實踐大學

資訊科技與通訊學系

專題學士論文

無人飛行船自動導航系統

Autonomous Navigation System of an Unmanned Airship

組員：

林亞澤　侯圳嶺　侯劭東

劉文茜　葉曉霈　鄧靜容

指導教授：

吳啟偉老師

民國 100 年

專題摘要

本專題以 Microchip 公司所生產的 PIC24FJ128GB106 板子為主體，搭配自行製作的子板銜接 GPS、GPRS 以及 Compass 等週邊模組，達成無人飛行船自動飛行的目標。

本研究已經能夠讓飛行船自動依據設定好的路線巡航，並利用 GPRS 回傳飛行即時動態資料至地面監測站，其中包含經緯度、高度、溫度、目前動作、下一目的地、以及航向、船身俯仰、偏傾等相關資訊，同時地面監測站也可以利用 GPRS 下達控制指令至飛船上，即時改變目前動作或者是更改目的地座標。

本專題從自行開發軟硬體控制系統後，歷經多次地面測試(Ground Test)及空中飛行測試(Fly Test)，已成功完成系統各項功能驗証，其中包括自動飛行導航、手動自動飛行模式切換、地面監控資料接收、地面資料人機介面及動畫即時顯示…等驗證。

目錄

表目錄

圖目錄

第一章 緒論

1.1. 論文架構

本論文第一章述及自動化飛行載具的現況，並提出研究的動機及希望能達到的目的，第二章論述飛行船主體架構及自動控制系統架構。第三章為核心模組主程式開發、馬達驅動，和系統公用程式初步規劃，第四章為週邊模組的開發，及地面站的架設和 Flash 介面設計，第五章是展現試飛時的狀態，分為地面測試與空中飛行測試。最後是本研究案未來的期許與展望。

1.2. 研究動機

國外無人飛行載具，近年來應用層面越來越廣泛，然而在台灣飛行船的運用受限於地面飛行控制器的有效控制範圍太小，或是人員操縱問題，以及許多外界環境變因，讓飛行船的運用受到許多限制及顧慮，尤其在人工操縱失控時，飛行船無法自主飛行及尋找合適地點安全降落，易造成飛行船失控而失聯，嚴重者甚至造成飛安問題。因此本研究動機即是要針對這些問題，開發一套能夠讓飛行船自主飛行的控制系統。

在完成飛船自動飛行控制系統後，可以利用飛船雷達截面積小(RCS)的特性執行國防上的任務如軍事偵察、情報蒐集等。另外在商業或民間用途上可作為廣告宣傳、溼度溫度監測、空中拍攝、環境監控等。

1.3. 目的

1. 飛行船自主飛行

本次專題最主要的目標，即為飛行船自主飛行，希望可以在輸入目標經緯度設定之後，控制系統即能自動航向該目標，到達目的地。

2. 讓飛行船的控制不再只有人工手動模式的單一控制方式

傳統飛行船皆為人工手動操縱，本專題要讓飛船具有自動與手動兩種飛行模式，並依據任務需求或現場狀況，進行手動或自動模式之切換。尤其在人為操作不當時，可以啟動自動降落功能，可讓飛行船安全降落。

3. 突破現有飛行船僅能在某一小範圍航行。

本專題在自動飛行時，可突破控制器操縱範圍的限制，達到在任意地點皆可航行的目的。

4. 及時動態飛行資料傳達地面站

本研究計劃在飛行船航行過程中，可以將經緯度、高度、溫度、航向、船身俯仰、偏傾等相關資訊，並記錄於地面站伺服器中，如同即時的飛航黑盒子般。

5. 地面即時模擬飛行動態

為解決當飛航高度過高，或飛行船在可視距外時，也能夠於地面端清楚檢視飛船目前飛行狀態。

第二章 飛船及自動控制系統設計

2.1. 飛船架構

2.1.1. 前言

本專題所使用的飛船主體，是由天興化工所提供的氦氣填充式飛船，利用 APP-026-3X 母板控制板作為整個自動化主架構，並搭配自製子板來銜接各種周邊配備，將各周邊設備所輸入的資訊送到主板做運算，達成自動控制的目標。

2.1.2. 船體操控及動力概述

本次專題所使用之飛船為天興化工所提供，飛船前進的結構，係藉由載台左右兩側的直流馬達轉動葉片，產生前進推力，使船體向前飛行。而飛船的上升與下降，則是透過固定葉片的軸承上下轉動，來控制俯仰角，進行上升與下降的動作。而軸承轉動的動作則是由升降伺服馬達負責控制。有關飛船轉向機構有三種版本，除了船體大小之外，主要差異在轉向模式的不同，另其主推力馬達也在最終版本中，有了大幅度的變更。

1. A 型船體：是將推進馬達架設於一轉向伺服馬達之上(如 圖 2-1)，因此可做出幅度左右各 75 度的風向吹送，而利用船尾螺旋槳吹向不同角度可達成船身角度的偏移，此種轉向方式優點是偏移的角度可以控制，且在角度零度時(直往後吹)，可作為船身前進加速使用，但經由多次試飛後，發現轉向能力不足，在風大時，或者是需要轉比較大角度彎角時，過慢的反應速度會造成路徑的偏移量加大，且於手動(遙控器)操作模式時，控制成效也不盡理想，因此此種轉向方式在幾次試飛之後，與廠商協議決定更換成另一種轉向模式，希望能改善其控制性。

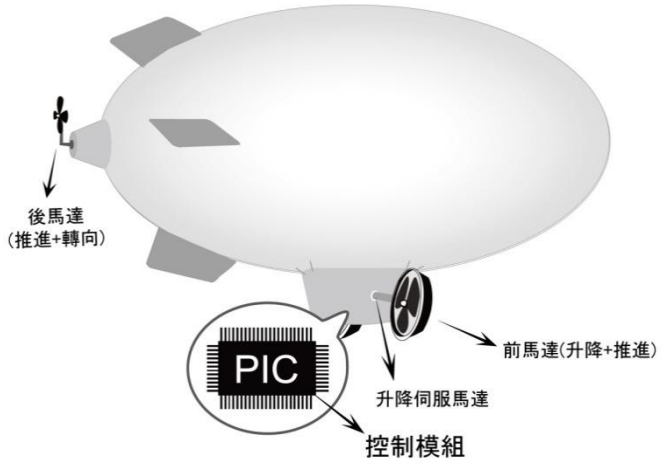


圖 2-1 A 型船體

2. B 型船體：是將一碳刷馬達，固定於下尾翼中間，利用碳刷馬達交換極性即可達成正反吹送的特性，來達成轉向功能，另外增加垂直襟翼，加強轉向效果(如圖 2-2)，此種轉向模式的優點是轉向迅速，相較於第一種轉向模式需等候伺服馬達將螺旋槳帶到不同吹送角度才能轉向，此種方式只是改變螺旋槳正反轉，改變出力方向較快、較直接。而此種機構有一種先天上的缺陷，相較於第一種轉向模式的無刷馬達，碳刷馬達雖然可以做兩種轉向的切換，但正反轉出力卻有著明顯的輸出落差，這也在轉向速度上有了明顯的差異。

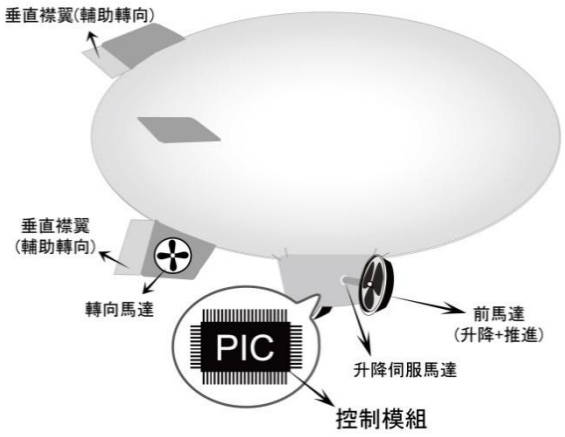


圖 2-2 B 型船體

3. C 型船體：此版主要著重於船體主馬達推進功能的改良，並增加水平襟翼，加強升降效能(如圖 2-3)。過去船體在飛行時，常常會遇到風大的狀況，若又處於逆風，則船身要朝目標前進則是會更為困難，且飛船重量輕，受風面積大，很容易被風所影響，停滯不前，這是很不樂見的情況。而幾次試飛，有時候會面臨障礙物的情形，為了避免船體受

損，通常會切到手動(遙控器)模式，將船體駛離障礙物，但風大時連手動都很難駕馭船身動態，更別說自動飛行，因此再度和廠商討論，最後改良為此種版本。

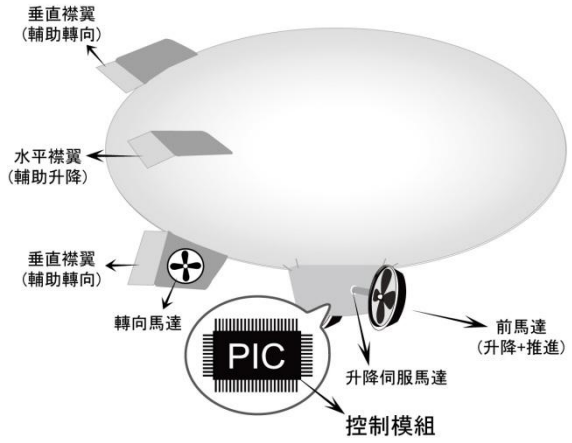


圖 2-3 C 型船體

除了提升馬達效能外,也加強了電力的輸出,並且將原本的三翼式螺旋槳改為四翼式，且翼長更大，可帶來更多的風力吹送，另於船尾四個尾舵後方，加上可動式襟翼，且分別由四顆伺服馬達控制，左右水平襟翼可用來輔助船體上升與下降，上下襟翼則與後方轉向伺服切換器共用一組 PWM 訊號，輔助船體的左右轉向，在此版本的演進後，船體與控制模組的搭配已有很大的成效，且另外還有襟翼的輔助，因此整體船身控制的性能有大幅度的改善，包含飛行的升降，在多了襟翼的輔助後，更為迅速。不過雖然性能提升了不少，卻也不是沒有缺點，在搭載了更複雜的轉向系統之後，第一個面臨的基本變化即是船體重量的增加，這使得要填入的氦氣需要增量，才能順利地將船身漂浮，且重心位置需要重新調配；此轉向模式由於多了四組伺服馬達，在 PWM 訊號的規劃上也更為複雜，且電力耗損也有所增加，會縮短飛行時間。

同時此版轉向機制，在一次的意外下有不一樣的發現(見 6.266.266.2 節)，即使主轉向馬達失效，於風大的環境下，單靠轉向襟翼的作動，也可以達到轉向的功能，雖然轉向較慢，但轉向角度變化卻比較緩和，比較穩定，算是意外下的收穫，雖然先前有預想到，但卻沒意料到會經由意外來獲得驗證。

2.1.3. 控制模組

控制模組母板是使用 Microchip 公司所生產的 PIC CPU 開發板(圖 2-4)來開發，開發板已內建下列基本周邊模組：

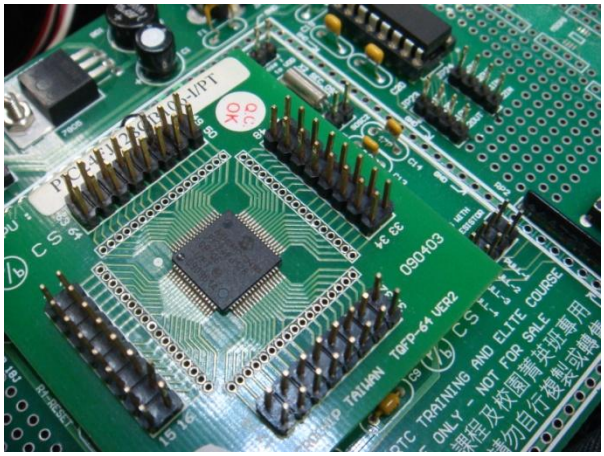


圖 2-4 PIC 64pinCPU

1. LCD：可用來顯示各種基本數值，以及 CPU 運作狀況，是控制模組運作狀態的主要參考依據(如 3.3.2)。
2. RS232 IO(含 MAX232 準位轉換晶片)：本模組所整合的 GPRS 以及 GPS 模組都是透過 RS232 通訊埠來進行溝通，其中 GPS 因為訊號準位(5V)與 CPU(3.3V)不同，因此需經過 MAX232 做準位的轉換，方可提供 CPU 進行運算。
3. 基本按鍵開關與 LED 燈號組：開發板上佈有四組基本按鍵開關以及四顆分雙色的燈號提供使用，利用不同的按鍵搭配不同的程式碼，再依照需求可以達成參數的切換或者是顯示資訊的交換，以提供使用者與開發板之間的互動，另四顆 LED 也具負責顯示目前控制板運作狀態的指示功能，目前設置為旗標的狀態顯示，提供快速的狀況表達，協助使用者了解周邊是否運作正常(如 3.3.2)。
4. 燒錄器除錯模組：提供電腦端與 PIC 端的連結，可經由此模組作程式燒錄，或直接經由電腦於 PIC 上做模擬除錯動作。

除了基本開發板以外，本研究另有自製子板來搭載各式周邊設備，其包括：

1. 電子羅盤 (Compass)：用來感知船體三維的動態，包含：
   - 船首方位 (Heading)，協助 GPS 定位，以便計算，此功能為系統之最關鍵功能，目的在立即得知飛船目前船首指向，不必靠 GPS 的位移距離，來判斷船首目前指向，進而修正飛行方向，此模組更簡化程式的撰寫，讓程式執行上效能更為提高。
   - 船身仰俯角 (Pitching)，藉由此資訊可以得知船體要爬升到指定高度時，船體上升的角度偏移，並可搭配螺旋槳出力方向的修正，此外也判斷船身是否處於上升或下降狀態。
   - 船身左右滾角 (Rolling)，感測目前船身是否處於不穩定狀態。
2. 全球衛星定位系統 (GPS)：此設備可在有限的誤差範圍內定位出飛船目前的所在位置，以及判定與目標點的距離，其中包含時間、高度、經度、緯度等資訊。

3. 通用封包無線服務技術 (GPRS)：將飛行資訊透過網路傳回由 Java 撰寫的地面監控站，並將所有資訊存入資料庫內。另有 Flash 動態展示介面，內嵌 Google Map App，經由網頁呼叫將資料庫參數讀入，並轉成動畫即時顯示目前飛航狀況，包括船體各種參數、馬達輸出、飛行動態、Heading、Pitching、Rolling，以及在 Google Map 的位置等資訊，若必要時，地面站可下達指令，經由 GPRS 傳送至飛行中的控制模組，以達到修改主板參數之功能，藉此在緊急時取得控制權來操控船身動作，甚至更改目標點，具有緊急備援的作用；目前主板由 CPU 直接輸出四組 PWM 訊號，分別控制各個伺服馬達；另有兩顆繼電器，搭配一組伺服馬達直接觸動微動開關來達成手動(遙控器)及自動模式的切換，以備飛船緊急狀態時，可透過繼電器的切換，將各馬達的輸入訊號，由主板轉為遙控接收機訊號(遙控器控制)。

2.1.4. 與他校類似研究比較

本研究與他校以飛船為載具之研究比較(如表 2-1)，雲林科技大學無人飛行載具的製作與導航[1.]，船體較小承受不住風速，因此飛行應用主要以室內為主。而成功大學航太系無人飛船自主性控制設計與實現[2.]，是在傳統的 RC 控制外設計一電腦飛行控制系統，且以手動控制模組為主。

表 2-1 與他校類似比較表

名稱 / 項目	本專題	成大	雲科大
船體大小	9 公尺	14.3 公尺	4 公尺
動力來源	電池馬達	汽油引擎	電池馬達
手自動切換機制	微動開關	FPGA	無
主要控制模組	PIC 開發板	PC586 電腦	AT89C51 單晶片系統
主螺旋槳葉片數	4	2	2
轉向方式	螺旋槳加襟翼	襟翼	襟翼

通訊方式	GPRS	無線電	無線電
研發年度	2010-2011 學士	2000-2002 碩士	2002-2004 碩士
軟體	自行開發	PC-Based 系統	C++Builder+PComm Pro
實際飛行測試	✓	模擬飛行	室內飛行測試

2.2. 自動控制系統架構

2.2.1. 系統主架構

圖 2-5 為系統主架構圖，系統使用元件及性能規格如表 2-2，因本研究之主要目標為自主飛行，故控制板必須做出目標判斷，根據目前船身的所在位置以及方位做出適當的船身控制，包含升降、轉向、前進等動作，對於目的地的距離以及目前動作輸出與實際反應做出最快速的回饋動作以保持船身穩定。

為了即時能掌握船身動態，方位的判定，以及飛船狀態的及時監控，控制板銜接了各項裝置來達成此一目的；利用 GPS 可取得目前飛船所在之座標，Compass 可取得目前船身的 Heading，以利方位判定：另感測船身 Pitching 以及 Rolling 可助於船身的穩定性控制與升降控制，另加上了 GPRS 可即時回傳飛行船的所有相關資訊至地面接收站，其中包含 Compass、GPS、以及目前控制板控制狀態及馬達輸出狀況等等的相關資訊。

此外，於地面接收站，是由 Java 所架設的 Server 接收飛船經由 GPRS 透過 Internet 所傳送下來的封包，將其解碼後分別將資訊交由 Flash 製作而成的監控介面來即時監控所有飛行船的飛行資訊，同時 Java Server 端也具備向上傳送控制指令的功能，可由地面端軟體控制介面送出控制訊號，或者即時更換目的地座標，來達成雙向傳送指令的目的。

表 2-2 使用元件列表

軟硬體	型號	資源
PIC24F	PIC24FJ128GB106	Microchip 研習取得
GPS	GARMIN GPS15xL-W	購買
GPRS	Create GPRS4SIM100	學校 ARM 的子板
電子羅盤	HONEYWELL HMC6343	購買
資料庫	MYSQL	MYSQL 建於學校之網頁伺服器
操作介面	Windows 視窗介面	JAVA、FLASH
飛船主體	6.5M、9M 飛行船	天興化工公司提供

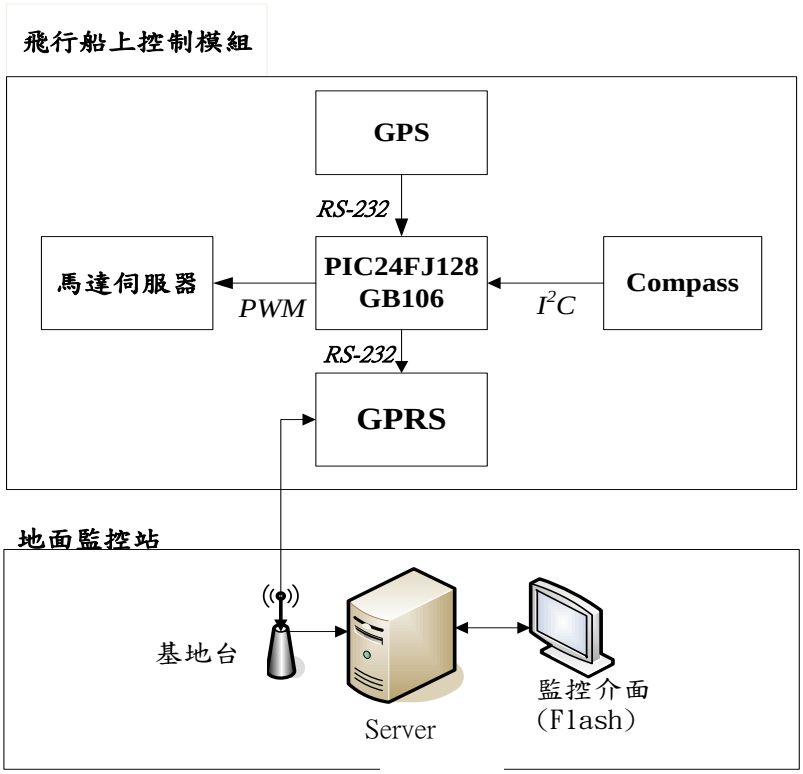


圖 2-5 系統主架構

2.3. 馬達動力控制設計

2.3.1. 馬達控制演進表

使用 PWM 模組控制馬達而經過不斷的測試與改良，實驗出最適用於本專題馬達控制的 PWM 模組程式，表 2-3 是本專題馬達控制演進表。

表 2-3 馬達控制演進表

版本 / PWM	PWM1	PWM2	PWM3	PWM4
第一版	升降伺服馬達俯仰控制	轉向伺服馬達控制	前馬達推進控制	後馬達推進
第二版	升降伺服馬達俯仰控制	轉向馬正反轉控制	前馬達推進控制	
第三版	升降伺服馬達俯仰控制	轉向馬正反轉控制+垂直襟翼伺服馬達擺動控制	前馬達推進控制(輔助轉向)	水平襟翼伺服馬達擺動控制(輔助升降)

2.3.2. 馬達動力控制設計

第一版：推進馬達架設於一轉向伺服馬達之上，如圖 2-6。

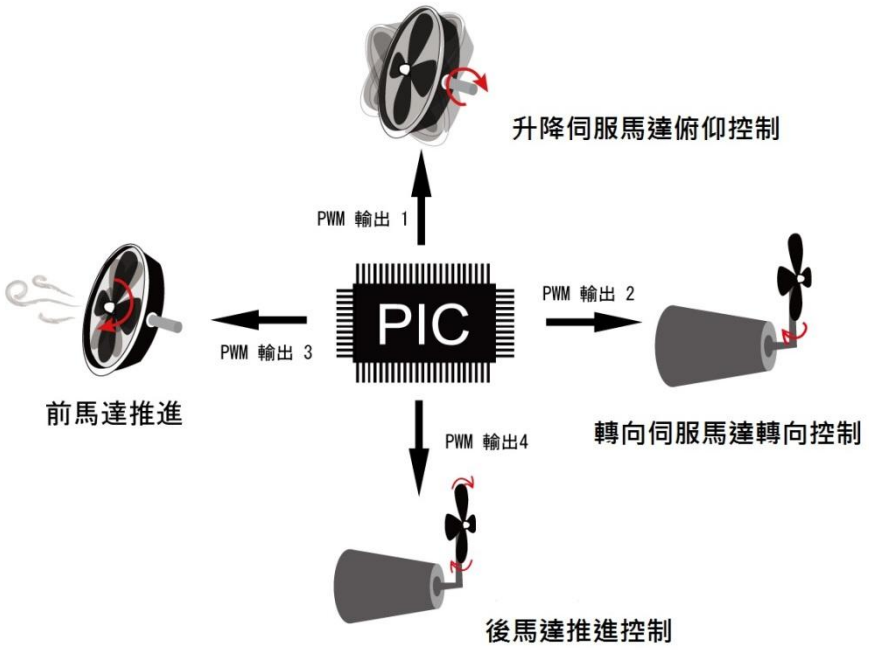


圖 2-6A 型船體馬達控制

第二版：轉向馬正反轉控制，如圖 2-7 **錯誤! 找不到參照來源。**。

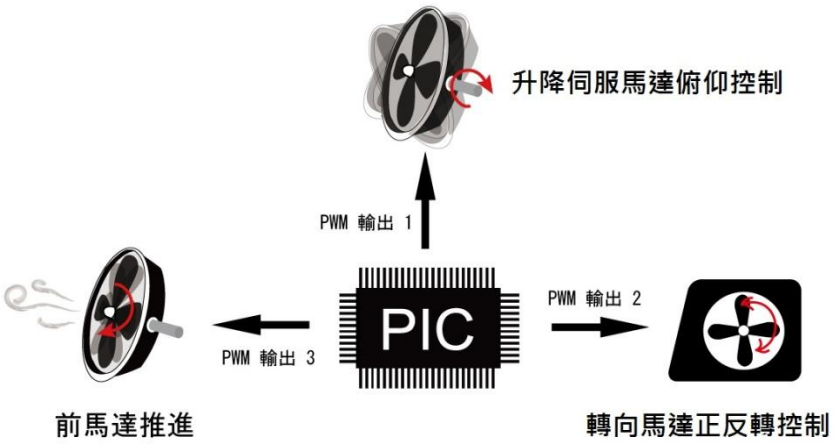


圖 2-7B 型船體馬達控制

第三版：兩組轉向舵，後轉向舵與後升降舵，前轉與後升降舵相連接，如圖 2-8。

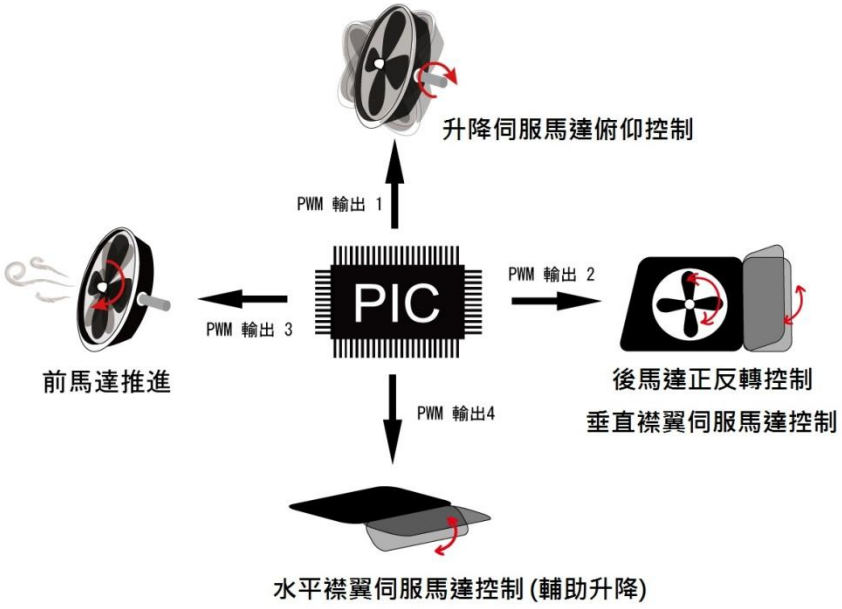


圖 2-8C 型船體馬達控制

飛行控制系統－PIC 母板與周邊整合架構圖，如圖 2-9。

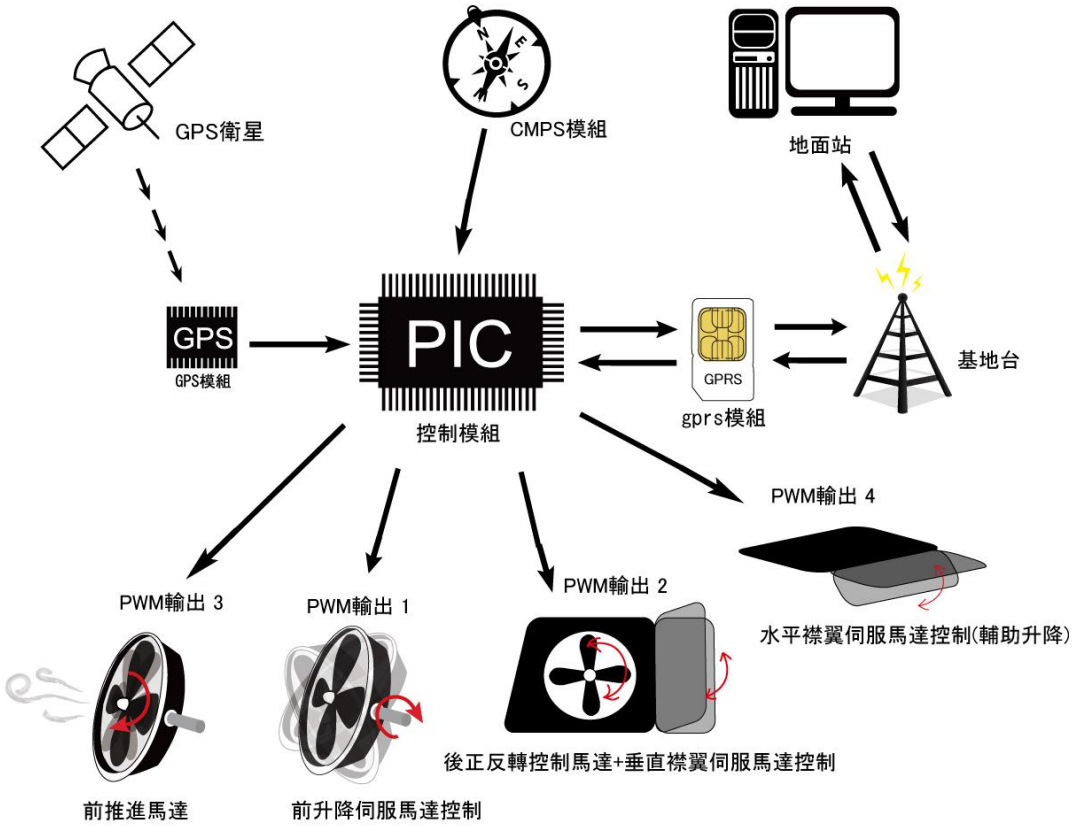


圖 2-9 整合圖

2.4. 系統手、自動切換模式

本研究所使用之手自動切換，係利用一伺服馬達直接觸發微動開關導通繼電器來達成 pwm 訊號源的切換，而此一伺服馬達之訊號來源為遙控接收機，切換動作可經由遙控器來達成。如圖 2-10 示意。

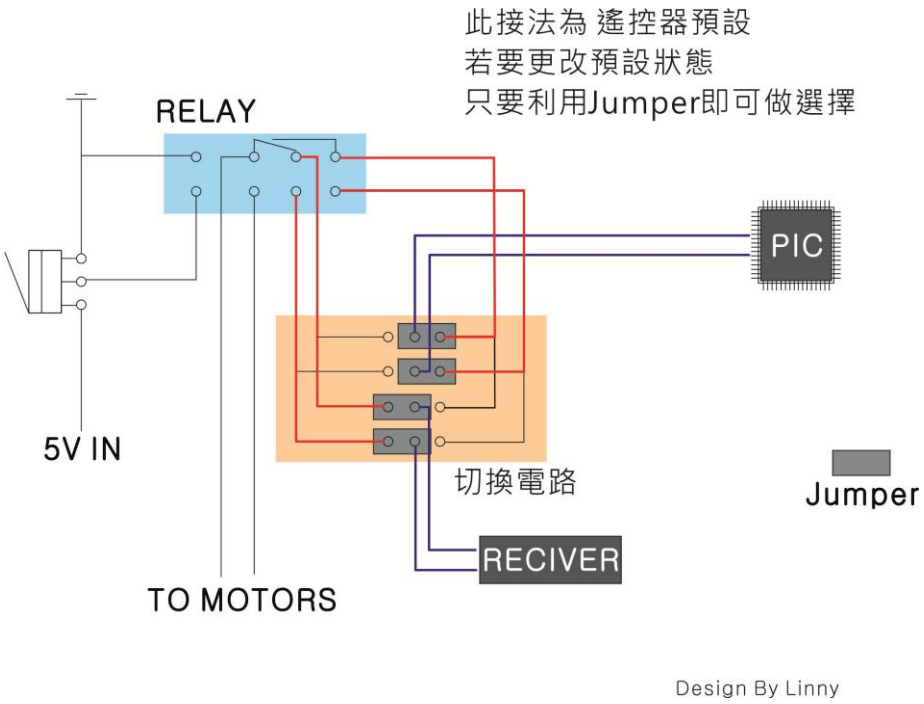


圖 2-10 切換電路示意圖(雙組切換)

2.5. 子板電路設計

2.5.1. 設計概述

本研究所使用之開發模組，需搭配各式周邊運作，初期於測試階段時，係利用杜邦端子將各個周邊做連結，包含訊號傳遞及電源供應，在各式周邊皆正常運作後，顧及日後操作方便以及線路穩固，開始有進一步的連接方式。

a. 連接線模組化：

由於線材眾多，初期是以杜邦一條一條做連結，此方法將各個模組所用之訊號線與電源線模組化後，只要將周邊所對應之連接線街上，便可順利安裝模組。此種方法雖然使用連接上較之前快速，但線材的雜亂程度卻沒有明確的改善，在模組運作開始時依樣要一一為周邊做連結，使用上還是不太方便，也不太穩固。

b. 子板連接(第一版)：

相較於將連接線模組化的方式，此方法直接將所有連接線部屬在 PCB 板上，利用杜邦基座直接對應 CPU pin 腳插上，再將各功能 pin 直接拉出並也設置杜邦插腳，以利周邊安裝，讓更為快速，不用再插線，不會有線材雜亂的問題。

此外，在製作此子板時加入了手自動切換電路，利用伺服馬達做動，可觸發微動開關，直接導通繼電器驅動兩組繼電器做 4 port PWM 訊號源(手自動)的切換，同時有一組繼電器擔任第五顆感應開關，將 3.3V 之訊號送入對應 pin 腳，使系統能偵測到目前的運作模式。

由於此板線材均使用 0.3mm 單芯線做佈線，當線路複雜時，容易纏繞錯亂，當有問題時不易除錯，且由於周邊是利用魔鬼氈做固定，雖然穩固，但依然不慎理想，尤其在拆卸時更是不便，此版本只用於地面測試。

c. 子板連接(第二版)：

此版改進了上一版的缺點，所有連接線均以排線做連接，並利用較細的線材，讓整體看起來清爽許多且容易除錯。同時第五顆開關(手自動狀態感應)也由繼電器改成了光耦合開關，除了減輕一個繼電器的重量外，更降低了耗電量，對於嚴苛的用電環境來說，是一項重要的改進。對於周邊裝置的固定方式，也由魔鬼氈改為塑膠螺絲固定，除了較為穩固之外，拆裝也變得容易許多。

第一版子版在操作時，因為沒有電於開關的設計，在需要暫時關閉電源時，須從電源線下手，容易造成短路的風險，因此在此版設計中，於主電源會路上加了一指撥開關，若需暫時切斷電源或者重開機，只需利用開關來切斷回路即可，不僅使用上較為便利，同時也較為安全。

上一版的地面測試中，在 PWM 馬達接上線後發現主版便有電源反應，有時會造成當機的狀況，檢查後發現是來自於遙控器接收機的電源，因此在 PWM 訊號輸出迴路上均加了二極體來防止電流逆向流入 CPU 內，以阻絕當機的風險。但第三四組的 PWM 訊號由於經過二極體後便無法被變電器所接收，故另設置跳線繞過二極體方能正常運作。

2.5.2. 電路圖

2.5.2.1 子板第一版電路圖

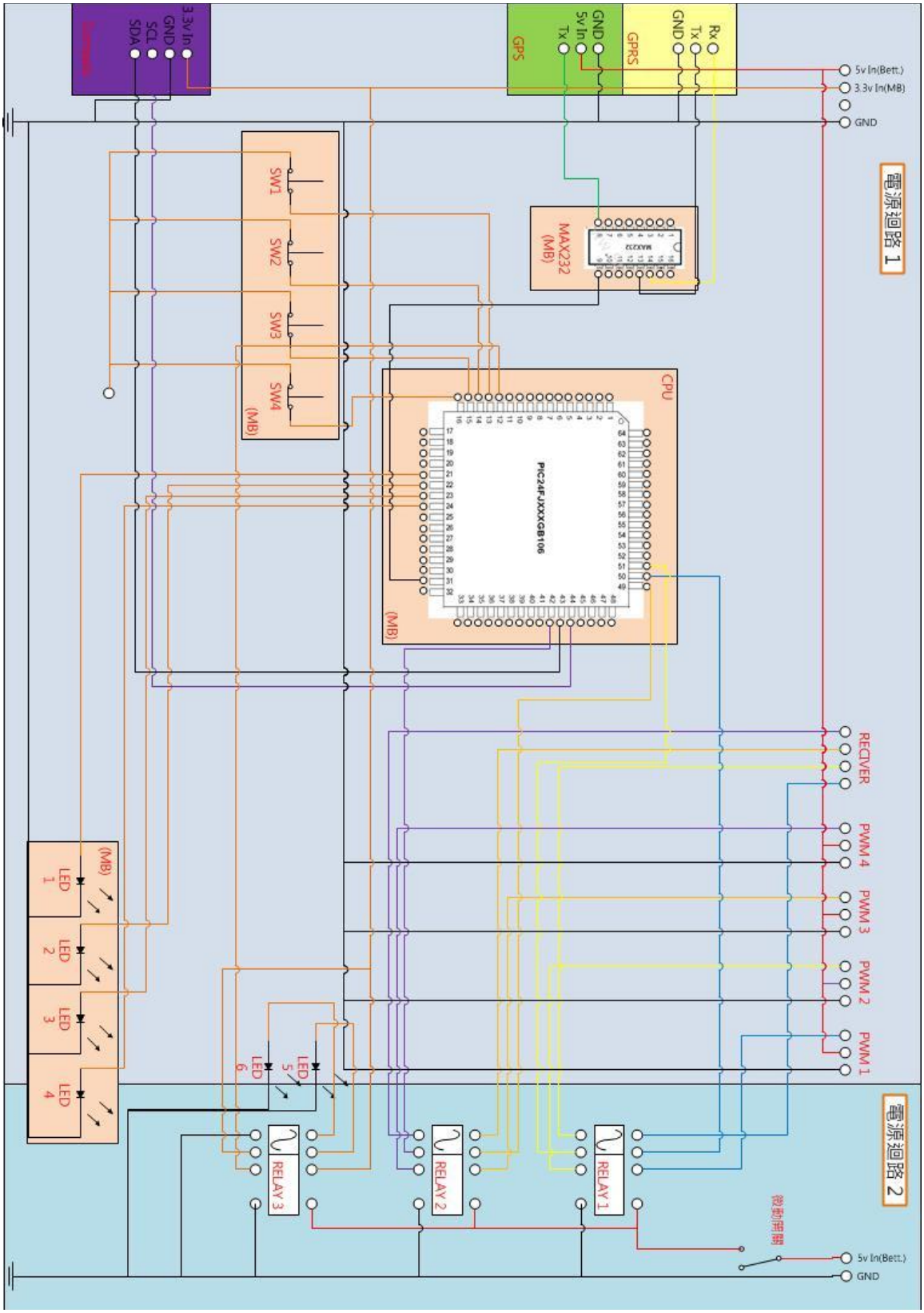


圖 2-11 子版第一版電路圖

2.5.2.2 子版第一版本體

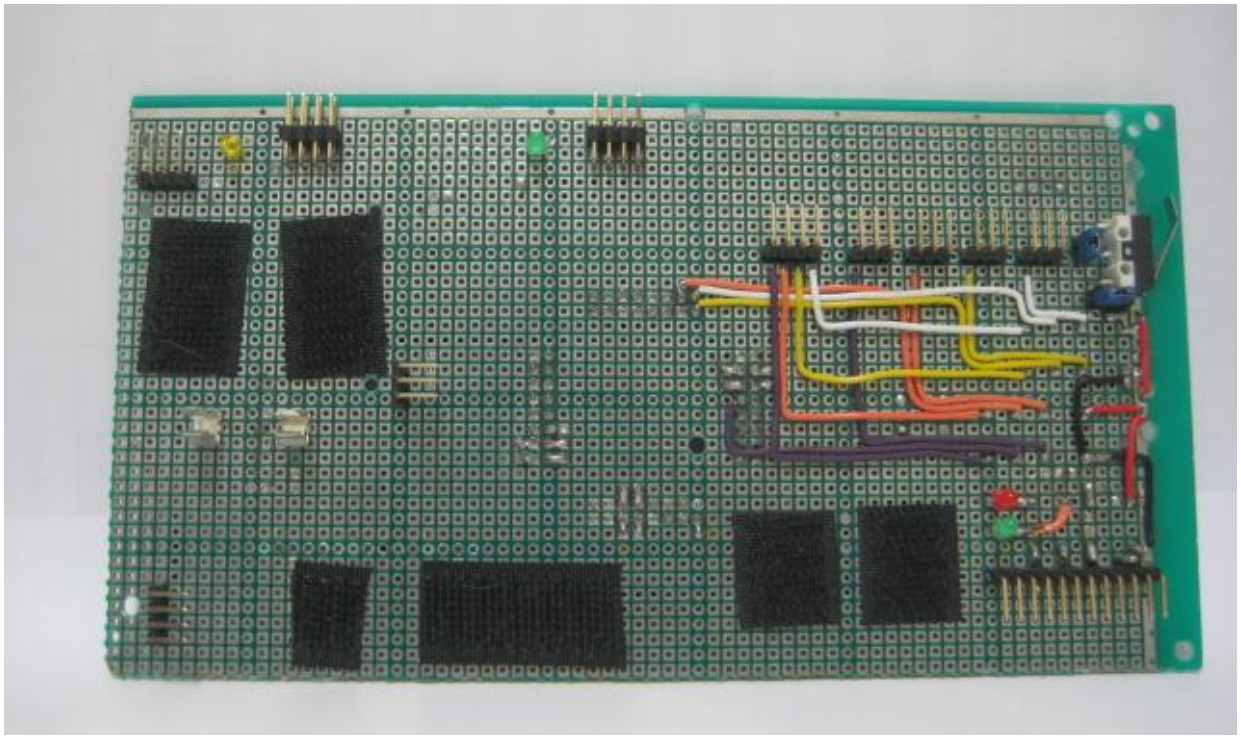


圖 2-12 子板第一版實體正面

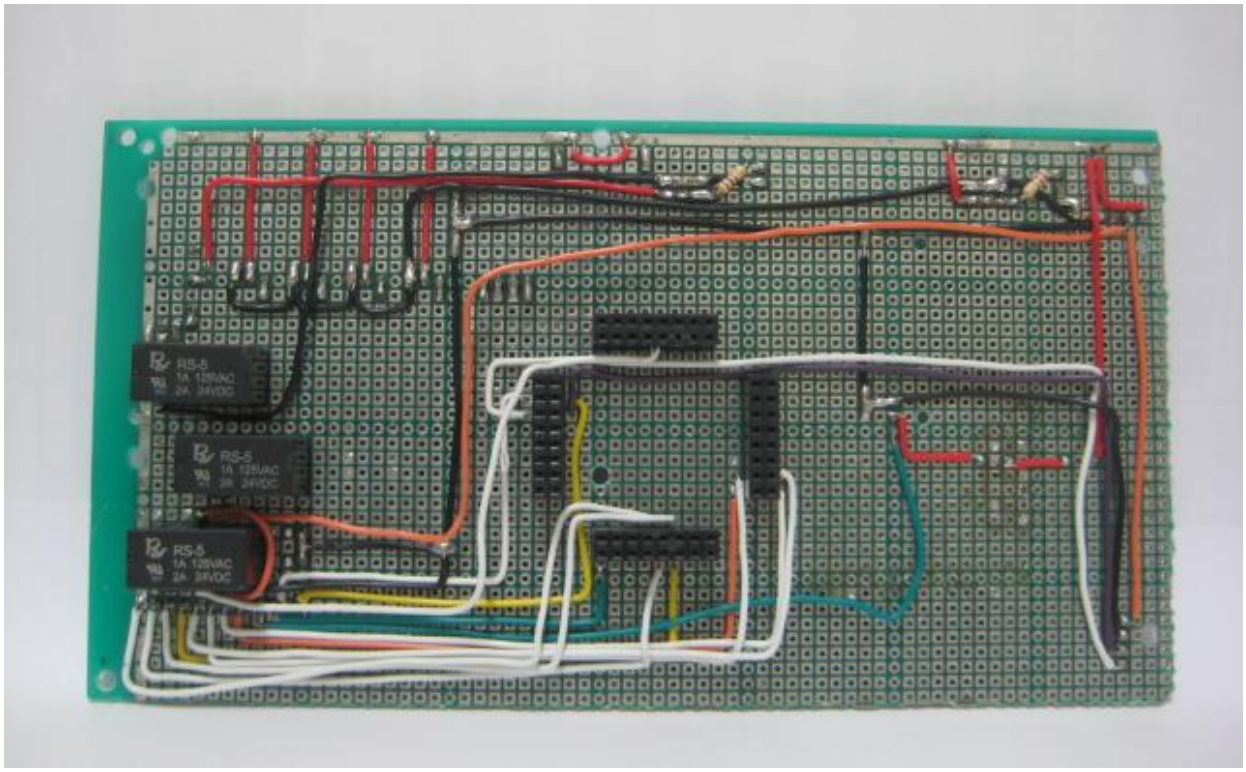


圖 2-13 子板第一版實體背面

2.5.2.3子板第二版電路圖

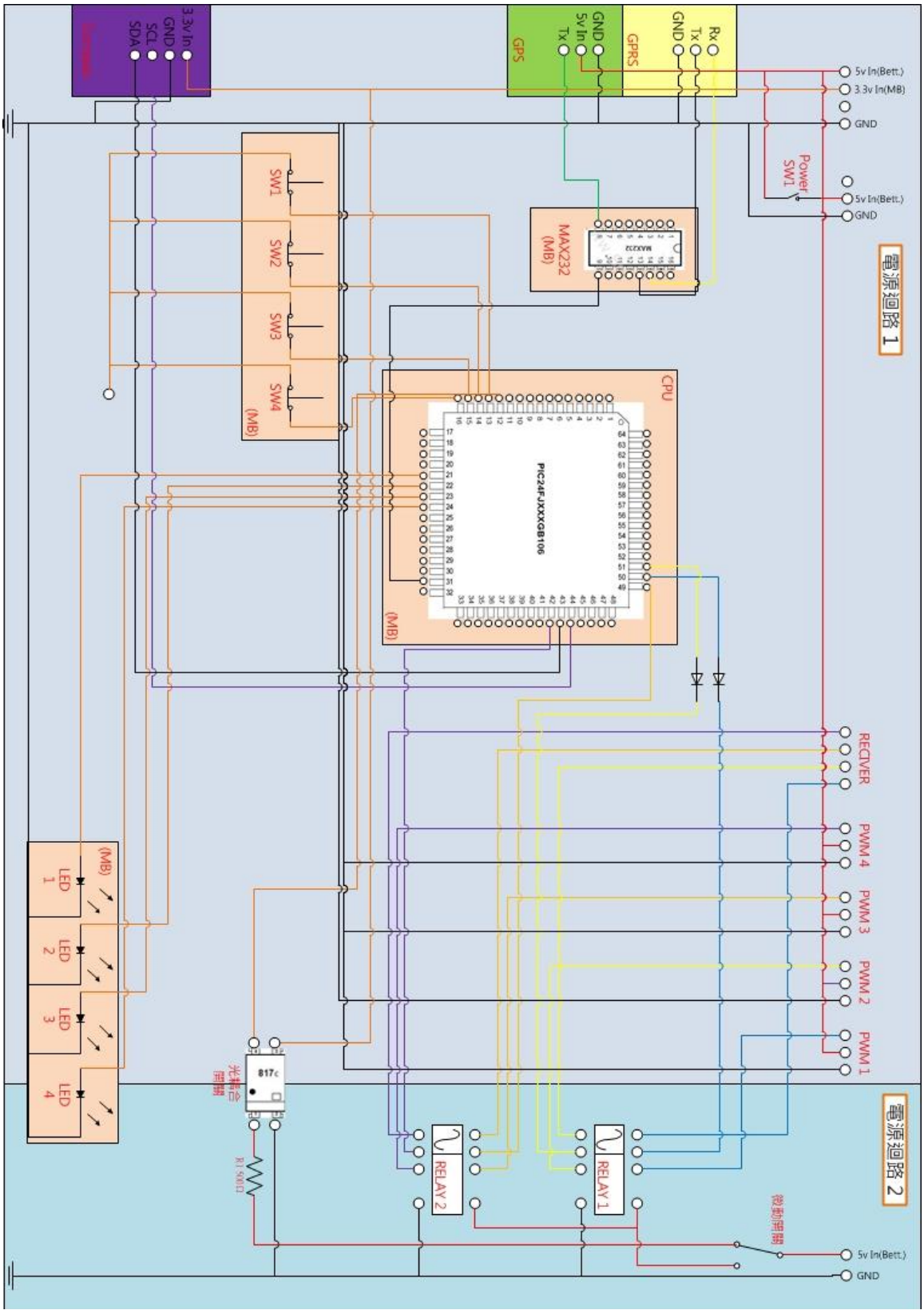


圖 2-14 子版第二版電路圖

2.5.2.4 子板第二版本體

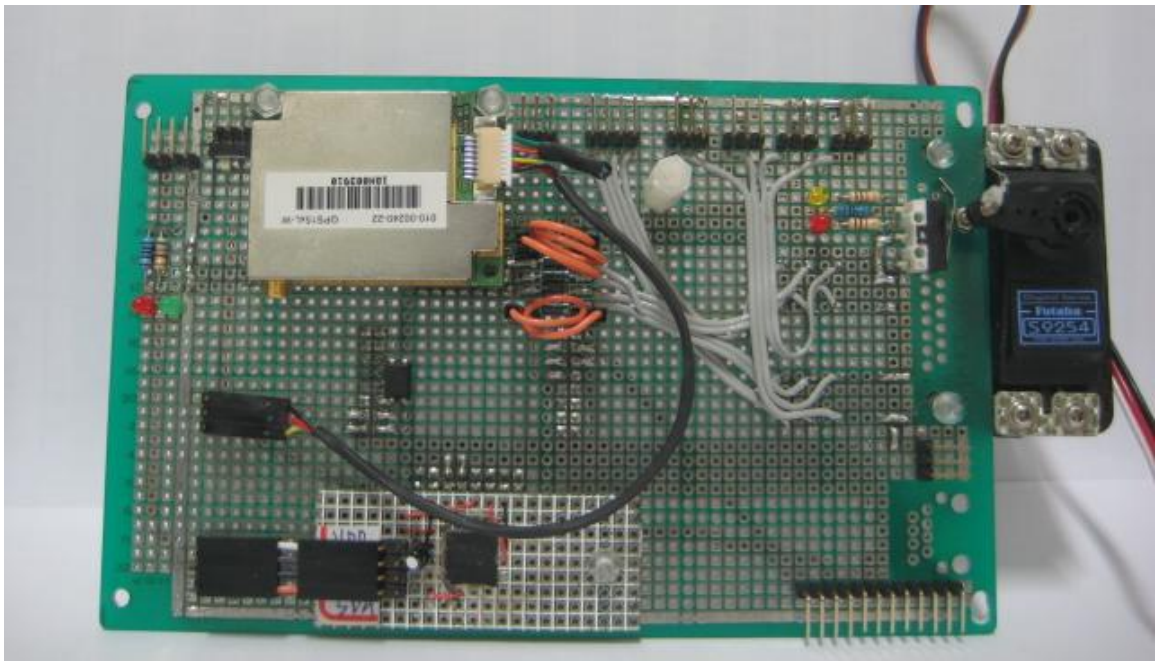


圖 2-15 子板第二版實體正面(含周邊)

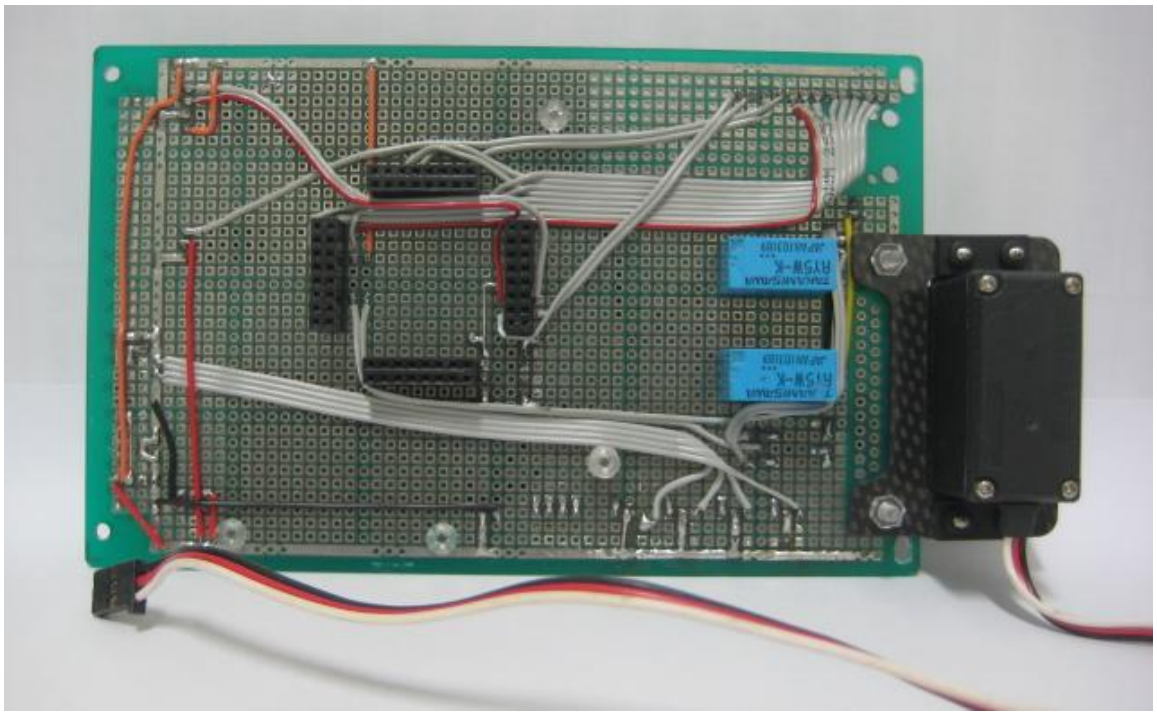


圖 2-16 子板第二版實體背面(含周邊)

2.6. 系統電力設計

本研究之控制模組電源迴路有三組：

1. 主電源迴路：提供主要電源，供應 PIC 主板運作，再經由主板提供 5V 及 3.3V 電源給周邊設備，包含 GPS(5V)、COMPASS(3.3V)。

2. 繼電器電源迴路：由於繼電器在切換之瞬間電流量會造成整個電源迴路崩潰，故將兩個繼電器之電源供應獨立自主，使其不影響整個主模組的運作。
3. GPRS 電源迴路：GPRS 由於耗電量較高，故獨立提供一電源供應其使用。

2.7. 地面 SERVER 系統

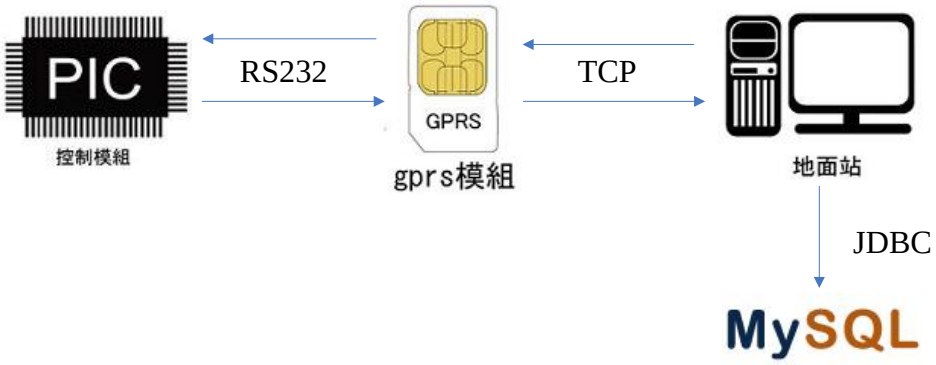


圖 2-17 系統圖

PIC 將資料透過 RS232 傳輸到 GPRS 將透過 TCP 原理把資料傳到地面監控站，地面監控站再以 JDBC 連接，把資料寫進資料庫，如圖 2-17。

2.8. 動態展示系統

動態展示系統所使用的 Flash 軟體本身無法直接與資料庫做連線，因此我們需要透過 PHP 來協助，按照 Flash 抓取資料的格式，將資料印出在網頁上，Flash 連結網頁抓取所需的資料，便可以完成此項動作，如圖 2-18。

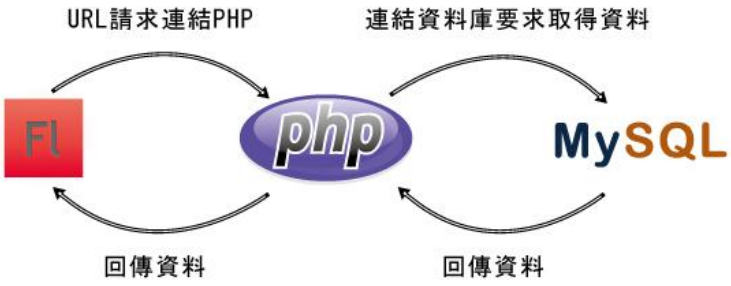


圖 2-18 動態展示系統

2.9. 系統開發期程表

表 2-4 進度甘特圖

識別碼	項目 / 時間	2010年 01月	02月	03月	04月	05月	06月	07月	08月	09月	10月	11月	12月	2011年 01月	02月	03月	04月
1	決定題目																
2	選定模組、器材(Compass、GPS、GPRS)																
3	PWM訊號接收模組測試																
4	GPS程式測試(一)																
5	地面監控程式開發																
6	I2C程式測試(DS1307時間晶片)																
7	GPS模組測試(二)																
8	GPRS模組通訊開發																
9	飛船動態展示系統																
10	I2C模組測試(電子羅盤)																
11	硬體子板電路製作焊接																
12	飛行控制模組程式																
13	飛船模組與飛行控制模組結合																
14	無人自動化控制模組地面測試(未搭載船體)																
15	無人飛行船自動導航地面測試(推車搭載飛行船船艙)																
16	無人飛行船自動導航飛行測試 第一次試飛(下操場) 第二次試飛(下操場) 後轉向馬達:推力 狀態：以遙控居多																
17	無人飛行船自動導航飛行測試 第三次試飛(下操場) 後轉向馬達：正反吹 狀態：關通訊程式，成功自動飛行																
18	無人飛行船自動導航飛行測試 第四次(操場和校園大草坪) 後轉向馬達：兩組轉向舵，後轉向舵與後升降舵 狀態：成功平穩飛行																
19	專題論文資料整理與撰寫																

第三章　主核心模組

3.1.　PIC24F PIN 腳定義圖表

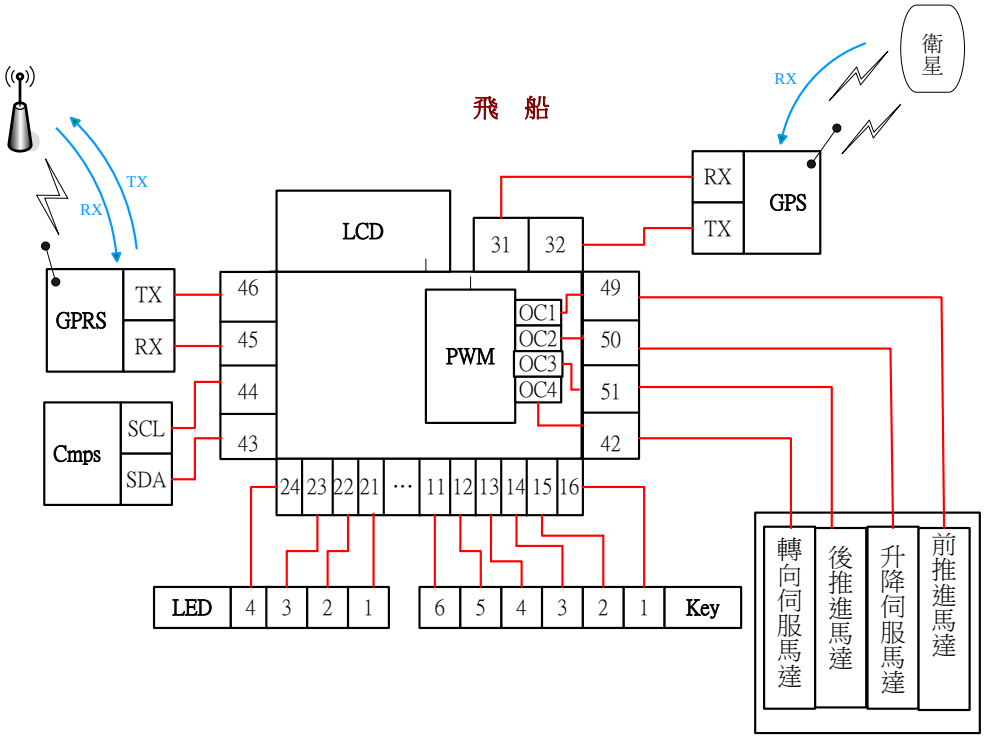


圖　3-1 PIN 腳定義圖表

對照圖　3-1

1. ＬＥＤ－ＯＵＴ腳
   輸出腳為：ＬＥＤ１：21、ＬＥＤ２：22、ＬＥＤ３：23、ＬＥＤ４：24。
2. ＫＥＹ－ＩＮ腳
   輸入腳為：ＫＥＹ１：16、ＫＥＹ２：15、ＫＥＹ３：14、ＫＥＹ４：13、ＫＥＹ５：12、ＫＥＹ６：11。
3. PWM－ＯＵＴ腳
   輸出腳為：ＯＣ１：49、ＯＣ２：50、ＯＣ３：51、ＯＣ４：42。
4. ＧＰＳ－ＲＸ－ＩＮ腳
   規劃腳位為：ＲＸ：31。
5. ＧＰＲＳ－ＲＸＴＸ－ＩＮ、ＯＵＴ腳
   規劃腳位為：ＴＸ：46、ＲＸ：45。
6. ＣＭＰ－ＳＤＡ、ＳＣＬ腳
   規劃腳位為：SDA：43，SCL：44。

