

# 初中「機械人」學習資源

基礎單元二：機械人的輸入和輸出



香港特別行政區政府教育局  
課程發展處科技教育組  
2019年5月

**如對本學與教資源有任何意見及建議，請致函：**

香港九龍塘沙福道 19 號西座 W101 室

教育局課程發展處科技教育組

總課程發展主任（科技教育）

本學習資源版權，除在鳴謝頁所列舉的圖片外，全屬於香港特別行政區政府教育局擁有。

教育局歡迎學校等教育團體使用本學習資源的內容作非牟利的教學用途。任何情況下使用本學習資源，需作出鳴謝，教育局保留本學習資源版權。

未經本局事先允許，不能以任何形式使用其中教材作出版或其他用途，否則教育局將保留一切追究的權利。

**© 版權所有 2019**

本學習資源由香港機械人學院製作。

# 目錄

章節一：機械人的輸入元件	P.3
章節二：超聲波感應器和溫濕度感應器	P.8
章節三：機械人的輸出元件	P.15
章節四：伺服馬達	P.20
章節五：單元專題研習 – 清潔機器人	P.25

# 基礎單元二：機械人的輸入和輸出

## 章節一：機械人的輸入元件

### I. 輸入

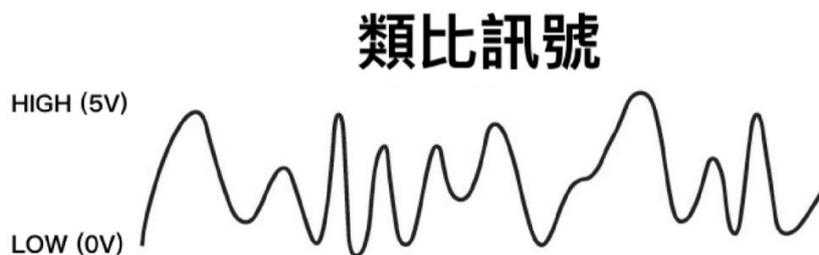
機械人是由許多不同的功能所組成，如輸入、輸出、控制、記憶、及運算邏輯等單元。電腦是一台機器，它可以用來接收資料（輸入），再處理成有用的資訊（輸出），並將其儲存到其他地方。

### II. 數碼訊號和類比訊號

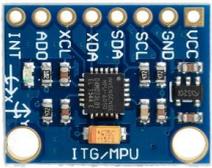
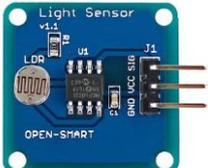
輸入裝置是能夠產生電流訊號給微控制器的電子元件。電流訊號一般分為數碼訊號和類比訊號兩種。訊號的變數只有最大值和最小值，而且變化的時間極短。數碼訊號可理解為固定的訊號，例如電壓為 5 伏特的話，一段數碼訊號會是：



相反，類比訊號可理解為浮動的訊號，同樣都是 5 伏特電壓的情況下，訊號的變數的值在 0 至 5 伏特之間有不同，變化的時間不一。一段類比訊號會是：



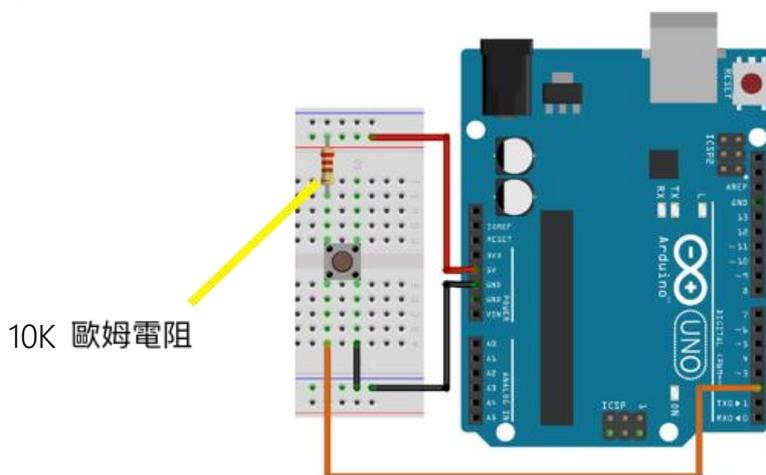
常用感應器列表：

	<p><b>超聲波感應器</b> 偵測距離</p>		<p><b>壓力感應器</b> 偵測膠片承受壓力</p>
	<p><b>紅外線感應器</b> 偵測顏色</p>		<p><b>酸鹼度感應器</b> 偵測酸鹼度 (pH 值)</p>
	<p><b>陀螺儀</b> 感測與維持方向</p>		<p><b>聲音感應器</b> 感測音量變化</p>
	<p><b>氣體感應器</b> 偵測某一種氣體</p>		<p><b>磁力感應器</b> 偵測磁場變化</p>
	<p><b>UV感應器</b> 偵測紫外線</p>		<p><b>水感應器</b> 偵測是否含水和 水壓</p>
	<p><b>溫濕度感應器</b> 偵測溫度和濕度</p>		<p><b>氧氣感應器</b> 偵測氧氣含量</p>
	<p><b>光感應器</b> 偵測光線強度</p>		<p><b>數碼壓力感應器</b> 偵測壓力</p>

以上的列表是我們在製作機械人時比較普遍會使用的感應器，除了開關以外，其餘都是類比訊號輸入。因為開關或按鈕只有兩個狀態：分別是開和關，所以訊號也比較明顯。以電壓來計算，只有 0 伏特和 5 伏特，是數碼輸入的一種。但其他感應器，例如溫濕度感應器、壓力感應器，接收的訊號電壓不限於 0 伏特和 5 伏特，有時是 2.4，亦有可能是 3.8，沒有一定的狀態，所以被歸類為類比輸出的一種。

### III. 動手做 – 按鈕訊號

本課第一個任務非常簡單，我們會連接一個按鈕在 Arduino 控制板上，並且利用 Arduino 篇寫程式感應按鈕是否按下。以下的連接方法會使控制板在沒有按下按鈕的時候感應到高電位訊號（5V），當按下按鈕的時候便能夠感應低電位訊號。下圖表示了如何接駁感應按鈕：



上圖的電線是用杜邦線來連接。按鈕、電阻全部駁在麵包板上面，最後再接上 Arduino UNO 微控制板。留意我們必須要加上上拉電阻（Pull-up resistor），否則當按鈕按下的時候會將 5V 與 GND 連在一起，形成短路，把 Arduino 控制板燒壞。

如果完成以上的接駁任務，你已經完成了基本的硬件設備。接着我們可以把 USB 線連入微控制板，再接上電腦。接上電腦後，可以進入編程的部份。

控制按鈕的程式碼全圖，可參考檔案 2\_1\_1\_control\_button

```
1 int button = 2;
2 int result = 0;
3
4 void setup() {
5   pinMode(button, INPUT);
6   Serial.begin(9600);
7 }
8
9 void loop() {
10  result = digitalRead(button);
11  if(result == LOW) {
12    Serial.println("Button was pressed!!!!!!");
13  } else {
14    Serial.println("Button was not pressed!!!!!!");
15  }
16 }
```

