LAB：車牌辨識專題

實驗模組作者：洪祐鈞、高效能計算實驗室

若是做了有問題，可以email聯絡課程助教。

實驗目標：

- 透過本實驗模組，建構一簡易的車牌辨識系統。
- 自動車牌辨識（automatic license plate recognition，ALPR）

術語：

本機：你自己的電腦，會和樹莓派遠端連線

需求：

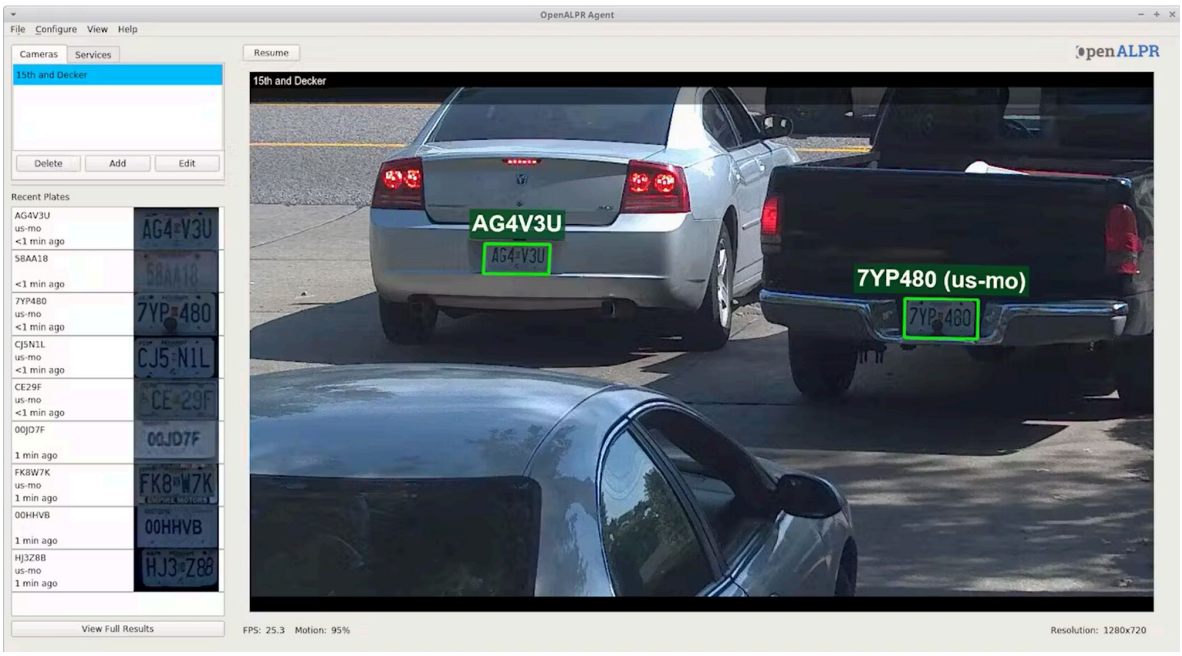
- 請先閱讀本文檔，了解要做的事情是什麼，有可能有些步驟你不需要也可以達到目標。
- 本機是Linux系統，強烈建議直接安裝Linux系統 (不要透過虛擬化的軟硬體技術)。
- 使用Windows或是WSL的理論上可以行得通，但必須理解整個實驗模組在做的事情，並且自行尋找解決方法。
- 建議先完成行人偵測實驗模組，再完成這個實驗模組。

