臺北市立大安高級工業職業學校專題實作競賽

「專題組」作品說明書

群別：電機與電子群

作品名稱：檢色筆

關鍵詞：顏色檢測、回歸分析、大數據

目錄

圖目錄

表目錄

【檢色筆】

壹、 摘要

本專題以檢測顏色為核心概念，設計出一支具有檢色功能的觸控筆，搭配自建伺服器和多款自行設計的繪畫軟體插件，組成一套完整的線上繪畫系統。只需將該產品連上網路，藉由伺服器的中繼，能讓使用者更輕鬆地把現實中的顏色轉換成精準的 RGB 值，並自由的在多款數位繪畫軟體中繪出檢測到的顏色。

為了提升 RGB 值的檢測精準度，且消除檢色晶片自身的誤差，本專題檢測 2000 多張色卡，建立自己的大數據，對原始測量值和色卡廠商提供的數值做對比，把兩者數據值的變化關係做比較後利用回歸分析，得出三組三元二次的轉換方程式，使原始測量值經過該函式轉換得到和色卡相似的 RGB 值。

貳、 研究動機

有沒有遇到過繪圖時覺得挑選顏色好難？在畫電繪和做 3D 模型時找不到準確的顏色？又或是每次找顏色時，都得用很複雜的方法，進而浪費一大堆時間呢？

在當今蓬勃發展的數位藝術領域，我們所處的世界充滿無限的色彩選擇。然而，找到理想的色彩往往是一項挑戰，創作者因在數位創作時找不到合適的顏色而煩惱，或許不同於現實的顏色能創造出特別的視覺效果，但在追求現實的繪者手中，如何快速找尋正確的顏色就是迫在眉間的問題，而這些需求是傳統觸控筆沒辦法滿足的。為了能更自然、更順暢地應付多元色彩的需求，我們打算開發一款能夠檢測顏色並即時應用於電繪的觸控筆，以提供更流暢、更直觀的創作體驗。

或許大家曾在網路上看過各式的檢色器，但那麼大顆的檢色器是否能稱得上靈活輕巧？如果每次出門都要多帶一顆檢色器是否相當的不便，並降低使用者的使用意願？但只要把檢色器和觸控筆結合，讓繪者在各類場合都能方便攜帶和使用，就能大幅增加其使用的意願。

我們希望做出一枝能大幅降低電繪入門程度的筆，讓對顏色較不敏感的人群也能輕鬆享受電繪的樂趣，同時讓已有一定程度的繪畫者能更快速的查找顏色，加快繪畫的步調，並讓需要準確標示顏色的使用者能用更加簡單的方式讀取物體顏色值，為此來拉近大眾和數位繪畫的距離，把藝術更好的推廣給大家。

參、　主題與課程之相關性或教學單元之說明

一、硬體製作

(一) 3D 繪圖與列印

我們利用高二彈性課程時所學的繪圖軟體Inventor繪製檢色筆的外觀如圖 1，並配合 3D 列印技術如圖 2 打造檢色筆的主要結構。

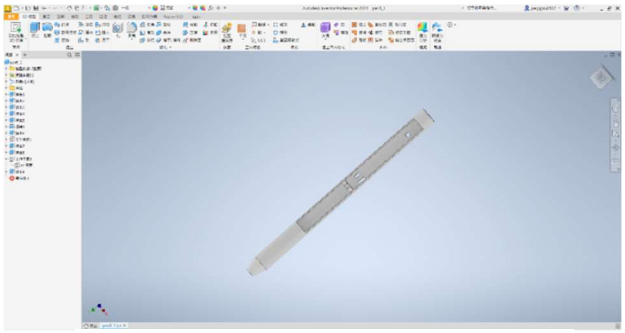


圖 1　Inventor 的繪畫介面

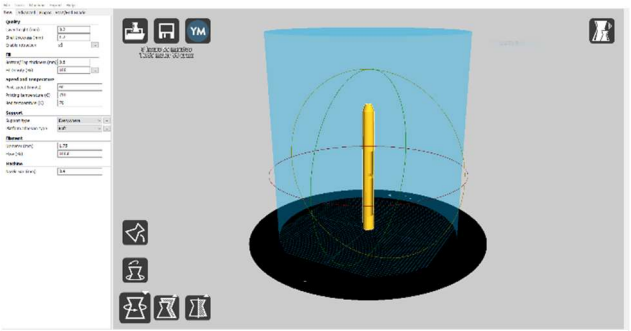


圖 2　3D 列印參數調整

(二) 雷射切割

我們利用高二彈性課程時所使用雷射切割軟體 RDworksV8 和繪圖軟體 Inventor 做結合，把 Inventor 中的 2D 草圖檔案如圖 3 輸出到 RDworksV8 中做雷射切割數據的設定如圖 4，最後用雷切機切割板材，製成檢色筆的展示架和專屬收納盒。

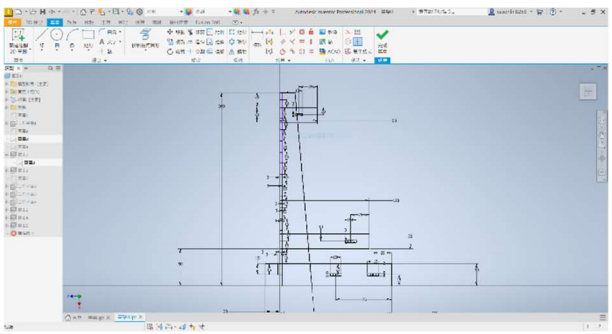


圖 3　Inventor 的草圖檔

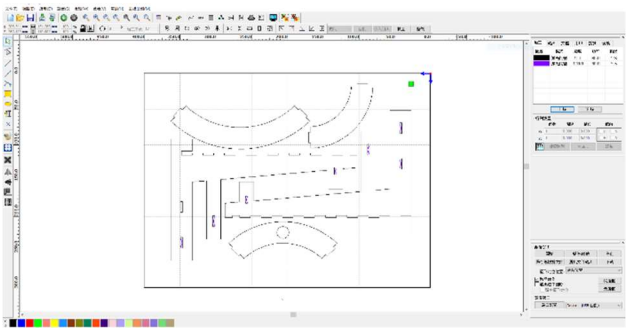


圖 4 雷射切割調整介面

(三) 電路板設計

我們利用高二電子學實習時使用的電路板切割技術 PCB，在電路設計軟體 Altium Designer 中繪製電路圖如圖 5 及設計 PCB 電路板如圖 6，來打造檢色筆專用的接線板，以取代傳統雜亂的導線連接。