定義變數

定義腳位及打開串行通訊頻道

閱讀腳位並在串行通訊頻道列印出按鈕動作

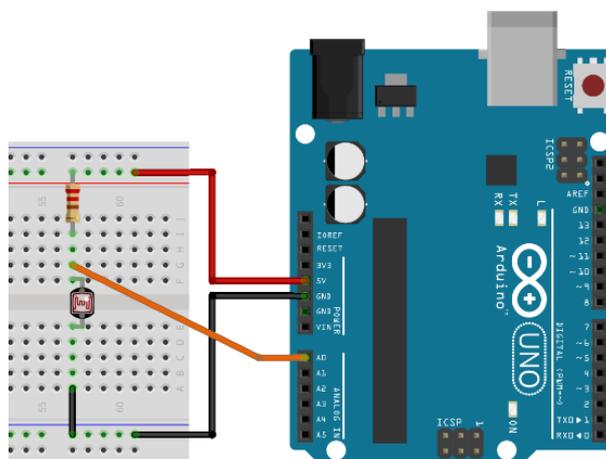
程式簡單來說分為三部份。第一部份，我們需要定義按鈕插腳的變數“button”，同時也把按鈕所讀到的訊號儲存為變數“result”。第二部份，我們定義按鈕插腳設定為輸入，並打開傳輸率是 9600 的串行通訊頻道，串行通訊頻道可透過程式右上方的按鈕開啟。第三部份，我們利用函數“digitalRead()”感應按鈕的數碼數值，並將此數值儲存在變數“result”上。我們利用變數“result”進行邏輯判定：假如變數“result”為低電位 (LOW) 的話，我們在串行通訊頻道打印出“Button was pressed!!!!!!”，否則我們會打印出“Button was not pressed!!!!!!”。完成程式後，可以上載到 Arduino UNO 微控制板進行測試。如果測試成功，你會於串行通訊頻道中看到以下資訊。

```
button was not pressed!!!!!!
button was not pressed!!!!!!
button was not pressed!!!!!!
button was not pressed!!!!!!
button was pressed!!!!!!
```

自動捲動      沒有行結尾      9600 baud      Clear output

#### IV. 動手做 – 感應按鈕光線

本課節希望同學能夠學習使用光線感應器，我們會使用光敏電阻 (Photoresistor) 來當感應器。光敏電阻的特性是當我們用光向光敏電阻照射時，光敏電阻的電阻值會變小；反之，當沒有光線照在光敏電阻上時，光敏電阻的電阻值會變大。根據下圖的連接方法中，當有光線照射在光敏電阻的時候，光敏電阻的電阻值會下降，A0 數碼輸入腳位的數值會接近 GND，即數字會接近 0；當環境變暗時，A0 數碼輸入腳位數值會變大。



我們要注意必須加上並聯的普通電阻，這樣是防止當有強光照射在光敏電阻上時，光敏電阻的電阻值變小所產生的過大電流會燒壞 Arduino 控制板。如果你完成以上的接駁任務，你已經完成了基本的硬件設備。接着，我們可以把 USB 線連入微控制板，再接上電腦。接上電腦後，即可以進入編程的部份。

控制光敏電阻的程式碼全圖，可參考檔案 2\_1\_2\_control\_photoresistor

```
1 int photoresistor = A0;           定義腳變數
2 int result = 0;
3 -----
4 void setup() {
5   pinMode(photoresistor, INPUT);   設定程式
6   Serial.begin(9600);
7 }
8 -----
9 void loop() {
10  result = analogRead(photoresistor); 不停讀取光敏光阻的數值並
11  Serial.println(result);             在串行通訊頻道中列印出來
12 }
```

程式簡單來說分為三部份。第一部份，我們需要定義按鈕光敏電阻的變數“photoresistor”，同時也把按鈕所讀到的訊號儲存為變數“result”。第二部份，我們定義光敏電阻插腳設定為輸入，並打開傳輸率是 9600 的串行通訊頻道。第三部份，我們利用函數“analogRead()”感應光敏電阻的數碼數值，並將此數值儲存在變數“result”上。我們在串行通訊頻道打印出“result”的數值。完成程式後，可以上載到 Arduino UNO 微控制板進行測試。如果測試成功，你就完成這個任務了。

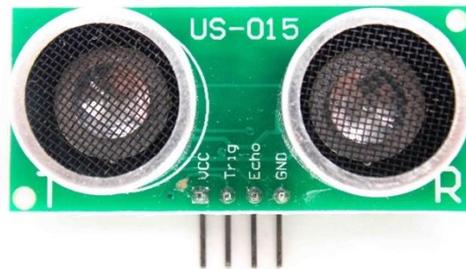
## V. 溫習問題

完成本課後你能夠回答以下的問題嗎？

1. 什麼是數碼訊號和類比訊號？
2. 你能夠舉出數碼訊號和類比訊號的例子嗎？
3. 你能夠利用 Arduino IDE 顯示數碼輸入訊號嗎？

## 章節二：超聲波感應器和溫濕度感應器

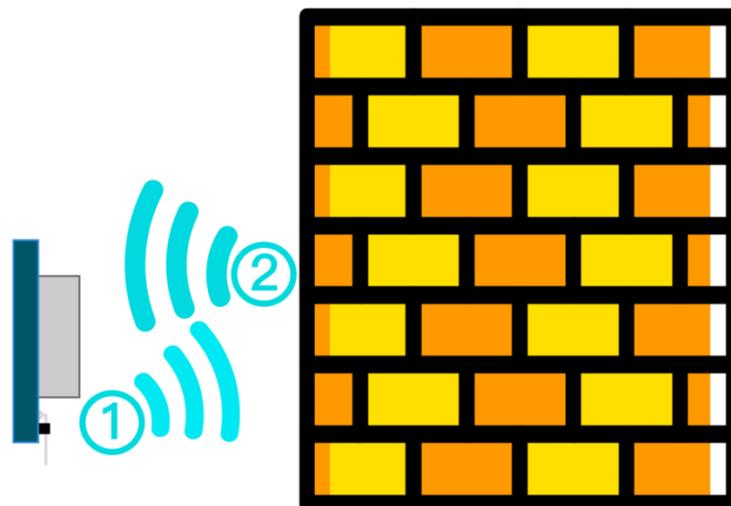
### I. 認識超聲波感應器



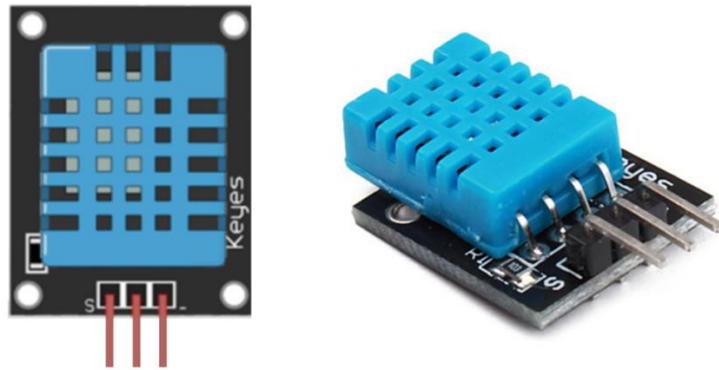
四針位的超聲波感應器

超聲波是高於人耳可聽到的頻率，一個正常的成人最高能夠聽到 20,000 Hz 以下的頻率。20,000 Hz 以上的頻率我們稱之為超聲波，因此人類無法聽到超聲波。

超聲波感應器的原理是有兩組硬件，一組負責發射超聲波，超聲波碰到障礙物時會回彈，回彈的超聲波會被另一組的接收器收到。收回來的數據是時間，假如我們能夠掌握超聲波的音速便能計算出時間等於多少距離。自然界裡也有很多使用超音波的原理，例如海豚和蝙蝠。一般情況下，公式為  $\text{distance(距離)} = \text{duration(持續時間)} \times 0.034/2$ 。下圖表示了超聲波感應器運作的原理：



