

# Final Project - Get my waffle!!

0751919 邱震譯 0856606 蔡孟勳 0856121 江沛澤

## 1. Introduction

我們的期末專題題目是"Get my waffle!", 其中需要克服的困難大概可以分為三項：人物與叫號機的辨識、叫號機的數字辨識、無人機的控制。透過無人機內建的鏡頭，我們使用YOLOv3將無人機拍攝到的畫面進行object detection，判斷客戶或叫號機是否存在目前的畫面，隨後調整無人機的位置進行叫號機的數字辨識，辨識成功後就會尋找客戶的位置，最後前往客戶的面前進行降落。

## 2. Motivation

交大小木屋其實相當的受歡迎，但因為鬆餅現做的關係，小木屋的熱門時段通常會等上20到30分鐘。為了減少大家因為排隊而浪費的時間，我們希望透過無人機來幫助我們運送鬆餅。

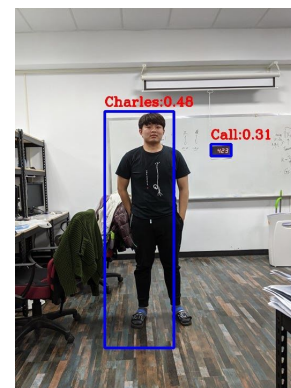
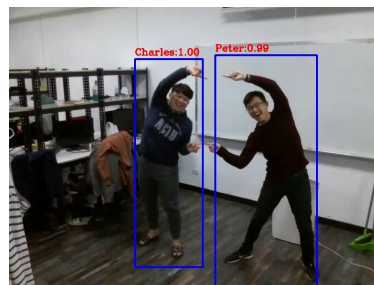
## 3. Method

### a. YOLOv3 - Object Detection

#### i. Model setting:

- 1) Pretrained weight: yolov3\_tiny.weights
- 2) Classes: [Charles, Allen, Peter, Calling\_machine]
- 3) Confidence threshold: 0.5
- 4) NMS threshold: 0.5
- 5) Training dataset: 150 images
- 6) Train iterations: 50000 (選表現比較好的)

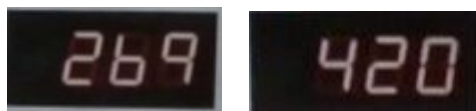
#### ii. Test result:






b. Traditional method - Number Recognition



i. Number Segmentation

7) 透過socket接收YOLOv3 object detection的結果。(如下圖)



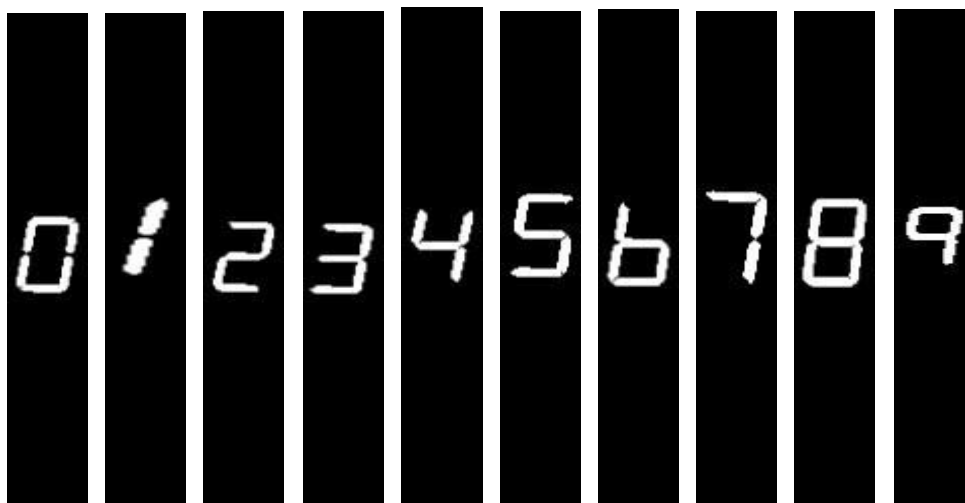
8) 進行傳統影像處理，像是：灰階化、邊緣偵測、閉運算、二值化，最後根據白色在結果圖中佔的比例來準確切割數字。

灰階化	邊緣偵測	閉運算
		

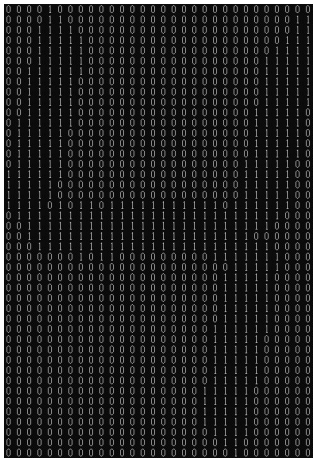
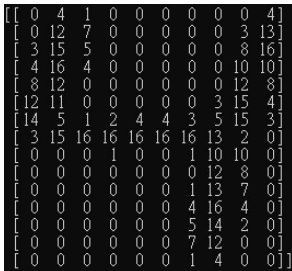
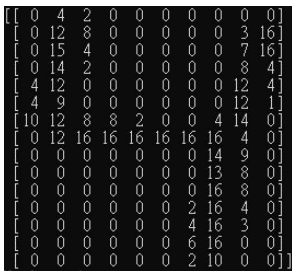
二值化	segmentation後的結果
	

ii. Number Recognition

1) 我們先拍幾張數字的照片作為ground truth



- 2) 並擷取segmentation後的結果與ground truth中包含數字的最大區域，並每間隔4\*4的區域取sum提取feature。

ground truth 4 包含數字的最大區域	ground truth 提取特徵的結果	prediction result 提取特徵的結果
		
prediction result與ground truth 0~9 計算兩者feature間的l2-norm		
Number: 4 Difference: 23.345235059857504 Differences: [85.71464285639881, 120.80149005703531, 87.71544903835355, 70.76722405181653, 23.345235059857504, 83.03011501858829, 87.14929718592113, 65.82552696332935, 83.916625289629, 56.52433104425032]		

