

智慧自駕車與模擬城市

Intelligent self-driving car and Simulated City

指導教授：李國川老師

學生：黃振益、鄭鈞維、廖崧竹、吳柏均

國立聯合大學 資訊工程學系

苗栗市南勢里聯大 2 號

flhuang@nuu.edu.tw

摘要

近年來 AI 蓬勃發展，而自駕車技術也隨著 AI 的進步越來越成熟，為了搭上現在的潮流，因此我們製作自駕車系統在模擬城市中行駛，並透過模擬城市中的路牌來驗證自駕車的功能。

為了讓自駕車的功能可以正確的在城市中驗證，我們設計了一座模擬城市跟城市中有著各種路牌，根據這些路牌，因此自家車有了路牌辨識、區域定位、車道矯正以及停車系統等功能，我們結合路牌辨識系統與車道矯正系統使自駕車可以穩定的在城市中行駛，透過區域定位可以得知車子現在位置。

我們也架設一個網站以及手機 APP 以便使用者可以觀看車輛的資訊、位置以及即時地圖，APP 有即時播報系統，使用者可以用聽的就得知現在位置。APP 停車系統可以讓使用者指定停車地點，例如：回家和上班。

關鍵字：自駕車、路牌辨識、車道矯正

Abstract

With the Artificial Intelligence developing and booming for the past few years, self-driving car technology is getting more and more mature. In order to keep up with the pace, we make a self-driving car system to drive in the simulated city and verify the function throughout our simulated city.

In order to verify the function of the self-driving car, we design a simulated city with various street signs. Self-driving car has Street Sign Recognition System, Regional Mapping System, Lane Correcting System and Parking System. We can realize the current position of the car throughout the Regional Mapping System.

We set up a website and Android operating system APP so that users can view the car information, car position and real-time map. APP has Real-Time Broadcast System to make users can know the car

current position with their ears and users can designate the parking place (be on duty or go home) throughout the Parking System.

Keyword : self-driving car、Street Sign Recognition、Lane Correcting System

一、前言

自駕車在近幾年中越來越受到人類的重視，雖然仍有些少許的車禍但根據美國的研究報告顯示出自駕車意外發生率小於一般汽車，因此我們決定製作智慧自駕車。

我們使用 YOLOv3-tiny 以及 Intel RealSense 攝像頭結合路牌與城市景觀(區域)辨識系統，讓車子可以辨識到城市中的交通號誌、路牌以及區域，並透過車道線矯正系統，使車子在偏移道路時可以自動矯正回來，結合路牌辨識以及車道線矯正系統，使自駕車可以穩定的在城市中行駛。

我們藉由路牌辨識系統，透過辨識區域以及路牌，可以達成區域定位，並透過區域定位系統可以得知車子現在位置。APP 停車系統可以讓使用者指定停車地點，並透過區域定位系統得知目前位置，當目前位置跟指定停車地點相符合時，自駕車就會自動停車。停車地點例如家裡或公司，因此根據選擇的點不同車子就會到選擇的地點停車。

使用者可以透過網頁以及 APP 來得知自駕車目前的車況、當前位置以及當下環境狀況。APP 的即時播報系統可以讓使用者可以用聽的就得知現在位置。

二、系統技術與架構

1. 深度學習與物件辨識：

路牌辨識採用深度學習的方式來進行實作，實作所採用的類神經演算法為 YOLOv3-tiny，YOLOv3 是基於深度學習的物件偵測。因此我們將路牌視為一個物件，共 15 種路牌 15 個物件。

透過 Intel RealSense D435i 串流拍攝，配合深

度學習建立的模型進行物件辨識，將辨識結果透過車子執行。

1-1 物件辨識

透過深度學習技術使機器學習辨識特定物件，將深度學習之輸出作為物件特徵用於辨識系統之辨識規則以利於辨識系統的輸出結果。基於此研究目的，本研究將分為兩部分，第一部分為透過深度學習實作物件分辨，第二為實作辨識系統的特定物件辨識。