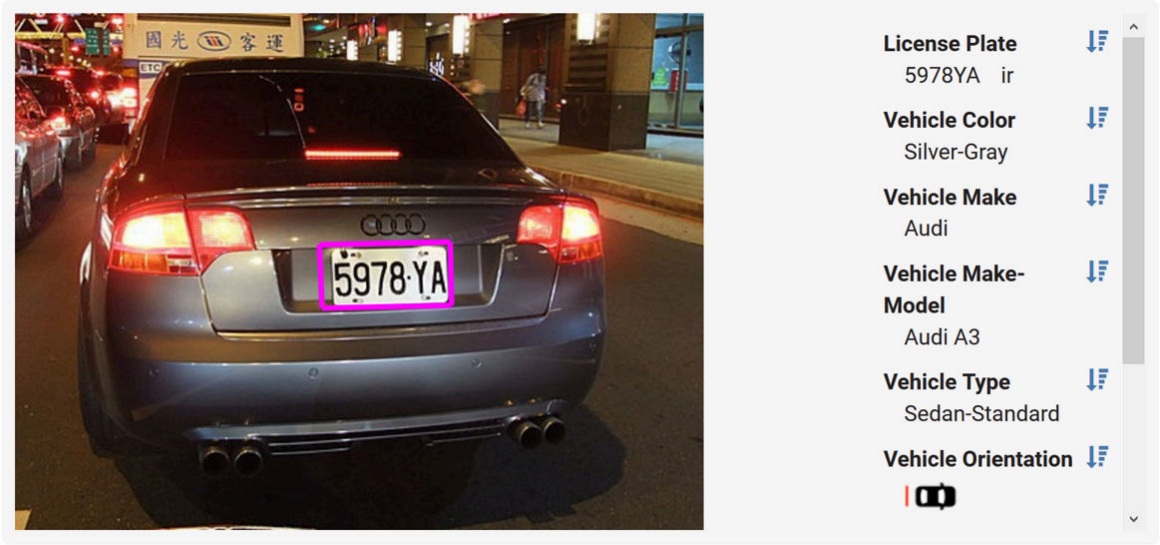
需要材料：

- 樹莓派
- 安裝好 Raspberry Pi OS 的 SD卡 (如行人偵測實驗模組一樣，不用重灌)
- WiFi環境

安裝OpenALPR

- OpenALPR is an open source Automatic License Plate Recognition library.
- The library written in C++ with bindings in C#, Java, Node.js, Go, and Python.
- The library analyzes images and video streams to identify license plates.
- The output is the text representation of any license plate characters.
- 有興趣可以進一步參閱 http://www.openalpr.com/cloud-api.html 。





接下來可以使用 ssh 遠端連線，使本機對樹莓派進行操作。

1. 安裝更新：

```
$ sudo apt update
$ sudo apt upgrade
```

2. 安裝必要套件：

```
$ sudo apt install autoconf automake libtool -y
$ sudo apt install build-essential cmake git libgtk2.0-dev pkg-config
libavcodec-dev libavformat-dev libswscale-dev -y
$ sudo apt install libatlas-base-dev gfortran -y
```

3. 若是接下來的安裝，若RAM不夠的話可以嘗試增加swap，但也要考慮你的樹莓派空間：(不一定要做！！！)

```
$ sudo fallocate -l 4G /swapfile
$ sudo chmod 600 /swapfile
$ sudo mkswap /swapfile
$ sudo swapon /swapfile
$ sudo swapon --show
```

4. 安裝leptonica：

   - **Leptonica** is a pedagogically-oriented open source library containing software that is broadly useful for **image processing** and **image analysis applications**。
   - 主要包括的操作有：點陣圖操作、仿射變換、形態學操作、連通區域填滿、圖像變換及圖元掩模、融合、增強、算數運算等操作。

```
$ cd
$ git clone https://github.com/DanBloomberg/leptonica.git
$ cd leptonica/
$ git checkout v1.71
$ chmod 777 configure
$ ./configure
$ make -j2
$ sudo make install
```

5. 安裝tesseract：

   Tesseract，一款由 HP 實驗室開發，由 Google 維護的開源 **OCR（Optical Character Recognition , 光學字元辨識）引擎**，可以不斷的訓練圖庫，使圖像辨識不斷增強；如果團隊深度需要，還可以以它為基底，開發出符合自身需求的 OCR 引擎。

```
$ cd
$ git clone https://github.com/tesseract-ocr/tesseract.git
$ cd tesseract
$ git checkout 3.04.01
$ ./autogen.sh
$ ./configure --enable-debug
$ make -j2
$ sudo make install
```

6. 安裝OpenCV:

