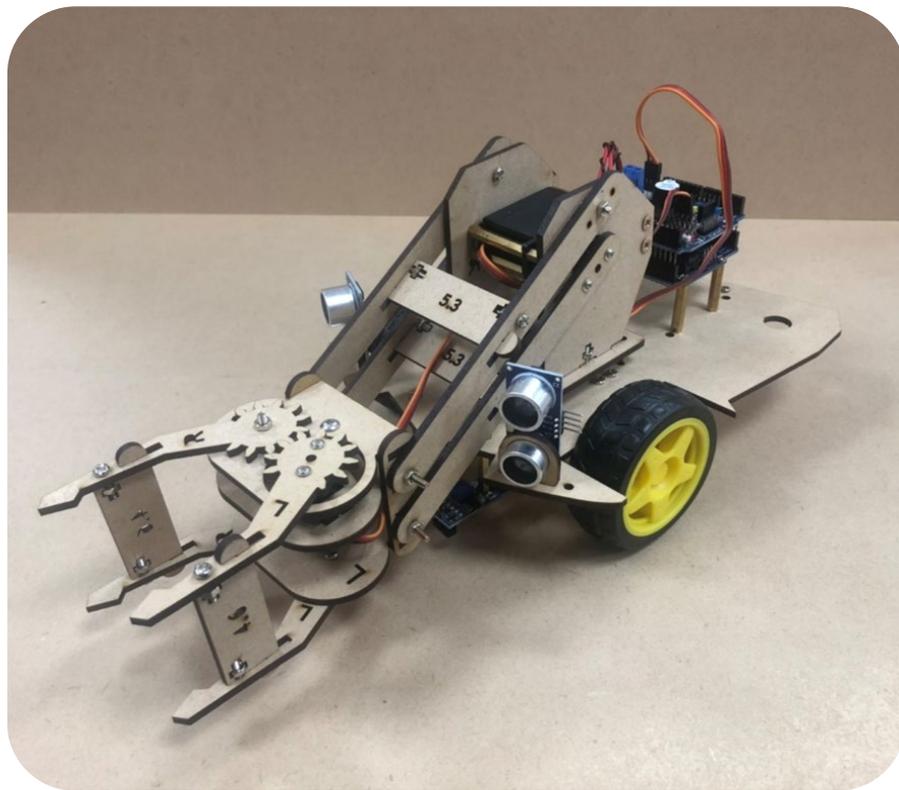


# START! 智慧小車



網站QRCODE ▲

自造大師團隊

網站連結：<https://reurl.cc/ybpRE>

資料下載：<https://reurl.cc/GjE5A>

# 目錄



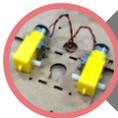
單元1 認識Arduino



單元2 智慧小車組裝



單元3 程式環境設定



單元4 TT直流馬達操作



單元5 紅外線感測器操作



單元6 超音波感測器操作



單元7 伺服馬達操作

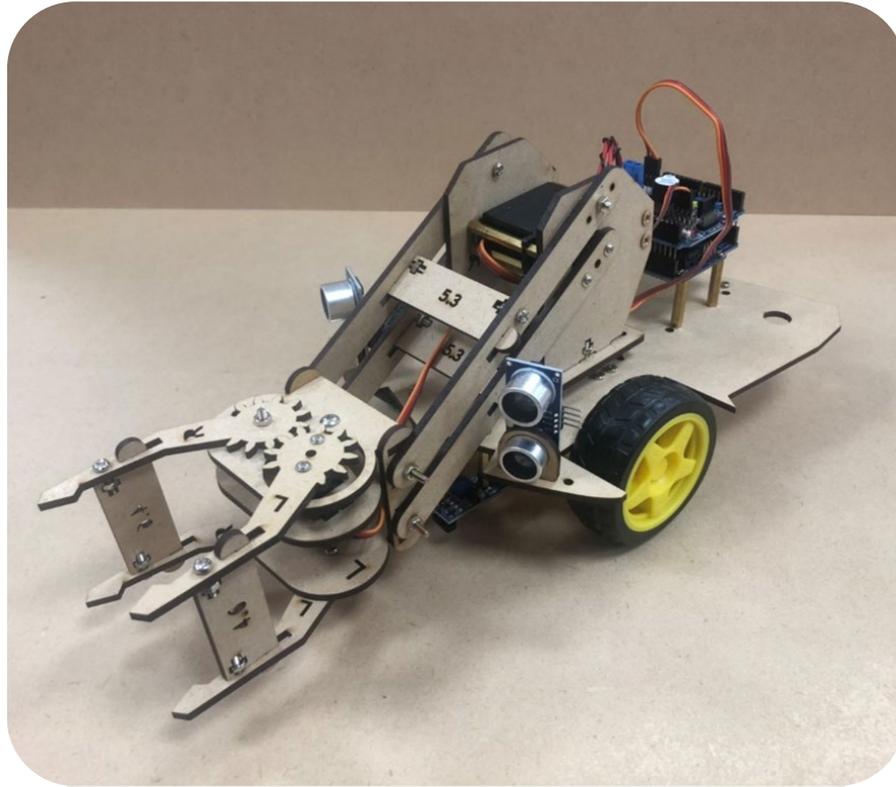


單元8 手機藍芽遙控操作



單元9 綜合練習

# START! 智慧小車

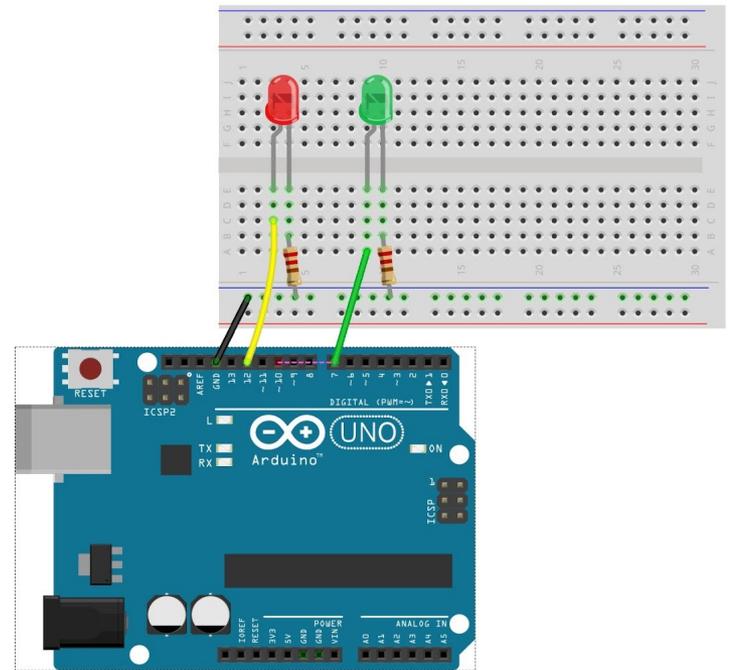
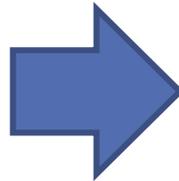
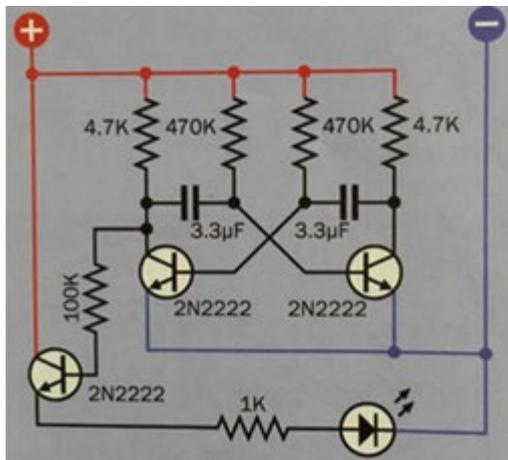
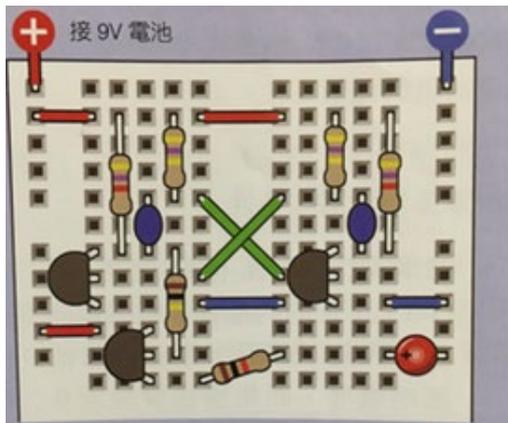


-單元1 認識**ARDUINO**-

# 由傳統數位邏輯電路到程式電路

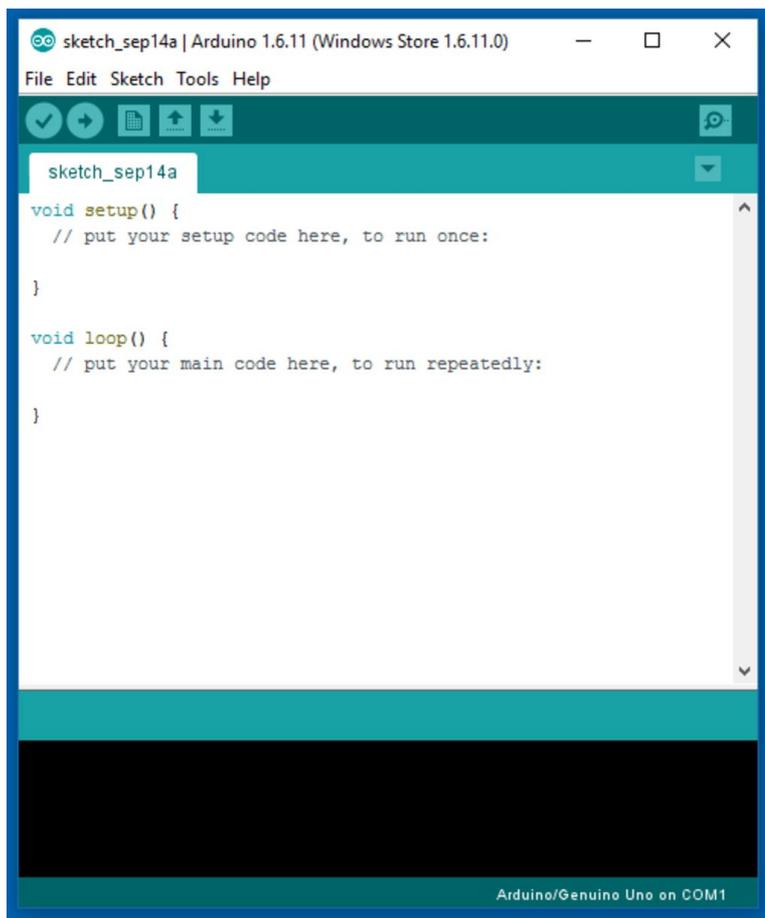
近年流行的 **Arduino**到底在夯什麼？

使用**Atmel AVR** 微處理器及**I/O**介面板，  
程式電路可以大幅降低電子配線的難度。



<<傳統數位邏輯電路使用較多電子元件  
且接線複雜

# Blink

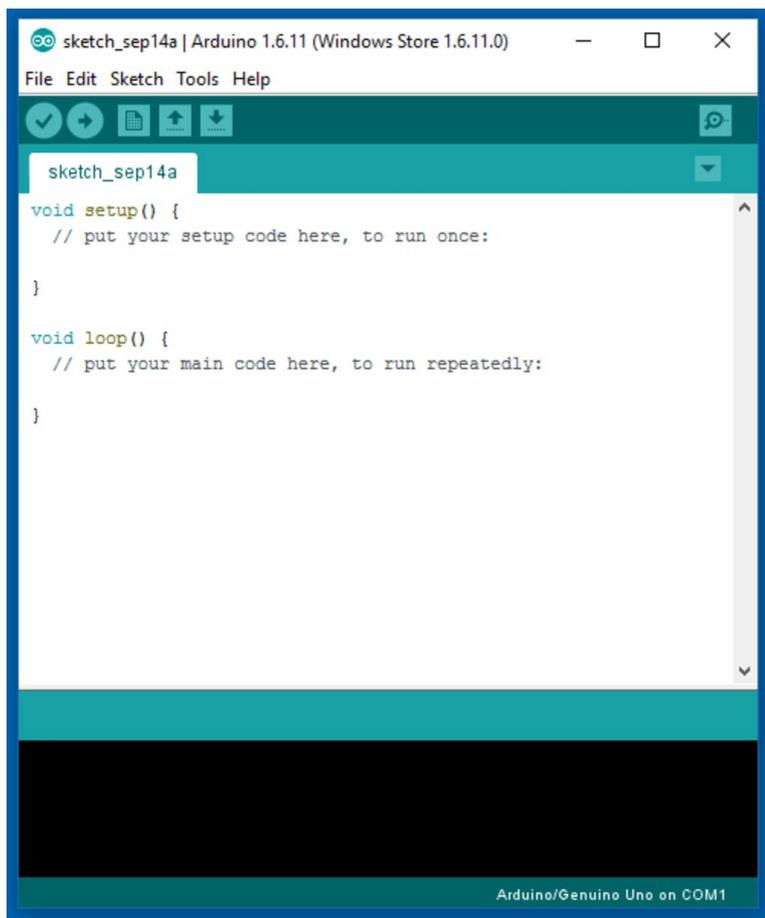


類似類似 C 語言的  
開發環境：

```
void loop () {
    digitalWrite (LED_1, HIGH);
    delay (1000 );
    digitalWrite (LED_1, LOW);
    delay (1000);
}
```

試著想想兩顆燈泡一明一滅  
程式碼要如何修改？

# Blink

A screenshot of the Arduino IDE interface. The window title is "sketch\_sep14a | Arduino 1.6.11 (Windows Store 1.6.11.0)". The menu bar includes "File", "Edit", "Sketch", "Tools", and "Help". Below the menu bar is a toolbar with icons for checkmark, play, upload, and download. The main text area shows the following code:

```
sketch_sep14a
void setup() {
  // put your setup code here, to run once:
}

void loop() {
  // put your main code here, to run repeatedly:
}
```

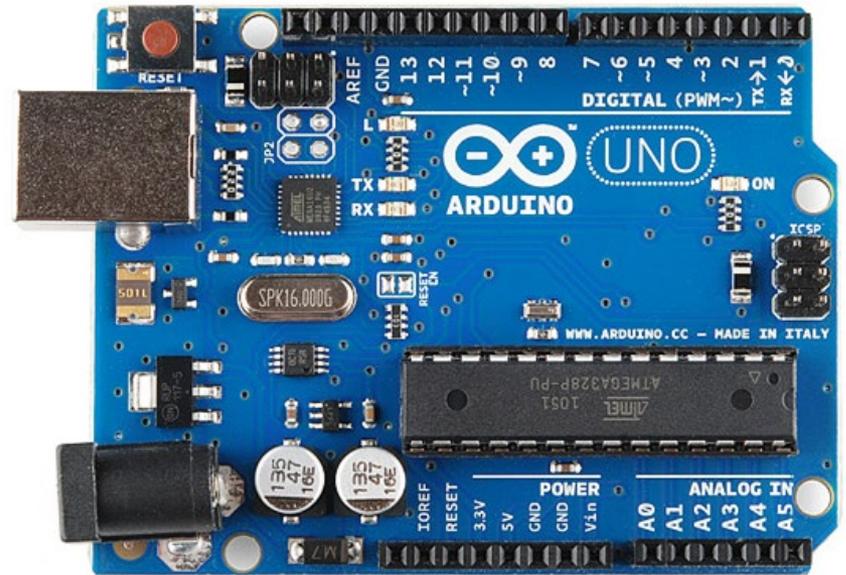
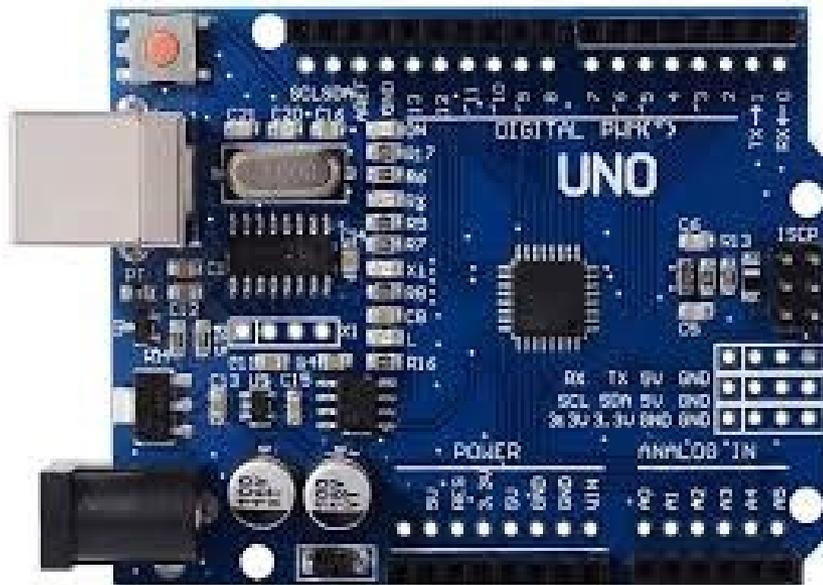
The status bar at the bottom indicates "Arduino/Genuino Uno on COM1".

```
void loop () {
  digitalWrite (LED_1, HIGH);
  delay (1000 );
  digitalWrite (LED_1, LOW);
  delay (1000};
  digitalWrite (LED_2, HIGH);
  delay (1000 );
  digitalWrite (LED_2, LOW);
  delay (1000};
}
```

Arduino 雖然多了程式控制，但比起傳統邏輯電路還是簡單許多。

# ARDUINO

簡單來說，**Arduino**利用針腳的訊號輸出，可以控制電燈明滅、馬達運轉、訊息傳遞等。利用針腳來讀取訊號，則可以接收訊息，或是量測環境狀況。

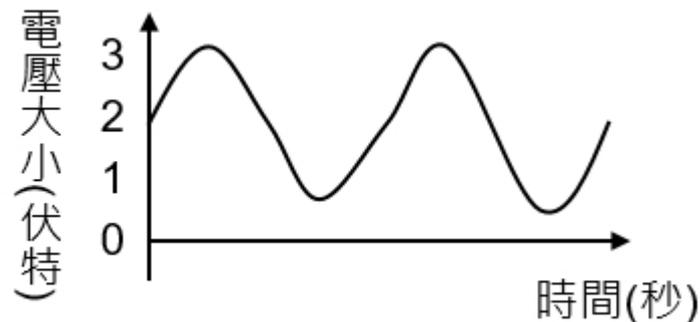


ps.副廠的主板需安裝CH340驅動程式

# 類比與數位

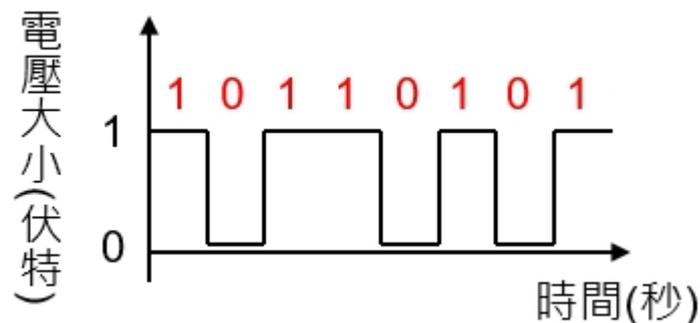
## 類比訊號 ( Analog signal )

日常生活中，我們聽到的聲音、看到的影像，大多屬於類比訊號，如右圖中這種「連續的訊號」即稱為類比訊號。像是人類講話的聲音是漸漸變大或漸漸變小，隨著聲音的大小，麥克風的電壓也會漸漸變大或漸漸變小，所以麥克風就是一種類比訊號的產品。



## 數位訊號 ( Digital signal )

就是我們經常說，在電腦的世界只有的0跟1，與類比訊號不同，數位訊號不是連續的，只有高電位(1)與低電位(0)的差別，可以想像成是電燈的開關，只有開燈跟關燈兩種狀態，不會有開一半或關一半的情況發生。

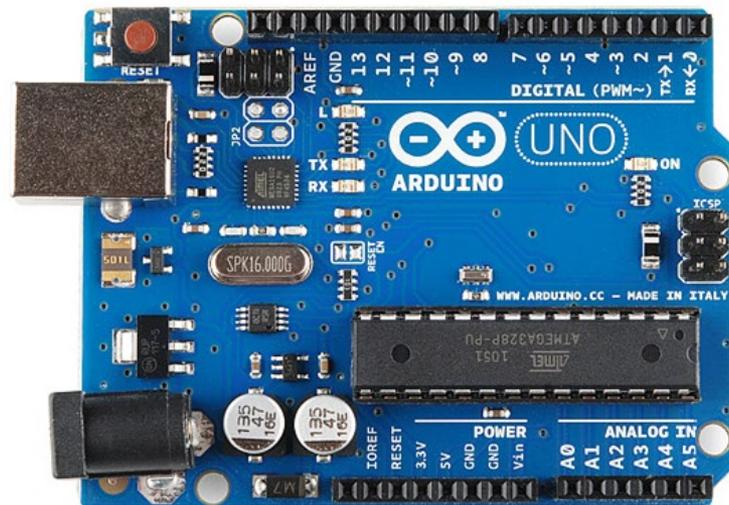
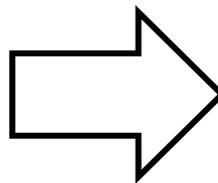


想想看：生活中有哪些資訊是類比的？又有哪些是數位的呢？

# ARDUINO

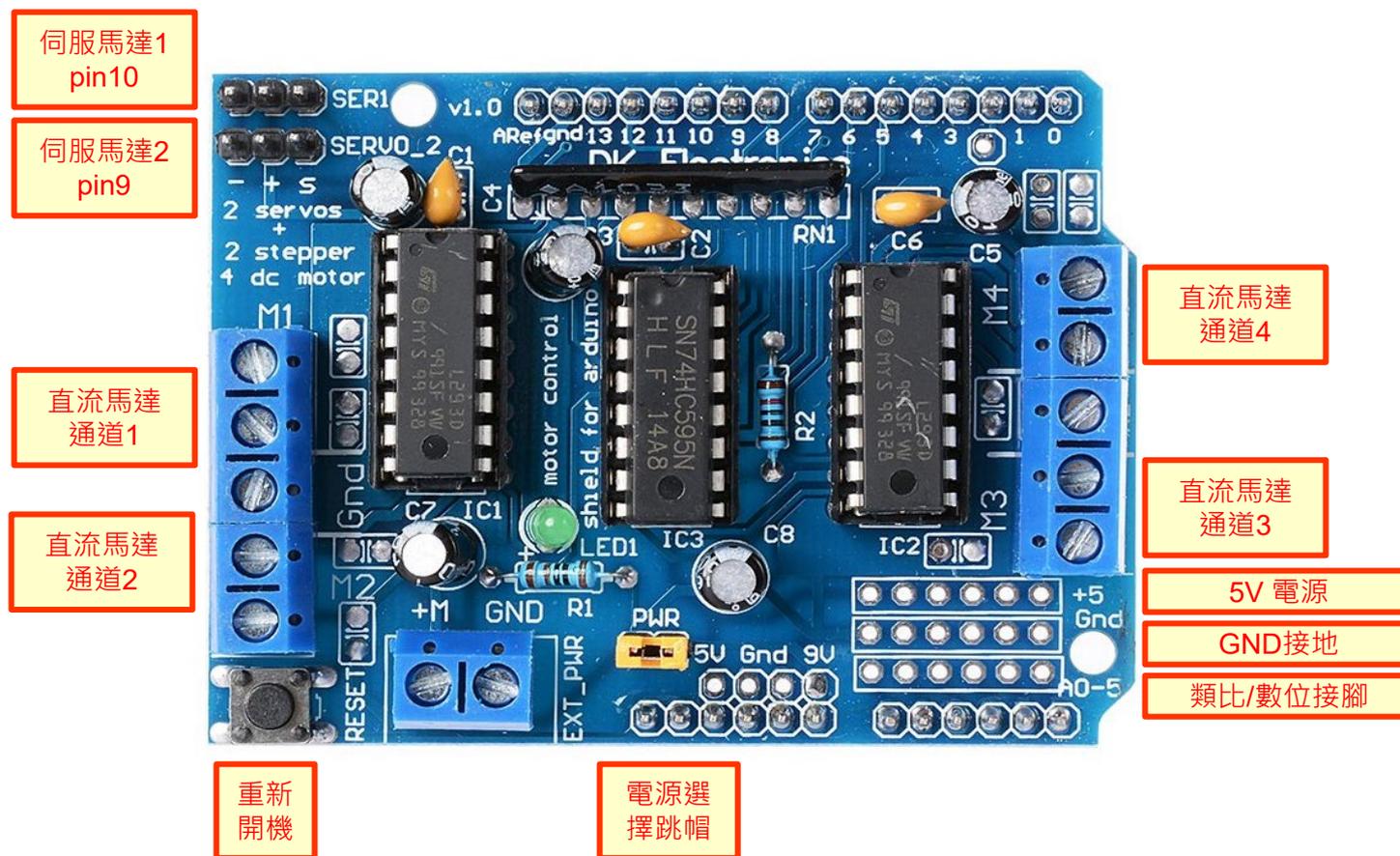
控制	腳位	用途範例
數位輸入	D0-D13 A0-A5	判斷開關是否按下
數位輸出		控制燈泡明滅
類比輸入	A0-A5	讀取環境狀態
類比輸出 (PWM)	D3、D11 D5、D6 D9、D10	調整強弱、頻率

仔細看，PWM針腳前有~記號



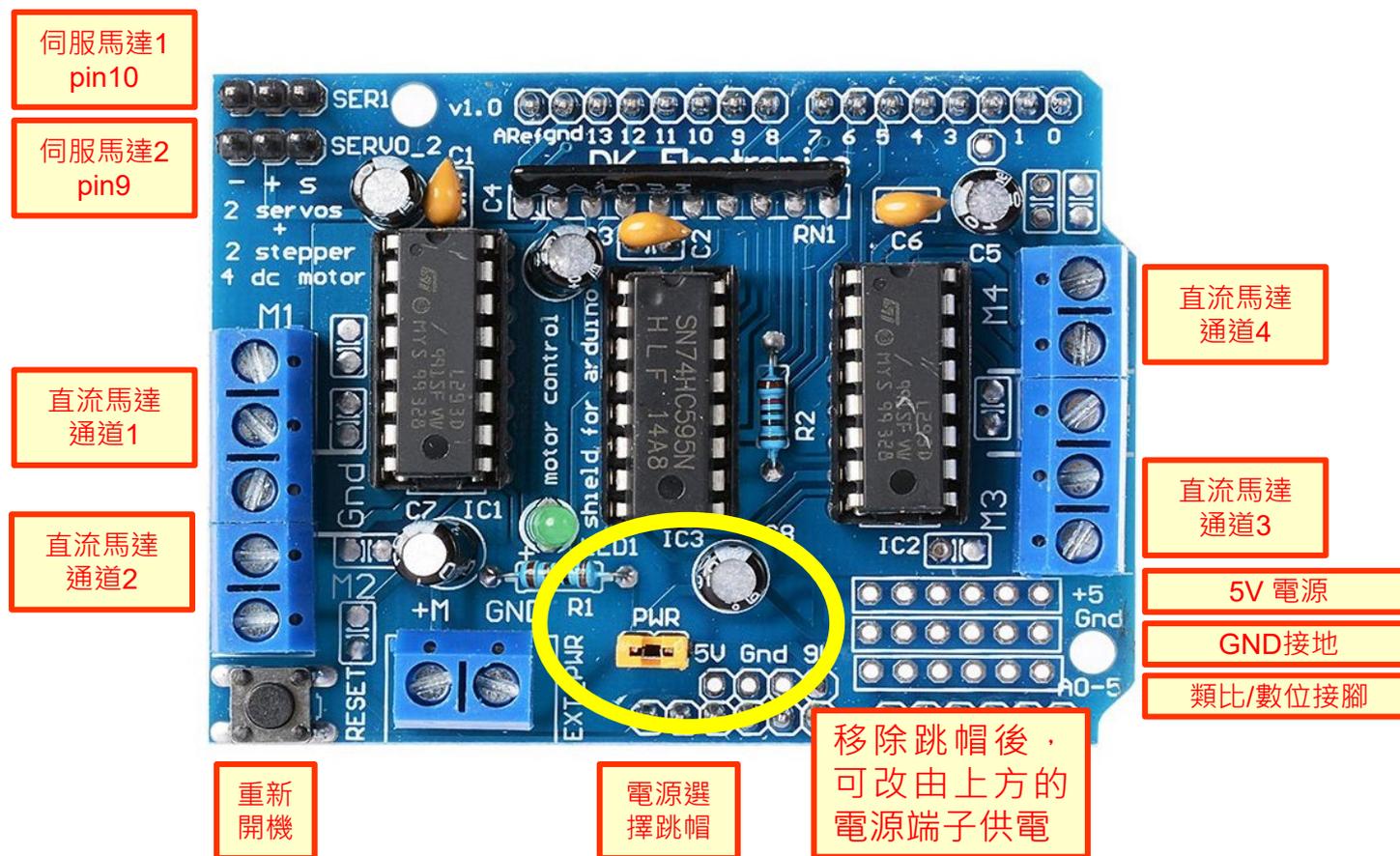
## 補充：L293D擴展板介紹

L293D電機控制擴展板支援四通道直流馬達、兩個伺服馬達，並保留A0-A5類比/數位接腳，



## 補充：電源供應來源選擇

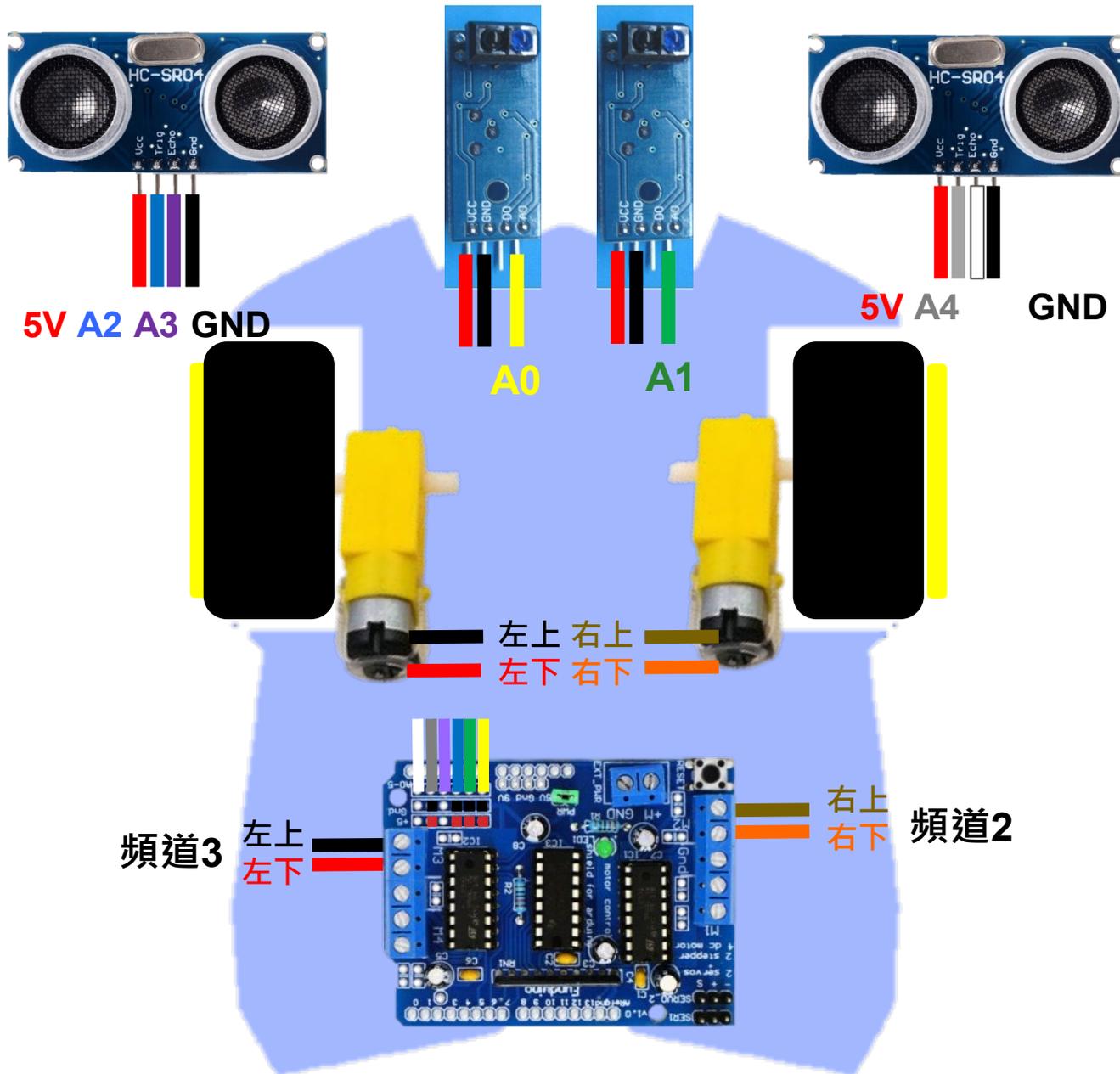
- 預設跳帽為接上，電源可由ARDUINO板上的USB-B插座或DC插座供應。
- 若移除跳帽，電源由L293D上的電源接線端子供應。



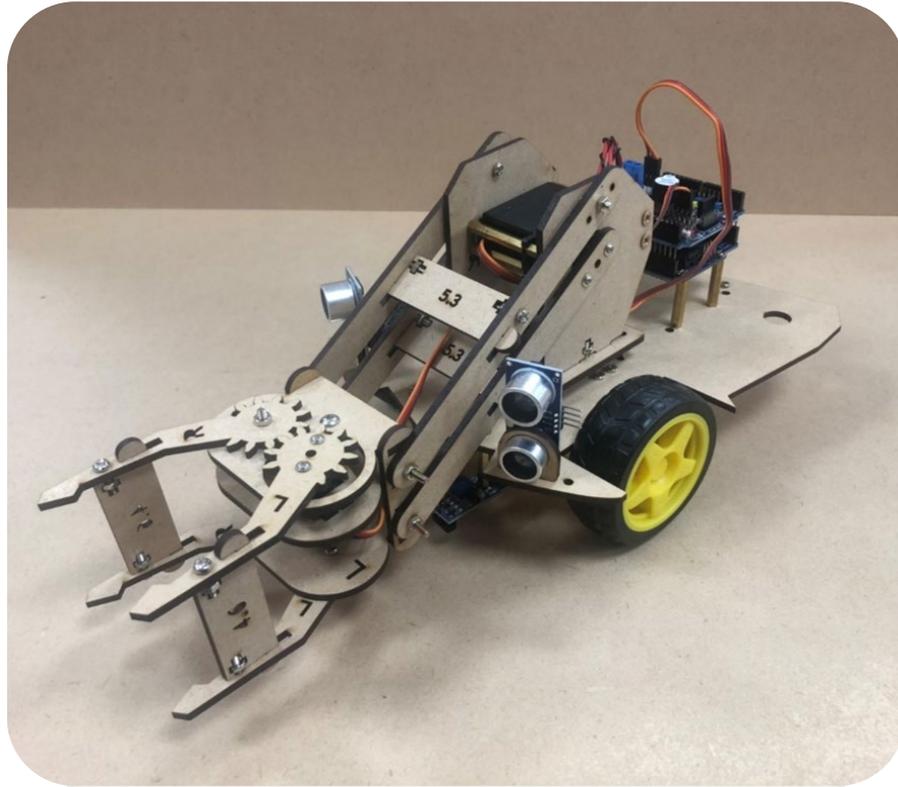
# 本教材使用腳位一覽(L293D擴展板)

	功能	備註
0	RX	Serial通訊
1	TX	Serial通訊
2		可使用
3	保留	請勿使用
4	保留	請勿使用
5	保留	請勿使用
6	保留	請勿使用
7	保留	請勿使用
8	保留	請勿使用
9	Servo 1	伺服馬達(抬昇)
10	Servo 2	伺服馬達(爪子)
11	保留	請勿使用
12	保留	請勿使用
13		板載LED

	功能	備註
A0	左AO	紅外線循跡
A1	右AO	紅外線循跡
A2	左TRIG	超音波
A3	左EHCO	超音波
A4	右TRIG	超音波
A5	右EHCO	超音波