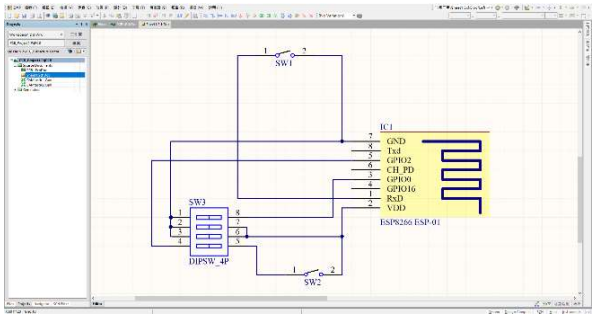


圖 5 電路圖設計

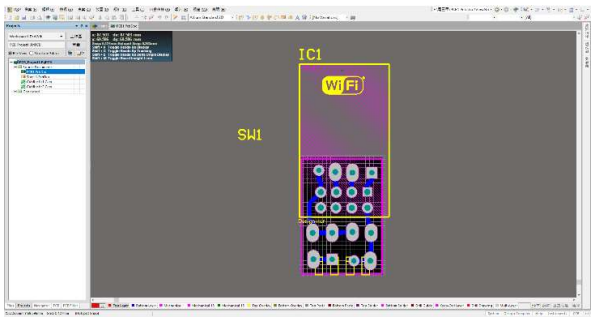


圖 6 PCB 排版設計

二、程式撰寫

(一) Arduino

我們利用高二智慧居家監控實習中學到的 Arduino 圖 7，作為檢色筆核心晶片 ESP-01 的驅動程式，使其能連接網路並完成檢測數值的讀取、轉換和傳遞。

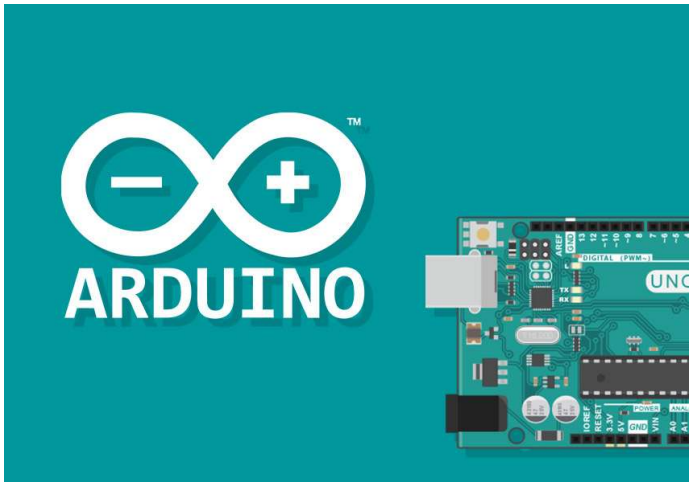


圖 7 Arduino

(二) Python

我們利用高一資訊課所學的基礎 Python 語法圖 8 並在課餘時間加以延伸，搭配網路上查找的相關資料寫出一段程式，把我們輸入至 Excel 的原始測量 RGB 值和色卡附的 RGB 數值做比對，得出三組數值轉換函式。讓原始測量值經過該函式的轉換，得到接近實際的 RGB 值。同時也使用該語言打造程式插件，讓檢色筆可以搭配多種開源繪軟體，達成檢色筆最重要的目的。

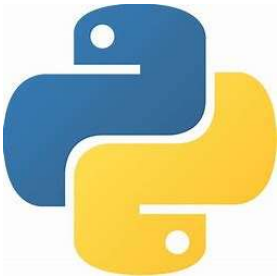


圖 8　Python

(三) JavaScript

我們利用課餘時間自學 JavaScript 語法圖 9，搭配 Node.js 編程，使用訂閱的網域名，在樹莓派上架設了專屬於檢色筆的繪畫伺服器，作為檢色筆和使用者繪畫設備的溝通橋樑，該伺服器也支援線上繪畫和同步檢色筆，作為數據校正的測試平台。

圖 9　JavaScript

肆、 研究步驟

一、研究流程

在六月底訂下專題題目後，組員們開始卯足全力查找相關網路資料，並在七月初買了第一批測試元件，著手麵包板接線測試和基礎程式編寫。九月初做第一次人眼數據校正。九月中旬基礎伺服器架設完成。在十月時進行 129 色的顏色校正且完成其數據分析，並優化軟體系統和硬體接線。到了十一月尾聲筆身也被設計完成，並使用 3D 列印技術做出第一版的檢色筆。其餘時間都花在顏色數據庫的收集和筆身外觀的微調，且在隔年一月成功完成最後一次大數據的回歸分析，最後測試了檢測穩定性和動作影片拍攝，整個檢色筆研究才算告一個段落。其大致時程如下圖 10。

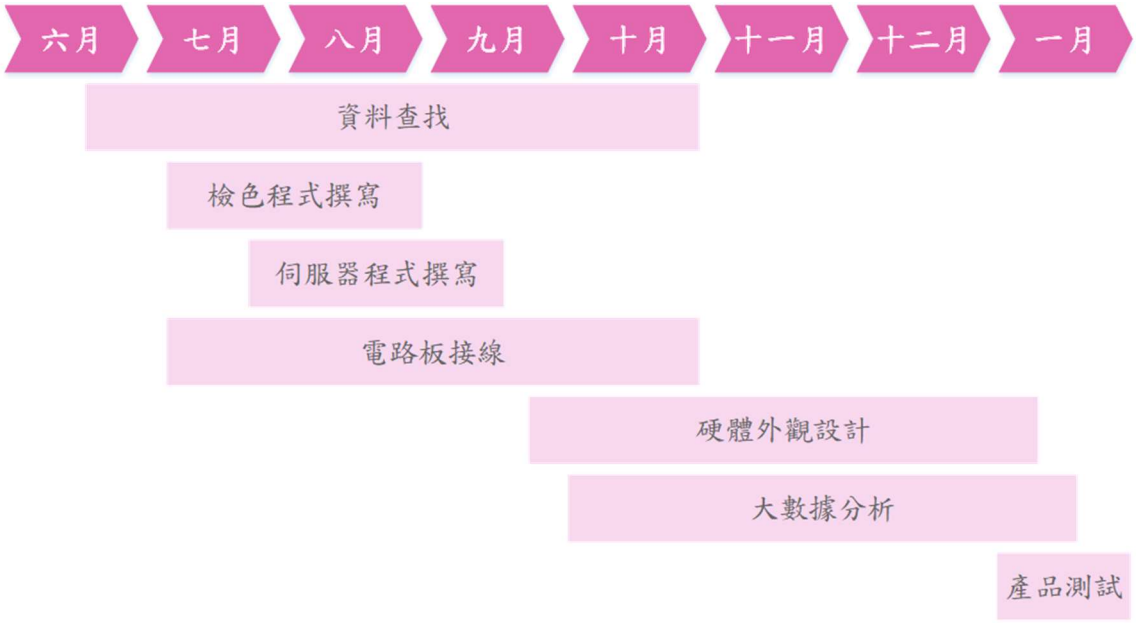


圖 10 時程表

二、使用材料

(一) ESP-01

我們使用該晶片作為檢色筆的核心晶片，憑藉它迷你的身形能輕鬆地放入檢色筆中，其中兩根類比接腳被用來讀取檢色晶片的數值，並換算該數值再傳送給伺服器。其外觀如圖 11 規格如表 1

