

元智大學工程學院機械工程學系
Department of Mechanical Engineering
College of Engineering
Yuan Ze University

機電整合期末專題企劃書

1100826 王子晨

指導教授：吳昌暉 博士

摘要

本研究旨在設計一款基於影像處理技術的循線自走車，透過攝影機捕捉地面標線影像並使用影像處理算法進行自動識別，實現車輛自動跟隨標線行駛。車輛運用了麥克納姆輪系統，具備多向移動能力，提供更高的靈活性與操作精度。在車輛運行過程中，攝影機會即時捕捉前方及周圍的環境影像，通過影像處理技術進行辨識，判斷標線位置及車輛偏差，並依此進行位置與方向的調整，確保車輛沿標線精確行駛。研究中選用了 OV2640 攝影鏡頭作為感測器，並搭配 TT 馬達和 Arduino 控制器進行運動控制。車輛的運行指令由個人電腦進行計算，並利用 Wi-Fi 進行車輛與控制器之間的通訊。預處理部分將利用透視變換、灰階處理、高斯模糊等技術來處理影像，為後續的車道偵測與控制算法提供準確的數據支持。

目錄

摘要	i
1 整體功能	1
2 機械結構草圖	1
3 致動器的功能與數目列表	2
4 感測器的功能與數目列表	3
5 控制器的功能與數目列表	4
6 演算法設計	5
6.1 預處理	5
6.2 偵測	12
6.3 控制	14
7 人機界面的功能概述	16
8 研究進度與預估花費	16
9 參考文獻	16

圖目錄

1	機械結構草圖	1
2	麥克納姆輪 (直徑 58mm)	2
3	TT 馬達 [1]	3
4	透視變換流程圖	5
5	尚未進行透視變換之原始圖片	7
6	經透視變換後影像	7
7	實際測量地板尺寸	8
8	待測試之原始圖片	8
9	經灰階處理之圖片	9
10	經高斯模糊處理之圖片	10
11	經二值化處理之圖片	11
12	經邊緣偵測之圖片	11
13	尚未滑動之視窗	12
14	滑動後視窗	13
15	擬合後曲線與車輛行駛方向比較	14
16	研究計畫甘特圖	16

表目錄

1	致動器的功能與數目列表	2
2	TT 馬達規格 [1]	3
3	感測器的功能與數目列表	3
4	OV2640 攝影鏡頭規格 [2]	4
5	致動器的功能與數目列表	4
6	控制器規格表 [3][4]	4
7	MacBook Air M1 與比較機種規格	4
8	透視變換誤差	8
9	V_x 控制規則表	15
10	V_y 控制規則表	15
11	ω 控制規則表	15
12	預估花費	16

1 整體功能

本期末專題旨在設計一款基於影像處理技術的循線自走車。車輛將利用攝影機捕捉地面標線影像，並通過影像處理算法實現自動跟隨標線行駛。專題將融合麥克納姆輪系統，讓車輛能夠實現多向移動，提供更高的靈活性和操作精度。

在車輛運行過程中，攝影機將即時捕捉前方及周圍的環境影像，並利用影像處理技術進行辨識，判斷標線位置和車輛的當前偏差。車輛將根據這些信息，進行位置與方向調整，從而自動修正其行駛路徑，確保沿著標線精確行駛。

