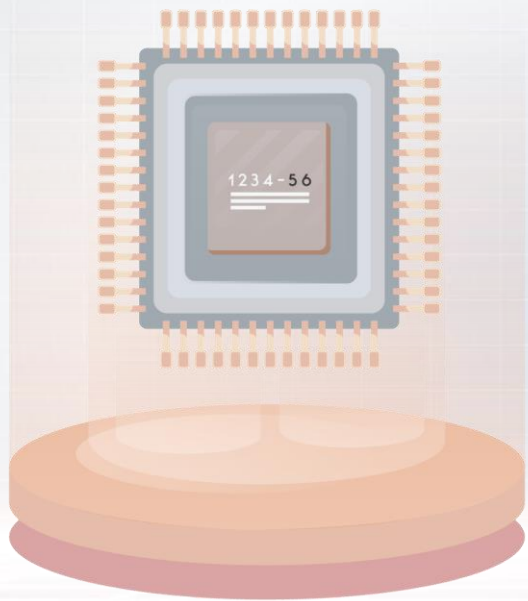
A stylized illustration of a dark grey rectangular block with the letters 'ai' in white, resting on a two-tiered orange and yellow pedestal. The background features a light grey grid with faint geometric shapes and lines, including concentric circles and a network of dots and lines on the left side.

# 無人載具結合 影像辨識飛行

邱俊翰



# 01. 前言

# 前言

本專題使用 DJI Tello 實作出數字辨識功能，Tello 是一台體積小、操作簡單、機動性高，且同時具備精準定位與影像拍攝的小型無人機，而我們運用此優點，將其結合影像識別的應用。

在技術上，基於 KNN 數字辨識模型，將自製資料集作為輸入進行訓練，並套用訓練成果於 Tello 中。辨識過程中，輸入影像會經由 OpenCV 前處理，提取影像特徵再做預測。最後，辨識結果與即時畫面一同顯示，使我們能快速取得測試成果。

