大　葉　大　學

資　訊　工　程　學　系

專　題　製　作　報　告

移動式手臂

學生：林建成 F0106036
王炫鈞 F0106213
劉俊延 F0106031
賴佑益 F0106011

指導老師：林仁勇 老師

中　華　民　國　１０４　年　１２　月

摘要

近年來物聯網蓬勃的發展，想必與物聯網相關技術在未來的趨勢一定不容小覷，所以我們就先往領域去研究，最後我們經由網路資訊，發現了 Arduino 在這裡技術領域裡佔有一席之地，又還有具備硬體便宜、網路有開放源範例，所以我們決定朝著去做專研。

在決定討論題目時，我們跟指導教授還有學長朝了許多方面去做探討，像是四軸直升機加攝影機組成的空拍機，或是加裝超音波以及光線感測器的循線自走車，還有插座電流量的計量，最後我們決定先從基本的方面去著手進行研究，所以我們決定了實作操控型的單臂。

在研究的過程中，我們遇到了許多問題點，像是機器手臂的組裝就遇到了動作無法像是預期的行動；Inventor 2 寫出來的 APP 跟手機品牌以及解析度的相容性問題；Arduino 程式方面從無到有的學習，在學習的歷程裡遇到許多疑問，多虧學長的熱心教導才得以解決的，最後將車子與手臂做組合時整體重量的配置、排線問題以及如何使他具備優良的活動性。

在這一年期間我們從無到有，最後有現在的成果，全多虧了教授跟學長們的不厭其煩地熱情指導，以及組員們努力地研究、討論、學習，以及即使爭吵也可以互相包容。相信這塊領域的技術會越來越成熟，未來我們的機器人不再是只有操控性，而是具有智慧、高機動性、可自動地去解決人類的問題。

致謝

這次的專題可以順利且完整的呈現給大家，首先要歸功於我們的指導教授(林仁勇 老師)，非常感謝他，當我們每一次的研究遇到瓶頸時，總是能在第一時間幫助我們找到合適的解決方案，並且提供我們當時所欠缺的研究相關器材，同時也謝謝與老師一同教導我們的學長姐們，多虧了學長姊們與我們分享的許多經驗，使我們在研究的過程中也吸收到了相當多的知識與技巧，感謝他們不管是平日或者是假日，都願意犧牲自己的時間來指導我們，只要我們有遇到解決不了的問題需要請教他們時，我們都可以跟他們約個時間，之後到學校與他們進行討論與測試。

最後也要感謝這組的每位成員(林建成、王炫鈞、劉俊延、賴佑益)，大家都竭盡所能做好自己被分配到的工作，發揮自己的專長，一起在不斷的測試中學習，雖然中間總是會遇到瓶頸，不過大家都會透過各別的意見，採取最好的應對方法。當然意見總會有分歧的時候，畢竟每個人的價值觀各不相同，所以講究的細節也會不同，但是每次到了最後，我們大家總會各退一步，因為大家覺得，每個人的意見都是值得受到尊重與包容的，在此多謝各組員的互相體諒與包容，非常感謝大家。

目錄

圖目錄

表目錄

第一章　簡介

1.1 專題研究動機

剛開始的時候，我們一起在未知領域尋找這次的主題是什麼，雖然我們事前都有聽過什麼是 Arduino，但是實際要我們舉例有哪些相關作品時，其實我們大家只能舉少數例子，之後我們大家各自收集了相關的資訊，並和老師討論未來作品的可行性，其中我們對機器手臂挺感興趣的，但是覺得只有一個機器手臂太單調了，所以我們決定研究機器手臂加上自走車，使它變得更特別。

1.2 專題目標

在多方面的探討後，我們決定研究由 Arduino 板子所衍生的作品，在這次的研究中，從組裝機器手臂與自走車至使用智慧型手機去控制。過程雖然很複雜且要逐一修正程式碼，但是我們從中學習到機器手臂與自走車之間的配合及操作細節，進而做出二合一的移動式機器手臂，希望此專題在未來可以擁有更多延伸，不僅能像電影裡的拆彈情節一樣，為了避免人員傷亡所設計的機器手臂，也能用在室內，提供看護的工作，幫助老人以及傷患所需要的幫助。

1.3 實驗計畫

本專題研究擬進行下列相關研究步驟：

1. 研究 Arduino 如何開發。
2. 研究藍牙協定以及連接。
3. 組裝機器手臂與自走車。
4. 撰寫機器手臂、自行車與藍牙的 code。
5. 調整手臂與自走車之間的配置與合適度。
6. 將成果結合，實際運作。

1.4 分配工作

表 1.1 成員工作分配表

林建成	蒐集相關資料、程式撰寫、硬體組裝、Arduino 測試、書面撰寫
王炫鈞	蒐集相關資料、程式撰寫、硬體組裝、Arduino 測試、書面撰寫
劉俊延	蒐集相關資料、程式撰寫、Arduino 測試、硬體組裝、書面編排、書面撰寫
賴佑益	蒐集相關資料、硬體組裝、書面編排、書面撰寫

1.5 實驗中所遭遇到的問題及解決方法

1　問題：

在進行藍牙撰寫時，由於相關軟體太多，不知道該用哪種撰寫。

解決問題：

請教老師或是對於藍牙撰寫比較強的學長姐們，針對我們的問題，對照他們以前的經驗來幫助我們認知個別差異，並選用對的軟體。

2　問題：

在測試機器手臂的遠端操作時，發現手臂有動，但是與指令不連貫，斷斷續續的。

解決問題：

請教老師關於這情況的緣由，並先檢查硬體方面有無接觸不良，沒問題後，藉由老師的指導，逐步改寫手臂與藍牙軟體程式碼。

3　問題：

在進行手臂與自走車組合時，自走車大小比例與手臂不合，而後續配重也有問題。

解決問題：

經由老師的幫忙，找到了一台以前經由學長姊所組裝而成，大小比例剛好與手臂也剛好吻合的自走車，之後，大家與老師一起討論配重問題。

第二章　相關研究

Arduino 是一塊基於開放原始碼的 Simple i/o 介面版，它使用了 Atmel AVR 單片機，並且具有使用類似 JAVA、C 語言的開發環境。

Arduino 可以使用 SuperCollider、Max/MSP、Processing、Java、Macromedia Flash 和 Pure Data…等軟體，結合電子元件，像是控制器件，例如：開關、感測器...等，或是其他輸出裝置，例如：LED、步進馬達...等，作出可互動作品。Arduino 可以是一個獨立運作且跟軟體溝通的介面。

2.1 Arduino UNO 板的規格介面

在大部分的使用情況下，所有的 Arduino 控制板其實並沒有差別，因為它們的核心都是由 ATmega328 型號的微控制器所組成的。而在安裝 Windows 版的 Arduino 軟體時，過程中可加入 UNO 的驅動程式。此外，Mac OS X 和 Linux 系統的使用者，可以不用安裝 UNO 的 USB 驅動程式。Arduino UNO 板的優勢在於使用者可以自行燒錄 USB 序列轉換控制器的韌體。最初的 UNO 板採用的是 MEGA8U2 晶片，它的記憶體容量比較小，快閃記憶體有 8KB，主記憶體與 EEPROM 只有 512Bytes，而 UNO ”R3”版則換成記憶體容量較大的 MEGA16U2 晶片，它的快閃記憶體有 16KB，主記憶體與 EEPROM 一樣只有 512Bytes，並新增燒錄韌體的 ICSP 接腳。

表 2.1 Arduino UNO 規格表

Arduino UNO 規格表	
微控制器晶片	ATmega328
工作電壓	5V
輸入電壓(建議值)	7-12V
輸入電壓（限制）	6-20V
數位 I/O 腳數	14（其中6個是PWM output）
類比輸入 Pins	6
I/O 腳直流電流	40 mA
3.3V 腳直流電流	50 mA
快閃記憶體	32 KB 其中 0.5 KB 用在 bootloader
靜態記憶體	2 KB
EEPROM	1 KB
時脈速度	16 MHz

