

國 立 虎 尾 科 技 大 學
機 械 設 計 工 程 系
專 題 製 作 報 告

大容量 3D 列印機系統設計與
實作

Design and Implementation of a Large Volume 3D Printer System

指導教授：嚴家銘老師

班級：四 設計 三乙

學生：潘巧昕 (40523206)

林韋翔 (40523219)

張堡祺 (40523226)

賴瑋傑 (40523243)

摘要

系上課程可分為機械機構、電子元件和程式軟體三大部分，為綜合以上內容本專題參考相當多的資料，在此同時發現系上各個實驗室內並無 3D 列印機有辦法一次列印高且大面積的零件，最後專題確定為大容積 3D 列印機系統設計與實作。在藉由實體的組立來從中發現問題並加以解決，與此同時和虛擬之間的整合，可以用虛擬程式先行驗證模擬列印，最後對比實體列印和虛擬列印之間所存在的不同，先以 Onshape 繪製零件組合做為實體機構的範本同時作為 V-rep 動態模擬的物件，電子元件的程式用 Arduino base 來對步進馬達和噴嘴等進行控制，再由 V-REP、Python 的結合為虛擬模擬通過以上方式來達到對所學內容的結合。

目錄

摘要.....	2
第一章 簡介與研究動機	6
第二章 設計流程	7
2.1 執行規劃	7
2.2 機械設計	8
2.3 電系設計	8
2.3.1 步進馬達	8
2.3.2 驅動器 TB6600	9
2.3.3 接近開關與極限開關之差別	10
第三章 虛擬介面設計	11
3.1 設計流程	11
第四章 V-REP 模擬	13
4.1 虛擬控制介面之軟體應用	13
4.2 問題與解決方法	20
4.3 V-rep 粒子模擬	22
第五章 軟韌體設計	26

第六章 組立流程	27
6.1 機構組立	27
6.1.1 基本骨架安裝	27
6.1.2 Z 軸組裝	28
6.2 傳動系統組立	30
6.3 電控系統裝配	33
6.3.1 步進馬達驅動與測試.....	33
6.3.2 測試 X 軸移動距離.....	36
6.3.3 馬達正反轉修改	37
6.3.4 X、Y 極限開關.....	38
第七章 軟體選用與配置	47
7.1 Cura	47
7.1.1 基本設定	47
7.1.2 列印參數設定	49
7.2 Pronterface.....	51
7.3 Python.....	53
7.4 Arduino.....	54

7.5 V-rep	55
7.6 Onshape	57
第八章 系統定位與校正	60
8.1 校正 X、Y 軸垂直水平精度	60
8.2 Z 軸平台校正	61
8.3 X Y Z 定位精度測試	66
第九章 列印測試	68
附錄	71

第一章 簡介與研究動機

隨著科技的進步，列印立體的物品已經不再是難事，3D 列印是一種趨勢，現在廣泛應用在生活及醫療上。此研究與列印技術一樣，皆從無到有，首先設計各部位零件再進行虛擬與實體的組裝模擬，到最後列印測試。目前普遍機種依結構型式可分為 Delta 型及 XY-Z 型式，雖然列印高度有些可達 600mm，但在大型物件下還是有所限制，這個研究以列印大容量系統為目標，XY-Z 型式為結構，底部有安裝腳輪可視環境情況自由移動移動，最大列印高度為 1000mm。

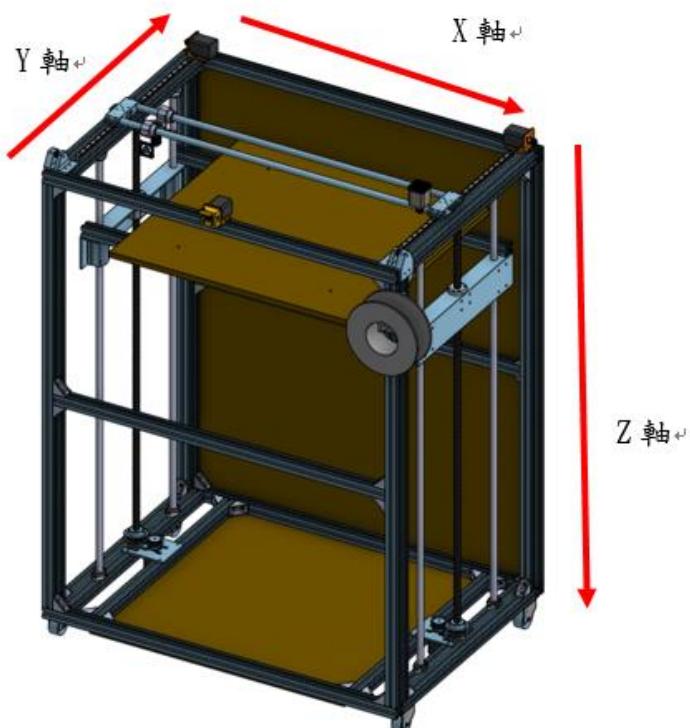


圖 1.1 3D 列印機虛擬圖

第二章 設計流程

2.1 執行規劃

上網找尋專題相關資料有了初步的構想，參考其他人 3D 列印機的運動方式和電路設置，列出所需要的材料且製成表並進行採購，在 Onshape 上建立模型並組裝做為日後實體建立的參考依據。

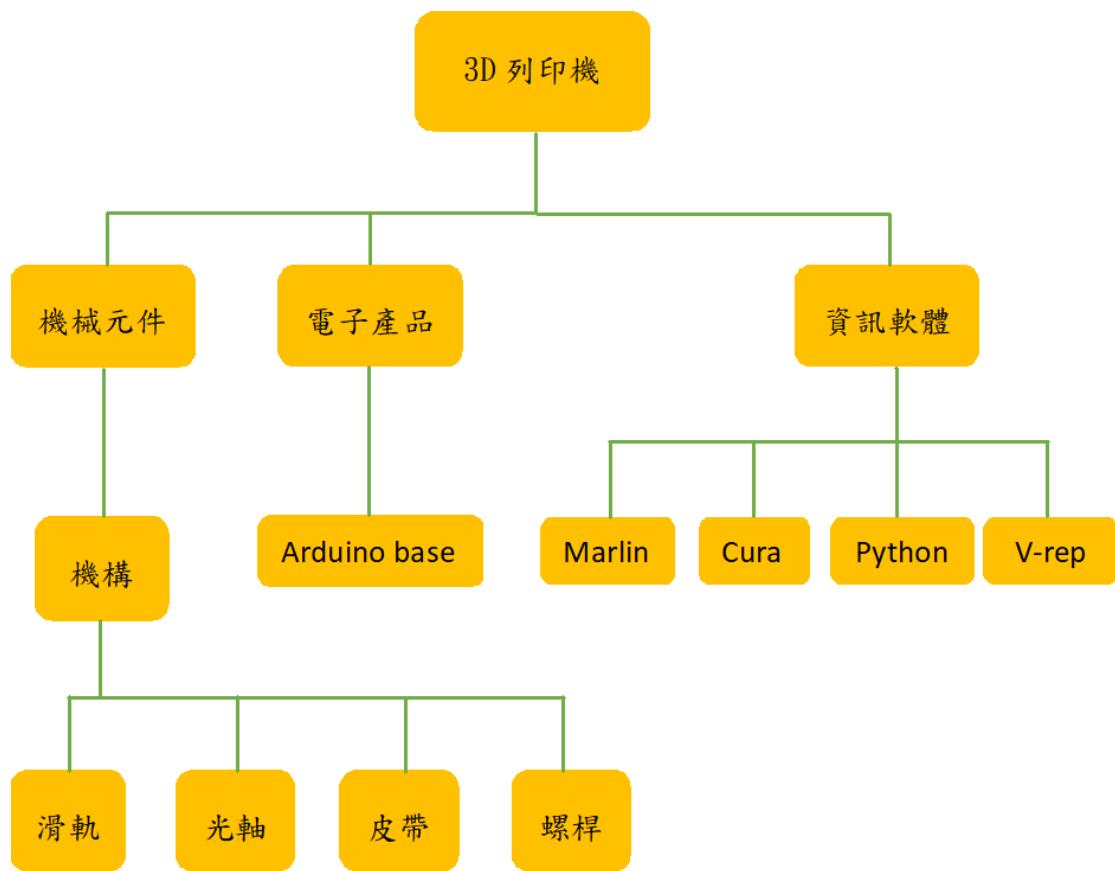


圖 2.1 3D 列印機系統設計

2.2 機械設計

將機台設計為箱型結構，與直角坐標系相似，各有馬達控制X、Y、Z軸，但箱型結構的平台是Z方向上下移動，噴頭由X-Y軸來控制，此架構較穩定，且Z軸行走路徑較長所以可列印物體的高度可以更廣，平台由Z軸控制較不易搖晃，列印品質較其他機構更為良好。

2.3 電系設計

在3D列印機上總共有六顆馬達，步進馬達通常會標示電壓和電流，一般不會將馬達直接與電源連接，而是連接驅動器，由驅動器傳送脈波訊號來控制電力，使馬達以一定的角度運作。驅動器是將所需要的輸出條件轉換為命令形式送至馬達，我們使用的驅動器為TB6600，它有六種細分方式，可驅動4線、六線、八線步進電機。

2.3.1 步進馬達

步進馬達有分成許多種類，其中有相數和接線數量的差別，有兩相四線式、兩相六線式…等，相數代表有多少組線圈，相數越多基本步進角就越小。而這個研究所使用的是兩相四線式步進馬達，即

基本步級角為 1.8 度(200 步為一圈)，且四線步進馬達為雙極馬達，雙極的意思是 A 相位跟 B 相位的電流會有兩個方向。馬達內部有相互嚙合的定子與轉子，對馬達的線圈通電流，切換通電順序產生不同方向的磁性使馬達中的永久磁鐵因為電流產生相對應不同極性而有轉動的結果。

1. A+通電，A-為出口(A+為正極，A-為負極)
2. B+通電，B-為出口(B+為正極，B-為負極)
3. A-通電，A+為出口(A-為正極，A+為負極)
4. A-通電，A+為出口(B-為正極，B+為負極)

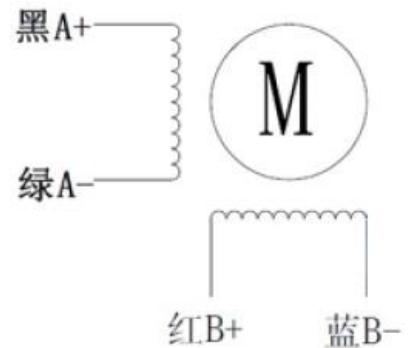


圖 2.2 馬達相位

2.3.2 驅動器 TB6600

為了控制步進馬達，需要使用驅動器，將所需要的輸出條件轉換為命令形式送至馬達。兩相步進馬達驅動器，支持速度和方向的控制，有八種電流控制及七種微步可調，例如 1/8 細分、1/16 細分、1/32 細分。



圖 2.3 TB6600 驅動器

2.3.3 接近開關與極限開關之差別

接近開關的原理是將檢測對象的移動資訊或存在資訊轉換成電性訊號。轉換成電性訊號的檢測方式包括：藉由電磁感應使金屬物檢測對象產生渦電流，並利用渦電流進行檢測；捕捉隨著檢測物體接近而變化的電容量…依據 JIS 的定義，感測器非接接觸檢測物體的情況下，能檢測是否有檢測物體接近者，統稱為接近開關

極限開關的原理是一個機電設備，其中有一個和一個接點機械連結的致動器。若物體接觸致動器，就會觸動接點，打開或是關閉電氣的連接。而 Z 軸使用接近開關的原因是 Z 軸平台是與噴頭去做碰觸，所以我們運用接近開關，因為接近開關不需要與其他東西需做碰觸，只需要用鐵製品去感應他，感應距離通常為 2mm 左右，且比極限開關高度調整容易許多。

第三章 虛擬介面設計

透過 Python 語言與 V-rep 之 API 設計開發訊號進行通訊及共用資訊，再利用 PyQt 的介面設計功能，將 Python 程式碼與 PyQt 虛擬元件進行連接，初步設計理念為透過虛擬的動態模式，觀察機器運行的軌跡及問題。

3.1 設計流程

- 首先 Python 基本架構後，須使 Python 和 V-rep 進行通訊，透過 V-rep 之 API 中提供的 **simxStart**，給予 ip、port 及部分設定讓通訊連接，再用 **simxFinish** 輔助 API 監視通訊或者作結束通訊的動作。
- 利用 PyQt 程式介面設計繪製出概念，有分割介面、伸縮介面、彈出介面、按鈕、選項、下拉式選單及輸出文字介面.... 等多機能，介面設計上相當容易上手。

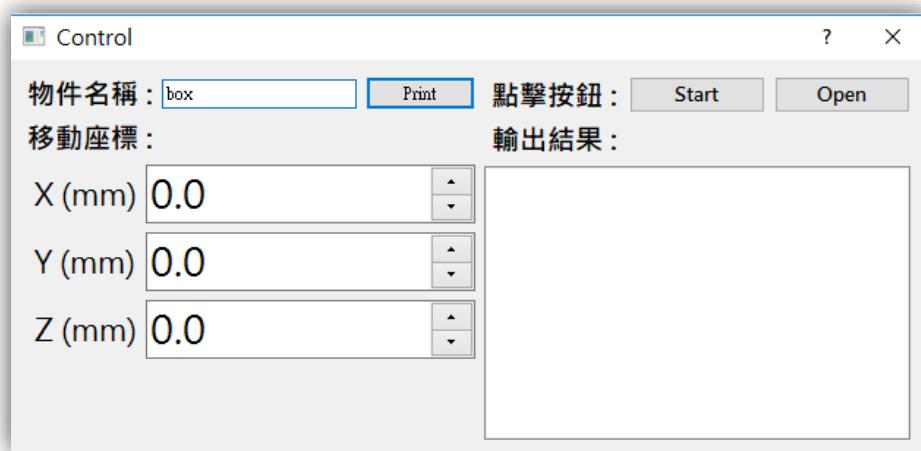


圖 3.1 介面之概念草圖設計

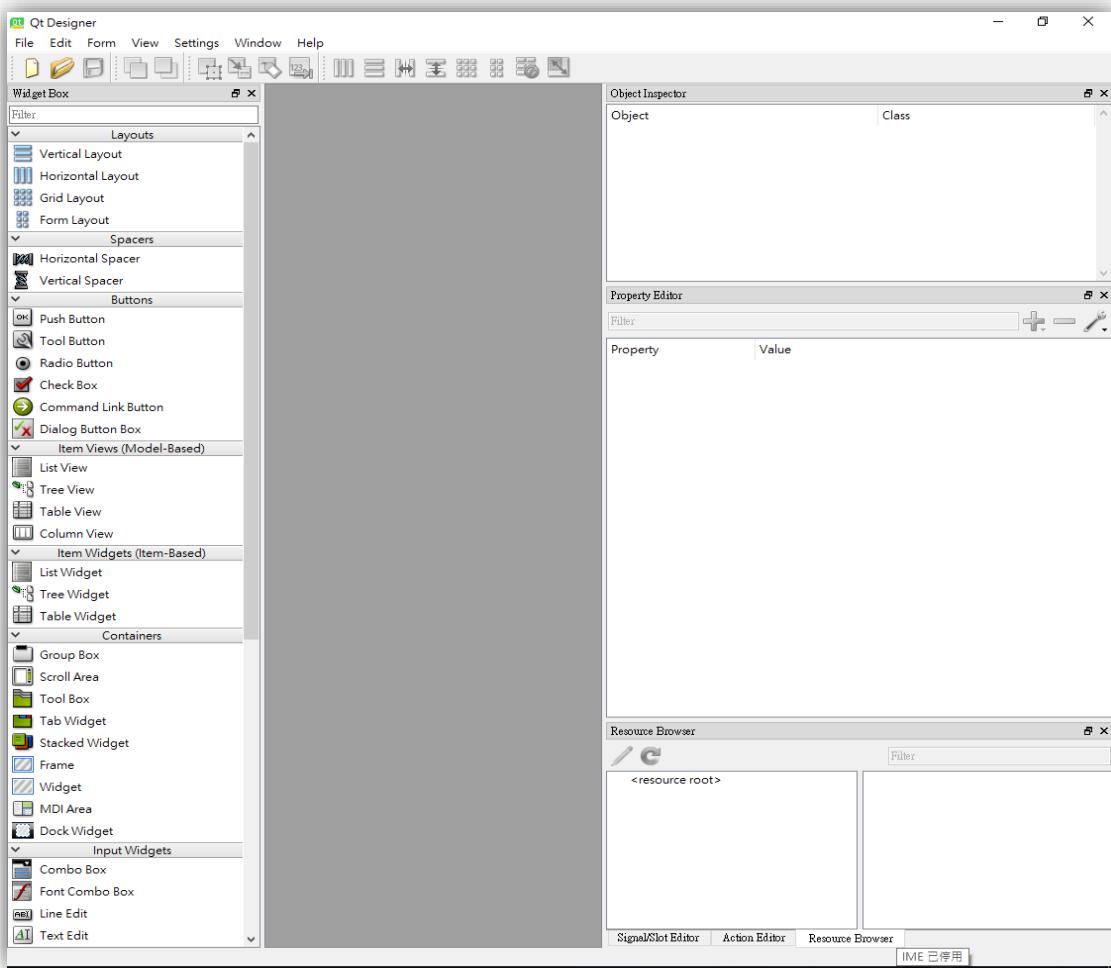


圖 3.2 介面之表現

3. 繼初版介面設計後撰寫程式，經由按鈕啟動 Python 程式碼為目的，進行 V-rep 的物件控制動作，而控制物件須透過 V-rep 之 API 中獲得物件的座標資訊。

simxGetObjectHandle - 鎖定物件

simxGetObjectPosition - 獲得物件座標資訊

simxSetObjectPosition - 設定物件座標資訊

第四章 V-REP 模擬

為 V-rep & PyQt & Python 之軟體結合，以虛擬的方式模擬測試結果，主要以 Python 撰寫程式，並且透過 V-rep API 與物件進行協定，達到動態模擬的效果。首先為計算快速，用簡化的塊件代替原本零件作動，但為避免外觀失真將塊件隱藏保留原本零件外觀，再用 python 導入 V-REP 下達路徑，控制 3D 列印機的噴嘴。