未來發展應用於大範圍或特殊環境條件下收集或辨識特定物件，在有限人力之下持續監測或追蹤的任務，如需減少支出，且避免任務的風險，此技術將會是不錯的選擇。

# 02.

## 使用器材



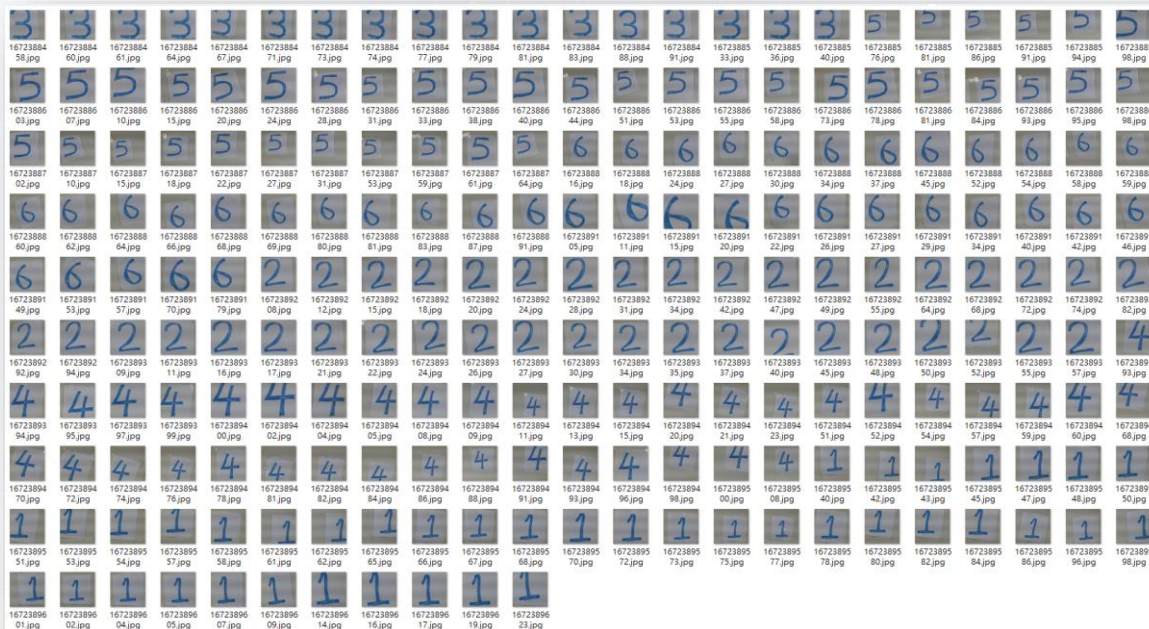
DJI Tello



Wi-Fi 模組



## 03. 自製資料集





# 04.

## 自製資料集利用KNN訓練

在分類問題中 KNN 演算法採多數決標準，利用  $k$  個最近的鄰居來判定輸入資料是屬於哪一群，其演算法流程非常簡單。首先決定  $k$  的大小，接著計算目前該筆資料與鄰近的資料間的距離。第三步找出跟自己最近的  $k$  個鄰居，查看哪一組鄰近的數量最多，就辨識為該族群。

1. 決定  $k$  值。
2. 求每個鄰居跟自己之間的距離。
3. 找出跟自己最近的  $k$  個鄰居，查看哪一組鄰居數量最多，就加入哪一組。

# 利用cv2.KNearest訓練

```
72
73 # 訓練集資料
74 print(x_train.shape[0])
75 x_train = x_train.reshape(x_train.shape[0],-1) # 轉換資料形狀
76 x_train = x_train.astype('float32')/255 # 轉換資料型別
77 y_train = y_train.astype(np.float32)
78 print(type(x_train))
79 print(type(y_train))
80
81 # 測試集資料
82 x_test = x_test.reshape(x_test.shape[0],-1) # 轉換資料形狀
83 x_test = x_test.astype('float32')/255 # 轉換資料型別
84 y_test = y_test.astype(np.float32)
85
86 knn=cv2.ml.KNearest_create() # 建立 KNN 訓練方法
87 knn.setDefaultK(5) # 參數設定
88 knn.setIsClassifier(True)
89
90 print('training...')
91 knn.train(x_train, cv2.ml.ROW_SAMPLE, y_train) # 開始訓練
92 knn.save('mnist_knn_test.xml') # 儲存訓練模型
93 print('ok')
94
95 print('testing...')
96 test_pre = knn.predict(x_test) # 讀取測試集並進行辨識
97 test_ret = test_pre[1]
98 test_ret = test_ret.reshape(-1,)
99 test_sum = (test_ret == y_test)
100 acc = test_sum.mean() # 得到準確率
101 print(acc)
```

```
-----Generate Datasets-----
-----Save Datasets-----

241
training...
ok
testing...
0.9253112033195021
```

