

國立臺灣海洋大學

電機工程學系

機器人控制專題實作報告

指導教授：鄭智湧 博士

報告主題：全向輪自走車

學生姓名：林昕瑋

中華民國 113 年 01 月 12 日

摘要

本研究旨在探討以微控制器與眾多感測器去控制馬達以操作麥克納姆輪的移動方向與避障情況。本研究亦對於微控制器、基礎 I/O 與介面設計、感測器、馬達驅動、麥克納姆輪移動方向控制、系統整合與控制器設計進行一系列的整合與操作應用。藉由 Arduino 微電腦配合程式語言驅動達到自走車可在接收手機指令後進行自主移動，且遇到障礙物時可精準避開而不受擦撞。

目錄

摘要.....	II
目錄.....	III
第一章 緒論.....	1
一、 研究背景與動機	1
二、 專題目標.....	1
第二章 系統架構.....	3
一、 系統架構介紹	3
二、 機構製作工具介紹	4
三、 開發軟體介紹	9
第三章 方法與實踐	10
一、 主要技術方法	10
二、 軟體開發流程	11
三、 硬體設計與搭建	15
第四章 實驗結果.....	16
一、 完整程式碼	16
二、 全向輪自走車成品	22
三、 實際結果.....	22
第五章 討論.....	23
一、 成果意義與局限性探討	23
二、 專題目標達成程度	23
第六章 結論與未來展望	24
一、 結論.....	24
二、 未來研究方向建議	24

三、	系統改進的可能性	24
第七章	參考文獻.....	26

圖目錄

圖 2-1-1	全向輪系統架構流程圖	3
圖 2-2-1	Arduino Mega2560 硬體介紹圖	4
圖 2-2-2	ADIO-L298N 馬達驅動板腳位介紹圖	6
圖 2-2-3	HC06 硬體介紹	6
圖 2-2-4	HC-SR04 超聲波感測器硬體介紹	8
圖 3-1-1	麥克納姆輪合力移動方向	10
圖 3-2-1	馬達腳位與轉速控制程式碼	11
圖 3-2-2	馬達輸出設定與飽率設定	12
圖 3-2-3	四個全向輪的輪子移動方向程式碼	13
圖 3-2-4	車體移動方向程式碼	13
圖 3-2-5	手機輸入特定指令時進入 Distance 迴圈(超聲波測距)	14
圖 3-2-6	超聲波測距程式碼	14
圖 4-1-1	完整程式碼 github 連結 Qrcode	16
圖 4-2-1	全向輪自走車前視圖與俯視圖	22

第一章 緒論

一、 研究背景與動機

隨著科技的日新月異，各種機器人技術的發展已逐漸滲透至各行各業，特別是物流、醫療、家居清潔等領域。隨著人工智慧、機器學習與感測技術的進步，機器人已不僅是執行簡單任務的工具，其逐漸發展成具備自主決策與靈活應對能力的智能系統。傳統機器人的移動方式以履帶或兩輪為主，儘管這些移動方式較為簡單，但在運動靈活性及控制精度上存在一定侷限性。

為解決上述問題，故決定以可實現任意方向運動的全向輪(麥克納姆輪)作為機器人的移動方式。此外，隨著無線通訊技術的發展，藍芽技術成為機器人控制系統中的重要手段之一。藍牙控制技術簡化了機器人的操控流程，使用者可藉由自身的智慧手機與機器人進行連線，實現遠程控制機器人，此方式不僅降低了硬體需求，還增加了機器人系統的可操作性並提高了機器人運行的安全性。

在機器人的導航系統中，避障技術是實現自主運行的關鍵因素之一，故在避障系統的方式選擇上有較多顧慮與考量。考慮到金錢成本與使用穩定性，暫且選擇使用超聲波感測器作為機器人的避障系統。超聲波感測器發展成熟度較高，且成本較低廉，可說是機器人避障系統的極佳選擇。