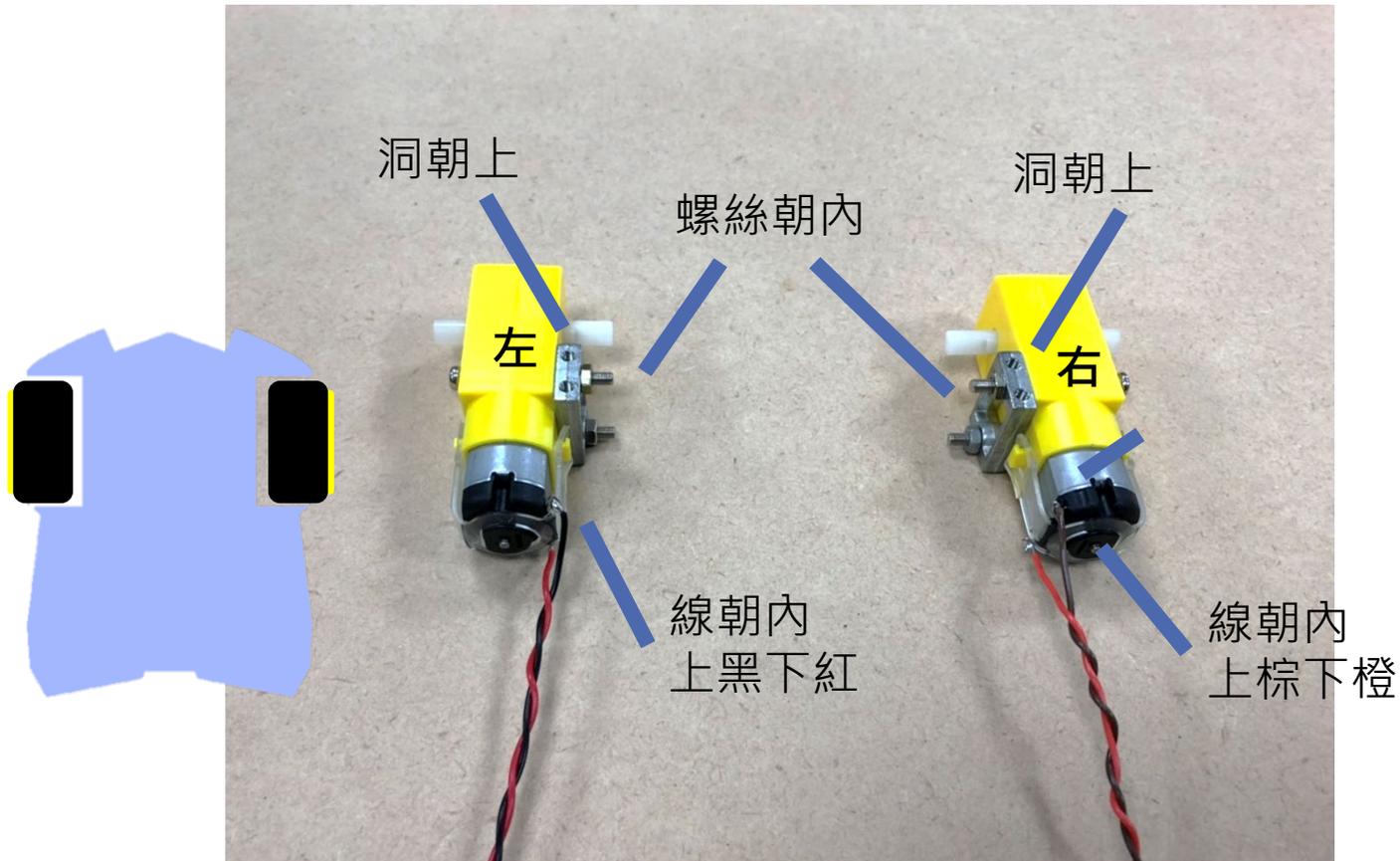
# START! 智慧小車



-單元2 智慧小車組裝-

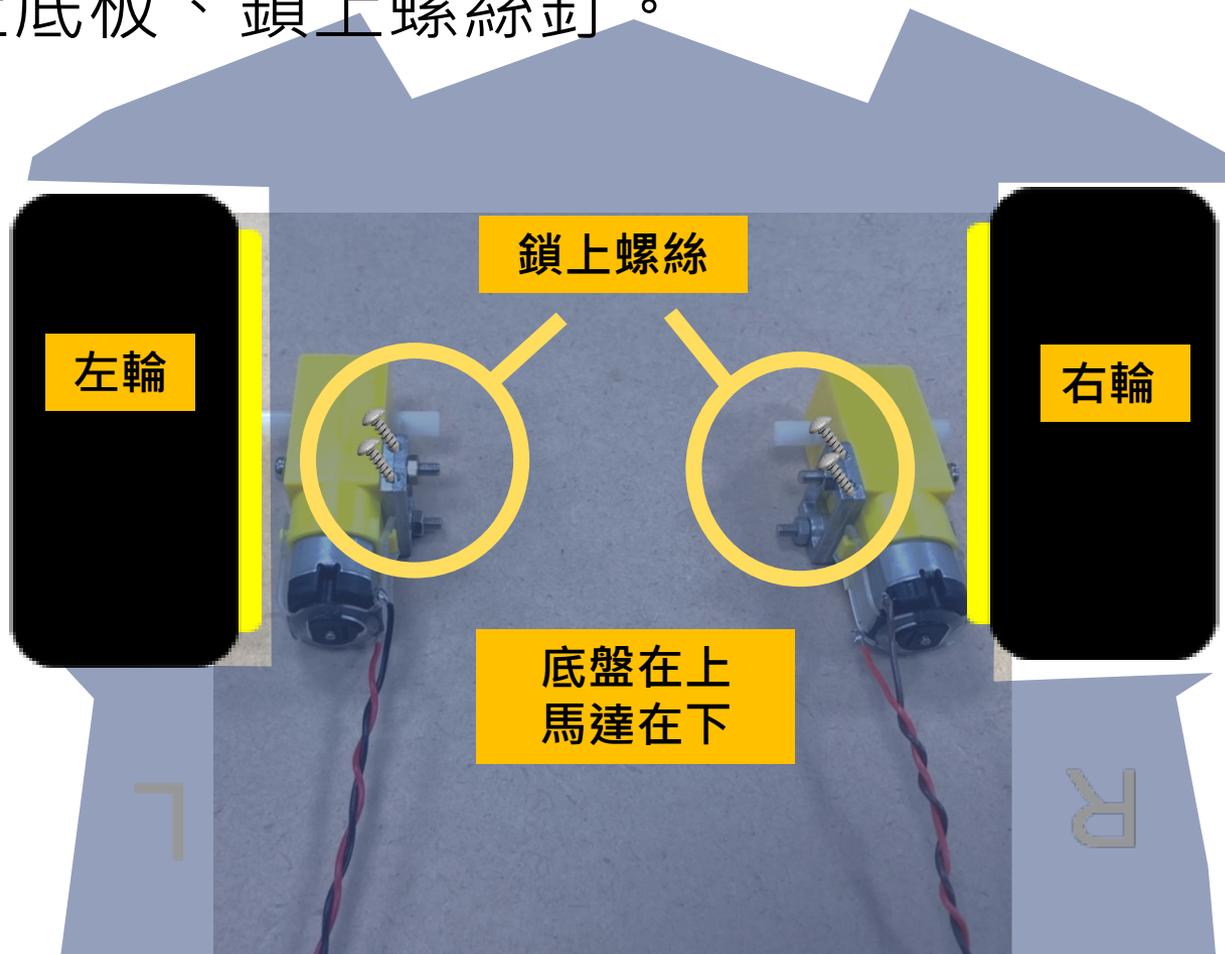
# STEP.1

鎖上TT馬達支架，注意螺絲孔朝上、螺絲要在內側、電線也要在內側。



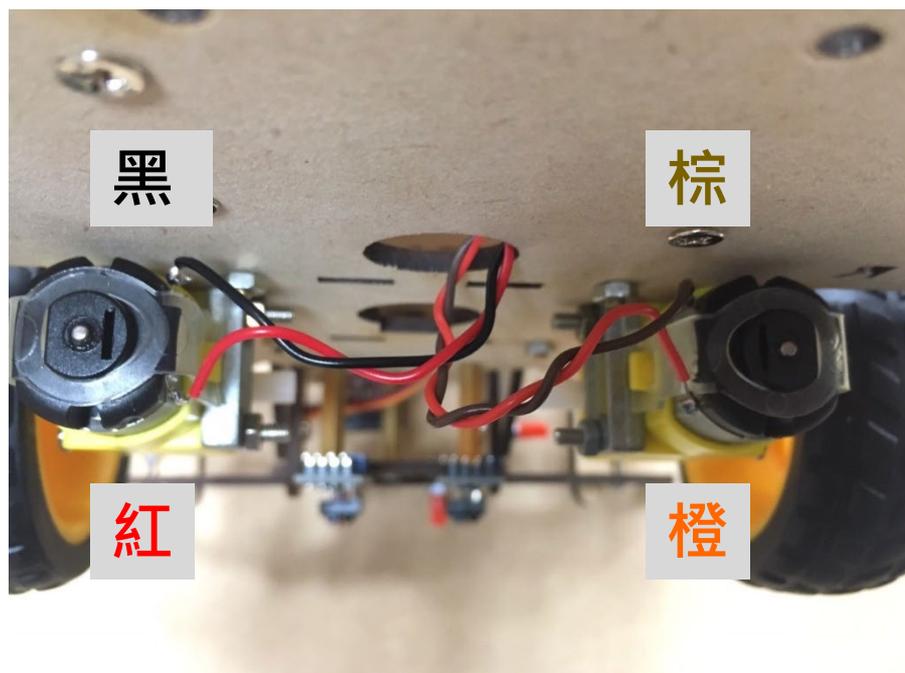
## STEP.2

蓋上底板、鎖上螺絲釘。



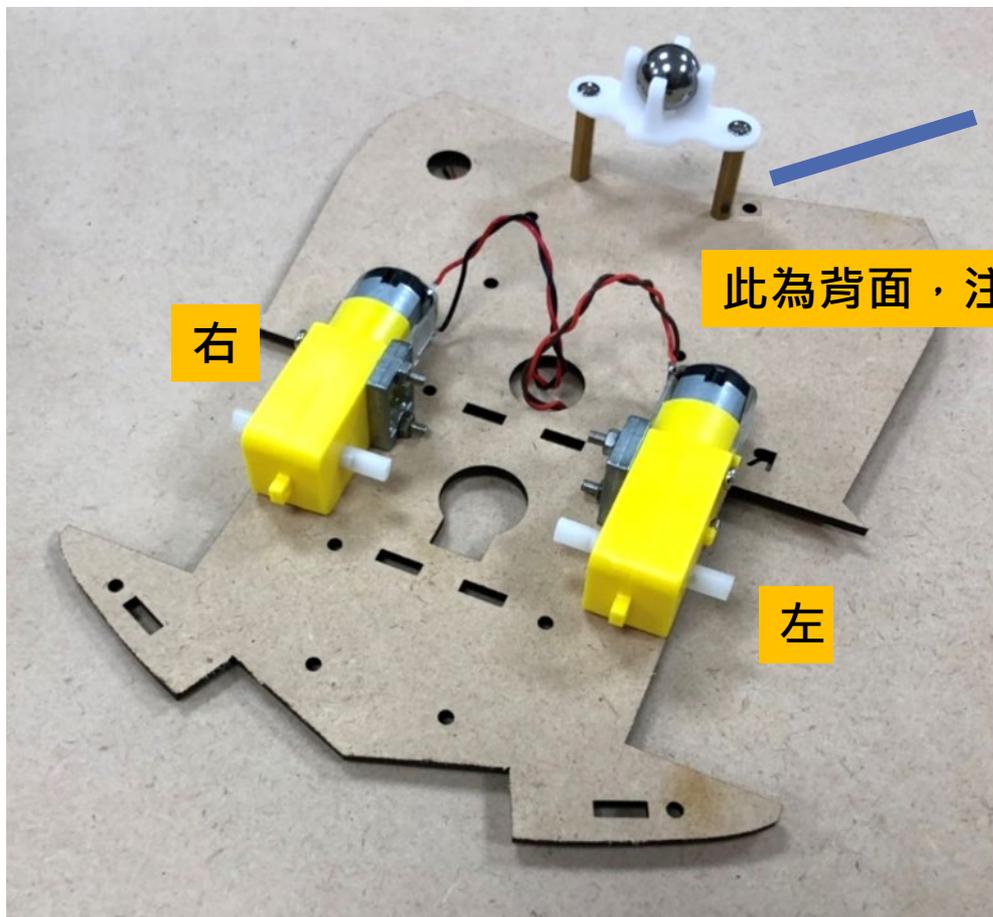
# 直流馬達連接

從車尾觀察小車的馬達顏色，以小寫n字的順序會是「黑紅棕橙」；  
在L293D擴展板上，若將重新開機鍵置於右上角，電線顏色以小寫n字的順序也是「黑紅棕橙」。



## STEP.3

將底盤翻過來，將鋼珠輪鎖至底盤



25mm 雙通銅柱  
10mm 平頭螺絲

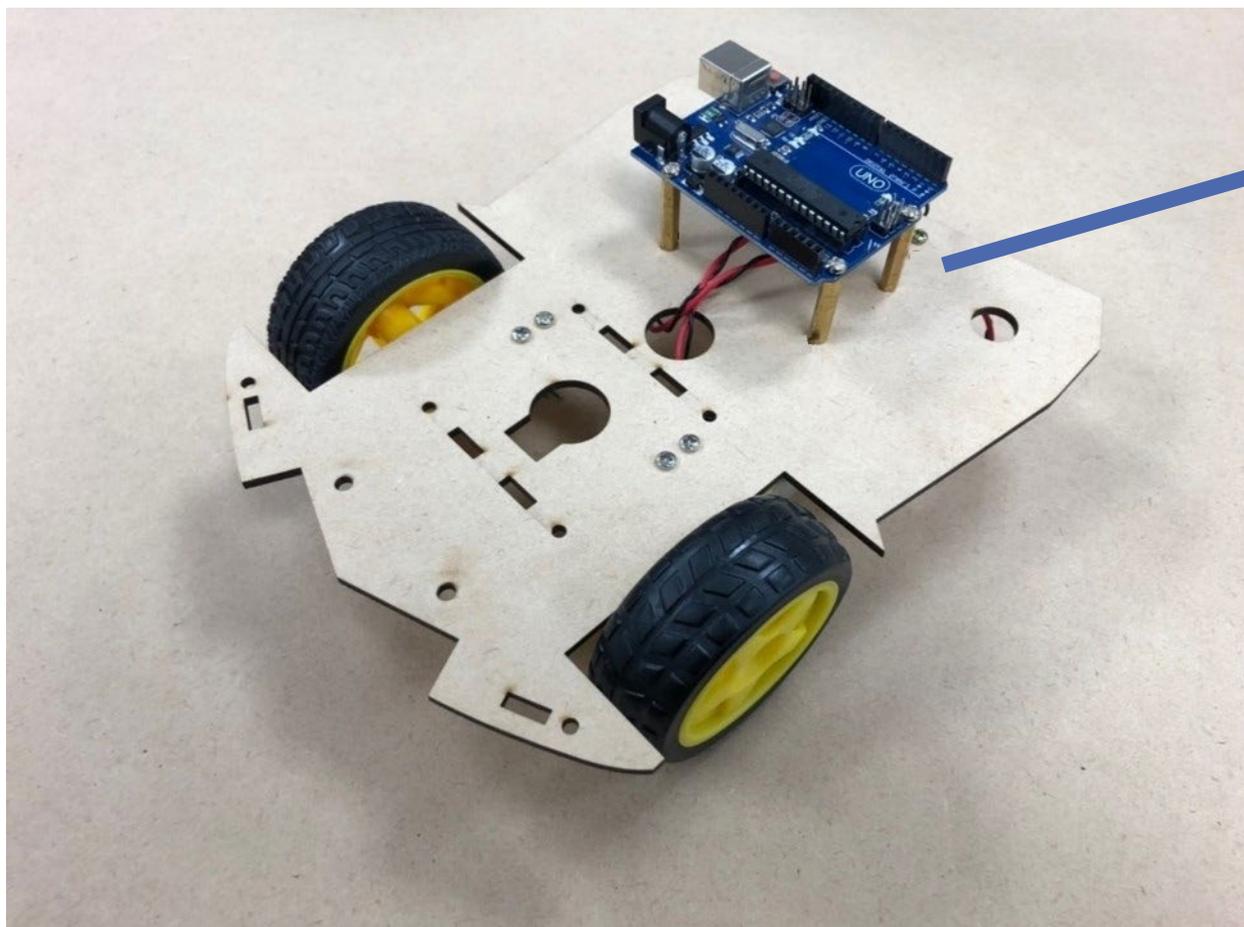
此為背面，注意電線顏色

右

左

## STEP.4

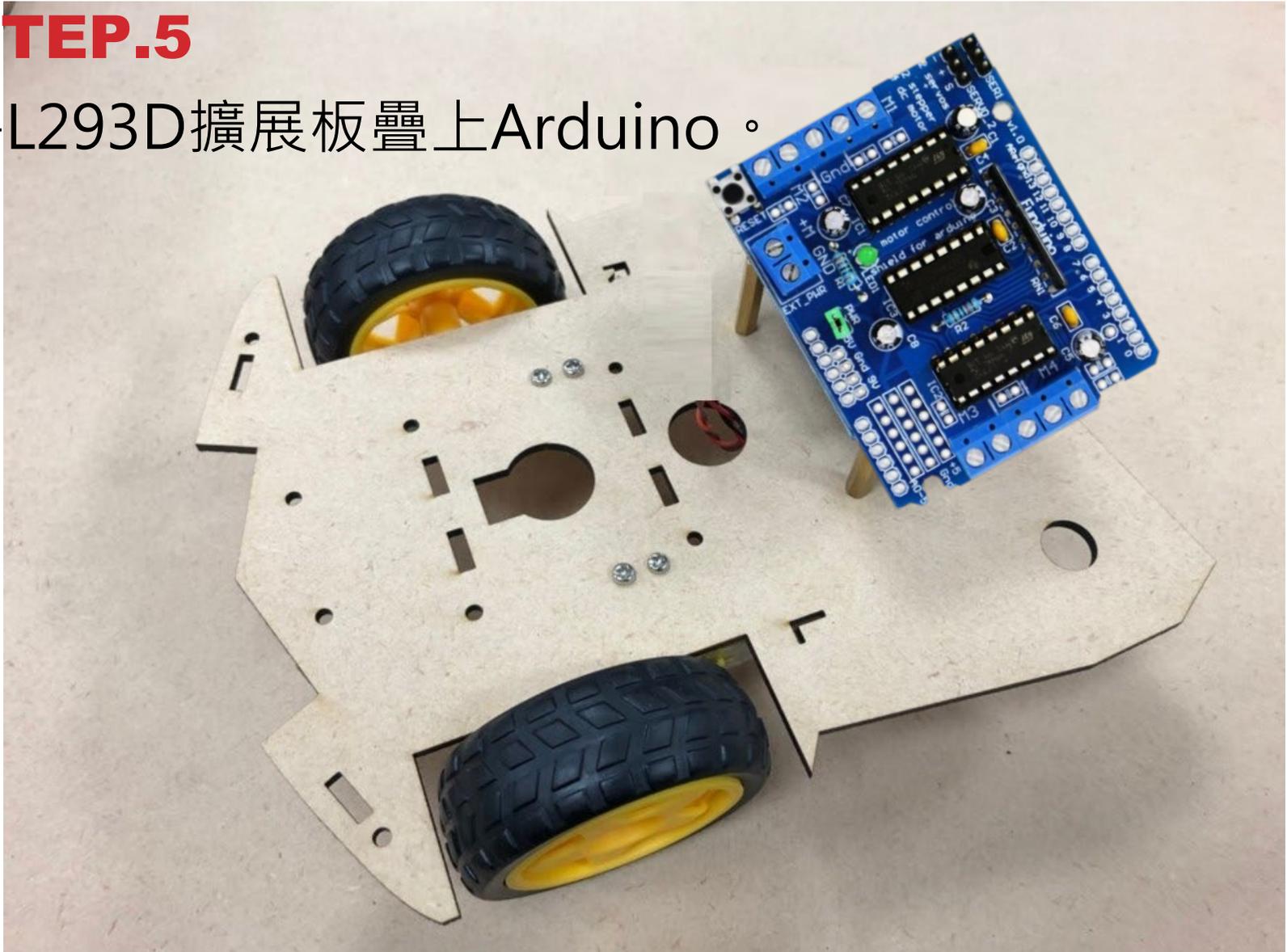
翻回正面，鎖上Arduino控制板



25mm雙通銅柱  
10mm平頭螺絲

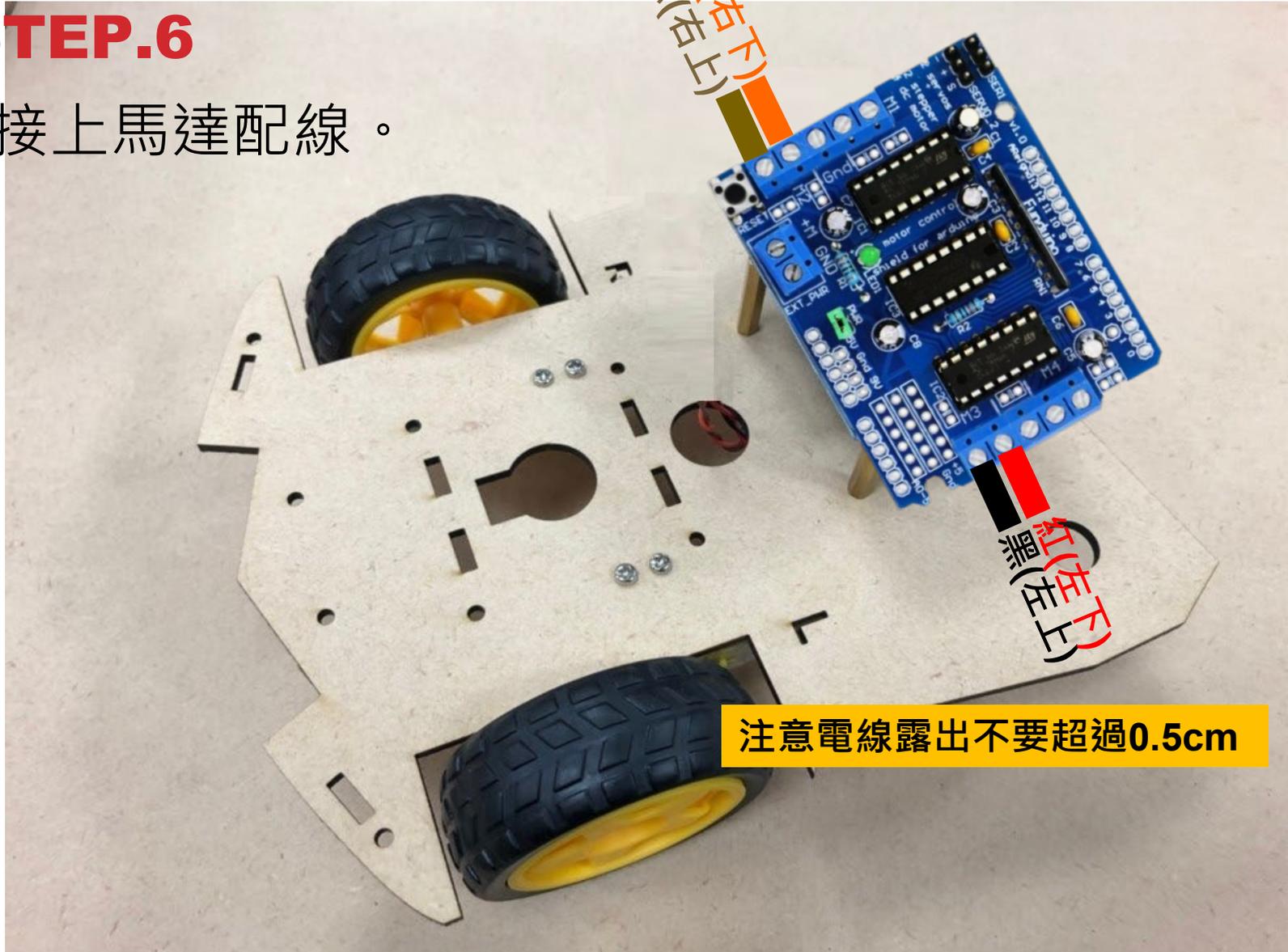
## STEP.5

將L293D擴展板疊上Arduino。



## STEP.6

接上馬達配線。



## STEP.7

鎖上循線感測器、超音波感測器。

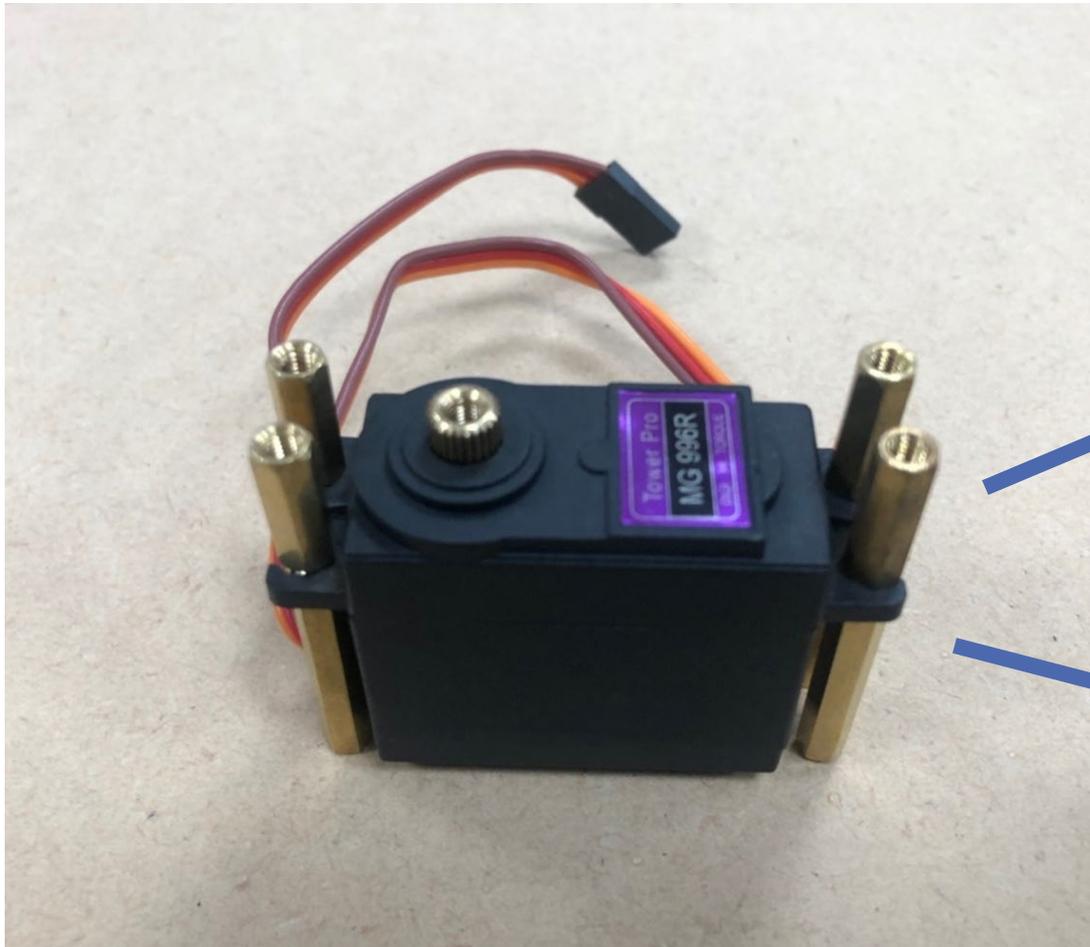


※可依據循跡狀況調整銅柱長度(12-25mm)  
銅柱短，Sensor離地遠，反射強度弱  
→對黑色較敏感，適合轉彎(缺點為容易擺頭)  
銅柱長，Sensor離地近，反射强度高  
→對白色較敏感，適合直線(缺點為容易出軌)

12mm銅柱(一公一母)  
10mm平頭螺絲、螺帽

## STEP.8

為MG996馬達鎖上銅柱。

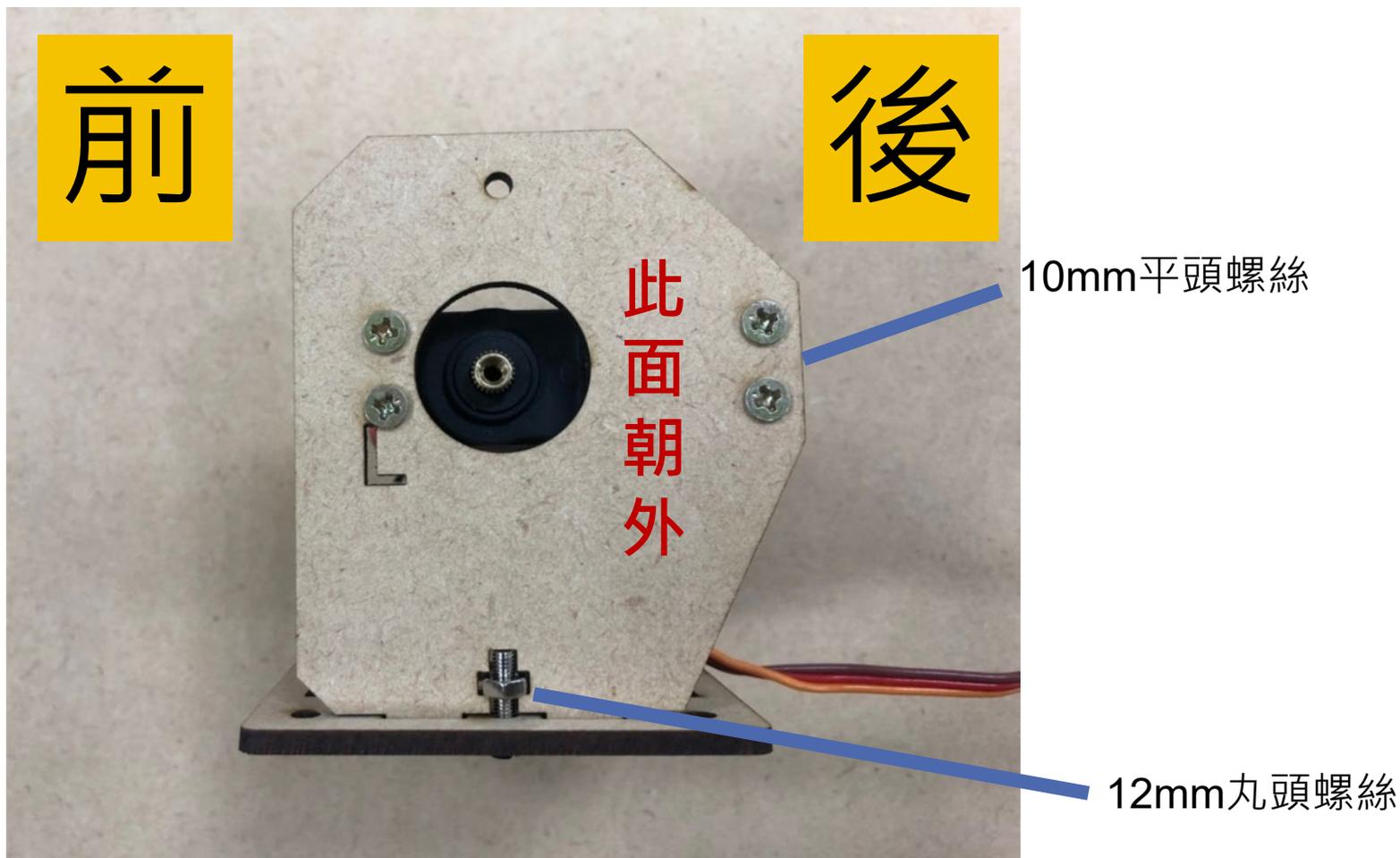


12mm銅柱  
(一公一母)

