# 2023 年臺灣國際科學展覽會
# 優勝作品專輯

作品編號　**190001**

參展科別　電腦科學與資訊工程

作品名稱　以機器學習增強無人機飛行準確度

得獎獎項　三等獎

巴西科學博覽會(MOSTRATEC)代表

就讀學校　國立花蓮高級中學

指導教師　趙義雄

作者姓名　林郁軒

關鍵詞　機器學習、無人機、仿真模擬

# 作者簡介

我是林郁軒，目前就讀國立花蓮高中，我在國中時曾經與資訊專題老師做了一系列機器學習與無人機相關的研究，而到了高中的科展研究則實際運用上了這些方法，雖然這次並沒有真正做出無人機而是使用程式模擬的方式，但是這種作法也讓我在研究進行的方法上開了眼界，期待未來能接觸到更多新的技術與研究方法。

# 摘要

無人機在進行定位時，多半是依靠內建 GPS 晶片與內建慣性測量單元(Inertial Measurement Unit, IMU)進行定位，然而高精度的 IMU 及 GPS 晶片受限於高成本無法在一般無人機上運行；此外，各種定位系統均有其適用範圍，若無人機運行於定位系統之適用環境外，其定位精確度會下降，進而導致無人機飛行時會與預期路線產生誤差。

在本研究中，我利用 Webots 模擬軟體進行無人機模擬，藉由無人機鏡頭所拍攝的連續兩幀圖片差異，產生差異與角度及距離間的關係資料集，並利用此資料集來訓練深度神經網路，將產生模型用以模型迴歸出連續圖片間的旋轉角度偏移量，以此偏移量輔助無人機進行飛行校正。

經過多次實驗與修改，我比較了幾種不同的資料處理與分類方法，找出當中最佳結果的機器學習模型後，將此模型套入模擬環境中輔助無人機飛行，使無人機飛行於複雜環境時，成功提升飛行準確度。

# Abstract

Nowadays, the methods of positioning drones mostly rely on the built-in GPS chip and the Inertial Measurement Unit (IMU). When it comes to budget consideration, these devices are too costly to run on general-purpose drones. Besides, using these devices beyond the scope of the original design will decrease flight accuracy.

In this research, we use the Webots app to simulate the route flight of general drones and generate the datasets by mapping the difference of angle and distance between every two consecutive image frames. Then we use these datasets to train a regression neural network model. The model will predict the change of rotation angle and help to enhance flight accuracy during route flight.

We get the best regression model after comparing several data processing methods and classification techniques. Finally, the model successfully enhances flight accuracy during the route flight in a complicated environment.

# 壹、前言

一、研究動機

現今四軸無人機及飛行穩定算法、自駕技術發達，然而，卻時不時耳聞無人機飛行意外，其中，造成最多墜機事件的為無人機自動返航時墜落及自駕型無人機在飛行時墜毀，追究其原因，最大宗為訊號受干擾及定位異常，而我曾經也因自製四軸無人機的運行動作不好而尋找許久，最後發現是感測器的誤差導致動作不精準，現今無人機雖然具有多感測器以加強定位精度，但礙於成本與技術考量，在無人機上的定位裝置有時會受限於地理環境、訊號干擾等受到嚴重影響。為此，本研究利用四軸無人機常見的鏡頭作為輔助感測器的依據，並與深度學習結合，製作能輔助無人機飛行的人工智慧程式，並期望其能對無人機飛行於較多障礙物等複雜的環境時，能以此方法提升飛行準確度。

目前有許多的研究團隊與無人機廠商正嘗試解決訊號干擾及定位異常問題，如有研究團隊嘗試透過雙頻連線增強無人機定位能力[4]，而著名的DJI空拍機即是以優秀的IMU調校及D-RTK(一種雙頻連線技術的變形)技術聞名，其能透過二次差分運算得出比傳統即時動態定位技術更高的穩定性及精確度，然而，即便是具有如D-RTK技術的DJI無人機，在有些情況下依舊會受到干擾產生無人機偏移，甚至墜機。

現今多數具有無人機定點視覺定位技術的無人機，多是採用光流定位法，雖然有距地高度的限制，但開啟後定位準確度多能比 GPS 定位更精準，於是此研究便採用了無人機鏡頭拍攝的影像作為無人機飛行修正依據，且為了能降低無人機製造成本，嘗試僅使用單攝影機進行定位增強，不過現今採用單攝影機進行定位的方法是基於 SLAM 技術(Simultaneous Location And Mapping, SLAM)，同時因為受到單攝影機無法辨別世界中物體的真實尺度影響，往往具有相當大的誤差。本研究嘗試以單攝影機的視覺資訊進行定位增強，而不是直接的定位功能，預期能加強無人機的動態定位能力。

二、研究目的

(一) 使用Webots模擬無人機飛行，並比較以幾種方法生成之影像資料作為訓練資料及不同機器學習模型，找出具有較佳判別無人機飛行誤差能力的學習模型與資料。

(二) 針對無人機的飛行偏差問題，製作一深度學習模型來判別飛行中的誤差，使無人機能準確地依照設定的路線飛行。

(三) 比較無人機之飛行路徑與無人機經由機器學習模型校正後之飛行路徑，探討無人機經學習模型加強定位後，是否能有效提升目前無人機之定位能力。

(四)　探討模型實作於真實世界之無人機時，其效率是否適合在真實世界使用。

三、文獻回顧

在進行本研究時，我先對無人機定位之相關研究進行爬梳，找到了一種以無人機鏡頭對無人機飛行於室內時進行立體視覺定位之方法[1]，其透過在飛行前以鏡頭在定點採集無人機周遭資訊，並建構成一個立體模型，使無人機能在飛行時使用建構完成之模型及鏡頭資訊進行室內定位，其方法與本研究所構想之方法同樣使用了鏡頭建構資訊，此研究相比其他使用算法或感測器增強定位的研究，與本研究更相似，不過此研究相比本研究而言，其能夠直接進行定位工作，且在特定狀況下具有良好的定位能力，不過其依靠對周遭的物體建模需要耗費大量運算能力，大幅增加無人機的運算負載，並且其所使用的飛時攝影機(Time-of-Flight Camera, ToF Camera)輔助方案將大幅提高無人機的製造成本，此外，該研究在飛行前對周遭環境進行建模，需要花費額外時間。

本研究經由與前述方法比較過後，使用鏡頭拍攝之連續圖像資訊針對無人機定位進行增強，相比前者本方法無法直接進行定位而不依賴其他定位系統，不過本方法旨在能實際運用於生活中之無人機，因此本研究採用AI模型針對連續圖像資訊進行旋轉幅度推測，由於不需要進行立體模型建構，故相比前者能大幅減少運算所需資源，本研究最後以算法模型大小，針對實際應用時可能的表現進行評估，並且比較模型對於應用環境的適用性，總結而言，本研究相比於前述研究更傾向於實際應用層面，雖然精確度上可能不及前者，但是更能符合真實所需。

# 貳、研究方法與過程

一、研究設備

(一) 硬體

1. 電腦

(1) CPU：i9-10900KF

(2) RAM：32GB

(3) GPU：GTX1080Ti-11GB

(4) SSD：1TB

(二) 軟體與作業系統

1. 作業系統：Windows 10 Pro 21H2

2. 開發與測試環境：

(1) IDE：Visual Studio Code 1.67.2

(2) Python直譯器：Conda-Python 3.8.5

(3) 仿真軟體：Webots R2022a

二、研究流程

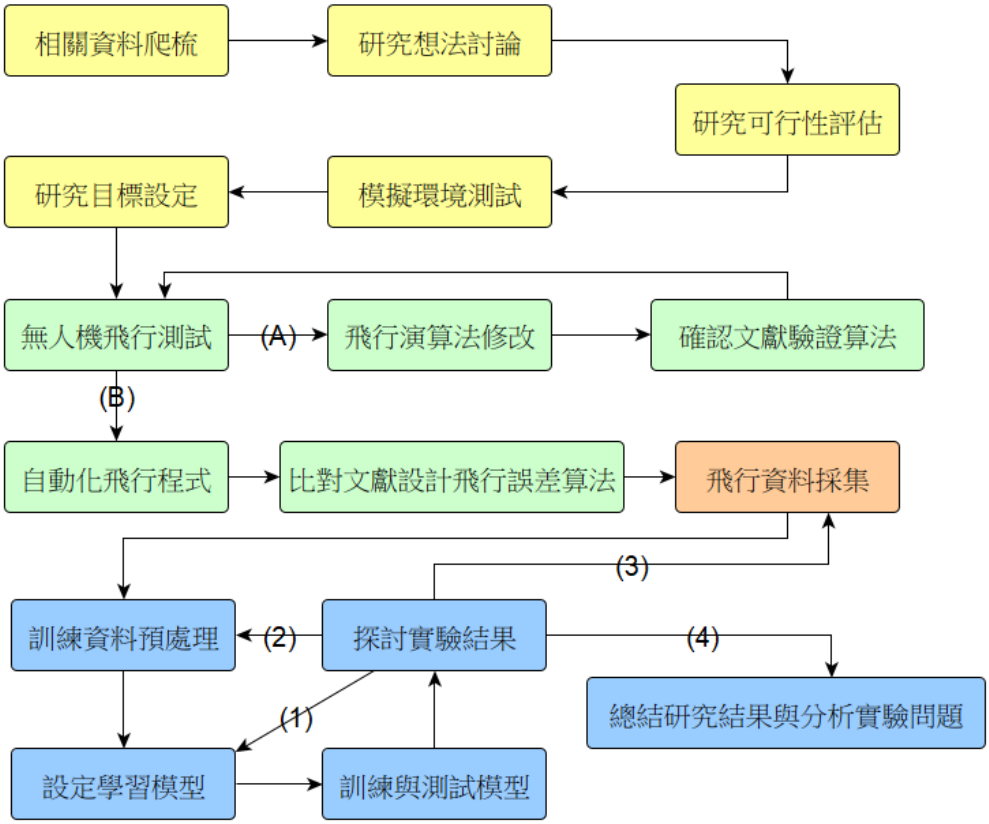



圖 1、研究流程圖

上圖為研究的大致流程圖，本研究開始前，首先查找了文獻與相關資訊，了解過去在相關學術論文或書籍中所提出的各種圖片處理方法、人工智慧模型原理，使我對此領域有較深入的認識，經過討論研究想法與可能的問題後，便決定研究實驗所需的程式並測試，最後設定目標，決定研究探討主軸與無人機的比較依據，然後開始進入研究階段，首先先設定模擬環境及無人機飛行程式，重複步驟(A)直到無人機能穩定飛行後，開始步驟(B)，接著訂定研究所使用的深度學習架構及資料採集方法，再開始製作訓練用的資料，並依序除錯、修正，測試效果並且對比，過程中依據測試之模型結果決定應該重新進行(1)或(2)步驟，如果有嚴重的辨別問題則走步驟(3)將模型重新設定並重新採集資料，在完成模型測試後，進行步驟(4)，以Webots模擬軟件中的一般無人機飛行軌跡為對象，比較其效果。

本研究主旨為以無人機鏡頭結合機器學習提高無人機之飛行定位能力，故研究進行時，大部分進行之主題多為機器學習與資料處理相關，其中又以機器學習為主要核心進行研究，探討不同作法之效果與以鏡頭增強無人機定位能力之可行性。

三、相關文獻探討

(一) 現今無人機概況

四軸無人機在生活中從軍事、工業到民生，無所不在，目前四軸無人機多採用旋翼兩正兩反的方式產生推力，並使無人機螺旋槳所產生的反作用力自相抵消，也能透過此反作用力轉動無人機，透過調整四顆馬達轉速，改變無人機平衡以產生無人機三軸 Pitch、Yaw、Roll 上不同方向的力，使無人機移動，如下圖：