二、 專題目標

本專題旨在設計並實現一款具備高度靈活性移動、自主避障與便捷操控能力的全向輪移動機器人系統，以滿足在多變環境中操作的需求與挑戰。具體目標包含以下幾點：

1. 設計全向輪移動平台

設計並建構基於全向輪技術的機器人底盤，以實現任意方向運動的能力並達到精確的運動控制，以提升機器人的靈活性。

2. 實現藍芽遠程控制

開發基於藍芽通訊的控制系統，並設計用戶友好的手機操作介面，以便於用戶通過智能手機實現對機器人的遠程操作，使用戶能對機體行進方向與速度進行控制。

3. 集成超聲波避障功能

藉由超聲波傳感技術以即時探測周遭環境中的障礙物，以確保機器人能夠

安全運行與移動，並嘗試開發避障演算法，使機器人能夠自主調整行進路徑，避免碰撞風險。

4. 系統整合與優化

完成機器人硬體與軟體的結合，其中包含控制系統的程式設計與算法實現，以確保機器人功能模組間的高效協作，並測試與優化系統性能，以確保其穩定性與可靠性。

5. 檢驗與應用場景探索

在不同場景中進行測試，驗證系統性能並評估實際應用的可行性，包含機器人在實驗環境中對移動的靈活性、避障準確性及藍芽控制穩定性進行測試。

第二章 系統架構

一、系統架構介紹

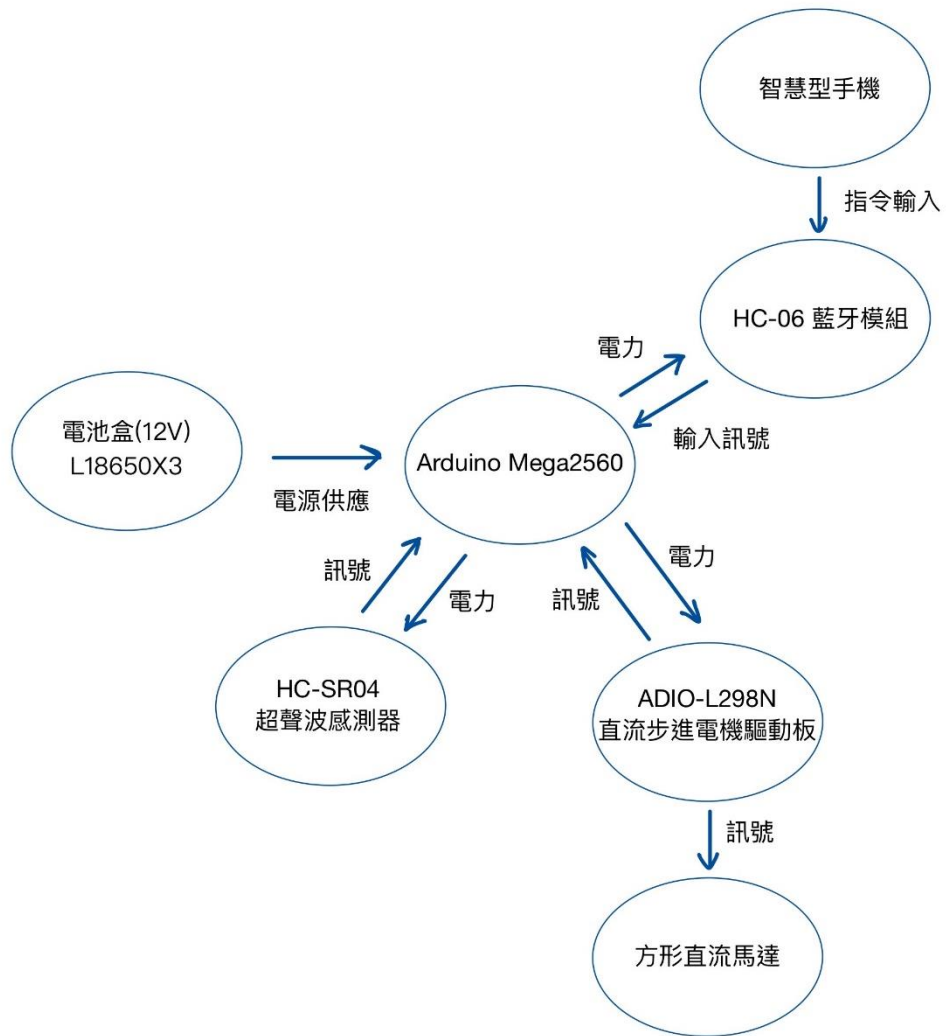


圖 2-1-1 全向輪系統架構流程圖

二、機構製作工具介紹

(一)Arduino Mega2560

1. Arduino Mega2560 硬體介紹

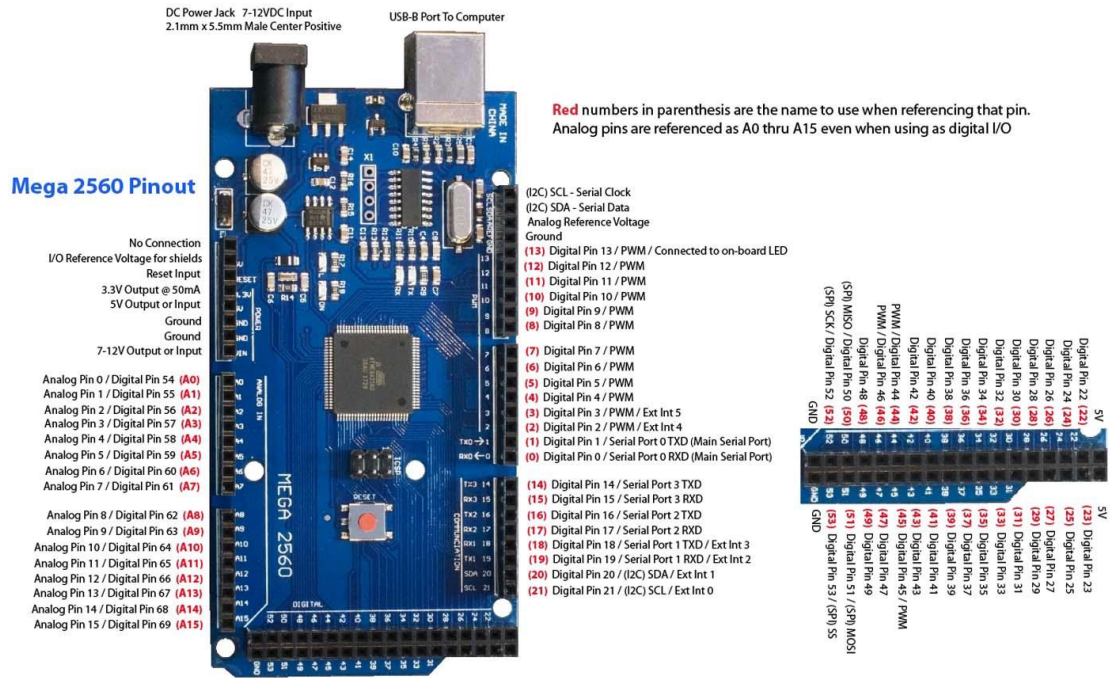


圖 2-2- 1¹Arduino Mega2560 硬體介紹圖

2. 核心微控制器

- (1) 微控制器: ATmega2560
- (2) 工作電壓: 5V
- (3) 輸入電壓(建議): 7-12V
- (4) 輸入電壓(極限): 6-20V
- (5) 時脈速度: 16MHz