此外，通過麥克納姆輪的設計，車輛將具備前進、後退、左右側向移動以及旋轉等多種運動模式，這使得車輛在複雜路徑中具有更高的操作靈活性。

2 機械結構草圖

機械結構草圖與使用之麥克納姆輪如下：

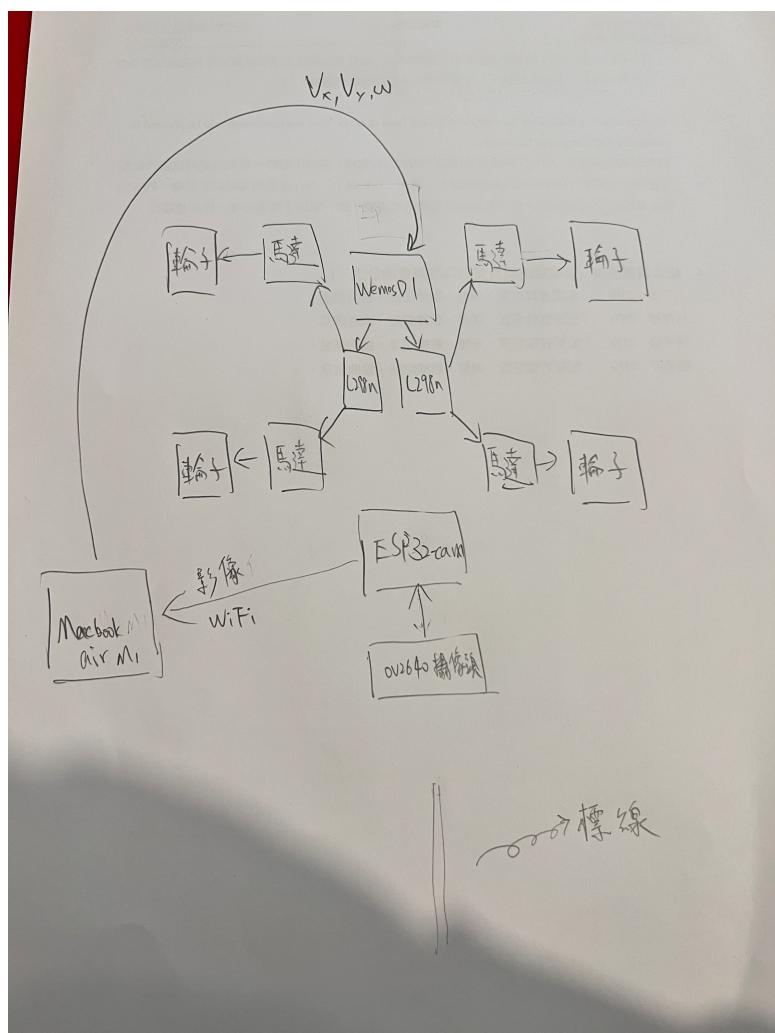


圖 1: 機械結構草圖



圖 2: 麥克納姆輪 (直徑 58mm)

3 致動器的功能與數目列表

採用 TT 馬達來驅動車輛，使其具備靈活的多向運動能力。下表是功能與數量 (如表 1 所示) 和規格表 (如表 2 所示)：

表 1: 致動器的功能與數目列表

項目	功能	數量
TT 減速馬達	每個麥克納姆輪提供獨立的驅動	4

表 2: TT 馬達規格 [1]

項目	規格
減速比	1:48
工作電壓	6V
空載電流	$\leq 240 \text{ mA}$
空載轉速	$270 \pm 10\% \text{ RPM}$
扭矩	$1.5 \text{ kgf} \cdot \text{cm}$



圖 3: TT 馬達 [1]

4 感測器的功能與數目列表

感測器部分將選用 OV2640 攝影鏡頭。以下是關於 OV2640 的功能與數目列表：

表 3: 感測器的功能與數目列表

項目	功能	數量
OV2640 攝影鏡頭	拍攝影像	1

表 4: OV2640 攝影鏡頭規格 [2]

項目	規格
感測器類型	1/4 吋 CMOS
解析度	1600×1200
幀率	15fps
水平視角	63°
垂直視角	47°

5 控制器的功能與數目列表

控制器的部分預計使用兩塊 Arduino 板子與一台個人電腦、搭配 l298n 進行控制。功能與數目列表如下表 (表 5、6、7) 所示。三台電腦間利用 Wi-Fi 進行通訊。

表 5: 致動器的功能與數目列表

項目	功能	數量
Wemos D1	接收個人電腦指令控制 l298n	1
ESP32-CAM	拍攝影像並傳輸至個人電腦	1
MacBook M1	影像偵測與控制演算法計算	1
l298n	麥克納姆輪	2

表 6: 控制器規格表 [3][4]

項目	Wemos D1	ESP32-CAM
處理器	ESP8266	ESP32
記憶體	4MB	4MB
無線連接	802.11 b/g/n Wi-Fi	802.11 b/g/n Wi-Fi
I/O 引腳	11	9
電源	5V	5V

表 7: MacBook Air M1 與比較機種規格

元件	MacBook Air M1
處理器 (CPU)	Apple M1 (8 核心)
記憶體 (RAM)	8GB Unified Memory
硬碟 (Storage)	256GB SSD
顯示卡 (GPU)	Apple M1 內建 GPU (7 核心)
作業系統 (OS)	macOS Sequoia
電池容量 (Battery)	49.9Wh

