

工科專論期末報告

組員: 姓名學號

張致瑜 左一
B05505011 李原祚 左二
盧廷偉 右二
邊鈞天 右一

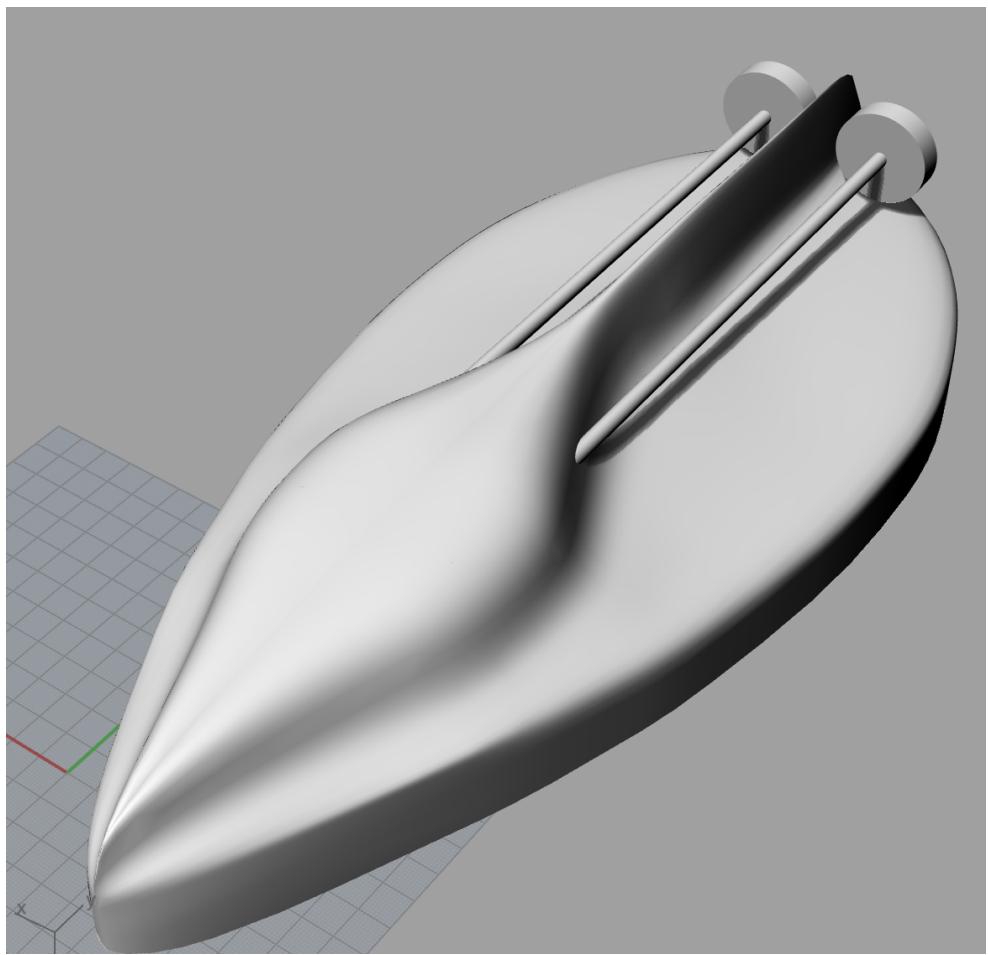
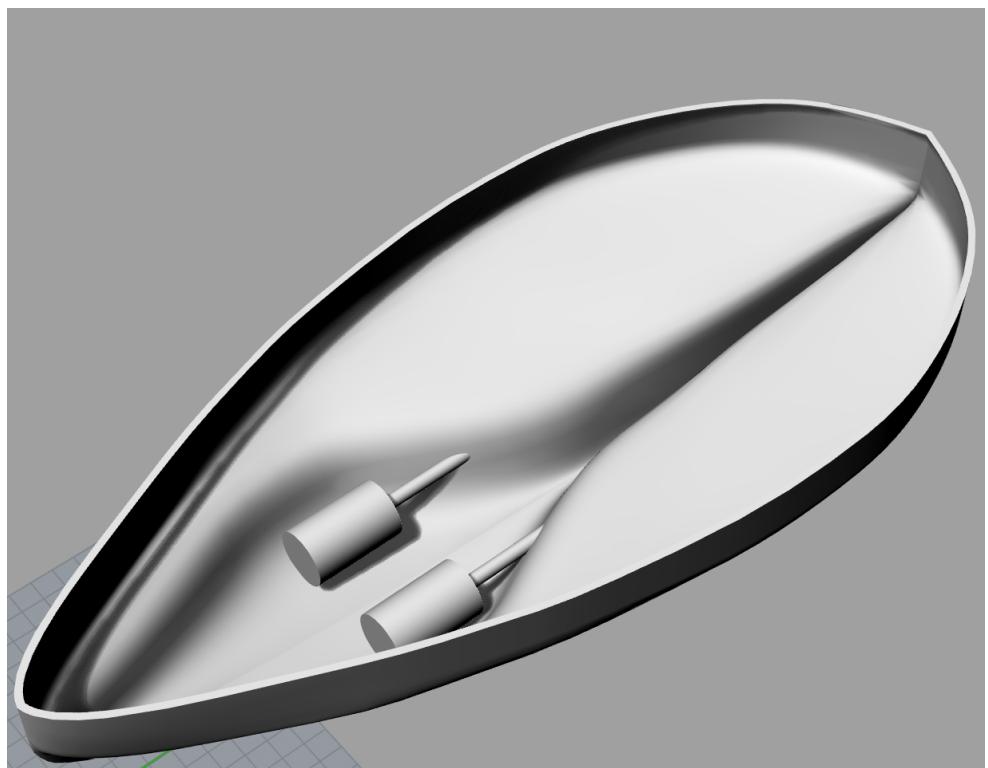