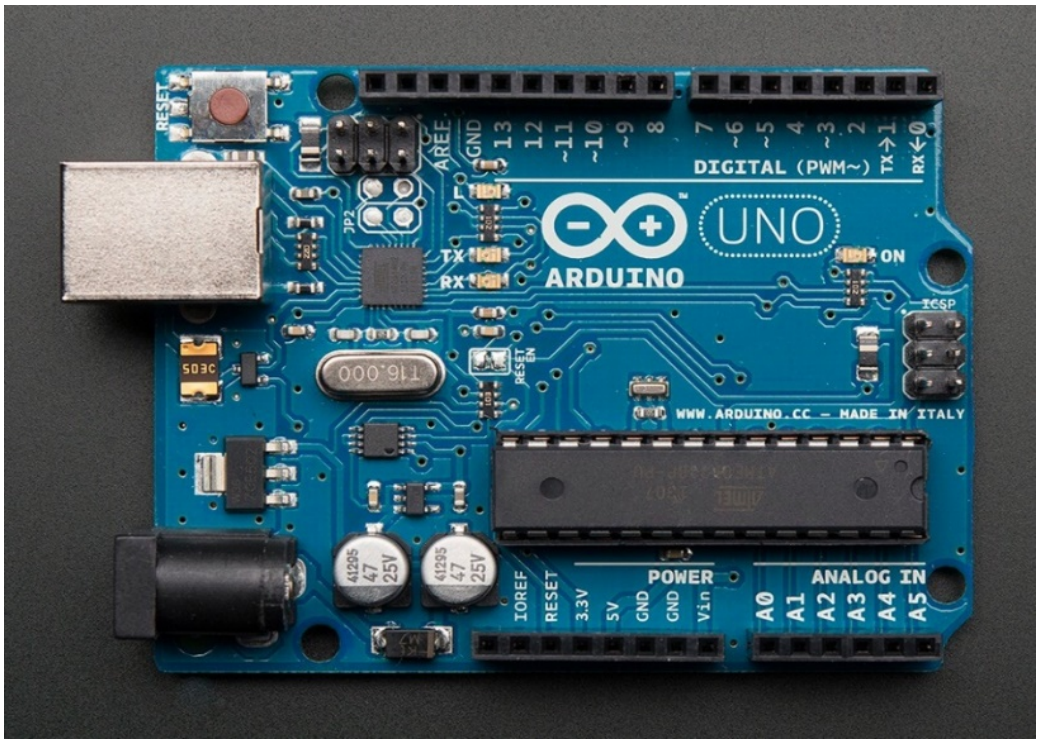

圖 2.1 Arduino UNO 外觀

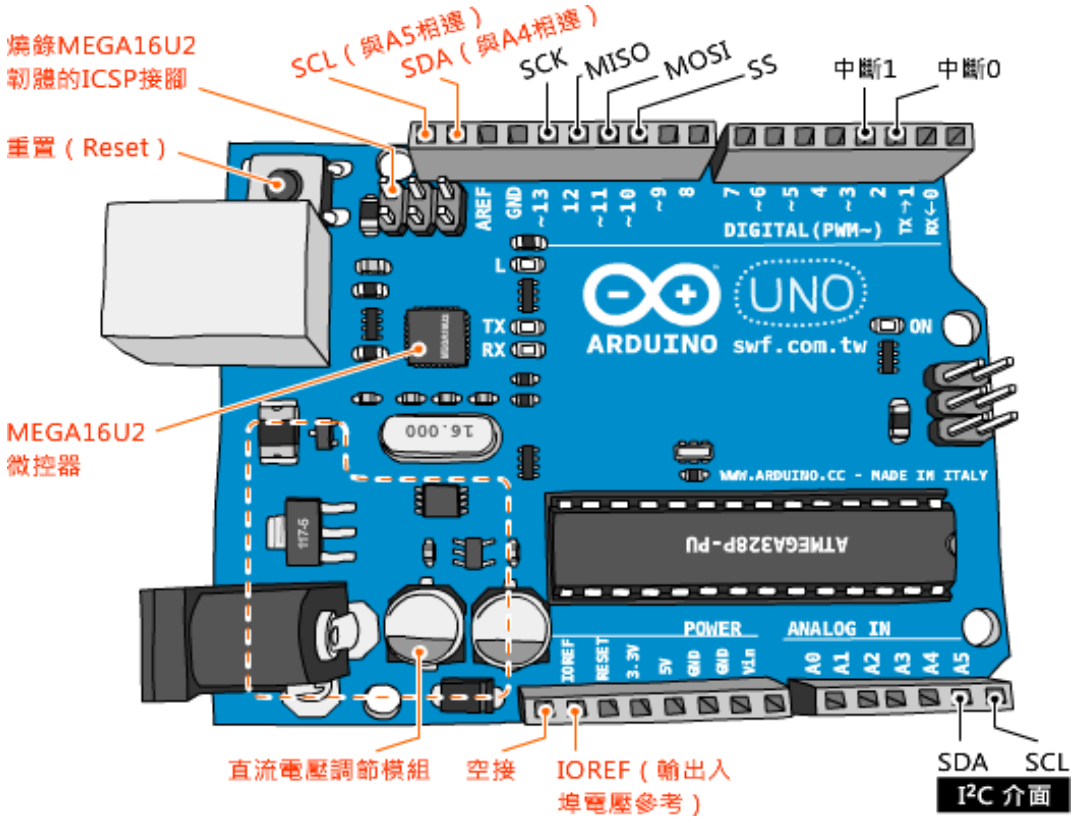

圖 2.2 Arduino UNO 內部介紹

2.2 Bluetooth(藍牙)

Bluetooth 是一種無線技術標準,可以用來讓固定或是行動的裝置，在短距離間內，互相交換資料，以便形成一個個人區域網路（PAN)。而它所使用的短波特高頻（UHF）無線電波，是經由 2.4 至 2.485 GHz 的 ISM 頻段來進行通訊的。近期間，Bluetooth 用於在不同的裝置上，與其它物件進行無線連線，像是連線電腦和外圍裝置，如：印表機、鍵盤.... 等，或是讓個人數位助理（PDA）與附近其它的 PDA 或電腦進行通訊。具備 Bluetooth 技術的手機可以連線到電腦、PDA 甚至免持聽筒。事實上，根據技術上的驗證，Bluetooth 可以支援功能更強的長距離通訊,也可以構成無線區域網路。每個 Bluetooth 裝置可同時維護 8 個連線，而每個裝置都被設定為可以向附近的裝置偵測並宣告自身存在，以便建立連線。另外也可以對二個裝置之間的連線進行密碼保護，以防止被其他裝置接收或入侵。

常見的兩種支援 SPP（Serial Port Profile，序列埠規範）的 Bluetooth 模組：

HC-05：

主/從（master/slave）一體型，出廠預設通常是「從端」模式，但是能自行透過 AT 命令修改，如圖 2.3。

HC-06：

主控端或從端模式，出廠前就設定好，不能更改；市面上販售的通常是「從端」模式，如圖 2.3。

不管是 HC-05,HC-06，對 Arduino 都沒有影響，控制程式都一樣，實際接線也只用到 4 條線：電源、接地、傳送（TxD）和接收（RxD）。而關於 Bluetooth 模組的操作模式有兩種，一種為自動連線（automatic connection），又稱為透通模式（transparent communication），另一種為命令回應（order-response），又稱為 AT 模式（AT mode）。

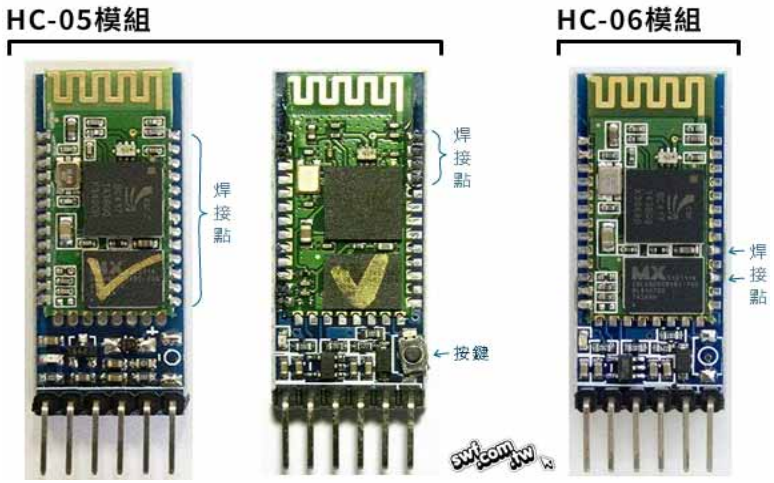

圖 2.3 近期常用的 Bluetooth 模組

表 2.2 常用的 Bluetooth 簡介

模組型號	用途	模式
HC-05	一般用(成本便宜)	主從一體，能自行透過 AT 命令修改。
HC-06	一般用(成本便宜)	單一模式(出廠前就設定好主端或從端，不能更改。)
HC-07	一般用(成本便宜)，精簡型 HC-06	單一模式(出廠前就設定好主端或從端，不能更改。)

第三章　Arduino UNO 版與手機 App 設計

3.1 手機遠端操控程式

本專題是使用 App Inventor 2 來撰寫 App 程式 ，並透過藍牙連線來遠端操控機械手臂車。

App Inventor 2 使用拼圖模式來開發 Android 裝置應用程式,並且也支援多國語言,能夠使得對於程式語言不熟悉的人能輕易地使用，而 App Inventor 2 是一個完全線上開發的 Android 程式環境，只要有連上網路，它可以在 Chrome、FireFox、Internet Explorer 這些一般的瀏覽器上進行開發，因為它原本是由 Google 所研發，之後所有在網際網路上所開發的設計專案，都會透過 Google Account 儲存在 Server 上。

圖 3.1App Inventor 2 圖樣

3.2 App Inventor 2 環境介紹

進入到 App Inventor 2 後，點擊左上角新建專案，如圖 3.2。

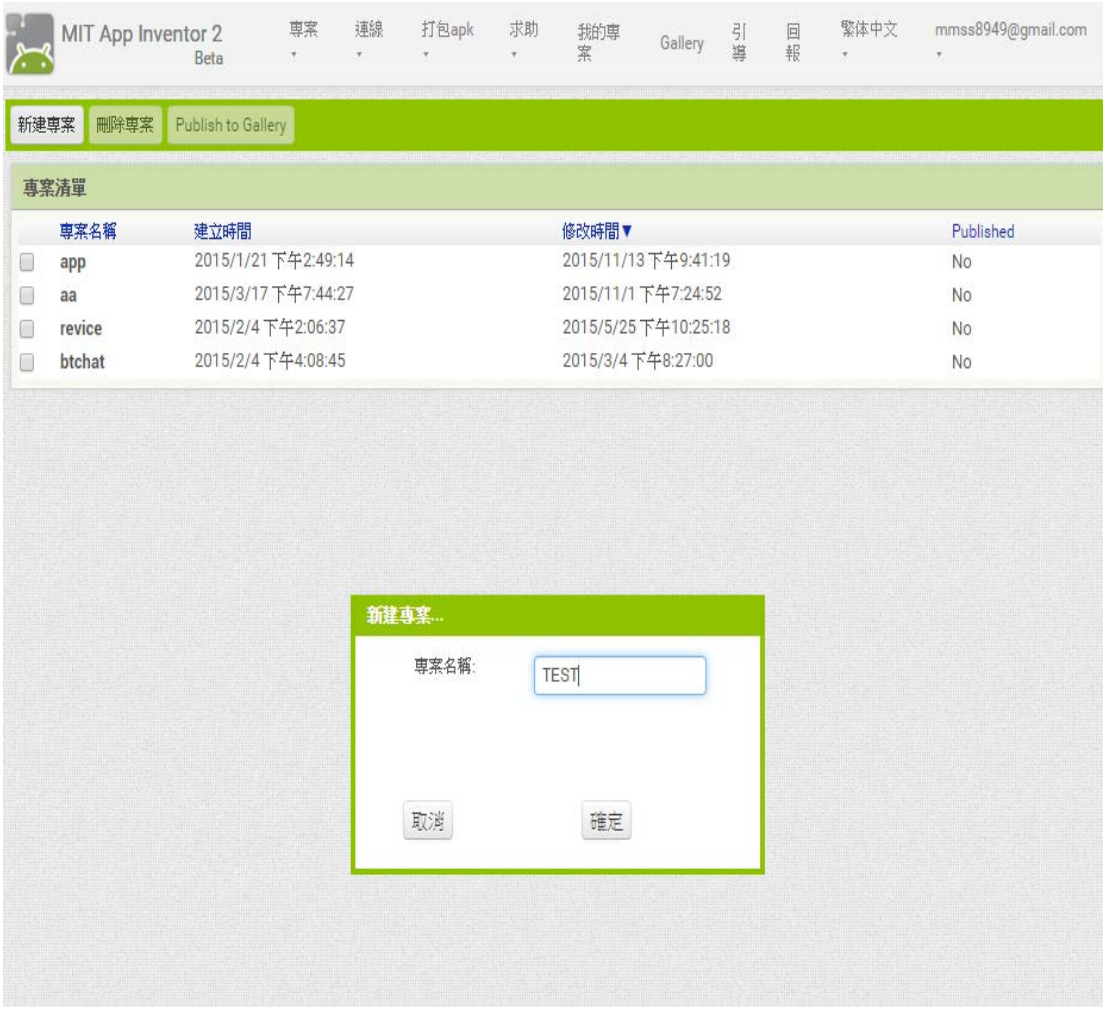

圖 3.2 App Inventor 2 新建專案

如圖 3.3，區塊 1 是元件面板，裡面包含了使用者介面、界面布局、多媒體...等工具欄。區塊 2 是工作面板，裡面有手機程式本身的畫面，可以藉由元件面板新增功能放置工作面板。區塊 3 是元件清單，由元件面板所新增的功能都會顯示在這條例表上面。區塊

4 是元件屬性，在點選元件清單上的元件後，將會在此顯示此元件的詳細屬性。區塊 5 是素材，所有上傳的圖片與素材皆會此顯示。區塊 6 是畫面增加與畫面程式碼設計切換，可以在此新增主程式畫面與切換成程式碼設計。

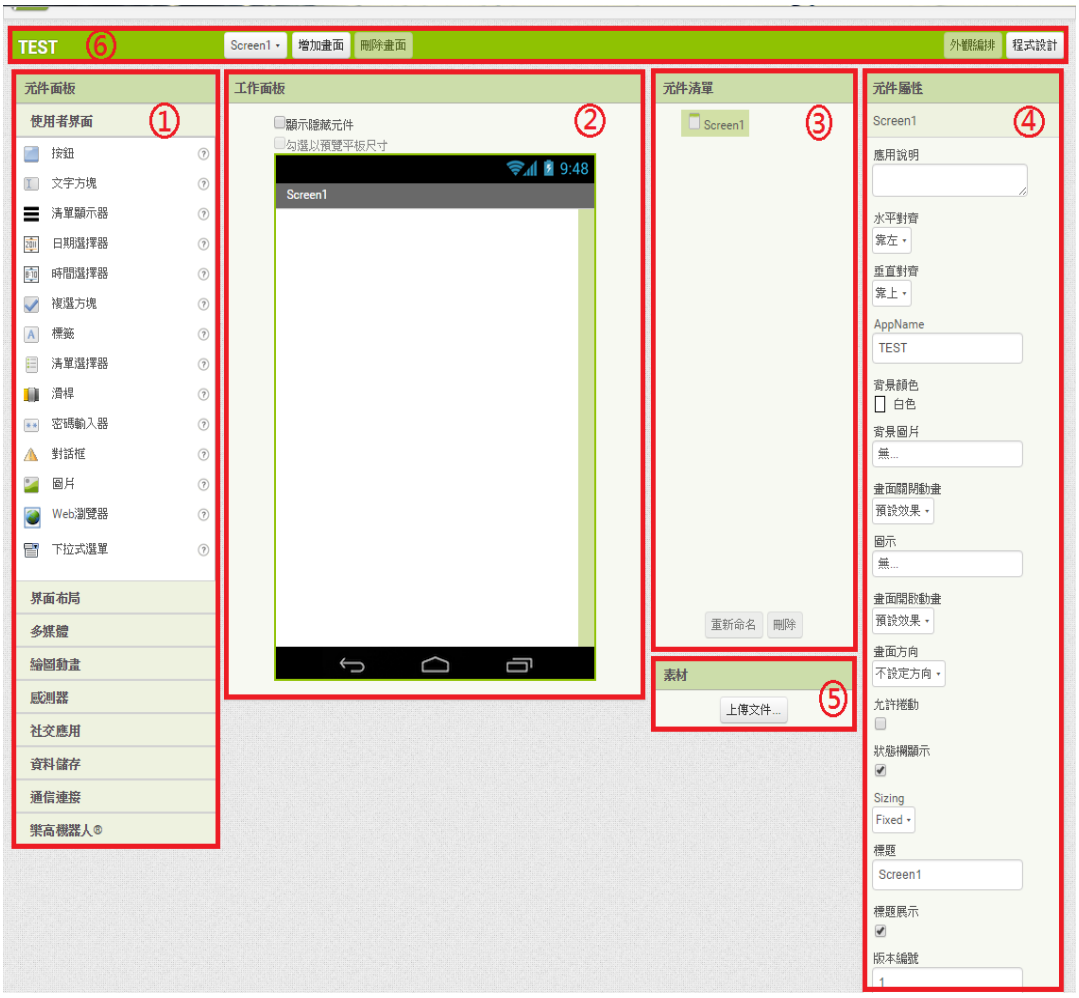

圖 3.3 App Inventor 2 介面設計說明

如圖 3.4，區塊 1 裡包含了內建區塊、Screen 畫面以及元件三大點，而內建區塊裡有流程控制、邏輯運算、算術運算、文字...等，而 Screen 畫面則是所有新增的畫面皆會在此，最後的元件會把所有