表 1　ESP-01

ESP-01	
尺寸	24.7 *14.4 *11.0 mm
供電電壓	3.0V ~ 3.6V
供電電流	> 300mA
天線規格	板載 PCB 天線

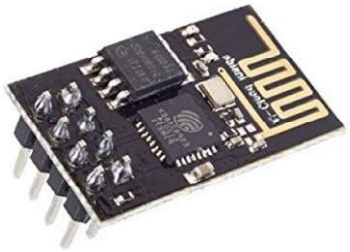


圖 11　ESP-01

(二) TCS34725 RGB Color Sensor

TCS34725 是一款顏色傳感器，用於測量紅色、綠色和藍色的強度，輸出對應的數值來表示物體的顏色，但並非標準的 0~255 RGB 值。為了配合圓柱形的筆桿，我們選用圓形的版本。其外觀如圖 12 規格如表 2。

表 2　Color Sensor

TCS34725 RGB Color Sensor	
尺寸	直徑 16mm
工作電壓	3.3-5V
工作電流	65 uA
檢測距離	3 ~ 10 mm

圖 12　Color Sensor

(三) Raspberry Pi （樹莓派）

我們選用樹莓派 4B 這款高性能且多功能的單板電腦來架設伺服器，當作器具間通訊的橋樑，讓測量到的值傳輸到軟體的插件中，並作為自製繪畫網站的運算核心。其外觀如圖 13 規格如表 3。

圖 13 Raspberry Pi

表 3 Raspberry Pi

Raspberry Pi 4B	
電壓	5V
電流	3A
處理器	Broadcom BCM2711
工作溫度	0-50ºC

三、使用軟體與服務

(一) Autodesk Inventor

一款能輕鬆入門的工業繪圖軟體，擔任這次專題的硬體設計擔當，我們用它來建模和繪製草圖，是硬體製作的根本。

(二) RDWorks V8

雷射切割的編譯程式，負責調整雷射切割機的各項參數。

(三) Ultimaker Cura

3D 列印的編譯程式，負責調整 3D 列印時的各項參數。

(四) Altium Designer

PCB 電路板設計軟體，能在上面設計線路、編排元件位子，並將脫機檔輸出至雕刻機中進行切削。我們用其設計自己的電路板以減少導線連接。

(五) Arduino IDE

作為 Arduino 語法的開發環境，能配合多種晶片進行編譯和多樣的程式庫可供使用。我們用其對 ESP-01 晶片進行程式的編譯。

(六) Visual Studio Code

一款跨平台免費原始碼編輯器簡稱 VS Code，本專題多數程式包括網頁、插件和迴歸分析都是由其進行編譯。

(七) Node.js

作為一個基於 Chrome 引擎的 JavaScript 運行環境，用於構建輕量級且高效的後端服務。被我們用於伺服器的架設和編輯。

(八) **Vue.js**

是一個進階的 JavaScript 框架，讓我們做出繪畫網站介面。

四、使用機具

(一) 3D 列印機

我們採用多種型號的 3D 列印機製作硬體外觀，這種方法具有極高的自由度，且無需使用傳統的模板，正是我們所青睞的特點。

(二) 雷射切割機

負責加工平面的木材或壓克力材料，相較傳統切割有著更高的精度，我們用其特性製作有卡榫的片狀物件，來組成架子及盒子。

(三) 電路板雕刻機

專門用於雕刻 PCB 板的機器，用於產出自製電路板。

伍、 研究結果

一、主體結構

以 PLA 材料為主體，做成一個裝載電子元件的圓柱形容器，大致構造如圖 14，實體如圖 15。

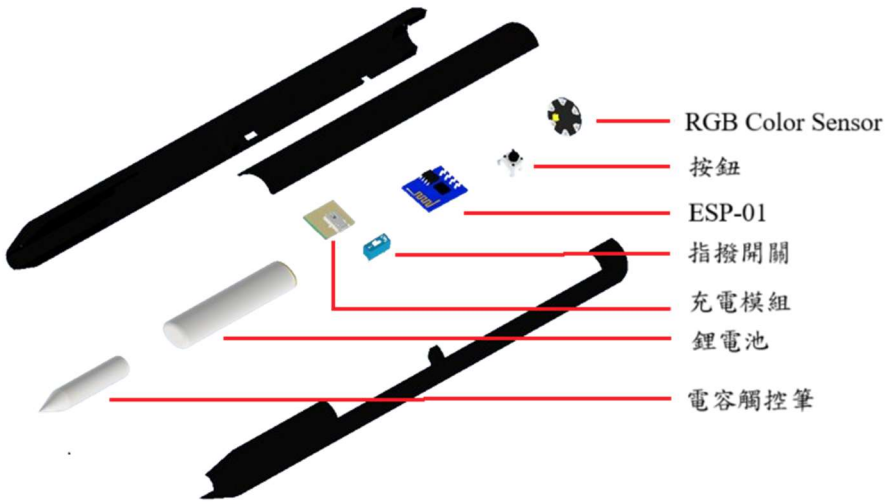


圖 14 檢色筆結構爆炸圖

圖 15 產品實體

二、軟體通訊架構

設備通訊方面，我們最終選擇使用網路通訊，原因有兩點：第一，網路伺服器我們已有相關架設經驗，會比使用藍牙系統更好上手和除錯；第二，藍牙模組有時會和晶片產生衝突並且來要另外多加一顆藍牙模組在系統中，大幅增加操作難度和整體大小。

在整個通訊過程中使用了三種程式語言來相互配合，從 Arduino 的類 C 語言，回傳到伺服器使用的 JavaScript 語言，最後傳輸到 Python 寫成的插件上，各元件的通訊方式大致如下圖 16。