6 演算法設計

演算法的設計包括三個部分，分別是預處理、偵測、控制，運算全都由個人電腦 (MacBook Air M1) 進行運算。

6.1 預處理

預處理部分包括透視變換、灰階處理、高斯模糊、二值化、Canny 邊緣偵測，將接收到的影像進行處理，提供後續偵測使用。

6.1.1 透視變換

首先，對捕獲的影像進行透視變換，將影像轉換為以車輛為原點的二維座標系統，這樣有助於確定車輛與道路的相對位置，並為後續的計算提供準確的基準。

本專題將利用鏡頭往斜下的角度拍攝。採用逆透視變換 (Inverse Perspective Mapping, IPM)[5][6][7] 技術處理影像，可以將像素座標轉換為地面座標，取得影像中精確的距離。技術處理影像，可以將像素座標轉換為地面座標，取得影像中精確的距離。透視變換的核心在於去除透視效應，以便將影像中的道路線條轉換為平行的直線。這樣的變換有助於車輛識別路徑，使後續演算法能夠更容易地進行路徑規劃與車道偵測。

首先將影像像素座標系 (u, v) 轉換為圖像座標系 (x, y) 。加入相機的水平視角 $HFOV$ 與相機的垂直視角 $VFOV$ 、相機俯仰角 θ_c ，將圖像座標轉換為相機座標 (x_c, y_c, z_c) 。接著將相機座標系轉換為車輛標系 (x_w, y_w, z_w) ，以取得拍攝影像中內容物與車輛之相對位置。流程圖如下：

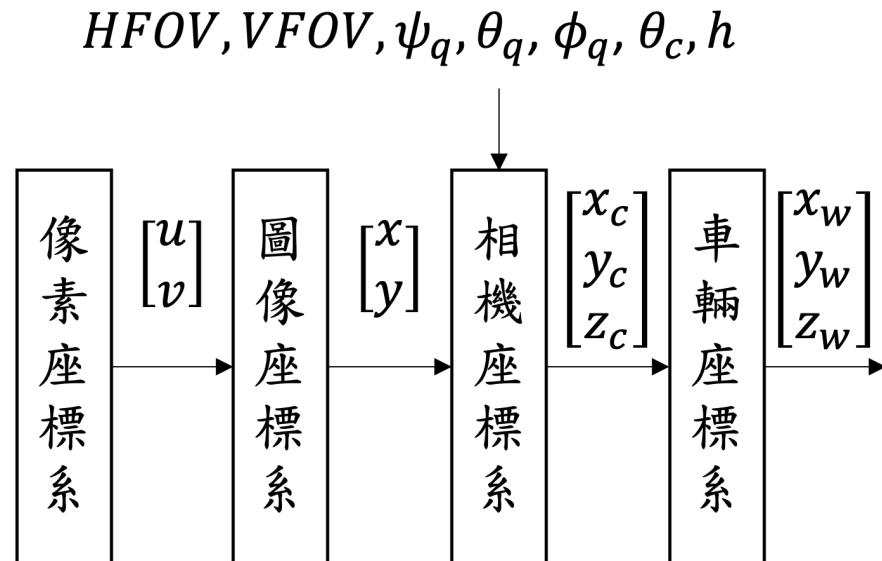


圖 4: 透視變換流程圖

將像素座標轉換為無人機相對於地平面的世界座標時，過程涉及相機的內參矩陣 K 、旋

轉矩陣 R 以及相機高度 h 。要算相機的內參矩陣首先需得出影像的水平焦距 f_x 、影像的垂直焦距 f_y 和影像的中心座標 (c_x, c_y) 。

影像的水平焦距 f_x 公式如式(1)， W 為影像的寬度，HFOV 為相機的水平視角。影像的垂直焦距 f_y 公式如式(2)， H 為影像的寬度，VFOV 為相機的垂直視角。

$$f_x = \frac{W}{2 \cdot \tan\left(\frac{HFOV}{2}\right)} \quad (1)$$

$$f_y = \frac{H}{2 \cdot \tan\left(\frac{VFOV}{2}\right)} \quad (2)$$

影像的中心座標 (c_x, c_y) 公式如式(3)、式(4)：

$$c_x = \frac{W}{2} \quad (3)$$

$$c_y = \frac{H}{2} \quad (4)$$

式(5)為相機內參矩陣 K 。

$$K = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \quad (5)$$

旋轉矩陣 R 用來表示相機相對於世界座標系的旋轉。此處忽略車輛進行中的晃動。旋轉矩陣 R 如下：

$$R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta_c) & -\sin(\theta_c) \\ 0 & \sin(\theta_c) & \cos(\theta_c) \end{bmatrix} \quad (6)$$