3.2. 飛行船自動飛行系統主程式

3.2.1. ＭＰＬＡＢ軟體簡介

本系統的軟體開發是利用 MPLAB IDE。它是一套微處理控制器軟體研發作業平台，架構於 Windows 的作業環境，提供多視窗文字編輯功能(如圖 3-2)；可直接在 MPLAB 環境下直接撰寫、修改原始程式(Source Code)或進行 C 的編譯(Compiler)或組譯(Assembler)工作，產生微處理機可執行的 EXE 檔。使用方法見(附錄)。

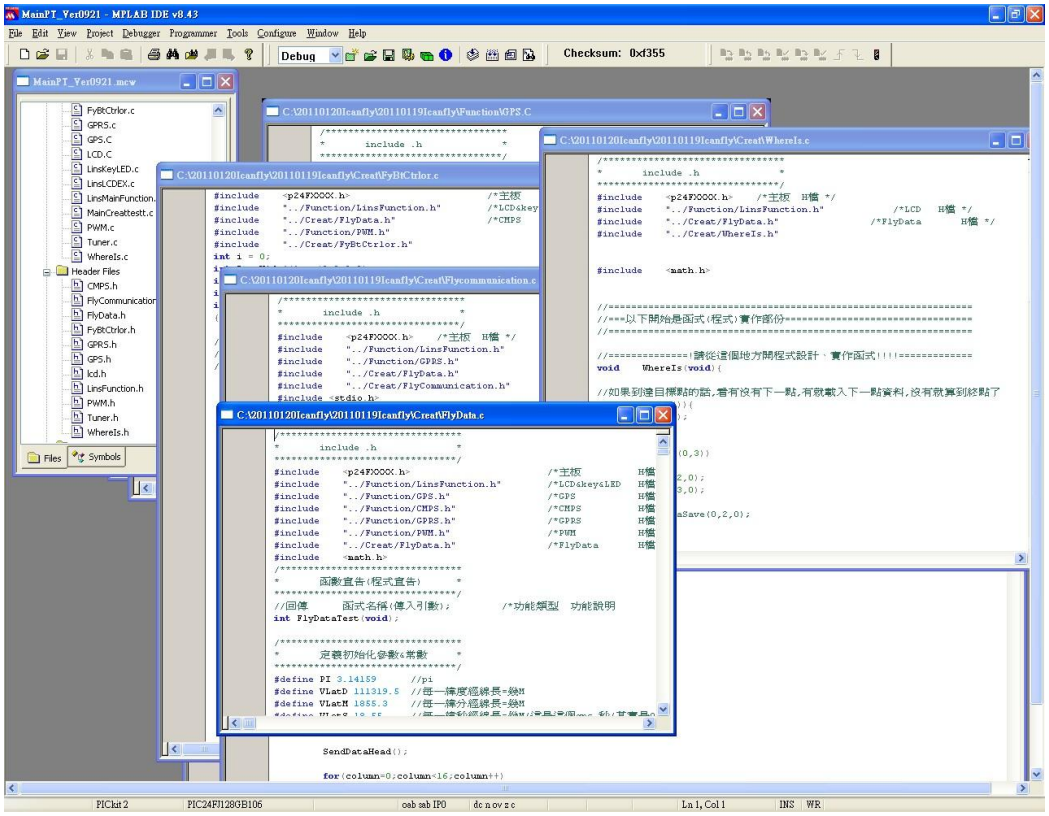


圖 3-2MPLAB 視窗介面

3.2.2. 自動飛行主程式流程

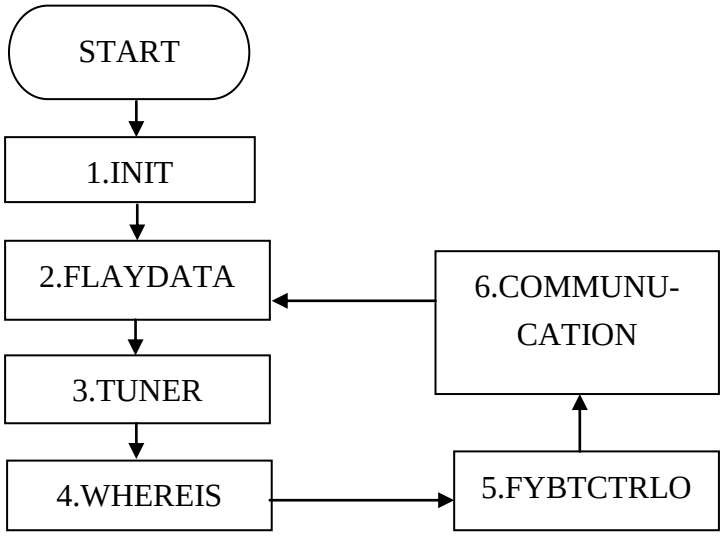


圖 3-3 主程式流程圖

飛行控制主程式，開始

1. 初始化 INIT
   各週邊模組基本初始化
   LED、LCD、KEY、GPS、GPRS、PWM、CMPS
2. 方向判斷 FLAYDATA
   方向判斷依據與動作指令下達方式
3. 微調 TUNER
   依飛行時天氣因素可在此設定需要增加或減少動力的方式
4. 目的地判斷 WHEREIS
   是否到達目的地
5. 馬達輸出 FYBTCTRLOR
   輸出 PWM 訊號於馬達
6. 地面通訊 COMMUNUCATION
   將飛行中所接收到ＧＰＳ資料與計算動作資料傳至地面ＳＥＲＶＥＲ，並接收回傳指令

3.2.3. 自動飛行主程式版本演進表

自動飛行主程式開發流程共經歷四大階段如圖 3-4，詳細更新內容如表 3-1。

第一階段：
公用程式開發時期，此階段將定義主板各 I/O、週邊相連接 PIN 腳，以及開發 LCD、LED、KEY 按鍵等基本開發功能需求。公用程式詳細開發說明請參閱 3.3。

第二階段：
模組程式開發時期，此階段將利用已開發之公用程式，對各週邊模組進行功能測試，並將其功能模組化，使其便於副程式中運用。各模組詳細版本說明於第 4 章。

第三階段：
副程式開發時期，此階段將本系統需解決之問題及需提供之功能進行切割，並依其先後順序進行歸類統整，分別撰寫副程式提供主程式所需解決問題之功能。詳細版本說明於 3.4。

第四階段：
整合測試階段，將以上各階段所撰寫之程式加以整合後行進地面測試及飛行測試，並依測試結果加以檢討以及對程式進行修改。整合測試結果於第 6 章詳加說明。

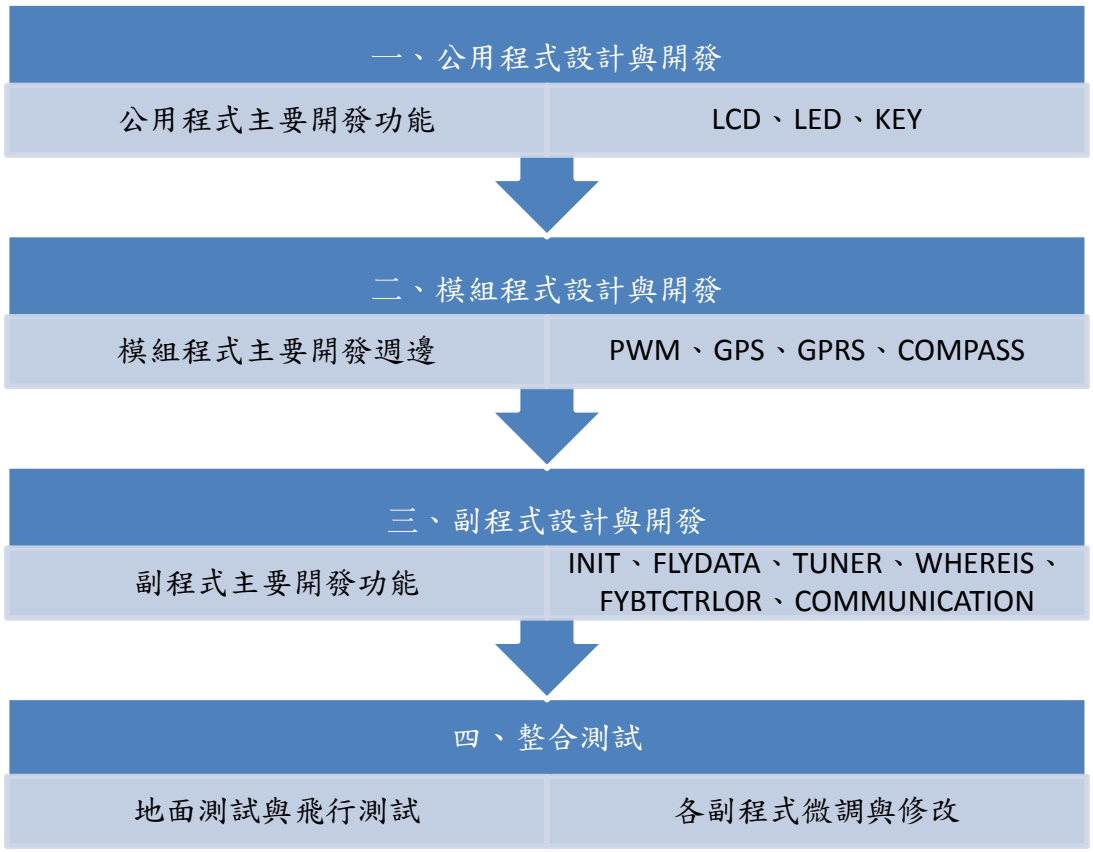


圖 3-4 開發流程

表 3-1 主程式版本更新表

版本	更新項目	新增需求
MainPT_Ver0712	InitAll、llStop、Show、Show7kAt、ShowAt、ShowERROR、ShowDelay、ShowSP、CleanLCD、LinsFunnyLCDL2R、LinsFunnyLCDR2L	ＬＥＤ燈控制、按鍵控制（增加程式開放的便利性）
MainPT_Ver0713	ShowNum、LEDCtrl、LEDON、LEDOFF、LEDChkFlag、IsKey、IsClick	開啟、關閉ＬＣＤ、ＫＥＹ、ＬＣＤ的功能（有時不需要使用）
MainPT_Ver0714	ShowEN、ShowDIS、LEDEN、LEDDIS、KeyEN、KeyDIS	ＬＣＤ印出不需宣告的字串、判別何按鍵被按下、等待任意或指定按鍵按鍵
MainPT_Ver0716	ShowLn、ShowDouble、WhichKey、WaitFor、	無正負號 CHAR 轉換
MainPT_Ver0731	ShowUnCh	
MainPT_Ver0801	加入 PWM 模組程式	
MainPT_Ver0816	加入 GPS 模組程式	
MainPT_Ver0820	加入 CMPS 模組程式	
MainPT_Ver0822	加入 GPRS 模組程式	
MainPT_Ver0921	整合所有副程式	

3.2.4. 飛行船自動飛行系統程式階層架構

本系統建立於 PIC24FJ128GB106 板上，程式階層架構如圖 3-5 所示。

第一層：頂層即為本自動飛行控制系統。

第二層：系統主程式，此層所載入的函式包括 PIC24FJ128GB106 主板所提供的函式，以及本系統所定義的副程式。

第三層：副程式，此層程式與各週邊模組程式相連結，並利用週邊模組所提供之資料加以運算，得以處理、解決主程式所切割之各種問題。

第四層：模組程式，此層模組程式將直接處理週邊模組如 PWM、GPS、GPRS、Compass 之資料傳輸與訊號輸出、入。

第五層：公用程式，此層所提供的功能包括 LCD 的顯示、LED 明、暗控制與按鍵開關的偵測，其中按鍵開關為控制手自動切換所使用，LED 燈為確認各週邊連結成功所使用，且於 LCD 上顯示目前系統狀態，並提供本系統開發階段的開發設計與除錯使用。

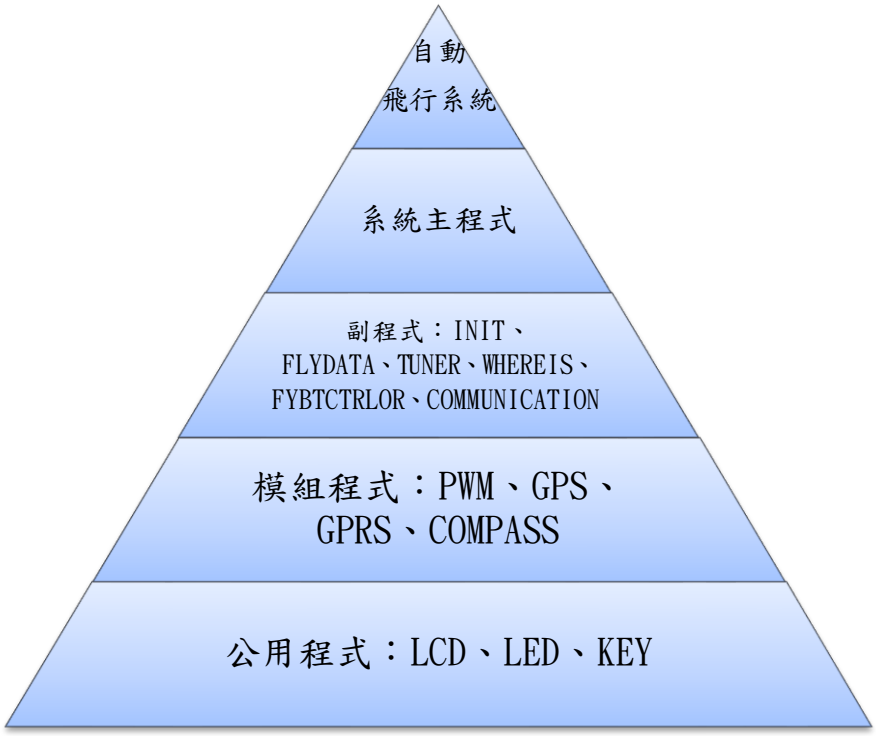


圖 3-5 程式階層示意圖

3.3. 公用程式

3.3.1. 系統公用程式規劃

1. ＳＴＯＰ－停止ＣＰＵ動作

   需求：讓程式在該行什麼事都不做

   設計：AllStop();

2. ＬＣＤ－顯示字母、符號於ＬＣＤ

   需求：印出字串、印出變數、印出發生的錯誤

   設計：Show();、ShowInt();、ShowERROR

3. ＬＥＤ－ＬＥＤ燈控制

   需求：用讓 LED 亮、暗，用 LED 確認旗標值

   設計：LEDON、LEDOFF、LEDChkFlag();

4. ＫＥＹ－按鍵控制

   需求：等待按鍵、尋找被按下的鍵、是否按住、是否按一下。

   設計：WaitFor() WitchKey() IsKey() IsClick()函式

3.3.2. 公用程式執行測試

1. LCD 執行測試－於 LCD 上印出ＬＣＤ　ＬＥＤ　ＫＥＹ　ＴＥＳＴ，如圖 3-6。

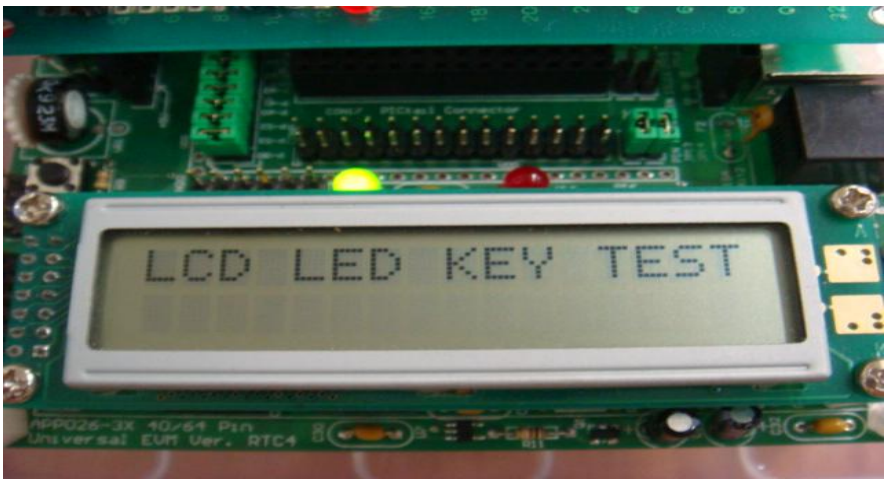


圖 3-6LCD 執行測試

2. LED 執行測試－按下ＫＥＹ１時ＬＥＤ１亮，如圖 3-7，
放開時暗，如圖 3-8。

3. KEY 執行測試－按下ＫＥＹ１時ＬＥＤ１亮，如圖 3-7，
放開時暗，如圖 3-8。

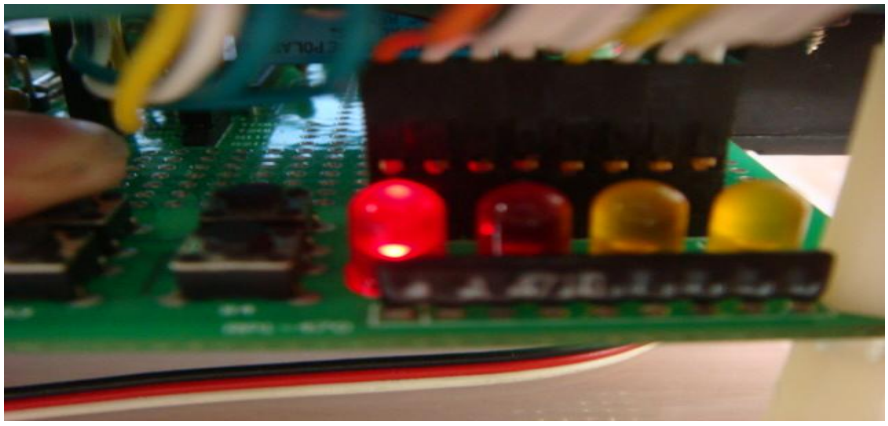


圖 3-7 按下ＫＥＹ１時ＬＥＤ１亮

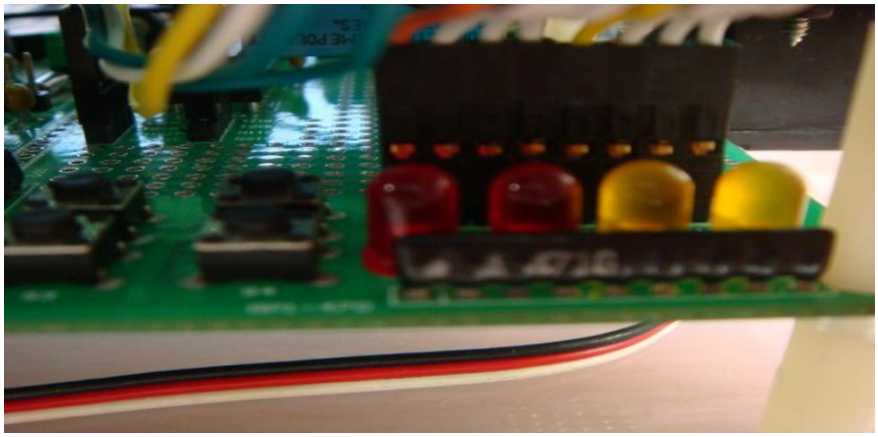


圖 3-8 放開ＫＥＹ１時ＬＥＤ１暗

3.4. 副程式

3.4.1. FLAYDATA

3.4.1.1 FLYDATA 版本演進

由於本專題目的為自動飛行，故需結合各週邊設備，得到所需之數據，並利用這些數據進行自動飛行之演算法，並對馬達進行控制，以達到自動飛行之目的。

FLYDATA 副程式所具備之功能為下列幾點。

1. 載入週邊模組提供之數據資料，如 GPS 所提供之經緯度、高度、行進速度、方向和 Compass 提供之 heading、pitching、rolling 與溫資資料。
2. 利用週邊所提供之資料，演算得知目標所在的方位，並判斷飛行船應做的補償動作。
3. 依 Compass 提供之溫度，斷判系統是否過熱，以及利用一組按鍵進行手動或自動的切換，最後依補償動作與所規定之轉向、動力輸出，表 3-2 **錯誤! 找不到參照來源。**為 FLYDATA 副程式的版本演進表。

表 3-2FLYDATA 副程式版本演進表

版本	功能	新增需求
MainPT _Ver0720	GetVD-算目前所在緯度之一(度分秒)的長度 GetXD-算目標與飛船間的距離(經緯) GetDegree-算目標與飛船間的角度 GetHD-算目標與飛船間的距離(高度差) GetRD-算目標與飛船間的距離(實際距離)	算出距離後，尚不知道要做何動作，不知道要輸出何種訊號控制馬達。其他副程式也無法得知此計算結果
MainPT _Ver0731	新增飛行資料陣列諸存飛行中計算、接受、判斷的資料。 函式： DataRec-更新資料-接收 DataMath-資料計算 FlyDataLoad-提供其他程式載入本程式之資料 FlyDataSave-提供其他程式輸入更新資料 GetNDegree 算目標與飛船間的角度 GotoLorRD-飛船要左或右多少度 GotoUorDD-飛船要升或降多少度 GetPitchDegree-目標與船的仰角 原函式 GetVD 更名：	將 LOAD 與 SAVE 函式完成後，需要讓其他程式試著存取，以確定函式正確運作。由於此時也完成了ＣＭＰＳ的模組程式，故可以進行目標判斷程式的測試，判斷向左或向右的動作，但

	GetLonUnit 算目前所在緯度之一(度分秒)的長度 原函式 GetXD 更名： Get2DM 算目標與飛船間的距離(經緯) 原函式 GetHD 更名： GetHeightDoX 算目標與飛船間的距離(高度差) 原函式 GetRD 更名： 新：Get3DM-資料計算 算目標與飛船間的距離(實際距離)	仍未連接 PWM 飛船馬達，故尚未完成 PWM 數值指令表的設定。
MainPT_Ver0822	因測試整合其他程式，飛行資料陣列大小進行調整，由 10*10 更改為 16*16	整合中發現程式會相互干擾，commnucation 與 pwm 模組的 init 內有相衝突的設定。以致於無法同時初始化成功。
MainPT_Ver0921	加入 PWM 數值輸出參數設定，完全整合所有模組與副程式。	發生１４秒問題此問題記錄：於主要是ｇｐｒｓ的 Rx 沒反應所以 delay 到時間。

3.4.1.2 FLYDATA 功能結構

Flydata 程式所具備之功能與結構分工，圖 3-9 可知此程式之流程，於 Flydata 程式之主要功能為下列三大部分：

1. DataRec：

   將 GPS 與 Compass 模組程式要求所需資料,可得知目前所在經緯度、高度、heading、pitching、rolling 與溫度…等資料，並存入 FLYDATA 內的陣列等待後續處理。

2. DataMath：

   將利用 DataRec 中抓取的 GPS 資料獲得目標點所在方位並與 Compass 資料加以計算，得知飛船目前所該進行的動作為何。飛行船應該升降幾度，左右轉動多少度…等。(計算與方向判斷於 3.4.1.3 詳細說明)

3. FlyDataSysUpt：

此部分會判斷系統是否過熱、飛行船為自動還是手動模式，以及依上述 DataMath 之判斷，決定實際推力輸出與馬達動作角度需要多大。(詳細說明請參閱 3.4.1.4)

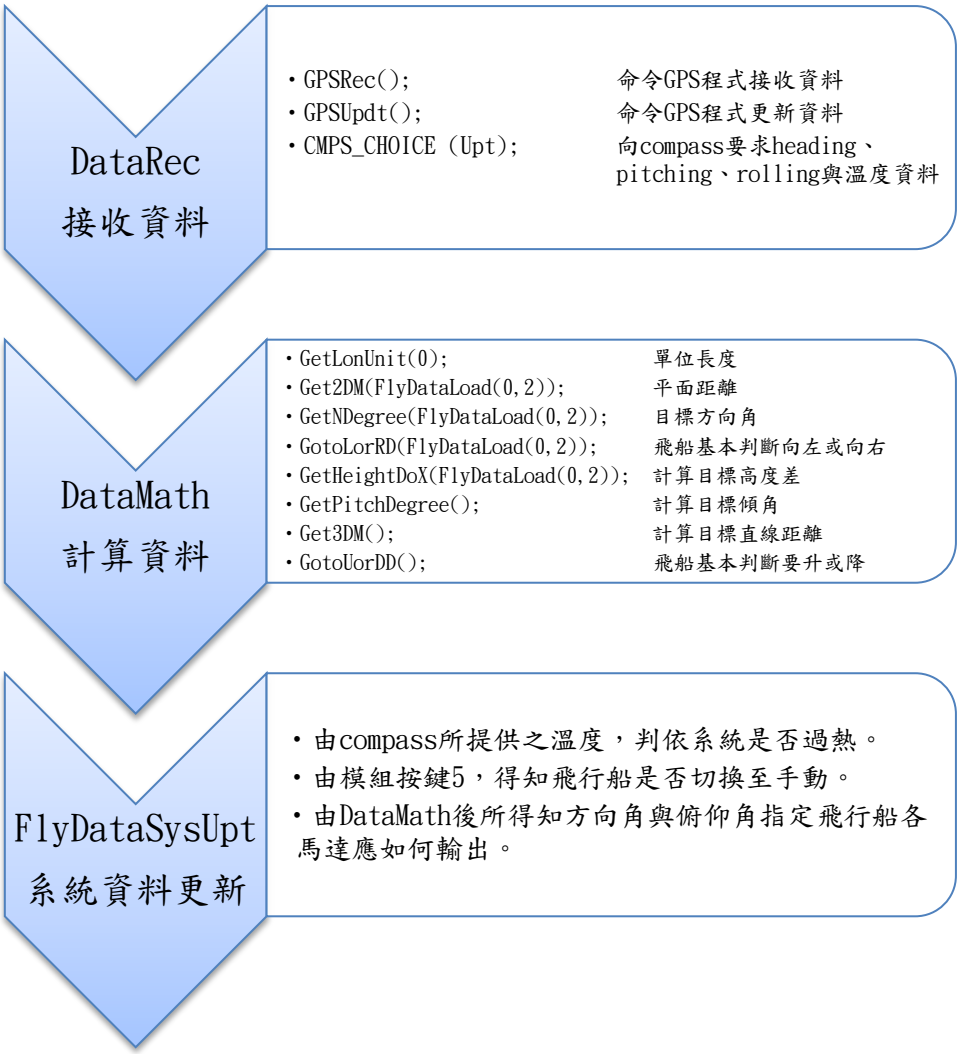


圖 3-9 Flydata 程式功能結構圖

3.4.1.3 DataMath

由於本研究為自主飛行，所以由程式來處理以下問題：目前在何處、航向，以及目標點在何方向、距離，最後判斷該往哪裡飛。目前在何處、航向，由 GPS 與 Compass 提供，而於目標點之方位、距離將於下列說明。

1. 目標點之距離

目標點之經緯度與高度於程式燒錄時會先行指定，升空飛行後仍可由地面站透過 GPRS 修改飛行目標點。我們將以 GPS 得知之目前經緯度與目標點之經緯度來計算距離與方向，但本研究所用之 PIC 板為 16bitMCU，故距離計算不能單純以地球上兩點距離公式

直接計算（因以度為單位，計算時兩點之距離太小，結果為 0），而需修改計算方式，經緯度以較小單位（分）為一基本單位。因飛行距離不長，故可將地球經緯線視為平面，以商高定理求得距離。

兩點距離公式如下：
點 1 之經度：Long1r
點 1 之緯度：Lat1r
點 2 之經度：Long2r
點 2 之緯度：Lat2r
地球半徑 r = 6371000 （單位M）
則地球上任兩點之距離

$$d = cos^{-1}\{[sin\,Lat1r \times sin\,Lat2r] + [cos\,Lat1r \times cos\,Lat2r \times cos(Long2r - Long1r)]\} \times r$$

又因在地球上之每條經線長度不同，但緯線長度相同（於程式可先行定義）。

故僅需計算經線長度即可利用商高定理得知目標點與目前飛行船所在位置之距離。

利用上述兩點公式可得知所在緯度之經線長，即可得知每 1 分為幾公尺。

並以商高定理得知平面距離。部分計算程式如下所示。

```
//GotoLonM          目標點之經度（以分為單位）
//NowLonM           飛行船之經度（以分為單位）
//GetLonUnit(1)   得知所在該緯度的 1 經線長（分）為幾公尺
//DLon              為目標與飛行船本身的經度差距（單位公尺）
DLon=(GotoLonM-NowLonM)*GetLonUnit(1)；

//GotoLatM          目標點之緯度（以分為單位）
//NowLatM           飛行船之緯度（以分為單位）
//VLatM             已定義之緯線（1 分）長度為幾公尺
//DLat              為目標與飛行船本身的緯度差距（單位公尺）
DLat=(GotoLatM-NowLatM)*VLatM；

//以商高定理算得平面距離（單位公尺）資料存於飛行資料陣列中
FlyDataSave(6，4，(int)sqrt(pow(DLon，2)+pow(DLat，2)));
```

2. 目標點在何方向

由上述計算方式即可知，兩點之經度差距與緯度差距，且單位同為公尺。故即可以簡單的歸納來得知目標的方向。

首先定義

經差＝目標經度-目前飛行船經度
緯差＝目標緯度-目前飛行船緯度
比值=經差/緯差

接下來依目標點所在位置在不同象限，tan 值會有所不同來得知其方位角度為何。

先讓比值都為正，然後以tan^{-1}反三角函式得知其在第一象限之角度，再依其所在象限另外計算。

若緯差大於 0 且經差大於 0，目標在目前的東北方

第 1 象限之方位角為： 方位角 $= \frac{tan^{-1}\text{比值}}{\pi} \times 180°$

若緯差大於 0 且經差小於 0，目標在目前的西北方

第 2 象限之方位角為：方位角 $= 360° - \left(\frac{\tan^{-1}(\text{-比值})}{\pi} \times 180°\right)$

若緯差小於 0 且經差大於 0，目標在目前的東南方

第 4 象限之方位角為：方位角 $= 180° - \left(\frac{\tan^{-1}(\text{-比值})}{\pi} \times 180°\right)$

若緯差小於 0 且經差小於 0，目標在目前的西南方

第 3 象限之方位角為：方位角 $= 180° + \left(\frac{\tan^{-1}(\text{比值})}{\pi} \times 180°\right)$

如上所述即可得知目標點所在之方位為何。

3. 決定航向

知道目標點的方位後，就要以目前的航向來做動作修正，而目前的航向就由 Compass 來提供。所以同樣參照以下圖 3-10 來歸納整理出下表。

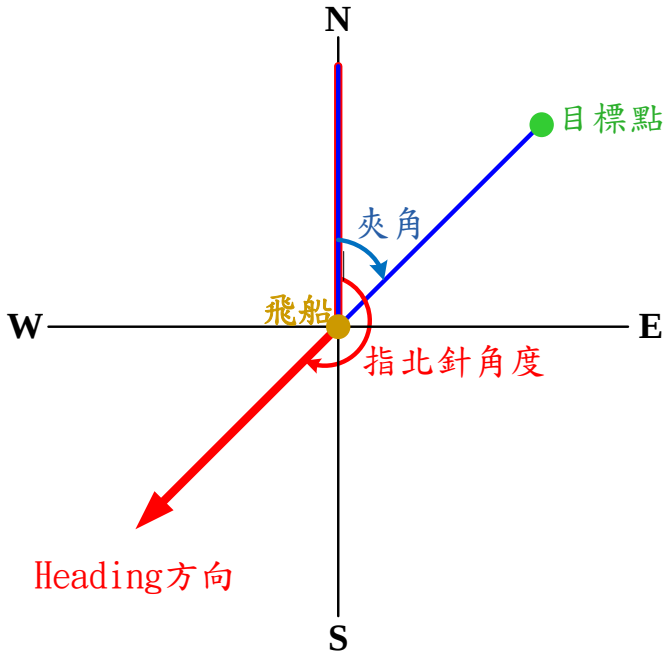


圖 3-10 決定航向示意圖

1、計算目標與飛船的相對方向是多少度，如圖 3-10 的夾角。

2、Compass 得到的 heading 數值為目前飛船指向的方位，如圖 3-10 的指北針角度。

3、以 Compass 減目標方位，得到應該補償的動作角度，補償動作歸納如下表 3-3。

表 3-3 補償動作

夾角（度）	0	90	180	270
1.指北針角度	0	0	0	0
補償動作角度	0	90	180	270
2.指北針角度	90	90	90	90
補償動作角度	－90	0	90	180
3.指北針角度	180	180	180	180
補償動作角度	－180	－90	0	90
4.指北針角度	270	270	270	270
補償動作角度	－270	－180	－90	0

動作角度 = (目標飛船的連線與北的夾角)－指北針角度

由上表 1.得知－180~0 左轉 0~180 右轉 <－180 右轉(比較近)

由上表 2.得知－180~0 左轉 0~180 右轉

由上表 3.得知 0~180 右轉－180~0 左轉
由上表 4.得知 0~180 右轉－180~0 左轉　>180 左轉(比較近)
故：　動作角度　0~180　右轉，180 以上左轉
動作角度　0~－180 左轉，－180 以下右轉
如此，heading 的航向修正算是解決了。

再來就是要上升或下降，上升或下降就以目標高度-目前高度即可知道，而升降的角度是由一伺服馬達所控制，且船身若不是水平，則升降轉動的伺服馬達應轉動不同的角度做為補償，故我們仍要算出 pitching 的角度。

計算的方式如圖 3-11 與下表 3-4 **錯誤! 找不到參照來源。**所列。

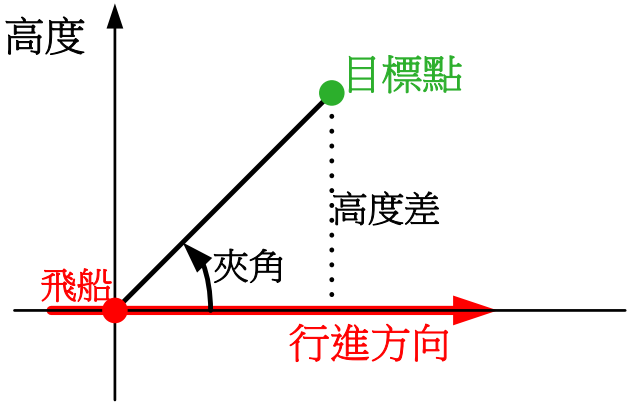


圖 3-11

藉由高度差與距離的比值可用反三角函數tan^{-1}來計算船身與目標的夾角，接著利用 Compass 得到的 pitching 為依據來判斷應該向上降或向下降，馬達應該轉動多少度，列表如下表 3-4。

表 3-4

	夾角（度）	90	0	－90
1.	指北針傾角	90	90	90
	補償動作角度	0	－90	－180
2.	指北針傾角	0	0	0
	補償動作角度	90	0	－90
3.	指北針傾角	－90	－90	－90
	補償動作角度	180	90	0

補償動作角度＝夾角－指北針傾角

補償動作角度的意思就是，要朝向目標的夾角角度轉的話，(指北針傾角)還差幾度。

由上面的兩個判斷可以得到飛行船目前應該做的動作，升或降幾度以及左轉或右轉幾度。

3.4.1.4 FlyDataSysUpt

在上一小節中我們已經得知了目標在何處以及該如何動作才能面向目標，但僅僅是知道，並未決定如何動作，而在 FlyDataSysUpt 將會以應該執行的動作以及馬達可做的動作列表，來決定要下達何種動作指令。此程式也會判斷系統是否過熱以及是否被搖控器改為手動飛行，並記錄於飛行資料陣列內，待 communication 副程式進行資料傳送。

1. 過熱判斷：由於 Compass 與主板在飛行中所放置的位置相差不遠，所以我們就直接使用 Compass 所提供的溫度來做為系統是否過熱的依據。若是系統發生過熱，則會將系統過熱旗標變為 true，並記錄於飛行資料陣列之指定位置。
2. 手/自動切換：我們所開發的系統後除自動飛行之外，仍可以搖控器控制，而於此程式中所說的手自動切換，實則手/自動模式記錄。程式依照目前是何種模式，如果切換成手動則記錄 true 否則記錄 false 於飛行資料陣列中之指定行列。而程式是內定開關，當切換手動時便會按下這組開關，程式進而得知目前是搖控器操縱的手動模式，若沒按下此開關，程式則設定為自動飛行模式（見 2.4 節）。程式內以公用程式中規劃之按鍵 5 來對應此組開關。若按鍵 5 被按下則記錄為手動，否則為自動。
3. 動作指令：在本自動飛行系統中，控制馬達的動作是以一動作指定參數來設定，其中分為兩大類，伺服馬達轉動角度控制類與無刷馬達轉速推力類。

伺服馬達中，前方升降的控制可以達到正負 90 度，而尾端的伺服馬達僅能到達正負 45 度。皆對照下表 3-5 **錯誤! 找不到參照來源。**做設定，並依表填參數值於飛行資料陣列中指定之行列。

表 3-5 馬達控制參數設定表

判斷動作	<-75	<-60	<-45	<-30	<-15	<0	0	>0	>15	>30	>45	>60	>75
指令參數	1	2	3	4	5	6	7	8	9	10	11	12	13
升降馬達	-90	-75	-60	-45	-30	-15	0	15	30	45	60	75	90
轉向馬達	-45	-45	-45	-45	-30	-15	0	15	30	45	45	45	45

而馬達轉速推力的決定則視當日的風力再自行做設定，設定以 0～100％來做設定，依表填參數值於飛行資料陣列中指定之行列。如表 3-6 所示。

表 3-6 馬達推力輸出百分比

需要推力	0%	10%	25%	40%	50%	75%	100%

指令參數	7	8	9	10	11	12	13

FlyDataSysUpt 所下達的指令會存於飛行資料陣列內，待 Fybtctrlor 副程式依指令參考我們量測後設定的 PWM 數值來做輸出，詳細的設定與輸出 PWM 訊號於各馬達之方式在 3.4.4FYBTCTRLOR 節與 4.1 節 PWM 中詳細說明。

3.4.1.5 飛行資料陣列內容表規畫 FyBtData

於本系統之週邊設備所抓取之資料，以及程式計算之結果皆會存於飛行資料陣列中，如表 3-7 所示，以提供各程式間溝通使用。

表 3-7 旗標表列

index	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	A	A	E	A	A	A	A	A	A	A	E					
1	B	B	B	B	B	B	C	C	C	A						
2	D	D	D	D	D	D	D	D	D							
3	E	E	E	E	E	E	E	E	E	E						
4	F	F	F	G	G	G	G			J						
5	I	I														
6	K	K			J	J	M	M								
7			H	H	M	M	M	M								
8	N	N	N	N		N	N	N	N							
9	O	O	O	O		O	O	O	O							
10	P	P	P	P	M	P	P	P	P	M						
11	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
12	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
13	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S
14	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T
15	U														U	U

依 row index 可畫分為以下幾部分加以說明。

A. 狀態旗標：

此為自機飛行系統中的狀態旗標所存放處。

FyBtData[0][0]：手/自動狀態旗標，1 為手動 0 為自動。
FyBtData[0][1]：地控/自動狀態旗標，1 為地控 0 為自動。
FyBtData[0][3] ：終點旗標，1 為抵達終點 0 為尚未抵達終點，即到達最後目標點則記錄到達終點。

FyBtData[0][4] ：過熱旗標，1 為過熱 0 為未過熱。
FyBtData[0][5] ：升降轉向不足旗標，1 為 true0 為 false。
FyBtData[0][6] ：左右轉向不足旗標，1 為 true0 為 false。
FyBtData[0][7] ：高度超出旗標，1 為 true0 為 false。
FyBtData[0][8] ：升降轉向過多旗標，1 為 true0 為 false。
FyBtData[0][9] ：左右轉向過多旗標，1 為 true0 為 false。

FyBtData[1][9] ：msg，此為 ROWINDEX 0 之部分旗標組合以 8bit 表示。如下表 3-8 所示。

表 3-8

FyBtData [0][1]	[0][3]	[0][4]	[0][5]	[0][6]	[0][7]	[0][8]	[0][9]
0	0	0	0	0	0	0	0

B. 日期與時間：

此行記錄 GPS 所提供之現在時間，即飛行主程式之系統時間。
FyBtData[1][0] ：日。
FyBtData[1][1] ：月。
FyBtData[1][2] ：年。
FyBtData[1][3] ：時。
FyBtData[1][4] ：分。
FyBtData[1][5] ：秒。

C. GPS 定位狀態：

FyBtData[1][6] ：GPRMC 格式定位資料是否可用，1 可用，0 不可用。
FyBtData[1][7] ：GPGGA 格式定位資料是否可用， 0 未定位，1 已定位，2 DGPS 精準定位。
FyBtData[1][8] ：GPGSA 格式之定位狀態，1 未定位，2 二維定位，3 三維定位。

D. 現在位置相關資料：

FyBtData[2][0] ：緯度資料，度，現在緯度幾度。
FyBtData[2][1] ：緯度資料，分，現在緯度為幾分。
FyBtData[2][2] ：緯度資料，目前緯度 0.0001 分。
FyBtData[2][3] ：緯度資料，南緯或北緯(0：南緯， 1：北緯)。
FyBtData[2][4] ：經度資料，度，現在經度幾度。
FyBtData[2][5] ：經度資料，分，現在經度為幾分。
FyBtData[2][6] ：經度資料，目前緯度 0.0001 分。

FyBtData[2][7] ：經度資料，西經或東經(0：西經， 1：東經)。

FyBtData[2][8] ：高度，單位 0.1 公尺。

E. 目標點相關資料：

FyBtData[0][2] ：目前目標點編號，預設 0 即第 1 目標點，抵達後則加 1，即目標下一點。

FyBtData[0][10] ：總目標點數，記錄總有多少目標點，即到達第幾目標點後算抵達終點。

FyBtData[3][0] ：緯度資料，(度)，現在目標緯度幾度。

FyBtData[3][1] ：緯度資料，(分)，現在目標緯度為幾分。

FyBtData[3][2] ：緯度資料，目前目標緯度 0.0001 分。

FyBtData[3][3] ：緯度資料，南緯或北緯 (0：南緯， 1：北緯)。

FyBtData[3][4] ：經度資料，(度)，現在目標經度幾度。

FyBtData[3][5] ：經度資料，(分)，現在目標經度為幾分。

FyBtData[3][6] ：經度資料，目前目標緯度 0.0001 分。

FyBtData[3][7] ：經度資料，西經或東經(0：西經， 1：東經)

FyBtData[3][8] ：高度，單位 0.1 公尺。

FyBtData[3][9] ：飛行高度範圍，在此範圍內飛行才安全，否則會記錄並警告超出飛行範圍，過低、過高。

F. 電子指北針 Compass 資料：

此為向 Compass 所要求的三維資料。

FyBtData[3][0]：heading，角度(度)。

FyBtData[3][1]：pitching，角度(度)。。

FyBtData[3][2]：rolling，角度(度)。。

G. 飛行船所判斷該如何動作：

FyBtData[3][3]：左右轉動作角度，0~180 度。

FyBtData[3][4]：升降轉動作角度，0~90 度。

FyBtData[3][5]：左右轉動作方向，1 右，－1 左。

FyBtData[3][6]：升降轉動作方向，1 升，－1 降。

H. 角度規範

FyBtData[3][7]：左右轉過頭角度定義。

FyBtData[3][8]：升降轉過頭角度定義。

FyBtData[7][2]：行進角度範圍〈左右〉。

FyBtData[7][3]：行進角度範圍〈升降〉。

I. 溫度：

FyBtData[5][0]：Compass 溫度。

FyBtData[5][1]：過熱溫度。

J. 距離：

FyBtData[4][9]：目標高度差，單位 M。
FyBtData[6][4]：目標平面距離，單位 M。
FyBtData[6][5]：目標直線距離，單位 M。

K. 移動速度方向：

FyBtData[6][0]：相對位移速度，節。
FyBtData[6][1]：相對位移方向，度。

L. 計算夾角：

FyBtData[7][0]：平面 heading 夾角。
FyBtData[7][1]：垂直 pitching 夾角。

M. 馬達輸出設定：

FyBtData[6][6]：前升降馬達推力倍率，(%)。
FyBtData[6][7]：後轉向馬達推力倍率， (%)。
FyBtData[7][4]：前升降推力最大角度。若大於此角度，將推力設至最大值。
FyBtData[7][5]：後轉向推力最大角度，若大於此角度，將推力設至最大值。
FyBtData[7][6]：前升降推力最小角度，若小於此角度，將推力設至最大值。
FyBtData[7][7]：後轉向推力最小角度，若小於此角度，將推力設至最大值。
FyBtData[10][4]：自動時輸出馬達 PWM 訊號時的變化量，數值越大反應越快
FyBtData[10][9]：地控時輸出馬達 PWM 訊號時的變化量，數值越大反應越快。

N. 馬達輸出設定：

自動飛行計算時存入：
FyBtData[8][0]：升降轉向馬達指令。
FyBtData[8][1]：左右轉向馬達指令。
FyBtData[8][2]：升降推力馬達指令。
FyBtData[8][3]：左右轉推力馬達指令。

自動飛行-地面站控制時存入：
FyBtData[8][5]：升降轉向馬達指令 。
FyBtData[8][6]：左右轉向馬達指令。
FyBtData[8][7]：升降推力馬達指令。
FyBtData[8][8]：左右轉推力馬達指令。

O. 馬達輸出補償結果：

自動飛行計算後，依當日風力做額外補償時存入：
FyBtData[9][0]：升降轉向馬達指令。
FyBtData[9][1]：左右轉向馬達指令。
FyBtData[9][2]：升降推力馬達指令。

FyBtData[9][3]：左右轉推力馬達指令。

自動飛行-地面站控制，依當日風力做額外補償時存入：

FyBtData[9][5]：升降轉向馬達指令。

FyBtData[9][6]：左右轉向馬達指令。

FyBtData[9][7]：升降推力馬達指令。

FyBtData[9][8]：左右轉推力馬達指令。

P. 馬達輸出實際 PWM 數值：

自動飛行時各馬達實際輸出之 PWM 數值：

FyBtData[10][0]：升降轉向馬達實際輸出之 PWM 數值。

FyBtData[10][1]：左右轉向馬達實際輸出之 PWM 數值。

FyBtData[10][2]：升降推力馬達實際輸出之 PWM 數值。

FyBtData[10][3]：左右轉推力馬達實際輸出之 PWM 數值。

自動飛行-地面站控制時各馬達實際輸出之 PWM 數值：

FyBtData[10][5]：升降轉向馬達實際輸出之 PWM 數值。

FyBtData[10][6]：左右轉向馬達實際輸出之 PWM 數值。

FyBtData[10][7]：升降推力馬達實際輸出之 PWM 數值。

FyBtData[10][8]：左右轉推力馬達實際輸出之 PWM 數值。

Q. 前升降轉向馬達指令對應 PWM 值設定：

FyBtData[11][0]：馬達轉動 0 度時 PWM 數值。

FyBtData[11][1]：0 度與－90 度時 PWM 差值。

FyBtData[11][2]：0 度與－75 度時 PWM 差值。

FyBtData[11][3]：0 度與－60 度時 PWM 差值。

FyBtData[11][4]：0 度與－45 度時 PWM 差值。

FyBtData[11][5]：0 度與－30 度時 PWM 差值。

FyBtData[11][6]：0 度與－15 度時 PWM 差值。

FyBtData[11][7]：0。

FyBtData[11][8]：0 度與 15 度時 PWM 差值。

FyBtData[11][9]：0 度與 30 度時 PWM 差值。

FyBtData[11][10]：0 度與 45 度時 PWM 差值。

FyBtData[11][11]：0 度與 60 度時 PWM 差值。

FyBtData[11][12]：0 度與 75 度時 PWM 差值。

FyBtData[11][13]：0 度與 90 度時 PWM 差值。

FyBtData[11][14]：限制最小角度時 PWM 數值。

FyBtData[11][15]：限制最大角度時 PWM 數值。

R. 後轉向轉向馬達指令對應 PWM 值設定：

FyBtData[12][0]：馬達轉動 0 度時 PWM 數值。

FyBtData[12][1]：0 度與－90 度時 PWM 差值。

FyBtData[12][2]：0 度與－75 度時 PWM 差值。
FyBtData[12][3]：0 度與－60 度時 PWM 差值。
FyBtData[12][4]：0 度與－45 度時 PWM 差值。
FyBtData[12][5]：0 度與－30 度時 PWM 差值。
FyBtData[12][6]：0 度與－15 度時 PWM 差值。
FyBtData[12][7]：0。
FyBtData[12][8]：0 度與 15 度時 PWM 差值。
FyBtData[12][9]：0 度與 30 度時 PWM 差值。
FyBtData[12][10]：0 度與 45 度時 PWM 差值。
FyBtData[12][11]：0 度與 60 度時 PWM 差值。
FyBtData[12][12]：0 度與 75 度時 PWM 差值。
FyBtData[12][13]：0 度與 90 度時 PWM 差值。
FyBtData[12][14]：限制最小角度時 PWM 數值。
FyBtData[12][15]：限制最大角度時 PWM 數值。

S. 前升降推力馬達指令對應 PWM 值設定：

FyBtData[13][0]：馬達怠速時 PWM 數值。
FyBtData[13][7]：0，怠速。
FyBtData[13][8]：怠速與 10%推力時 PWM 差值。
FyBtData[13][9]：怠速與 25%推力時 PWM 差值。
FyBtData[13][10]：怠速與 40%推力時 PWM 差值。
FyBtData[13][11]：怠速與 50%推力時 PWM 差值。
FyBtData[13][12]：怠速與 75%推力時 PWM 差值。
FyBtData[13][13]：怠速與 100%推力時 PWM 差值。
FyBtData[13][14]：限制最小角度時 PWM 數值。
FyBtData[13][15]：限制最大角度時 PWM 數值。

T. 後轉向推力馬達指令對應 PWM 值設定：

FyBtData[14][0]：馬達怠速時 PWM 數值。
FyBtData[14][7]：0，怠速。
FyBtData[14][8]：怠速與 10%推力時 PWM 差值。
FyBtData[14][9]：怠速與 25%推力時 PWM 差值。
FyBtData[14][10]：怠速與 40%推力時 PWM 差值。
FyBtData[14][11]：怠速與 50%推力時 PWM 差值。
FyBtData[14][12]：怠速與 75%推力時 PWM 差值。
FyBtData[14][13]：怠速與 100%推力時 PWM 差值。
FyBtData[14][14]：限制最小角度時 PWM 數值。
FyBtData[14][15]：限制最大角度時 PWM 數值。

U. 升降舵翼 PWM 控制參數：

C型飛行船尾翼升降舵轉動角度變化較小,故只使用置中、最大角度與最小角度，三個動作。

FyBtData[15][0]：0 度，水平。

FyBtData[15][14]：下轉極限角度，飛行船需要下降時輸出。

FyBtData[15][15]：上轉極限角度，飛行船需要上升時輸出。

3.4.2. TUNER

這部分程式是要依當日天氣狀況來做調整。由於飛行時天氣因素的不同，可在此設定不同的微調參考,增加或減少動力,讓自動飛行系統更靈活。以下為 TUNER 程式片段,程式如附錄。

目前版本尚未進行微調。直接依原 flydata 計算之動作下達指令

```
FlyDataSave(9，0，FlyDataLoad(8，0));
FlyDataSave(9，1，FlyDataLoad(8，1));
FlyDataSave(9，2，FlyDataLoad(8，2));
FlyDataSave(9，3，FlyDataLoad(8，3));
```

3.4.3. WHEREIS

用來判斷是否到達目的地。先判斷是否到達目標點。若抵達目標點則判斷是否還有下一點，若還有下一點則載入下一點資料，若沒有下一點則代表到達終點，如圖 3-12。

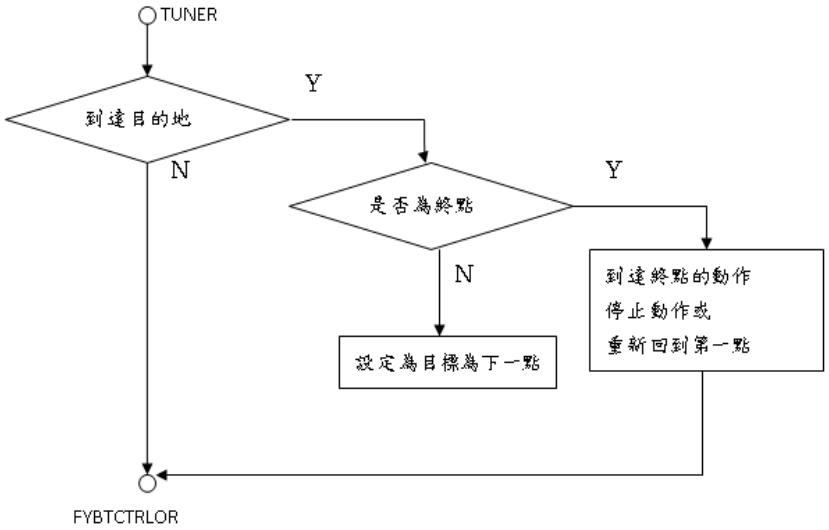


圖 3-12WHEREIS 程式流程

3.4.4. 馬達輸出　FYBTCTRLOR

3.4.4.1　FYBTCTRLOR 概述

此程式目的在將陣列裡的動作依據設定值轉換為各馬達所對應的 PWM 數值，並輸出至各馬達，讓運算出來的動作能夠正確的輸出做動。

另程式內附加有一漸變值列入計算，此數值之參與主要影響動作間的改變速度，若漸變值愈大，則動作切換愈快，反之則愈慢，目的在不同的環境內，能有不同的反應速度可作為調。

3.4.4.2　FYBTCTRLOR 摘要

在本程式內，主要分為兩組不同輸出計算對應不同的馬達，其中一、二組為前後轉向馬達輸出，因為只有轉向角度的計算，因此在 flaydata 陣列內導入需求動作代碼後，直接依照漸變值加減對應之 pwm 數值後即可完成做動，其程式碼如下：

Req2Pwm[i] = Pwm_Cur [i] + FlyDataLoad(Cur_S，PIC_Ofset);

Req2Pwm[i] = Pwm_Cur [i] - FlyDataLoad(Cur_S，PIC_Ofset);

另三、四組輸出為馬達推力輸出，此種輸出方式為百分比之計算，依照不同需求會得到不一樣的輸出百分比，會依照相對應的 pwm 數值計算出對應的百分比輸出，詳細程式碼如下：

```
Req2Pwm[i] = Pwm_Cur [i] * FlyDataLoad( MPow_S，MPow_F ) / 100 + FlyDataLoad( Cur_S，PIC_Ofset );
Req2Pwm[i] = Pwm_Cur [i] * FlyDataLoad( MPow_S，MPow_F ) / 100 - FlyDataLoad( Cur_S，PIC_Ofset );
```

由於此為百分比計算結果，因此計算出來數值可能會超過或小於馬達輸出所能接受之數值，故另外加入判斷是來判斷是否漁範圍之外，若超出，則等於所設定之最大/最小臨界值，詳細程式碼如下：

```
超過最大臨界值
 if( Req2Pwm[i] > FlyDataLoad( Req_S，  15 ) {
        Req2Pwm[i] = FlyDataLoad( Req_S，  15 );
}
小於最小臨界值
 if( Req2Pwm[i] < FlyDataLoad( Req_S，14 ) ){
        Req2Pwm[i] = FlyDataLoad( Req_S，  14 );
        }
```

3.4.4.3FYBTCTRLOR 流程圖

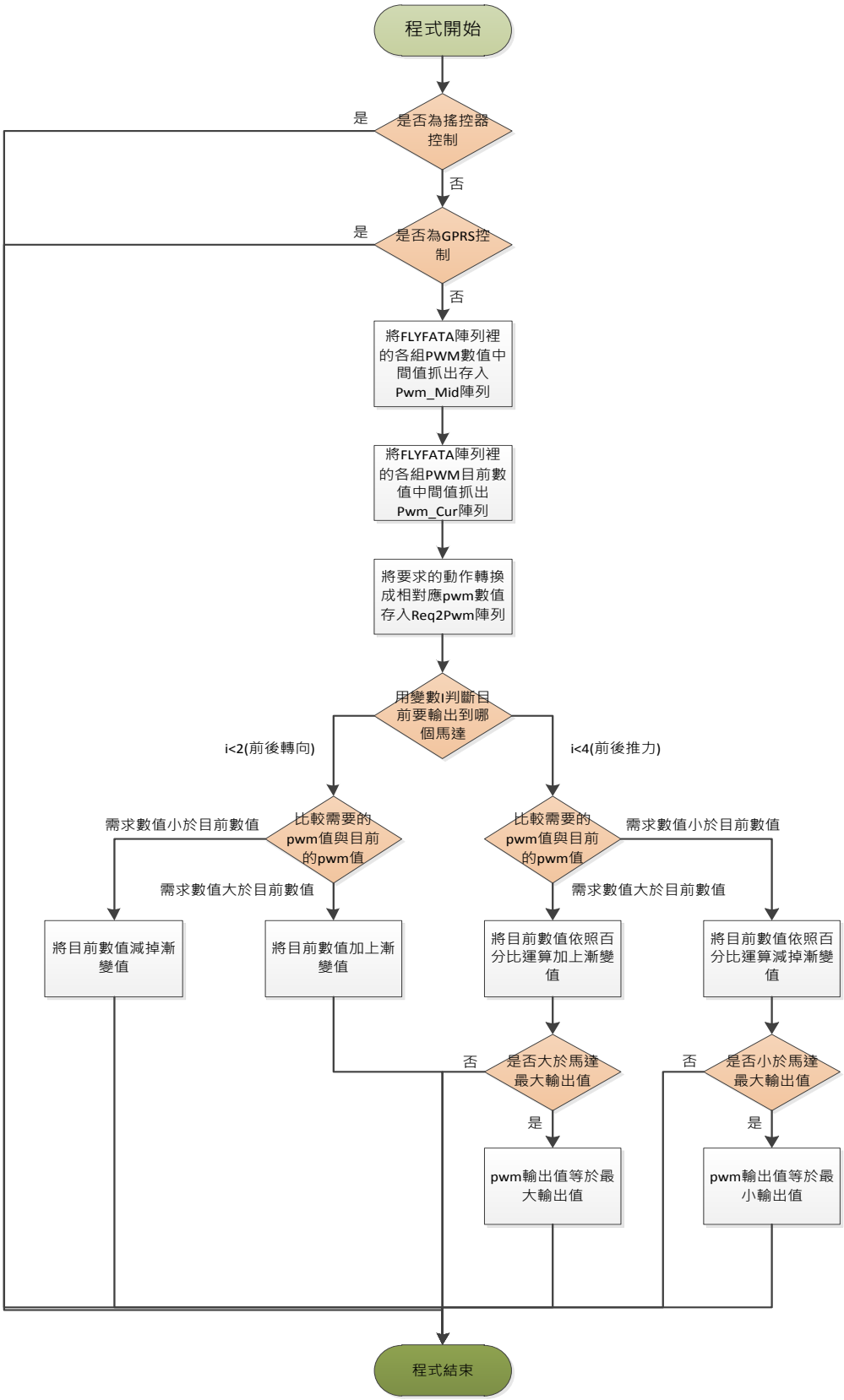


圖 3-13FYBTCTRLOR 流程圖

3.4.5. 地面通訊　COMMUNICATION

此程式主要功能為將飛行中所接收到 GPS 資料與計算動作資料傳至地面 Server，並接收地面 Server 回傳指令，將指令加以分析，存回 FlyData 陣列裡，如圖 3-14。

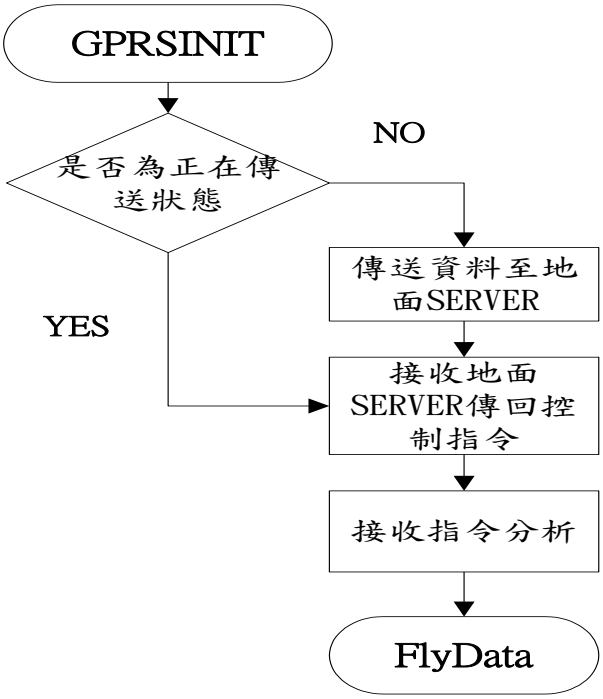


圖 3-14COMMUNICATION 流程

第四章 週邊模組控制設計

在完成主核心模組設計後，本章要針對周邊模組設計進行討論，主要的模組有：PWM、GPS、GPRS、CMPS。

4.1. 模組程式－PWM

待補：馬達與伺服器介紹、變電器。

4.1.1. 控制原理：脈波寬度調變 (Pulse Width Modulation ，PWM)