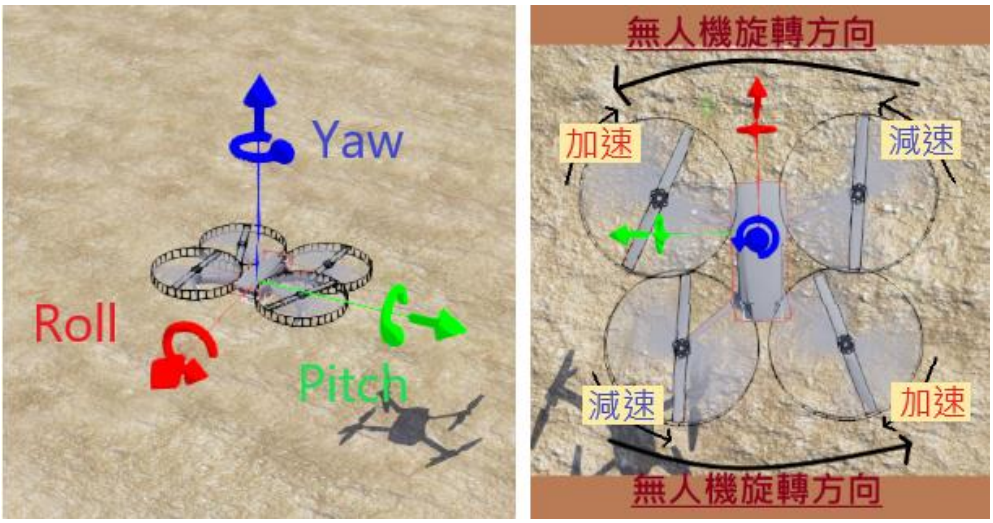



圖 2、無人機飛行控制

目前無人機也能透過 PID 控制器(Proportional-Integral-Derivative, PID)，將目標位置與目前位置的差轉換為無人機各個馬達所需轉速，以此增強無人機定位飛行時精準度及穩定性，再透過姿態算法綜合各項參數操縱無人機，在本研究中，將會在後續對於無人機飛行誤差狀況進行測試。

(二) 無人機定位誤差

無人機在一般情況下主要以 GPS 作為定位依據，然而，由於 GPS 信號在某些情況下受電磁波干擾、差分定位受高程異常影響，亦或是環境干擾電子羅盤等，會導致無人機定位出的值與真實位置有差異，而隨著受到的干擾程度越強，無人機的定位精確度也會隨之下降，在部分情況下，若使無人機飛行一原先訂好的飛行路線，若飛行路線與周遭障礙物距離不足，定位誤差甚至可能導致墜機等嚴重後果，故此研究打算以機器學習的方式增強無人機的定位能力，不過每種定位方法本身也都會

有適用的最佳情況或較差的情況，故此研究最後會比較本研究定位法之優缺點，若能與無人機常見的定位方法互相補足，則判定此方法有效，反之則無效。

(三) 無人機定位方法

1. RTK 定位法

為了能讓無人機在戶外移動時能及時定位到目前的位置，在外飛行時能夠仰賴全球衛星導航系統(Global Navigation Satellite System, GNSS)，並透過即時動態定位技術(Real-Time Kinematic, RTK)定位精度可達公分級別，屬於一種差分定位法，當無人機與基準站的距離越近，定位越精準，反之，當距離越遠則誤差會隨之增大。

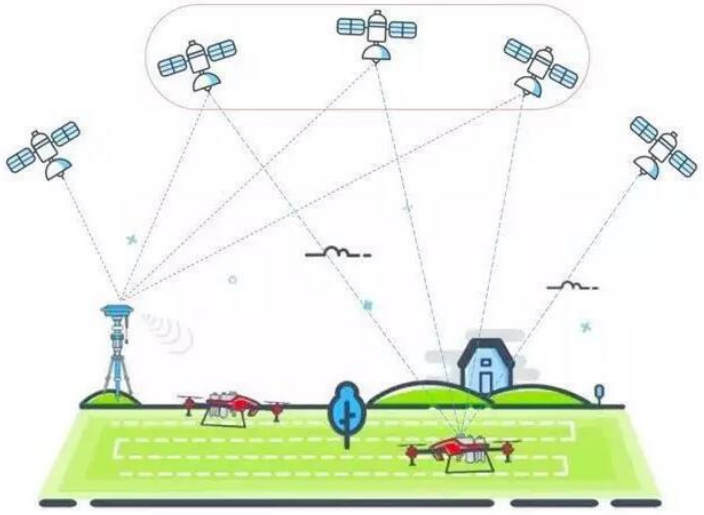


圖 3、RTK 定位法

如上圖，RTK 定位法透過多顆已知位置的衛星對無人機上訊號接收器的距離，以及一座基準站同時接收這些衛星訊號，再將基準站所接收到的數據發送至無人機，無人機利用這些數據完成差分計算，以此得知與基準站的相對位置關係，透過已知基準站的位置，完成定位。不過此定位法受限於對空通視環境影響衛星訊號、高程異常會導致計算時出現誤差等問題，容易因此受干擾，另外，若無人機處於強干擾環境附近，也會使無人定位出現偏移。

2. 光流定位與超音波定高法

在無人機飛行時定點有些位置可能因衛星訊號不佳等因素無法使用 RTK 定位法定位，這時部分類型無人機下方裝有攝影機用以採集影像資料，再利用光流演算法找到兩幀間像素點亮度的連續變化情形(如圖 4 所示)，得到兩圖的二維平移連續變化關係，並將此水平位移做修正。

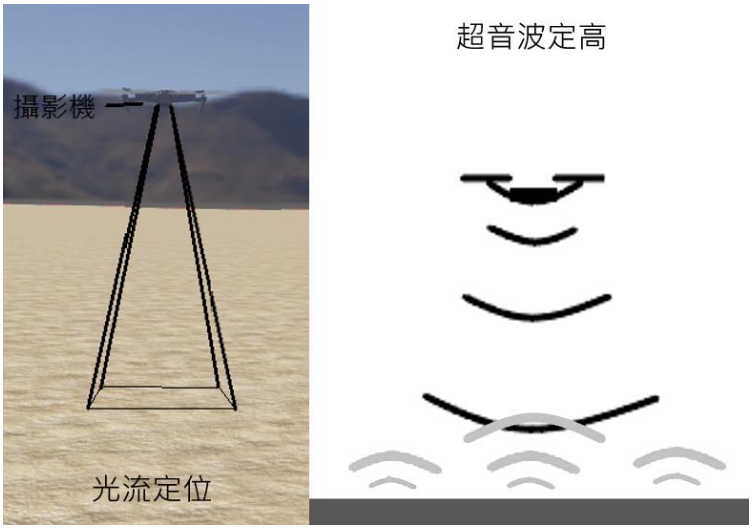



圖 4、光流定位與超音波定高

此種方法雖然能夠抓出無人機的連續位移資訊以此做定位依據，而其缺點為當物件為具反光性質之物件時，由於反光的映象會隨著無人機移動而改變，因此會對無人機之定位產生負面影響，並且由於超音波感測器與攝影機的靈敏度限制，通常以此方法定位時距地的高度限制為十五公尺以內，否則定位效果會大幅降低。

3. 慣性導航技術

這種方法是基於無人機的陀螺儀、加速度計的連續變化資訊做為定位依據，並以航位推測法算出目前的位置，然而陀螺儀、加速度計等感測器均有一定程度的誤差，而每次推算均是由前次推估的位置加上當前飛行狀態得出目前位置，故此方法會隨著飛行時間越久誤差會不停累加，此方法對於無人機內建的感測器靈敏度要求極高，故只有少數高空或航太級無人機會採用，且通常只有在不得已的情況下才會使用此方法定位。

(四) Webots 仿真模擬軟體

為了能實行此研究，我們有實作無人機與模擬無人機兩種方案，若採用前者，將會造成研究成本與研究複雜度太高，並且會耗費大量時間於非方法實作與探討，而把研究焦點置於無人機設計及平台整合，偏離了主要研究目標，故本研究採用模擬系統作為實作方案。

此外，為了控制成本與技術考量，採用了支援大量模擬功能並且完全免費的 Webots 模擬軟體，雖然此軟體性能調用之效率較差，且無法透過插件程式直接評估 AI 性能表現，但相比其他模擬軟體如 Gazebo，它較好取得且兼容性佳，同樣能夠設

定許多環境物理條件，如：風、重力大小和碰撞等現象，並能以常用的程式語言，如 C++與 Python 等程式語言控制機器人，其軟體畫面如圖 5 所示：

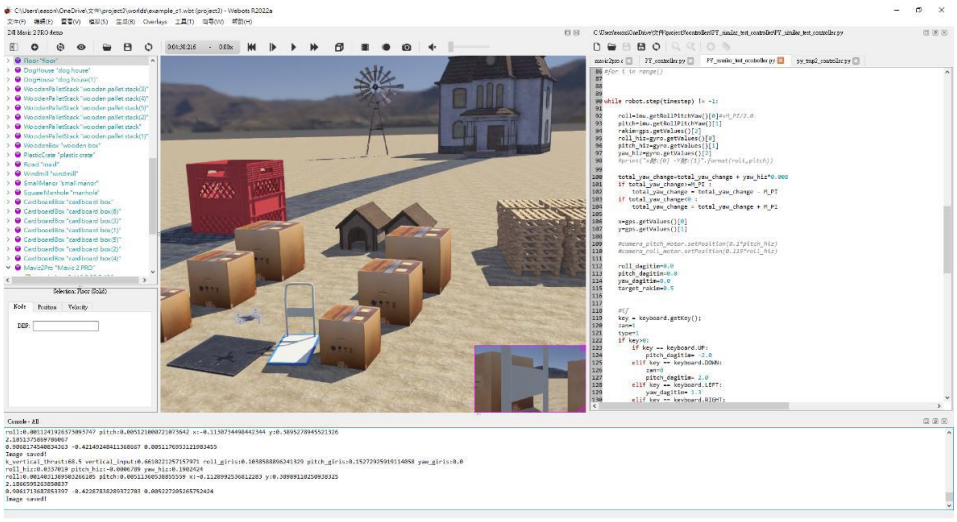


圖 5、Webots 仿真模擬器操作介面

Webots 仿真模擬器已經開源，故能使用許多額外插件、特殊選項，並且相容於機器人作業系統(Robot Operating System, ROS)等較大型的模擬系統，故此研究採用 Webots 作為無人機的飛行模擬程式，並以此軟體生成機器學習所使用的仿真圖像訓練資料。

(五) 卷積神經網路(Convolutional Neural Network, CNN)

為了能夠快速及有效率地處理資訊量較為龐大的圖像問題，卷積神經網路透過卷積層與池化層縮減圖像的資訊量並保留原圖像的重要特徵，最後進入全連結層經由運算得出分類或迴歸之結果，其架構如下圖所示：

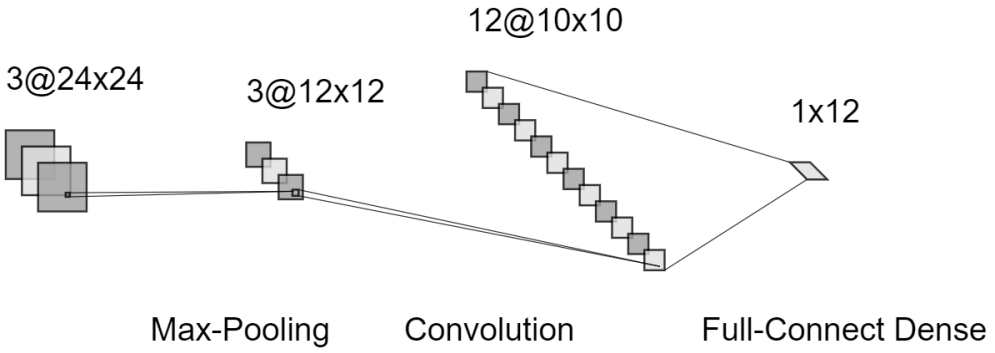



圖 6、卷積神經網路架構範例

透過卷積神經網路所得出的學習成果，當分類不多且有效訓練資料足夠時通常可以訓練出能有效處理圖像問題的神經網路，不過此研究將會嘗試以此卷積神經網路與深度神經網路結合後進行圖像迴歸，並經由測試後得出較佳的機器學習架構，再作為最終的模型放入模擬器中實驗。

(六) 資料迴歸

當輸入資料與輸出資料在特定維度上屬於線性關係時，使用資料迴歸能預判未知資料接下來可能的分布，在此研究中，由於無人機生成的圖像資料隨變化程度不同具有連續關係，而訓練後的模型期望功能也是用以判斷兩張連續圖片之差而得出移動或角度數值，若採用分類則可能因為資料量不足而導致無法訓練成功，故研究中的機器學習模型採用資料迴歸的方式處理問題。

(七) Roofline 模型與機器學習模型效能評估

在現實生活中，不論是甚麼平台對於深度學習模型都會有模型執行速度的限制，當一個具有強大預測能力的模型被放在性能孱弱的小型平台上，就可能導致模型運行過慢的問題，為此，Roofline 模型能透過模型計算量與訪存量，並與計算平台之運算能力、內存帶寬結合，計算出計算強度等資訊，得出模型運行時的理論效能上限，如下圖：