新增的按鈕或元件都在此設計。區塊 2 裡是素材，所有上傳的圖片或素材皆會在此顯示。區塊 3 是工作面板，可以從區塊拉選區塊或元件等等來進行拼湊程式碼。

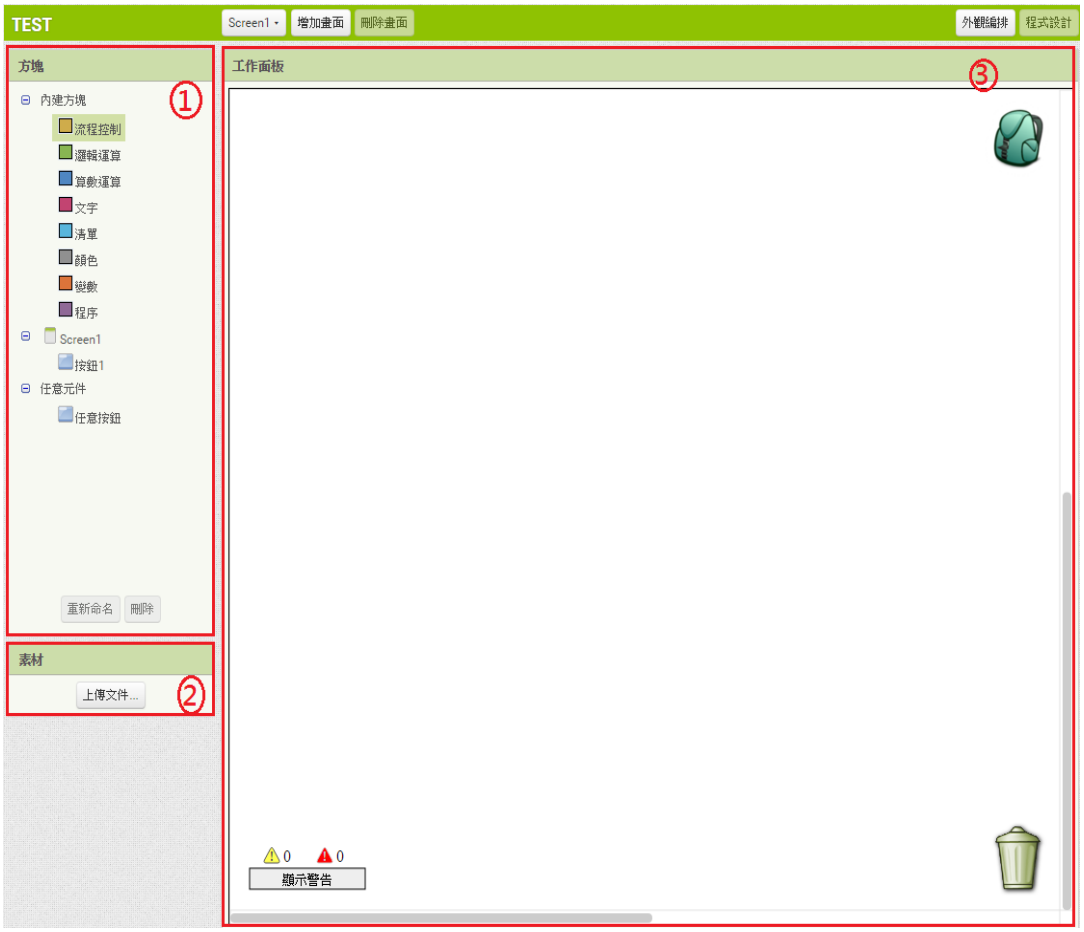

圖 3.4 App Inventor 2 程式碼設計說明

3.3 遠端程式使用步驟

首先開啟 App 並開啟藍牙點選連接裝置，如圖 3.5。

圖 3.5 程式畫面

連接機械手臂自走車的藍牙，如圖 3.6。

圖 3.6 連接選擇畫面

連接成功，如圖 3.7。

圖 3.7 連接成功後的畫面

點選圖中的 on/off 按鈕即可斷開連結，如圖 3.8。

圖 3.8 藍牙斷開後的畫面

3.4 App Inventor 2 程式碼

當軟體開啟時，只有藍牙連接按鈕將設置成可以使用，如圖 3.9。

圖 3.9 螢幕初始化後的程式碼設定

將清單選擇器的內容設定為藍牙連接，如圖 3.10。

圖 3.10 連接按鈕內容設定

若藍牙連線成功，所有按鈕將會設置成可以使用，除了藍牙裝置連接按鈕外，開關按鈕會切至 on，如圖 3.11。

圖 3.11 連接成功後 程式碼設定

若點選 dbt 按鈕將會斷開藍牙連接，且所有按鈕將會關閉，除了藍牙連接按鈕外，開關按鈕則會切製成 off，如圖 3.12。

圖 3.12 連藍牙線斷開後 程式碼設定

如表 3.1，此表格為按鈕按下後，所傳輸對應的變數名稱。

表 3.1 圖片按鈕名稱與變數名稱對照表

按鈕名稱	變數名稱
F	F
L	L
R	R
B	B
O	O
C	C
W	W
A	A
S	S
D	D
U	U
N	N
連接裝置	RE
開關按鈕(ON/OFF)	DBT