常用於直流馬達的控制、電源變換器之穩壓控制、甚至是直流轉換交流弦波的控制等，是控制直流馬達轉速最常見的方法。原理如圖 4-1 與圖 4-2 所示，圖中高電位的部份是馬達「動作(active)」時間，或叫做「責任週期(Duty Cycle)」，低電位的部份是馬達「停止(stop)」時間，兩者時間和即為一週期。當使用者想降低馬達轉速時，只要減少動作的時間、增加停止的時間，並保持週期不變即可，如圖 4-1 所示。反之，如果想加快馬達的轉速，則需要加長動作的時間、縮短停止的時間，並且保持週期不變，如圖 4-2 所示。由於改變轉速

是透過改變動作的時間比例，也就是圖 4-1 與圖 4-2，因此這樣的控制方式稱作脈波寬調變。

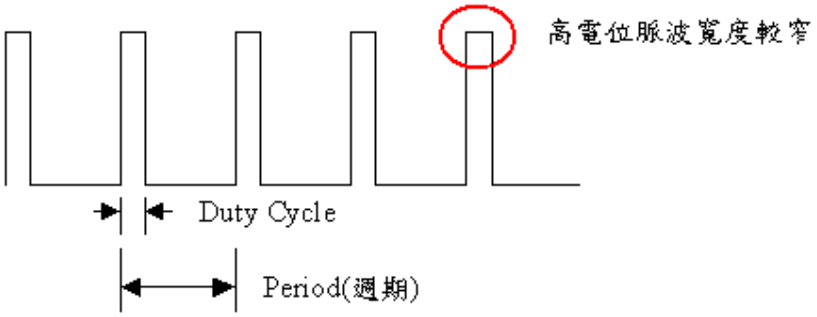


圖 4-1PWM 動作原理於低轉速時

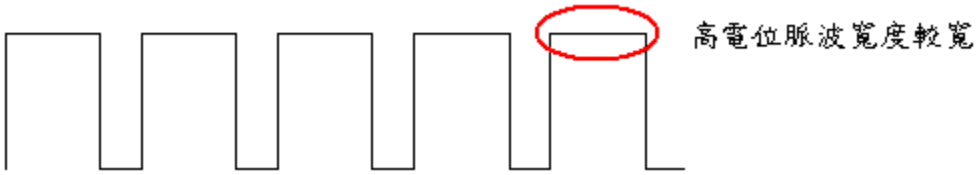


圖 4-2 動作原理於高轉速時

本專題載具所使用之轉向伺服馬達，升降伺服馬達，變電器，均使用 PWM 控制模式，來達成各馬達完成指定動作，由 PIC 產生相對應各馬達的 PWM 輸出數值，來操作目前飛船的動態，包含轉向控制，前進控制，升降控制等動作。

4.1.2. 模組所需提供功能

由於船身所有控制機構，都是由 PWM 訊號所達成，包括各式馬達，轉向伺服馬達，升降伺服馬達等，因此本專題之自動控制，是將資料運算過後之結果，利用模組所產生之 PWM 輸出，傳送至各個馬達辨識，以實現船體之控制。

4.1.3. PWM 程式開發研析

表 4-1 PWM 開發研析

項目	開發研析	說明
1	初版 PWM 輸出	已可利用 PIC 做出 PWM 控制訊號
2	可調整式 PWM 輸出	此版本可利用實體案件即時挑整 PWM 脈波寬，並將數值顯示至 LCD 螢幕上，主要功能在測試各個伺服馬達的接收範圍，以及對應之作動狀態，以便紀錄及操作
3	PWM 正式版模組化	將 PWM 程式正式模組化，包含初始化及模組化程式，利用程式呼叫及參數純入，即可操作 PWM 周邊
4	PWM 船身控制程式	將船身個馬達動作化成不同輸出範圍，並切個為 14 個等級，依照狀況經由陣列值計算出各組輸出數值，並輸出至各個馬達，且有漸變值設計，讓馬達作動能有不同的反應速度

4.2. 模組程式－GPS

4.2.1. 全球衛星定位系統（Global Positioning System）

全球衛星定位系統（Global Positioning System，簡稱 GPS），由 24 顆衛星組成，其以每天繞行地球兩週的速度，分別在六個軌道上運行；軌道面的傾角約 60 度，因此無論地球的任何一個角落、任何時間，只要接收器位於室外，均可以接收到數顆 GPS 衛星的訊號。GPS 定位的計算方式，採用三角測量的方法確定接收器的位置，利用三顆衛星所發出的球體訊號，三個球體訊號所相交的點即為接收器的位置，如圖 4-3，其能夠精確提供目標物的位置資料、速度、以及時間，完成導航之功用。

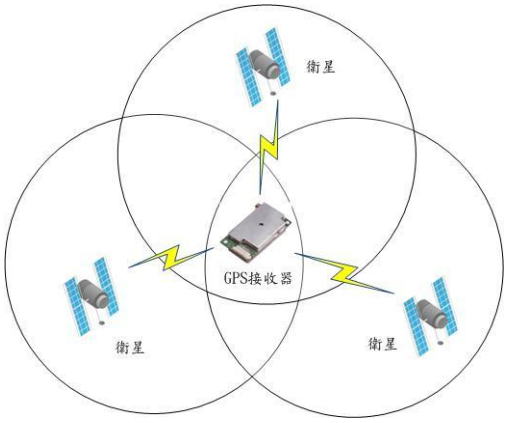


圖 4-3GPS 示意圖

4.2.2. GPS 訊號處理

GPS 接收器都具備有美國國家海洋電子學會 NMEA（National Marine ElectronicsAssociation）0183 ASCII 碼格式協議所制定的標準規格，是一種關於數位信號傳遞的標準。這些 GPS 提供串列通信介面，經信號處理後可以獲得用戶位置，速度等資訊，最終達到利用 GPS 進行導航和定位的目的。來自衛星的訊號有很多種，其發送的資料主要由封包頭，封包尾和封包內資料所組成，依據資料封包的不同，封包頭也不相同，主要有$GPGGA，$GPGSA，$GPRMC，以及$GPGSV 等，這些封包頭標示了後續封包資料的組成結構。在本研究中需要取得經緯度、日期、時間、高度、相對方向以及相對速度的資訊，所以擷取$GPGGA，$GPGSA 以及$GPRMC 訊號，而解析這些訊號會抓到像是這類字串$GPXXX，<1>，<2>，<3>，<4>，<5>，<6>，<7>，<8>，<9>，<10>，<11>，<12>*hh，每個逗號隔開視為一個欄位，而每個欄位值所代表的意義，用範例的方式來說明 GPS 接收器會接收到的 NMEA 數據資料格式(見附錄[1.])。

4.2.3. GPS 接收器

1. GPS15XL-W

   本研究採用 GARMIN 公司型號為 GPS15XL-W 的 GPS 接收器(圖 4-4)，用來進行飛行船定位及導航控制，其規格如(表 4-2)。

圖 4-4 GPS15XL 接收器

表 4-2 GPS15XL 規格

型號	GPS15XL-W
連接介面	RS-232
外觀大小(mm)	35.56 x 45.85 x 8.31
重量(g)	14.1
輸入電壓範圍(Volt)	3.3~ 5
資料更新頻率 (second)	1
輸出通訊協定	NMEA 0183
GPS 精確度(m)	< 15（95% typical）
DGPS 精確度(m)	< 3~5（95% typical）
Reacquisition	< 2 seconds
同時追蹤衛星數	12

2. LR9548S

本研究過程原先承接學長使用的 Leadtek LR9548S 模組(圖 4-5)，其規格如(表 4-3 **錯誤! 找不到參照來源。**)，後來因為腳位損壞，於是改採用前面提到的 GARMIN GPS15XL-W 模組。

圖 4-5 LR9548S

表 4-3 LR9548S 規格

型號	LR9548S
連接介面	RS-232
外觀大小(mm)	24 x 20 x 2.9
重量(g)	2.5
輸入電壓範圍(Volt)	3.2~5
資料更新頻率(second)	1
輸出通訊協定	NMEA 0183
GPS 精確度(m)	< 10
DGPS 精確度(m)	< 5（50% typical）
Reacquisition	0.1 seconds
同時追蹤衛星數	20

4.2.4. GPS 程式開發研析

表 4-4 GPS 開發研析

項目	開發研析	資料項目	備註
1	抓取 GPGGA 格式資料，分析陣列內容，依資料項目放入各別陣列。	時間 緯度(GGA 格式) 經度(GGA 格式)	原本是抓取 RMC 格式資料，但因定位時間過久，所以改成 GGA 格式嘗試。

		高度	
2	陣列資料轉為數字型態。 時間為 UTC 型式，於是把「時」的部分加 8，變成台灣時間。	同上。	1.資料轉為數字型態目的在便於計算。 2.發現定位時間過久是因為實驗室走廊位置遮蔽較多，且天候因素也有影響。
3	把各資料項目放入 GPSgetData 函數。	時間 緯度(GGA 格式) 經度(GGA 格式) 高度 北半球或南半球 東半球或西半球	資料統一放到陣列內，便於飛控程式抓取。
4	增加 GPRMC、GPGSA、GPGSV 格式。	時間 緯度(GGA 及 RMC 格式) 經度(GGA 及 RMC 格式) 高度 北半球或南半球 東半球或西半球 定位狀態 getStatus() 對地速度 Speed() 對地方向 Direction() 磁極變量 Mag()	比較 GGA 及 RMC 格式的經緯度資料，發現兩者數值相差約 2~3 度。
5	增加 GPVTG 格式。 GPRMV 格式的速度、方向、日期由 Array()函數擷取。 由 Resolve(char str[])函數分析。	時間 緯度(GGA 及 RMC 格式) 經度(GGA 及 RMC 格式) 高度 北半球或南半球 東半球或西半球 定位狀態 getStatus() 速度、方向、日期 Array()	除了 VTG 格式抓取的資料仍為字元外，其餘資料皆為數字型態。
6	刪掉 GPVTG 格式，全部資料轉為數字型態。	同上。	因為 Leadtek 的 LR9548S 模組損壞，所以改為 GARMIN 的 GPS15XL-W 。後者的出廠預設 NMEA 格式沒有 GPVTG，於是不擷取該格式資料。

4.2.5. GPS 程式流程

GPS 主程式先進行初始化動作，設定 UART(Universal Asynchronous Receiver/Transmitter)的 RX、TX 腳位及鮑率，初始化成功後開始接收資料，將 UART 讀取資料暫存器的內容放進陣列，分析 GPS 訊號格式並分別作整理，再來更新資料就是讀取出經緯度、高度、時間等的各個資料。檢查旗標的目的在於能夠清楚知道程式流程哪些部分出現問題，透過 LED 或 LCD 的顯示方便在研究過程容易觀看，若資料更新沒有問題，則繼續接收下一筆資料。GPS 主程式流程圖(如圖 4-6)。

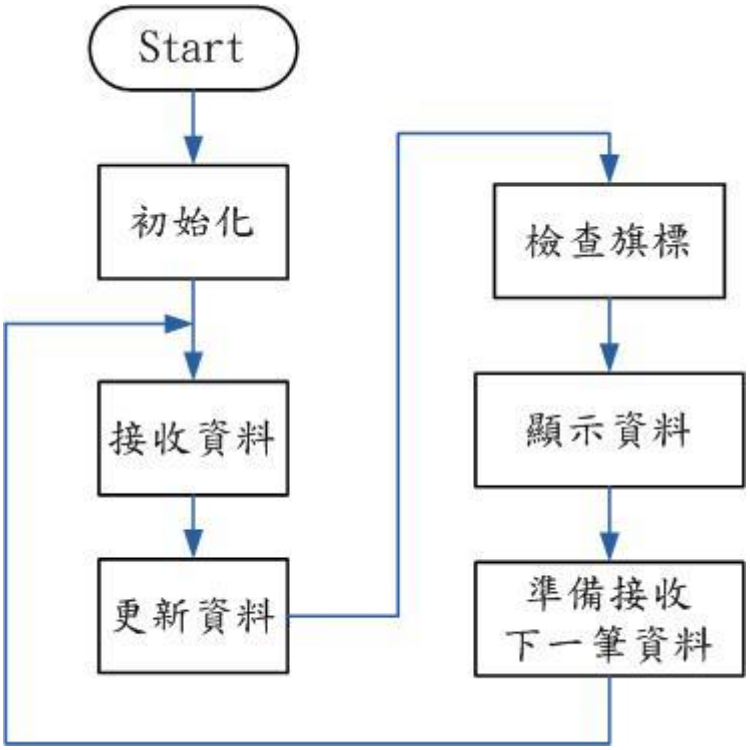


圖 4-6 GPS 流程圖

4.2.6. GPS 程式開發過程

GPS 接收器收到 NMEA 數據資料後，須經過分析再擷取所需要之資料，如 4.2.2 節所述。透過 LCD 顯示可便於開發過程查看資料數值，限制於 LCD 螢幕大小，無法一次呈現要看的數值，可設計按鍵切換，顯示各種資訊如表 4-5。為提升撰寫程式中除錯的效率，檢查旗標利用 LED 亮燈可知程式出錯於哪個流程，以便直接針對問題點來做解決。

表 4-5 GPS 資料顯示於 LCD 說明

按鍵	列	說明圖片資訊	圖片
1	第一列	1→GPGGA 北緯 2254.1723→GPGGA 緯度	圖 4-7

	第二列	1→GPRMC 北緯 2254.1721→GPRMC 緯度	
2	第一列	1→GPGGA 東經 12027.9572→GPGGA 經度	圖 4-8
	第二列	1→GPRMC 東經 12027.9726→GPRMC 經度	
3	第一列	1→GPGGA 定位型態 S： 1→GPRMC 定位型態 3→GPGSA 定位型態 H： 3202→高度(乘以 10 倍過後)	圖 4-9
	第二列	12 21 43：時間 190810：日期	
4	第一列	Speed 0：GPRMC 速度(乘以 10 倍過後)	圖 4-10
	第二列	Dire 1771：GPRMC 方向(乘以 10 倍過後)	

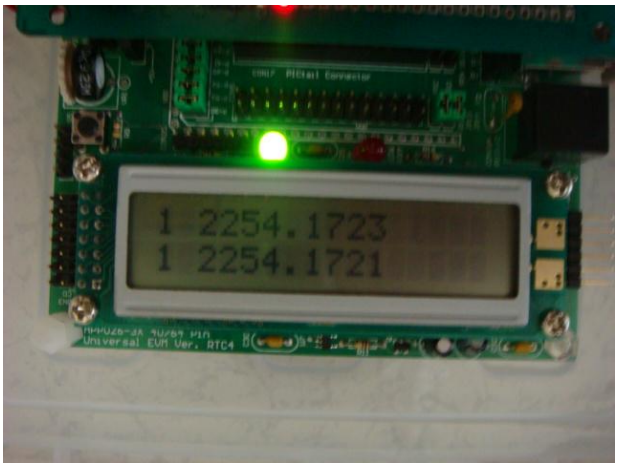


圖 4-7


1 12027.9572
1 12027.9726

圖 4-8


S:113 H:3202
12 21 43 190810

圖 4-9


Speed 0
Dire 1771

圖 4-10

4.3. 模組程式－GPRS

4.3.1. 通用封包無線服務通訊原理

通用封包無線服務(General Packet Radio Service， GPRS)

GPRS 是利用 GSM 網路中未使用的 TDMA 通道，提供中速的數據傳輸。GPRS 是採用封包交換（Packet Switch），即是多名用戶可以共享一個相同的傳輸通道，每位用戶只有在傳輸時才會佔用通道。如此一來。所有可用頻寬可以立刻分配給當前發送數據的用戶，這樣更多的間隙發送或接收數據的用戶可以共享頻寬。

本專題使用 Create GPRS4SIM100 為通訊模組(圖 4-11)，以下表 4-6 為規格表。

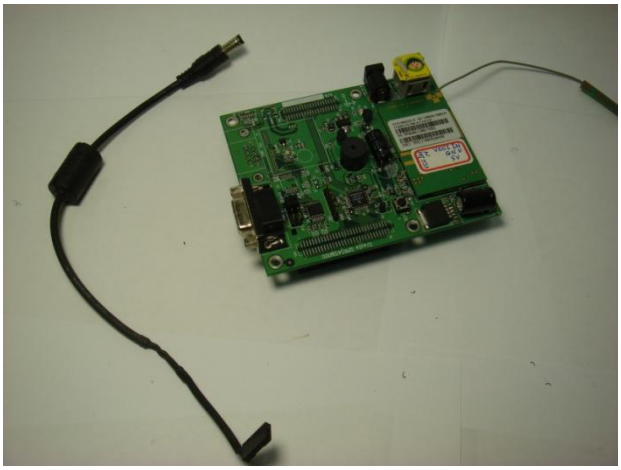


圖 4-11 GPRS 模組與電源線

表 4-6 GPRS4SIM100 規格

項目	規格
重量	11 公克
外觀大小	53 x 33x 3.0 mm
連接介面	RS-232
電力供應	3.4V-4.5V
頻寬	SIM100S Tri-band： EGSM 900， DCS 1800， PCS 1900 Compliant to GSM Phase 2/2+
GPRS 連接	GPRS multi-slot class 10 GPRS mobile station class B
SIM 介面	Supported SIM card： 1.8V ，3V
GPRS 傳輸	下載： 85.6kbps(max) 上傳：42.8 kbps(max)
外接天線	Connected via 50 Ohm antenna connector or antenna pad

4.3.2. GPRS 接收資料格式

1.手自動切換

手動

"@，"+"0"+"，"+"1"+"，"+"#"+'\n'如下表 4-7。

自動

"@，"+"0"+"，"+"0"+"，"+"#"+'\n'如下表 4-7。

表 4-7

名稱	實例
起始字元	@
手動或自動判斷	0
自動	0
手動	1
結束字元	#
Enter	\n

2. 馬達

"@，"+"1"+"，"+degreechange1+"，"+degreechange2+"，"+powermode1+"，"+powermode2+"，"+"#"+'\n 如下表 4-8。

表 4-8

名稱	實例
起始字元	@
馬達狀況	1
升降角度	degreechange1
左右轉角度	degreechange2
升降推力馬達%數	powermode1
左右轉向推力馬達%數	powermode2
結束字元	#
Enter	\n

3.改變行列

"@，"+"2"+"，"+columnchange+"，"+rowchange+"，"+data+"，"+"#"+'\n';如下表 4-9。

表 4-9

名稱	實例
起始字元	@
改變行列	2
列	columnchange
行	rowchange
資料	data
結束字元	#

Enter	\n

4.改變點

"@,"+"3"+","+changepoint+","+E0+","+E1+","+E2+","+E+","+N0+","+N1+","+N2+","+N+","　　+high+","+"#"+'\n'如下表 4-10。

表 4-10

名稱	實例
起始字元	@
地面改變點	3
改變點	changepoint
經度	E0
分	E1
秒	E2
東西經	E
緯度	N0
分	N1
秒	N2
南北緯	N
高度	HIGH
結束字元	#
Enter	\n

4.3.3. AT 指令與終端機測試

1. AT 指令使用

表 4-11

指令	作用	回傳字元	意義
AT\r\n	裝置 RS232 連接測試	OK	GPRS 模組已連接
AT+CGATT=1\r\n	GPRS 連接狀態	OK	連接 GPRS 模組
		ERROR	未連接
AT+CGDCONT=1，\"IP\"，\"INTERNET\"\r\n	定義 PDPcontext	OK	設定成功
AT+CSTT=\"INTERNET\"\r\n"	啟動任務並設置 APN、USER ID、PASSWORD	OK	啟動成功
AT+CIICR\r\n	啟用 GPRS 或 CSD 無線連線	OK	GPRS 已連上

		ERROR	GPRS 未連接
AT+CIFSR\r\n	取得 IP 位置	IP	已取得 IP 位置
AT+CIPSTART=\"TCP\"，\"IP\"，\"PORT\"\r\n	跟遠端 IP 做連線	OK	嘗試連接遠端 IP
		CONNECT OK	已連接成功
AT+CIPSEND\r\n	傳送訊息前的準備	>	準備完成
AT+CIPCLOSE\r\n	切斷與主機的聯繫	CLOSE OK	已切斷遠端網路連線
AT+CIPSHUT\r\n	切斷連線放棄取得 IP	SHUT OK	已切斷

2. 終端機操作 GPRS 連線

開啟終端機，如 圖 4-12。

圖 4-12

3. 設定通訊速率為 57600，如圖 4-13。

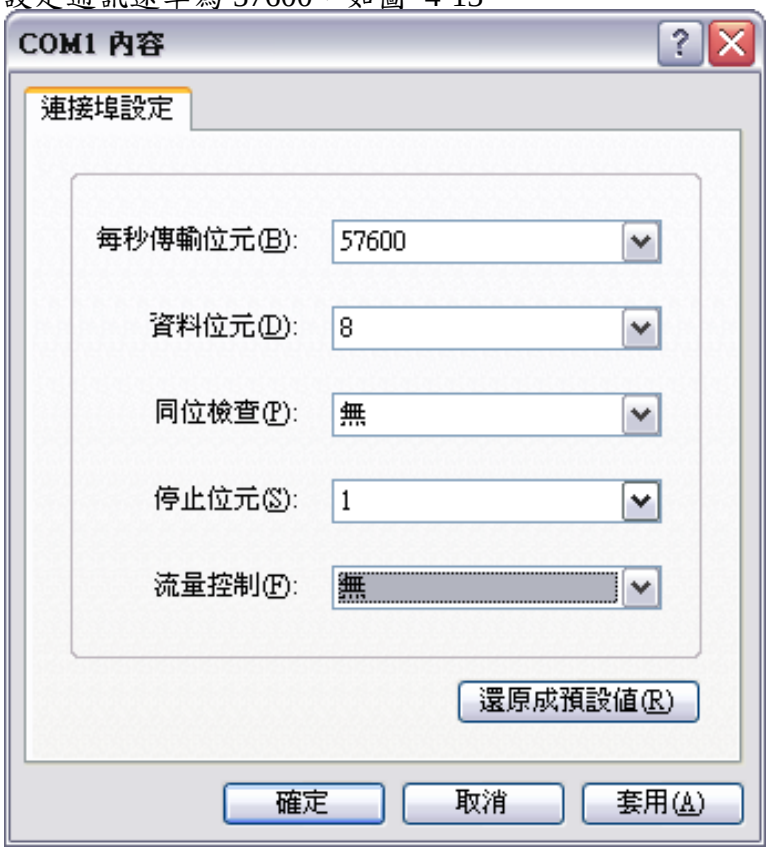


圖 4-13

4. 下達 AT 指令，連線成功，資料順利傳出去，圖 4-14。

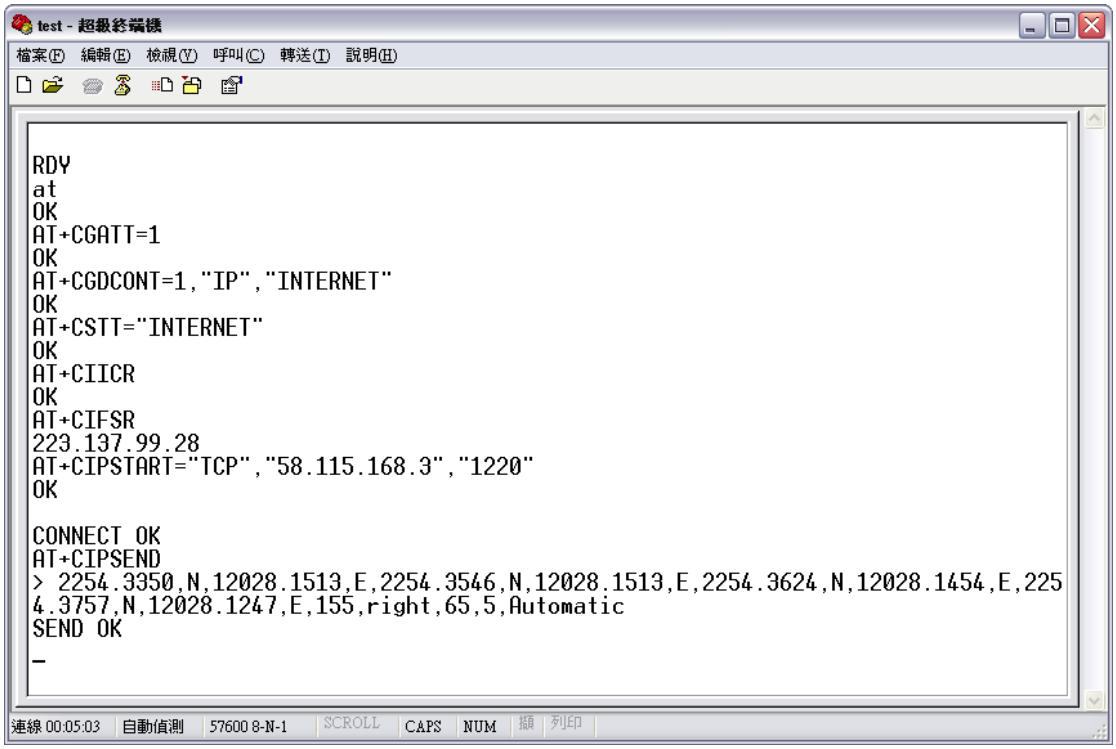


圖 4-14

5. 連接到關閉的過程，如圖 4-15。

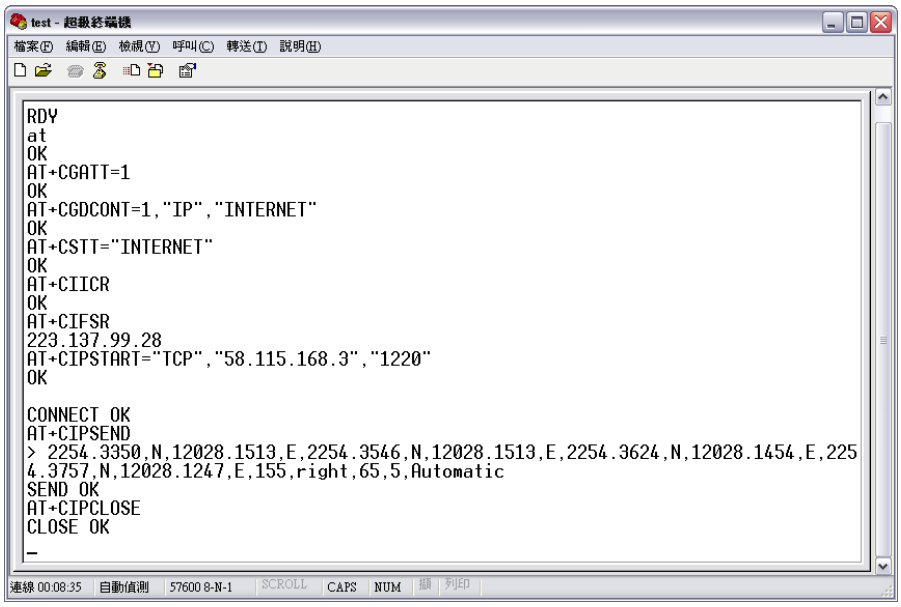


圖 4-15

終端機與主板相較，終端機可用來事先設定好 GPRS 的數值，例如改傳輸率或是模組內部設定的優先權。

4.3.4. GPRS 程式流程

程式開始先初始化，設定 RX 與 TX 腳位，傳 AT 指令啟動 GPRS 模組，下達一次命令，RX 會回傳字元，利用回傳字元設定檢查旗標來判斷指令是否有執行正確，檢查旗標正確後會繼續往下執行下一個 AT 指令，當連上網路時會收到基地台傳來的 LOCAL IP，設定連接 SERVER 的 IP 與 PORT，就可以連線開始傳送資料如圖 4-16 流程，從 FlyData 裡將 GPS、Compass 接收到的資料傳至地面的 Server，在 Server 端可以下達命令，Server 接收到資訊後便會回傳至 GPRS　RX，從 RX 存取資料以“@”為判斷字元，當“@”字元出現時便會開始擷取資料至通訊程式裡做資料的判斷分析，做好處理的資料會存回 FlyData，更新後的資料會再傳回地面的 Server。

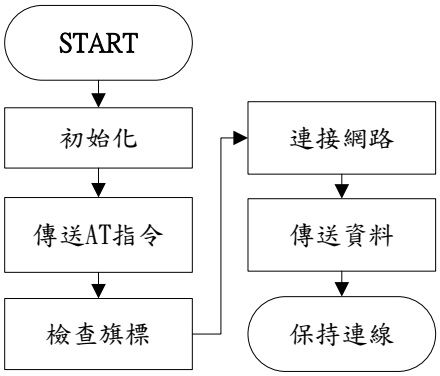


圖 4-16 GPRS 初始化連線

4.3.5. GPRS 程式開發研析

表 4-12 GPRS 開發研析

項目	開發研析	備註
1	換了中華電信的 SIM Card，成功連接網路	
2	利用 UART 的 TX 傳送 AT 給 GPRS 模組,有利用 DATASHEET 的 DEMO 程式做出陽春型的傳送。	之前都只使用 RX，對於中斷的模式很不清楚，加上 UART 的初始化的設定。
3	程式一次只能傳一個指令，TX 跟 RX 的狀況都不穩定，容易陷入 WHILE 迴圈出不來	由於是多個 DEMO 程式湊出來的，很多定義都不清楚或是重覆了
4	利用陣列來傳 AT 指令，並加入檢查旗標，發現有些旗標跟 DEMO 程式原先設的有重覆到，只要檢查旗標不過就陷入 WHILE 迴圈無窮盡了。	AT 指令結尾都要加\r\n Ctrl Z 表示法為\032
5	為了增加程式的準確性，檢查回傳值，錯誤比對和正確比對。雖然增加了比對陣列，但發現不是每個比對字元程式都會認識。傳 AT 指令目前只能傳一次要重新關開程式，沒辦法連續執行，太浪費時間於傳指令。	錯誤比對"RROR"，"FAIL" 正確比對"AT"，"OK"，"SE"，"on"
6	把 TX 中斷的內容程式，拉出中斷獨立 Funtion 執行，發現結果是可行的,為了要連續傳送 AT 指令,嘗試將主程式加入 WHILE 迴圈,開始了一連串無限迴圈的噩夢,原因是檢查程式沒寫好，測試了 RX 的回傳值，證實了程式只能辨別 SERVER 傳的字元	錯誤比對和正確比對是沒有用的。
7	基礎程式已經出來了,AT 指令回傳的時間有差異,有些指令回傳時間會延遲久一點，測試需要等待的回傳時間，盡可能地縮短等待時間，而 RX 跟 TX 也穩定了	程式整個大變身，發現可以藉由 SERVER 回傳帶回地面資料。
8	可以接收地面傳來的命令，加入確認旗標，避免掉資料。	程式測試成功但傳輸時間還是不夠快
9	將程式模組化，GPRS 程式初步完成	開始進行通訊程式
10	GPRS 將地面的命令、資料藉由 RX 帶回來,RX 會不斷的清空，因此通訊程式要去分析資料做分類，存入資料陣列裡	
11	通訊程式可以抓到 GPRS 的 RX 資料，開始進行判斷自動、手動，升降馬達與轉向馬達資料分析	通訊程式雛形完成，加入資料分析
12	將分析好的可用資料上傳到資料陣列裡。	
13	從 FlyData 裡要傳給 server 的資料，一個一個的抓下來，存入跟讀出功能都可以使用了。	
14	通訊程式模組化完成,與 GPRS 搭配使用,傳輸的穩定性有了，但時間還是有點慢，來回會花大約六秒。	傳輸時間越久越不利於自動飛行，還需要再修

		改
15	進行程式的局部修改，多餘的測試程式都拿掉，方便與母版搭配	

4.4. 模組程式－CMPS

4.4.1. I²C 通訊原理：

I2C 是由數據線 SDA 和時鐘 SCL 構成的串行，可發送和接收數據。在 CPU 與被控 IC 之間、IC 與 IC 之間進行雙向傳送，最高傳送速率 100kbps。各種被控制電路均並聯在這條上，但就像電話機一樣只有撥通各自的號碼才能工作，所以每個電路和模塊都有唯一的地址，在信息的傳輸過程中，I2C 上並接的每一模塊電路既是主控器（或被控器），又是發送器（或接收器），這取決於它所要完成的功能。CPU 發出的控制信號分為地址碼和控制量兩部分，地址碼用來選址，即接通需要控制的電路，確定控制的種類；控制量決定該調整的類別（如對比度、亮度等）及需要調整的量。這樣，各控制電路雖然掛在同一條上，卻彼此獨立，互不相關。

I2C 在傳送數據過程中共有三種類型信號，它們分別是：開始信號、結束信號和應答信號。

1. 開始信號：SCL 為高電位時，SDA 由高電位向低電位跳變，開始傳送數據。
2. 結束信號：SCL 為低電位時，SDA 由低電位向高電位跳變，結束傳送數據。

4.4.2. 概述

霍尼韋爾 HMC6343 電子羅盤模組內包含 3 軸磁阻感測器與 3 軸 MEMS 加速感測器，及必要的類比和數位支援電路和航向值計算法則。各個感測器元件、處理電子管件和硬體封裝在一個 9.0mm×9.0mm×1.9mm 的 LCC 元件內，如圖 4-17。規格見表 4-13。

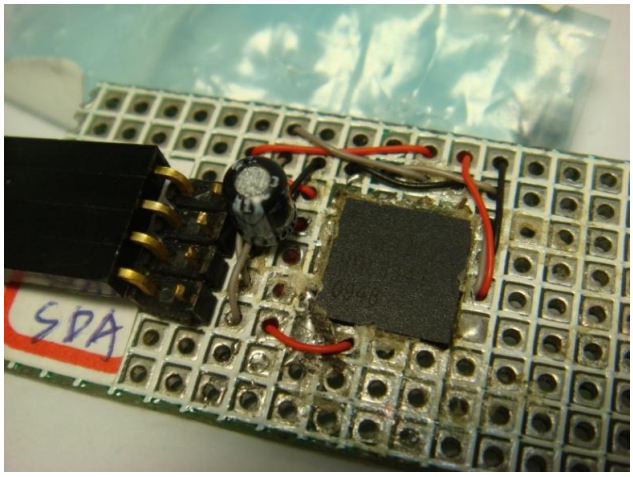


圖 4-17 HMC6343 電子羅盤

表 4-13 HMC6343 技術規格表

特性	條件*	最小值	標準值	最大值	單位

供電電壓	VDD 以接地為基準	2.7	3.0	3.6	V
電流	與所有 VDD 引線插腳連接在一起持續峰值 (0.5ms)		12	25	mA mA
磁場範圍	全部磁場		±1		gauss（高斯）
航向精度	水平 ±15°傾斜 ±60°傾斜		3.0 待定 待定		±度(°)
航向，解析度	輸出數值		0.1		度(°)
更新率	持續模式		10		Hz
傾斜範圍	從垂直方向		±80		度(°)
傾斜精度	0°至±15° ±15°至±60°		±1 ±待定		度(°)
傾斜解析度	輸出數值		0.1		度(°)
工作溫度	周圍環境	-20		80	℃
儲存溫度	周圍環境，無偏差的	-55		125	℃
重量			0.32		g
MSL	濕度的靈敏性等級		3		

*除非另有規定外均在 25℃時進行測試。

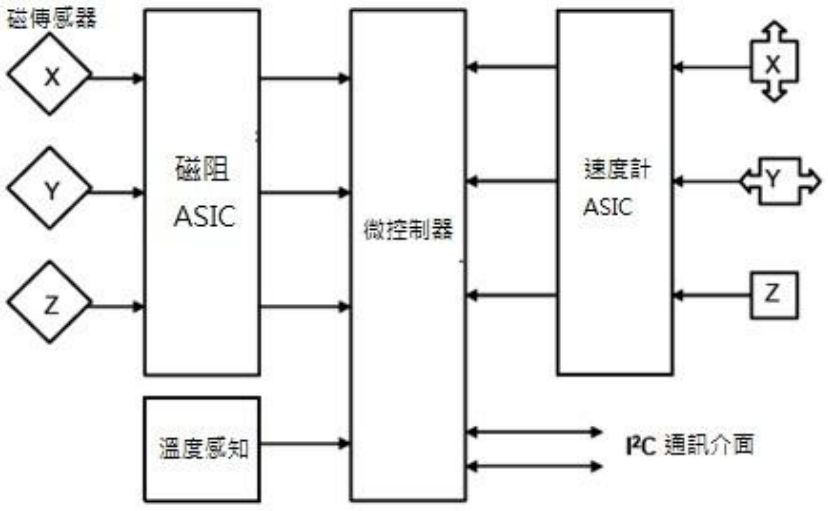


圖 4-18 功能圖

4.4.3. 指令參考表

表 4-14

指令	參數 1	參數 2	回應格式	說明
(0XF1)	EPROM 地址	數據		寫入 EEPROM (0×E1)
(0XE1)	EPROM 地址		數據(1 位元組)	從 EEPROM 讀取

(0X70)				進入使用者校準模式
(0X7F)				退出使用者校準模式
(0X72)				液位方向(X=向前,+Z=向上)(預設值)
(0X73)				向上一側的方向(X=向前，Y=向上)
(0X74)				向上平面向前的方向(Z=向前，-X=向上)
(0X82)				處理器重置
(0X45)			MSL/LSB(6 位元組)	位置 Mag 資料。MxMSB、MxLSB、MyMSB、MyLSB、MzMSB、MzLSB
(0X50)			MSL/LSB(6 位元組)	位置航向資料。航向 MSB、航向 LSB、縱傾 MSB、縱傾 LSB、橫滾 MSB、橫滾 LSB

4.4.4. 程式摘要

本電子羅盤模組程式根據 I2C 之通訊協定所撰寫而成，其流程大致規劃如下：先初始化 I2C 通訊介面，接著開啟一個 I2C 連線，先將設備位址送上資料線且告知狀態為寫入，這時等待 HMC6343 的回應，當確認回應後，就會在將要寫入的資料位址(指令)送上資料線給 HMC6343 接收，當此程序完成之後，會再將 I2C 重新啟動一次，接著再度送出資料位址等待回應，且將狀態設為讀取，當再度確認 HMC6343 的回應後，就會開始接收由 HMC6343 所接收的資料，資料接收經過位移計算後，便可得到需要的數值。

以下則列出一個標準的 HMC6343 溝通程式碼片段

順序	程式碼	說明
1	Cmps_Initial();	初始化 I2C
2	Cmps_Start();	啟動
3	Cmps_SetRW(D_WRITE);	呼叫裝置，設定狀態為"寫入"
4	Cmps_SetRW_Flag=FlagNO;	清除自定旗標
5	Cmps_WriteData(Command);	寫入"0X50"，這是 HMC6343 的指令
6	Delay(10);	Delay
7	Cmps_ReStart();	重新啟動
8	Cmps_SetRW(D_READ);	設定為讀取，接收
9	Delay(100);	
10	Cmps_GetDatas(6);	接收 6Byte 的資料
11	Cmps_Stop()	/中止 I^2C 通訊

4.4.5. Compass 程式開發研析

表 4-15 Compass 開發研析

項目	開發研析	備註
1	為了避免毀損高單價之 Compass IC，本研究先行利用同樣是 I^2C 傳輸介面的 DS1307 時間晶片嘗試利用 I^2C 與 PIC 溝通，在測試完成無誤後，再加入 Compass IC 模組。	I^2C DS1307 版
2	螢幕開始顯示數值，但部分亂碼	研判可能是時脈震盪器問題
3	已可成功抓取到時間，但時間不會跳動	
4	成功抓取時間，並可寫入設定，且時間準確。	原因出在抓取資料的 DELAY 時間
5	開始嘗試與電子羅盤 HMC6343 溝通	
6	可以找到 HMC6343，但卻無法與其成功建立資料連接	無法抓取各維度資料
7	重新更改程式架構，並參考 PIC 範例程式做修改，且將其各傳輸步驟模組化溝通	依然無法抓取資料。
8	嘗試各個資料地址寫入，並接收	抓取到亂碼，但會跳動
9	更改接收方式，加入 DELAY 函式，並將資料型態轉型，以利讀取格式	
10	經過轉型後的數值，已可成功讀取出來	
11	開始抓取各維度資料，並將其模組化，以利於主程式銜接	
12	發現更新數值有誤，且誤差值大開始進行修正	研判是 DELAY 太長，取樣率不足所造成。
13	修改 DELAY 時間，經誤差幅度及更新時間縮小	

4.4.6. Compass 程式開發過程

以 LCD 顯示 Compass 資訊，如圖 4-19 的”H： ”為 Heading，數值 3582 即 358.2 度之意，與指北針方向相同，在 360 度左右;”P： ”為 Pitching 角度，”R： ”為 Rolling 角度。

圖 4-19 LCD 顯示 Compass 資訊

第五章 地面監控

5.1. 地面 Server 建置

5.1.1. TCP 原理

傳輸控制協定（Transmission Control Protocol， TCP）

TCP 連線包括三個狀態：連線建立、資料傳送和連線終止。TCP 用三路握手（three-way handshake）過程建立一個連線。在連線建立過程中，很多參數要被初始化，例如序號被初始化以保證按序傳輸和連線的穩定性。

TCP 連線的正常建立一對終端同時初始化一個它們之間的連線是可能的。但通常是由一端開啟一個介面(socket)然後監聽來自另一方的連線，這就是通常所指的被動開啟(passive open)。伺服器端被被動開啟以後，使用者端就能開始建立主動開啟(active open)。詳細過程如圖 5-1。

1. 用戶端透過向伺服器端發送一個 SYN 來建立一個主動開啟，作為三路握手的一部分。
2. 伺服器端應當為一個合法的 SYN 回送一個 SYN/ACK。

3. 最後，用戶端再發送一個 ACK。這樣就完成了三路握手，並進入了連線建立狀態。

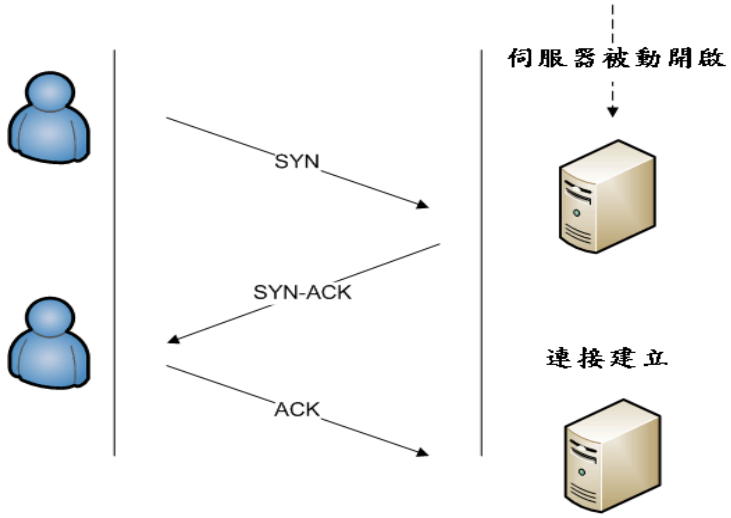


圖 5-1TCP 連線

5.1.2. Java－Mysql 資料庫存取原理

Java 資料庫連線，（Java Database Connectivity，簡稱 JDBC）JDBC 是 Java 與資料的連接。因為 ODBC 是完全用 C 語言編寫的，而 Java 中實現與 C 語言程式的通信是比較困難的，因此就產生了由 Java 語言編寫的用於 Java 程式與資料庫連接的介面技術。JDBC 與資料庫的連接 JDBC 與具體的某種資料庫連接，是直接通過 JDBC API 類別庫中的類別來自動裝載驅動程式的。此類別庫一般在 Java.sql 類別中，它包含了用於實現與資料庫連接的功能，包括與資料庫建立連接、傳送查詢和接受查詢結果。

監控站程式流程圖(圖 5-2)

程式一開啟後等待 GPRS 的連線，連線之後就可以開始下指令或收資料，資料寫進資料庫，如果連線中斷還會回到等待連線的地方再次等待連線。

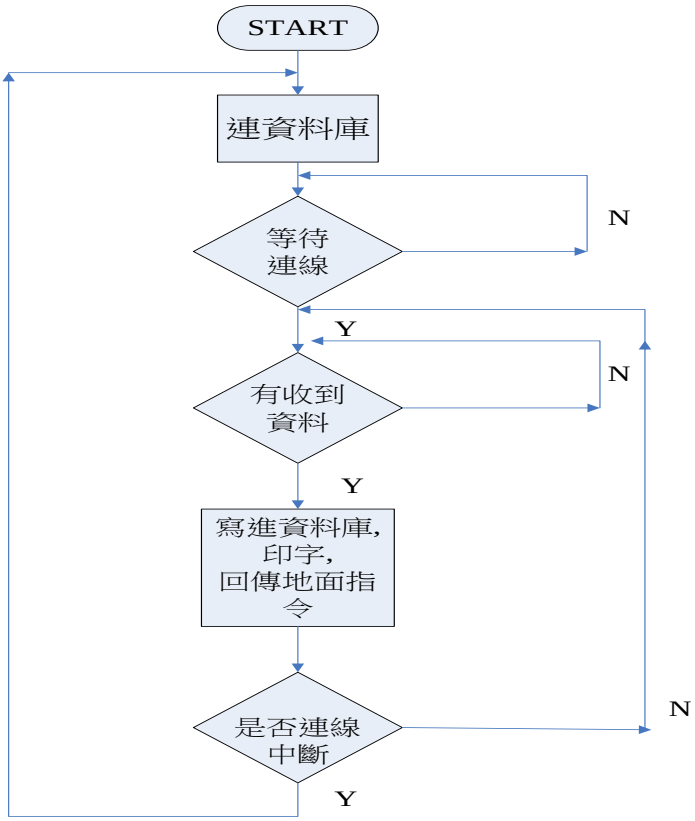


圖 5-2 監控站程式流程圖

5.2. Google Maps

5.2.1. Google Maps API Java Script

為了可以及時把飛船目前現在的位置顯示出來所以使用了 Google Maps 的一些開發套件，這些套件都是 Google 所提供的， 金鑰的部分只要加入 Google 的會員並且登入之後就可以取得，地圖上有左邊有可以拉近遠的控制項，右上則可以切換地圖、衛星、混合地圖、地形， Google Maps API 的網頁上有提供基本的範例程式可以自行參考。本專題研究重點為飛行船自動飛行系統，故 Google Maps API 建置方式請參閱附錄[2.]。圖 5-3 是使用 Java Script 做的 Google Maps 測試圖。

圖 5-3 使用 Java Script 做的 Google Maps

5.2.2. Google Maps Flash

Google Maps 也有提供 Google Maps API　Flash 版，金鑰的部分跟之前一樣只要登入會員就行了，基本範例程式 Google Maps API 的網頁上也有提供，範例畫面如圖 5-4，使用 Adobe Flash CS4 編輯。本專題研究重點為飛行船自動飛行系統，故 Google Maps API for Flash 建置步驟請參閱附錄[3.]。

圖 5-4

5.3.　監控站 Server 版本演進

最早之前用 VB 做的監控站(圖 5-5)，只能收的到資料存到文字檔而已，測試的時後是先模擬 1 個傳送端傳資料(圖 5-6)，所以沒有用 GPRS 測試過。此 VB 版本監控站傳輸介面格式說明如表 5-1。

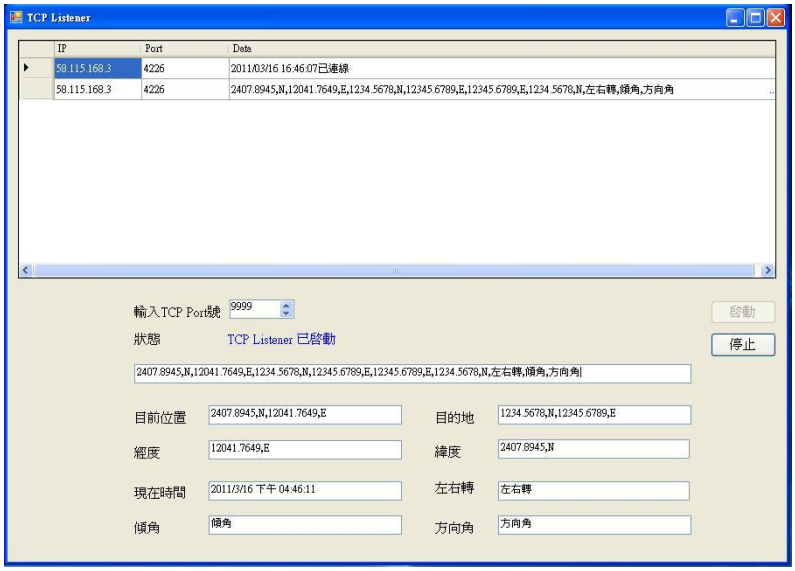


圖 5-5 VB Server

表 5-1 VB Server 格式說明

Server 上半部分	
IP	CLIENT 的 IP
PORT	CLIENT 的 PORT
DATA	CLIENT 傳的資料
Server 下半部分	
開放的 PORT	讓 CLIENT 進來的 PORT
目前狀態	紅字是目前未啟動，藍字是已經啟動
資料	全部收到的資料
目前位置	目前位置座標
目的地	目的地座標
經度	現在經度
緯度	現在緯度
現在時間	現在時間
左右轉	看收到的資料來判斷飛船是左轉還右轉
傾角	判斷目前飛船的傾斜角度
方向角	判斷目前的方向

圖 5-6 用來測試的 Client

圖 5-6 為測試 Client 端傳輸介面，表 5-2 **錯誤! 找不到參照來源。**為 Client 端之格式說明。

表 5-2 VB Client 格式說明

CLIENT	
狀態	紅字是連線中，藍字的還沒連線
遠端伺服器 IP	Server 的 IP
遠端伺服器 PORT	Server 的 PORT
要發送的文字	要傳送的資料

測試完成後改用 Java 撰寫先把資料存成文字檔，再用 GPRS 測試傳送資料成功存入文字檔(圖 5-7)， 表 5-3 為 Java 初版介面說明。

圖 5-7 Java 改寫後初版

表 5-3 Java 初版介面說明

清空文字框	把所有文字框清空
接收的資料	全部收到的資料都會印出來
目前位置	目前的經緯度，南北緯，東西經
目的地	目標的經緯度，南北緯，東西經
經度	目前經度
緯度	目前緯度
現在時間	現在時間
高度	目前 GPS 抓到的高度
左右轉	收到的資料來判斷飛船是左轉還右轉
傾角	判斷目前飛船的傾斜角度
方向角	判斷目前的方向
手動/自動	判斷目前是手動還是自動飛行

完成後的監控軟體畫面，可以下指令改變點，改變行列的資料，及操作飛船馬達升降角度以及馬達的出力%數，並將資料都存進了資料庫。

- 完成版監控站

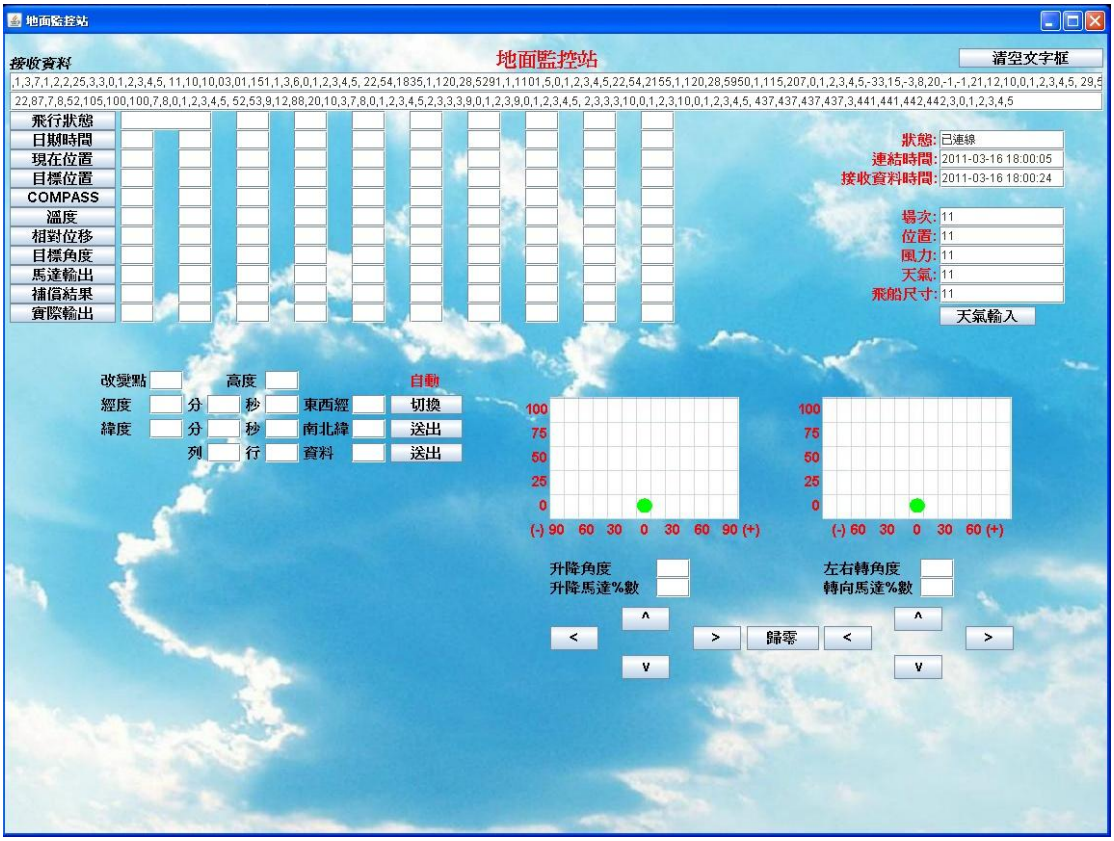


圖 5-8 完成版的監控站

操作介面(圖 5-8)：介面部分是由文字框，按鈕以及標籤所組成的，透過 GPRS 把資料傳送到地面站然後先分割後再把資料分別放置在分類好的文字框內，由於文字框過多所以在分類的資料標題有設個按鈕可以顯示現在那行是有什麼資料，介面右上的部分有顯示目前狀態，連結時間和接收資料時間，比較重要的是測試場次、位置、風力、天氣飛船尺寸，這些都要在一開啟時就要先輸入因為這些都要存進資料庫，要注意的是資料庫不能有空值，如果是空值整個程式就會出現錯誤，無法將資料存進資料庫，下面的部分是控制飛船的方式，按鈕上下是馬達出力%數左右是轉的角度，改變點的話必須輸入經緯度，高度然後送出等收到資料後就會把這些指令傳上去給板子，板子再對飛船做出動作。表 5-4 為 Java 完成版監控站介面說明。

表 5-4 Java 完成版介面說明

Server 上半部分	
清空文字框	把所有文字框清空
接收資料	全部收到的資料都會印出來
飛行狀態	自動/手動，地控，目標點，終點，過熱，轉向不足，(升降)，轉向不足(左右)，高度超出， 轉過頭(升降)，轉過頭(左右) **從這可以了解飛船目前的情況是手動/自動還是地控，目標點是第幾點，馬達是**

	否過熱目前升降和轉向不足多少度，高度超出值，以及升降和左右轉過頭多少度
日期時間	**日，月，年，時，分，秒，可用，A，三維，MSG** **記錄GPS的更新日期時間，可用是指GPGGA的定位裝態，A是** **GPRMC的定位狀態，三維是代表GPGSA的定位狀態，MSG是磁極** **變量**
現在位置	緯度，分，秒，南北緯，經度，分，秒，東西經，高度，雜訊比 **記錄目標點緯度、經度與高度資料，以及飛行高度限制範圍** **雜訊比目前是沒有用到**
目標位置	緯度，分，秒，南北緯，經度，分，秒，東西經，高度，飛行高度範圍 **記錄目標點緯度、經度與高度資料，以及飛行高度限制範圍**
COMPASS	存放的資料有：（HEAD），（PITCH），（ROLL），動作角度(HEAD)， 動作角度(PITCH) 動作方向(左右)，動作方向(上下)，方向角轉過頭角度，仰角轉過頭角度，高度差 **從這邊可以看到 COMPASS 上抓到的 Heading，Pitching，Rolling 的數值，目前動作的角度，動作左右上下，判斷方向角和仰角如果超過多少度就算是轉過頭，目前跟目標點的高度差**
溫度	COMPASS 溫度，溫度上限，預設轉角 1，預設轉角 2 **了解目前溫度情況，預設轉角是指若實際轉角小於預期轉角則代表轉向不足，推力或角度需增加**
相對位移	GPS 速度，GPS 方向，2D 距離，3D 距離，馬達出力倍率(前)，馬達出力倍率(後) **這些可以了解目前 GPS 抓到的速度，方向，速度計算，方向計算，2D 距離，3D 距離，馬達出力前後的結果**
目標角度	北夾角目標，船身夾角，行進角度範圍(左右)，行進角度範圍(升降)，推力最大角度(HEAD)， 推力最大角度(PITCH)，推力最小值(HEAD)，推力最小值(PITCH) **經判斷後與北邊的夾角是多少度和確認目標在哪邊 ，目前行** **進角度範圍(左右) 、行進角度範圍(升降) ，行進角度範圍：** **在此範圍內不進行角度或升降角度修正〈防止快速反覆動作〉** **最大推力角度：目標角度差超過此數值則推力增至最大值** **此數值是指這個欄位的值** **推力最小值：和目標角度差超過此數值則推** **力減至最小值**
馬達輸出	馬達 1，馬達 2，馬達 3，馬達 4，命令 1，命令 2，命令 3，命令 4 **自動飛行計算時結果，馬達 1 是升降轉向馬達，馬達 2 是左右轉向馬達，馬達 3 是升降推進馬達，馬達 4 是後轉向推進馬達** **通訊接收指令後** **命令 1 是接收升降轉向馬達的指令** **命令 2 是接收左右轉向馬達的指令**

	命令 3 是接收升降推進馬達的指令 **命令 4 是接收後轉向馬達的指令**
補償結果	馬達 1，馬達 2，馬達 3，馬達 4，命令 1，命令 2，命令 3，命令 4 **經過微調後結果馬達 1 是升降轉向馬達，馬達 2 是左右轉向馬達，馬達 3 是升降推進馬達，馬達 4 是後轉向推進馬達** **通訊接收指令後** **命令 1 是接收升降轉向馬達的指令** **命令 2 是接收左右轉向馬達的指令** **命令 3 是接收升降推進馬達的指令** **命令 4 是接收後轉向馬達的指令**
實際輸出	PWM1，PWM2，PWM3，PWM4，推力可漸變值，通訊接收輸出 1，通訊接收輸出 2，通訊接收輸出 3，通訊接收輸出 4，推力後漸變值 **這邊可以知道實際的輸出 pwm 數值和通訊接收輸出 pwm 值，漸變值是馬達要到達下一個動作時改變的速度**
狀態	判斷是否連線
連結時間	顯示連結時間
接收資料時間	顯示收到資料的時間
天氣輸入	依照場次，位置，風力，天氣，飛船尺寸輸入
Server 下半部分	
切換	用來切換手動/自動的按鈕
送出按鈕 1	用來改變點的按鈕，需先輸入 經緯度，和高度等資料
送出按鈕 2	用來改變資料行列中的數值
↑	升降馬達或是左右轉馬達增加出力%數
↓	升降馬達或是左右轉馬達減少出力%數
←	減少升降或是左右轉的角度
→	增加升降或是左右轉的角度
歸零	全部歸零

