

全國高級中等學校專業群科 107 年專題及創意製作競賽  
「專題組」作品說明書



群別：電機電子群

作品名稱：PS2 無線遙控教導式機械手臂

關鍵詞：無線遙控、教導式、機械手臂

## 目錄

壹、摘要.....	2
貳、研究動機.....	2
參、主題與課程之相關性或教學單元之說明.....	2
肆、研究方法.....	3
一、研究設備與材料.....	3
二、研究過程.....	6
伍、研究結果.....	10
一、軟體說明.....	10
二、馬達說明.....	12
三、動作流程.....	13
陸、討論.....	14
柒、結論.....	15
捌、參考資料.....	15

## 壹、摘要

PS2 無線遙控教導式機械手臂，透過流暢的自動學習系統，提升機械手臂流程修改上的效率，本團隊利用整合 PS2 無線遙控模組與 LCD 顯示器，做出簡單的人機控制介面，PS2 控制器用來操控機械手臂的所有動作，LCD 則用以顯示動作資訊，使用者便能夠更簡單地對機械手臂進行操控與教導，讓功能完善的自組機械手臂自動化做出任何使用者教導的動作流程，令其持續工作，當然也能隨時教導新的動作流程。而使用者也能夠在設定流程時選擇自動流程所需的次數，模擬工廠自動設備的工作要求。

## 貳、研究動機

在現代的製造業裡，以機器取代人力已經成為主流。而機械手臂的應用也是其大宗。機械手臂是具有模仿人類手臂功能並可完成各種作業的自動控制設備，且能一機多用，不限於單一種工作。另外，可程式化的控制也可以減少純人工的失誤。

但是若純粹只以固定的程式動作一套流程並不完全便利，想要進行其他動作的話就需要編寫新的整套程式。於是本團隊便想以人為的一次操作來對機器進行教導，製造出可教導式的機械手臂，透過單次的學習來使機器能夠無限反覆進行學習到的動作流程，流程完全仿照使用者設定操作的樣子，且能夠隨時重新教導新的動作，省下編寫新的程式所需的耗費。而方便的遙控操作與顯示介面也能讓使用者在對機器進行教導時更容易上手。

## 參、主題與課程之相關性或教學單元之說明

在高二的課程中，我們學習到 PLC 可程式控制來完成機電整合，而本學期有了 Arduino 及 8051 的軟體課程。由於機電整合五站程序中就多達四站的搬運程序，於是本團隊最初的想法是如何讓搬運工程減少並將其融為一體，最後則是透過讓一隻機械手臂來達到整合全部之搬運工作。軟體部份本團隊之所以從三者中選出 Arduino 使用，原因為使用 PLC 可程式控制器需由多顆感測器來完成搬運工程，此方法無法減少機台的需求量，並無法達成本團隊之需求，而 8015 及

Arduino 則是因為 8015 的函式庫需全部親自完成，相較 Arduino 而言編輯所需時間又更長，最後便選擇了用 Arduino 控制機械手臂來達成將多個搬運站合而為一之目的。

## 肆、研究方法

### 一、研究設備與材料

(表 1)研究材料

項目	零件	
機械手臂主體	機器人零組件	多功能支架
		長U支架
		L支架
		U型樑板
	MG99X系列	MG996R伺服馬達 13KG單軸承 180度
		MG995伺服馬達 20KG雙軸承 180度
	機械夾爪	
	彈簧	
人機介面	PS2無線遙控模組	
	LCD液晶顯示器	
	I2C串列通訊匯流排	
供電	電源線	
	AC/DC 5V10A變壓器	
	5.5x2.1mm 電源充電供電接口	
程式控制	Arduino軟體	
	Arduino Mega 2560板	
	PS2X無線遙控模組接收器接線圖	
其他	麵包板	
	電木底座	