## II. 認識溫濕度感應器



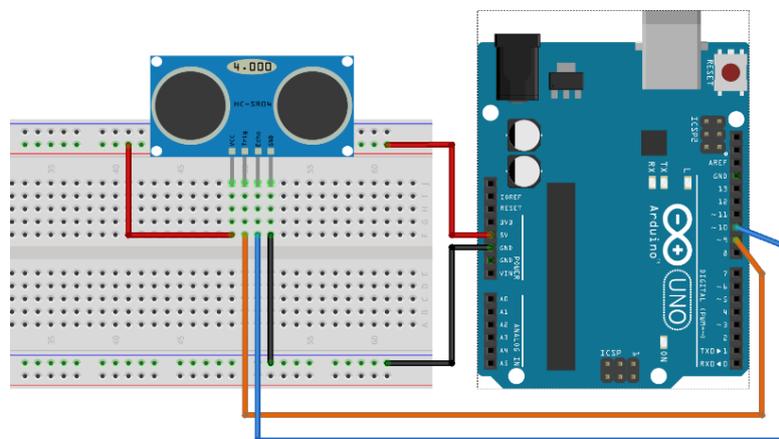
本文採用的是能檢測溫度和濕度的 DHT 11，是一款結合溫度及濕度的感測器。溫濕度感應器能夠把量度的溫度和濕度轉換成電子訊號，每一組訊號代表一個特定溫度或濕度。透過分析感應器輸出的訊號，我們便能知道環境的溫度和濕度。

## III. 動手做 – 超聲波感應器

本課任務會嘗試透過超聲波感應器量度距離，並把量度距離列印在串行監控視窗（Serial Monitor）上。這次任務共分為兩個部分，分別是連接 Arduino 和超聲波感應器，以及使用 Arduino 編程量度距離。

### 一 連接 Arduino 和超聲波感應器

我們根據以下方法連接超聲波感應器 US-015 和 Arduino。四針位的超聲波感應器有 Trig 針位及 Echo 針位。當數值出現異常時，我們能夠根據針位的讀數進行除錯。



這次任務只需要用到 Arduino UNO、麵包板和超聲波感應器 US-015。超聲波感應器 VCC 腳位需要連接 Arduino 的 5V 電源，而 GND 腳位則要連向 Arduino 的 GND。超聲波感應器其餘兩個腳位分別是“Trig”和“Echo”腳位。“Trig”腳位負責啟動超聲波感應器，並連接在 Arduino 的 9 號腳位。“Echo”腳位負責計算超聲波發射和反回感應器之間的時間，連接在 Arduino 的 10 號腳位。

## 二 使用 Arduino 編程量度距離

我們打開 Arduino IDE，並把 Arduino 連接電腦。在工具裡選擇 Arduino UNO 為開發板，並檢查序列埠是否已連接 Arduino。連接 Arduino 後，我們需要輸入以下程式，並把程式上傳到 Arduino。

超聲波感應器測距的程式碼全圖，可參考檔案 2\_2\_1\_ultrasonic\_sensor

```
1 int trigPin = 9;
2 int echoPin = 10;
3
4 long duration;
5 int distance;
6 -----
7 void setup() {
8     pinMode(trigPin, OUTPUT);
9     pinMode(echoPin, INPUT);
10    Serial.begin(9600);
11 }
12 -----
13 void loop() {
14     digitalWrite(trigPin, LOW);
15     delayMicroseconds(2);
16     digitalWrite(trigPin, HIGH);
17     delayMicroseconds(10);
18     digitalWrite(trigPin, LOW);
19
20     duration = pulseIn(echoPin, HIGH);
21     distance= duration*0.034/2;
22
23     Serial.print("Distance: ");
24     Serial.println(distance);
25 }
```

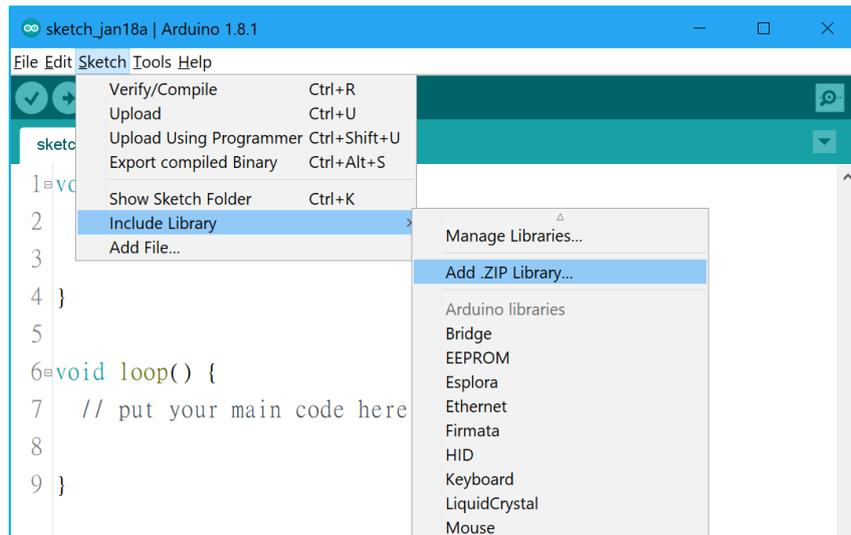
定義變數

定義超聲波感應器腳位

發射超聲波和計算距離

## IV. 動手做 – 溫濕度感應器

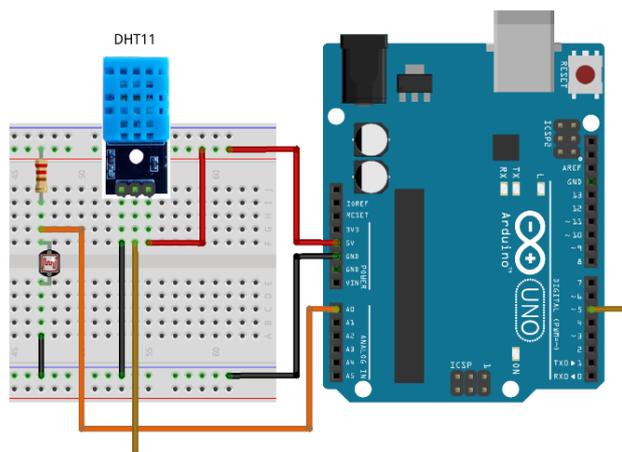
本課介紹同學這個有趣的感應器：溫濕度感應器。溫濕度感應器可以讓同學感測到環境的濕度及溫度。與之前的那些感應器不同，感謝 Arduino 開發人員在網絡上十分活躍，他們會為不同的感應器編寫函數庫。通過函數庫，同學可以很簡單的提取溫濕度感應器的數值。本課主要希望能同時處理兩種不同的訊號輸入。以下是下載函數庫的地址。<https://github.com/adafruit/DHT-sensor-library/archive/master.zip>



按下 Add .ZIP Library...，選取上述下載了的.ZIP 檔。這樣能把 DHT 的函數庫加進 Arduino IDE，接著我們便可以編寫程式了。

### 一 連接 Arduino 和溫濕度感應器

我們根據以下方法連接溫濕度感應器 DHT11 和 Arduino。



## 二 使用 Arduino 編程量度溫度及濕度

我們打開 Arduino IDE，並把 Arduino 連接電腦。在工具裡選擇 Arduino UNO 為開發板，並檢查序列埠是否已連接 Arduino。連接 Arduino 後，我們需要輸入以下程式，並把程式上傳到 Arduino。

光敏電阻及溫濕度感應器測量的程式碼全圖，可參考檔案  
[2\\_2\\_2\\_photoreis\\_temp\\_humi\\_sensor](#)

```
1 #include <dht.h>
2
3 int photoresistor_pin = A0;
4 int dht11_pin = 5;
5
6 dht DHT;
7
8 int light_result = 0;
9 int light_threshold = 300;
10 float temp_result = 0;
11 float temp_threshold = 28;
12
13 void setup() {
14     pinMode(photoresistor_pin, INPUT);
15     Serial.begin(9600);
16 }
17
18 void loop() {
19     light_result = analogRead(photoresistor_pin);
20     Serial.begin("Current light intensity is: ");
21     if(light_result >= light_threshold){
22         Serial.println("dark");
23     } else {
24         Serial.println("bright");
25     }
26
27     DHT.read11(dht11_pin);
28     temp_result = DHT.temperature;
29     Serial.print("Current temperature is: ");
30     if(temp_result >= temp_threshold) {
31         Serial.println("hot");
32     } else {
33         Serial.println("cold");
34     }
35
36     delay(1000);
37 }
```