每種路牌各 30 張照片，並將光影、角度考慮進去，使照片不重複以達成較好的結果。將共計 450 張照片作為訓練的資料，並用 labeling 標註要辨識的物件及編輯物件的標籤(圖一)，接下來會生成含有座標的 txt 檔(表一)，下一步切割訓練與測試集，訓練集 80%，測試集 20%，並撰寫 YOLOv3-tiny 的設定檔，開始訓練建立模型。

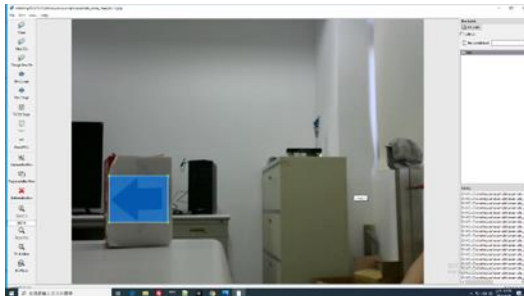


圖 1. bounding box 圖

Category number	Object center in X	Object center in Y	Object width in X	Object width in Y
1	0.837853773 5849056	0.479166666 6666667	0.121462264 1509434	0.191666666 6666668

圖 2.座標 TXT 圖

YOLOv3 使用 resent 網路(Residual Network) 新的基底網路是改良 YOLOv2 的 Darknet-19，由連續的 3x3 和 1x1 的捲積層組合而成，共 53 層所以稱為 Darknet-53，然而因為硬體效能的關西，我們使用 YOLOv3-tiny，YOLOv3-tiny 是 YOLOv3 的輕量版，是以 YOLOv3 網路層為基礎，去掉一些特徵層，故速度較快。

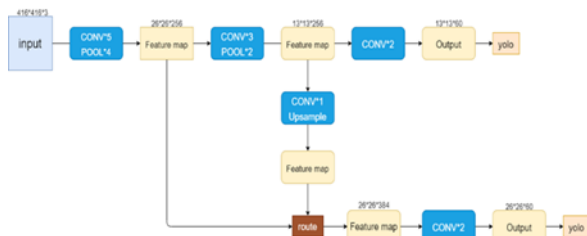


圖 3.YOLOv3-tiny 網路圖

YOLOv3 在訓練的過程中會產生權重檔，本專題在設定上採取每 1000Epoch 存檔一次，而一次完

整的訓練為 502000Epoch，但事實上並不需要這麼多，因此我們會檢查遺失率(loss)，當遺失率達到 0.02 時就行止訓練。

最後將訓練好的權重檔轉成 TensorRT 所使用的格式，TensorRT 是深度學習優化器，其主要目的為加速推理辨識，並提高 FPS 數達到即時辨識。

透過 Intel RealSense D435i 串流拍攝，配合深度學習建立的模型進行物件辨識，將辨識結果傳送到資料庫，藉由網頁和 APP 呈現相關資料，並透過車子執行。