3. 記憶體

- (1) Flash 記憶體: 256KB
- (2) SRAM: 8KB

¹ 圖片來源:

https://blogger.googleusercontent.com/img/b/R29vZ2xl/AVvXsEiKntB5sY3CSxTXwEgrtBKii43OB_P3DinlOnZyZ8-sjvCUpwGHmDQCtJzF5c1wbjNY_czNIPuy2_3_uXp1Erh2NYg4vMRnUx6d4NHRyhKfxeIXtRegHAux4zWCFX38FkoRAw68W1F0ufw/s1400/Mega2560%25E8%2585%25B3%25E4%25BD%258D%25E5%259C%2596.jpg, 最後參考日期:2025 年 1 月 11 日。

- (3.) EEPROM: 4KB
- 4. I/O 接口
 - (1.) 數位 I/O 腳位: 54 個(其中 15 個可作 PWM 輸出)
 - (2.) 類比輸入腳位: 16 個(10-bit 精度)
 - (3.) PWM 輸出腳位: 15 個
 - (4.) 硬體 UART(串口): 4 個
 - (5.) SPI 接口: 1 個
 - (6.) I2C 接口: 1 個
 - (7.) 外部中斷腳位: 6 個
- 5. 電器特性
 - (1.) 每個 I/O 腳位最大電流: 40mA
 - (2.) 3.3V 腳位最大電流: 50mA
- 6. 電源系統
 - (1.) 電源選項:
 - USB 供電(5V)
 - 外部供電(透過 DC 插座或 Vin 腳位)
 - (2.) DC 插座電壓範圍: 7-12V(建議)
 - (3.) Vin 腳位電壓範圍: 7-12V
- 7. 通訊功能
 - (1.) USB 接口
 - (2.) 4 個硬體串口(UART)
 - (3.) SPI 通訊
 - (4.) I2C 通訊
 - (5.) 支援其他軟體串口
- 8. 額外功能
 - (1.) LED 指示燈
 - L(13 腳位上的 LED)
 - 電源警示燈
 - RX/TX
 - (2.) 重置按鈕: 可手動重啟板子
 - (3.) ICSP 接口: 用於燒錄 bootloader 或與其他 AVR 微控制器通訊
 - (4.) 擴展板相容性: 支援大多數 Arduino Shield

(二)ADIO-L298N 馬達驅動板

1. ADIO-L298N 馬達驅動板介紹

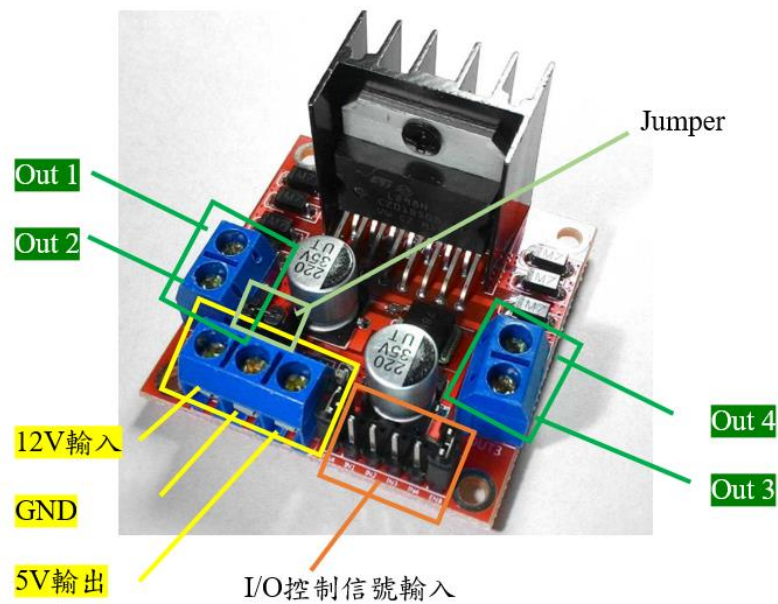


圖 2-2- 2ADIO-L298N 馬達驅動板腳位介紹圖

2. 腳位

- (1.) 12V 輸入與電源接上(電源供應器或是電池盒)
- (2.) GND 接腳與 Arduino 的 GND 接腳一同接至電源的 GND
- (3.) OUT1~4 四個腳位負責控制馬達轉向
- (4.) EnableA、EnableBrup 針腳控制馬達轉速

(三)HC06-藍芽模組

1. HC06 硬體介紹

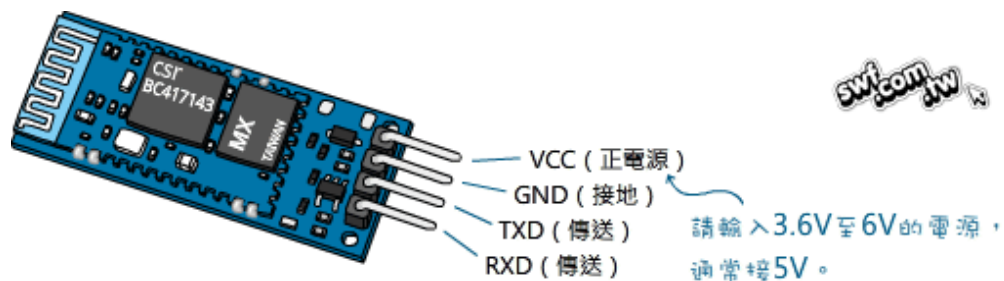


圖 2-2- 3²HC06 硬體介紹

² 圖片來源: https://swf.com.tw/images/books/arduino/2ed/hc-05_module_4pins.png，最後參考日期:2025 年 1 月 11 日