4.1 虛擬控制介面之軟體應用

虛擬介面操作過程

1. 開啟 V-rep 動態模擬介面如圖 4.1



圖 4.1 v-rep 動態模擬介面

2. 開啟虛擬控制介面如圖 4.2

控制介面主要分為控制圖 4.2 和模擬圖 4.3 兩種模式

[控制模式]

物件名稱 - 鎖定移動目標物件

原點復歸 - 物件往自定義虛擬物件(0, 0, 0)一直線移動。

參數控制 - 輸入 XYZ 移動量進行相對運動。

按鈕控制 - 點擊對應的軸向按鈕進行移動。

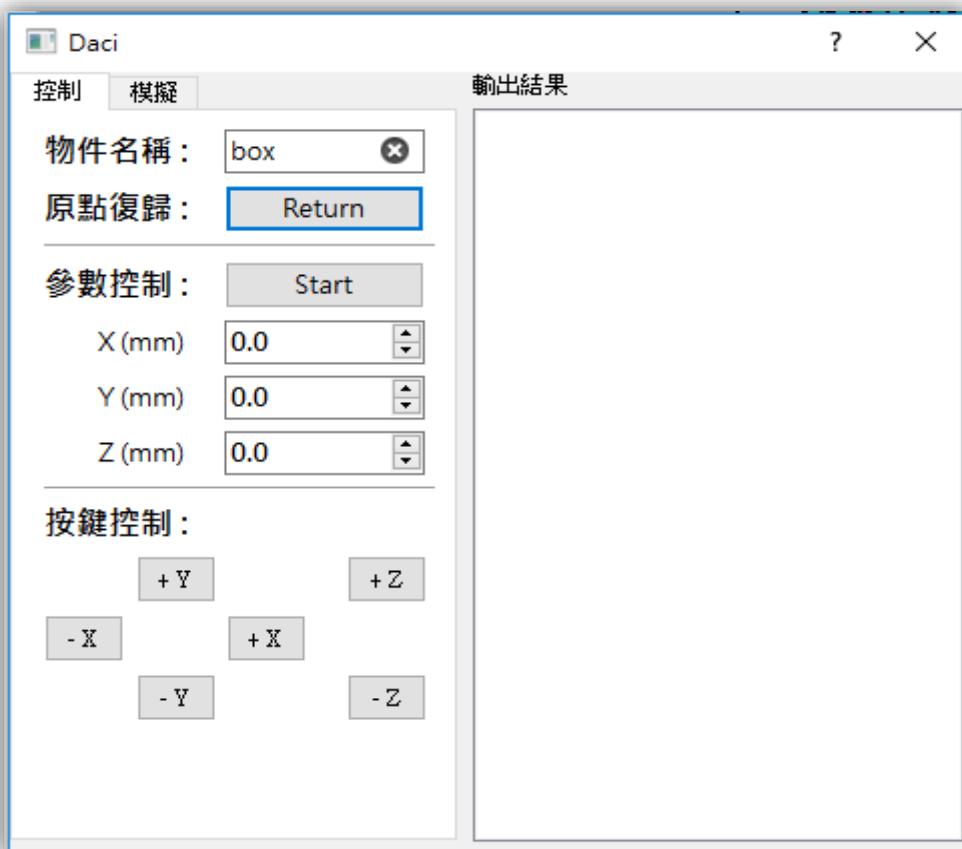


圖 4.2 虛擬控制介面-控制模式

[模擬模式]

原點復歸 - 與控制模式較為不同之處是分割式原點復歸，將

軸向分為 XY/Z 進行兩段式原點復歸。

開啟檔案 - 讀取 “.gcode” 文檔，進行動態模擬測試。

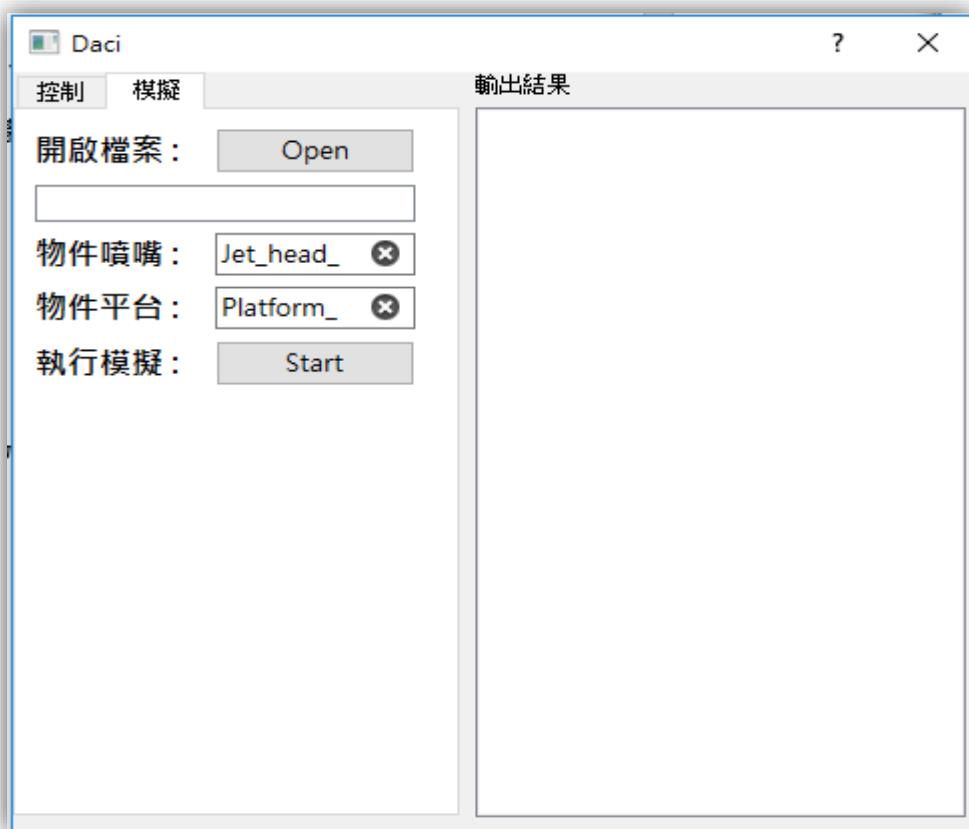


圖 4.3 虛擬控制介面-模擬模式

3. 於介面 A 開啟 3D 列表機 TTT 檔

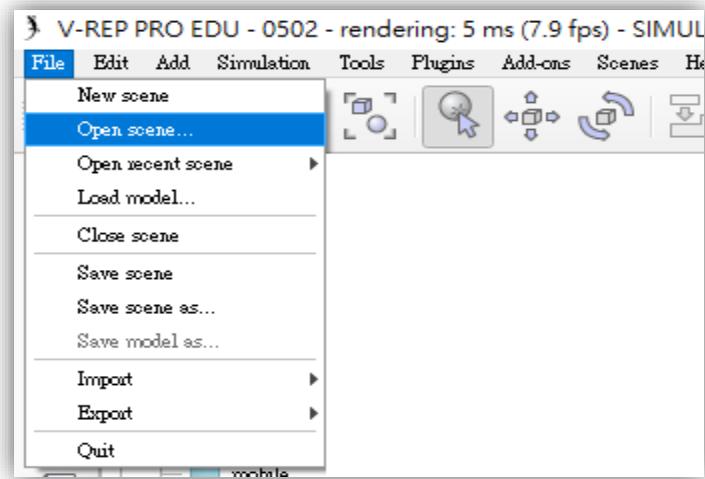


圖 4.4 開啟.ttt 檔

4. 建立一個虛擬模塊 box0 (原點復歸用)

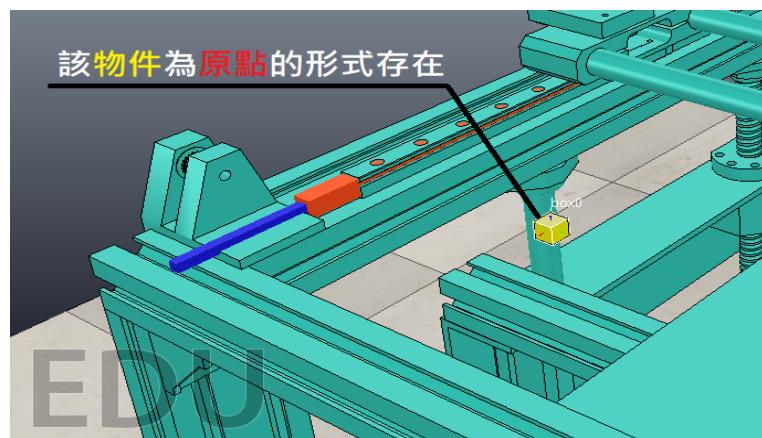


圖 4.5 建立模塊

5. 開始執行動態模擬 (Start/resume simulation)

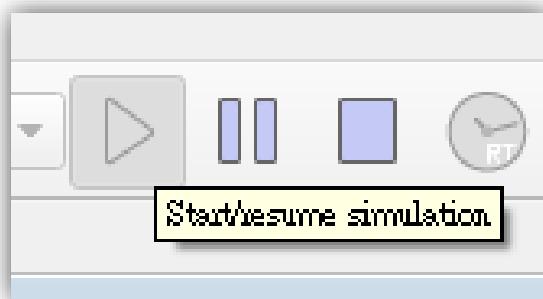


圖 4.6 執行模擬

6. 於控制模式介面輸入物件名稱，再點擊 Return 執行原點復歸

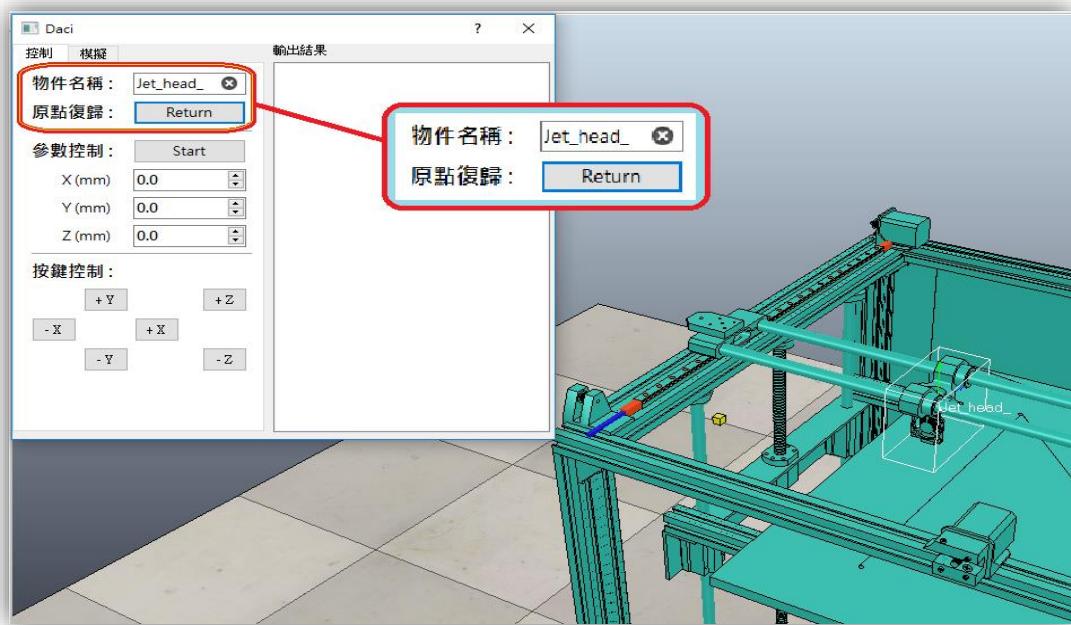


圖 4.7 復歸

7. 於介面 B 輸入物件名稱及 XYZ 移動量，再點擊 Start 執行動作

(1) 輸入物件名稱

(2) 三軸移動座標

(3) 點擊 Start 執行

(4) 物件移動至指定座標並且回報當前座標，即為完成

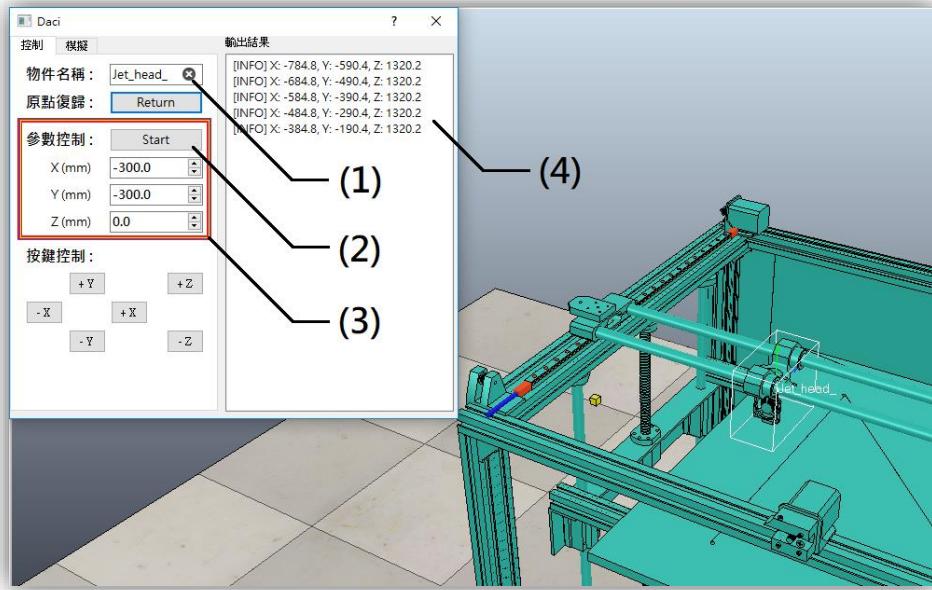


圖 4.8

8. 重複步驟六，原點復歸後將介面 B 切換至模擬模式

9. 點擊開啟檔案，選取.gcode 文檔，並執行

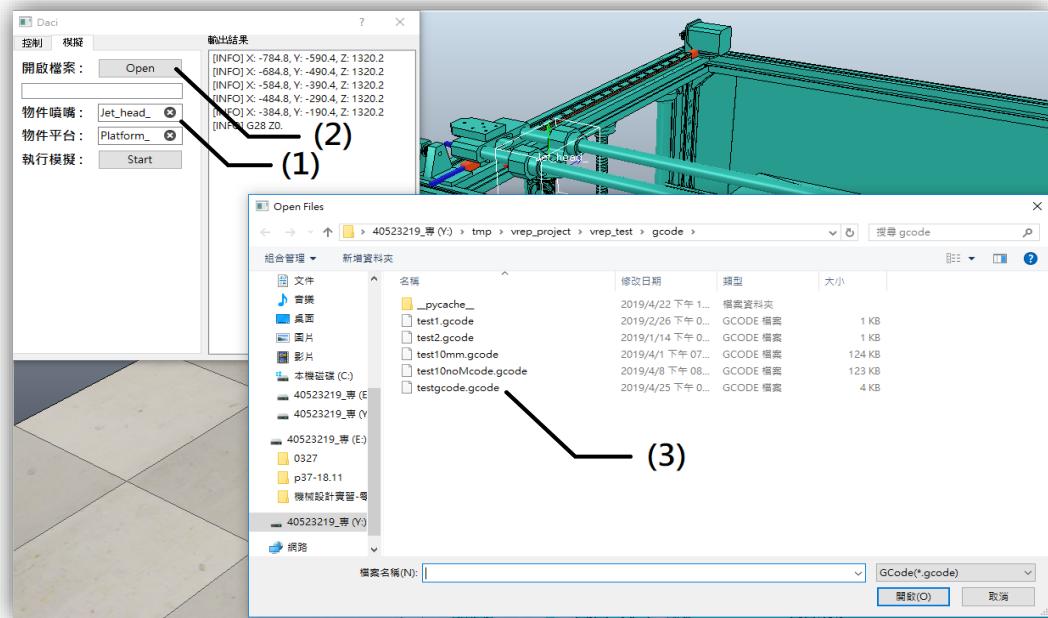


圖 4.9

模擬介面狀態

物件鎖定成功 - [INFO] print object OK!!!

