Embedded System Design

Final Project Report

PID Balance Ball and Beam

106030009 葉蓁

一張含有 牆, 室內 的圖片

自動產生的描述

1. Goal

我製作一個一維的平衡球和支架，目的是：

平衡樑要能夠讓球停在指定的setpoint。

Operating Philosophy:

Distance sensor get ball position(Input) -> Tell Arduino and compute the PID value-> Tell servo motor to move and try to stabilize the ball.(Output)

My design flow:

一張含有 文字 的圖片

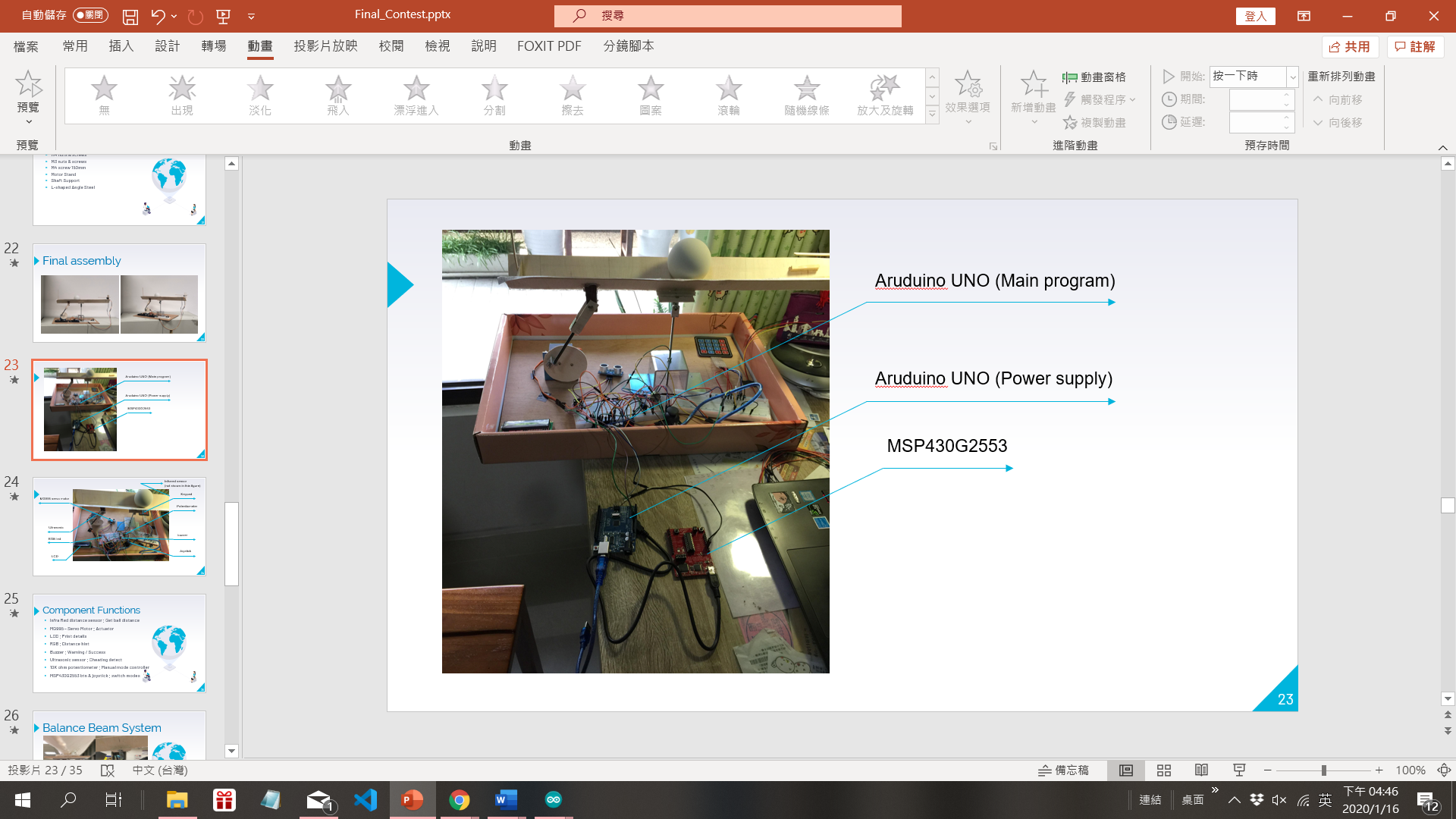
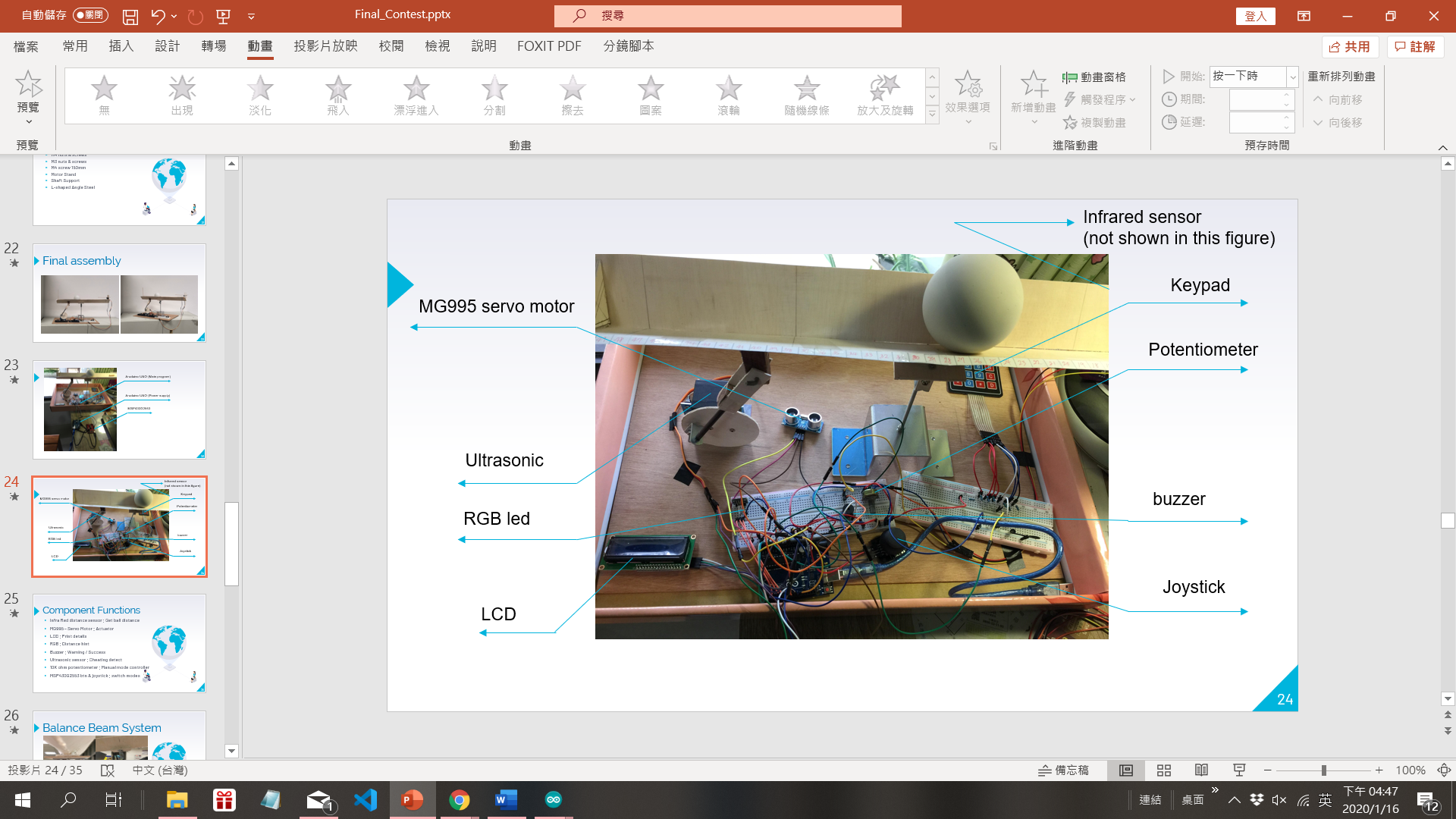
自動產生的描述一張含有 螢幕擷取畫面 的圖片