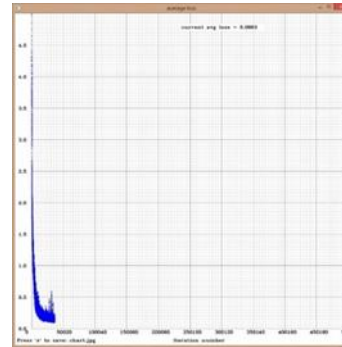


圖 4.平均 Loss 圖

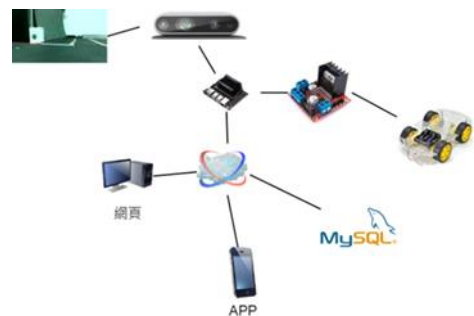


圖 5.自駕車架構圖

2.道路線辨識：

2-1.資料裁切

因為需要辨識的車道線顏色僅需眼前一小部分的影像，如果影像的範圍太廣會使影像無法辨識所以需要做影像的裁剪。



圖 6. 影像裁切圖

2-2.邊緣偵測

我們採用了 canny edge detection 此為一種多階段的演算法

2-2-1.降噪

邊緣檢測容易受到圖像中噪音的影響，因此須使用 5x5 高斯濾波器消除圖像中的噪音。

2-2-2. 查找圖像的強度梯度

使用 Sobel Kernel 在水平和垂直方向上對降噪過後(平滑圖像)進行濾波以在水平方向上獲得一階導數 (GX) 和垂直方向得一階導數 (GY)。藉由 GX 與 GY 能夠得知他的邊緣梯度(edge gradient)與每個像素的方向

(angle) 公式如下所示

$$Edge_Gradient (G) = \sqrt{G_x^2 + G_y^2}$$

$$Angle (\theta) = \tan^{-1} \left(\frac{G_y}{G_x} \right)$$

2-2-3. 非最大抑制(設為零)

漸變方向始終垂直於邊緣，在獲得梯度大小和方向後，將對圖像進行全面掃描，以去除可能不構成邊緣的所有不需要的像素，檢查像素是否在梯度方向上構成附近的局部最大值。

2-2-4. Hysteresis Thresholding

該階段確定哪些邊緣全部是真正的邊緣，哪些不是。我們需要兩個閾值 minVal 和 maxVal。強度梯度大於 maxVal 的任何邊緣必定是邊緣，而小於 minVal 的那些邊緣必定是非邊緣。介於這兩個閾值之間的對象根據其連通性被分類為邊緣或非邊緣。(圖 7)

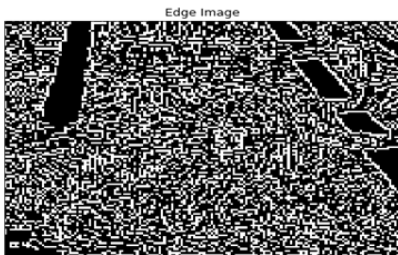


圖 7. 結果

2-3. 設定 hsv 空間並定義所需顏色的閾值 (threshold)(顏色辨識)

我們需要的顏色為紅色及黃色，所以僅需定義二種顏色，紅色較為複雜，需要定義到二種空間因為其跨越 255 跟 0，將影像的 rgb 空間轉換成 hsv 空間並定義他們的閾值(threshold)，根據圖像輸出的資料即可判定圖中的紅色與黃色在哪个區域並顯示出來。

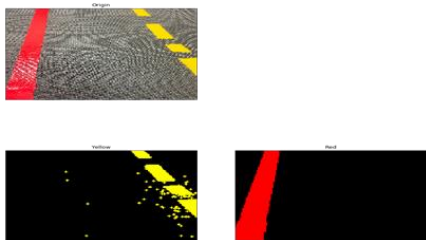


圖 8. 顏色判斷結果

2-4. 霍夫線轉換

2-4-1.

霍夫直線檢測的原理在於利用點與線的對偶性，影像空間中的直線與引數空間中的點是互相對應的。因此霍夫直線檢測演算法就是把在影像空間中的直線檢測問題轉換到引數空間中對點的檢測問題，通過在引數空間裡尋找峰值來完成直線檢測任務。

2-4-2.

對於霍夫變換，我們使用極座標系統表達線。引數空間是不能選擇直角座標系的，因為原始影像直角座標空間中的特殊直線 $x=c$ (垂直 x 軸，直線的斜率為無窮大) 是沒辦法在基於直角座標系的引數空間中表示的，因此線方程式表達如下圖公式，其中