   `sudo apt install libopencv-dev`

7. 安裝OpenALPR

```
$ cd
$ git clone https://github.com/openalpr/openalpr.git
$ cd openalpr
$ sudo apt install libcurl4-openssl-dev liblog4cplus-dev -y
$ cd src
$ mkdir build
$ cd build
$ cmake ..
$ make -j2
$ sudo make install
$ sudo ldconfig
```

8. 測試OpenALPR:

   `alpr plate.jpeg`

   將會輸出：

```
plate0: 10 results
    - 786PO      confidence: 89.1473
    - 786PO      confidence: 79.1358
    - 786PQ      confidence: 78.6103
    - 786PD      confidence: 77.5103
    - 786O       confidence: 73.9903
    - 786PB      confidence: 72.9523
    - 786PU      confidence: 71.1389
    - 786PG      confidence: 70.8614
    - 786P       confidence: 69.5352
    - 786O       confidence: 63.9789
plate1: 10 results
    - D1D1       confidence: 81.3787
    - DDD1       confidence: 80.7326
    - D1DI       confidence: 80.5026
    - 11D1       confidence: 80.1846
    - DDDI       confidence: 79.8565
    - 1DD1       confidence: 79.5384
    - 11DI       confidence: 79.3085
    - 1DDI       confidence: 78.6623
    - Q1D1       confidence: 75.8495
    - D1O1       confidence: 75.6745
```

如何手寫一個車牌辨識的程式？

sftp

若是可以ssh遠端連入，那就可以sftp互傳檔案，將plate_recog之專案傳到樹莓派。

以下只是重點提示：

1. sftp連線到樹莓派

   `sftp pi@192.168.xxx.xxx`

2. 使用put指令將檔案由本機傳到樹莓派。

   `put -r plate_recog/`

plate_recog專案

plage_recog是助教為了課程課餘開發的小專案，專注在辨識七位數的第八代台灣車牌。

設置環境及安裝套件

在樹莓派上安裝：

1. 設置虛擬環境

   樹莓派有預設安裝Python，現在可以藉由虛擬環境來隔離環境。

   ```
   $ cd plate_recog
   $ python -m venv env
   $ source env/bin/activate
   ```

2. 安裝必要套件

   ```
   $ pip install opencv-python
   ```

3. 測試

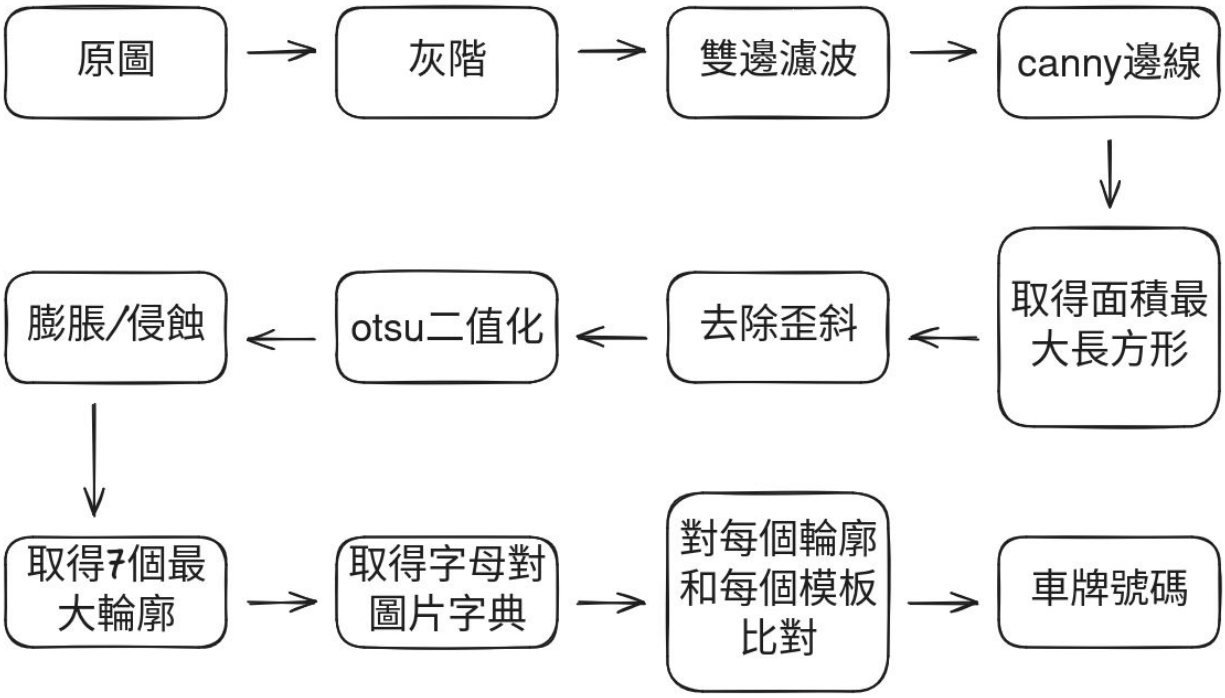
   ```
   $ python plate_recog.py
   ```