自動產生的描述

簡單解釋一下：受控場(plant)是平衡架，由伺服馬達(actuator)驅動它，得到的output是球的位置(Ball Position)。藉由一個IR sensor去讀取output，然後回授到input固定的位置(set point)，兩者相減得到一個誤差值。這個誤差值如果最簡單的控制方法，就是用一個單純的P gain去放大然後送給驅動器，讓他去追命令，減少誤差。這種只有一個放大倍率的控制叫做P control。但這種簡易的控制器有它的限制，可能會有太多的過衝量(overshoot)，或是因為system type是0的原因，即使input只是一個step還是會有穩態誤差。所以我們想用PID control，就是PD加上PI，D是Differential微分，將誤差微分後乘以比例係數(ki)，再加上I是Integral積分，最後送給驅動器。

PID三個項的目的：以我們project為例：P 能夠讓我們的Ball去趨近中間setpoint的位置，D能夠考慮球的速度(位置微分)改變馬達輸出的大小，減少球過衝停不下來的問題，而I則是避免球卡再一個很靠近(誤差3-5公分卻停住不動)但錯的位置的窘境。

1. Component Used (Schematics)

1. Component:
2. Electrical components.

Sensors :

SHARP GP2Y0A21 YK0F infrared distance sensor(10-80cm)

Joysticks

10KOhm potentiometer

Actuators:

MG995 Servo motor

Liquid crystal display

DAC:

RGB led light

Buzzer

MCU:

Arduino UNO R3 \*2 (one for program, one for power supply for servo)

MSP430G2553S \*1

1. Mechanical components.

Several M3, M4 screws and nuts.

一些鋁的沖孔版(支架用)