1. 分工表:

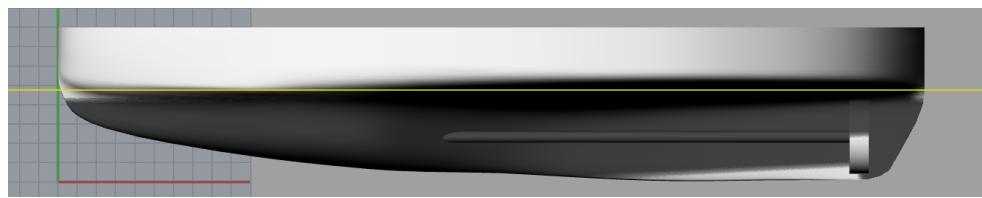
李原祚: 船體設計、馬達裝配、以及最後的參數調整(PWM和閾值)
邊鈞天: 演算法設計、程式實作編排
張致瑜: 船體元件配置、(多段轉彎)演算法設計、馬達螺槳調校
盧廷偉: 演算法設計、船體空間擺放設計，馬達螺槳調校

2. 船體設計:

1.Rhino的設計圖(包含馬達配置)



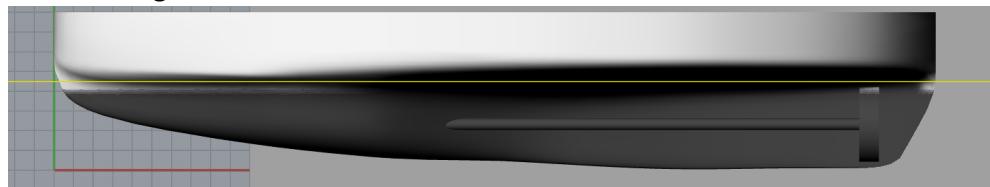
水線面700g :



吃水47.5mm

排水體積704051. mm^3

水線面600g :