2. 電器參數
 - (1.) 工作電壓: 3.6-6V
 - (2.) 工作電流: 約 30mA
 - (3.) 腳位工作電壓: 3.3V (若與 5V 系統通訊，需使用電壓分壓電路)
3. 硬體介面
 - (1.) 腳位數: 4 個主要腳位
 - VCC: 電源輸入
 - GND: 接地
 - TXD: 模組的串口傳輸數據(輸出，需連接控制器的 RX)
 - RXD: 模組的串口接收數據(輸入，需連接控制器的 TX)
 - (2.) 串口通訊
 - 預設鮑率: 9600
 - 支援範圍: 1200-1382400
 - 通訊協議: UART，8 位數據位，1 位停止位，無校驗位
4. LED 指示燈
 - (1.) 閃爍狀態: 未連接(等待配對)
 - (2.) 恆亮狀態: 配對成功
5. 配對與名稱
 - (1.) 預設為 HC-06
 - (2.) 預設密碼: 1234 或 0000
 - (3.) 使用方式: 單向主/從模式
6. AT 指令
 - (1.) 檢查模組: AT
 - (2.) 設定鮑率: AT+BAUDX
 - (3.) 更改名稱: AT+NAME<name>
 - (4.) 更改密碼: AT+PIN<xxxx>0
7. 注意事項
 - (1.) 無法使用 ios 系統進行連線，僅接受安卓系統手機進行連線

(四)HC-SR04-超聲波感測器

1. HC-SR04 超聲波感測器硬體介紹

Pin	Description
VCC	power supply pin
Trig	trigger pin
Echo	Echo pin
GND	GND



圖 2-2- 4³HC-SR04 超聲波感測器硬體介紹

2. 電器參數

- (1.) 工作電壓: 5V (DC)
- (2.) 靜態電流: 小於 2mA
- (3.) 工作頻率: 40kHz
- (4.) 訊號輸出類型: TTL 級別訊號(5V)

3. 硬體介面

- (1.) 腳位數: 4 個
 - VCC: 電源輸入(5V)
 - GND: 接地
 - TRIG: 觸發腳位，輸入控制訊號
 - ECHO: 回應腳位，輸出高電平訊號持續時間表示測量距離
- (2.) 觸發條件: TRIG 腳位輸入至少 10 μ s 的高電平訊號(超聲波輸出訊號)

4. 時間參數

- (1.) 測量週期: 60ms
- (2.) 最小回波時間: 200 μ s (2cm)
- (3.) 最大回波時間: 25ms (400cm)

(五)L18650X3 電池盒

1. 基本資訊

- (1.) 適合 18650 鋰電池
- (2.) 單顆電池電壓: 3.7V

³ 圖片來源: https://shop.mirotek.com.tw/wp-content/uploads/2022/03/ESP32_10010_2-1024x307.png，最後參考日期:2025 年 1 月 11 日

(3.) 單顆電池容量範圍: 2000mAh~3500mAh

(4.) 電池數量: 3 個

(5.) 結構類型: 串聯與並聯

2. 串聯與並聯配置

(1.) 串聯:

- 輸出電壓: $3.7V \times 3 = 11.1V$
- 容量: 2500mAh (與單顆電池相同)

(2.) 並聯:

- 輸出電壓: 3.7V
- 容量: $3 \times \text{單顆電池容量}(2500mAh) = 7500mAh$

三、開發軟體介紹

Arduino IDE

Arduino 是一個開源電子原型開發平臺，源自於義大利的 Ivrea 互動設計研究所，組合了硬體與軟體，能夠幫助使用者設計和構建互動式的電子裝置，其簡化了複雜的程式撰寫流程，使其開發介面(Arduino IDE)容易上手與使用。通過它，開發者能快速地把想法實現，專注在創造與學習。Arduino 不僅是一種工具，更是一個社群，有許多協作與資源可供使用。

Arduino 最主要的軟體環境是 Arduino IDE，也就是綜合開發環境。Arduino IDE 提供一個簡單易用的編程界面，適合專業人士和初學者。它支援多種平臺，如 Windows、MacOS 和 Linux，並使用 C 和 C++ 語言進行程式編寫。通過 IDE 中的內建範例，使用者可以快速熟悉 Arduino 的基本功能和實現方式。它還提供串列通信的監視工具，可以用來觀察或測試數據，提升實作的便利性。

為了給開發者更大的自由，Arduino IDE 也提供了強大的函數庫管理功能。通過安裝庫，使用者可以寫出複雜的工程測試，如串列通信、SPI 和 I2C 等通訊協定。一些庫更是充分利用了社群的集體智慧，讓開發者能夠將模塊化的解決方案直接應用在實際項目中。

第三章 方法與實踐

一、 主要技術方法

1. 麥克納姆輪

麥克納姆輪是一種特殊設計的萬向輪，可以讓載具實現全向運動(包含前進、後退、側移或是任意角度的斜向運動)。此種輪子是由多個裝配在輪緣上的滾輪組成，滾輪的表面有一定的角度(通常為 45 度)，相對於輪子的旋轉平面。其輪框通常採用金屬或高強度塑膠製成；滾輪以橡膠或聚胺酯材料製造而成，提供一定程度的摩擦力。

四個輪子中，將兩兩相同滾輪傾斜角度的輪子各自安裝在車體底盤斜對角，利用輪子上的滾輪產生力的分量(分解成水平方向與垂直方向)，再結合馬達對輪子運轉的轉速可以達到操控車體全方位的移動。下圖為四顆輪子的操控方向所產生的合力移動方向：