- 3) 透過socket回傳預測結果給main function

c. Drone Controlling:

- rotate\_drone: 旋轉無人機使calling machine/customer bounding box置於畫面中央。
- translate\_drone: 直線調整無人機使calling machine/customer bounding box置於畫面中央。
- zoom\_drone: 前後移動無人機使calling machine/customer bounding box佔畫面於一定比例。

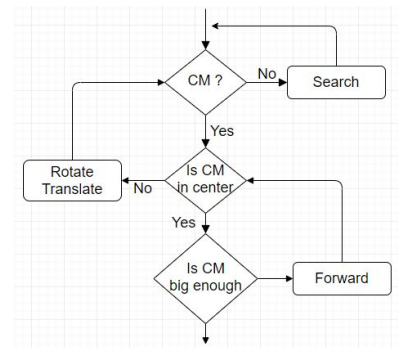
d. Helper Function:

- pixel\_to\_meter: 將pixel轉換meter。
- check\_cm\_ratio: 檢查calling machine佔畫面是否足夠大。

#### e. Overall process

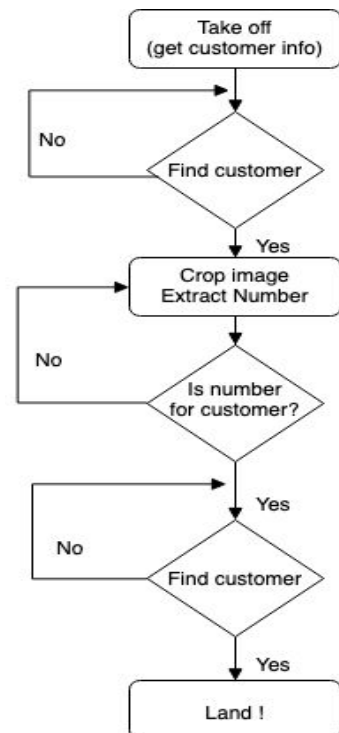
##### i. 定位流程

- 1) 確認目標有在視線內
- 2) 調整目標至視線中央
- 3) 檢查物體是否夠近  
(為了取得更清晰的圖片)



##### ii. 整體流程

- 1) 起飛無人機
- 2) 每次順時針旋轉30度尋找叫號機
- 3) 如果找到叫號機，就會根據bbox調整無人機的位置
- 4) 如果夠靠近叫號機，就會辨識數字
- 5) 等待直到辨識到正確的客戶號碼
- 6) 如果辨識到客戶號碼，就會每次順時針旋轉30度尋找客戶
- 7) 如果找到客戶，就會根據bbox調整無人機的位置
- 8) 如果夠靠近客戶，就會準備降落



#### 4. Experiment:

[https://drive.google.com/open?id=1VqvGjJxIMXrn4p-n0kvV-38olwva\\_LZN](https://drive.google.com/open?id=1VqvGjJxIMXrn4p-n0kvV-38olwva_LZN)

#### 5. Challenges:

- a. Tello 環境設定: 一開始在安裝環境的時候遇到了許多很麻煩的問題，需要一一上網排解問題，其中最大的問題是Tello無人機本身的韌體更新問題，這並沒有在內附的FAQ或助教的解說PDF內提到，查了很久也沒有看到別人有類似的問題，後來是在嘗試用手機控制Tello之後，Tello才提示需要連網更新韌體，Tello最後才得以用電腦控制，順利起飛。

- b. YOLOv3訓練問題: 我們一開始在訓練YOLOv3時, 所用的 training data 太少且不夠多樣化, 導致YOLOv3最後學到每個 class 的 feature 都不一樣, 例如class: Peter 是學到大面積紅色物體, 有大面積的紅色物體就有很高的可能被判定是Peter。這部分的問題我們分析是可以透過training data 的多樣化來解決, 使YOLOv3不會只學到一種feature。
- c. Drone Control: 由於Tello在完成動作之前不會接受第二條指令, 所以在下第二條指令之前, 最好都是要設置一定時間的 `time.sleep()` 來確保動作已經完成。

## 6. Conclusion:

- a. 一開始我們很煩惱該如何從frame中取得可以控制drone的訊號, 最後結論是發現可以利用bounding box 作為無人機的方向指引, 雖然有蠻多參數需要調整, 像是我們要如何轉換pixel information 成我們熟知的公制單位, 又或是無人機鏡頭在多遠的距離取的照片可以讓YOLOv3得到信心較高的判定, 透過一次次的測量與實驗, 我們才能在程式中設定適合的參數。
- b. 這次使用Tello作為完成Project的工具才讓我們意識到現實與電腦環境的區別其實很大, 現實環境會有更多需要考慮的突發狀況, 有些在設計程式時並不會想到, 需要一步步的測試並改寫程式。現實世界的響應通常也並不那麼即時, 限制也很多, 這也大大延長了我們每一次的實驗時間, 這點在之後想必也是排定進度時需要考慮的。

## 7. Team Work

邱正義 : Tello environment setting, Drone control, 整合

蔡孟勳 : Call machine recognition, 整合

江沛澤 : Train YOLOv3 model, 整合

## 8. Reference

### a. Yolo

#### i. Model

<https://github.com/pjreddie/darknet>

#### ii. Network setting

<https://reurl.cc/b6KpXE>

### b. Number segmentation and recognition

#### i. <https://reurl.cc/EK5dZa>

#### ii. <https://reurl.cc/9zm858>

#### iii. <https://reurl.cc/ILK65E>