包含 DHT 函數庫、定義變數及 DHT 物件

開啟串行通訊及定義腳位運件模式

重覆讀取光敏電阻的數值，比較我們所設立的值，作出判定

重覆讀取溫濕度感應器的數值，比較我們所設立的值，作出判定

暫停 1 秒讓數據變得易讀

### 三 程式內容說明

1. 包含 DHT 函數庫、定義變數及 DHT 物件。

```
1 #include <dht.h>
2
3 int photoresistor_pin = A0;
4 int dht11_pin = 5;
5
6 dht DHT;
7
8 int light_result = 0;
9 int light_threshold = 300;
10 float temp_result = 0;
11 float temp_threshold = 28;
```

第一個部分包含函數庫 `dht.h`，這個函數庫能讓我們用簡單的函數提取溫濕感應器的數值。其次定義 `photoresistor pin` 及 `dht11 pin`，這種定義的方法能讓程式變得更容易明白和編程。下一句是創建了名叫 `DHT` 的物件，這個物件會在之後用作存取溫濕感應器的數值。接著 `result` 的兩個變數是用作儲存程式在運行中所收到感應器的數值。`threshold` 是指臨界點，若結果的數值比臨界點大，程式會根據量度的結果進行動作。

2. 開啟串行通訊及定義腳位運作模式。

```
13 void setup() {
14   pinMode(photoresistor_pin, INPUT);
15   Serial.begin(9600);
16 }
```

第二部分連接光敏電阻的腳位，並定義為輸入。開啟串行通信頻道。

3. 重覆讀取光敏電阻的數值，比較我們所設立的值，作出判定。

```
18 void loop() {
19   light_result = analogRead(photoresistor_pin);
20   Serial.begin("Current light intensity is: ");
21   if(light_result >= light_threshold){
22     Serial.println("dark");
23   } else {
24     Serial.println("bright");
25   }
```

在重覆的循環內，光敏電阻的結果會放在 `light result`。若結果大於臨界點會列印出 `dark`；臨界點小於結果則會列出 `bright`。

4. 重覆讀取溫濕度感應器的數值，比較我們所設立的值，作出判定。

```
27 | DHT.read11(dht11_pin);
28 | temp_result = DHT.temperature;
29 | Serial.print("Current temperature is: ");
30 | if(temp_result >= temp_threshold) {
31 |     Serial.println("hot");
32 | } else {
33 |     Serial.println("cold");
34 | }
```

在重複的循環內，溫濕感應器的結果會放在 `temp result`。若結果大於臨界點會列印出 `hot`；結果小於臨界點則會列出 `cold`。

5. 暫停 1 秒讓數據變得易讀。

```
36 | delay(1000);
37 | }
```

可以開啟串行監控視窗觀看以上結果。

## V. 溫習問題

完成本課後你能夠回答以下的問題嗎？

1. 什麼是超聲波感應器？
2. 什麼是溫濕度感應器？
3. 你能夠利用 Arduino IDE 顯示超聲波感應器訊號嗎？
4. 你能夠利用 Arduino IDE 顯示溫濕度感應器訊號嗎？

## 章節三：機械人的輸出元件

### I. 輸出

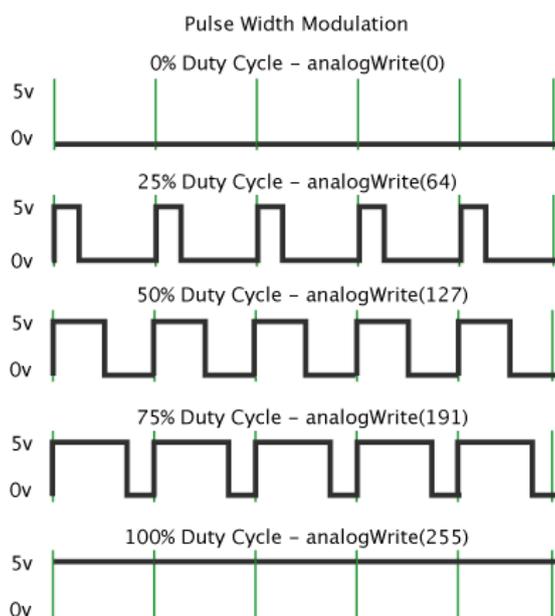
機械人是由許多不同的功能所組成，如輸入、輸出、控制、記憶及運算邏輯等單元。因為電腦是一台機器，它可以用來接受資料（輸入），再處理成有用的資訊（輸出），並將其儲存到其他地方。

### II. 數碼輸出

輸出裝置是能夠根據程式所設定的電腦訊號而改變動作的電子元件，電腦訊號一般分為**數碼輸出**和**類比輸出**兩種。所謂輸出是一個固定的電子訊號，假如電壓為 5 伏特的話，一段數碼輸出電子訊號只會出現最大值伏特和最小值伏特，因此信號比較明顯和容易用編程來控制。我們試想想在輸入一個高電位一秒，然後輸入一個低電位一秒，只用到三條函數分別是“digitalWrite (OUTPUT, HIGH)”、“digitalWrite (OUTPUT, LOW)”和“delay(1000)”，加上適當的流程控制，便能產生複雜的數碼輸出訊號。

### III. 類比輸出與脈寬調變 (Pulse Width Modulation, PWM)

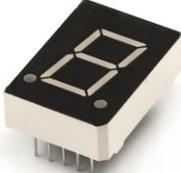
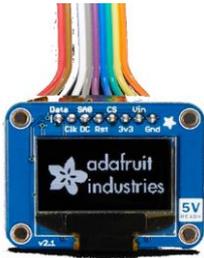
相反，假如我們想用類比(analog) 輸出訊號，現階段並不能達到，因為電腦本身也不能產生類比輸出。因此，我們的解決方法是利用一種稱為**脈沖寬度調變信號的(PWM)**技術。



這種訊號是數碼輸出的一種，讓我們在電腦中模擬類比訊號。簡單來說，這包含了兩個主要的參數：一個是工作週期，另一個是頻率。工作週期是指訊號中層產生一個高電位所佔的時間和完成一個信號週期的比例；而頻率即是能夠最快用多少時間完成一個週期。根據上圖，綠線代表高電位（5 伏特）且綠線和綠線之間為一個工作週期。因為上圖定義的 `AnalogWrite()` 的範圍只能有 0 至 255，所以 `AnalogWrite(64)` 代表用了百分之二十五高電位的工作週期。藉著這種一定頻率的數碼輸出高低的變化，再加上自定義的工作週期，該輸出便能在電腦呈現出一個固定電壓的類比輸出訊號。

以產生 2.5 伏特作為例子，我們可以用這種技術去定義一個工作周期 50% 的 5 伏特訊號， $5 \times 0.5 = 2.5$ ，便會變成 2.5V。這樣的話，我們就模擬了一個類比訊號。這種技術的用途相當廣泛，對於上一單元的發光二極管、蜂鳴器和馬達都非常有用處例如利用脈沖寬度調變信號來實行馬達精準地轉 90 度，能夠為機械人做出更加準確的動作。

以下是常見輸出的列表。一般輸出只有數碼訊號，但亦可以利用 PWM 模擬類比輸出，所以其實所有輸出同樣可以做到數碼或類比輸出，純粹只是在編程上有分別。

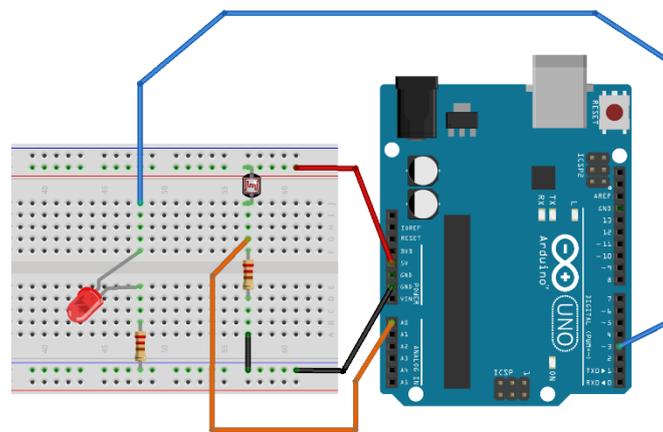
	發光二極管		步進馬達
	蜂鳴器		七段發光二極管
	直流馬達		有機發光二極管
	伺服馬達		揚聲器