(一) MG995 伺服馬達：本專題採用雙軸承 MG995 伺服馬達以提高機械穩定度。



(圖 1) MG995 伺服馬達

(二) PS2 手把：操控所有動作。

(三) PS2 無線接收器：接收搖桿發出的訊息。



(圖 2) PS2X 無線遙控模組

(四) LCD 顯示器：顯示當前的動作資訊。



(圖 3) LCD 顯示器

(五) I2C 串列通訊匯流排：將 LCD 的所有腳位濃縮至四支。



(圖 4) I2C

(六) Arduino：是一個開放原始碼的單晶片微控制器，它使用了 Atmel AVR 單晶片，採用了開放原始碼的軟硬體平台，建構於簡易輸出/輸入 (simple I/O) 介面板，並且具有使用類似 Java、C 語言的開發環境，即使不懂電腦編程，也能用 Arduino 做出很酷的東西，比如對感測器作出回應，閃爍燈光，還能控制馬達。



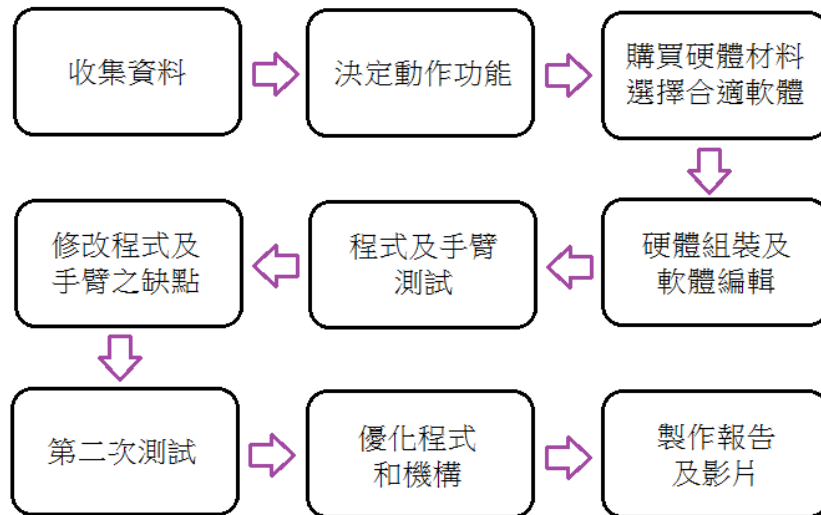
(圖 5) Arduino 程式

(七) Arduino Mega 2560 板：手臂的主要核心。接收 PS2 無線接收器發送的訊息，以控制整體機械手臂。



(圖 6) Arduino Mega 2560 板

## 二、 研究過程



(圖 7)專題製作流程

### (一) 收集資料

機械手臂是具有模仿人類手臂功能並可完成各種作業的自動控制設備，這種機器人系統有多關節連結並節允許在平面或三度空間進行運動或使用線性位移移動。構造上由機械主體、控制器、伺服機構和感應器所組成，並由程式根據作業需求設定其一定的指定動作。機器人的運作由電動機驅動移動一隻手臂，張開或關閉一個夾子的動作，並精確的回饋至可程式邏輯的控制器。這種自動裝置機械以完成「腕部以及手部」的動作為主要訴求，可以由熟練的操作者將作業順序輸入後，就能依樣照作並且反覆完成無數次的正確規律運作。

### (二) 決定動作功能

遙控：

利用方便使用者操控的介面，對機械手臂進行無線遙控。

教導式：

此手臂主要功能是依照使用者在工廠的不同需求，經由一次的操作後，讓機械手臂記住該動作，並設定動作次數，來完成後續重複性的動作，以節省人力及時間。



### （三）購買硬體材料，選擇合適軟體

硬體：

起初手臂主體本來想自己設計後裁切壓克力板，但發現壓克力不夠堅固，容易斷裂，所以找了鋁合金支架作為支撐手臂的主體。

馬達選用了MG996R伺服馬達13KG單軸承180度，

而裝設LCD目的是為了方便使用者觀察當前動作之馬達角度，並告知手臂目前狀態，以及教導式流程中，設定動作次數、需動作次數、當前完成次數。

軟體：

之所以從Arduino與8051中選擇了Arduino是因為它已經設計好接腳讓使用者使用，而且擁有簡潔的語法與眾多便利且開源的函式庫，例如PS2無線模組就有已經設定好腳位的函式庫可供使用，伺服馬達也能以伺服專用的控速函式庫做簡單的控速，諸如此類的優點變成為首選。

