

嵌入式實驗期末報告：智能溫室

B08901001 樊樟

B08901038 施泓羽

1. 動機：

因為目前有在家中陽台種植各類植物，但可能比較忙碌的時候就會疏於照顧，且因為種植在戶外，所以有大風大雨的話很常會枯萎死掉。所以我們才打算做一個能夠抗風與且能夠自動控制澆水、照明的智能溫室。

2. 功能與做法：

server:

i. 功能:

server 是用來接收 stm32 及 rpi 傳過來的資訊，除了將傳來的資料繪製在 UI 上外，也可以主動控制燈光及風扇。

ii. 實作:

我們是用 python 中的 socket、select 這兩個 library 做出一個可以同時監聽多個 client 的 socket server，然後將 stm32 及 rpi 用 socket 連上 server 傳遞跟接收資料。

而 UI 跟繪圖的部分則是用 tkinter、matplotlib 來完成。因為 socket server、UI、繪圖三者要一起執行，但三者都是 blocking 的，所以還有用到 threading 這個 library。

控制澆水:

i. 材料:

抽水馬達*1、一路繼電器*1、土壤溼度感測器*1、塑膠軟管*1

ii. 功能:

以土壤溼度感測器讀取土壤中的濕度，並以此判斷是否要啟動抽水馬達進行澆水。

iii. 實作:

先以程式判斷感測器偵測到的溼度有沒有超過門檻(0.3)，再以 RPi 的 gpio 控制繼電器的開關，讓抽水馬達進行澆水。

控制燈光:

i. 材料:

白色 led * 12、適當長度麵包版 * 2、杜邦線 * 4

ii. 實作:

在硬體上，我們是拿麵包版側邊兩條來並聯燈泡，每個麵包版上有6顆led，並用雙面膠黏於容器頂部。控制部分則是用stm32的pwm功能，為了產生改變亮度的功能，我們是將週期設為0.01，然後再根據server傳過來的值去改變亮度，假如收到亮度l, $0 \leq l \leq 99$ ，則將dutycycle設為 $101*l / 10000$ ，藉此來增加其亮度的改變幅度。

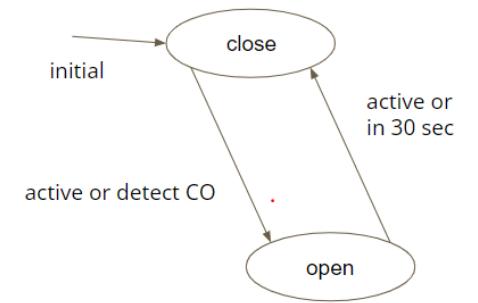
控制風扇:

i. 材料:

12V電腦風扇、一路繼電器 * 1

ii . 實作:

我們將容器側面開一個大小與風扇差不多的洞，並將風扇裝在該處並向外吹風。我們會透過主動或是被動偵測到一氧化碳的方式啟動它，當它打開後，為了避免被動打開後無法關閉，我們將其設定為只要開啟30秒，server就會傳一個要風扇關閉的訊息給stm32。



偵測濕度:

i. 材料:

土壤溼度感測器*1、MCP3008*1

ii. 功能

偵測土壤濕度，並傳到電腦端以供監測

iii. 實作

使用土壤溼度感測器偵測土壤的濕度，用MCP3008轉成數位訊號後，以RPi的gpio讀進來，再使用socket將數值從RPi(client)回傳到電腦(server)，再使用電腦進行監測

偵測水位：

i. 材料

土壤溼度感測器*1、MCP3008*1、抽水馬達*1、塑膠軟管*1

ii. 功能

偵測集水槽水位高度，傳回電腦進行監測，並以程式判斷是否要啟動抽水馬達進行廢水回收

iii. 實作

以土壤溼度感測器(因為水位感測器解析度太低)偵測集水槽水位高度，用MCP3008轉成數位訊號後，以RPi的gpio讀進來，再使用socket將數值從RPi(client)回傳到電腦(server)，再使用電腦進行監測，並在RPi內以程式判斷是否要接通繼電器，啟動抽水馬達進行抽水

偵測空氣：

i. 材料

CO感測器*1、MCP3008*1

ii. 功能

偵測容器內CO濃度，並回傳給電腦進行監測和風扇控制

iii. 實作

使用CO感測器偵測容器內的CO濃度，用MCP3008轉成數位訊號後，以RPi的gpio讀進來，再使用socket將數值從RPi(client)回傳到電腦(server)，使用電腦進行監測，並進行風扇的控制

容器製作：

i. 材料

壓克力板*1、熱熔膠與矽膠

ii. 實作

以雷射切割切出外殼並組裝，使用熱熔膠與矽膠堵住縫隙進行防漏

3.遇到問題與解決：

1.漏水

使用壓克力板作為容器，水很容易就從縫隙滲出，我們一開始使用熱熔膠進行防漏，但效果不彰，後來我們使用了矽膠進行第二次防漏處理，結果好了很多，但還是有微微的滲漏，我們認為我們缺乏足夠的技術完全解決這個問題。

2.水壓問題

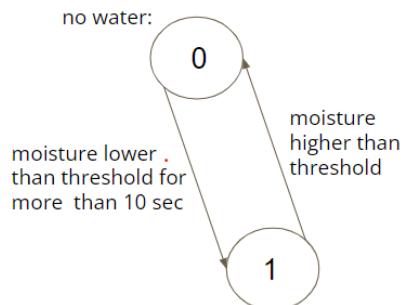
再澆水時，因為水槽的位置較高，水會因為水壓的差距不斷從水管流出，無法停止，我們後來在水管的頂端戳了一個洞，讓空氣能夠進入，解決了這個問題。