3.5 自走車與機器手臂的藍牙程式介紹

3.5.1 App Inventor 2 自走車部分

當 F 按鈕被按壓後，會送出文字 F 至機械手臂車，車子會有往「前」的動作，鬆開按鈕後，會送出文字 X 至機械手臂車，停止動作，如圖 3.13。

圖 3.13 App Inventor 2 車往前的程式

當 B 按鈕被按壓後，會送出文字 B 至機械手臂車，車子會有往「後」的動作，鬆開按鈕後，會送出文字 X 至機械手臂車，停止動作，如圖 3.14。

圖 3.14 App Inventor 2 車往後的程式

當 L 按鈕被按壓後，會送出文字 L 至機械手臂車，車子會有往「左」的動作，鬆開按鈕後，會送出文字 X 至機械手臂車，停止動作，如圖 3.15。

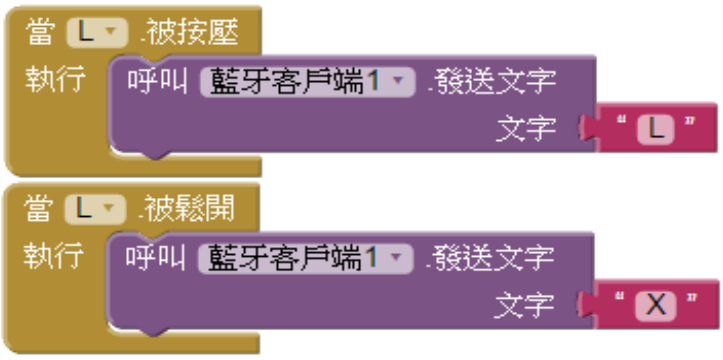

圖 3.15 App Inventor 2 車往左的程式

當 R 按鈕被按壓後，會送出文字 R 至機械手臂車，車子會有往「右」的動作，鬆開按鈕後，會送出文字 X 至機械手臂車，停止動作，如圖 3.16。

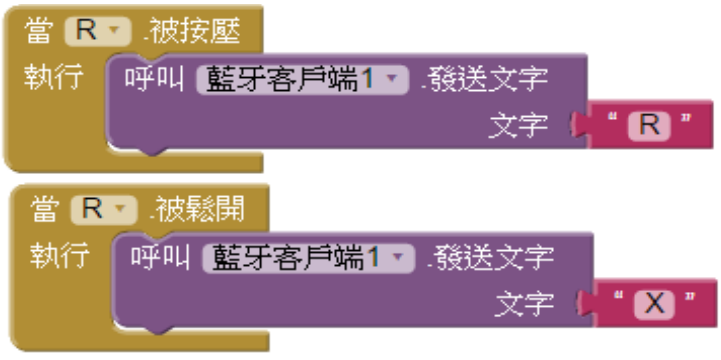

圖 3.16 App Inventor 2 車往右的程式

3.5.2 App Inventor 2 機器手臂部分

當 A 按鈕被按壓後，會送出文字 A 至機械手臂車，手臂底部會有往「左」的動作，鬆開按鈕後，會送出文字 X 至機械手臂車，停止動作，如圖 3.17。

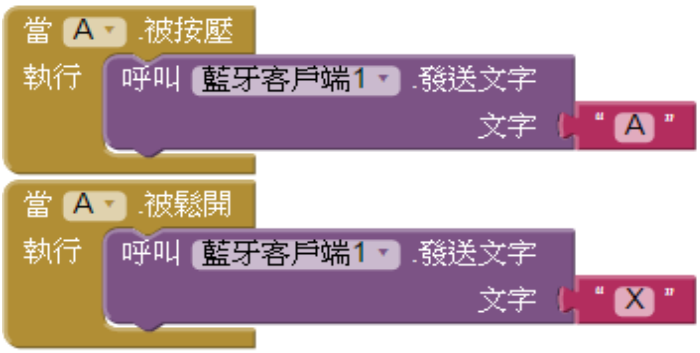

圖 3.17 App Inventor 2 手臂底部往左的程式

當 D 按鈕被按壓後，會送出文字 D 至機械手臂車，手臂底部會有往「右」的動作，鬆開按鈕後，會送出文字 X 至機械手臂車，停止動作，如圖 3.18。

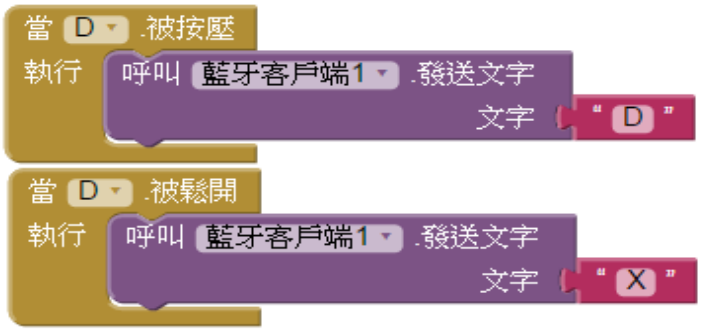

圖 3.18 App Inventor 2 手臂底部往右的程式

當 U 按鈕被按壓後，會送出文字 U 至機械手臂車，手臂會有往「上」的動作，鬆開按鈕後，會送出文字 X 至機械手臂車，停止動作，如圖 3.19。

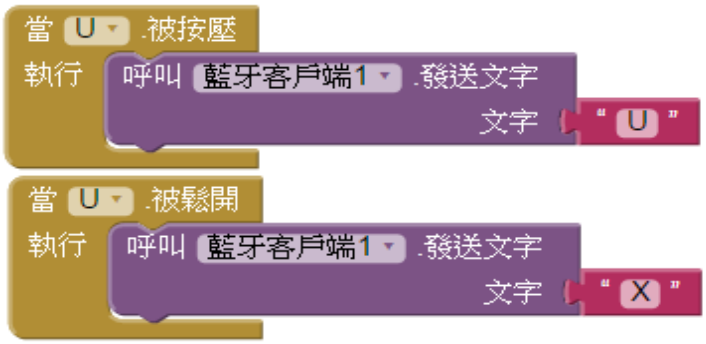

圖 3.19 App Inventor 2 手臂往上的程式

當 N 按鈕被按壓後，會送出文字 N 至機械手臂車，手臂會有往「下」的動作，鬆開按鈕後，會送出文字 X 至機械手臂車，停止動作，如圖 3.20。

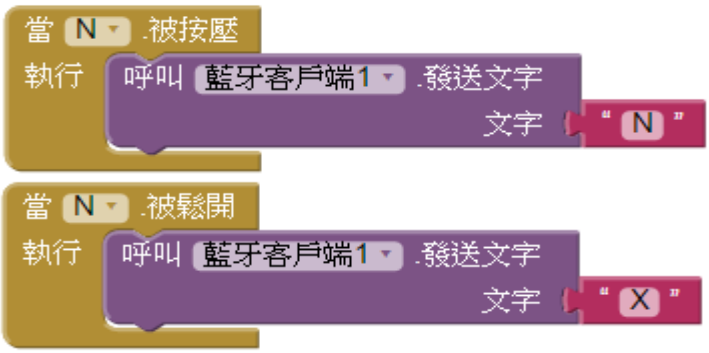

圖 3.20 App Inventor 2 手臂往下的程式

當 W 按鈕被按壓後，會送出文字 W 至機械手臂車，手臂會有往「前」的動作，鬆開按鈕後，會送出文字 X 至機械手臂車，停止動作，如圖 3.21。

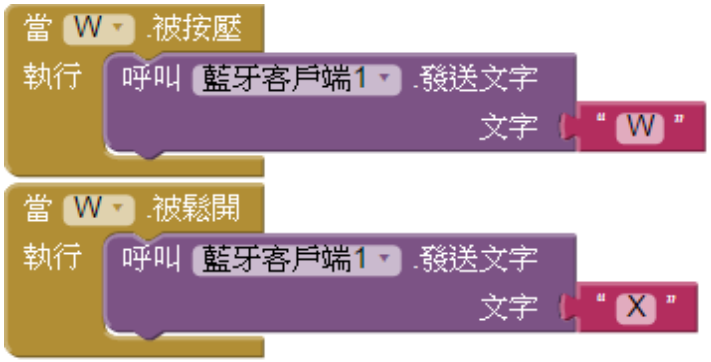

圖 3.21 App Inventor 2 手臂往前的程式

當 S 按鈕被按壓後，會送出文字 S 至機械手臂車，手臂會有往「後」的動作，鬆開按鈕後，會送出文字 X 至機械手臂車，停止動作，如圖 3.22。

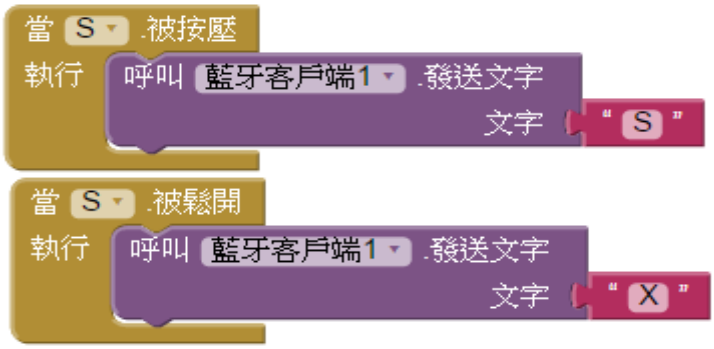

圖 3.22 App Inventor 2 手臂往後的程式

當 O 按鈕被按壓後，會送出文字 O 至機械手臂車，手臂會有「開夾」的動作，鬆開按鈕後，會送出文字 X 至機械手臂車，停止動作，如圖 3.23。

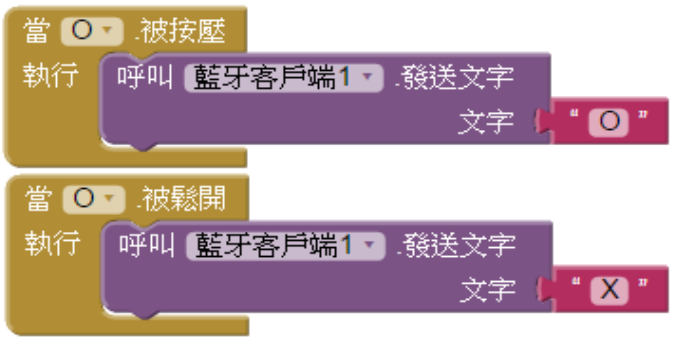

圖 3.23 App Inventor 2 手臂開夾的程式

當 C 按鈕被按壓後，會送出文字 C 至機械手臂車，手臂會有「關夾」的動作，鬆開按鈕後，會送出文字 X 至機械手臂車，停止動作，如圖 3.24。

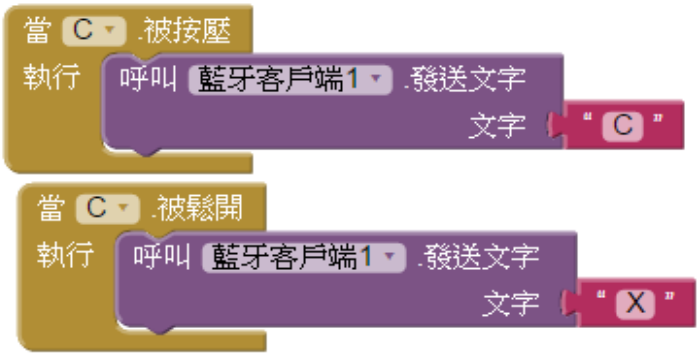

圖 3.24 App Inventor 2 手臂關夾的程式

3.6 Arduino 開發環境介紹和設定

請先到 Arduino 的官方網站下載 Arduino Software，請依照自己作業系統來下載安裝，如圖 3.25。

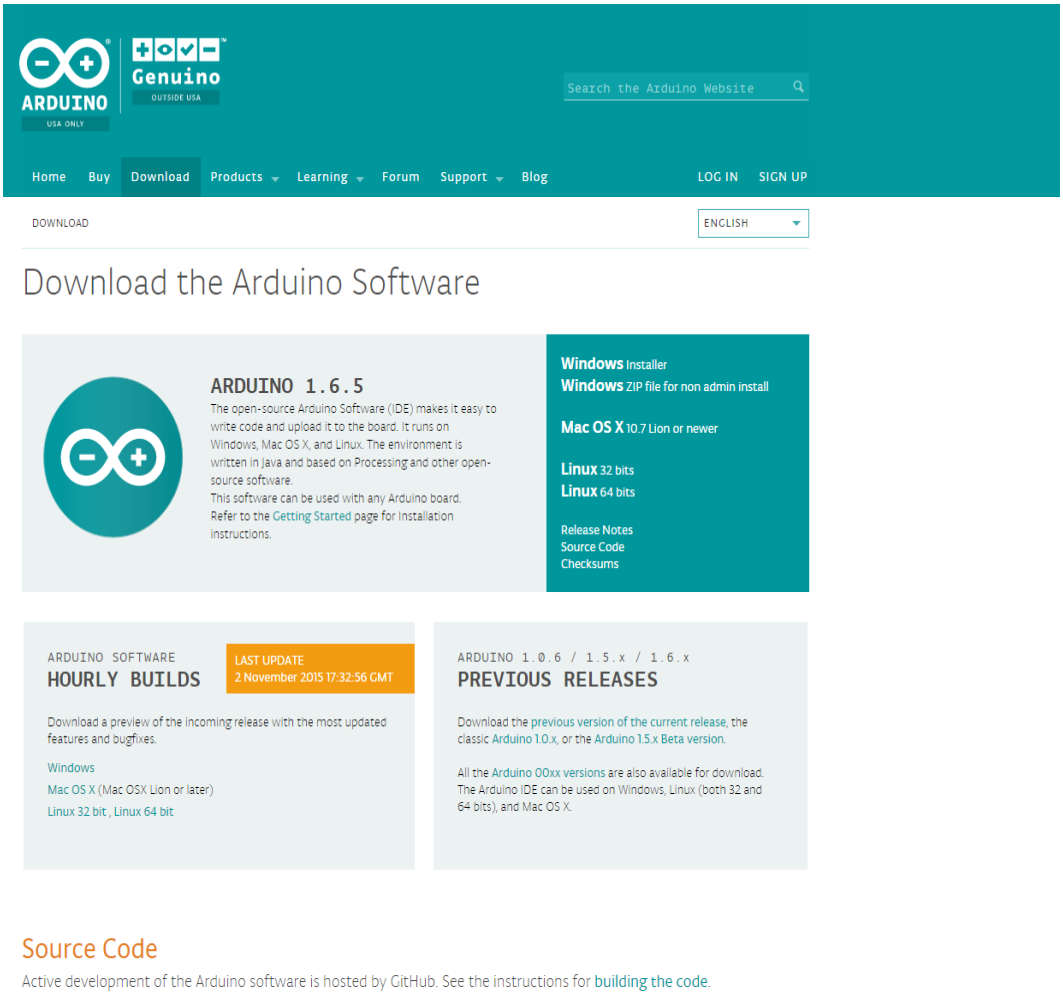

圖 3.25 Arduino Software 主頁

安裝好之後，開啟 Arduino 的 IDE 介面，畫面如圖 3.26。

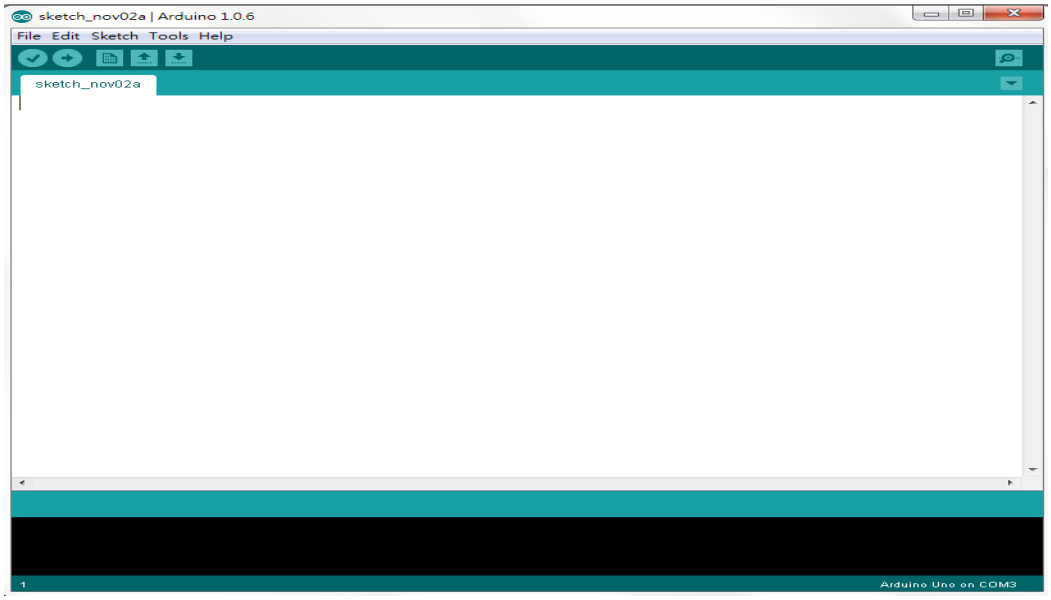

圖 3.26 設計 Arduino 版的介面

再來連接 Arduino 的 UNO 版，設定 Port，連接時請確認是否有驅動成功，並且記住 Port 的名稱，如圖 3.27。

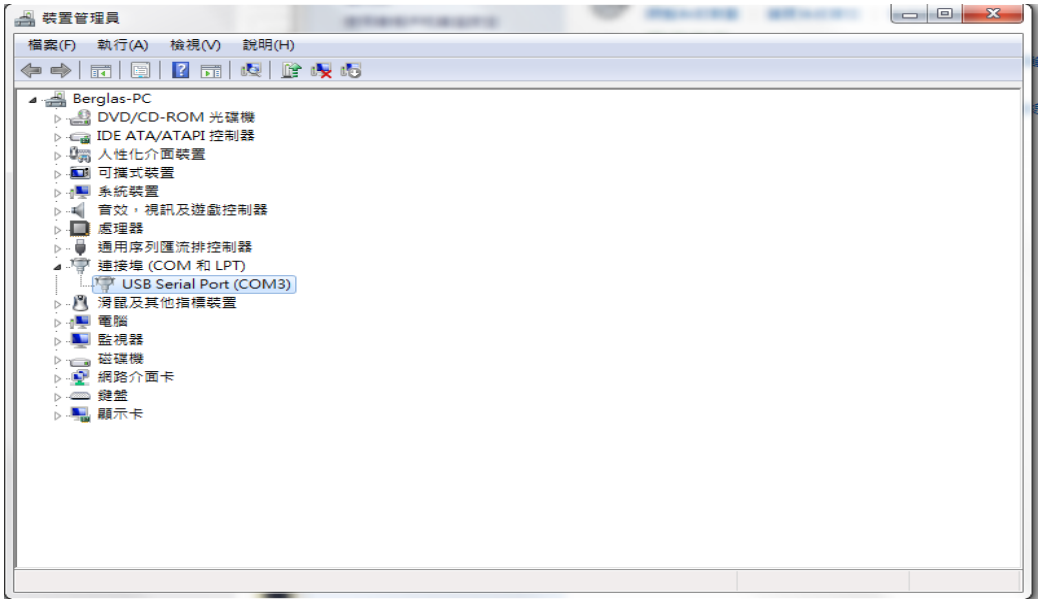

圖 3.27 連接 UNO 版

在編譯器介面時選擇：Tools→Serial Port→選擇 Port(在這是使用 COM3) ，如圖 3.28。

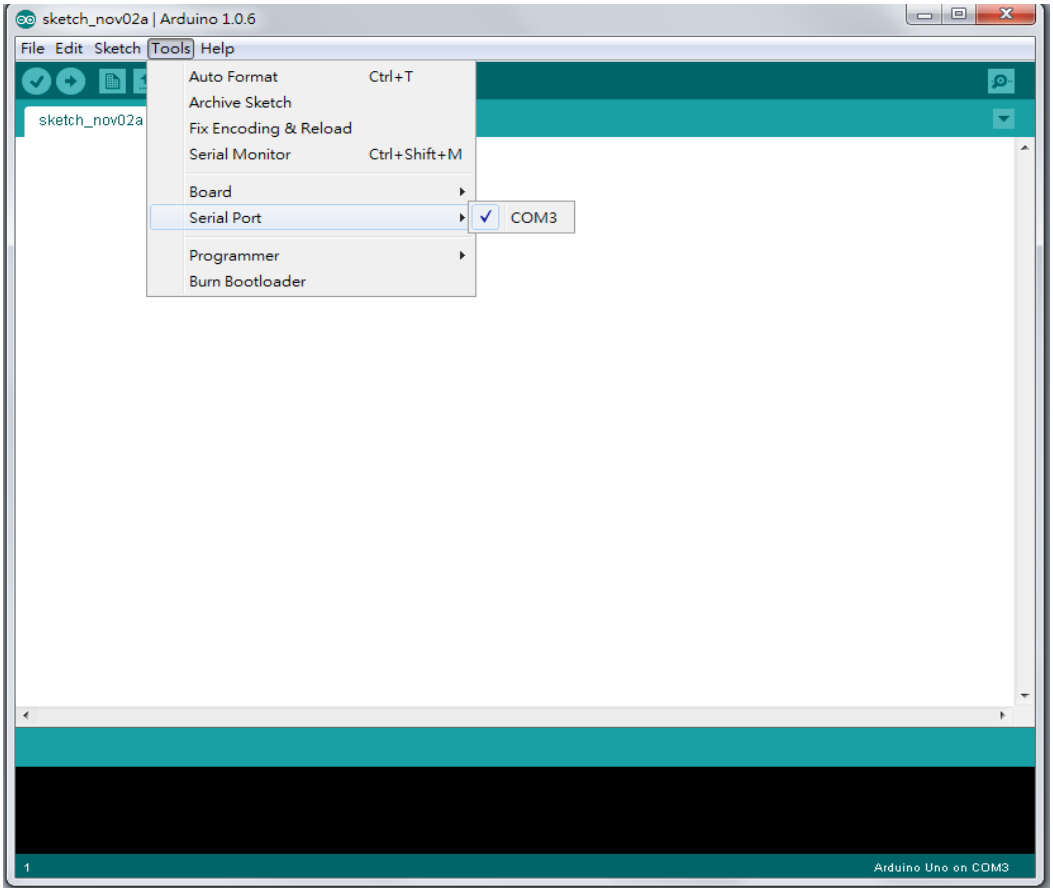

圖 3.28 選擇編譯器

這樣環境就設定完成，可以開始撰寫程式碼了。

#include 是了為引用函示庫，而#include <SoftwareSerial.h>是使用藍牙模組時所需的引用，另外，#include <Servo.h>是使用伺服馬達時所需的引用，如圖 3.29。

ArmCar §

```
#include <SoftwareSerial.h>
#include <Servo.h>



void setup()
{
  int RM_F=8;
  int RM_B=6;
  int LM_F=5;
  int LM_B=7;
  pinMode(RM_F,OUTPUT);
  pinMode(RM_B,OUTPUT);
  pinMode(LM_F,OUTPUT);
  pinMode(LM_B,OUTPUT);

  SoftwareSerial I2CBT(0,1);
  I2CBT.begin(9600);

  Servo servoClaw,servoLeft,servoRight,servoBottom;
  servoBottom.attach(3);
  servoBottom.write(90);
  servoLeft.attach(11);
  servoLeft.write(100);
  servoRight.attach(10);
  servoRight.write(110);
  servoClaw.attach(9);
  servoClaw.write(20);

  int Claw=20,Left=100,Right=110,Bottom=90;
}
```

圖 3.29 專題整體程式碼解析

初始化車子輪子對應腳位，如圖 3.30。

```
int RM_F=8;
int RM_B=6;
int LM_F=5;
int LM_B=7;
pinMode(RM_F,OUTPUT);
pinMode(RM_B,OUTPUT);
pinMode(LM_F,OUTPUT);
pinMode(LM_B,OUTPUT);
```

圖 3.30 車子輪子對應腳位的程式碼

初始化藍牙版對應腳位&設定連接鮑率，如圖 3.31。

```
SoftwareSerial I2CBT(0,1);
I2CBT.begin(9600);
```

圖 3.31 藍牙版對應腳位和設定連接鮑率的程式碼

初始化伺服馬達和設定腳位，如圖 3.32。

```
Servo servoClaw,servoLeft,servoRight,servoBottom;