- Mysqly 資料庫

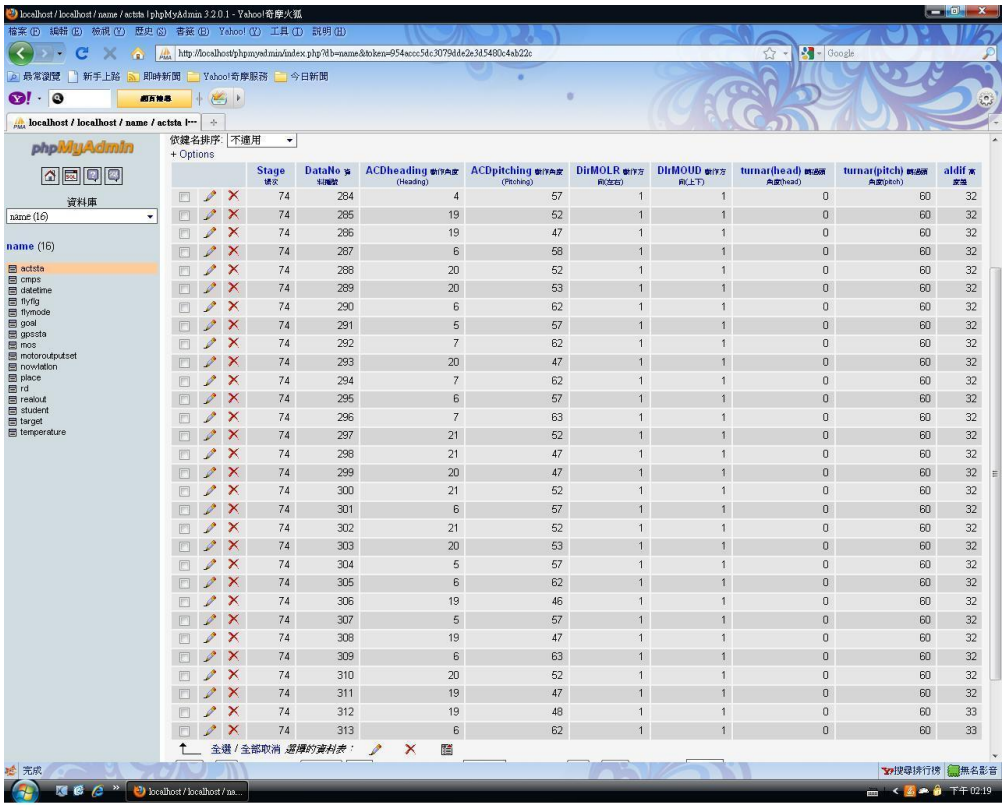


圖 5-9 資料庫內資料畫面

Mysql 資料庫(圖 5-9)：

首先要先上網抓 Mysql 所提供 Java 用的驅動程式檔(Mysql-connector-java-5.1.3) ，然後放在 C：\Program Files\java\jdk1.6.0_18\jre\lib\ext 資料夾底下(通常是沒選安裝路徑時後預設的) ，驅動程式的部分是依照提供 Mysql 服務的 WampServer 軟體所使用的版本，資料庫部分是用 WampServer 上提供的，可以直接在上面建立資料庫，最後只要在 Java 編輯軟體上寫程式對應哪個資料表，哪個欄位就可以把資料寫進資料庫，要注意的是資料庫要先建好，類型是數字還是文字也都要先設定好，否則在執行的時候會一直出錯。

5.4. 資料庫建置

資料庫裡總共有 15 個資料表，每個資料表所屬欄位都有欄位註解，如下表。

1. 動作狀態：存放目前動作的角度，動作左右上下，是否轉過頭與目前跟目標點的高度差，如表 5-5 所示。

表 5-5 動作狀態格式

actsta				
欄位	型態	Null	預設值	註解
Stage	int(3)	否		場次
DataNo	int(5)	否		資料編號
ACDheading	int(3)	否		動作角度(Heading)
ACDpitching	int(3)	否		動作角度(Pitching)
DirMOLR	int(2)	否		動作方向(左右)
DIrMOUD	int(2)	否		動作方向(上下)
turnar(head)	int(3)	否		轉過頭角度(head)
turnar(pitch)	int(3)	否		轉過頭角度(pitch)
aldif	int(3)	否		高度差

2. Compass：存放從 COMPASS 上抓到的 Heading，Pitching，Rolling 的數值，如表 5-6 所示。

表 5-6 Compass 格式

cmps				
欄位	型態	Null	預設值	註解
Stage	int(3)	否		場次
DataNo	int(5)	否		資料編號
Heading	int(3)	否		Heading
Pitching	int(3)	否		Pitching
Rolling	int(3)	否		Rolling

3. 日期：存放目前時間日，月，年，時，分，秒，如表 5-7 所示。

表 5-7 日期格式

datetime				
欄位	型態	Null	預設值	註解
Stage	int(3)	否		場次
DataNo	int(5)	否		資料編號
day	int(2)	否		日
month	int(2)	否		月
year	int(2)	否		年
hour	int(2)	否		時
minu	int(2)	否		分
second	int(2)	否		秒

4. 飛行旗標：存放資料有 Msg(詳見表 3-8)，判斷目前是不是地面操作，是否到達終點，馬達是否過熱，目前升降和左右是否轉向不足，高度超出值，以及升降和是否左右轉過頭，如表 5-8 所示。

表 5-8 飛行旗標格式

flyflg				
欄位	型態	Null	預設值	註解
Stage	int(3)	否		場次
DataNo	int(5)	否		資料編號
Msg	int(1)	否		Msg
Grocon	int(1)	否		地控
End	int(1)	否		終點
OvHeat	int(1)	否		過熱
UndsterUpDown	int(1)	否		轉向不足(升降)
UndsterLR	int(1)	否		轉向不足(左右)
Exheit	int(1)	否		高度超出
TurnUpDown	int(1)	否		轉過頭(升降)
TurnLR	int(1)	否		轉過頭(左右)

5. 飛行模式目標：紀錄目前是自動/手動和目標點，如表 5-9 所示。

表 5-9 飛行模式目標格式

flymode				
欄位	**型態**	**Null**	**預設值**	**註解**
Stage	int(3)	否		場次
DataNo	int(5)	否		資料編號
AutoManu	int(1)	否		自動/手動
Tarp	int(3)	否		目標點

6. 目標點資料：此表記錄目標點緯度、經度與高度資料，以及飛行高度限制範圍，如表 5-10 所示。

表 5-10 目標點資料格式

goal				
欄位	**型態**	**Null**	**預設值**	**註解**
Stage	int(3)	否		場次
DataNo	int(5)	否		資料編號
Lat	int(2)	否		緯度(度)
LatMinu	int(2)	否		緯度(分)
LatSec	int(4)	否		緯度(秒)
NorS	int(2)	否		南北緯
Lon	int(3)	否		經度(度)
LonMinu	int(2)	否		經度(分)
LonSec	int(4)	否		經度(秒)
EorW	int(2)	否		東西經
height	int(4)	否		高度
Alhrange	int(4)	否		飛行高度範圍

7. GPS 狀態：存放從 GPS 中抓到的 GPGGA 的狀態、GPRMC 的狀態、GPGSA 的狀態、NOISE 後來沒用到是 GPS 的雜訊比，如表 5-11 所示。

表 5-11 GPS 狀態格式

gpssta				
欄位	**型態**	**Null**	**預設值**	**註解**
Stage	int(3)	否		場次
DataNo	int(5)	否		資料編號
GGAsta	int(1)	否		GPS 中 GPGGA 的狀態
RMCsta	int(1)	否		GPS 中 GPRMC 的狀態
GSAsta	int(1)	否		GPS 中 GPGSA 的狀態
Noise	int(1)	否		GPS 的雜訊比

8. 微調後馬達指令：存放經過微調後 PWM 的輸出值和通訊接收指令後的輸出值，如表 5-12 所示。

表 5-12 微調後馬達指令格式

mos				
欄位	**型態**	**Null**	**預設值**	**註解**
Stage	int(3)	否		場次
DataNo	int(5)	否		資料編號
PWM1	int(3)	否		PWM1 輸出值
PWM2	int(3)	否		PWM2 輸出值
PWM3	int(3)	否		PWM3 輸出值
PWM4	int(3)	否		PWM4 輸出值
ComuReOut1	int(3)	否		通訊接收輸出 1
ComuReOut2	int(3)	否		通訊接收輸出 2
ComuReOut3	int(3)	否		通訊接收輸出 3
ComuReOut4	int(3)	否		通訊接收輸出 4

9. 自動判斷馬達指令:存放程式判斷後 4 顆馬達的輸出結果和通訊接收命令,如表 5-13 所示。

表 5-13 自動判斷馬達指令格式

motoroutputset				
欄位	**型態**	Null	**預設值**	**註解**
Stage	int(3)	否		場次
DataNo	int(5)	否		資料編號
Motor1	int(1)	否		馬達 1
Motor2	int(1)	否		馬達 2
Motor3	int(1)	否		馬達 3
Motor4	int(1)	否		馬達 4
Cmd1	int(1)	否		命令 1
Cmd2	int(1)	否		命令 2
Cmd3	int(1)	否		命令 3
Cmd4	int(1)	否		命令 4

10. 現在經緯度:目前的緯度(度) ,緯度(分) ,緯度(秒) ,南北緯,經度(度) ,經度(分) ,經度(秒) ,東西經,高度,如表 5-14 所示。

表 5-14 現在經緯度格式

nowlatlon				
欄位	**型態**	Null	**預設值**	**註解**
Stage	int(3)	否		場次
DataNo	int(5)	否		資料編號
Lat	int(2)	否		緯度(度)
LatMinu	int(2)	否		緯度(分)
LatSec	int(4)	否		緯度(秒)
NorS	int(2)	否		南北緯
Lon	int(3)	否		經度(度)
LonMinu	int(2)	否		經度(分)
LonSec	int(4)	否		經度(秒)
EorW	int(2)	否		東西經
height	int(4)	否		高度

11. 場地天氣:存放地面監控站輸入的飛測位置,目前風力,天氣,飛船尺寸,如表 5-15 所示。

表 5-15 場地天氣格式

place				
欄位	型態	Null	預設值	註解
Stage	int(3)	否		場次
Local	char(20)	否		位置
Wind	int(11)	否		風力
weather	char(10)	否		天氣
Size	int(3)	否		飛船尺寸

12. 方向距離：存放資料目前 GPS 抓到的速度，方向，速度計算，方向計算，2D 距離，3D 距離，馬達出力前後的結果，如表 5-16 所示。

表 5-16 方向距離格式

rd				
欄位	型態	Null	預設值	註解
Stage	int(3)	否		場次
DataNo	int(5)	否		資料編號
Speed	int(5)	否		GPS 速度
Dire	int(5)	否		GPS 方向
Sprat	int(5)	否		速度計算
Dircal	int(3)	否		方向計算
2Ddis	int(5)	否		2D 距離
3Ddis	int(5)	否		3D 距離
MoOutRate1	int(5)	否		馬達出力倍率(前)
MoOutRate2	int(5)	否		馬達出力倍率(後)

13. 實際輸出：目前馬達實際輸出數值和通訊接收輸出值，漸變值是馬達要到達下一個動作時改變的速度，如表 5-17 所示。

表 5-17 實際輸出格式

realout				
欄位	**型態**	**Null**	**預設值**	**註解**
Stage	int(3)	否		場次
DataNo	int(5)	否		資料編號
PWM1	int(3)	否		PWM1 輸出值
PWM2	int(3)	否		PWM2 輸出值
PWM3	int(3)	否		PWM3 輸出值
PWM4	int(3)	否		PWM4 輸出值
Thvach	int(1)	否		推力前漸變值
ComuReOut1	int(3)	否		通訊接收輸出 1
ComuReOut2	int(3)	否		通訊接收輸出 2
ComuReOut3	int(3)	否		通訊接收輸出 3
ComuReOut4	int(3)	否		通訊接收輸出 4
Athvach	int(1)	否		推力後漸變值

14. 目標點資料：經判斷後北夾角目標(head)，船身夾角(pitch) ，行進角度範圍(左右) ，行進角度範圍(升降) ，馬達推力最大角度，馬達推力最小值，如表 5-18 所示。

表 5-18 目標點資料格式

target				
欄位	**型態**	**Null**	**預設值**	**註解**
Stage	int(3)	否		場次
DataNo	int(5)	否		資料編號
Nrang	int(3)	否		北夾角目標(head)
Hulang	int(3)	否		船身夾角(pitch)
ScopangleLR	int(3)	否		行進角度範圍(左右)
ScopangleUD	int(3)	否		行進角度範圍(升降)
MaxAngthA	int(2)	否		推力最大角度
MaxAngthB	int(2)	否		推力最大角度
MinthA	int(2)	否		推力最小值
MinthB	int(2)	否		推力最小值

15. 溫度：存放資料有 Compass 溫度，溫度上限，如表 5-19 所示。

表 5-19 溫度格式

temperature				
欄位	型態	Null	預設值	註解
Stage	int(3)	否		場次
DataNo	int(5)	否		資料編號
CmpsTemp	int(4)	否		Cmps 溫度
TempLim	int(4)	否		溫度上限

5.5. 飛行人機介面與開發研析

表 5-20 飛行人機介面問題研析表

項目	開發研析	備註
1	用 VB 所做的，可以收資料和存到記事本，並沒有實際用 GPRS 連線成功過。	如**錯誤! 找不到參照來源。**。
2	根據之前 VB 所做的達到一樣的功能。	存到筆記本的時候會無法一筆一筆的存，會把上一筆收到的資料覆蓋掉。
3	可以由 GPRS 送資料到監控站來並且儲存到筆記本內。	先前無法一筆一筆存到筆記本的問題已解決。
4	重新改寫，新增了許多空白的文字框，還有可以下指令改變點，改變行列的資料，還有操作飛船馬達升降角度以及馬達的出力%數。	要把存入筆記本改成存進資料庫的部分，還在研究中。
5	可以把資料改成存進資料庫。	
6	新增可以輸入場次、位置、風力、天氣飛船尺寸。	目前不能輸入文字，只能輸入數字。
7	場次、位置、風力、天氣飛船尺寸改成文字輸入皆可以。	

5.6. Flash 動態展示使用工具

5.6.1. PHP 簡介

PHP 是 Hypertext Preprocessor 的縮寫，中文可以譯為「超文本預處理器」，基本上來說 PHP 的程式是在 Server 端執行的，在執行完後會依照程式碼的功能轉換成一般的網頁格式，所以一般使用者只能看見結果，而無法知道 PHP 程式碼是如何編寫或運作的，因此，被廣泛地運用在網頁程式的撰寫，專門應用於動態網頁的製作、以及存取資料庫的互動式網頁之開發上。而且，PHP 可以在大多數 Unix、Linux 和 Windows 等系統平台上完全運行，與 ASP、JSP、Cold Fusion 等 Script 動態網頁開發平台極為相似。

PHP 是一種伺服端（Server-side）、跨平台（Cross-platform）、易學易用的 HTML 嵌入式描述語言（HTML embedded scripting language）。由於這種在伺服器上執行 Script 語法的特性，讓 PHP 不需事先經過編譯就可以直接執行，加上它對 HTML 的支援，可以將 Script 結合在網頁中執行與運用，因此，更符合了現今所有軟體必須整合於 Web 環境的實現！此外，擁有跨平台以及與網站伺服器（Web Server）的結合特性，提供了多種連結資料庫的介面，諸如：MYSQL、MicrosoftSQL、Sybase、Informix、PostgreSQL、InterBase 等。

5.6.2. Wamp 簡介

所謂 Wamp 是指：w=window(視窗軟體)，a=apache2(伺服器軟體)，m=Mysql(資料庫)，p=php(網站程式語言)，總體來說，Wamp 是在 windows 環境下所存在，它允許你在 apache2 平台上，使用 php 語言，開發動態 web 應用程式；並且整合 php 語言所搭配的資料庫軟體 -MYSQL，而且內附 phpMyAdmin 方便管理 Mysql。

5.6.3. Flash CS4 簡介

Flash 是 Adobe 公司開發的網頁多媒體製作軟體，向量繪圖與動畫編輯功能，簡易地製作連續動畫、互動按鈕、繪圖與音效，可以不需要任何程式腳本即可在網頁中增加互動式多媒體。互動式的動畫和影音同步效果使網頁繪圖更加生動活潑，使用了 Flash 製作的任何物件，皆可以時間軸與動態路徑的動畫設計方式，由淺入深。只有以向量為基礎的 Flash 多媒體，才能流暢地呈現在 Internet 上，即使放大縮小也不降低本身品質。

Flash 主要有以下幾點特性：

1. 使用向量式圖形技術來製作動畫，它使檔案容量較小（因為向量圖形是使用數學函數來記錄圖形中的屬性）；而且將向量圖放大或縮小，也不會失真，最重要的是檔案容量也不會改變。
2. 可為網頁配上悅耳動聽的音效，而且是以 MP3 的音樂壓縮格式壓縮，可大幅降低聲音所佔據的檔案容量，也可保有高品質的音質。(可匯入 WAV (WIndows) 的聲音檔)。
3. 採用 Stream 資料流傳送方式，在檔案下載的同時即可流暢的播放，不須等到資料全部下載完畢才能觀看動畫。
4. 提供 Actions 指令設定環境，可使網頁作到極佳的互動性。
5. 為了讓使用者可以在完成動畫之後，立即看到動畫在網頁中的效果，Flash 可直接將動畫出版成網頁，產生 HTML。
6. 具有抗鋸齒的功能，可讓文字或影像的邊緣都非常平滑。
7. 加強與支援點陣式圖形處理 (Enhanced Bitmap Support)，使之可旋轉、拉長等功能。

5.6.4. PHP 程式設計

本研究，初次接觸 php 程式設計，起初為了熟悉撰寫方法及為專題研究所需做準備，選擇一些實用的範例做練習，並搭配 wamp 一起使用，完成簡易小網站，如：99 乘法表、計算機、聊天室、井字遊戲。而在熟悉語言後，便開始著手專題本身所需的程式。

本研究 PHP 程式開發研析，如表 5-21。

表 5-21 PHP 程式問題研析表

項目	開發研析	備註
1	載入文字資料，將自定格式的測試資料存成文字檔，以 php 程式讀取後，依格式輸出在網頁上。用於版本１。	把自定格式的假資料存成 txt 檔,按照格式抓取資料印出在網頁上。
2	載入資料庫飛行資料，以 php 程式讀取後，依格式輸出在網頁上。用於版本 6。	按照查詢指令,找出資料位置，並按資料格式抓取資料,最後依照 flash 指定格式印出在網頁上,為方便不同場景抓取資料共分成 5 支 php 程式： ConnectTest(連結資料庫)、 sql_connect(進入資料庫)、 left(近景資料載入專用) top(中景資料載入專用) right(遠景資料載入專用) (程式碼請查詢附錄 5.6.5)

5.7. Flash 動態展示平台設計

5.7.1. Flash 動態展示系統建置開發研析

表 5-22 Flash 動態展示系統建置開發研析表

項目	開發研析	備註
1	載入資料庫資料測試	利用 php 載入資料庫資料，回傳給 flash 使用。
2	三種視角觀測飛船	設計三種視角場景於一畫面，並且將飛船各種飛行資訊以小圖示呈現。
3	新增說明按鈕	新增說明文字按鈕，利用滑鼠事件與小圖示物件作相對應動作來說明圖示資訊。
4	動畫觀看舒適度檢測及改善	新增場景切換按鈕，依場景切換呈現畫面，提高畫面觀看的舒適度。
5	動畫流暢度檢測及改善	移除地版物件。
6	畫面協調性檢測及改善	更改切換場景按鈕的樣式。 新增移入物件彈出說明文字框功能。

5.7.2. 動畫場景設計

1. 動畫版本演進概述

當飛航高度過高，飛行船不在可視距離內時，檢視飛行船飛行資訊不易，傳送進資料庫的數據繁多也不易查詢，為了能更清楚的了解現在飛船飛行資訊，便設置了此系統。

然而為了使展示系統同時擁有美觀與傳遞資訊功能，在設計中多次修改，大略分為六種版本：

版本 1 為最初簡易版，為了測試連結 php 載入資料庫資料的功能，並給控制模組測試用，故整體設計內容都以資料為主，如表 5-23 所示。

表 5-23 版本 1 場景畫面

場景畫面	
	
出發時間	飛船出發時間，依照西元年月日顯示
出發位置	飛船出發座標，依照「度度，分分，分分分分」 東西經或南北緯顯示
中繼站	飛船下一個飛行目標的座標，依照「度度分分，分分分分」 東西經或南北緯顯示
終點站	飛船最後的飛行目標，依照「度度，分分，分分分分」 東西經或南北緯顯示
目前位置	即飛船飛行的當下座標，依照「度度分分，分分分分」 東西經或南北緯顯示
已花費時間	從飛船出發開始計算至當下共花費的時間，依照時分秒顯示

時速	飛行速度，依照每小時幾公里顯示
已行走距離	從飛船出發開始計算至當下共行走的距離，單位為公尺
高度	目前飛船飛行高度，單位為公尺
傾角	飛船與水平線夾角。
方向角	飛船水平轉向角度。

版本 2 的展示系統，為了讓畫面達到更佳的展示效果，分別設計了中景 (rear view)飛船後方視角、近景 (side view)飛船側方視角、遠景 (bird view)飛船上方視角，用三種不同視角呈現飛船即時動態，在展出資訊的同時希望畫面不會過於繁雜，故將資訊皆用小圖示配合數據表示。而在製作過程中發現，資訊呈現雖然詳細，其圖示卻無法使觀看者清楚了解含意，故演進為版本 3，增加了說明文字(彩虹按鈕)，依照滑鼠指向按鈕上的文字，其對應物件會發亮表示位置，藉此幫助使用者了解圖示。

由於加入彩虹按鈕使畫面更加擁擠，故版本 4 把三種場景分為三個部份，增加近中遠按鈕，可利用按鈕選擇要看的場景，各景有專用的彩虹按鈕，在移入按鈕時更容易注意相對物件的動畫效果。

在版本 4 中，展示平台動畫在播放時速度異常的緩慢，Flash cs4 軟體不穩定，常常因為不明原因發生錯誤必須強制關閉，導致作業上無法順利進行，軟體不穩成了最嚴重的問題，並且慢速播放讓展示平台的即時性大幅下降，在多次檢查及修改後，發現問題癥結是地板物件，此物件中大量的補間動畫拖慢整體速度，並因為每次播放時一次處理太多補間動畫導致軟體不穩，故將中景飛船畫面改為無地板空中飛行，成為了版本 5。

最後的版本為版本 6，經由協調性檢測，發現彩虹按鈕的畫面與中景 (rear view)以外的場景有著過度的差異化，使畫面轉到近景 (side view)及遠景 (bird view)時產生不協調感，故省略整個彩虹按鈕物件，並將近中遠按鈕改為簡約風英文按鈕，說明部分改為移入物件彈出文字框說明，提高整體畫面協調。(版本 2 至版本 5 請查閱附錄)

2. 版本 6 展示系統

(1). 版本 6 場景畫面解說，如表 5-24 所示

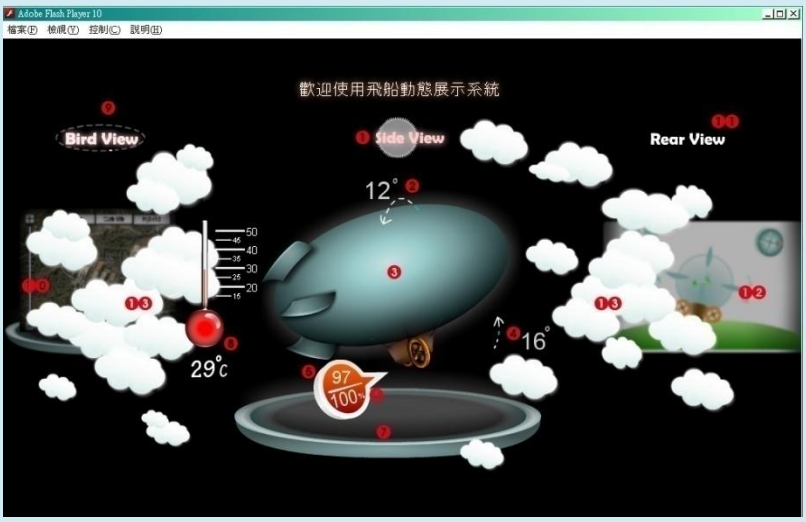
表 5-24 版本 6 場景畫面

進場畫面 1	**進場畫面 2**

進場畫面 3(rear view)

1. 晶片控制
2. 飛船自動系統
3. 離目標點距離差
4. 離目標點高度差
5. 方向角(方向+角度)
6. rear view 飛船
7. 轉角(方向+角度)
8. 說明框
9. 指北針
10. rear view 按鈕 (按下)
11. side view 按鈕
12. side view
13. bird view
14. 地板
15. 雲
16. Bird view 按鈕

進場畫面 4(Side view)



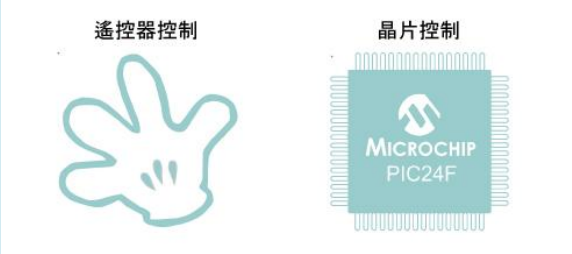
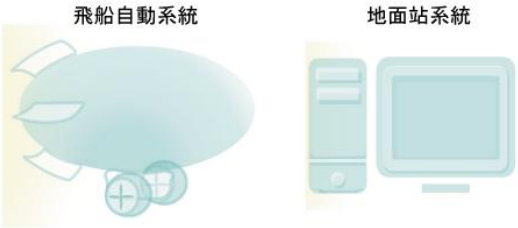
1. Side view 按鈕(按下)
2. 轉角(方向+角度)
3. Side view 飛船
4. 傾角(方向+角度)
5. 推力
6. 馬達輸出倍率
7. 旋轉底盤
8. 晶片溫度+度數
9. Bird view 按鈕(移入)
10. Bird view
11. Rear view 按鈕
12. Rear view
13. 雲

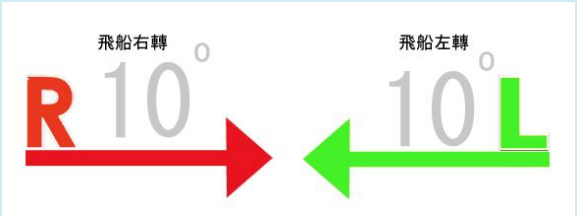
進場畫面 5(Bird view)

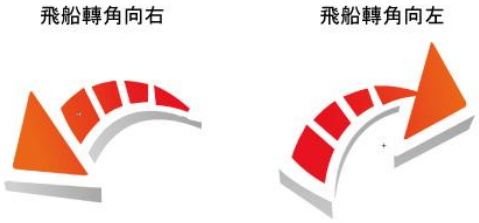
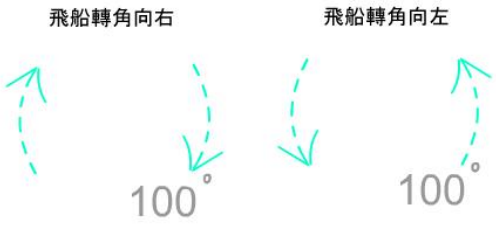
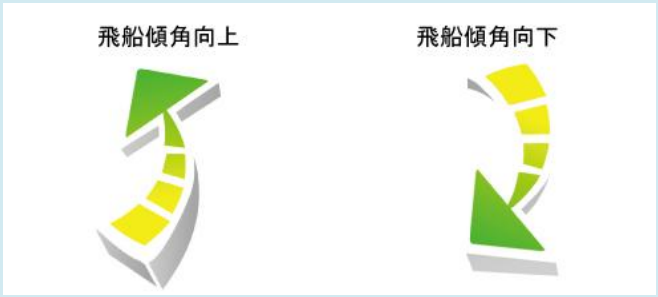
1. Bird view 按鈕(按下)
2. Rear view 按鈕
3. Side view 按鈕
4. 飛行軌跡
5. 飛船位置
6. Rear view
7. Side view
8. 雲

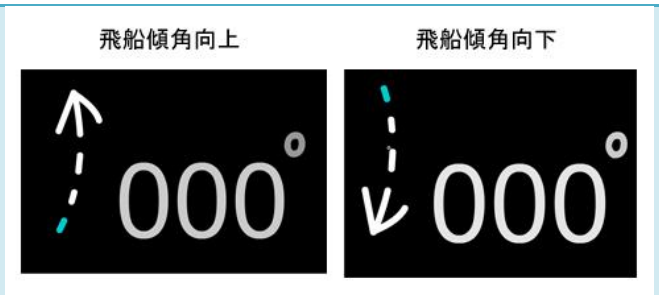
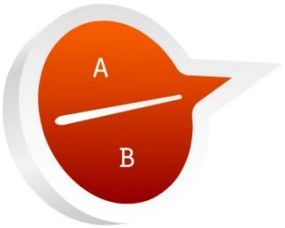
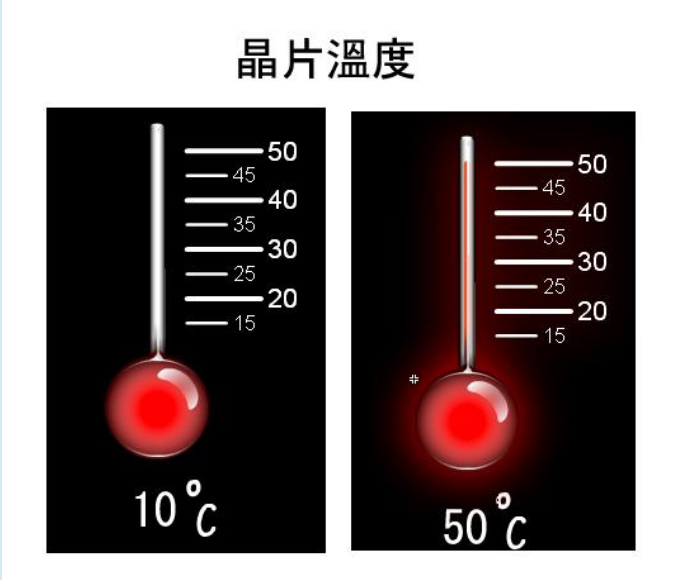
(2). 場景物件解析，如表 5-25 所示。

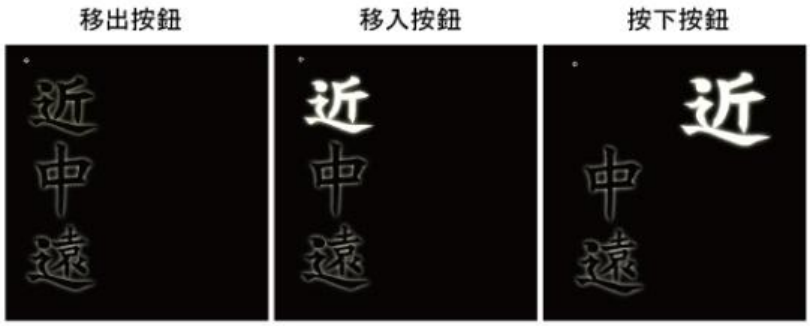
表 5-25 版本 6 場景物件解析

項目	說明
控制系統 (手動/自動)	物件動畫會依照飛船的飛行資料做變動。 手動-晶片控制：由控制模組控制飛船動作 自動-遙控器控制：使用無線搖控器控制飛船動作 
控制系統 (飛控/地控)	物件動畫會依照飛船的飛行資料做變動。 飛控-飛船自動系統：飛船自動飛行狀態 地控-地面站系統：由地面站操作飛行狀態 
雜訊比	物件動畫會依照飛船的飛行資料做變動。依照 GPS 收訊的良好度做變化。 (在版本 6 中因實測時未記錄此資料，故刪除)

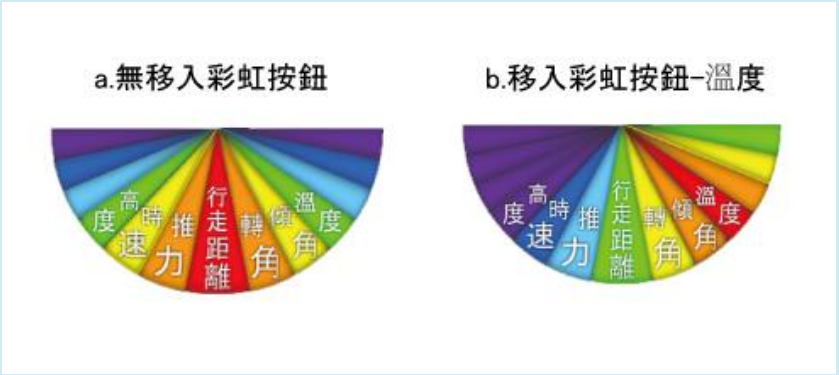
	
指北針	物件動畫會依照飛船的飛行資料做變動。指北針上 N 的方向為北方，指針會依照北方的方位轉動。
中景飛船	物件動畫會依照飛船的飛行資料做變動。當進入動畫時會有飛行動態動畫，並根據飛船的轉角角度急轉角方向做轉動。 
距離	物件動畫會依照飛船的飛行資料做變動。上方數字為與目標的距離差，下方數字為與目標的高度差。 
方向角	物件動畫會依照飛船的飛行資料做變動。當飛船方向角變化時會根據轉向顯示動畫，字母顯示方向，度數為移動角度。 。 
轉角	物件動畫會依照飛船的飛行資料做變動。當飛船轉角變化時會根據方向顯示動畫，並顯示轉動的度數。

	甲、　近景轉角 1.版本 2 新增，版本 4 因風格不搭刪除  2. 版本 4 新增  乙、　中景轉角(版本 4 新增) 
傾角	物件動畫會依照飛船的飛行資料做變動。當飛船傾角變化時會根據方向顯示動畫。 A.　版本 2 新增，版本 4 因風格不搭刪除  B.　版本 4 新增

	
推力及馬達出力倍率	物件動畫會依照飛船的飛行資料做變動。A 部分為馬達推力，會依照推力顯示數據，B 部分為馬達出力倍率，會依照馬達出力倍率顯示數據。 
晶片溫度	物件動畫會依照飛船的飛行資料做變動。會依照飛船資料有不同的溫度動畫，當溫度達到 50 度即為過熱時，會閃爍紅光表示警示。 
場景按鈕	A. 中文按鈕(版本 4 中新增，版本 6 因風格不搭刪除) 甲、 展示平台畫面預設為中景按下狀態。 乙、 一般移出時如圖中是白邊黑底字，滑鼠移入後會有發光白字效果，滑鼠按下過後會放大並在其他按鈕右方顯示。

<table>
<tr><td></td><td>


B.　英文按鈕(版本 6 新增)
1.　展示平台畫面預設為中景按下狀態。
2.　按鈕固定在對應場景上方，並跟隨場景移動。
3.　滑鼠移出時為普通白底英文，移入場景內，按鈕周圍有虛線動畫播放，按下場景時會跑出按下畫面。


</td></tr>
<tr><td>說明文字</td><td>丙、　彩虹按鈕(版本 3 新增)
無移入彩虹按鈕時，會以正中間按鈕為紅色以放射狀向兩邊依照彩虹顏色(紅橙黃綠藍靛紫)順序延伸，而滑鼠移入彩虹按鈕時，如圖中 b 項，依照移入按鈕(溫度)為紅色並以放射狀向兩邊依照彩虹顏色順序延伸。


丁、彩虹按鈕改版(版本 4 中配合場景切換改版,版本 6 中因風格不搭刪除)
近景：溫度、傾角、轉角、行走距離、推力、時速、高度</td></tr>
</table>

	中景：轉角、訊號比、控制系統、指北針、方向角 遠景：高度、飛行軌跡、位置  戊、　　說明框（版本 6 中新增，代替彩虹按鈕的功能） 如下圖，當滑鼠移入物件時後出現說明框，並在框內顯示說明內容以及資料數據。 
	己、

(3). 動畫整體架構，如圖 5-10。

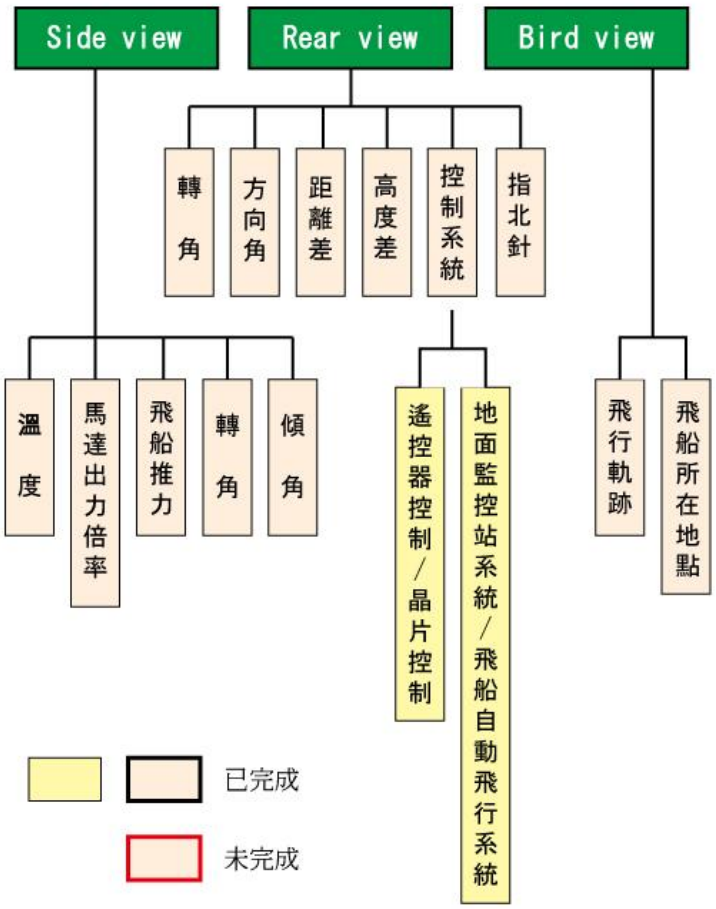


圖 5-10 動畫架構

(4). 物件導向程式架構，圖 5-11。

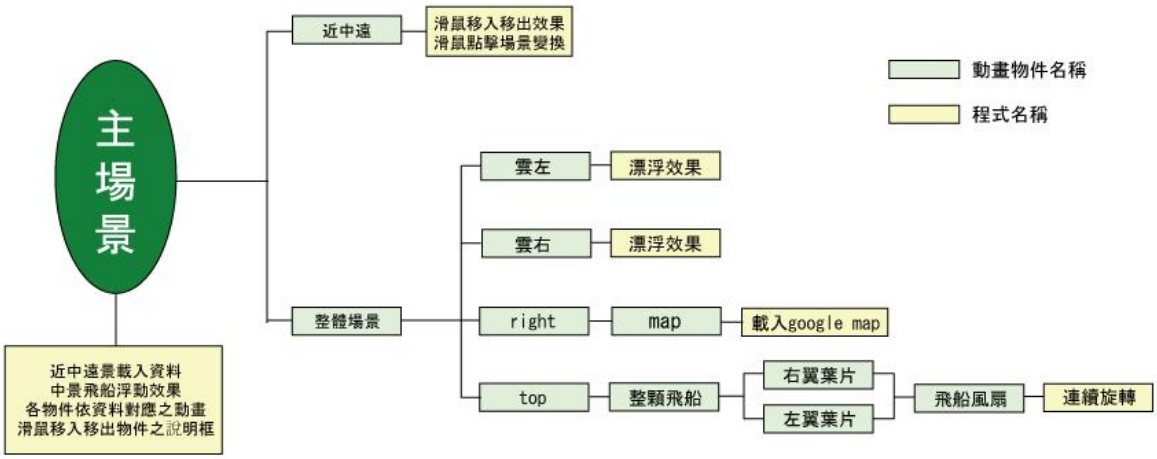


圖 5-11 物件導向程式架構

第六章 系統整合測試與驗證

6.1. 地面整合測試（PIC+Server＋動態展示）

6.1.1. 室內地面測試

主核心模組(詳述於第三章)與周邊模組(詳述於第四章)開發完成後，於 2010 年 12 月 10 日將控制模組搭載於飛行船船艙，並與地面監控站連結，測試整合後系統是否有需要調整的部份，此步驟為飛測前的準備項目。

測試方式第一部分以移動控制模組來模擬船身移動狀態，如圖 6-1，而馬達確實有做出相對應動作，如 Pitching 仰角改變，升降馬達做出升降轉的變動。因為自動模式中目標點設定於操場，而此次測試在專題教室，所以距離沒有在移動，以致補償動作一直讓馬達增加推力，要使飛行船全速前進到操場。

第二部份測試為遙控器控制與自動模式的切換，是為了確認硬體部分在切換過程沒有問題。再來就是由地面站切換地面控制與自動模式的測試，透過飛行人機介面看到飛行狀態變成地面控制，由此介面更改角度以及馬達%數，經過 GPRS 傳送指令，成功控制馬達做出相對應動作，再將推力改到 0%，則回到怠速狀態。地面站的 Flash 動態展示，也確實根據控制系統回傳給 Server 的資訊做出即時飛行動畫擬態。

上述室內地面測試項目及綜整摘要如表 6-1。

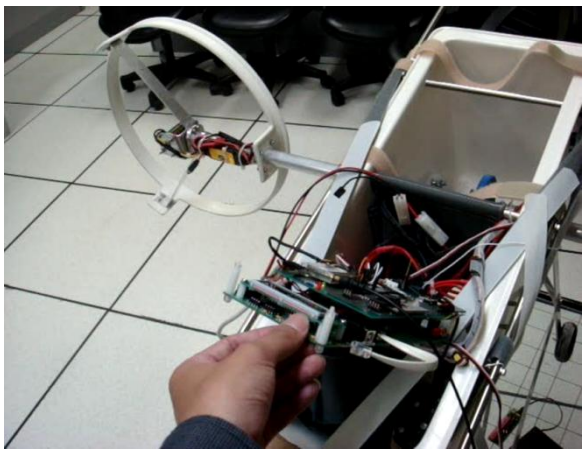


圖 6-1 主版與船艙

表 6-1 室內地面測試表

測試項目	說明	是否完成
升降伺服馬達自動控制	升降伺服馬達是否有依照載台仰俯角做出穩定船身的反饋控制動作	✓
手自動切換	硬體確認：遙控器控制切換模式	✓
載台與監控站連結	GPRS 連結測試	✓
地面控制與自動模式切換	從自動模式切換成地面下達指令更改飛行控制	✓
監控站更改馬達升降及推力	馬達推力是否依下達數據作相對改變	✓
Flash 動態展示	是否依據載台動態做出即時動畫展示	✓
測試馬達推力自動控制	馬達是否有依照目標點方向做出船身的補償動作	✓

6.1.2. 室外地面測試

室外地面測試分為兩大類，一是未搭載飛行船艙，僅測試自動飛行判斷的部分，另一類為搭載船艙後結合馬達與伺服器，依其反應動作預估其行進方向，推測是否能夠順利抵達目標點。

以下將就幾次重要的地面室外測試過程、結果，綜整摘要敘述如後:

測試場次

- 第 1 場次
  日期：2010 年 12 月 12 日
  時間：晚上 10 點 55 分 至 11 點 33 分
  程式版本：Main_PT_0921_1211_1
  記錄資料筆數：767 筆
  船體：未搭載飛行船船艙
  路徑圖：圖 6-2

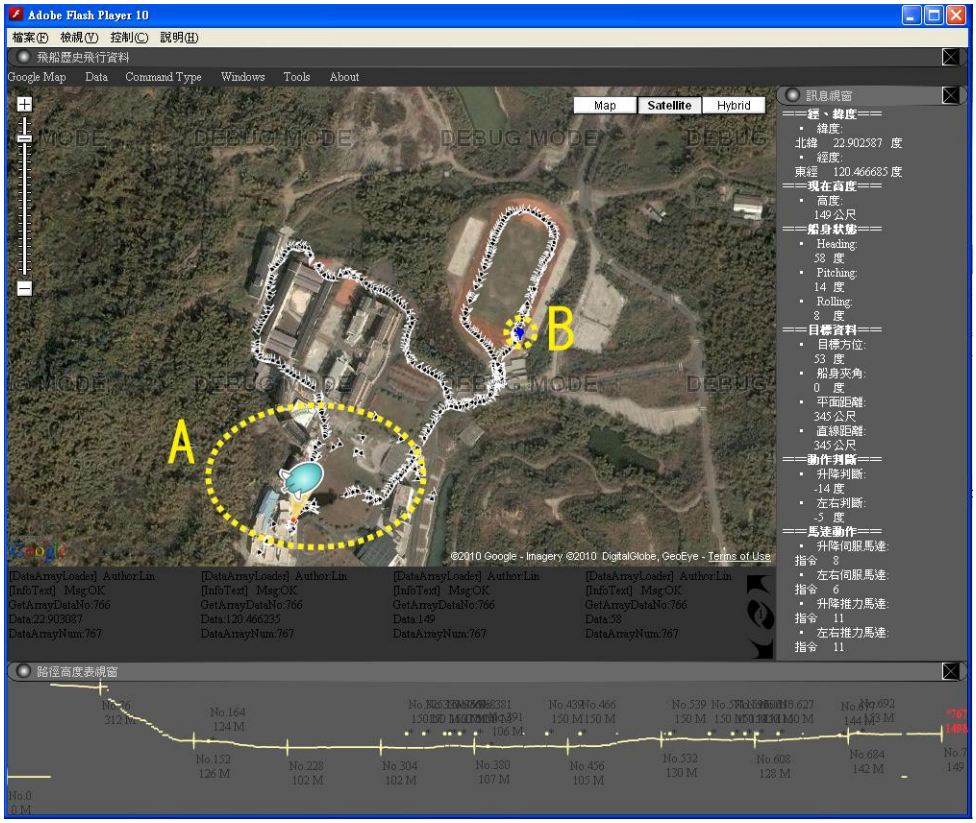


圖 6-2 地測第一場次路徑圖

測試項目	1. GPS 定位精準度 2. 資料傳送速度 3. 目標距離判斷準確度 4. 抵達目標判斷精準度
測試結果	1. 靠近 J 棟、大草坪與羅馬廣場時(圖中 A 範圍)，不時出現許多定位錯誤點 2. 資料傳送至地面站每筆間格約三秒 3. 目標距離判斷僅判斷第一目標點(圖中 B 點) 4. 抵達目標點判斷，無法到達目標點
分析與修改	1. 近 J 棟時可能受到雜訊干擾，導致定位點錯誤，擇日再測。 2. 綜合測試結果 3、4 點，判斷可能是目標點設定過高，導致一直無法到達第一目標點。 3. 修改目標點高度值，設定操場海拔高度約為 110 公尺。

- 第 2 場次

  日期：2010 年 12 月 15 日

  時間：晚上 8 點 48 分 至 9 點 17 分

  程式版本：Main_PT_0921_1211_2

  記錄資料筆數：597 筆

  船體：未搭載飛行船船艙

  路徑圖：圖 6-3

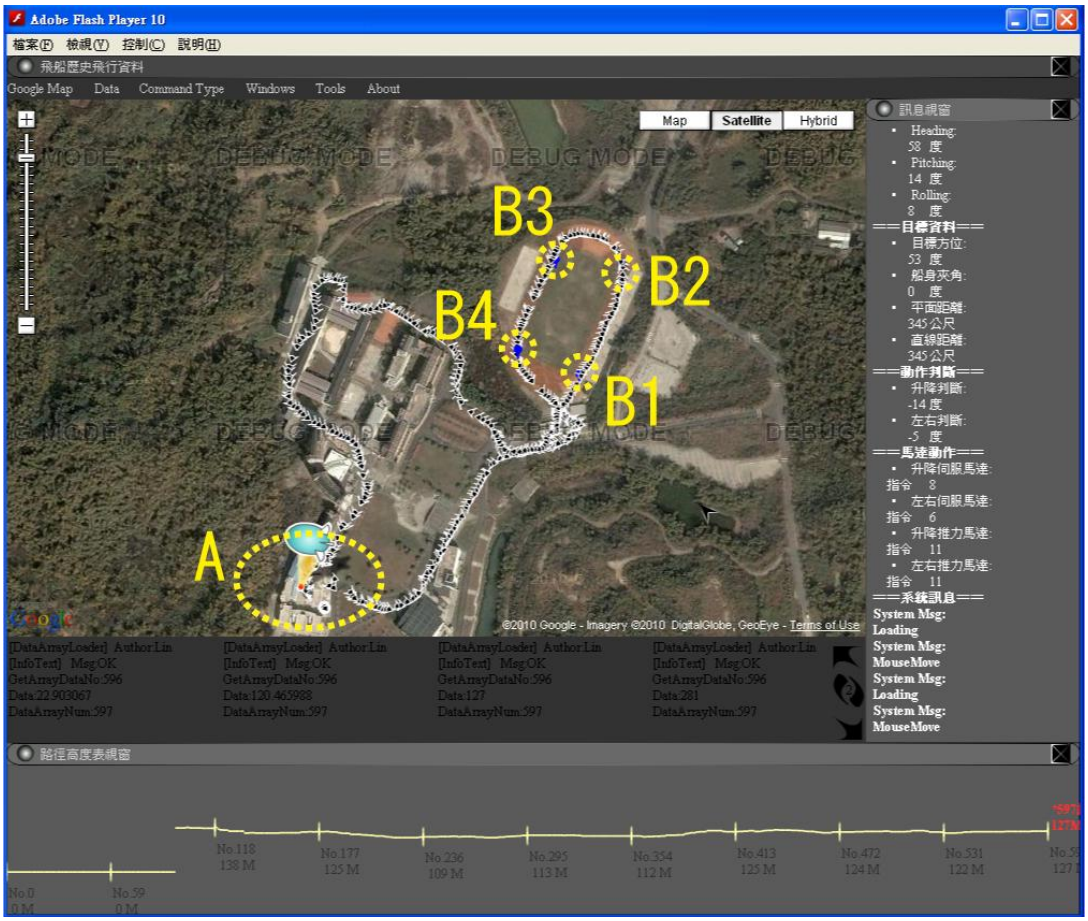


圖 6-3 地測第二場次路徑圖

測試項目	1. GPS 定位精準度 2. 資料傳送速度 3. 目標距離判斷準確度 4. 抵達目標判斷精準度
測試結果	1. 靠近 J 棟電梯處(圖中範圍 A)，出現許多定位錯誤點 2. 資料傳送至地面站每筆間格約三秒 3. 目標距離判斷，四個定位點皆可計算出距離 4. 抵達目標點判斷(圖中 B1~B4 點)，將目標點設於水平面上，可判斷抵達
分析與修改	1. 進 J 棟電梯時，可能受到雜訊干擾，導致定位點錯誤。 2. 目標距離判斷正確，事後由資料庫內記錄之座標以 GoogleEarth 軟體計算距離相差在 3 公尺內。 3. 於資料庫內記錄之目標點，在抵達後也有確實更新為下一目標點。

- 第 3 場次

日期：2010 年 12 月 15 日
時間：晚上 9 點 33 分 至 10 點 6 分
程式版本：Main_PT_0921_1211_2
記錄資料筆數：655 筆
船體：未搭載飛行船船艙
路徑圖：圖 6-4

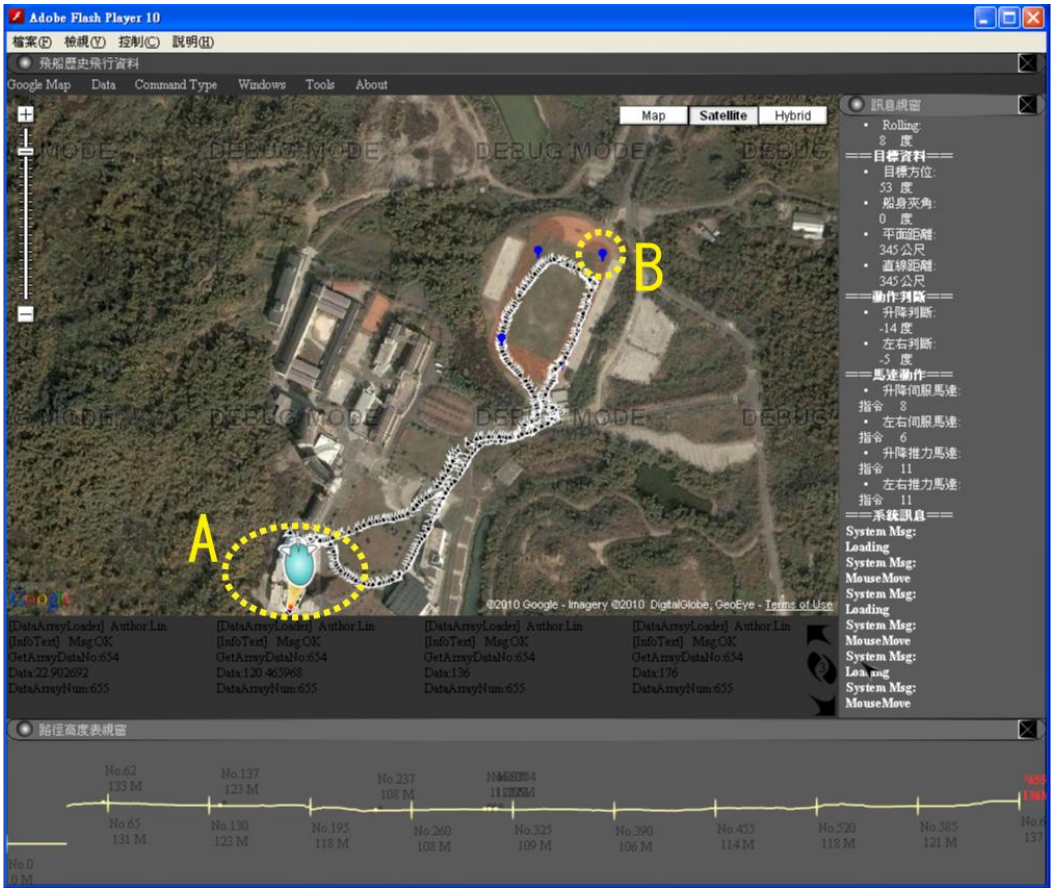


圖 6-4 地測第三場次路徑圖

測試項目	1. GPS 定位精準度 2. 資料傳送速度 3. 飛行應採取之動作方向判斷 4. 目標點範圍大小(平面距離 30M 且直線距離 40M)精準度
測試結果	1. 靠近 J 棟電梯處(圖中 A 點)，仍出現許多錯誤點 2. 資料傳送至地面站每筆間格約三秒，相當穩定 3. 飛行應採取之動作方向判斷正確，依其判斷方位直行，可達預定目標。 4. 目標點範圍大小精準度，可判斷抵達，約距目標 25 公尺內判斷抵達，如圖中 B 範圍)。
分析與修改	1. 由羅馬廣場直線前進 J 棟途中，未出現任何錯誤定位點。但進 J 棟電梯時，可能受到雜訊干擾，出現許多錯誤定位點，判斷為電梯電力干擾。 2. 資料傳送穩定，方向判斷正確，目標點範圍在 GPS 誤差範圍(10 公尺)內。 3. 去除電梯干擾因素，各項判斷皆正確，故擬進行船體裝載實地測試。

- 第 5 場次

  日期：2010 年 12 月 17 日

  時間：下午 3 點 23 分 至 3 點 34 分

  程式版本：Main_PT_0921_1211_3

  記錄資料筆數：226 筆

  船體：以推車搭載飛行船船艙(A 型，詳見第 2 章)

  路徑圖：圖 6-5

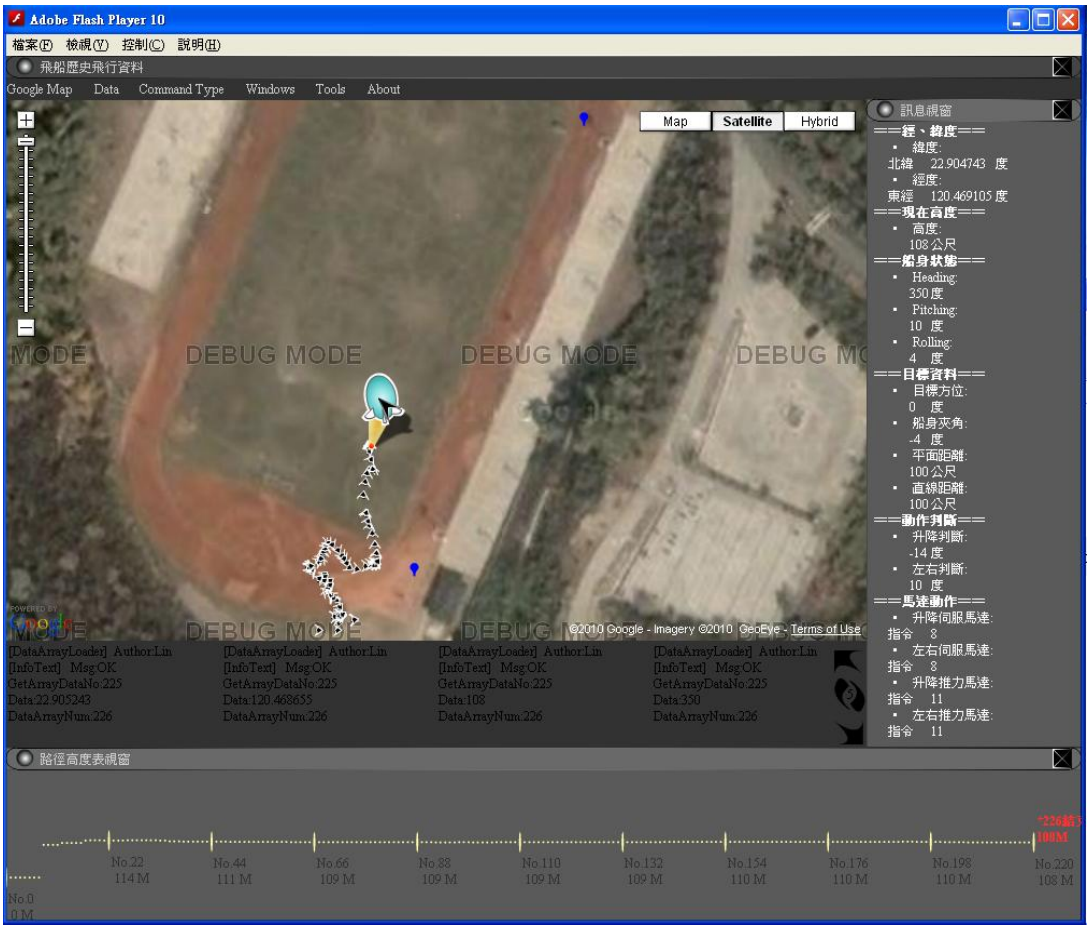


圖 6-5 地測第五場次路徑圖

測試項目	1. 後方轉向馬達實際動作 2. 前方升降馬達實際動作 3. 地面站連接與資料傳送速度 4. 由地面站以 GPRS 控制馬達動作之反應速度
測試結果	1. 馬達動作輸出不如預期。 2. 升降馬達轉動角度過大。 3. 資料傳送至地面站速度穩定每筆間格 3 秒。 4. 由地面站下達動作指定反應速度相當慢(數十秒至數分鐘)。
分析與修改	1. 因與第 3 場次測試期間曾修改方向判斷程式，可能修改有誤，故改燒錄第 3 場次地測之程式，以前一次地測結果確保程式無誤，再進行測試。 2. 升降馬達轉動角度過大，判斷為 PWM 數值有誤，應重新量測。 3. 資料傳至地面站速度穩定，但由地面指令送至控制板時相當慢，判斷可能原因為 GPRS 之 RX 優先權過低，故調高後再行測試。

- 第 6 場次

日期：2010 年 12 月 17 日
時間：下午 3 點 36 分 至 3 點 47 分
程式版本：Main_PT_0921_1211_3
記錄資料筆數：221 筆
船體：以推車搭載飛行船船艙(A 型，詳見第 2 章)
路徑圖：圖 6-6

