之類的
2. fragment shader卻沒有作用，甚至連基本的color = (1.0,0.0,0.0,1.0) 都沒有跑出紅色

- 解決 >> 發現沒用glUseProgram(program)

3. Texcoord的offset給錯，vec2的資料

   卻給vec3的offset結果就是材質跑不出來

   只有一隻灰伊布，還以為資料沒傳到fragment shader

- 解決 >> 重新看程式邏輯的時候發現offset錯誤

Additional function

- light() 因為看助教的demo範例basis好像有光照陰影的樣子，所以把HW1的light()搬過來了
- keyboard()使用空白鍵可以讓整個模型停止旋轉

成品