$$r = x \cos \theta + y \sin \theta$$

對於每個點 (x_0, y_0) ，我們可以將通過該點的線族定義為以下公式：

$$r_{\theta} = x_0 \cdot \cos \theta + y_0 \cdot \sin \theta$$

2-5 結合霍夫直線偵測與顏色辨識

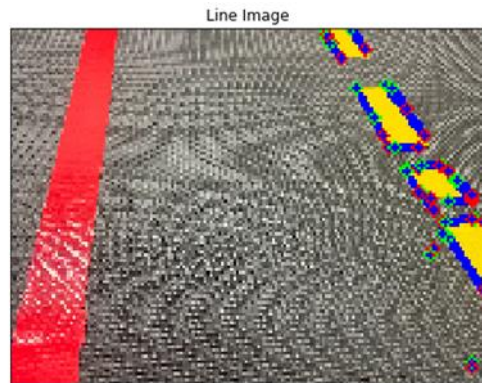


圖 9. 以偵測黃色線為例

3. L298N 車體控制：

L298N 模組用來驅動車子的左右輪，可以分別控制車子左右輪前進、倒退及停止，控制車子行進方向是靠 IN1~IN4 的接腳控制，控制轉速則是靠 ENA 跟 ENB 控制。

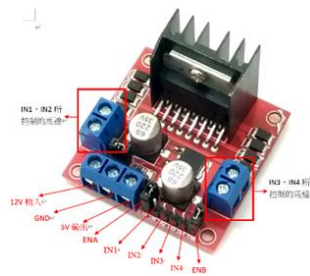


圖 10. L298N

將 L298N 的 IN1~IN4 接上 JetsonNano

的 pin 腳，ENA 及 ENB 則需接上特定的腳位才能控制馬達轉速，接下來使用 Python 的 GPIO 套件控制輪胎的轉向，IN1 及 IN2 控制左邊輪胎，IN3 及 IN4 控制右邊的輪胎，透過改變 IN1~IN4 的 HIGH、LOW 來控制正轉、反轉或停止。

使用 ChangeDutyCycle 函式來控制輪胎轉速，透過控制轉速及轉動方向，即可實現車體的各種行動，例如：左右轉。

4.資料呈現：

4-1 網頁

網頁為響應式網頁，並使用動態更新的方式來更新及呈現我們的資料。使用程式語言有 HTML、JavaScript、JQuery、CSS、Bootstrap 框架，並架設 MySQL 資料庫，用 PHP 來抓取資料庫中的資料。

4-2 APP

手機 APP 使用 Android Studio 製作，使用 PHP 作為 API 抓取資料庫資料，並將資料轉換成 JSON 檔格式回傳給 APP，才能使 APP 可以讀取到資料，並且呈現車子的相關資料在畫面上。

三、專題實作部分

(1)模擬自駕車系統

1.車道線矯正:

根據車道線辨識的結果，分別出了幾種不同情況，根據以下情況的不同，車子會採取不同行動，例如當車資偏離車道時會自動偏回正向或是當車子在非十字路口的轉彎處時會正確的轉向。