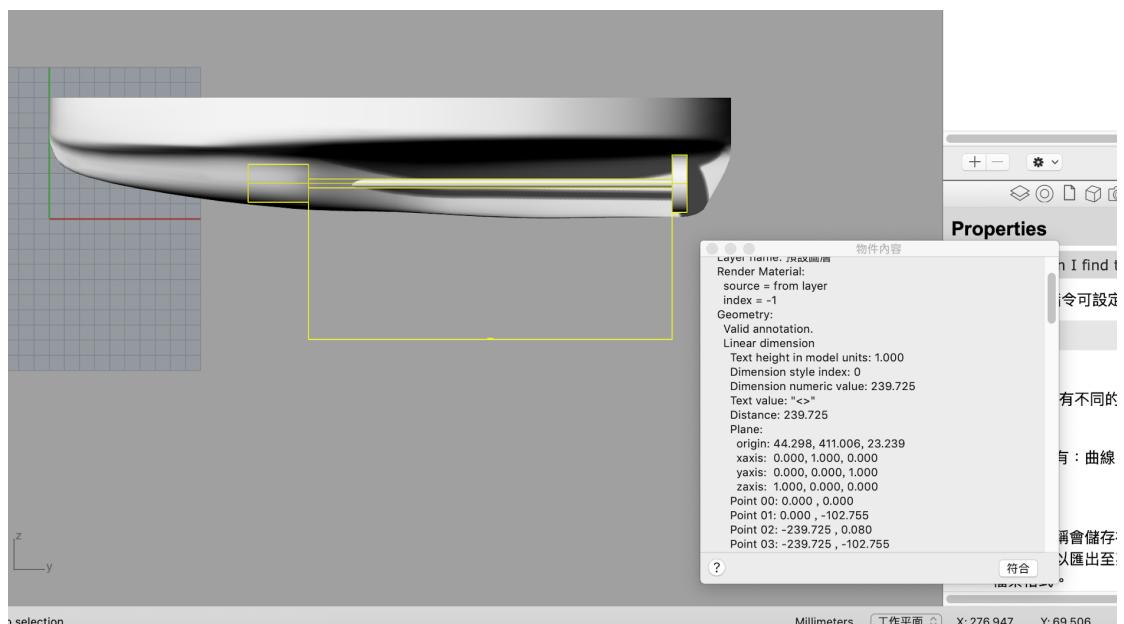


吃水45mm

排水體積592107 mm^3

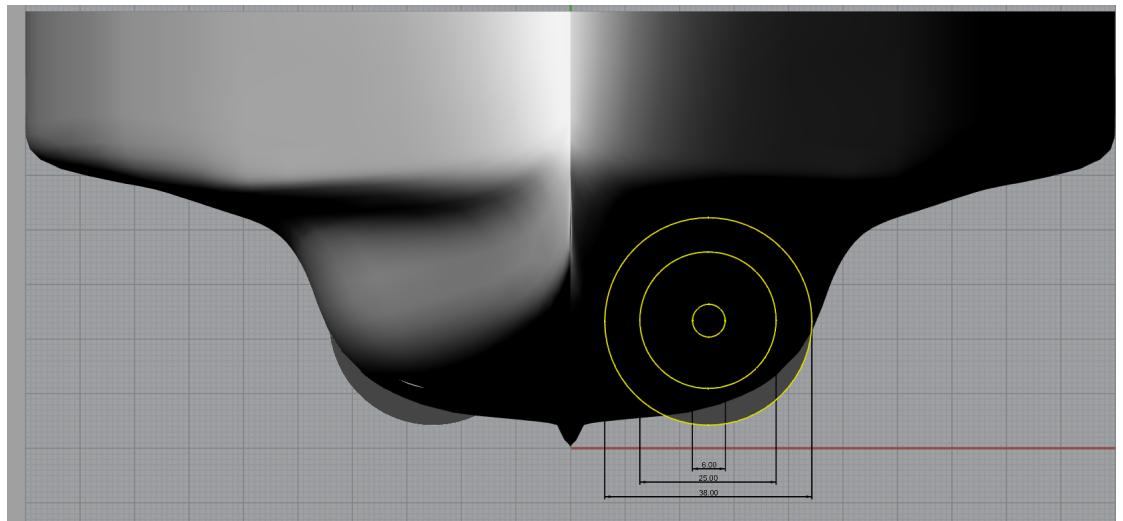
2.三視圖

船模側視圖：



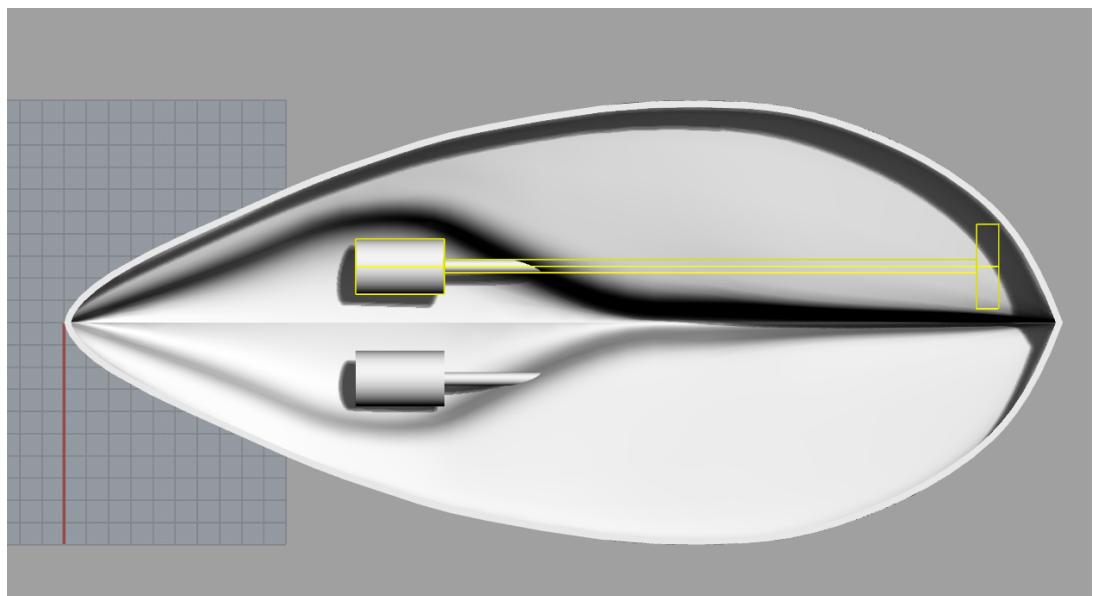
此模型中螺桿的長度為239.725mm

船模正視圖：



此模型中馬達的直徑為25cm, 螺桿的直徑為6cm, 螺槳的直徑為38cm

船模俯視圖:



3.解釋為何要這樣設計船體, 如何去考慮馬達配置

此船型設計首要考慮的重點是當排水體積達 $600cm^3$ ，螺槳須完整沒入水中，這樣的規定是當初文件所要求的。螺槳也完整在船體之中，受到船殼保護。於一般正常情況下，即使擱淺，螺槳均可受到船殼保護，免除碰撞。

當初有詢問重量 $700g$ (原本是 $600g$)是重量下限、上限還是規定要剛好 $700g$ ，得到的回答都是要剛好 $700g$ 。因此在船型設計方面盡量縮小水下體積，以達到排水體積 $700cm^3$ 時水線可達 $4cm$ 以上，使螺槳完整沒入水中。為此原因本組的船型設計受到相當的限縮。不過本組測量船重時，數值為 $701g$ ，確實達成當初船型設計的目標。