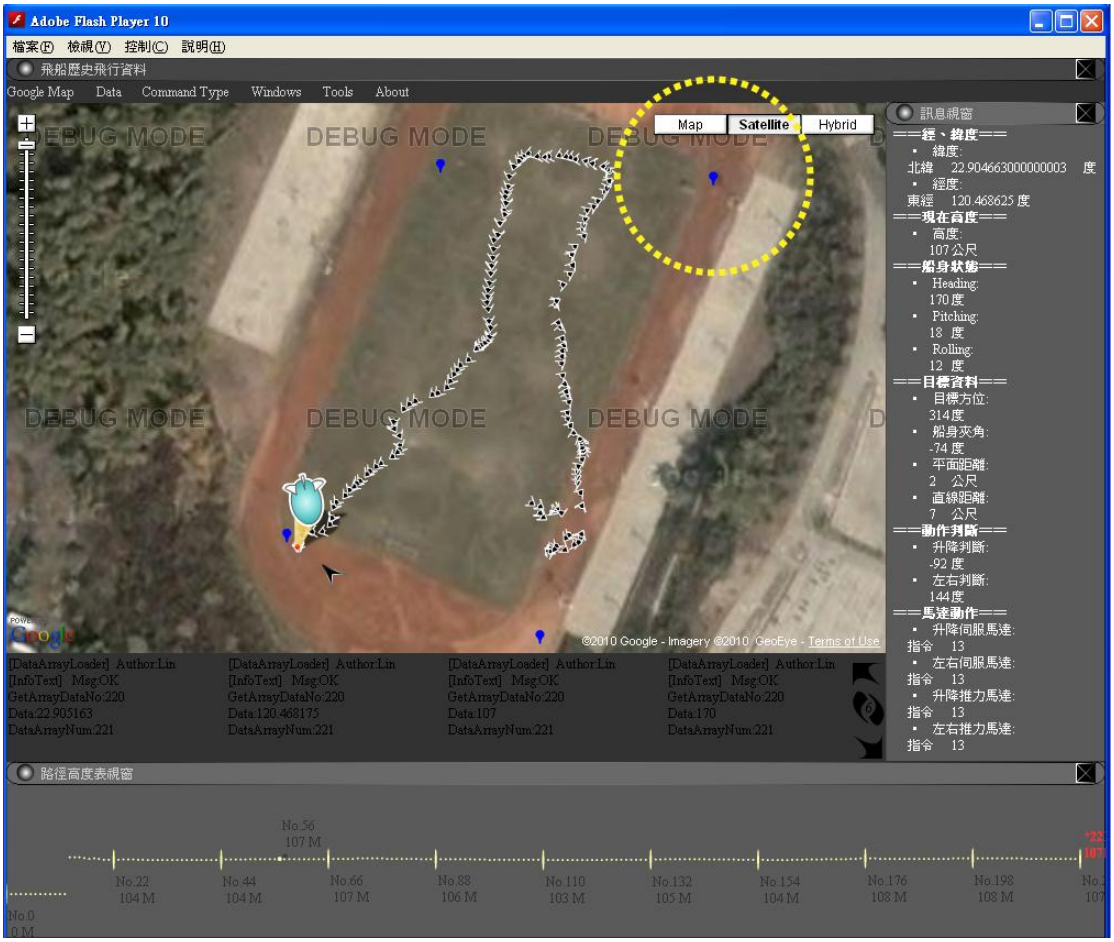


圖 6-6 地測第六場次路徑圖

測試項目	1. 後方轉向馬達實際動作 2. 前方升降馬達實際動作 3. 地面站連接與資料傳送速度 4. 抵達目標點之範圍大小
測試結果	1. 馬達動作輸出不如預期。 2. 升降馬達轉動角度過大。 3. 資料傳送至地面站速度穩定每筆間格 3 秒。 4. 抵達目標點之範圍大小，在規定範圍(30 公尺)內，如圖中黃圈。
分析與修改	1. 因與第 3 場次測試期間曾修改方向判斷程式，可能修改有誤，故改燒錄第 3 場次地測之程式，以前一次地測結果確保程式無誤，再進行測試。 2. 升降馬達轉動角度過大，判斷為 PWM 數值有誤，應重新量測。

- 第 7 場次

日期：2010 年 12 月 17 日

時間：下午 4 點 00 分 至 4 點 12 分
程式版本：Main_PT_0921_1211_2
記錄資料筆數：245 筆
船體：以推車搭載飛行船船艙(A 型，詳見第 2 章)
路徑圖：圖 6-7

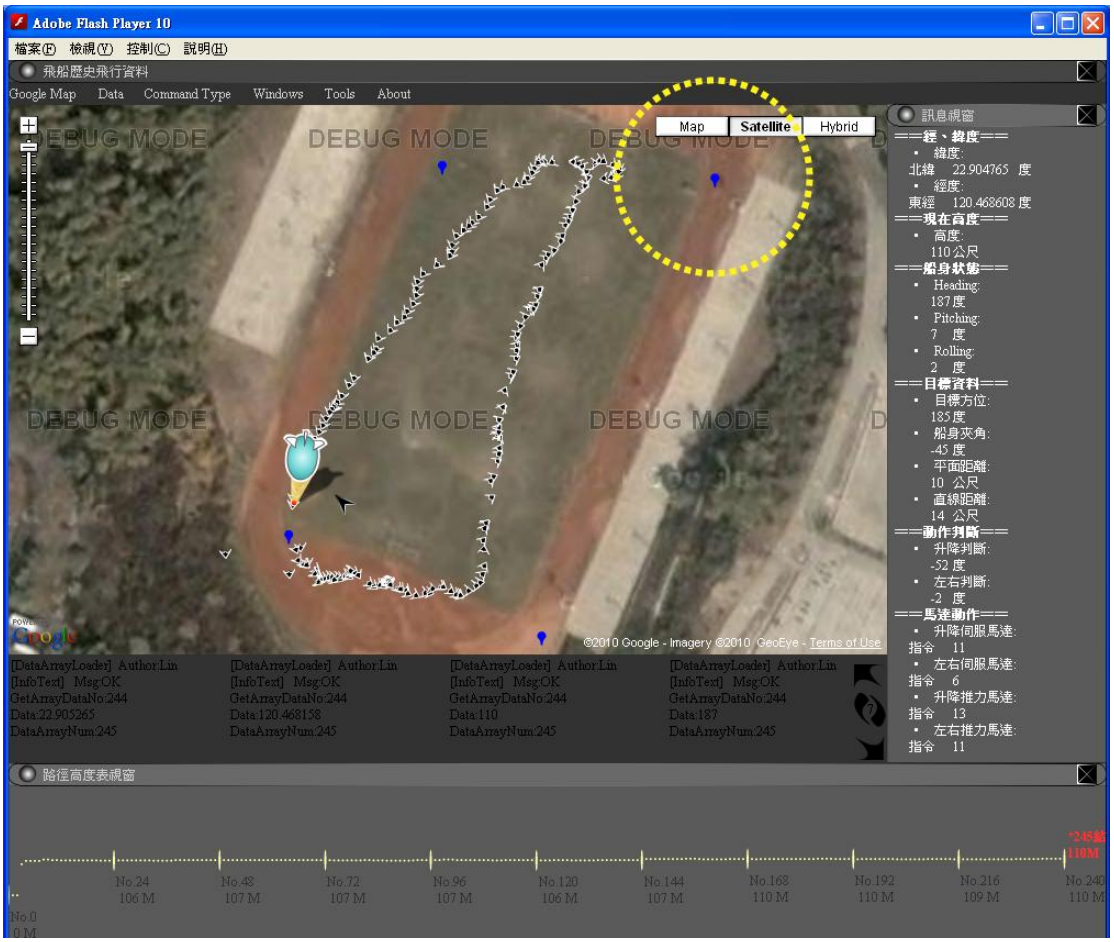


圖 6-7 地測第七場次路徑圖

測試項目	1. 後方轉向馬達實際動作 2. 前方升降馬達實際動作 3. 地面站連接與資料傳送速度 4. 抵達目標點之範圍大小
測試結果	1. 馬達動作輸出不如預期。 2. 升降馬達轉動角度過大。 3. 資料傳送至地面站速度穩定每筆間格 3 秒。 4. 抵達目標點之範圍大小，在規定範圍(30 公尺)內。
分析與修改	1. 升降馬達轉動角度過大，判斷為 PWM 數值有誤，應重新量測。

- 第 13 場次
  日期：2010 年 12 月 24 日
  時間：下午 5 點 11 分 至 5 點 32 分
  程式版本：Main_PT_0921_1222
  記錄資料筆數：907 筆
  船體：以推車搭載飛行船船艙(A 型，詳見第 2 章)
  路徑圖：圖 6-8

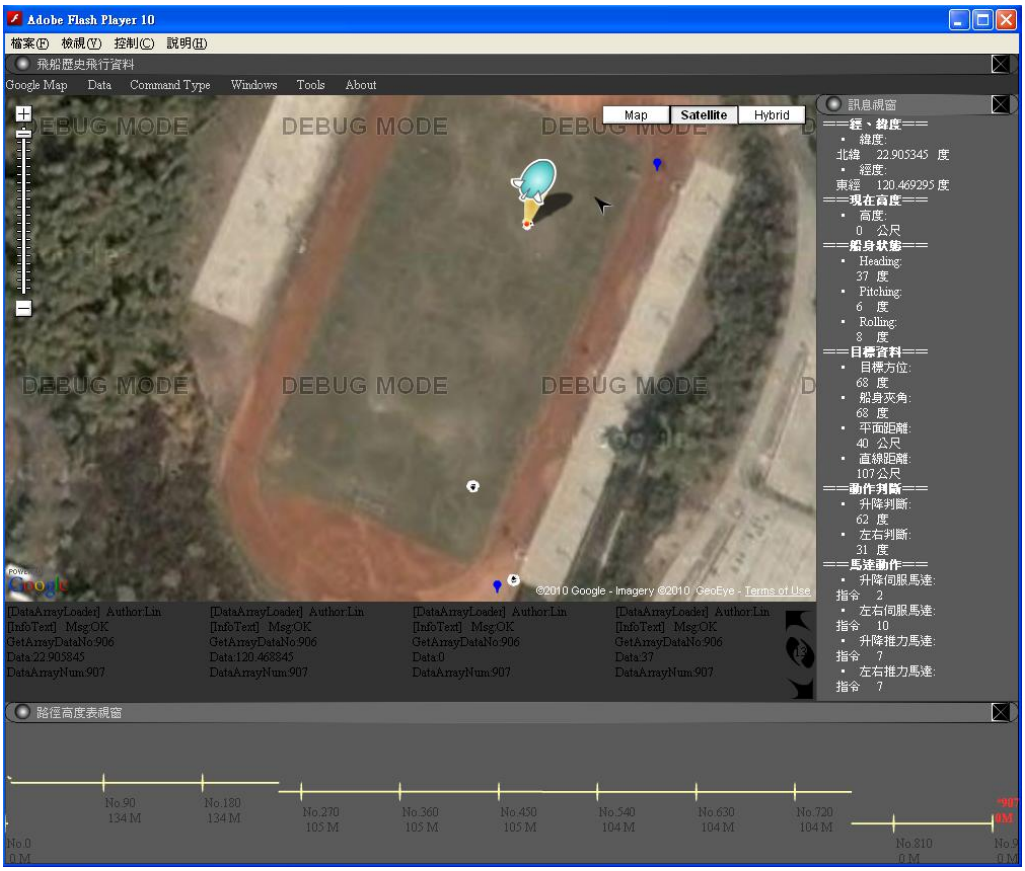


圖 6-8 地測第十三場次路徑圖

測試項目	1. 後方轉向馬達實際動作 2. 前方升降馬達實際動作 3. 搖控器切換手自動控制之反應速度 4. 地面以 GPRS 控制馬達動作之反應速度
測試結果	1. 依指北針之方向補償動作正常，但 GPS 發生問題。 2. GPS 定位僅數點，但時間正常更新，直至測試結束前數分鐘止。 3. 搖控器切換時，地面站顯示手/自動之狀態更新時快時慢(4~20 秒)。 4. 以地面站控制飛行船馬達動作之反應速度約 4~5 秒。
分析與修改	1. GPS 只定位到幾個點，可能是電力不足造成，GPRS 與 COMPASS 功能皆正常。 2. 應將電池充滿後再次進行測試。 3. 由於是以硬體切換手/自動飛行模式，雖地面站之資料更新速度較慢，但實際手自動已切換，故對實際飛行之安全性較無影響，唯對資料庫內資料造成幾筆錯誤記錄，與實際手/自動狀態有些許出入。

- 第 17 場次
  日期：2010 年 12 月 26 日
  時間：晚上 9 點 57 分 至 10 點 41 分
  程式版本：Main_PT_0921_1225
  記錄資料筆數：917 筆
  船體：以推車搭載飛行船船艙(A 型，詳見第 2 章)
  路徑圖：圖 6-9

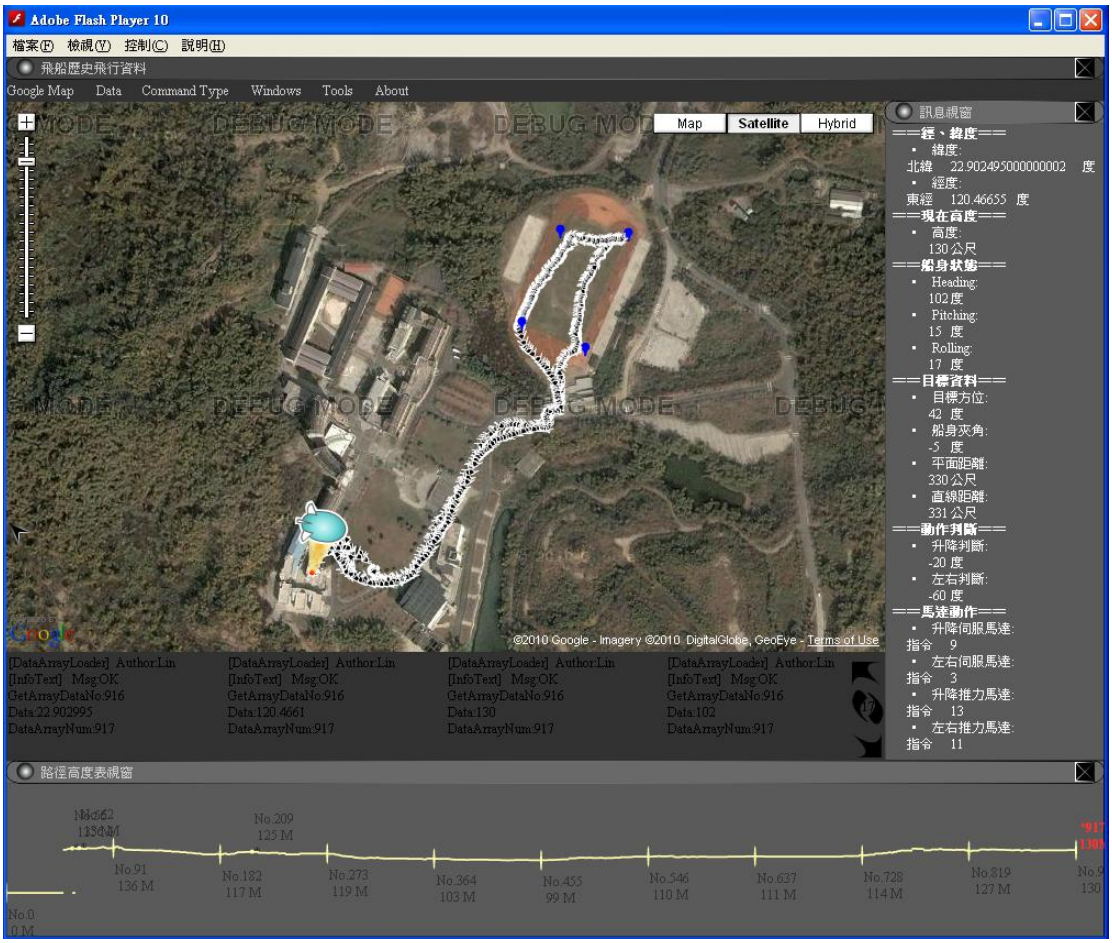


圖 6-9 地測第十七場次路徑圖

測試項目	1. 後方轉向控制馬達實際動作 2. 後方轉向推力輸出馬達實際動作 3. 資料傳送速度 4. 抵達目標點之範圍大小(平面 15 公尺，直線 20 公尺距離)
測試結果	1. 轉向控制馬達動作輸出角度在面北時，向左偏差約 10 度。 2. 後方轉向推力輸出馬達需先以搖控器啟動後，程式才能自動控制。 3. 資料傳送至地面站速度穩定每筆間格 3 秒。 4. 抵達目標點之範圍大小，在規定範圍(約 15 至 20 公尺)內。
分析與修改	1. 可能磁北與正北有些許偏差，或是 PWM 數值有誤，越接進目標點時所判定方向則無偏差影響。 2. 可能多訂目標點，行進軌跡會較依照預定點前進。

上述幾場室外地測測試項目及綜整摘要如表 6-2。

表 6-2 室外地面測試表

測試項目	說明	是否完成
1. GPS 定位精準度	誤差小於 15 公尺	✓
2. 地面站連接與資料傳送速度	約 3 秒	✓
3. 目標距離判斷準確度	實際距離差距小於 10 公尺	✓
4. 抵達目標判斷精準度	約 15~20 公尺	✓
5. 飛行應採取之動作方向判斷	左右轉及角度判斷	✓
6. 目標點範圍大小精準度	平面距離 30M 且直線距離 40M	✓
7. 後方轉向、推力馬達實際動作	轉動角度與推力	✓
8. 前方升降、推力馬達實際動作	轉動角度與推力	✓
9. 由地面站以 GPRS 控制馬達動作之反應速度	約 4~5 秒	✓
10. 搖控器切換手自動控制之反應速度	小於 4 秒	✓

6.2. 飛行整合測試

6.2.1. 飛行測試流程

因系統設計之初，雖與廠商確認飛行時之動力結構為 A 型，但仍不排除修改動力結構的情況況，故於系統設計時便以模組化、靈活可調整的方向進行設計。

飛行測試之目的為驗証本自動飛行系統之完整性，故針對自動飛行部分做以下流程規劃以利於進行充份的測試並驗証，直至完成自動飛行之目的，如圖 6-10。

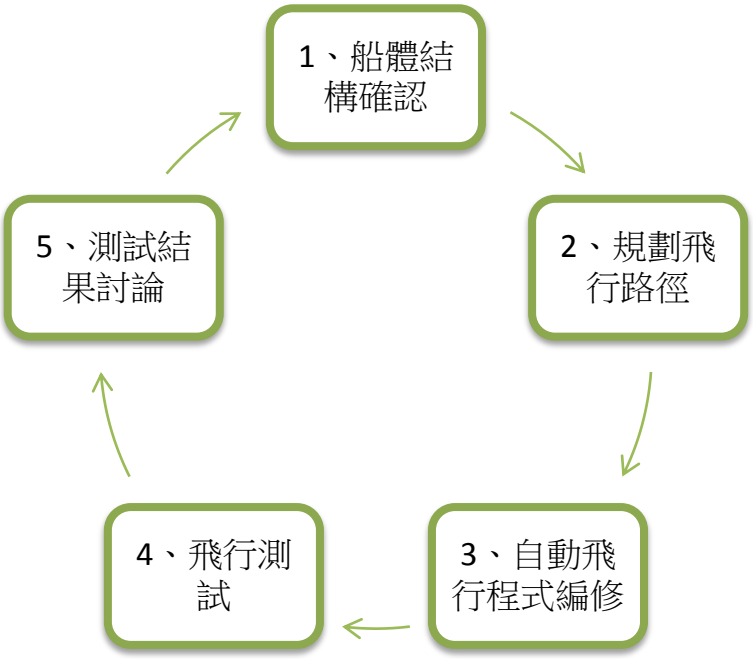


圖 6-10 飛行測試流程圖

測試流程

1. 船體結構確認：
   船體動力系統的輸出方式。確認各馬達的輸出動作對於飛行船飛行姿態之改變為何。
2. 規劃飛行路徑：
   確認飛行測試之地點，於測試前先預定飛行目標之經緯度資料。
3. 自動飛行程式編修：
   以達成上述兩點之目標，對自動飛行系統進行初步編修，動力輸出變更與目標點變更。
4. 進行飛行測試：
   以圖影記錄飛行測試結果，飛行系統也會將各種系統資料回傳於地面站，並儲存於資料庫中。
5. 測試結果討論：
   利用測試所記錄之圖影及資料庫中之系統資料，提出假設、討論與修改船體、程式部分皆需進行檢討。

6.2.2. 飛行測試準備工作

飛行前先地面整合測試(見 6.1.1 節)，確認設備運作是否正常並修正問題。飛船部份只於第一天測試填充氦氣，如圖 6-11，之後的幾場再依據浮力做氣體的調整，比較不浪費，並在飛船綁上襟翼與船艙，如圖 6-12。

圖 6-11 填充氦氣

圖 6-12 綁上襟翼

系統部分要確認設備及其他器材是否齊全，準備事項例如電源電力是否充足、監控站是否開啟等項目，皆需列表單整理，如附錄[4.]的行前確認單。圖 6-13 為遠端監控站畫面確認開啟。

圖 6-13 遠端監控站畫面

6.2.3. 飛行測試記錄與討論

由於飛行測試之記錄很多，但有許多場次記錄為同一天所測，且硬體軟體更新多為不同飛測日期才更新，同日之不同場次飛行船之軟硬體大多不更新，故以下列出各測試之重點場次，並討論各場次之問題點與解決辦法，以飛測日期為分隔而不以場次作區分。

1. 第一次飛行測試記錄

日期：2011/01/08

重點場次記錄：29

圖 6-14 飛行資料點圖

圖 6-14，使用程式將所記錄於資料庫之資料以 Google Map 點出，因點數相當多，故只可知其飛行範圍，而看不出飛行路徑的先後順序。而圖 6-15 為本次飛行測試之飛行高度圖表，依記錄之資料點順序及記錄之高度，排成一線並於高點標示高度。

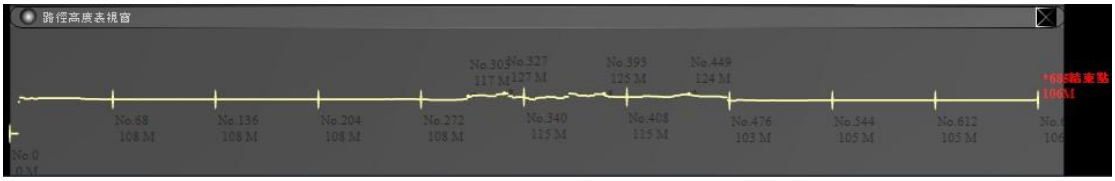


圖 6-15 高度圖 29 場次

由圖 6-15 可知本次飛試海平面高度約 105M，起飛時最高高度可至 127M，此場次幾乎皆為自動飛行，切換至手動時因控縱者與地面站溝通不良以至未能如期測試。

以下為資料庫所記錄之飛行資料，以自製的路徑軌跡圖程式繪制之飛行軌跡。

飛行軌跡圖：

(1). 本次飛測約於第 243 筆資料為起飛點，如圖 6-16。

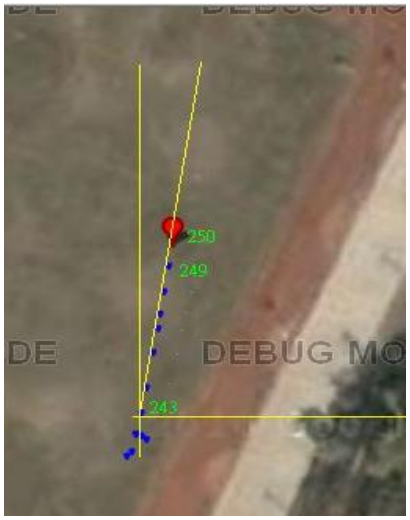


圖 6-16 起飛點

(2). 飛行資料第 258~278 點，如圖 6-17。

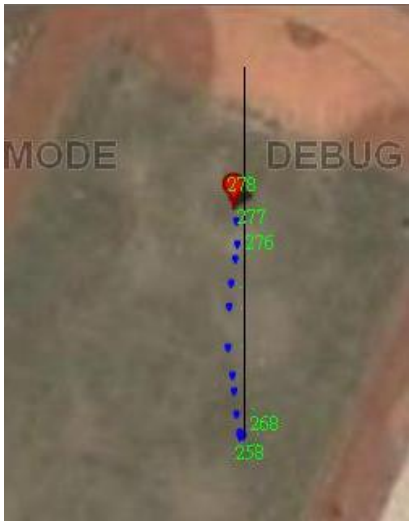


圖 6-17 飛行資料第 258~278 點

(3). 飛行資料第~298~312 點，如圖 6-18。

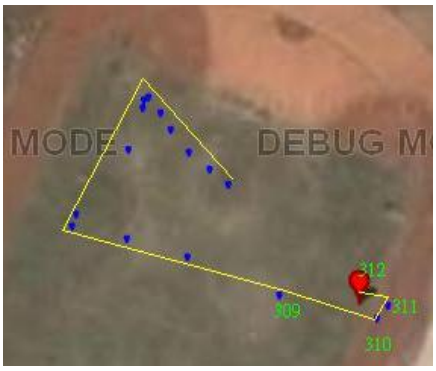


圖 6-18 飛行資料第~298~312 點

(4). 飛行資料第 310~330 點，如圖 6-19。

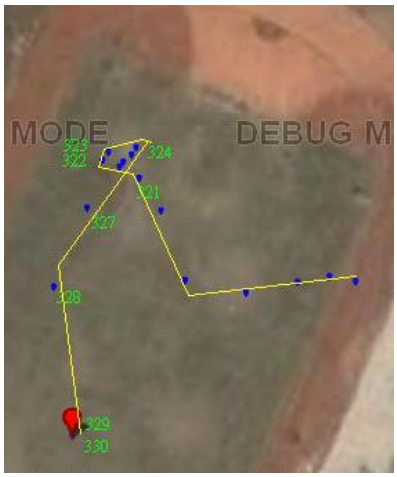


圖 6-19 飛行資料第 310~330 點

(5). 飛行資料第 324~344 點，如圖 6-20。

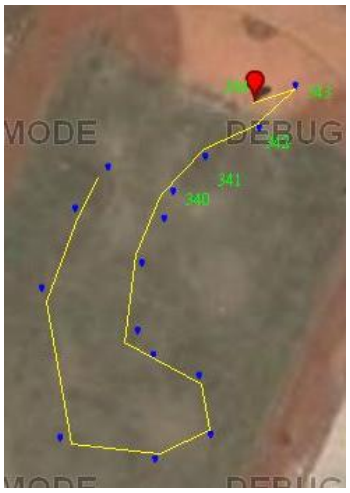


圖 6-20 飛行資料第 324~344 點

(6). 飛行資料第 345~363 點，如圖 6-21。

圖 6-21 飛行資料第 345~363 點

(7). 飛行資料第 364~383 點，如圖 6-22。

圖 6-22 飛行資料第 364~383 點

(8). 飛行資料第 381~400 點，如圖 6-23。

圖 6-23 飛行資料第 381~400 點

(9). 飛行資料第 404~423 點，如圖 6-24。

圖 6-24 飛行資料第 404~423 點

(10). 飛行資料第 423~443 點，如圖 6-25。

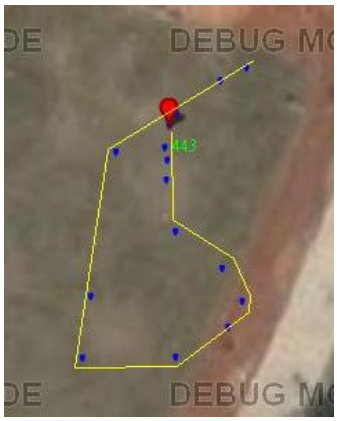


圖 6-25 飛行資料第 423~443 點

(11). 飛行資料第 442~462 點，如圖 6-26。

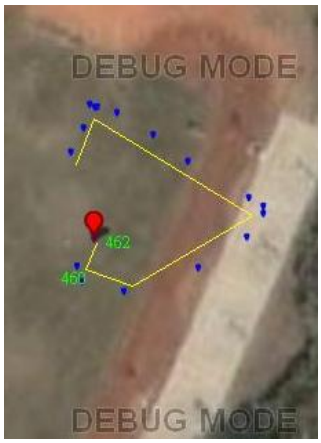


圖 6-26 飛行資料第 442~462 點

(12). 飛行資料第 462~482 點，如圖 6-27。

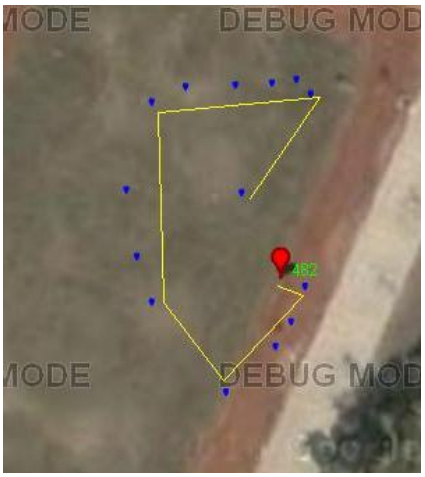


圖 6-27 飛行資料第 462~482 點

飛行範圍與路徑分析：

於搖控器操作之飛行範圍多在操場中間偏北，如圖 6-28，所以此次預定之 6 個目標點，操場跑道逆時針方向定六點，皆未抵達。本次飛測在緊急時均切換人工手動操作模式，但仍多次發生失控情形，推測為馬達推力不足及轉向伺服馬達轉動不理想。

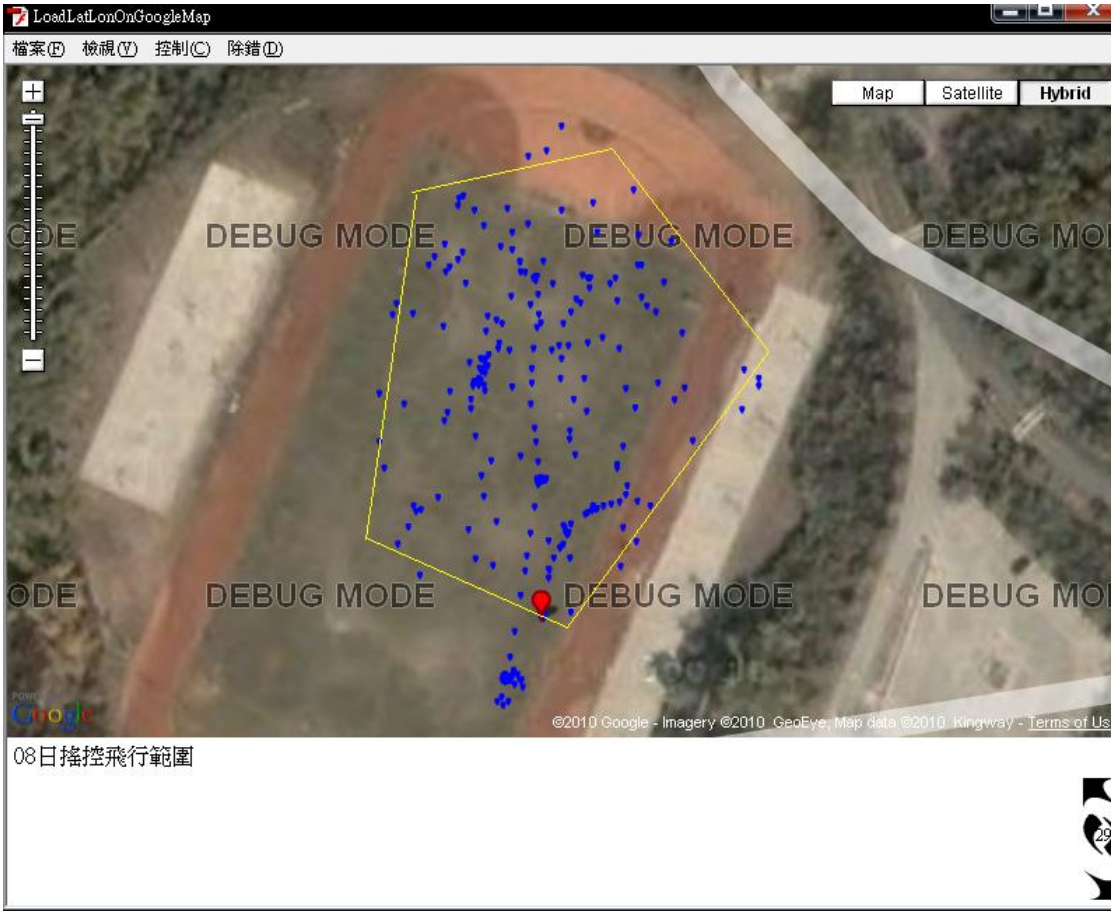


圖 6-28 飛行範圍

此外考慮到安全問題，因操場南方為階梯，外加風力吹向至此處時，會向上抬升可能遠成飛行船失控而被吹走的危險，故擬定下次飛行範圍應設於操場中間偏北，較為安全、適宜。圖 6-29 為重新規劃之下一次飛測之目標點與判斷範圍圖。

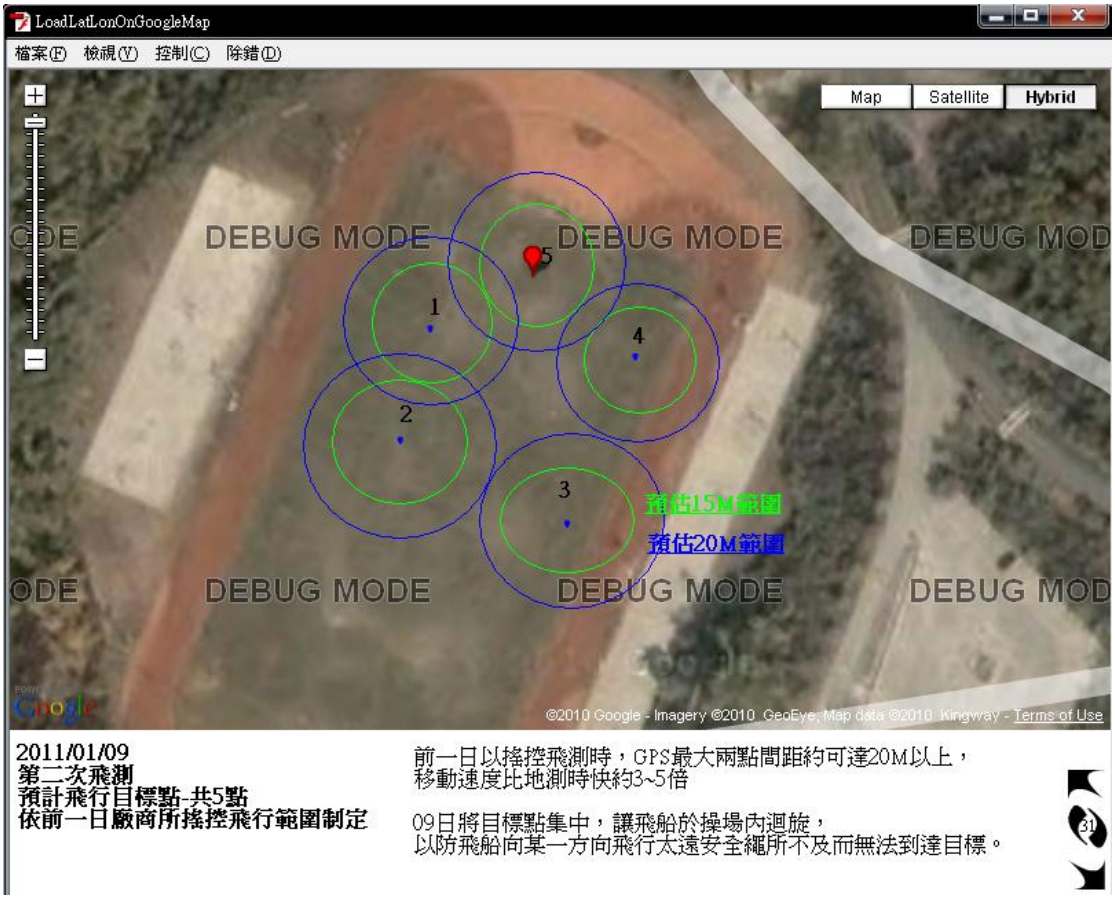


圖 6-29 預計飛行目標點

第一次飛測問題與討論：

(1). 馬達推力不足。

(2). 浮力不足、氦氣未充飽、船艙過重（多加兩組推進馬達但未使用到）。

(3). 行前準備器材不足：電池、無線電…等。

改進方向：

(1). 廠商將船艙帶回，將多餘的推進馬達拆下。

(2). 氦氣填充。

(3). 規畫行前準備表，廠商準備地測時供電之電池(如前面章節所提之型號)。

2. 第二次飛行測試記錄

日期：2011/01/09

重點場次記錄：42、43

圖 6-30 為第 42 場次之飛行範圍圖，目前這場次仍設定六個目標點，與第 29 場次相同，可以看到此次飛行因為受風力影響，飛行船被向上吹至校門口附近，飛行路徑也想目當凌亂，幾乎不受自動飛行程式控制，轉向的反應速度相當慢。

圖 6-30 飛行資料點圖。

圖 6-30 是由本次飛測時第 42 場次記錄於地面資料庫的飛行資料點。

圖 6-31 為本次飛行測試之該場次飛行高度圖表。

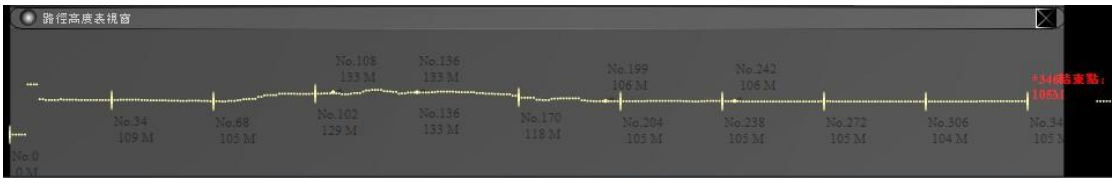


圖 6-31 高度圖 42 場次

因 42 場次與第 29 場次程式相同，只有浮力較充足與船艙較輕之差別故 42 場次之軌跡在此不列出。

圖 6-32 為第 42 場次之飛行範圍圖，此場次即燒錄本次飛測之五點預定飛行目標程式，但可見未抵達目標點，且飛行路徑依然相當凌亂，飛測時仍然不受自動飛行程式控制，但因為已將目標點訂於操場中間偏北的位置，所以在路徑有些影響，飛行範圍分佈較北，但仍未到達預計目標。

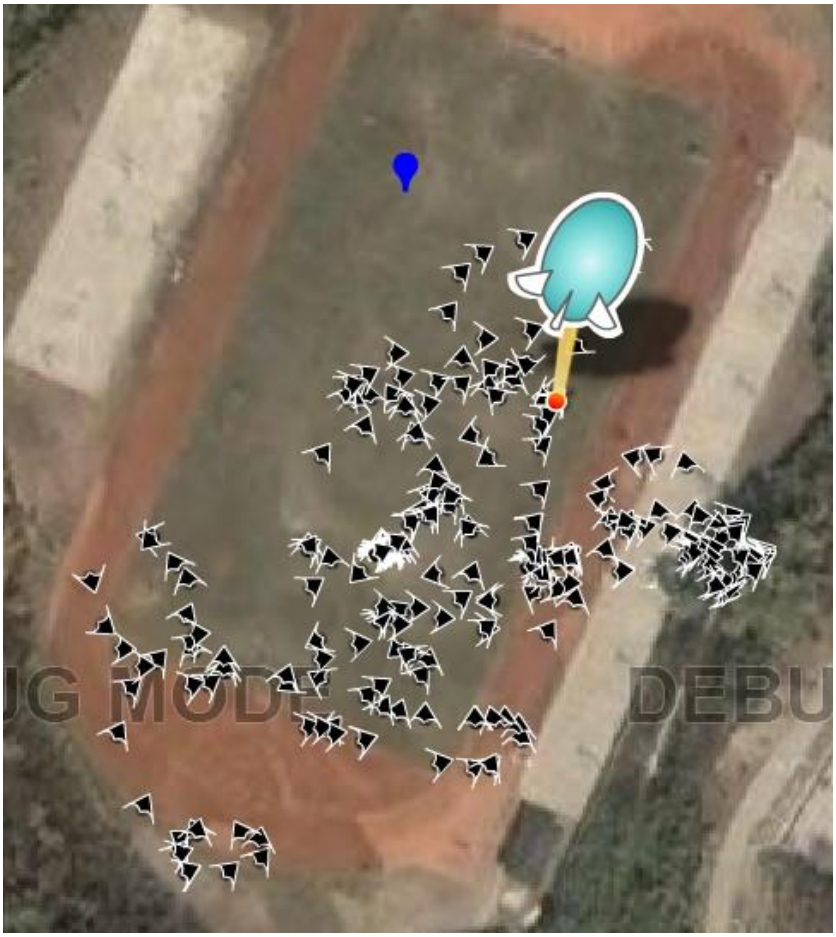


圖 6-32 飛行範圍 42 場次

圖 6-33 為第 43 場次的飛行高度圖表。

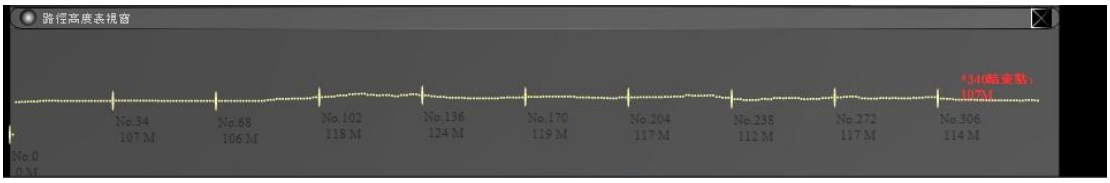


圖 6-33 飛行高度圖表 43 場次

下列為第 43 場次飛行路徑軌跡圖，皆為手動遙控器操控：

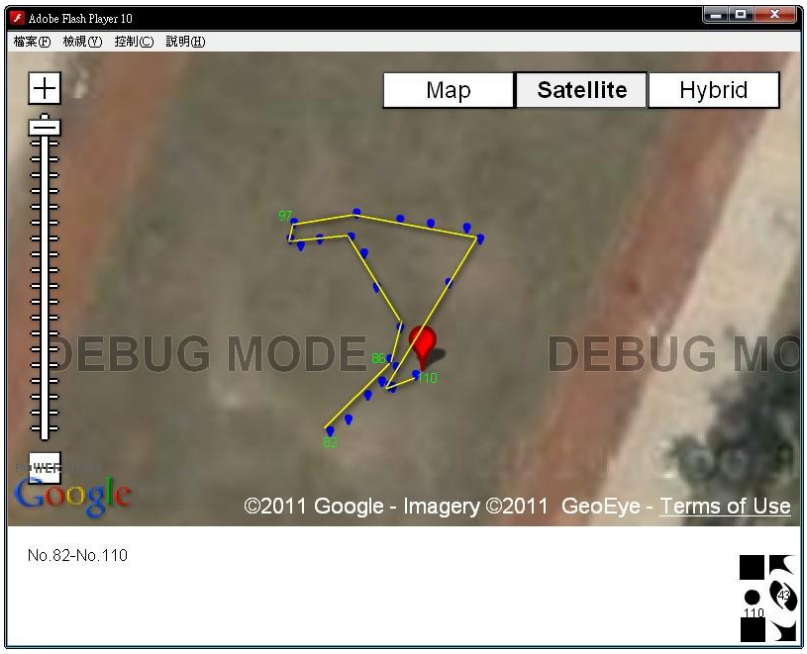


圖 6-34 飛行路徑軌跡圖第 82~110 點

圖 6-35 飛行路徑軌跡圖第 112~139 點

圖 6-36 飛行路徑軌跡圖 第 140~168 點

圖 6-37 飛行路徑軌跡圖第 169~197 點

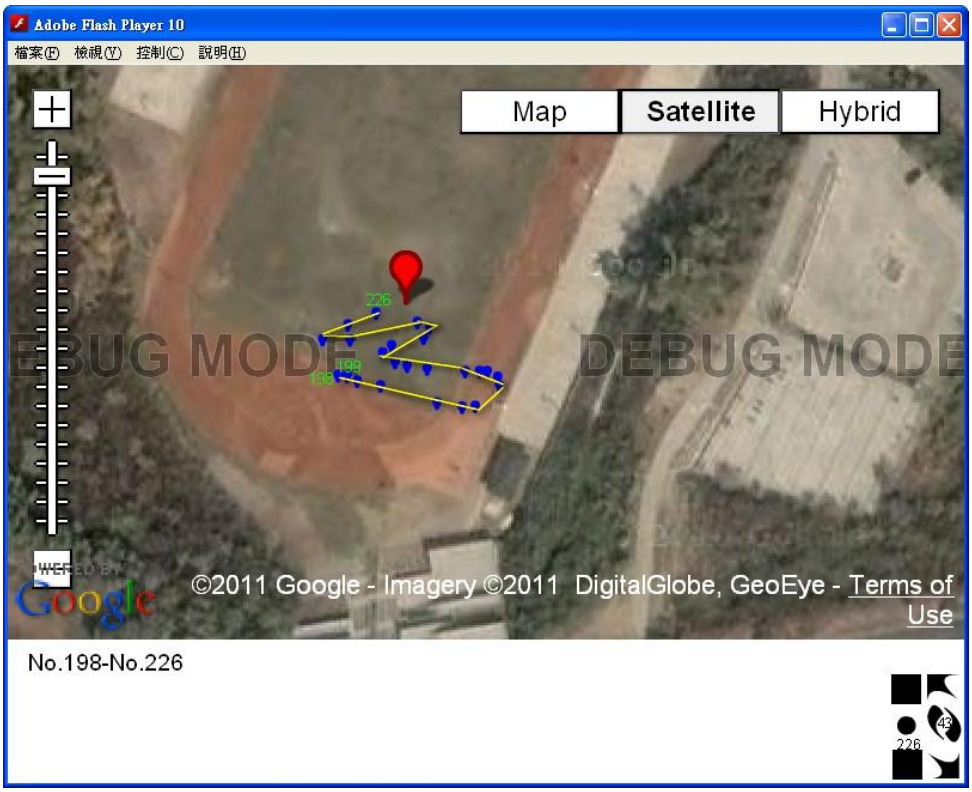


圖 6-38 飛行路徑軌跡圖第 198~226 點

圖 6-39 飛行路徑軌跡圖第 227~252 點

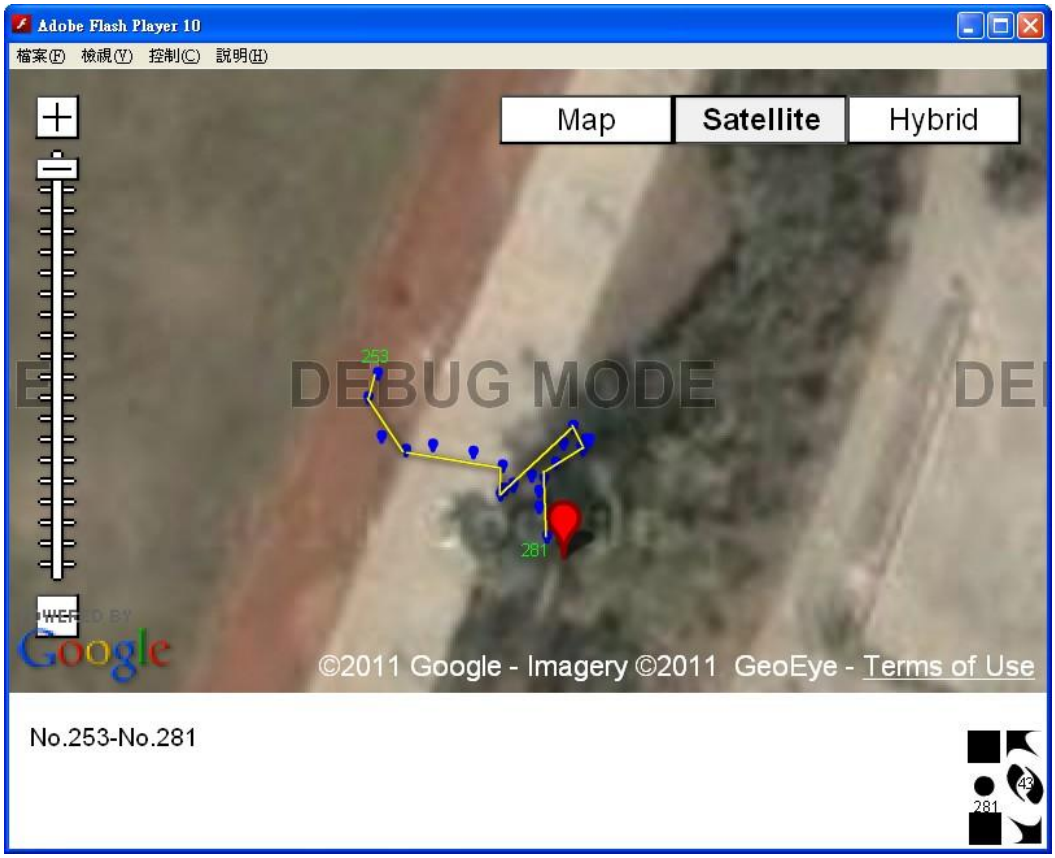


圖 6-40 飛行路徑軌跡圖第 253~281 點

圖 6-41 飛行路徑軌跡圖第 282~315 點

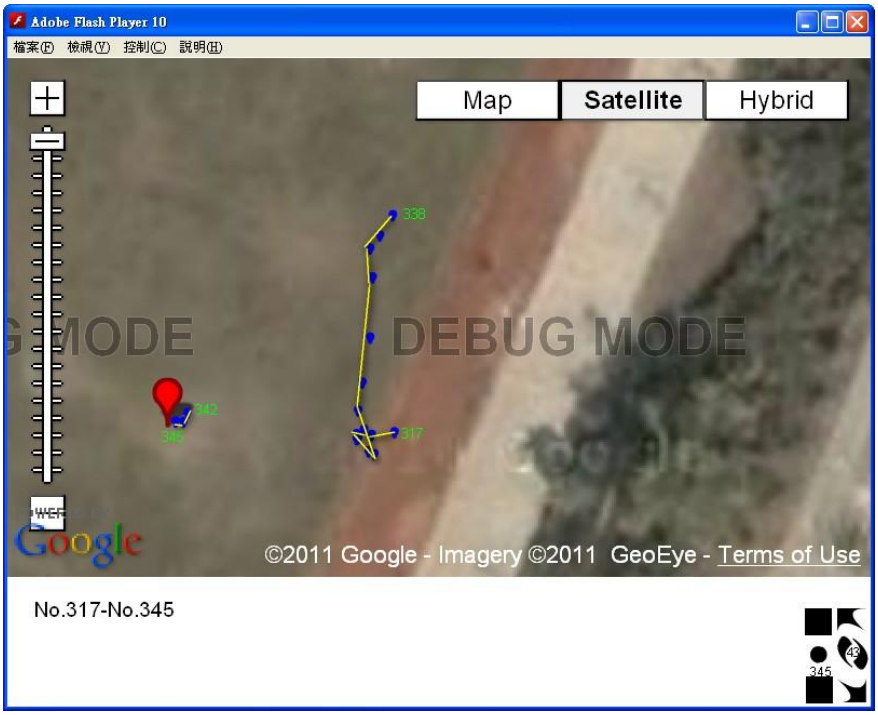


圖 6-42 飛行路徑軌跡第 317~345 點

由飛行軌跡之平滑度、流暢度可知飛測時飛船的穩定度並不佳，以及人為搖控與自動飛行模式切換時動作反應之差異，也可看出飛行船並未向著目標點前進，本次飛行測試之整體表現並不佳，可規納出以下幾個問題點。

(1). 馬達推力仍不足。
(2). 電力問題仍未解決，電池有問題，使用不當造成充電電池損毀。
(3). 左右轉時反應過慢，受風影響相當大，軌跡圖中有幾部分是由北向南一直線，是受風力影響一路從北吹向南，由搖控操作至在場北後，切換成自動時無法抵抗風力並且轉向速度過慢造成順風往南飄，未能即時作出補償動作。
(4). 由於操場東西向範圍不夠大，飛行範圍超過操場之東西向範圍，可能是目標點設定過多，應設定較少點並定在在場較中間部分。

解決方案：

(1). 預計下次飛行測試，廠商將改變尾部動力結構，使用 B 型船尾結構，正反轉的碳刷馬達來控制左右轉。
(2). 預計使用新的充電電池。
(3). 訂定新的目標點，從南端球門至操場正中間再到北端的球門，共三點來回航行。

3. 第三次飛行測試記錄

日期：2011/01/16

重點場次記錄：

有通訊程式：47、49

無通訊程式：53、54

本次飛行測試之 47、49 場依原訂計畫燒錄三點目標點之飛行程式，但 47 場次之抵達目標點判斷有誤，以致於抵達後仍未更新目標點為下一目標，於操場手動、自動四處移動測試仍不能到達目標點如圖 6-43。

圖 6-43 飛行範圍圖 47 場次

待修正後即 49 場次則發現動作仍然太慢，如圖 6-44。

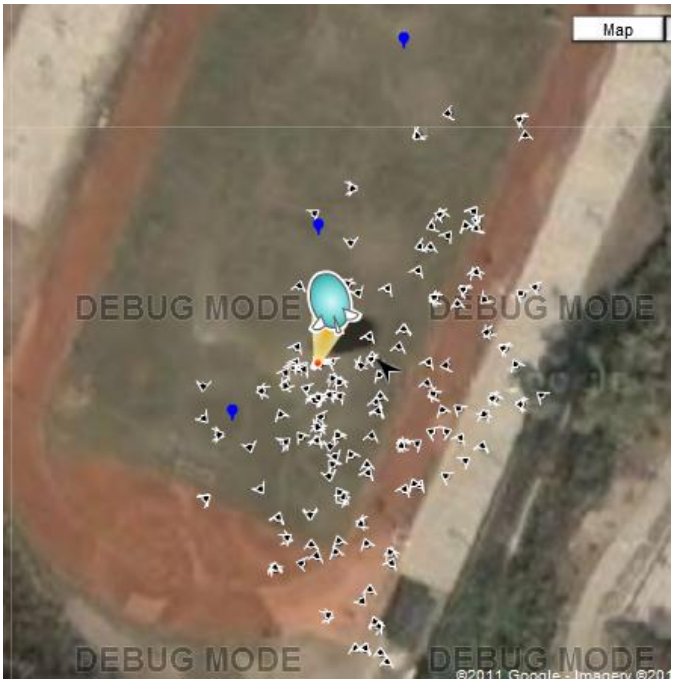


圖 6-44 飛行範圍圖 49 場次

最後將較花時間之通訊程式關閉後補償動作變為較靈敏，最後修改左右轉向的馬達控制，取消原先程式預計以漸變值控制馬達的設計，因飛船結構改變，船尾已非原 A 型船尾之結構，不需判斷馬達的出力與角度，所以將程式修改為需要左右轉時，控制船尾吹反正的伺服馬達直接轉至輸出動力角度。經過上述修改，於第 53、54 場次飛行時，與先前的飛行測試比就有相當大的進步，因逆風故無法筆直前進，但已能面對目標點迂迴航行。不過仍能發現以下問題點：

(1). 電力消耗很快，船尾的馬達不斷輸出。
(2). 飛行船仍舊推力不足，最大推力逆風時僅能抗衡，若電力消耗後則會越飛越遠。（詳影片記錄 53、54 場次）

解決方案：

(1). 飛行時多載一組電池並聯。
(2). 尾部增加舵翼輔助，即 C 型船尾。
(3). 推進馬達之葉片增加，增為四片。

4. 第四次飛行測試記錄

日期：2011/01/19

重點場次：在場 67、71、校內 74

場次：67、71

飛行範圍：

圖 6-45 為第 67 場，電力充足，可抵抗風力，自動飛行且皆在操場範圍內。

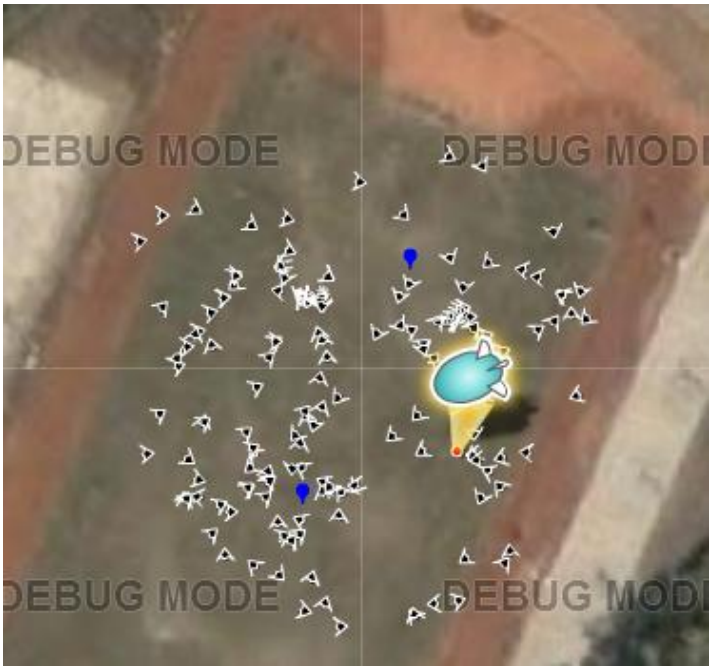


圖 6-45 飛行範圍圖 67 場次

圖 6-46 為第 71 場，風力較強時。

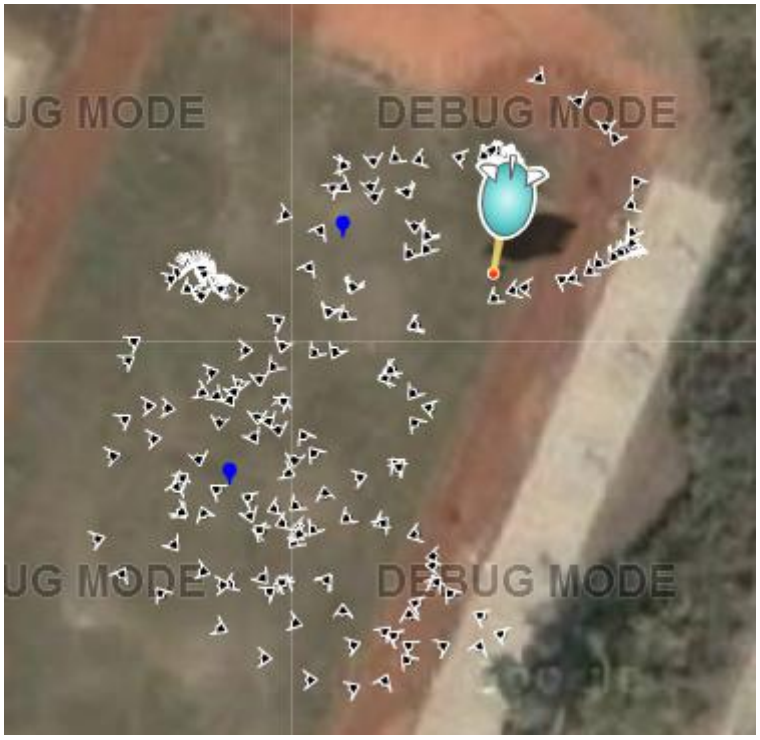


圖 6-46 飛行範圍圖 71 場次

第四次飛行測試，其中第 67 場次已經達成自動飛行之目標，而第 71 場次則發生向下風力太強，馬達輸出不足（GPS 抓到之高度與目標高度差不夠大，以致補償推進動力未設定至最大值），故飛行船被壓向地面的情形，即圖 6-46 右上方的弧形軌跡，此場次發生通訊斷線，但斷線後仍有影片記錄此次自動飛行，仍達成兩點往返航行動作，如影片資料。