30mm雙通銅柱

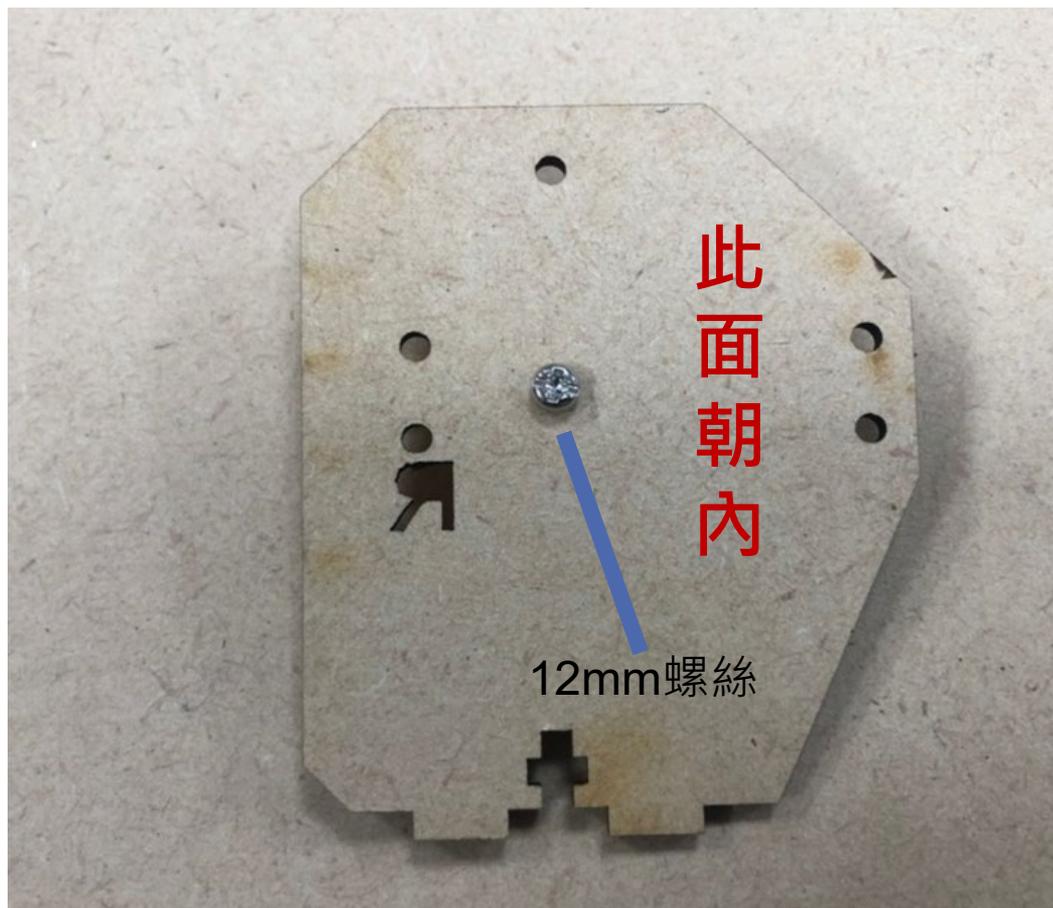
## STEP.9

組裝左側蓋板及底板



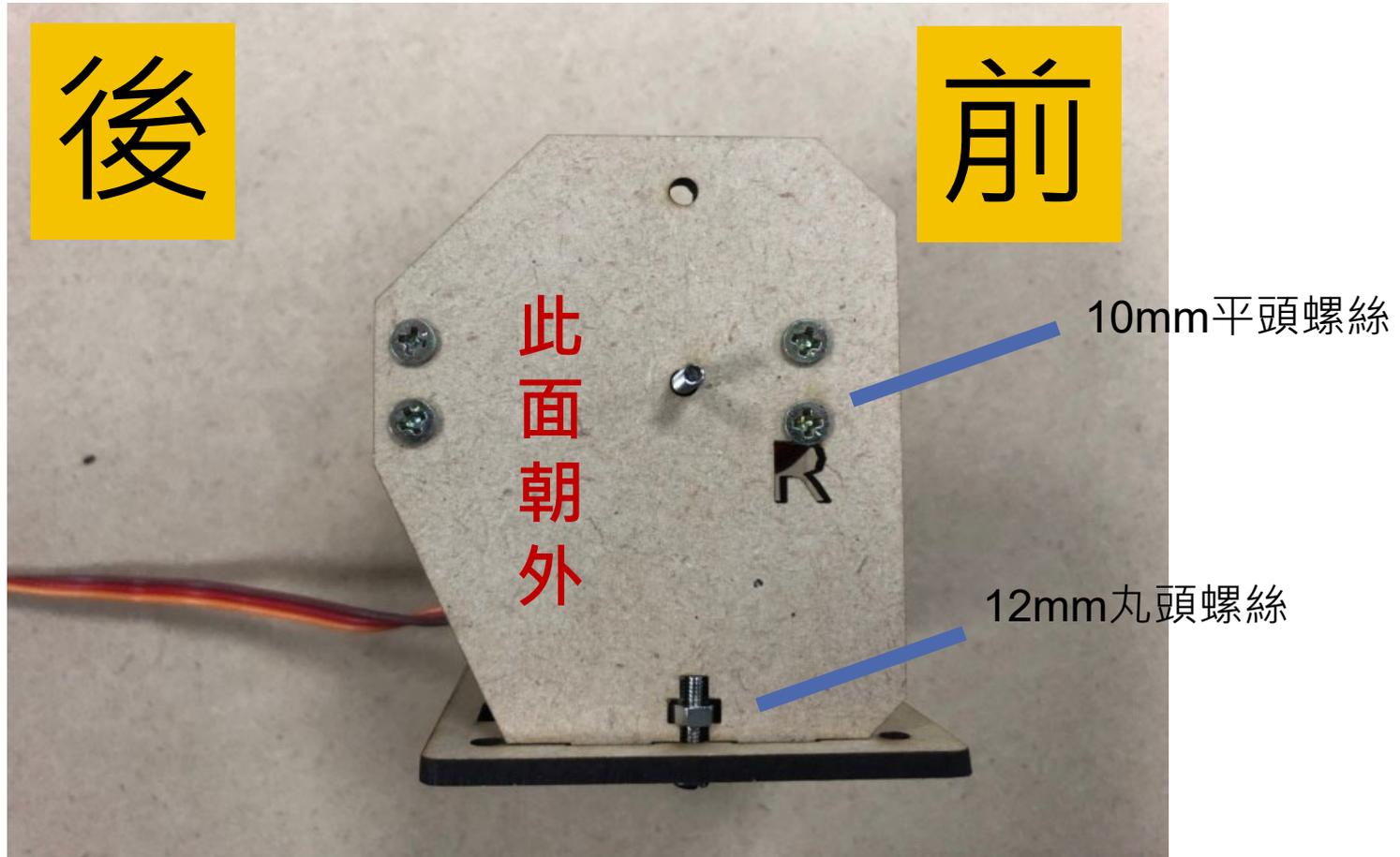
## STEP.10

拿出右側蓋板，翻到內側，預置一個螺絲。



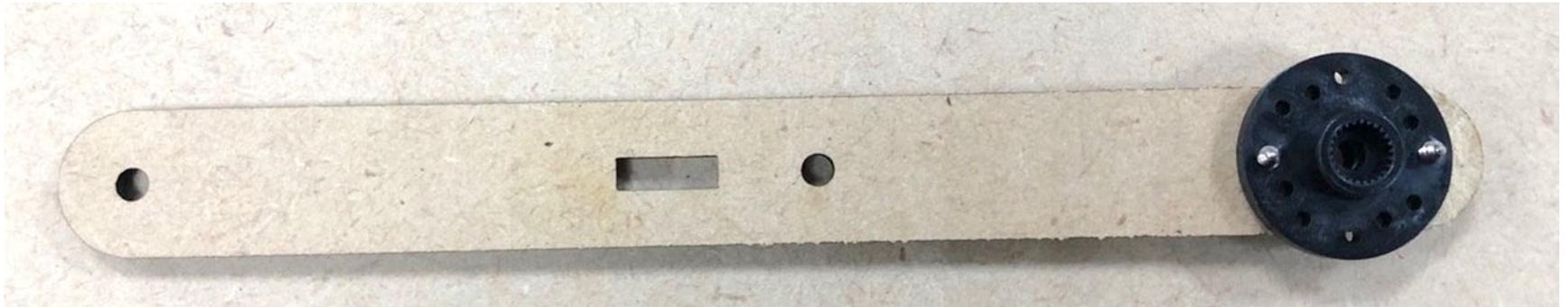
## STEP.11

組裝右側蓋板及底板



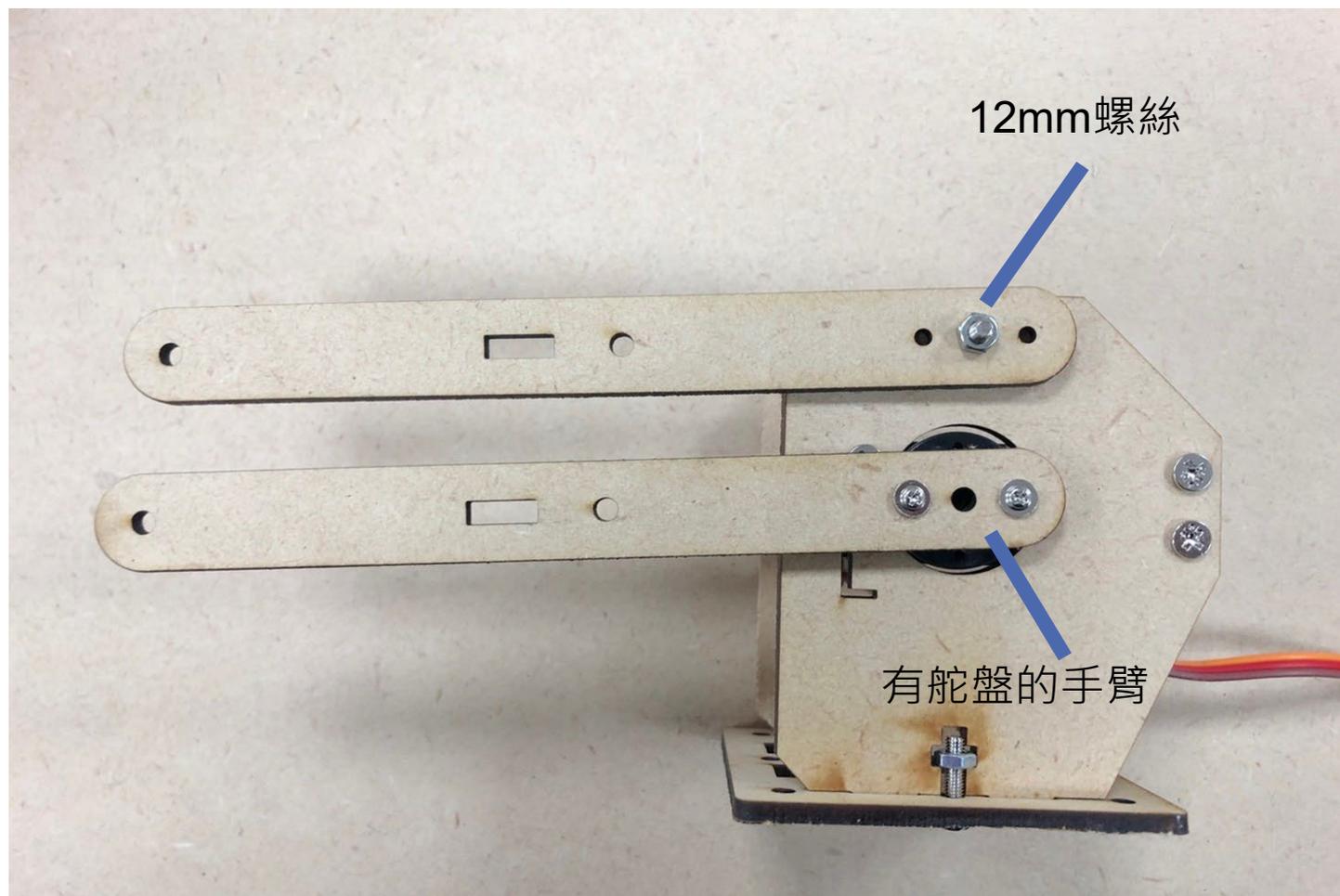
## STEP.12

將舵盤鎖至手臂上



## STEP.13

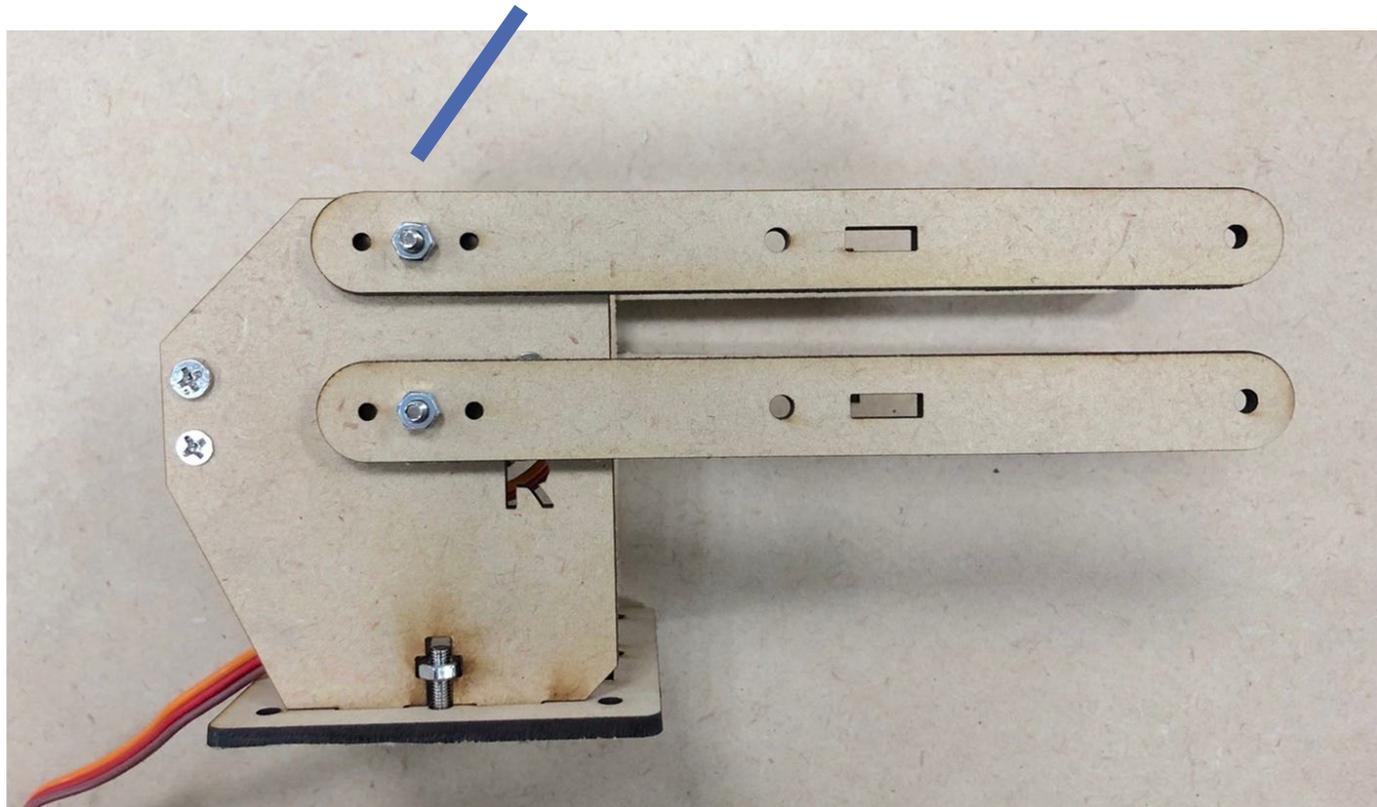
### 組裝左側手臂



# STEP.14

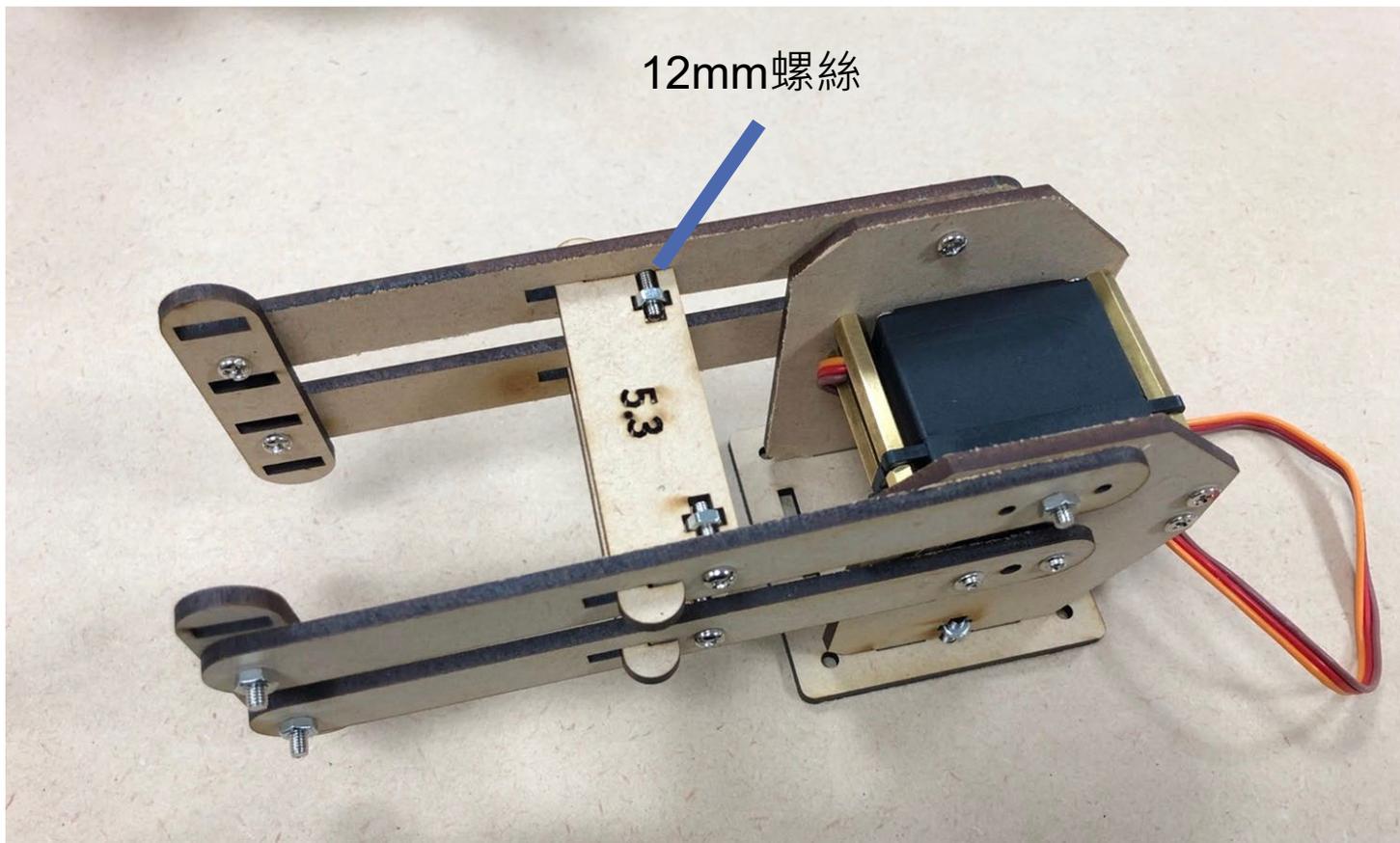
## 組裝右側手臂

12mm螺絲



## STEP.15

### 組裝手臂橫桿



# STEP.16

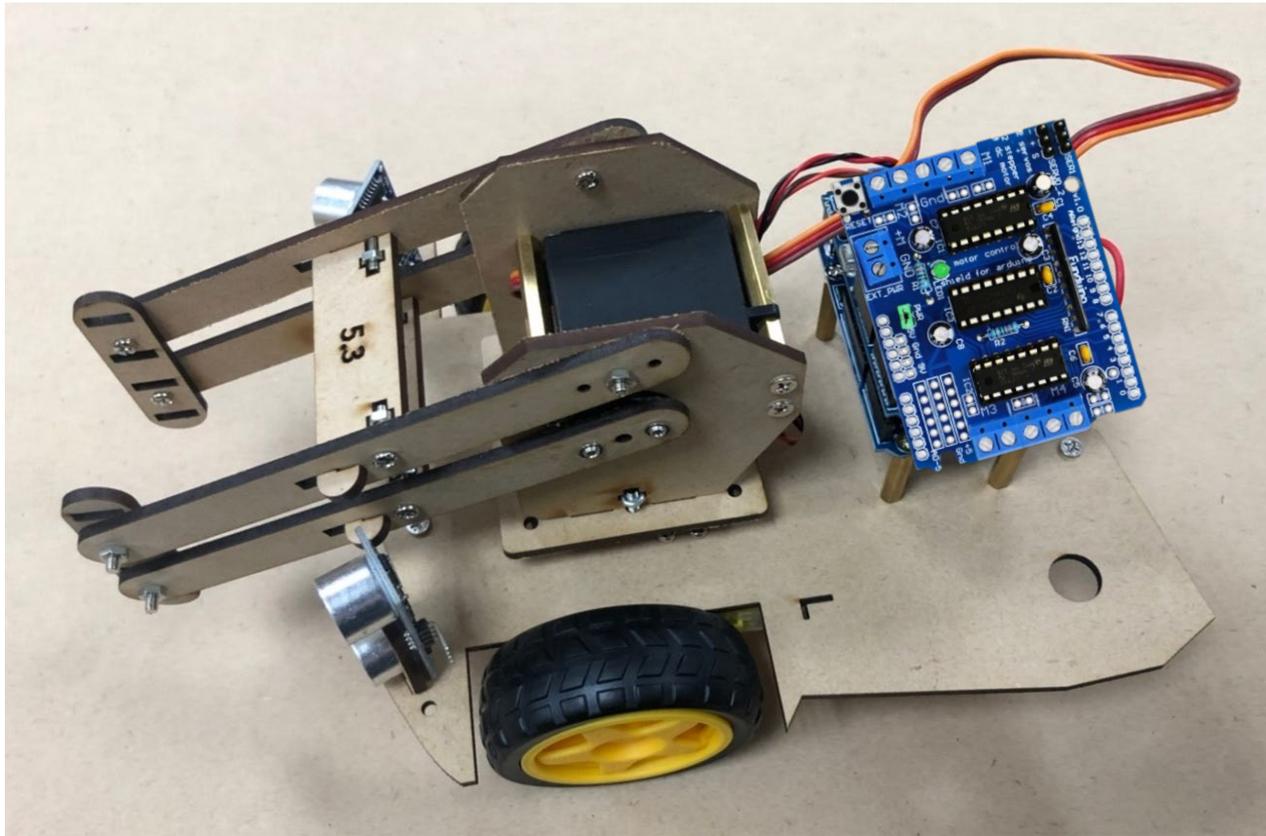
放上快拆支架。



亦可使用10mm銅柱。

## STEP.17

利用快拆將手臂扣至車身底盤。



## STEP.18

將MG90馬達套到方孔內。



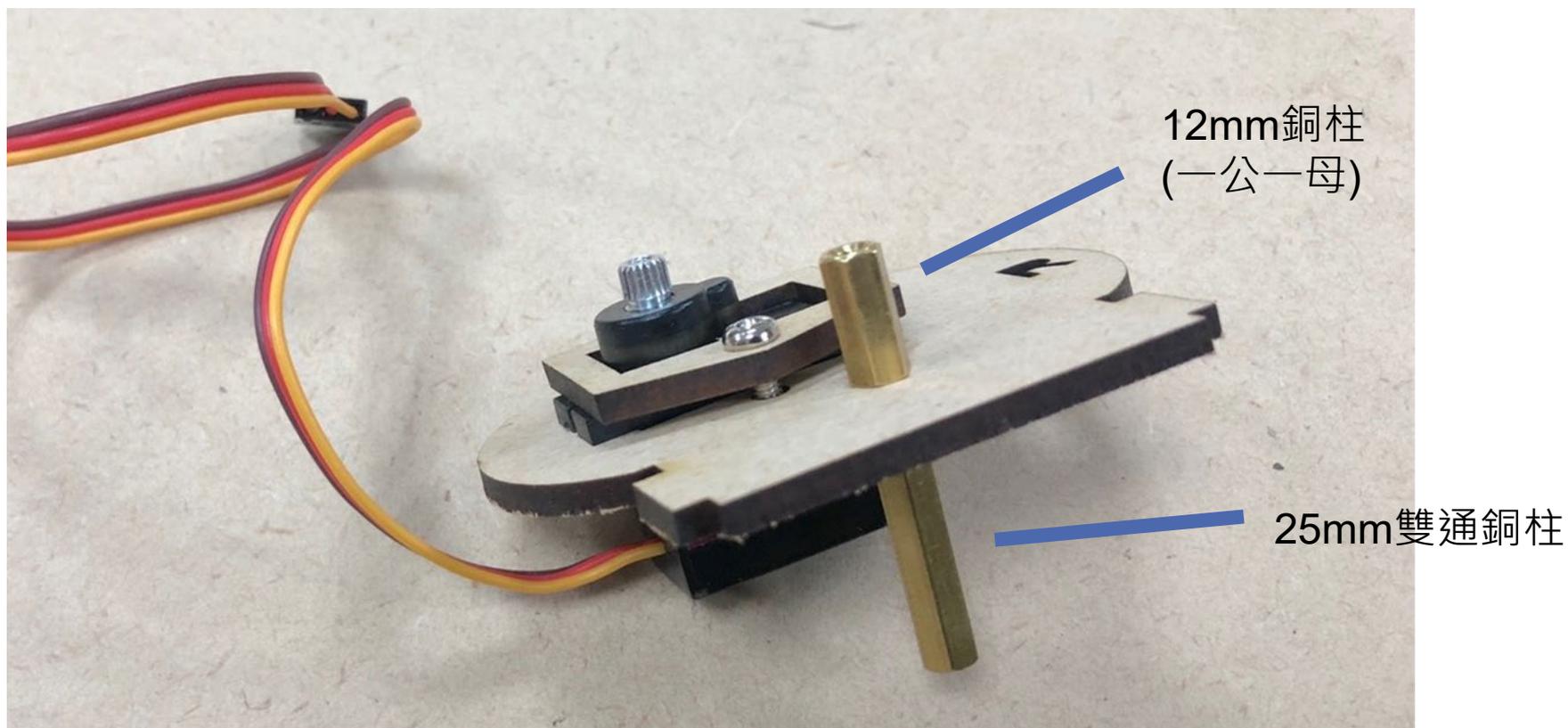
## STEP.19

固定MG90馬達。



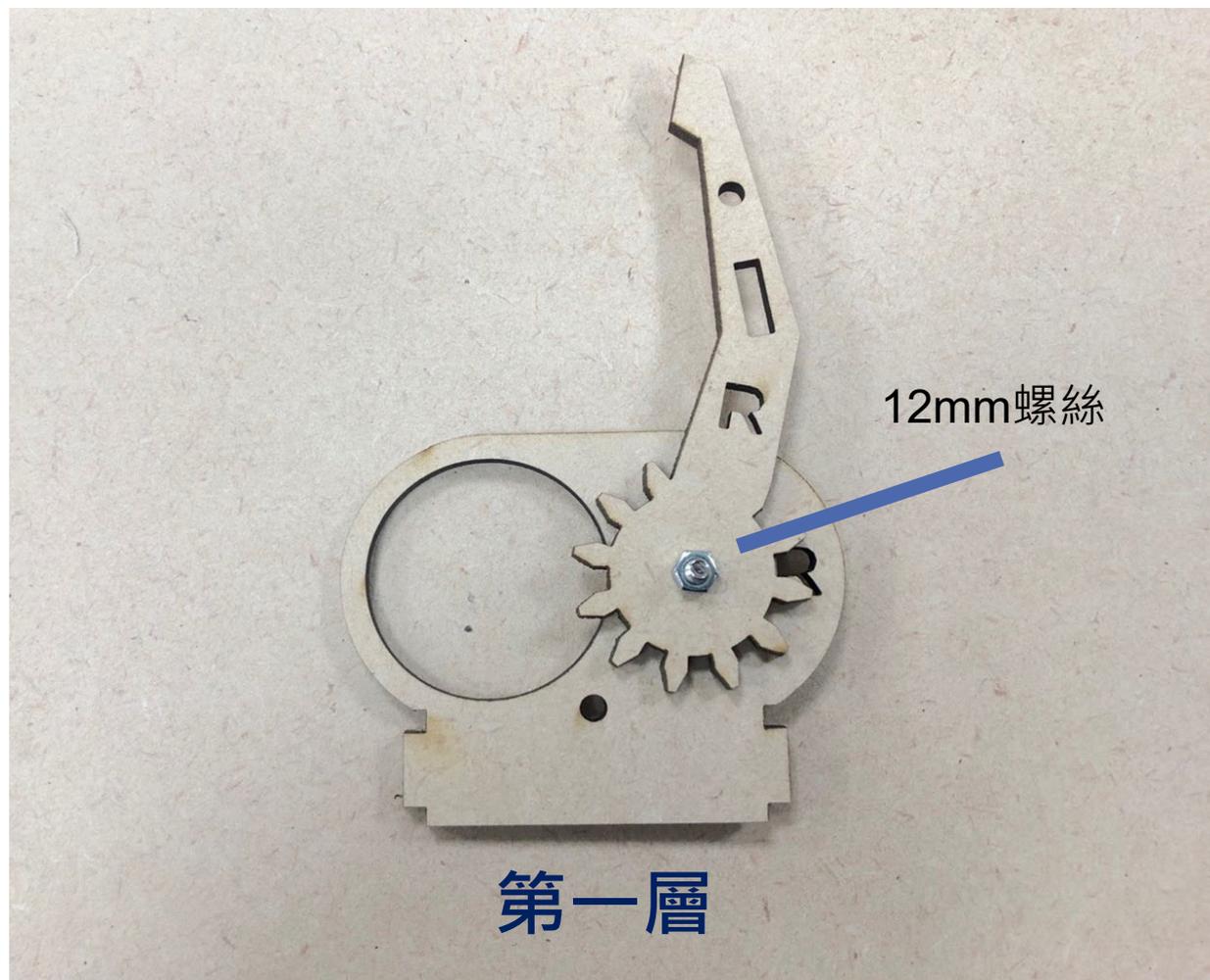
## STEP.20

鎖上隔離柱至支架。



## STEP.20

鎖上第一層右邊爪子。



## STEP.21

組裝第一層與第二層。



## STEP.22

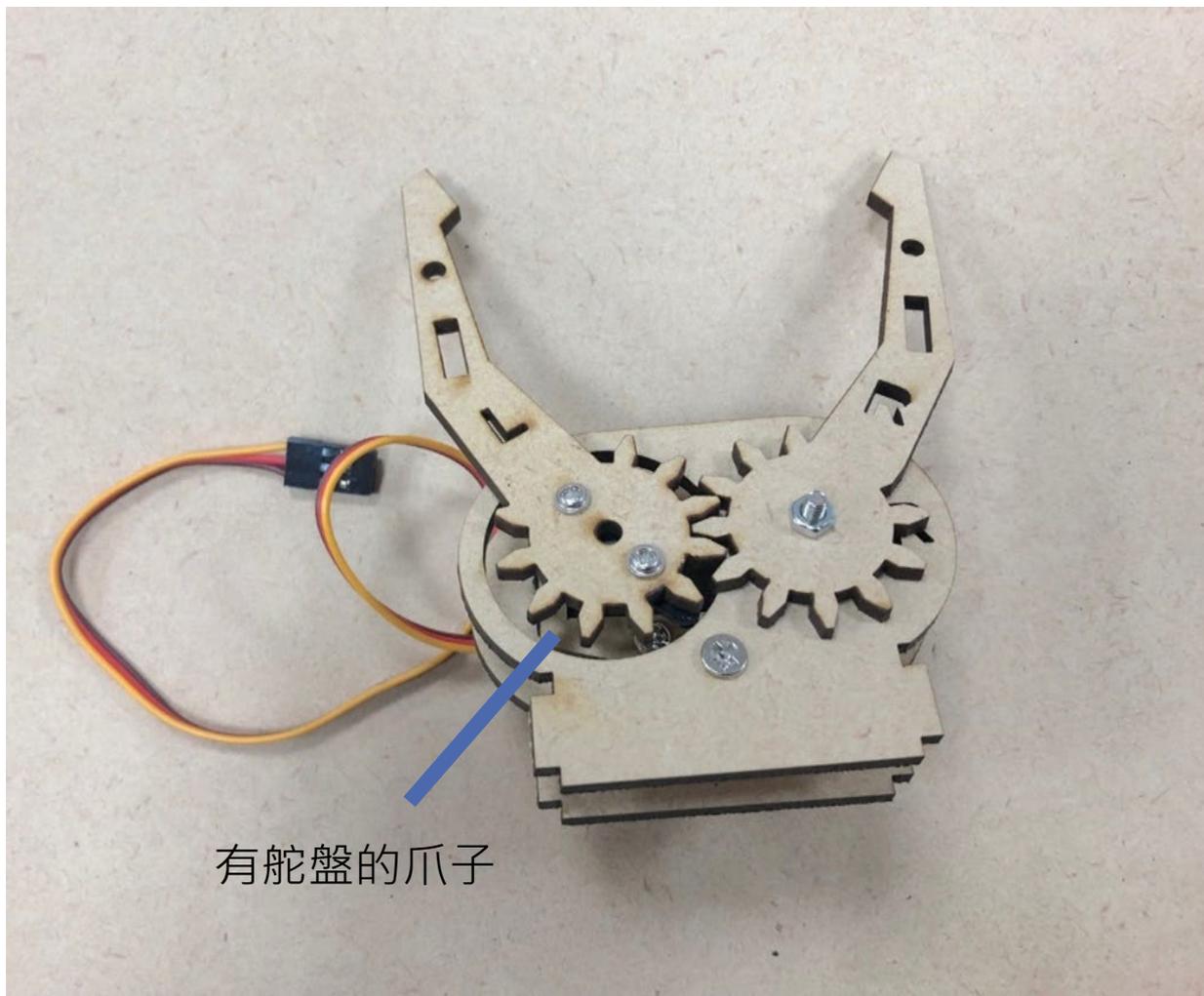
將舵盤鎖至爪子上。



此面朝內

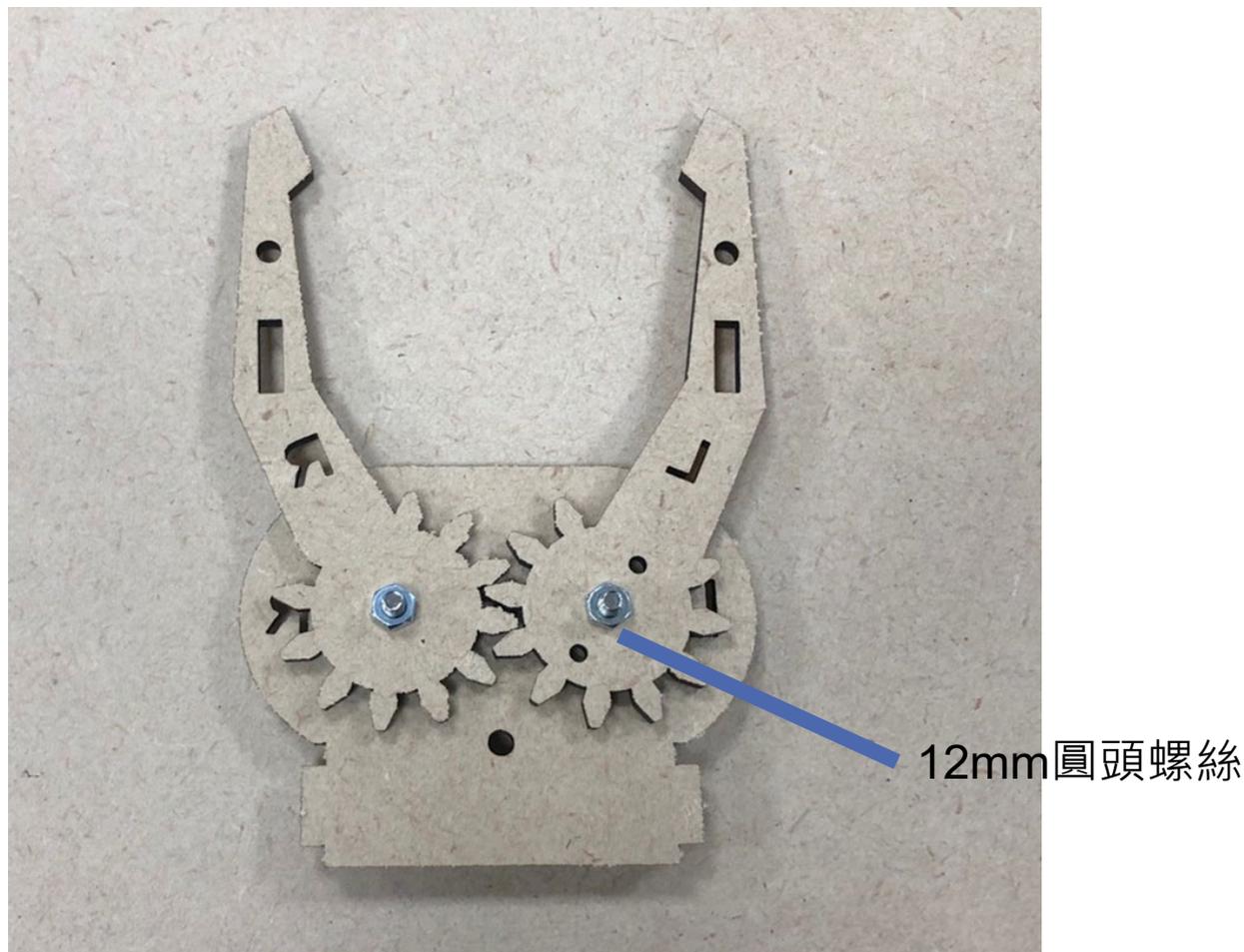
## STEP.23

鎖上第一層左邊爪子。



## STEP.24

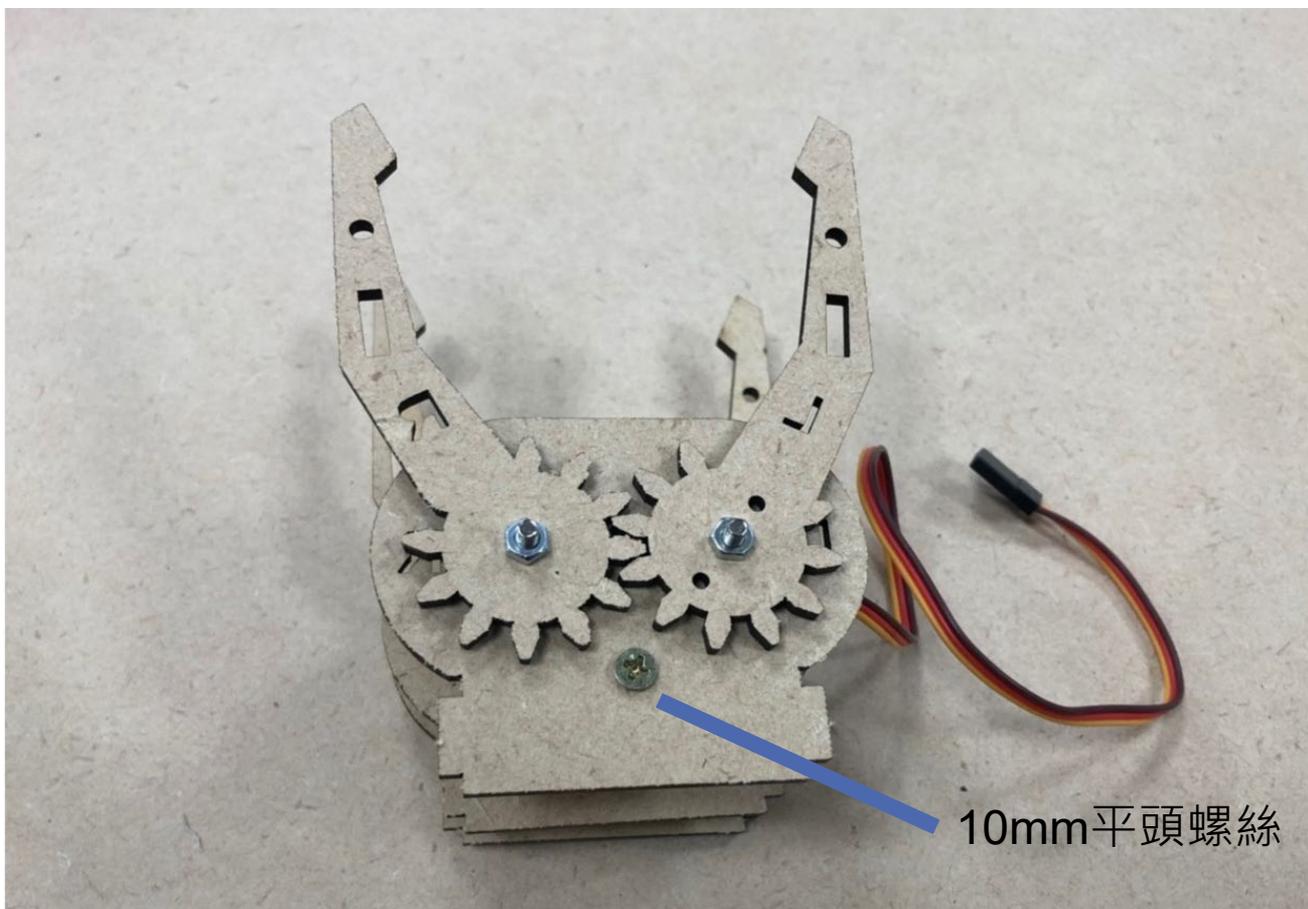
組裝第三層爪子。



第三層

## STEP.25

組裝第三層，注意齒輪在外側。



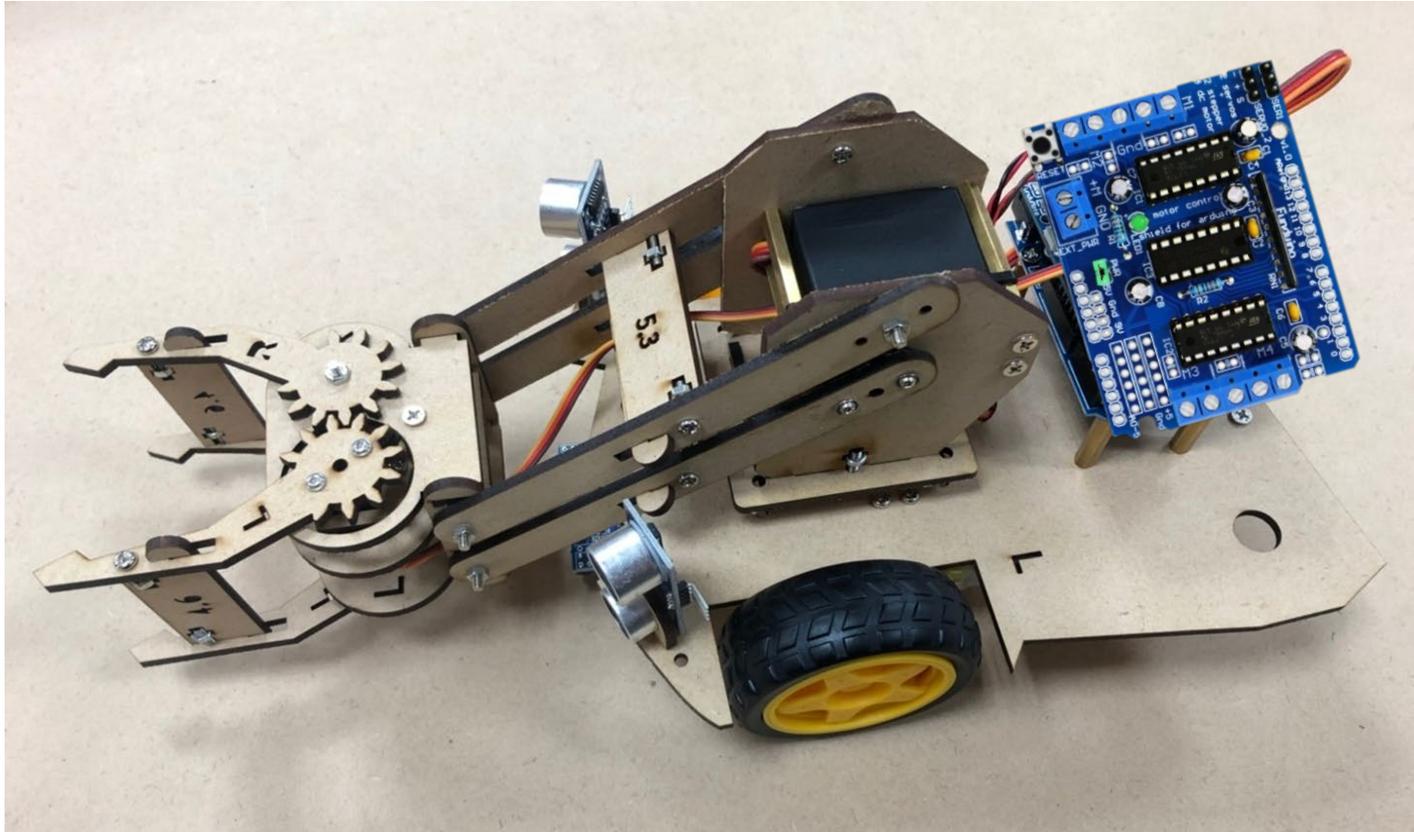
## STEP.26

組裝橫桿。



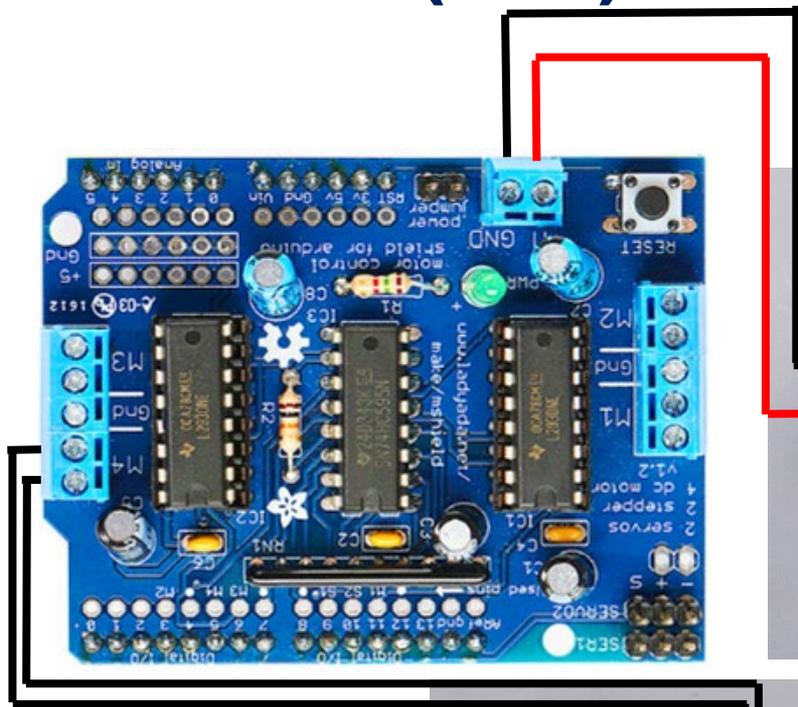
## STEP.27

將爪子固定至手臂，組裝完成。



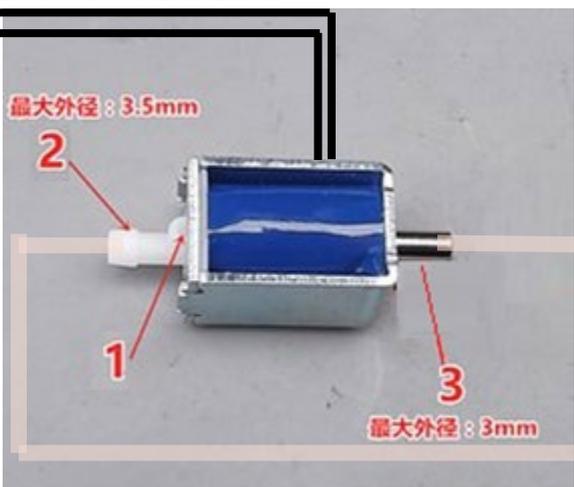
# 補充：吸盤模組(選配)

① 將抽氣馬達用熱融膠黏在車底盤  
電源接到L293D板上(無分正負極)



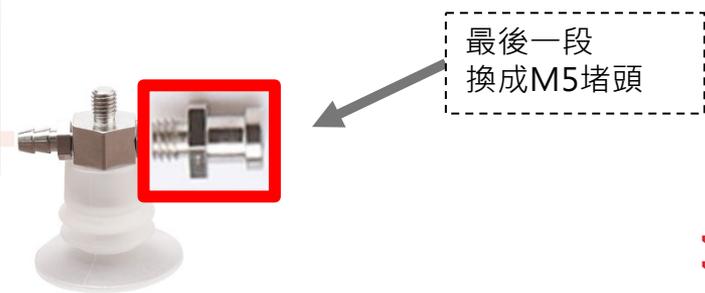
矽膠管長約8-10cm

② 將電磁閥電源接到  
L293D板(無分正負極)



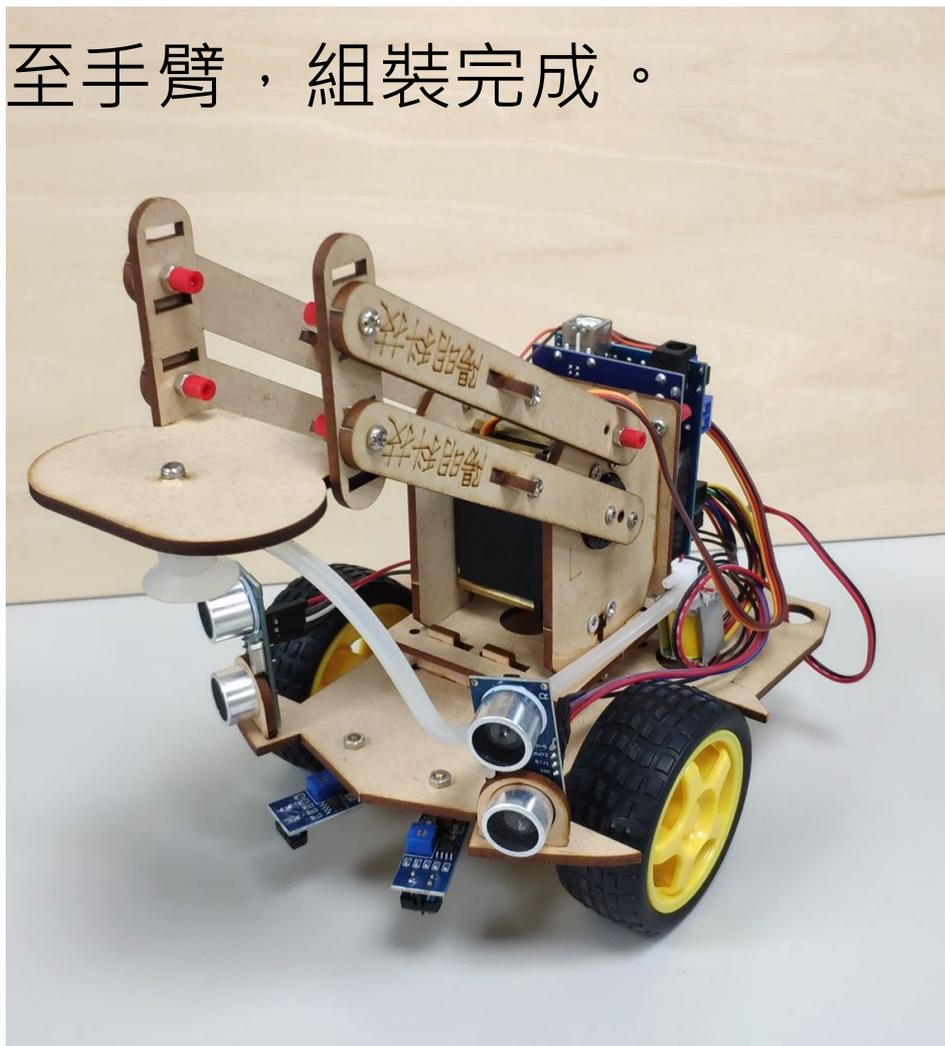
③ 剪取適當長度矽膠管  
連接幫浦、電磁閥及  
吸盤

矽膠管長約20-22cm

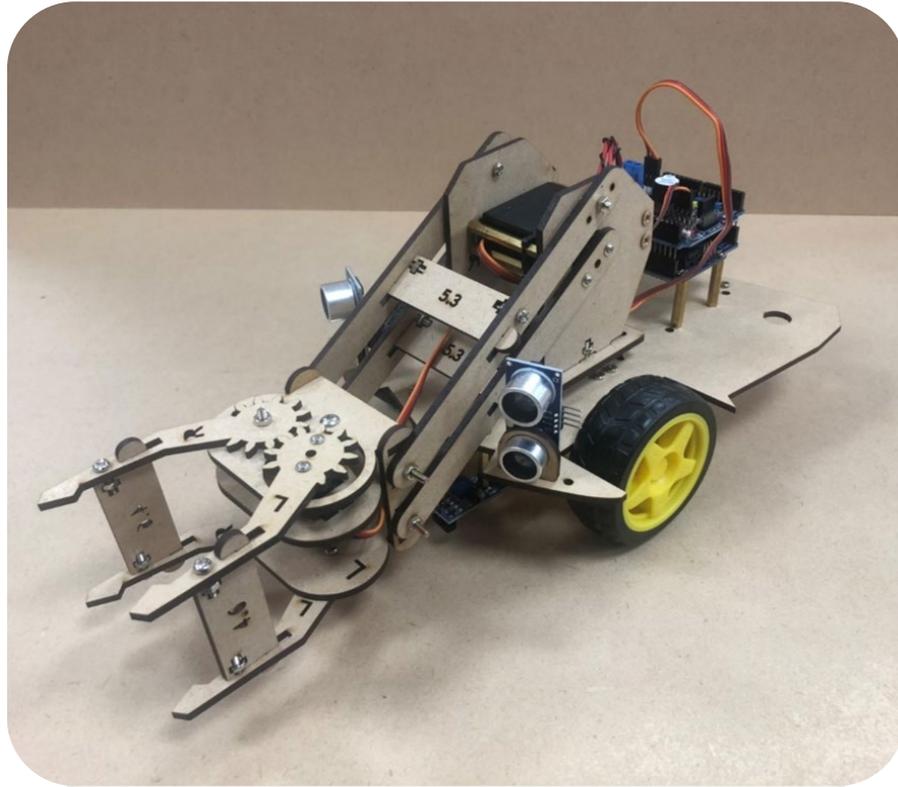


## 補充：吸盤模組(選配)

將吸盤鎖至手臂，組裝完成。



# START! 智慧小車



-單元3 程式環境設定-

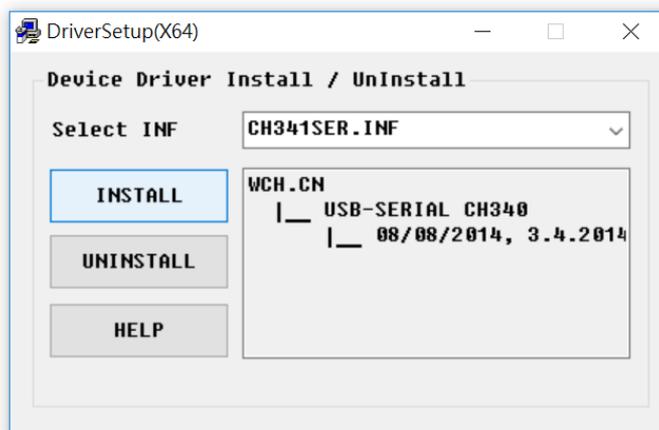
# 程式環境設定

## 下載並安裝CH341SER.EXE

我的雲端硬碟 > 智慧小車 > 01.Ardublockly 程式 ▾

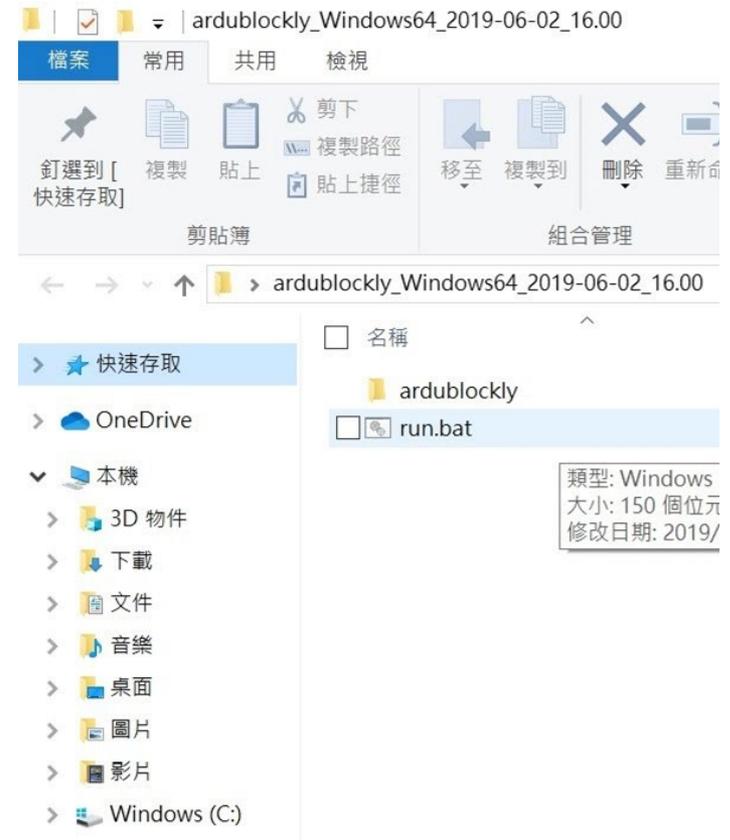
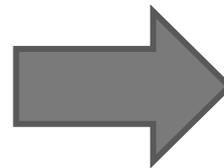
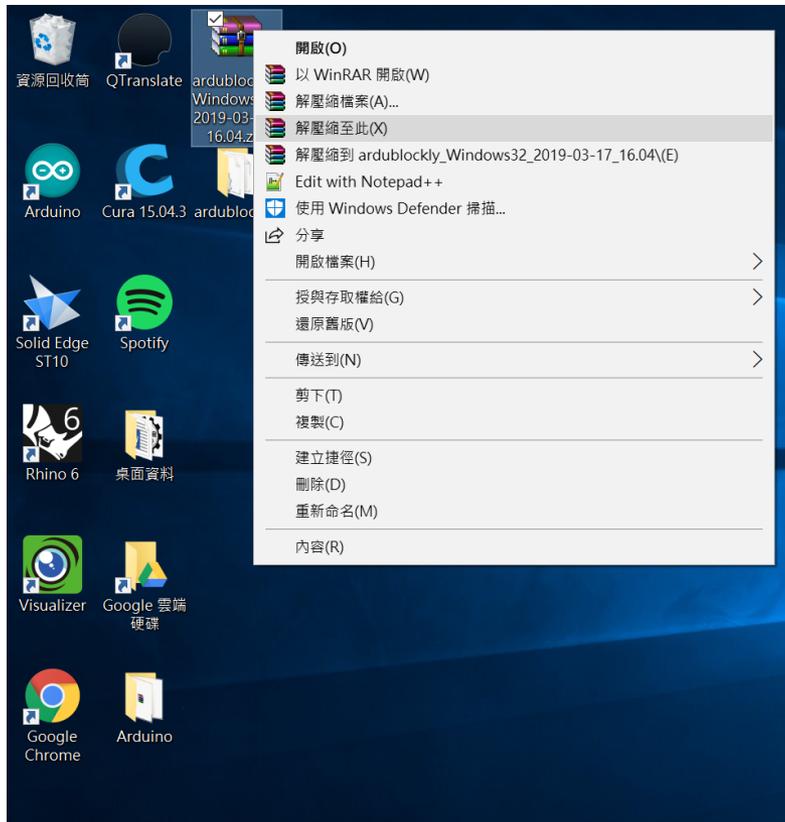


名稱 ↑	擁有者	上次修改時間	檔案大小
ardublockly_Windows32_2019-03-17_16.04.zip	我	下午7:48	256 MB
<b>CH341SER.EXE</b>	我	下午11:02	238 KB



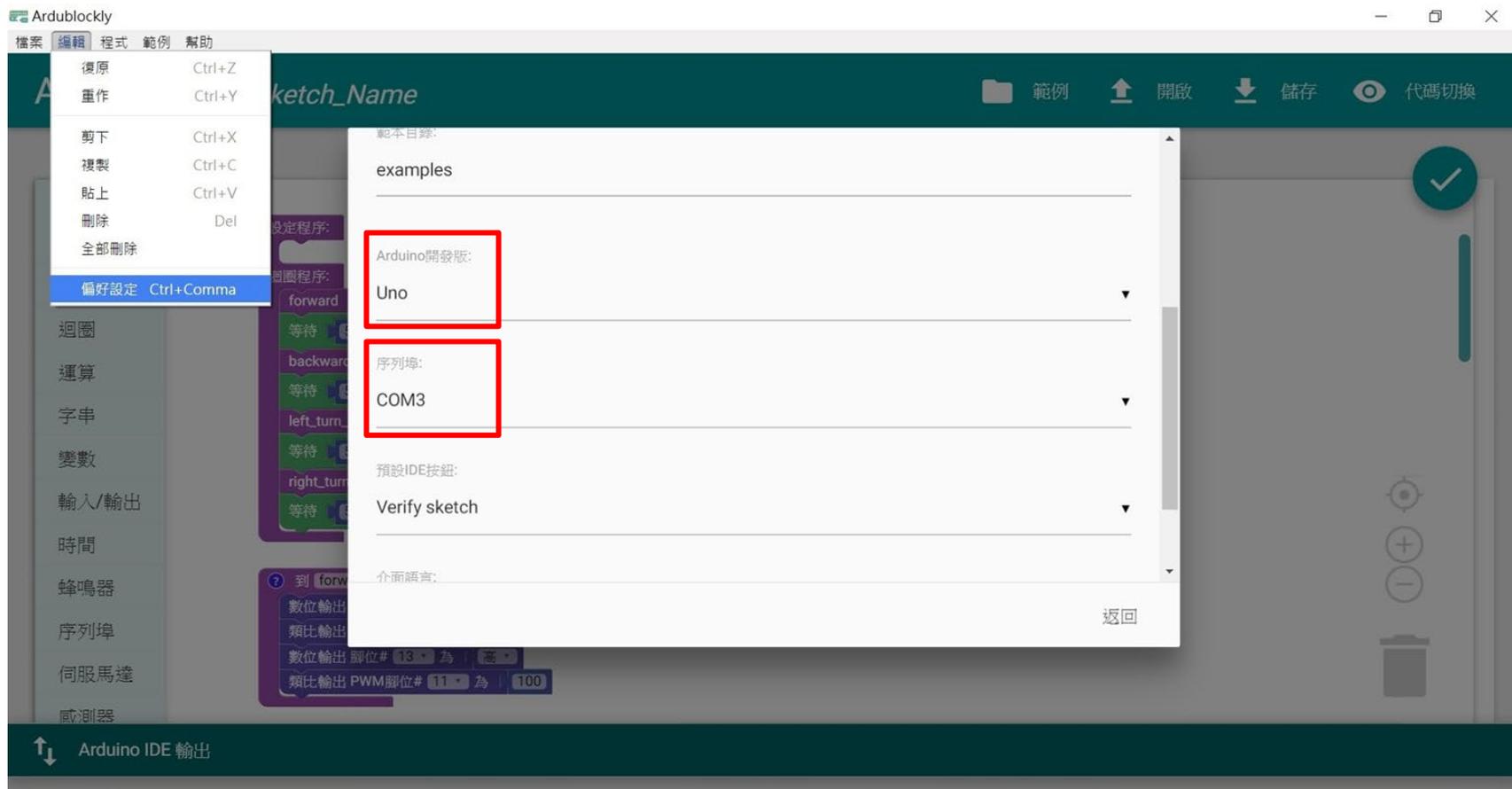
# 程式環境設定

下載並解壓縮Ardublockly，執行ardublockly\_run.bat。



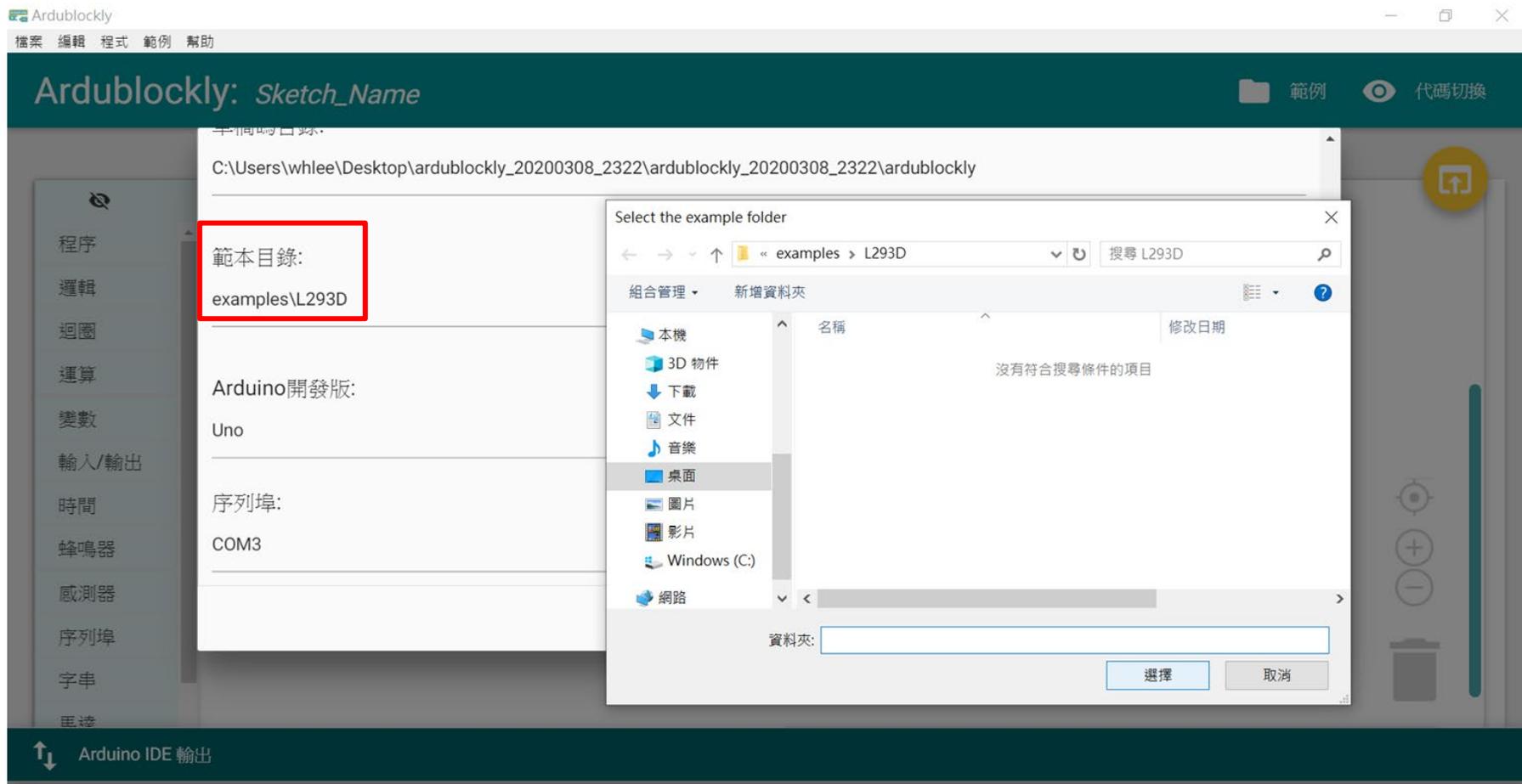
# 快速使用ARDUBLOCKLY

點選編輯>偏好設定，確定開發板&序列埠是否正確。



# 快速使用ARDUBLOCKLY

## 檢查範本目錄位置至ardublockly / examples / L293D資料夾



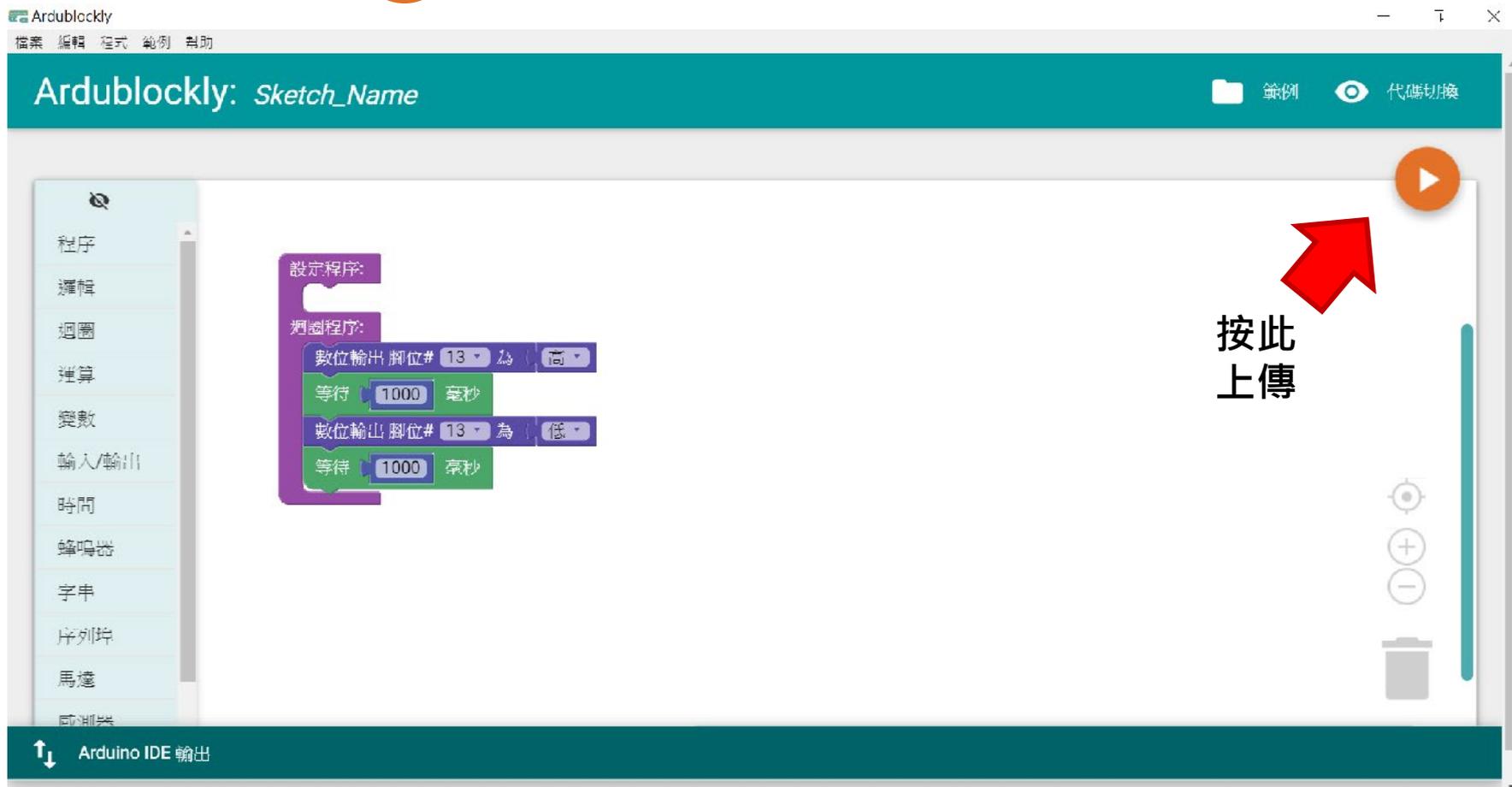
# 快速使用ARDUBLOCKLY

點選範例，點選清單匯入範本



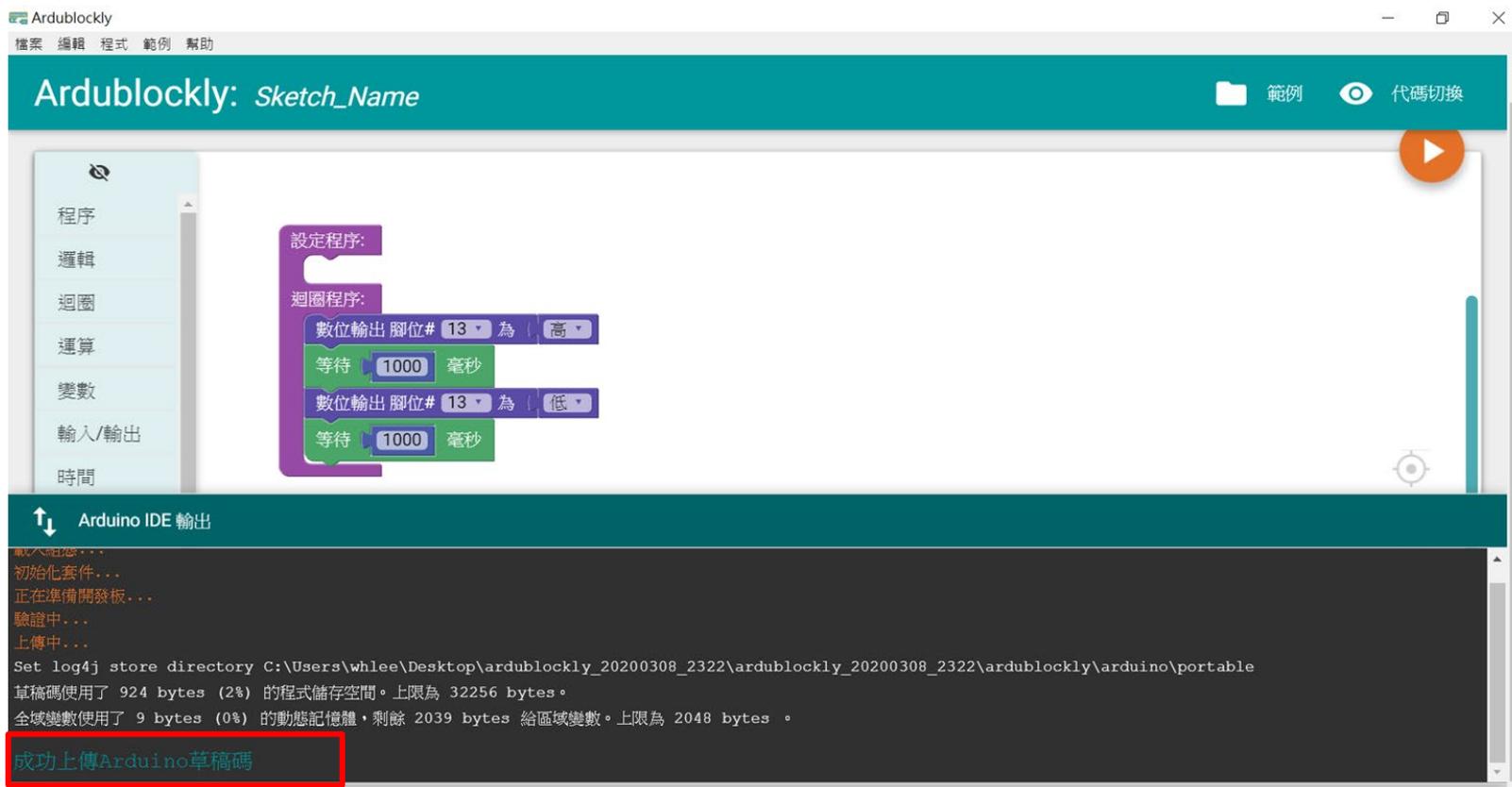
# 快速使用ARDUBLOCKLY

點選上傳圖示 ，傳送程式碼至Arduino。



# 快速使用ARDUBLOCKLY

點選下方 Arduino IDE輸出監控視窗，確認上傳狀態。



The screenshot displays the Ardublockly web application interface. At the top, the title bar reads "Ardublockly: Sketch\_Name". Below the title bar is a navigation menu with options: "檔案", "編輯", "程式", "範例", and "幫助". The main workspace shows a block-based sketch with the following code:

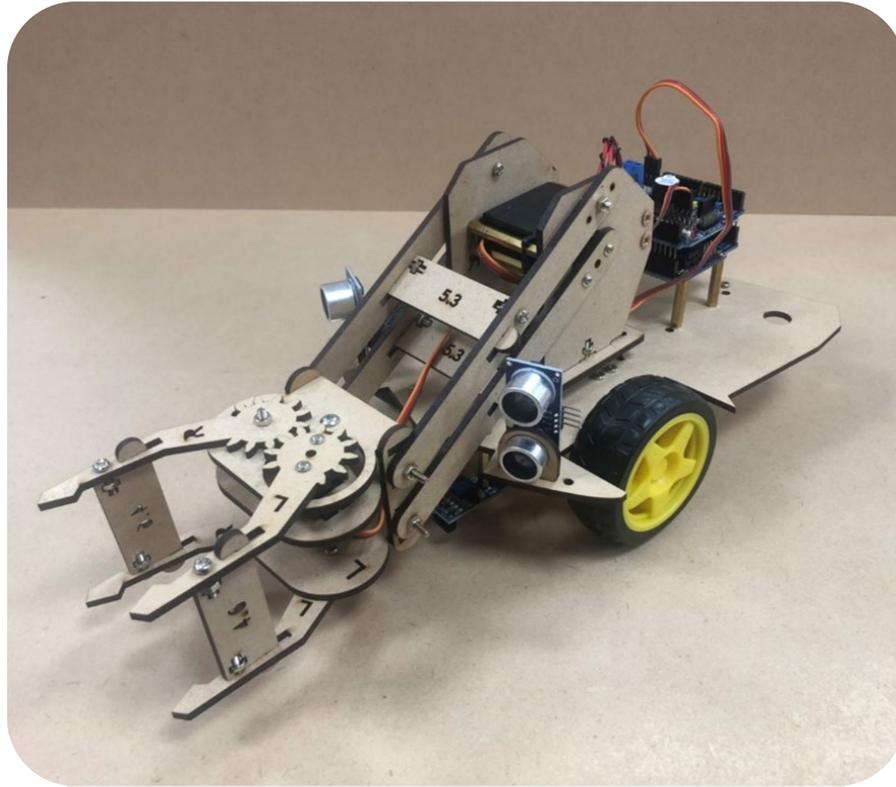
```
設定程序:  
迴圈程序:  
 數位輸出 腳位# 13 為 高  
 等待 1000 毫秒  
 數位輸出 腳位# 13 為 低  
 等待 1000 毫秒
```

On the right side of the workspace, there is a play button icon. Below the workspace is the "Arduino IDE 輸出" (Arduino IDE Output) window, which shows the following text:

```
Set log4j store directory C:\Users\whlee\Desktop\ardublockly_20200308_2322\ardublockly_20200308_2322\ardublockly\arduino\portable  
草稿碼使用了 924 bytes (2%) 的程式儲存空間。上限為 32256 bytes。  
全域變數使用了 9 bytes (0%) 的動態記憶體，剩餘 2039 bytes 給區域變數。上限為 2048 bytes。  
成功上傳Arduino草稿碼
```

The last line, "成功上傳Arduino草稿碼", is highlighted with a red rectangular box.