servoBottom.attach(3);
servoBottom.write(90);
servoLeft.attach(11);
servoLeft.write(100);
servoRight.attach(10);
servoRight.write(110);
servoClaw.attach(9);
servoClaw.write(20);
```

圖 3.32 初始化伺服馬達和設定腳位的程式碼

宣告使用一般參數，如圖 3.33。

```
int Claw=20,Left=100,Right=110,Bottom=90;
```

圖 3.33 宣告使用一般參數的程式碼

依照接受到的藍牙訊息，進行控制車子移動的程式碼，如圖3.34。

```
void loop()
{
  char cmd;
  if (I2CBT.available()>0)
  {
       cmd=I2CBT.read();
  }
    switch (cmd)
    {
      case 'F': //車往前
        digitalWrite(RM_F,LOW);
        digitalWrite(RM_B,HIGH);
        digitalWrite(LM_F,LOW);
        digitalWrite(LM_B,HIGH);
        break;
      case 'B':  //車往後
        digitalWrite(RM_F,HIGH);
        digitalWrite(RM_B,LOW);
        digitalWrite(LM_F,HIGH);
        digitalWrite(LM_B,LOW);
        break;
      case 'L': //車左轉
        digitalWrite(RM_F,HIGH);
        digitalWrite(RM_B,LOW);
        digitalWrite(LM_F,LOW);
        digitalWrite(LM_B,HIGH);
        break;
      case 'R': //車右轉
        digitalWrite(RM_F,LOW);
        digitalWrite(RM_B,HIGH);
        digitalWrite(LM_F,HIGH);
        digitalWrite(LM_B,LOW);
        break;
      case 'X': //停止動作(各按鍵放開發送)
        digitalWrite(RM_F,LOW);
        digitalWrite(RM_B,LOW);
        digitalWrite(LM_F,LOW);
        digitalWrite(LM_B,LOW);
        servoClaw.detach();
        servoLeft.detach();
        servoRight.detach();
        servoBottom.detach();
        break;
```

LOW 為停止　HIGH 為啟動

digitalWrite(RM_F,HIGH) 為正轉右輪;

digitalWrite(RM_B,LOW); 為反轉右輪;

digitalWrite(LM_B,HIGH);為正轉左輪;

digitalWrite(LM_F,LOW); 為反轉左輪;

圖 3.34 Arduino 自走車程式碼

下圖 3.35，此段程式碼用於控制手臂，此段 case 為接受到藍牙訊息為'W'時(接收藍牙程式碼於行標 96 至 99 行；判斷收到訊息程式碼於行標 100 至 103)執行手臂往前之動作，至於外部 while 迴圈是為了使得手臂一直重複執行此動作。行標 105 至 110 此段程式碼為操控右邊的伺服馬達角度，第 105 行為設定安全轉動動範圍，去控制能轉動的最大值，第 107 行 Right=Right+2 為增加伺服馬達的角度，delay 的部分是為了延遲馬達轉動的速度，其時間單位為毫秒。

```
93      case 'W':  //臂往前
94        while(1)
95        {
96          if (I2CBT.available()>0)
97          {
98               cmd=I2CBT.read();  //讀取藍芽訊息
99          }
100         if(cmd!='W')
101         {
102           break;  //如果接受到訊息不為'W'則離開此While()迴圈
103         }
104         servoRight.attach(10);
105         if(Right<=220)  //設定安全轉動範圍，即轉動最大值不得超過220
106         {
107           Right=Right+2;  //轉動伺服馬達兩個單位
108           servoRight.write(Right);
109           delay(50);
110         }
111         else
112         {
113           break;
114         }
115       }
116       break;
```

圖 3.35 Arduino 手臂往前程式碼

下圖 3.36，此段程式碼用於控制手臂，此段 case 為接受到藍牙訊息為’S’時(接收藍牙程式碼於行標 120 至 123 行；判斷收到訊息程式碼於行標 124 至 127 行)執行手臂往前之動作，至於外部 while 迴圈是為了使得手臂一直重複執行此動作。行標 129 至 134 此段程式碼為操控右邊的伺服馬達角度，第 129 行為設定安全轉動動範圍，去控制能轉動的最大值，第 131 行 Right=Right.2 為減少伺服馬達的角度，delay 的部分是為了延遲馬達轉動的速度，其時間單位為毫秒。

```
117       case 'S':  //臂往後
118         while(1)
119         {
120           if (I2CBT.available()>0)
121           {
122              cmd=I2CBT.read();  //讀取藍芽訊息
123           }
124           if(cmd!='S')
125           {
126             break;  //如果接受到訊息不為'S'則離開此While()迴圈
127           }
128           servoRight.attach(10);
129           if(Right>=50)  //設定安全轉動範圍，即轉動最小值不得小於50
130           {
131             Right=Right-2;  //轉動伺服馬達兩個單位
132             servoRight.write(Right);
133             delay(50);
134           }
135           else
136           {
137             break;
138           }
139         }
140         break;
```