1. ESP-01 感測到檢色按鈕按下
2. 檢色晶片發出強光開始檢測
3. 原始數值(0-65535)回傳給 ESP-01
4. ESP-01 經過回歸公式轉換原始數值變成標準 RGB 值(0-255)
5. ESP-01 把轉換後的 RGB 值傳回伺服器
6. 伺服器把訊號傳給使用者設備
7. 最後經由插件將數值傳入開源繪圖軟體中

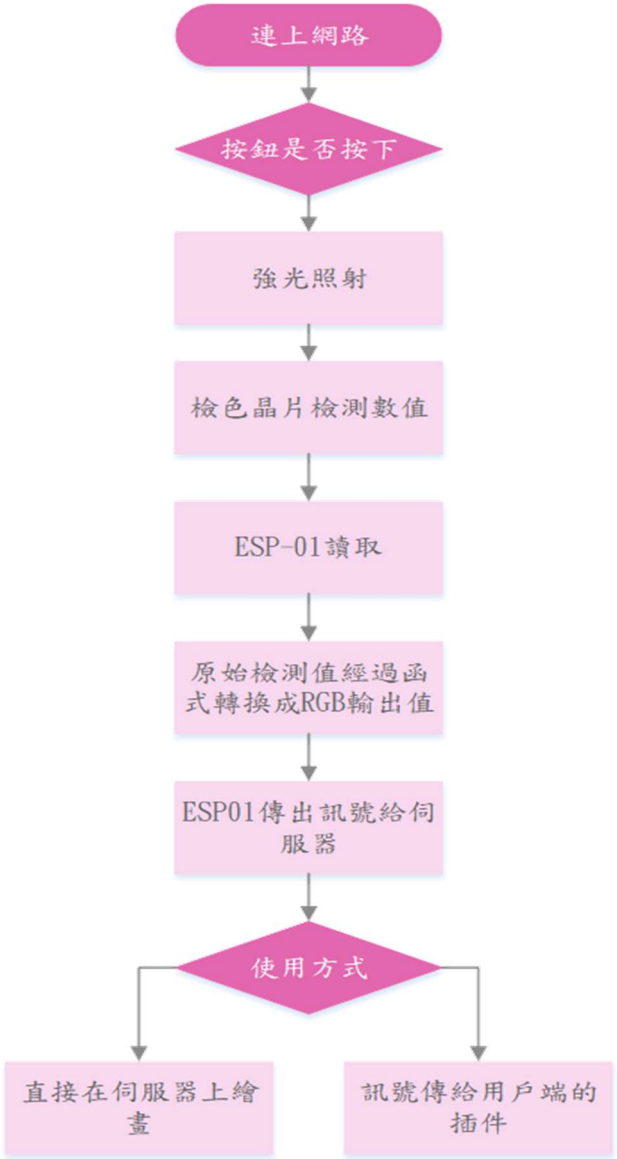


圖 16 通訊流程圖

(一) ESP-01 端

負責執行數值的轉換，把晶片檢測到的值，經過數據回歸推出的公式轉換成 0-255 的 RGB，並將其透過網路傳送到伺服器，如圖 17 為網路連線和回傳的程式。

7777.ino | ColorUpdater.cpp | ColorUpdater.h

```cpp
#include "ColorUpdater.h"

ColorUpdater::ColorUpdater() {}

bool ColorUpdater::updateColor(uint8_t red, uint8_t green, uint8_t blue) {
    if (WiFi.status() != WL_CONNECTED) {
        return false;
    }

    HTTPClient http;
    http.begin(_client,_serverURL);
    http.addHeader("Content-Type", "application/x-www-form-urlencoded");

    String postData = "red=" + String(red) + "&green=" + String(green) + "&blue=" + String(blue);
    int httpResponseCode = http.POST(postData);

    http.end();

    return (httpResponseCode == 200);
}
```

圖 17　ESP-01 中的程式

(二) 伺服器端

伺服器端分成兩邊，一邊為溝通伺服器負責接收檢色筆的訊號並讓插件讀取，插件能藉由特定路徑得到一組 16 進制色彩標示的.json 格式字串進行，另一邊為繪畫伺服器負責繪畫網站的運作，用來檢色效果的校正和檢驗。如圖 18 為伺服器的端口和啟動程序。

```cpp
const API_PORT = 8002;
const VUE_PORT = 80;

apiApp.listen(API_PORT, '0.0.0.0', () => {
    console.log(`API 伺服器在 http://0.0.0.0:${
API_PORT}`);
});

vueApp.listen(VUE_PORT, '0.0.0.0', () => {
    console.log(`Vue 伺服器在 http://0.0.0.0:${
VUE_PORT}`);
});
```

圖 18　兩種伺服器

(三) 自製插件

為了讓檢色筆能作畫在各平臺中，我們自行開發插件，目前做了兩款，分別適用於大眾熟悉的 3D 和 2D 繪圖軟體——Blender 和 Inkscape，只要連上網路並下載插件，就能在兩款軟體中分別找到窗口圖 19 Blender 操作介面和圖 20 Inkscape 操作介面，來使用檢色筆測量到的值畫出自己想要的顏色。圖 2121 為 Blender 插件程式碼片段，將資料轉換為 Blender 中的 RGB 值；圖 22 為 Inkscape 插件程式碼片段，功能為從伺服器取得的顏色應用到 Inkscape 中。