### （四）硬體組裝與軟體編輯

硬體：

了解每顆馬達之功能及鋁合金支架之形狀，配合當初設計之動作，從最底部的馬達依序往上拼裝，並將機械手臂調整至可測試狀態，電線整理至不會干擾測機，外觀整齊不凌亂，以便程式測機。



（圖8）手臂初步整理

軟體：

利用Arduino編寫程式給機械手臂，當中重點包含遙控控制所有馬達、初始化回預設狀態、顯示器顯示、教導準位設定、自動流程次數設定與自動流程動作。



### (五) 程式及手臂測試

#### 硬體：

用搖桿測試每顆馬達的力矩，夾爪之鬆緊程度，機械手臂動作的流暢度，運作時，電線干擾程度，以及夾取物品時手臂配重狀態。

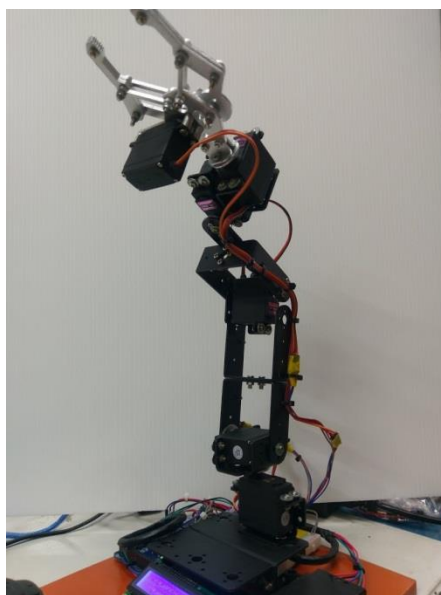
#### 軟體：

測試搖桿每顆按鈕是否依照程式設計，讓對應馬達動作，並試驗所有流程是否出錯或與要求不符，若有問題則修改。

### (六) 修改程式及手臂之缺點

#### 硬體：

機械手臂在前傾夾取物品時，會有重心不穩之情形，故在底座加上電木，以維持機械手臂之重心。



(圖 9)機械手臂加電木

#### 軟體：

1. 伺服馬達在直接給數值使其快速移動時可能出現劇烈搖晃，便需要微調馬達極限角度，並以可調速之函式庫取代原本常用的函式庫。
2. 修正程式語法使 LCD 出現亂碼的錯誤。
3. 微調使用者設定自動流程時，次數設定的緩衝時間。

## （七）第二次測試

硬體：

經過改良後，確實解決了之前的問題，使得這次測機更為順利，但部分時候馬達還是稍嫌無力。

軟體：

原本使用陣列來進行自動流程時，只能將手臂從設定的起始點移至終點，馬達動作順序與設定並不相同，且可能有撞機之風險，於是改採用二維陣列來判別遙控設定時的馬達動作順序，雖然花了點時間，但確實能達到完美的複製效果。

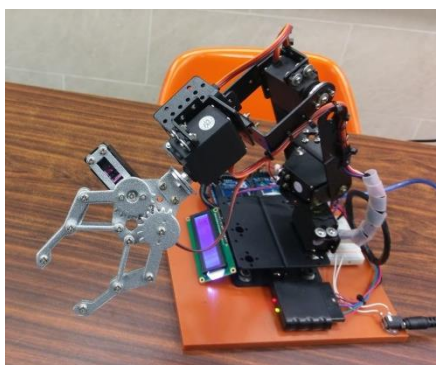
## （八）優化程式和機構

硬體：

在完成機械手臂組裝後，為了使手臂活動更為順暢，我們將原來的 13KG 996R 單軸承馬達換為 20KG 的 995 雙軸承馬達，來提供更大的轉矩與更為穩固的穩定度，並且把所用之電線，一樣以不干擾動作為前提，將它重新地整齊繞至於手臂及電木上，再更進一步加強整體的美觀。



（圖 10）手臂側面圖



（圖 11）手臂全圖



（圖 12）手臂後視圖

軟體：

自動流程判別改採用二維陣列後，的確能較完美的複製使用者操作的動作順序，所以再經過更細小的微調檢查後，程式最終完成了。

## 伍、研究結果

### 一、軟體說明

以 Arduino 編寫程式，整體程式太過繁雜，故不完整說明。遙控部分利用無線手把與接收器，配合 ps2x.h 的函式庫將遙控訊號傳至 Arduino，對手臂進行所有控制。當五顆不同馬達的操作鈕分別被使用時，if else 迴圈將檢視左側搖桿的數值變化使對應按鈕的馬達數值進行加減，藉此達到遙控旋轉的效果(如圖 13)，當然各馬達的極限角度也經過限制，當角度數值加減超過極限值時，強制回到極限值，以防撞機。