碳纖棒

木片、木頭少許

冰棒棍

膠帶、電火布、熱熔槍、焊錫

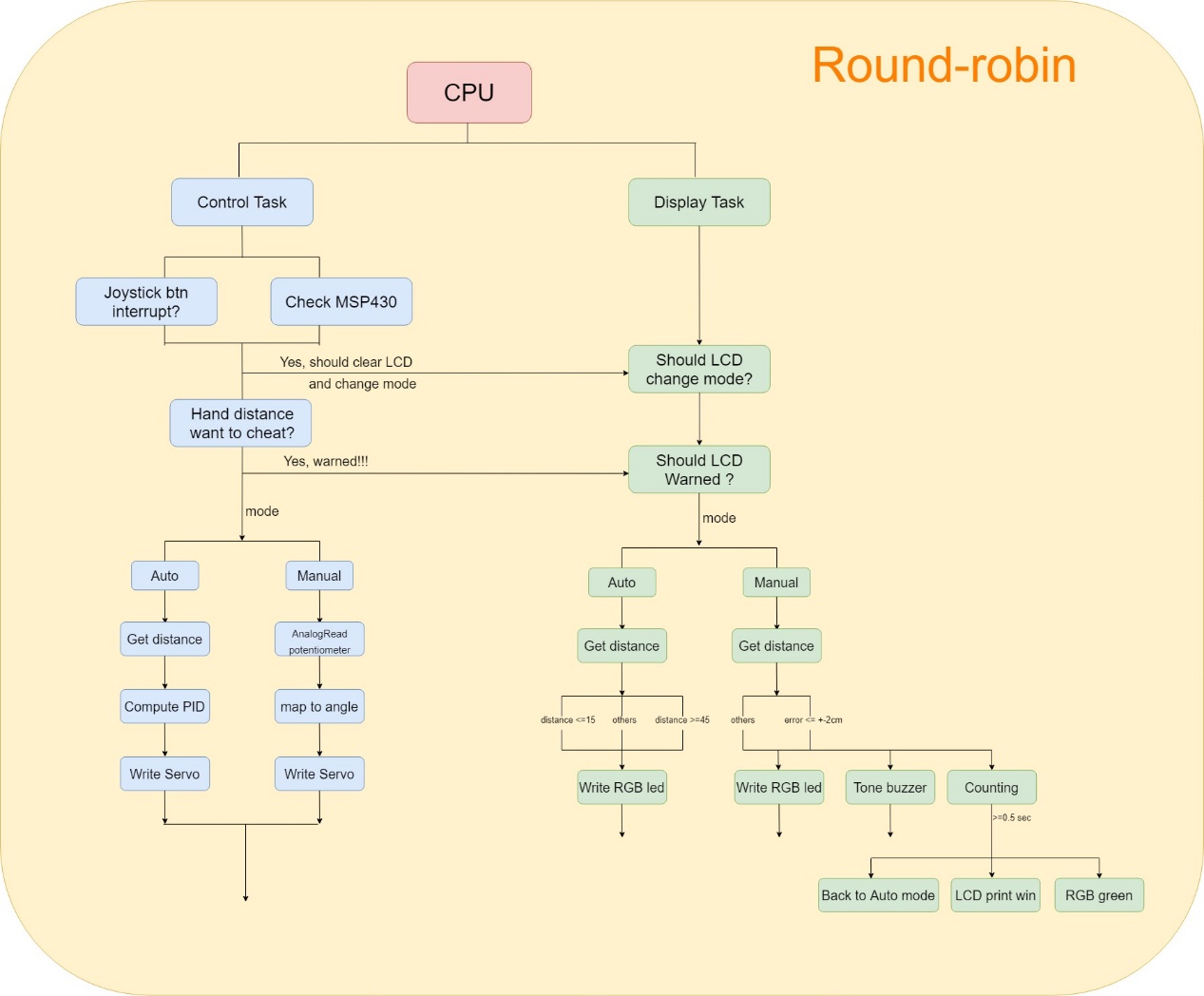


1. Schematics: (How I connect everything with Arduino board)

一張含有 文字, 地圖 的圖片

自動產生的描述

1. Program Flow



我用FreeRTOS寫，宣告兩個Task,分別是controlTask跟displayTask。任何跟控制Beam相關的程式都寫在於controlTask裡面，跟display相關的都放到displayTask裡面。兩個Task的priority都宣告成1，CPU排程是用Round-robin scheduling。記憶體分配分別是160跟128byte。

(1)Control Task : 每次進來這個Task，都會先跑兩個function：

第一個function是確認MSP430有沒有按按鈕，如果有按會換mode。

第二個function是讀取距離超音波感測器的距離，如果手很靠近(想作弊)

