

Homework 2 Report

張仁杰 | Computer Graphic | 11/6, 2018

# The OpenGL Shaders

在這次的作業中，我將三種效果寫在分別的shader中，而每種效果又有各自的vertex shader與fragment shader，故宜共有六個shader file，統一放在CG\_HW2/Shader目錄下。其中的三個vertex shader是完全相同的，這三種效果實際上都是利用fragment shader來實作。

Vertex shader的功能較單純，最主要就是將vertex的位置進行transformation以及projection。我是分別將view matrix以及model matrix以uniform的方式傳給shader，因為在計算normal matrix時需要用到model matrix。計算fragment座標的部分也是時坐在vertex shader中，由於我是在world space中進行lighting的計算，所以需要先將座標轉換到world space。在vertex shader中計算出頂點的world space座標後，OpenGL會先將此座標進行內插在傳到fragment shader，也就是fragment的實際座標。

三種效果的Fragmet shader皆不相同，以下分別進行說明。

# Phong Shading

由於spec中規定material的三個component Ka, Kd, Ks皆為(1, 1, 1)，故計算時可以直接不考慮這三個項，由於燈光除了位置之外的安數都不會變，因此都以常數的方式寫在fragment shader中。計算三個component以及attenuation的方法如下：

* Ambient：由於material的ambient項為(1,1,1)，燈光的ambient(0.2,0.2,0.2)即最終的結果。
* Diffuse：需要用到normal以及light direction兩個vector。Normal是由vertex shader將頂點的normal vector乘上normal matrix後，再內差而得。Light direction是由fragment指向light的vector，將燈光位置減fragment位置即可得到。最後將normal以及light direction做內積，乘上燈光的diffuse項即是結果。
* Specular：與view direction以及reflection direction有關。View direction是由fragment指向camera的vector，reflection direction是光線反射的方向，可以由light direction對normal進行鏡射得到，鏡射的部分GLSL有內建函數可使用(reflect)
* Attenuation：只跟fragment與光源的距離有關，此處就只是單純距離平方的倒數。

# Dissolving

我將dissolving的效果已自動化的方式呈現，物體會從原本的樣子逐漸消失但完全不見，再逐漸回到完整的樣子。為了達到此效果，我再main.cpp中新增了一個變數dissolvingThresh，以uniform的方式傳入shader，並且在idle callback中不斷更新它，dissolvingThresh會在0~1之間不斷來會變動。在fragment shader種首先要對noise texture進行sampling得到一個亮度值，若noise小於dissolvingThresh，則discard此fragment，若noise只比dissolvingThresh稍大(此數我選的是0.3)，代表fragment位於邊界，黃色作為fragmanet color，其他case則是正常繪製。

# Ramp Shading

主要是對diffuse的亮度進行調整，首先與phong shading相同先計算出diffuse component，之後利用diffuse的亮度對ramp texure進行採樣，得到新的亮度，最後將diffuse乘以新的亮度在除以原本的亮度，即可達到ramp shading效果。