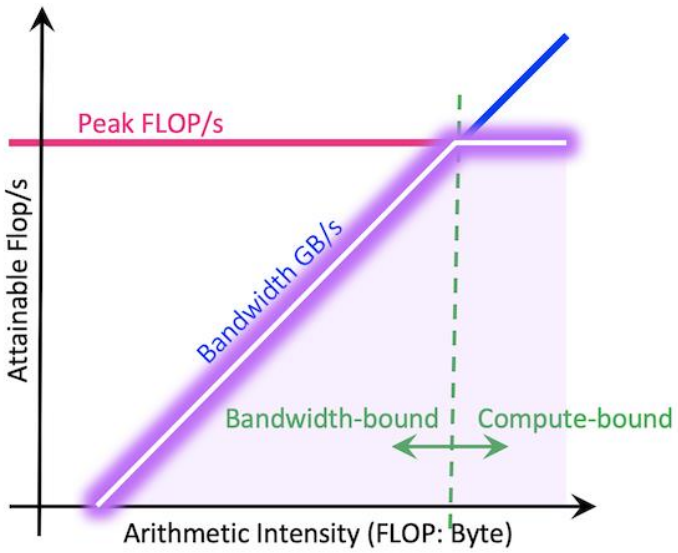



圖 7、Roofline 模型主要架構 (圖片取自 NERSC[7])

從 Roofline 模型可以得知模型的性能上限即平台所具有的計算能力，然而若平台所具有的內存帶寬不足，模型將會受限於帶寬，帶寬為圖 7 中的斜率部分，而帶寬瓶頸的模型將會停在平台最大帶寬與模型所需帶寬的斜率部分，達到效能表現的上限。

為了能評估本研究之方法撇除模型本身之設計問題，還須兼顧能放在無人機上即時運算，本研究將透過 Roofline 模型原理評估本實驗方法運作在無人機上之效率與可行性。

四、研究流程

(一) Webots 無人機基礎模擬環境設定

本研究主要是想透過無人所搭載之鏡頭影像經機器學習後作為無人機的定位增強依據，為此，首先要蒐集可供模型作為學習依據之飛行時的影像資料及飛行位置資料，此研究先在 Webots 預設情況下導入 DJI Mavic 2 Pro 無人機，此無人機為許多媒體及無人機愛好者所採用，而模擬程式中內建之無人機，其最快移動速度及最快自旋速度，也都與真實世界中的 DJI Mavic 2 Pro 無人機相同，故本研究直接採用此無人機作為實驗對象。

首先，使用內建之物體模型如：箱子、拖車、房子、風車、道路，製作相對於城市或鄉村等地形，較為簡易的無人機飛行場地(如圖 8 所示)，背景則開啟預設的藍天與山，將地圖簡化目的是先以簡單的飛行地圖測試並採集資料、訓練模型，使資料採集過程較為快速且較少場景也能降低模擬器負擔、加快模擬速度。


圖 8、無人機飛行地圖設計

接著，開始撰寫無人機之控制程式，為了之後能夠較容易與廣泛使用 Python 語言作為設計語言的 TensorFlow 訓練完成之模型結合，研究時將控制無人機飛行之程式也同樣使用 Python 撰寫，雖然執行效率不及環境測試範例的 C++版本，但是能較容易進行開發，其中無人機的平衡控制參數，按照 Webots 文獻[9]之性能設定，較詳細的作法細節請見附錄一。

(二) 無人機定位誤差算法編寫

在飛行誤差模擬方面，本研究依照 DJI 技術參數紀載[10]之 Mavic 2 Pro 定位能力，其誤差平均 1.5 公尺，我們設計一個 GPS 誤差的函式來模擬真實環境的定位誤差，此函式是使用時間為種子的隨機取值函式，其取隨機值之範圍為正負 1.5 公尺內，並透過設定基礎偏差值為 1.5，最後得出以下 Random 範圍限制公式(1)：

$$GPS_{Noise} = \left(\frac{\pm 1.5}{1000/TimeStep} \pm 1.5\right)\ \text{(單位：公尺)} \quad (1)$$

運算式(1)中 TimeStep 其值代表模擬之基本單位時間，預設最小為 1，單位為毫秒，這樣設定代表模擬器將以無人機世界時間，每 TimeStep 毫秒模擬一次地圖(包括感測器資訊更新)，最後得出的 GPS_Noise 代表每次取 GPS 值時的誤差值，本研究中 TimeStep 設為 8，代表每一秒會讀取 GPS 資訊 125 次，每次間隔均為 8 毫秒，而每次模擬即使用此公式算一次誤差，模擬此無人機真實運行情況，對於 TimeStep 更詳細的補充請見附錄三。

(三) 無人機飛行定位算法設定

為了能使無人機飛行到任意指定位置，我們使用了 PID 進行無人機控制，PID 的三項參數分別代表移動比例(Kp)、積分增益(Ki)與微分增益(Kd)，調整此三項數值能改變無人機馬達的靈敏度及旋轉反應幅度，若 Kp 太大會導致無人機反應過大，可能使無人機無法精確到達目標位置，太小則會使無人機移動過慢，而 Ki 值過大則會使無人機產生震盪，太低則會導致無人機移動或旋轉遲鈍，而 Kd 值可以減少移動過衝的問題，需要注意的是，在本研究中的模擬系統無人機之 PID 並非為模擬控制馬達電流等物理上的控制，而是透過改變馬達轉速影響無人機飛行。

最後，透過幾次 Kp、Ki、Kd 值的搜索，找到能使無人機穩定移動至定點的數值，將無人機的三軸及旋轉控制的 Kp、Ki、Kd 值設定如下表：

| | **Kp** | **Ki** | **Kd** |
|---|---|---|---|
| **PitchPID** | 1 | 0.1 | 1 |
| **RollPID** | 2 | 0.1 | 2 |
| **YawPID** | 2 | 0.1 | 2 |
| **ThrottlePID** | 10 | 0.1 | 5 |

表 1、PID 調校值設定

接著，只需將此數值套用至模擬系統中，並將程式精簡化，即能快速地指定無人機飛行目標位置，且無人機也能精準地到達目標位置(目前的 PID 能使無人機在沒有定位誤差的狀況下，以誤差小於 0.1 公分的性能到達目標位置)，詳細作法請見附錄二。

(四) 飛行資料採集

在本研究中，為了對比無人機飛行於景色細節較多與較少時，高空定位能力的差異，本研究除了在飛行路徑結果針對飛行於不同位置時的路徑做比較，也嘗試採用均為景色細節較多的圖片與景色細節多寡混雜的圖片作為訓練集，並比較其訓練

結果。

模擬無人機在一般拍攝時，不一定都在房屋等物件密集的區域飛行，為了生成細節數量多寡混雜的圖片訓練集，我們採用人為操作模擬器內的無人機，令其持續飛行 4 分 30 秒，過程中飛行範圍控制在物件密集區之 1.5 公尺範圍，使無人機能有部分圖像拍攝到物件，而部分圖像僅拍攝模擬器之世界背景，資料採集的飛行範圍為下圖橘色線條圈起的部分(由於是人工操作無人機，故飛行範圍也是大略手繪)：

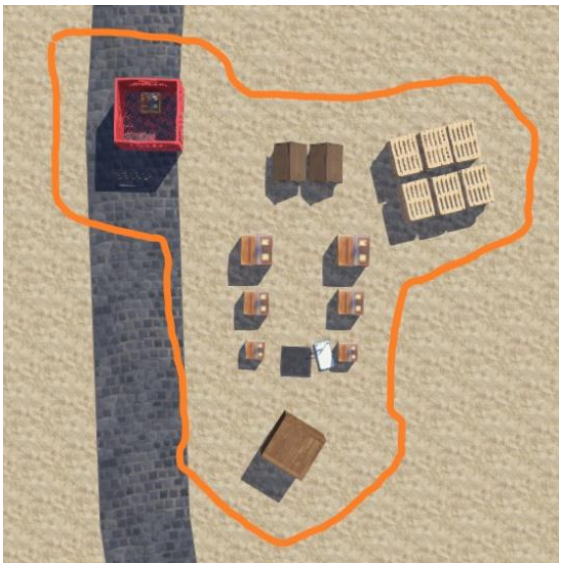


圖 9、手動操作無人機飛行時的飛行範圍

至於資料密集區的採集，則透過隨機飛向圖 10 之橘色方框圈起範圍內之任意點，總共飛行 4 分 45 秒後結束採集，雖然比前次採集之時間多了約 15 秒，不過後面訓練時會取固定數量的資料做訓練以確保兩者訓練資料量相同，因此影響不大，採集範圍圖如下：

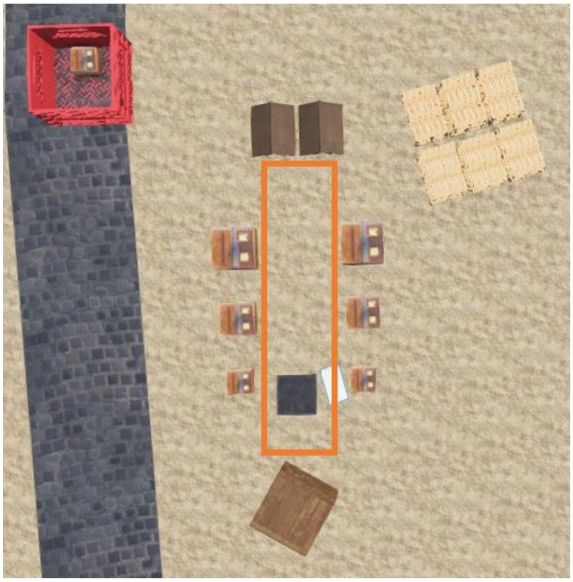


圖 10、無人機自動隨機飛行時的飛行範圍

透過上述兩種方法，採集無人機以 TimeStep 速度模擬並保存每次所拍攝的原始圖片，再於另一文字檔存下對應照片所在位置、角度資訊，以備後續方便調整資料預處理方法及深度學習模型的輸入圖像格式，在本研究中，圖像的尺寸為 240*400，而圖像資料的採集程式做法則放在附錄三。

(五) 資料分類與預處理

為了能快速地標記資料，我將生成出的資料透過改變角度的不同作為分類準則，以精度為小數點後四位的資料開始分類，取小數點後四位作為最大精度的原因是此無人機依照現實中的 DJI Mavic 2 Pro 無人機技術參數進行設定，而採用 Webots 中小數點後四位相當於最小可以測量至 0.1 毫米的移動，相當於最高精度比 GPS 每次採樣最大誤差的 1.2 公分精細度高了 120 倍，使模型的性能上限有機會超越 GPS 定位，同時減少位數能避免資料類別過多，進而導致各類別資料不足造成訓練不易。

程式部份則透過 OpenCV 及 NumPy 函式庫，將圖像轉成矩陣後做運算再將最終資料存到該類別應屬的資料夾，若尚無該資料夾則自動建立，因此最後生成的 DataSet (訓練用資料)將會是具有各種無人機飛行資料且沒有多餘空資料夾的資料。此研究中將無人機主鏡頭所得之影像，與鏡頭前一次採樣所得之影像做相減，為了能有效看出圖像間的差異訊息，研究時採集之訓練圖像，均會轉為灰階圖像，再進行差異操作，最後透過式(2)生成每個像素值：

$$\boldsymbol{V_{Pixel}} = \left(\frac{\textit{當前圖像之像素}}{\mathbf{2}} + \mathbf{127}\right) - \left(\frac{\textit{前張圖像之像素}}{\mathbf{2}}\right) \quad \textbf{(2)}$$

運算式(2)中得到的$\boldsymbol{V_{Pixel}}$將會依像素逐位進行，功能為使因相加或相減後超出溢位的值限定在 0~255 的範圍中以加強圖像變化之特徵，同時避免溢位造成圖像中部份信息丟失，目的為使學習模型能更有效地抓出圖像特徵。

最終，依序將每張連續圖片依照圖像生成公式處理後，再透過 OpenCV 的 imwrite 寫入資料，完成所有訓練資料的標記及分類，本研究一共生成了約 30000 張圖像，如下表：

| 類別 | 手動控制無人機產生之資料 | 無人機自動飛行產生之資料 |
|---|---|---|
| **資料夾數量** | 216 | 166 |
| **總檔案數量** | 32859 | 36741 |

表 2、訓練資料數量

可以從上表發現，無人機自動飛行時雖然資料量較多，但類別卻較少，在此推斷無人機由於自動飛行程式限制了角加速度的上限，故得出此結果，但也能同時得

知此資料符合無人機自動飛行時的真實情況，並推斷使用此類別資料訓練的模型表現會較佳。

在模型訓練完成後，依據模型的判斷能力與原先 GPS 做定位修正，不過預設目標是只有在 GPS 訊號不佳時或是才會啟用此方法輔助定位，當使用訓練後的機器學習模型輔助定位時，將會透過式(3)對無人機之定位進行修正：