就會出現Warning的訊息，叫你不可以作弊(buzzer會響)。

跑完這兩個function後，有個if\_else判斷進哪個mode。所以我

#define Auto 0

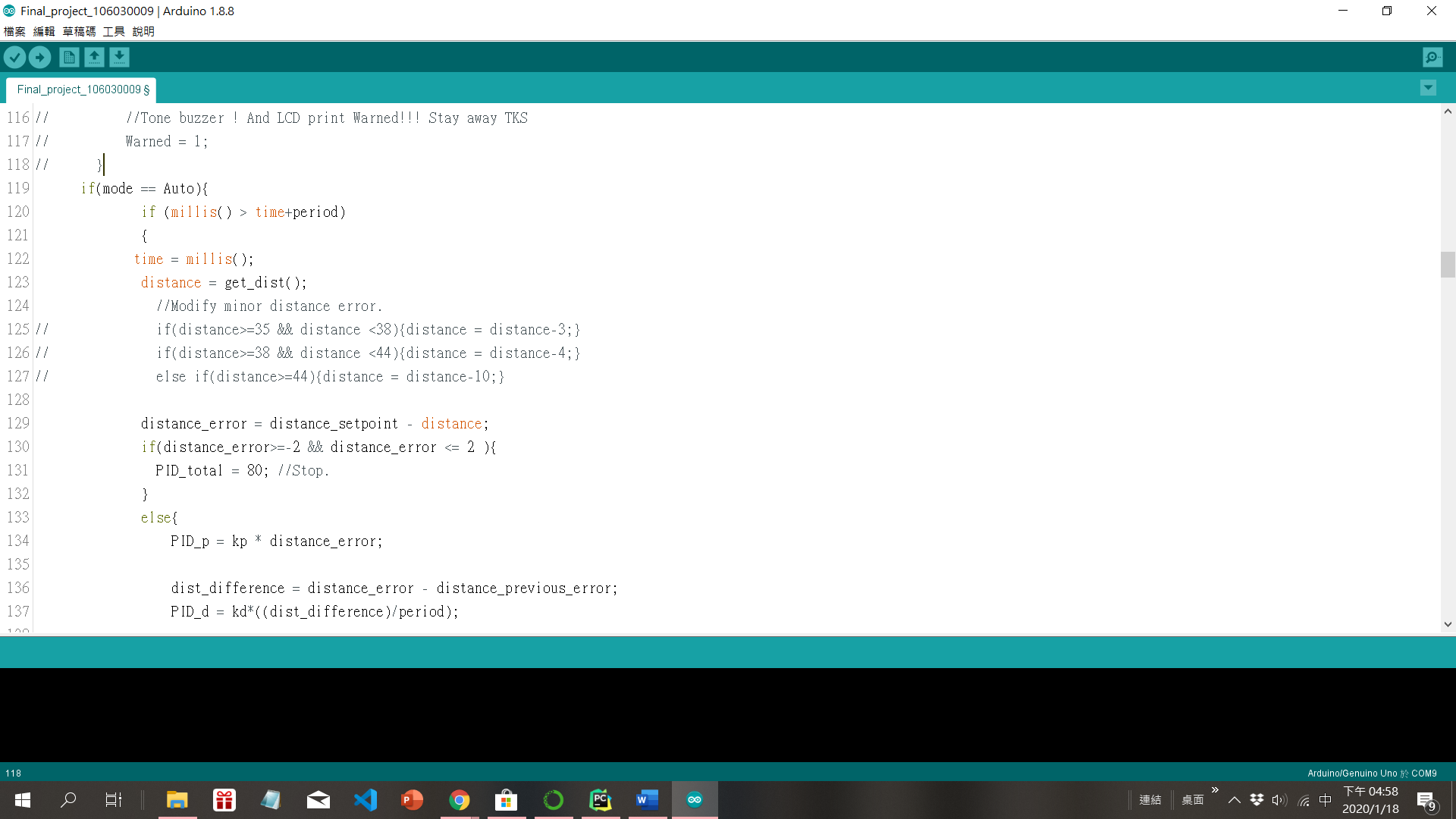
#define Manual 1

如果是自動(Auto)模式，平橫樑會自己依照球的誤差去平衡，使球回到setpoint。

如果是手動(Manual)模式，使用者用10K歐姆的可變電阻去平衡球的位置。如果在可接受誤差內(正負2cm)維持超過一定秒數，機器會告訴使用者你成功了，然後回到自動模式。

至於如何切換模式，有兩種方法，可以按Joystick的sw，也可以按MSP430的Button。可以各別按一下回到本來的mode。E.x.: 預設Auto mode，按一下Msp進入Manual mode，再按一下Joystick變回Auto mode。兩個按鈕不會衝突。Joystick的btn偵測是用Interrupt寫的。

說明PID實作方式：附上部分code截圖：



如果是Auto mode，準備要進來平衡。當超過sample的週期(period我設50ms)重新跑一次流程：

1. 取得距離(get\_dist)
2. 可能要修正距離(因為我的sensor在大於30公分後會有不定的誤差，我用人工修正方法讓他更貼近真實距離
3. 誤差 = SetPoint – distance
4. 如果誤差小於正負2公分，視為平衡，讓馬達維持在水平靜止
5. 如果誤差大於正負2公分：計算三個PID值：

