How to operate my program

"Keyboard :"

"Press T to enter geometrical Translation mode”
"Press R to enter geometrical Rotation mode"
"Press S to enter geometrical Scaling mode"
"Press E to enter eyes' position mode"
"Press L to increase x coord."
"Press J to decrease x coord."
"Press I to increase y coord."
"Press K to decrease y coord."
"Press M to increase z coord."
"Press O to decrease z coord."
"Press P to toggle projection mode"
"Press Q to reset "
"Press left arrow or right arrow to change current model pointer"

"Mouse :"

"Press E/other keys(T, S, R) to switch between geometry mode and eye position mode"
“Press middle button to reset.”
“Press T, R or S to enter geometry mode”
"If in geometry mode,"
"Press left button and drag to translate current model"
"Press right button and drag to rotate current model"
"Wheel up and down to scale current model"
“Press E to enter eye position mode”
"If in eye position mode,"
"Press left button and drag to move both eye position and center position"
"Press right button and drag to move only eye position in the x and y axes"
"Wheel up and down to move eye position in the z axis"

Implementation and problems I met

首先先將所有 OBJ 都讀進程式中，並為每個 model maintain 一個 structure，讀進來的同時算出每個 model 的長寬高及圖的中心記錄在 structure 中。接著進入計算 MVP 矩陣的階段。我將 Model Matrix 分成五個矩陣，分別是 Init、R、S、T 與 displace_matrix。Init matrix 的功能是將 model 移到中心點 (0, 0, 0) 的位置，並 normalize；R 是 Rotation matrix；S 是 scaling matrix；T 是 translation matrix；displace_matrix 的作用是當我有五張圖時，若我要把他們都擺在 window 中，不可能把他們全部都擺在中心，所以我在 normalize 與 model matrix 乘完以後會再進行一次 translation，並以 flag 決定每張圖應該在位置。

```
Matrix4 D = Matrix4(
    1, 0, 0, dx[(index-currentModel+5)%5],
    0, 1, 0, dy[(index-currentModel+5)%5],
                    0, 0, 1, 0,
                    0, 0, 0, 1);
```

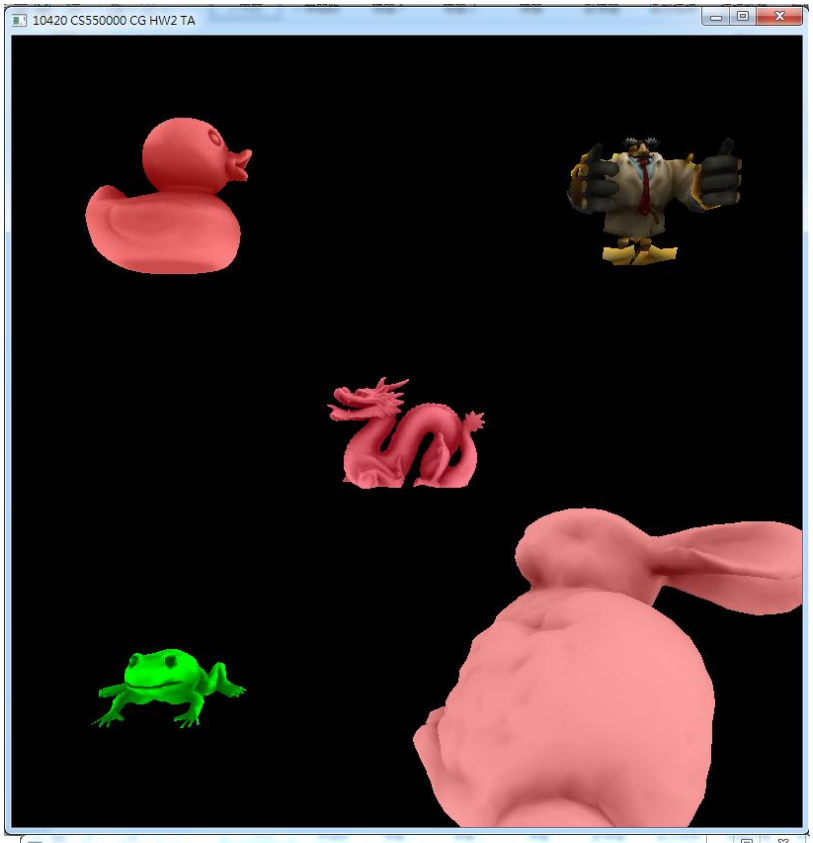
透過 mod 的方式，使 geometry 可以操作的圖永遠位於 window 正中間，即座標 (0, 0, 0)，最後 M = displace_matrix * T * S * R * Init，即可得到最後的 Model Matrix。View Matrix 也是在每次畫之前，根據 eye position render 出一個新的 matrix。Projection matrix 會根據 flag 切換兩種投影矩陣，其中 perspective matrix 是用 frustum 的方式實作，透過參數可以設定 view volume 的大小。當所有矩陣都計算完成後，mvp = P*V*M，再傳入 shader program 就能畫出我們想要的圖案了。