下列為第 67 場（於操場完成自動飛行），飛行軌跡圖，黃線末端之紅色標記為各小段軌跡之結束點。

圖 6-47 飛行路徑軌跡第 27~57 點

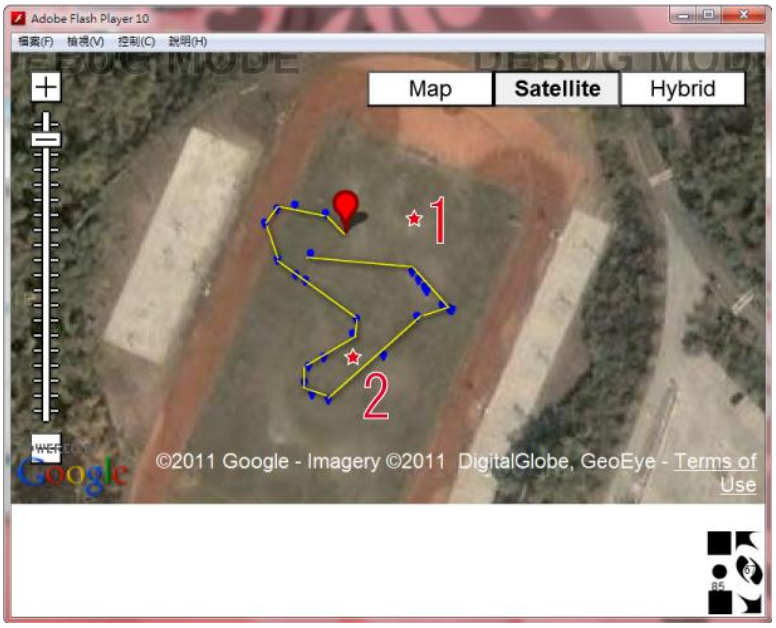


圖 6-48 飛行路徑軌跡第 56~86 點

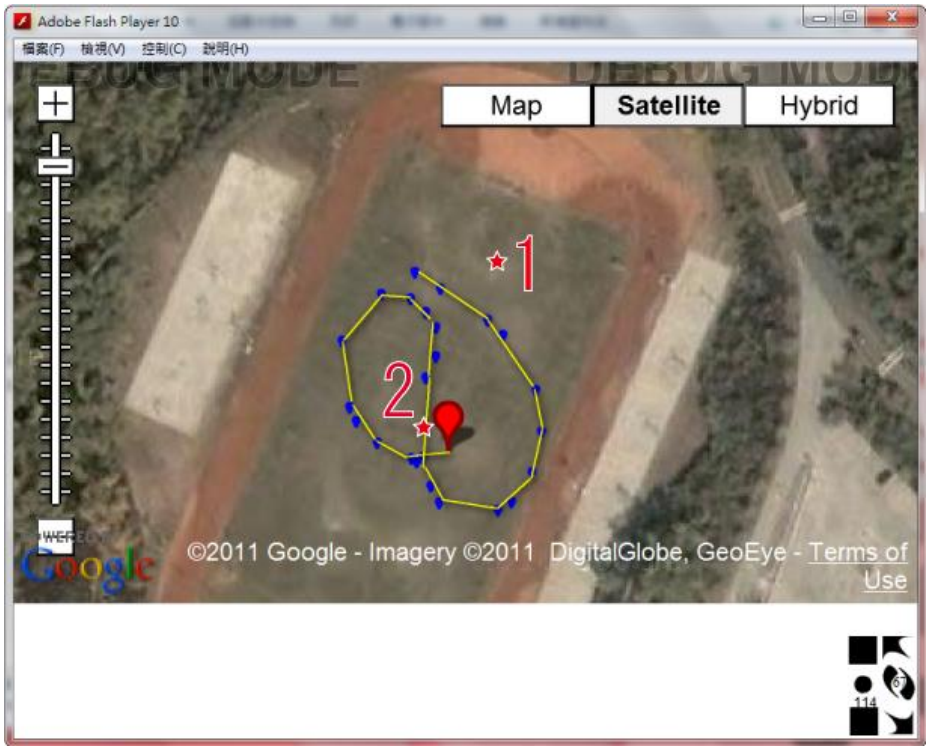


圖 6-49 飛行路徑軌跡第 84~114 點

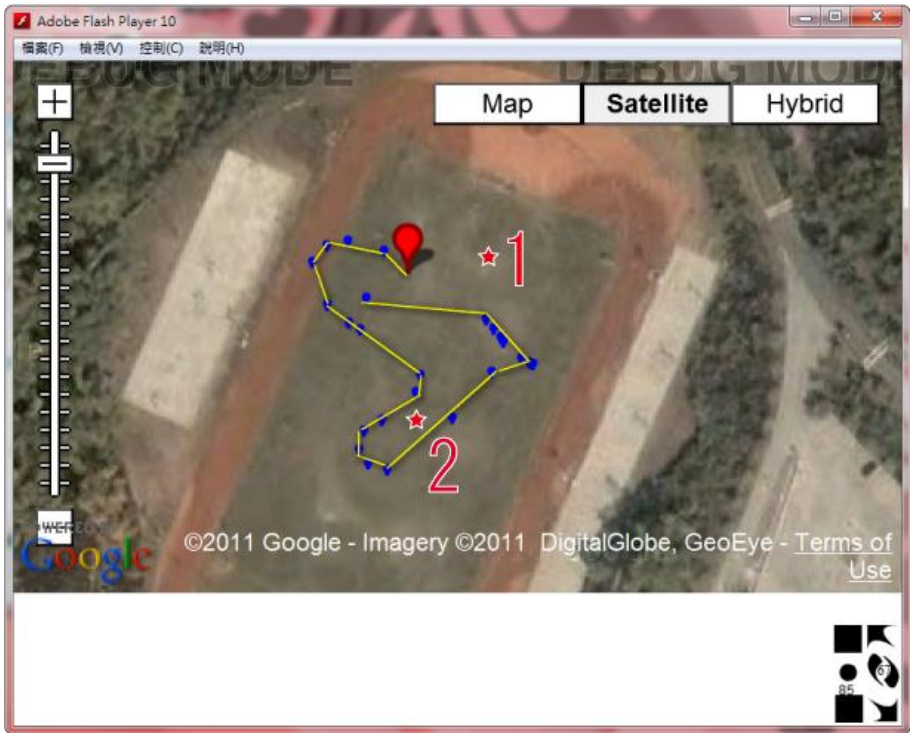


圖 6-50 飛行路徑軌跡第 113~143 點

圖 6-51 飛行路徑軌跡第 141~171 點

圖 6-52 飛行路徑軌跡第 180~210 點

由上列飛行軌跡圖可以很清楚的看到與前幾次飛行測試的不同，由於皆為自動飛行且飛行狀態相當穩定，故所繪軌跡相當平滑，以平滑的曲線於兩點間來回往返。

- 大草坪

經過上述之飛行測試，飛行範圍也都能控制在約 100M左右，故將飛行船移至大草坪於校內測試，記錄如下。

場次：74

以下為第 74 場次之飛行範圍圖：

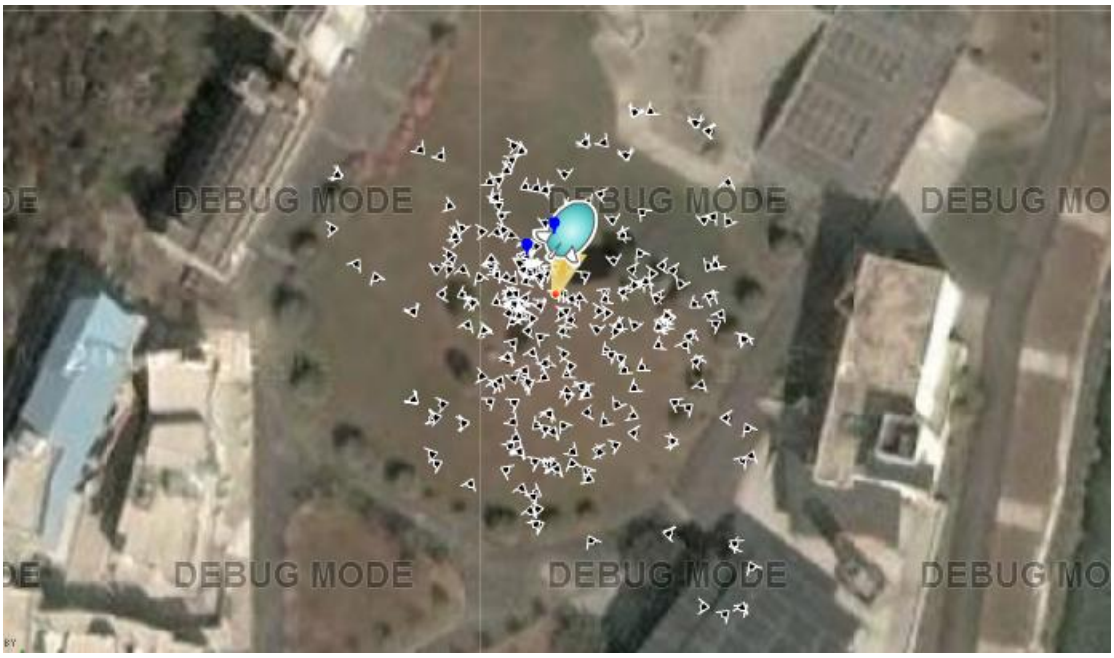


圖 6-53 飛行範圍 74 場次

此次飛行測試，尾端之正反吹碳刷馬達故障，無法正常運轉，故此次飛行轉向只依賴尾端舵翼，但仍能在一定範圍內進行自動飛行。可以推測當電力較不足時，自動飛行系統將較難控制飛行船於固定範圍內，輸出之動力較無法與風力抗衡。

詳細飛行軌跡圖可見附錄與飛測影片。

第 4 次飛測問題討論：

(1). 電力仍是最大問題點，若電力充足則可以保證飛行船能夠與風力抗衡，而依自動飛行程式所預定的方向前進，若電力不足則可能會被風力越帶越遠，或待陣風之風力減弱時而向目標點推進。

(2). 若能加入電壓感測，或許能在電壓不足時就自動降落，以免沒電時被風吹走。

(3). 未測試以地面站控制飛行船之動作，但依地測結果飛行船約３～４秒反應動作，以飛測之飛行速度及風力影響力，推測應不可行，僅能在緊急失控時要求飛行船做出標準向下降落動作，安全降落。

有關整體飛測的測試項目及綜整列表如表 6-3。

表 6-3 飛測綜整列表

項目	說明	是否完成
1. GPS 定位精準度	誤差小於 15 公尺	✓
2. 地面站連接與資料傳送速度	約 3 秒	✓
3. 目標距離判斷準確度	實際距離差距小於 10 公尺	✓
4. 抵達目標判斷精準度	約 15~20 公尺	✓
5. 飛行應採取之動作方向判斷	左右轉及角度判斷	✓
6. 目標點範圍大小精準度	平面距離 30M 且直線距離 40M	✓
7. 後方轉向、推力馬達實際動作	轉動角度與推力	✓
8. 前方升降、推力馬達實際動作	轉動角度與推力	✓
9. 由地面站以 GPRS 控制馬達動作之反應速度	約 4~5 秒	✓
10. 搖控器切換手自動控制之反應速度	小於 4 秒	✓
11. 兩點往返(定點)飛行測試	67、71 場次路徑	✓
12. 單點(定點)飛行測試	校內飛行 74 場次	✓
13. 全自動飛行	未包含降落部份	✓

第七章 結論與未來展望

本研究從各模組系統建立、測試，架設地面 Server 系統、動態展示系統，至各模組整合為一個完整的自動控制系統，期間經歷了約一年時間，逐漸由個體變為團體，共同擊破了許多難關，不斷的實作模擬、改良，完成了自動飛行導航、突破電路障礙焊出與程式搭配完美的子板達到手動自動飛行模式切換、地面監控資料接收、地面資料人機介面及動畫即時顯示，克服馬達、電池硬體問題、氣候完成了試飛。

本研究開發初期即將進入尾聲，討論有些待改進的部分。

- 電力仍是最大問題點，若電力充足則可以保證飛行船能夠與風力抗衡，而依自動飛行程式所預定的方向前進，若電力不足則可能會被風力越帶越遠，或待陣風之風力減弱時而向目標點推進。
- 無線通訊 GPRS：GPRS 的不穩定，在硬體限制下兩秒至三秒傳輸一次資料，且後期 GPRS 模組老舊常發生焊點脫落或是燒毀，故希望可換新模組，嘗試看看傳輸速度是否會因此加快，使資料更即時更新。本研究有第三控制模式：地面控制模式，由於 GPRS 的不穩與傳輸速度不夠快，故此模式並無於試飛時測試。此模式主要通訊程式部分則已撰寫至可由地面 Server 指定行列改變資料，此程式還可增加改變經緯度、高度等。地面

控制是一安全機制，如果飛船航行偏離遙控器控制範圍，就可由地面控制接手，避免飛船失控。故期許通訊程式能開發完整、成熟。

- GPRS 自動斷線後，沒有自動連線設定。
- 尚未有母板當機自動啟動裝置。
- 自動降落測試目前未實作。
- 飛船展示系統的呈現方式，希望未來可以以全 3D 方式展示飛行船動態，利用滑鼠拖曳可以從各種視角觀看飛船的飛行動態，讓畫面更有臨場感。

文獻探討

[1.]許家瑋“無人飛行載具的製作與導航”，碩士論文，雲林科技大學電機工程系，2004。

[2.]楊元坤，“無人飛行船自主性控制設計與實現”，碩士論文，成功大學航空太空工程學系，2002。

[3.]伍紹之，“無人飛行船系統之研發”，國立成功大學航空太空工程研究所碩士論文，2001。

[4.]趙弘文，“無人旋翼機自動駕駛系統之研發”，國立成功大學航空太空工程研究所碩士論文，2009。

附錄

GPS 訊號格式

GPGGA 範例如下：
$GPGGA，161229.487，3723.2475，N，12158.3416，W，1，07，1.0，9.0，M， ， ， ，0000*18。

GPGGA 範例說明

名稱	實例	單位	敘述
訊息代號	$GPGGA		GGA 規範抬頭
標準定位時間 (UTC)	161229.487		時時分分秒秒.秒秒秒
緯度	3723.2475		度度分分.分分分分
北半球或南半球指示器	N		北半球（N）或南半球（S）
經度	12158.3416		度度度分分.分分分分
東半球或西半球指示器	W		東（E）半球或西（W）半球
定位代號指示器	1		0 = fix not available， 1 = Non-differential GPS fix available， 2 = Differential GPS (WAAS) fix available，6 = Estimated
使用中的衛星數目	07		00 至 12
水平精度	1.0		0.5 至 99.9 米
海拔高度	9.0	米	-9999.9 至 99999.9 米
單位	M	米	
地表平均高度		米	-999.9 至 9999.9 米
單位	M	米	
差分修正 DGPS			(RTCM SC-104)資料年限，上次有效的 RTCM 傳輸至今的秒數（若非 DGPS，則數字為 0）

偏差修正（DGPS）			參考基地台代號，0000 至 1023。（0 表非 DGPS）
插分參考基站程式	0000		
總和檢查碼	*18		

GPRMC 範例如下：

$GPRMC，161229.487，A，3723.2475，N，12158.3416，W，0.13，309.62，120598， 003.1，W *10。

GPRMC 範例說明

名稱	實例	單位	敘述
訊息代號	$GPRMC		RMC 規範抬頭
標準定位時間(UTC)	161229.487		時時分分秒秒.秒秒秒
定位狀態	A		A = 資料可用，V = 資料不可用
緯度	3723.2475		度度分分.分分分分
北半球或南半球指示器	N		北半球（N）或南半球（S）
經度	12158.3416		度度度分分.分分分分
東半球或西半球指示器	W		東（E）半球或西（W）半球
對地速度	0.13	節	0.0 至 1851.8 節
對地方向	309.62	度	實際值
日期	120598		日日月月年年
磁極變量	003.1	度	000.0 至 180.0 度
磁方位角			東（E）半球或西（W）半球
總和檢查碼	*10		

GPGSA 範例如下：

$GPGSA，A，3，07，02，26，27，09，04，15， ， ， ， ， ，1.8，1.0，1.5*33 。

GPGSA 範例說明

名稱	實例	單位	敘述
訊息代號	$GPGSA		GSA 規範抬頭
定位模式 1	A		M：手動模式；A：自動模式
定位模式 2	3		1：位置不可用；2：二度空間定位；3：三度空間定位
PRN 數字	07		01 至 32 表天空使用中的衛星編號，最多可接收 12 顆衛星資訊
位置精度值 PDOP	1.8		0.5 至 99.9
水平精度值 HDOP	1.0		0.5 to 99.9
垂直精度值 VDOP	1.5		0.5 to 99.9
總和檢查碼	*33		

Google Maps API 建置方式

1. Google Maps API Java Script 製作步驟

(1). 使用網址申請 Google 地圖 API 金鑰

申請網址 http：//code.google.com/intl/zh-TW/apis/maps/signup.html

(2). 申請完畢後，系統會給一個 Google 地圖的 key 和範例

```
<!DOCTYPE html "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http：//www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http：//www.w3.org/1999/xhtml">
  <head>
    <meta http-equiv="content-type" content="text/html; charset=utf-8"/>
    <title>Google Maps JavaScript API Example</title>
    <script src="http：
//maps.google.com/maps?file=api&amp;v=2&amp;key=abcdefg&sensor=true_or
_false"
            type="text/Javascript"></script>
    <script type="text/Javascript">

    function initialize() {
      if (GBrowserIsCompatible()) {
        var map = new GMap2(document.getElementById("map_canvas"));
        map.setCenter(new GLatLng(37.4419， -122.1419)， 13);
      }
    }

    </script>
  </head>
  <body onload="initialize()" onunload="GUnload()">
    <div id="map_canvas" style="width： 500px; height： 300px"></div>
  </body>
</html>
```

網頁顯示 500x300 地圖，中心位於加州的 Palo Alto：

2. 載入 Google Maps API

```
<script src="http：
//maps.google.com/maps?file=api&v=2&key=abcdefg&sensor=true_or_false"
        type="text/Javascript">
</script>
```

http：//maps.google.com/maps?file=api&v=2&key=abcdefg 網址指向 JavaScript 檔案的位置，這些檔案包括使用「Google 地圖 API」時所需要的所有符號和定義。您的網頁必須包含指向此網址的 script 標籤，當您申請使用 API 時，請使用收到的金鑰。在此範例中，金鑰顯示為「abcdefg」。要注意的是，用傳送 sensor 參數，以指出此應用程式是否使用感應器來判斷使用者的位置。特別用 true_or_false 做為範例中變數的名稱，以強調**必須**明確地將此值設定為 true 或 false。

3. 地圖 DOM 元素

```
<div id="map_canvas" style="width： 500px; height： 300px"></div>
```

在網頁上必須預留顯示地圖的位置。通常，我們會建立名稱為 div 的元素，並取得瀏覽器文件物件模型 (DOM) 中此元素的參照。

在上述範例中，定義名稱為「map_canvas」的 div，並使用樣式屬性來設定它的大小。除非在建構函式中使用 GMapOptions 為地圖明確地指定大小，否則地圖會使用容器本身的大小。

4. GMap2 - 基本物件

```
var map = new GMap2(document.getElementById("map_canvas"));
```

代表地圖的 JavaScript 類別為 GMap2 類別。此類別的物件會定義網頁上的單一地圖。建立新的地圖實例時，可以指定網頁中的 DOM 節點 (通常是 div 元素) 做為地圖的容器。HTML 節點是 JavaScript document 物件的子項，而可以透過 document.getElementById() 方法來取得此元素的參照。

此程式碼會定義一個變數 (名稱為 map) 並指派該變數為新的 GMap2 物件。函式 GMap2() 稱為「建構函式」。

5. 初始化地圖

```
map.setCenter(new GLatLng(37.4419， -122.1419)， 13);
```

透過 GMap2 建構函式建立地圖後，需要再做一件事：將它初始化。使用 setCenter() 方法來完成地圖的初始化。setCenter() 方法需要 GLatLng 座標和縮放等級，而且此方法必須在地圖執行其他任何操作之前 (包括設定地圖本身的任何其他屬性)，就先行傳送。

6. 載入地圖

```
<body onload="initialize()" onunload="GUnload()">
```

呈現 HTML 網頁時，會建立文件物件模型 (DOM)，接收任何外部的圖片和指令碼，並與 document 物件進行整合。如果想確認地圖要在網頁完全載入後才會放上網頁，只要在 HTML 網頁的 <body> 元素接收到 onload 事件時，執行建構 GMap2 物件的函式即可。這樣做可以避免無法預期的行為，並讓我們更

能控制繪製地圖的方式和時機。onload 屬性是事件處理常式的範例。GUnload() 函式是一種公用程式，其目的是為了防止記憶體流失。

7. 控制項總覽

(1). GLargeMapControl -「Google 地圖」上使用的大型平移/縮放控制項。按照預設，會出現在地圖的左上角。

(2). GSmallMapControl -「Google 地圖」上使用的小型平移/縮放控制項。按照預設，會出現在地圖的左上角。

(3). GSmallZoomControl - 小型縮放控制項 (沒有平移控制項)，在「Google 地圖」上小型跳出視窗中，用來顯示行車導航。

(4). GScaleControl - 地圖比例尺。

(5). GMapTypeControl - 讓使用者切換地圖類型的按鈕 (例如 [地圖] 和 [衛星])

(6). GHierarchicalMapTypeControl - 巢狀按鈕和功能表項目的選項，可放置多種地圖類型選取器。

(7). GOverviewMapControl - 位於畫面角落、可摺疊的總覽地圖。

這些控制項都是實作 GControl 物件。

GMapTypeControl 和 GHierarchicalMapTypeControl 是特殊案例，因為它們也可以進行設定。在「Google 地圖 API」中，這些控制項會新增功能性，對目前地圖使用的 GMapType 進行變更。

目前支援地圖類型的清單顯示如下：

(1). G_NORMAL_MAP 顯示 Google 地圖的正常、預設 2D 地圖方塊

(2). G_SATELLITE_MAP 顯示圖形地圖方塊

(3). G_HYBRID_MAP 顯示圖形地圖方塊和特殊圖徵 (道路、城市名稱) 地圖方塊圖層的混合圖

(4). G_PHYSICAL_MAP 根據地形資訊顯示實際地圖方塊

根據預設，「Google 地圖 API」提供三種地圖類型：G_NORMAL_MAP、G_SATELLITE_MAP 以及 G_HYBRID_MAP。可以透過 GMap2.removeMapType() 移除地圖類型，或透過 GMap2.addMapType() 新增它們，來變更地圖上可用的地圖類型。不論何時建立地圖類型控制項，它都會使用目前附加的地圖類型，並讓這些地圖類型可透過該控制項使用。請注意，在新增控制項之前，必須指定地圖類型之間的任何關聯，讓地圖類型控制項知道那些關聯。

8. 新增控制項至地圖

```
map.addControl(new GLargeMapControl());
```

9. 繪製折線

```
var polyline = new GPolyline([
  new GLatLng(37.4419， -122.1419)， //第一點
  new GLatLng(37.4519， -122.1519) //第二點
]， "#ff0000"， 10);
map.addOverlay(polyline);
```

折線是由地圖上一系列直線段所繪製而成。可以指定線條的自訂色彩、粗細以及透明度。色彩為十六進位數字 HTML 格式，例如使用 #ff0000，而非 red。GPolyline 不瞭解命名色彩。

Google Maps API for Flash 建置步驟

1. 使用網址申請 Google 地圖 API 金鑰

申請網址 http：//code.google.com/intl/zh-TW/apis/maps/signup.html

範例：

```
import com.google.maps.LatLng;
import com.google.maps.Map;
import com.google.maps.MapEvent;
import com.google.maps.MapType;

var map：Map = new Map();
map.key = "your_api_key";

map.addEventListener(MapEvent.MAP_PREINITIALIZE，
onMapPreinitialize);

function onMappreinitialize(event：MapEvent)：void {
  setInitOptions(
    new MapOptions({
      center： new LatLng(40.736072，-73.992062)，
      zoom： 14，
      mapType： MapType.NORMAL_MAP_TYPE
    }));
}
```

2. 匯入程式庫

```
import com.google.maps.LatLng;
import com.google.maps.Map;
import com.google.maps.MapEvent;
import com.google.maps.MapType;
```

ActionScript 程式庫會以 import 宣告匯入。在上述範例程式碼中，匯入了幾個「Google 地圖」Flash 程式庫。如果要成功編譯 SWF 檔，必須匯入範例程式碼中所使用類型的程式庫。

3. 宣告地圖

```
var map：Map = new Map();
```

```
map.key = "your_api_key";
```

com.google.maps.Map 物件是 Google Maps API for Flash 程式庫中的基本物件。這裡會具現化 Map 物件，並將此物件指派給 map 變數。接下來將 map 的 key 屬性設定為已註冊的 API 金鑰。

4. 設定地圖大小

```
map.setSize(newPoint(stage.stageWidth，
stage.stageHeight));
```

5. 設定事件監聽程式

```
map.addEventListener(MapEvent.MAP_READY， onMapReady);
```

在此處為 Map 物件的 MapEvent.MAP_READY 事件新增事件接聽程式。此事件處理常式扮演著橋樑的角色，讓 Google Maps API for Flash 應用程式能夠順利初始化。只要地圖接收到該事件，就會呼叫 onMapReady 函式（如下所示）。

```
function onMapReady(event：MapEvent)：void { setCenter(new
LatLng(40.736072，-73.992062)，14，MapType.NORMAL_MAP_TYPE);}
```

onMapReady() 函式傳送類型 MapEvent（在此情況下會忽略）的 event 參數，然後使用指定的參數（定義位置、縮放等級以及要顯示的地圖類型）呼叫 setCenter()。

地圖控制項總覽

(1). PositionControl - 平移控制項（如同「Google 地圖」一般)。根據預設，這會出現在地圖的左上角。

(2). ZoomControl - 縮放控制滑桿（如同「Google 地圖」一般)。

(3). MapTypeControl - 讓使用者切換地圖類型的按鈕（例如［地圖］和［衛]）。

(4). ScaleControl - 比例尺控制項是視覺的指示器，可以指出目前地圖的解析度和縮放等級。

(5). OverviewMapControl - 位於畫面角落、可摺疊的總覽地圖。

6. 新增控制項至地圖

```
map.addControl(new PositionControl());
```

使用 Map 方法 addControl() 新增控制項至地圖。

除了 PositionControl 新增內建的 ZoomControl、PositionControl 及 MapTypeControl 控制項，可讓使用者平移/縮放地圖並在不同的地圖類型間切換。這些標準控制項只要包含至地圖即可正常運作。

```
private function onMapReady(event：MapEvent)：void {
```

```
  map.setCenter(new LatLng(42.366662，-71.106262)， 11，
MapType.NORMAL_MAP_TYPE);
  map.addControl(new ZoomControl());
  map.addControl(new PositionControl());
  map.addControl(new MapTypeControl());
}
```

7. 設定標準控制項

部分控制項（例如 MapTypeControl）可進行設定。例如，Google Maps API 提供四種地圖類型:NORMAL_MAP_TYPE、SATELLITE_MAP_TYPE、HYBRID_MAP_TYPE 及 PHYSICAL_MAP_TYPE。可以透過 Map.removeMapType() 移除現有類型或 Map.addMapType() 新增它們，來變更地圖上可用的地圖類型。不論何時建立地圖類型控制項，它都會使用目前附加的地圖類型，並讓這些地圖類型可透過該控制項使用。

下方程式碼可從地圖附加的可用地圖類型中移除 HYBRID_MAP_TYPE，只留下三種地圖類型。一旦使用者新增 MapTypeControl，就只能使用這三種地圖類型。

```
private function onMapReady(event：MapEvent)：void {
  map.setCenter(new LatLng(42.366662，-71.106262)， 11，
MapType.NORMAL_MAP_TYPE);
  map.removeMapType(MapType.HYBRID_MAP_TYPE);
  map.addControl(new MapTypeControl());
}
```

8. 繪製折線

折線是由地圖上一系列直線段所繪製而成。可以為線條指定自訂的色彩、線條粗細和透明度。色彩應該使用 com.google.maps.Color 中的數值。Polyline 不瞭解命名色彩。

折線屬性指定於 PolylineOptions 中。以下是目前支援的折線選項：

- geodesic 指定兩點之間的折線應描繪為測地線（「大圓」）。
- strokeStyle 指定繪製折線時使用的 StrokeStyle。

下方程式碼片段會在兩點之間建立 4 像素寬的紅色折線：

```
        private function onMapReady(event：MapEvent)：void {
          map.setCenter(new LatLng(37.4419， -122.1419)， 13，
        MapType.NORMAL_MAP_TYPE);
```

```
  // Polyline overlay.
  var polyline:Polyline = new Polyline([
      new LatLng(37.4419, -122.1419),
      new LatLng(37.4519, -122.1519)
      ], new PolylineOptions({ strokeStyle: new
StrokeStyle({
            color: 0xFF0000,
            thickness: 4,
            alpha: 0.7})
  }));

  map.addOverlay(polyline);
}
```

無人飛行船自動導航系統 行前確認單

日期:_____

器材:

名稱	數量	確認	備註
主板	1		
GPRS(加 SIM 卡)	1		
筆電	2		
網卡	1		
程式檔案			
燒錄線	1		
RJ11	1		
RS232 線	1		
電源延長線(三孔)	2		
電源捲線器	1		
GPS 天線	1		
攝影機+腳架	1		
電池(主板+GPRS)	2		
電池(繼電器)	1		
遙控器	1		
充電器(遙控器)	1		
充電器(主板)	1		
充電器(繼電器)	1		
充電器(馬達)	1		
筆電電源線	2		
魔鬼氈			
程式燒錄機	1		

確認事項:

編號	事項	確認	備註
1.	電源電力是否充足?		
2.	遙控器電力是否充足?		
3.	DV 電源充足?		
4.	主板確認線材無缺損?		
5.	主板固定於飛船上?		
6.	教室監控介面是否開啟?		
7.	Wamp 開啟?		
8.	操場鐵門開放?		

9.	體育器材室借用?		
10.	操場樹林高度?		
11.	測量山下空地 GPS 位置?		
12.	低空飛行測試?		
13.	GPS 天線貼於飛船位置?		
14.			

動態展示各版本詳細資訊

- 場景設計

A. 版本 2

進場畫面 1

進場畫面 2

進場畫面 3

1. 控制系統(手動/自動)-晶片控制
2. 控制系統(飛控/地控)-地面站系統
3. 訊號比
4. 指北針
5. 中景飛船
6. 地板
7. 近景 (side view)
8. 轉角
9. 傾角
10. 推力及馬達出力倍率
11. 晶片溫度
12. 旋轉底座
13. 遠景 (bird view)-Google map(標示點為飛船所在地)
14. 雲

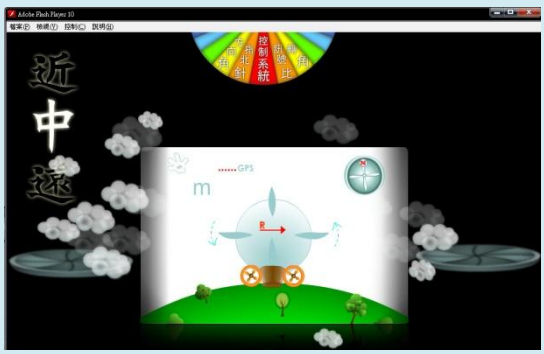
B. 版本 3

進場畫面 1	進場畫面 2
	

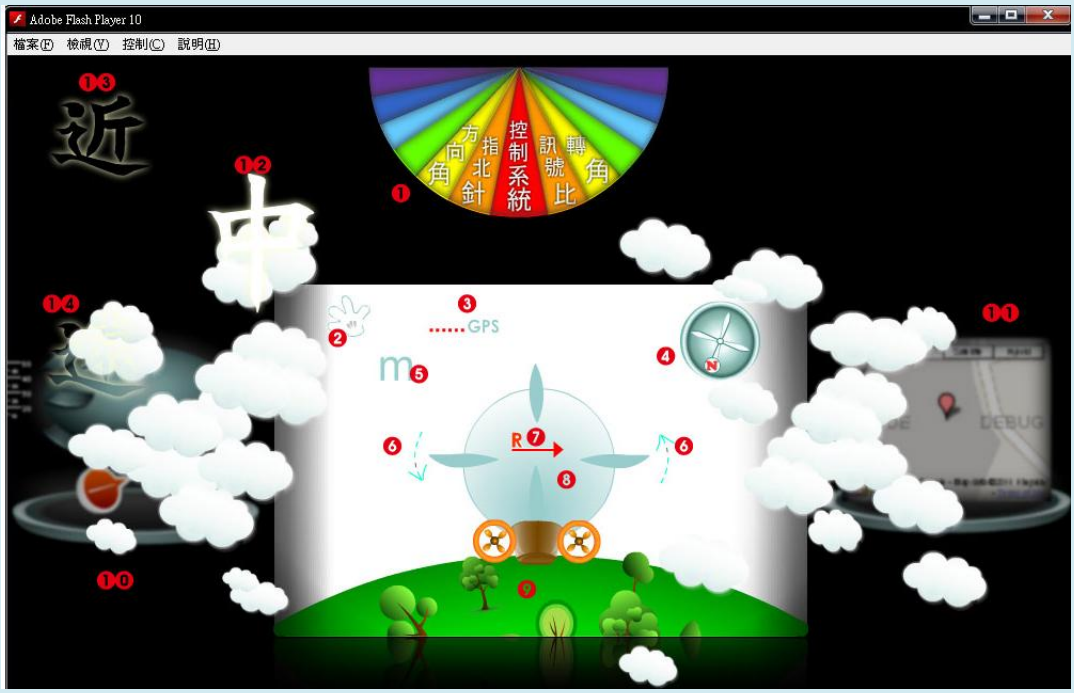
進場畫面 3

1. 彩虹按鈕
2. 指北針
3. 中景飛船
4. 地板
5. 近景飛船
6. 轉角
7. 傾角
8. 推力及馬達出力倍率
9. 晶片溫度
10. 旋轉底座
11. 遠景 (bird view)-Google map(標示點為飛船所在地)
12. 雲

C. 版本 4

進場畫面 1	進場畫面 2
	

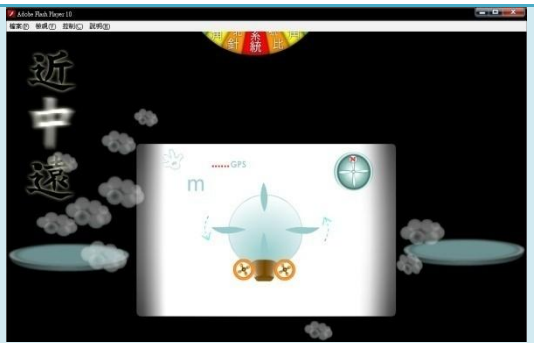
進場畫面 3



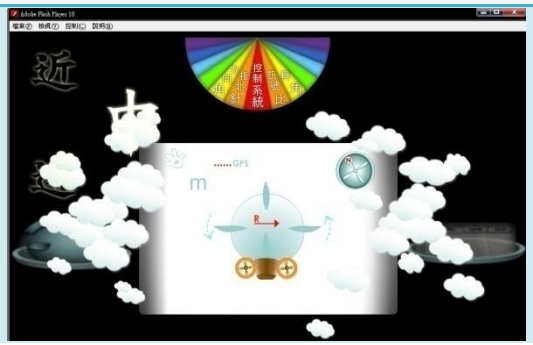
1. 彩虹按鈕
2. 控制系統(手動/自動)-遙控器控制
3. 訊號比
4. 指北針
5. 距離
6. 轉角
7. 方向角
8. 中景
9. 地板
10. 近景(side view)
2. 遠景 (bird view)-Google map(標示點為飛船所在地)
11. 中景按鈕
12. 近景按鈕
13. 遠景按鈕

D. 版本 5

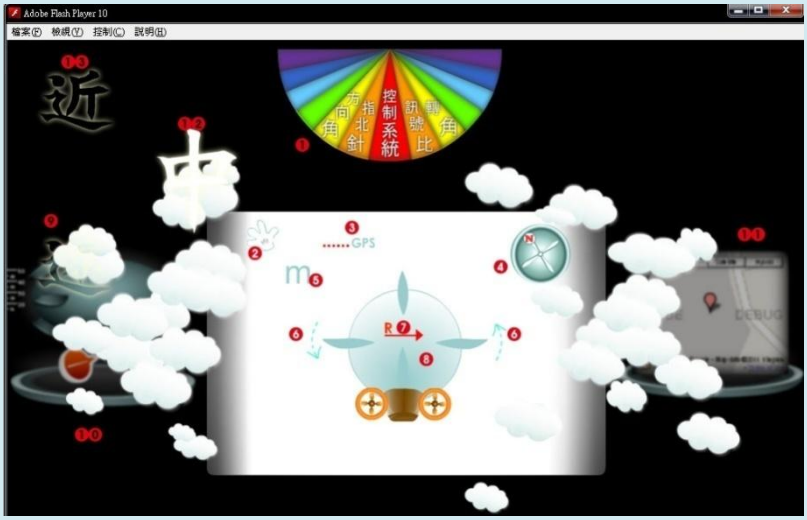
進場畫面 1



進場畫面 2

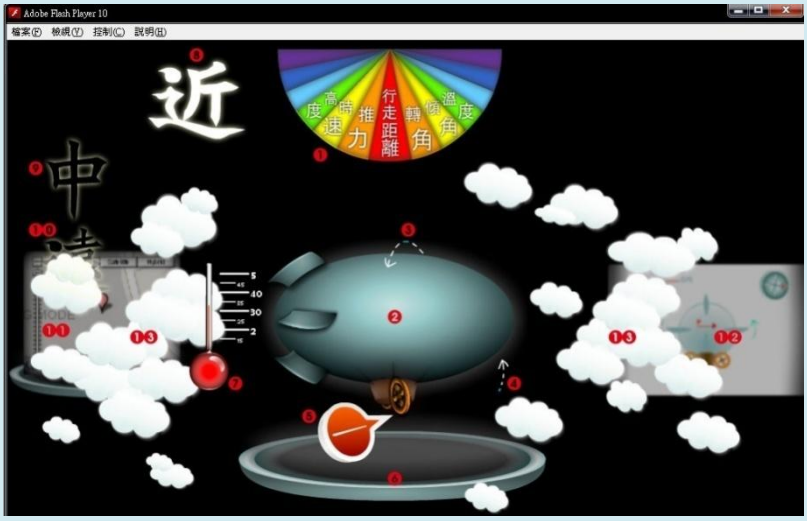


進場畫面 3(中景)



1. 中景-彩虹按鈕
2. 遙控器控制
3. 訊號比
4. 指北針
5. 距離
6. 轉角
7. 方向角
8. 中景飛船
9. 遠景按鈕
10. 近景
11. 遠景
12. 中景按鈕(按下)
13. 近景按鈕

進場畫面 4(近景)



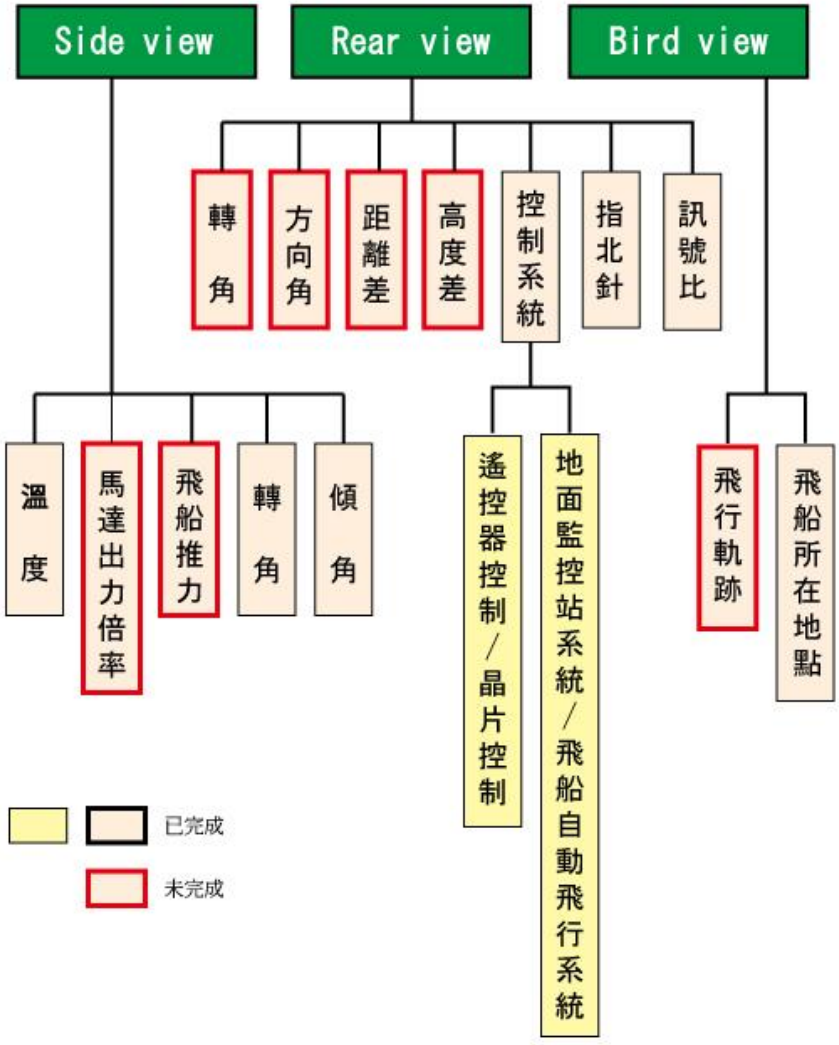
1. 近景-彩虹按鈕
2. 近景飛船
3. 轉角
4. 傾角
5. 推力+馬達輸出倍率
6. 旋轉底盤
7. 晶片溫度
8. 近景按鈕(按下)
9. 中景按鈕
10. 遠景按鈕
11. 遠景
12. 中景
13. 雲

進場畫面 5(遠景)

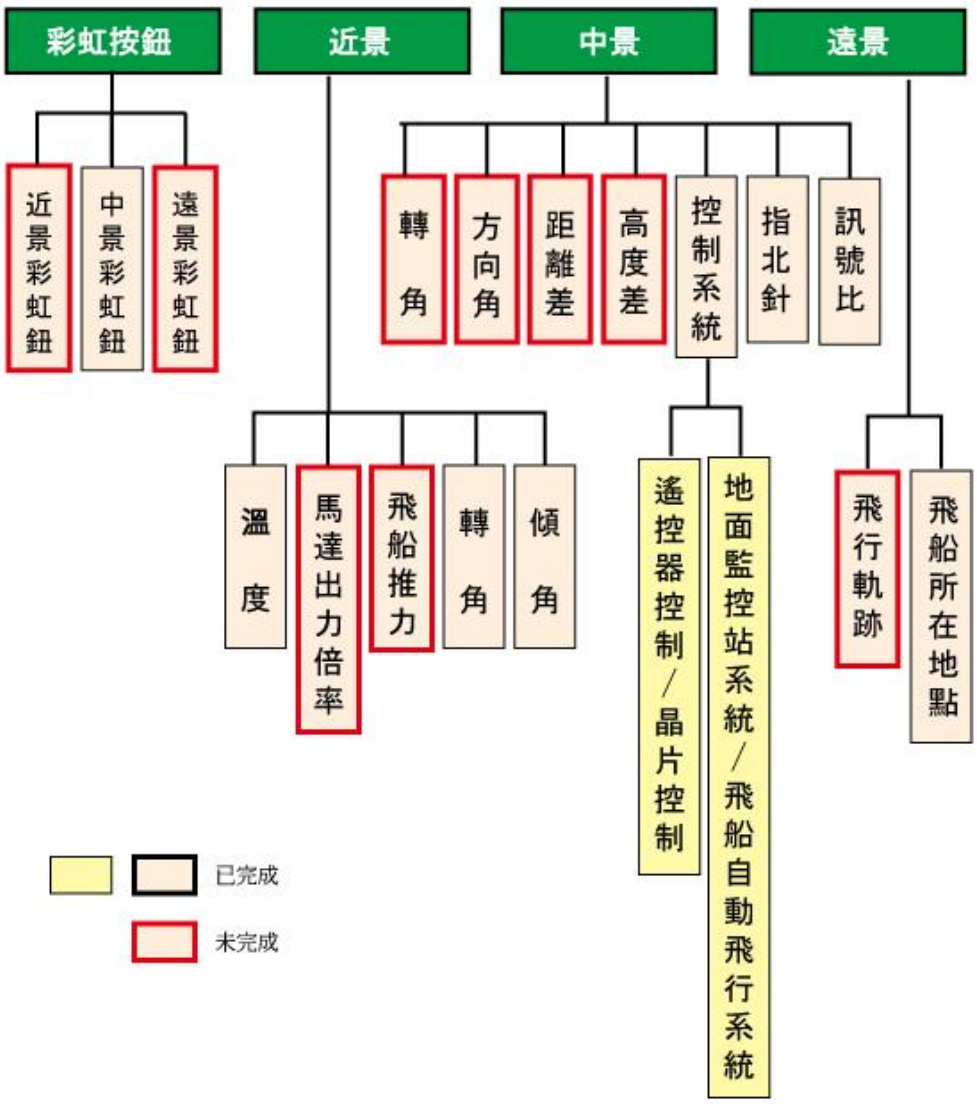
1. 遠景-彩虹按鈕
2. 遠景-Google map
3. 近景按鈕
4. 中景按鈕
5. 遠景按鈕(按下)
6. 中景
7. 近景
8. 雲

- 動畫架構

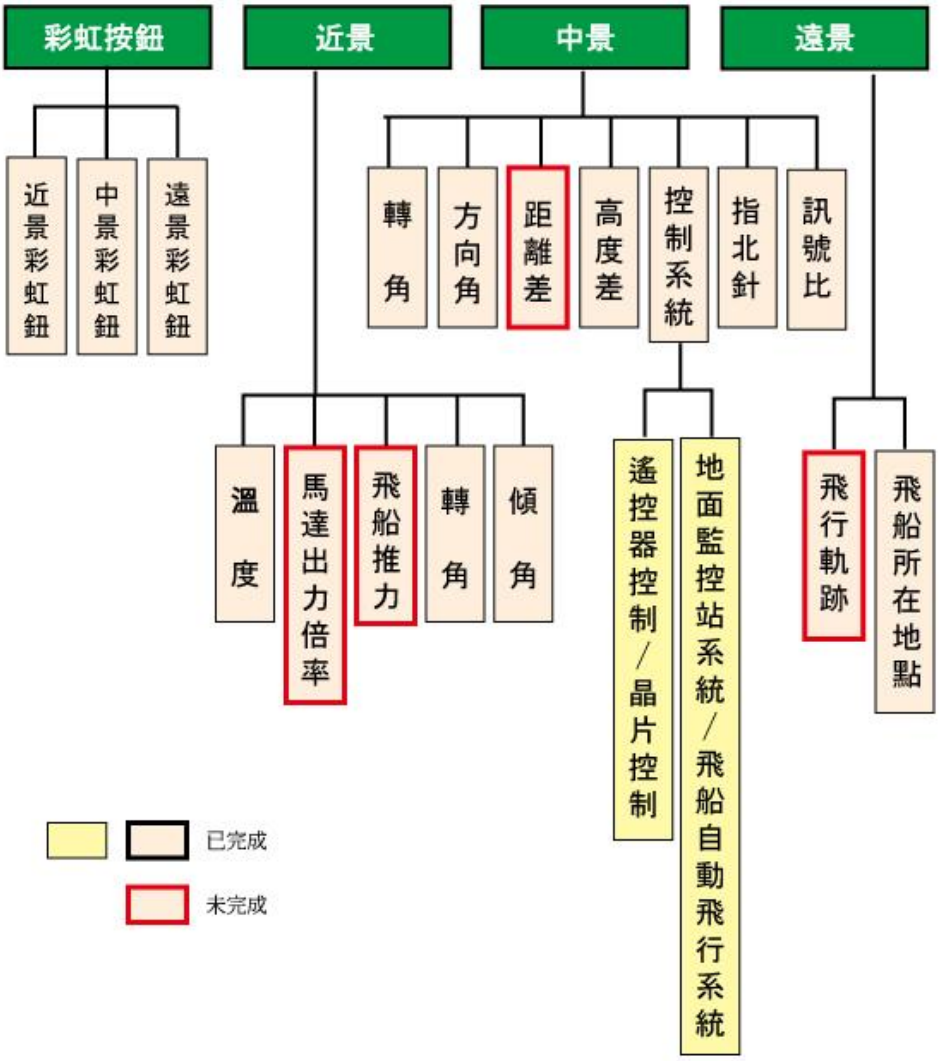
  A. 版本 2


Side view
Rear view
Bird view
轉角
方向角
距離差
高度差
控制系統
指北針
訊號比
溫度
馬達出力倍率
飛船推力
轉角
傾角
遙控器控制／晶片控制
地面監控站系統／飛船自動飛行系統
飛行軌跡
飛船所在地點
已完成
未完成

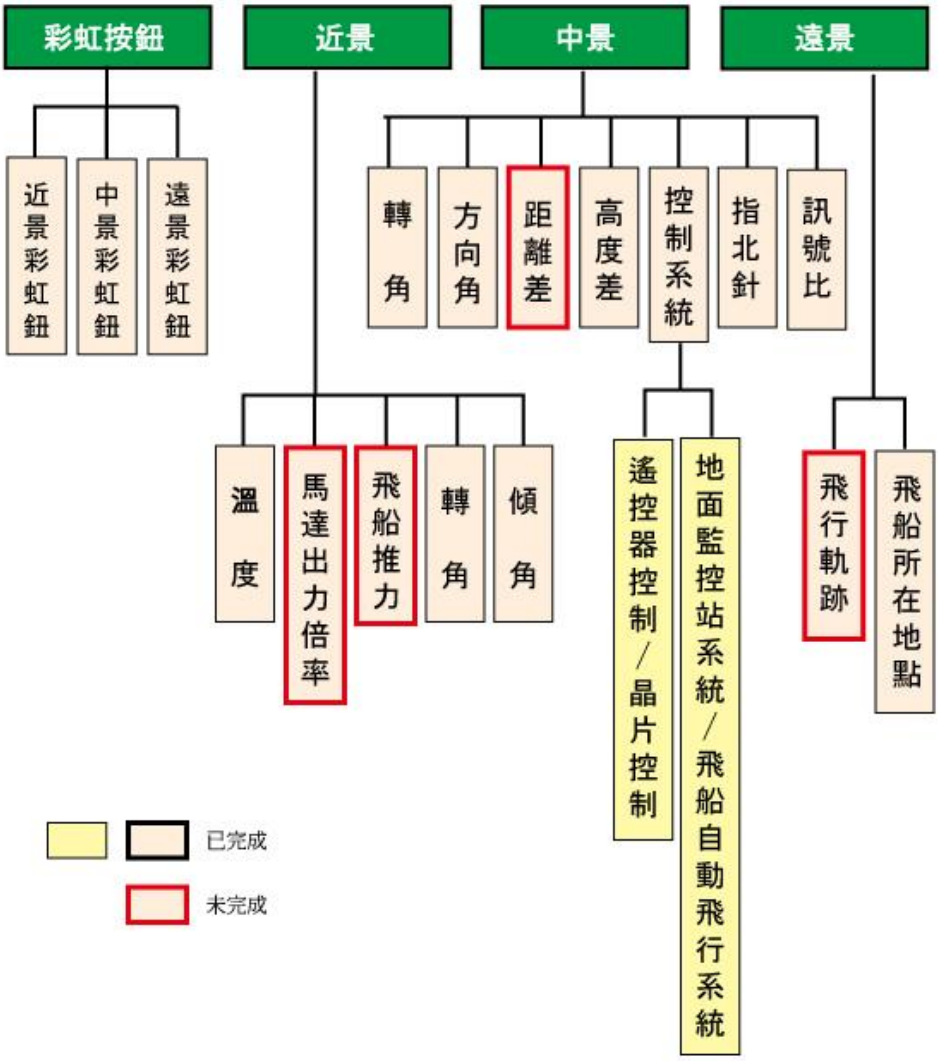
B. 版本 3


彩虹按鈕
近景
中景
遠景
近景彩虹鈕
中景彩虹鈕
遠景彩虹鈕
轉角
方向角
距離差
高度差
控制系統
指北針
訊號比
溫度
馬達出力倍率
飛船推力
轉角
傾角
遙控器控制／晶片控制
地面監控站系統／飛船自動飛行系統
飛行軌跡
飛船所在地點
已完成
未完成

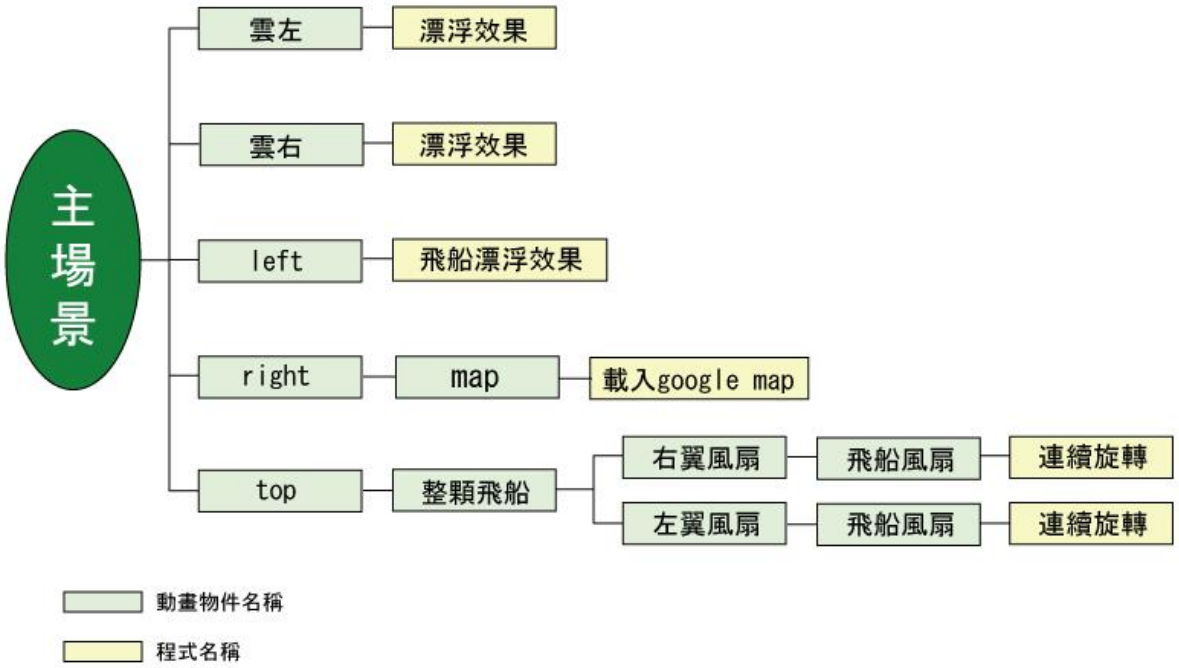
C. 版本 4


彩虹按鈕
近景
中景
遠景
近景彩虹鈕
中景彩虹鈕
遠景彩虹鈕
轉角
方向角
距離差
高度差
控制系統
指北針
訊號比
溫度
馬達出力倍率
飛船推力
轉角
傾角
遙控器控制／晶片控制
地面監控站系統／飛船自動飛行系統
飛行軌跡
飛船所在地點
已完成
未完成

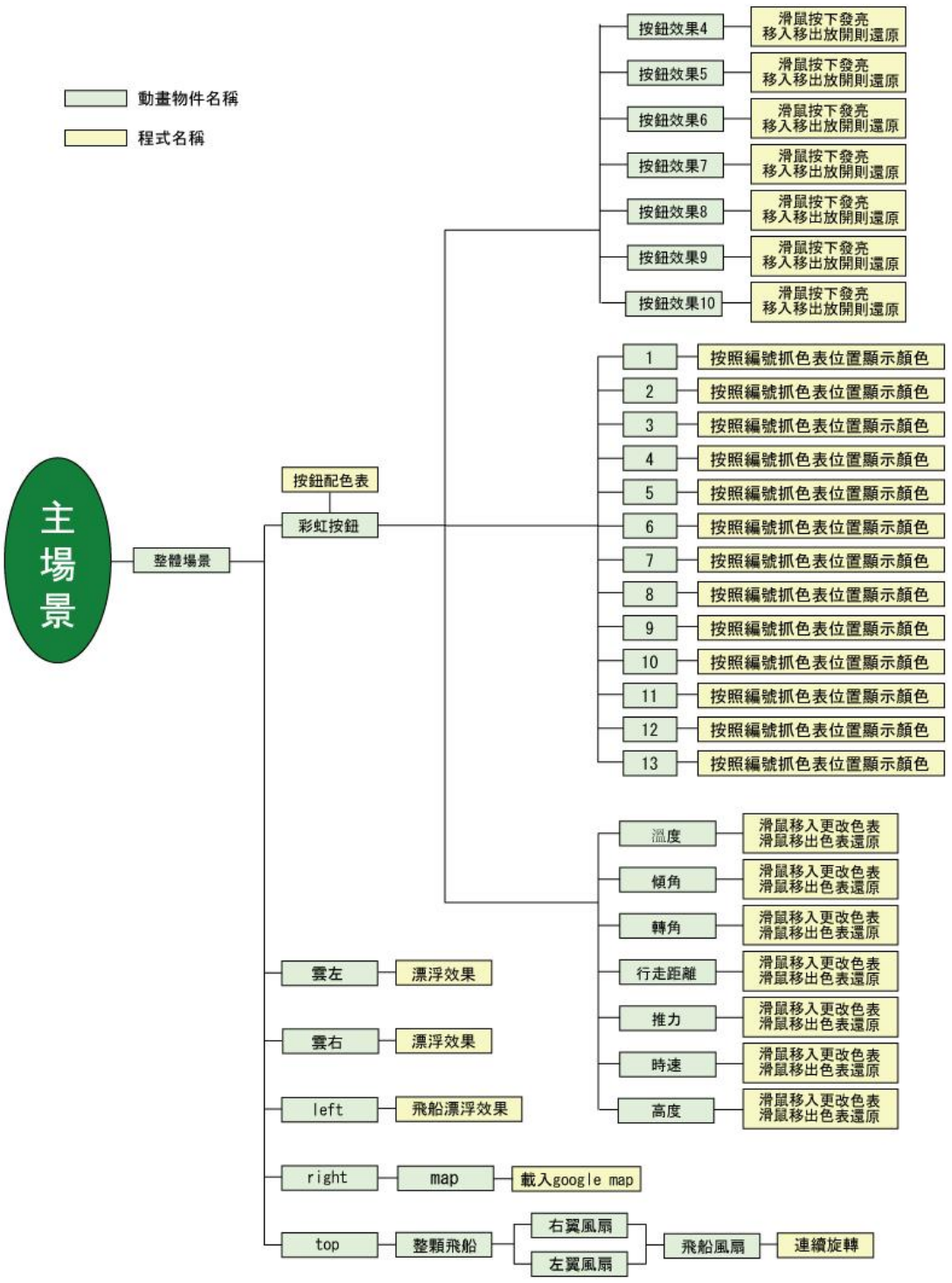
D. 版本 5


彩虹按鈕
近景
中景
遠景
近景彩虹鈕
中景彩虹鈕
遠景彩虹鈕
轉角
方向角
距離差
高度差
控制系統
指北針
訊號比
溫度
馬達出力倍率
飛船推力
轉角
傾角
遙控器控制／晶片控制
地面監控站系統／飛船自動飛行系統
飛行軌跡
飛船所在地點
已完成
未完成