# START! 智慧小車



-單元4 TT 直流馬達操作-

# 車輛前進

設定程序:

L293D 頻道 2 指令 正轉 速度 255

L293D 頻道 3 指令 正轉 速度 255

迴圈程序:

頻道2(右輪) 方向 速度

頻道3(左輪) 方向 速度

先不要放到迴圈裡，觀察一下馬達運轉情形。

# 車輛前進

設定程序:

L293D 頻道 2 ▾ 指令 正轉 ▾ 速度 255

L293D 頻道 3 ▾ 指令 正轉 ▾ 速度 255

頻道2(右輪) 方向 速度

頻道3(左輪) 方向 速度

迴圈程序:

為什麼沒有放在迴圈卻能不停運轉？

馬達控制與電燈開關相同，是一種「狀態設定」。

打開後便會一直維持著，跟賽車遊戲放開按鍵便會停止不一樣。

# 車輛停止



**指令設為停止，或將速度設為0。**

在正轉或反轉時，速度設定100以下時，車輛也會停止。

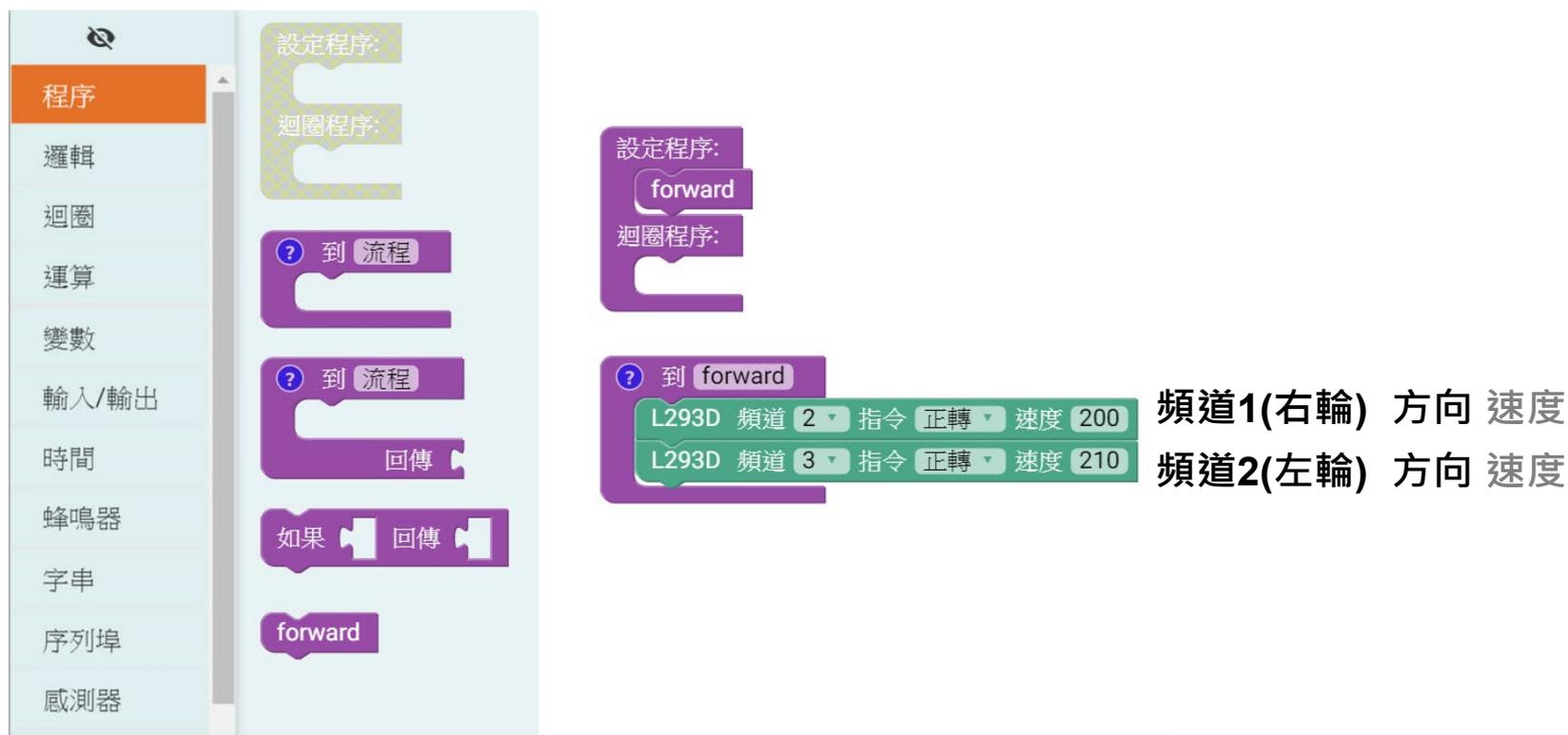
這時馬達仍有送電，只是電壓不足以推動車輪。



**想想看：車輛後退該如何設定？**

# 副程式的使用

我們可以將常用的程式碼包裝成一個副程式



The image shows a programming interface with a sidebar on the left containing categories: 程序 (Program), 邏輯 (Logic), 迴圈 (Loop), 運算 (Math), 變數 (Variables), 輸入/輸出 (Input/Output), 時間 (Time), 蜂鳴器 (Buzzer), 字串 (Strings), 序列埠 (Serial), and 感測器 (Sensors). The main workspace displays several sub-program blocks:

- A '設定程序:' (Set Program) block with a 'forward' sub-block.
- An '迴圈程序:' (Loop Program) block with a '到 流程' (Go to Flow) block.
- A '到 流程' (Go to Flow) block with an '回傳' (Return) block.
- An '如果' (If) block with an '回傳' (Return) block.
- A 'forward' block.

Two detailed views of sub-program blocks are shown to the right:

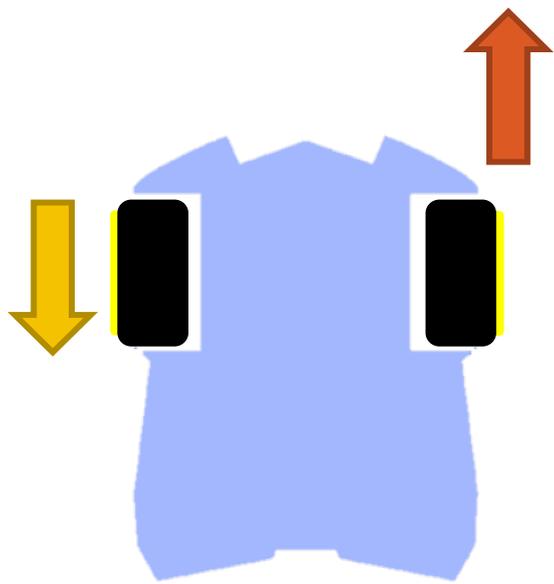
- The first view shows a '設定程序:' block containing a 'forward' sub-block and an '迴圈程序:' block.
- The second view shows a '到 forward' block containing two configuration blocks:
  - L293D 頻道 2 指令 正轉 速度 200
  - L293D 頻道 3 指令 正轉 速度 210

Labels for the configuration blocks are:

- 頻道1(右輪) 方向 速度
- 頻道2(左輪) 方向 速度

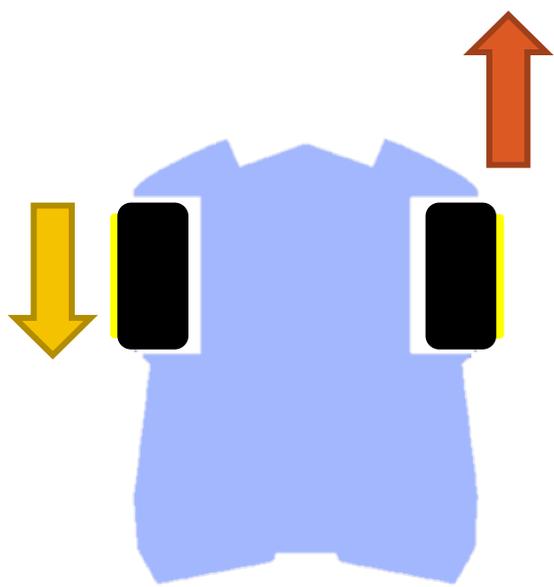
由於左右輪可能因為馬達、輪胎、摩擦力等原因出現誤差，導致車輛無法穩定的走直線，這時可以**稍微調整轉速**來修正。

# 轉彎的方式

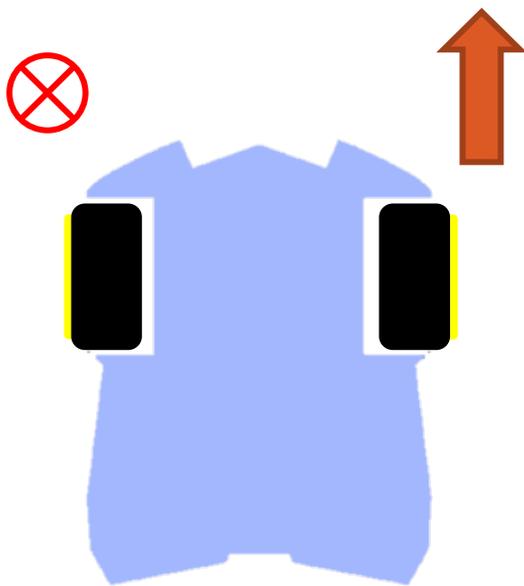


原地自轉  
turn\_0

# 轉彎的方式

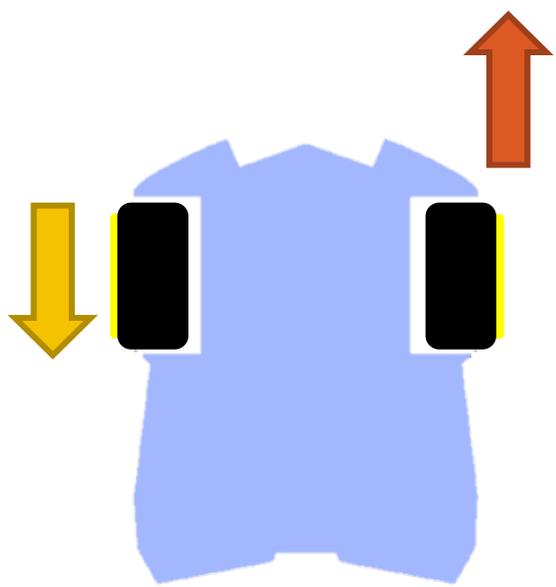


原地自轉  
turn\_0

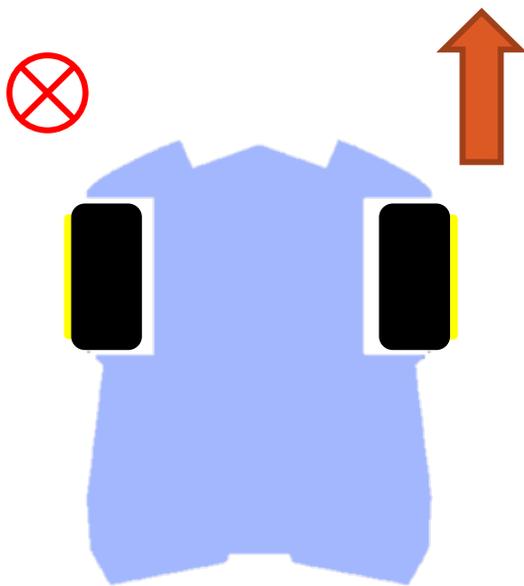


單輪轉彎  
turn\_1

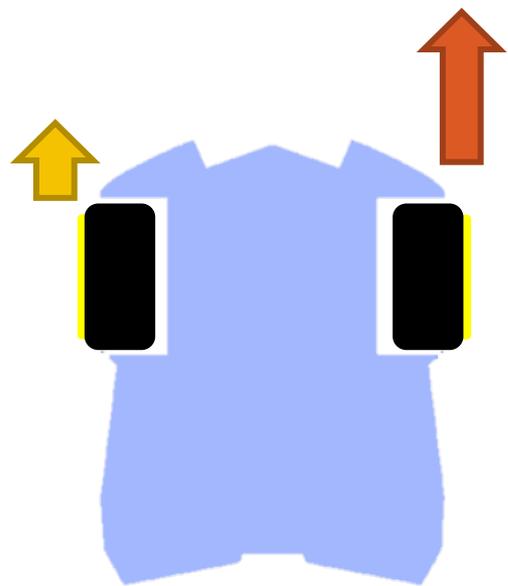
# 轉彎的方式



原地自轉  
turn\_0



單輪轉彎  
turn\_1



差速轉彎  
turn\_2

# 轉彎的方法

比較一下不同的  
轉彎方法有無差異？

```
設定程序:  
迴圈程序:  
forward  
等待 3000 毫秒  
left_turn_0  
等待 3000 毫秒  
left_turn_1  
等待 3000 毫秒  
left_turn_2  
等待 3000 毫秒
```

```
? 到 forward  
L293D 頻道 2 指令 正轉 速度 200  
L293D 頻道 3 指令 正轉 速度 200
```

```
? 到 left_turn_0  
L293D 頻道 2 指令 正轉 速度 200  
L293D 頻道 3 指令 反轉 速度 200
```

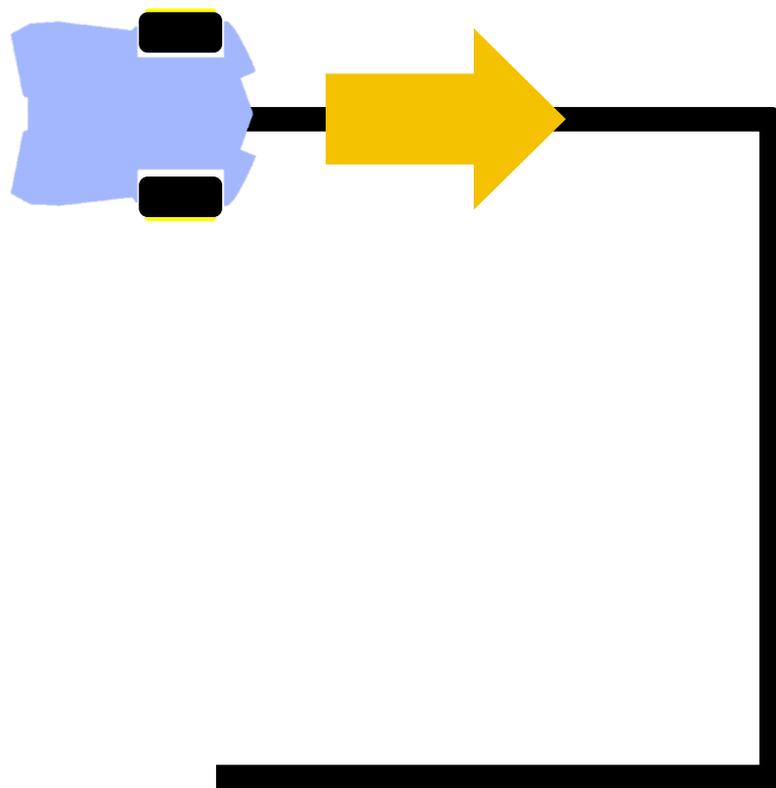
```
? 到 left_turn_1  
L293D 頻道 2 指令 正轉 速度 200  
L293D 頻道 3 指令 停止 速度 0
```

```
? 到 left_turn_2  
L293D 頻道 2 指令 正轉 速度 200  
L293D 頻道 3 指令 正轉 速度 150
```

# 活動1：□字轉彎練習(國小)

提示：

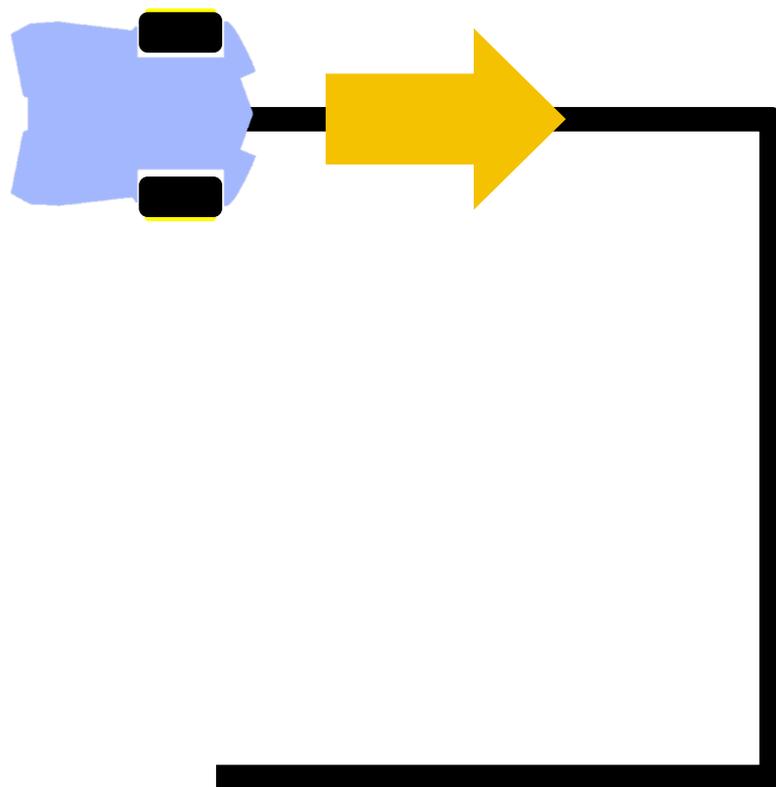
試著調整秒數，讓車輛可以精確地轉90度。



想想看：哪一種方式比較能夠完美地完成90度轉彎？

# 活動1：□字轉彎練習(國小)

```
設定程序:  
forward  
等待 1000 毫秒  
right_turn_0  
等待 500 毫秒  
forward  
等待 1000 毫秒  
right_turn_0  
等待 500 毫秒  
forward  
等待 1000 毫秒  
stop  
迴圈程序:
```



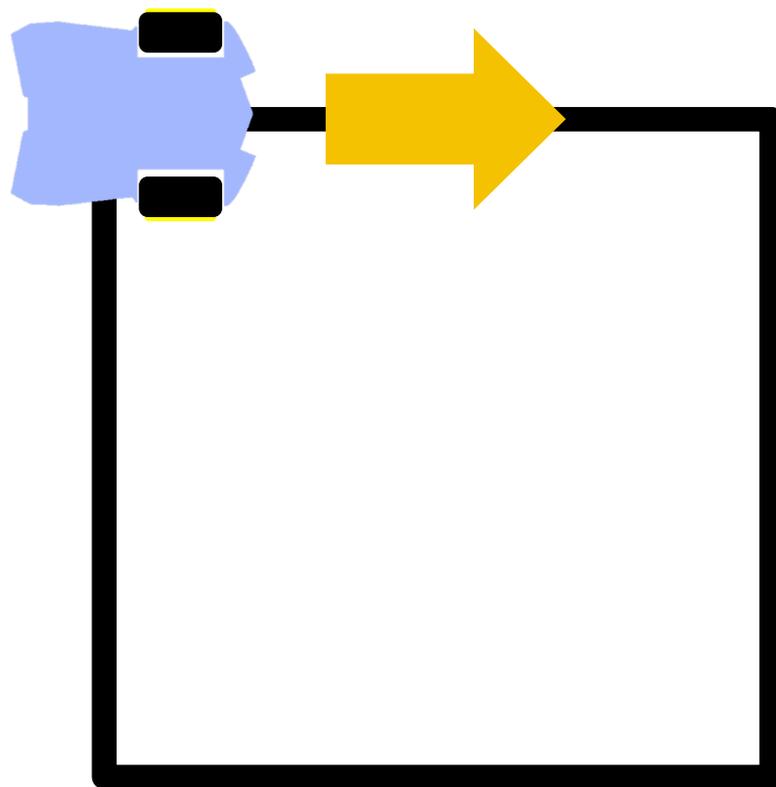
## 延伸：口字轉彎練習(國中)

※先測試要原地旋轉多久，  
能剛好轉向90度，最後再  
加上直線線段部分



想想看：若移動軌跡變成  
三角形，可能原因是？

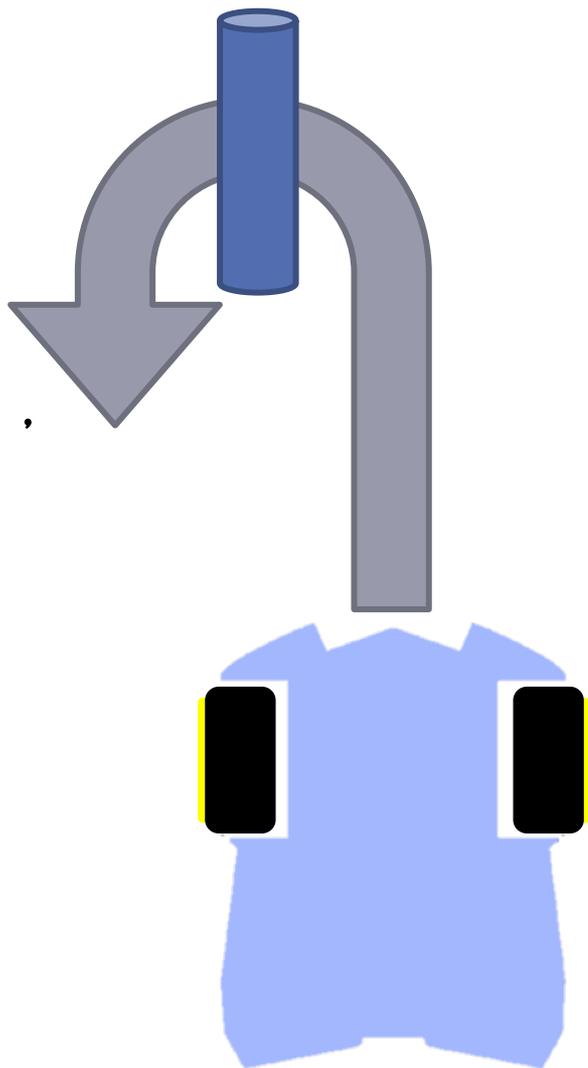
想想看：若移動軌跡變成  
五角形，可能原因是？



# 差速轉彎練習(國小)

※練習：繞過瓶子折返

※先測試要差速旋轉多久，  
能剛好繞半圈，最後再  
加上直線線段部分

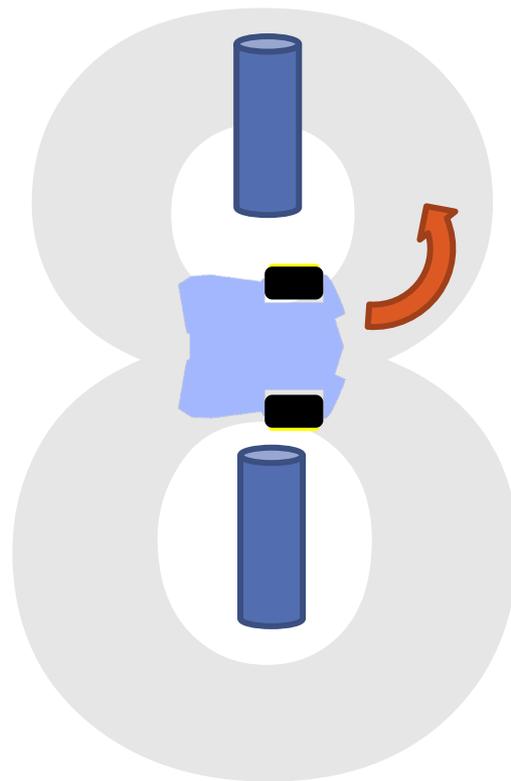


```
設定程序:  
forward  
等待 1000 毫秒  
left_turn_2  
等待 5000 毫秒  
forward  
等待 1000 毫秒  
stop  
迴圈程序:
```

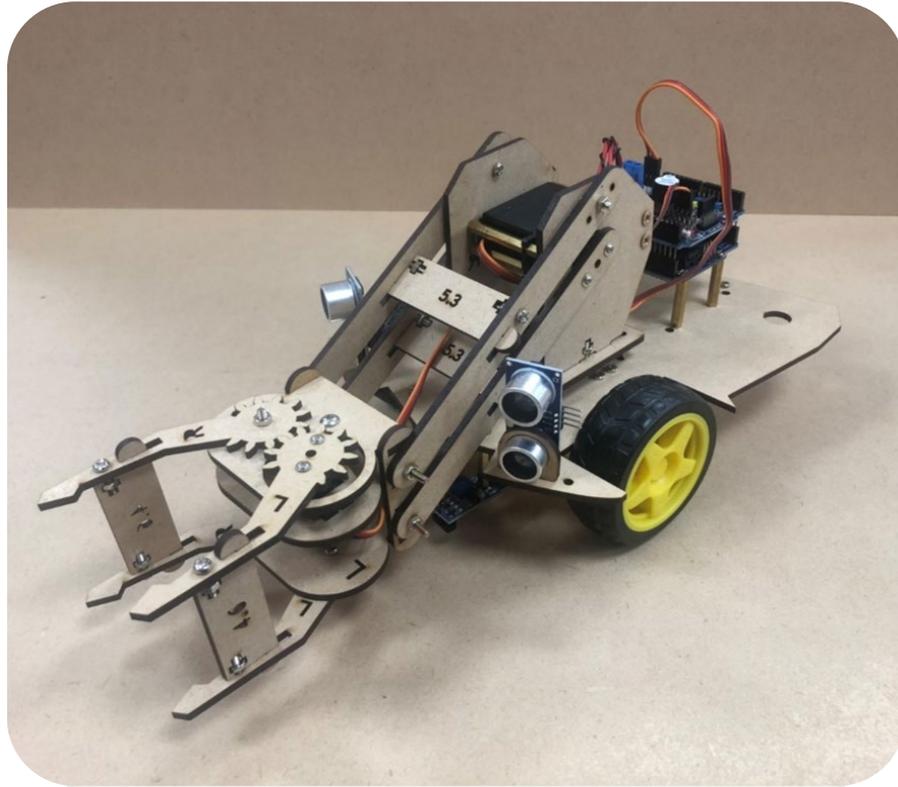
## 活動2：繞8字形(國中)

挑戰：

利用差速轉彎在兩個罐子間繞8字



# START! 智慧小車

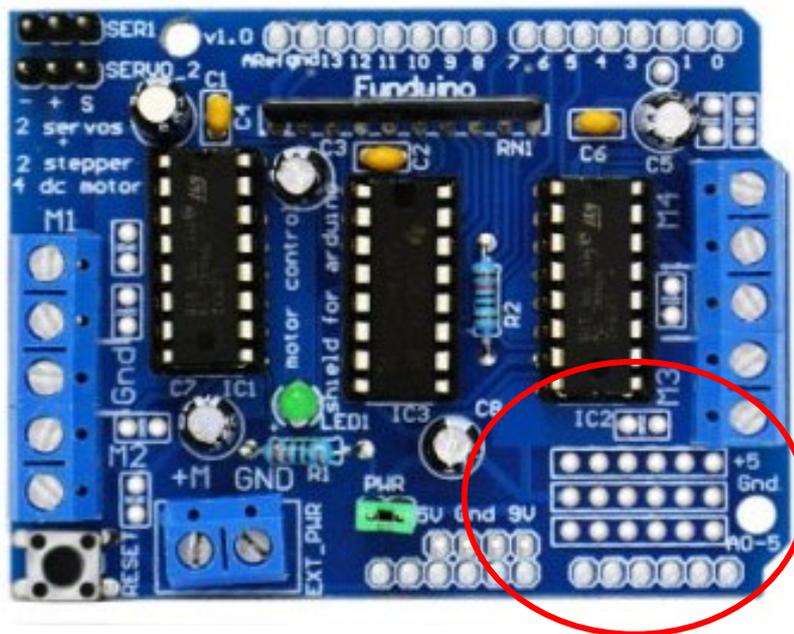


-單元**5** 紅外線感測器操作-

## 類比/數位接腳

**A0 - A5**接腳可用於模擬輸入，  
亦可作為**D14-D19**數位輸入輸出接腳，  
可支援兩種類型感測器。

※L293D的類比/數位接腳在供電處設計(**GND**置中**5V**在側)，與一般配置不同，  
請務必小心以免錯接燒毀元件，尤其不能直插MG90/MG995伺服馬達。



5V 電源

GND接地

類比/數位接腳

# 紅外線感測模組

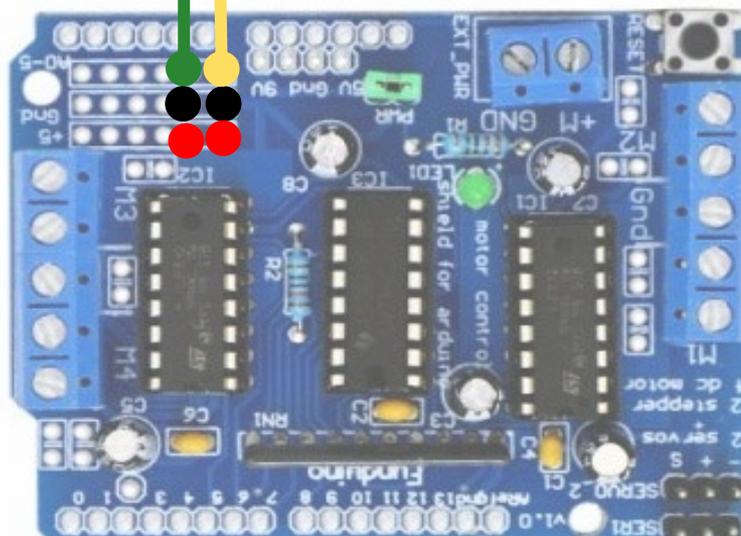
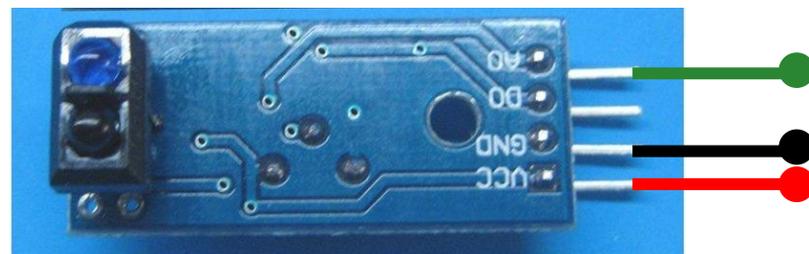
## 功能介紹：

- 發射紅外線後，讀取反射回來的紅外線強度：
  - 黑色區域：容易吸收光線，反射回來的光線較少。
  - 白色區域：容易反射光線，反射回來的光線較多。

左側(A0)感測器接  
法

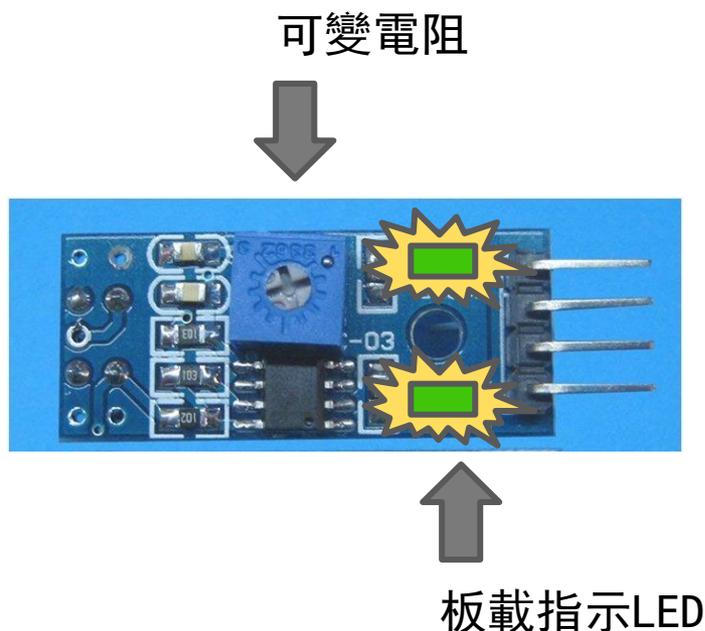


右側(A1)感測器接  
法



## 補充：紅外線感測模組校正

- 紅外線感測模組容易受到場地照明、黑線材質等因素影響，每次使用一定要記得先做以下檢測：
- 使用類比模式時，可用程式讀取數值。
- 使用數位模式時，可用板載指示燈來觀察是否能判斷黑線，必要時可用板載的可變電阻調整校正數值。



請用手在感測器下方揮一揮  
手在下方時，有光線反射，亮兩顆燈。  
手放開時，無光線反射，亮一顆燈。

## 讀取紅外線感測器數值

請將紅外線感測器接線至控制板

使用類比數值，左邊為A0，右邊為A1

設定程序:

設定 serial 的序列通訊速度為 9600 位元/秒

迴圈程序:

- serial 送出 “ left value = ” 換行
- serial 送出 從 A0 讀取紅外線光感測器的數值 - 換行
- serial 送出 “ , ” 換行
- serial 送出 “ right value = ” 換行
- serial 送出 從 A1 讀取紅外線光感測器的數值 -  換行

等待 1000 毫秒



# 讀取紅外線感測器數值

程式上傳完畢後，點選序列埠監控視窗，即可觀察小車傳回的數值。

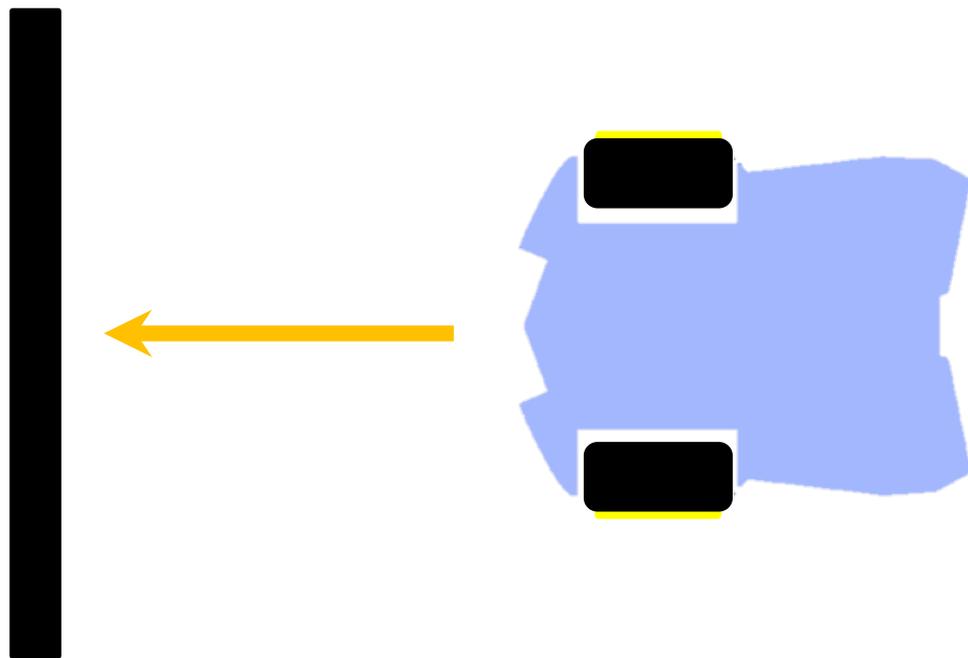
The screenshot shows the Ardublockly v20200526\_0006 interface. The top toolbar includes icons for 'Serial Monitor' (序列埠監控), 'Examples' (範例), 'Code Switch' (代碼切換), and 'Example Programs with Arduino Compatible' (範例程式與arduino可攜版). A yellow arrow points to the 'Serial Monitor' icon, with the text '按此監看' (Click here to monitor) next to it. The main workspace contains a code block for '設定程序:' (Setup) and '迴圈程序:' (Loop). The 'Setup' block sets the serial communication speed to 9600 bps. The 'Loop' block contains the following steps: 1. Send 'left value = ' to the serial port. 2. Read the value from A0. 3. Send ', ' to the serial port. 4. Send 'right value = ' to the serial port. 5. Read the value from A1. 6. Wait 1000 milliseconds.

```
設定程序:  
設定 serial 的序列通訊速度為 9600 位元/秒  
迴圈程序:  
serial 送出 "left value = "  
serial 送出 從 A0 讀取紅外線光感測器的數值  
serial 送出 ", "  
serial 送出 "right value = "  
serial 送出 從 A1 讀取紅外線光感測器的數值  
等待 1000 毫秒
```

# 前進遇到黑線停下

練習：

前進直到紅外線感測器  
偵測到黑線後停下



# 前進遇到黑線停下



想想看：停下後若將車輕輕推離黑線會發生什麼事？



想想看：如果將Block移到設定程序會發生什麼事？

# 當...重複執行...

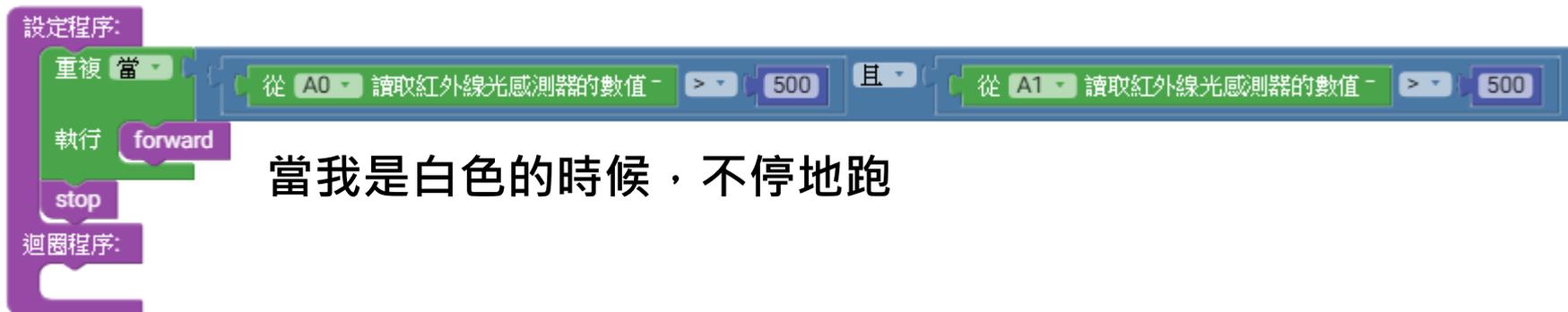
while()函式的應用。



試試看：「當」跟「直到」兩種寫法會產生什麼不一樣的結果？

## 當...重複執行...

while()函式的應用。



設定程序:

重複 當

從 A0 讀取紅外線光感測器的數值 - > 500 且 從 A1 讀取紅外線光感測器的數值 - > 500

執行 forward

當我是白色的時候，不停地跑

stop

迴圈程序:



設定程序:

重複 直到

從 A0 讀取紅外線光感測器的數值 - < 500 且 從 A1 讀取紅外線光感測器的數值 - < 500

執行 forward

直到遇到黑色前，不停地跑

stop

迴圈程序:

補充：在C++語法中，「當」就是while()，「直到」就是while(!())。

「當」是滿足()的條件才執行迴圈，「直到」是達成()內的條件離開迴圈  
在小車任務中，我們習慣使用「直到」來撰寫程式。

## 活動3：計算黑線數量(國小國中)

從起點出發

請停在第三條黑線上

停在這一條



## 活動3：計算黑線數量

設定程序：

第一條



第二條



第三條



迴圈程序：

強制走一段路

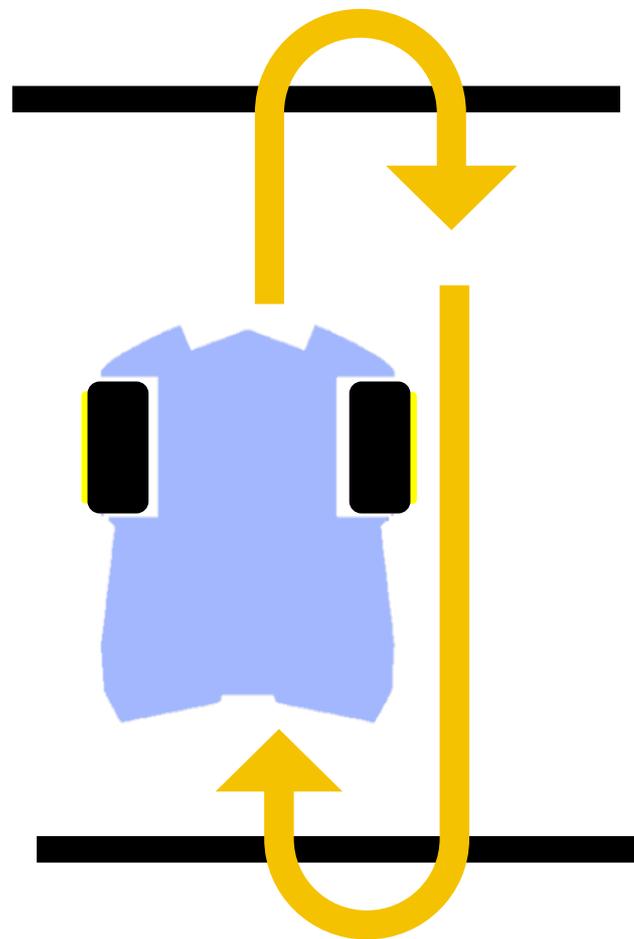
強制走一段路

需要視黑線間隔距離調整等待秒數，  
若間距太小時容易失敗。

## 活動4：折返跑(國小國中)

練習：

在兩條黑線之間折返跑



## 活動4：折返跑(國小國中)

練習：

在兩條黑線之間折返跑

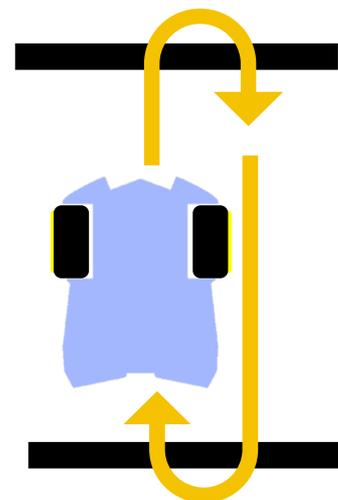
設定程序：

迴圈程序：

```
repeat (until) loop {
  from A0 read infrared sensor value < 500 and from A1 read infrared sensor value < 500
  forward
  left_turn_0
  wait 500 ms
}
```