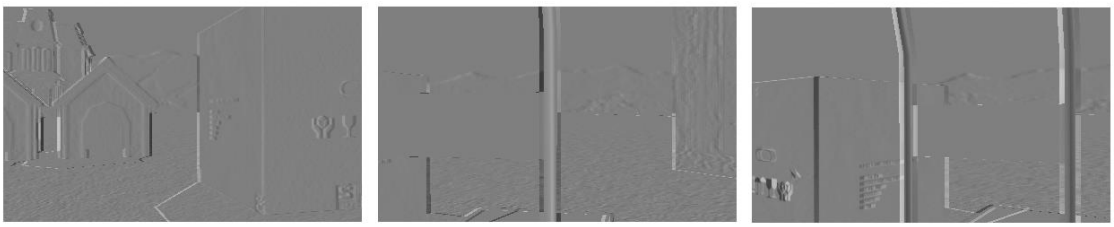
$$\boldsymbol{V_{Final}} = \frac{\textit{無人機定位精確度} \times \textit{無人機定位結果} + \textit{模型定位精確度} \times \textit{模型定位結果}}{\textit{無人機定位精確度} + \textit{模型定位精確度}} \quad \textbf{(3)}$$

運算式(3)中$\boldsymbol{V_{Final}}$是最終定位結果，其中包含無人機之位置及方向資訊，此式是為了確保學習模型能達到對無人機偏差修正的功能，並維持定位系統的綜合穩定性，使無人機每次定位時均有模型加入無人機輔助判斷，達到增強定位準確度的目的，式中透過將兩者的定位準確度之比例做為無人機的定位結果，能確保無人機經由模型校正後平均定位能力最少不會變差，而在實際用於真實世界時，此模型主要是用在 GPS 訊號不佳時啟用，因此理論上能做到比不套入模型時好。

(六) 評估訓練資料

訓練模型前，我們先確認了生成之資料是否與當初預想的相符，雖然透過人眼觀察不一定能看出特徵，但至少能確認資料分布及特徵是跟預想的一樣。

從先前生成的資料中，隨機選擇了資料夾 60、80 及 100 中的圖片作分析，原因是此研究目前嘗試的是以迴歸方式訓練機器學習模型解決問題，選擇連續且等距的資料較容易看出輸入與結果的關係是否屬於連續變化，若有則較為可能訓練成功，若無則代表在當前維度上，資料可能屬於離散關係，但是由於神經網路是做多維度的運算，因此若無法以肉眼看出連續關係，並不代表不可能學習成功，只是若能以肉眼看出連續關係，可猜測訓練成功的機率會越高。



資料夾60之圖片　資料夾80之圖片　資料夾100之圖片

圖 11、三種不同類別資料圖

如上圖，比較三種不同資料，三張圖片在橫向的顏色變化層大小屬於連續關係，類別 60 的圖片變化層最小，80 的中等，100 的最大，比較橫向是因為此數字所屬的資料類別代表水平旋轉的程度，故比較其橫向的顏色變化層。這裡資料類別的數字

名稱代表資料之連續變化，而顏色變化層大小在本研究指的是兩張連續圖片重疊時顏色圖層重疊的部分，如下圖：

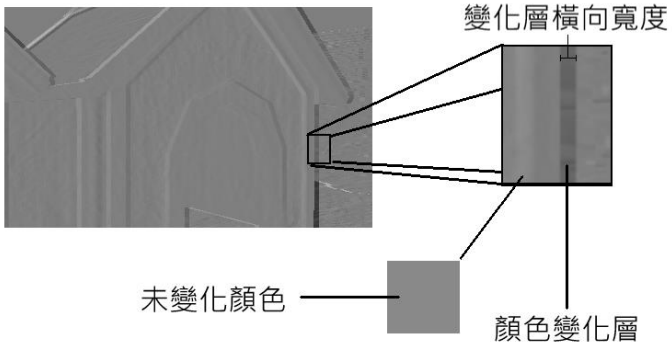



圖 12、顏色變化層

雖然大部分資料都屬於上述有顏色變化層的圖片，不過當無人機面朝地圖外時，由於背景是固定底圖，且顏色很淡，造成無人機不論角度轉動多少，拍攝的圖片都極為接近純灰色圖(如圖 13)，最終導致當無人機若無法拍到障礙物，所得出之圖片都十分相像，不過此資料僅出現於手動飛行模擬一般飛行情況所生成的資料，在由電腦飛行於中心區域的資料中，並無此類型之資料分布，符合預期。

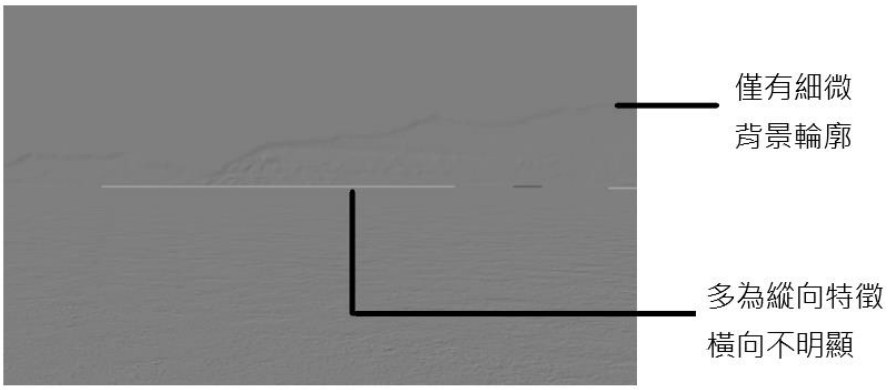



圖 13、不利於模型學習的訓練資料

(七) 機器學習模型設計

在設計用於無人機之 ML 模型時，訓練無人機在無人機上的機器學習必須注意 ML 算法要能在無人機上及時運行，研究中為了能兼顧學習模型訓練成效，同時避免學習模型過於複雜導致無法在無人機運行，或造成過擬合現象，此研究依據[5]研究中所提，同樣採用輸入神經元數加輸出神經元數開根號，隱藏層神經元 876 個，並設定其具有兩層隱藏層，使模型能夠表現更多種邏輯或推理關係，後續實驗中也會測試不同隱藏層節點數之模型，比較是否有可能透過改變節點數修改模型以增強模型之預測能力或優化對於平台運算性能之要求，並在完成實驗與訓練後，透過 RoofLine 模型的深度學習模型執行效率估算方法，對當前訓練之模型進行評估是否可能在現今無人機運行及其效率表現如何。

在本研究中，如果卷積層的部分直接採用 VGG16 等經典物體識別模型的設計，除了此結構不一定能識別這種長條型的特徵外，龐大的模型也對無人機運算量造成不小的負擔，透過參考 VGG 架構中採用多個 3*3 卷積核取代少而大的卷積核以提升感知視野，在本研究中主要的線條等較細緻的圖形特徵可能也可以透過這種策略保留更完整的資訊，而且本研究中的圖片多數僅部分具有線條特徵，其餘多為灰底，為此，我們採用了透過多次的池化層減少參數量以降低運算量，同時透過增加特徵圖(Feature Map)數量的方式避免重要圖像資訊在進行多次的池化時遺失，並使用回歸的方式作為輸出結果，透過上述方法使用了以下參數來訓練模型:

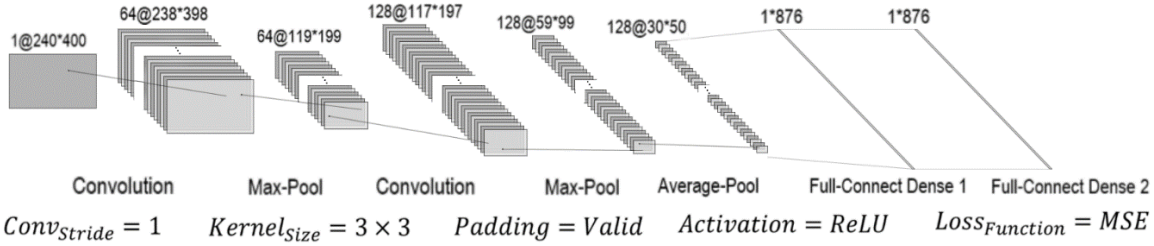



圖 14、本研究使用的機器學習模型

上圖中，過濾器數量及池化層的擺放除了透過前述想法進行設計外，也是基於多次的訓練嘗試後，從各個架構之模型訓練完時的損失函數為依據，找出表現相對較佳的組別，成為最終的模型，而模型設計時的詳細程式做法請見附錄四。

# 參、研究結果與討論

一、在 Webots 模擬中無人機所飛行之理想路徑與誤差路徑比較

為了能確認無人機飛行時的誤差情況，本研究在開始進行模型測試及後續研究前，利用完成的 GPS 及電子羅盤誤差函數並設計路徑使無人機飛行，最後，無人機的飛行結果如下圖所示：

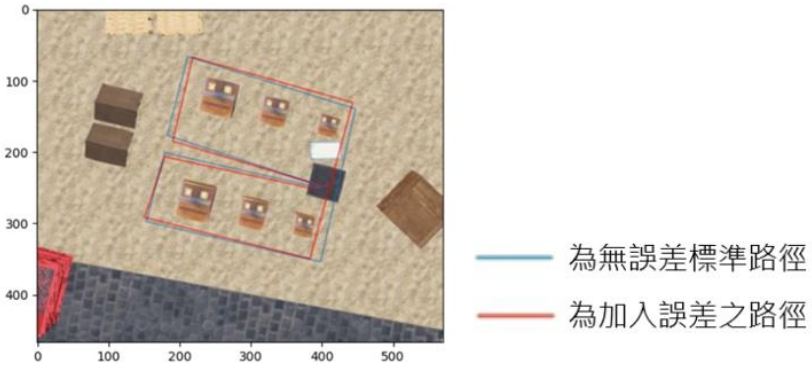



圖 15、定位有誤差情況下的無人機飛行路徑圖

由上圖可發現無人機在飛行時確實會因為感測器誤差而導致飛行路徑與預期有所偏

差，不過由於 GPS 取得之值為絕對位置，因此無人機到各點的誤差大多都還在標準路徑之一定範圍內。

圖 15 為透過無人機定位至定點後再讓程式傳回真正的當前無人機位置所繪的圖片，因此路徑看起來簡潔俐落，然而若將無人機在飛行時每一刻所記錄的位置全部加入圖後，由於程式碼為了能準確定位採用了 PID 控制，且 GPS 感測器並非在無人機正中心，因此會出現許多無人機轉彎時及直線移動微調時的路徑，半圓狀為無人機轉彎，而直線則為無人機進行位移微調時的路徑，如圖 16 所示：

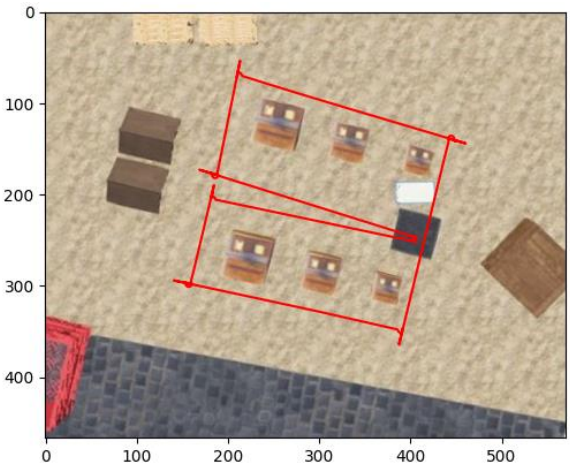



圖 16、無人機飛行時的完全路徑

在本研究中，後續的無人機定位結果將會使用如圖 15 的路徑表示法，原因是採用完整路徑會導致較難辨別無人機到定點時的位置，並且相比於第一種表示法，完整路徑並未提供特別的優勢，我們的研究核心，在於當無人機到達目標點時的位置與正確位置之誤差關係，與無人機到達目標點前之移動路徑較無關係，且多餘之微調路徑為 PID 調校問題，當前 PID 控制器已經能使無人機最終以 0.1 公分之內的誤差抵達任意點，而修正過程不影響主要實驗結果，故不多做更進一步的優化。

二、深度學習模型之訓練結果

全程訓練了約 1.5 個小時後，手動生成資料集的訓練程式在訓練第 89 次迭代後停止了訓練，最後結束時損失函數從一開始的 1261 降低為 146，驗證集的損失函數則從 1239 降低為 129，得出的 RMSE 值約為 10.15，遠低於基準模型的 35.17，且 R2 Score 也達到了 0.916，可得知模型對於將圖像做線性回歸的結果為有效，不過泛化性及其他性能則需後續實驗得出結果，同時，為了確認第一次實驗之準確性，本研究又以相同資料再度進行了二次訓練，而訓練完成後，各項訓練結果指標皆與前次差不多，故認定此模型能有效收斂，結果如下表：

| 收斂結果 | 第一次 | 第二次 | 第三次 |
|---|---|---|---|
| **Loss** | 146 | 131 | 137 |
| **Val_Loss** | 129 | 119 | 131 |
| **RMSE** | 10.15 | 9.58 | 10.21 |
| **R^2 Score** | 0.916 | 0.925 | 0.913 |