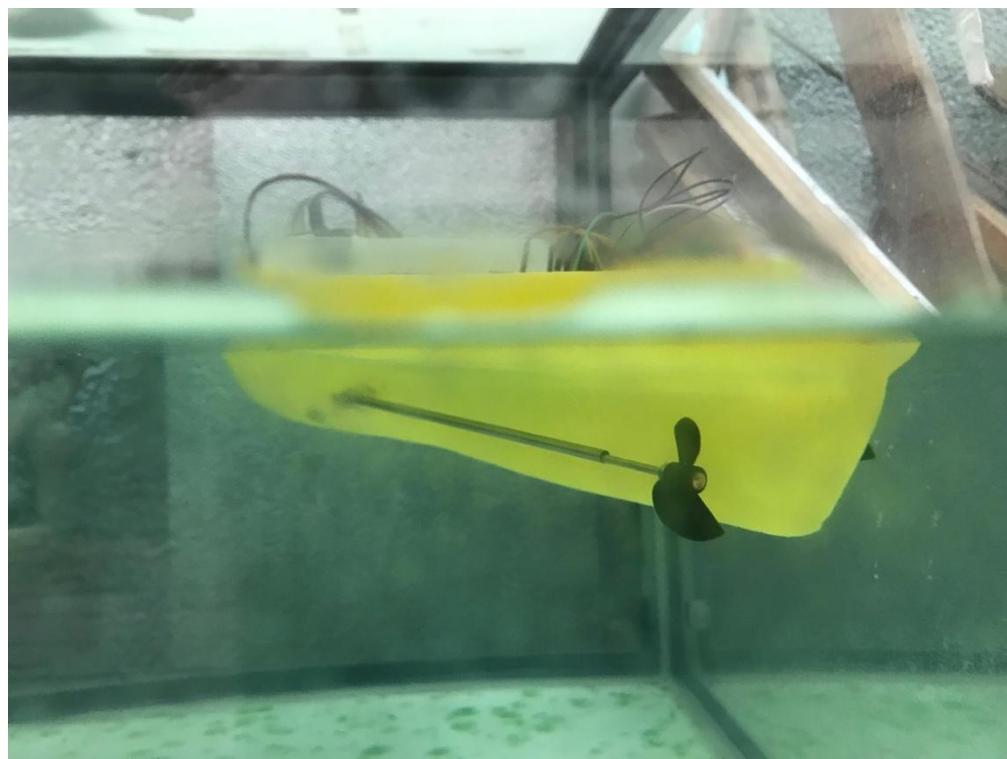
此船型設計，在如此嚴苛的條件下，依然可以提供足夠空間裝配馬達與螺槳。本船體有非常大的儲備浮力，也可以迅速修正船隻的rolling，使船於轉彎時，不易傾覆，也可在同學於水池邊造浪時，穩定航行。

至於坡道挑戰，當初是預計以船底之細長龍骨滑過坡面，因其接觸面積非常小，摩擦力理應也非常小，且此結構可以有效保護螺槳避免與坡道撞擊，損害到螺槳。然獲得規格後，坡道將高於水面5公分，將此數據帶入簡單的動能位能公式 $\frac{1}{2}mv^2 = mgh$ ，其中 $h \approx 0.08$, $g = 9.98$ (若船要越過此坡道，船的重心需被提高坡道高度加上船的重心到船底的距離，在此保守假設為3公分)。解得速度 v 須達 $1.264 m/s$ ，也就是就算沒有摩擦力耗損，船速須達126公分每秒，在我們測試馬達之後，及果斷放棄此項目標。雖然後來坡道降到在水面以下，使得某些船隻得以越過。大部分越過的船隻都有一共同特徵，那就是船頭重船尾輕，只要船頭越過之後，船體即不會與坡道有接觸(除螺槳)，且螺槳必定與坡道產生重大撞擊，與我們當初選擇保護螺槳的設計構想完全相反，導致我們無法以此技巧順利越過。螺槳也比我們預想的耐撞許多。

船底的狹長結構物設計除了應用在坡道以外，我們也預期此結構可以隔開左右螺槳，避免操作時水流的互相干擾，藉此提升轉彎時的表現。於水上實作時，確實也是我們船的操縱性比別組優良許多，類似的控制程式，我們的船得以漂亮過彎，別組的不一定可以。這樣的結構，也使得下文的轉彎策略得以有效實現。