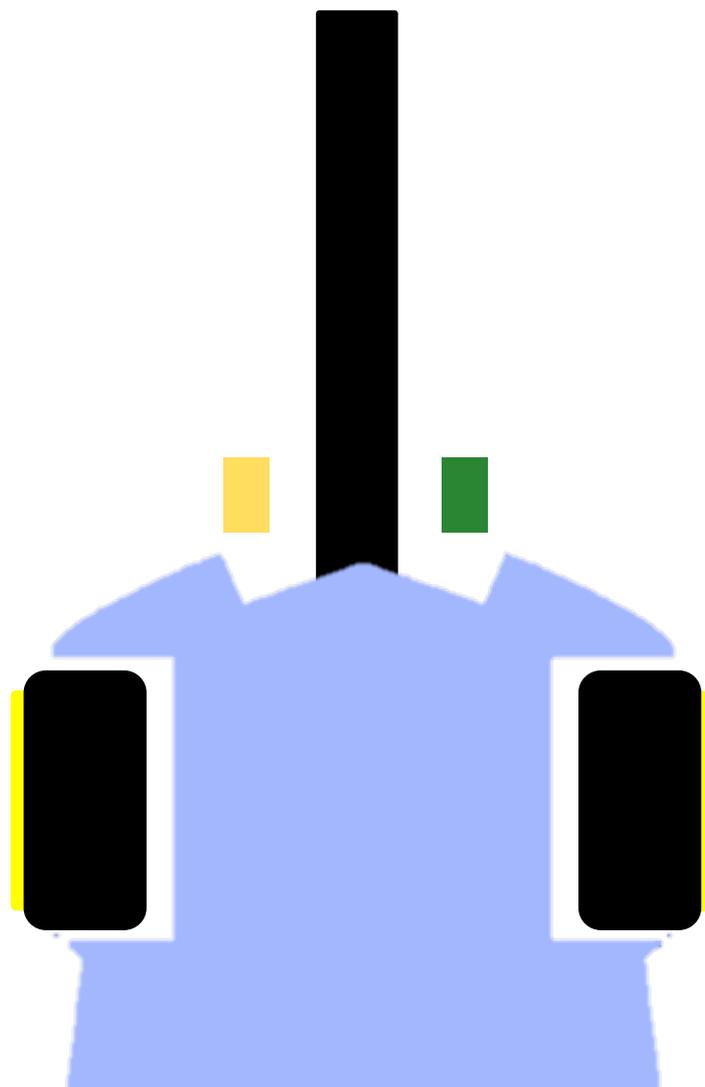
← 原地自轉

觀察旋轉結果調整等待秒數。

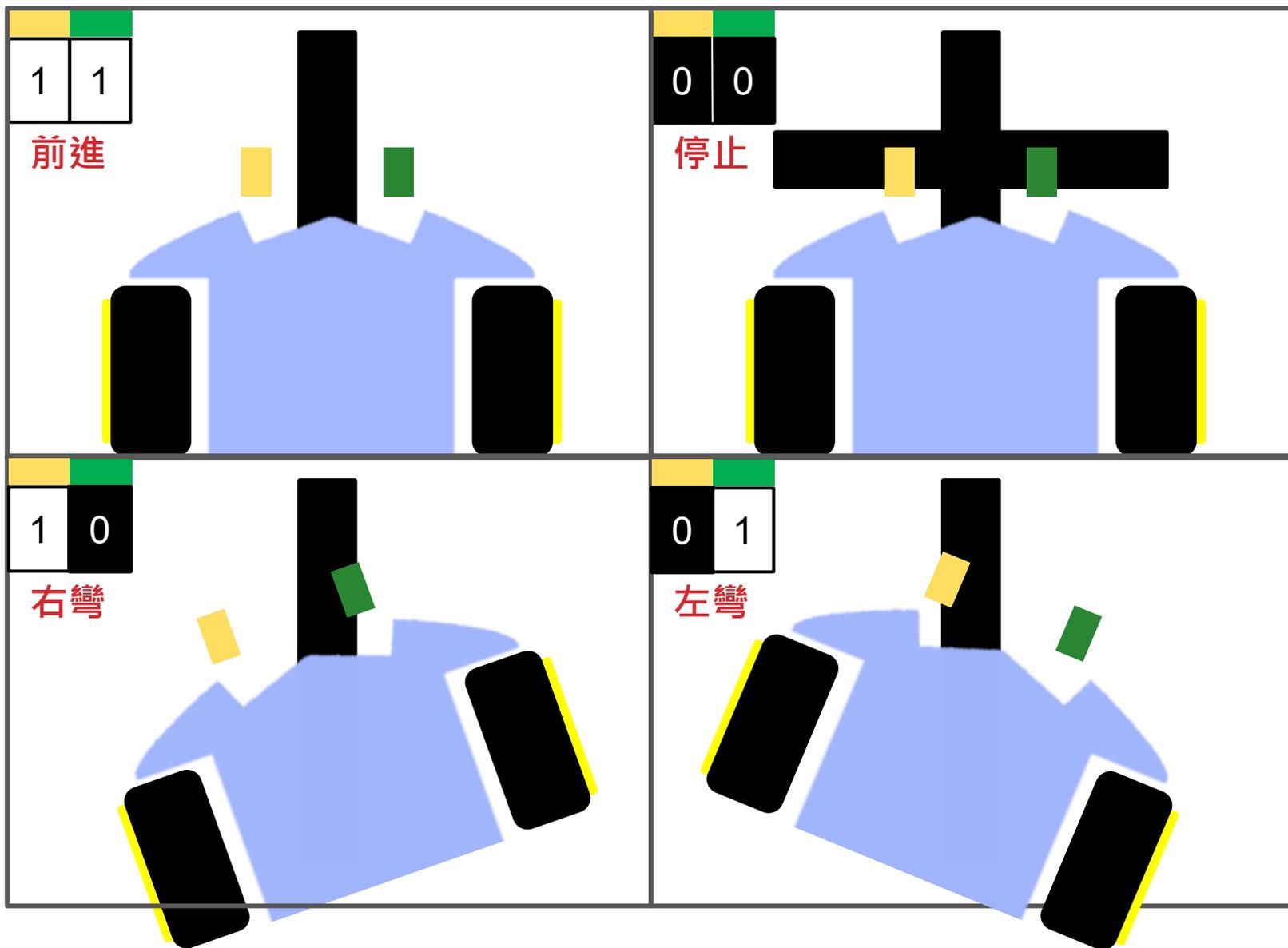


# 紅外線循跡模式

	黃色	綠色
前進	1	1
左彎	0	1
右彎	1	0
停止	0	0

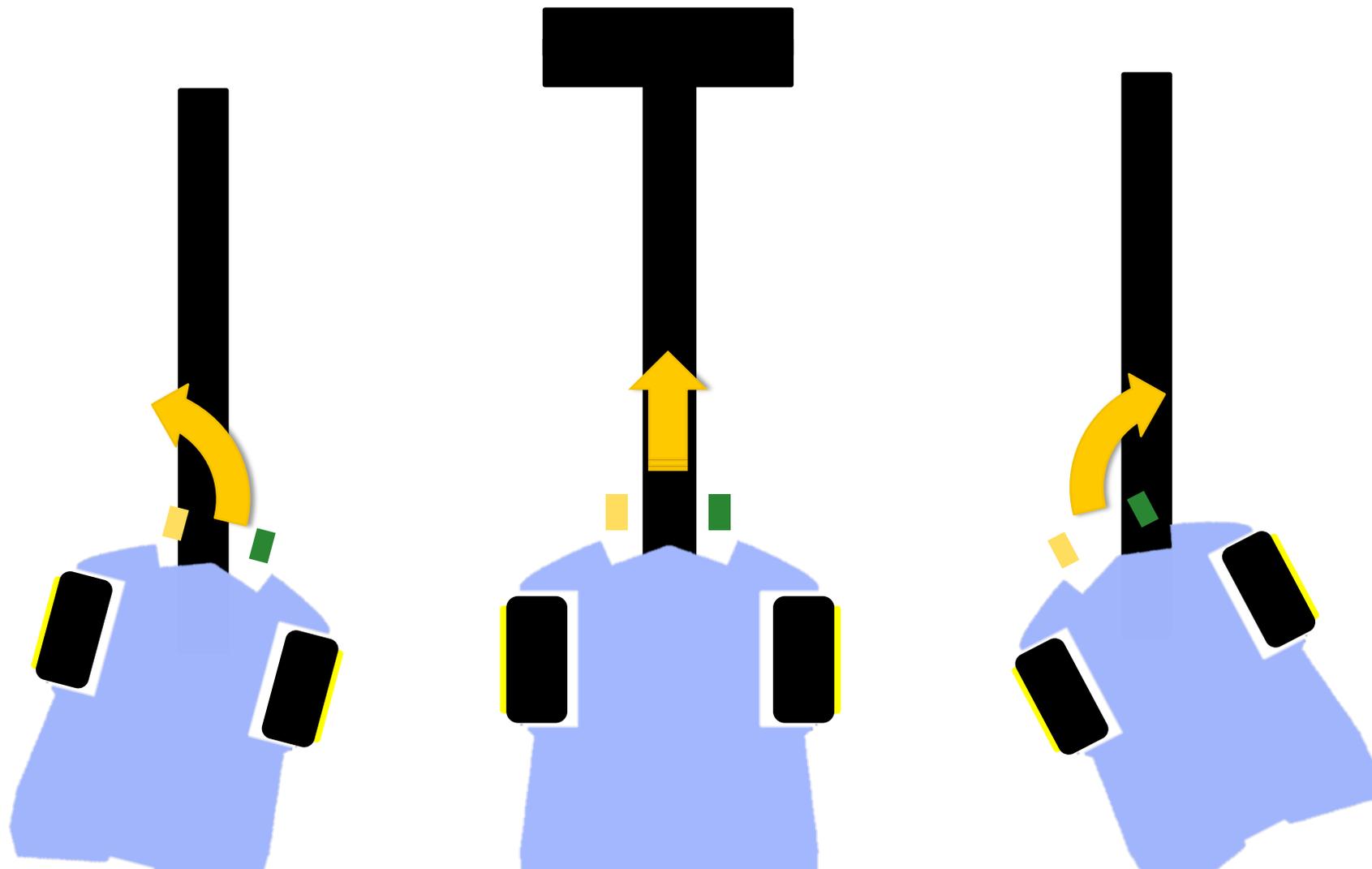


# 紅外線循跡模式 口訣：哪邊壓線往哪邊轉



# 紅外線循跡模式

口訣：哪邊壓線往哪邊轉



# 紅外線循跡

設定程序:

- 口訣**哪邊壓線往哪邊轉**

迴圈程序:

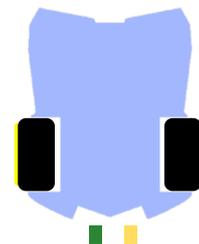
```
if (A0 >= 500 and A1 >= 500)
  forward
else if (A0 < 500 and A1 >= 500)
  left_turn_0
else if (A0 >= 500 and A1 < 500)
  right_turn_0
else if (A0 < 500 and A1 < 500)
  stop
```



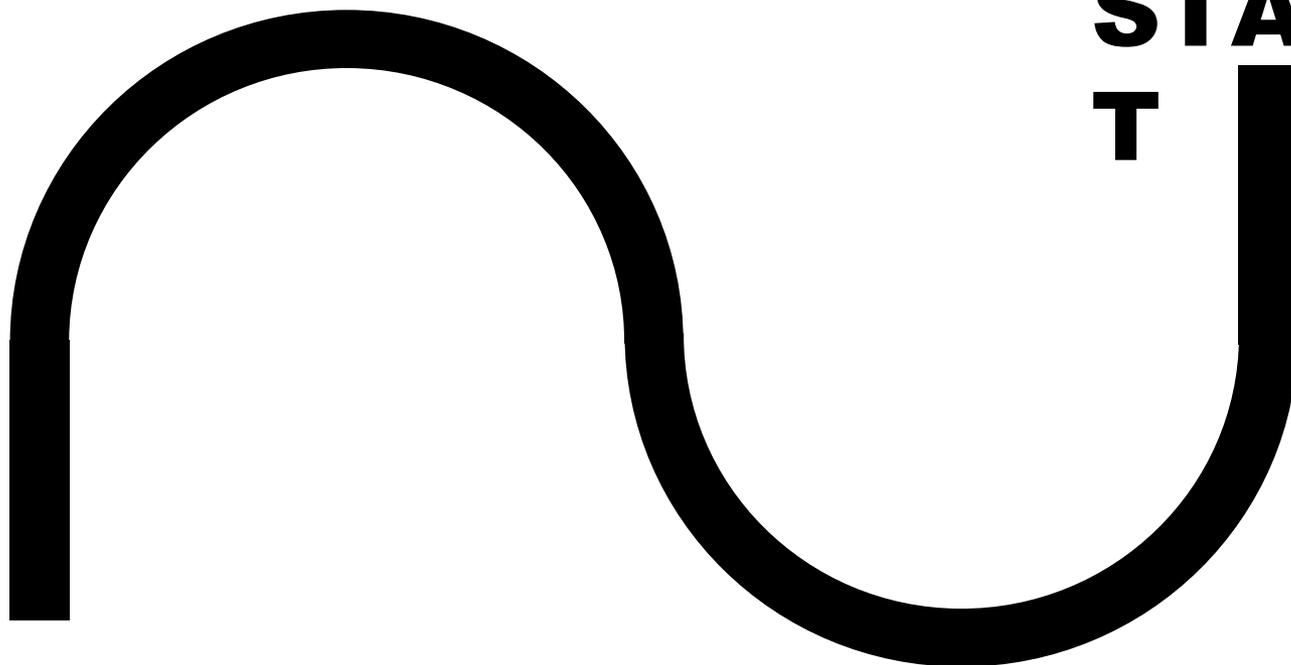
想想看：使用哪一種轉彎方式較佳？  
如果黑底白線該如何處理？  
如果線比車寬該如何處理？

# 紅外線循跡

任務：  
在S型黑線上做循跡。



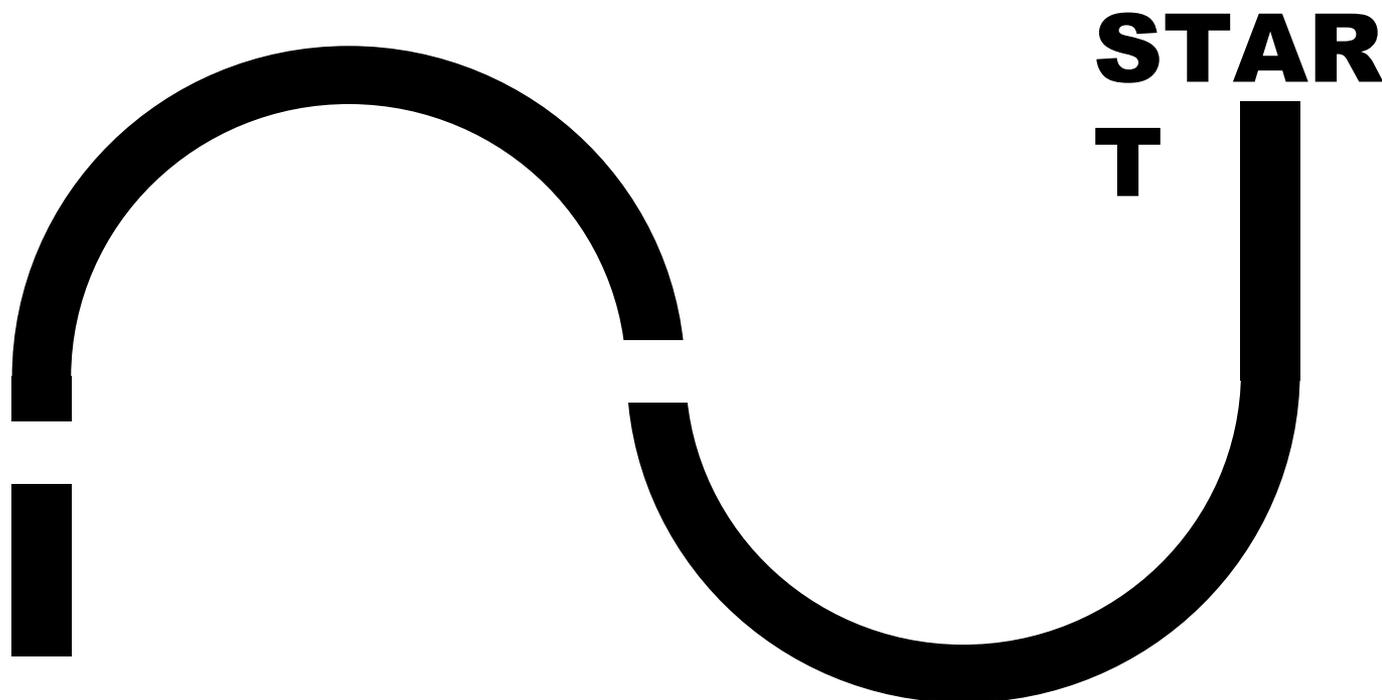
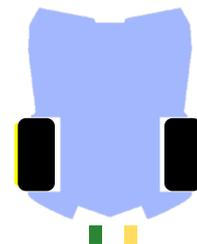
**STAR**  
**T**



# 紅外線循跡



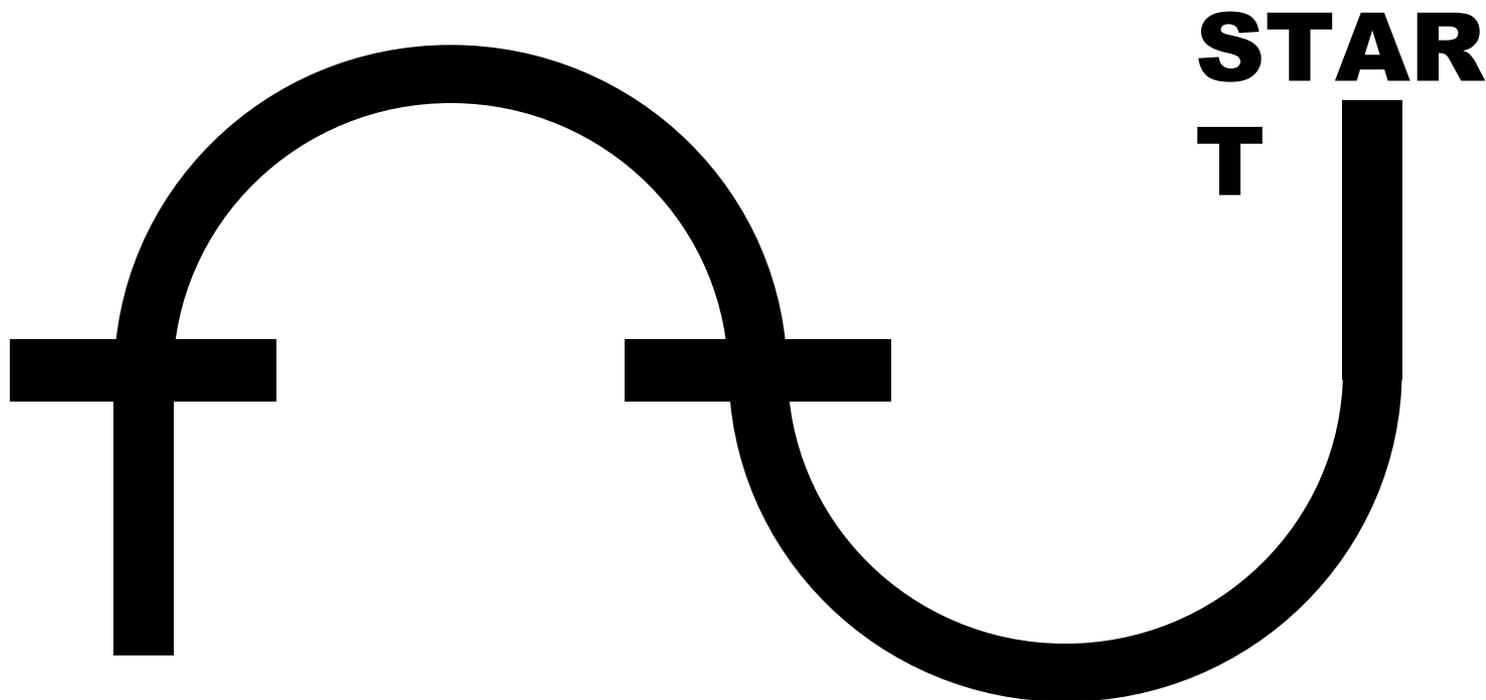
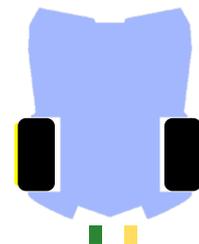
想想看：若黑線有斷線，會有什麼影響？你該怎麼辦？



# 紅外線循跡

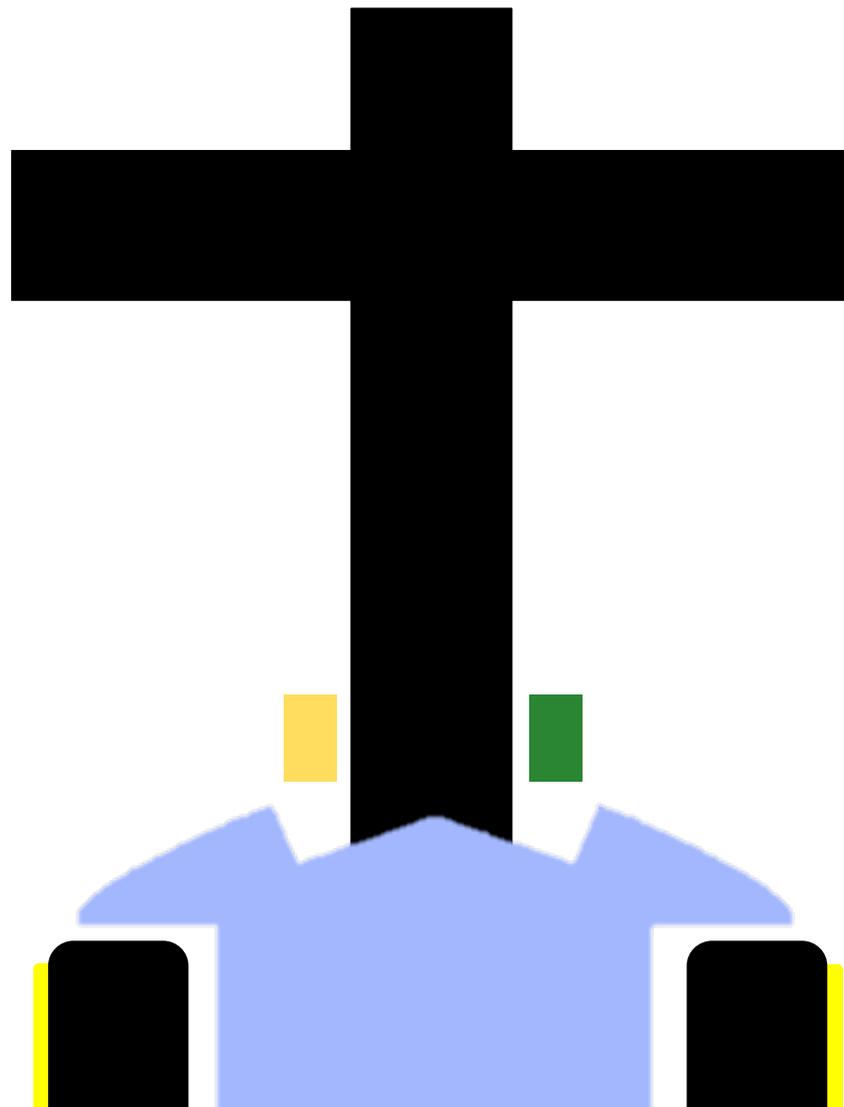


想想看：若黑線有橫線，會有什麼影響？你該怎麼辦？



## 活動5：循跡跨橫線

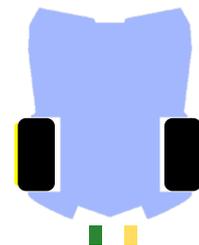
遇到橫線後  
強迫走一段路



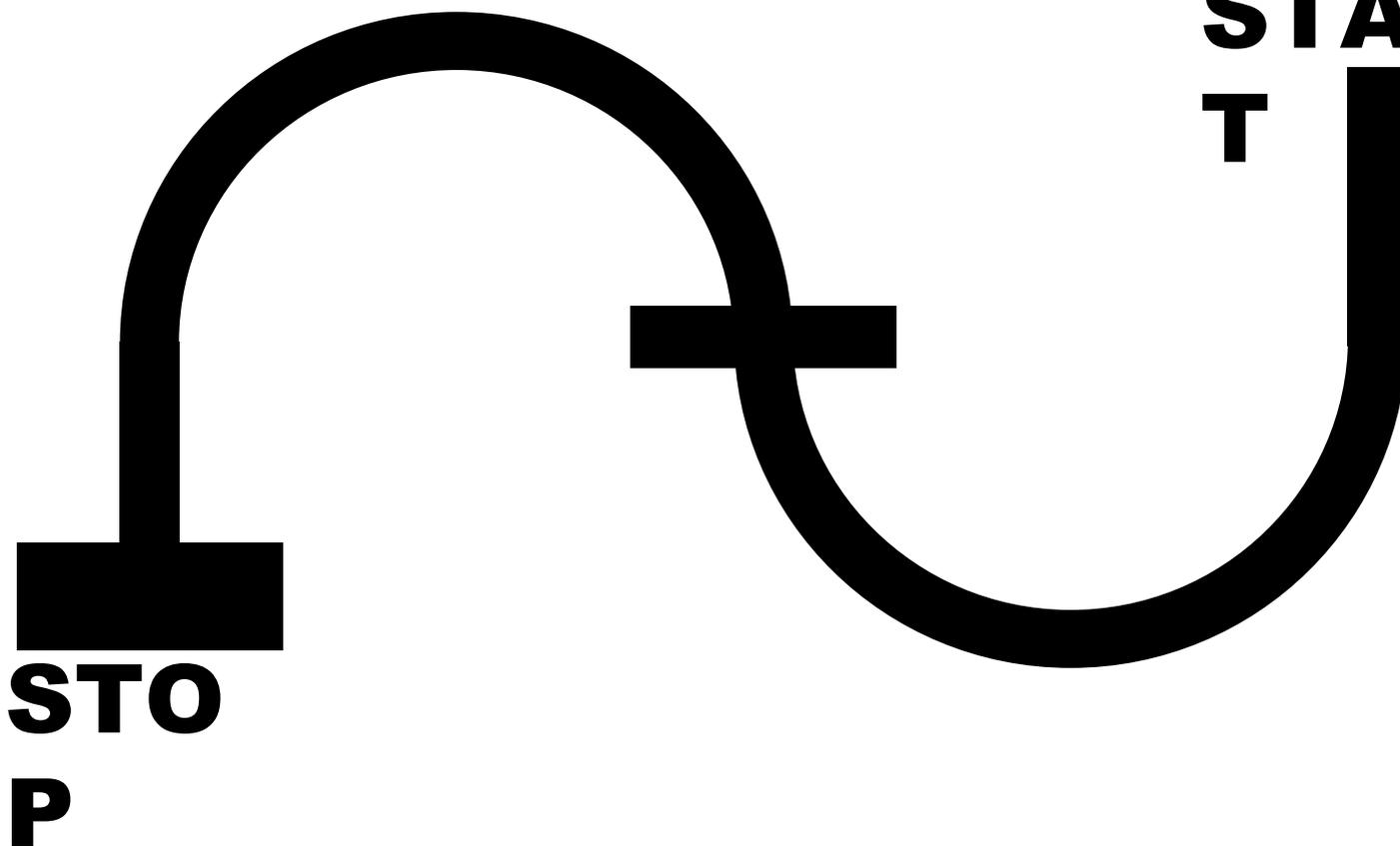
## 活動5：循跡跨橫線

練習：

在S型黑線上做循跡，必須跨越中段的橫線，  
並停在最終的寬橫線。



**STAR**  
**T**



## 活動5：循跡跨橫線

設定程序:

重複 直到

執行 如果

執行 forward

執行 如果

執行 left\_turn\_0

執行 如果

執行 right\_turn\_0

forward

等待 200 毫秒

stop

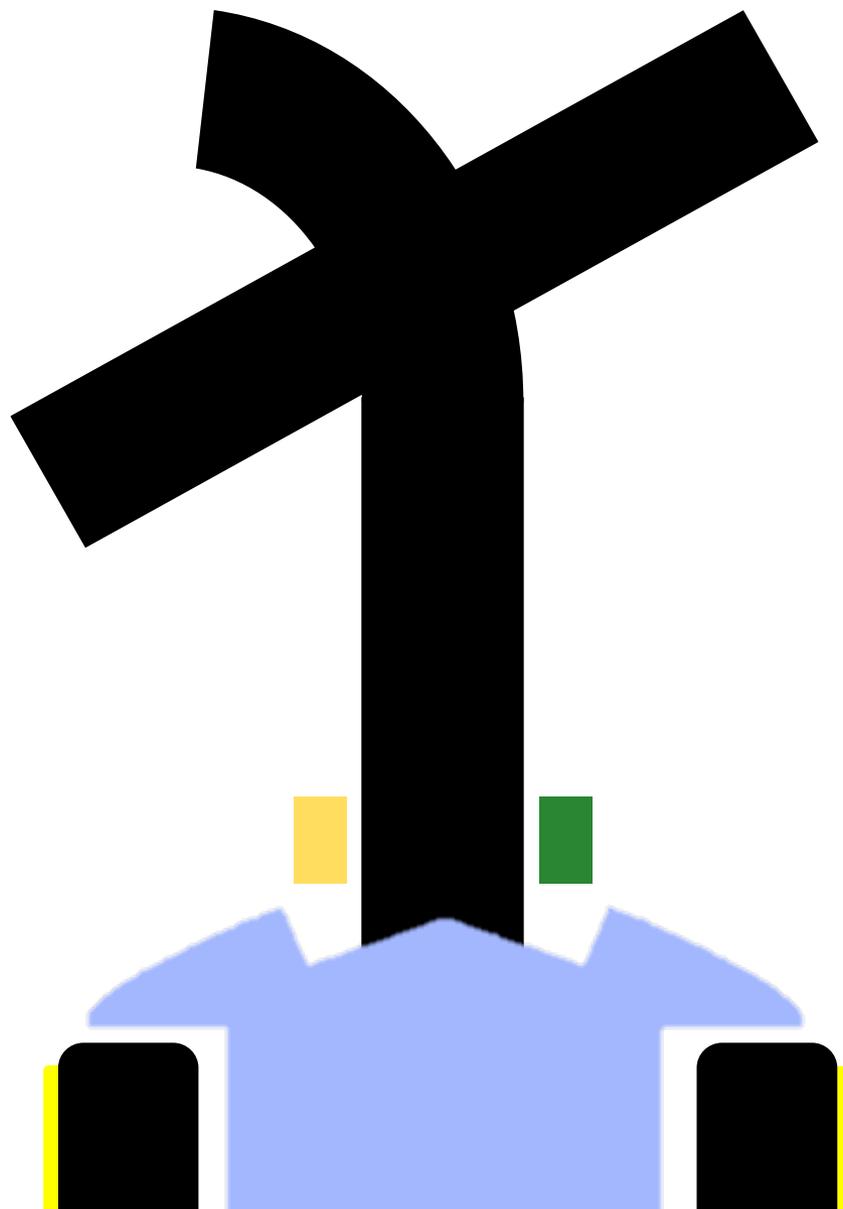
迴圈程序:

**強制走一段路**

## 特別注意：彎道中遇到橫線...

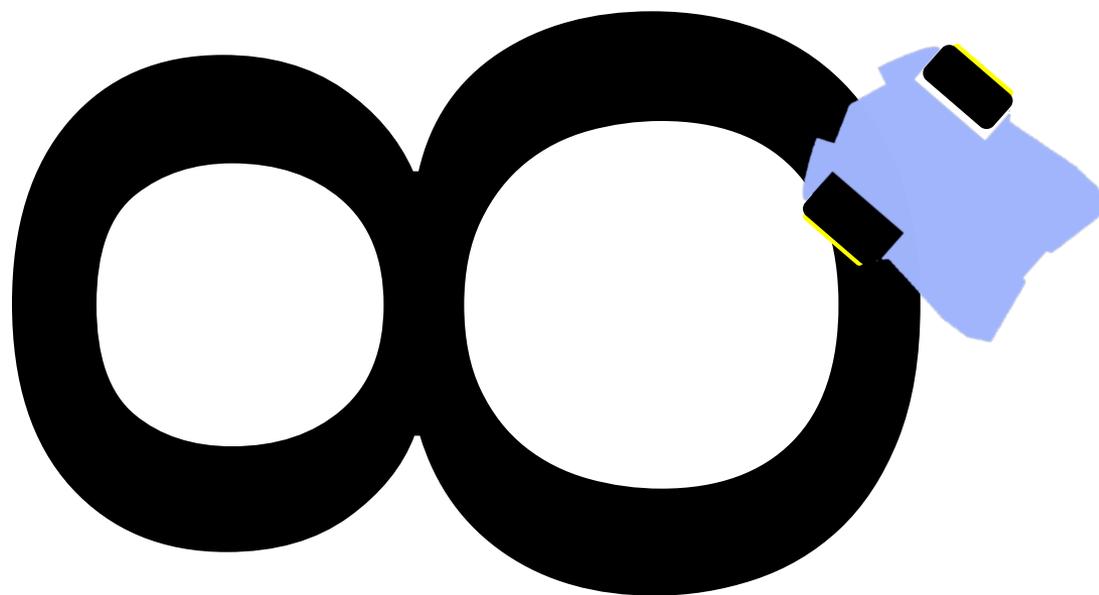


想想看：  
仔細調整前進距離避免  
感測器脫離黑線...



## 延伸挑戰(國小)：

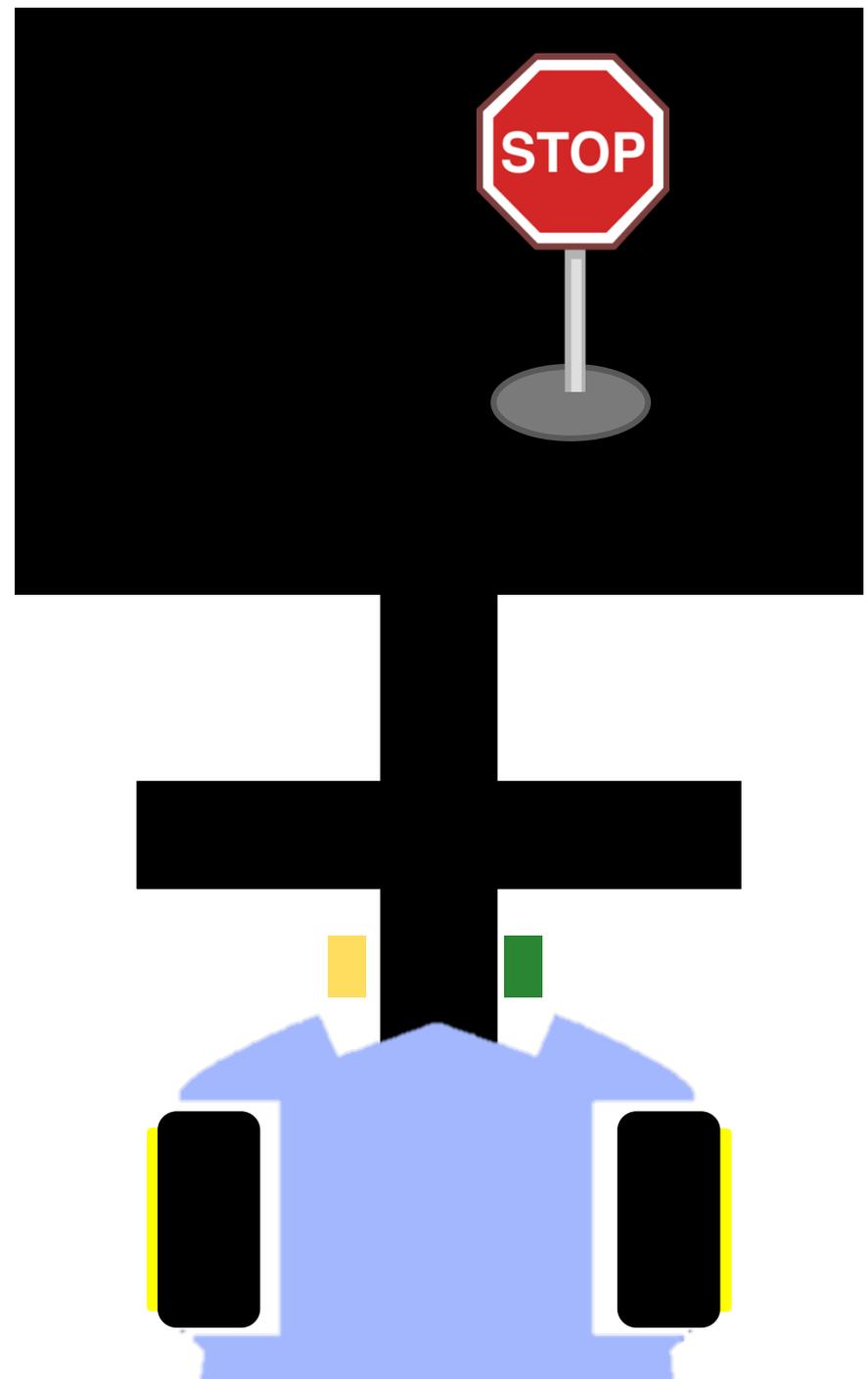
在8字型黑線上做循跡。



## 延伸挑戰 跨橫線但停在黑色區域

任務：  
車輛前進途中遇到橫線能跨過去  
但若為黑色區域則須停下來。

有時比賽的關卡會以寬度兩倍的黑線當作啟動提示。

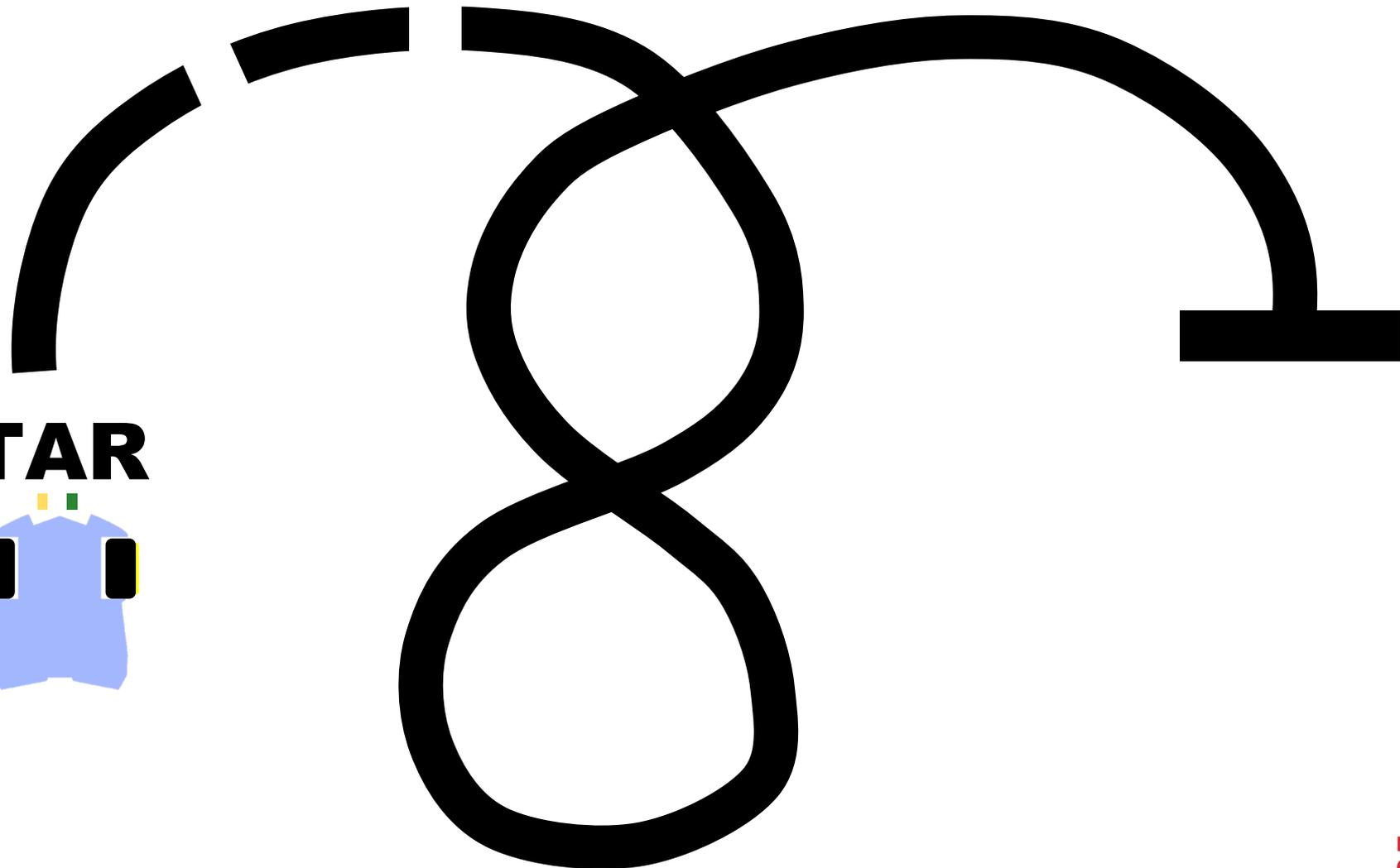


## 延伸挑戰(國中)：

從起點出發，循跡過斷線與交叉。

**STAR**

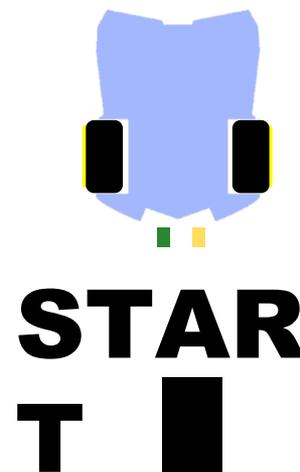
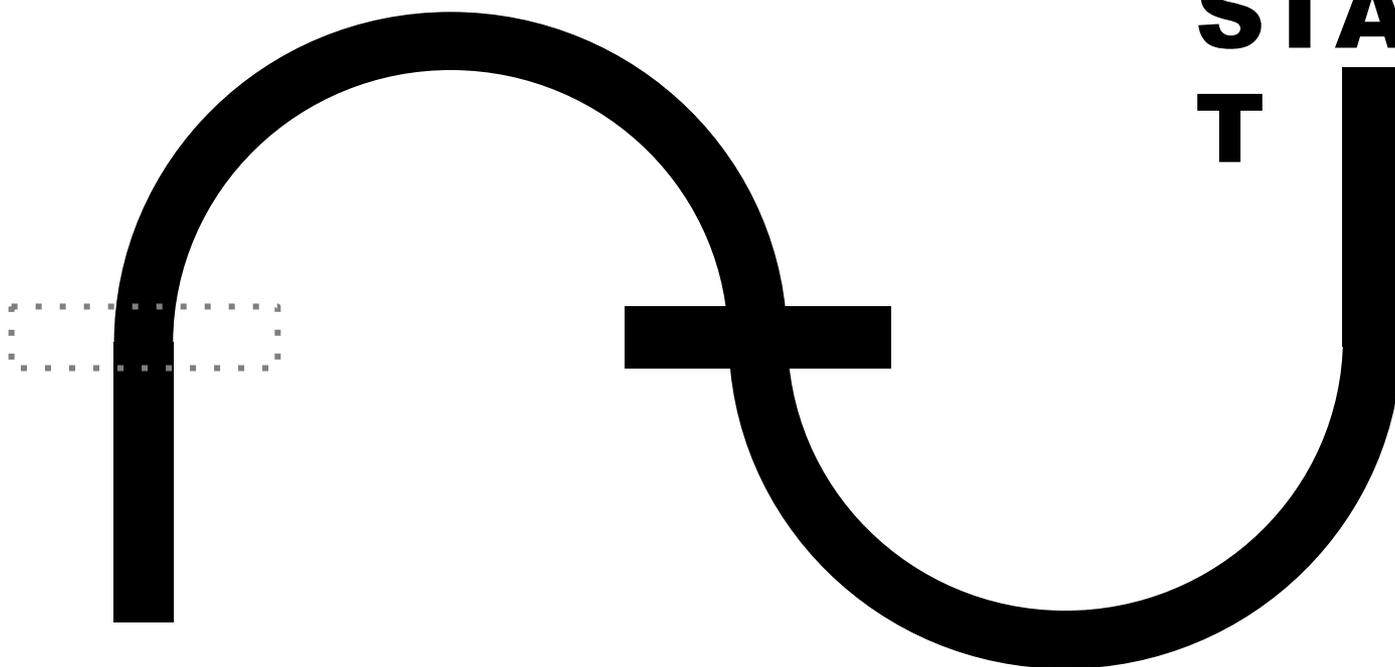
**T**