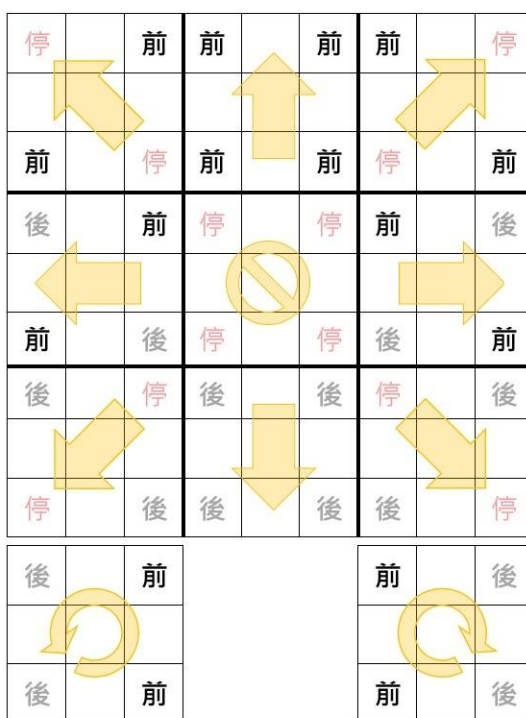


圖 3-1- 1⁴麥克納姆輪合力移動方向

⁴ 圖片來源: <https://lh3.googleusercontent.com/UHoIOPGmEDaWFLNNP->

2. 手機藍芽連線控制

藍芽模組 HC-06 是一種支援藍芽 2.1+EDR 規範的藍芽晶片模組，可讓安卓系統的裝置與此種模組進行藍芽連線

將藍芽模組 HC-06 與安卓手機進行藍芽連線，並利用手機程式 Arduino Bluetooth Control 使我們能以手機輸入指令並且經由藍芽模組將指令傳送到 Arduino Mega2560 中，藉此控制麥克納姆輪的移動方向來操控車體的移動方向。

3. 超聲波避障

超聲波感測器是一種藉由超聲波反射來進行測距的感測器，當超聲波遇到障礙物時會反射，感測器接收反射後的超聲波之後藉由 Arduino Mega2560 進行運算距離達到量測障礙物與車體之間距離的功能。

通常的距離計算公式為：

$$\text{距離}(cm) = \frac{\text{高電平持續時間}(\mu s)}{58} \quad \text{或} \quad \text{距離}(m) = \frac{\text{高電平持續時間}(\mu s)}{5800}$$

二、軟體開發流程

使用 Arduino IDE 介面進程式碼開發，以下為程式碼開發流程：

1. 程式碼調整方形馬達轉速

```
int ENA1 = 2;      int ENA2 = 8;
int IN1 = 3;        int IN5 = 9;
int IN2 = 4;        int IN6 = 10;
int PWMSpeed1 = 90; int PWMSpeed3 = 150;
//左後輪--1       //左前輪--3
int IN3 = 6;        int IN7 = 12;
int IN4 = 5;        int IN8 = 11;
int ENB1 = 7;       int ENB2 = 13;
int PWMSpeed2 = 145; int PWMSpeed4 = 120;
//右後輪--2       //右前輪--4
```

圖 3-2-1 馬達腳位與轉速控制程式碼

```

void setup() {
  Serial.begin(9600); //設定監控視窗鮑率
  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);
  pinMode(ENA1, OUTPUT);
  pinMode(IN3, OUTPUT);
  pinMode(IN4, OUTPUT);
  pinMode(ENB1, OUTPUT);

  pinMode(IN5, OUTPUT);
  pinMode(IN6, OUTPUT);
  pinMode(ENA2, OUTPUT);
  pinMode(IN7, OUTPUT);
  pinMode(IN8, OUTPUT);
  pinMode(ENB2, OUTPUT);
}

```

圖 3-2-2 馬達輸出設定與鮑率設定

2. 將馬達運轉方向確認後對照麥克納姆輪運轉方向讓程式碼指令可使四顆方形馬達依照全向輪移動方向要求進行運轉與操控

<pre> void FRONT1() { digitalWrite(IN1, HIGH); digitalWrite(IN2, LOW); analogWrite(ENA1, PWMspeed1); } //1前 void FRONT2() { digitalWrite(IN3, HIGH); digitalWrite(IN4, LOW); analogWrite(ENB1, PWMspeed2); } //2前 void BACK1() { digitalWrite(IN2, HIGH); digitalWrite(IN1, LOW); analogWrite(ENA1, PWMspeed1); } //1後 void BACK2() { digitalWrite(IN4, HIGH); digitalWrite(IN3, LOW); analogWrite(ENB1, PWMspeed2); } //2後 </pre>	<pre> void FRONT3() { digitalWrite(IN5, LOW); digitalWrite(IN6, HIGH); analogWrite(ENA2, PWMspeed3); } //3前 void FRONT4() { digitalWrite(IN7, LOW); digitalWrite(IN8, HIGH); analogWrite(ENB2, PWMspeed4); } //4前 void BACK3() { digitalWrite(IN6, LOW); digitalWrite(IN5, HIGH); analogWrite(ENA2, PWMspeed3); } //3後 void BACK4() { digitalWrite(IN8, LOW); digitalWrite(IN7, HIGH); analogWrite(ENB2, PWMspeed4); } //4後 </pre>
---	---

```

void STOP1()
{
    digitalWrite(IN1, LOW);
    digitalWrite(IN2, LOW);
} //1停

void STOP2()
{
    digitalWrite(IN3, LOW);
    digitalWrite(IN4, LOW);
} //2停

void STOP3()
{
    digitalWrite(IN5, LOW);
    digitalWrite(IN6, LOW);
} //3停

void STOP4()
{
    digitalWrite(IN7, LOW);
    digitalWrite(IN8, LOW);
} //4停