表 3、模型之收斂結果

由於模型中每單位是 0.0001 弧度，將前者的 RMSE 換算成角度約為 0.06 度，相當於每次進行判斷時會有約 0.06 度的誤差，將 0.06 度的誤差乘以每秒 125 次預測，相當於平均每秒產生約 7.2 度的誤差，雖然高於 DJI Mavic 2 Pro 電子羅盤的平均 6 度，且低於陀螺儀之精確度，不過已經超越一般千元入門級無人機的平均 12 度誤差了，考量到此模型採用了各種資料混雜的訓練集，有可能出現圖像幾乎不變卻角度轉變很大的情況(如拍攝到無障礙物之背景)，此模型 RMSE 成績也包含了與預期不符的資料，因此尚無法判斷模型之實用性，實驗至此尚無法下定論。

除了前述結果外，基於自動飛行於地圖中心生成資料所訓練的模型，其 RMSE 也下降到了 8.61，相比前者模型表現又更進一步，而 R2 Score 則高達 0.936，模型收斂成功，若將第二種模型 RMSE 換算成平均每秒角度誤差則達到了優秀的每秒 6.16 度，雖然相比於電子陀螺儀尚有段距離，但是已經高於電子羅盤之平均誤差了，理論上能夠為實際應用於無人機時帶來不錯的效果。

雖然經由上述數據可推斷模型之訓練成效良好，不過若是模型損失函數下降時跳動幅度太大或不合理，也可能表示模型訓練具有異常，因此，我將模型訓練時每筆 Epoch 之損失函數及驗證集損失函數畫成下圖：

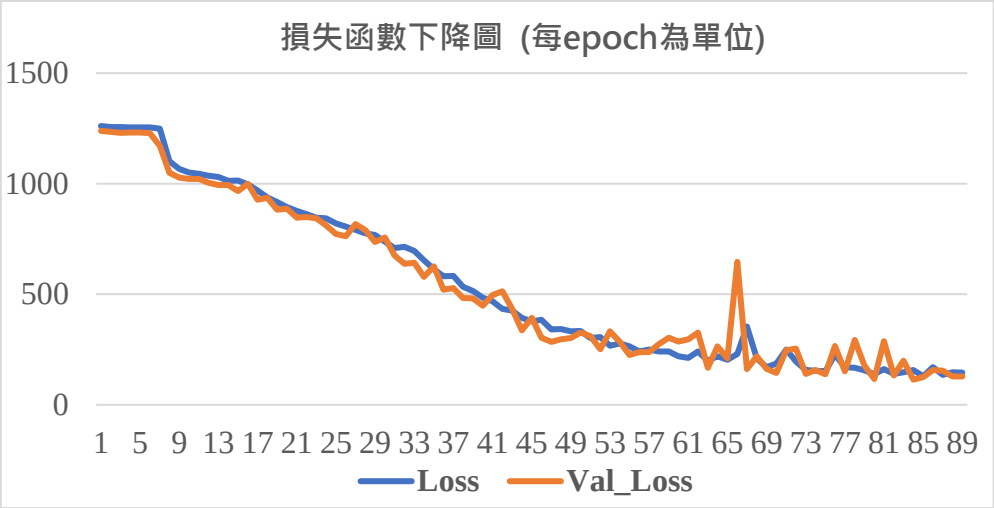



圖 17、損失函數下降圖

由上圖可以得知，此模型到訓練後期驗證集震動起伏較大，可推斷模型對於特定資料產生瓶頸，抑或是模型設計不夠完善，可能導致模型容易受某些特徵大幅影響輸出，至此，可以對學習模型得出初步結果：

(一) 模型能有效收斂，且其最終之 RMSE 表現不錯。

(二) 模型或訓練資料可能存在部分錯誤或具有不佳的資料與結構，尚需模擬實驗測試模型之真實表現。

三、訓練完成的模型之資料預測能力

為了驗證模型對於生成資料之預測能力，撇除模型設計時預先準備不放入訓練的測試集資料，本研究也額外針對先前進行無人機誤差路徑實驗時，無人機採集之飛行資料，將其套入模型中進行預測，得出各類別每張照片的預測之平均誤差值，結果整理如下圖所示：

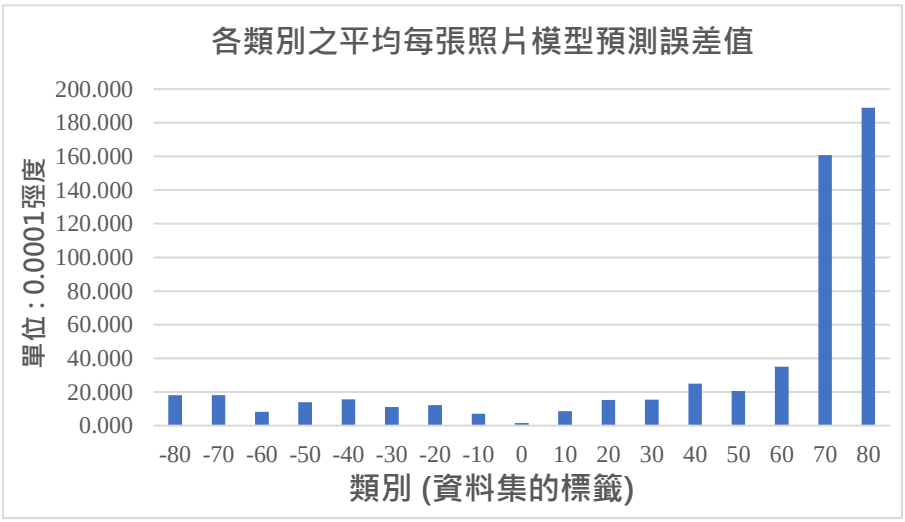



圖 18、各類別之平均每張照片模型預測誤差值圖

由上表可以發現，模型在多數情況下預測圖像之誤差均小於 20，不過在預測類別 70 與 80 時，誤差急遽上升，其誤差約為正確答案之值的兩倍以上，然而，當我將此二類別之圖像獨立放入模型中並比較其結果時，發現模型似乎將此二類別的正負預測完全顛倒，以正解為 80 之圖像為例，模型平均將其預測為-86.937，本研究判斷此二類別是導致模型最後損失函數抖動的原因，因此，我又設計了一程式對模型輸出資料進行評估，算出模型輸出正負相反的圖像在各類別所佔的比例，得出如下結果：

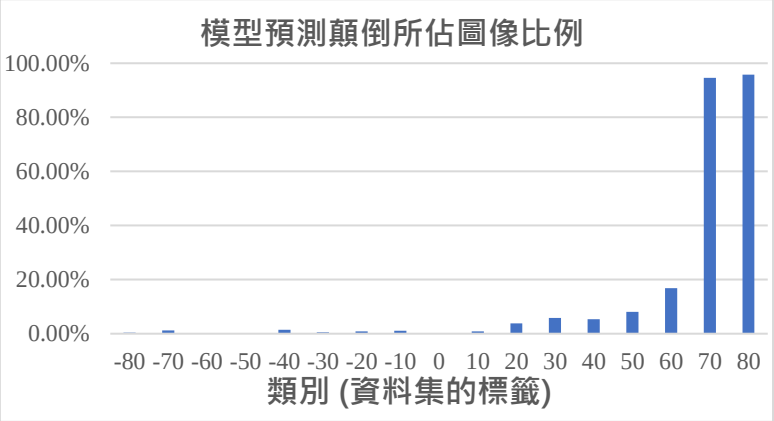



圖 19、模型預測各類別正負顛倒所佔圖像比例

接著，我又檢查了圖像資料分布情況，最終找出原因，由於訓練之資料為無人機自動生成，而無人機自動飛行時由於程式設定因素，大多是優先向左(負數方向)旋轉，因此生成之資料大多都帶負數標記，多數情況下，快速向右移動占比相較於向右微調情況又更少，而模型設計時並未將各類資料的權重占比平均，導致模型對於預測快速向右旋轉之圖像資料，相較於將資料預測為正數，更傾向預測為負數，下圖為各類別之訓練資料在-80~80 十七個類別(由於總類別數量太多，故以 10 為單位取類別方便製圖)的占比：

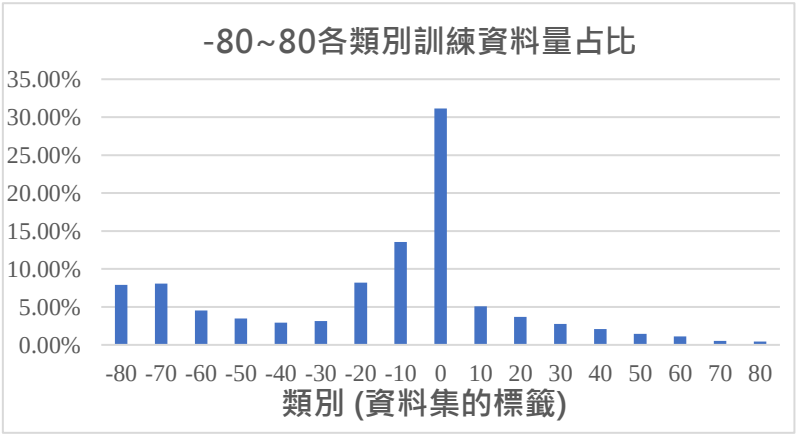



圖 20、-80~80 各類別訓練資料量占比圖

雖然模型對於單張照片預測對於部分資料存在正負相反的問題，不過模型在針對無人機修正時是取旋轉時整段時間之資料，而無人機本身在旋轉時均為向左，只有當與目標座標很接近時才會使用少量的向右移動，而模型若不考慮向右移動之情況，單純向左移動時每次採樣圖像之平均誤差僅為 7.88，因此，本研究繼續進行下一步的無人機飛行路徑與平均飛行準確度之實驗，以驗證模型對於當前飛行系統飛行準確度是否有益。

## 四、套入模型後之無人機飛行路徑

本研究設計了兩種路線給予無人機飛行，一種為將飛行路徑全部限縮於障礙物之內，如圖 21-(2)之移動路徑，其中座標為隨機生成，另一種則為使無人機針對整張地圖環繞

一圈後回到原點，如圖 21-(1)之移動路徑，而為了防止隨機路徑經過障礙物或過於接近障礙物，因此圖 21-(1)之路徑座標為人工設定，如下：

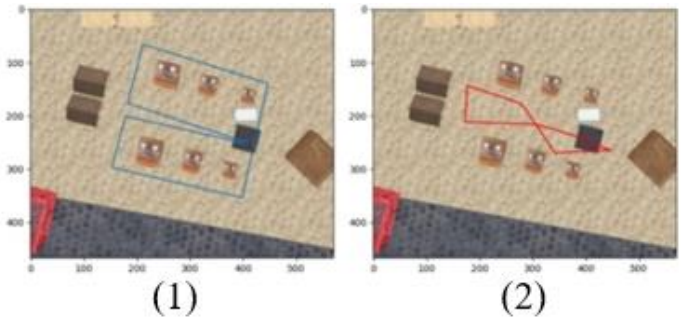



圖 21、設計之無人機飛行路徑

接著，使用每次無人機飛行系統認為到達目標點時，與無人機實際與目標點之距離作為單次飛行誤差，並透過飛行整個路徑時的多個誤差結果取平均，得到單趟飛行的平均誤差，舉例來說：無人機必須飛行至 B 點及 C 點，當無人機認為已到達 B 點時，實際與 B 點的準確位置相距 1.5 公尺，則單次飛行誤差為 1.5 公尺，並假設 C 點飛行時的單次飛行誤差為 2.5 公尺，則可算出無人機單趟飛行平均誤差為 2.0 公尺。

雖然本研究未將無人機飛行過程之路徑也納入無人機飛行誤差的統計中，不過目前所設計之 PID 飛行系統能在感測器無誤差的前提準確到達任意點，因此，使用此方法用於測試無人機飛行誤差結果具有可信度。在後續實驗開始前，本研究先對當前 PID 控制無誤差版無人機之飛行表現進行測試，無人機在未套入誤差與模型校正前，由於並無其他干擾條件，因此平均誤差僅 0.01 公分不到，產生自程式中函數對浮點數多次轉換產生的精度誤差，由於其值單位太小對結果影響極小，因此不特別進行排除。