物件完成復歸 - [INFO] Have arrived at the destination

物件回饋座標 - [INFO] X: , Y: , Z:

物件水平座標原點復歸 - [INFO] G28 X0. Y0.

物件垂直座標原點復歸 - [INFO] G28 Z0.

移動座標尚未輸入 - [WARN] Please set value!!

移動座標值過小 - [WARN] The value is too small!!

(總移動量至少 5mm 以上)

4.2 問題與解決方法

問題：如何抓取物件並且移動物件？

解：利用 V-rep API 指定物件進行互動，搜尋關鍵字

【vrep python】。API 網址：

<http://www.coppeliarobotics.com/helpFiles/en/remoteApiFunctionsPython.htm>

simxGetObjectHandle – 鎖定物件

simxGetObjectPosition – 獲得物件座標資訊

simxSetObjectPosition – 設定物件座標資訊

問題：如何使 V-rep 定位原點座標？

解：在 V-rep 中已經有本身的座標，於是給予它一個固定的

虛擬物件，將其表示為原點座標(0, 0, 0)，再使物件以相

對位置進行復歸動作。

問題：V-rep 物件無法照指定座標及斜率移動（或物件跳針）

解：注意 V-rep 的環境設定，預設移動單位為公尺，將使用者

輸入座標單位，釐米轉為公尺。

問題：控制物件(噴嘴)位置不精準，初步認為受到 V-rep 物件

層次約束關係，使控制位置受到影響。

解：將 V-rep 中的組合件之間的從屬關係(包含 Dummy)，重新檢查一遍，修正後即可正常運行至準確位置。

問題：Gcode 碼呈現方式為絕對座標

解：因此每次移動都先抓取距離原點的相對位置

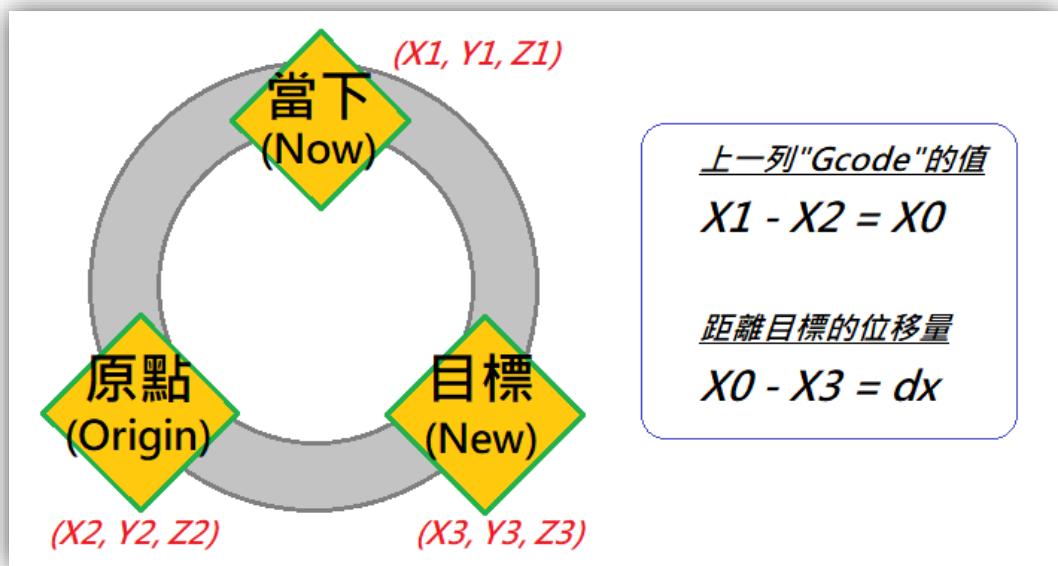


圖 4.10

4.3 V-rep 粒子模擬

在軸的選用上有 Revolute joints(旋轉軸)、Prismatic joints(平移軸)Spherical joints(球型接頭)，因噴嘴和平版的移動只有平移，故用平移軸來設定 x, y, z 移動時的自由度。

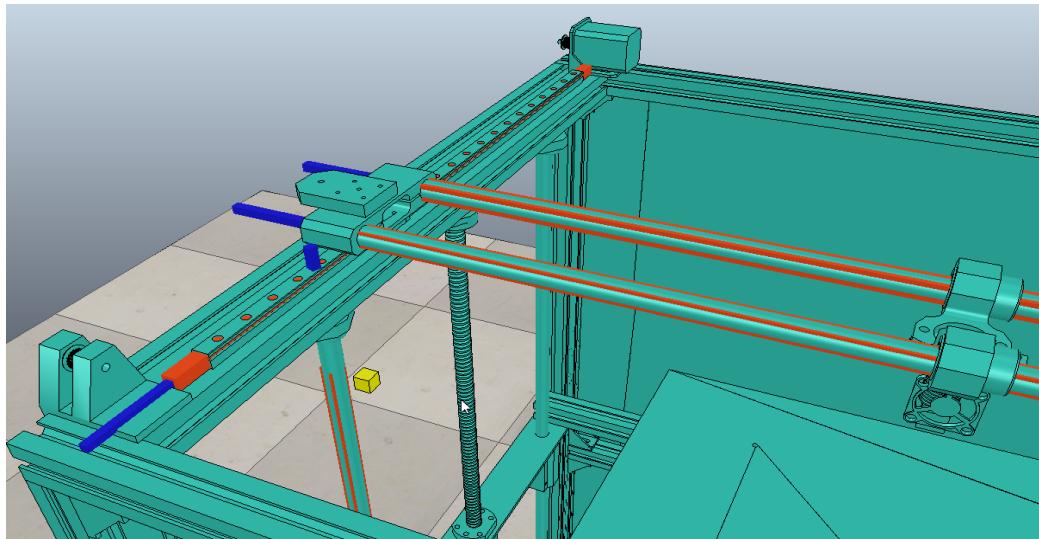


圖 4.11 加入平移軸

在模擬擠料的部份，是用 V-rep 範例的 components 中 locomotion and propulsion 裡的 Propeller 的粒子模組表示擠出塑料，以下是粒子的腳本參數設定介面。

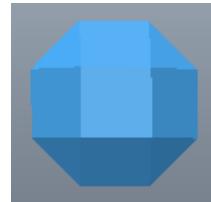


圖 4.12 粒子

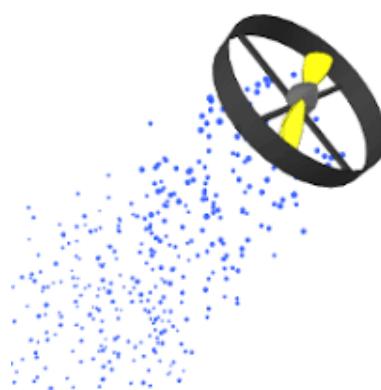


圖 4.13 粒子模組

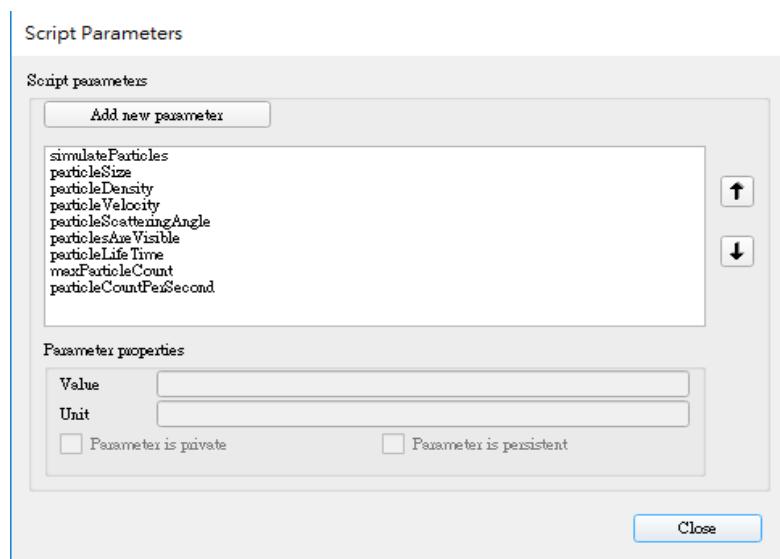


圖 4.14 粒子腳本參數設定

表 4.1 粒子參數說明

名稱	說明
simulateParticles	模擬粒子
particleSize	粒子大小
ParticleDensity	粒子密度
ParticleVelocity	粒子速度
ParticleScatteringAngle	粒子散射角
ParticleAreVisible	粒子可見
ParticleLifeTime	粒子壽命
MaxParticleCount	最大粒子數量
particleCountPerSecond	粒子每秒數量

目前針對腳本程式的修改有兩處，對 params 的部分不確定是不是對應 V-rep 的粒子的 API ，對 pos 的部分則控制了粒子是從單一位置射出。

```

26     notFullParticles=0
27     params={0,0,0,0,0,0,0,0,0,1000,0,0,0,1,1000,1000,1000,0}
28 end
29
30 function sysCall_cleanup()
31 end
32
33
34 function sysCall_actuation()
35     local t=sim.getSimulationTime()
36     sim.setJointPosition(propellerJoint,t*10)
37     s=sim.getObjectSizeFactor(propeller) -- current size factor
38     if (s~is) then
39         if (particleObject) then
40             sim.removeParticleObject(particleObject)
41             particleObject=nil
42         end
43         particleSize=sim.getScriptSimulationParameter(sim.handle_se
44         is=s
45     end
46     ts=sim.getSimulationTimeStep()
47
48     if (particleObject==nil)and(simulateParticles) then
49         particleObject=sim.addParticleObject(type,particleSize,part
50     end
51
52     m=sim.getObjectMatrix(propeller,-1)
53     particleCnt=0
54     pos={0,0,0}
55     dir={0,0,1}
56
57     requiredParticleCnt=particleCountPerSecond*ts+notFullParticles
58     notFullParticles=requiredParticleCnt % 1
59     requiredParticleCnt=math.floor(requiredParticleCnt)
60     while (particleCnt<requiredParticleCnt) do
61         -- we want a uniform distribution:
62         x=(math.random()-0.5)*2
63         y=(math.random()-0.5)*2
64         if (x*x+y*y<=1) then
65             if (simulateParticles) then
66                 pos[1]=x*0*s
67                 pos[2]=y*0*s
68                 pos[3]=-particleSize*0.1

```

圖 4.15 粒子模組腳本

這是當 pos 不為零時造成粒子射出時位置的不固定

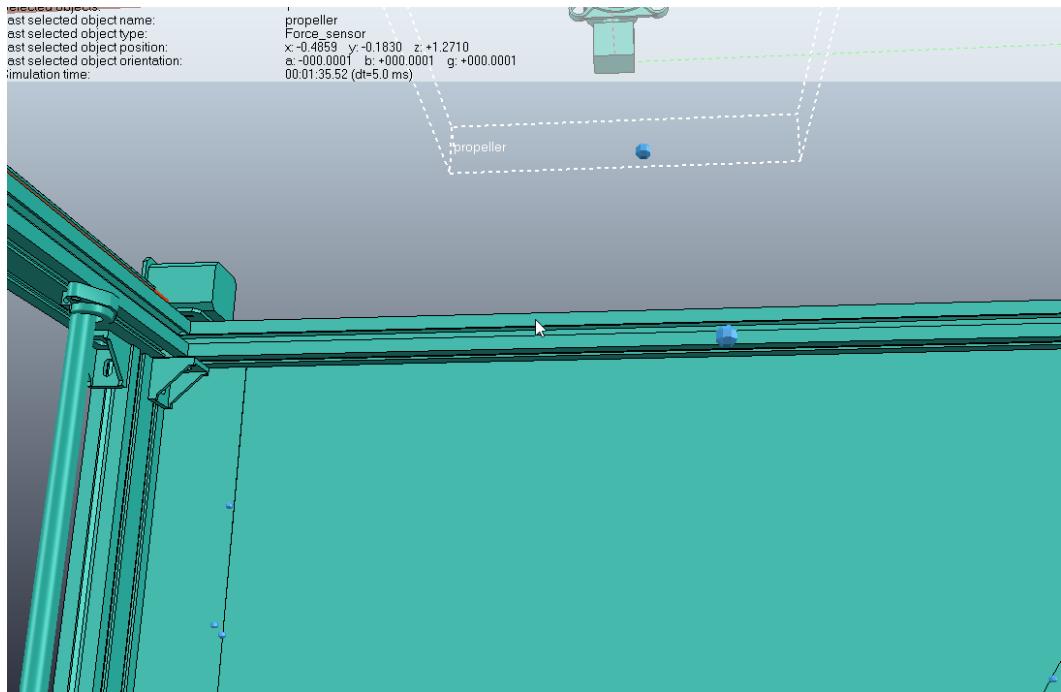


圖 4.16 粒子射出位置不固定

目前的問題是對粒子在射出後，對平台產生作用力而彈起不是黏在板上，及當平版移動時粒子會四處飄移，還要再對腳本裡的程式進行修改。

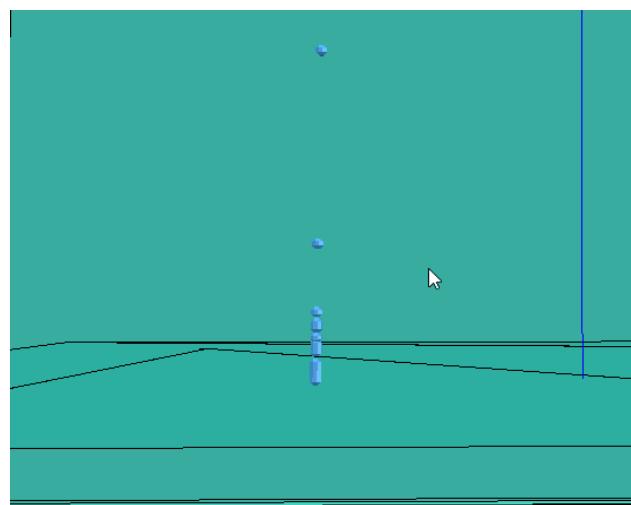


圖 4.17 粒子彈起

第五章 軟韌體設計

5.1 Marlin 韌體



圖 5.1 marlin

Marlin 為 3D printer 的韌體，由 Scott Laheine 為主要開發人員，在 2011 年為了 RepRap 和 Ultimaker 列印機所創造。在主機板上運作，管理機器運動時的活動，包括馬達驅動器、加熱器、傳感器、LCD 顯示器與按鈕的移動。支援大多數 3D printer 的硬體平台。

將 Arduino 下載安裝，選擇 Arduino Mega 2560 的開發板，以 USB 連接線與主機板連接，在 Marlin 中選擇 Configuration.h 做機器相關設定的修改