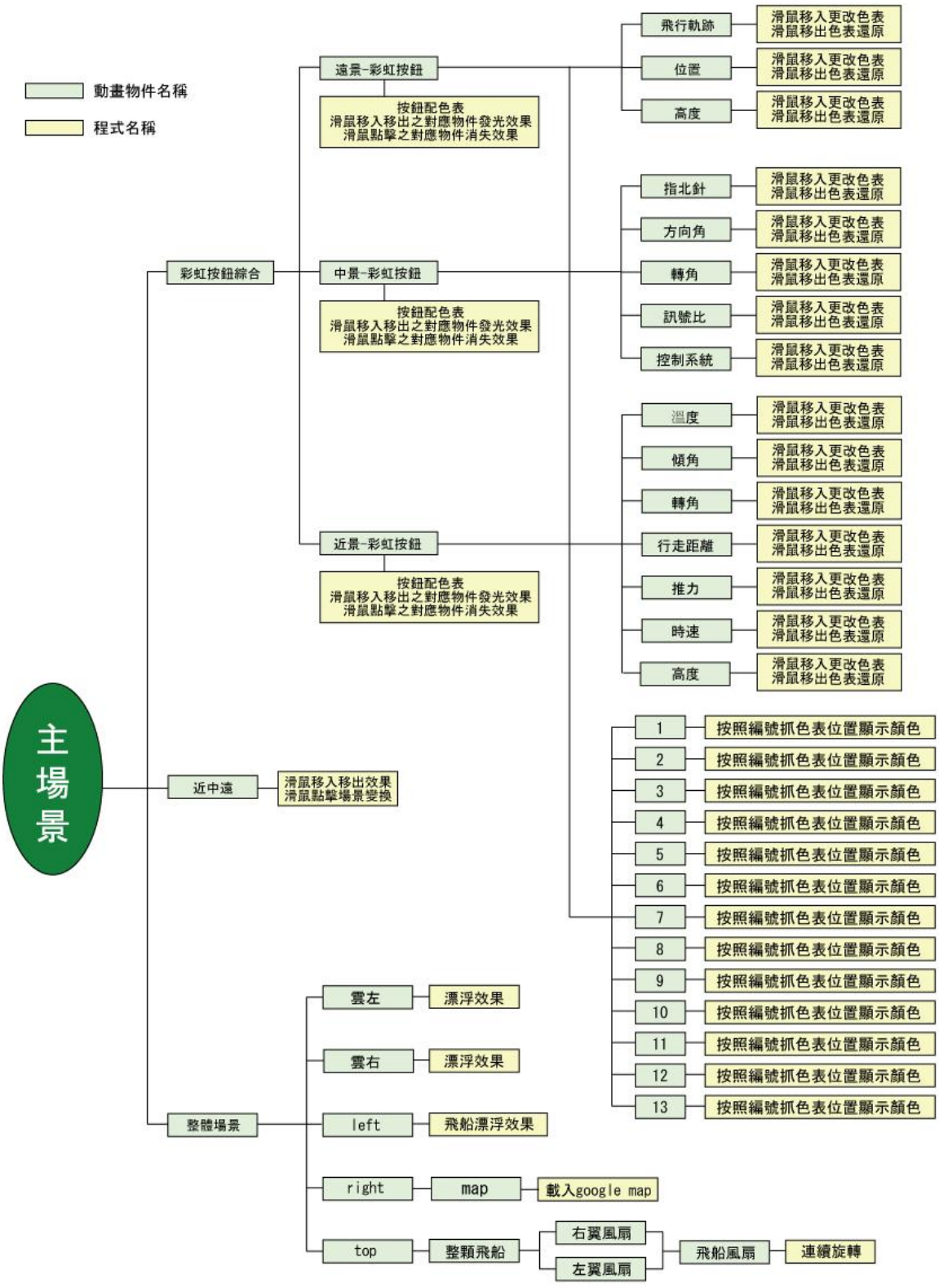
- 物件導向程式設計
  A. 版本 2



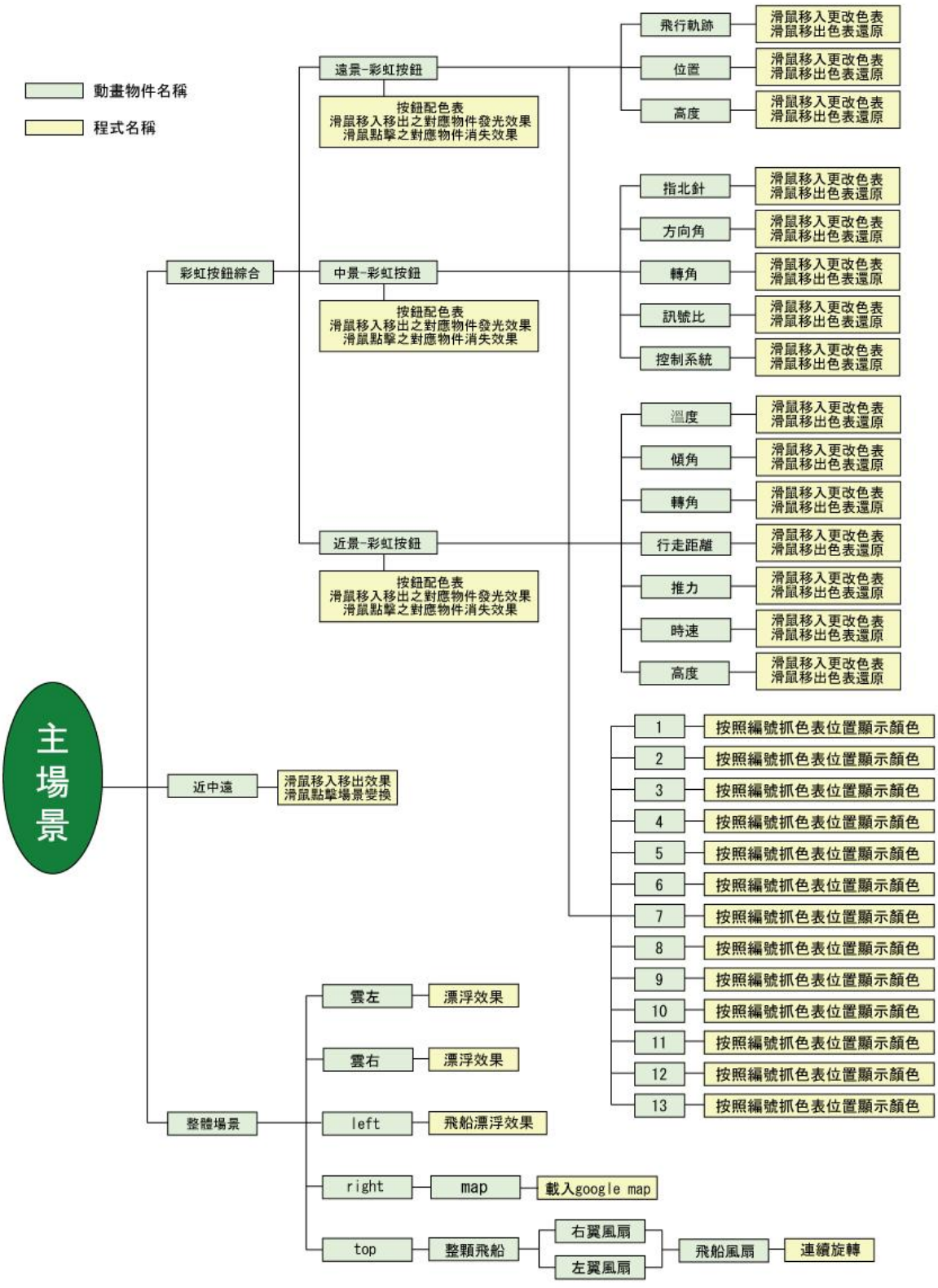
B. 版本 3


動畫物件名稱
程式名稱
主場景
整體場景
按鈕配色表
彩虹按鈕
雲左
漂浮效果
雲右
漂浮效果
left
飛船漂浮效果
right
map
載入google map
top
整顆飛船
右翼風扇
左翼風扇
飛船風扇
連續旋轉
按鈕效果4
按鈕效果5
按鈕效果6
按鈕效果7
按鈕效果8
按鈕效果9
按鈕效果10
滑鼠按下發亮
移入移出放開則還原
1
2
3
4
5
6
7
8
9
10
11
12
13
按照編號抓色表位置顯示顏色
溫度
傾角
轉角
行走距離
推力
時速
高度
滑鼠移入更改色表
滑鼠移出色表還原

C. 版本 4


動畫物件名稱
程式名稱
主場景
彩虹按鈕綜合
遠景-彩虹按鈕
按鈕配色表
滑鼠移入移出之對應物件發光效果
滑鼠點擊之對應物件消失效果
中景-彩虹按鈕
按鈕配色表
滑鼠移入移出之對應物件發光效果
滑鼠點擊之對應物件消失效果
近景-彩虹按鈕
按鈕配色表
滑鼠移入移出之對應物件發光效果
滑鼠點擊之對應物件消失效果
飛行軌跡
位置
高度
指北針
方向角
轉角
訊號比
控制系統
溫度
傾角
轉角
行走距離
推力
時速
高度
滑鼠移入更改色表
滑鼠移出色表還原
1
2
3
4
5
6
7
8
9
10
11
12
13
按照編號抓色表位置顯示顏色
近中遠
滑鼠移入移出效果
滑鼠點擊場景變換
整體場景
雲左
雲右
left
right
top
漂浮效果
飛船漂浮效果
map
載入google map
整顆飛船
右翼風扇
左翼風扇
飛船風扇
連續旋轉

D. 版本 5



附錄 5.6.2.2-5

Flash 展示系統各版本內含程式碼

版本 1

主場景-載入資料及顯示

```
//建立一個 URL 載入容器
var url 載入器:URLLoader = new URLLoader();//url=網址連結
//設定為一變數對應一數值
url 載入器.dataFormat=URLLoaderDataFormat.VARIABLES;//資料格式=載入變數
//位置為遊戲主機位置
var server:String='http://localhost/opening2.php';
//建立 URL 請求物件
var  要求連結:URLRequest=new URLRequest(server);
//設定傳送方式
要求連結.method=URLRequestMethod.POST;//改 post 或 get 就可以用不同方式傳，也要用不同方式收
//建立傳送數值物件，也是用來存變數的容器
var  資料封包:URLVariables = new URLVariables();
//傳送的東西完成了的事件
url 載入器.addEventListener(Event.COMPLETE,loaded);
//完成了就做下面的動作
function loaded(e:Event) {
      //trace(myloader.data.msg);//php 處理後回報怎樣的訊息，秀在輸出面版
出發時間.text=url 載入器.data.ymdhms;//秀在 page6 的 remsg 文字框了 1
出發位置 ns.text=url 載入器.data.stptns;//秀在 page6 的 remsg 文字框了 2
出發位置 ew.text=url 載入器.data.stptew;//秀在 page6 的 remsg 文字框了 2
中繼站 ns.text=url 載入器.data.nextptns;//秀在 page6 的 remsg 文字框了 3
中繼站 ew.text=url 載入器.data.nextptew;//秀在 page6 的 remsg 文字框了 3
終點站 ns.text=url 載入器.data.endptns;//秀在 page6 的 remsg 文字框了 4
終點站 ew.text=url 載入器.data.endptew;//秀在 page6 的 remsg 文字框了 4
目前位置 ns.text=url 載入器.data.nowptns;//秀在 page6 的 remsg 文字框了 5
目前位置 ew.text=url 載入器.data.nowptew;//秀在 page6 的 remsg 文字框了 5
時速.text=url 載入器.data.hkm;//秀在 page6 的 remsg 文字框了 6
已花費時間.text=url 載入器.data.diff;//秀在 page6 的 remsg 文字框了 7
已行走距離.text=url 載入器.data.dm;//秀在 page6 的 remsg 文字框了 8
高度.text=url 載入器.data.high;//秀在 page6 的 remsg 文字框了 9
頃角.text=url 載入器.data.ln1;//秀在 page6 的 remsg 文字框了 10
方向角.text=url 載入器.data.ln2;//秀在 page6 的 remsg 文字框了 11
轉彎方向.text=url 載入器.data.turn;//秀在 page6 的 remsg 文字框了 12
}
```

```
this.addEventListener(Event.ENTER_FRAME,重新載入);//監聽程式-進入影格開始     this=這個物件本身
function  重新載入(e:Event) {
        var i:int=0;
        if(i%50==0){ //(影格跑 25 格)i 跟 25 取餘數=0，則重新載入
        url 載入器.load(要求連結);
        }i++;
}
```

2. 版本 2

雲左物件-漂浮效果

```
stop();
var ix:int=1;
var iy:int=1;
var jx:int=-1;
var jy:int=1;
var kx:int=1;
var ky:int=-1;
var lx:int=-1;
var ly:int=-1;
this.addEventListener(Event.ENTER_FRAME,flyyy);
//這個物件.設一監聽程式(事件類型.進入影格.做"rota")
function flyyy(e:Event):void {
        /*left1.alpha=1;
        left2.alpha=1;
        left3.alpha=1;
        left4.alpha=1;*/


        left1.x+=0.05*ix;//進入 1 影格.移動 1 像素
        left1.y+=0.05*iy;

        left2.x+=0.05*jx;
        left2.y+=0.05*jy;

        left3.x+=0.05*kx;
        left3.y+=0.05*ky;
```

```
        left4.x+=0.05*lx;
        left4.y+=0.05*ly;

        if (left1.x>=-25||left1.x<=-27) {
                ix=- ix;
        }
        if (left1.y>=-244||left1.y<=-246) {
                iy=- iy;
        }
        if (left2.x>=7||left2.x<=5) {
                jx=- jx;
        }
        if (left2.y>=-259||left2.y<=-261) {
                jy=- jy;
        }
        if (left3.x>=-53||left3.x<=-55) {
                kx=- kx;
        }
        if (left3.y>=-76||left3.y<=-78) {
                ky=- ky;
        }
        if (left4.x>=-13||left4.x<=-15) {
                lx=- lx;
        }
        if (left4.y>=-162||left4.y<=-164) {
                ly=- ly;
        }
}
```

雲右物件-漂浮效果

```
stop();
var ix:int=1;
var iy:int=1;
var jx:int=-1;
var jy:int=1;
var kx:int=1;
var ky:int=-1;
var lx:int=-1;
```

```
var ly:int=-1;
this.addEventListener(Event.ENTER_FRAME,flyy);
//這個物件.設一監聽程式(事件類型.進入影格.做"rota")
function flyy(e:Event):void {
	/*rigjt1.alpha=1;
	rigjt2.alpha=1;
	rigjt3.alpha=1;
	rigjt4.alpha=1;*/


	rigjt1.x+=0.05*ix;//進入 1 影格.移動 1 像素
	rigjt1.y+=0.05*iy;

	rigjt2.x+=0.05*jx;
	rigjt2.y+=0.05*jy;

	rigjt3.x+=0.05*kx;
	rigjt3.y+=0.05*ky;

	rigjt4.x+=0.05*lx;
	rigjt4.y+=0.05*ly;

	if (rigjt1.x>=103||rigjt1.x<=100) {
		ix=- ix;
	}
	if (rigjt1.y>=-108||rigjt1.y<=-110) {
		iy=- iy;
	}
	if (rigjt2.x>=29||rigjt2.x<=27) {
		jx=- jx;
	}
	if (rigjt2.y>=-263||rigjt2.y<=-265) {
		jy=- jy;
	}
	if (rigjt3.x>=111||rigjt3.x<=108) {
		kx=- kx;
	}
	if (rigjt3.y>=-181||rigjt3.y<=-183) {
```

```
            ky=- ky;
        }
        if (rigjt4.x>=84||rigjt4.x<=82) {
            lx=- lx;
        }
        if (rigjt4.y>=-247||rigjt4.y<=-249) {
            ly=- ly;
        }
}
```

Left 物件-飛船漂浮效果

```
stop();
var ix:int=1;
var iy:int=1;
this.addEventListener(Event.ENTER_FRAME,flyb);
//這個物件.設一監聽程式(事件類型.進入影格.做"rota")
function flyb(e:Event):void {
        flyboatxl.x+=2*ix;//進入 1 影格.移動 1 像素
        flyboatxl.y+=0.5*iy;

        if (flyboatxl.x>=6||flyboatxl.x<=-6) {
            ix=- ix;
        }
        if (flyboatxl.y>=3||flyboatxl.y<=-3) {
            iy=- iy;
        }
}
```

Map 物件-載入 google map

```
import com.google.maps.Map;
import com.google.maps.LatLng;
import com.google.maps.MapEvent;
import com.google.maps.MapMouseEvent;
import com.google.maps.MapType;
import com.google.maps.overlays.Marker;
import com.google.maps.overlays.MarkerOptions;
import com.google.maps.overlays.Polyline;
import com.google.maps.overlays.PolylineOptions;
```

```
import com.google.maps.overlays.GroundOverlay;
import com.google.maps.overlays.GroundOverlayOptions;
import com.google.maps.LatLng;
import com.google.maps.LatLngBounds;
import com.google.maps.styles.FillStyle;
import com.google.maps.styles.StrokeStyle;
import com.google.maps.InfoWindowOptions;
import com.google.maps.Color ;
import com.google.maps.controls.ZoomControl;
import com.google.maps.controls.ZoomControlOptions;
import com.google.maps.controls.PositionControlOptions;
import com.google.maps.controls.PositionControl;
import com.google.maps.controls.MapTypeControl;
import com.google.maps.controls.MapTypeControlOptions;
import com.google.maps.controls.ControlPosition;
import flash.display.Loader;
import flash.display.LoaderInfo;
import flash.net.URLRequest;



var map:Map = new Map();
map.key =
"ABQIAAAAB5gTmMnSQeb_llJwdOJkuhQevdxZWrZCE2EIyPUBGw1wKNFKlxSkAs73wWuXLQtQX_0LzfnrY
D2WeQ";
//map.addControl(new ZoomControl());
//map.addControl(new PositionControl());
map.addControl(new MapTypeControl());
map.setSize(new Point(280, 220));
map.addEventListener(MapEvent.MAP_READY, onMapReady);
this.addChild(map);

  function onMapReady(event:Event):void
{
      var topLeft:ControlPosition =    new ControlPosition(ControlPosition.ANCHOR_TOP_LEFT, 16, 10);
      var myZoomControl:ZoomControl = new ZoomControl(new ZoomControlOptions({position:
topLeft}));
        map.addControl(myZoomControl);
```

```
        map.removeMapType(MapType.PHYSICAL_MAP_TYPE);
        map.setCenter(new LatLng(22.905926,120.469207),18, MapType.NORMAL_MAP_TYPE);
        var markerA:Marker = new Marker
    (
            new LatLng(22.905926,120.469207),
            new MarkerOptions
            ({
                    strokeStyle: new StrokeStyle({color: 0x000000}),
                    fillStyle: new FillStyle({color: 0xFF8888, alpha:1}),
                    radius: 10,
                    hasShadow: true
            })
    );
      markerA.addEventListener(MapMouseEvent.CLICK, function(event:Event):void {
      markerA.openInfoWindow(new InfoWindowOptions({content:"緯度:22.905926:經度:120.469207"}));
    });
    map.addOverlay(markerA);

}
```

整顆飛船-連續旋轉

```
var i:int=20;
var j:int=1;
var a:int=40;
this.addEventListener(Event.ENTER_FRAME,rota);
    //這個物件.設一監聽程式(事件類型.進入影格.做"rota")
function rota(e:Event):void{    //
        this.rotation = this.rotation - a;
//      j++;
//      a=i*j;
}
```

版本 3

雲左物件-漂浮效果

```
stop();
var ix:int=1;
var iy:int=1;
var jx:int=-1;
```

```
var jy:int=1;
var kx:int=1;
var ky:int=-1;
var lx:int=-1;
var ly:int=-1;
this.addEventListener(Event.ENTER_FRAME,flyyy);
//這個物件.設一監聽程式(事件類型.進入影格.做"rota")
function flyyy(e:Event):void {
	/*left1.alpha=1;
	left2.alpha=1;
	left3.alpha=1;
	left4.alpha=1;*/


	left1.x+=0.05*ix;//進入 1 影格.移動 1 像素
	left1.y+=0.05*iy;

	left2.x+=0.05*jx;
	left2.y+=0.05*jy;

	left3.x+=0.05*kx;
	left3.y+=0.05*ky;

	left4.x+=0.05*lx;
	left4.y+=0.05*ly;

	if (left1.x>=-25||left1.x<=-27) {
		ix=- ix;
	}
	if (left1.y>=-244||left1.y<=-246) {
		iy=- iy;
	}
	if (left2.x>=7||left2.x<=5) {
		jx=- jx;
	}
	if (left2.y>=-259||left2.y<=-261) {
		jy=- jy;
	}
```

```
        if (left3.x>=-53||left3.x<=-55) {
                kx=- kx;
        }
        if (left3.y>=-76||left3.y<=-78) {
                ky=- ky;
        }
        if (left4.x>=-13||left4.x<=-15) {
                lx=- lx;
        }
        if (left4.y>=-162||left4.y<=-164) {
                ly=- ly;
        }
}
```

雲右物件-漂浮效果

```
stop();
var ix:int=1;
var iy:int=1;
var jx:int=-1;
var jy:int=1;
var kx:int=1;
var ky:int=-1;
var lx:int=-1;
var ly:int=-1;
this.addEventListener(Event.ENTER_FRAME,flyy);
//這個物件.設一監聽程式(事件類型.進入影格.做"rota")
function flyy(e:Event):void {
        /*rigjt1.alpha=1;
        rigjt2.alpha=1;
        rigjt3.alpha=1;
        rigjt4.alpha=1;*/


        rigjt1.x+=0.05*ix;//進入 1 影格.移動 1 像素
        rigjt1.y+=0.05*iy;

        rigjt2.x+=0.05*jx;
        rigjt2.y+=0.05*jy;
```

```
        rigjt3.x+=0.05*kx;
        rigjt3.y+=0.05*ky;

        rigjt4.x+=0.05*lx;
        rigjt4.y+=0.05*ly;

        if (rigjt1.x>=103||rigjt1.x<=100) {
                ix=- ix;
        }
        if (rigjt1.y>=-108||rigjt1.y<=-110) {
                iy=- iy;
        }
        if (rigjt2.x>=29||rigjt2.x<=27) {
                jx=- jx;
        }
        if (rigjt2.y>=-263||rigjt2.y<=-265) {
                jy=- jy;
        }
        if (rigjt3.x>=111||rigjt3.x<=108) {
                kx=- kx;
        }
        if (rigjt3.y>=-181||rigjt3.y<=-183) {
                ky=- ky;
        }
        if (rigjt4.x>=84||rigjt4.x<=82) {
                lx=- lx;
        }
        if (rigjt4.y>=-247||rigjt4.y<=-249) {
                ly=- ly;
        }
}
```

Left 物件-飛船漂浮效果

```
stop();
var ix:int=1;
var iy:int=1;
this.addEventListener(Event.ENTER_FRAME,flyb);
```

```
//這個物件.設一監聽程式(事件類型.進入影格.做"rota")
function flyb(e:Event):void {
	flyboatxl.x+=2*ix;//進入 1 影格.移動 1 像素
	flyboatxl.y+=0.5*iy;

	if (flyboatxl.x>=6||flyboatxl.x<=-6) {
		ix=- ix;
	}
	if (flyboatxl.y>=3||flyboatxl.y<=-3) {
		iy=- iy;
	}
}
```

Map 物件-載入 google map

```
import com.google.maps.Map;
import com.google.maps.LatLng;
import com.google.maps.MapEvent;
import com.google.maps.MapMouseEvent;
import com.google.maps.MapType;
import com.google.maps.overlays.Marker;
import com.google.maps.overlays.MarkerOptions;
import com.google.maps.overlays.Polyline;
import com.google.maps.overlays.PolylineOptions;
import com.google.maps.overlays.GroundOverlay;
import com.google.maps.overlays.GroundOverlayOptions;
import com.google.maps.LatLng;
import com.google.maps.LatLngBounds;
import com.google.maps.styles.FillStyle;
import com.google.maps.styles.StrokeStyle;
import com.google.maps.InfoWindowOptions;
import com.google.maps.Color ;
import com.google.maps.controls.ZoomControl;
import com.google.maps.controls.ZoomControlOptions;
import com.google.maps.controls.PositionControlOptions;
import com.google.maps.controls.PositionControl;
import com.google.maps.controls.MapTypeControl;
import com.google.maps.controls.MapTypeControlOptions;
import com.google.maps.controls.ControlPosition;
```

```
import flash.display.Loader;
import flash.display.LoaderInfo;
import flash.net.URLRequest;


var map:Map = new Map();
map.key =
"ABQIAAAAB5gTmMnSQeb_IlJwdOJkuhQevdxZWrZCE2EIyPUBGw1wKNFKlxSkAs73wWuXLQtQX_0LzfnrY
D2WeQ";
//map.addControl(new ZoomControl());
//map.addControl(new PositionControl());
map.addControl(new MapTypeControl());
map.setSize(new Point(280, 220));
map.addEventListener(MapEvent.MAP_READY, onMapReady);
this.addChild(map);

 function onMapReady(event:Event):void
{
      var topLeft:ControlPosition =    new ControlPosition(ControlPosition.ANCHOR_TOP_LEFT, 16, 10);
      var myZoomControl:ZoomControl = new ZoomControl(new ZoomControlOptions({position:
topLeft}));
       map.addControl(myZoomControl);

      map.removeMapType(MapType.PHYSICAL_MAP_TYPE);
      map.setCenter(new LatLng(22.905926,120.469207),18, MapType.NORMAL_MAP_TYPE);
      var markerA:Marker = new Marker
  (
        new LatLng(22.905926,120.469207),
        new MarkerOptions
        ({
              strokeStyle: new StrokeStyle({color: 0x000000}),
              fillStyle: new FillStyle({color: 0xFF8888, alpha:1}),
              radius: 10,
              hasShadow: true
        })
  );
    markerA.addEventListener(MapMouseEvent.CLICK, function(event:Event):void {
    markerA.openInfoWindow(new InfoWindowOptions({content:"緯度:22.905926:經度:120.469207"}));
```

```
    });
    map.addOverlay(markerA);


}
```

整顆飛船-連續旋轉

```
var i:int=20;
var j:int=1;
var a:int=40;
this.addEventListener(Event.ENTER_FRAME,rota);
     //這個物件.設一監聽程式(事件類型.進入影格.做"rota")
function rota(e:Event):void{    //
        this.rotation = this.rotation - a;
//      j++;
//      a=i*j;
}
```

彩虹按鈕物件-按鈕配色表

```
var  移到誰:int=6;
var  色表:Array=new Array();
色表=[[255,0,0],
      [255,153,0],
      [255,255,0],
      [102,255,0],
      [102,204,255],
      [51,102,204],
      [102,51,153]];
var  顯示顏色:Array=new Array();
顯示顏色=[[0,1,2,3,4,5,6,6,6,6,6,6,6],
      [1,0,1,2,3,4,5,6,6,6,6,6,6],
      [2,1,0,1,2,3,4,5,6,6,6,6,6],
      [3,2,1,0,1,2,3,4,5,6,6,6,6],
      [4,3,2,1,0,1,2,3,4,5,6,6,6],
      [5,4,3,2,1,0,1,2,3,4,5,6,6],
      [6,5,4,3,2,1,0,1,2,3,4,5,6],
      [6,6,5,4,3,2,1,0,1,2,3,4,5],
      [6,6,6,5,4,3,2,1,0,1,2,3,4],
      [6,6,6,6,5,4,3,2,1,0,1,2,3],
```

```
    [6,6,6,6,6,5,4,3,2,1,0,1,2],
    [6,6,6,6,6,6,5,4,3,2,1,0,1],
    [6,6,6,6,6,6,6,5,4,3,2,1,0]];//列:移到誰,行:自己編號
var  大家的透明度:Array=new Array();
大家的透明度=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];

//(移到誰-自己編號)的絕對值
this.addEventListener(Event.ENTER_FRAME,彈出顏色);
 function  彈出顏色(e:Event):void{
        var  彈出速度:Number=0.3;
        大家的透明度[0]=1;
        for(var i:int=1;i<=12;i++){
            if(大家的透明度[i-1]>0.7){
                大家的透明度[i]+=彈出速度;//大家的透明度[i]=彈出速度+大家的透明度[i]
                 if(大家的透明度[i]>=0.9){
                        大家的透明度[i]=1;
                        }
                }
             }
        }
```

溫度物件-滑鼠移入更改色表+滑鼠移出色表回正

```
var  自己編號:int=9;
this.addEventListener(MouseEvent.ROLL_OVER,移到誰);
this.addEventListener(MouseEvent.ROLL_OUT,移出誰);
function  移到誰(E:MouseEvent):void {
        parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
        parent['移到誰']=自己編號;
        trace(自己編號);
}
function  移出誰(E:MouseEvent):void {
        parent['移到誰']=6;
        parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
}
```

傾角物件-滑鼠移入更改色表+滑鼠移出色表回正

```
var  自己編號:int=8;
this.addEventListener(MouseEvent.ROLL_OVER,移到誰);
```

```
this.addEventListener(MouseEvent.ROLL_OUT,移出誰);
function  移到誰(E:MouseEvent):void {
        parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
        parent['移到誰']=自己編號;
        trace(自己編號);
}
function  移出誰(E:MouseEvent):void {
        parent['移到誰']=6;
        parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
}
```

轉角物件-滑鼠移入更改色表+滑鼠移出色表回正

```
var  自己編號:int=7;
this.addEventListener(MouseEvent.ROLL_OVER,移到誰);
this.addEventListener(MouseEvent.ROLL_OUT,移出誰);
function  移到誰(E:MouseEvent):void {
        parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
        parent['移到誰']=自己編號;
        trace(自己編號);
}
function  移出誰(E:MouseEvent):void {
        parent['移到誰']=6;
        parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
}
```

行走距離物件-滑鼠移入更改色表+滑鼠移出色表回正

```
var  自己編號:int=6;
this.addEventListener(MouseEvent.ROLL_OVER,移到誰);
this.addEventListener(MouseEvent.ROLL_OUT,移出誰);
function  移到誰(E:MouseEvent):void {
        parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
        parent['移到誰']=自己編號;
        trace(自己編號);
}
function  移出誰(E:MouseEvent):void {
        parent['移到誰']=6;
        parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
}
```

推力物件-滑鼠移入更改色表+滑鼠移出色表回正

```
var  自己編號:int=5;
this.addEventListener(MouseEvent.ROLL_OVER,移到誰);
this.addEventListener(MouseEvent.ROLL_OUT,移出誰);
function  移到誰(E:MouseEvent):void {
        parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
        parent['移到誰']=自己編號;
        trace(自己編號);
}
function  移出誰(E:MouseEvent):void {
        parent['移到誰']=6;
        parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
}
```

時速物件-滑鼠移入更改色表+滑鼠移出色表回正

```
var  自己編號:int=4;
this.addEventListener(MouseEvent.ROLL_OVER,移到誰);
this.addEventListener(MouseEvent.ROLL_OUT,移出誰);
function  移到誰(E:MouseEvent):void {
        parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
        parent['移到誰']=自己編號;
        trace(自己編號);
}
function  移出誰(E:MouseEvent):void {
        parent['移到誰']=6;
        parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
}
```

高度物件-滑鼠移入更改色表+滑鼠移出色表回正

```
var  自己編號:int=3;
this.addEventListener(MouseEvent.ROLL_OVER,移到誰);
this.addEventListener(MouseEvent.ROLL_OUT,移出誰);
function  移到誰(E:MouseEvent):void {
        parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
        parent['移到誰']=自己編號;
        trace(自己編號);
}
```

```
function  移出誰(E:MouseEvent):void {
        parent['移到誰']=6;
        parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
}
```

1-按照編號抓色表位置顯示顏色

```
var red:int;
var green:int;
var blue:int;
var  自己編號:int=0;
var  透明度:Number=0;
this.addEventListener(Event.ENTER_FRAME,變色);
function  變色(e:Event):void {
//進入影格後開始按照下方指令配色


        red=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][0];
        //依照老爸的[色表],以及用[移到誰(列)]和[自己編號(行)]判斷[顯示顏色]裡的位置,色表裡紅色參數的位置
        green=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][1];
        blue=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][2];


        this.transform.colorTransform=new ColorTransform(1,1,1,parent['大家的透明度'][Math.abs(parent['移到誰']-自己編號)],red,green,blue,0);
        //前三個 1 是紅綠藍的偏移量,第四個是透明度的偏移量<<這行主要再改透明度



}
```

2-按照編號抓色表位置顯示顏色

```
var red:int;
var green:int;
var blue:int;
var  自己編號:int=1;
var  透明度:Number=0;
this.addEventListener(Event.ENTER_FRAME,變色);
function  變色(e:Event):void {
```

```
        red=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][0];
        //依照老爸的[色表],以及用[移到誰(列)]和[自己編號(行)]判斷[顯示顏色]裡的位置,色表裡紅色參數的位置
        green=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][1];
        blue=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][2];

        this.transform.colorTransform=new ColorTransform(1,1,1,parent['大家的透明度'][Math.abs(parent['移到誰']-自己編號)],red,green,blue,0);//前三個 1 是紅綠藍的偏移量,第四個是透明度的偏移量


}
```

3-按照編號抓色表位置顯示顏色

```
var red:int;
var green:int;
var blue:int;
var  自己編號:int=2;
var  透明度:Number=1.0;
this.addEventListener(Event.ENTER_FRAME,變色);
function  變色(e:Event):void {


        red=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][0];
        //依照老爸的[色表],以及用[移到誰(列)]和[自己編號(行)]判斷[顯示顏色]裡的位置,色表裡紅色參數的位置
        green=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][1];
        blue=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][2];

        this.transform.colorTransform=new ColorTransform(1,1,1,parent['大家的透明度'][Math.abs(parent['移到誰']-自己編號)],red,green,blue,0);//前三個 1 是紅綠藍的偏移量,第四個是透明度的偏移量


}
```

4-按照編號抓色表位置顯示顏色

```
var red:int;
```

```
var green:int;
var blue:int;
var  自己編號:int=3;
var  透明度:Number=1.0;
this.addEventListener(Event.ENTER_FRAME,變色);
function  變色(e:Event):void {



      red=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][0];
      //依照老爸的[色表],以及用[移到誰(列)]和[自己編號(行)]判斷[顯示顏色]裡的位置,色表裡紅色參數的位置
      green=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][1];
      blue=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][2];

      this.transform.colorTransform=new ColorTransform(1,1,1,parent['大家的透明度'][Math.abs(parent['移到誰']-自己編號)],red,green,blue,0);//前三個 1 是紅綠藍的偏移量,第四個是透明度的偏移量



}
```

5-按照編號抓色表位置顯示顏色

```
var red:int;
var green:int;
var blue:int;
var  自己編號:int=4;
var  透明度:Number=1.0;
this.addEventListener(Event.ENTER_FRAME,變色);
function  變色(e:Event):void {



      red=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][0];
      //依照老爸的[色表],以及用[移到誰(列)]和[自己編號(行)]判斷[顯示顏色]裡的位置,色表裡紅色參數的位置
      green=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][1];
      blue=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][2];

      this.transform.colorTransform=new ColorTransform(1,1,1,parent['大家的透明度
```

']Math.abs(parent['移到誰']-自己編號)],red,green,blue,0);//前三個 1 是紅綠藍的偏移量,第四個是透明度的偏移量

}

6-按照編號抓色表位置顯示顏色

```
var red:int;
var green:int;
var blue:int;
var  自己編號:int=5;
var  透明度:Number=1.0;
this.addEventListener(Event.ENTER_FRAME,變色);
function  變色(e:Event):void {


      red=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][0];
      //依照老爸的[色表],以及用[移到誰(列)]和[自己編號(行)]判斷[顯示顏色]裡的位置,色表裡紅色參數的位置
      green=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][1];
      blue=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][2];

      this.transform.colorTransform=new ColorTransform(1,1,1,parent['大家的透明度']Math.abs(parent['移到誰']-自己編號)],red,green,blue,0);//前三個 1 是紅綠藍的偏移量,第四個是透明度的偏移量


}
```

7-按照編號抓色表位置顯示顏色

```
var red:int;
var green:int;
var blue:int;
var  自己編號:int=6;
var  透明度:Number=1.0;
this.addEventListener(Event.ENTER_FRAME,變色);
function  變色(e:Event):void {
```

```
        red=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][0];
        //依照老爸的[色表],以及用[移到誰(列)]和[自己編號(行)]判斷[顯示顏色]裡的位置,色表裡紅色參數的位置
        green=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][1];
        blue=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][2];

        this.transform.colorTransform=new ColorTransform(1,1,1,parent['大家的透明度'][Math.abs(parent['移到誰']-自己編號)],red,green,blue,0);//前三個 1 是紅綠藍的偏移量,第四個是透明度的偏移量


}
```

8-按照編號抓色表位置顯示顏色

```
var red:int;
var green:int;
var blue:int;
var  自己編號:int=7;
var  透明度:Number=1.0;
this.addEventListener(Event.ENTER_FRAME,變色);
function  變色(e:Event):void {


        red=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][0];
        //依照老爸的[色表],以及用[移到誰(列)]和[自己編號(行)]判斷[顯示顏色]裡的位置,色表裡紅色參數的位置
        green=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][1];
        blue=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][2];

        this.transform.colorTransform=new ColorTransform(1,1,1,parent['大家的透明度'][Math.abs(parent['移到誰']-自己編號)],red,green,blue,0);//前三個 1 是紅綠藍的偏移量,第四個是透明度的偏移量


}
```

9-按照編號抓色表位置顯示顏色

```
var red:int;
var green:int;
var blue:int;
var  自己編號:int=8;
var  透明度:Number=1.0;
this.addEventListener(Event.ENTER_FRAME,變色);
function  變色(e:Event):void {


        red=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][0];
        //依照老爸的[色表],以及用[移到誰(列)]和[自己編號(行)]判斷[顯示顏色]裡的位置,色表裡紅色參數的位置
        green=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][1];
        blue=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][2];

        this.transform.colorTransform=new ColorTransform(1,1,1,parent['大家的透明度'][Math.abs(parent['移到誰']-自己編號)],red,green,blue,0);//前三個 1 是紅綠藍的偏移量,第四個是透明度的偏移量


}
```

10-按照編號抓色表位置顯示顏色

```
var red:int;
var green:int;
var blue:int;
var  自己編號:int=9;
var  透明度:Number=1.0;
this.addEventListener(Event.ENTER_FRAME,變色);
function  變色(e:Event):void {


        red=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][0];
        //依照老爸的[色表],以及用[移到誰(列)]和[自己編號(行)]判斷[顯示顏色]裡的位置,色表裡紅色參數的位置
        green=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][1];
        blue=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][2];
```

```
        this.transform.colorTransform=new ColorTransform(1,1,1,parent['大家的透明度
'][Math.abs(parent['移到誰']-自己編號)],red,green,blue,0);//前三個 1 是紅綠藍的偏移量,第四個是透
明度的偏移量



}
```

11-按照編號抓色表位置顯示顏色

```
var red:int;
var green:int;
var blue:int;
var  自己編號:int=10;
var  透明度:Number=1.0;
this.addEventListener(Event.ENTER_FRAME,變色);
function  變色(e:Event):void {



        red=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][0];
        //依照老爸的[色表],以及用[移到誰(列)]和[自己編號(行)]判斷[顯示顏色]裡的位置,色表裡紅色
參數的位置
        green=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][1];
        blue=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][2];

        this.transform.colorTransform=new ColorTransform(1,1,1,parent['大家的透明度
'][Math.abs(parent['移到誰']-自己編號)],red,green,blue,0);//前三個 1 是紅綠藍的偏移量,第四個是透
明度的偏移量



}
```

12-按照編號抓色表位置顯示顏色

```
var red:int;
var green:int;
var blue:int;
var  自己編號:int=11;
var  透明度:Number=1.0;
this.addEventListener(Event.ENTER_FRAME,變色);
function  變色(e:Event):void {
```

```
        red=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][0];
        //依照老爸的[色表],以及用[移到誰(列)]和[自己編號(行)]判斷[顯示顏色]裡的位置,色表裡紅色參數的位置
        green=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][1];
        blue=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][2];

        this.transform.colorTransform=new ColorTransform(1,1,1,parent['大家的透明度'][Math.abs(parent['移到誰']-自己編號)],red,green,blue,0);//前三個 1 是紅綠藍的偏移量,第四個是透明度的偏移量


}
```

13-按照編號抓色表位置顯示顏色

```
var red:int;
var green:int;
var blue:int;
var  自己編號:int=12;
var  透明度:Number=1.0;
this.addEventListener(Event.ENTER_FRAME,變色);
function  變色(e:Event):void {


        red=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][0];
        //依照老爸的[色表],以及用[移到誰(列)]和[自己編號(行)]判斷[顯示顏色]裡的位置,色表裡紅色參數的位置
        green=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][1];
        blue=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][2];

        this.transform.colorTransform=new ColorTransform(1,1,1,parent['大家的透明度'][Math.abs(parent['移到誰']-自己編號)],red,green,blue,0);//前三個 1 是紅綠藍的偏移量,第四個是透明度的偏移量


}
```

按鈕效果 4、按鈕效果 5、按鈕效果 6、按鈕效果 7、按鈕效果 8、按鈕效果 9、按鈕效果 10-滑鼠按下發亮，移入移出放開則還原

```
var red:int;
var green:int;
var blue:int;
var  自己編號:int=12;
var  透明度:Number=1.0;
this.addEventListener(Event.ENTER_FRAME,變色);
function  變色(e:Event):void {



      red=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][0];
      //依照老爸的[色表],以及用[移到誰(列)]和[自己編號(行)]判斷[顯示顏色]裡的位置,色表裡紅色參數的位置
      green=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][1];
      blue=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][2];

      this.transform.colorTransform=new ColorTransform(1,1,1,parent['大家的透明度'][Math.abs(parent['移到誰']-自己編號)],red,green,blue,0);//前三個 1 是紅綠藍的偏移量,第四個是透明度的偏移量



}
```

版本 4(版本 5)

雲左物件-漂浮效果

```
stop();
var ix:int=1;
var iy:int=1;
var jx:int=-1;
var jy:int=1;
var kx:int=1;
var ky:int=-1;
var lx:int=-1;
var ly:int=-1;
this.addEventListener(Event.ENTER_FRAME,flyyy);
//這個物件.設一監聽程式(事件類型.進入影格.做"rota")
function flyyy(e:Event):void {
```

```
/*left1.alpha=1;
left2.alpha=1;
left3.alpha=1;
left4.alpha=1;*/


left1.x+=0.05*ix;//進入 1 影格.移動 1 像素
left1.y+=0.05*iy;

left2.x+=0.05*jx;
left2.y+=0.05*jy;

left3.x+=0.05*kx;
left3.y+=0.05*ky;

left4.x+=0.05*lx;
left4.y+=0.05*ly;

if (left1.x>=-25||left1.x<=-27) {
	ix=- ix;
}
if (left1.y>=-244||left1.y<=-246) {
	iy=- iy;
}
if (left2.x>=7||left2.x<=5) {
	jx=- jx;
}
if (left2.y>=-259||left2.y<=-261) {
	jy=- jy;
}
if (left3.x>=-53||left3.x<=-55) {
	kx=- kx;
}
if (left3.y>=-76||left3.y<=-78) {
	ky=- ky;
}
if (left4.x>=-13||left4.x<=-15) {
	lx=- lx;
```

```
	}
	if (left4.y>=-162||left4.y<=-164) {
		ly=- ly;
	}
}
```

雲右物件-漂浮效果

```
stop();
var ix:int=1;
var iy:int=1;
var jx:int=-1;
var jy:int=1;
var kx:int=1;
var ky:int=-1;
var lx:int=-1;
var ly:int=-1;
this.addEventListener(Event.ENTER_FRAME,flyy);
//這個物件.設一監聽程式(事件類型.進入影格.做"rota")
function flyy(e:Event):void {
	/*rigjt1.alpha=1;
	rigjt2.alpha=1;
	rigjt3.alpha=1;
	rigjt4.alpha=1;*/


	rigjt1.x+=0.05*ix;//進入 1 影格.移動 1 像素
	rigjt1.y+=0.05*iy;

	rigjt2.x+=0.05*jx;
	rigjt2.y+=0.05*jy;

	rigjt3.x+=0.05*kx;
	rigjt3.y+=0.05*ky;

	rigjt4.x+=0.05*lx;
	rigjt4.y+=0.05*ly;

	if (rigjt1.x>=103||rigjt1.x<=100) {
```

```
            ix=- ix;
        }
        if (rigjt1.y>=-108||rigjt1.y<=-110) {
            iy=- iy;
        }
        if (rigjt2.x>=29||rigjt2.x<=27) {
            jx=- jx;
        }
        if (rigjt2.y>=-263||rigjt2.y<=-265) {
            jy=- jy;
        }
        if (rigjt3.x>=111||rigjt3.x<=108) {
            kx=- kx;
        }
        if (rigjt3.y>=-181||rigjt3.y<=-183) {
            ky=- ky;
        }
        if (rigjt4.x>=84||rigjt4.x<=82) {
            lx=- lx;
        }
        if (rigjt4.y>=-247||rigjt4.y<=-249) {
            ly=- ly;
        }
}
```

Left 物件-飛船漂浮效果

```
stop();
var ix:int=1;
var iy:int=1;
this.addEventListener(Event.ENTER_FRAME,flyb);
//這個物件.設一監聽程式(事件類型.進入影格.做"rota")
function flyb(e:Event):void {
        flyboatxl.x+=2*ix;//進入 1 影格.移動 1 像素
        flyboatxl.y+=0.5*iy;

        if (flyboatxl.x>=6||flyboatxl.x<=-6) {
            ix=- ix;
        }
```

```
        if (flyboatxl.y>=3||flyboatxl.y<=-3) {
                iy=- iy;
        }
}
```

Map 物件-載入 google map

```
import com.google.maps.Map;
import com.google.maps.LatLng;
import com.google.maps.MapEvent;
import com.google.maps.MapMouseEvent;
import com.google.maps.MapType;
import com.google.maps.overlays.Marker;
import com.google.maps.overlays.MarkerOptions;
import com.google.maps.overlays.Polyline;
import com.google.maps.overlays.PolylineOptions;
import com.google.maps.overlays.GroundOverlay;
import com.google.maps.overlays.GroundOverlayOptions;
import com.google.maps.LatLng;
import com.google.maps.LatLngBounds;
import com.google.maps.styles.FillStyle;
import com.google.maps.styles.StrokeStyle;
import com.google.maps.InfoWindowOptions;
import com.google.maps.Color ;
import com.google.maps.controls.ZoomControl;
import com.google.maps.controls.ZoomControlOptions;
import com.google.maps.controls.PositionControlOptions;
import com.google.maps.controls.PositionControl;
import com.google.maps.controls.MapTypeControl;
import com.google.maps.controls.MapTypeControlOptions;
import com.google.maps.controls.ControlPosition;
import flash.display.Loader;
import flash.display.LoaderInfo;
import flash.net.URLRequest;



var map:Map = new Map();
map.key =
"ABQIAAAAB5gTmMnSQeb_IlJwdOJkuhQevdxZWrZCE2EIyPUBGw1wKNFKlxSkAs73wWuXLQtQX_0LzfnrY
```

```
D2WeQ";
//map.addControl(new ZoomControl());
//map.addControl(new PositionControl());
map.addControl(new MapTypeControl());
map.setSize(new Point(280, 220));
map.addEventListener(MapEvent.MAP_READY, onMapReady);
this.addChild(map);

 function onMapReady(event:Event):void
{
      var topLeft:ControlPosition =    new ControlPosition(ControlPosition.ANCHOR_TOP_LEFT, 16, 10);
      var myZoomControl:ZoomControl = new ZoomControl(new ZoomControlOptions({position:
topLeft}));
       map.addControl(myZoomControl);

      map.removeMapType(MapType.PHYSICAL_MAP_TYPE);
      map.setCenter(new LatLng(22.905926,120.469207),18, MapType.NORMAL_MAP_TYPE);
      var markerA:Marker = new Marker
  (
        new LatLng(22.905926,120.469207),
        new MarkerOptions
        ({
              strokeStyle: new StrokeStyle({color: 0x000000}),
              fillStyle: new FillStyle({color: 0xFF8888, alpha:1}),
              radius: 10,
              hasShadow: true
        })
  );
    markerA.addEventListener(MapMouseEvent.CLICK, function(event:Event):void {
    markerA.openInfoWindow(new InfoWindowOptions({content:"緯度:22.905926:經度:120.469207"}));
  });
  map.addOverlay(markerA);

}
```

整顆飛船-連續旋轉

```
var i:int=20;
var j:int=1;
```

```
var a:int=40;
this.addEventListener(Event.ENTER_FRAME,rota);
    //這個物件.設一監聽程式(事件類型.進入影格.做"rota")
function rota(e:Event):void{    //
        this.rotation = this.rotation - a;
//      j++;
//      a=i*j;
}
```

近景按鈕-按照配色表，滑鼠移入移出之對應物件發光效果，點擊滑鼠之對應物件消失效果

```
var  移到誰:int=6;
var  色表:Array=new Array();
色表=[[255,0,0],
    [255,153,0],
    [255,255,0],
    [102,255,0],
    [102,204,255],
    [51,102,204],
    [102,51,153]];
var  顯示顏色:Array=new Array();
顯示顏色=[[0,1,2,3,4,5,6,6,6,6,6,6,6],
    [1,0,1,2,3,4,5,6,6,6,6,6,6],
    [2,1,0,1,2,3,4,5,6,6,6,6,6],
    [3,2,1,0,1,2,3,4,5,6,6,6,6],
    [4,3,2,1,0,1,2,3,4,5,6,6,6],
    [5,4,3,2,1,0,1,2,3,4,5,6,6],
    [6,5,4,3,2,1,0,1,2,3,4,5,6],
    [6,6,5,4,3,2,1,0,1,2,3,4,5],
    [6,6,6,5,4,3,2,1,0,1,2,3,4],
    [6,6,6,6,5,4,3,2,1,0,1,2,3],
    [6,6,6,6,6,5,4,3,2,1,0,1,2],
    [6,6,6,6,6,6,5,4,3,2,1,0,1],
    [6,6,6,6,6,6,6,5,4,3,2,1,0]];//列:移到誰,行:自己編號
var  大家的透明度:Array=new Array();
大家的透明度=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];

//(移到誰-自己編號)的絕對值
this.addEventListener(Event.ENTER_FRAME,彈出顏色);
```

```
function  彈出顏色(e:Event):void {
	var  彈出速度:Number=0.3;
	大家的透明度[0]=1;
	for (var i:int=1; i<=12; i++) {
		if (大家的透明度[i-1]>0.7) {
			大家的透明度[i]+=彈出速度;//大家的透明度[i]=彈出速度+大家的透明度[i]
			if (大家的透明度[i]>=0.9) {
				大家的透明度[i]=1;
			}
		}
	}
}
var  物件名字陣列:Array=new Array();
//物件名字陣列=[物件名稱,沒有亮的影格,有亮的影格,物件不見的影格,說明文字]
物件名字陣列=[['物件名稱','沒有亮的影格','有亮的影格','物件不見的影格','說明文字'],
  ['物件名稱','沒有亮的影格','有亮的影格','物件不見的影格','說明文字'],
  ['物件名稱','沒有亮的影格','有亮的影格','物件不見的影格','說明文字'],
  ['高度','沒有亮的影格','有亮的影格','物件不見的影格','說明文字'],
  ['時速','沒有亮的影格','有亮的影格','物件不見的影格','說明文字'],
    ['推力',1,2,3,'推力說明文字'],
  ['行走距離','沒有亮的影格','有亮的影格','物件不見的影格','說明文字'],
  ['轉角',1,2,3,'轉角說明文字'],
  ['傾角',1,2,3,'傾角說明文字'],
  ['溫度',1,2,3,'溫度說明文字']];
var  滑鼠移到彩虹裡:int=0;
var loop:int=4;
var  推力:int=0;
var  轉角:int=0;
var  傾角:int=0;
var  溫度:int=0;
this.addEventListener(MouseEvent.MOUSE_OVER,滑鼠移到彩虹上);
function  滑鼠移到彩虹上(e:MouseEvent):void {
	滑鼠移到彩虹裡=1;
	for (loop=5; loop<=9; loop++) {
		if (loop==移到誰) {
			if (移到誰==5&&推力==0) {
				this.parent.parent['main'].left_.推力.gotoAndStop(物件名字陣列[移到誰][2]);
			} else if (移到誰==7&&轉角==0) {
```

```
                    this.parent.parent['main'].left_.轉角.gotoAndStop(物件名字陣列[移到誰][2]);
                } else if (移到誰==8&&傾角==0) {
                    this.parent.parent['main'].left_.傾角.gotoAndStop(物件名字陣列[移到誰][2]);
                } else if (移到誰==9&&溫度==0) {
                    this.parent.parent['main'].left_.溫度.gotoAndStop(物件名字陣列[移到誰][2]);
                }
                trace(物件名字陣列[loop][4]);
                trace(物件名字陣列[loop][0]);
                trace(物件名字陣列[loop][2]);
            }
        }
}
this.addEventListener(MouseEvent.MOUSE_OUT,滑鼠移開彩虹上);
function  滑鼠移開彩虹上(e:MouseEvent):void {
        滑鼠移到彩虹裡=0;
        for (loop=5; loop<=9; loop++) {
            if (loop==移到誰) {
                if (移到誰==5&&推力==0) {
                    this.parent.parent['main'].left_.推力.gotoAndStop(物件名字陣列[移到誰][1]);
                } else if (移到誰==7&&轉角==0) {
                    this.parent.parent['main'].left_.轉角.gotoAndStop(物件名字陣列[移到誰][1]);
                } else if (移到誰==8&&傾角==0) {
                    this.parent.parent['main'].left_.傾角.gotoAndStop(物件名字陣列[移到誰][1]);
                } else if (移到誰==9&&溫度==0) {
                    this.parent.parent['main'].left_.溫度.gotoAndStop(物件名字陣列[移到誰][1]);
                }
                trace(物件名字陣列[loop][4]);
                trace(物件名字陣列[loop][0]);
                trace(物件名字陣列[loop][2]);
            }
        }
}
this.addEventListener(MouseEvent.MOUSE_DOWN,點擊彩虹按鈕);
function  點擊彩虹按鈕(e:MouseEvent):void {
        for (loop=5; loop<=9; loop++) {
            if (loop==移到誰) {
                if (移到誰==5) {
                    if (推力==0) {
```

```
                        this.parent.parent['main'].left_.推力.gotoAndStop(物件名字陣列[移到
誰][3]);
                        推力=1;
                    } else if (推力==1) {
                        this.parent.parent['main'].left_.推力.gotoAndStop(物件名字陣列[移到
誰][1]);
                        推力=0;
                    }
                } else if (移到誰==7) {
                    if (轉角==0) {
                        this.parent.parent['main'].left_.轉角.gotoAndStop(物件名字陣列[移到
誰][3]);
                        轉角=1;
                    } else if (轉角==1) {
                        this.parent.parent['main'].left_.轉角.gotoAndStop(物件名字陣列[移到
誰][1]);
                        轉角=0;
                    }
                } else if (移到誰==8) {
                    if (傾角==0) {
                        this.parent.parent['main'].left_.傾角.gotoAndStop(物件名字陣列[移到
誰][3]);
                        傾角=1;
                    } else if (傾角==1) {
                        this.parent.parent['main'].left_.傾角.gotoAndStop(物件名字陣列[移到
誰][1]);
                        傾角=0;
                    }
                } else if (移到誰==9) {
                    if (溫度==0) {
                        this.parent.parent['main'].left_.溫度.gotoAndStop(物件名字陣列[移到
誰][3]);
                        溫度=1;
                    } else if (溫度==1) {
                        this.parent.parent['main'].left_.溫度.gotoAndStop(物件名字陣列[移到
誰][1]);
                        溫度=0;
                    }
```

```
                    trace(物件名字陣列[loop][4]);
                    trace(物件名字陣列[loop][0]);
                    trace(物件名字陣列[loop][2]);
                }
            }
        }
}
```

遠景按鈕-按照配色表，滑鼠移入移出之對應物件發光效果，點擊滑鼠之對應物件消失效果

```
var  移到誰:int=6;
var  色表:Array=new Array();
色表=[[255,0,0],
    [255,153,0],
    [255,255,0],
    [102,255,0],
    [102,204,255],
    [51,102,204],
    [102,51,153]];
var  顯示顏色:Array=new Array();
顯示顏色=[[0,1,2,3,4,5,6,6,6,6,6,6,6],
    [1,0,1,2,3,4,5,6,6,6,6,6,6],
    [2,1,0,1,2,3,4,5,6,6,6,6,6],
    [3,2,1,0,1,2,3,4,5,6,6,6,6],
    [4,3,2,1,0,1,2,3,4,5,6,6,6],
    [5,4,3,2,1,0,1,2,3,4,5,6,6],
    [6,5,4,3,2,1,0,1,2,3,4,5,6],
    [6,6,5,4,3,2,1,0,1,2,3,4,5],
    [6,6,6,5,4,3,2,1,0,1,2,3,4],
    [6,6,6,6,5,4,3,2,1,0,1,2,3],
    [6,6,6,6,6,5,4,3,2,1,0,1,2],
    [6,6,6,6,6,6,5,4,3,2,1,0,1],
    [6,6,6,6,6,6,6,5,4,3,2,1,0]];//列:移到誰,行:自己編號
var  大家的透明度:Array=new Array();
大家的透明度=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];

//(移到誰-自己編號)的絕對值
this.addEventListener(Event.ENTER_FRAME,彈出顏色);
function  彈出顏色(e:Event):void {
```

```
        var  彈出速度:Number=0.3;
        大家的透明度[0]=1;
        for (var i:int=1; i<=12; i++) {
            if (大家的透明度[i-1]>0.7) {
                大家的透明度[i]+=彈出速度;//大家的透明度[i]=彈出速度+大家的透明度[i]
            }
        }
}
```

中景按鈕-按照配色表，滑鼠移入移出之對應物件發光效果，點擊滑鼠之對應物件消失效果

```
var  移到誰:int=6;
var  色表:Array=new Array();
色表=[[255,0,0],
    [255,153,0],
    [255,255,0],
    [102,255,0],
    [102,204,255],
    [51,102,204],
    [102,51,153]];
var  顯示顏色:Array=new Array();
顯示顏色=[[0,1,2,3,4,5,6,6,6,6,6,6,6],
    [1,0,1,2,3,4,5,6,6,6,6,6,6],
    [2,1,0,1,2,3,4,5,6,6,6,6,6],
    [3,2,1,0,1,2,3,4,5,6,6,6,6],
    [4,3,2,1,0,1,2,3,4,5,6,6,6],
    [5,4,3,2,1,0,1,2,3,4,5,6,6],
    [6,5,4,3,2,1,0,1,2,3,4,5,6],
    [6,6,5,4,3,2,1,0,1,2,3,4,5],
    [6,6,6,5,4,3,2,1,0,1,2,3,4],
    [6,6,6,6,5,4,3,2,1,0,1,2,3],
    [6,6,6,6,6,5,4,3,2,1,0,1,2],
    [6,6,6,6,6,6,5,4,3,2,1,0,1],
    [6,6,6,6,6,6,6,5,4,3,2,1,0]];//列:移到誰,行:自己編號
var  大家的透明度:Array=new Array();
大家的透明度=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];

//(移到誰-自己編號)的絕對值
this.addEventListener(Event.ENTER_FRAME,彈出顏色);
```

```
function  彈出顏色(e:Event):void {
	var  彈出速度:Number=0.3;
	大家的透明度[0]=1;
	for (var i:int=1; i<=12; i++) {
		if (大家的透明度[i-1]>0.7) {
			大家的透明度[i]+=彈出速度;//大家的透明度[i]=彈出速度+大家的透明度[i]
			if (大家的透明度[i]>=0.9) {
				大家的透明度[i]=1;
			}
		}
	}
}
var  物件名字陣列:Array=new Array();
//物件名字陣列=[物件名稱,沒有亮的影格,有亮的影格,物件不見的影格,說明文字]
物件名字陣列=[['物件名稱','沒有亮的影格','有亮的影格','物件不見的影格','說明文字'],
  ['物件名稱','沒有亮的影格','有亮的影格','物件不見的影格','說明文字'],
  ['物件名稱','沒有亮的影格','有亮的影格','物件不見的影格','說明文字'],
  ['高度','沒有亮的影格','有亮的影格','物件不見的影格','說明文字'],
  ['時速','沒有亮的影格','有亮的影格','物件不見的影格','說明文字'],
    ['推力',1,2,3,'推力說明文字'],
  ['行走距離','沒有亮的影格','有亮的影格','物件不見的影格','說明文字'],
  ['轉角',1,2,3,'轉角說明文字'],
  ['傾角',1,2,3,'傾角說明文字'],
  ['溫度',1,2,3,'溫度說明文字']];
var  滑鼠移到彩虹裡:int=0;
var loop:int=4;
var  推力:int=0;
var  轉角:int=0;
var  傾角:int=0;
var  溫度:int=0;
this.addEventListener(MouseEvent.MOUSE_OVER,滑鼠移到彩虹上);
function  滑鼠移到彩虹上(e:MouseEvent):void {
	滑鼠移到彩虹裡=1;
	for (loop=5; loop<=9; loop++) {
		if (loop==移到誰) {
			if (移到誰==5&&推力==0) {
				this.parent.parent['main'].left_.推力.gotoAndStop(物件名字陣列[移到誰][2]);
			} else if (移到誰==7&&轉角==0) {
```

```
                    this.parent.parent['main'].left_.轉角.gotoAndStop(物件名字陣列[移到誰][2]);
                } else if (移到誰==8&&傾角==0) {
                    this.parent.parent['main'].left_.傾角.gotoAndStop(物件名字陣列[移到誰][2]);
                } else if (移到誰==9&&溫度==0) {
                    this.parent.parent['main'].left_.溫度.gotoAndStop(物件名字陣列[移到誰][2]);
                }
                trace(物件名字陣列[loop][4]);
                trace(物件名字陣列[loop][0]);
                trace(物件名字陣列[loop][2]);
            }
        }
}
this.addEventListener(MouseEvent.MOUSE_OUT,滑鼠移開彩虹上);
function  滑鼠移開彩虹上(e:MouseEvent):void {
        滑鼠移到彩虹裡=0;
        for (loop=5; loop<=9; loop++) {
            if (loop==移到誰) {
                if (移到誰==5&&推力==0) {
                    this.parent.parent['main'].left_.推力.gotoAndStop(物件名字陣列[移到誰][1]);
                } else if (移到誰==7&&轉角==0) {
                    this.parent.parent['main'].left_.轉角.gotoAndStop(物件名字陣列[移到誰][1]);
                } else if (移到誰==8&&傾角==0) {
                    this.parent.parent['main'].left_.傾角.gotoAndStop(物件名字陣列[移到誰][1]);
                } else if (移到誰==9&&溫度==0) {
                    this.parent.parent['main'].left_.溫度.gotoAndStop(物件名字陣列[移到誰][1]);
                }
                trace(物件名字陣列[loop][4]);
                trace(物件名字陣列[loop][0]);
                trace(物件名字陣列[loop][2]);
            }
        }
}
this.addEventListener(MouseEvent.MOUSE_DOWN,點擊彩虹按鈕);
function  點擊彩虹按鈕(e:MouseEvent):void {
        for (loop=5; loop<=9; loop++) {
            if (loop==移到誰) {
                if (移到誰==5) {
                    if (推力==0) {
```

```
                        this.parent.parent['main'].left_.推力.gotoAndStop(物件名字陣列[移到
誰][3]);
                        推力=1;
                    } else if (推力==1) {
                        this.parent.parent['main'].left_.推力.gotoAndStop(物件名字陣列[移到
誰][1]);
                        推力=0;
                    }
                } else if (移到誰==7) {
                    if (轉角==0) {
                        this.parent.parent['main'].left_.轉角.gotoAndStop(物件名字陣列[移到
誰][3]);
                        轉角=1;
                    } else if (轉角==1) {
                        this.parent.parent['main'].left_.轉角.gotoAndStop(物件名字陣列[移到
誰][1]);
                        轉角=0;
                    }
                } else if (移到誰==8) {
                    if (傾角==0) {
                        this.parent.parent['main'].left_.傾角.gotoAndStop(物件名字陣列[移到
誰][3]);
                        傾角=1;
                    } else if (傾角==1) {
                        this.parent.parent['main'].left_.傾角.gotoAndStop(物件名字陣列[移到
誰][1]);
                        傾角=0;
                    }
                } else if (移到誰==9) {
                    if (溫度==0) {
                        this.parent.parent['main'].left_.溫度.gotoAndStop(物件名字陣列[移到
誰][3]);
                        溫度=1;
                    } else if (溫度==1) {
                        this.parent.parent['main'].left_.溫度.gotoAndStop(物件名字陣列[移到
誰][1]);
                        溫度=0;
                    }
```

```
                    trace(物件名字陣列[loop][4]);
                    trace(物件名字陣列[loop][0]);
                    trace(物件名字陣列[loop][2]);
                }
            }
        }
}
```

