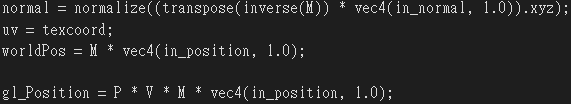
0816028 周孟謙

Phongshading

Vertex shader:



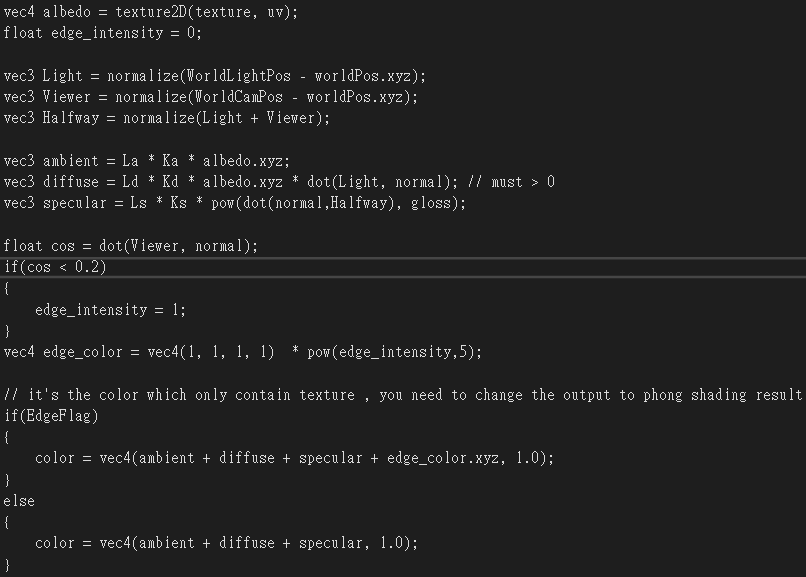
normal的部分由(Nn)\*(Mv)=I推導出N=(M^(-1))^T。

uv則直接由texcoord帶入。

worldPos=modelview乘上in\_position。

gl\_Position則與原本code一樣，沒做更改。

Fragment shader:



先將Light vector, Viewer vector用pos相減後得出再做normalize，halfway則依照公式h = ( l + v )/ | l + v |(l為Light vector，v為Viewer vector)算出。

Ambient, diffuse, specular皆代入公式:

ambient = La \* Ka \* object\_color;

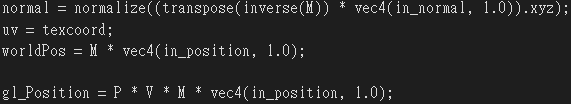
diffuse = Ld \* Kd \* object\_color \* dot(L,N);

specular = Ls \* Ks \* pow(dot(V,R), gloss);

Edge effects的部分則是當Viewer vector和normal vector之間的cos小於0.2時將edge\_intensity設為1，最後再用EdgeFlag判斷要不要將edge\_color加到color裡。

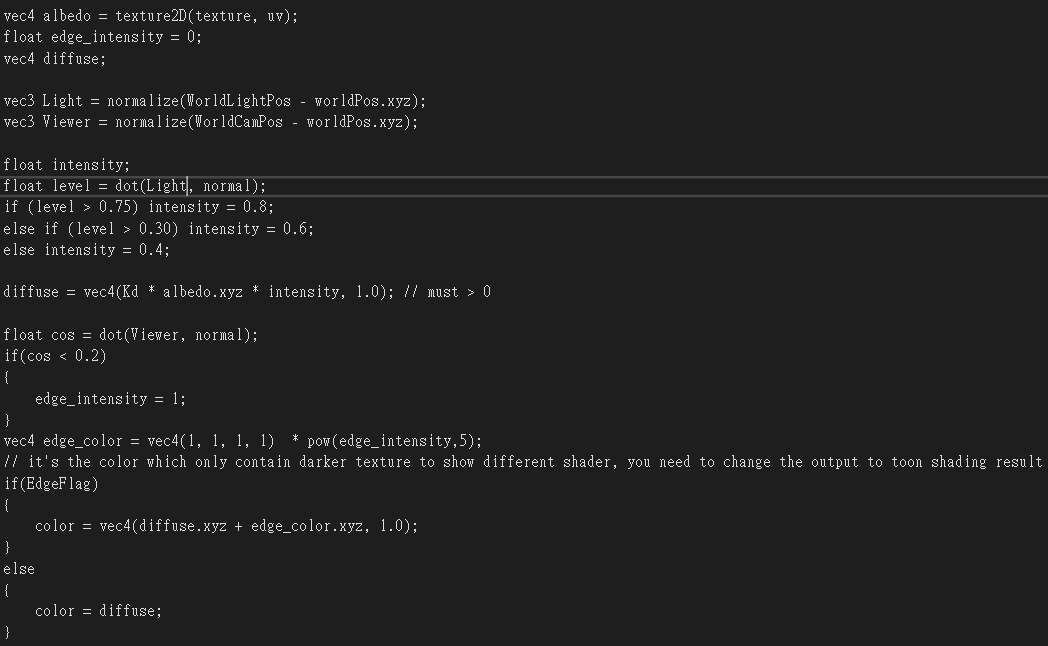
Toon

Vertex shader:



與Phongshading的Vertex shader一樣。

Fragment shader:



先將Viewer vector用pos相減後得出再做normalize。

level的部分則是用Light vector和normal vector之間cos的大小來作區間，>0.75, 0.75>= && >0.3, >=0.3這三個區間分別設三種intensity(0.8, 0.6, 0.4)。

diffuse代入公式Kd \* object\_color \* intensity。

Edge effects的部分則是與Phongshading相同，當Viewer vector和normal vector之間的cos小於0.2時將edge\_intensity設為1，最後再用EdgeFlag判斷要不要將edge\_color加到color裡。

Describe the problems you met and how you solved them.

某些vec3和vec4的轉換沒有做好，只要將每個vec看清楚維度再做轉換就沒問題了。還有在不理解公式的情況下亂代入，解決辦法則是完全搞懂每個公式的意義以及它在幹甚麼，問題便迎刃而解。