```

圖 3-2-3 四個全向輪的輪子移動方向程式碼

```

void FRONT() //往前移動
{
    FRONT1();
    FRONT2();
    FRONT3();
    FRONT4();
}

void BACK() //往後移動
{
    BACK1();
    BACK2();
    BACK3();
    BACK4();
}

void LF() //左前移動
{
    FRONT1();
    STOP2();
    STOP3();
    FRONT4();
}

void RF() //右前移動
{
    STOP1();
    FRONT2();
    FRONT3();
    STOP4();
}

void L() //往左平移
{
    FRONT1();
    BACK2();
    BACK3();
    FRONT4();
}

void LB() //左後移動
{
    STOP1();
    BACK2();
    BACK3();
    STOP4();
}

void R() //往右平移
{
    BACK1();
    FRONT2();
    FRONT3();
    BACK4();
}

void RB() //右後移動
{
    BACK1();
    STOP2();
    STOP3();
    BACK4();
}

void STOP() //停止
{
    STOP1();
    STOP2();
    STOP3();
    STOP4();
}

void Lround() //逆時針自轉
{
    BACK1();
    FRONT2();
    BACK3();
    FRONT4();
}

void Rround() //順時針自轉
{
    FRONT1();
    BACK2();
    FRONT3();
    BACK4();
}

```

圖 3-2-4 車體移動方向程式碼

3. 結合藍芽模組程式碼達到輸入特定指令時可操控馬達轉向以達到機器人準確移動

```
void loop()
{
  // FRONT();
  if(Serial.available())//當輸入任意指令時
  {
    Distance();//啟動超聲波測距
  }
  // L();
}
```

圖 3-2- 5 手機輸入特定指令時進入 Distance 迴圈(超聲波測距)

4. 加諸超聲波測距程式碼，使機器人在遇見障礙物後能移動輪子以達到避障
- 使用測距公式：

$$\text{距離(m)} = \frac{\frac{\text{高電平持續時間}(\mu\text{s})}{2}}{29} = \frac{\text{高電平持續時間}(\mu\text{s})}{58}$$

```
void Distance()
{
  float duration, distance;
  digitalWrite(trig, HIGH);
  delayMicroseconds(100);
  digitalWrite(trig, LOW);
  duration = pulseIn(echo, HIGH);
  distance = (duration/2)/29;
  Serial.print("distance = ");
  Serial.print(distance);
  Serial.println("cm");
  if(distance >= 20)//與障礙物距離20cm以上時
  {
    FRONT();//向前行
    delay(10);
  }else //與障礙物距離小於20cm時
  {
    STOP();//停下
    delay(500);
    L();//向左平移
    delay(2800);
  }
}
```

圖 3-2- 6 超聲波測距程式碼

三、 硬體設計與搭建

下列為硬體設計相關介紹：

1. 底盤設計

使用壓克力板作為機器人底盤，藉由雷切機將壓克力板切割置合適大小

2. 馬達驅動

以方形馬達作為驅動麥克納姆輪的驅動方式，將四顆馬達分別安裝於底盤下方的四個角落，再將方形馬達連接馬達驅動板(ADIO-L298N)達到控制馬達轉速與轉向的效果

3. 感測模組

(1.) 安裝藍芽模組 HC-06 使手機可藉由輸入指令來操控馬達轉向

(2.) 安裝超聲波感測器使機器人達到避開障礙物的效果

4. 控制模組

使用 Arduino Mega2560 來進行系統控制

5. 電源系統

將電池盒正極端連接 Arduino Mega2560 的 Vin 腳位以及其他需要電源端的腳位，接地端連接在其 GND 端，達到為整體系統供電的效果

第四章 實驗結果

一、完整程式碼



圖 4-1-1 完整程式碼 github 連結 Qrcode

以下為完整程式碼:

```
int ENA1 = 2;
int IN1 = 3;
int IN2 = 4;
int PWMspeed1 = 90; //轉速設定
//左後輪--1
int IN3 = 6;
int IN4 = 5;
int ENB1 = 7;
int PWMspeed2 = 145; //轉速設定
//右後輪--2

int ENA2 = 8;
int IN5 = 9;
int IN6 = 10;
int PWMspeed3 = 150; //轉速設定
//左前輪--3

int IN7 = 12;
int IN8 = 11;
int ENB2 = 13;
int PWMspeed4 = 120; //轉速設定
//右前輪--4
char incomingChar;
```

```

////////超聲波感測器設定////////
const int trig = A0;
const int echo = A1;
const int inter_time = 1000;
int time = 0;

void setup() {
  Serial.begin(9600);
  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);
  pinMode(ENA1, OUTPUT);
  pinMode(IN3, OUTPUT);
  pinMode(IN4, OUTPUT);
  pinMode(ENB1, OUTPUT);

  pinMode(IN5, OUTPUT);
  pinMode(IN6, OUTPUT);
  pinMode(ENA2, OUTPUT);
  pinMode(IN7, OUTPUT);
  pinMode(IN8, OUTPUT);
  pinMode(ENB2, OUTPUT);

  pinMode(echo, INPUT);
  pinMode(trig, OUTPUT);

}

void loop()
{
  if(Serial.available())//手機輸入任意字母時啟動超聲波測距
  {
    Distance();//超聲波測距

  }
}

////////////////////////////////////
void Distance()
{
  float duration, distance;
  digitalWrite(trig, HIGH);
  delayMicroseconds(100);
  digitalWrite(trig, LOW);
  duration = pulseIn(echo, HIGH);
  distance = (duration/2)/29;
}