而若在搖控中按下左鍵，也就是動作複製的開始鍵時，motionRecord 變數將啟動(如圖 13 與圖 14)，程式將會把馬達和角度資訊寫入二維陣列，記錄所使用的馬達名稱與其開始和最終的角度，方便自動流程時取用。相反地當按下右鍵結束複製時，motionRecord 將停止，動作複製也記錄到此(如圖 14)。

```
//Servo Start
if ((ps2x.Button(PSE_RED)) && (kill168!=true)) {           //當壓著紅色圓形時動作
    if (motionRecord)                                     //當複製動作開始時
    {
        pos[k][0] = 1;                                   //編排二維陣列數值寫入位置
        if (k == 0)
        {
            pos[k][1] = angle1;                           //寫入數值
            k++;                                           //編排二維陣列數值寫入位置
        }
        else if ((pos[k][0] != pos[k - 1][0]) && ((k - 1) >= 0)) //判斷servo是否與上次使用servo重複
        {
            pos[k][1] = angle1;
            angleWrite(pos[k - 1][0]);                     //覆蓋角度數值
            k++;
        }
    }
    if (ps2x.Analog(PSS_LY) < 120) {                       //當左搖桿數值小於120時
        if (angle1 < 180)                                   //當servo1角度數值小於180
        {
            angle1 += 2;                                    //servo1角度數值+2
        }
        else
        {
            angle1 = 180;                                   //servo1角度強制等於180，防止轉超過極限角度
        }
    }
}
```

(圖 13) 局部遙控程式

---

```

//Motion Record Start
if ((ps2x.ButtonPressed(PSB_PAD_LEFT) == true) && (motionRecord == false)) //當按下左鍵且複製動作未開始時
{
    motionRecord = true; //複製動作開始
    k = 0; //設定二維陣列初值
    for (a = 0; a < 30; a++) //二維陣列共30行
    {
        for (b = 0; b < 3; b++) //二維陣列共3列
        {
            pos[a][b] = 0; //清除先前二維陣列資料
        }
        Serial.println();
    }
    Serial.println("Start set"); //監控視窗列印字串
    lcd.clear(); //LCD清除先前畫面
    lcd.setCursor(0,0); //設定顯示座標
    lcd.print("Start set"); //LCD顯示字串
}
else if ((ps2x.ButtonPressed(PSB_PAD_RIGHT) == true) && (motionRecord == true)) //當按下右鍵且複製動作已開始時
{
    motionRecord = false; //複製動作結束
    pos[k][0] = 0;
    angleWrite(pos[k - 1][0]); //寫入最後一顆servo數值
    for (a = 0; a < 30; a++) //二維陣列共30行
    {
        for (b = 0; b < 3; b++) //二維陣列共3列

```

---

(圖 14) 局部 motionRecord 程式

自動流程中所使用的函式則是以 switch case 編成(如圖 15)，利用二維陣列裡的馬達與角度資訊並配合控速函式庫做出順暢的馬達轉動效果。從複製起始點往終點也就是正向動作時，switch case 配合 for 迴圈照設定順序取用二維陣列的馬達名稱與其最終角度，進行自動的動作。而當要從終點返回至起始點時。則逆向取用二維陣列裡的資料，藉此達到依照複製流程路徑往返的效果(如圖 15)。

```

}
void servoAutoMotion(int servoNO, int servoAngle) { //自動化函數
    switch (servoNO) //選擇動作Servo
    {
        case 1:
            Servo1.write(servoAngle, SPEED); //利用控速函式庫控速旋轉至寫入的角度
            Servo1.wait(); //等待旋轉至角度才進行下個動作
            break; //跳出case
        case 2:
            Servo2.write(servoAngle, SPEEDservo2); //利用控速函式庫控速旋轉至寫入的角度
            Servo2.wait(); //等待旋轉至角度才進行下個動作
            break; //跳出case
        case 3:
            Servo3.write(servoAngle, SPEED); //利用控速函式庫控速旋轉至寫入的角度
            Servo3.wait(); //等待旋轉至角度才進行下個動作
            break; //跳出case
        case 4:
            Servo4.write(servoAngle, SPEED); //利用控速函式庫控速旋轉至寫入的角度
            Servo4.wait(); //等待旋轉至角度才進行下個動作
            break; //跳出case
        case 5:
            Servo5.write(servoAngle, SPEED); //利用控速函式庫控速旋轉至寫入的角度
            Servo5.wait(); //等待旋轉至角度才進行下個動作
            break; //跳出case
        case 6:
            Servo6.write(servoAngle, SPEED); //利用控速函式庫控速旋轉至寫入的角度