P : Kp \* error

D : Kd \* (error – last\_error)/period

I : 如果距離setpoint小於10公分或大於5公分(這部分要由做實驗得到你球跟馬達配合的結果來決定)，讓Integral項作用。

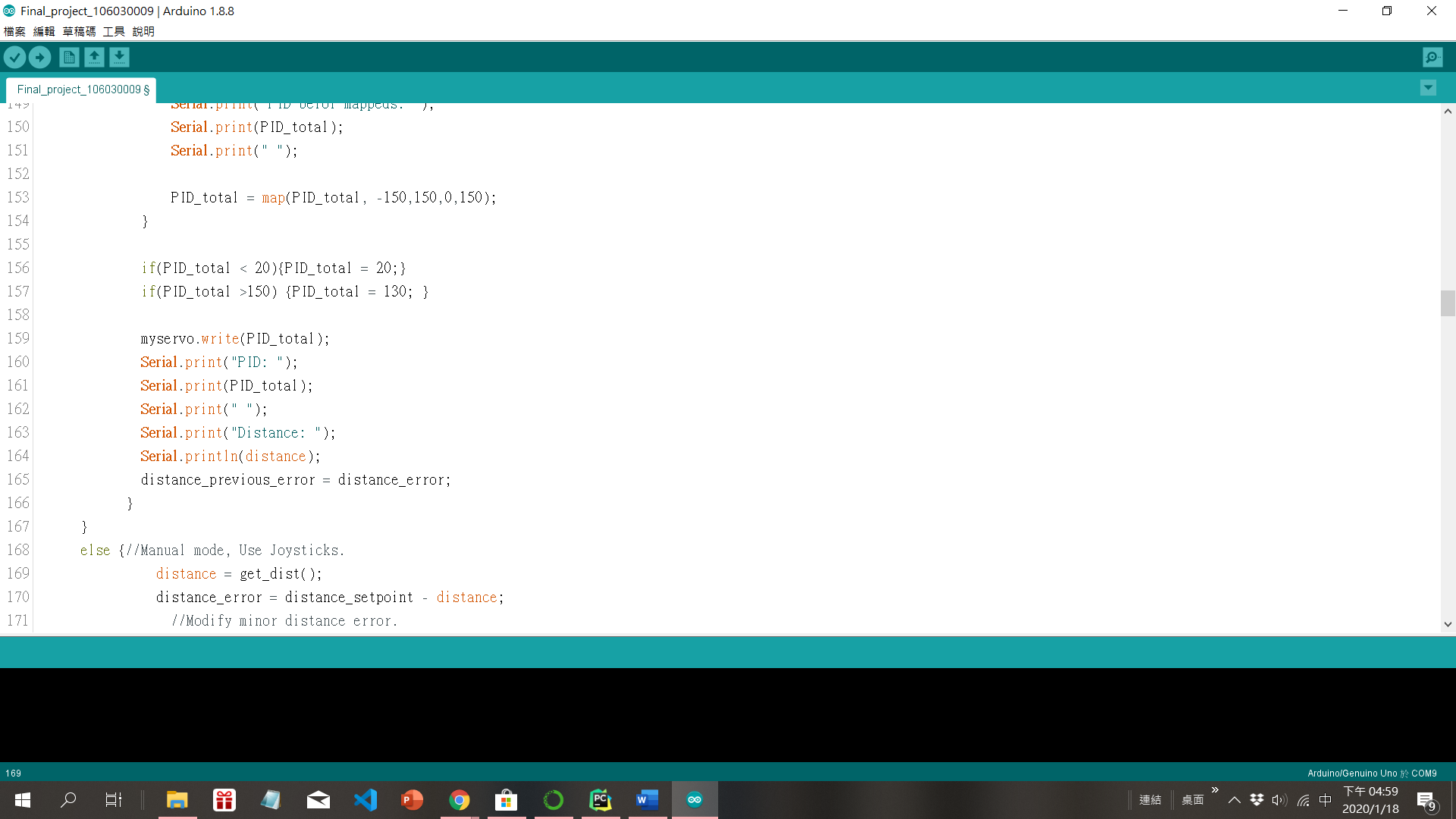
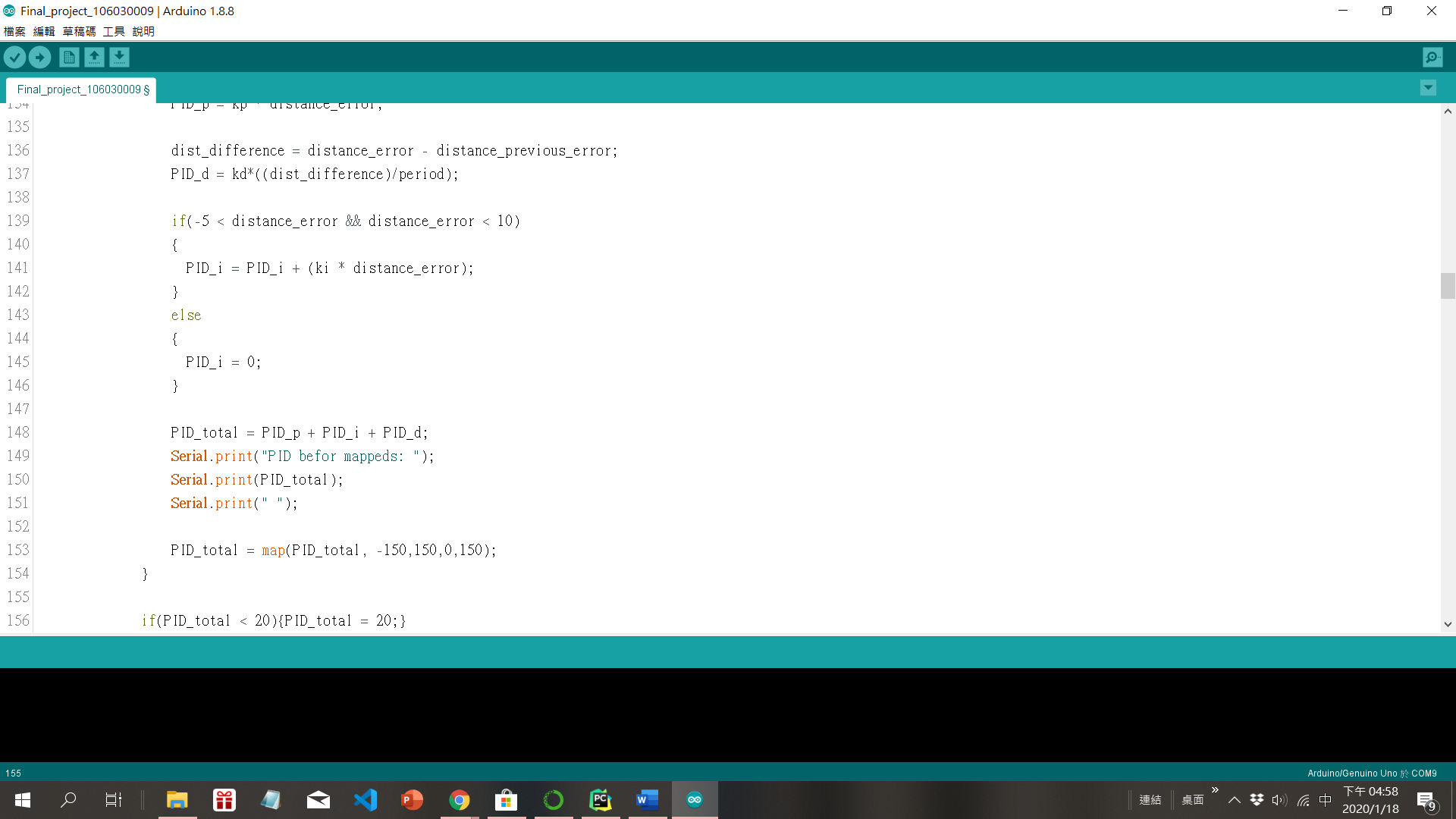
Ki \* error + 上一次的I term。(積分效果)