最後進行無人機飛行路徑實驗，首先是無人機飛行於地圖中心部分的路徑，飛行路線及其誤差值如下圖：

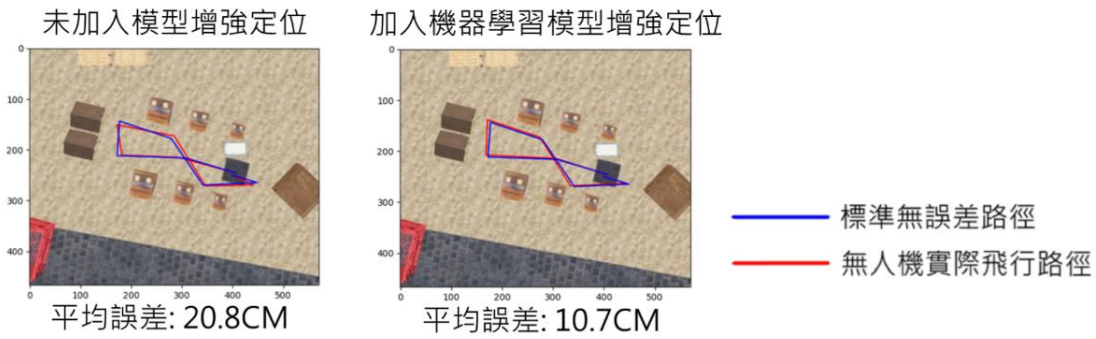



圖 22、無人機飛行於地圖內部之飛行路徑

由上圖可以發現，無人機即便處於理論上誤差會較低的低速模式飛行，仍與實際預期之軌跡產生了肉眼明顯可見之一定程度的誤差，然而將模型套入無人機後進行飛行測

試後，飛行誤差顯著下降。無人機在飛行於地圖中央時，透過模型輔助定位，成功使無人機之飛行路徑平均誤差下降，其中並無出現當初測試模型時，向右旋轉產生較大程度誤差而導致的明顯路徑誤差，從此得知先前對模型評估結果的顧慮確實可以放心，透過模擬並實時監控模型輸出狀況，可以發現雖然有時會忽然出現幾筆正負相反的資料，不過馬上又會被數十筆較正確的資料蓋過去，由於無人機角度判斷是由多筆模型輸出結果平均後得出，因此少部分的錯誤對無人機實際表現影響不大。

然而，當無人機繞行地圖飛行時，不加入模型時與先前飛行於地圖中心時表現差不多，不過加入機器學習模型的表現則與前者大相逕庭，其飛行路徑與平均誤差結果如下：

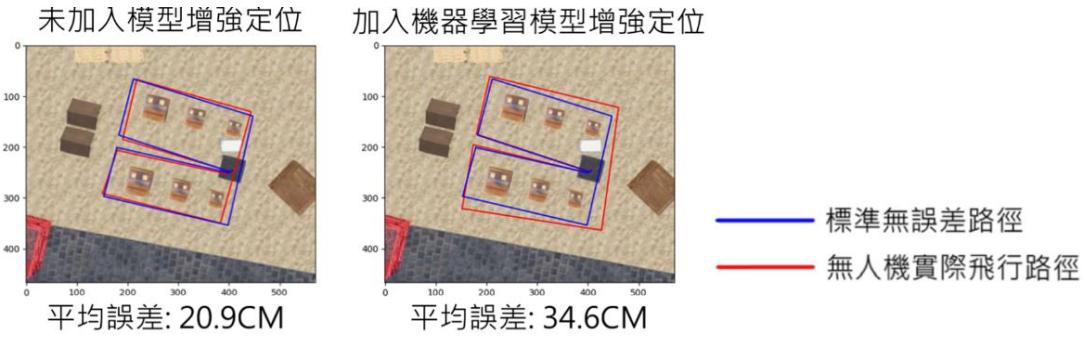



圖 23、無人機繞行於地圖外部之飛行路徑

當無人機在飛行於地圖外圍時，無人機飛行之路徑平均誤差上升至 34.6 CM，準確度遠低於未加入模型的表現，不過由路徑圖可以發現，加入機器學習模型的無人機在飛行於障礙物內時準確度表現高於未加入模型，由此得知模型確實能在特定情形增強無人機定位能力。

定位技術對於無人機而言具有舉足輕重的地位，一套定位方法在實際應用前需要經過多項基準測試才能合格，而目前的研究僅針對了一種小型地圖進行測試，但還有許多需要驗證的問題，如：模型撇除對於地形複雜度之要求，是否有其他情況也可能有問題需要加強。故本研究未來將會朝向多種地理環境甚至加入天氣等變因進行測試，以驗證此方法在真實世界的各種情況下是否合格。

五、設定 ML 模型啟用門檻後的無人機定位表現

有鑑於前項實驗中無人機飛行於有障礙物環境時，其飛行準確度確實獲得了良好的提升，然而無法拍攝到障礙物時，準確度大幅下降，我們推測其受機器學習模型判斷影響，因此，**我們又嘗試以連續兩個影格中像素點相減的平均變化量為門檻**，作為啟用機器學習模型的依據，使用像素平均變化量的原因，是本研究的模型只有在針對足夠複雜的畫面變化時，才有比較好的判斷能力，在經由相減的圖像中，若一個點並未改變，則其值為 127(根據第 14 頁式(2))，能夠從預處理後的圖片中，各個像素與 127 相差的絕對

值之平均，推斷此圖像所含帶的資訊量。

本實驗透過測試先前標註的四百多張無人機高速移動且無法拍攝到障礙物時生成的圖像，以這些圖片的像素點的變化量作為機器學習模型是否啟用的判斷門檻，在此類的圖像中，圖像的平均像素點變化量約在 1.75 內，此值為一個閾值，此值可能隨著不同的無人機飛行環境有所變動，故後續進行定位實驗時將以圖像中像素之平均變化差是否達到 1.75 作為啟用模型的依據。

透過程式大量將先前生成的圖片平均像素變化量不足 1.75 的圖片篩出，我們發現除了有單純未拍攝到障礙物以及無人機緩慢轉動微調而導致特徵太少的圖像外，也有一些具有長條特徵但是因具有特徵的部位不足而被篩掉的圖像，這表示此方法可能導致一些具有部分可用資訊的圖像被浪費掉，但是此實驗之目的為使無人機定位表現不要出現過度要求環境而導致泛用性不佳的情況，在未來時間更充裕後，會嘗試使用不同方法作為啟用門檻測試，不過目前由於時間因素，我們選擇犧牲掉部分資訊以嘗試換取更高的泛用性，最終，透過增加模型啟用門檻並進行模擬飛行實驗，取得了以下結果:

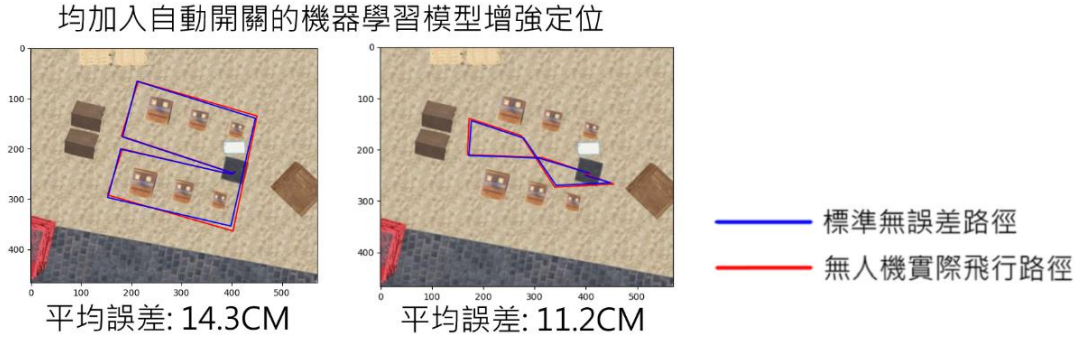



圖 24、加入機器學習模型啟用門檻後的無人機飛行路徑

由上圖可以發現雖然無人機在針對飛行於障礙物內時，其定位能力相比未加入啟用門檻，其平均誤差值略微增加了 0.5 公分，但是此舉使得無人機即便環繞地圖飛行，有部分位置無法拍攝到障礙物資訊，由於定位時不會受沒有充足辨別能力的機器學習模型影響，其定位能力相比最原始的 20.9 公分依舊獲得了顯著的提升，其誤差比對如下表:

| | 原始飛行狀態 | 加入機器學習模型 | 自動開關機器學習模型 |
|---|---|---|---|
| **路徑一** | 20.8 cm | **10.7 cm** | **11.2 cm** |
| **路徑二** | 20.9 cm | 34.6 cm | **14.3 cm** |

表 4、三種定位方法與對應路徑之誤差比對圖

由上表統整之定位誤差結果，無人機從最開始的定位能力，相比於後來加入了機器學習模型，其定位能力有所提升，然而在特定情況下定位能力卻不增反減，為了解決這

個問題，我們透過設計閾值來自動開關模型，使無人機在兩種狀況下均能獲得顯著地定位精確度提升，讓我們的輔助定位模型更加實用。

總結上述兩種飛行路徑之結果，透過模型增強無人機後，無人機抵達目的地時的平均誤差相比未加入前，減少了約 39%，雖然相比高精度陀螺儀等設備尚不夠好，但是已經能達到增強無人機飛行準確度之目的，且模型之應用範圍主要為障礙物較多的區域，待此技術更加成熟後，推斷本研究之定位增強方法未來可以應用於幫助無人機城市飛行更加普及、工廠或電廠等設施發生意外時輔助勘查，使無人機較不會因為 GPS 或電子羅盤受到嚴重干擾而無法前進該區域，此外，透過此技術進行圖像間的角度識別方法，未來也可以用於影像穩定等更多方面的功能。

在多數情況下，不同的無人機通常都有專門處理特定問題的功能，這也代表無人機在許多情況下，總是飛行在相似的環境中，即便是救災也有分抗高溫或特定功能的無人機，比如抗高溫型之偵查無人機可能較常飛行於火災現場或工廠等地方出現事故時進行偵查，這也代表無人機在許多情況下所拍攝到之畫面可能較類似，未來本研究在完成模型更進一步的各項檢測後，將嘗試修改模型讓其能進行自適應學習(Adaptive learning)，使無人機能夠在飛行幾次後，能對環境定位增強表現越來越好。

六、模型運算速度

為了能了解模型之實際應用時可能的執行效率，本研究使用 Roofline 模型套用至神經網路模型本身的理論數據，並以現有相似等級之無人機運算效能套用至模型，計算模型在真實世界運行時的理論速度上限。

本研究以 DJI Spark 無人機所採用之晶片為範例進行比較，此無人機之性能定位比實驗時所採用之 Mavic 2 Pro 低，不過其運算晶片採用 Intel Movidius Myriad 2，此晶片之資料相比其他訂製晶片較公開，然而此晶片為 6 年前的產物，為了使性能表現與現代無人機更加接近，本研究以同等級的新一代晶片 Intel Movidius Myriad X 最為比對對象，其性能相關資料如下表所示：

| 運算能力數據 | | 其他主要數據 | |
|---|---|---|---|
| **FP16 半精度浮點** | 4 TFlops/s | **最大帶寬** | 450 GB/s |
| **渲染處理核心** | 16 個 | **記憶體支援** | LPDDR4 2GB |
| **計算能力** | 4 TOP/s | | |

表 5、Intel Movidius Myriad X 晶片運算能力

而模型本身所需之計算量，透過將每層卷積之大小相乘，並將各層所需之計算量相加可得到模型總計算量，在 Roofline 模型中，每層卷積層的計算量公式(參考自[7])如下：

$$\boldsymbol{Cov_{Time} \sim O(M^2 K^2 C_{in} C_{out})} \qquad \textbf{(4)}$$

在式(4)中，M 代表每個卷積核輸出之特徵圖的邊長(在特徵圖為正方形之前提，本研究為長方形，故在本研究中代表特徵圖的長乘寬)，而 K 則代表卷積核本身之邊長，$C_{in}$代表卷積核之通道數，而$C_{out}$則代表卷積核所具有之輸出通道數(即卷積核數量)，對本研究而言，特徵圖為長方形，故不使用$M$而直接使用$M^2$代表特徵圖的輸出結果。