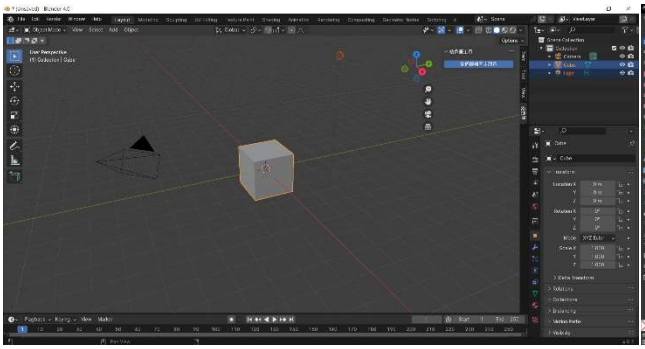


圖 19 Blender 操作介面

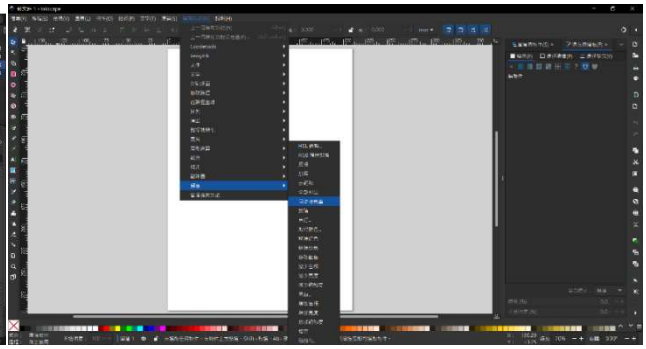


圖 20 Inkscape 操作介面

```python
import bpy
import requests

class COLORPICKER_OT_request_color(bpy.types.Operator):
    bl_idname = "material.request_color"
    bl_label = "從伺服器同步顏色"

    def execute(self, context):
        url = "http://color-magic.bw-legend.net:8002/color"
        response = requests.get(url)

        if response.status_code != 200:
            self.report({'ERROR'}, "Failed to retrieve color from server!")
            return {'CANCELLED'}

        color_hex = response.json().get('color', '#FFFFFF')[1:]  # 假設如果沒有取得顏色就默認為白色
        color_rgb = tuple(int(color_hex[i:i+2], 16) / 255.0 for i in (0, 2, 4))

        if bpy.context.object and bpy.context.object.active_material:
            bpy.context.object.active_material.diffuse_color = (*color_rgb, 1)  # RGBA
            self.report({'INFO'}, "Color set from server!")
        else:
            self.report({'ERROR'}, "Object has no active material!")

        return {'FINISHED'}
```

圖 21 Blender 插件

```xml
<?xml version="1.0" encoding="UTF-8"?>
<inkscape-extension xmlns=
"http://www.inkscape.org/namespace/inkscape/extension">
  <name>同步檢色筆</name>
  <id>org.inkscape.tutorial.make_red_extension</id>
  <effect>
      <effects-menu>
          <submenu name="Color"/>
      </effects-menu>
  </effect>
  <script>
    <command location="inx" interpreter="python">
my_effect_extension.py</command>
  </script>
</inkscape-extension>
```

圖 22 Inkscape 插件

三、相關系統和配件

(一) 線上繪畫網站

我們還在伺服器上架設了一個網頁，並用 vue.js 對其編輯，訂閱網域名使其能被外界使用，主要功能為繪畫和測試檢色效果，使用介面如圖 23 和功能如圖 24。

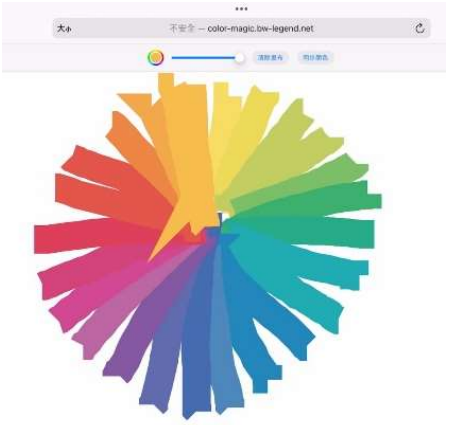


圖 23 網頁介面

圖 24 網頁功能

(二) 筆架和筆盒

為了能更好展示本產品和攜帶，我們用壓克力和木板分別做了為圖 5 直立式展示架和圖 26 為簡易收納盒。

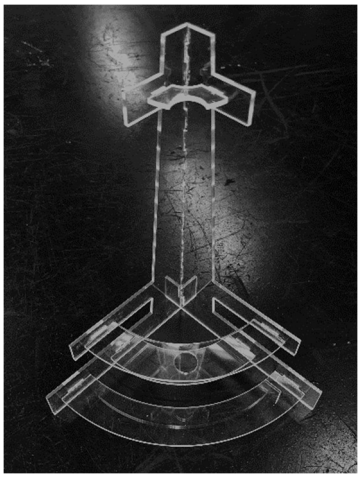


圖 25 筆架成品外觀

圖 26 收納盒成品外觀

四、大數據分析

顏色檢測晶片測量出來的值是三組 0~65535 的類比訊號，分別表示紅、藍和綠三種光譜的反射強度，但我們較常使用的 RGB 值是 0~255，為此我們必須對該數值做轉換，最初我們最直接的使用 map 語法，這語法會把 65535 平均成 255 個小格，來做到數值映射的功效，但結果慘不忍睹，經過該方法轉換並輸出的顏色會在兩極端出現極大誤差，這也使我們意識到無法直接做線性轉換，因此我們開始了大數據收集，希望能分析出其變化的趨勢。

起初我們嘗試做了 129 色的變化趨勢圖，發現曲線會相似一條一元二次方程式。我們以此為基礎做顏色校正並進行數據分析，得出了三組方程式分別對應紅、藍和綠三種顏色數值，但實際使用發現顏色還是存在些許誤差，經過一個月的測試，我們發現三種數值是會互相影響的，因此三組一元二次的公式被改成三組三元二次的公式，結果相當令人滿意。抱持著數據越多，換算結果越精準的想法，我們開始了 2000 多色的大數據收集，結果如圖 27。