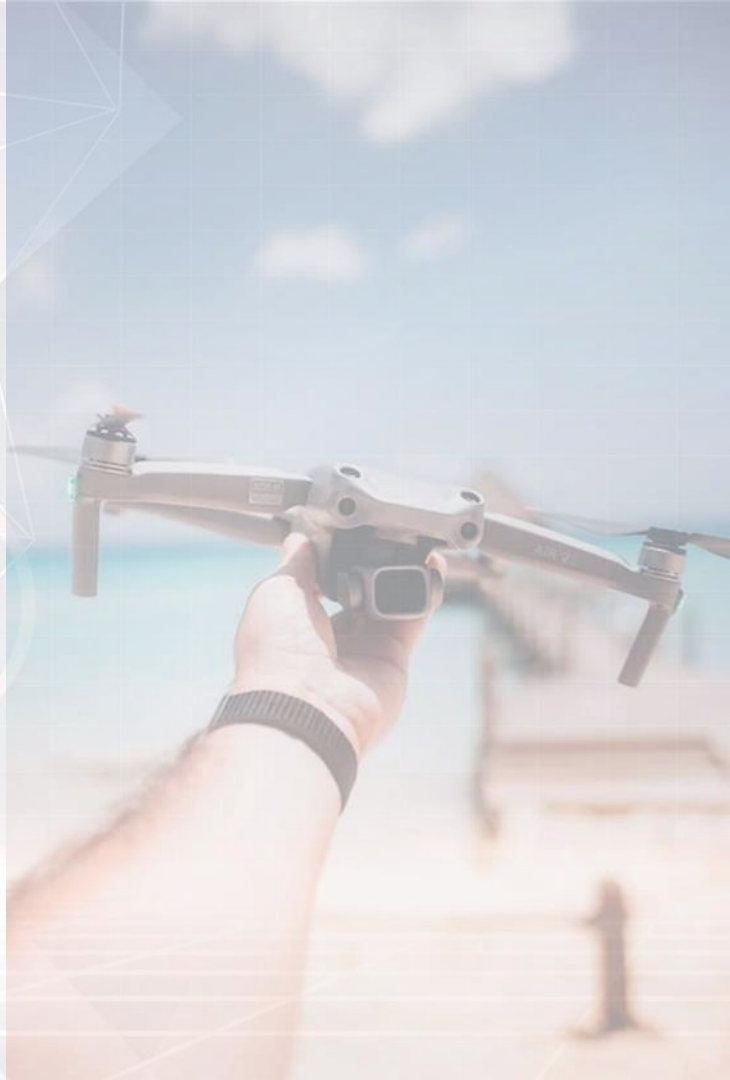


# 05.

## 無人機辨識與控制

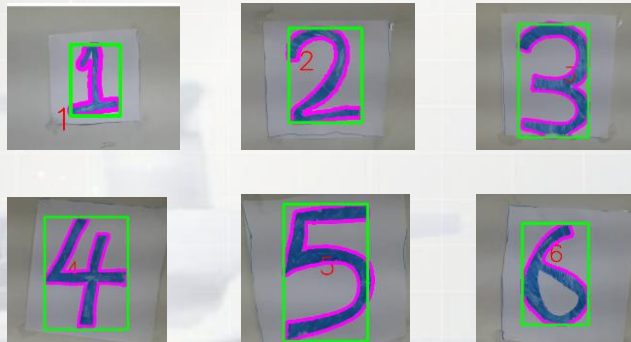
為了凸顯輸入影像的特徵，我們使用 OpenCV 進行一連串의影像處理。過程如下，將前處理的二質化影像，依序做濾波、侵蝕、膨脹和縮放後，再放入 KNN 模型中訓練。經上述的方式，可大幅度彰顯數字의特徵，以利於分類의成效。

無人機部分是以 Wi-Fi 做連接，調用 Tello 所提供的 SDK 開發，完成無人機起降、懸停等控制。藉此，實作出無人機影像辨識功能。

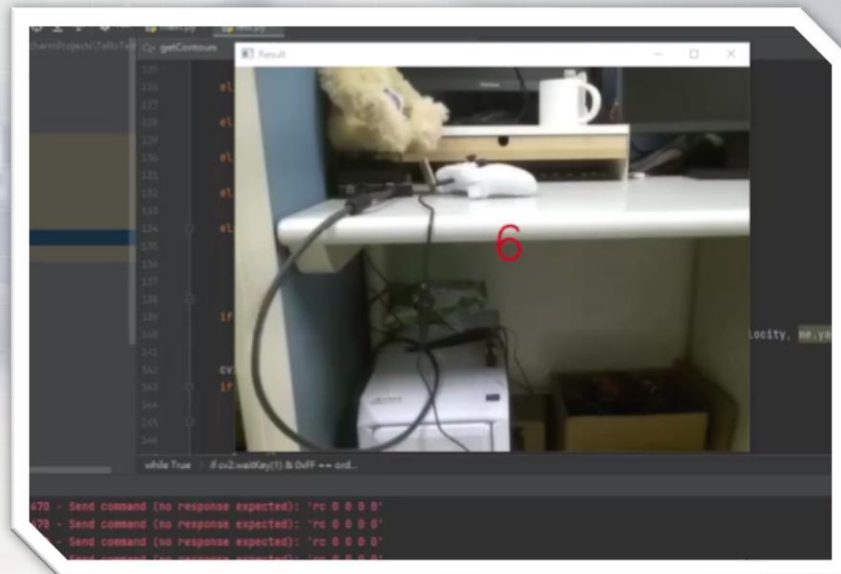
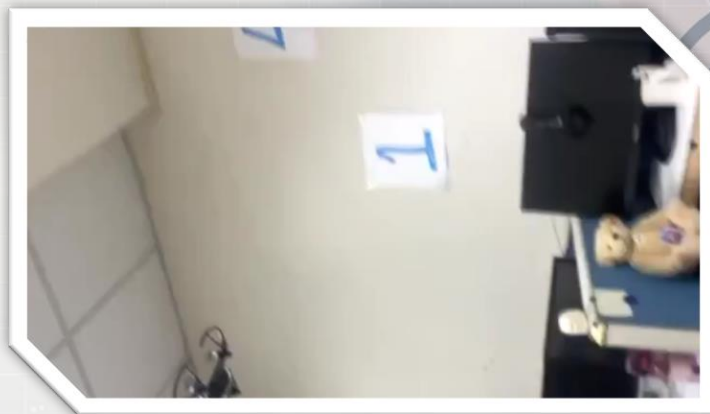