在本研究中各項參數分別為：第一層卷積$M^2 = 94724$、$K^2 = 9$、$C_{in} = 1$(本研究使用灰階圖像)、$C_{out} = 64$；第二層卷積$M^2 = 23049$、$K^2 = 9$、$C_{in} = 64$、$C_{out} = 128$，而透過總複雜度估算的兩層卷積運算計算次數之和為 1,753,917,696 次，過程中由於池化的複雜度相對較低故不列入計算，故卷積處理的預估計算量約為 1.633 GFlops，最後加入兩層具有 876 個神經元的隱藏層，由於隱藏層為全連接式，因此預估計算次數為兩層神經元數量相乘，共 759,736 次，相加後可得理論總運算量約為 1,754,685,072 次，每次對模型進行正向傳遞時，總共必須計算約 1.634 GFlops，然而本研究之模型運行於 FP32 的全精度浮點數模式，而晶片僅支援 FP16 的半精度運算，根據 NVIDIA 的文獻[8]指出，當神經網路運行於 FP16 模式時，執行效率相比 FP32 最高可以提升 8 倍，而模型本身的訓練表現幾乎不會產生下降，由於本研究並未對 FP16 下的模型進行測試，因此本研究假設模型以 FP32 在支援此架構的無人機特規晶片進行運算，並假設模型處於最差情況下，平台運算能力僅發揮 1/8，以此作為後續評估依據。

由 Roofline 模型可知模型的效能除了平台本身的計算能力，同時受到平台帶寬與模型計算強度影響，不過由於本研究所測試之模型還有大量模型優化的空間，故暫時先不對模型進行各個細部節點的評估(如：每個單一維度的 Filter 訪存量)，而是對模型各層總訪存量與計算強度進行基礎的評估，而所有卷積層的總訪存量評估(參考自[7])如下：

$$\boldsymbol{Cov_{Space} \sim O(4(\sum D\ K^2 C_{l-1} C_l + \sum D\ M^2 C_l))} \qquad \textbf{(5)}$$

在式(5)中，$K^2$、$M^2$等參數之涵義均與式(4)各參數之涵義相同，由於多層卷積層相連也會增加訪存量，其值為卷積層通道數的連乘，故多出了$\sum D$表示多層卷積網路連乘之積，$C_l$代表第$l$層時的通道數，在神經網路內，基本存儲單位為 Byte，然而深度學習時之參數類別為 Float32，Float32 每單位占用 4 個 Byte，故須將參數量乘四成為總內存佔用空間(即訪存量)，最後，透過上述公式可得出卷積層之總訪存量約為 17.27MB，並加入隱藏層運算時往前推進所需的 161.1MB，可得每次正向傳播計算約需要 178.3MB 的空間。

綜合以上結果，可以算得本模型之計算強度約為 9.38 Flops/Byte，大約介於現今的大型卷積神經網路與輕型神經網路之間，而晶片所具之計算強度為 8.8Flops/Byte，故模型能發揮平台運算能力之 100%，其 Roofline 模型如下圖所示: (為了限制範圍故以 log 表示)

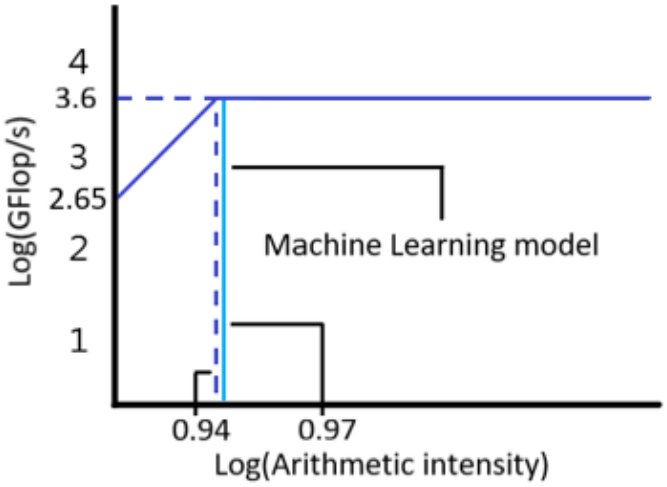



圖 25、本研究中機器學習模型之 Roofline 模型

在最差情況下，若晶片與模型結合後發揮不佳，無人機晶片計算能力僅能發揮 1/8，平台每秒約能計算 500 GFlops，由於當前模型本身輸入圖片規模較小，所需計算量相比大型模型較低，在運算核心無法被完整利用的情況下，平台每秒約能預測 306 張圖像，由於若將此研究之方法置於無人機，還須兼顧無人機之硬體運算條件，而卷積層涉及圖像處理，若修改不當即使提高了模型表現也可能導致模型運算量過大不適合運行於無人機上，依據實驗結果可得知修改模型隱藏層並不會顯著增加運算量，不過也會對現有的模型之訪存量產生明顯的影響，可能會降低模型的計算強度，此外，實驗時，模型在隱藏層節點數過多後，並非產生過擬合而是不收斂，代表模型隱藏層至目前為止之設計還有許多問題待解決，未來，我們也會朝此方向提高模型之預測表現。

本研究之網路輸入的圖像解析度僅為 240*400，真實世界中之無人機往往具有 720P、1080P 甚至 4K 等解析度，若單純對圖像大小直接換算，真實世界輸入神經網路之圖像節點數約為本研究使用圖像的 10~80 倍，若修改模型部分架構使晶片運算能力能被完整運用，並以更高解析度之圖像進行訓練，最後使模型運行於 720P 解析度下，假設模型表現與當前一致，模型每秒約能預測 252 張圖像，依舊高於模擬時所使用之每秒 125 張圖像，此外，由於更高解析度的圖像也具有更多細節的資訊，可以推測若增加圖像解析度，模型的預測能力可能也會更好。

透過比對現有無人機之硬體設備，得知此模型對於現今具有一定程度運算效能而言，並經由計算模型理論運行速度得知此模型能實現於無人機實時運行之能力，即使圖像解

析度僅有 240*400 像素，依舊具有充足的預測精準度，若以當前解析度及 125 幀的拍攝速度換算，等於**僅消耗約 5.1%的運算性能便能減少 39%的定位誤差**，總體定位增強性能與運行速度表現優異。

現今自動駕駛系統發達，例如汽車之自動駕駛系統，撇除透過訊號進行數位交通，這些系統大多也會搭載攝影機以拍攝該裝置之周遭情況，而本研究所設計之方法僅需依靠攝影機取得之資訊，且透過前述之運算量估算，得知此模型運算性能要求不高，未來本研究也會嘗試將此方法應用於這些系統上。

七、修改模型大小與其成效

為了能對模型進一步優化、提升辨別表現的空間能有大略的了解，且由於研究時間因素，不同卷積層或不同圖像解析度之比對暫時無法進行，本研究最後將不同隱藏層節點數量的神經網路分別進行訓練，比較其收斂結果，並以測試集之 RMSE 作為該模型表現，其中，為了方便比對，本研究直接採用二的冪次作為隱藏層節點數，並將神經元數量比例改為 1:2，原因是主要的記憶體開銷來自第一層隱藏層與卷積層的連結，故嘗試以減少第一層隱藏層並增加第二層隱藏層的神經元數量來修改原本的模型。

由於依據前一章之推測結果，可發現模型運行速度已經足夠快，且隱藏層節點數量更動對於模型運行速度影響相對較小(雖然可能增加訪存量並減少計算強度，但目前的模型中沒有性能問題)，因此本實驗將以最佳化模型收斂跟測試集測驗表現為目標，在不產生過擬合等收斂異常之範圍內盡量提高模型表現，實驗中，以(X,Y)表示兩層隱藏層分別具有 X 及 Y 個神經元，其餘參數則與先前模型訓練時的設定相同，得出如下圖表：

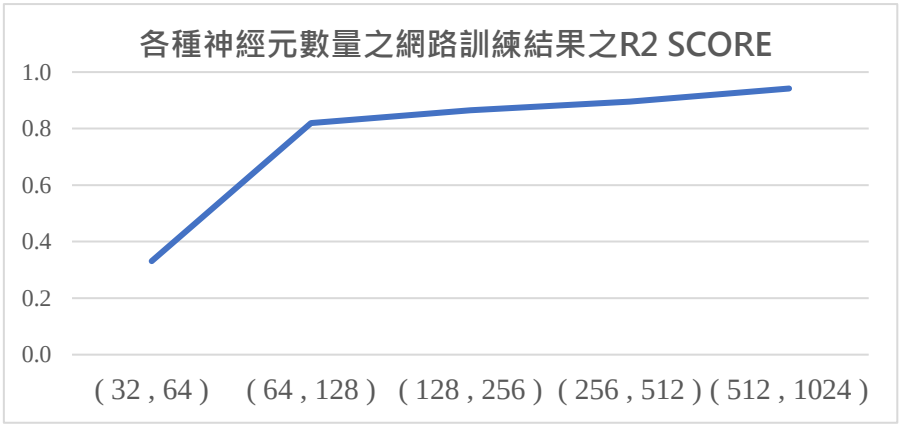



圖 26、各種神經元數量之網路訓練結果之 R2 SCORE

上圖中，由於神經元多於(1024,2048)的組合經過數次嘗試均會產生不收斂自動停止訓練，而神經元少於 32 之模型也會產生相同問題(R2 SCORE 為負數)，故不列入比較。神經網路收斂情形在隱藏層節點數量下降至 32 後大幅降低，而其餘神經元數量越多的模

型收斂情形則呈現穩定之連續上升關係。

接著為了求出模型在能有效提升無人機飛行準確度的前提下，模型大小的下限，依序將各模型訓練結果之 RMSE 製成下圖：

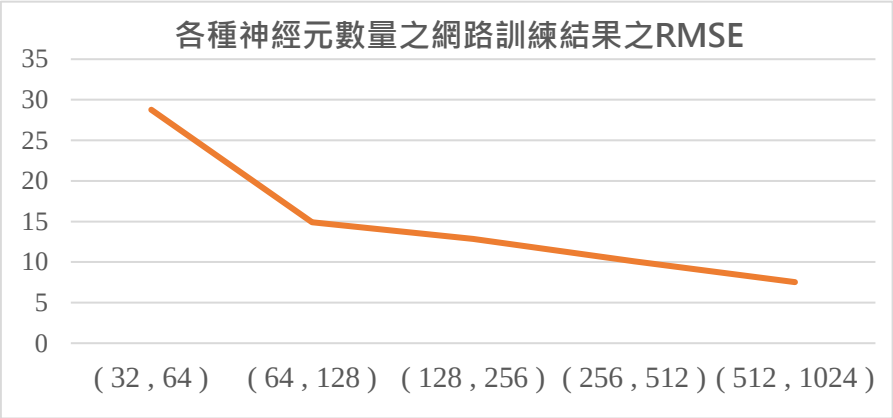



圖 27、各種神經元數量之網路訓練結果之 RMSE

圖中之 RMSE 在隱藏層小於(128,256)之組合後開始大於 15，當 RMSE 等於 15 時，模型判斷能力大約與電子羅盤相等(每秒 11 度)，而模型神經元數量在 64~1024 之間不會有明顯落差，由此可知，若想將模型實用於真實世界中，模型隱藏層之大小下限大約在(256,512)之組合，並整合前一章節的模型理論運行速度計算，若想使模型辨別能力最佳化，最好使用節點數為(512,1024)之組合的隱藏層設定模型，在此情況下，模型的 RMSE 約為 7.525，相比研究時所測試之模型表現又更好了。

依照以上實驗結果計算，若以(512,1024)之模型對圖像進行辨別，平均每秒約會產生 5.38 度的偏差，估算模型最終表現大約能使無人機飛行於障礙物較多的環境時，平均飛行路徑偏差能下降至未加入模型前的一半。

## 八、方法實用性

透過本研究之方法增強無人機定位後，能使目前無人機所具之定位能力獲得有效提升，且其運行效率能實現於一般常規無人機，然而本研究之方法對於鏡頭所拍攝之景色具有高度要求，若無法在鏡頭內拍取夠多的周遭物體資訊以利圖片相減時產生足夠特徵，則模型可能多數時間處於不啟用的狀態因而效果有限，有別於現今常見的視覺定位系統，本研究的方法雖然只能做為定位輔助而不能獨立運作，但其不受到距地高度限制，且不需要額外使用較為昂貴的飛時攝影機或複雜且運算量高的視差算法或 MVS 算法(Multi-View Stereo, MVS)，便能透過無人機畫面的資訊進行定位增強，在不需要任何額外製造成本的前提下，預估只要不是性能太弱之低端無人機，便能為該無人機提供一種新的定位輔助功能，使無人機定位系統能夠加完善。

# 肆、結論與應用

一、結論

(一) 透過深度學習模型能有效增強無人機飛行準確度

(二) 提升預測性能時可優先修改隱藏層設計

(三) 模型能以僅消耗約 5.1%之運算性能便能帶來約 39%的定位誤差縮減

二、未來展望

(一) 針對各種場景並使用分布更好的訓練資料進行測試

(二) 使用更好的機器學習模型啟用門檻判定方式

(三) 將方法升級成自適應學習

(四) 將方法應用於其他領域，如:視頻穩定技術

# 伍、參考文獻


[1] Young-Hoon Jin, Kwang-Woo Ko, &Won-Hyung Lee. (2018). An Indoor Location-Based Positioning System Using Stereo Vision with the Drone Camera. Hindawi. 5160543

[2] Erol Duymaz, A. Ersan Oğuz, Hakan Temeltaş. (2020). Exact flow of particles using for state estimations in unmanned aerial systems' navigation. PLOS ONE. 0231412.