否則I項為零。

1. 將三項加起來得到總PID\_total
2. 將有效的PID值mapping到馬達轉的角度(0~150度)

Mapping這邊也要靠trial and errror來嘗試，最為困難。

1. 剔除極端值後，讓馬達轉
2. 更新上次的error為現在的error，準備下一次的sampling。



1. Display Task :

任何跟顯示有關的(LCD, RGB,buzzer)都用這個Task控制。以下分用這三個元件說明：

***LCD*** : 用mode分要印甚麼。第一行會印是甚麼mode(Auto mode : )，第 二行會印距離幾公分(e.x. 22 cm)。如果有換mode的話，才會把lcd.clear()，第一行會洗掉重印，這樣才不會太頻繁的clear導致螢幕閃爍。

另外如果有cheat行為，Lcd會警告，印出”Warning! Do not Cheat”來告訴使用者。在Manual mode如果你進入平衡範圍，他會告訴你”You balance”，成功平衡後會印出”Congratulation”，然後回到Auto mode。

***RGB*** : 用mode來分，

Auto mode :

球大於45公分的話，會亮紅燈；小於15公分的話，會熄滅。

中間距離想用成比例關係亮燈，如下：

Manual mode:

球在balance 區內會亮藍燈，這個區間以外則不亮燈。

***Buzzer*** : 不論在哪個mode，如果超音波偵測到人手想要cheat，buzzer會教出高頻的警告聲。

另外在Manual mode，如果玩家讓球停在平衡區範圍內，buzzer會發區低頻聲表示現在是在平衡區，超過指定時間，會發出一段旋律表示你成功了! 然後又回到Auto mode。

兩個Task寫好之後，loop沒有東西，記憶體調整一下，原本我都只给128，結果controlTask不太夠，所以調大一點，就可以了!

1. Difficulties

我整理以下幾個困難點：

1. 機構設計：這個機構最困難的地方是：如何將伺服馬達旋轉的運動，轉為直線運動，再轉成推動蹺蹺板(木軌道)的運動。這會需要一個多連桿系統。從側面看過去，碳纖棒盡量要垂直於木片，才不會有側向剪應力，造成應力集中機構毀損，或是馬達做無效的力量傳送。
2. 調整PID參數：PID有三個最重要的參數：KP、KI跟KD。調整的順序為，先調KP，再加KD，最後加KI。但除了這三個參數之外，計算後的PID\_total如何mapping到讓伺服馬達動的Servo.write()裡面也是一大挑戰。要先觀察自己馬達尚未mapping前的初始值，PID\_total的範圍是多少，剔除極端值後，選出有效範圍，然後做mapping。Arduino很方便，有內建的map API可以叫，不過你的Domain和Range要設有效範圍，才會讓平衡裝置成功。我光調整這個mapping就花了不少時間。調完還要回去看KP, KI, KD三個值可不可以，頗花心力的。
3. 讀取距離的精準度：由於這是一個一維的平衡裝置，單靠距離就可以做到平衡，但也因為這樣距離的精準度變得相當重要。原本我是用常見的超音波感測器，但對於球狀構造，波可能會繞射或產生其他非預期行為，我後來查了資料發現紅外線比較精準，但解析度沒有超音波那麼好(只能讀到cm等級)。有時候如果接地線鬆開，量測距離的sensor會瞬間壞掉(把50多公分顯示成9公分等)讓我非常緊張。這是我實驗過非常多次後得到的結論。未來如果這個project要做的更強健，可能不能用杜邦線，要直接焊接以減少接觸不良的問題。

p.s.這個紅外線模組初始沒有杜邦接頭，也是我自己焊接的XD生平第一次焊電線QQ

5. Discuss & Feedback

關於我的project我想到了三個問題：

* 1. 可以動態(不用改code)就改變setpoint位置嗎?

我有想到用keypad去寫入，但不曉得因為我用FreeRTOS的方式寫，memory會有不夠的問題，導致這個目前還沒做出來。如果用mega或更大記憶體的板子，也許可以成功。

* 1. 可以展延成2D的平衡板嗎？

是可以，但一來機構上會變得很複雜(二維彼此不能干涉)，軟體上也變得相當複雜，因為首先：Sensor不只是用測距離就可以解決，必須用三軸陀螺儀去偵測，這個模組的code應該也相當複雜，需要一些時間研究。不過既然網路上查的到相關成功的影片，表示一定做得出來。只是我時間跟經費不夠。

* 1. 可以用更好的控制方法嗎？

如果我得到精準的Transfer function，也許我可以嘗試用lead-lag控制方式。不過這目前超出我的能力範圍XD期許未來自己有機會可以實現更好的控制器。

這學期辛苦各位助教了，我真的感謝無比，把我手把手從超級廢帶起來，沒想到自己可以修完這門課! 雖然現在還是很廢，期許自己可以帶著嵌入式學到的Coding Style和Debug method在未來任何地方解決問題! 謝謝助教，辛苦了!