將像素座標系 (x, y) 轉換為圖像座標系的公式如式(7)：

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = K^{-1} \cdot \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \quad (7)$$

將圖像座標系轉換為相機座標系的公式如式(8)：

$$\begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = R \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad (8)$$

實際測試結果如下圖所示。將實際拍攝影像投影至以相機為原點的平面上，即可計算拍

攝目標物與車輛之距離。



圖 5：尚未進行透視變換之原始圖片



圖 6：經透視變換後影像

下圖 (如圖 7所示) 與下表 (如表 8所示) 為實際測量之結果，並計算誤差，若參數設置準確，絕對誤差大約為 0.7 公分左右。



圖 7: 實際測量地板尺寸

表 8: 透視變換誤差

MAE(m)	0.007
MRE(%)	1.168

6.1.2 灰階處理

將彩色影像轉換為灰階影像，這樣能有效減少計算量，同時保留影像的基本結構特徵，便於後續處理。這樣的轉換方式基於人眼對不同顏色敏感度的權重選擇，使得轉換後的影像仍然保持較高的可辨識度。

由行車圖片進行測試，圖 8為原始圖片，圖 9為經灰階處理之圖片。



圖 8: 待測試之原始圖片



圖 9: 經灰階處理之圖片

6.1.3 高斯模糊

使用高斯模糊濾波器來去除影像中的噪聲，這有助於平滑影像，減少後續處理中的干擾，尤其是在邊緣檢測時。高斯模糊的公式如下：

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \quad (9)$$

其中， σ 為標準差，決定模糊的強度。較大的 σ 值會導致更強的平滑效果，但可能會模糊掉重要的邊緣資訊。



圖 10: 經高斯模糊處理之圖片

6.1.4 二值化

將影像轉換為二值圖像，即只保留黑白兩種顏色，這有助於強調出車道線等重要特徵，便於後續的分析與處理。固定閾值法：設定一個固定的閾值 T ，當像素強度高於 T 時設為白色，否則設為黑色。固定閾值法的公式如下。

$$I_{bin}(x, y) = \begin{cases} 1, & I(x, y) > T \\ 0, & I(x, y) \leq T \end{cases} \quad (10)$$



圖 11: 經二值化處理之圖片

6.1.5 Canny 邊緣偵測

使用 Canny 邊緣偵測算子來提取影像中的邊緣信息，這有助於識別出車道的邊界或其他障礙物，為自走車的導航提供關鍵信息。



圖 12: 經邊緣偵測之圖片

6.2 偵測

圖像經過預處理後，接下來的步驟是偵測車道標線，並計算車輛的偏移量與偏移角度。透過適當的演算法，能夠從二值化影像或邊緣偵測結果中提取關鍵特徵，並進一步分析車輛的行進狀態，以利於後續控制。

6.2.1 掃描特定區域像素

將首先，從影像中掃描特定的區域，通過檢測這些區域的像素值來獲取相關的幾何特徵。在偵測車道標線時，並非整張影像都具有相同的重要性，因此可以選擇關鍵區域進行分析。例如，車輛前方的道路中央區域可能包含主要的行駛標線，而影像上方區域則可能包含過多的背景資訊，對偵測無實質幫助。將使用滑動視窗法，使用固定大小的窗口在影像中移動，對每個區域進行局部分析，以提取潛在的車道標線位置。

