車體設計及循跡

我們的車的硬體設備與其他組別相同。全車由2顆3.7V鋰電池供電，電池盒連接到升壓模組，我們這組的工作電壓是10.0V, 高於其他組別。Arduino 及其上連接的擴充模組，用來控制其他諸多設備，如馬達控制模組L298N, 紅外線sensor, RFID版, 以及藍芽模組。

循跡

我們的車子能前進行走，也可以倒退行走，兩個方向都能循跡。

我們使用PID Alpha循跡。車子的中間偏前方設置5個sensor. 當sensor看到黑線時，讀值為1, sensor 看到白紙讀值是0. 由於sensor只會是0 or 1, 因此error所有的可能值只會有9種，這讓要求精密度的I control 增加了一些難度。

Error= r2\*(\_w2)+r1\*(\_w1)+l1\*(-\_w1)+l2\*(-\_w2). 若同時亮兩燈，則error=error/2.

dV= vR(t­­0)-vR(t0-0.1)

sumError 則是把error從上一個node積分到現在的時間點。積分方法為每隔1ms 加總error.

我們用這些變數去計算馬達的轉速，公式如下：

前進時，

vR = Tp+power/2

vL = Tp-power/2

倒退時，

vR = -Tp+power/2

vL = -Tp-power/2

由於接線問題，vR 實際代表左輪速度，vL 則是右輪速度。大約在6月初，我們發現左右馬達轉速不同，經過實驗，前進時MotorWriting (vR\*1.4, vL), 能保持直線行走，後退時MotorWriting (vR, vL), 能保持直線行走。循跡穩定性對於正確轉彎非常重要，如果沒辦法準確進入node, 則轉彎非常容易失敗。

6/14 demo時的參數: w2=16, w1=8, Kp=24, K­d= 6.4, Ki= 0.0004, Kα =0.72, T­p= 128

以下稱以速度(179,128)前進, (-128,-128)後退為 “正常速度”.

轉彎及倒車

當連續3個sensor 看到黑色，我們就當作遇到node. 遇到node有四種動作：車會直走(指令1)、右轉、左轉(指令 2, 3)、讀RFID並改變行進方向 (前進變倒退，倒退變前進)(指令4). 指令0代表結束, 讀完RFID以後車子便停下。

我們認為迴轉會花費太多時間，且不穩定，因此改用倒車取代迴轉，雖然會使的循跡的工作量加倍，且倒退循跡遠比前進循跡困難，但確實節省了時間，也成了我們這組最大的特色及創意。

前進

收到指令1, 車保持前進方向，並以正常速度花費600毫秒通過node, 通過node期間不能循跡，故有時離開node會有較大的error。

轉彎

轉彎分為前進、後退兩種。原先我們把轉彎的時間寫死，後來我們在轉彎時，serial print 各個sensor每5ms的讀值，並進行大量的數據分析，從而改用兩階段判斷。轉彎分為4個步驟:

1. Sensor 看到node 時，輪子仍沒有對到路口中央，因此我們會讓車子以正常速度前進一段時間 tenter, 讓車輪對準正中央。之後停止一段時間。tenter 由實驗獲得，前進為90ms, 後退為60ms。

2. 車子以(-64, 64)左轉或(64, -64)右轉一段時間t­turn­, 這個時間由實驗決定，通常介於 600-700 ms.

3. 第2步驟完成後，若中央的sensor讀值不是1, 則繼續以相同速度轉，直到中央sensor看到黑線為止。

4. 車子以(255,255)走一小段時間 tleave, 離開node, 回到循跡模式。

無論轉彎前是前進還是後退，轉彎後皆變為前進。後退轉彎與前進轉彎第2步驟以後都相同。前進右轉及倒車左轉指令2, 前進左轉及倒退右轉為指令3, 整個轉彎過程約費時1.2秒。

讀RFID並改變行進方向

若車子前進進入端點，則當sensor看到node以後便立刻停下，隨即讀RFID; 若車子倒車抵達端點，則需要以正常速度行進150 ms, 才能讓RFID接收器對準卡。花費短暫時間讀RFID以後隨即前進變倒退，倒退變前進，並以正常速度，花費300ms 離開node.