圖 3.36 Arduino 手臂往後程式碼

下圖 3.37，此段程式碼用於控制手臂，此段 case 為接受到藍牙訊息為'A'時(接收藍牙程式碼於行標 144 至 147 行；判斷收到訊息程式碼於行標 148 至 151 行)執行手臂往前之動作，至於外部 while 迴圈是為了使得手臂一直重複執行此動作。行標 153 至 158 此段程式碼為操控底下的伺服馬達角度，第 153 行為設定安全轉動動範圍，去控制能轉動的最大值，第 155 行 Bottom=Bottom+4 為增加伺服馬達的角度，delay 的部分是為了延遲馬達轉動的速度，其時間單位為毫秒。

```
141     case'A': //底左轉
142       while(1)
143       {
144         if (I2CBT.available()>0)
145         {
146               cmd=I2CBT.read();  //讀取藍芽訊息
147         }
148         if(cmd!='A')
149         {
150           break;  //如果接受到訊息不為'A'則離開此While()迴圈
151         }
152         servoBottom.attach(3);
153         if(Bottom<=150)  //設定安全轉動範圍，即轉動最大值不得超過150
154         {
155           Bottom=Bottom+4;  //轉動伺服馬達四個單位
156           servoBottom.write(Bottom);
157           delay(50);
158         }
159         else
160         {
161           break;
162         }
163       }
164       break;
```

圖 3.37 Arduino 手臂底部往左程式碼

下圖 3.38，此段程式碼用於控制手臂，此段 case 為接受到藍牙訊息為’D’時(接收藍牙程式碼於行標 168 至 171 行；判斷收到訊息程式碼於行標 172 至 175 行)執行手臂往前之動作，至於外部 while 迴圈是為了使得手臂一直重複執行此動作。行標 177 至 182 此段程式碼為操控底下的伺服馬達角度，第 177 行為設定安全轉動動範圍，去控制能轉動的最大值，第 179 行 Bottom=Bottom.4 為增加伺服馬達的角度，delay 的部分是為了延遲馬達轉動的速度，其時間單位為毫秒。

```
165       case'D': //底右轉
166         while(1)
167         {
168           if (I2CBT.available()>0)
169           {
170              cmd=I2CBT.read();    //讀取藍芽訊息
171           }
172           if(cmd!='D')
173           {
174             break;  //如果接受到訊息不為'D'則離開此While()迴圈
175           }
176           servoBottom.attach(3);
177           if(Bottom>=30)  //設定安全轉動範圍，即轉動最小值不得小於30
178           {
179             Bottom=Bottom-4;
180             servoBottom.write(Bottom);    //轉動伺服馬達四個單位
181             delay(50);
182           }
183           else
184           {
185             break;
186           }
187         }
188         break;
```

圖 3.38 Arduino 手臂底部往右程式碼

下圖 3.39，此段程式碼用於控制手臂，此段 case 為接受到藍牙訊息為'O'時(接收藍牙程式碼於行標 192 至 195 行；判斷收到訊息程式碼於行標 196 至 199 行)執行手臂往前之動作，至於外部 while 迴圈是為了使得手臂一直重複執行此動作。行標 201 至 206 此段程式碼為操控底下的伺服馬達角度，第 201 行為設定安全轉動動範圍，去控制能轉動的最大值，第 203 行 Claw=Claw+2 為增加伺服馬達的角度，delay 的部分是為了延遲馬達轉動的速度，其時間單位為毫秒。

```
189       case'O': //開夾
190         while(1)
191         {
192           if (I2CBT.available()>0)
193           {
194              cmd=I2CBT.read();  //讀取藍芽訊息
195           }
196           if(cmd!='O')
197           {
198             break;  //如果接受到訊息不為'O'則離開此While()迴圈
199           }
200           servoClaw.attach(9);
201           if(Claw<=60)  //設定安全轉動範圍，即轉動最大值不得超過60
202           {
203             Claw=Claw+2;  //轉動伺服馬達二個單位
204             servoClaw.write(Claw);
205             delay(50);
206           }
207         }
208         break;
```

圖 3.39 Arduino 手臂開夾程式碼

下圖 3.40，此段程式碼用於控制手臂，此段 case 為接受到藍牙訊息為'C'時(接收藍牙程式碼於行標 212 至 214 行；判斷收到訊息程式碼於行標 216 至 219 行)執行手臂往前之動作，至於外部 while 迴圈是為了使得手臂一直重複執行此動作。行標 221 至 226 此段程式碼為操控底下的伺服馬達角度，第 221 行為設定安全轉動動範圍，去控制能轉動的最大值，第 223 行 Claw=Claw.2 為增加伺服馬達的角度，delay 的部分是為了延遲馬達轉動的速度，其時間單位為毫秒。

```
209       case'C': //關夾
210         while(1)
211         {
212           if (I2CBT.available()>0)
213           {
214                cmd=I2CBT.read();  //讀取藍芽訊息
215           }
216           if(cmd!='C')
217           {
218             break;  //如果接受到訊息不為'C'則離開此While()迴圈
219           }
220           servoClaw.attach(9);
221           if(Claw>=5)  //設定安全轉動範圍，即轉動最小值不得小於5
222           {
223             Claw=Claw-2;   //轉動伺服馬達二個單位
224             servoClaw.write(Claw);
225             delay(50);
226           }
227           else
228           {
229             break;
230           }
231         }
232         break;
```

圖 3.40 Arduino 手臂關夾程式碼

下圖 3.41，此段程式碼用於控制手臂，此段 case 為接受到藍牙訊息為'N'時(接收藍牙程式碼於行標 260 至 263 行；判斷收到訊息程式碼於行標 264 至 267 行)執行手臂往前之動作，至於外部 while 迴圈是為了使得手臂一直重複執行此動作。行標 269 至 274 此段程式碼為操控底下的伺服馬達角度，第 269 行為設定安全轉動動範圍，去控制能轉動的最大值，第 271 行 Left=Left.3 為增加伺服馬達的角度，delay 的部分是為了延遲馬達轉動的速度，其時間單位為毫秒。

```
257       case'N': //臂往下
258         while(1)
259         {
260           if (I2CBT.available()>0)
261           {
262              cmd=I2CBT.read();    //讀取藍芽訊息
263           }
264           if(cmd!='N')
265           {
266             break;  //如果接受到訊息不為'N'則離開此While()迴圈
267           }
268           servoLeft.attach(11);
269           if(Left>=85)    //設定安全轉動範圍，即轉動最小值不得小於85
270           {
271               Left=Left-3;    //轉動伺服馬達三個單位
272             servoLeft.write(Left);
273             delay(50);
274           }
275           else
276           {
277             break;
278           }
279         }
280         break;
```

圖 3.41 Arduino 手臂往下程式碼

下圖 3.42，此段程式碼用於控制手臂，此段 case 為接受到藍牙訊息為'U'時(接收藍牙程式碼於行標 236 至 239 行；判斷收到訊息程式碼於行標 240 至 243 行)執行手臂往前之動作，至於外部 while 迴圈是為了使得手臂一直重複執行此動作。行標 245 至 250 此段程式碼為操控底下的伺服馬達角度，第 245 行為設定安全轉動動範圍，去控制能轉動的最大值，第 247 行 Left=Left+3 為增加伺服馬達的角度，delay 的部分是為了延遲馬達轉動的速度，其時間單位為毫秒。

```
233       case'U': //臂往上
234         while(1)
235         {
236           if (I2CBT.available()>0)
237           {
238              cmd=I2CBT.read();  //讀取藍芽訊息
239           }
240           if(cmd!='U')
241           {
242             break;  //如果接受到訊息不為'U'則離開此While()迴圈
243           }
244           servoLeft.attach(11);
245           if(Left<=150)  //設定安全轉動範圍，即轉動最大值不得超過150
246           {
247             Left=Left+3;   //轉動伺服馬達三個單位
248             servoLeft.write(Left);
249             delay(50);
250           }
251           else
252           {
253             break;
254           }
255         }
256         break;
```

圖 3.42 Arduino 手臂往上程式碼

第四章　移動式手臂完整操作

4.1 地圖規劃及介紹

如圖 4.1，這是移動式手臂的實作地圖，利用膠帶所構成的起點、路線和終點，而移動式手臂可以準確的從起點沿著路線到達終點，再從終點取得物品之後，沿著原路回到起點，並且把物品放置於起點上。如圖 4.2，正方形是地圖的起點，起點中央為取得物品後所放置的地方。

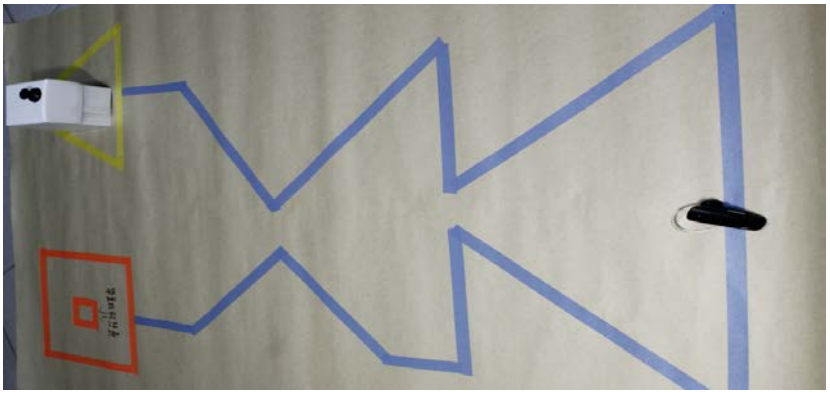

圖 4.1 整張地圖

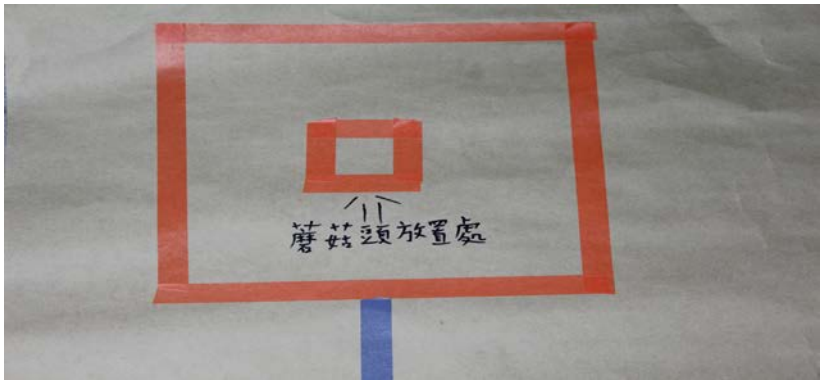

圖 4.2 地圖起點

如圖 4.3，這是路線中設有的障礙，我們將藍牙耳機作為行進中的路障，移動式手臂必須將障礙物移至路線外放置並繼續前進，直到抵達終點為止。如圖 4.4，三角形是地圖的終點，終點處所放置的物品是一個蘑菇頭，為了要表現我們的移動式手臂可以能屈能伸，可拿高也可放低，我們將物品放置盒子上端，再操作移動式手臂去取得物品。

圖 4.3 途中障礙物

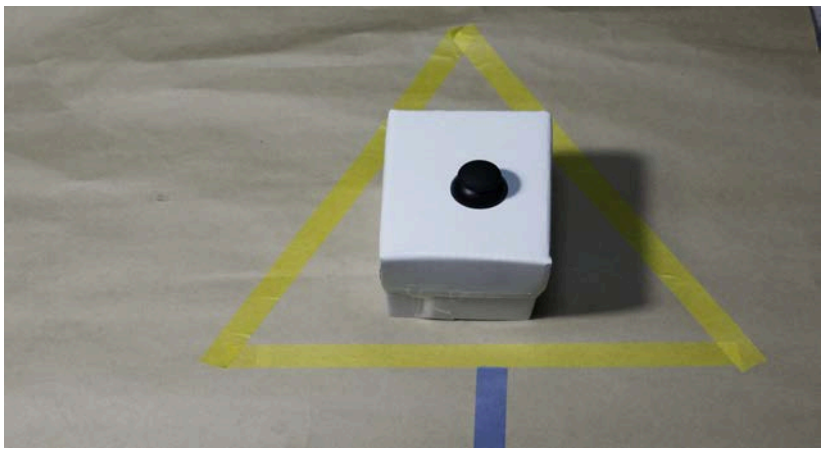

圖 4.4 地圖終點

4.2 實際完整操作

我們可以看到圖 4.5 至圖 4.6 表示，移動式手臂從起點出發，準確的沿著膠帶前進，車體可以左轉、右轉、前進、後退，而它的手臂可以收在中間放著，避免長期舉著手臂而造成馬達消耗或受損。