```

---

(圖 15) 局部 switch case 程式

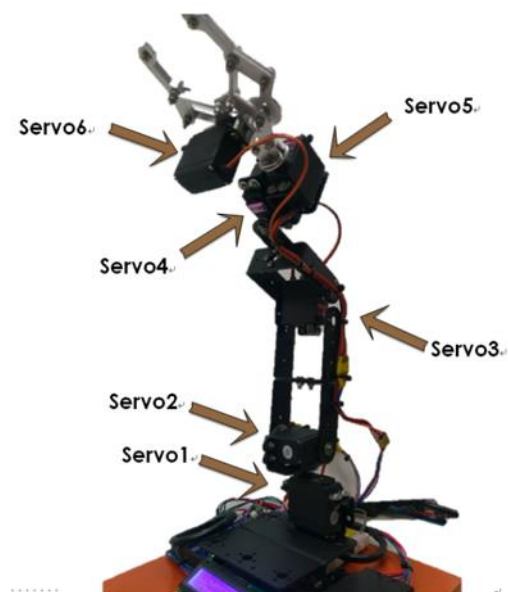
初始化部分則是也使用控速函式庫將各馬達調整至預設角度。LCD 顯示器也會像圖中一樣全程顯示當前資訊，不管是在遙控還是自動流程中(如圖 16)。

```
void initial(void) {  
    lcd.clear(); //lcd清除先前畫面  
    lcd.setCursor(0, 0); //設定第一個字顯示座標  
    lcd.print("Initial"); //lcd顯示字串  
    lcd.setCursor(0, 1); //設定第二個字顯示座標  
    lcd.print("Start"); //lcd顯示字串  
    Serial.println("Initial"); //監控視窗列印字串  
    Serial.println("Start"); //監控視窗列印字串  
    delay(2000);  
    angle2 = 95; //servo2角度設定為預設狀態值  
    Servo2.write(angle2, SPEEDservo2); //以SPEEDservo2的速度轉至angle2角度  
    Servo2.wait(); //等待servo2動作結束  
    Serial.println("Servo2"); //監控視窗列印字串  
    Serial.println(angle2); //監控視窗列印角度數值  
    lcd.clear(); //lcd清除先前畫面  
    lcd.setCursor(0, 0); //設定第一個字顯示座標  
    lcd.print("Servo2"); //lcd顯示字串  
    lcd.setCursor(0, 1); //設定第二個字顯示座標  
    lcd.print(angle2); //lcd顯示角度數值  
    delay(1000); //緩衝時間  
    angle3 = 50; //servo3角度設定為預設狀態值  
    Servo3.write(angle3, SPEED); //以SPEED的速度轉至angle3角度  
    Servo3.wait(); //等待servo3動作結束  
    Serial.println("Servo3"); //監控視窗列印字串  
    Serial.println(angle3); //監控視窗列印角度數值  
    lcd.clear(); //lcd清除先前畫面  
}
```

(圖 16)初始化與 LCD 顯示舉例

## 二、馬達說明

- Servo1：控制整隻手臂左、右轉。
- Servo2：手臂後段前、後動作。
- Servo3：手臂前段前、後動作。
- Servo4：夾爪左、右轉。
- Servo5：夾爪順、逆時針旋轉。
- Servo6：夾爪夾、放。