## IV. 動手做 – LED 漸變燈

本課任務會嘗試透過 Arduino 控制 LED 漸變燈。LED 漸變燈能夠使用光敏電阻感應外在環境的光度，從而控制 LED 燈自身的亮度。環境光度越光，LED 便會越暗；環境光度越暗，LED 便會越光。這次任務共分為兩個部分，分別是 LED 漸變燈的電路設計與連接和使用 Arduino 編程控制 LED。

### 一 LED 漸變燈的電路設計與連接

我們根據以下方法連接 LED、電阻、光敏電阻和 Arduino。



這次任務的接駁方法簡單。首先，紅色 LED 接駁 Arduino 的方法與 LED 閃光燈任務一樣。我們先把一盞紅色 LED 安插在麵包板上，然後把 LED 燈的正極連向 Arduino 的 3 號腳位，並將負極連向一個電阻後再連接到 Arduino 的 GND。請留意，Arduino 3 號腳位旁邊有一個「~」符號標示，這個「~」符號代表 3 號腳位能夠支援脈沖寬度調變信號 (PWM) 訊號輸出。光敏電阻的連接方法使用電壓分配定則 (Voltage Divider) 的原則來接駁，光敏電阻的一端會以串聯方式連接一個電阻，另一端會連接 Arduino 的 5V 電源，而電阻的另一端則會連接 Arduino 的 GND。在光敏電阻和電阻之間，我們需要使用一條電線把這個位置連接向 Arduino 的 A0 腳位。當環境光度越光，光敏電阻的電阻率就會越細，令 A0 腳位的電壓提高；當環境光度越暗，光敏電阻的電阻率就會越大，令 A0 腳位的電壓減少。通過判斷 A0 腳位電壓的數值，我們能夠感應外在環境的光度，控制 LED 燈的光暗。

## 二 使用 Arduino 編程控制 LED

首先打開 Arduino IDE，並把 Arduino 連接電腦。在工具裡選擇 Arduino UNO 為開發板，並檢查序列埠是否已連接 Arduino。連接 Arduino 後，我們需要輸入以下程式，並把程式上傳到 Arduino。

LED 漸變燈的程式碼全圖，可參考檔案 2\_3\_analog\_led

```
1 int led_pin = 3;
2 int photores_pin = A0;
3
4 int led_result = 0;
5 int photores_result = 0;
6 -----
7 void setup() {
8   pinMode(led_pin,OUTPUT);
9   pinMode(photores_pin, INPUT);
10 }
11 -----
12 void loop() {
13   photores_result = analogRead(photores_pin);
14   led_result = map(photores_result,0,1023,255,0);
15   analogWrite(led_pin,led_result);
16 }
```

### 1. 定義變數。

```
1 int led_pin = 3;
2 int photores_pin = A0;
3
4 int led_result = 0;
5 int photores_result = 0;
```

第一部分我們分別定義變數“led\_pin”和“photores\_pin”為整數 3 和 A0。這兩個變數代表 LED 和光敏電阻連向 Arduino 的腳位位置。其次，我們也定義變數“led\_result”和“photores\_result”，並預設它們為整數 0。這兩個變數代表 LED 的光度和光敏電阻的感應數值。

### 2. 定義 LED 和光敏電阻的腳位。

```
7 void setup() {
8   pinMode(led_pin,OUTPUT);
9   pinMode(photores_pin, INPUT);
10 }
```

第二部分，我們定義 LED 和光敏電阻連接 Arduino 的腳位。上一部分變數“led\_pin”和“photores\_pin”定義為整數 3 和 A0，這部分我們便使用“pinMode()”把 3 號腳位定義為輸出腳位，A0 腳位定義為輸入腳位。

3. 根據環境光度改變 LED 的亮度。

```
12 void loop() {  
13   photores_result = analogRead(photores_pin);  
14   led_result = map(photores_result,0,1023,255,0);  
15   analogWrite(led_pin,led_result);  
16 }
```

這一部分，我們使用“analogRead()”來閱讀在 A0 腳位的類比訊號輸入，並把閱讀的數值儲存在變數“photores\_result”當中。由於“photores\_result”所儲存的數值範圍是由 0 到 1023，這個數值範圍對我們控制 LED 來說範圍過大，因此我們用到“map()”這個功能把 0 到 1023 這個範圍縮小到 0 至 255。因為環境光度和 LED 的亮度是成反比關係，所以 0 到 1023 這個數值範圍應該變成 255 至 0 這個範圍才是合理。最後，我們使用“analogWrite()”把調整後的光度輸出到 LED，LED 燈的亮度便會根據環境的亮度而改變。

## V. 溫習問題

完成本課後你能夠回答以下的問題嗎？

1. 什麼是數碼輸出？
2. 什麼是脈寬調變（PWM）技術？

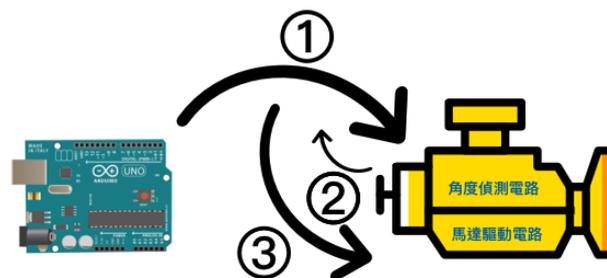
## 章節四：伺服馬達

### I. 認識伺服馬達

簡單來說，伺服馬達就是一個直流馬達再加上一個偵測馬達轉動角度的電路，以及一組齒輪去進行控制角位置。伺服馬達一般比直流馬達大一點，而且還有三條連接線，分別是**正電源**、**接地**和**數據線**。數據線提供一個偵測角度的電流，我們可以利用頻寬調節訊號技術來控制伺服馬達準確地轉出特定的角度。下圖是伺服馬達：



馬達的流程如下：當電流經過馬達後，會回饋一個角度的讀數。根據讀數和我們編程上的數值差異調整電壓大小，直至回饋的角度與定立的角度相符。



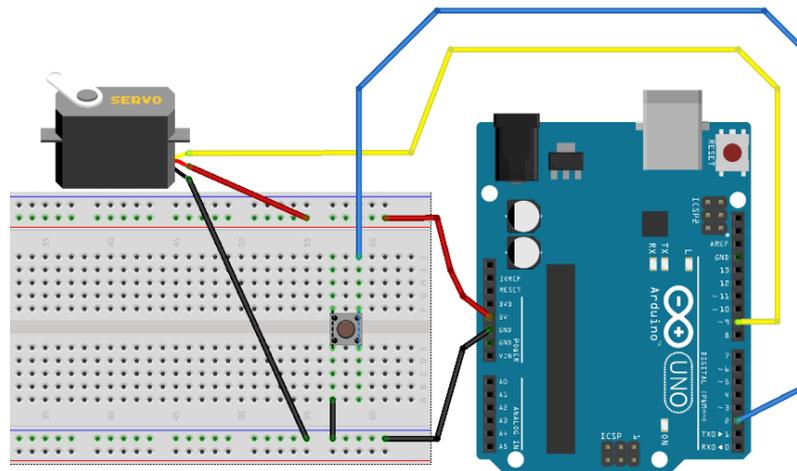
1. 角度偵測
2. 傳回角度訊號
3. 馬達驅動電流調節

## II. 動手做 – 控制伺服馬達

本課希望同學能透過使用按鈕來控制伺服馬達的動作，而本次的任務是當同學按下按鈕時，伺服馬達會慢慢打開，然後等待五秒，伺服馬達接著便會關上。若同學希望更易使用伺服馬達，Arduino IDE 中有內置函數庫 <Servo.h> 供同學使用。