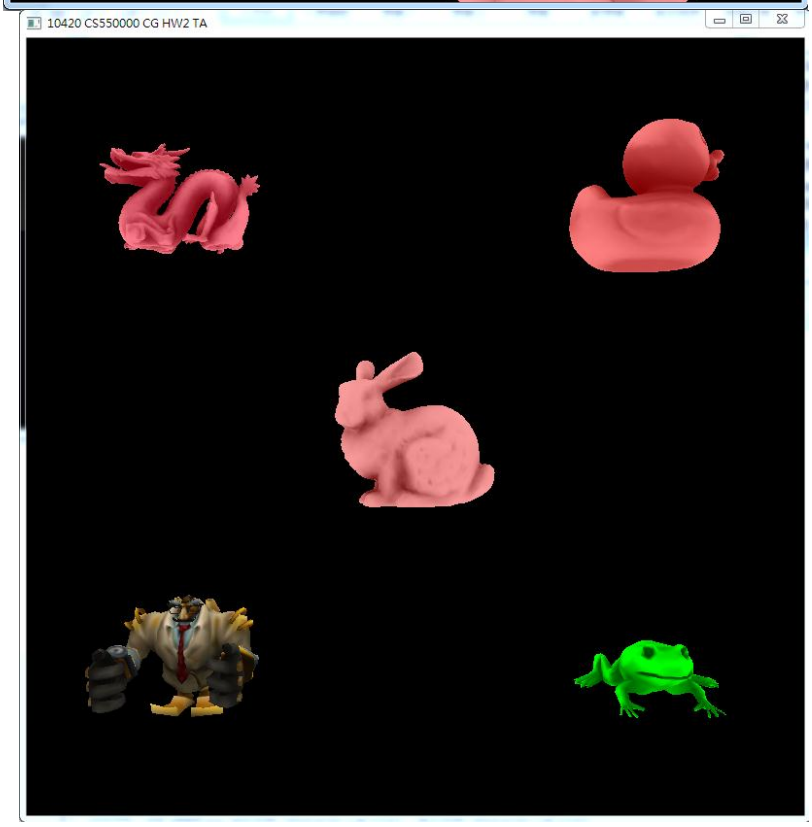
鍵盤操作部分很單純，照著 spec 所規定，按按鍵就能正確更改參數。滑鼠操作的部分，當按下不同按鍵時，我就設定一個 state flag 記錄目前應該是哪個操作，當按鍵被放開時，我就把 flag 變回初始化狀態 0 。然後在 onMouse function 裡面 trace 滑鼠移動後的位子並與初始位子相減，並根據 state flag 顯示現在是 rotation、translation 或特定操作後，將相差的值做正確的 scale 並加到對應的參數去，使下一次計算 mvp 時能確實反映參數的改變。