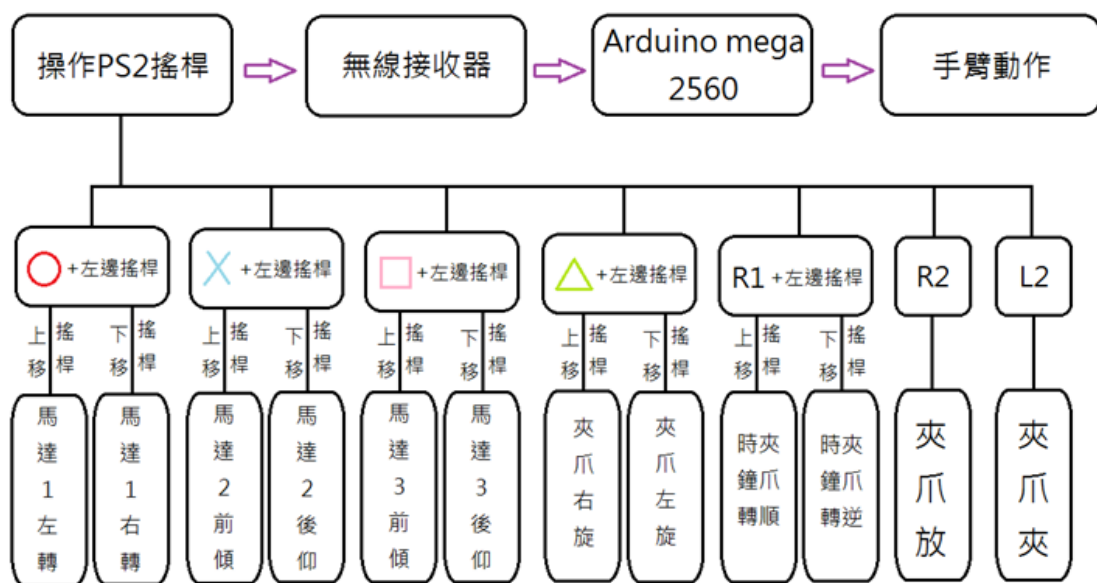
(圖 17)機構示意圖



### 三、動作流程





#### (一) 操作說明

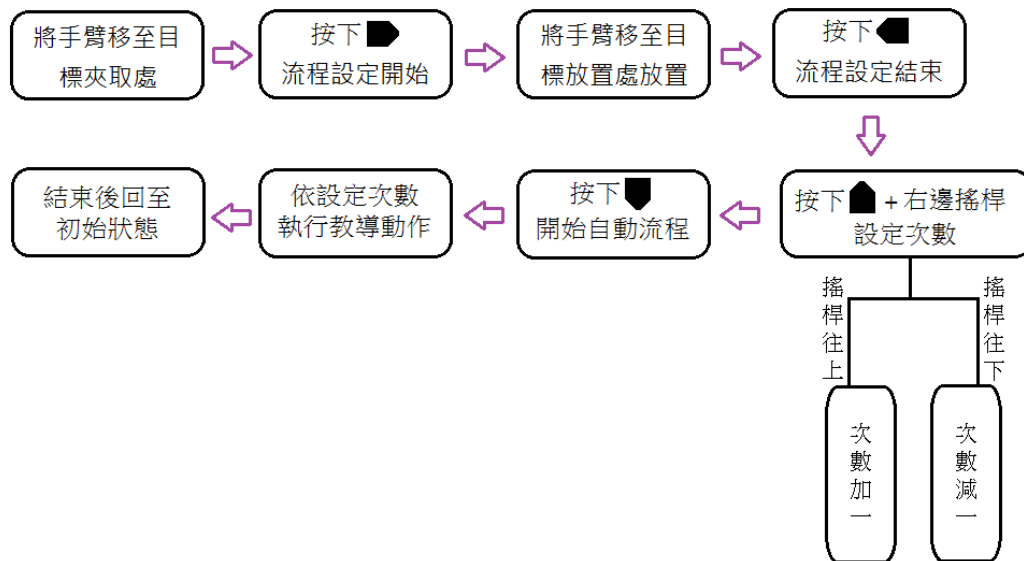
用 PS2 搖桿，透過無線接收器傳輸信號給 Arduino 以操控機械手臂。機械手臂可全自由操控，也可用以設定自動動作流程。