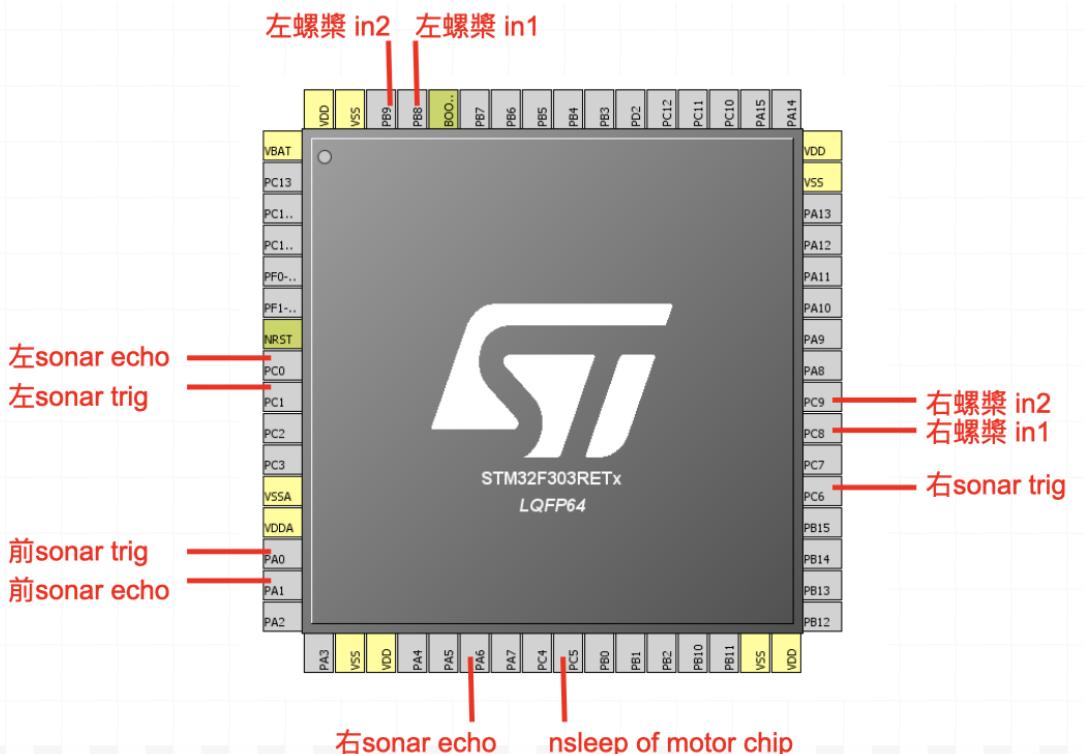
4. 船體的實際圖片





螺槳完整沒入水中

3. 電路設計：



Pin 腳：

我們簡單用以上pin作為開發板與超音波感測器的溝通腳位與馬達的PWM控制。

因使用mbed(為物件導向化)，所以我們僅需新增三個超音波感測的物件，並且確定以上腳位並未被佔用即可。

超音波供電的部分用開發板上的5v與GND分別作為超音波的VCC與VDD，並以麵包版作多點電源供應。

Motor chip:

同電子電路二，對應的腳位接上馬達，並針對特定腳位作PWM控制即可。

超音波感測器：

由超音波感測器的datasheet可得知，因聲音來回需要時間，故每個超音波感測器間發射間隔至少需 0.06 s ，以防止彼此干擾。因為有三個超音波感測器，需三倍時間 $0.06 * 3 = 0.18 \sim 0.2$ ，我們將一個超音波感測週期設定為 0.2 s 。

再由datasheet得知，超音波感測角範圍約30度，超過此感測角將發生回傳數字異常。我們一致認為超音波感測的擺放角度是最重要的，我們發現多數組別其感測器固定後就不再改動了，但其實擺放位置差異對於code的反應差距非常大，因此是一個調整空間非常大的變數，故經多次調整角度後得到一最佳擺放角度，使得控制的演算法能得到最佳效益。