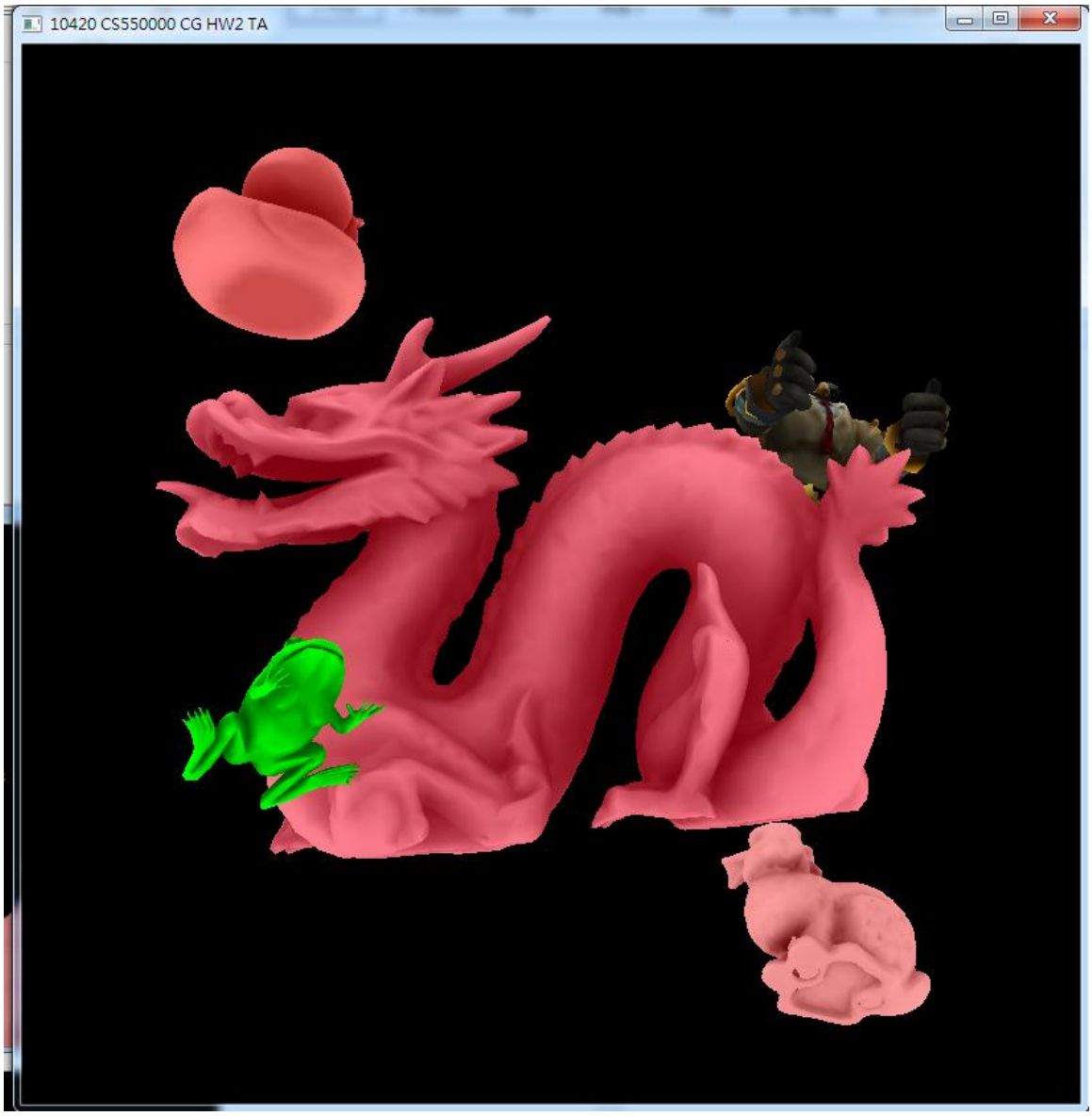
Other Efforts I have made

除了透過鍵盤控制 eye position 外，滑鼠也支援 eye position 的改變。按 E 進入 eye position mode 後，按住左鍵拖曳可使 center position 與 eye position 同時移動，相當於視野的平移；按住右鍵可以只更動 eye position 而不改變 center position，相當於 camera 的旋轉；滾動滾輪可以調整 eye z-position；按下滾輪則可以重製所有改變。

- **Screenshots**






10420 CS550000 CG HW2 TA