高度-滑鼠移入更改色表+滑鼠移出色表回正

```
var 自己編號:int=7;
this.addEventListener(MouseEvent.ROLL_OVER,移到誰);
this.addEventListener(MouseEvent.ROLL_OUT,移出誰);
function 移到誰(E:MouseEvent):void {
    parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
    parent['移到誰']=自己編號;
    trace(自己編號);
}
function 移出誰(E:MouseEvent):void {
    parent['移到誰']=6;
    parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
}
```

飛行軌跡-滑鼠移入更改色表+滑鼠移出色表回正

```
var 自己編號:int=6;
this.addEventListener(MouseEvent.ROLL_OVER,移到誰);
this.addEventListener(MouseEvent.ROLL_OUT,移出誰);
function 移到誰(E:MouseEvent):void {
    parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
    parent['移到誰']=自己編號;
    trace(自己編號);
}
function 移出誰(E:MouseEvent):void {
    parent['移到誰']=6;
    parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
}
```

位置-滑鼠移入更改色表+滑鼠移出色表回正

```
var 自己編號:int=5;
```

```
this.addEventListener(MouseEvent.ROLL_OVER,移到誰);
this.addEventListener(MouseEvent.ROLL_OUT,移出誰);
function  移到誰(E:MouseEvent):void {
        parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
        parent['移到誰']=自己編號;
        trace(自己編號);
}
function  移出誰(E:MouseEvent):void {
        parent['移到誰']=6;
        parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
}
```

溫度-滑鼠移入更改色表+滑鼠移出色表回正

```
var  自己編號:int=9;
this.addEventListener(MouseEvent.ROLL_OVER,移到誰);
this.addEventListener(MouseEvent.ROLL_OUT,移出誰);
function  移到誰(E:MouseEvent):void {
        parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
        parent['移到誰']=自己編號;
        trace(自己編號);
}
function  移出誰(E:MouseEvent):void {
        parent['移到誰']=6;
        parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
}
```

傾角-滑鼠移入更改色表+滑鼠移出色表回正

```
var  自己編號:int=8;
this.addEventListener(MouseEvent.ROLL_OVER,移到誰);
this.addEventListener(MouseEvent.ROLL_OUT,移出誰);
function  移到誰(E:MouseEvent):void {
        parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
        parent['移到誰']=自己編號;
        trace(自己編號);
}
function  移出誰(E:MouseEvent):void {
        parent['移到誰']=6;
        parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
```

```
}
```

轉角-滑鼠移入更改色表+滑鼠移出色表回正

```
v var  自己編號:int=7;
this.addEventListener(MouseEvent.ROLL_OVER,移到誰);
this.addEventListener(MouseEvent.ROLL_OUT,移出誰);
function  移到誰(E:MouseEvent):void {
        parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
        parent['移到誰']=自己編號;
        trace(自己編號);
}
function  移出誰(E:MouseEvent):void {
        parent['移到誰']=6;
        parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
}
```

行走距離-滑鼠移入更改色表+滑鼠移出色表回正

```
var  自己編號:int=6;
this.addEventListener(MouseEvent.ROLL_OVER,移到誰);
this.addEventListener(MouseEvent.ROLL_OUT,移出誰);
function  移到誰(E:MouseEvent):void {
        parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
        parent['移到誰']=自己編號;
        trace(自己編號);
}
function  移出誰(E:MouseEvent):void {
        parent['移到誰']=6;
        parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
}
```

推力-滑鼠移入更改色表+滑鼠移出色表回正

```
var  自己編號:int=5;
this.addEventListener(MouseEvent.ROLL_OVER,移到誰);
this.addEventListener(MouseEvent.ROLL_OUT,移出誰);
function  移到誰(E:MouseEvent):void {
        parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
        parent['移到誰']=自己編號;
        trace(自己編號);
```

```
}
function  移出誰(E:MouseEvent):void {
        parent['移到誰']=6;
        parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
}


}
```

時速-滑鼠移入更改色表+滑鼠移出色表回正

```
var  自己編號:int=4;
this.addEventListener(MouseEvent.ROLL_OVER,移到誰);
this.addEventListener(MouseEvent.ROLL_OUT,移出誰);
function  移到誰(E:MouseEvent):void {
        parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
        parent['移到誰']=自己編號;
        trace(自己編號);
}
function  移出誰(E:MouseEvent):void {
        parent['移到誰']=6;
        parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
}
```

高度-滑鼠移入更改色表+滑鼠移出色表回正

```
var  自己編號:int=3;
this.addEventListener(MouseEvent.ROLL_OVER,移到誰);
this.addEventListener(MouseEvent.ROLL_OUT,移出誰);
function  移到誰(E:MouseEvent):void {
        parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
        parent['移到誰']=自己編號;
        trace(自己編號);
}
function  移出誰(E:MouseEvent):void {
        parent['移到誰']=6;
        parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
}
```

訊號比-滑鼠移入更改色表+滑鼠移出色表回正

```
var  自己編號:int=7;
```

```
this.addEventListener(MouseEvent.ROLL_OVER,移到誰);
this.addEventListener(MouseEvent.ROLL_OUT,移出誰);
function  移到誰(E:MouseEvent):void {
        parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
        parent['移到誰']=自己編號;
        trace(自己編號);
}
function  移出誰(E:MouseEvent):void {
        parent['移到誰']=6;
        parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
}
```

控制系統-滑鼠移入更改色表+滑鼠移出色表回正

```
var  自己編號:int=6;
this.addEventListener(MouseEvent.ROLL_OVER,移到誰);
this.addEventListener(MouseEvent.ROLL_OUT,移出誰);
function  移到誰(E:MouseEvent):void {
        parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
        parent['移到誰']=自己編號;
        parent['真的移入誰']=1;
        trace(自己編號);
}
function  移出誰(E:MouseEvent):void {
        parent['移到誰']=6;
        parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
}
```

指北針-滑鼠移入更改色表+滑鼠移出色表回正

```
var  自己編號:int=5;
this.addEventListener(MouseEvent.ROLL_OVER,移到誰);
this.addEventListener(MouseEvent.ROLL_OUT,移出誰);
function  移到誰(E:MouseEvent):void {
        parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
        parent['移到誰']=自己編號;

        trace(自己編號);
}
function  移出誰(E:MouseEvent):void {
```

```
        parent['移到誰']=6;
        parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
}']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
}
```

方向角-滑鼠移入更改色表+滑鼠移出色表回正

```
var  自己編號:int=4;
this.addEventListener(MouseEvent.ROLL_OVER,移到誰);
this.addEventListener(MouseEvent.ROLL_OUT,移出誰);
function  移到誰(E:MouseEvent):void {
        parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
        parent['移到誰']=自己編號;
        trace(自己編號);
}
function  移出誰(E:MouseEvent):void {
        parent['移到誰']=6;
        parent['大家的透明度']=[0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3];
}
```

按鈕效果 5、按鈕效果 6、按鈕效果 7、按鈕效果 8、按鈕效果 9-滑鼠按下發亮，移入移出放開則還原

```
var red:int;
var green:int;
var blue:int;
var  自己編號:int=12;
var  透明度:Number=1.0;
this.addEventListener(Event.ENTER_FRAME,變色);
function  變色(e:Event):void {



        red=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][0];
        //依照老爸的[色表],以及用[移到誰(列)]和[自己編號(行)]判斷[顯示顏色]裡的位置,色表裡紅色參數的位置
        green=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][1];
        blue=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][2];


        this.transform.colorTransform=new ColorTransform(1,1,1,parent['大家的透明度
'][Math.abs(parent['移到誰']-自己編號)],red,green,blue,0);//前三個 1 是紅綠藍的偏移量,第四個是透
```

明度的偏移量

}

近中遠按鈕物件-滑鼠移入宜出效果，滑鼠點擊場景變換

```
var i:int=0;
var j:int=1;
var k:int=0;
var main:int=0;//預設值為中景
var d:int=30;
var m:int=10;
var u:int=20;
var all_rb_bt:int=0;
var rinbow_f:int=0;
var goto:int=0;
mid.gotoAndPlay(3);
parent['all_rinbow_bt'].gotoAndPlay(2);
up.addEventListener(MouseEvent.ROLL_OVER,移入 up);
up.addEventListener(MouseEvent.ROLL_OUT,移出 up);
up.addEventListener(MouseEvent.CLICK,點擊 up);
function  移入 up(e:MouseEvent):void {
	if (i!=1) {
		up.gotoAndStop(2);
	}
}
function  移出 up(e:MouseEvent):void {
	if (i!=1) {
		up.gotoAndStop(1);
	}
}
function  點擊 up(e:MouseEvent):void {
	if (i!=1) {//按扭轉換判斷
		i=1;
		up.gotoAndPlay(3);
		if (j==1) {
			mid.gotoAndPlay(28);
			j=0;
```

```
            }
            if (k==1) {
                down.gotoAndPlay(28);
                k=0;
            }
        }
        if (main==0) {//場景變換判斷
            parent['main'].gotoAndPlay(2);
            main=1;//畫面移到近景
        } else if (main==-1) {//如果畫面在遠景
            parent['main'].addEventListener(Event.ENTER_FRAME,goback);
            function goback(e:Event):void {
                if (u>=11) {
                    parent['main'].prevFrame();
                    u--;
                } else if (u==10) {
                    parent['main'].removeEventListener(Event.ENTER_FRAME,goback);
                    u=20;
                }
            }
            main=1;
        } else if (main==1) {
            parent['main'].gotoAndStop(10);
            main=1;
        }
        if (all_rb_bt==1) {//如果彩虹鈕在近景
            parent['all_rinbow_bt'].gotoAndStop(31);
            goto=1;
        } else if (all_rb_bt==0) {//如果彩虹鈕在中景
            parent['all_rinbow_bt'].gotoAndPlay(12);
            goto=1;
        } else if (all_rb_bt==-1) {//如果彩虹鈕在遠景
            parent['all_rinbow_bt'].gotoAndPlay(52);
            goto=1;
        }
}//function 的
mid.addEventListener(MouseEvent.ROLL_OVER,移入 mid);
mid.addEventListener(MouseEvent.ROLL_OUT,移出 mid);
```

```
mid.addEventListener(MouseEvent.CLICK,點擊 mid);
function  移入 mid(e:MouseEvent):void {
	if (j!=1) {
		mid.gotoAndStop(2);
	}
}
function  移出 mid(e:MouseEvent):void {
	if (j!=1) {
		mid.gotoAndStop(1);
	}
}
function  點擊 mid(e:MouseEvent):void {
	if (j!=1) {
		j=1;
		mid.gotoAndPlay(3);
		if (i==1) {
			up.gotoAndPlay(28);
			i=0;
		}
		if (k==1) {
			down.gotoAndPlay(28);
			k=0;
		}
	}
	if (main==1) {//如果畫面在近景
		parent['main'].addEventListener(Event.ENTER_FRAME,goback);
		function goback(e:Event):void {
			if (m>=2) {
				parent['main'].prevFrame();
				m--;
			} else if (m==1) {
				parent['main'].removeEventListener(Event.ENTER_FRAME,goback);
				m=10;
			}
		}
		main=0;//畫面移到中景
	} else if (main==-1) {
		parent['main'].gotoAndPlay(21);
```

```
            main=0;//畫面移到中景
        } else if (main==0) {
            parent['main'].gotoAndStop(1);
            main==0;
        }
        if (all_rb_bt==0) {//如果彩虹鈕在中景
            parent['all_rinbow_bt'].gotoAndStop(11);
            goto=0;
        } else if (all_rb_bt==1) {//如果彩虹鈕在近景
            parent['all_rinbow_bt'].gotoAndPlay(32);
            goto=0;
        } else if (all_rb_bt==-1) {//如果彩虹鈕在遠景
            parent['all_rinbow_bt'].gotoAndPlay(52);
            goto=0;
        }
}//function 的
down.addEventListener(MouseEvent.ROLL_OVER,移入 down);
down.addEventListener(MouseEvent.ROLL_OUT,移出 down);
down.addEventListener(MouseEvent.CLICK,點擊 down);
function  移入 down(e:MouseEvent):void {
        if (k!=1) {
            down.gotoAndStop(2);
        }
}
function  移出 down(e:MouseEvent):void {
        if (k!=1) {
            down.gotoAndStop(1);
        }
}
function  點擊 down(e:MouseEvent):void {
        if (k!=1) {
            k=1;
            down.gotoAndPlay(3);
            if (j==1) {
                mid.gotoAndPlay(28);
                j=0;
            }
            if (i==1) {
```

```
                up.gotoAndPlay(28);
                i=0;
            }
        }
        if (main==1) {//如果畫面在近景
            parent['main'].gotoAndPlay(11);
            main=-1;//畫面移到遠景
        } else if (main==0) {//如果畫面在中景
            parent['main'].gotoAndStop(30);
            parent['main'].addEventListener(Event.ENTER_FRAME,goback);
            function goback(e:Event):void {
                if (d>=21) {
                    parent['main'].prevFrame();
                    d--;
                } else if (d==20) {
                    parent['main'].removeEventListener(Event.ENTER_FRAME,goback);
                    d=30;
                }
            }
            main=-1;//畫面移到遠景
        } else if (main==-1) {
            parent['main'].gotoAndStop(20);
            main=-1;
        }//畫面移到遠景
        if (all_rb_bt==-1) {//如果彩虹鈕在遠景
            parent['all_rinbow_bt'].gotoAndStop(51);
            goto=-1;
        } else if (all_rb_bt==1) {//如果彩虹鈕在近景
            parent['all_rinbow_bt'].gotoAndPlay(32);
            goto=-1;
        } else if (all_rb_bt==0) {//如果彩虹鈕在中景
            parent['all_rinbow_bt'].gotoAndPlay(12);
            goto=-1;
        }
}
parent['all_rinbow_bt'].addEventListener(Event.ENTER_FRAME,gootherway);
function gootherway(e:Event):void {
        if (goto==1) {
```

```
            if (parent['all_rinbow_bt'].currentFrame==21||parent['all_rinbow_bt'].currentFrame==61) {
                parent['all_rinbow_bt'].gotoAndPlay(22);
                all_rb_bt=1;
            }
        } else if (goto==0) {
            if (parent['all_rinbow_bt'].currentFrame==41||parent['all_rinbow_bt'].currentFrame==61) {
                parent['all_rinbow_bt'].gotoAndPlay(2);
                all_rb_bt=0;
            }
        } else if (goto==-1) {
            if (parent['all_rinbow_bt'].currentFrame==41||parent['all_rinbow_bt'].currentFrame==21) {
                parent['all_rinbow_bt'].gotoAndPlay(42);
                all_rb_bt=-1;
            }
        }
}//function 的
```

1-按照編號抓色表位置顯示顏色

```
var red:int;
var green:int;
var blue:int;
var  自己編號:int=0;
var  透明度:Number=0;
this.addEventListener(Event.ENTER_FRAME,變色);
function  變色(e:Event):void {
//進入影格後開始按照下方指令配色

        red=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][0];
        //依照老爸的[色表],以及用[移到誰(列)]和[自己編號(行)]判斷[顯示顏色]裡的位置,色表裡紅色參數的位置
        green=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][1];
        blue=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][2];

        this.transform.colorTransform=new ColorTransform(1,1,1,parent['大家的透明度'][Math.abs(parent['移到誰']-自己編號)],red,green,blue,0);
        //前三個 1 是紅綠藍的偏移量,第四個是透明度的偏移量<<這行主要再改透明度
```

```
}
```

2-按照編號抓色表位置顯示顏色

```
var red:int;
var green:int;
var blue:int;
var  自己編號:int=1;
var  透明度:Number=0;
this.addEventListener(Event.ENTER_FRAME,變色);
function  變色(e:Event):void {



        red=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][0];
        //依照老爸的[色表],以及用[移到誰(列)]和[自己編號(行)]判斷[顯示顏色]裡的位置,色表裡紅色參數的位置
        green=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][1];
        blue=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][2];

        this.transform.colorTransform=new ColorTransform(1,1,1,parent['大家的透明度'][Math.abs(parent['移到誰']-自己編號)],red,green,blue,0);//前三個 1 是紅綠藍的偏移量,第四個是透明度的偏移量



}
```

3-按照編號抓色表位置顯示顏色

```
var red:int;
var green:int;
var blue:int;
var  自己編號:int=2;
var  透明度:Number=1.0;
this.addEventListener(Event.ENTER_FRAME,變色);
function  變色(e:Event):void {



        red=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][0];
        //依照老爸的[色表],以及用[移到誰(列)]和[自己編號(行)]判斷[顯示顏色]裡的位置,色表裡紅色參數的位置
```

```
        green=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][1];
        blue=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][2];

        this.transform.colorTransform=new ColorTransform(1,1,1,parent['大家的透明度
'][Math.abs(parent['移到誰']-自己編號)],red,green,blue,0);//前三個 1 是紅綠藍的偏移量,第四個是透
明度的偏移量


}
```

4-按照編號抓色表位置顯示顏色

```
var red:int;
var green:int;
var blue:int;
var  自己編號:int=3;
var  透明度:Number=1.0;
this.addEventListener(Event.ENTER_FRAME,變色);
function  變色(e:Event):void {


        red=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][0];
        //依照老爸的[色表],以及用[移到誰(列)]和[自己編號(行)]判斷[顯示顏色]裡的位置,色表裡紅色
參數的位置
        green=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][1];
        blue=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][2];

        this.transform.colorTransform=new ColorTransform(1,1,1,parent['大家的透明度
'][Math.abs(parent['移到誰']-自己編號)],red,green,blue,0);//前三個 1 是紅綠藍的偏移量,第四個是透
明度的偏移量


}
```

5-按照編號抓色表位置顯示顏色

```
var red:int;
var green:int;
var blue:int;
var  自己編號:int=4;
```

```
var  透明度:Number=1.0;
this.addEventListener(Event.ENTER_FRAME,變色);
function  變色(e:Event):void {



        red=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][0];
        //依照老爸的[色表],以及用[移到誰(列)]和[自己編號(行)]判斷[顯示顏色]裡的位置,色表裡紅色參數的位置
        green=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][1];
        blue=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][2];

        this.transform.colorTransform=new ColorTransform(1,1,1,parent['大家的透明度'][Math.abs(parent['移到誰']-自己編號)],red,green,blue,0);//前三個 1 是紅綠藍的偏移量,第四個是透明度的偏移量



}
```

6-按照編號抓色表位置顯示顏色

```
var red:int;
var green:int;
var blue:int;
var  自己編號:int=5;
var  透明度:Number=1.0;
this.addEventListener(Event.ENTER_FRAME,變色);
function  變色(e:Event):void {



        red=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][0];
        //依照老爸的[色表],以及用[移到誰(列)]和[自己編號(行)]判斷[顯示顏色]裡的位置,色表裡紅色參數的位置
        green=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][1];
        blue=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][2];

        this.transform.colorTransform=new ColorTransform(1,1,1,parent['大家的透明度'][Math.abs(parent['移到誰']-自己編號)],red,green,blue,0);//前三個 1 是紅綠藍的偏移量,第四個是透明度的偏移量
```

```
}
```

7-按照編號抓色表位置顯示顏色

```
var red:int;
var green:int;
var blue:int;
var  自己編號:int=6;
var  透明度:Number=1.0;
this.addEventListener(Event.ENTER_FRAME,變色);
function  變色(e:Event):void {



        red=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][0];
        //依照老爸的[色表],以及用[移到誰(列)]和[自己編號(行)]判斷[顯示顏色]裡的位置,色表裡紅色參數的位置
        green=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][1];
        blue=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][2];

        this.transform.colorTransform=new ColorTransform(1,1,1,parent['大家的透明度'][Math.abs(parent['移到誰']-自己編號)],red,green,blue,0);//前三個 1 是紅綠藍的偏移量,第四個是透明度的偏移量



}
```

8-按照編號抓色表位置顯示顏色

```
var red:int;
var green:int;
var blue:int;
var  自己編號:int=7;
var  透明度:Number=1.0;
this.addEventListener(Event.ENTER_FRAME,變色);
function  變色(e:Event):void {



        red=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][0];
        //依照老爸的[色表],以及用[移到誰(列)]和[自己編號(行)]判斷[顯示顏色]裡的位置,色表裡紅色
```

```
參數的位置
        green=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][1];
        blue=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][2];

        this.transform.colorTransform=new ColorTransform(1,1,1,parent['大家的透明度
'][Math.abs(parent['移到誰']-自己編號)],red,green,blue,0);//前三個 1 是紅綠藍的偏移量,第四個是透
明度的偏移量


}
```

9-按照編號抓色表位置顯示顏色

```
var red:int;
var green:int;
var blue:int;
var  自己編號:int=8;
var  透明度:Number=1.0;
this.addEventListener(Event.ENTER_FRAME,變色);
function  變色(e:Event):void {


        red=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][0];
        //依照老爸的[色表],以及用[移到誰(列)]和[自己編號(行)]判斷[顯示顏色]裡的位置,色表裡紅色
參數的位置
        green=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][1];
        blue=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][2];

        this.transform.colorTransform=new ColorTransform(1,1,1,parent['大家的透明度
'][Math.abs(parent['移到誰']-自己編號)],red,green,blue,0);//前三個 1 是紅綠藍的偏移量,第四個是透
明度的偏移量


}
```

10-按照編號抓色表位置顯示顏色

```
var red:int;
var green:int;
var blue:int;
```

```
var  自己編號:int=9;
var  透明度:Number=1.0;
this.addEventListener(Event.ENTER_FRAME,變色);
function  變色(e:Event):void {



        red=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][0];
        //依照老爸的[色表],以及用[移到誰(列)]和[自己編號(行)]判斷[顯示顏色]裡的位置,色表裡紅色參數的位置
        green=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][1];
        blue=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][2];

        this.transform.colorTransform=new ColorTransform(1,1,1,parent['大家的透明度'][Math.abs(parent['移到誰']-自己編號)],red,green,blue,0);//前三個 1 是紅綠藍的偏移量,第四個是透明度的偏移量



}
```

11-按照編號抓色表位置顯示顏色

```
var red:int;
var green:int;
var blue:int;
var  自己編號:int=10;
var  透明度:Number=1.0;
this.addEventListener(Event.ENTER_FRAME,變色);
function  變色(e:Event):void {



        red=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][0];
        //依照老爸的[色表],以及用[移到誰(列)]和[自己編號(行)]判斷[顯示顏色]裡的位置,色表裡紅色參數的位置
        green=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][1];
        blue=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][2];

        this.transform.colorTransform=new ColorTransform(1,1,1,parent['大家的透明度'][Math.abs(parent['移到誰']-自己編號)],red,green,blue,0);//前三個 1 是紅綠藍的偏移量,第四個是透明度的偏移量
```

```
}
```

12-按照編號抓色表位置顯示顏色

```
var red:int;
var green:int;
var blue:int;
var  自己編號:int=11;
var  透明度:Number=1.0;
this.addEventListener(Event.ENTER_FRAME,變色);
function  變色(e:Event):void {


        red=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][0];
        //依照老爸的[色表],以及用[移到誰(列)]和[自己編號(行)]判斷[顯示顏色]裡的位置,色表裡紅色參數的位置
        green=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][1];
        blue=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][2];

        this.transform.colorTransform=new ColorTransform(1,1,1,parent['大家的透明度'][Math.abs(parent['移到誰']-自己編號)],red,green,blue,0);//前三個 1 是紅綠藍的偏移量,第四個是透明度的偏移量


}
```

13-按照編號抓色表位置顯示顏色

```
var red:int;
var green:int;
var blue:int;
var  自己編號:int=12;
var  透明度:Number=1.0;
this.addEventListener(Event.ENTER_FRAME,變色);
function  變色(e:Event):void {


        red=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][0];
```

```
        //依照老爸的[色表],以及用[移到誰(列)]和[自己編號(行)]判斷[顯示顏色]裡的位置,色表裡紅色參數的位置
        green=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][1];
        blue=parent['色表'][parent['顯示顏色'][parent['移到誰']][自己編號]][2];

        this.transform.colorTransform=new ColorTransform(1,1,1,parent['大家的透明度'][Math.abs(parent['移到誰']-自己編號)],red,green,blue,0);//前三個 1 是紅綠藍的偏移量,第四個是透明度的偏移量


}
```

版本 6

近中遠按鈕-滑鼠移入移出效果、滑鼠點擊場景變換

```
var i:int=0;
var j:int=1;
var k:int=0;
var up_a:int=0;
var mid_a:int=0;
var down_a:int=0;
var main:int=0;//預設值為中景
var d:int=30;
var m:int=10;
var u:int=20;
var all_rb_bt:int=0;
var rinbow_f:int=0;
var goto:int=0;
mid.gotoAndStop(3);
//parent['all_rinbow_bt'].gotoAndPlay(2);
var au1:Number=1;
var au2:Number=0;
var ad1:Number=1;
var ad2:Number=0;
var am1:Number=1;
var am2:Number=0;
this.addEventListener(Event.ENTER_FRAME,按鈕位置);
function  按鈕位置(e:Event):void {
```

```
if (up_a==1) {
        if (au1>0) {
                //如果近景被按下  且透明度 1 大於 0.
                au1-=0.1;
                up.alpha=au1;
                down.alpha=au1;
                mid.alpha=au1;
                up.removeEventListener(MouseEvent.CLICK,點擊 up);
                mid.removeEventListener(MouseEvent.CLICK,點擊 mid);
                down.removeEventListener(MouseEvent.CLICK,點擊 down);
        }
        trace('au1='+au1);
        if (au1<=0) {
                up.x=428;
                down.x=30;
                mid.x=790;
                mid.rearview.y=274;
                down.b_view_t.y=269;
                up.sideview.y*=-1;
                up.gotoAndStop(3);
        }
        if (up.x==428&&down.x==30&&mid.x==790&&au2<1) {
                au2+=0.1;
                up.alpha=au2;
                down.alpha=au2;
                mid.alpha=au2;
        }
        trace('au2='+au2);
        if (au2>1) {
                up.addEventListener(MouseEvent.CLICK,點擊 up);
                mid.addEventListener(MouseEvent.CLICK,點擊 mid);
                down.addEventListener(MouseEvent.CLICK,點擊 down);
                au1=1;
                au2=0;
                up_a=0;
        }
}
if (mid_a==1) {
```

```
        if (am1>0) {
            //如果中景按下
            am1-=0.1;
            up.alpha=am1;
            down.alpha=am1;
            mid.alpha=am1;
            up.removeEventListener(MouseEvent.CLICK,點擊 up);
            mid.removeEventListener(MouseEvent.CLICK,點擊 mid);
            down.removeEventListener(MouseEvent.CLICK,點擊 down);
        }
        if (am1<=0) {
            mid.x=428;
            up.x=30;
            down.x=790;
            mid.rearview.y*=-1;
            down.b_view_t.y=269;
            up.sideview.y=269;
            mid.gotoAndStop(3);
            }
        if (up.x==30&&down.x==790&&mid.x==428&&am2<1) {
            am2+=0.1;
            up.alpha=am2;
            down.alpha=am2;
            mid.alpha=am2;
        }
        if (am2>1) {
            up.addEventListener(MouseEvent.CLICK,點擊 up);
            mid.addEventListener(MouseEvent.CLICK,點擊 mid);
            down.addEventListener(MouseEvent.CLICK,點擊 down);
            am1=1;
            am2=0;
            mid_a=0;
        }
    }
    if (down_a==1) {
        if (ad1>0) {
            //如果遠景按下
            ad1-=0.1;
```

```
                up.alpha=ad1;
                down.alpha=ad1;
                mid.alpha=ad1;
                up.removeEventListener(MouseEvent.CLICK,點擊 up);
                mid.removeEventListener(MouseEvent.CLICK,點擊 mid);
                down.removeEventListener(MouseEvent.CLICK,點擊 down);
        }
        if (ad1<=0) {
                down.x=410;
                mid.x=30;
                up.x=790;
                down.b_view_t.y*=-1;
                mid.rearview.y=274;
                up.sideview.y=269;
                down.gotoAndStop(3);
        }
        if (up.x==790&&down.x==410&&mid.x==30&&ad2<1) {
                ad2+=0.1;
                up.alpha=ad2;
                down.alpha=ad2;
                mid.alpha=ad2;
        }
        if (ad2>1) {
                up.addEventListener(MouseEvent.CLICK,點擊 up);
                mid.addEventListener(MouseEvent.CLICK,點擊 mid);
                down.addEventListener(MouseEvent.CLICK,點擊 down);
                ad1=1;
                ad2=0;
                down_a=0;
        }

    }
}
up.addEventListener(MouseEvent.ROLL_OVER,移入 up);
up.addEventListener(MouseEvent.ROLL_OUT,移出 up);
up.addEventListener(MouseEvent.CLICK,點擊 up);
function 移入 up(e:MouseEvent):void {
    if (i!=1) {
```

```
            //近景按鈕沒有被按下
            up.gotoAndStop(2);
        }
}
function  移出 up(e:MouseEvent):void {
        if (i!=1) {
            up.gotoAndStop(1);
        }
}
function  點擊 up(e:MouseEvent):void {
        if (i!=1) {
            //按扭轉換判斷.近景按鈕沒有被按下
            i=1;
            up_a=1;
        if (j==1) {
                //如果中景按鈕被按下了.做以下動作
                mid.gotoAndStop(1);
                j=0;
            }
            if (k==1) {
                //如果遠景按鈕被按下了.做以下動作
                down.gotoAndStop(1);
                k=0;
            }
        }
        if (main==0) {//場景變換判斷
            parent['main'].gotoAndPlay(2);
            main=1;//畫面移到近景
        } else if (main==-1) {//如果畫面在遠景
            parent['main'].addEventListener(Event.ENTER_FRAME,goback);
            function goback(e:Event):void {
                if (u>=11) {
                    parent['main'].prevFrame();
                    u--;
                } else if (u==10) {
                    parent['main'].removeEventListener(Event.ENTER_FRAME,goback);
                    u=20;
                }
```

```
            }
            main=1;
        } else if (main==1) {
            parent['main'].gotoAndStop(10);
            main=1;
        }
        /*if (all_rb_bt==1) {//如果彩虹鈕在近景
            parent['all_rinbow_bt'].gotoAndStop(31);
            goto=1;
        } else if (all_rb_bt==0) {//如果彩虹鈕在中景
            parent['all_rinbow_bt'].gotoAndPlay(12);
            goto=1;
        } else if (all_rb_bt==-1) {//如果彩虹鈕在遠景
            parent['all_rinbow_bt'].gotoAndPlay(52);
            goto=1;
        }*/
}//function 的
mid.addEventListener(MouseEvent.ROLL_OVER,移入 mid);
mid.addEventListener(MouseEvent.ROLL_OUT,移出 mid);
mid.addEventListener(MouseEvent.CLICK,點擊 mid);
function  移入 mid(e:MouseEvent):void {
        if (j!=1) {
            //如果中景按鈕沒有被按下
            mid.gotoAndStop(2);
        }
}
function  移出 mid(e:MouseEvent):void {
        if (j!=1) {
            mid.gotoAndStop(1);
        }
}
function  點擊 mid(e:MouseEvent):void {
        if (j!=1) {
            //如果中景按鈕沒有被按下
            j=1;
            mid_a=1;
                        if (i==1) {
                    //如果近景按鈕被按下了
```

```
                up.gotoAndStop(1);
                i=0;
        }
        if (k==1) {
                //如果遠景按鈕被按下了
                down.gotoAndStop(1);
                k=0;
        }
}
if (main==1) {//如果畫面在近景
        parent['main'].addEventListener(Event.ENTER_FRAME,goback);
        function goback(e:Event):void {
                if (m>=2) {
                        parent['main'].prevFrame();
                        m--;
                } else if (m==1) {
                        parent['main'].removeEventListener(Event.ENTER_FRAME,goback);
                        m=10;
                }
        }
        main=0;//畫面移到中景
} else if (main==-1) {
        parent['main'].gotoAndPlay(21);
        main=0;//畫面移到中景
} else if (main==0) {
        parent['main'].gotoAndStop(1);
        main==0;
}
/*if (all_rb_bt==0) {//如果彩虹鈕在中景
        parent['all_rinbow_bt'].gotoAndStop(11);
        goto=0;
} else if (all_rb_bt==1) {//如果彩虹鈕在近景
        parent['all_rinbow_bt'].gotoAndPlay(32);
        goto=0;
} else if (all_rb_bt==-1) {//如果彩虹鈕在遠景
        parent['all_rinbow_bt'].gotoAndPlay(52);
        goto=0;
}*/
```

```
}//function 的
down.addEventListener(MouseEvent.ROLL_OVER,移入 down);
down.addEventListener(MouseEvent.ROLL_OUT,移出 down);
down.addEventListener(MouseEvent.CLICK,點擊 down);
function  移入 down(e:MouseEvent):void {
	if (k!=1) {
		down.gotoAndStop(2);
	}
}
function  移出 down(e:MouseEvent):void {
	if (k!=1) {
		down.gotoAndStop(1);
	}
}
function  點擊 down(e:MouseEvent):void {
	if (k!=1) {
		//如果遠景按鈕沒有被按下
		k=1;
		down_a=1;
			if (j==1) {
			//如果中景按鈕被按下了
			mid.gotoAndStop(1);
			j=0;
		}
		if (i==1) {
			//如果近景按鈕被按下了
			up.gotoAndStop(1);
			i=0;
		}
	}
	if (main==1) {//如果畫面在近景
		parent['main'].gotoAndPlay(11);
		main=-1;//畫面移到遠景
	} else if (main==0) {//如果畫面在中景
		parent['main'].gotoAndStop(30);
		parent['main'].addEventListener(Event.ENTER_FRAME,goback);
		function goback(e:Event):void {
			if (d>=21) {
```

```
                parent['main'].prevFrame();
                d--;
            } else if (d==20) {
                parent['main'].removeEventListener(Event.ENTER_FRAME,goback);
                d=30;
            }
        }
        main=-1;//畫面移到遠景
    } else if (main==-1) {
        parent['main'].gotoAndStop(20);
        main=-1;
    }//畫面移到遠景
    /*if (all_rb_bt==-1) {//如果彩虹鈕在遠景
        parent['all_rinbow_bt'].gotoAndStop(51);
        goto=-1;
    } else if (all_rb_bt==1) {//如果彩虹鈕在近景
        parent['all_rinbow_bt'].gotoAndPlay(32);
        goto=-1;
    } else if (all_rb_bt==0) {//如果彩虹鈕在中景
        parent['all_rinbow_bt'].gotoAndPlay(12);
        goto=-1;
    }*/
}
/*parent['all_rinbow_bt'].addEventListener(Event.ENTER_FRAME,gootherway);
function gootherway(e:Event):void {
    if (goto==1) {
        if (parent['all_rinbow_bt'].currentFrame==21||parent['all_rinbow_bt'].currentFrame==61) {
            parent['all_rinbow_bt'].gotoAndPlay(22);
            all_rb_bt=1;
        }
    } else if (goto==0) {
        if (parent['all_rinbow_bt'].currentFrame==41||parent['all_rinbow_bt'].currentFrame==61) {
            parent['all_rinbow_bt'].gotoAndPlay(2);
            all_rb_bt=0;
        }
    } else if (goto==-1) {
        if (parent['all_rinbow_bt'].currentFrame==41||parent['all_rinbow_bt'].currentFrame==21) {
            parent['all_rinbow_bt'].gotoAndPlay(42);
```

```
                all_rb_bt=-1;
            }
        }
}//function 的*/
```

New-載入 google map

```
var googleMap:MovieClip=new googlemap();
this.addChild(googleMap);

var reLoadTime:uint=40;
var loop:uint=0;

this.addEventListener(Event.ENTER_FRAME,upt);
function upt(e:Event) {
        reLoadTime--;
        if (reLoadTime<1) {
            if (this.googleMap.ok) {
                //stageno.text=loop+'';
                //googleMap.map.clearOverlays();

        googleMap.MapLon.setServer('http://120.118.93.149/getLon.php?stage='+loop+'&table=nowlatlon
');

        googleMap.MapLat.setServer('http://120.118.93.149/getLat.php?stage='+loop+'&table=nowlatlon')
;

                googleMap.addEventListener(Event.ENTER_FRAME,googleMap.setMap);
                this.googleMap.ok=false;
            }
            reLoadTime=100;
        }
}
```

飛船葉片-整顆飛船

```
var i:int=20;
var j:int=1;
var a:int=40;
```

```
this.addEventListener(Event.ENTER_FRAME,rota);
    //這個物件.設一監聽程式(事件類型.進入影格.做"rota")
function rota(e:Event):void{    //
        this.rotation = this.rotation - a;
}
```

主場景-近中遠景載入資料、中景飛船浮動效果、各物件依資料對應之動畫、滑鼠移入移出物件之說明框動畫

```
var  中景載入器:URLLoader = new URLLoader();
中景載入器.dataFormat=URLLoaderDataFormat.VARIABLES;//資料格式=載入變數
var top_server:String='http://120.118.93.149/top.php';//建立 URL 請求物件
var  中景要求連結:URLRequest=new URLRequest(top_server);//設定傳送方式
中景要求連結.method=URLRequestMethod.POST;//改 post 或 get 就可以用不同方式傳，也要用不同方式收
var  中景資料封包:URLVariables = new URLVariables();//建立傳送數值物件，也是用來存變數的容器
//傳送的東西完成了的事件
var heading 角度:int;
var heading 方向:int;
var  中景 rolling:int;
var rolling 方向:int;
var  自動_手動:int;
var  地控_飛控:int;
var  指北針:int;
var  離目標點距離:int;
var  離目標點高度:Number;
var i:int=0;
var j:int=0;
var u:int=0;
//飛船浮動效果
var boat_x:int = 1;
var boat_y:int = 1;
this.main.topall.topup.top_boat.addEventListener(Event.ENTER_FRAME,boatmove);
function boatmove(e:Event):void
{
        this.main.topall.topup.top_boat.x +=    0.45 * boat_x;//進入 1 影格.移動 1 像素
        this.main.topall.topup.top_boat.y +=    0.5 * boat_y;
        if (this.main.topall.topup.top_boat.x >= 3 || this.main.topall.topup.top_boat.x <= -3){
```

```
            boat_x =    -    boat_x;
        }
        if (this.main.topall.topup.top_boat.y >= 3|| this.main.topall.topup.top_boat.y <= -3){
            boat_y =    -    boat_y;
        }
}
//中景載入器
=============================================================================================
=========

中景載入器.addEventListener(Event.COMPLETE,toploaded);//完成了就做下面的動作
function toploaded(e:Event) {
        //trace(myloader.data.msg);//php 處理後回報怎樣的訊息，秀在輸出面版
        heading 角度=中景載入器.data.heading;
        heading 方向=中景載入器.data.movelr;
        中景 rolling=中景載入器.data.rolling;
        rolling 方向=中景載入器.data.rolling_way;
        自動_手動=中景載入器.data.control;
        地控_飛控=中景載入器.data.grocon;
        指北針=中景載入器.data.north;
        離目標點距離=中景載入器.data.rd2d;
        離目標點高度=中景載入器.data.howhigh;
        //heading 方向
        if(heading 方向==1){
            this.main.topall.topup.top_heading.headingall.gotoAndPlay(1);//播放右轉動畫
        }else if(heading 方向==-1){
            this.main.topall.topup.top_heading.headingall.gotoAndPlay(9);//撥放左轉動畫
            }
        //heading 角度
        this.main.topall.topup.top_heading.headingall.topheading_doc.text=heading 角度;
        //rolling 方向.飛船旋轉角度
        if(rolling 方向==1){
            this.main.topall.topup.rolling.rollingall.gotoAndPlay(13);//播放向右轉角
            //this.main.topall.topup.top_boat.rotation=中景 rolling;//飛船轉角向右
        }else if(rolling 方向==-1){
            this.main.topall.topup.rolling.rollingall.gotoAndPlay(1);//播放向左轉角
            //this.main.topall.topup.top_boat.rotation=(360+中景 rolling);//飛船轉角向左
        }
```

```
	//rolling 角度
	this.main.topall.topup.rolling.rollingall.top_rolling_do.text=中景 rolling;
	//控制系統
	if(自動_手動==1){
		this.main.topall.topup.control.controlall.gotoAndStop(1);//手動動畫
	}else if(自動_手動==0){
		if(地控_飛控==1){
			this.main.topall.topup.control.controlall.gotoAndStop(2);//地面控制
		}else if(地控_飛控==0){
			this.main.topall.topup.control.controlall.gotoAndStop(3);//飛船控制
		}
	}
	//指北針
	//this.main.topall.topup.compass.compassall.compassya.rotation=指北針;

	trace(heading 角度);
	//離目標點距離
	this.main.topall.topup.howlong.rd2d.text=離目標點距離;
	//離目標點高度
	this.main.topall.topup.howlong.howhigh.text=離目標點高度;

}
this.addEventListener(Event.ENTER_FRAME,中景資料載入);//監聽程式-進入影格開始     this=這個物件本身
function  中景資料載入(e:Event) {

	if (i%24==0) {
		//(24 影格載一次=時間軸跑 1 秒)i 跟 24 取餘數=0，則重新載入
		中景載入器.load(中景要求連結);
		i=0;
	}
	i++;
}
//近景載入器
=============================================================================
=========

var  近景載入器:URLLoader = new URLLoader();
```

```
近景載入器.dataFormat=URLLoaderDataFormat.VARIABLES;//資料格式=載入變數
var left_server:String='http://120.118.93.149/left.php';//建立 URL 請求物件
var  近景要求連結:URLRequest=new URLRequest(left_server);//設定傳送方式
近景要求連結.method=URLRequestMethod.POST;//改 post 或 get 就可以用不同方式傳，也要用不同方式收
var  近景資料封包:URLVariables = new URLVariables();//建立傳送數值物件，也是用來存變數的容器
//傳送的東西完成了的事件
var pitching:int;
var pitching 方向:int;
var  近景 rolling:int;
var  近景 rolling 方向:int;
var  溫度:int;
var  過熱:int;
var  近景高度:int;
var  推力:int;
var  馬達輸出倍率:int;
近景載入器.addEventListener(Event.COMPLETE,leftloaded);//完成了就做下面的動作
function leftloaded(e:Event) {
        //trace(myloader.data.msg);//php 處理後回報怎樣的訊息，秀在輸出面版
        pitching=近景載入器.data.pitching;
        pitching 方向=近景載入器.data.moveud;
        近景 rolling=近景載入器.data.rolling;
        近景 rolling 方向=近景載入器.data.rolling_way;
        溫度=近景載入器.data.do_c;
        過熱=近景載入器.data.overhot;
        近景高度=近景載入器.data.now_high;
        推力=近景載入器.data.moto_push;
        馬達輸出倍率=近景載入器.data.moout1;
        //已行走距離=中景載入器.data.rd2d;
        //時速=近景載入器.data.;
        //傾角方向
        if(pitching 方向==1){
        this.main.left_f.傾角.pitching.gotoAndPlay(1);
        //傾角向上
        }else if(pitching 方向==-1){
        this.main.left_f.傾角.pitching.gotoAndPlay(6);
        //傾角向下
        }
```

```
	//傾角度數
	this.main.left_f.傾角.pitching_do.text=pitching;
	//轉角方向
	if(近景 rolling 方向==1){
	this.main.left_f.轉角.left_rolling.gotoAndPlay(1);
	}else if(近景 rolling 方向==-1){
	this.main.left_f.轉角.left_rolling.gotoAndPlay(8);
	}
	//轉角度數
	this.main.left_f.轉角.left_rolling_do.text=Math.abs(近景 rolling);
	//過熱指標
	if(過熱==1){
		this.main.left_f.溫度.left_do_doc.text=溫度;
		this.main.left_f.溫度.gotoAndPlay(2);
		this.main.left_f.溫度.left_do_doc.textColor=0xFF0000;
	}else if(過熱==0){
		this.main.left_f.溫度.left_do_doc.text=溫度;
		this.main.left_f.溫度.left_do_doc.textColor=0xFFFFFF;
	}
	//推力  7=0;8=15;9=25;10=35=;11=50;12=75;13=100



}
//溫度----------------------------------------------------------------------------------------------
var do_mov:int=0;
var fly_mov:int=10;
var p_power:int=0;
var ture_power:int=0;
var top_boat_mov:int=0;
var top_compass:int=0;
var top_a:int=0;
var left_boat:int=0;
var left_boat_rotation:int=0;
this.addEventListener(Event.ENTER_FRAME,資料動畫);
function  資料動畫(e:Event) {
	if(溫度<=50&&溫度>=15){
		if(do_mov==0){
```

```
                do_mov=溫度;
            }
            if(do_mov>溫度){
                //如果舊的溫度>新的溫度
                do_mov-=1;
                this.main.left_f.溫度.left_do.gotoAndStop((do_mov-9));
            }else if(do_mov<溫度){
                //如果舊的溫度<新的溫度
                do_mov+=1;
                this.main.left_f.溫度.left_do.gotoAndStop((do_mov-9));
            }else if(do_mov==溫度){
                //如果舊的溫度=新的溫度
                this.main.left_f.溫度.left_do.gotoAndStop((do_mov-9));
                do_mov=溫度;
            }
        }
//近景飛船
rolling=================================================================================
=========
        if(近景 rolling 方向==-1&&fly_mov>1){
            fly_mov+=1;
            this.main.left_f.flyboatxl.gotoAndStop((fly_mov));
        }else if(近景 rolling 方向==1&&fly_mov<20){
            fly_mov-=1;
            this.main.left_f.flyboatxl.gotoAndStop((fly_mov));
        }else if(近景 rolling 方向==0){
            if(fly_mov>10){
                fly_mov+=1;
                this.main.left_f.flyboatxl.gotoAndStop((fly_mov));
            }else if(fly_mov<10){
                fly_mov-=1;
                this.main.left_f.flyboatxl.gotoAndStop((fly_mov));
            }else if(fly_mov==10){
                fly_mov=10;
                this.main.left_f.flyboatxl.gotoAndStop((fly_mov));
            }

        }
```

```
//推力 7=0;8=15;9=25;10=35=;11=50;12=75;13=100----------
	if(推力==7){
		if(p_power>推力){
			ture_power--;
			if(ture_power==0){
				ture_power=0;
				p_power=7;
			}
		}
	}else if(推力==8){
		if(p_power<推力){
			ture_power++;
		}else if(p_power>推力){
			ture_power--;
		}if(ture_power==15){
				ture_power=15;
				p_power=8;
		}
	}else if(推力==9){
		if(p_power<推力){
			ture_power++;
		}else if(p_power>推力){
			ture_power--;
		}if(ture_power==25){
				ture_power=25;
				p_power=9;
		}
	}else if(推力==10){
		if(p_power<推力){
			ture_power++;
		}else if(p_power>推力){
			ture_power--;
		}if(ture_power==35){
				ture_power=35;
				p_power=10;
		}
	}else if(推力==11){
		if(p_power<推力){
```

```
                ture_power++;
            }else if(p_power>推力){
                ture_power--;
            }if(ture_power==50){
                        ture_power=50;
                        p_power=11;
            }
        }else if(推力==12){
            if(p_power<推力){
                ture_power++;
            }else if(p_power>推力){
                ture_power--;
            }if(ture_power==75){
                        ture_power=75;
                        p_power=12;
            }
        }else if(推力==13){
            if(p_power<推力){
                ture_power++;
            }else if(p_power>推力){
                ture_power--;
            }if(ture_power>=100){
                        ture_power=100;
                        p_power=13;
            }
        }
        this.main.left_f.推力.push_power.text=ture_power;
        trace(ture_power);

//中景飛船
rolling=================================================================================
=========
        if(top_boat_mov>中景 rolling){
            //如果舊的角度>新的角度
            top_boat_mov--;
            this.main.topall.topup.top_boat.rotation=top_boat_mov;
        }else if(top_boat_mov<中景 rolling){
            //如果舊的角度<新的角度
```

```
            top_boat_mov++;
            this.main.topall.topup.top_boat.rotation=top_boat_mov;
      }else if(top_boat_mov==中景 rolling){
            //如果舊的角度=新的角度
            top_boat_mov=中景 rolling;
            this.main.topall.topup.top_boat.rotation=top_boat_mov;
      }
      trace('中景 rolling='+top_boat_mov);
//  指北針
================================================================================
==============
      if(top_compass>指北針){
            //如果舊的角度大於新角度
            if(Math.abs(top_compass-指北針)>=180){
                  //新舊角度的距離差大於或等於 180 度
                  top_compass++;
                  //舊角度++
                  if(top_compass==360){
                        top_compass=0;
                        }
                  this.main.topall.topup.compass.compassall.compassya.rotation=top_compass;

      this.main.topall.topup.compass.compassall.compassya.north.rotation=(360-top_compass);
            }else if(Math.abs(top_compass-指北針)<180){
                  top_compass--;
                  if(top_compass<0){
                        top_compass=top_compass+360;
                        }
                        this.main.topall.topup.compass.compassall.compassya.rotation=top_compass;

      this.main.topall.topup.compass.compassall.compassya.north.rotation=(360-top_compass);
            }
      }else if(top_compass<指北針){
            //如果舊的角度小於新角度
            if(Math.abs(top_compass-指北針)>=180){
                  top_compass--;
                        if(top_compass<0){
                        top_compass=top_compass+360;
```

```
                    }
                this.main.topall.topup.compass.compassall.compassya.rotation=top_compass;

    this.main.topall.topup.compass.compassall.compassya.north.rotation=(360-top_compass);
        }else if(Math.abs(top_compass-指北針)<180){
                top_compass++;
                if(top_compass==360){
                    top_compass=0;
                    }
                this.main.topall.topup.compass.compassall.compassya.rotation=top_compass;

    this.main.topall.topup.compass.compassall.compassya.north.rotation=(360-top_compass);
        }
    }
//近景飛船
pitching=============================================================================
==========
    if(this.main.left_f.currentFrame==50){
         if(pitching 方向==1){
/*                  pitching*=-1;
                  left_boat*=-1;*/
                if(left_boat>pitching){
                    //如果舊的角度>新的角度
                    left_boat--;
                    this.main.left_f.flyboatxl.rotation=-left_boat;
                }else if(left_boat<pitching){
                    //如果舊的角度<新的角度
                    left_boat++;
                    this.main.left_f.flyboatxl.rotation=-left_boat;
                }else if(left_boat==pitching){
                    left_boat=pitching;
                    this.main.left_f.flyboatxl.rotation=-left_boat;
                }
        }else if(pitching 方向==-1){
                if(left_boat>pitching){
                    //如果舊的角度>新的角度
                    left_boat--;
                    this.main.left_f.flyboatxl.rotation=left_boat;
```

```
                }else if(left_boat<pitching){
                    //如果舊的角度<新的角度
                    left_boat++;
                    this.main.left_f.flyboatxl.rotation=left_boat;
                }else if(left_boat==pitching){
                    left_boat=pitching;
                    this.main.left_f.flyboatxl.rotation=left_boat;
                }
            }else if(pitching 方向==0){
                    left_boat=0;
                    this.main.left_f.flyboatxl.rotation=left_boat;
            }
        }
            trace('pitching='+left_boat);
//以下括號是 fanction 的
}
this.addEventListener(Event.ENTER_FRAME,近景資料載入);//監聽程式-進入影格開始     this=這個物件本身
function  近景資料載入(e:Event) {

        if (j%24==0) {//(影格跑 25 格)i 跟 25 取餘數=0，則重新載入
            近景載入器.load(近景要求連結);
        }j++;
}

//  遠景載入器
================================================================================
=====

var  遠景載入器:URLLoader = new URLLoader();
遠景載入器.dataFormat=URLLoaderDataFormat.VARIABLES;//資料格式=載入變數
var right_server:String='http://120.118.93.149/right.php';//建立 URL 請求物件
var  遠景要求連結:URLRequest=new URLRequest(right_server);//設定傳送方式
遠景要求連結.method=URLRequestMethod.POST;//改 post 或 get 就可以用不同方式傳，也要用不同方式收
var  遠景資料封包:URLVariables = new URLVariables();//建立傳送數值物件，也是用來存變數的容器
//傳送的東西完成了的事件
var  目標位置 ns:Number;
```

```
var  目標位置 we:Number;
var  現在位置 ns:Number;
var  現在位置 we:Number;
var  遠景高度:int;

遠景載入器.addEventListener(Event.COMPLETE,rightloaded);//完成了就做下面的動作
function rightloaded(e:Event) {
	//trace(myloader.data.msg);//php 處理後回報怎樣的訊息，秀在輸出面版
	/*目標位置 ns=遠景載入器.data.goal_yyy+""+遠景載入器.data.goalns;
	目標位置 we=遠景載入器.data.goal_xxx+""+遠景載入器.data.goalwe;
	現在位置 ns=遠景載入器.data.yyy+""+遠景載入器.data.nowns;
	現在位置 we=遠景載入器.data.xxx+""+遠景載入器.data.nowwe;
	遠景高度=近景載入器.data.now_high;*/
	//已行走距離=中景載入器.data.2drd;
	//時速=近景載入器.data.;
}
this.addEventListener(Event.ENTER_FRAME,遠景資料載入);//監聽程式-進入影格開始     this=這個物件本身
function  遠景資料載入(e:Event) {

	if (u%24==0) {//(影格跑 25 格)i 跟 25 取餘數=0，則重新載入
		遠景載入器.load(遠景要求連結);
	}
	u++;
}

//說明框-說明文字區
================================================================================

var  說明框說明:int=0;
說明框.alpha=0;
this.addEventListener(Event.ENTER_FRAME,說明框專用);
function  說明框專用(e:Event):void {
	說明框.x=mouseX+60;
	說明框.y=mouseY+30;
	說明框.說明框文字.autoSize='center';//文字自動調整大小
	說明框.說明框底.width=說明框.說明框文字.width+30;//文字底框與文字框一樣大
	說明框.說明框底.height=說明框.說明框文字.height+24;
```

```
//中景部份----------------------------------------------------------------------------------------
	if(說明框說明==1){
		//中景距離
		說明框.alpha=1;
		說明框說明=0;
		說明框.說明框文字.text='離目標點差\n'+離目標點距離+' M\n 與目標點高度差  '+離目標點高度+' M';
		trace('離目標點距離'+離目標點距離+'M\n 與目標點高度差'+離目標點高度+'M');
	}else if(說明框說明==2){
		//控制系統
		說明框.alpha=1;
		說明框說明=0;
		if(自動_手動==1){
		說明框.說明框文字.text='遙控器控制';//手動動畫
		}else if(自動_手動==0){
			if(地控_飛控==1){
			說明框.說明框文字.text='晶片控制\n 地面監控站系統';//地面控制
			}else if(地控_飛控==0){
			說明框.說明框文字.text='晶片控制\n 飛船自動系統';//飛船控制
			}
		}
	}else if(說明框說明==3){
		//指北針
		說明框.alpha=1;
		說明框說明=0;
		說明框.說明框文字.text='指北針';
	}else if(說明框說明==4){
		//中景 rolling
		說明框.alpha=1;
		說明框說明=0;
		if(rolling 方向==1){
			說明框.說明框文字.text='轉角向右\n'+中景 rolling+'  度';//播放向右轉角
		}else if(rolling 方向==-1){
			說明框.說明框文字.text='轉角向左\n'+(中景 rolling*-1)+'  度';//播放向左轉角
		}else if(rolling 方向==0){
			說明框.說明框文字.text='轉角\n'+中景 rolling+'  度';//轉角 0 度
		}
	}else if(說明框說明==5){
```

```
            //heading 角度
            說明框.alpha=1;
            說明框說明=0;
            if(heading 方向==1){
                說明框.說明框文字.text='方向角向右\n'+heading 角度+'  度';//播放右轉動畫
            }else if(heading 方向==-1){
                說明框.說明框文字.text='方向角向左\n'+heading 角度+'  度';//撥放左轉動畫
            }
//近景部份----------------------------------------------------------------------------------------------
        //轉角
        }else if(說明框說明==6){
                        說明框.alpha=1;
                        說明框說明=0;
                    if(近景 rolling 方向==1){
                        說明框.說明框文字.text='轉角向右\n'+近景 rolling+'  度';
                    }else if(近景 rolling 方向==-1){
                        說明框.說明框文字.text='轉角向左\n'+(近景 rolling*-1)+'  度';
                    }
        //傾角
        }else if(說明框說明==7){
            說明框.alpha=1;
            說明框說明=0;
            if(pitching 方向==1){
                說明框.說明框文字.text='傾角向上\n'+pitching+'  度';
            }else if(pitching 方向==-1){
                說明框.說明框文字.text='傾角向下\n'+pitching+'  度';
            }
        //溫度
        }else if(說明框說明==8){
            說明框.alpha=1;
            說明框說明=0;
            if(過熱==1){
                說明框.說明框文字.text='危險!!\n 目前飛船溫度\n 攝氏'+溫度+'  度\n 飛船  過熱';
            }else if(過熱==0){
                說明框.說明框文字.text='目前飛船溫度\n 攝氏'+溫度+'  度';
            }
        }else if(說明框說明==9){
            說明框.alpha=1;
```

```
            說明框說明=0;
            說明框.說明框文字.text='飛船推力\n '+ture_power+' %\n 馬達出力倍率\n100 %';

        }/*else if(){
            說明框.alpha=1;
            說明框說明=0;
        }*/
//說明框--按鈕區
================================================================================
//中景部份--------------------------------------------------------------------------
}
this.main.topall.topup.howlong.addEventListener(MouseEvent.ROLL_OUT,移出距離);
this.main.topall.topup.howlong.addEventListener(MouseEvent.ROLL_OVER,在距離上);
function  移出距離(e:MouseEvent):void {
    說明框.alpha=0;
}
function  在距離上(e:MouseEvent):void {
    說明框說明=1;
}
this.main.topall.topup.control.controlall.addEventListener(MouseEvent.ROLL_OUT,移出控制系統);
this.main.topall.topup.control.controlall.addEventListener(MouseEvent.ROLL_OVER,在控制系統上);
function  移出控制系統(e:MouseEvent):void {
    說明框.alpha=0;
}
function  在控制系統上(e:MouseEvent):void {
    說明框說明=2;
}
this.main.topall.topup.compass.compassall.addEventListener(MouseEvent.ROLL_OUT,移出指北針);
this.main.topall.topup.compass.compassall.addEventListener(MouseEvent.ROLL_OVER,在指北針上);
function  移出指北針(e:MouseEvent):void {
    說明框.alpha=0;
}
function  在指北針上(e:MouseEvent):void {
    說明框說明=3;
}
this.main.topall.topup.rolling.addEventListener(MouseEvent.ROLL_OUT,移出中景 rolling);
this.main.topall.topup.rolling.addEventListener(MouseEvent.ROLL_OVER,在中景 rolling 上);
function  移出中景 rolling(e:MouseEvent):void {
```

```
		說明框.alpha=0;
}
function  在中景 rolling 上(e:MouseEvent):void {
		說明框說明=4;
}
this.main.topall.topup.top_heading.addEventListener(MouseEvent.ROLL_OUT,移出中景 heading);
this.main.topall.topup.top_heading.addEventListener(MouseEvent.ROLL_OVER,在中景 heading 上);
function  移出中景 heading(e:MouseEvent):void {
		說明框.alpha=0;
}
function  在中景 heading 上(e:MouseEvent):void {
		說明框說明=5;
}
//近景部份----------------------------------------------------------------------------------------
this.main.left_f.轉角.addEventListener(MouseEvent.ROLL_OUT,移出近景 rolling);
this.main.left_f.轉角.addEventListener(MouseEvent.ROLL_OVER,在近景 rolling 上);
function  移出近景 rolling(e:MouseEvent):void {
		說明框.alpha=0;
}
function  在近景 rolling 上(e:MouseEvent):void {
		說明框說明=6;
		trace('移入 rolling');
}
this.main.left_f.傾角.addEventListener(MouseEvent.ROLL_OUT,移出近景 pitching);
this.main.left_f.傾角.addEventListener(MouseEvent.ROLL_OVER,在近景 pitching 上);
function  移出近景 pitching(e:MouseEvent):void {
		說明框.alpha=0;
}
function  在近景 pitching 上(e:MouseEvent):void {
		說明框說明=7;
		trace('移入近景 pitching');
}
this.main.left_f.溫度.addEventListener(MouseEvent.ROLL_OUT,移出溫度);
this.main.left_f.溫度.addEventListener(MouseEvent.ROLL_OVER,在溫度上);
function  移出溫度(e:MouseEvent):void {
		說明框.alpha=0;
}
function  在溫度上(e:MouseEvent):void {
```

```
	說明框說明=8;
	trace('移入溫度');
}
this.main.left_f.推力.addEventListener(MouseEvent.ROLL_OUT,移出推力);
this.main.left_f.推力.addEventListener(MouseEvent.ROLL_OVER,在推力上);
function  移出推力(e:MouseEvent):void {
	說明框.alpha=0;
}
function  在推力上(e:MouseEvent):void {
	說明框說明=9;
	trace('移入推力');
}
```