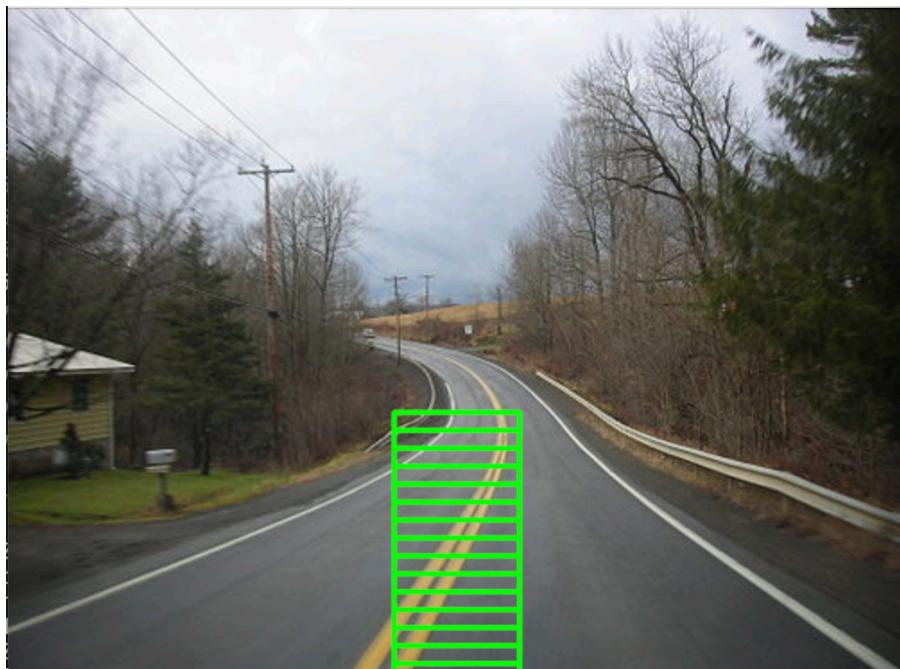


圖 13: 尚未滑動之視窗

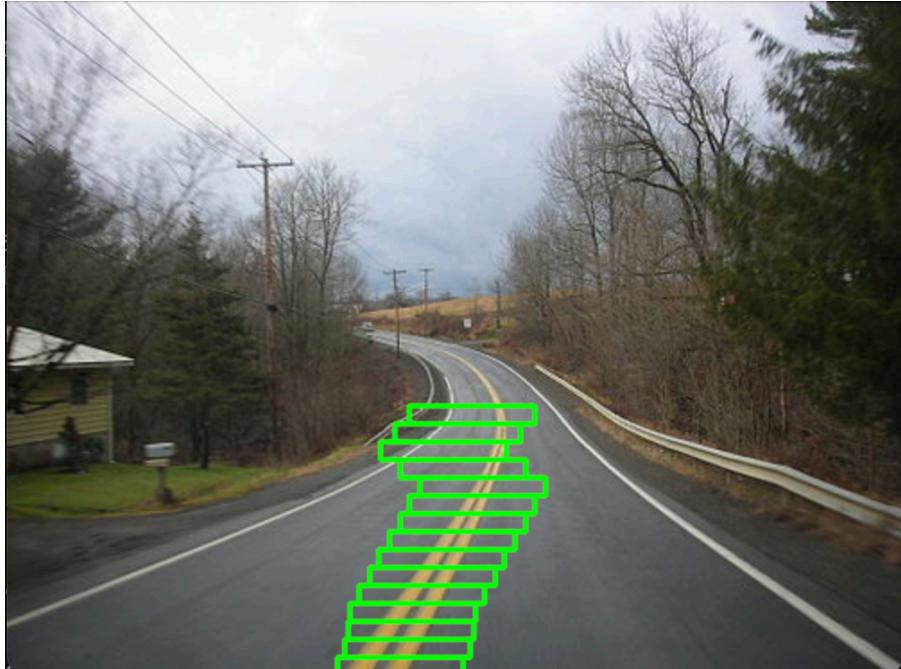


圖 14: 滑動後視窗

6.2.2 擬合曲線

根據掃描到的像素資料，使用多項式擬合來擬合出車道線或相關物體的曲線，這樣可以準確地描述其形狀。

當獲取到標線像素點後，下一步是利用數學模型來擬合標線的形狀。由於車道標線通常具有一定的連續性與平滑度，可以使用多項式擬合（Polynomial Fitting）來描述其形狀。