```

```

// Serial.print("Data:");
// Serial.print(time/1000);
Serial.print("distance = ");
Serial.print(distance);
Serial.println("cm");
// time = time + inter_time;
// delay(inter_time);
if(distance >= 20) //檢測到物體距離機器人距離超過 20cm 時
{
    FRONT(); //向前行
    delay(10);
}else
{
    STOP(); //停止移動
    delay(500);
    L(); //向左移動
    delay(2800);
}
}

void FRONT1()
{
    digitalWrite(IN1, HIGH);
    digitalWrite(IN2, LOW);
    analogWrite(ENA1, PWMspeed1);
} //1 前

void FRONT2()
{
    digitalWrite(IN3, HIGH);
    digitalWrite(IN4, LOW);
    analogWrite(ENB1, PWMspeed2);
} //2 前

void FRONT3()
{
    digitalWrite(IN5, LOW);
    digitalWrite(IN6, HIGH);
    analogWrite(ENA2, PWMspeed3);
} //3 前

void FRONT4()
{
    digitalWrite(IN7, LOW);
    digitalWrite(IN8, HIGH);
    analogWrite(ENB2, PWMspeed4);
} //4 前

```

```
void BACK1()
{
    digitalWrite(IN2, HIGH);
    digitalWrite(IN1, LOW);
    analogWrite(ENA1, PWMspeed1);
} //1 後
```

```
void BACK2()
{
    digitalWrite(IN4, HIGH);
    digitalWrite(IN3, LOW);
    analogWrite(ENB1, PWMspeed2);
} //2 後
```

```
void BACK3()
{
    digitalWrite(IN6, LOW);
    digitalWrite(IN5, HIGH);
    analogWrite(ENA2, PWMspeed3);
} //3 後
```

```
void BACK4()
{
    digitalWrite(IN8, LOW);
    digitalWrite(IN7, HIGH);
    analogWrite(ENB2, PWMspeed4);
} //4 後
```

```
void STOP1()
{
    digitalWrite(IN1, LOW);
    digitalWrite(IN2, LOW);
} //1 停
```

```
void STOP2()
{
    digitalWrite(IN3, LOW);
    digitalWrite(IN4, LOW);
} //2 停
```

```
void STOP3()
{
    digitalWrite(IN5, LOW);
    digitalWrite(IN6, LOW);
} //3 停
```

```
void STOP4()
{
```

```

    digitalWrite(IN7, LOW);
    digitalWrite(IN8, LOW);
} //4 停

////////////////////////////////////

void FRONT()
{
    FRONT1();
    FRONT2();
    FRONT3();
    FRONT4();
}

void BACK()
{
    BACK1();
    BACK2();
    BACK3();
    BACK4();
}

void LF()
{
    FRONT1();
    STOP2();
    STOP3();
    FRONT4();
}

void RF()
{
    STOP1();
    FRONT2();
    FRONT3();
    STOP4();
}

void L()
{
    FRONT1();
    BACK2();
    BACK3();
    FRONT4();
}

void R()
{

```

```
    BACK1();  
    FRONT2();  
    FRONT3();  
    BACK4();  
}
```

```
void LB()  
{  
    STOP1();  
    BACK2();  
    BACK3();  
    STOP4();  
}
```

```
void RB()  
{  
    BACK1();  
    STOP2();  
    STOP3();  
    BACK4();  
}
```

```
void STOP()  
{  
    STOP1();  
    STOP2();  
    STOP3();  
    STOP4();  
}
```

```
void Lround()  
{  
    BACK1();  
    FRONT2();  
    BACK3();  
    FRONT4();  
}
```

```
void Rround()  
{  
    FRONT1();  
    BACK2();  
    FRONT3();  
    BACK4();  
}
```

二、 全向輪自走車成品

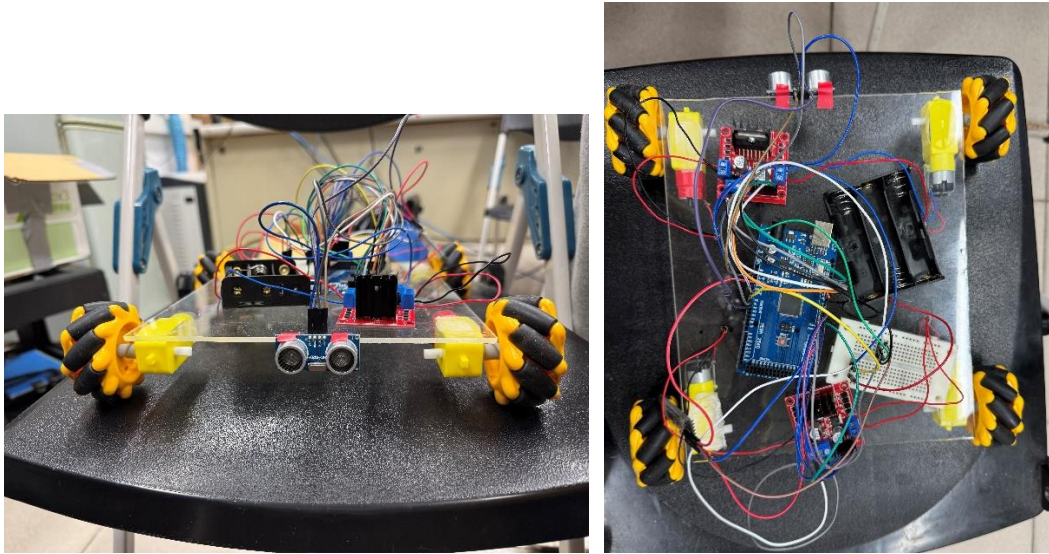


圖 4-2-1 全向輪自走車前視圖與俯視圖

三、 實際結果

成果影片：[全向輪自走車成果影片](#)