2. 解釋你們的演算法與方位判斷的架構設計
我們的演算法分兩個部分

(1) 方位判斷

我們的方位判斷是利用左右兩顆感測器，如果某一側的距離大於另一側一個閥值 d 以上則往那個方向轉($d=|\text{左感測器測距} - \text{右感測器測距}|$)，否則就直行。為了區別直行時的小修正及大迴轉所需馬力的不同，如果 $5 < d < 15$ 則用70%馬力轉彎， $d > 15$ 則全力轉。另外一個跟其他組比較不同的是，為了避免超音波感測器的noise所產生的方向震盪(錯物值)，我們的距離計算是利用當下跟前 k 次的 moving average，最後我們使用 $k = 2$ 。

(2) 自動煞車

為了避免船發生意外而撞牆壁(意外通常來自於轉180度大彎時轉彎馬力不夠)，如果船頭的感測器發現距離小於45公分，則會進行反轉來作急煞車。

整體的Pseudo Code如下：

```
left_history, right_history, front_history = LOOKUP TABLE

while(True):
    Require front_distance, left_distance, right_distance from sensor

    # 計算moving average
    left_history.update(left_distance) # push new data & pop oldest data
    right_history.update(right_distance) # push new data & pop oldest data
    front_history.update(front_distance) # push new data & pop oldest data

    left_average = left_history.average()
    right_average = right_history.average()
    front_average = front_history.average()

    # 若前方快撞牆 -> 自動煞車
    if (front_average < 45):
        REVERSE()

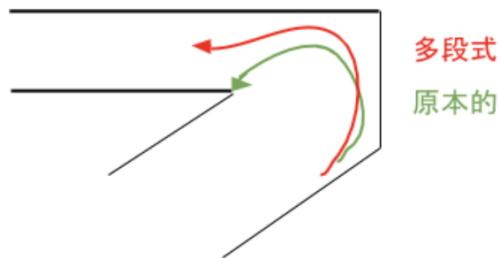
    # 否則判斷運行方向
    else if (left_average - right_average > 5):
        if (left_average - right_average > 15):
            TURN_LEFT(power = 1)
        else:
            TURN_LEFT(power = 0.7)
    else if (right_average - left_average > 5):
        if (right_average - left_average > 15):
            TURN_RIGHT(power = 1)
        else:
            TURN_RIGHT(power = 0.7)
    else:
        FORWARD()
```

3. 附上你們的所有的code，並特別標示出每一個function的功能
額外放在檔案中。

4. 額外設計: 如果你們有任何特別設計可以在此詳述

重量分佈對於船隻轉彎能力非常重要，經多次調整後，將電池位置固定，得到最好的重量分佈-轉彎能力關係(重量幾乎由電池提供)。

其中，我們最後能以最少次數撞牆過關，是因為一個突發奇想：多段式轉彎，其能在直線道上以小角度修正前進，而在中間大彎道同樣保有原本的轉彎能力，但又不至於過度轉彎使之回頭。



原本的單一轉彎可能造成過度轉彎，較為不穩定。

與其說把船想成是具有視野，不如把船想成是一位有導盲杖的視障人士。我們的策略是把船的PWM打滿，搭配我們以上提到的演算法，以及船體立即且良好的轉彎性能，利用船隻的移動及旋轉慣性，左右搖擺。此策略可以使船行進時如視障人士使用導盲杖的方式一樣，不停的變換導盲杖(超音波感測器)的方向，有效地判斷地形。雖然說這樣的方法會讓船走一些額外路徑（船隻路徑會有小幅的S型），但在計算撞牆的秒數加成後，**我們的移動時間是最短的，撞擊牆壁的次數僅有一次**（非常輕輕地碰一下），也是所有組別中最少的。甚至我們可以這樣說，只有我們的控制策略是有可能不撞到牆壁，他組多是以巧妙的撞擊角度與力道來修正船隻行進的方向。

分析影片後，我們推論會撞到牆是因為超聲波感測器感測角(30度)不足所致，右感測器與前感測器中有個相當角度的死角。平時難以進入死角，而比賽時稍微不幸。