第六章 組立流程

6.1 機構組立

6.1.1 基本骨架安裝

先定位底部骨架，再接四支支架，切記連接部分要與底部垂直與平行

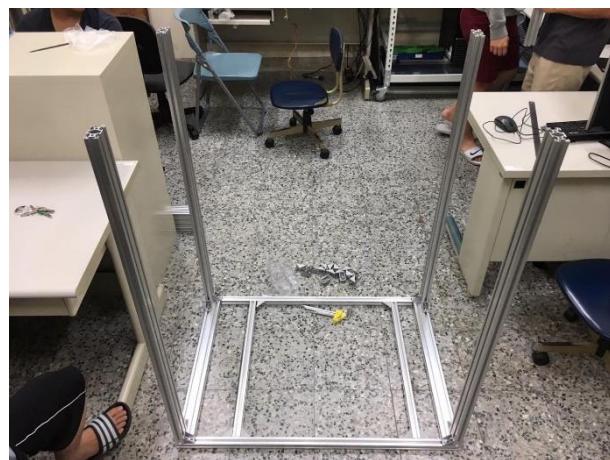


圖 6.1 基本骨架

安裝上方平台切記要與支架垂直與水平



圖 6.2 架構組裝完成

6.1.2 Z 軸組裝

先將 L 型鐵板畫完線後鑽孔，依指定尺寸去鑽，但不能鑽剛好的孔一定要比預定的孔尺寸再增加 2mm-3mm 左右，這樣才不會去卡到光軸及滾珠螺桿



圖 6.3 L 型鐵板

接著裝上光軸連接器，且要加上墊片否則會對不上孔，之後再裝光軸。



圖 6.4 裝上光軸連接器

裝上滾珠螺桿及鎖上



圖 6.5 安裝滾珠螺桿

裝上後，切記光軸及滾珠螺感的平行度不能差很多否則印表機再作動時馬達會轉不動。



圖 6.6 滾珠螺感及光軸互相平行

6.2 傳動系統組立

組裝 X、Y 皮帶、滑軌及微固定

安裝步進馬達 L 型座需定位與惰輪固定座，切記皮帶需與滑軌平行(Y 軸安裝)。



圖 6.7 Y 軸步進馬達及惰輪固定座

安裝 X 軸光軸與光軸固定座(單邊，光軸固定座需先鎖上滑軌)。

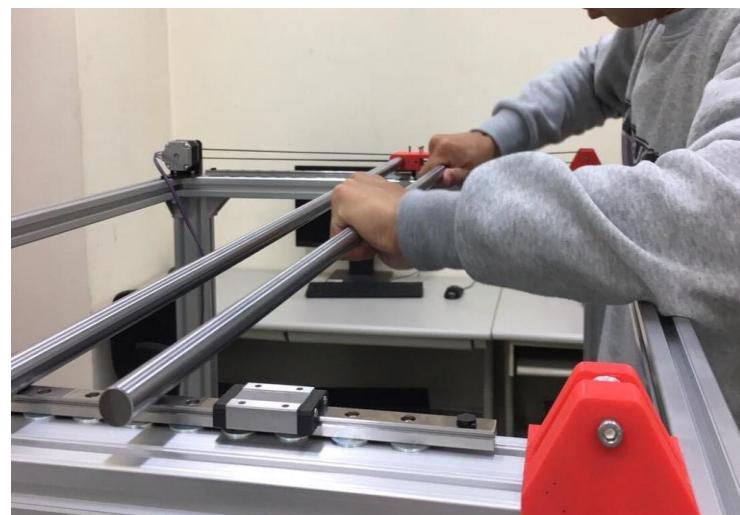


圖 6.8 安裝 X 軸光軸與光軸固定座

加裝噴頭固定座。



圖 6.9 噴頭固定座之安裝

安裝另外一邊的光軸固定座及皮帶(Y 軸)。



圖 6.10 安裝光軸固定座及 Y 軸皮帶

裝上 X 軸惰輪及皮帶與馬達。



圖 6.11 裝上 X 軸惰輪、皮帶及馬達

裝上噴頭與風扇

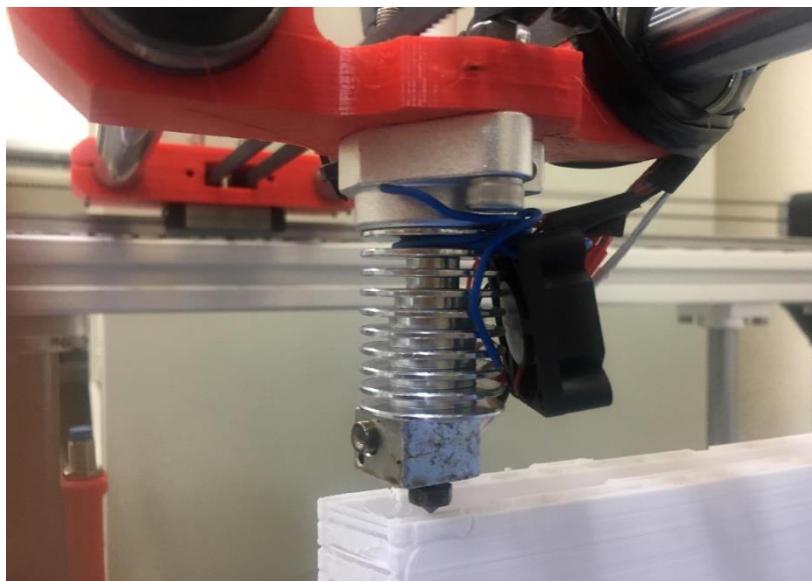


圖 6.12 噴頭與風扇

6.3 電控系統裝配

6.3.1 步進馬達驅動與測試

首先我們控制單一馬達的運轉，使用 Arduino UNO 控制板配合簡單的程式對馬達作正反轉測試，在轉動時會過度震動，是程式在編寫的過程有誤且電流過小使馬達轉動時震動。再來配合列印機之控制板找出相對應之腳位，連接後測試旋轉一圈的所需參數。

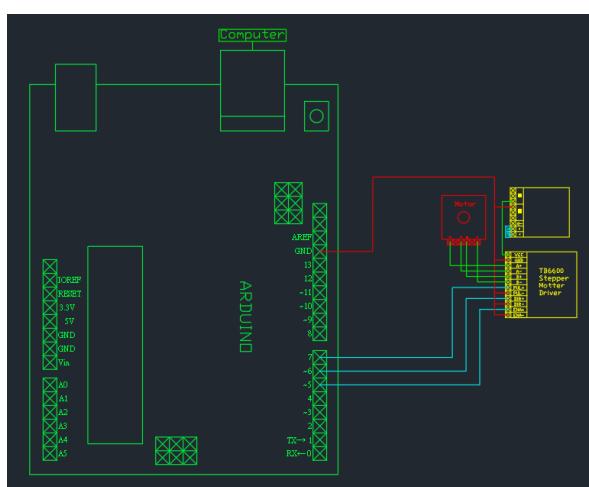


圖 6.13 馬達與 Arduino 腳位

```
1 int PUL=7; //define Pulse pin
2 int DIR=6; //define Direction pin
3 int ENA=5; //define Enable Pin
4 void setup() {
5   pinMode (PUL, OUTPUT);
6   pinMode (DIR, OUTPUT);
7   pinMode (ENA, OUTPUT);
8 }
9 }
10 void loop() {
11   for (int i=0; i<6400; i++)    //Forward 5000 steps
12   {
13     digitalWrite(DIR,LOW);
14     digitalWrite(ENA,HIGH);
15     digitalWrite(PUL,HIGH);
16     delayMicroseconds(50);
17     digitalWrite(PUL,LOW);
18     delayMicroseconds(50);
19   }
20 }
21 for (int i=0; i<6400; i++)    //Backward 5000 steps
22 {
23   digitalWrite(DIR,HIGH);
24   digitalWrite(ENA,HIGH);
25   digitalWrite(PUL,HIGH);
26   delayMicroseconds(50);
27   digitalWrite(PUL,LOW);
28   delayMicroseconds(50);
29 }
30 }
```

圖 6.14 測試馬達之

要讓步進馬達旋轉一圈，需先知道步級角為 1.8° ，轉一圈為 360° ，將兩者相除可得馬達轉一圈需 200 步，此時驅動器的微步調整會影響參數，我們將驅動器選擇為 32 微步，試算出來的數值輸入程式中，測試後即可讓馬達選轉一圈。

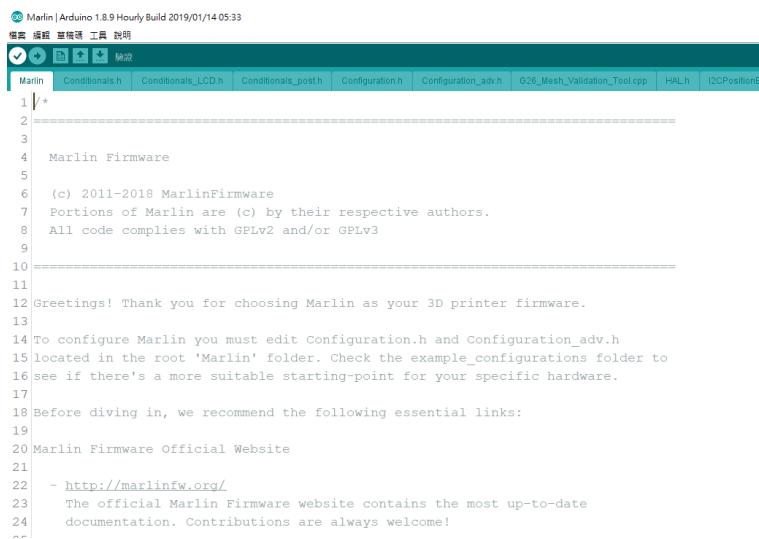
步進馬達之步級角=1.8°、馬達轉一圈=360°

$$\longrightarrow \frac{360^\circ}{1.8^\circ} = 200\text{步(轉一圈所需之步數)}$$

驅動器設為 32 微步

$$\longrightarrow 200\text{步} \times 32\text{微步}=6400\text{脈波/圈(馬達轉 } 90^\circ)$$

將 Arduino 換成 3D 列印機的控制板，要能控制馬達先找到馬達的腳位。從 Marlin 中找到與控制板型號的腳位 Pin，開啟 Marlin 程式，在右邊的下拉式選單中找與控制板相符的檔案。



The screenshot shows the Arduino IDE interface with the Marlin firmware code open. The title bar reads "Marlin | Arduino 1.8.9 Hourly Build 2019/01/14 05:33". The menu bar includes "檔案" (File), "編輯" (Edit), "工具" (Tools), and "說明" (Help). The toolbar has icons for "undo", "redo", "refresh", and "search". The code editor tab bar shows files: Main, Conditionals.h, Conditionals_LCD.h, Conditionals_posit.h, Configuration.h, Configuration_adv.h, G26_Mesh_Validation_Tool.cpp, HAL.h, and I2CPositionE. The main code area displays the Marlin Firmware startup message:

```
1 /*
2 -----
3
4 Marlin Firmware
5
6 (c) 2011-2018 MarlinFirmware
7 Portions of Marlin are (c) by their respective authors.
8 All code complies with GPLv2 and/or GPLv3
9
10 -----
11
12 Greetings! Thank you for choosing Marlin as your 3D printer firmware.
13
14 To configure Marlin you must edit Configuration.h and Configuration_adv.h
15 located in the root 'Marlin' folder. Check the example_configurations folder to
16 see if there's a more suitable starting-point for your specific hardware.
17
18 Before diving in, we recommend the following essential links:
19
20 Marlin Firmware Official Website
21
22 - http://marlinfw.org/
23 The official Marlin Firmware website contains the most up-to-date
24 documentation. Contributions are always welcome!
^r
```

圖 6.15 Marlin 程式

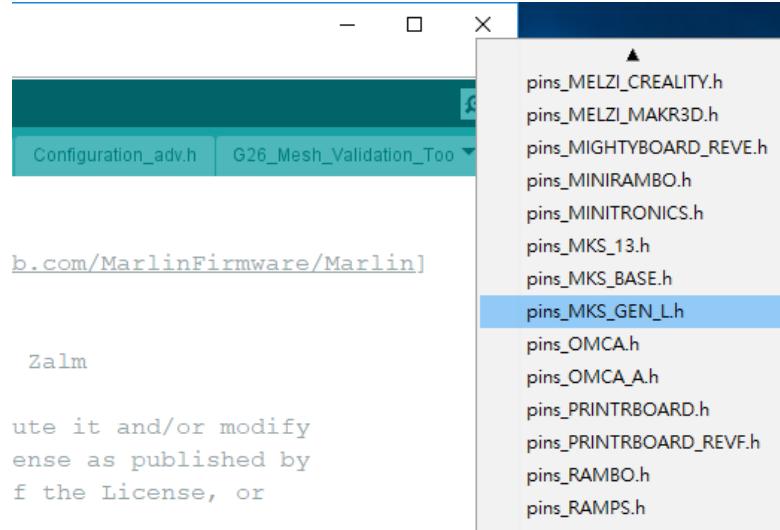


圖 6.16 右側分頁

點選後內容會告訴你該使用哪個程式，在分頁中找到該檔案所有馬達的腳位編號。

```

Marlin - pins_MKS_GEN_L.h | Arduino 1.8.9 Hourly Build 2019/01/14 05:33
檔案 儲存 編輯 單純碼 工具 說明
pins_MKS_GEN_L.h Conditional.h Conditional_LCD.h Conditional_post.h Configuration.h
21 /*
22
23 /**
24 * MKS GEN L - Arduino Mega2560 with RAMPS v1.4 pin
25 */
26
27 #if HOTENDS > 2 || E_STEPPERS > 2
28   #error "MKS GEN L supports up to 2 hotends / E-ste
29 #endif
30
31 #define BOARD_NAME "MKS GEN L"
32
33 /**
34 // Heaters / Fans
35 /**
36 // Power outputs EFBF or EFBE
37 #define MOSFET_D_PIN 7
38
39 #include "pins_RAMPS.h"

```

圖 6.17 控制板選用

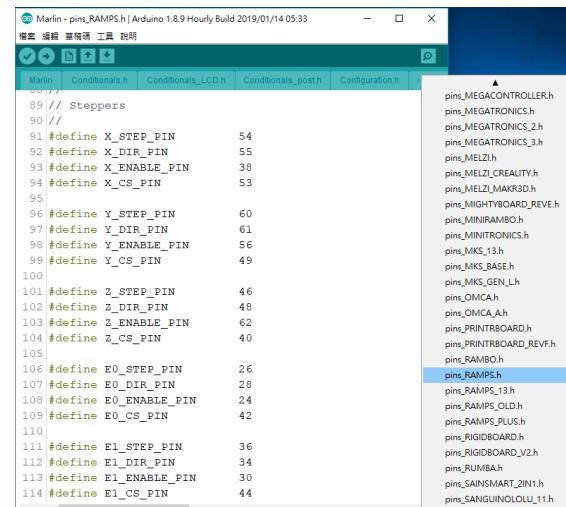


圖 6.18 腳位編號

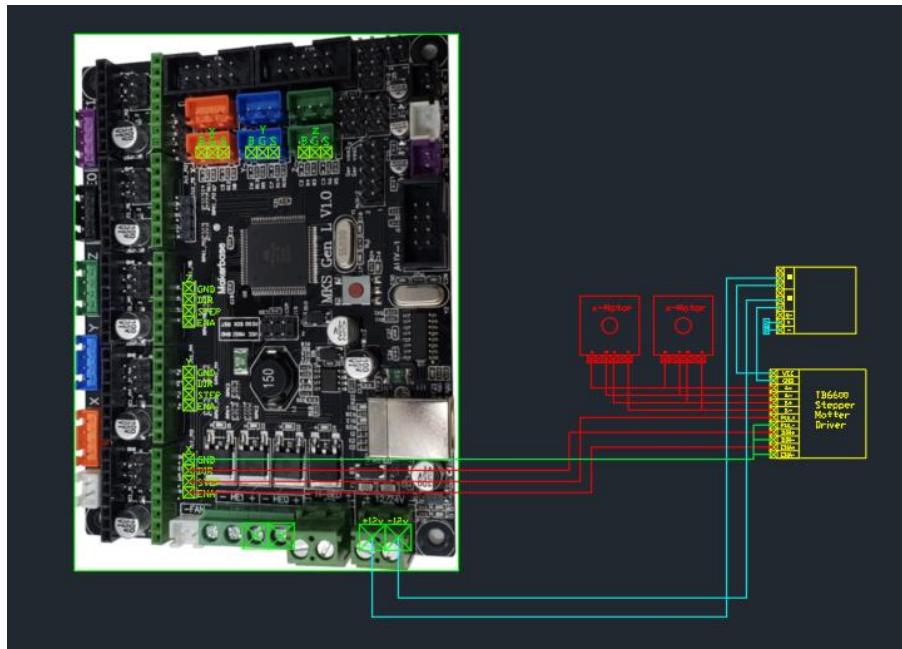


圖 6.19 控制板與馬達

6.3.2 測試 X 軸移動距離

我們將控制板連接上一台 3D 列印機上，在 Marlin 程式中點選 Configuration.h 分頁，可使用搜尋功能找 Movement setting，並修改程式。

```

609 *                                     X, Y, Z, E0 [, E1[, E2[, E3[, E4]]]]
610 */
611 #define DEFAULT_AXIS_STEPS_PER_UNIT    { 160, 160, 1920, 93.80 }
612
613 /**
614 * Default Max Feed Rate (mm/s)
615 * Override with M203
616 *                                     X, Y, Z, E0 [, E1[, E2[, E3[, E4]]]]
617 */
618 #define DEFAULT_MAX_FEEDRATE           { 30, 30, 5, 25 }
619
620 /**
621 * Default Max Acceleration (change/s) change = mm/s
622 * (Maximum start speed for accelerated moves)
623 * Override with M201
624 *                                     X, Y, Z, E0 [, E1[, E2[, E3[, E4]]]]
625 */
626 #define DEFAULT_MAX_ACCELERATION       { 3000, 3000, 3000, 2000 }