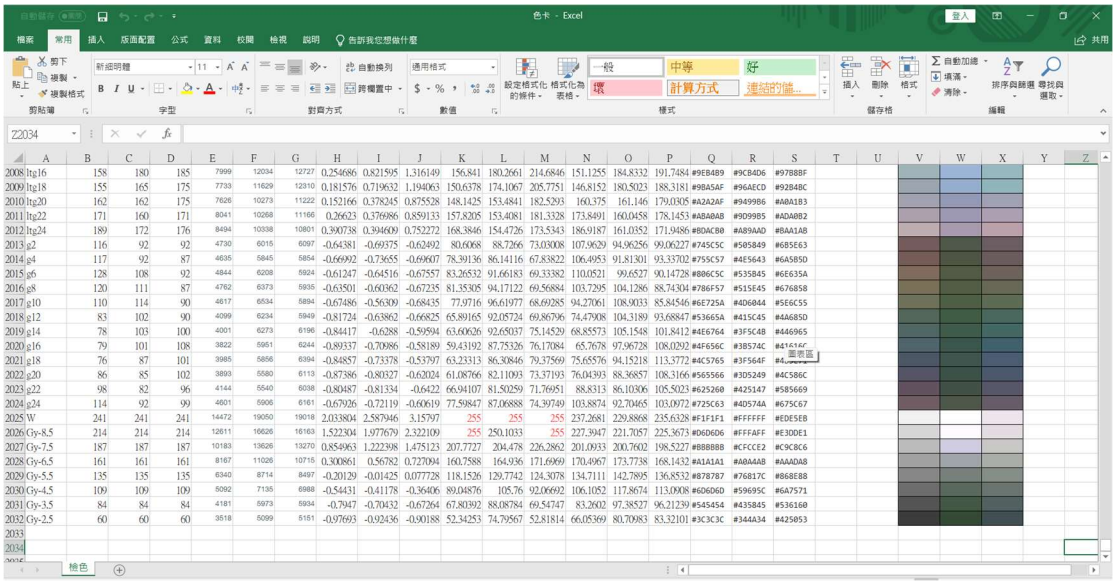


圖 27　2000 多筆色彩數據

(一) 數據庫

下表 4、表 5 為 Excel 表格中的片段，用以方便說明。
A 欄為：該色卡的變化
B～D 欄為：色卡公司的官方給的數值
E～G 欄為：晶片測量出的原始數值
H～J 欄為：原始數值標準化後的數值
K～M 欄為：為一元一次的回歸後的結果
N～P 欄為：為三元二次的回歸後的結果
Q～S 欄為：為 16 進制色彩的標示法
V 欄為：為色彩表示，顯示 B～D 欄的結果
W 欄為：為色彩表示，顯示 K～M 欄的結果
X 欄為：為色彩表示，顯示 N～P 欄的結果

表 4　數據庫 Excel 表格前

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	PCCS	R	G	B	r	g	b	x0	x1	x2	R'	G'	B'
2	v1	185	31	87	6747	4645	5254	-0.0894	-1.0387	-0.8717	127.644	67.89105	55.0188
3	v2	208	47	72	8641	5082	5488	0.43114	-0.9286	-0.8032	171.813	74.53713	60.0184
4	v3	221	68	59	10272	5678	5575	0.87943	-0.7786	-0.7777	209.848	83.60136	61.8772
5	v4	233	91	35	11276	6357	5406	1.15538	-0.6076	-0.8272	233.262	93.92788	58.2664
6	v5	230	120	0	12181	8233	5805	1.40412	-0.1353	-0.7104	254.367	122.4589	66.7913

表 5 數據庫 Excel 表格後

N	O	P	Q	R	S	T	U	V	W	X	Y	Z
R"	G"	B"	#16	#16'	#16"			real	line	final		
190.567	49.5029	79.242116	#B91F57	#7F4337	#BE314F							
219.7155	52.2191	72.120129	#D02F48	#AB4A3C	#DB3448							
228.0454	61.7807	57.578955	#DD443B	#D1533D	#E43D39							
229.5941	76.3251	33.679404	#E95B23	#E95D3A	#E54C21							
236.5696	117.225	18.890804	#E6780	#FE7A42	#EC7512							

(二) 落點分析

圖 28、圖 29 和圖 30 的 Y 軸為原始測量值；X 軸為官方提供之色卡數值，我們分析其落點發現圖形像一條一元二次的曲線圖形，但經過我們的實驗發現，三種色光會互相影響，所以實際上應該是三元二次，圖中的虛直線為第一次迴歸分析的結果(一元一次回歸公式)，虛曲線為最終迴歸分析的結果(三元二次的回歸曲線公式)，此公式也是目前被我們使用的轉換公式。

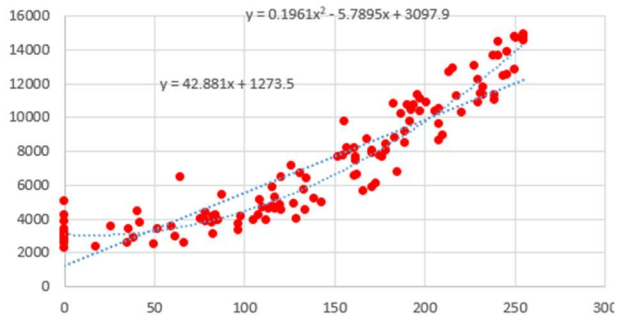


圖 28　R 值分布-1

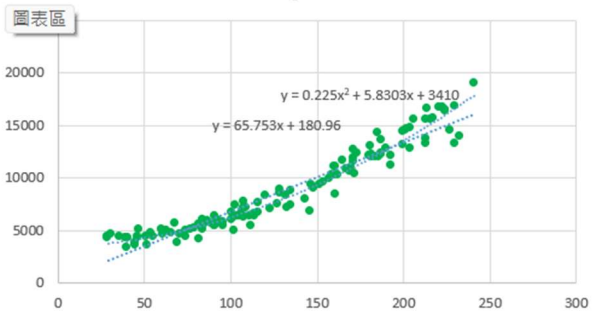


圖 29　G 值分布-1

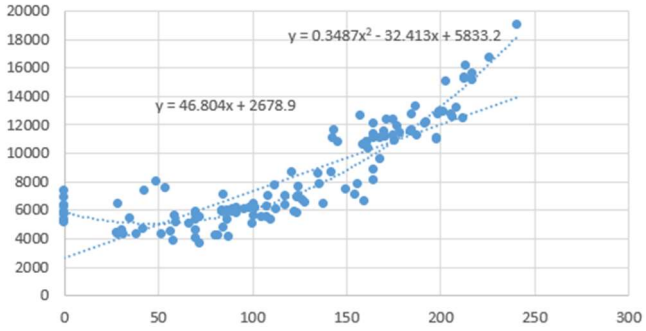


圖 30　B 值分布-1

圖 31、圖 32 和圖 33 的 Y 軸為檢色筆實際輸出的值；X 軸為官方提供之色卡數值，原始測量值經過公式的轉換後就是實際輸出的值，而中間的直線為理想轉換線(斜率等於 1)，越趨近中間直線表示轉換效果越好，也就是兩者的色差越小，該圖用來證明轉換函數的效果和準確度。