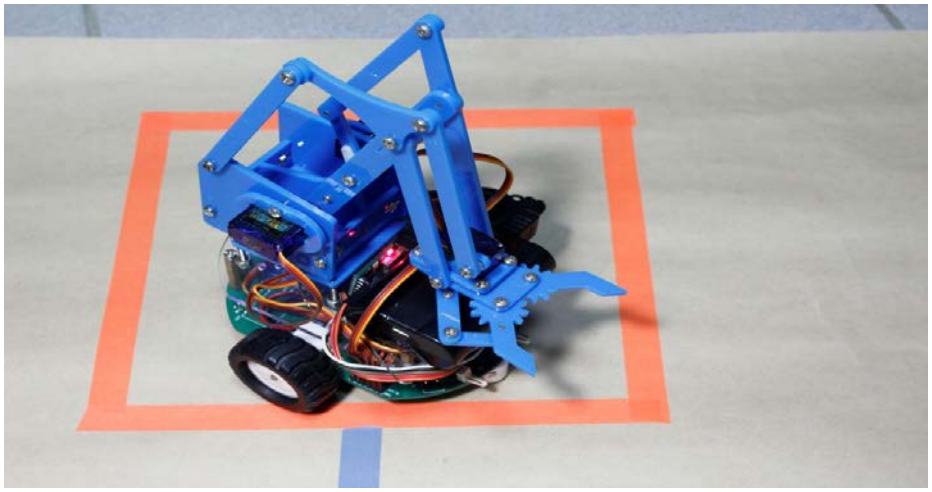

圖 4.5 準備出發

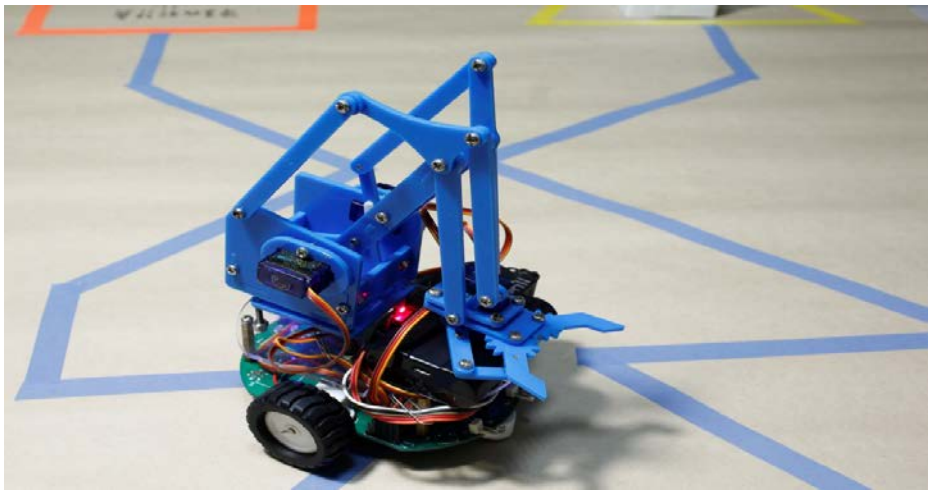

圖 4.6 正式出發

我們可以看到圖 4.7 至圖 4.10 表示，在移動式手臂持續移動中，途中遇到了一個障礙，它可以不用讓車體轉向，只須讓手臂底盤轉向就可以輕鬆把障礙移除，並且輕放在旁邊，而不是直接拋出去，最後導正姿態，繼續前進。

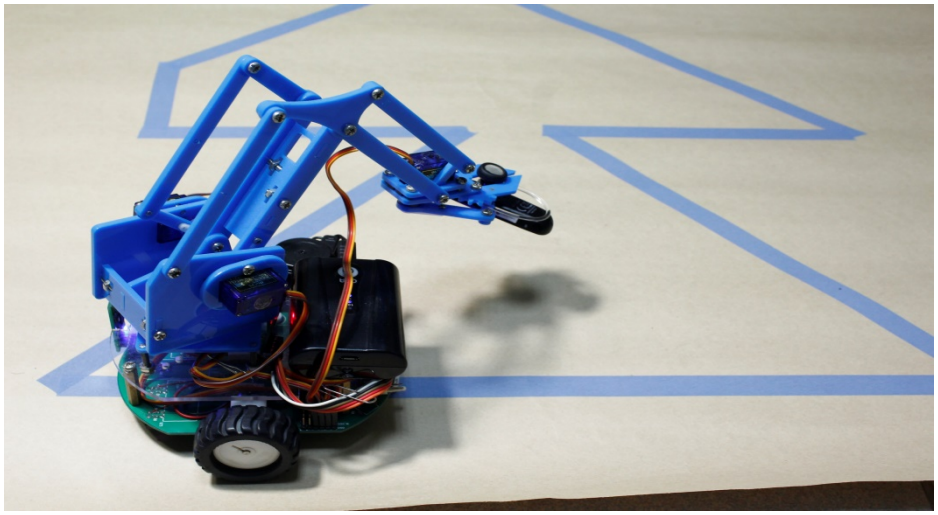

圖 4.7 夾起路障

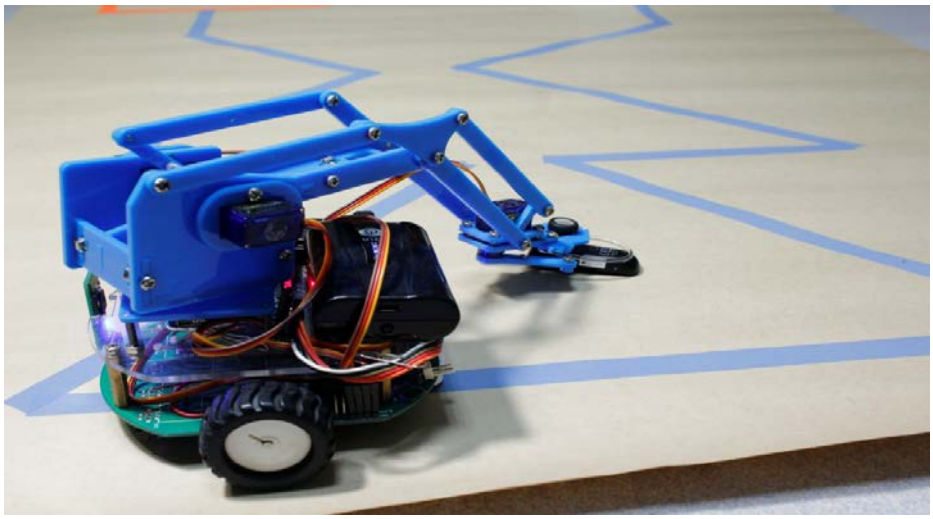

圖 4.8 把路障移至線路旁

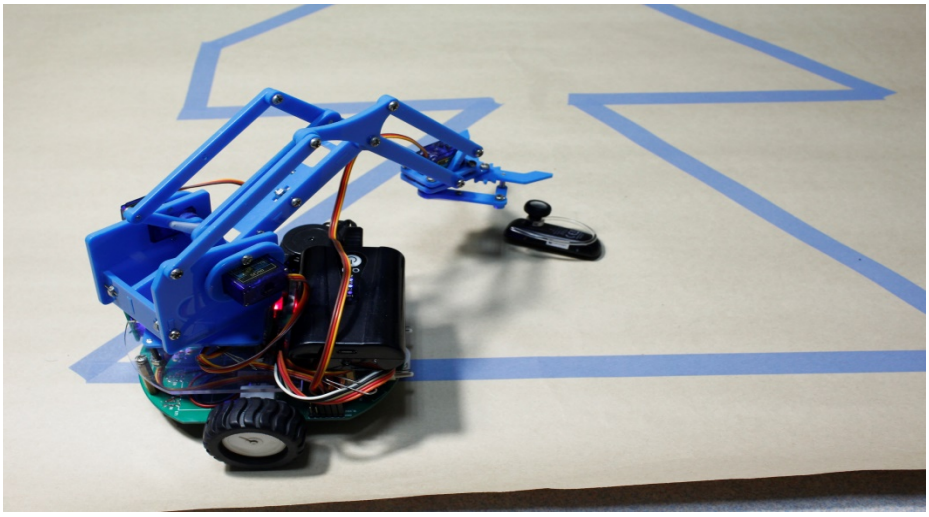

圖 4.9 放置完成

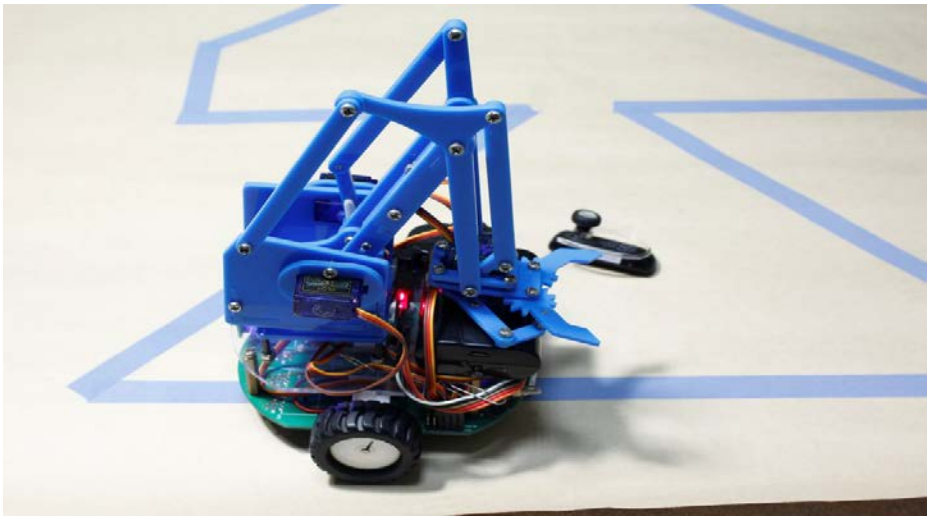

圖 4.10 導正姿態

我們可以看到圖 4.11 至圖 4.14 表示，當移動式手臂到達終點時，要先取好與白色盒子的距離，之後將手臂升到與蘑菇頭相等的高度，並把蘑菇頭夾起來，之後倒車轉向，返回起點將物品放置中間。