## 活動6：循跡到指定位置

練習：

在S型黑線上做循跡，必須跨越中段的橫線，並停在指定位置(但沒有橫線標示)。



## 活動6：循跡到指定位置

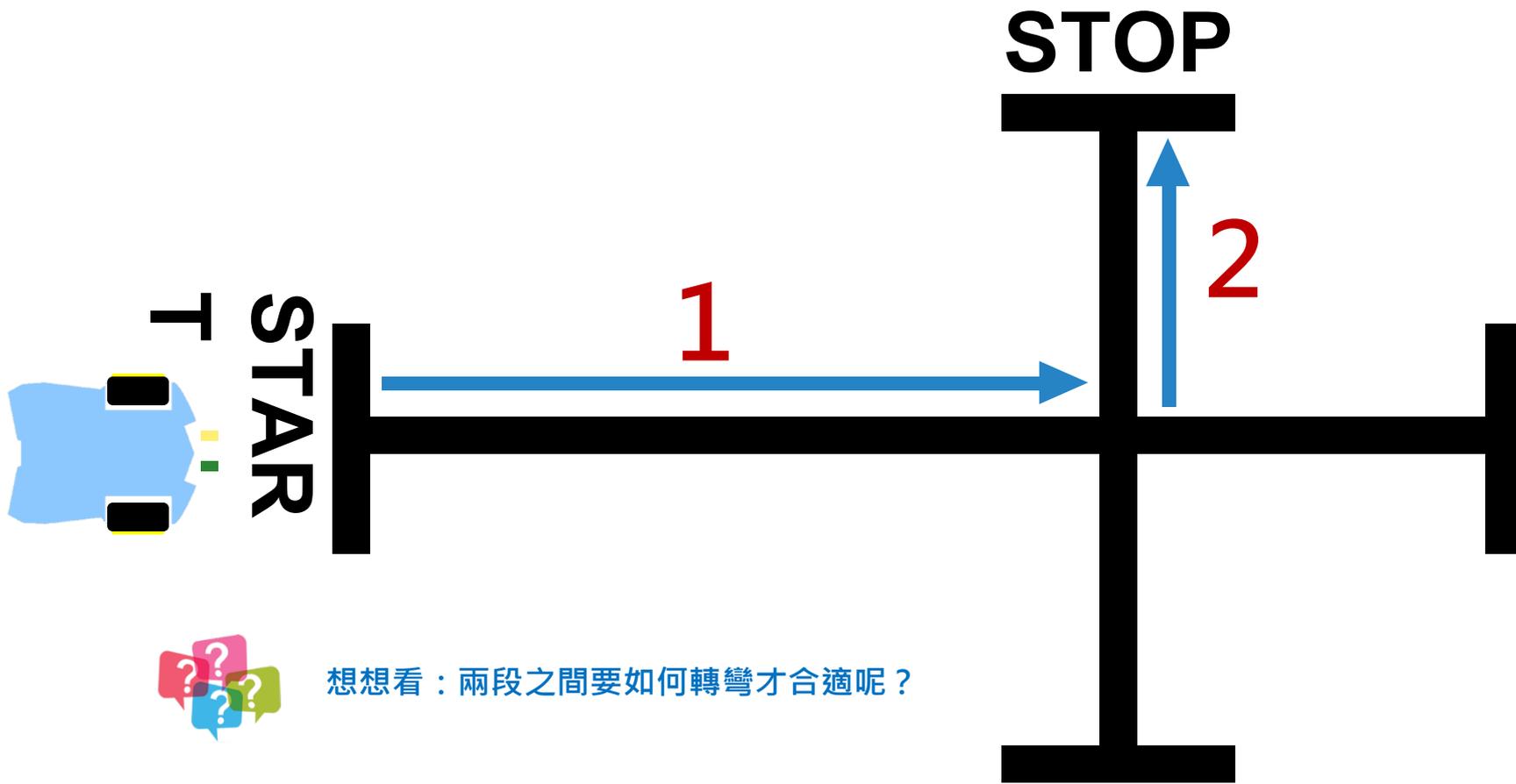
設定程序:

```
設定程序:  
重複 直到  
  從 A0 讀取紅外線光感測器的數值 < 500 且 從 A1 讀取紅外線光感測器的數值 < 500  
執行  
  如果 從 A0 讀取紅外線光感測器的數值 ≥ 500 且 從 A1 讀取紅外線光感測器的數值 ≥ 500  
    執行 forward  
  如果 從 A0 讀取紅外線光感測器的數值 < 500 且 從 A1 讀取紅外線光感測器的數值 ≥ 500  
    執行 left_turn_0  
  如果 從 A0 讀取紅外線光感測器的數值 ≥ 500 且 從 A1 讀取紅外線光感測器的數值 < 500  
    執行 right_turn_0  
forward  
等待 20 毫秒  
重複執行 10 秒  
  如果 從 A0 讀取紅外線光感測器的數值 ≥ 500 且 從 A1 讀取紅外線光感測器的數值 ≥ 500  
    執行 forward  
  如果 從 A0 讀取紅外線光感測器的數值 < 500 且 從 A1 讀取紅外線光感測器的數值 ≥ 500  
    執行 left_turn_0  
  如果 從 A0 讀取紅外線光感測器的數值 ≥ 500 且 從 A1 讀取紅外線光感測器的數值 < 500  
    執行 right_turn_0  
stop  
迴圈程序:
```

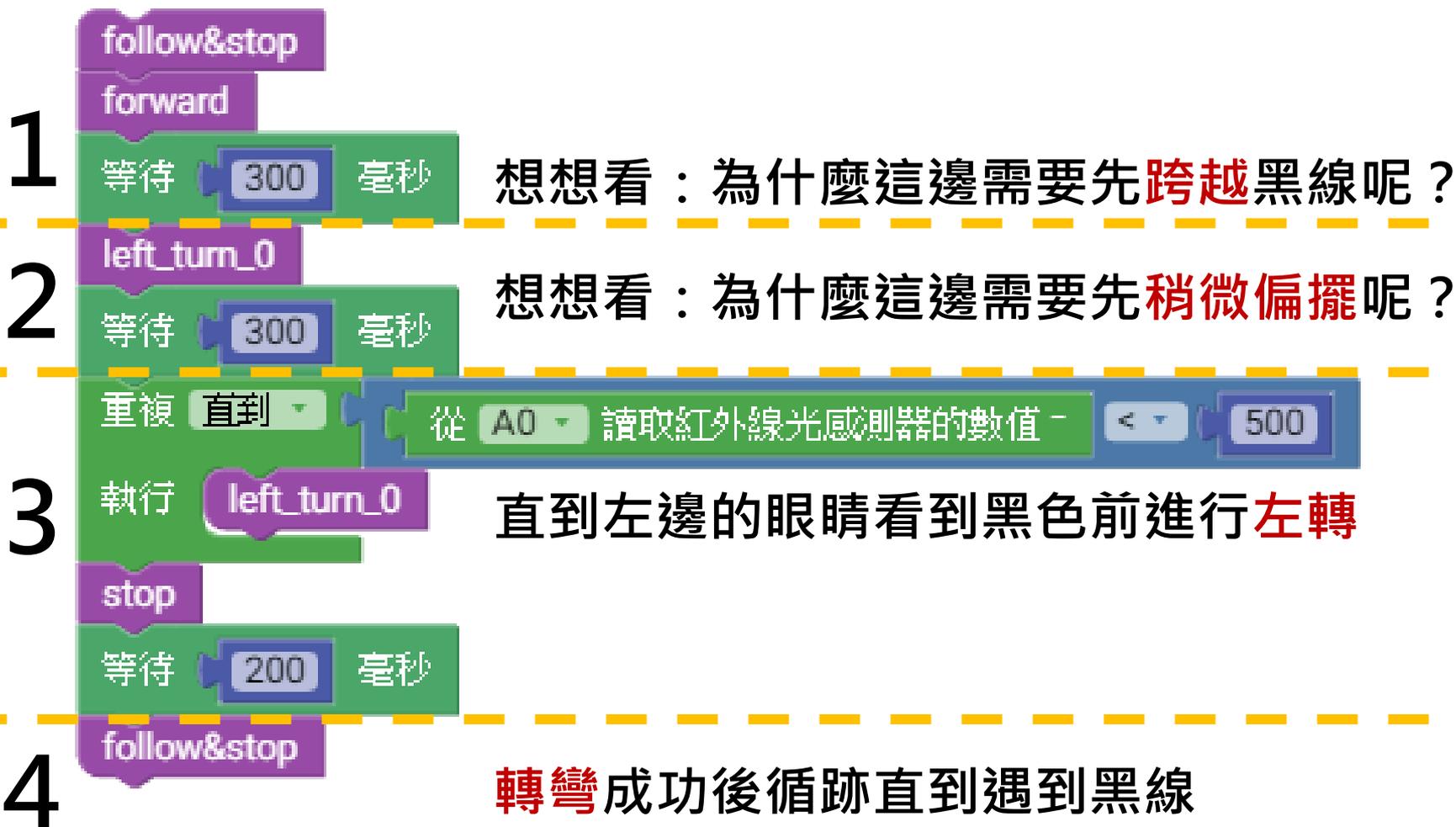
計算需要多久才能走到指定位置?

## 活動7：十字轉彎90度 — 左轉到叉路

循跡到十字處，90度左轉後繼續循跡，最後停在橫線處。

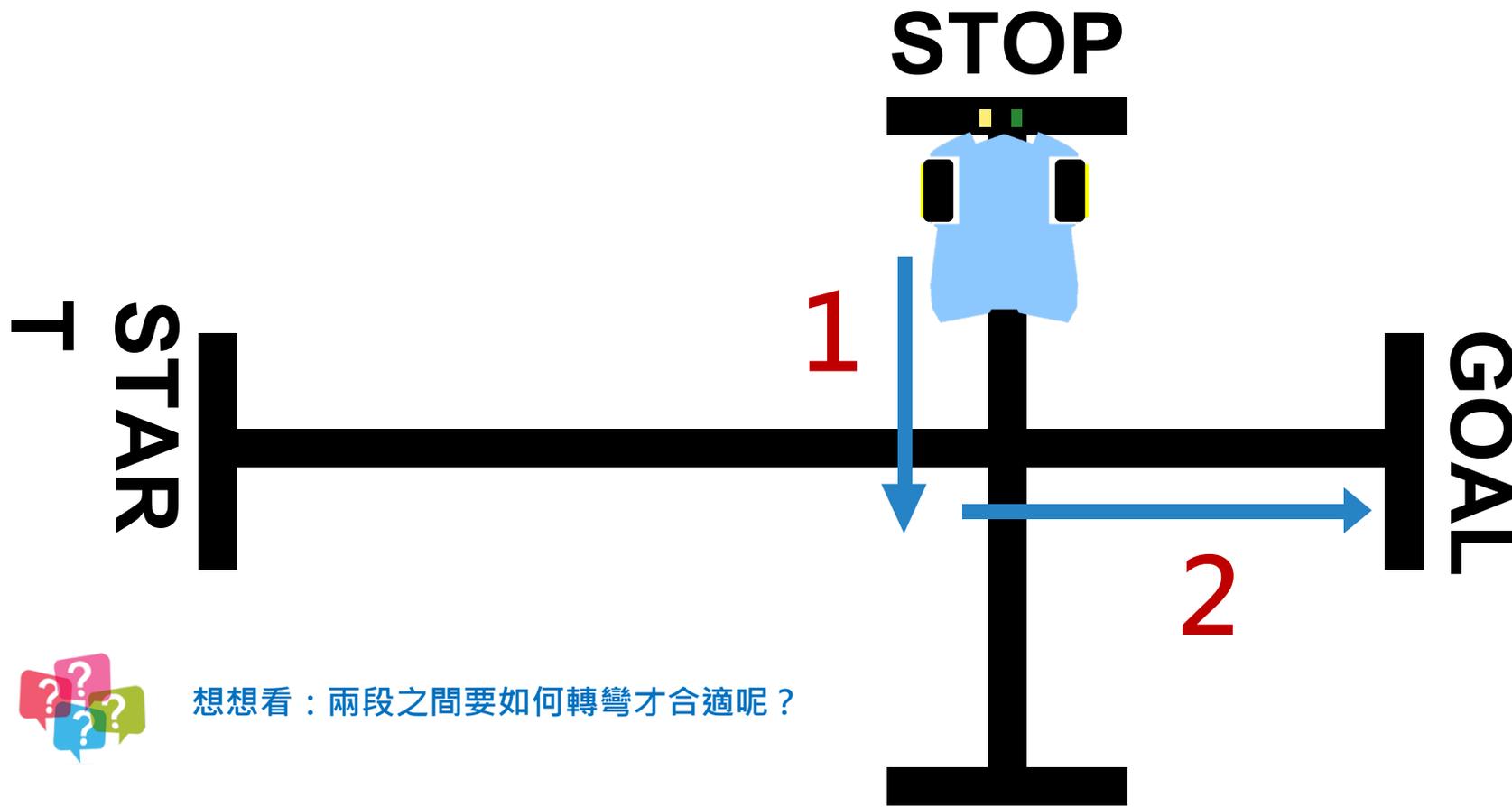


## 活動7：十字轉彎90度 — 左轉到叉路



## 活動7：十字轉彎90度 — 回到原路

從橫線處退回，右轉回原路線。



## 活動7：十字轉彎90度 — 回到原路

1 backward  
等待 300 毫秒 想想看：為什麼這邊需要先後退呢？

2 right\_turn\_0  
等待 300 毫秒 想想看：為什麼這邊需要先稍微偏擺呢？

3 重複 直到 從 A1 讀取紅外線光感測器的數值 < 500  
執行 right\_turn\_0 直到左邊的眼睛看到黑色前進行右轉  
stop

4 等待 200 毫秒  
轉彎成功後循跡直到遇到黑線

# 活動7：十字轉彎90度－綜合

1

follow&stop

循跡到十字

forward

2

等待 300 毫秒

左轉到叉路

X-Left

follow&stop

3

等待 5000 毫秒

循跡到橫線

backward

4

等待 300 毫秒

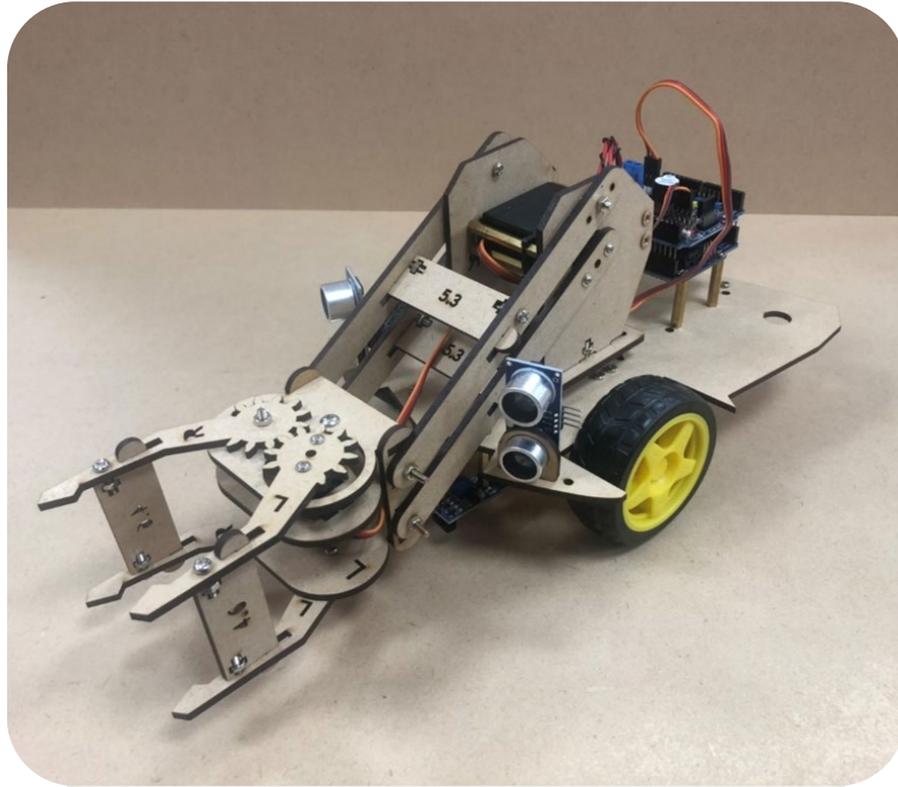
回到原路

X-Right

follow&stop

循跡至下一關

# START! 智慧小車



-單元6 超音波感測器操作-

# HC-SR04 超音波感測器介紹

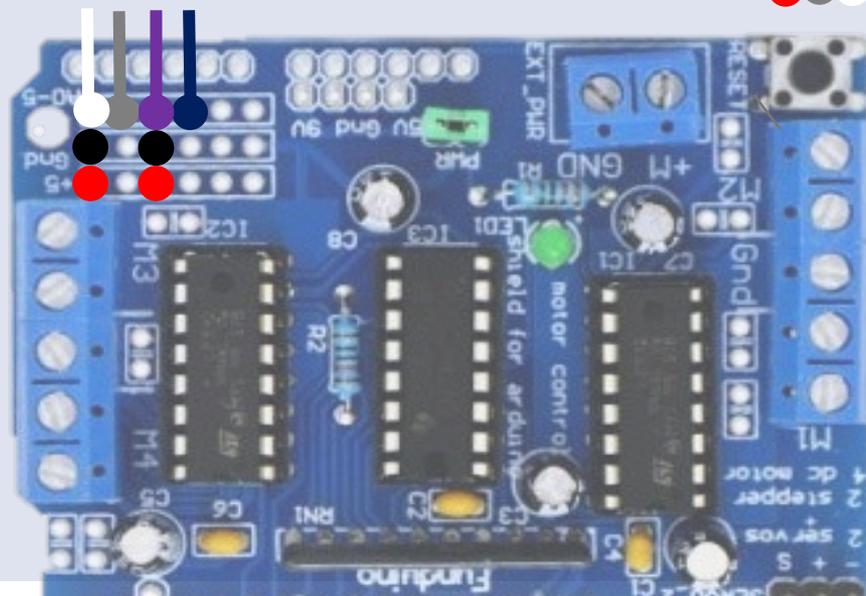
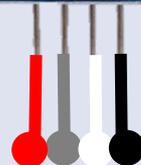
超音波感測器是由超音波發射器、接收器和控制電路所組成。會發射一連串 40 kHz 的聲波，從離它最近的物體接收回音。HC-SR04大約可以測距5~60cm。



左側超音波接法  
TRIG >> A2 藍  
ECHO >> A3 紫



右側超音波接法  
TRIG >> A4 灰  
ECHO >> A5 白



## HC-SR04 超音波感測器介紹

時間 **X** 速度 = 距離

聲音在空氣中傳播速度大約是每秒 **340** 公尺，傳播 **1** 公分所需時間 **29 microseconds** (百萬分之一秒)。

由於超音波從發射到返迴是兩段距離，因此在計算時必須將結果除以 **2** 才是正確的物體距離。

大自然中有很多動物皆是使用這樣的方式來偵測環境。



# 超音波測距

同樣使用IDE右上方的序列埠監控視窗，在電腦上觀察小車傳回的測距數值

設定程序:

設定 serial 的序列通訊速度為 9600 位元/秒

迴圈程序:

- serial 送出 “ left distance = ” 換行
- 超音波(HC-SR04)腳位設定 換行
  - Trig腳位 A2
  - ECHO腳位 A3
  - 超聲波回傳距離 公分
- serial 送出 “ , ” 換行
- serial 送出 “ right distance = ” 換行
- 超音波(HC-SR04)腳位設定 換行
  - Trig腳位 A4
  - ECHO腳位 A5
  - 超聲波回傳距離 公分
- 等待 1000 毫秒

# 讀取超音波感測器數值

程式上傳完畢後，點選序列埠監控視窗，即可觀察小車傳回的數值。

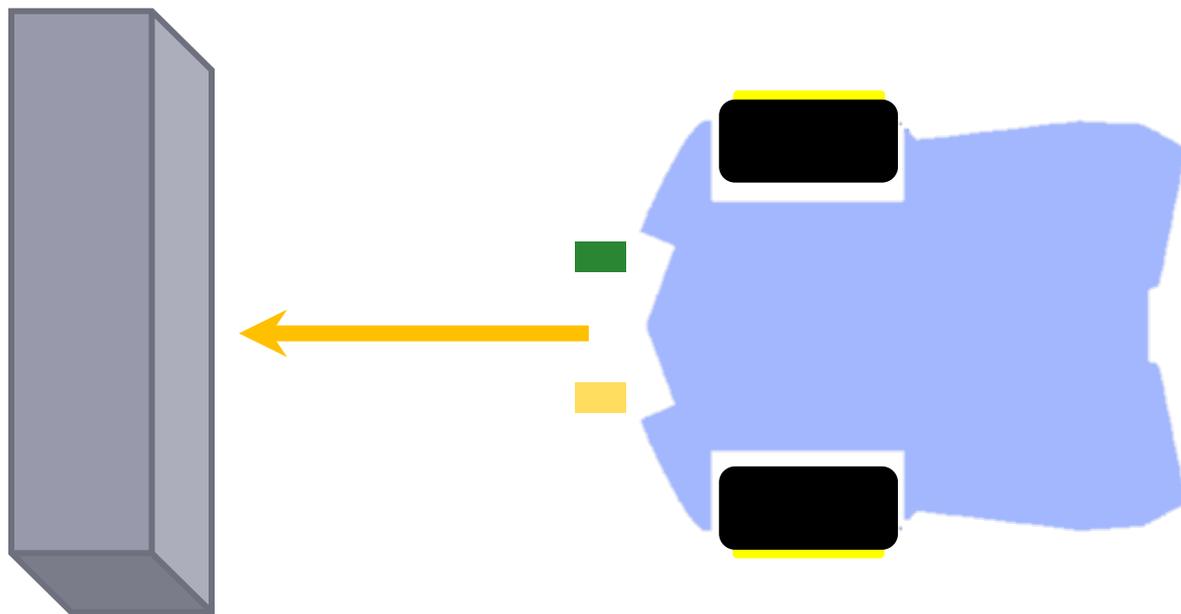
The screenshot shows the Ardublockly v20200526\_0006 interface. The top toolbar includes icons for 'Serial Monitor' (序列埠監控), 'Examples' (範例), 'Code Switch' (代碼切換), and 'Example Programs with Arduino Portable' (範例程式與arduino可攜版). A yellow arrow points to the 'Serial Monitor' icon, with the text '按此監看' (Click here to monitor) next to it. The main workspace contains a Scratch-style block-based program:

```
設定程序:  
設定 serial 的序列通訊速度為 9600 位元/秒  
迴圈程序:  
serial 送出 " left distance = "  
serial 送出 超音波 (HC-SR04) 腳位設定  
    Trig 腳位 A2  
    ECHO 腳位 A3  
    超音波回傳距離 公分  
serial 送出 " "  
serial 送出 " right distance = "  
serial 送出 超音波 (HC-SR04) 腳位設定  
    Trig 腳位 A4  
    ECHO 腳位 A5  
    超音波回傳距離 公分  
等待 1000 毫秒
```

At the bottom, there is a status bar with an up/down arrow and the text 'Arduino IDE 輸出'.

## 偵測障礙物

請參考遇到黑線停下來程式  
寫出偵測到前方10cm有障礙物  
會停下的程式



# 偵測障礙物

設定程序:

迴圈程序:

如果

超音波(HC-SR04)腳位設定 > 10 且 超音波(HC-SR04)腳位設定 > 10

Trig腳位 A2  ECHO腳位 A3  超音波回傳距離 公分

Trig腳位 A4  ECHO腳位 A5  超音波回傳距離 公分

執行 forward

如果

超音波(HC-SR04)腳位設定 ≤ 10 或 超音波(HC-SR04)腳位設定 ≤ 10

Trig腳位 A2  ECHO腳位 A3  超音波回傳距離 公分

Trig腳位 A4  ECHO腳位 A5  超音波回傳距離 公分

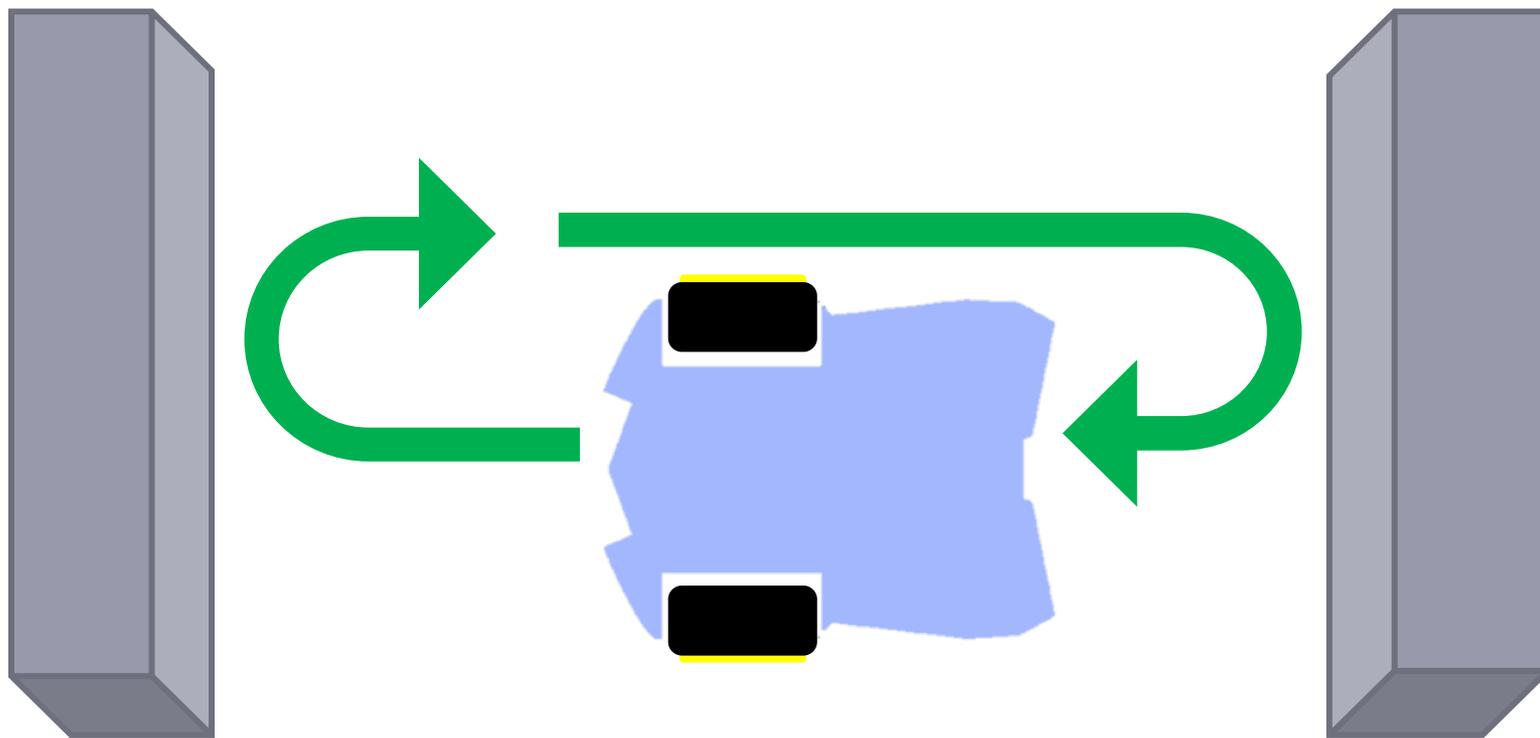
執行 stop



想想看：為什麼這裡的第二個如果要用「或」？

## 練習：

在兩面障礙物之間做折返跑。



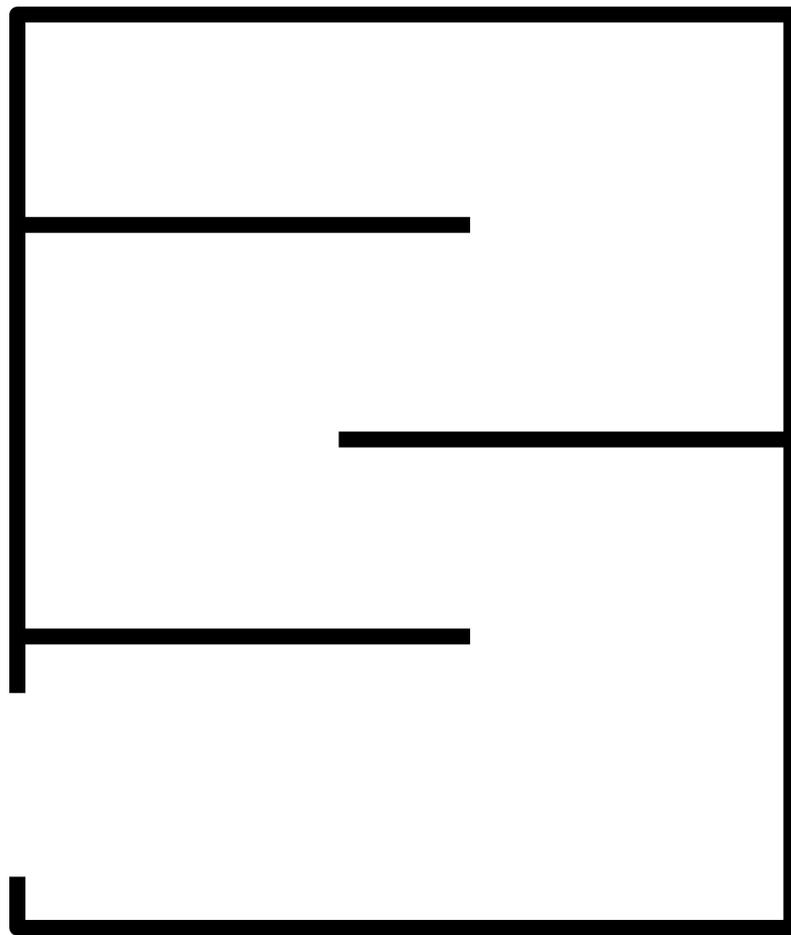
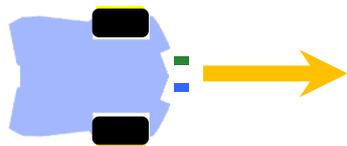
## 活動8：碰撞迴避

- 口訣**哪邊沒牆往哪邊轉**
- 遇到障礙先後退再轉彎  
(預留迴轉空間)



想想看：建議轉彎90度，  
為什麼？

- 受困在牆角時如何處理？



## 活動8： 碰撞迴避

- 避障距離應該設為多少？
- 避障距離越遠越好嗎？
- 考慮走道寬度

The image displays three Scratch code blocks for obstacle avoidance, each starting with an 'if' block and two 'ultrasonic sensor' blocks. The sensor blocks are configured with 'Trig' pins A2 and A4, and 'ECHO' pins A3 and A5, with a range of 10. The first block has an empty 'if' block and a 'forward' block. The second block has an 'if' block with a 'left' condition and a sequence of 'backward', 'left\_turn\_0', and 'backward' blocks. The third block has an 'if' block with a 'right' condition and a sequence of 'backward', 'right\_turn\_0', and 'backward' blocks.

設定位序  
迴避位序

如果

超音波(HC-SR04)腳位設定  
Trig腳位 A2  
ECHO腳位 A3  
超聲波回傳距離 10

超音波(HC-SR04)腳位設定  
Trig腳位 A4  
ECHO腳位 A5  
超聲波回傳距離 10

執行 forward

兩邊空，前進。

如果

超音波(HC-SR04)腳位設定  
Trig腳位 A2  
ECHO腳位 A3  
超聲波回傳距離 10

超音波(HC-SR04)腳位設定  
Trig腳位 A4  
ECHO腳位 A5  
超聲波回傳距離 10

執行 backward

等待 300 毫秒

left\_turn\_0

等待 300 毫秒

左邊空，先後退再左轉。

如果

超音波(HC-SR04)腳位設定  
Trig腳位 A2  
ECHO腳位 A3  
超聲波回傳距離 10

超音波(HC-SR04)腳位設定  
Trig腳位 A4  
ECHO腳位 A5  
超聲波回傳距離 10

執行 backward

等待 300 毫秒

right\_turn\_0

等待 300 毫秒

右邊空，先後退再右轉

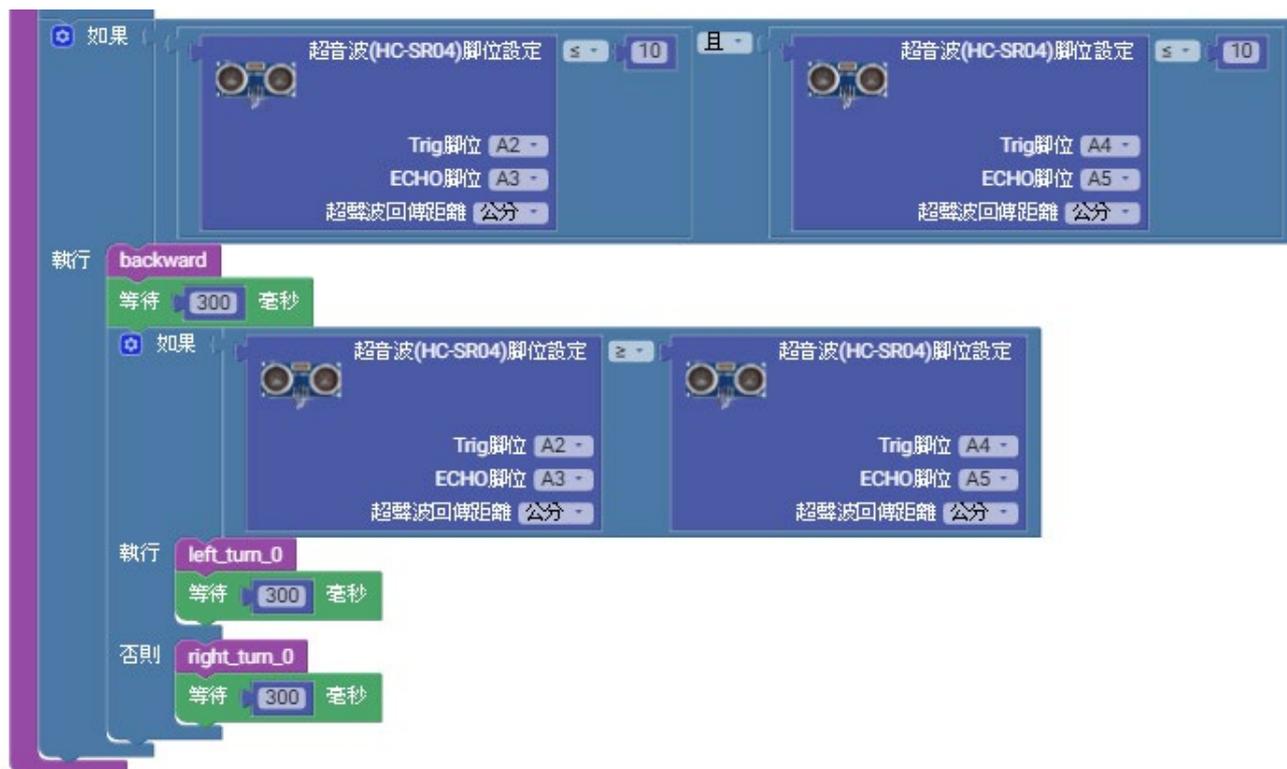


想想看：兩邊同時偵測到障礙物時....

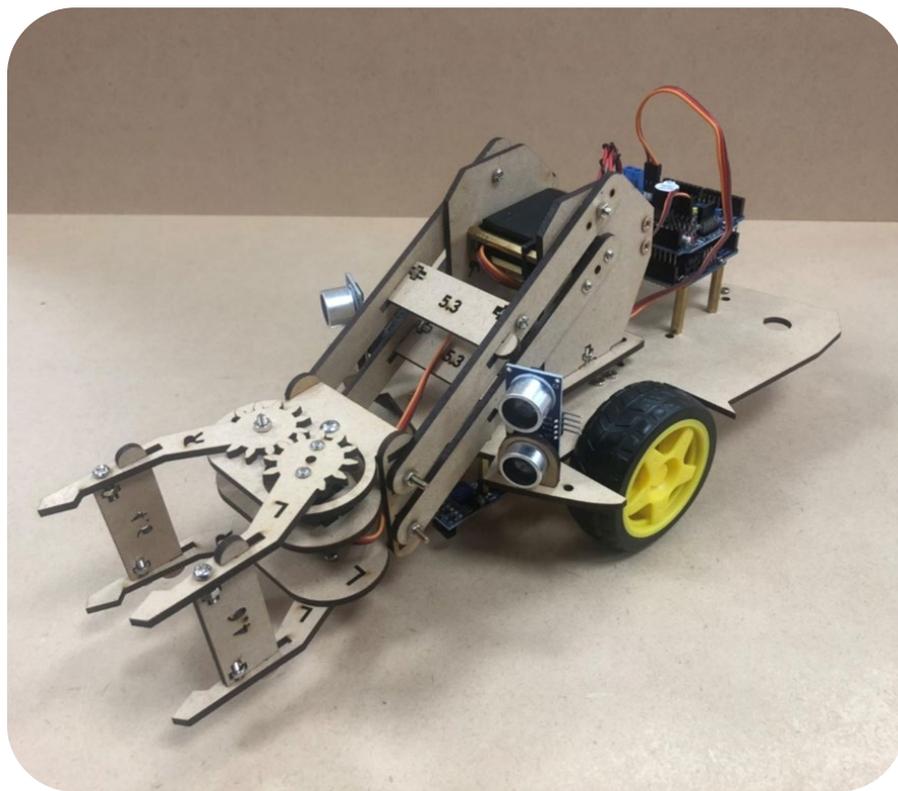
## 活動8： 碰撞迴避

兩邊同時偵測到  
障礙時...

判斷兩側牆壁遠近  
往距離較遠者轉彎



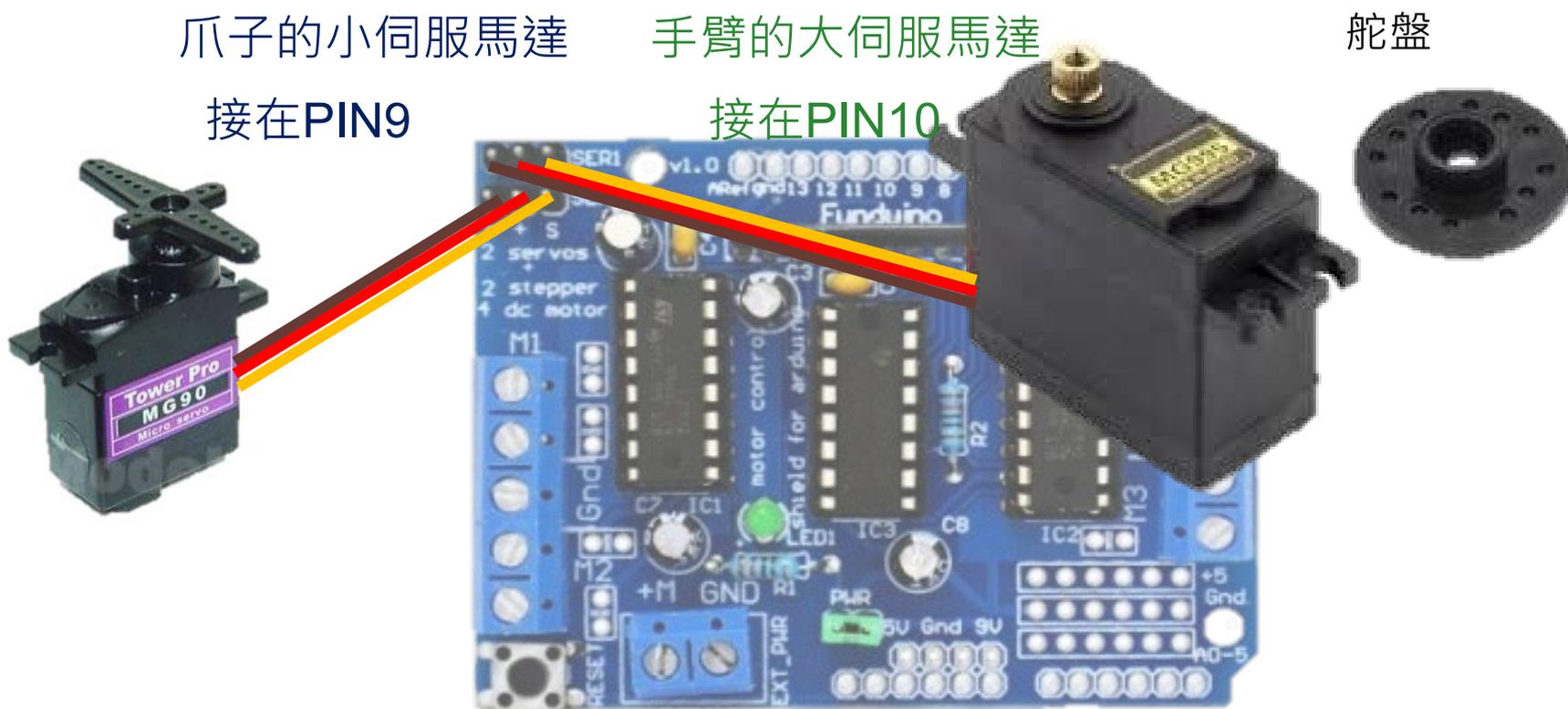
# START! 智慧小車



-單元7 伺服馬達操作-

# 伺服馬達

- 精準地控制0度~180度
- L293D擴展板上有引出SERVO接腳(使用PIN9及PIN10接腳), 可連接伺服馬達。



## 伺服馬達角度校正：

試著寫出右圖的程式，  
並依以下步驟進行校正。

- 設定伺服馬達角度為90度，
  - 1.手臂應抬升至水平
  - 2.爪子應張開至水平

若手臂角度不符預期

→拔除伺服馬達電源線

→將舵盤拔下

→調整至正確位置

→接上舵盤

```
設定程序:  
設定伺服馬達腳位 servo_9 , 腳位 9  
設定伺服馬達腳位 servo_10 , 腳位 10  
伺服馬達 servo_9 旋轉到 90 度  
等待 1000 毫秒  
伺服馬達 servo_10 旋轉到 90 度  
等待 1000 毫秒  
迴圈程序:
```

※記得預留轉動時間，  
一度需6毫秒以上

# 控制伺服馬達角度

試著調整伺服馬達角度  
找出手臂舉起、放下、  
爪子打開、收合時的  
角度為何？

提示：  
看著伺服馬達軸心時，  
角度增加 > 逆時針轉  
角度減少 > 順時針轉



馬達	PIN9 伺服馬達		PIN10 伺服馬達	
動作	打開爪子	收合爪子	舉起手臂	放下手臂
角度	90	0 ~ 30	90	120

爪子收合時記得考慮物體大小，不要夾到底。

## 活動9：伺服馬達練習

口訣: **Down&Open**、**Close&Up**

預設角度

>放下手臂

>打開爪子

>收起爪子

>抬起手臂



設定程序:

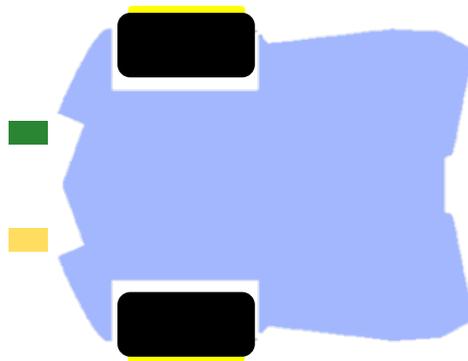
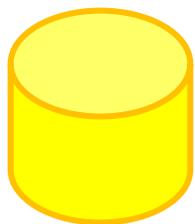
- 設定伺服馬達腳位 servo\_9 , 腳位 9
- 設定伺服馬達腳位 servo\_10 , 腳位 10
- 伺服馬達 servo\_9 旋轉到 90 度
- 等待 1000 毫秒
- 伺服馬達 servo\_10 旋轉到 90 度
- 等待 1000 毫秒

迴圈程序:

- 伺服馬達 servo\_10 旋轉到 120 度
- 等待 500 毫秒
- 伺服馬達 servo\_9 旋轉到 90 度
- 等待 500 毫秒
- 伺服馬達 servo\_9 旋轉到 30 度
- 等待 500 毫秒
- 伺服馬達 servo\_10 旋轉到 90 度
- 等待 500 毫秒