## 06. 實驗成果

# 實驗成果



預測成果，示意圖





## 07. 問題與解決

# 問題與解決

原先想要實踐的使用 DarkNet 套入 TinyYolo，但因虛擬機始終無法顯示飛行影像，有成功接收影像資訊並成功套入模型，故最終放棄此模型建置與訓練。實作過程如下：

Step 1：無法接收影像。

```
keyframe
Tello: 07:10:10.518: Error: video recv: timeout
Tello: 07:10:15.524: Error: video recv: timeout
Tello: 07:10:20.530: Error: video recv: timeout
Tello: 07:10:25.535: Error: video recv: timeout
Tello: 07:10:30.541: Error: video recv: timeout
Tello: 07:10:35.548: Error: video recv: timeout
Tello: 07:10:40.552: Error: video recv: timeout
Tello: 07:10:45.558: Error: video recv: timeout
-S
```



# 問題與解決

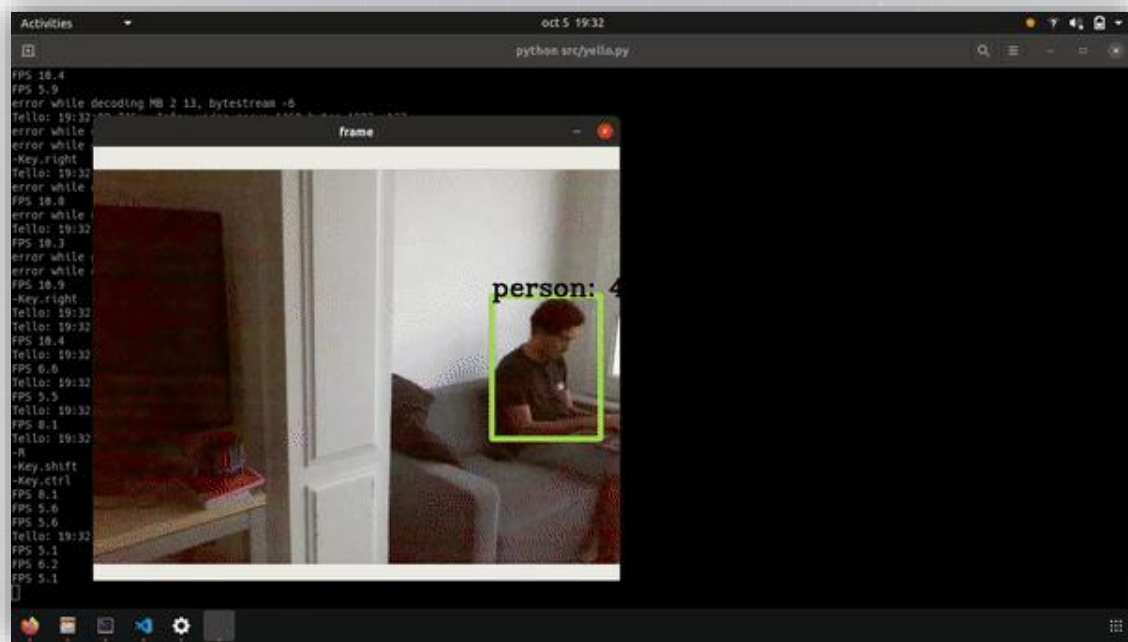
Step 2 : 解決 Step 1 問題成功橋接網路至虛擬機，完成接收影像資訊。

```
eternal@ubuntu: ~/Yello
File Edit View Search Terminal Help
error while decoding MB 54 40, bytestream -6
error while decoding MB 10 10, bytestream -6
Tello: 08:58:24.958: Info: video rcv: 1460 bytes bd02 +304
Tello: 08:58:25.262: Info: video rcv: 1460 bytes bd03 +304
Tello: 08:58:25.567: Info: video rcv: 1460 bytes bd04 +304
Tello: 08:58:25.872: Info: video rcv: 1460 bytes bd05 +305
Tello: 08:58:26.177: Info: video rcv: 1460 bytes bd06 +304
Tello: 08:58:26.367: Info: video rcv: 1460 bytes bd07 +189
Tello: 08:58:26.367: Info: video data 51261 bytes 24.7KB/sec loss=471
error while decoding MB 4 14, bytestream -9
Tello: 08:58:26.786: Info: video rcv: 1384 bytes ee88 +222
Tello: 08:58:26.887: Info: video rcv: 15 bytes fc80 +101
Tello: 08:58:26.988: Info: video rcv: 10 bytes fd80 +101
Tello: 08:58:27.293: Info: video rcv: 1460 bytes ff00 +304
Tello: 08:58:27.699: Info: video rcv: 1460 bytes ff01 +406
Tello: 08:58:28.004: Info: video rcv: 1460 bytes ff02 +304
Tello: 08:58:28.308: Info: video rcv: 1460 bytes ff03 +303
Tello: 08:58:28.386: Info: video data 43606 bytes 21.1KB/sec loss=477
error while decoding MB 18 12, bytestream -6
Tello: 08:58:29.127: Info: video rcv: 1460 bytes 3c03 +555
Tello: 08:58:29.432: Info: video rcv: 1460 bytes 3c04 +305
Tello: 08:58:29.834: Info: video rcv: 1460 bytes 3c05 +402
Tello: 08:58:30.036: Info: video rcv: 708 bytes 3c86 +202
```