對於普通道路，車道標線通常可以近似為二次曲線：

$$y = ax^2 + bx + c \quad (11)$$

其中， a, b, c 為待求的係數。使用最小二乘法（Least Squares Method, LSM）來找到最適合這些數據點的曲線。

處理後如下圖所示，即可計算車輛的偏移量與偏移角度。

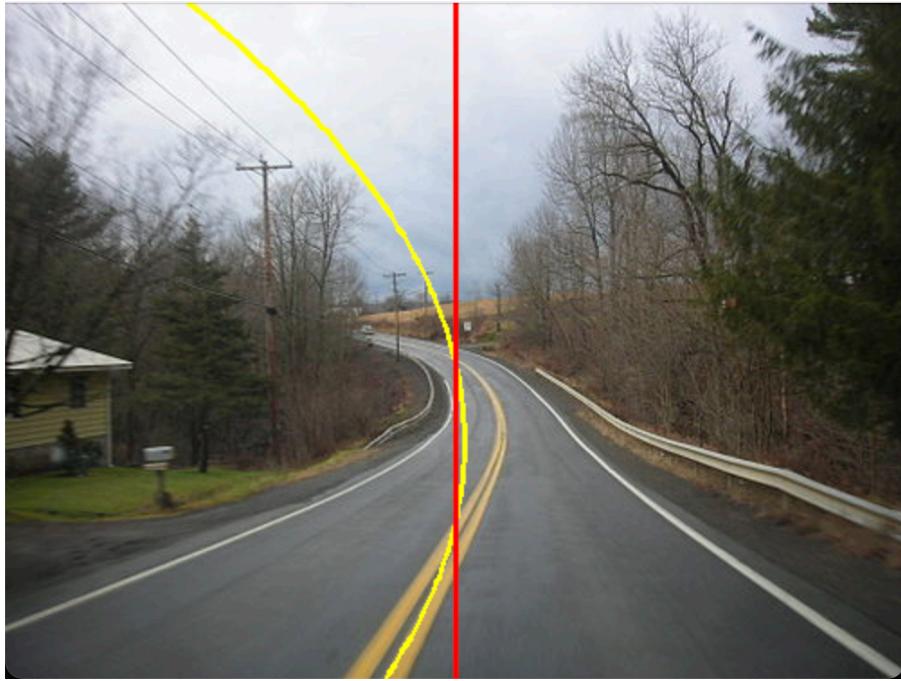


圖 15: 擬合後曲線與車輛行駛方向比較

6.2.3 計算偏移量、偏移角度

根據擬合的曲線，計算出車輛當前的偏移量（與目標軌跡的水平距離）和偏移角度（與車道或目標方向的夾角）。這些數據將作為車輛調整行駛路徑的依據。

6.3 控制

控制這一步分將使用模糊邏輯來實現全向輪之控制。

首先定義輸入變數與輸出變數。輸入變數為偏差角度 θ 即位置偏移量 p 。輸入變數為前進速度 V_x 、側向數度 V_y 與角速度 ω 。

隸屬函數將使用三角形隸屬函數 (Triangular Membership Function, Trimf)。

模糊規則設計如下表 (如表 9、表 10、表 11所示)，前進速度分為 S (Slow, 慢速)、M (Medium, 中速)、F (Fast, 快速)。側向速度分為 LL (Leftmost, 最左)、L (Left, 左)、Z (Zero, 中間)、R (Right, 右)、RR (Rightmost, 最右)。角速度分為 CCW2 (Counterclockwise 2, 強烈逆時針)、CCW (Counterclockwise, 輕微逆時針)、Z (Zero, 無旋轉)、CW (Clockwise, 輕微順時針)、CW2 (Clockwise 2, 強烈順時針)。

表 9: V_x 控制規則表

Angle($^\circ$) \ Position(cm)	BL	SL	Z	SR	BR
BL	S	S	M	S	S
SL	S	M	M	M	S
Z	M	M	F	M	M
SR	S	M	M	M	S
BR	S	S	M	S	S

表 10: V_y 控制規則表

Angle($^\circ$) \ Position(cm)	BL	SL	Z	SR	BR
BL	RR	R	Z	L	LL
SL	RR	R	Z	L	LL
Z	RR	R	Z	L	LL
SR	RR	R	Z	L	LL
BR	RR	R	Z	L	LL

表 11: ω 控制規則表

Angle($^\circ$) \ Position(cm)	BL	SL	Z	SR	BR
BL	CW2	CW2	CW2	CW2	CW2
SL	CW	CW	CW	CW	CW
Z	Z	Z	Z	Z	Z
SR	CCW	CCW	CCW	CCW	CCW2
BR	CCW2	CCW2	CCW2	CCW2	CCW2