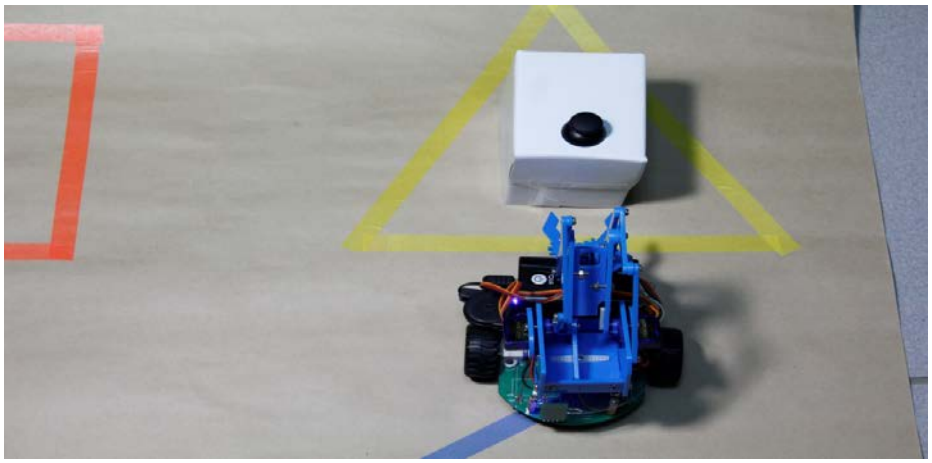

圖 4.11 到達終點

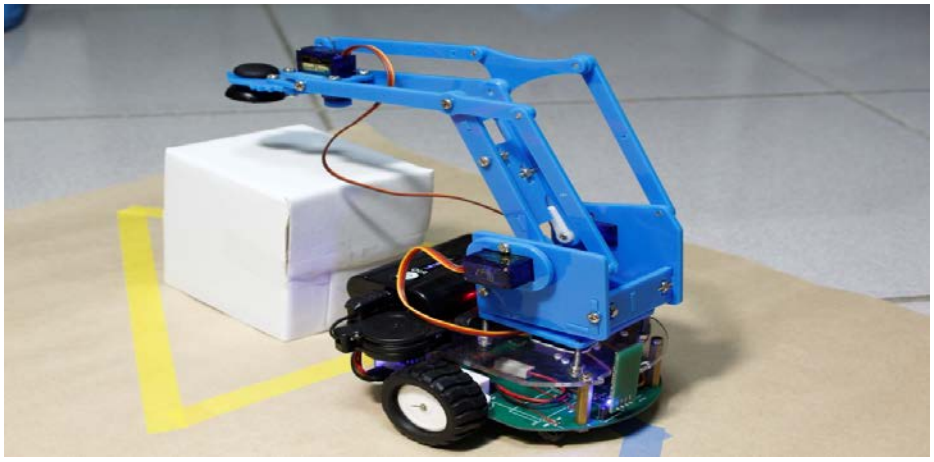

圖 4.12 調整手臂高度

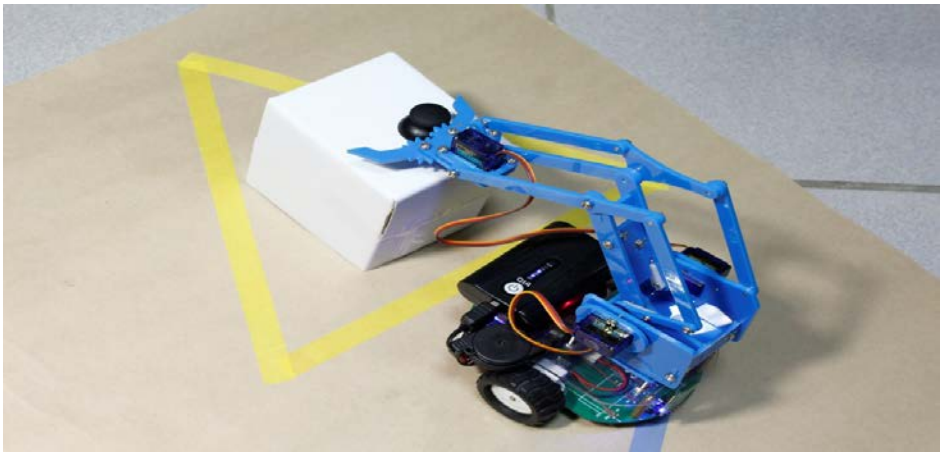

圖 4.13 夾起物品

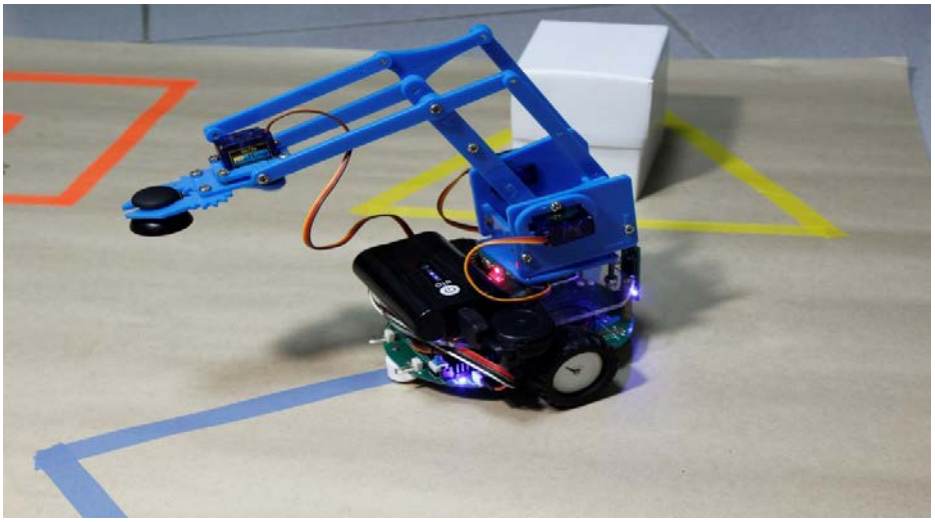

圖 4.14 準備回程

我們可以看到圖 4.15 至圖 4.18 表示，移動式手臂夾著物品沿著原來路線返回到起點，在抵達起點後，可以經由車體和手臂的調整，將物品準確地放置到指定的位子上，完成這次實作。

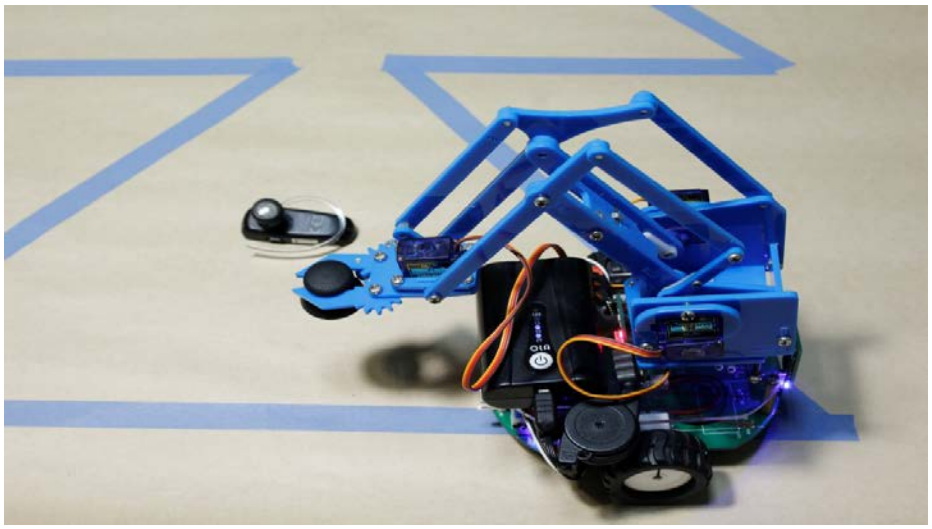

圖 4.15 返回起點的路中

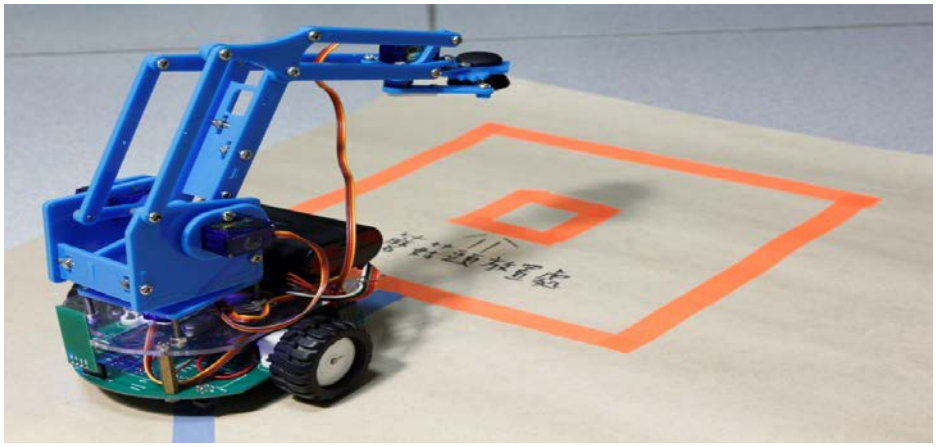

圖 4.16 到達起點

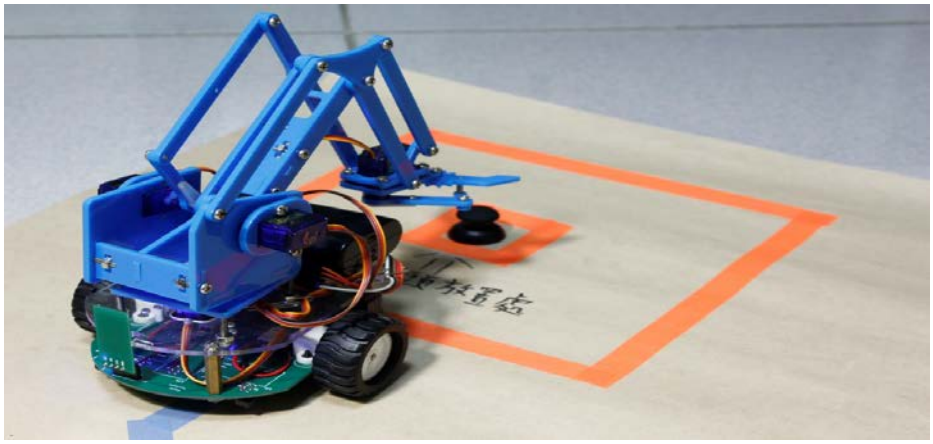

圖 4.17 放置物品

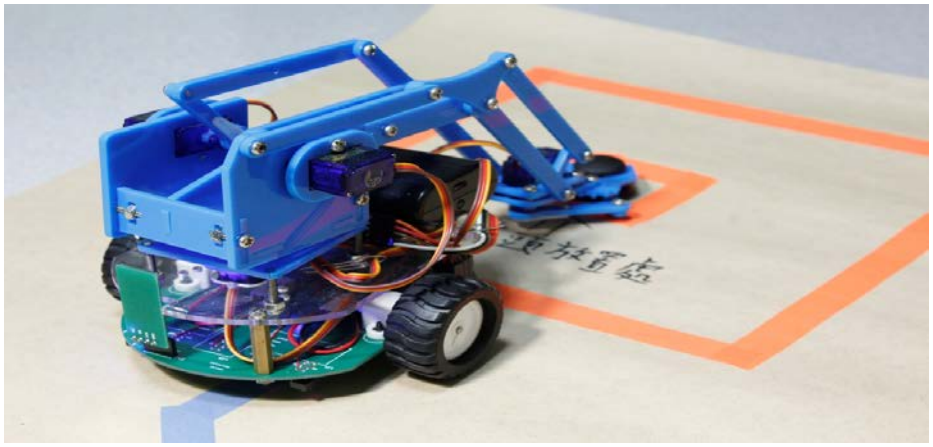

圖 4.18 完成

第五章　結論

此次的專題研究是由一塊 Arduino UNO 晶片燒錄在自製的自走車電路板上，在與 ARM 機器手臂的連接，並用寫一個 APP 去做遠端操控。我們在研究的過程裡發現實作上的比預想還要來的困難，遇到了我們沒預測到的問題，但慶幸的是我們經由學長還有教授的從中指導，不管是 Arduino Software、App Inventor 2、硬體設備的整合所產生的問題，我們最後都可以依照學長跟指導教授所教導知識去做解決。

經由這次研究我們大家都有不錯的收穫，讓我們更加了解了資訊產業的未來發展，相信我們的成品在未來裡可以擁有更多的發展空間，像是增加伸縮支架提高機動性，這樣就可以克服地形上更多的障礙；或是將藍牙模組改為 WIFI 模組這樣可以使操作距離大幅的提升，最後可以在加裝上更多的智慧模組，像是安裝超音波感測器、紅外線感測器，使他可以自行避開更多的障礙。我們之後也會更積極投入這項研究，努力實現我們大家的理想。

參考文獻

[1] Arduino . Wikiwand

http：//www.wikiwand.com/zh.hk/Arduino

[2] Arduino UNO 控制板簡介. 網昱多媒體

http：//swf.com.tw/?p=569

[3] 藍牙. 維基百科，自由的百科全書 . Wikipedia

https：//zh.wikipedia.org/wiki/%E8%97%8D%E7%89%99

[4] 藍牙 Bluetooth Archives . 網昱多媒體

http：//swf.com.tw/?tag=%E8%97%8D%E7%89%99.bluetooth

[5] App inventor 教學講義 chapter1 - SlideShare

http://www.slideshare.net/hotwusir/app-inventor-chapter1