```

圖 6.20 馬達移動距離程式

再使用 3D 列印模擬軟體 Pronterface，測試移動距離，一開始可移動 1mm 為基準再慢慢增加一動 10mm。

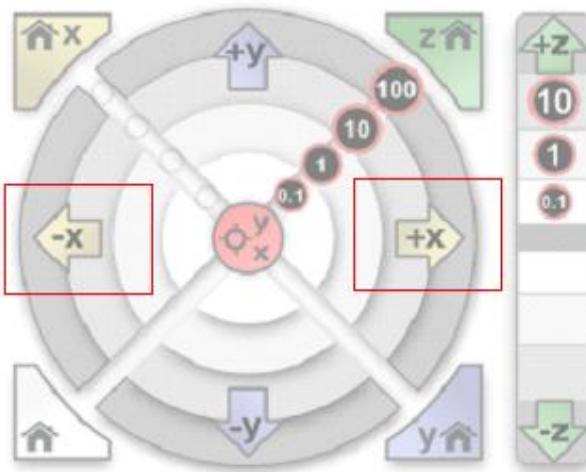


圖 6.21 Pronterface X 軸移動

6.3.3 馬達正反轉修改

測試時須以馬達轉動之方向去定義正負，要先設好原點位置

```
735 // Disables axis stepper immediately when it's not being used.  
736 // WARNING: When motors turn off there is a chance of losing position accuracy!  
737 #define DISABLE_X false  
738 #define DISABLE_Y false  
739 #define DISABLE_Z false  
740 // Warn on display about possibly reduced accuracy  
741 // #define DISABLE_REDUCED_ACCURACY_WARNING  
742
```

圖 6.22 馬達正反轉程式

6.3.4 X、Y 極限開關

安裝極限開關時，控制板上有三個腳位，只需要接上兩個即可。控制板上分別有 X、Y、Z 軸的位置，上面的 "S" 腳位代表訊號源，"-" 腳位代表接地線， "+" 腳位代表電源；極限開關上有三個腳位， "C" 為共同接腳，NO 為 Normal Open 是常開接腳，NC 為 Normal Close 是常閉接腳，常閉接腳為按壓後 C 接腳斷開不通電，所以可依據接線方式來決定線路為通電或斷電。

我們在 X 軸與 Y 軸使用的為 NC 接腳，並因 Marlin 程式的定義將以下程式改成 " false "，並在 Pronterface 中使用 " M119 " 的指令來查看極限開關的狀態，若出現 " triggerd" 代表極限開關被壓著，若出現 " open" 則表示極限開關沒有被壓著。

極限開關 C 接到控制板上 " - " 接點

極限開關 NC 接到控制板上 " S " 接點

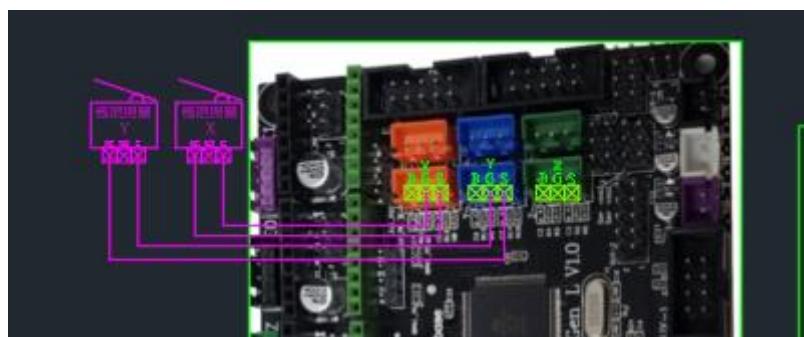


圖 6.23 極限開關配線圖

X_MIN_ENDSTOP_INVERTING 改成 false

Y_MIN_ENDSTOP_INVERTING 改成 false

```
530 // Mechanical endstop with COM to ground and NC to Signal uses "false" here (most common setup).  
531 #define X_MIN_ENDSTOP_INVERTING false // set to true to invert the logic of the endstop.  
532 #define Y_MIN_ENDSTOP_INVERTING false // set to true to invert the logic of the endstop.  
533 #define Z_MIN_ENDSTOP_INVERTING true // set to true to invert the logic of the endstop.  
534 // #define X_MAX_ENDSTOP_INVERTING false // set to true to invert the logic of the endstop.  
535 // #define Y_MAX_ENDSTOP_INVERTING false // set to true to invert the logic of the endstop.  
536 // #define Z_MAX_ENDSTOP_INVERTING false // set to true to invert the logic of the endstop.  
537 // #define Z_MIN_PROBE_ENDSTOP_INVERTING false // set to true to invert the logic of the probe.  
538
```

6.3.5 Z 軸接近開關

接近開關的電源為 12V，不能直接與控制板連接，因為控制板只能提供 5V 電源，需要降壓再接到控制板上，但是訊號電流至少要 4.5V 以上不然無法觸發訊號。

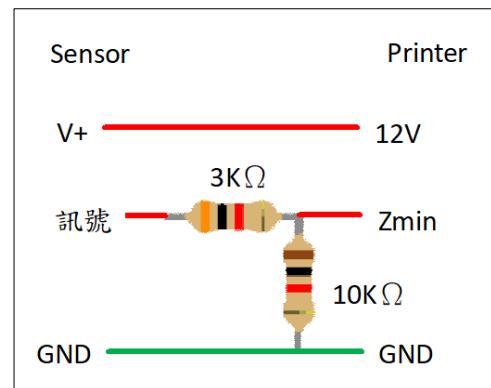


圖 6.24 降壓電路圖

接近開關 C 接到 RAMPS X end stop -

接近開關 NO 接到 RAMPS X end stop S

Z_MIN_ENDSTOP_INVERTING 改成 true

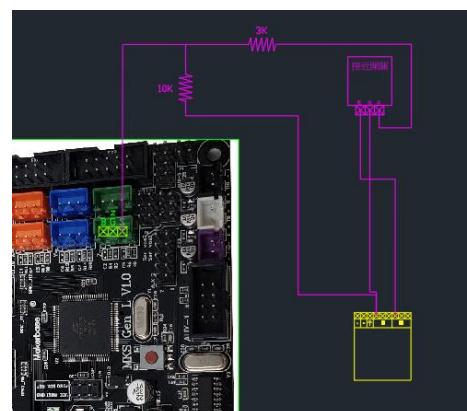


圖 6.25 接近開關配線圖

6.3.6 LCD 螢幕驅動

下載 u8glib_arduino_v1.18.1 加入 Marlin 後，將語言改成需要的，每個語言前有代表之代號，將他輸入即可，另外螢幕的線路與控制板連接時，有防呆的設計可直接連上，若螢幕沒反應除了是程式問題也可以檢查線路連接是否有誤。

```
//=====
//===== LCD and SD support =====
//=====

// @section lcd

/**
 * LCD LANGUAGE
 *
 * Select the language to display on the LCD. These languages are available:
 *
 * en, an, bg, ca, cn, cz, cz_utf8, de, el, el-gr, es, eu, fi, fr, fr_utf8, gl,
 * hr, it, kana, kana_utf8, nl, pl, pt, pt_utf8, pt-br, pt-br_utf8, ru, sk_utf8,
 * tr, uk, zh_CN, zh_TW, test
 *
 * :{ 'en':'English', 'an':'Aragonese', 'bg':'Bulgarian', 'ca':'Catalan', 'cn':'Chinese',
 *
#define LCD_LANGUAGE cn
```

圖 6.26 LCD 螢幕程式

6.3.7 加熱頭

調整 PID 值，有三個參數分別為 KP、KI、KD，marlin 程式內有預設數值，但是需要根據機台做校正。使用 Pronterface 右下角指令區輸入” M303 C8 200”，S=目標溫度、C=震盪次數，震盪越多次，算出來的值越準確。特別注意的是加熱頭在接接線時，若要延伸需使用多芯線才不會燒掉。

```

382 // Ultimaker
383 #define DEFAULT_Kp 25.92
384 #define DEFAULT_Ki 2.24
385 #define DEFAULT_Kd 74.86
...

```

圖 6.27 PID 值設定

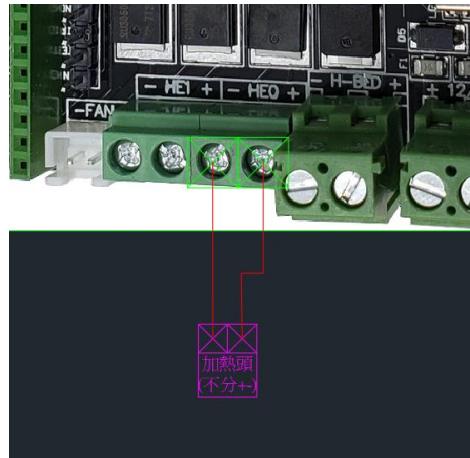


圖 6.28 加熱頭配線圖

6.3.8 風扇

風扇有正、負極兩個腳位，正極與正極相接、負極與負極相接，並在 Pronterface 右下角指令區輸入指令”M106 P0 S35”測試是能運轉，也可把風扇直接接上電源，一樣的接線方式，在 3D 列印機開機時就會運作，S=速度。熱敏電阻與風扇一樣，正極相接、負極相接。

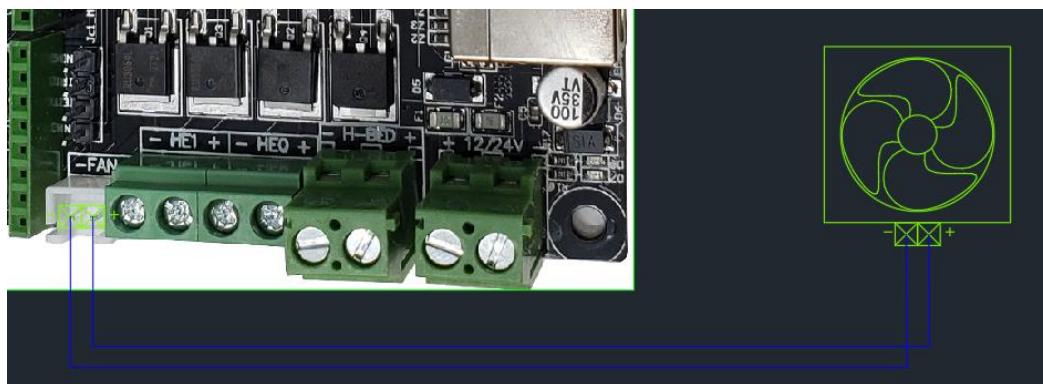


圖 6.29 風扇配線圖

6.3.8 热敏電阻

热敏電阻非常脆弱，安装時要小心不要拉扯或折損。

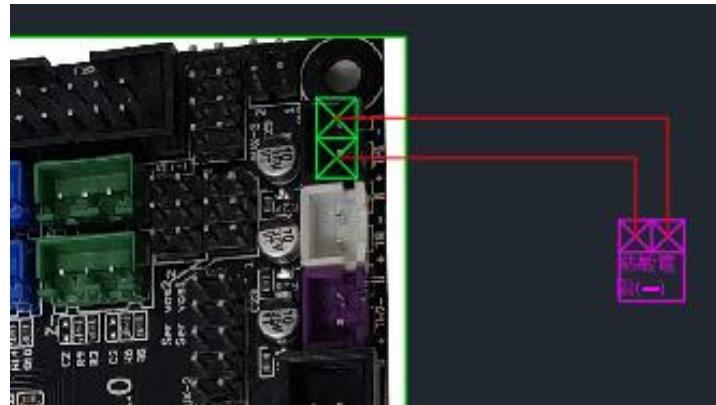


圖 6.30 热敏電阻配線圖

6.3.9 挤出機設定

須裝一個馬達驅動器 A4988 在控制板上來驅動擠出機馬達，測試
擠出量，輸入指令”M302 S0”，擠出量設 100mm 接著按下 Extrude，
擠出機開始運作。

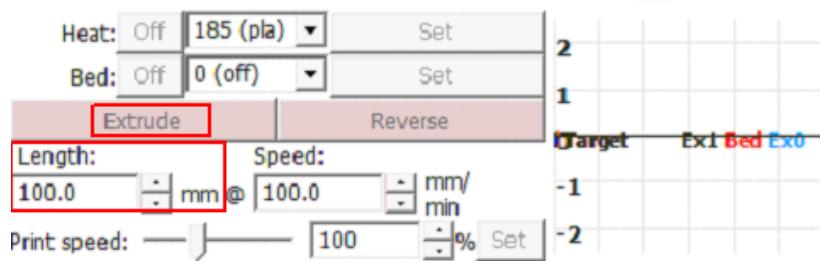


圖 31 挤出機測試

完成後將材料剪下並量測，若擠出長度為 533 而 Marlin 原本的值是 500 則 $500/533=0.9380$ ，將這個值乘上 100 為 93.80 即為輸入值

```
528 * Default Axis Steps Per Unit (steps/mm)
529 * Override with M92
530 *
531 */
532 #define DEFAULT_AXIS_STEPS_PER_UNIT { 160, 160, 1920, 93.80 }
533
534 /**
```

圖 6.32 擠出機設定

6.4 問題與解決方法

問題：一開始裝上步進馬達後無法控制轉幾度幾度。

解一：要輸入正一開始程式中就是會不停的循環與做動。

解二：如果控制精度(1mm)，則是 marlin 程式修改需要改成對的數值，然後運用 prонterface 的應用程式去操控，記得主機板要連上電腦。

問題：移動為何只能走(+)方向不能往(-)方向。

解：因為要裝極限開關且回歸原點後才能往(+、-)方向移動。裝極限開關是為了定義機械原點在哪個位置，所以電腦不知道原點在何處。

問題：X、Y 無法回正確的回歸原點。

解：因為一開始我們的 Y 軸兩顆馬達的方向是相反的，但是兩邊都改成同方向就能解決這個問題了。

問題：接近開關無法準確觸發。

解：當時測試直接把接近開關接到板子中彈無法讓機器回原點時觸發，所以我們把接近開關接到 12V，本來主機板

供電只有供應 5V 電壓所以無法觸發，但接到 12V 之後就能觸發，記得要連接主機板時要家電阻降壓以免主機板燒壞，然後降壓不能降到 4.5V 以下，否則也無法觸發。

問題：LCD 屏幕無法顯示。

解：當時購買的 LCD 螢幕後的接頭，是本來可以直接上，插上後卻無法顯示，將接頭相反後則就可以正常顯示了。

問題：加熱溫度不夠。

解：因為當時在列印時溫度只設定 175 度，但是要讓材料確實融化且能黏上底板需要 190 度才能確實融化材料。

問題：擠出機送料不穩。

解：因為那時候擠出機沒有正確的接上擠出機而造成送料不穩。

問題：marlin 無法上傳主機板。

解：連接電腦與連接主機板的插頭重新插緊即可，如果還是無法上傳的話則是更換連接線。

問題：加熱頭無法正確擠出材料。

解：因為加熱頭內部卡料，造成加熱頭無法正確擠出材料，

需要拿工具把加熱頭內部材料弄出才能正確擠料。

問題：馬達轉動忽快忽慢。

解：因為板子所提供的電流只有 5V 所以需要 12V 的電源供應

器幫忙提供電量才行。

問題：Z 軸無法同步做動。

解：要檢查 Z 軸馬達接線是否有接好，或者有些線路斷掉。

問題：溫度無法確實感應。

解：要查看熱敏電阻線路是否有折損或拉到，熱敏電阻非常

脆弱要小心。如果沒有感應到，溫度會顯示 -14 度，如果

有感應到則會顯示常溫 20-30 度左右。

第七章 軟體選用與配置

7.1 Cura

Cura 為 Ultimaker 公司設計的 3D 列印軟體，免費的
軟件且功能強大輕鬆上手。可將模型切成一層一層並將
指令參數轉為 G-code 讓列印機進行列印。