3.澆水不平均

如果只使用一條直的水管很容易造成澆水不平均的問題，所以我們將水管繞成了一個圈並固定住，放在花盆的開口端，並在內側戳許多洞，用熱溶膠將管口堵小，這樣出水就會比較均勻。

4.水槽缺水問題

我們做完之後發現，我們的儲水槽內沒有感測器，這樣如果儲水槽內沒有水，馬達就會一直空轉不會停，後來我們使用程式去判斷，如果抽水一段時間後土壤濕度沒有增加，就代表缺水，將強制停止馬達轉動，成功解決這個問題。



5.RPi重啟問題

我們發現RPi常常會毫無理由的重啟，後來發現是因為模組裸露的腳位造成短路，使用絕緣膠帶做好絕緣後解決。

6.土壤感測器氧化問題

土壤感測器因為長時間在水中，金屬電極部分很容易氧化，造成感測器失靈，且因為每次實驗後並未將其擦拭，才導致其問題。因為遇到此問題時已經將該感測器黏死，難以替換，因此為了Demo我們將感應的高度標準降低。

7.各式空氣感測器難以使用

我們一開始是打算偵測空氣中多種有害氣體，因此我們有試過CO₂、CO感測器、MQ135等等，但因為這些感測器都要先預熱一段很長的時間才能感測到較穩定的數值外，也必須要透過準確濃度的氣體來進行校正，因為那些有害氣體難以取得且預熱時間過長，所以我們只拿了CO感測器來實作。

為了解決需要校正的問題，我們是先觀察在一般狀況下讀取到的數值，然後再透過點香煙產生的少許CO去觀察讀取到的數值去設定風扇開啟的threshold。

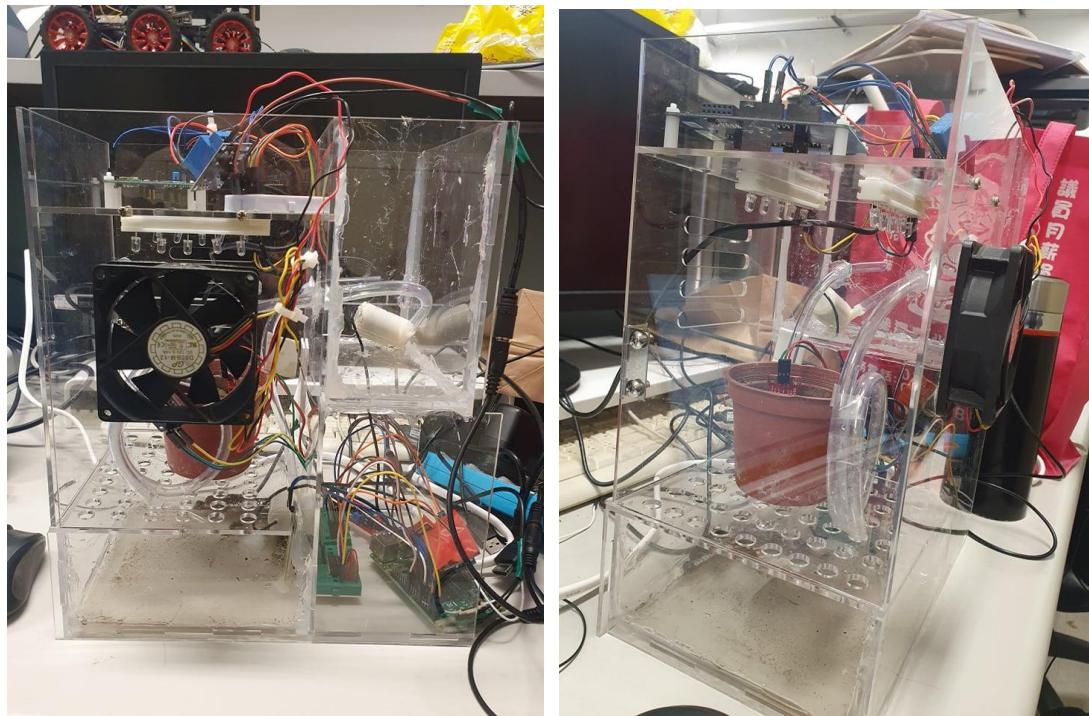
8. server、UI、繪圖難以同步進行

我們一開始在實做時，是直接在UI中開一個socket server，但當server打開後，卻發現UI當掉無法繼續執行，後來在網路上就查到應該要用mutithread的方式執行，所以我們便讓main thread執行UI，打開UI後便另外開一個thread讓server執行，而畫圖則是每次收到資料後就開起一個thread畫圖。

9. server不穩定

一開始我們是打算開一個server去接收所有client的request，但因為我們只想要傳給stm32，而又難以判斷目前是誰寄的request，而且如果有一個client斷線後，server無法偵測到並且持續傳資料的話就會產生error，使整個server當掉。所以我們後來是採用select這個library來處理，select可以自動偵測目前所有socket object是否有動作，像是有其他裝置連線到server、辨認那些client發request，並且可以很好處理斷線問題讓整個server更穩定的運作。

4. 成果及Demo影片



1. [嵌入式系統實驗 期末專題 Demo影片1 B08901038施泓羽 B08901001樊樺 - YouTube](#)
2. [嵌入式系統實驗 期末專題 Demo影片2 B08901038施泓羽 B08901001樊樺 - YouTube](#)

5. 參考資料

[multithreading - Python/socket/threading/tkinter: How to know the message sender? - Stack Overflow](#)