[3] Hayyan Afeef Daoud, Aznul Qalid Md. Sabri, Chu Kiong Loo, Ali Mohammed Mansoor. (2018). SLAMM: Visual monocular SLAM with continuous mapping using multiple maps. PLOS ONE. 0195878.

[4] 王士益、劉瑋傑、顏永哲、歐陽盟、林修國(民109)。應用低成本雙頻GNSS RTK技術於無人機定位定向之研究。航測及遙測學刊，4，241-252。

[5] 晉鳳山(民91)。改良式倒傳遞類神經網路之研究。NTCUIR: 47 28-34

[6] 廖茂文、潘志宏。用TensorFlow實作最棒的GAN應用。深智數位股份有限公司出版。

[7] Roofline Performance Model。NERSC Documentation。取自：https://docs.nersc.gov/tools /performance/roofline

[8] Deep Learning Performance。NVIDIA Documentation。取自：https://docs.nvidia.com /deeplearning /performance/mixed-precision-training/index.html

[9] DJI' Mavic 2 PRO。Webots Documentation。取自：https://cyberbotics.com/doc/guide/robots

[10] DJI Mavic 2 Pro 技術參數。DJI。取自：https://www.dji.com/tw/mavic-2/info#specs

# 陸、附錄

一、無人機控制程式設計

在程式碼中，導入了用來儲存資料及運算的 os、NumPy、CV2 等資料庫，再將 Webots 內用以操縱感測器及機器人的幾個程式庫導入，並定義數值範圍限制函式 CLAMP 及小數位數切割函式 truncate (如圖 28)，輔助後續編程。

```
def CLAMP(n,minn,maxn):
    if n<minn:
        return minn
    elif n>maxn:
        return maxn
    else:
        return n
```

```
def truncate(num,n):
    temp = str(num)
    for x in range(len(temp)):
        if temp[x] == '.':
            try:
                return float(temp[:x+n+1])
            except:
                return float(temp)
    return float(temp)
```

圖 28、輔助編程函式

接著進入無人機平衡主程式，基礎平衡控制採用 DJI Mavic 2 Pro 的文獻數據及單位，並透過修改不同方向之目標值後運算，算出無人機四顆馬達推力值，並以此改變無人機四顆馬達之推力，並將推力以先前定義的 CLAMP 函式限制推力以符合 DJI Mavic 2 Pro 的性能，以下參數按照 Webots 文獻[9]之性能設定，符合此無人機性能數據，此外，由於找到之控制無人機相關文獻在專有名詞上多採用土耳其文，故少部分變數命名上也跟進採用以方便程式撰寫，最終程式如下圖 29：

```
#各方向輸入值
roll_giris=k_roll_p*CLAMP(roll,-1.0,1.0)+roll_hiz+roll_dagitim
pitch_giris=k_pitch_p*CLAMP(pitch,-1.0,1.0)+pitch_hiz+pitch_dagitim
yaw_giris=yaw_dagitim
clamped_yukseklik=CLAMP(target_rakim-rakim+k_vertical_offset,-1.0,1.0)
vertical_input=k_vertical_p*math.pow(clamped_yukseklik,3.0)
#各馬達輸入值
solMotorileri_giris=k_vertical_thrust+vertical_input-roll_giris+pitch_giris-yaw_giris
sagMotorileri_giris=k_vertical_thrust+vertical_input+roll_giris+pitch_giris+yaw_giris
solMotorGeri_giris=k_vertical_thrust+vertical_input-roll_giris-pitch_giris+yaw_giris
sagMotorGeri_giris=k_vertical_thrust+vertical_input+roll_giris-pitch_giris-yaw_giris
```

```
#無人機平衡之相關系數
k_pitch_p=30.0
k_roll_p=50.0
k_vertical_p=3.0
k_vertical_thrust=68.5
k_vertical_offset=0.6
```

```
#無人機三個方向之期望值及飛行高度期望值
roll_dagitim=0.0
pitch_dagitim=0.0
yaw_dagitim=0.0
target_rakim=0.5
```

圖 29、無人機飛行控制程式碼

二、無人機飛行位置指定程式精簡化

我們在程式碼中使用了 Python 的 Simple-PID 程式庫，將先前設計的無人機飛行控制程式與 PID 結合，並以 TargetX、TargetY 代表目標位置，為了能簡易化後續的軌跡比較，本研究無人機的 Z 軸(飛行高度)將從頭到尾保持一致，且為了能簡化結果對比時的難度，系統中沒有加入風等第三外力，同時不設計 TargetZ 避免改變飛行高度，這樣路徑

比對出來的結果即是定位系統定位能力的差距。

經過流程設定後，每次移動時，只需指定 PID 控制器的目標 x 軸、y 軸位置，便能使無人機自動前往該位置，而在移動過程中，無人機每次移動前先轉向目標位置，然後筆直前行直到抵達目標位置，便能抵達目標位置，將 PID 輸出值放入到先前的飛行控制程式碼後，位置控制方法僅需設定目標位置及何時須飛行至指定位置，並將變數 go 初始化為 0 無人機便會開始運行，如下圖：

```
if robot.getTime() ==7.0:
    target_x=0
    target_y=0
    go=0
```

圖 30、指定無人機飛行位置程式

上圖程式碼中，無人機會在模擬運行開始後第 7 秒進行移動，並精準移動至 x、y 座標(0,0)的位置，至此，基礎飛行控制的所有程式碼已完成。

三、無人機圖像採集程式與 TimeStep 設定

在飛行圖片採集部分，本研究使用了 Webots 的 Camera 中的 saveImage 函式，儲存大小為 240*400 的照片 ，如圖 31：

```
if Needsave==True :
    #camerafile=camerafile+1
    with open(FilePath + "everynum.txt", 'a') as f:
        f.write(txt_input)
        f.close()
    FileName=str(camerafile)
    FileName=FileName + ".png"
    Needsave=False
    camerafile=camerafile + 1
    Camera.getImage(camera)
    Camera.saveImage(camera,ABFilePath+FileName,1)
```

圖 31、Webots 圖像生成程式碼

在上述程式碼完成後，本研究將 TimeStep 設定為 8，值設定越高雖然能模擬更加快速，但由於移動及物理碰撞基準也為每 TimeStep 毫秒進行一次運算，所以設定越高也會導致模擬越不精確，此無人機之最快水平移動速度為 72KM/h，換算後為 2CM/ms，不過無人機預設處於低速檔，為 3.6KM/h，換算後為 0.1CM/ms，相當每次模擬最多移動 0.8 公分，其值使無人機不會有單次模擬即撞到物體又移動長距離造成較難辨別無人機飛行狀態，並能讓模擬器以尚可的現實世界 0.3 倍速度進行模擬，在無人機模擬器中飛行約 4 分 45 秒，現實中花了約 16 分鐘。

四、ML 模型的程式製作

在本研究中，採用了 VSCode 作為主要的機器學習訓練平台，雖然在深度學習方便性上不如 Google Colaboratory 等網頁、筆記型訓練平台，但是 VSCode 具有的終端機及自動程式碼除錯能加快程式測試，並能幫助研究時處理環境設定問題，因此本研究將以

VSCode 建立與訓練模型。

而在程式設計部分，首先，先從類別的手動飛行模擬一般飛行情況所生成的資料開始訓練，在程式中，首先先設定圖像路徑，並設定採用所有資料中的其中 30000 張圖像，再將取出的圖像打亂，以避免訓練時總是同類別的資料一起訓練，導致每次學習損失函數下降地太快，結果當不同總資料開始訓練又因先前模型訓練方向錯誤又跳回前前個最佳解重複訓練，造成效率不佳。

接著，設定將其中 80%的資料作為訓練集，20%的資料做為測試集，同時也有將資料進行洗亂以確保能正常訓練，而 Dropout 或是其他新型激勵函數(如：Leaky ReLU)並未加入至模型中，而是若模型有出現過擬合或不收斂等現象，才會考慮加入，這麼作目的是使神經網路先精簡化，後續若有問題也不會不知從何下手。

完成訓練資料的設定後，接著透過設定訓練集、驗證集與測試集分別的檔案路徑、圖像大小、顏色種類，檔案路徑不變、而圖像大小為 400*240 (因設定關係輸入要做轉換)、顏色種類為灰階，每次以 32 筆資料進行批次學習，程式碼如下圖：

```
test_images = test_generator.flow_from_dataframe(
    dataframe=test_df,
    x_col='Filepath',
    y_col='Angle',
    target_size=(400, 240),
    color_mode='grayscale',
    class_mode='raw',
    batch_size=32,
    shuffle=False
)
```

測試集程式

```
val_images = train_generator.flow_from_dataframe(
    dataframe=train_df,
    x_col='Filepath',
    y_col='Angle',
    target_size=(400, 240),
    color_mode='grayscale',
    class_mode='raw',
    batch_size=32,
    shuffle=True,
    seed=42,
    subset='validation'
)
```

驗證集程式

```
train_images = train_generator.flow_from_dataframe(
    dataframe=train_df,
    x_col='Filepath',
    y_col='Angle',
    target_size=(400, 240),
    color_mode='grayscale',
    class_mode='raw',
    batch_size=32,
    shuffle=True,
    seed=42,
    subset='training'
)
```

訓練集程式

圖 32、訓練器及驗證器及測試器程式碼

接著設定模型，此模型採用卷積核大小為 3*3 的卷積神經網路，兩次卷積的濾波器分別設定為 64 及 128 (此值為實驗前進行幾次預測試後，模型表現較好之濾波器大小)，其中濾波器的大小將決定提取出的特徵數量，最後，設定兩層隱藏層的神經網路有 876 個神經元及激活函數 ReLU，並設定模型輸出為線性代表以回歸方式解決此問題，程式碼如下圖：

```
inputs = tf.keras.Input(shape=(240, 400, 1))
x = tf.keras.layers.Conv2D(filters=64, kernel_size=(3, 3),  activation='relu')(inputs)
x = tf.keras.layers.MaxPool2D()(x)
x = tf.keras.layers.Conv2D(filters=128, kernel_size=(3, 3), activation='relu')(x)
x = tf.keras.layers.MaxPool2D()(x)
x = tf.keras.layers.GlobalAveragePooling2D()(x)
x = tf.keras.layers.Dense(1024, activation='relu')(x)
x = tf.keras.layers.Dense(2048, activation='relu')(x)
outputs = tf.keras.layers.Dense(1, activation='linear')(x)
model = tf.keras.Model(inputs=inputs, outputs=outputs)
```

圖 33、TensorFlow 訓練設定

最後設定訓練時採用的損失函數值為平方誤差(MSE)，能較容易地與訓練完成後得出的總模型 RMSE 結合後看出訓練時的損失函數曲線兩者之間的關係，完成損失函數設定後，再設定訓練循環為 100 個 epochs，並且設定 patience 為 5，代表五次迭代失敗後停止訓練以避免過擬合，最後使用深度學習領域中較常見的 Adam 優化器輔助訓練，完成模型架構設定。

在設定好以上 TensorFlow 的編譯方式後，最後便使用 Conda 版本的 Python 以 GPU 開始訓練，設定的編譯器如下圖所示：

```
model.compile(
    optimizer='adam',
    loss='mse'
)
history = model.fit(
    train_images,
    validation_data=val_images,
    epochs=100,
    callbacks=[
        tf.keras.callbacks.EarlyStopping(
            monitor='val_loss',
            patience=5,
             restore_best_weights=True
        )
    ]
)
```

圖 34、TensorFlow 訓練設定

在設定的編譯器中，回傳部分將會在程式提早完成訓練時(五次梯度下降失敗後)，使神經網路自動回復到訓練過程中表現最好的權重，成為最終訓練完成之模型。

# 【評語】190001

這個作品具有實用價值，但目前只是在模擬軟體建立環境場景來驗證功能和量測誤差，若移到真實環境且採用真實無人機來飛行，是否能能達到一樣好的定位誤差，值得比較和探究。另外，飛行一段距離所累積的定位誤差和此段的飛行距離是否有正比的關係在作品中沒有進行實驗探討，在未來可以進行研究。