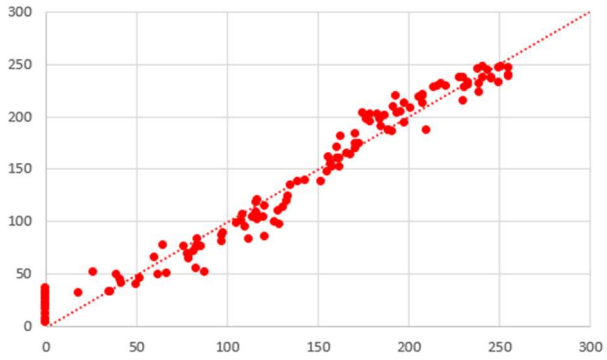


圖 31　R 值分布-2

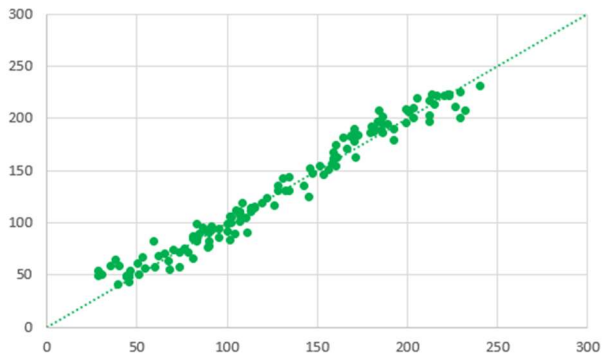


圖 32　G 值分布-2

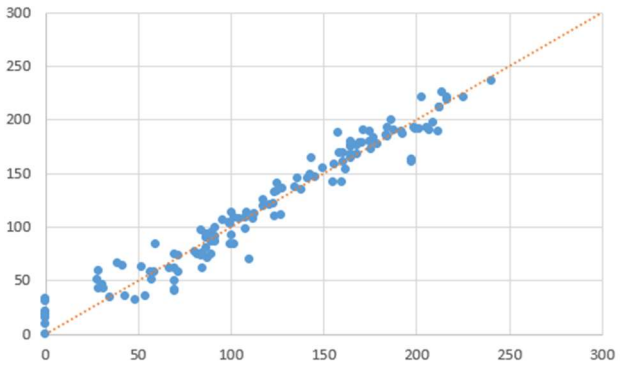


圖 33　B 值分布-2

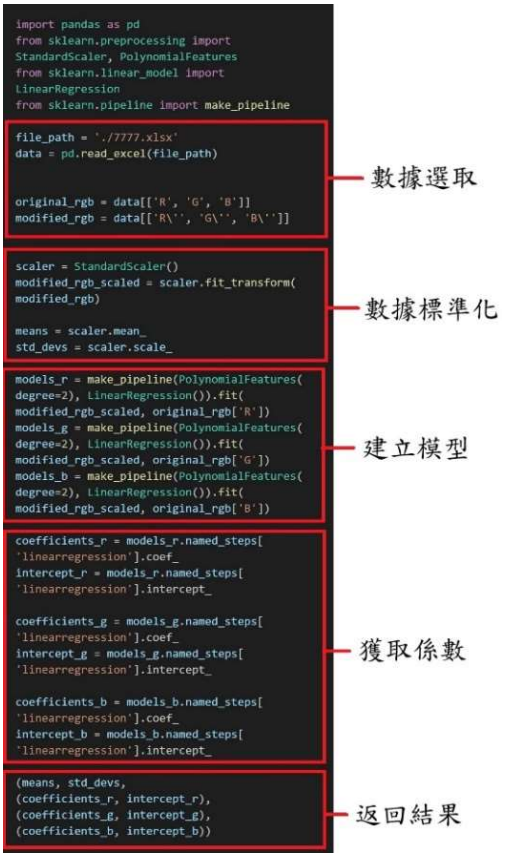
(三) 數據回歸程式

轉換公式應用於數值不相等需要轉換時，只要將數值帶入該公式，其公式的解即為我們要的輸出。這次的案例我們知道其輸入的值，也知道該輸出的值，我們要求其中間的變化關係，而這變化關係即是回歸公式，只要我們找出此公式，檢色晶片檢測到的數值就能被精準得轉換。

我們使用 Python 寫出一組程式碼，對我們統計出的數據做數據回歸，只要把數據輸入 Excel 中並導入公式，程式就能算出該公式的各項係數。其大致步驟如下：

1、 求回歸公式係數

先把 Excel 中的數值導入 VS Code 中用 Python 語法寫成的推導公式，搭配上多種程式庫，其公式功能如下圖 34。



數據選取
從 Excel 中讀取數據，並從原始數據中選取'R'、'G'、'B'和修改後數據中的'R"、'G"、'B"列

數據標準化
使用 StandardScaler 程式庫進行數據標準化，並獲取標準化後的數據－均值、標準差

建立模型
使用多項式特徵轉換和回歸模型構建三組模型，分別對'R'、'G'、'B'進行擬合

獲取係數
獲取三個模型的擬合結果的係數和截距

圖 34 推導程式

2、 數字標準化

數據標準化是一種數據預處理技術，旨在使數據集中的數據具有相似的尺度，以助於提高模型的性能和穩定性。我們將測量數值導入 Python 寫成的程式運算，圖 35 數值標準化 35 為檢色筆內部 ESP-01 中的換算數值 X_0 為紅色；X_1 為綠色；X_3 為藍色，轉換成均值為 0，標準差為 1 的標準正態分佈。這樣的轉換有助於消除數據中的尺度效應，確保不同特徵的值在模型訓練過程中被平等對待。

$X_0=(r-9107.36)/4233.43$
$X_1=(g-11858.98)/4610.62$
$X_3=(b-11433.57)/3948.27$

```
double predictRed(double x0, double x1, double x2) {

    double mean_x0 = 9107.36, mean_x1 = 11858.98, mean_x2 = 11433.57;
    double std_x0 = 4233.43, std_x1 = 4610.62, std_x2 = 3948.27;

    x0 = (x0 - mean_x0) / std_x0;
    x1 = (x1 - mean_x1) / std_x1;
    x2 = (x2 - mean_x2) / std_x2;
```

圖 35 數值標準化

3、 轉換公式

經過方的上述的步驟，我們求出三組轉換公式和標準化數值，並使用在了檢色筆上，圖 36 轉換公式 36 為其中紅色數值的轉換公式。

紅色輸出值 $= 166.18 + 126.23 * X_0 - 44.51 * X_1 - 2.47 * X_2 - 56.7 * X_0^2 + 59.65 * X_0 * X_1 - 9.52 * X_0 * X_2 - 27.72 * X_1^2 + 16.78 * X_1 * X_2 - 4.63 * X_2^2$

```
double predictRed(double x0, double x1, double x2) {

    double mean_x0 = 9107.36, mean_x1 = 11858.98, mean_x2 = 11433.57;
    double std_x0 = 4233.43, std_x1 = 4610.62, std_x2 = 3948.27;

    x0 = (x0 - mean_x0) / std_x0;
    x1 = (x1 - mean_x1) / std_x1;
    x2 = (x2 - mean_x2) / std_x2;

    double intercept = 166.18;
    double coefs[] = {0.0, 126.23, -44.51, -2.47, -56.70, 59.65, -9.52, -27.72, 16.78, -4.63};

    double prediction = intercept +
                        coefs[1] * x0 + coefs[2] * x1 + coefs[3] * x2 +
                        coefs[4] * x0 * x0 + coefs[5] * x0 * x1 + coefs[6] * x0 * x2 +
                        coefs[7] * x1 * x1 + coefs[8] * x1 * x2 + coefs[9] * x2 * x2;
    return prediction;
}
```