### 一 連接 Arduino、按鈕和伺服馬達

我們根據以下方法連接 Arduino、按鈕和伺服馬達。



## 二 使用 Arduino 編程控制伺服馬達

我們打開 Arduino IDE，並把 Arduino 連接電腦。在工具裡選擇 Arduino UNO 為開發板，並檢查序列埠是否已連接 Arduino。連接 Arduino 後，我們需要輸入以下程式，並把程式上傳到 Arduino 當中。

控制伺服馬達的程式碼全圖，可參考檔案 2\_4\_control\_servo\_motor

```
1 #include <Servo.h>
2
3 int button_pin = 2;
4 int servo_pin = 9;
5 int pos = 0;
6
7 Servo servo;
8 -----
9 void setup() {
10   servo.attach(servo_pin);
11   servo.write(0);
12   pinMode(button_pin, INPUT_PULLUP);
13 }
14 -----
15 void loop() {
16   if(digitalRead(button_pin) == LOW){
17     delay(20);
18     turn_servo_180_cw();
19     delay(5000);
20     turn_servo_180_ccw();
21   }
22 }
23 -----
24 void turn_servo_180_cw(){
25   for (pos = 0; pos <= 180; pos += 1) {
26     servo.write(pos);
27     delay(5);
28   }
29 }
30
31 void turn_servo_180_ccw(){
32   for (pos = 180; pos >= 0; pos -= 1) {
33     servo.write(pos);
34     delay(15);
35   }
36 }
```

包含 Servo 函數庫、定義變數及 Servo 物件

設定伺服馬達所使用的腳位及設定腳位模式

感應程式

自行定義轉動函數

### 三 程式內容說明

1. 第一部分包含函數庫、定義變數名稱以及創建伺服馬達物件。

```
1 #include <Servo.h>
2
3 int button_pin = 2;
4 int servo_pin = 9;
5 int pos = 0;
6
7 Servo servo;
```

2. 第二部分把創建了的伺服馬達物件和伺服馬達腳位連在一起，並把伺服馬達的角度轉為零。按鈕的腳位模式為輸入上拉電阻，同學請留意接駁電路時不用加上電阻，但是在程式上腳位模式必須選擇 INPUT\_PULLUP。

```
9 void setup() {
10   servo.attach(servo_pin);
11   servo.write(0);
12   pinMode(button_pin, INPUT_PULLUP);
13 }
```

3. 第三部分程式不停探測按鈕腳位有沒有低電壓的情況出現，有的話便會觸發開門的動作，然後在五秒後馬達會以逆時針（Counter-clock Wise, CCW）轉 180 度，並重複這個動作。

```
15 void loop() {
16   if(digitalRead(button_pin) == LOW){
17     delay(20);
18     turn_servo_180_cw();
19     delay(5000);
20     turn_servo_180_ccw();
21   }
22 }
```

4. 第四部分是自行定義以順時針（Clockwise, CW）轉 180 度及以逆時針（Counter-clockwise, CCW）轉 180 度的函數。為了讓伺服馬達能慢慢地轉動，我們需要在程式中每一個角度轉動時加上 delay() 函數，以減慢馬達的轉速。

```
24 void turn_servo_180_cw(){
25   for (pos = 0; pos <= 180; pos += 1) {
26     servo.write(pos);
27     delay(5);
28   }
29 }
30
31 void turn_servo_180_ccw(){
32   for (pos = 180; pos >= 0; pos -- 1) {
33     servo.write(pos);
34     delay(15);
35   }
36 }
```

### III. 溫習問題

完成本課後你能夠回答以下的問題嗎？

1. 什麼是伺服馬達？
2. 你能如何控制伺服馬達呢？

## 章節五：單元專題研習 – 清潔機器人

清潔及整潔的形象在文明社會是十分重要的，可是清潔的工作一般比較厭倦和重複，似乎不一定要人去做。因此，我們想以機械人的角度嘗試去解決問題。

機器人學中包含了不同的元素，如科學、科技、工程等。同學進行單元專題研習時，需要利用不同的學科知識，例如一開始你們需要由科學研究做起，找出如何利用科學去找出不同的污漬和相應的清潔方式。其次你們需要構思設計，如何利用單元一所學到的電子元件去設計一個機械人。製造完原型之後，需要進行測試。最後，你們會根據測試結果去改良你們的設計。在進行專題研習中，期望同學可以得到更多科學知識和工程知識，亦希望透過與人合作，增強溝通技巧，解難能力和領導能力。

工程設計階段／步驟	相關知識	探究和設計考慮	應用相關知識和探究結果在工程設計上
界定問題 (確認要求和限制)	瞭解家中日常地板清潔狀況，並提出打掃家務的長短處，以指出清潔機械人的可行性		釐定明確的設計、要求和限制 (參考資料一的例子)
研究	<ul style="list-style-type: none"> <li>清潔家居的方法</li> <li>使用不同布料對清潔的影響</li> <li>一般清潔使用的物料</li> <li>清潔家居的路線，能讓機械人不會撞上傢俱</li> <li>甚麼傳感器能讓清潔機械人完成工作</li> </ul>		選擇處理方法 (如在課堂上進行，可選一項或多項，視乎教學目標、課時及材料限制而定)
構想設計	研究資料顯示，超聲波感應器能有效感測傢俱	預測：超聲波感應器能讓機械人躲過障礙物	先使用簡單的程式測試感應器及機械人的運作
測試模型	<ul style="list-style-type: none"> <li>傳感器安裝的位置對清潔機械人效能影響</li> <li>布料的安放位置對清潔的影響</li> </ul>	超聲波感應器安放於機械人的不同位置，進行測試並量度結果  公平測試方法： <ul style="list-style-type: none"> <li>找出感應器安裝位置如何影響清潔效率</li> <li>獨立變數：安裝位置</li> </ul>	

		<ul style="list-style-type: none"> <li>· 因變數：清潔效果</li> <li>· 控制變數：傢俱位置、馬達位置及速度等</li> </ul>	
解決設計／製作／測試過程中遇到的問題		程式必須讓機械人能覆蓋家中的每個角落才能把家居打掃乾淨	將這些發現納入為是項工程設計的必要程序
分析及評估測試結果及出現的問題	隨機的結果	<ul style="list-style-type: none"> <li>· 利用隨機安排機械人行走，並評估設計是否已達到既定目標</li> <li>· 但機械人行走的路線不能預測</li> </ul>	參考分析及評估所得，研究改良方法
改良	除了安裝單個超聲波感應器外，亦可安裝多個不同感應器，幫助機械人尋找路線	利用電腦視覺及人工智能改善及尋找路線，以此增加清潔機械人的覆蓋率	一起探討現今科技是如何制作清潔械人