## 延伸練習(國小)：

在白色區域直走，遇到橫線停下來，並將目標物舉起再放下。



## 延伸練習(國小)：

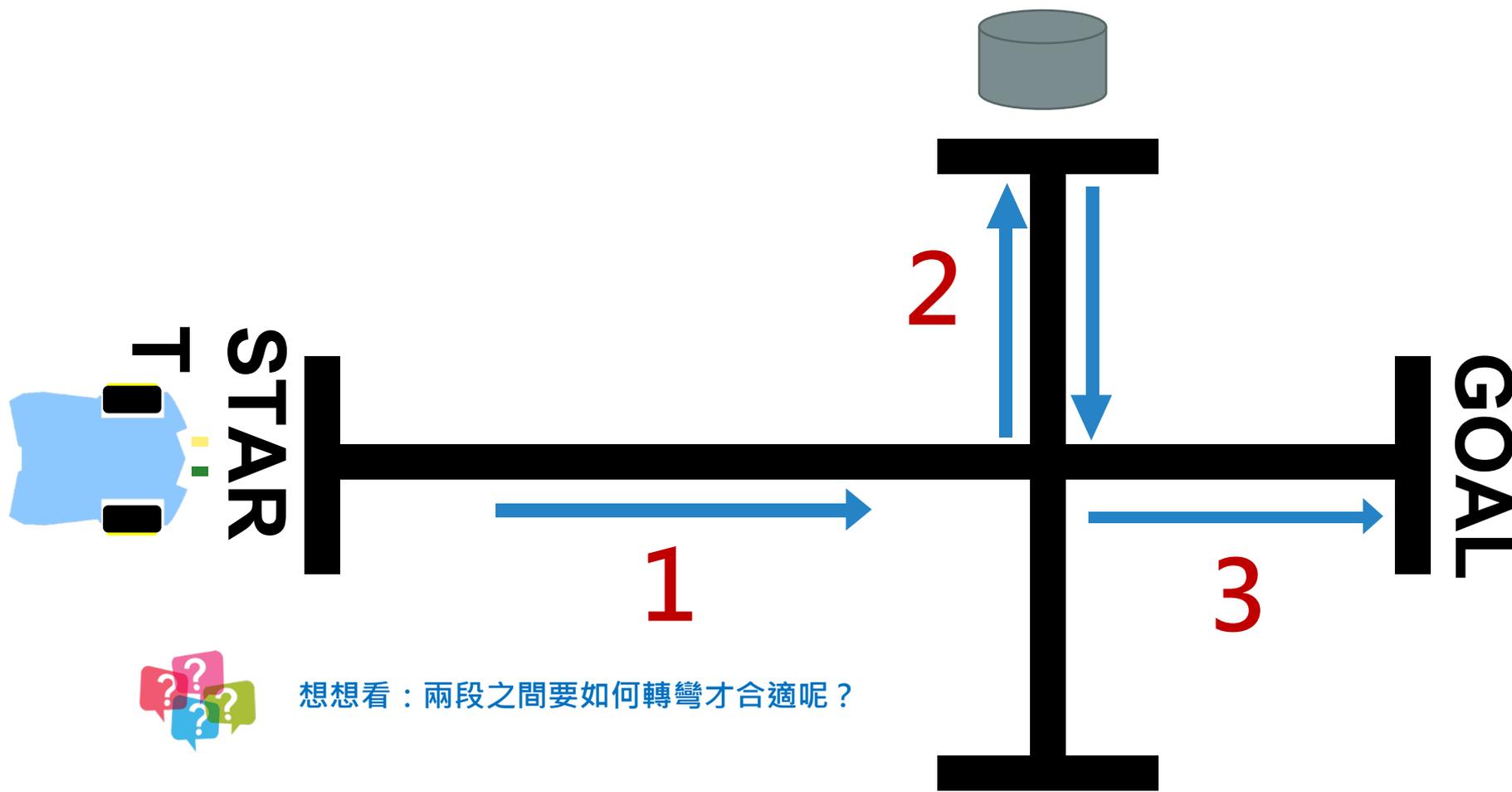
在白色區域直走，遇到橫線停下來，並將目標物舉起再放下。

設定程序：

一開始就讓爪子打開並放在最底下的角度，在取物時可以避免爪子在運作過程中被目標物干擾。

迴圈程序：

# 十字轉彎+取物



想想看：兩段之間要如何轉彎才合適呢？

# 十字轉彎+取物



放下手臂、循跡到十字

左轉到叉路

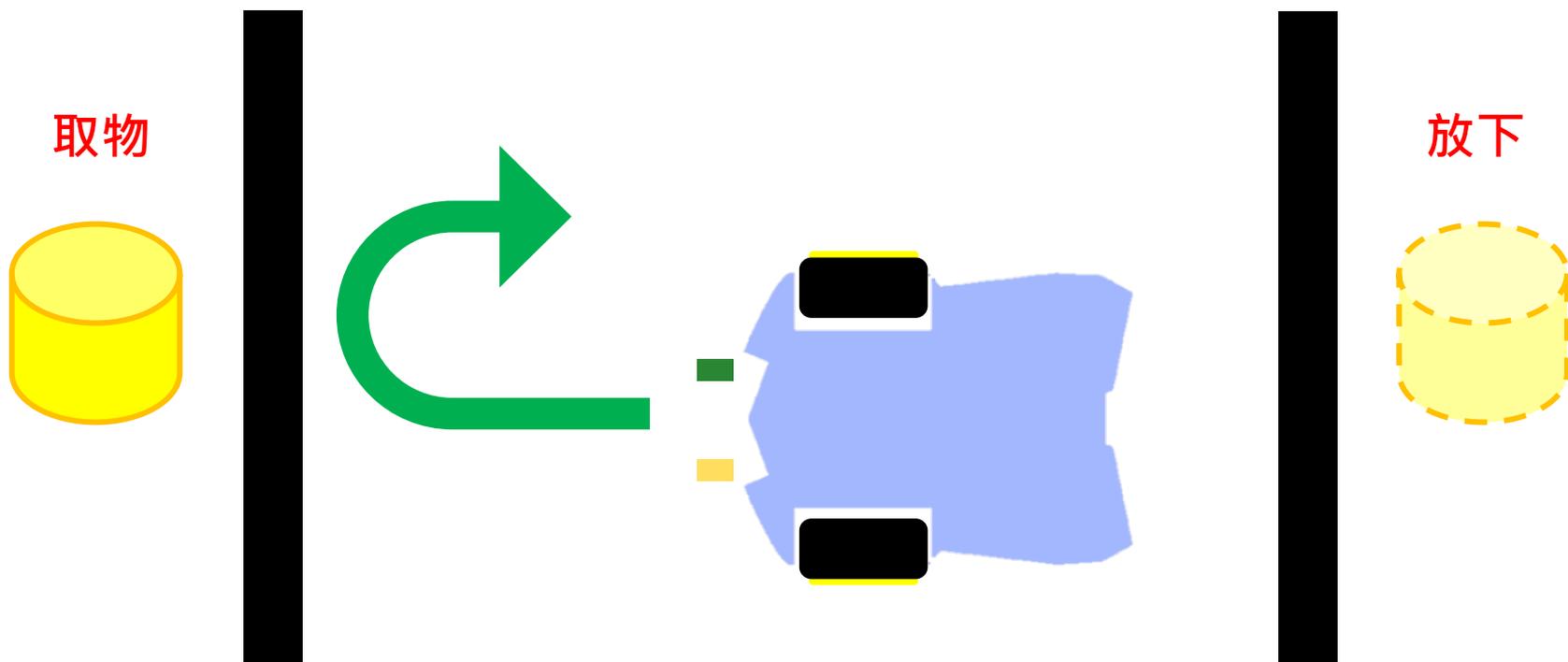
循跡到橫線

抓起道具

回到原路、循跡至下一關

## 練習(國小/國中)：

在白色區域直走，遇到橫線停下來，並將目標物舉起，迴轉後繼續前進，遇到黑線後停下來，並將目標物放下。(提示：兩條黑線間折返跑)



## 練習(國小/國中)：

在白色區域直走，遇到橫線停下來，並將目標物舉起，迴轉後繼續前進，遇到黑線後停下來，並將目標物放下。(提示：兩條黑線間折返跑)

設定程序:

可將爪子的一系列動作寫為副程式

重複 直到

從 A0 讀取紅外線光感測器的數值 - 500 且 從 A1 讀取紅外線光感測器的數值 - 500

執行 forward

stop

等待 500 毫秒

close&up

等待 1000 毫秒

right\_turn\_0

原地旋轉90度

等待 800 毫秒

重複 直到

從 A0 讀取紅外線光感測器的數值 - 500 且 從 A1 讀取紅外線光感測器的數值 - 500

執行 forward

stop

等待 500 毫秒

down&open

到 down&open

伺服馬達 servo\_10 旋轉到 120

等待 1000 毫秒

伺服馬達 servo\_9 旋轉到 120

等待 1000 毫秒

到 close&up

伺服馬達 servo\_9 旋轉到 30

等待 1000 毫秒

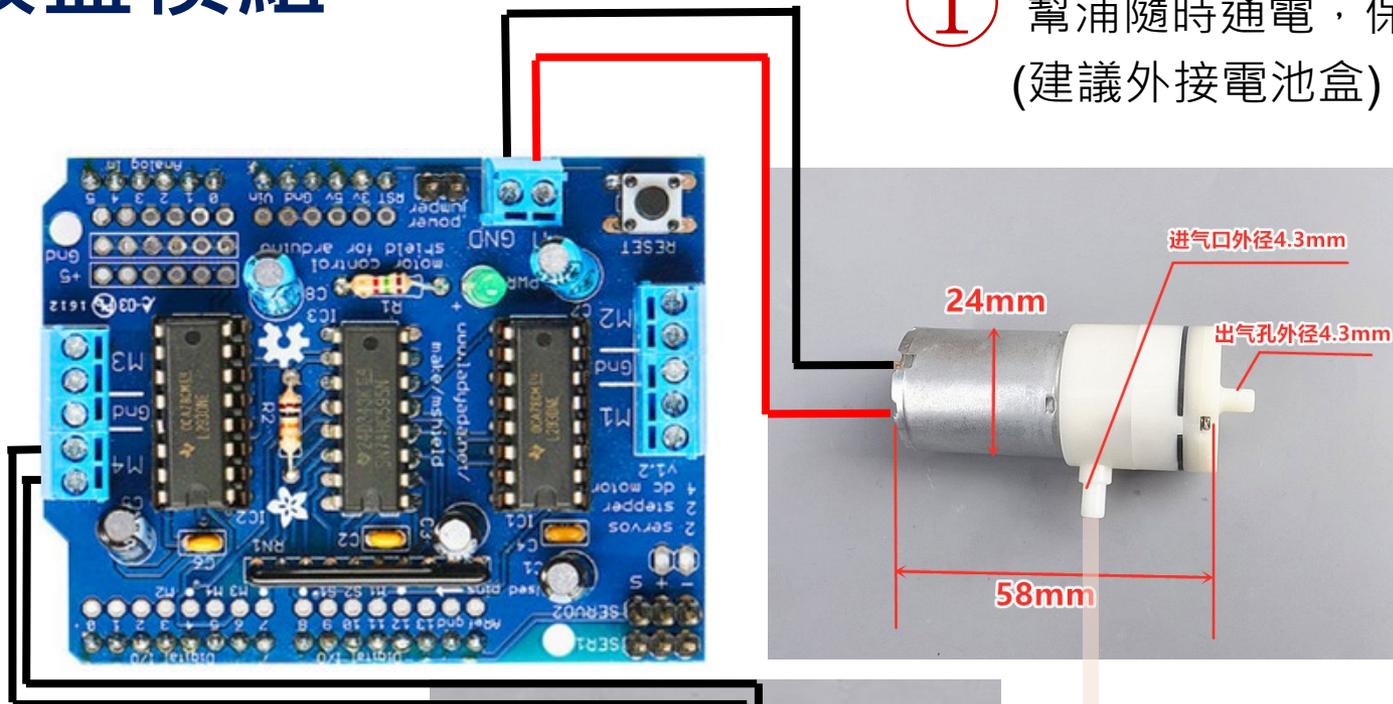
伺服馬達 servo\_10 旋轉到 90

等待 1000 毫秒

迴圈程序:

# 吸盤模組

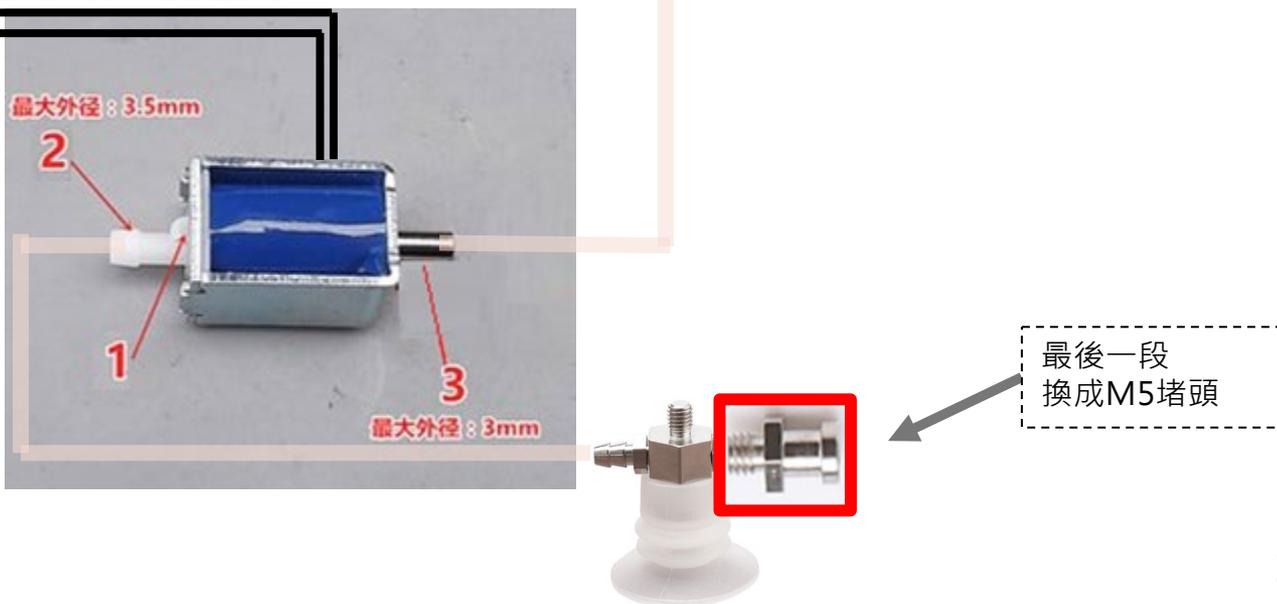
① 幫浦隨時通電，保持抽氣  
(建議外接電池盒)



② 利用電磁閥控制  
東西放下

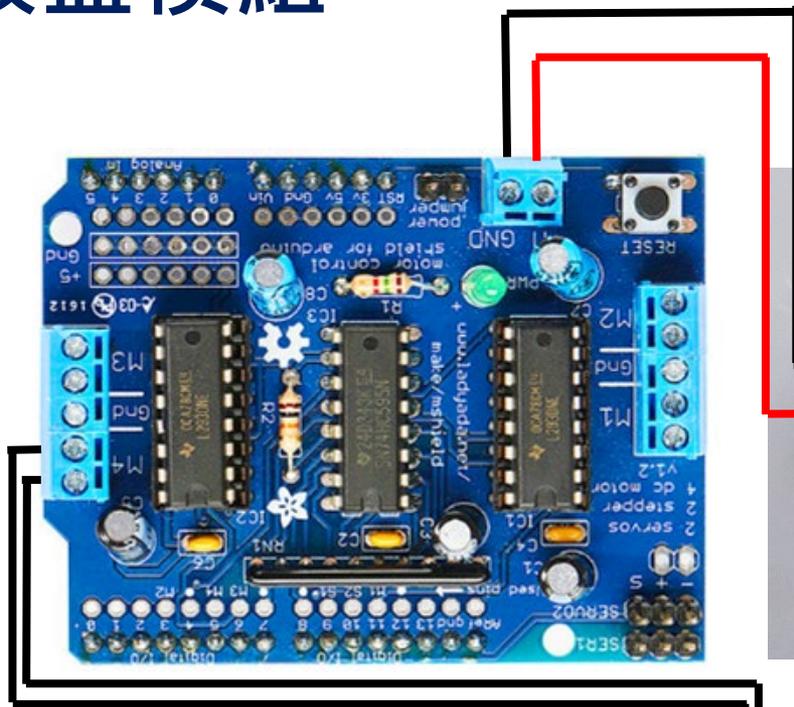
③ 電磁閥斷電時  
1通3，2被封堵  
管線接抽氣馬達

電磁閥通電時  
1通2，3被封堵  
管線與外界連通



# 吸盤模組

① 幫浦隨時通電，保持抽氣  
(建議外接電池盒)



② 利用電磁閥控制  
東西放下



③ 電磁閥斷電時  
1通3，2被封堵  
管線接抽氣馬達

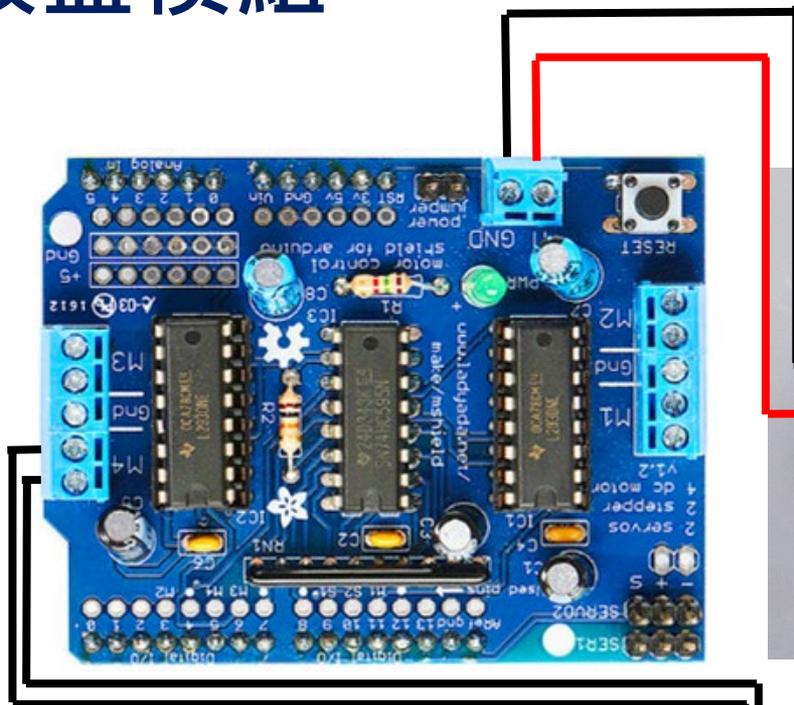
電磁閥通電時  
1通2，3被封堵  
管線與外界連通



最後一段  
換成M5堵頭

# 吸盤模組

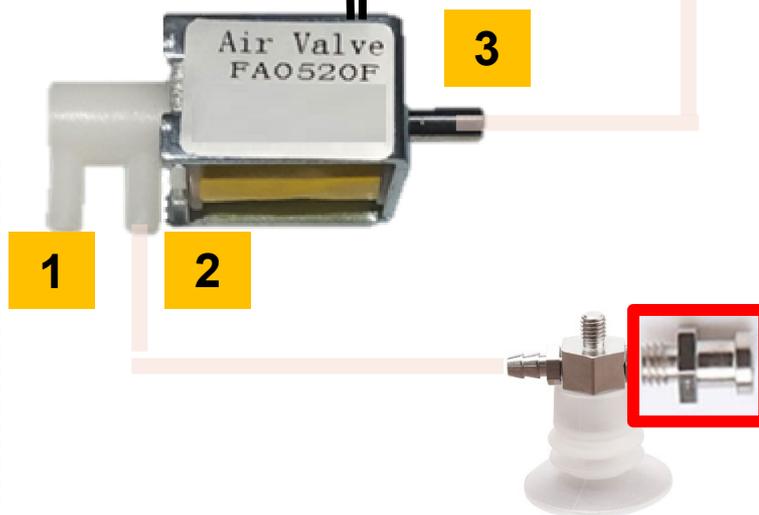
① 幫浦隨時通電，保持抽氣  
(建議外接電池盒)



② 利用電磁閥控制  
東西放下

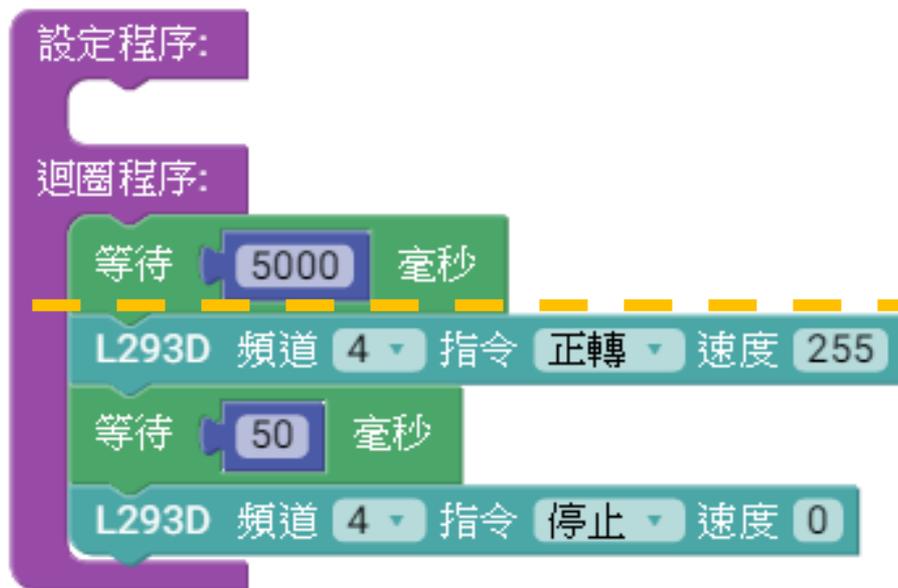
③ 電磁閥斷電時  
1通3，2被封堵  
管線接抽氣馬達

電磁閥通電時  
1通2，3被封堵  
管線與外界連通



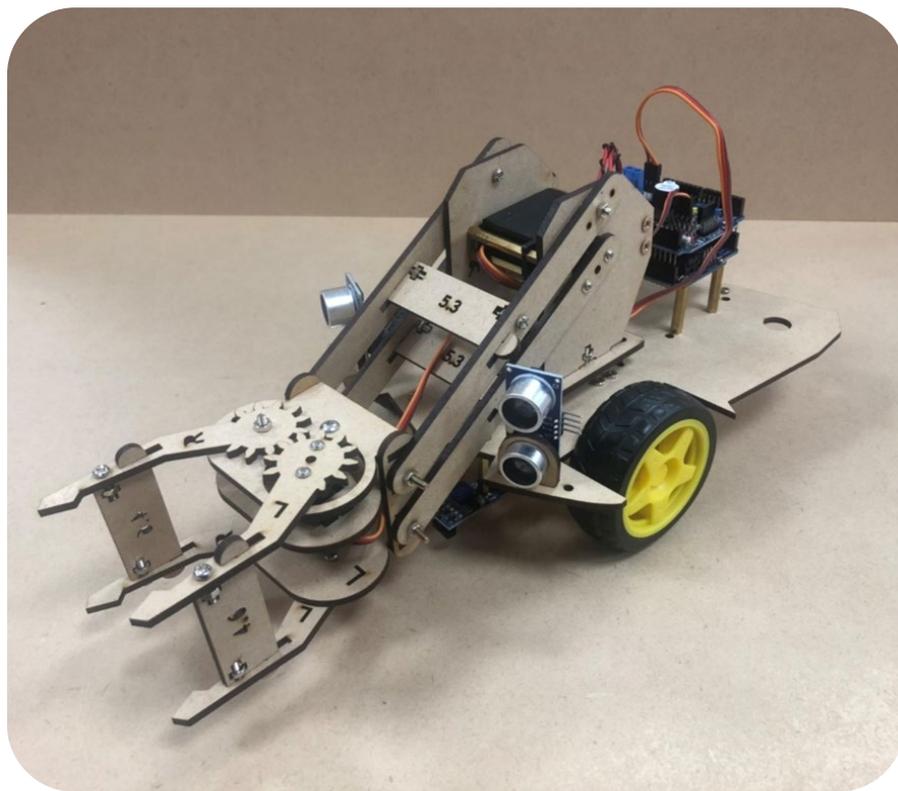
最後一段  
換成M5堵頭

## 測試抽氣馬達模組



無指令時，幫浦保持抽氣  
電磁閥通電(阻斷抽氣)  
調整秒數至物品順利落下  
電磁閥斷電(回復抽氣)

# START! 智慧小車



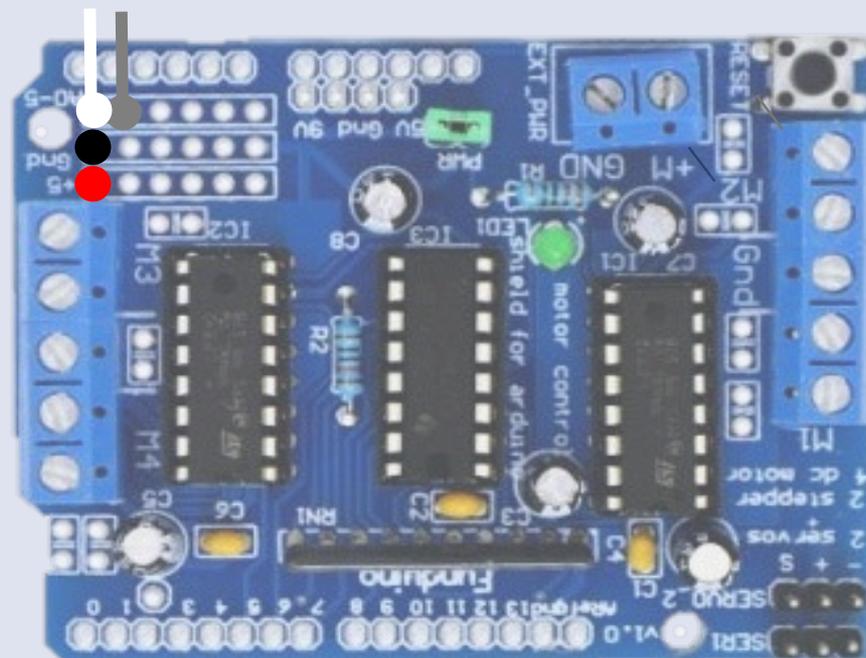
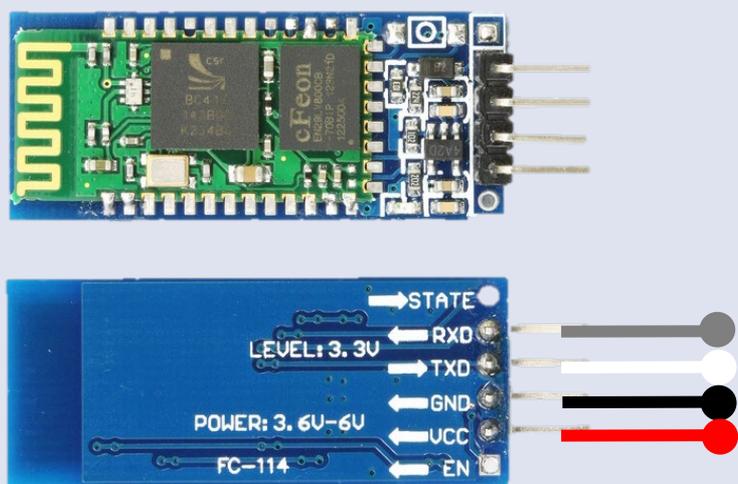
-單元8 手機藍牙操作-

# HC-06 藍牙模組介紹

HC-06 支援藍牙2.1，採用Serial Port通訊協定，與使用USB線直連電腦時相同。

※L293D供電為5V請注意藍牙接收範圍3.6V – 6V是否相容。

※由於頻道有限，多人連線時容易干擾，請盡量將沒有連線的手機藍牙關閉。



# STEP1.

## 將藍牙遙控車範例程式上傳至Arduino

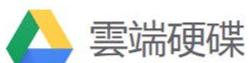
The screenshot shows the Ardublockly web application interface. At the top, there is a navigation bar with the text "Ardublockly: #鍵盤控制車" and a "範例" (Examples) button. A dropdown menu is open, displaying a list of example programs:

- 00.藍芽遙控車.xml (highlighted with a red circle)
- 00.馬達測試.xml
- 01.前後左右停.xml
- 02.不同的轉彎方式.xml
- 03.紅外線\_測量黑線數值.xml
- 04.遇到黑線停下來.xml

The background shows a block-based programming environment with various components like "設定程序:", "藍芽模組 傳送", "接收", "序列通訊速度", "伺服馬達變數", "orig", "迴圈程序:", "如果", "執行", "賦值", "否則如", "執行", "left\_turn\_0", and "否則如果".

# STEP.2

至雲端硬碟中，點選「手機遙控」資料夾



智慧小車

名稱 ↑	擁有者	上次修改時間	檔案大小
 0.智慧小車講義	李文宏	2019年3月17日 李文宏	-
 01.Ardublockly 程式	李文宏	2019年3月17日 李文宏	-
 02.手機遙控	李文宏	2019年3月17日 李文宏	-
 03.範例程式	Si-Yu Chen	2019年3月17日 Si-Yu Chen	-
 04.雷射切割檔	李文宏	2019年5月5日 李文宏	-
 05.材料清單	李文宏	2019年6月15日 李文宏	-
 06.motoBlockly	Si-Yu Chen	2019年6月15日 Si-Yu Chen	-

## STEP.3

將「START智慧小車.apk」  
下載手機APP安裝



智慧小車 > 02.手機遙控

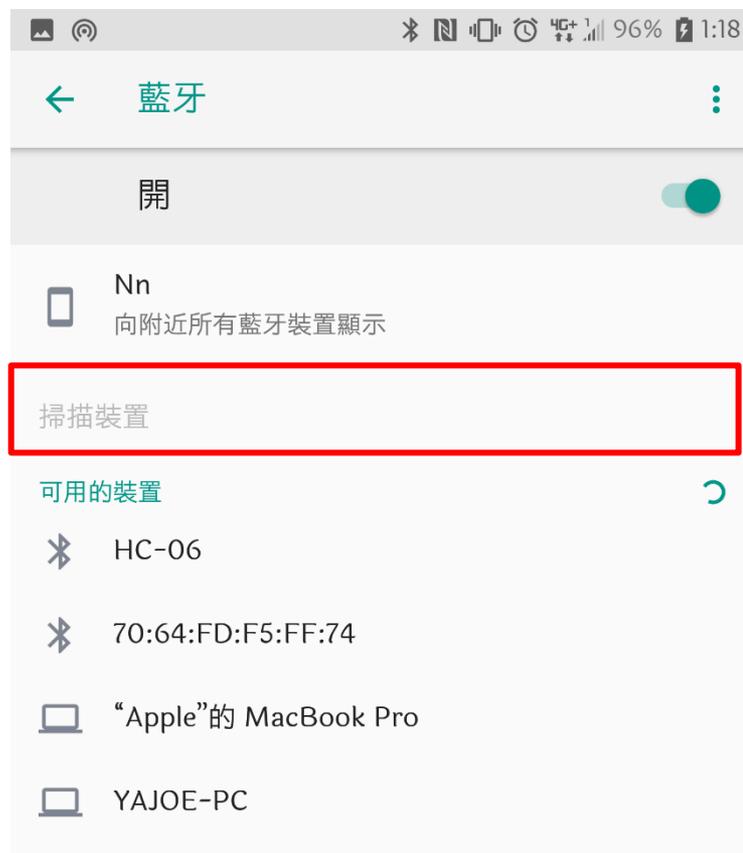
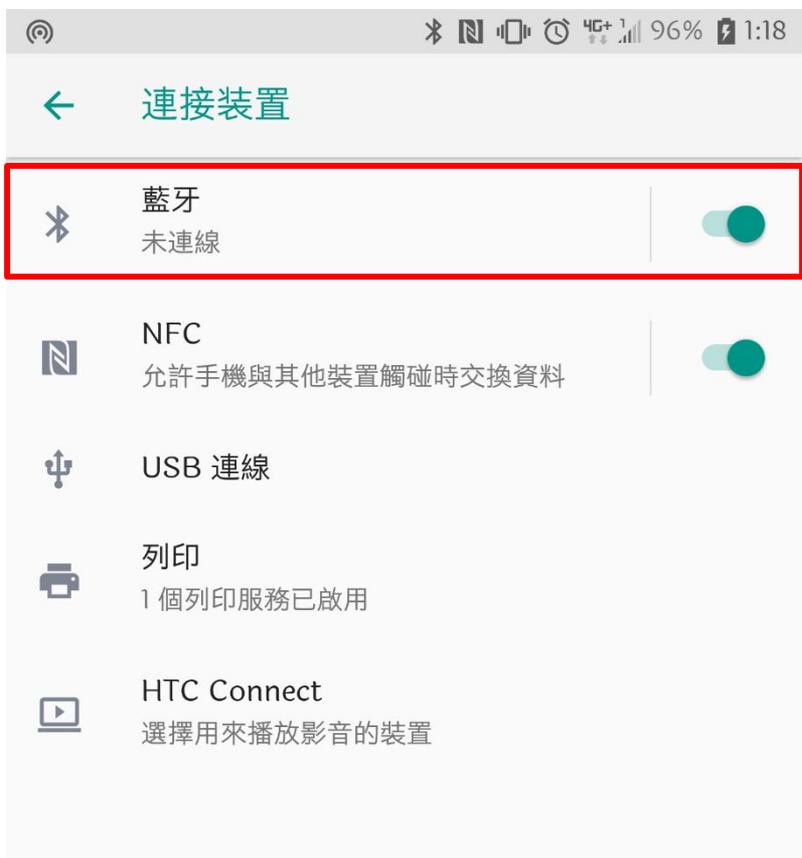
名稱 ↑	擁有者
 BTcontroll.ino 	李文宏
 START智慧小車手機遙控APP.a... 	Zhien Wang
 StartControl.aia 	Zhien Wang



【註】手機APP使用 AppInventor 編寫，  
可用 StartControl.aia 自行進行修改

## STEP.4

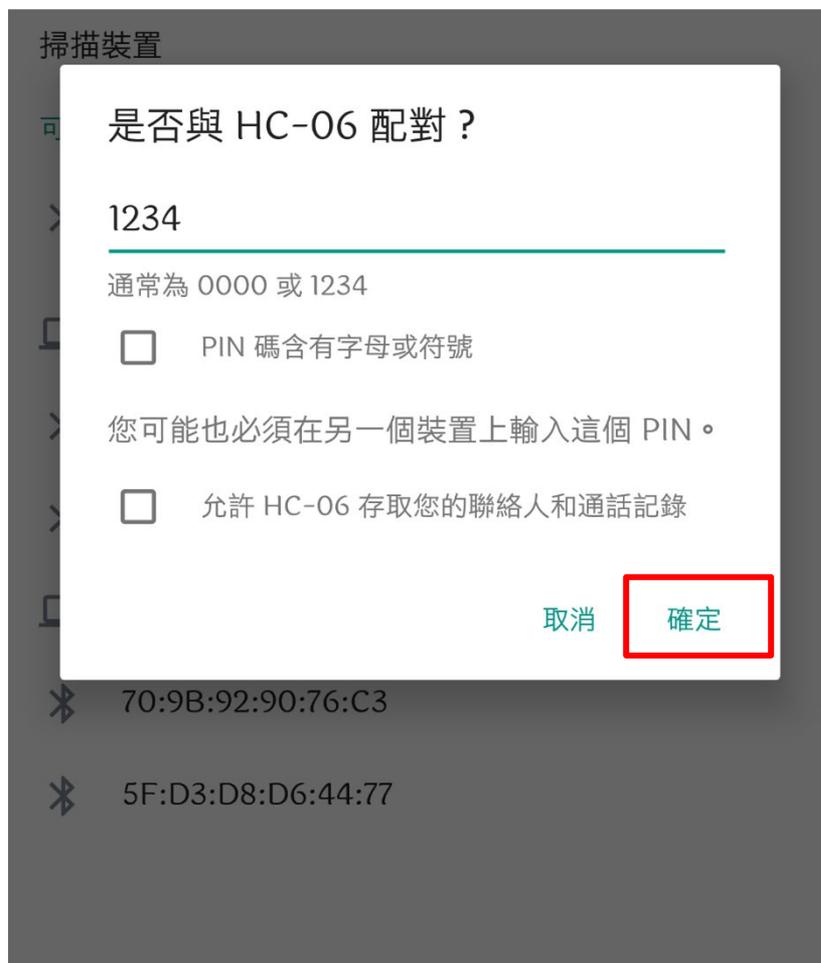
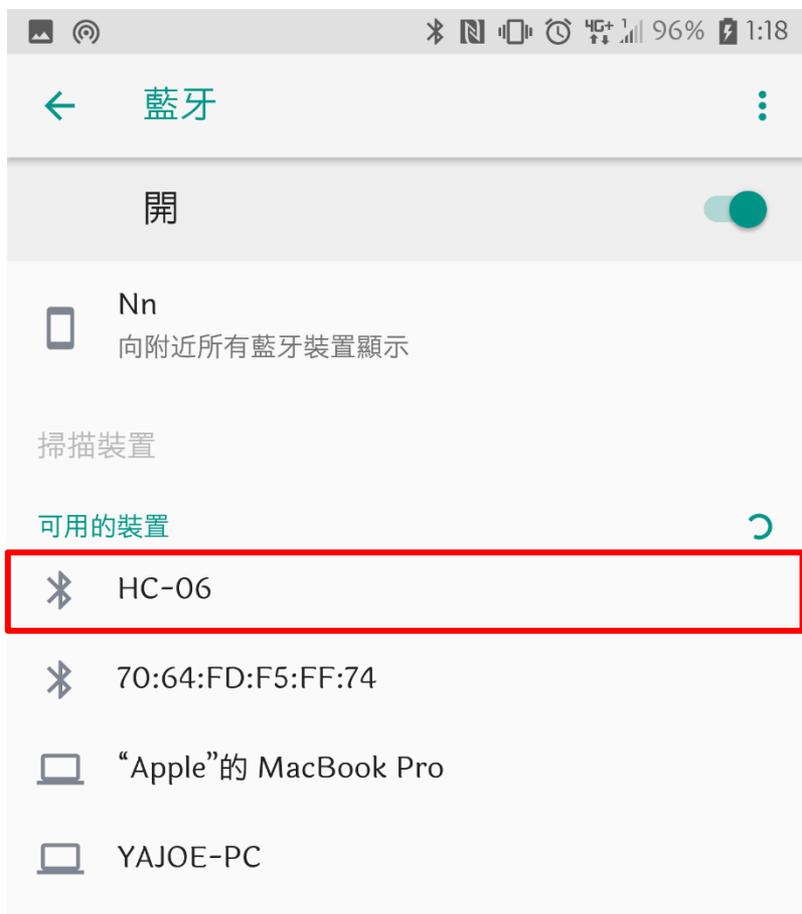
開啟手機設定中的藍牙，點選掃描裝置搜尋藍牙  
(每家手機畫面不同，圖片為HTC手機示範，僅供參考)



# STEP.5

## 與HC-06進行配對，密碼預設為1234

( HC-06為藍牙型號，當多台機器同時使用時，最上面為最靠近你的訊號 )



## STEP.6

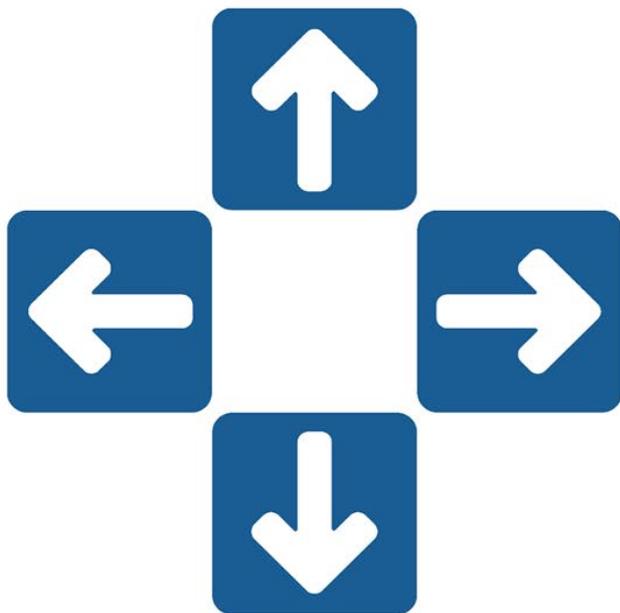
開啟手機遙控APP，點選藍牙列表



藍芽列表



中斷連接



## STEP.7

點選HC-06，即可進行遙控

START！智慧小車遙控

00:21:13:02:83:8E HC-06

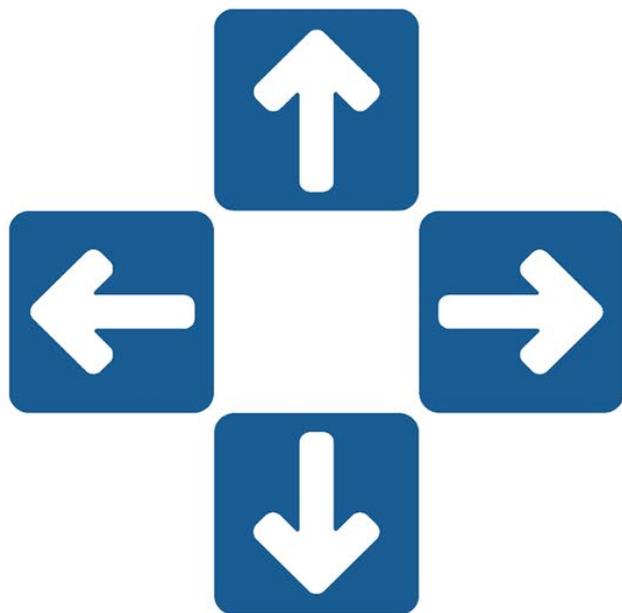
# STEP.8



藍芽列表



中斷連接



方向鍵  
控制方向



UP/DOWN  
控制手臂



OPEN/CLOSE  
控制夾爪



# 活動10：藍牙遙控小車

## 小車遙控機制

1. 手機APP依據使用者按下的按鍵

發送命令字元

2. Arduino 透過藍芽晶片收到命令字元後

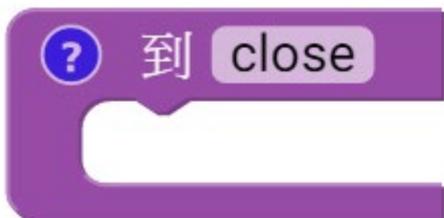
根據字元執行相對應的副程式

迴圈程序:

```
迴圈程序:  
如果 藍芽有效資料?  
執行 賦值 c 到 從藍芽讀取字元  
  如果 c = 'w'  
    執行 forward  
  否則如果 c = 'x'  
    執行 backward  
  否則如果 c = 'a'  
    執行 left_turn_0  
  否則如果 c = 'd'  
    執行 right_turn_0  
  否則如果 c = 's'  
    執行 stop  
  否則如果 c = '0'  
    執行 up  
  否則如果 c = '1'  
    執行 down  
  否則如果 c = '2'  
    執行 open  
  否則如果 c = '3'  
    執行 close  
  否則如果 c = '8'  
    執行 orig
```

## 活動10：藍牙遙控小車(吸盤版)

請找到爪子的副程式，並改寫其內容。



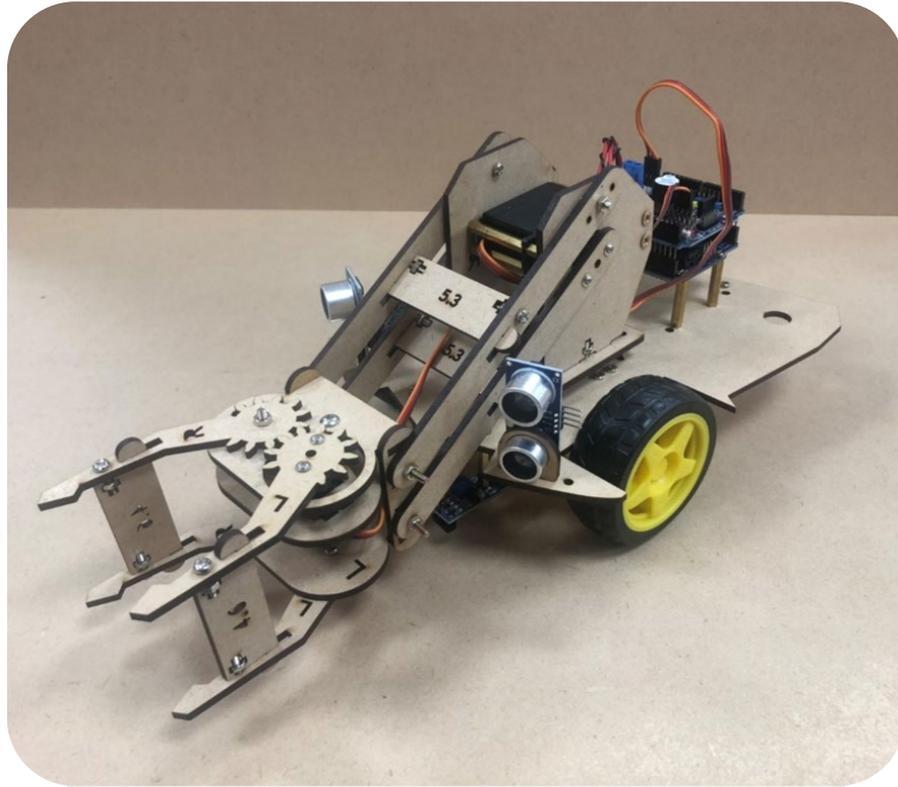
抓取物品：  
幫浦保持抽氣，不須給任何指令。



放下物品：  
電磁閥通電阻斷抽氣管線

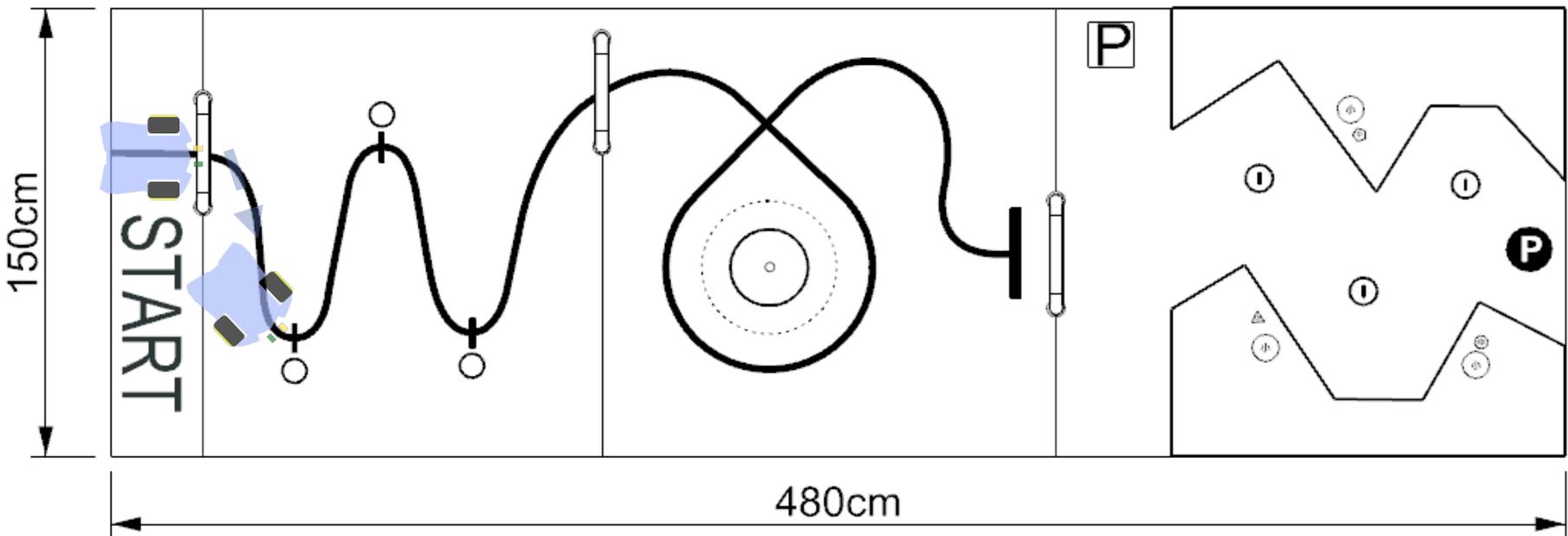
ps.請調整秒數至物品可以順利放下

# START! 智慧小車



-單元9 綜合練習-

直到雙黑線前不停地循跡

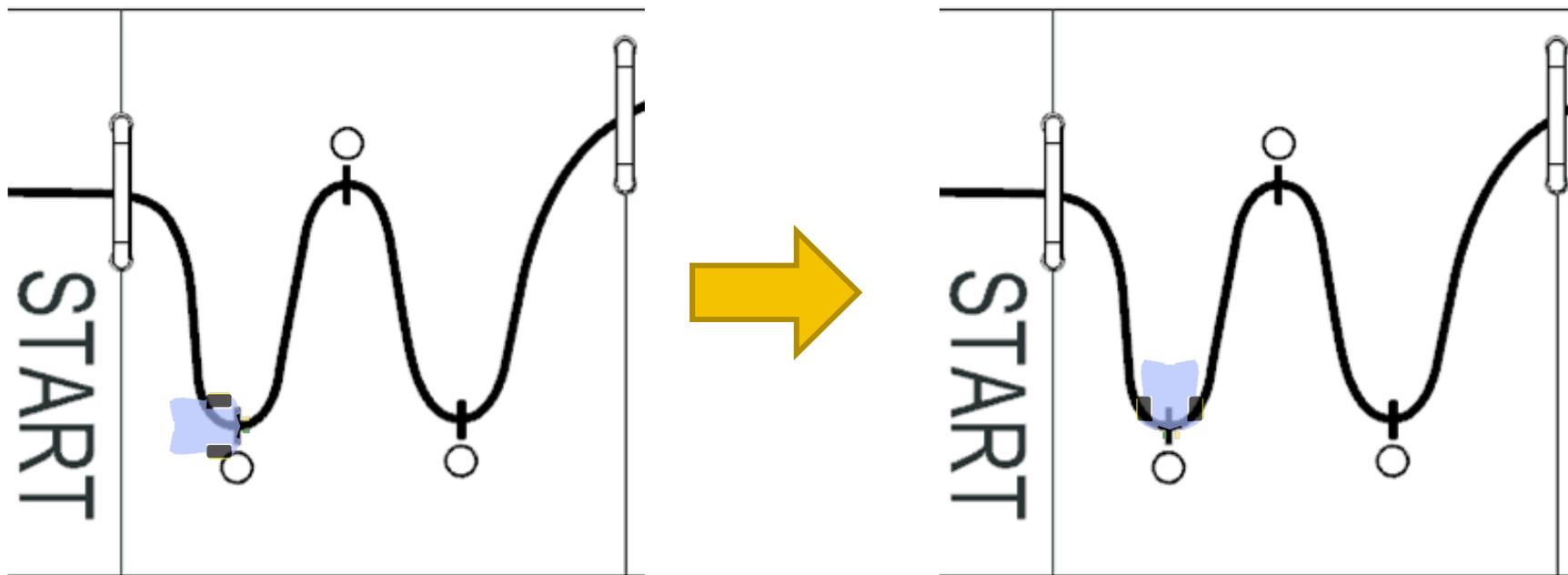


# 直到雙黑線前不停地循跡

```
到 follow&stop
重複 直到
  從 A0 讀取紅外線光感測器的數值 < 500 且 從 A1 讀取紅外線光感測器的數值 < 500
執行 如果
  從 A0 讀取紅外線光感測器的數值 ≥ 500 且 從 A1 讀取紅外線光感測器的數值 ≥ 500
  執行 forward
  如果
    從 A0 讀取紅外線光感測器的數值 < 500 且 從 A1 讀取紅外線光感測器的數值 ≥ 500
    執行 left_turn_0
  如果
    從 A0 讀取紅外線光感測器的數值 ≥ 500 且 從 A1 讀取紅外線光感測器的數值 < 500
    執行 right_turn_0
stop
等待 200 毫秒
```

```
到 follow&pass
重複 直到
  從 A0 讀取紅外線光感測器的數值 < 500 且 從 A1 讀取紅外線光感測器的數值 < 500
執行 如果
  從 A0 讀取紅外線光感測器的數值 ≥ 500 且 從 A1 讀取紅外線光感測器的數值 ≥ 500
  執行 forward
  如果
    從 A0 讀取紅外線光感測器的數值 < 500 且 從 A1 讀取紅外線光感測器的數值 ≥ 500
    執行 left_turn_0
  如果
    從 A0 讀取紅外線光感測器的數值 ≥ 500 且 從 A1 讀取紅外線光感測器的數值 < 500
    執行 right_turn_0
forward
```

