網路嵌入式系統應用Final Project

《實作農用自走車的檢測元件》



組別：5

姓名：郭富凱、沈文凱

導師：黃悅民

內文目錄

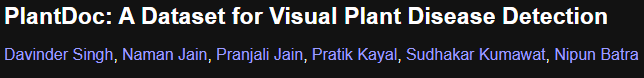
**前言**

本專題旨在將檢測元件安裝在噴灑農藥的農用自走車上。

協助判斷是否有受感染之葉子，使農民能提早剪枝以預防感染擴大。

並且我們希望以麥克風控制Jetson Nano是否進入物件辨識階段 (若不需要時可以聲控關閉) 。

**資料集**



本專題使用《PlantDoc》資料集

在2019年，由印度理工學院提供，是一個包含13種植物物種和27種類別（17 個疾病，10 個健康）的2598張圖像的數據集。

|  |  |
| --- | --- |
| healthy\_plants = [  'Apple leaf',  'Bell\_pepper leaf',  'Blueberry leaf',  'Cherry leaf',  'Corn leaf',  'Grape leaf',  'Peach leaf',  'Raspberry leaf',  'Soyabean leaf',  'Strawberry leaf',  'Tomato leaf'  ] | diseased\_plants = [  'Apple Scab Leaf',  'Apple rust leaf',  'Bell\_pepper leaf spot',  'Corn Gray leaf spot',  'Corn leaf blight',  'Corn rust leaf',  'Grape leaf black rot',  'Potato leaf early blight',  'Potato leaf late blight',  'Squash Powdery mildew leaf',  'Tomato Early blight leaf',  'Tomato Septoria leaf spot',  'Tomato leaf bacterial spot',  'Tomato leaf late blight',  'Tomato leaf mosaic virus',  'Tomato leaf yellow virus',  'Tomato mold leaf',  'Tomato two spotted spider mites leaf'  ] |

**使用工具**

1. Demo environment: Jupyter Notebook

2. Speech2Text: SpeechRecognition (Using Chinese)

3. Threading

4. Object Detection: Yolo v7 (Self-trained model, merge detect.py in one file.)

5. Camera: limited to 1 frame per second.

6. GPIO: LCD, LED, Light sensor, Supersonic sensor, BME280

.

**環境建置過程**

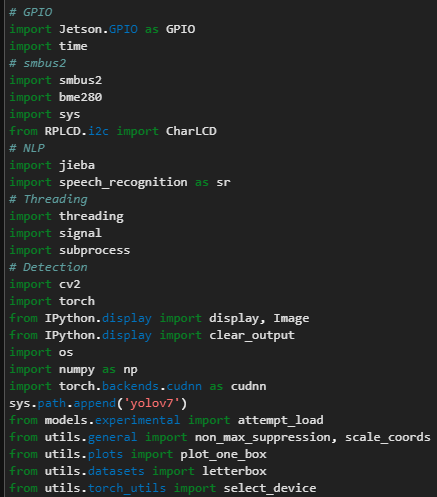
我們主要將程式分成控制器、主程式流程、執行緒 (影像辨識階段)