(圖 18) 操作說明

#### (二) 功能說明

操作者利用 PS2 搖桿將手臂移至欲夾取的目標位置後，按下 ，顯示器顯示 Start set，表示目標位置設定完成，便可開始操作欲複製之流程，複製結果將完全按照此時至按下完成鍵前之動作，當然若馬達在遙控時不小心轉錯邊，程式只會儲存最終馬達定位點，並不影響流程。移至存放位置後，也就是自動流程欲結束之動作，按下 ，顯示器顯示 Done set，表示自動流程設定完成。接著按住  + 右邊搖桿設定欲動作次數，搖桿往上次數加一，搖桿往下次數減一，顯示器全程顯示當前設定次數。而次數設定完成後，按下 ，手臂會先回預設初始點接下來便會開始執行所設定之動作，此時顯示器將顯示” Auto start” 與 “手臂已完成次數/設定完成次數”，當做完設定次數，手臂回至初始狀態。



(圖 19) 自動流程說明

## 陸、討論

### 問題一：

在解決完手臂傾倒的問題後，又發現 996R 馬達的力矩在力臂伸到最長的情況下完全不夠，沒有辦法以馬達本身的力量拉回垂直，需要人為稍微施予力量才能舉起。

### 解決方法：

馬達力矩不夠最直觀的想法就是換顆扭力更大的馬達來驅動，但由於馬達的價格也會隨著扭力的上升而增加，所以我們最多只能替換成 20 公斤的雙軸承 995 馬達，但後來又發現 20 公斤的 995 馬達雖然已經很有力，但在夾取物品時還是稍嫌不足，便多方面的請教了同學及老師，當中最讓我們覺得驚喜的答案就是用拉力取代扭力，比起再換一顆馬達，加裝一個彈簧也可以達到同樣的效果，而且既省錢又省換馬達的時間。

### 問題二：

原先使用普通陣列來儲存起始與終點位，手臂只能依照由下往上的馬達順序來在兩個點為之間移動，不僅沒有達到完美複製之效果，也有可能撞機。

### 解決方法：

改用二維陣列來儲存馬達動作順序與角度位置，再搭配 switch case 來使馬達依設定的順序動作，從終點回起始點時則反向動作即可，藉此達到在起始與終點間來回操作又不失其流程順序的效果。



## 柒、結論

遙控教導式的機械手臂，確實能省下編寫新程式的時間，且不僅能使得整體自動化系統更有效率，更能去除完全人工的失誤。只要操控教導過一次之後，便能無限使用。方便的遙控指令也能讓使用者容易上手，而流暢的機械手臂也能順利地動作整套使用者希望的流程，實為便利。

然而隨著越深入的研究，優化教導式機械手臂的想法也越來越多，未來可能發展出能夠加裝記憶體的教導式機械手臂，配合儲存裝置的記憶及讀取，讓手臂能隨時更換成先前做過的任何動作，如此可以減少重複教導一次的時間。控制方面則可以以人手親自虛擬操作一次取代搖桿，達到更為精確的控制。

## 捌、參考資料

### 一、機械手臂介紹

<https://zh.wikipedia.org/wiki/%E6%A9%9F%E6%A2%B0%E6%89%8B%E8%87%82>

### 二、995 馬達

[https://images-na.ssl-images-amazon.com/images/I/5187rqPrbZL.\\_SX342\\_.jpg](https://images-na.ssl-images-amazon.com/images/I/5187rqPrbZL._SX342_.jpg)

### 三、Arduino

<https://zh.wikipedia.org/wiki/Arduino>

### 四、小狐狸事務所 Arduino 基本語法筆記

[http://yhhuang1966.blogspot.tw/2015/09/arduino\\_14.html](http://yhhuang1966.blogspot.tw/2015/09/arduino_14.html)

### 五、自造學堂 Arduino 如何透過 i2c 控制 LCD

<https://makerpro.cc/2017/02/how-arduino-use-i2c-to-control-lcd-module/>

### 六、Arduino 使用 1602iic(i2c)LCD 點陣液晶模組

<https://blog.gtwang.org/iot/ywrobot-arduino-lcm-1602-iic-v1-lcd-display/>

### 七、柯博文 (2014)。ARDUINO 互動設計專題與實戰 深入 ARDUINO 的全方位指南。碁峰資訊股份有限公司