Computer Graphics HW1

104062203 陳涵宇

[TODO] Put structure value to 1D array:

由於 for loop 的次數是三角形個數、每個三角形有 3 個點、每點有 3 個座標，因此我先用 3 個變數(vertex_idx1,2,3)記錄目前三角形的 3 個座標的存放位置，再根據這 3 個座標利用 vertex_idx1,2,3 * 3 + 0,1,2 的公式依序取得座標值(x,y,z)，最後再利用公式 i * 9 + 0~9 存入對應的位置。color 的部分作法同 vertices index。

公式解釋:

1. 取得座標

vertex_idx 記錄的是座標的存放 index，由於每個座標有 3 個座標值，因此要從 m.obj->vertices 取值時需要乘以 3 才會指到正確的座標值。公式後面的+0,1,2 則是因為同座標的值會存放在連續位置，因此可利用這種方式取得同座標的 x,y,z。

2. 存放座標

由於存入的矩陣是一維的，因此每個三角形都需要連續 9 個位置存放資料(3 個點*3 個座標值)。當 for loop 在處理第 i 個三角形時，矩陣對應的位置即為 i*9+0~8，只要依序放入即可。

[TODO] Normalize each vertex value:

這邊的 for loop 會重複三角形個數*3 次，因此從 m.vertices 取值時只需要 i*3+0,1,2。後面則是利用公式計算新值並存回相同的位置。

[TODO] Link vertex and color matrix to shader parameter

void **glVertexAttribPointer**(GLuint *index*, GLint *size*, GLenum *type*, GLboolean *normalized*, GLsizei *stride*, const GLvoid * *pointer*);

index: 起始 index(iLocPosition/iLocColor)

size: 每個 vertex 的 component 數量(3/3)

type: 資料型別(GL_FLOAT/ GL_FLOAT)

normalized: 定點資料是否要 normalized(GL_FALSE/ GL_FALSE)

stride: 連續資料之間的偏移量(0/0)

pointer: 指向第一筆資料的第一個 component(model->vertices/ model->colors)

[TODO] Drawing command:

void **glDrawArrays**(GLenum *mode*, GLint *first*, GLsizei *count*);
mode: 繪製的基本型別(GL_TRIANGLES)
first: 起始 index(0)
count: 繪製資料數量(model->obj->numtriangles*3)

[TODO] Change polygon mode (GL_LINE or GL_FILL):

每次執行時利用 not 將 use_wire_mode 的值在 true, false 之間轉換，並根據變數值改變使用的 mode。