7.1.1 基本設定

圖 7.1 Cura

安裝完 Cura 4.0 之後，即可開始設定印自己所使用的列印機，
Cura 4.0 內部已經有非常多種列印機，所以可以直選，例如，
我們是自己組裝的列印機，所以要按 Cura 左上角中，有一個設
定，裡面有印表機，印表機點下去之後就可以新增一個印表機，
我們是使用 Prusa i3 的機台所以選用 Prusa i3，選完列印機後點
選自己的印表機(Prusa i3)，之後要設定列印機的基本設定。

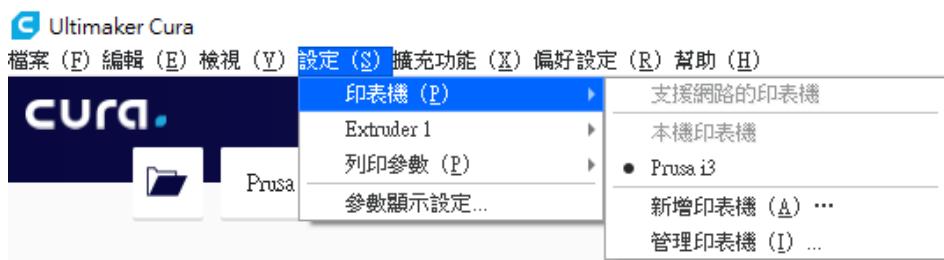


圖 7.2 新增列印機

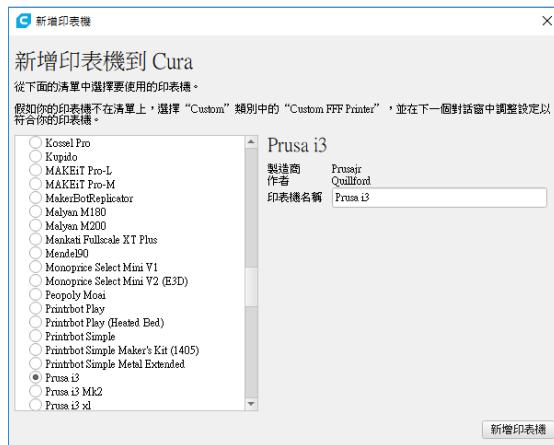


圖 7.3 選用適合的列印機

在設定中的參數顯示設定，設定列印機的最大行程

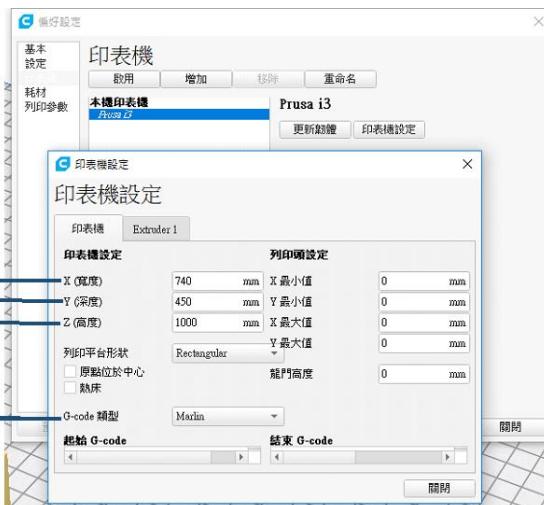


圖 7.4 基本設定

表 7.1 機台設定說明

代碼	名稱	說明
1	X(寬度)	X(寬度)是指，機台本身 X 軸的最大行程
2	Y(深度)	Y(深度)是指，機台本身 Y 軸的最大行程
3	Z(高度)	Z(高度)是指，機台本身 Z 軸的最大行程
4	G-cord 類型	指主機板內部是使用哪一種韌體

7.1.2 列印參數設定

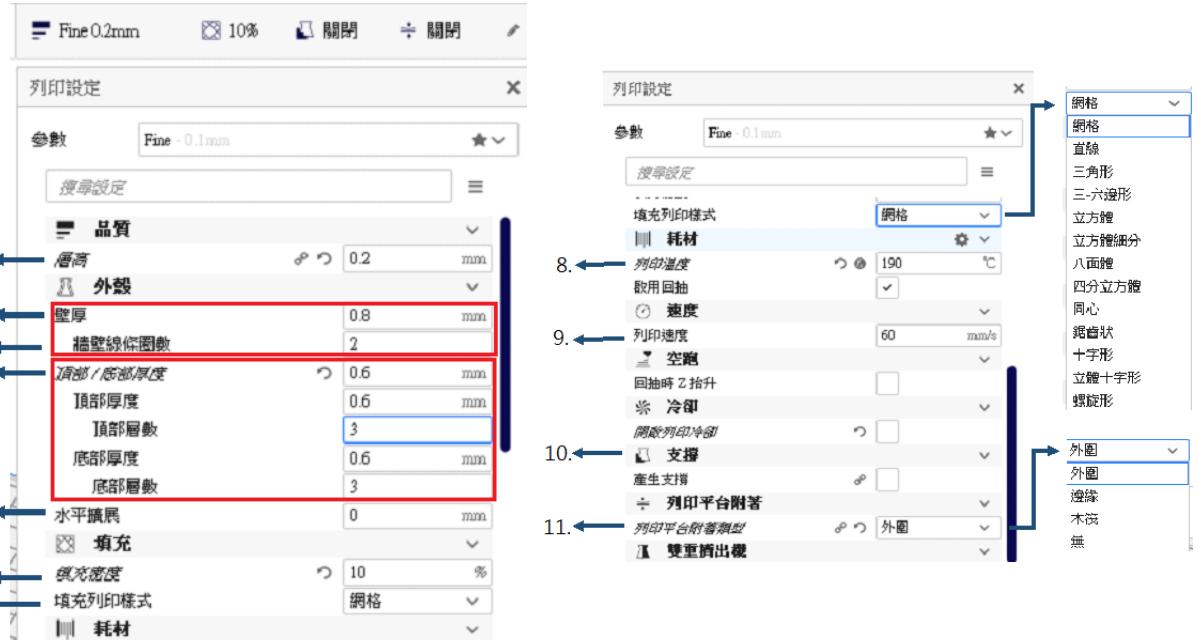


圖 7.5 列印參數設定

圖 7.6 列印參數設定

表 7.2 列印參數說明

代碼	名稱	說明
1	層高	列印時，每層高度為 0.2mm
2	壁厚	牆壁厚度，簡稱外圍厚度，與牆壁線條圈數是相對應的，所以如果機台噴頭設定是 0.4mm 的，然後壁厚設定為 0.8mm，牆壁線條圈數 Cura 會自動算出。
3	牆壁線條圈數	

4	頂部/底部厚度	頂部/底部厚度跟層高有關，所以若層高設定為 0.2mm，然後頂部/底部厚度 Cura 會自動幫你算需要列印幾層
5	水平擴展	有關於模型中有孔那些需要作微調的部分
6	填充密度	填充密度就是有關模型強度，如果值越大，模型強度就會越大
7	填充列印樣式	填充列印樣式，基本上就是你可以選取模型密度中想要的填充樣式
8	列印溫度	列印溫度是在列印時，噴頭之溫度
9	列印速度	列印速度就是在列印時 X、Y 移動的速度
10	支撐	支撐簡單來說就是假設如過有像 V 型的模型，在列印時需要有東西讓他支撐，列印出來的東西才會完美
11	列印平台附著	列印平台附著就是，如果你的模型是屬於薄狀且很高的狀態下，你需要用列印平台附著去增加列印時，模型與平台的附著狀態

7.2 Pronterface

Pronterface 為 3D 列印機控制軟體，有基本的

控制功能，幾乎適用於所有 3D 列印機，可做

輸出控制與機器校正，也可直接下達指令給 3d

列印機在接收回傳的資料。



圖 7.7 Pronterface

下載安裝後先在左上角點擊設定，修改機器的工作範圍及擠出機數量，

擠出機通常為 1，Settings>option

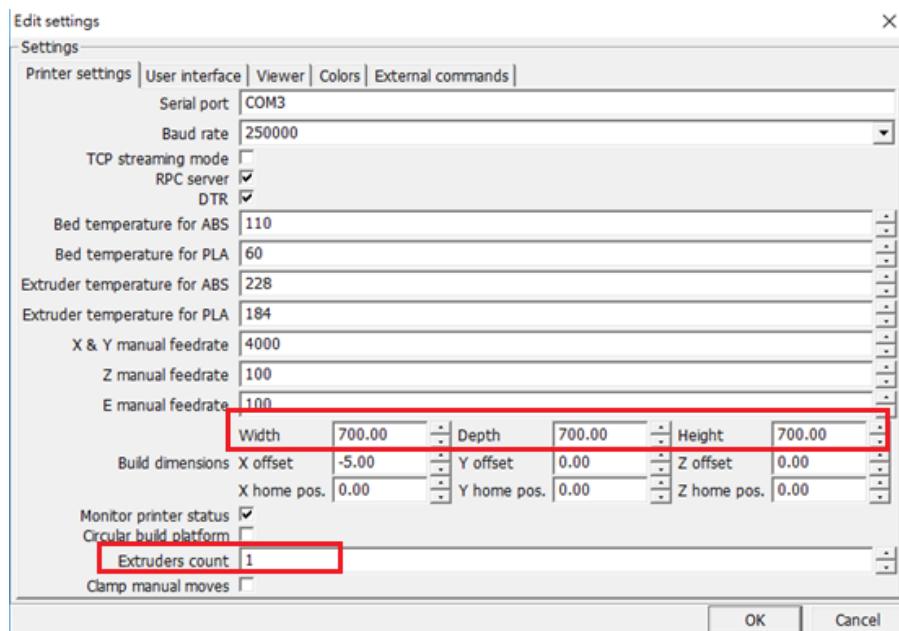


圖 7.8 機台設定

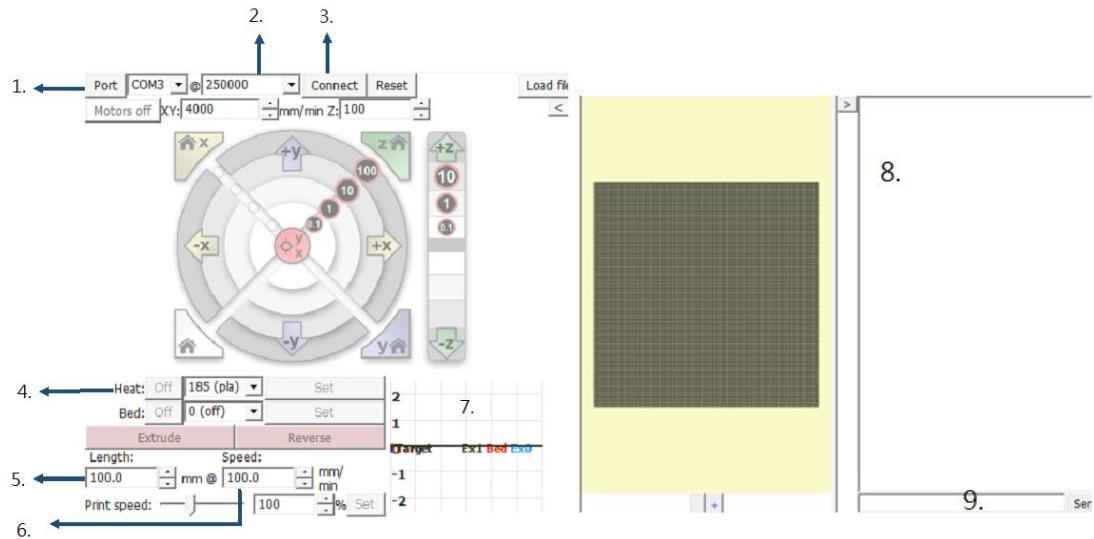


圖 7.9 Pronterface 介面

表 7.3 Pronterface 介面說明

代碼	名稱	說明
1	Port	連接埠
2	Baud Rate	傳送速率，指資料與電控板傳輸之間的速度
3	Connect	按下連接後與 3D 列印機連接
4	Heat	列印機噴嘴溫度
5	Length	擠出機輸出長度
6	Speed	列印機列印速度
7	溫度圖表	當噴嘴在加熱時，可以查看溫度的上升
8	指令區	可在此處輸入指令，測試機台
9	顯示區	輸入指令的回饋會陳列在此處

7.3 Python

Python 是物件導向程式的較易入門語言，也是直譯式程式語言。

Python 強調對程式語言的語句易讀、易懂、易學、簡潔和清晰的語法特點及加快程式開發的時效，方便使用，可以完成各種難度的應用，並可在大多數的系統中運行，以減少開發及維護成本的觀念進行發展。



圖 7.10 Python

Python 同時支援 modules 和 packages，另外 Python 為跨平台程式語言也支援 unicode 字元。功能強大而完善的通用型語言，可以用於很多種軟體開發動態程式，使得 Python 非常有吸引力，發展至今已有十多年的歷史，成熟且穩定，並提供了許多自行開發的 library(函式庫)以提供其他 Python 程式設計者下載使用。

參考 <https://sls.weco.net/CollectiveNote20/Python/Intro>

7.4 Arduino

Arduino 電路板設計使用各種微處理器和控制器。這些電路板配有一組數字和類比 I/O 引腳，可以連接各種擴充板或麵包板（封鎖板）和其他電路。這些電路板具有串列埠，包括某些型號上的通用串列匯流 (USB)，也用於從個人電腦載入程式。微控制器通常使用 C/C++ 程式語言。



圖 7.11 Arduino

The screenshot shows the Arduino IDE interface. The title bar says "123 | Arduino 1.8.9 Hourly Build 2019/01/14 05:33". The menu bar includes "檔案" (File), "編輯" (Edit), "草稿" (Sketch), "工具" (Tools), and "說明" (Help). The code editor contains the following C/C++ code:

```
1 int PUL=7; //define Pulse pin
2 int DIR=6; //define Direction pin
3 int ENA=5; //define Enable Pin
4 void setup() {
5   pinMode (PUL, OUTPUT);
6   pinMode (DIR, OUTPUT);
7   pinMode (ENA, OUTPUT);
8 }
9
10 void loop() {
11   for (int i=0; i<6400; i++)    //Forward 5000 steps
12   {
13     digitalWrite(DIR,LOW);
14     digitalWrite(ENA,HIGH);
15     digitalWrite(PUL,HIGH);
16     delayMicroseconds(50);
17     digitalWrite(PUL,LOW);
18     delayMicroseconds(50);
19 }
```

代號	說明
1	編寫完程式 編譯並上傳
2	程式編寫區
3	訊息回饋

表 7.4 Arduino 程式界面說明

圖 7.12 Aeduino 程式介面

7.5 V-rep

為 V-rep & PyQt & Python 之軟體結合，以虛擬的方式模擬測試結果，主要以 Python 撰寫程式，並且透過 V-rep API 與物件進行協定，達到動態模擬的效果。



圖 7.13 v-rep

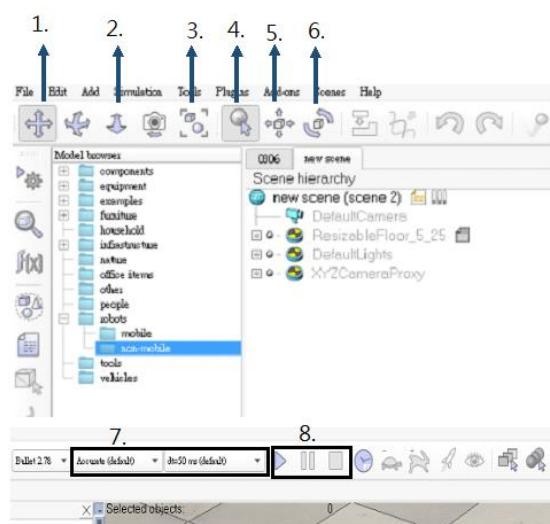


圖 7.13 v-rep 功能

表 7.5 v-rep 功能說明

代號	說明	代號	說明
1	移動場景	5	物件平移
2	旋轉場景	6	物件旋轉
3	正視物件	7	物理引擎配置
4	選擇點擊	8	模擬(開始、暫停、結束)