# 問題與解決

Step 3：在本地端無法顯示 Frame 視窗(嘗試清理環境，並依據作者 Github 上Release 方式，依舊無法實現)



# 問題與解決



在小型無人機平台進行預測與辨識雖然是可行的，但因鏡頭錄製影響與 Wi-Fi 傳輸等並須兼顧飛行，Tello 無人機的可用電力大約只能飛行 5 分鐘，就必須進行充電，這也是我們所遭遇的一項困境。

# 08. 結論

本次專題中整合機器學習、影像處理及無人機飛行控制，為了更精確的進行辨識與飛行定位，我們嘗試了許多的方法並加以改進，但受限於設備與時間上的限制。因此，盡可能的將最佳的成果呈現出來。透過這類的學習，累積與無人機相關的專案開發。





## 09. 參考資料

# 參考資料

1. [https://docs.opencv.org/3.4/d5/d26/tutorial\\_py\\_knn\\_understanding.html](https://docs.opencv.org/3.4/d5/d26/tutorial_py_knn_understanding.html)
2. <https://ithelp.ithome.com.tw/m/articles/10269826>
3. <https://github.com/murtazahassan/Tello-Object-Tracking/blob/master/ObjectTrackingTello.py>
4. <https://ithelp.ithome.com.tw/articles/10307704?sc=iThelpR>
5. <https://ithelp.ithome.com.tw/articles/10297550>
6. <https://blog.hashteacher.com/?p=1048>
7. [https://blog.csdn.net/qq\\_36108664/article/details/107087068?fbclid=IwAR0EK2CdeY8W2xJBphp8YzNHfDAI\\_VnJC6cshigGE7bPdIFVJtShzgUeziA](https://blog.csdn.net/qq_36108664/article/details/107087068?fbclid=IwAR0EK2CdeY8W2xJBphp8YzNHfDAI_VnJC6cshigGE7bPdIFVJtShzgUeziA)
8. <https://github.com/adriacabeza/Yello>
9. <https://github.com/damiafuentes/DJITelloPy>

Thank you  
for listening.