# Report

1. main.cpp

draw basis:

- 用GL\_POLYGON畫20邊形

- 用GL\_POLYGON畫長方形

- glActiveTexture()選擇texture通道。

- glBindTexture(a,b)指定a型texture的b texture

- glCoord2f()連接texture座標(0,0)~(1,1)與vertex座標

draw Umbreon

- calculate matrices including projection, modelview and TRS matrices

- glGetUniformLocation(program,U)where U is uniform variable in vertex shader

- glUniformMatrix4fv(Mloc,,,) pass the matrices to vertex shader.

shaderInit():

- create vertex , fragment shader and program

- bind program to two shaders

bindbufferInit():

- generate VBO, VAO

- set VBO data(vertices positions, texture coordinates and normals)

VBO

- record the data of vertices data

- pass the data to vertex shader

- glEnableVertexAttribArray(i) , glVertexAttribPointer(i,…) connect VBO to vertex shader in location i.

VAO

- record all VBOs

2. vertexShader.vert

- layout(location = i) in vec3: receiving the data from VBO in tunnel i

- uniform mat4 : receiving matrices from openGL

- out vec2/3 : output fragment data (texture coord , color etc.) to fragment shader

- main() : vertex calculation (Matrices multiplication)

3. fragmentShader.frag

- in vec2/3 : receiving data from vertex shader

- out vec4 : output color

- main() : calculating the texture data.