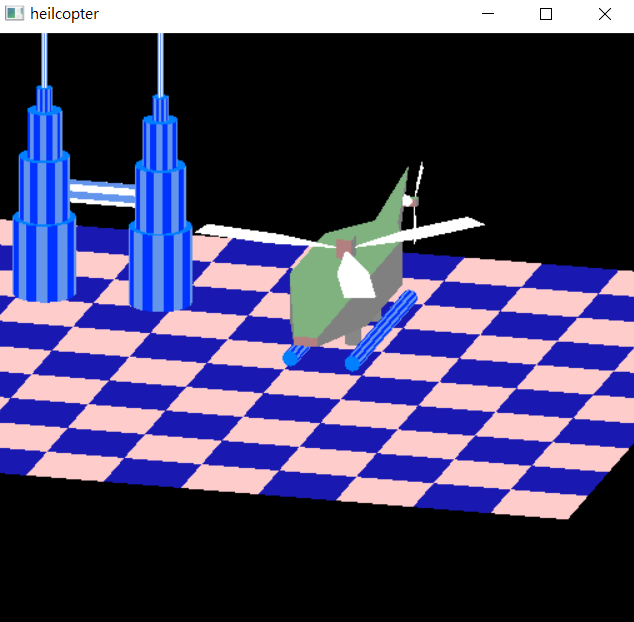
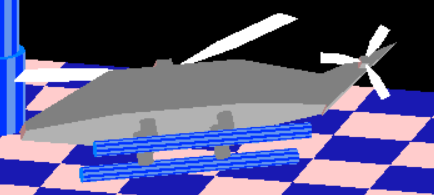
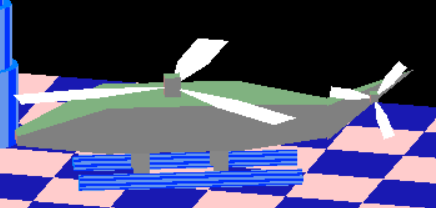
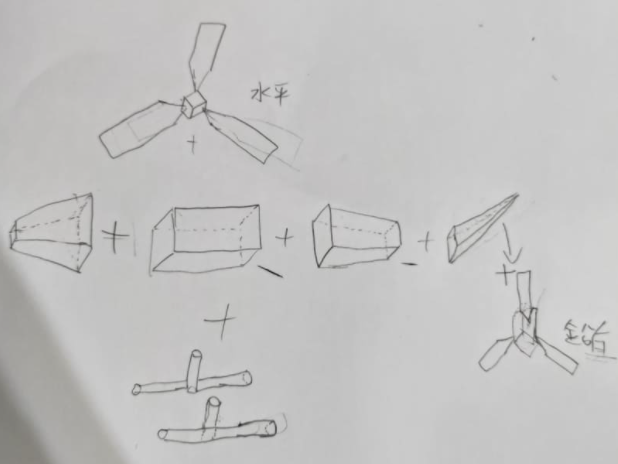
初始畫面:



直升機構成:





其中腳架橫向的為二十面體

按鍵說明:

|  |  |
| --- | --- |
| **按鍵** | **功能** |
| Esc | 退出程式 |
| w | 往上1格 |
| s | 往下1格 |
| W | Y方向向前翻滾10度 |
| S | Y方向向後翻滾10度 |
| Q q | 向左旋轉10度 |
| E e | 向右旋轉10度 |
| A a | 向左平移0.5格 |
| D d | 向右平移0.5格 |
| 8 | 向前平移0.5格 |
| 5 | 向後平移0.5格 |
| R r | 切換為擬真模式 |

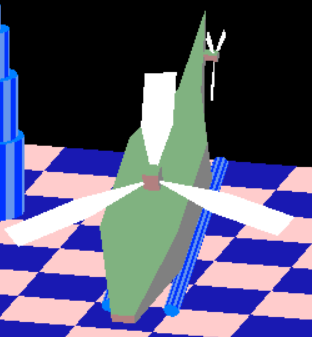
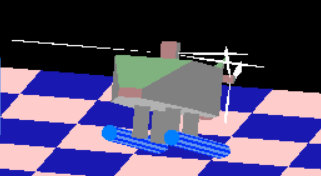
擬真模式按鍵說明:

|  |  |
| --- | --- |
| 按鍵 | 功能 |
| Esc | 退出程式 |
| W w | 上升 |
| S s | 下降 |
| A a | 向左移動 |
| D d | 向右移動 |
| 8 | 向前移動 |
| 5 | 向後移動 |
| Q q | 向左旋轉 |
| E e | 向右旋轉 |
| Else | 係數歸零 |

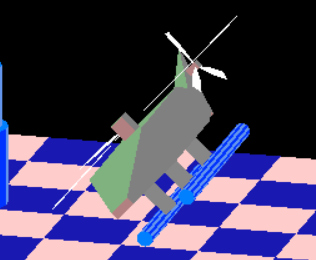
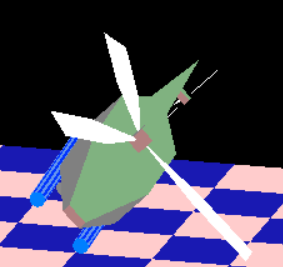
擬真模式說明:

模擬真實直升機飛行方式，並新增加速度，移動時觸發

前進/後退

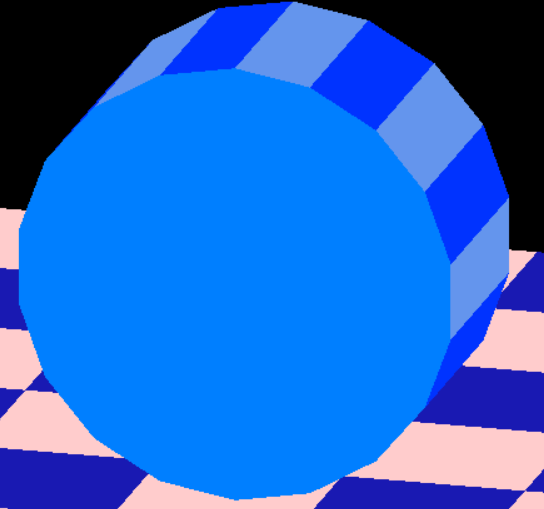
右移/左移

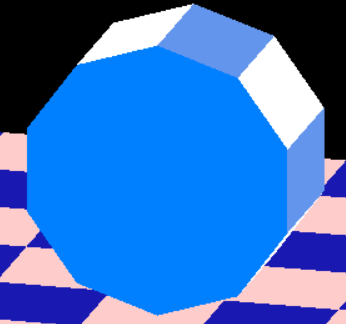
場景說明:

--基本元件

18角柱(20面體)



10角柱(12面體)



--成品(左) 右邊為參考原型

Bridge跟floor5是用12面體，其餘為20面體

程式碼說明:

直升機:

--上螺旋槳:

葉片部分參考sample code(wind\_mill)，放大倍率(2,1,2.5)，中心為放大倍率(2,3,2)的cube

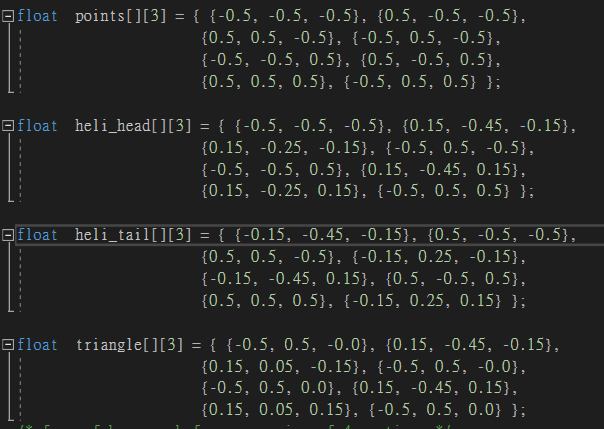
--尾螺旋槳:

葉片部分參考sample code(wind\_mill)，放大倍率(0.7,0.7,0.7)，中心為放大倍率(1,1,3)的cube

--機身

由三個六面體，一個四面體組成

採用資料集(由sample code car中的資料集修改):



--起落架:

圓柱參照sample code(wind\_mill)

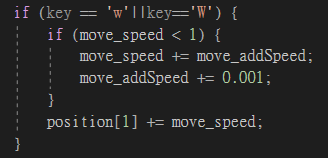
由兩個橫放的圓柱，倍率(4,1,1)和四個直放的圓柱(未放大)組成

擬真模式:

定義以下參數，分別是移動和旋轉的速度和加速度

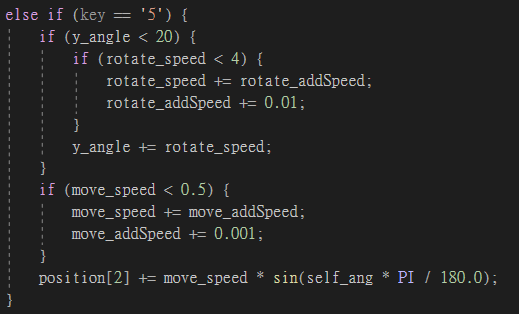


上升/下降 左/右旋轉



定義速度上界為1，當速度未超過上界，加速度增加，速度加上加速度，反之則維持不變

前進/後退 左/右平移:

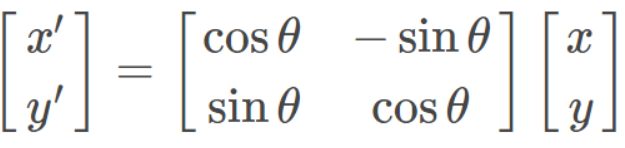


此類操作會用到傾斜機身，所以除了上面的處裡還要增加機身角度的處裡

只是鍵盤事件的處裡我沒處理好，所以每做完一個動作要手動重設速度參數

--背景:

10角柱(12面體)/18角柱(20面體):

旋轉矩陣可以求出x, y 以原點為中心旋轉theta度後的點座標，於是我設原點為該正n邊形中心，旋轉n次，這樣就可以求出構成該正n邊形所需的所有點。

在z=0時做一次該運算，z=1時做一次該運算，x=0, y=1，這樣即可得到構成邊長為1的n角柱中所需的所有點。

心得:

以往計算都在二維座標上計算，突然換成三維有點不習慣。

OpenGL 的push pop和 model view摸索了一大段時間

Sample code給得真好