第五章 討論

一、 成果意義與局限性探討

1. 成果意義

本次專題實作的成果展現了全向輪在平坦地面上全方位移動的功能，結合手機藍芽連線操控與超聲波避障讓我們看到未來自走車自主移動的可能性，同時也可以去深入思考，此項移動與控制技術的應用與發展可能性。

2. 局限性探討

(1.) 藍芽連線裝置匹配問題

在本次實作中，可發現藍芽連線系統存在一定程度的不穩定性，除了經常斷開連線外另一個致命缺點是 HC-06 模組僅限於安卓系統的裝置，因此使用 ios 系統的裝置接無法使用。在現今社會中，使用 ios 系統裝置的受眾眾多，因此將藍芽模組更新可能是需要改善的項目之一。

(2.) 超聲波避障偵測盲區

在本次實作中遇到的另外一項問題是超聲波避障。儘管超聲波避障在成本上較為低廉，使用上也較為簡單，但考慮到車體的大小與超聲波的單面感測存在一定程度的偵測盲區，改善超聲波感測器或是改以其他技術進行檢測障礙物可能成為未來需要進行鑽研的項目。

二、 專題目標達成程度

本專題旨在設計並實現一款具備高度靈活性移動、自主避障與便捷操控能力的全向輪移動機器人系統，從實驗成果來說，儘管專題目標項目皆有實現，然而在精準控制移動與自主避障的準確性上仍有待加強，亦屬於未來需要再對此次專題研究進行深入研究與製作的內容。

第六章 結論與未來展望

一、 結論

本次專題實作中比較令我印象深刻的內容是硬體設備上的修繕與更換，首先是方形馬達，由於其連接的銅片很容易斷裂，而且斷裂後很難和皆回來，因此每次作專題前都要忐忑地確認馬達是否完好無損，很不幸地，每次去作專題時，當天必定有至少一顆馬達的銅片斷裂，導致每次都要花時間更換馬達並轉移車輪再重新焊接與連接底板。再者是底板大小，由於車輪較小，底板一開始選得太大導致車輪被底板壓著，無法正常運轉，後面在進行底板切割時學長姐還讓我嘗試到許多除了雷切機以外的切割與打磨新奇工具，算是蠻特殊的經驗。

本次專題製作花費大約一個學期的時間，從最初車體大小設計，底盤材料使用，馬達組裝，程式碼撰寫與系統整合等，每個階段皆花費了一定程度的時間去進行設計與改良，乃至到後來的加裝避障系統以及多次的測試亦或是硬體設備維修等皆是本次專題研究所探索並學習到的內容。儘管過程有些艱辛，甚至專題成果有些不如預期，但反而讓我更堅定地想將研究成果進行更為深入且完整的研究與探討。

二、 未來研究方向建議

根據實作成果與討論結果，我認為未來可朝著車體移動方向準確度控制以及避障系統精確度控制兩主題進行深入研究，不僅是因為這兩個項目是本次專題實作尚未達到的目標，更是因為車體移動方向的準確控制與避障系統的精確度對於真正的自走車具有一定程度的重要性與參考必要性。

三、 系統改進的可能性

以下幾點為系統改進的可能性探討

1. 馬達轉速改善

我認為馬達轉速改善可基於 PID 控制系統進行調整，以達到準確控制

馬達轉速，以避免實作中經常遇到不同條件下馬達轉速差異導致車體移動方向不穩定的情況。

2. 藍芽連接模組改善

我認為未來可以將藍芽模組替換成 HM-10 或 ESP32 等相關模組以達到無論是安卓系統還是 ios 系統皆可進行控制的效果。

3. 避障系統改善

此項目為我未來想優先深入研究的內容，由於前一章所題到的超聲波感測器造成偵測盲區的問題，我認為也許未來能以其他偵測障礙物的模組或感測器來替代超聲波感測器並使得偵測效果更加準確與完整。目前有參考的一個可能項目是改使用光達偵測，此感測器不但可使偵測精確度上升，亦可以擴大偵測範圍，使偵測範圍擴張至車體週圍，不再僅限於車體前方。

第七章 參考文獻

[1.] Arduino Mega2560 介紹:

<https://ee543.blogspot.com/2020/12/arduino-mega2560.html>

[2.] 麥克納姆輪介紹:

<https://sites.google.com/site/whlee1990/mechatronics/%E9%BA%A5%E5%85%8B%E7%B4%8D%E5%A7%86%E8%BC%AA%E5%B0%8F%E8%BB%8A/%E9%BA%A5%E5%85%8B%E7%B4%8D%E5%A7%86%E8%BC%AA%E5%85%A8%E5%90%91%E8%BC%AA%E5%B0%8F%E8%BB%8A%E9%81%8B%E4%BD%9C%E5%8E%9F%E7%90%86>

[3.] HC-06 模組介紹:<https://swf.com.tw/?p=693>

[4.] HC-SR04 模組介紹: <https://blog.jmaker.com.tw/sr04/>

[5.] HC-SR04 模組介紹: <https://shop.mirotek.com.tw/iot/esp32-start-10/>

[6.] L298N 模塊介紹: <https://www.ariat-tech.tw/blog/l298n-dc-motor-drive-module-features,pinout,usage-and-application.html>