整體程式碼，以物件導向為主，兼顧封裝、易讀性，以便後續再擴充功能。

專題程式碼為Project.ipynb這單一檔案。

-

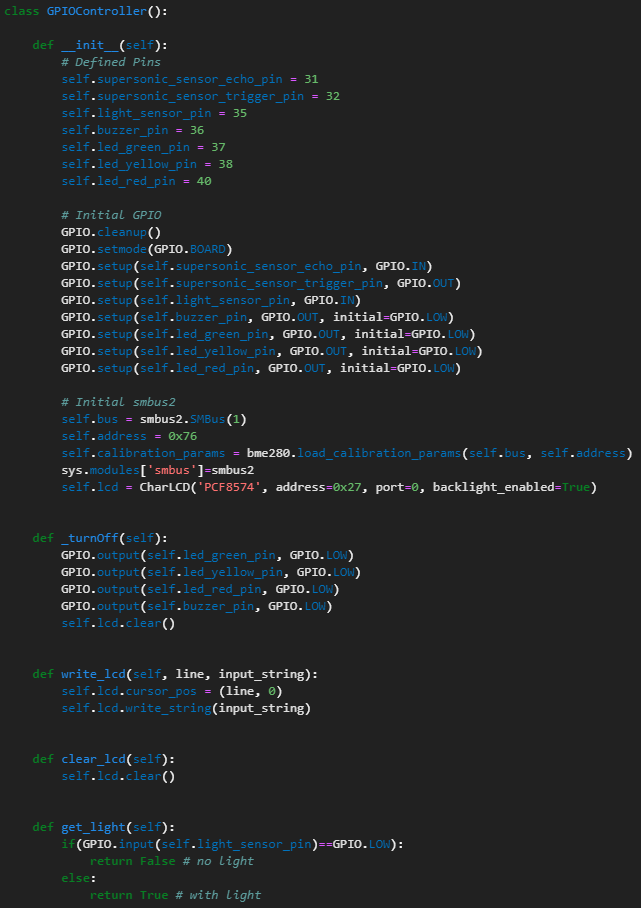
Import時將功能區分清楚



-

1. GPIO Controller: LCD, LED, Light sensor, Supersonic sensor, Bme280

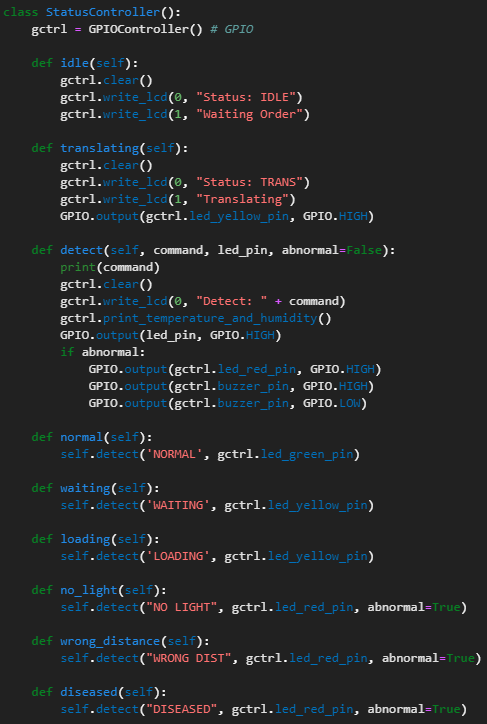
將GPIO相關之功能整合在GPIOController中



-

2 .Status Controller: 狀態控制器 (Idle, Translating, no\_light, wrong\_distance, Loading, Normal, Diseased , Waiting)

將程式目前的狀態，該處理的事情皆寫在StatusController中



-

3. Detect Controller: 鏡頭初始化、載入Yolo v7模型

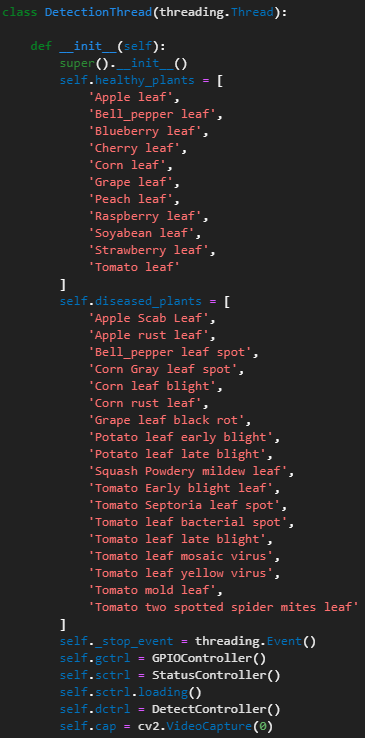
將物件辨識的部分寫在DetectController中



-

4. Detection Thread: 物件辨識

將檢測部分寫在執行緒中，以確保語音檢測與物件辨識能同時執行



-

5. Main Function: 語音控制

主要的無窮迴圈，可以再這邊設定要用甚麼樣的關鍵字語言來執行相對的動作。

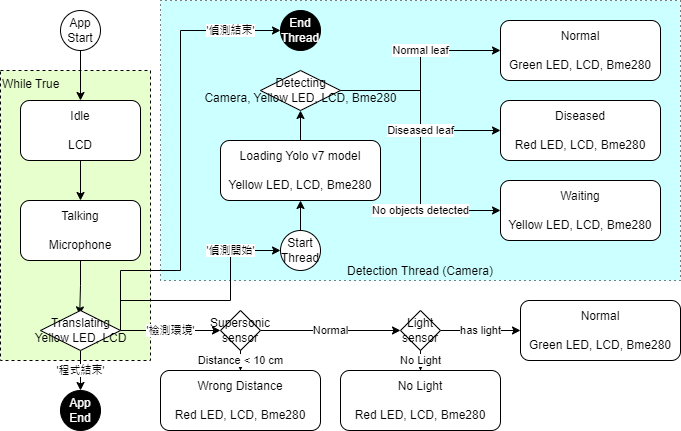


**程式說明**

本專題使用Jupyter Notebook運行即可，專題程式碼為Project.ipynb這單一檔案。

-

下圖為程式的Flow Chart：



-

於程式啟用後，進入閒置狀態，LCD開始顯示'idle'。

-

喊: '檢測環境'。

程式使用Supersonic sensor檢測鏡頭距離是否正確，以及使用是否處在照明充足的環境：

若任一條件符合，則LCD顯示Wrong Distance或No Light， 紅燈亮起，蜂鳴器發出警報。

若條件皆不符合，則LCD顯示Normal，綠燈亮起。

-

喊: '偵測開始'，進入偵測階段

程式運行鏡頭以及載入我們自行訓練的Yolov7 Model，且LCD開始顯示環境溫濕度。

-

於偵測階段。

若檢測正常，則 LCD 顯示 Normal，LED 綠燈亮起。

若檢測異常，則 LCD 顯示 Disease Detected，LED 紅燈亮起，蜂鳴器發出警報。

若無檢測到物件，則 LCD 顯示 Waiting，LED 黃燈亮起。

若須停止偵測，使程式待命，則喊: '偵測結束'，使程式結束偵測。

-

若需完整退出程式，則喊: '程式結束'，退出無窮迴圈。

**專題限制**

1. 語音辨識限制

使用’偵測結束’要關閉Thread時，會因為物件偵測記憶體使用過量。

而導致無法使用語音辨識功能，此問題於專題報告前都無法解決。

但若是用subprocess方式，將物件偵測改為使用yolov7/detect.py，反而語音辨識功能都還能正常使用，目前也不確定原因為何。

1. Yolo v8

本專題在物件辨識的套件上，本希望是使用Yolov8，因其能直接導入Library使用。

而Yolov7，需用到Subprocess呼叫detect.py的方式較不直觀 。

但因Yolov8需要將Jetson Nano的Python更新到3.8的版本，試了很多方式，最終於辨識階段遇到系統卡死的狀況，只好放棄。

最終我們選擇，改將Yolov7的detect.py的邏輯實做到我們的檔案中，才能在不需要Subprocess的方式於程式內部使用。

**專題分工**

郭富凱: 程式撰寫(全)、模型訓練、概念PoC、主題發想、簡報製作、文件製作

沈文凱: 程式撰寫(協助GPIO部分、協助物件辨識部分)