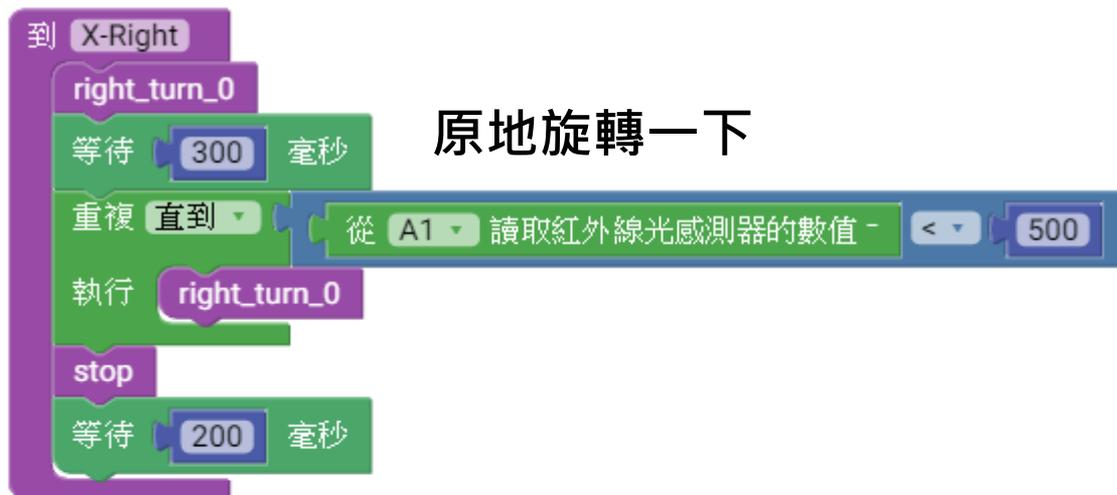
## 交叉點轉90度



循跡過黑線 (超過線讓車身中心對準交叉點)

- > 原地旋轉一下
- > 原地旋轉到感測器對準黑線

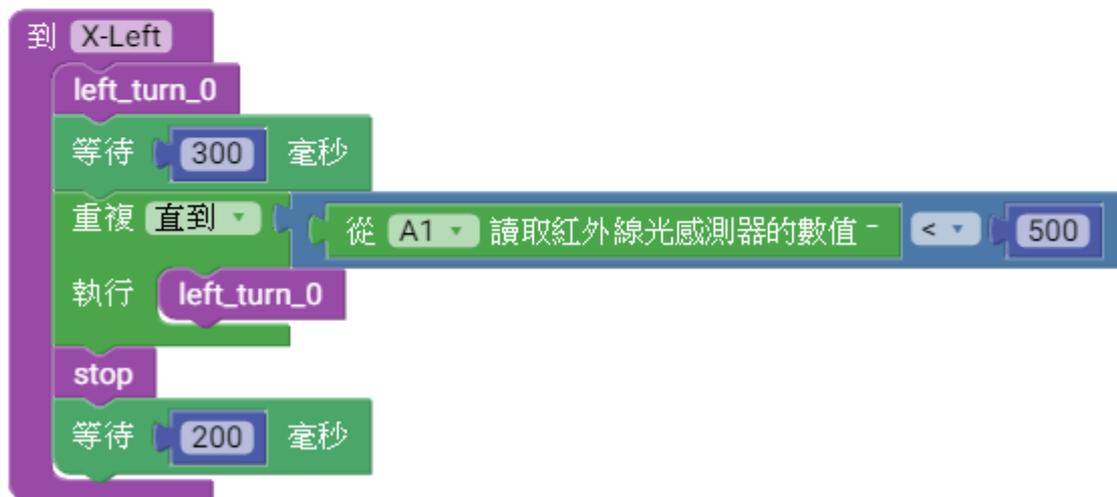
# 交叉點轉90度



```
到 X-Right  
  right_turn_0  
  等待 300 毫秒  
  重複 直到 從 A1 讀取紅外線光感測器的數值 - < 500  
  執行 right_turn_0  
  stop  
  等待 200 毫秒
```

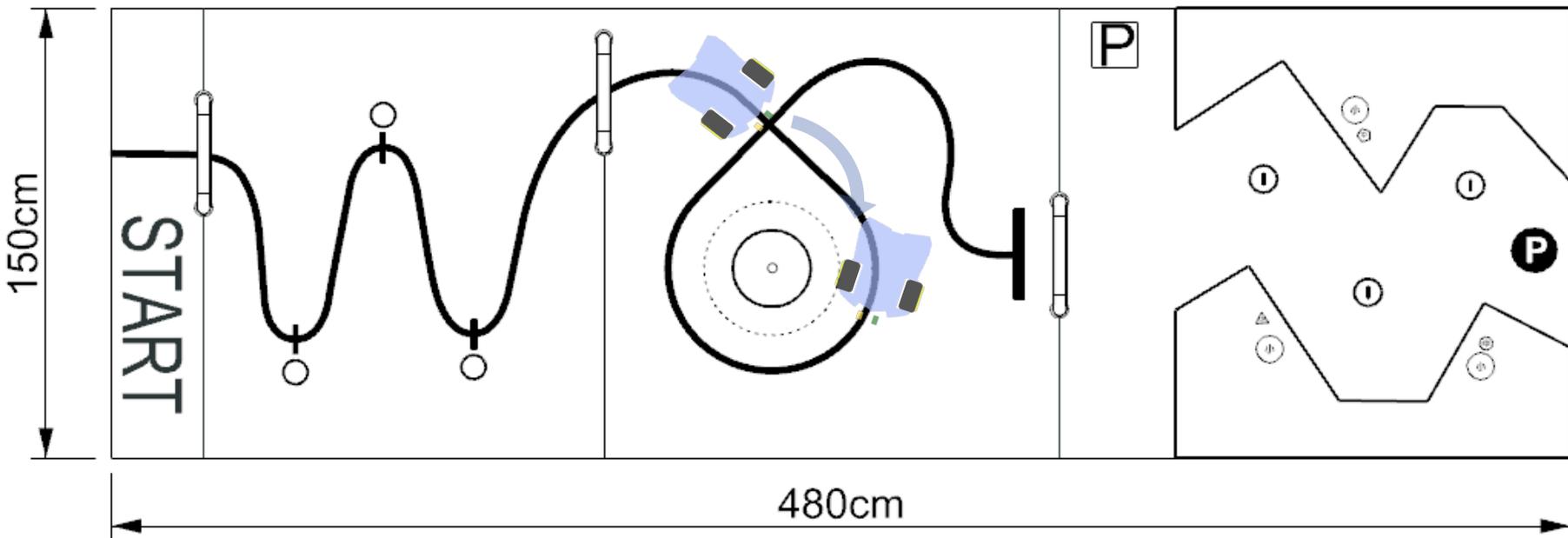
原地旋轉一下

原地旋轉至感測器對準黑線



```
到 X-Left  
  left_turn_0  
  等待 300 毫秒  
  重複 直到 從 A1 讀取紅外線光感測器的數值 - < 500  
  執行 left_turn_0  
  stop  
  等待 200 毫秒
```

# 單邊循跡繞1/4圈

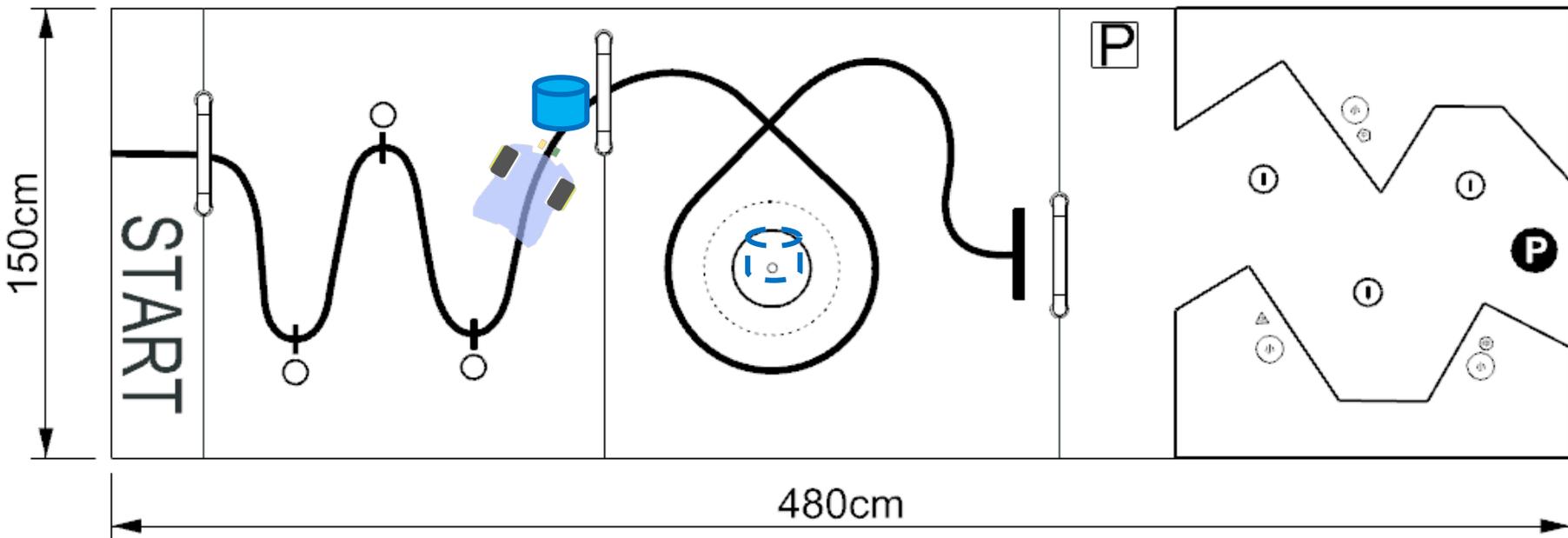


# 單邊循跡繞1/4圈

計算需要多久才能走1/4圈？



# 將道具放置圓圈中



# 將道具放置圓圈中

設定伺服馬達腳位 servo\_9 , 腳位 9

設定伺服馬達腳位 servo\_10 , 腳位 10

close&up

follow&pass

重複執行 5 秒

```
if (從 A0 讀取紅外線光感測器的數值 - ≥ 500 且 從 A1 讀取紅外線光感測器的數值 - ≥ 500)
  執行 forward
else if (從 A0 讀取紅外線光感測器的數值 - < 500 且 從 A1 讀取紅外線光感測器的數值 - ≥ 500)
  執行 left_turn_0
else if (從 A0 讀取紅外線光感測器的數值 - ≥ 500 且 從 A1 讀取紅外線光感測器的數值 - < 500)
  執行 right_turn_0
```

stop

right\_turn\_0

等待 200 毫秒

stop

down&open

left\_turn\_0

等待 200 毫秒

stop

follow&stop

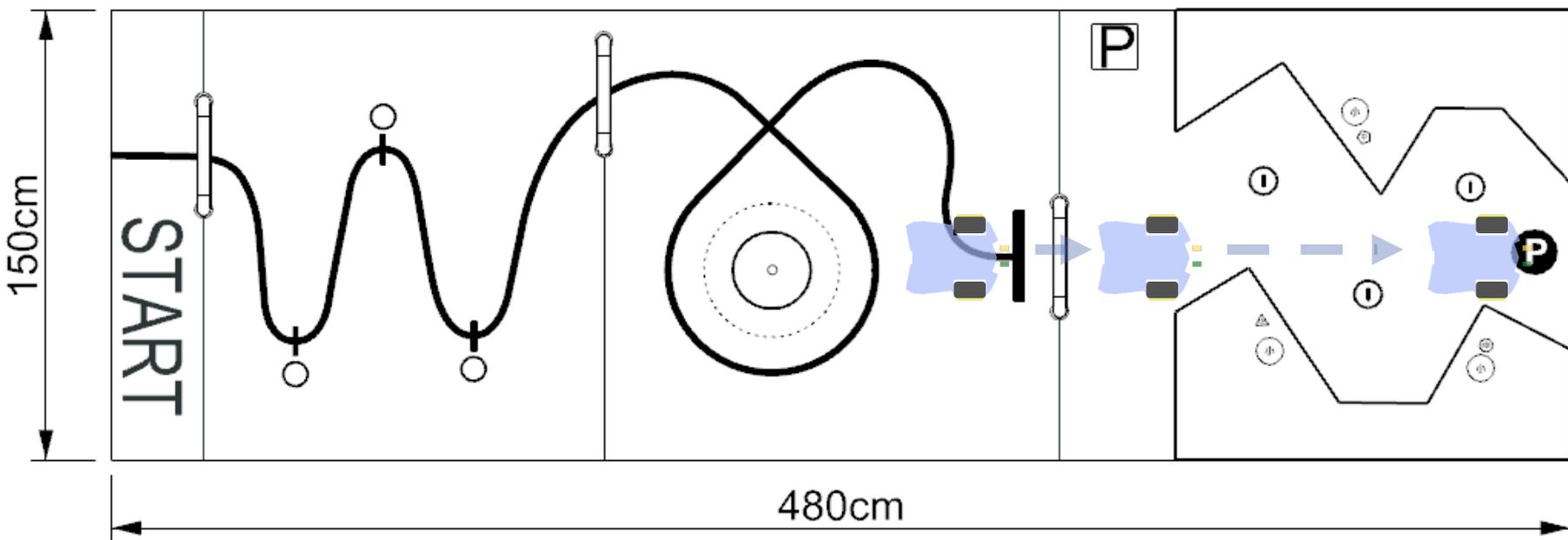
抓起道具

循跡至指定位置(秒數)

放下道具

回到原路、循跡至下一關

# 直到黑點前不停地避障



# 直到黑點前不停地避障(1)

The image shows a Scratch script for obstacle avoidance. It starts with a 'Repeat Until' loop with the condition '直到黑點前不停避障'. Inside the loop, there are two 'If' blocks. The first 'If' block checks if the distance from sensor A0 is greater than 800 OR the distance from sensor A1 is greater than 800. If true, it sets two HC-SR04 sensors (A2/A3 and A4/A5) to a distance of 10 cm. The second 'If' block checks if the distance from sensor A2 is greater than 10 cm AND the distance from sensor A4 is less than or equal to 10 cm. If true, it also sets both sensors to 10 cm. After the 'If' blocks, there is a 'forward' movement block, followed by a 'backward' movement block, a 'wait 300 ms' block, a 'left\_turn\_0' block, and another 'wait 300 ms' block.

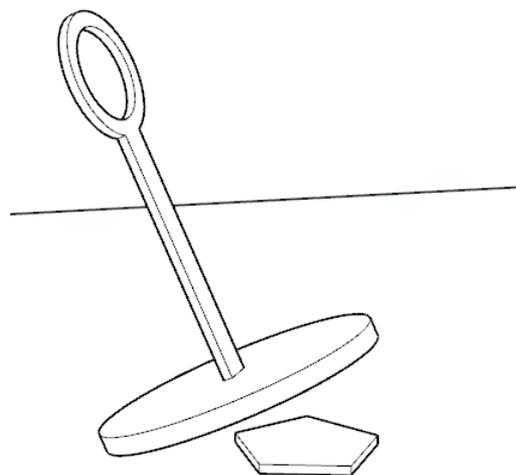
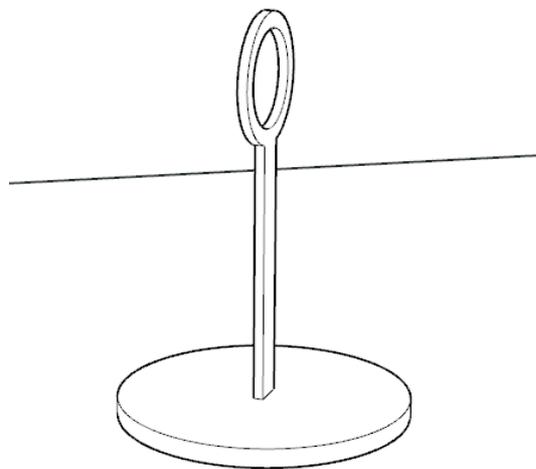
```
repeat_until(直到黑點前不停避障)
  read_value(A0) > 800 or read_value(A1) > 800
  if true
    set_distance(HC-SR04, A2, 10)
    set_distance(HC-SR04, A4, 10)
  if true
    set_distance(HC-SR04, A2, 10)
    set_distance(HC-SR04, A4, 10)
  forward
  backward
  wait(300, ms)
  left_turn_0
  wait(300, ms)
```

## 直到黑點前不停地避障(2)

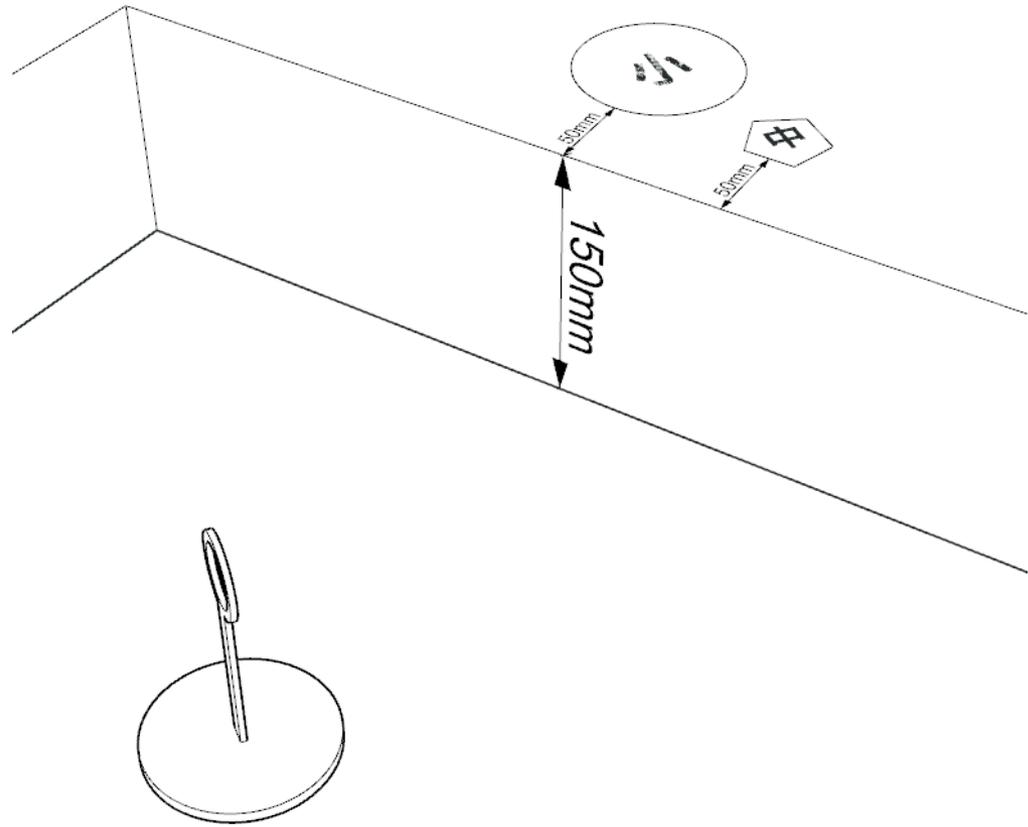


# 吸盤模組練習

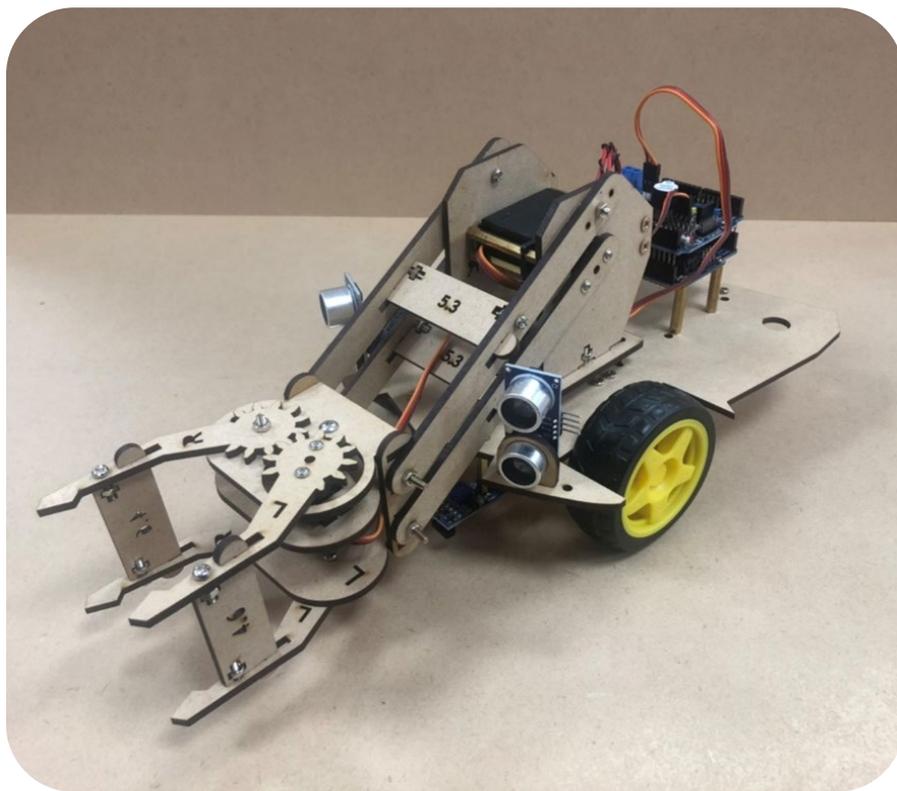
移除吊牌、吸取下方之零件



將吊牌/零件放回指定位置



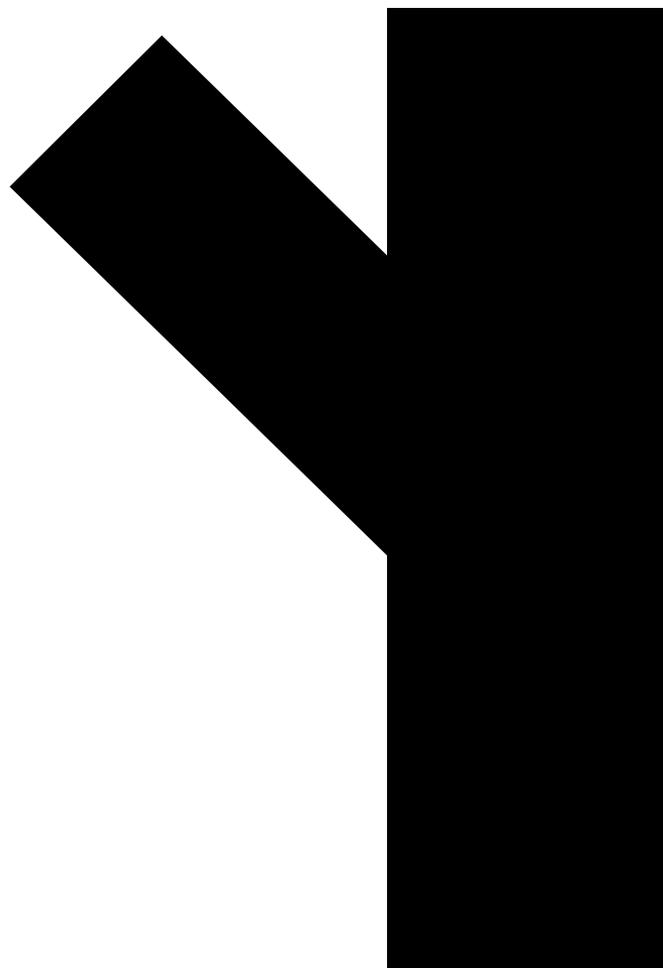
# START! 智慧小車



-單元10 高手挑戰-

## 循跡過Y字分岔(國中)

- 前提：已知要往左彎
- 偵測到分岔時情形：  
等同於遇到雙黑
- 利用單邊循跡  
避開分叉



# 循跡過Y字岔

設定程序:

```
重覆 直到
  執行 如果 (從 A0 讀取紅外線光感測器的數值 < 500 且 從 A1 讀取紅外線光感測器的數值 < 500)
    執行 forward
  執行 如果 (從 A0 讀取紅外線光感測器的數值 ≥ 500 且 從 A1 讀取紅外線光感測器的數值 ≥ 500)
    執行 left_turn_0
  執行 如果 (從 A0 讀取紅外線光感測器的數值 < 500 且 從 A1 讀取紅外線光感測器的數值 ≥ 500)
    執行 left_turn_0
  執行 如果 (從 A0 讀取紅外線光感測器的數值 ≥ 500 且 從 A1 讀取紅外線光感測器的數值 < 500)
    執行 right_turn_0
  重覆 10 次
    執行 重覆 直到 (從 A0 讀取紅外線光感測器的數值 > 500)
    執行 left_turn_1
    執行 重覆 直到 (從 A0 讀取紅外線光感測器的數值 < 500)
    執行 right_turn_1
stop
```

迴圈程序:

左側單邊循跡，預估十次可以轉彎成功

直到白以前左轉

直到黑以前右轉

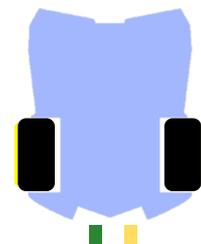
單邊循跡建議使用單輪轉彎



想想看：原地轉彎為什麼容易失敗？

## 高手挑戰任務(一)

在S型黑線上做循跡，必須跨越中段的橫線，並在最後的Y字形選擇右轉，停在最終的橫線(皇冠處)。



**STAR**

**T**

# 高手挑戰任務(一) 主程式、循跡過橫線、循跡停黑線副程式

設定程序:

follow&pass

follow&pass

follow&Y-Right

follow&stop

迴圈程序:

```
到 follow&pass
重複 直到
  從 A0 讀取紅外線光感測器的數值 - < 500 且 從 A1 讀取紅外線光感測器的數值 - < 500
執行 如果
  從 A0 讀取紅外線光感測器的數值 - ≥ 500 且 從 A1 讀取紅外線光感測器的數值 - ≥ 500
執行 forward
執行 如果
  從 A0 讀取紅外線光感測器的數值 - < 500 且 從 A1 讀取紅外線光感測器的數值 - ≥ 500
執行 left_turn_0
執行 如果
  從 A0 讀取紅外線光感測器的數值 - ≥ 500 且 從 A1 讀取紅外線光感測器的數值 - < 500
執行 right_turn_0
forward
等待 1000 毫秒
```

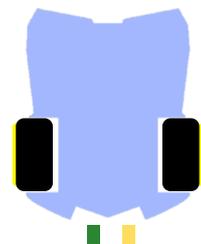
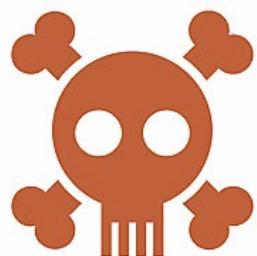
```
到 follow&stop
重複 直到
  從 A0 讀取紅外線光感測器的數值 - < 500 且 從 A1 讀取紅外線光感測器的數值 - < 500
執行 如果
  從 A0 讀取紅外線光感測器的數值 - ≥ 500 且 從 A1 讀取紅外線光感測器的數值 - ≥ 500
執行 forward
執行 如果
  從 A0 讀取紅外線光感測器的數值 - < 500 且 從 A1 讀取紅外線光感測器的數值 - ≥ 500
執行 left_turn_0
執行 如果
  從 A0 讀取紅外線光感測器的數值 - ≥ 500 且 從 A1 讀取紅外線光感測器的數值 - < 500
執行 right_turn_0
stop
```

# 高手挑戰任務(一) 單邊循跡副程式

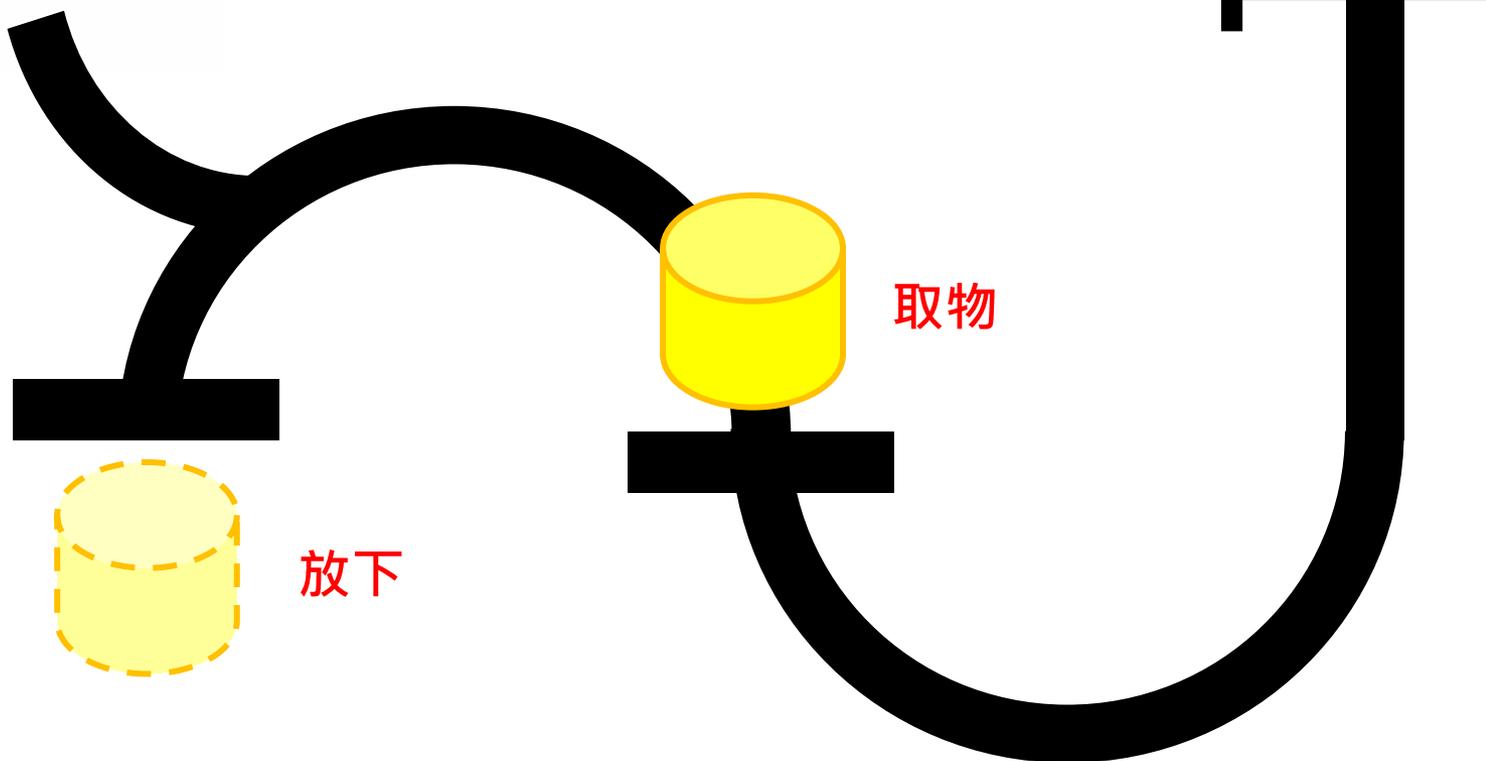
The image shows a Scratch script for a sub-program named "follow&Y-Right". The script is contained within a "到" (Go to) block. It begins with a "重複 直到" (Repeat until) loop that continues as long as the infrared sensor values from A0 and A1 are both less than 500. Inside this loop, there are three conditional blocks: 1) An "如果" (If) block where both A0 and A1 values are greater than or equal to 500, followed by an "執行 forward" (Execute forward) block. 2) An "如果" (If) block where A0 is less than 500 and A1 is greater than or equal to 500, followed by an "執行 left\_turn\_0" (Execute left\_turn\_0) block. 3) An "如果" (If) block where A0 is greater than or equal to 500 and A1 is less than 500, followed by an "執行 right\_turn\_0" (Execute right\_turn\_0) block. After the "直到" loop, there is a "重複 10 次" (Repeat 10 times) block. Inside this block, there are three "直到" loops: 1) A "重複 直到" loop where A1 is greater than or equal to 500, followed by an "執行 right\_turn\_1" (Execute right\_turn\_1) block. 2) A "重複 直到" loop where A1 is less than 500, followed by an "執行 left\_turn\_1" (Execute left\_turn\_1) block.

## 高手挑戰任務(一) 改

在S型黑線上做循跡，必須跨越中段的橫線並將目標物舉起，繼續循跡後於Y字形選擇左轉，停在最終的橫線並放下目標物。



**STAR**



# 高手挑戰任務(一) 改

設定程序:

- down&open
- follow&pass
- follow&stop
- close&up
- follow&Y-Left
- follow&stop
- down&open

迴圈程序:

除了爪子的動作之外，  
可以將循跡過黑線、  
循跡停黑線、  
循跡過Y字左轉  
都寫成副程式，使得主  
程式更簡略好懂。

到 down&open

- 伺服馬達 servo\_10 旋轉到 120
- 等待 1000 毫秒
- 伺服馬達 servo\_9 旋轉到 120
- 等待 1000 毫秒

到 close&up

- 伺服馬達 servo\_9 旋轉到 30
- 等待 1000 毫秒
- 伺服馬達 servo\_10 旋轉到 90
- 等待 1000 毫秒

# 高手挑戰任務(一)改

到 follow&pass

重複 直到

- 從 A0 讀取紅外線光感測器的數值 - < 500 且 從 A1 讀取紅外線光感測器的數值 - < 500

執行

- 如果 從 A0 讀取紅外線光感測器的數值 - ≥ 500 且 從 A1 讀取紅外線光感測器的數值 - ≥ 500
- 執行 forward
- 如果 從 A0 讀取紅外線光感測器的數值 - < 500 且 從 A1 讀取紅外線光感測器的數值 - ≥ 500
- 執行 left\_turn\_0
- 如果 從 A0 讀取紅外線光感測器的數值 - ≥ 500 且 從 A1 讀取紅外線光感測器的數值 - < 500
- 執行 right\_turn\_0

forward

等待 500 毫秒

到 follow&stop

重複 直到

- 從 A0 讀取紅外線光感測器的數值 - < 500 且 從 A1 讀取紅外線光感測器的數值 - < 500

執行

- 如果 從 A0 讀取紅外線光感測器的數值 - ≥ 500 且 從 A1 讀取紅外線光感測器的數值 - ≥ 500
- 執行 forward
- 如果 從 A0 讀取紅外線光感測器的數值 - < 500 且 從 A1 讀取紅外線光感測器的數值 - ≥ 500
- 執行 left\_turn\_0
- 如果 從 A0 讀取紅外線光感測器的數值 - ≥ 500 且 從 A1 讀取紅外線光感測器的數值 - < 500
- 執行 right\_turn\_0

stop

等待 500 毫秒

# 高手挑戰任務(一)改

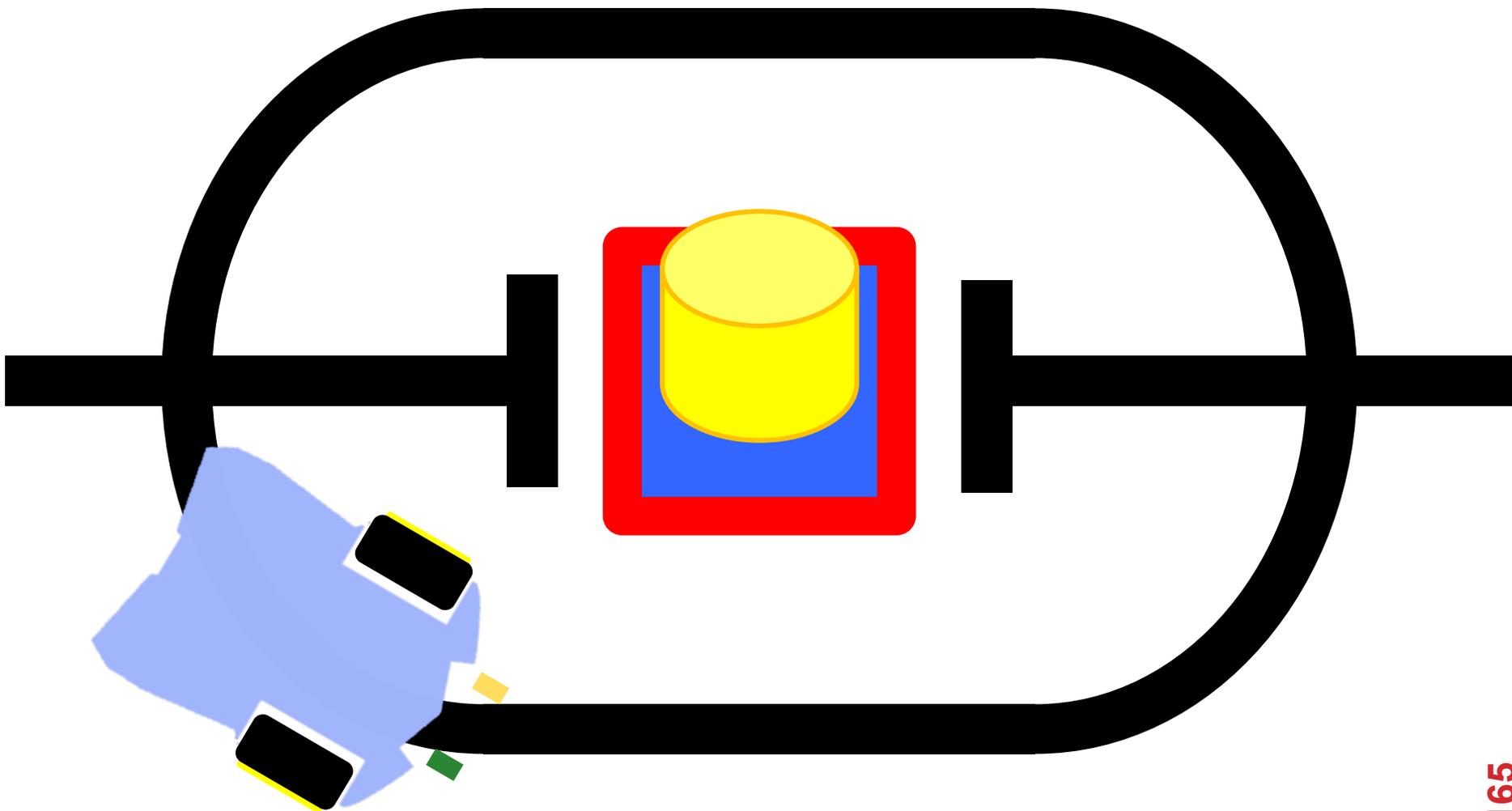
The code is organized into several nested loops and conditional blocks:

- Initial Loop:** A "Repeat Until" block with a dropdown set to "直到" (Until). It contains two "Read Infrared Sensor Value" blocks for ports A0 and A1, both with comparison operators set to "<" (less than) and the value 500.
- Forward Movement:** An "If" block with a dropdown set to "如果" (If). It contains two "Read Infrared Sensor Value" blocks for ports A0 and A1, both with comparison operators set to "≥" (greater than or equal to) and the value 500. Below it is an "Execute" block labeled "forward".
- Left Turn:** An "If" block with a dropdown set to "如果" (If). It contains two "Read Infrared Sensor Value" blocks for ports A0 and A1. The A0 block has a "<" operator and the value 500, while the A1 block has a "≥" operator and the value 500. Below it is an "Execute" block labeled "left\_turn\_0".
- Right Turn:** An "If" block with a dropdown set to "如果" (If). It contains two "Read Infrared Sensor Value" blocks for ports A0 and A1. The A0 block has a "≥" operator and the value 500, while the A1 block has a "<" operator and the value 500. Below it is an "Execute" block labeled "right\_turn\_0".
- Final Loop:** A "Repeat" block with a dropdown set to "重複" (Repeat) and a count of 10. It contains two sub-loops:
  - Left Turn Loop:** A "Repeat Until" block with a dropdown set to "直到" (Until). It contains a "Read Infrared Sensor Value" block for port A0 with a "≥" operator and the value 500, followed by an "Execute" block labeled "left\_turn\_1".
  - Right Turn Loop:** A "Repeat Until" block with a dropdown set to "直到" (Until). It contains a "Read Infrared Sensor Value" block for port A0 with a "<" operator and the value 500, followed by an "Execute" block labeled "right\_turn\_1".

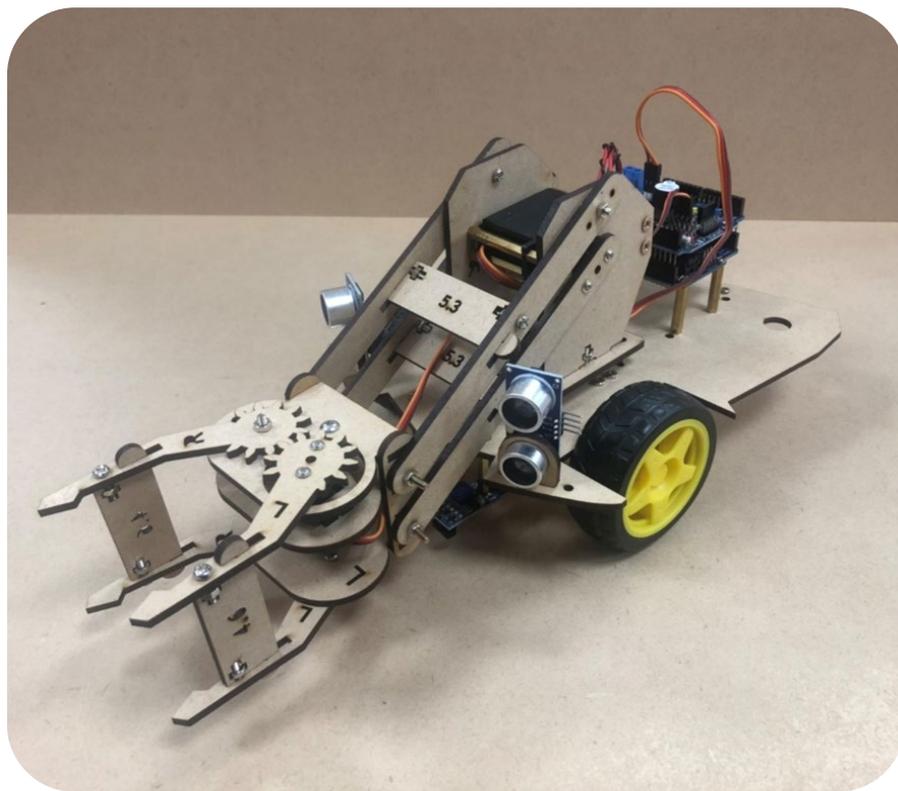
← 過Y字記得使用單輪轉彎

## 高手挑戰任務(二)

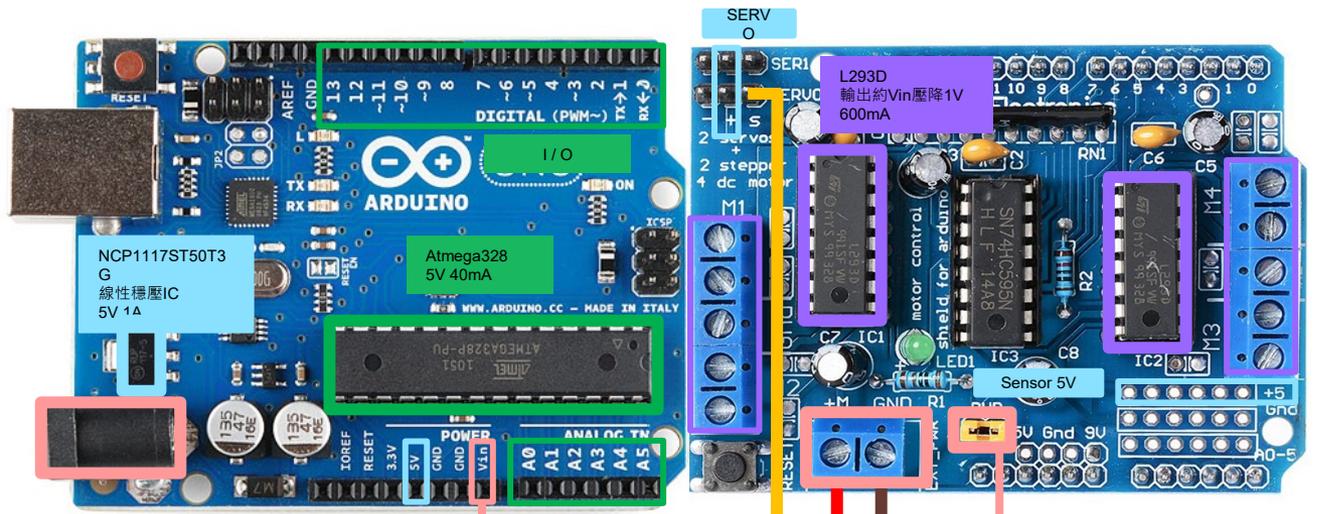
環狀循跡半圈後，在十字交接處轉彎進到取物區，將目標物夾起並舉起，回到環狀循跡繼續走半圈，下一個十字交接處轉彎進取物區放下目標物，再回到環狀循跡，周而復始。



# START! 智慧小車



- 伺服馬達電源強化 -



18650 2C  
8.4V 串聯電池盒  
2A

18650 2C  
8.4V 串聯電池盒  
2A