最後進行模糊推理與去模糊化，給定輸入 θ_{in} 和 p_{in} ，計算它們在模糊集合中的的隸屬度：

$$\mu_A(\theta_{in}), \quad \mu_B(p_{in}) \quad (12)$$

然後對於每個模糊規則 R_i ，使用最小推理法（Mamdani 推理法）計算輸出的強度：

$$\mu_{rule} = \min(\mu_A, \mu_B) \quad (13)$$

最後，使用重心法（Centroid Method）計算輸出，其中 y_i 是輸出變數的取值， μ_i 是對應的隸屬度。

$$y^* = \frac{\sum \mu_i y_i}{\sum \mu_i} \quad (14)$$

即可得到了模糊控制輸出的 V_x, V_y, ω 。已知麥克納姆輪的速度模型如下，可得知每一顆輪子之速度，即可換算成輪子所需轉速。

$$\begin{bmatrix} V_{FL} \\ V_{FR} \\ V_{RL} \\ V_{RR} \end{bmatrix} = \begin{bmatrix} 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 \\ -1 & 1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \begin{bmatrix} V_x \\ V_y \\ \omega \end{bmatrix} \quad (15)$$

7 人機界面的功能概述

人機界面將主要提供使用者監控控自走車，顯示車輛的實時影像及運行狀態（如行駛速度、偏差角度等）。顯示車輛是否正確沿著標線行駛，若有偏離，顯示偏離的程度。

8 研究進度與預估花費

表 12: 預估花費

項目	花費 (元)
esp32-cam	330
Wemos D1	200
tt 減速馬達	50*4

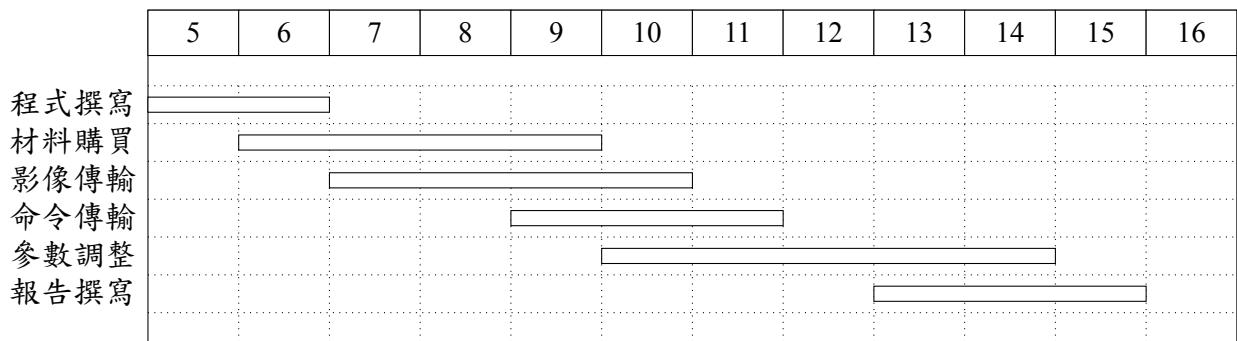


圖 16: 研究計畫甘特圖

9 參考文獻

- [1] JustMakeIt, Tt 玩具馬達. <https://hackmd.io/@JustMakeIt/HJI5gkpSo>.

- [2] 台灣感測器, *Arduino 200 萬像素 ov2640 影像擷取模組*. <https://www.taiwansensor.com.tw/product/arduino-200>
- [3] 台灣物聯科技, *Lolin (wemos) d1 wifi arduino uno 開發板*. <https://www.taiwaniot.com.tw/product/wemos-d1-wifi-arduino-uno-esp8266>
- [4] ICShop, *Esp32-cam - 高效能物聯網相機模組，內建 wi-fi 和藍牙功能*. <https://www.icshop.com.tw/products/368030501448>.
- [5] S.-F. Lin, J.-Y. Chen, and H.-X. Chao, “Estimation of number of people in crowded scenes using perspective transformation,” *IEEE*, vol. 31, no. 6, 2001.
- [6] M. Bertozzi, A. Broggi, and A. Fascioli, “Stereo inverse perspective mapping: Theory and applications,” *Image and vision computing*, vol. 16, no. 8, pp. 585–590, 1998.
- [7] H. A. Mallot, H. H. Bülthoff, J. J. Little, and S. Bohrer, “Inverse perspective mapping simplifies optical flow computation and obstacle detection,” *Biological cybernetics*, vol. 64, no. 3, pp. 177–185, 1991.