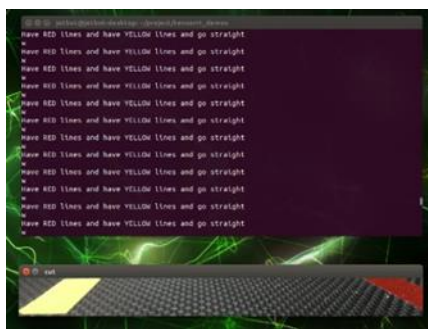


圖 11.偵測到紅線黃線直走

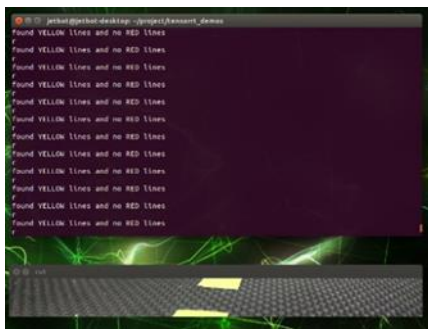


圖 12. 只偵測黃線右偏圖

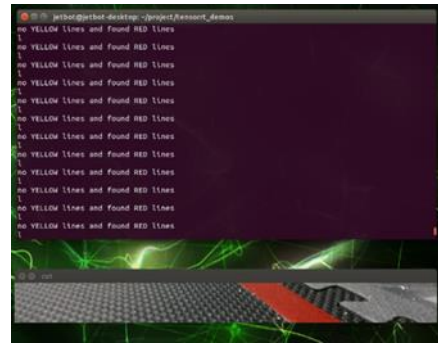


圖 13.只偵測到紅線左偏

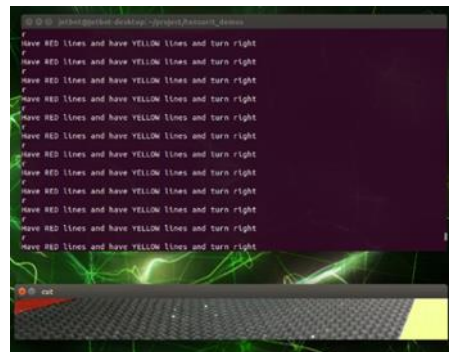


圖 14.偵測到對向紅線與黃線右偏

2.路牌與城市景觀(區域)辨識:

目前我們可辨識的物件有 15 種，例如車子藉由攝像頭拍攝到左轉時，先計算與路牌物件的距離，當車子到達與路牌間適當的距離時會辨識路牌種類，加入計算距離的目的是模擬真實道路駕駛時，駕駛不會一看到路牌就執行該動作，而是等到路口處才執行。接下來將辨識結果透過 Python 套件 GPIO 來驅動 L298N 控制馬達，使車體左轉，並回傳資料到資料庫。



圖 15.路牌與物件示意圖



圖 16.辨識到右轉

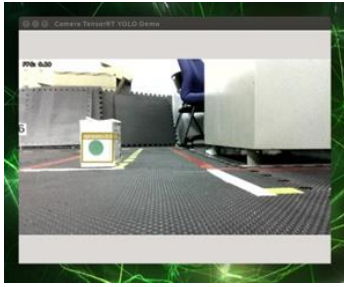


圖 17.辨識到綠燈

3.模擬定位與模擬該地區資料

我們在地區方面，劃分成三個區域，共有住宅、工業和郊區。根據不同的區域該地區的資料也有所不同，住宅區的速限為 30，因為人口架密集，工業區及郊區分別為 60 跟 90，而其他模擬資料 PM2.5(由於再同一房間 PM2.5 數值不會不同)工業區為最高，其次住宅、郊區。

街道的部分，我們設置了六個街道，分別為一街到六街，再透過街道和區域交集來計算出車子現在的所在位置。

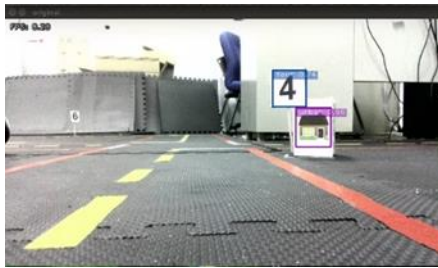


圖 18. 住宅區 4 街

根據圖 18 區域與街道交集的結果，我們會傳送資料「住宅區 4 街」到資料庫，並在網頁和 APP 上以地圖及文字顯示車子位置達到定位效果。

4.停車系統

使用者可以決定在哪一地點停車，例如:公司與住宅，當使用者在 APP 上按下住宅的按鈕時，車子自動行駛到住宅的車庫停放，而停放條件須為看到停車場的圖示才適用。

5.登入系統

一個家中可能有不同的使用者，例如爸爸、媽媽、哥哥等，這時我們就可以透過使用者登入系統，來了解目前是誰在使用這台自駕車。

(2)網頁及 APP 資料呈現與功能

利用 Python 連結 MySQL 資料庫實作數據分析，將所有使用者資料、所在地區、PM 值、車速、目前狀態，放入資料庫中，再透過網頁抓取資料，最後將其統整並列出資料於網頁及 APP 上。