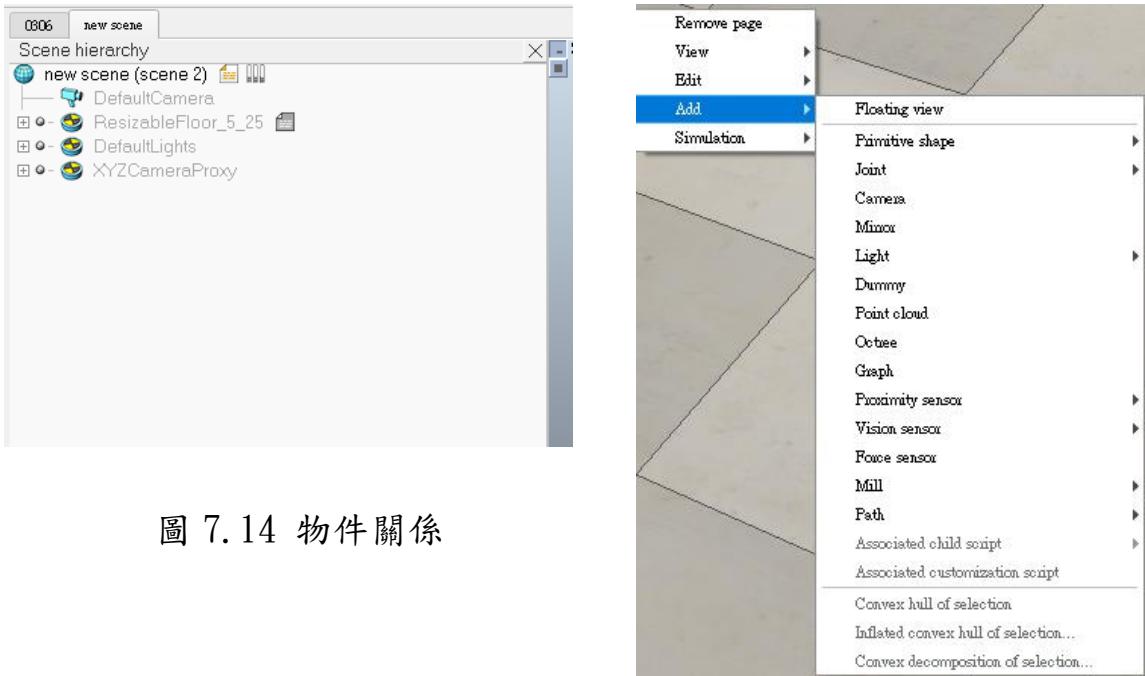


圖 7.14 物件關係

圖 7.15 增加其他附件

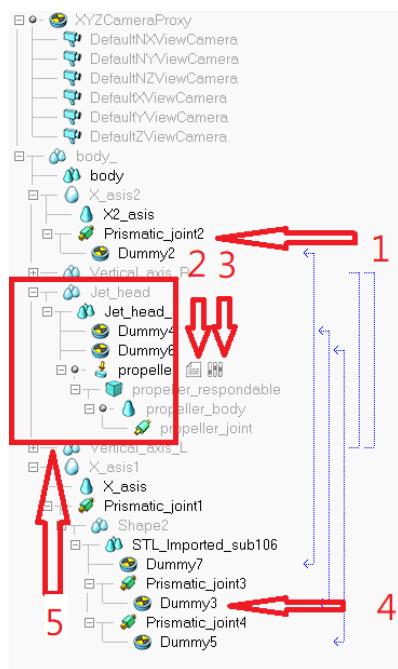


圖 7.16 物件從屬關係

表 7.6

代號	說明
1	軸
2	腳本圖標
3	腳本模擬參數圖標
4	虛擬連結
5	場景層次結構

7.6 Onshape

為繪圖軟體，介面與 Solidworks 雷同但為免費提供使用，且設計者能在線上與他人協同共享作品並有歷程記錄和版本。



圖 7.17 Onshape

開啟新專案：建立 > 文件



圖 7.18 建立新文件

選擇一基準面，點選草圖進行繪製

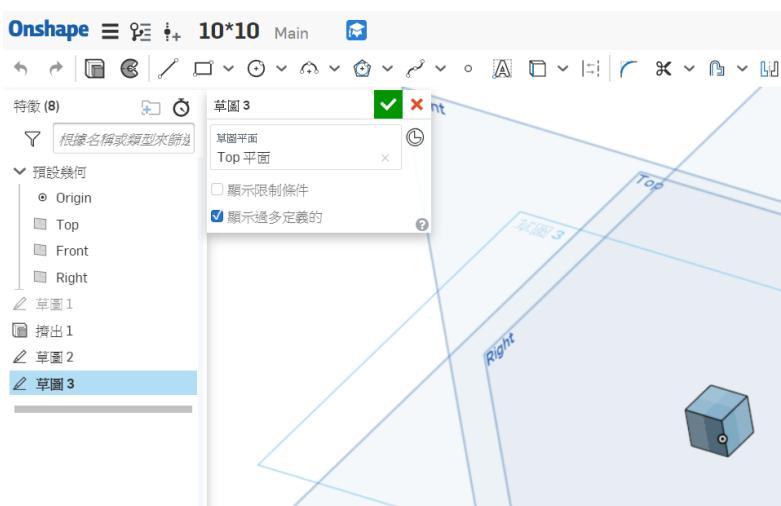


圖 7.19 繪製零件

表 7.7 常用繪圖功能

圖示	繪圖功能	圖示	特徵功能
	直線		伸長填料、廚廖
	圓形		旋轉
	圓弧		導圓角
	矩形		導角
	標註尺寸		肋材
	圓角		掃出填料、除料
	修剪、延伸		拔模
	鏡射		疊層拉伸
	線性複製排列、 環狀複製排列		螺旋線
	偏移、溝槽		薄殼
	重合共點		
	同心圓弧		
	相切		

在下方列表右鍵點選匯出，格式選擇 STL 後確定即可存為 STL 檔

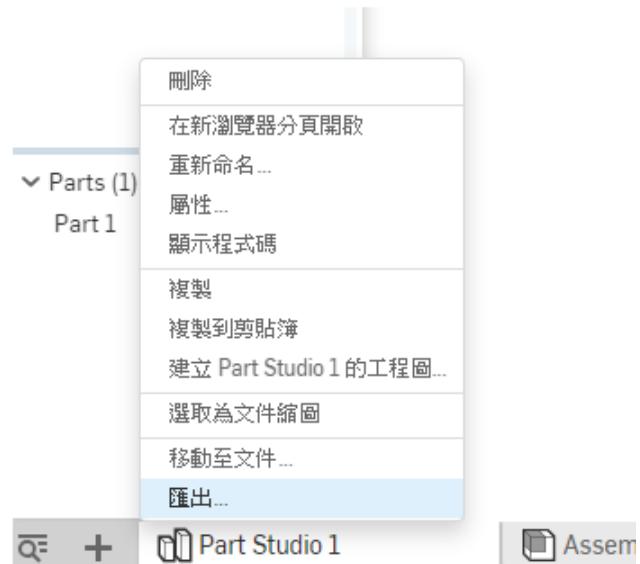


圖 7.20 匯出檔案

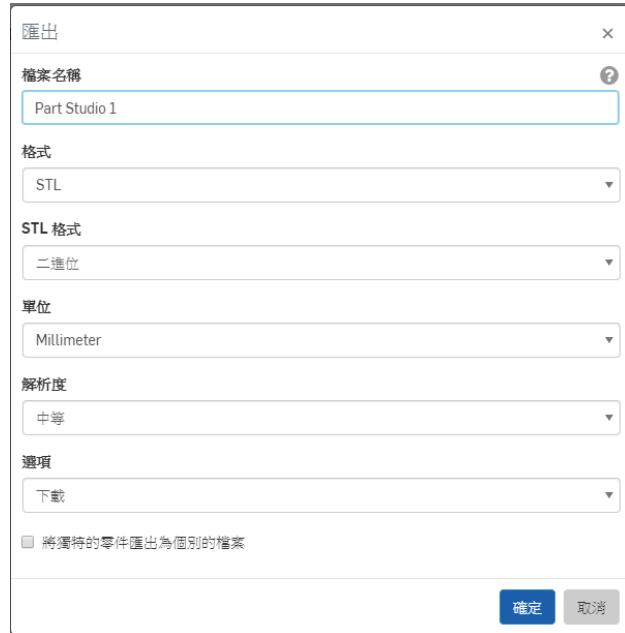


圖 7.21 儲存成. STL 檔

第八章 系統定位與校正

8.1 校正 X、Y 軸垂直水平精度

1. 首先鎖緊光軸固定座之螺絲，並且放鬆兩條固定滑軌的絲。
2. 將光軸固定座試滑，試滑完後，把光軸固定座移動到中間偏左或右邊，鎖緊靠近光軸固定座的螺絲(1 條滑軌)。
3. 延續光軸移動左邊或右邊鎖緊螺絲(光軸固定座平台須保持移動，滑軌固定需從中間到兩側螺絲連續鎖緊)。
4. 一條滑軌鎖緊後，將光軸固定座移動到滑軌中間試滑會不會有干涉或卡住之情形。
5. 另一條滑軌延續步驟(2)、(3)，之後滑動檢測看看會不會有干涉或卡住之情形。
6. X 軸鎖緊前與鎖緊後須試滑噴頭固定座，看會不會有干涉之情形。

8.2 Z 軸平台校正

1. 接下來組裝完光軸及滾珠螺桿後需要先量好滾珠螺桿底座大約尺寸，量好之後將中心部位畫線找出中心點，上下座都要，接著把所有剛才組裝的光軸及螺桿放上剛剛的滾珠螺桿座，切記滾珠螺桿座一定要先定位完成才能做光軸定位。



圖 8.1 量測下方滾珠螺桿底座尺寸



圖 8.2 量測上方滾珠螺桿底座尺寸

2. 滾珠螺桿固定完成之後，把 L 型鐵板轉到最下面，轉到下面之後定位兩邊光軸固定座(上方光軸固定座先不要固定)。

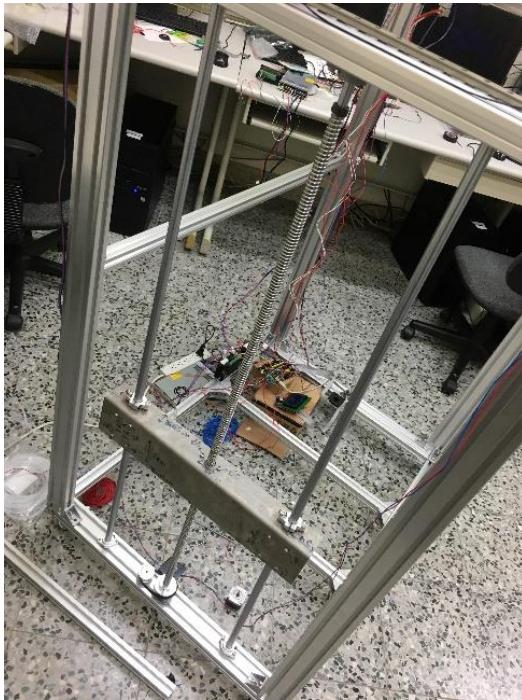


圖 8.3 L 型鐵板移至下方

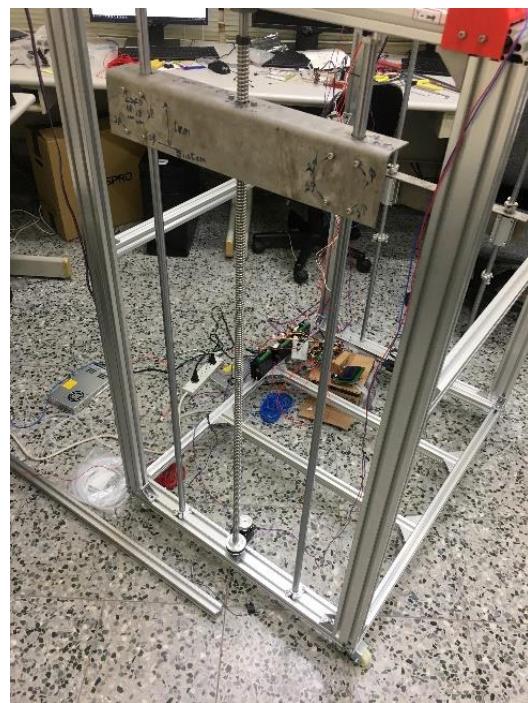


圖 8.4 定位光軸固定座

3. 接著把 L 型鐵板移動到最上面，再來固定上方的光軸固定座。

固定完成後，轉動看看是否會有卡住的問題。如有卡住之問題，再把 L 型鐵板上的螺絲鬆開去做微調處裡即可。也可以裝上馬達去控制微調(較省力)。

4. 放上平台鋁擠製與壓克力板還有木條，然後用水平儀去校正水平，記得外骨架一定要先用水平儀去校正，校正完才能去校正平台，之後把噴頭裝上掛表(精度 0.01mm)，去做精度最佳化(平台誤差正負 20mm)，四邊都要調平，否則印出來的東西會不太好，主要還是以噴頭去做精度校正。

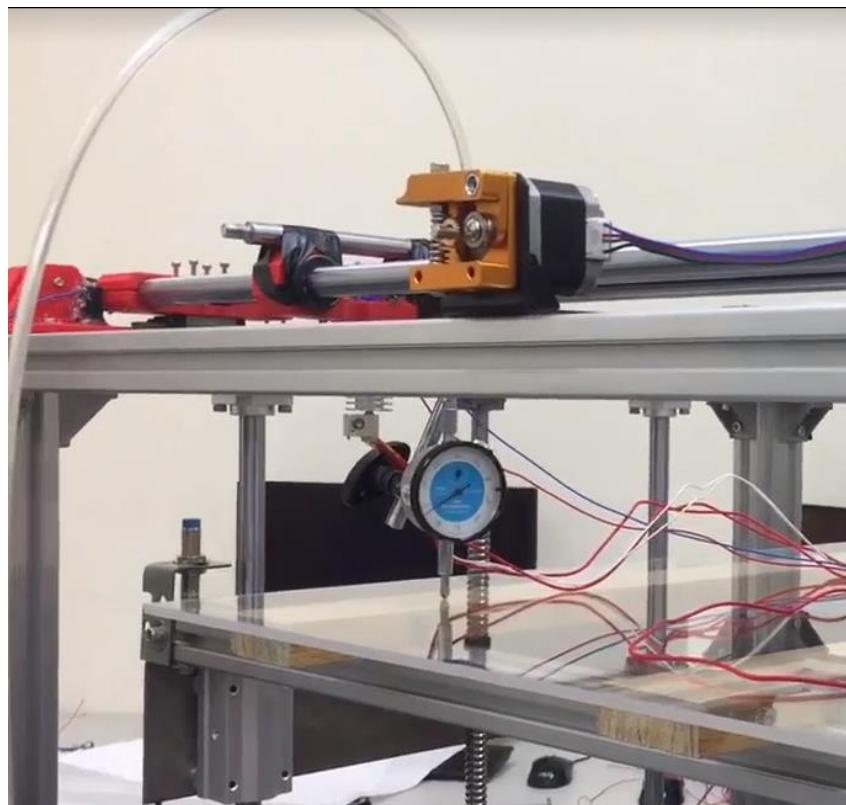


圖 8.5 水平儀校正平台及骨架水平



平台部分則是用壓克力板與木條，木條部分是為了讓壓克力板保持平行。

圖 8.6 平台與壓克力板

5. 裝上接近開關後進行回歸原點測試(3 軸)，然後以噴頭為基準去察看是否與平台距離差很多，如果差很多就去調接近開關的高低即可，要調到平台與噴頭能有稍微的干涉與不干涉即可，若有厚薄規校正則更加精準。