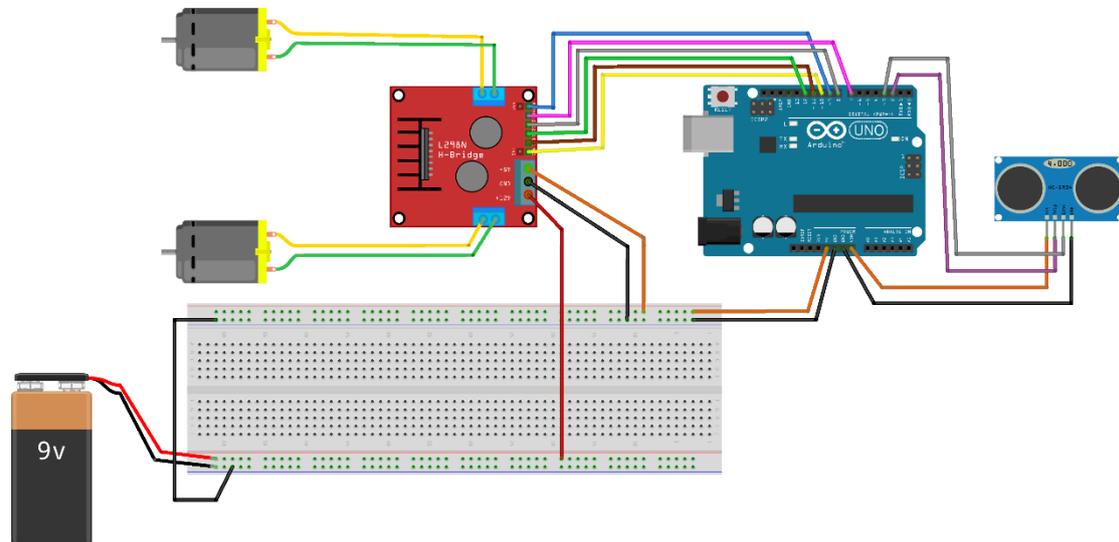
本課任務會整合過去所學的知識，運用超聲波感應器、馬達、馬達控制模組和 **Arduino** 來製作清潔機械人。這次任務共分為三個部分，當中包括：清潔機械人車身結構製作、機械人電路製作和機械人程式編寫。

這次任務的清潔機械人以簡單結構為主，務求讓每一位同學就算使用普通材料都能夠容易製作清潔機械人。清潔機械人主要由雪條棍做成，讓機械人的重量較輕，行走更方便。機械人的行動方式由兩個馬達推動，馬達由馬達控制模組所控制，前方的萬向輪令機械人更易轉變方向。清潔機械人的清潔物料會放置在萬向輪的前方，當機械人移動時會帶動清潔物料，清潔機械人經過的地方。由於 **Arduino** 的輸出電壓和輸出電流只有 5V 和 40mA，所以要推動機械人，我們需要額外加上 9V 電池為機械人提供大約 9V 電壓，機械人才會運作。

清潔機械人運作時，機械人會隨機移動清潔地面，直到超聲波感應器感應前方有障礙物少於 20cm，機械人便會後退，再以隨機方式清潔其他地方。

## 一 機械人電路製作

我們根據以下方法連接超聲波感應器、馬達、馬達控制模組、9V 電池和 Arduino。完成後我們將會把電子零件安裝在清潔機械人的底盤上。



## 二 清潔機械人車身結構製作

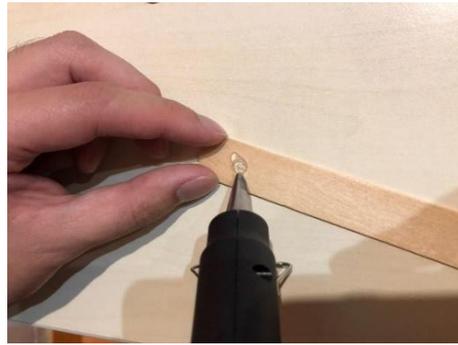
同學可根據自己的構思，為你的清潔機械人設計及製作其外殼，用以放置 Arduino 板及其他電子原件。以下是製作清潔機械人底盤的參考：

需要工具：13 條雪條棍、熱溶膠、萬向輪、超聲波感應器、直流馬達、馬達控制模組、車輪、Arduino、電線、電池盒、9V 電池、清潔材料和車蓋。

1. 拿出 8 條雪條棍，並且排列在一起



2. 使用熱溶槍塗滿雪條棍



3. 塗滿後的雪條棍



4. 把它壓在整排雪條棍，並固定雪條棍



5. 最後加上斜的雪條棍作固定



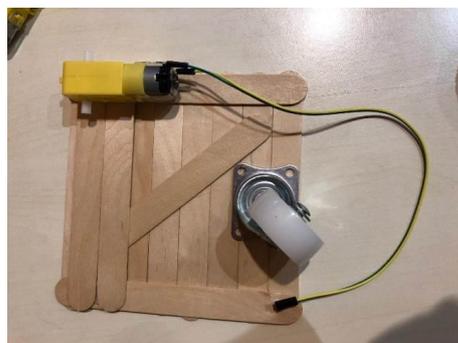
6. 使用熱溶膠槍把萬向輪貼在雪條棍上



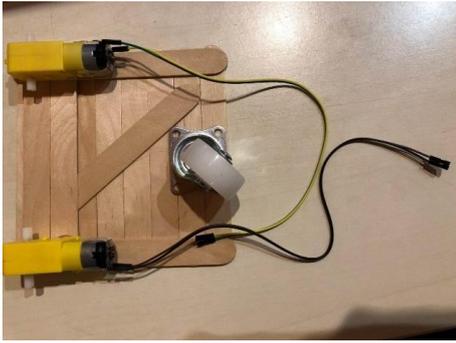
7. 在後方貼上 2 條雪條棍加高位置



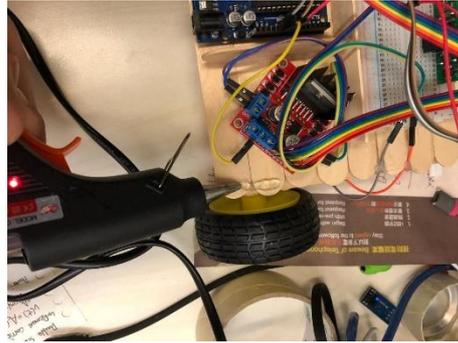
8. 把黃色馬達貼在機械人的後方



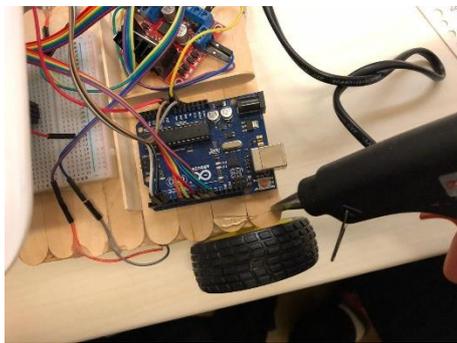
9. 重覆以上動作



10. 在機械人左邊馬達旁的木條塗熱溶膠



11. 在機械人右邊馬達旁的木條塗熱溶膠



12. 先把塑膠蓋剪開並塗熱溶膠



13. 最後替機械人加上後蓋



14. 在鋁罐中心打一個孔。



15. 利用鐵線固定鋁罐並使其能轉動。



16. 加上清潔用具例如吸塵紙於鋁罐上。



### 三 機械人程式編寫

我們打開 Arduino IDE，並把 Arduino 連接電腦。在工具裡選擇 Arduino UNO 為開發板，並檢查序列埠是否已連接 Arduino。連接 Arduino 後，我們需要輸入以下程式，並把程式上傳到 Arduino。

清潔機械人的程式碼全圖，可參考檔案 2\_5\_cleaning\_robot

```
1 const int motorENA = 10;
2 const int motorIN1 = 11;
3 const int motorIN2 = 12;
4 const int motorENB = 9;
5 const int motorIN3 = 8;
6 const int motorIN4 = 7;
7
8 const int trigPin = 2;
9 const int echoPin = 3;
10
11 byte LwheelSpeed = 100;
12 byte RwheelSpeed = 100;
13 bool LwheelisForward = true;
14 bool RwheelisForward = true;
15
16 long duration;
17 int distance;
18 -----
19 void setup(){
20   pinMode(motorENA, OUTPUT);
21   pinMode(motorIN1, OUTPUT);
22   pinMode(motorIN2, OUTPUT);
23   pinMode(motorENB, OUTPUT);
24   pinMode(motorIN3, OUTPUT);
25   pinMode(motorIN4, OUTPUT);
26   pinMode(trigPin, OUTPUT);
27   pinMode(echoPin, INPUT);
28   randomSeed(analogRead(0));
29   Serial.begin(9600);
30 }
31 -----
32 void loop(){
33   motor(LwheelSpeed, LwheelisForward, RwheelSpeed, RwheelisForward);
34   debug();
35
36   LwheelSpeed = random(0,127);
37   RwheelSpeed = random(0,127);
38
39   if(ultrasonicDetect(<20){
40     motor(random(0,100),0,random(0,200),0);
41     delay(500);
42   }
43 }
```