   應該會出現

   ```
   best match: A, value: 0.8525890111923218
   best match: A, value: 0.8694927096366882
   best match: A, value: 0.872961699962616
   best match: 4, value: 0.35498046875
   best match: S, value: 0.4684656858444214
   best match: 8, value: 0.7091379165649414
   best match: 5, value: 0.7426824569702148
   plate number: AAA4S85
   ```

基本上就代表範例程式成功運行了

How does it work?



1. 灰階

   使用`cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)`轉換

2. 雙邊濾波

   使用`cv2.bilateralFilter`來模糊圖片，可降低雜訊。

3. Canny邊線

   使用`cv2.Canny`以取出邊線。

4. 取得面積最大長方形

   使用`cv2.findContours`找出輪廓
   並且使用`cv2.approxPolyDP`找出4個邊

5. 去除歪斜

   使用`cv2.getPerspectiveTransform``取出透視變換矩陣`
   `並且使用`cv2.warpPerspective```將車牌傳換成正的長方形

6. otsu二值化

   使用`cv2.threshold(img, 0, 255, cv2.THRESH_BINARY + cv2.THRESH_OTSU)`
   針對像素為雙峰分佈的灰階像素做二值化

7. 膨脹/侵蝕

   使用

   ```
   cv2.erode
   cv2.dilate
   ```

形態學嘗試消除過小雜訊區塊，並連接小縫隙。

8. 取得7個最大的輪廓

   使用`cv2.connectedComponentsWithStats`取得連通圖，簡單來說就是連續的像素單位
   並且排序面積後篩選7個最大的輪廓，回傳排序的圖片集

9. 取得字母對字典

   從檔名讀取字母和圖片，以字典資料結構將兩者關聯在一起，可藉著字母查詢對應圖片。

10. 針對每個輪廓和每個模板比對

    遍歷每個字母，使用`cv2.matchTemplate`對於每個模板比對，找最大的相似度分數的字母。

作業:

完成下面問題，簡述你的作法，並繳交最終程式碼。(不要連環境都丟進來了)，截圖需要包含Raspberry Pi的使用者名稱的終端機指令、日期和時間。

會需要閱讀程式碼和自行搜尋有關的資料以完成本次的作業。

1. 請截圖成功執行OpenALPR的結果(10%)
2. 請執行並截圖plate_recog的結果，其結果需要：
   1. 終端機輸出，如同測試（能跑成功就出來了）(10%)
   2. 出現匡選圖（看得懂程式打指令就可以出來了）(10%)

      如：

   3. 在這個匡選圖上，請用程式的方式，嘗試在圖片上印出方框的四個點的座標數字 （需要看懂程式大約在做什麼並做相對應實作）(10%)

4. 對於程式做時間和空間的數據量測：(10%)

   跑完全部但不含產圖的時間是什麼？(Hint: 可在python程式使用`time.time()`)

   佈署這個專案包含虛擬環境(env資料大小)，程式碼本身和模板的大小總共佔多少空間（可使用`du`指令）

3. 請針對plate_recog專案中辨識車牌的方法，提出三個可能造成問題的點。(10%)

   （如車牌和背景太過相近可能導致canny抓不到車牌邊線，這裡提出的不算在三點之內）

4. 針對你提出的問題中挑選一個並進行實作以改善專案。(20%)

5. 心得（每個人各一）： (20%)