1.網頁

將拍攝的物件影像回傳至電腦，並利用

YOLOv3-tiny 進行深度學習，建構 DNN 網路，配合學習好的 model 依照辨識出的路牌、區域、號誌，放入各項數據參數進行智慧解析，可以知道目前自駕車所在區域與在哪一條街道上，並把資料透過 Python 傳到資料庫上，網站會即時呈現資料，並且還會用語音進行播報以防視障者無法觀看網頁上的即時地圖。

利用網站平台可以讓使用者不需親自坐在車裡，就可以即時查看目前車體所在地區位置，而進行使用者想做的事。例如戰爭時，透過網路，不須親赴戰場前線，即可進行偵查；或是空氣汙染嚴重地區，會危害人體，不須觀察員親自前往，即可遠端查看位置以及測量 PM 值。

網頁還設有公佈欄，管理員可以發布公告訊息，而所有的使用者皆可看見；也有聯絡我們的功能，透過問題回報，我們可以更清楚問題所在，對系統或車體來進行維護及改善，使我們能更貼近使用者。



圖 19.網頁首頁

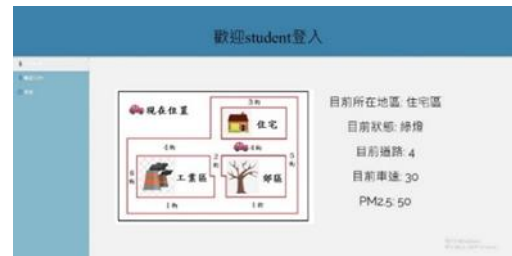


圖 20.網頁呈現資料與小地圖



圖 21.網頁功能列表及聯絡我們

2.APP

使用者若是身旁沒有電腦也可以利用 APP 得知車子的資訊和即時地圖，讓使用者可以得知目前所在位置，也有自動語音播報可以目前位置。

在得知目前車體所在位置之後，可以透過 APP 的停車按鈕，讓車子停在使用者想停放的位置，使用者在車上按下 APP 的回家按鈕，自駕車會自動開回車庫；而使用者在車上按下 APP 的上班按鈕，自駕車會自動開到公司的車庫停放。



圖 22 .呈現資料與功能按鈕

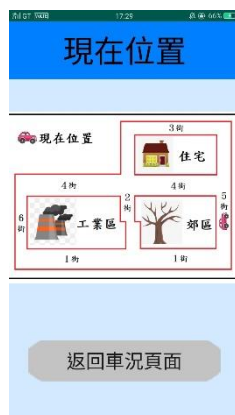


圖 23 .APP 呈現地圖

四、結論

本次的專題實作透過辨識號誌、路牌以及區域，可以達成定位系統，定位系統可以得知車子現在位置，使用者也可以藉由停車系統來指定停車的地點。車道線矯正使用了影像處理的技術，使車體能自動偏回軌道。APP 以及網頁可以讓使用者查看有關車子的相關資訊，再加上即時語音播報系統，能讓使用者用聽的就能知道車子目前的狀況。

在製作專題的過程中，我們學到了很多之前不曾接觸過的技術，在這過程中，碰到了許多的挫折，例如:程式的撰寫與除錯、環境的架設、車體電池的供電不足、程式吃到的效能太高導致 CPU 使用量接近 100%。在這跌倒又爬起的過程中，讓我們思考如何去運用大學這四年間所學的種種，並加以融會貫通。

五、參考文獻

參考網站:

【1】 Yolo : 基於深度學習的物件偵測 (含 YoloV3)

<https://mropengate.blogspot.com/2018/06/yolo-yolov3.html>

【2】 YOLO: Real-Time Object Detection

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

【3】 YoloV3cfg 檔解讀(一)

https://medium.com/@chih.sheng.huang821/深度學習-物件偵測_yolov1-yolov2和_yolov3-cfg-檔解讀-75793cd61a01

【4】 邊緣偵測懶人包-Canny 演算法

<https://medium.com/@bob800530/opencv-%E5%AF%A6%E4%BD%9C%E9%82%8A%E7%B7%A3%E5%81%B5%E6%B8%AC-canny%E6%BC%94%E7%AE%97%E6%B3%95-d6e0b92c0aa3>