定義變數

定義超聲波感應器和馬達  
控制模組的連接腳位

控制機械人移動

```

45 void debug(){
46   Serial.print("L: ");
47   Serial.print(LwheelSpeed);
48   Serial.println(LwheelisForward);
49   Serial.print("R: ");
50   Serial.print(RwheelSpeed);
51   Serial.println(RwheelisForward);
52 }
53 -----
54 void motor(byte leftSpeed, bool isLForward, byte rightSpeed, bool isRForward){
55
56   if(isLForward){
57     digitalWrite(motorIN1, HIGH);
58     digitalWrite(motorIN2, LOW);
59   }else{
60     digitalWrite(motorIN1,LOW);
61     digitalWrite(motorIN2,HIGH);
62   }
63   analogWrite(motorENA, leftSpeed);
64
65   if(isRForward){
66     digitalWrite(motorIN3, HIGH);
67     digitalWrite(motorIN4, LOW);
68   }else{
69     digitalWrite(motorIN3,LOW);
70     digitalWrite(motorIN4,HIGH);
71   }
72   analogWrite(motorENB, rightSpeed);
73 }
74 -----
75 int ultrasonicDetect(){
76   digitalWrite(trigPin, LOW);
77   delayMicroseconds(2);
78   digitalWrite(trigPin, HIGH);
79   delayMicroseconds(10);
80   digitalWrite(trigPin, LOW);
81
82   duration = pulseIn(echoPin, HIGH);
83   distance= duration*0.034/2;
84   return distance;
85 }

```

檢查程式

設定馬達移動函數

設定超聲波感應器函數 ·  
發射超聲波和計算距離

## 四 程式內容說明

### 1. 定義變數。

```
1 const int motorENA = 10;
2 const int motorIN1 = 11;
3 const int motorIN2 = 12;
4 const int motorENB = 9;
5 const int motorIN3 = 8;
6 const int motorIN4 = 7;
7
8 const int trigPin = 2;
9 const int echoPin = 3;
10
11 byte LwheelSpeed = 100;
12 byte RwheelSpeed = 100;
13 bool LwheelisForward = true;
14 bool RwheelisForward = true;
15
16 long duration;
17 int distance;
```

在第一部分，我們定義變數分別為超聲波感應器和馬達控制模組設定連接到 Arduino 腳位的位置。每一個變數的內容都是整數。使用“const int”來定義變數的原因是確保這些變數的內容在整個程式中都不會有任何變化。我們同時預設“LwheelSpeed”和“RwheelSpeed”為整數 100，這兩個變數分別代表左邊和右邊馬達的轉動速度。至於“LwheelisForward”和“RwheelisForward”都需要預設為“true”，這兩個變數能控制左右馬達轉動的方向。“duration”負責儲存超聲波由發出到返回感應器需要的時間，而“distance”則負責儲存超聲波感應器和物件之間的距離。

### 2. 定義超聲波感應器和馬達控制模組的連接腳位。

```
19 void setup(){
20   pinMode(motorENA, OUTPUT);
21   pinMode(motorIN1, OUTPUT);
22   pinMode(motorIN2, OUTPUT);
23   pinMode(motorENB, OUTPUT);
24   pinMode(motorIN3, OUTPUT);
25   pinMode(motorIN4, OUTPUT);
26   pinMode(trigPin, OUTPUT);
27   pinMode(echoPin, INPUT);
28   randomSeed(analogRead(0));
29   Serial.begin(9600);
30 }
```

在這部分我們首先把所有 Arduino 連接馬達控制模組的腳位都定義為輸出腳位。我們定義超聲波感應器的“trigPin”為輸出腳位，因為 Arduino 會透過這個腳位向超聲波感應器傳送訊號，啟動超聲波感應器。而“echoPin”會被定義為輸入腳位，因為超聲波感應器需要向 Arduino 輸入訊號，計算超聲波發出和返回的時間。由於需要

機械人隨機移動，所以我們使用 “randomseed(analogRead(0))” 來製造隨機數字。“Serial.begin(9600)” 則設定序列埠連接速率，方便 Arduino 與電腦進行通訊，在序列埠監控視窗上顯示數據。

### 3. 設定馬達移動函數。

```
54 void motor(byte leftSpeed, bool isLForward, byte rightSpeed, bool isRForward){
55
56     if(isLForward){
57         digitalWrite(motorIN1, HIGH);
58         digitalWrite(motorIN2, LOW);
59     }else{
60         digitalWrite(motorIN1,LOW);
61         digitalWrite(motorIN2,HIGH);
62     }
63     analogWrite(motorENA, leftSpeed);
64
65     if(isRForward){
66         digitalWrite(motorIN3, HIGH);
67         digitalWrite(motorIN4, LOW);
68     }else{
69         digitalWrite(motorIN3,LOW);
70         digitalWrite(motorIN4,HIGH);
71     }
72     analogWrite(motorENB, rightSpeed);
73 }
```

第三部分的程式是一個馬達移動函數。當主程式使用這個函數時，馬達移動函數便會根據輸入控制左右馬達轉動的方向和速度。請留意，要控制馬達速度，我們可以使用 “analogWrite()” 製作一個 PWM 訊號給 “motorENA” 或 “motorENB” 腳位。

### 4. 設定超聲波感應器函數，發射超聲波和計算距離。

```
75 int ultrasonicDetect(){
76     digitalWrite(trigPin, LOW);
77     delayMicroseconds(2);
78     digitalWrite(trigPin, HIGH);
79     delayMicroseconds(10);
80     digitalWrite(trigPin, LOW);
81
82     duration = pulseIn(echoPin, HIGH);
83     distance= duration*0.034/2;
84     return distance;
85 }
```

第四部分的程式是一個超聲波感應器函數，函數會透過超聲波感應器計算物件距離，然後把距離數值傳輸到主函數加以分析。函數中主要通過 “trigPin” 來啟動超聲波感應器，繼而使用 “echoPin” 來量度超聲波由感應器發出到經物件反彈後反射回超聲波感應器的時間，再計算距離。由於函數需要返回數據給主函數，因此在函數的末端，我們使用 “return distance” 來返回儲存在變數 “distance” 中的數據。

5. 控制機械人移動。

```
32 void loop(){
33   motor(LwheelSpeed, LwheelisForward, RwheelSpeed, RwheelisForward);
34   debug();
35
36   LwheelSpeed = random(0,127);
37   RwheelSpeed = random(0,127);
38
39   if(ultrasonicDetect(<20){
40     motor(random(0,100),0,random(0,200),0);
41     delay(500);
42   }
43 }
```

第五部分是整個程式的核心，也是程式中的主函數。首先，我們使用馬達轉動函數來設定機械人移動的方向。為了令機械人隨機移動，我們使用“random()”來改變變數“LwheelSpeed”和“RwheelSpeed”的數值，讓左邊和右邊馬達轉動的速度不一樣。接著，我們使用超聲波感應器量度距離。當超聲波感應器量度的距離少於 20cm 時，清潔機械人便會後退，然後再清潔其他地方。

6. 檢查程式。

```
45 void debug(){
46   Serial.print("L: ");
47   Serial.print(LwheelSpeed);
48   Serial.println(LwheelisForward);
49   Serial.print("R: ");
50   Serial.print(RwheelSpeed);
51   Serial.println(RwheelisForward);
52 }
```

第六部分是檢查程式。我們在序列埠監控視窗上列印左右馬達的轉動方向和速度，檢查程式運行途中有沒有發生意外或出現我們意想不到的問題。

7. 完成程式後，我們便能為清潔機械人安裝電池，機械人會在地上四處走動，幫助我們清潔。