圖 36 轉換公式

陸、 討論

一、更大的檢色範圍

RGB（紅色、綠色、藍色）是一種加色光模型，它表達三種色光的強度關係——用 0 到 255 來表示。

理論上，通過變化 R、G、B 的強度，可以混合出多種不同的顏色，包括各種飽和度和亮度的色彩。然而，人眼對顏色的感知是複雜的，而單純的 RGB 模型可能無法完全模擬一些特定的顏色和光學效果。

RGB 顯示的色域有限，因此無法精確表示某些特殊的飽和度和色相，尤其在實現廣色域的顯示或打印時。且 RGB 表示的顏色無法完全符合人眼的感知，在一些精細的圖形和色彩細節上，RGB 會存在色度失真。

為了解決這些問題，我們希望能使用一些更複雜的色彩模型和色彩空間，例如 CMYK（青、洋紅、黃、黑）用於印刷、Lab 色彩空間用於描述人眼感知的色彩等。

二、不同材質的檢色穩定性

經過上述的大數據實驗，我們明顯發現數值在不同物質的測量上，會出現嚴重的偏差，因為檢色晶片是利用強光照射物體，並用光敏二極體接收物體反射的光來讀取數值。所以如果測量物會反光或透光，都會使其失真。

為此，我們希望能想出其他的檢色原理來取代目前的強光反射測量，在現階段我們討論出的最佳做法是把檢色晶片換成高精度攝影機，並把拍攝到的圖片交由 AI 辨識其顏色和輸出數值。

三、更小巧的產品

我們專題的初衷是讓藝術走向人群，並讓人們方便攜帶和隨心所欲的繪畫，但礙於各類晶片的體積，我們的產品始終無法做到符合我們期望的大小。

為此我們希望研發自己的晶片。事實上經過討論檢色晶片不難自製，我們都知道它的原理，搭配上電路板雕刻技術，實現自製更小的晶片應該可行，不然直接更換檢色方法也是值得嘗試的選擇。

柒、 結論

本專題以「檢色」為主題，旨在解決數位藝術創作者在選擇準確顏色上的困難。透過整合硬體製作（3D 繪圖與列印、雷射切割、電路板雕刻）和程式撰寫（Arduino、Python、JavaScript），開發出一枝能檢測現實中顏色並將其轉成 RGB 數值，進而即時應用於電繪軟體的觸控筆。

硬體製作涵蓋 3D 列印的外殼、雷射切割的筆架、電路板設計等技術，並透過 Arduino 控制晶片、Python 製成插件和數據回歸程式、JavaScript 架設伺服器和網站來實現各項功能。另外，自製插件使檢色筆兼容兩款開源繪畫軟體 Blender 和 Inkscape，提供更豐富的應用場所。

主要功能為顏色檢測晶片 TCS34725 進行測量，並透過 ESP-01 晶片中自創的轉換函數進行數值轉換後由網路傳輸，經過伺服器的中繼，傳到用戶的軟體插件中。

研究時，我們進行了大量數據收集、硬體製作、軟體開發和測試，包括搭建伺服器、研發插件、設計展示架和收納盒等相關系統和配件。此外，透過 2000 多組大數據的分析並配合數據回歸技術得到一組能轉換數值的函式，使其在極端狀態下的數值變化也能穩定，優化了檢色晶片的讀取數值轉換成 RGB 值的能力，提升了檢測精準度。

未來會朝向，擴大檢色範圍、提高檢測穩定性、探索更小巧的產品和檢色晶片設計。這些方向將進一步優化產品性能，提高使用者體驗，符合專題初衷，拉近藝術和大眾的距離。

這個專題結合了藝術和科技，成功地開發了一款具有檢色功能的觸控筆，為數位藝術創作者提供更直觀、準確的顏色操作體驗，同時展望未來有進一步的發展和改進空間。

捌、 參考資料和其他

書面資料

1.陳佑瑄（2024/01/18）。Vue.js 3 前端開發不踩雷：Composition API×Vue Router×Pinia，帶你快速升級進階開發者！。博碩文化股份有限公司。
2.王玉樹（2023/08）。Raspberry Pi 最佳入門與應用(Python)。全華圖書股份有限公司。
3.任鏡翔（2020/09/21）。進階程式設計-使用 Python、C++(電子書)。Qi feng zi xun gu fen you xian gong si。
4.Tony Lu （2017/07/26）。Vue.js 漸進式 JavaScript 框架入門。TonyCube。
5.陳會安 （2017/08/31）。JavaScript+jQuery +Node.js 網頁設計與物聯網應用開發教本(電子書)。碁峰資訊。
6.謝邦昌宋龍華李紹綸（2017/09/22）。大數據分析 SQL Server 2016 與 R 全方位應用(電子書)。碁峰資訊。
7.謝邦昌鄭宇庭宋龍華陳妙華（2019/07/22）。大數據分析 Excel Power BI 全方位應用(第三版)(電子書)。Qi feng zi xun gu fen you xian gong si。
8.洪煌佳（2022/04/07）。Python 論文數據統計分析。Wu nan tu shu chu ban gu fen you xian gong si。

網路資料

1.深圳色彩管理有限公司（2023）。立邦漆色查詢頁電子版 1988 色。千通色彩管理。https://www.qtccolor.com/secaiku/dir/28
2.Jason Chu（2020/04/29）。用 ESP-01 進行 ESP8266 開發，完全攻略。傑森創工。https://blog.jmaker.com.tw/esp8266-esp01/
3.Python Software Foundation (2001-2024)。Beginner's Guide to Python。Python。https://www.python.org/
4.TP-Link（2023/07/25）。如何將獨立模式下的交換器設定為 DHCP 伺服器？。TP-Link。https://www.tp-link.com/tw/support/faq/3656/
5.OpenAI (2015-2014)。Color Update HTTP Request / Excel XY 散點折線.。ChatGPT。https://chat.openai.com/
6.Blender（2024/01/09）。Blender 4.0 Python API Documentation。Blender 。https://docs.blender.org/api/current/index.html
7.Inkscape（2024）。How to Write Extensions。Inkscape。https://inkscape.org/develop/extensions/