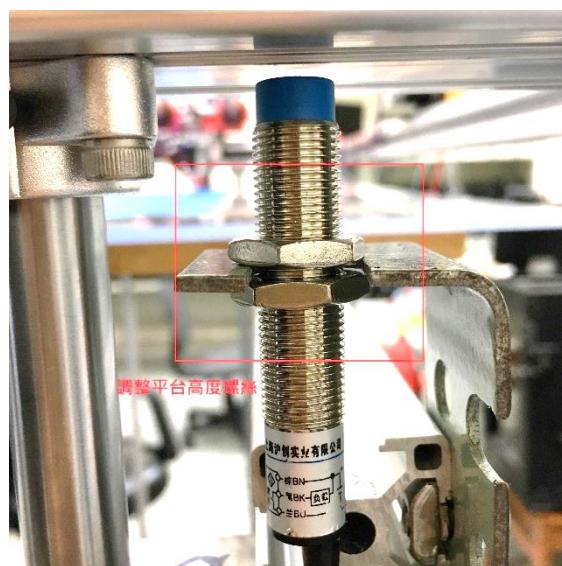


圖 8.7 接近開關校正

6. 以上都校正完成之後，回原點，回完之後把 Z 軸往下降
0.3mm(0.3 為印表機第一層高度印製)然後把噴頭移動到平台上，
做手動微調及校正，兩邊都要。

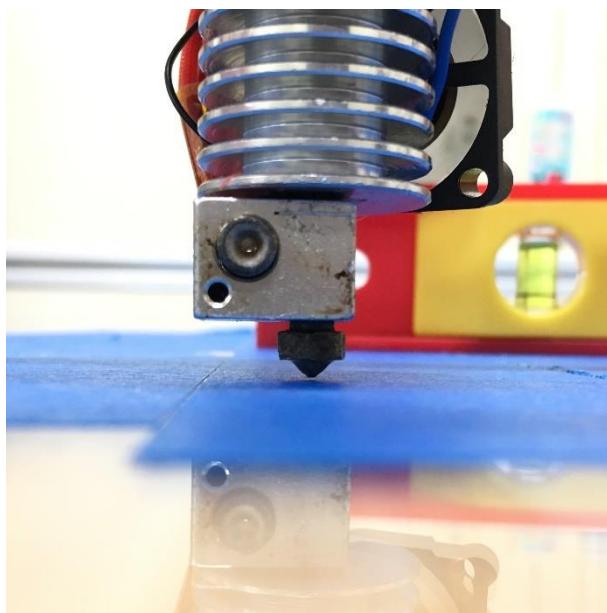


圖 8.8 平台右方水平

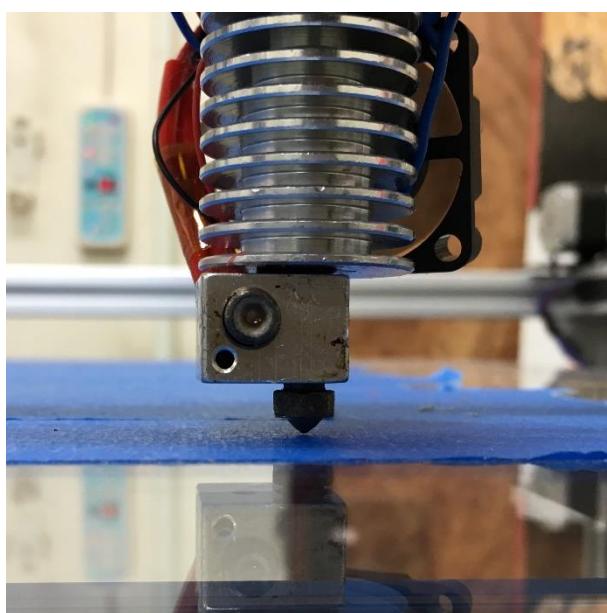


圖 8.9 平台左方水平

8.3 X Y Z 定位精度測試

一開始要先把機台回歸原點，之後掛表固定好，固定好後，即可開始測試定位精度。

1. 一開始要先把機台與掛表接觸，並且掛表歸零

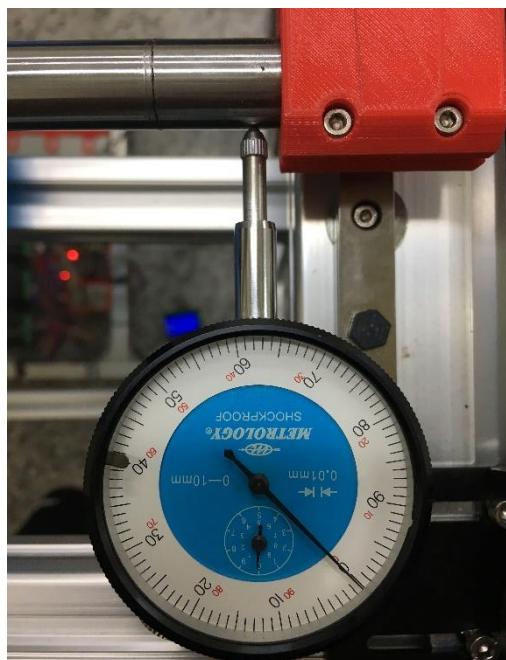


圖 8.10 掛表歸零

2. 掛表歸零完後，即可開始測試定位精度，調整 LCD 螢幕中的準備 > 移動軸 > 移動 Y 軸 > 移動 1mm(上圖示針對 Y 軸的定位精度去做測試的)

3. 移動 5mm 後，查看機台上的掛表上是否定位出正確的尺寸。



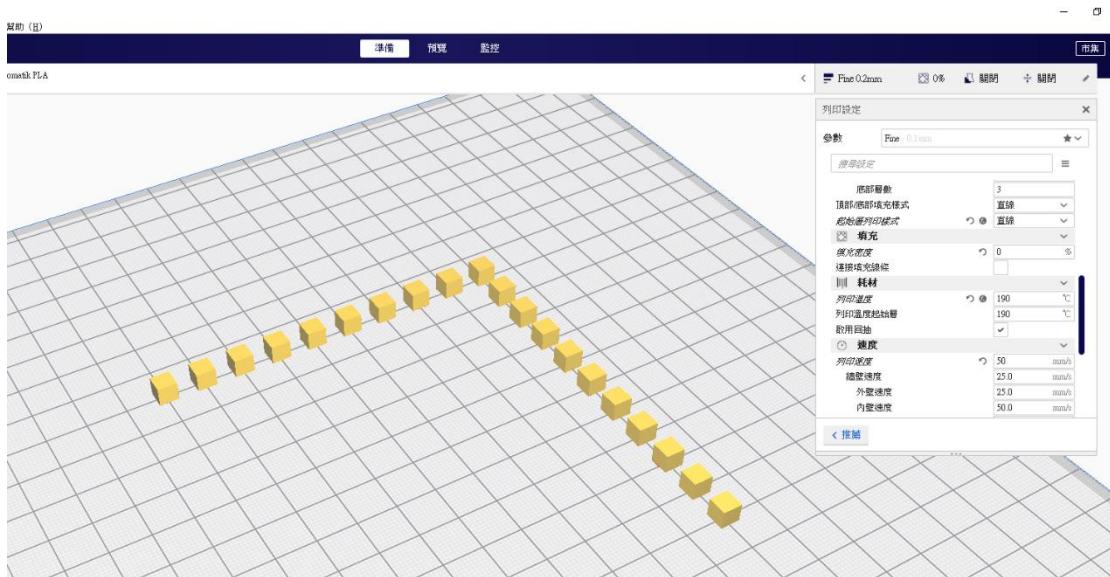
圖 8.11 移動 Y 軸後查看表是否準確

4. 定位有誤差如果在正負 0.02mm，代表機台精度是有的，韌體的步進馬達部分有算精確，那如果誤差過大例如正負 1mm 以上代表韌體的步進馬達部分數值有算錯(參照步進馬達計算)，如果誤差在正負 0.1mm 內這幾 mm 有可能是因為固定掛表的部分沒有固定好，而導致掛表位移，或者是接觸面有鬆動而造成的。(X、Y、Z 軸參照這種方式即可)

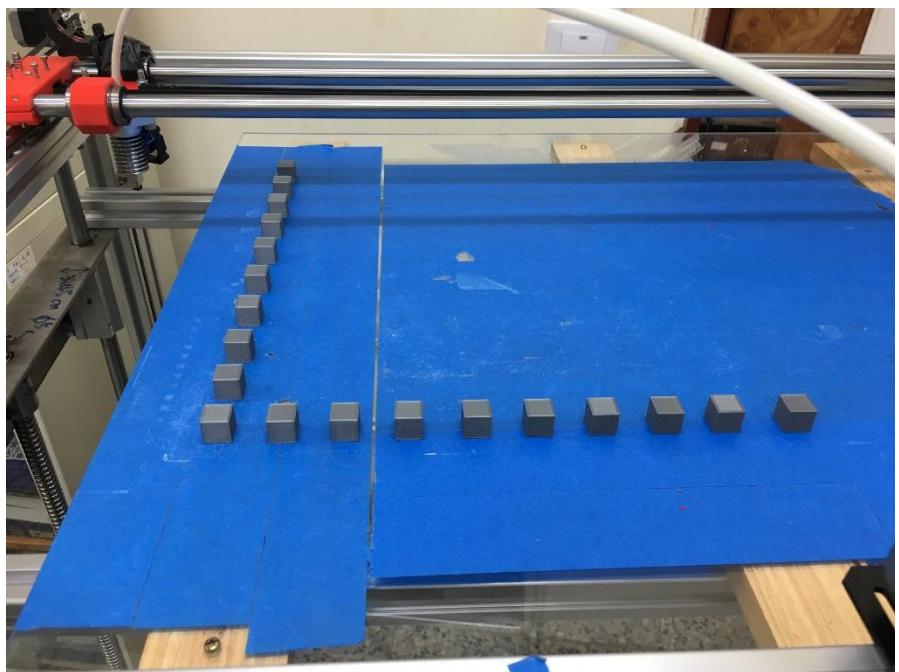
第九章 列印測試



一次列印 19 顆 長 15 寬 15 高 15(單位 mm)的方塊，並且以排列的方式去列印，每顆的間距是 30mm。



接著上機去實際列印測試



列印出來後拿游標卡尺去量測每顆方塊長寬高之尺寸並記錄下來。

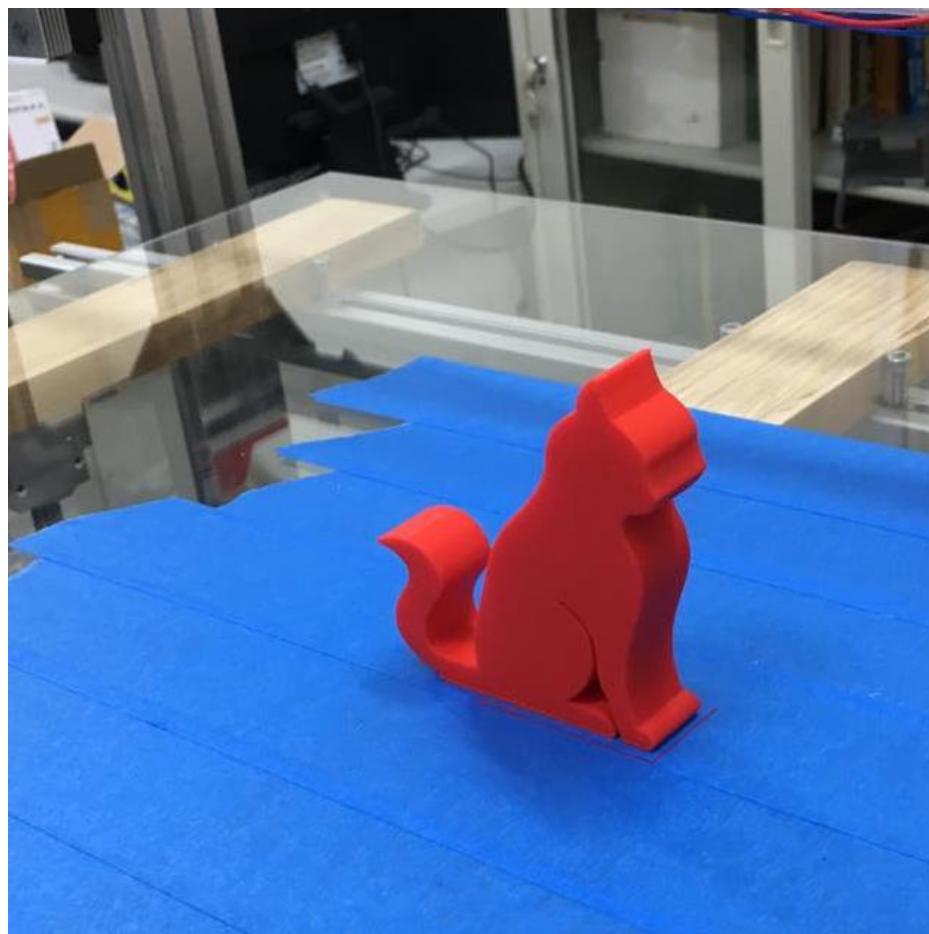
編號	X	Y	Z
1	15	15	14.8
2	15	15	14.7
3	15	15	15
4	15	15	15
5	15	15	15
6	15	15	14.85
7	15	15	15
8	15	15	14.8
9	15	15	14.8
10	15	15	14.7
11	15	15	14.75
12	15	15	14.7
13	15	15	15
14	15	15	15
15	15	15	14.75
16	15	15	15
17	15	15	14.7
18	15	15	15
19	15	15	14.9



單位:mm

上表表示出 Z 軸座標的準確度是有誤差的，但是我們使用的是
5mm 厚的壓克力透明板，壓克力的變形量很大所以我們所做的
3D 印表機誤差在正負 50mm。

動物造型手機架型



第十章 總結

目前我們組立的列印機已經能夠運作，利用傳輸線來直接連線列印，並能達到大量列印，因此可以將整個系統從無到有自己設計組裝，對於機台的參數設定、校正和維修，能夠完全掌控及解決問題。

未來若有多個機台能夠彈性製造，加裝 Webcam 能夠遠端控制並做影像辨識的功能，有問題能夠紀錄以便除錯也可以馬上將物件換一台機器繼續執行，朝向彈性製造發展。

附錄

機械零件

件號	名稱	材料	規格（單位: mm）	數量	備註
1	鋁擠製	鋁	30x30x1000	4	
2	鋁擠製	鋁	30x60x1200	4	
3	鋁擠製	鋁	30x30x660	2	
4	鋁擠製	鋁	30x60x660	4	
5	鋁擠製	鋁	30x30x950	2	
6	鋁擠製	鋁	30x30x940	2	
7	光軸	軸承鋼	Ø16x950	2	X 軸用
8	光軸	軸承鋼	Ø16x1200	4	Z 軸用
9	光軸滑塊	鋁合金	SCS16LUU	4	
10	軸承	軸承鋼	Ø16	2	
11	軸座	鋁合金	SHF16	9	
12	軸座	鋁合金	Ø12(UFL001)	4	
13	滾珠螺桿		Ø16x1200	2	頭尾車 Ø12 長 45
14	齒傳動輪	不鏽鋼	GT2-20T 內徑 5 帶 寬 6	4	馬達用
15	傳動帶輪 A	不鏽鋼	GT2-20T 內徑 5 帶 寬 6	3	非馬達用
16	傳動帶輪 B	不鏽鋼	XLGT2-20T 內徑 5	2	非馬達用
17	傳動帶輪 C	不鏽鋼	XLGT2-30T 內徑 12	2	非馬達用
18	皮帶 A	橡膠	長 1800 帶寬 6	1	X 軸用
19	皮帶 B	橡膠	長 1450 帶寬 6	2	Y 軸用
20	皮帶 C	橡膠	GXP-100XL 環形皮 帶 帶寬 10	2	Z 軸用
21	線性滑軌		MGN-12H 滑軌長 550	2	
22	直角連接件(套餐 4 含螺帽 * 2、螺絲 * 2、墊片 * 2)		30x20x30	32	
23	噴頭		0.4	1	
24	MK8 擠出機 2 鋁座	鋁合金	左手	1	
25	42 步進馬達固定座			6	

26	散熱鳍片	鋁	69x69x36	1	
27	螺母座	鋁合金	SFU1605	2	
28	六角承頭螺栓		M6x10	100	
29			M5x40	100	
30			M5x32 或 M5x30	100	
31			M5x24 或 M5x22	200	
32			M5x20	200	
33			M5x16	100	
34			M5x14	100	
35			M4x25	100	
36			M4x16	100	
37			M4x12	100	
38			M4x10	100	
39			M3x36 或 M3x35	100	
40			M3x18	100	
41			M3x12	100	
42			M2.5x6	100	
43	T型螺帽		M6	50	
44			M5	100	
45			M4	100	
46			M3	100	
47	六角螺帽		M5	20	
48			M4	10	
49			M3	20	
50	腳輪	鋁合金	2吋	4	

電子材料

件號	名稱	材料	規格（單位: mm）	數量	備註
1	42 步進馬達			6	
2	3D 主板(KS Gen-1 兼容 ramps 開源 marlin)			1	
3	電源供應器		LRS-350-24 NES- 350-24 350W 24V 15A	2	
4	馬達驅動器		42V 1/32 微步	3	
5	LCD 控制板 RAMPS 3D 印表機			1	
6	極限開關			2	
7	DCV 接近開關			1	常開型
8	降溫風扇		30x30x10	1	
9	A4899			1	

訂製材料

件號	名稱	材料	規格（單位: mm）	數量	備註
1	支撐版	鋼材		2	L型
2	惰輪座	PLA		2	3D 列印
3	線材座	PLA		1	3D 列印
4	線材座的鉤子	PLA		1	3D 列印
5	線材座的銷	PLA		1	3D 列印
6	噴座頭	PLA		1	3D 列印
7	光軸固定塊	PLA		2	3D 列印

其他

件號	名稱	材料	規格（單位: mm）	數量	備註
1	木板		600x60x20	3	
2	壓克力板		50x70x1	1	
3	坦克鍊			2	

硬體接線配線圖

