計算型智慧 HW1 說明文件

104502518 劉冠聲

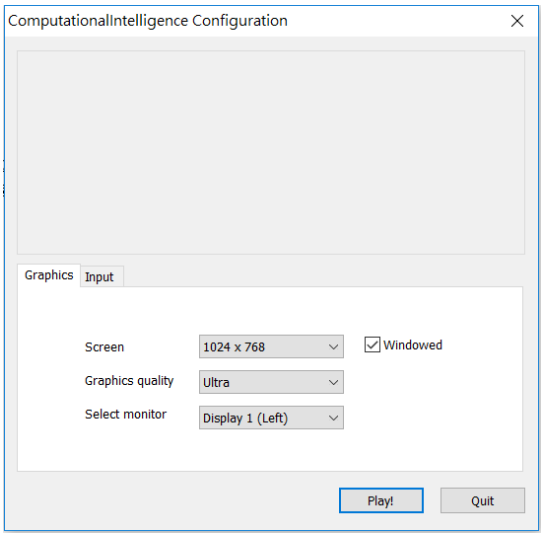
- *前置步驟*

  請將測試用的資料放置至

  104502518_劉冠聲_HW1_V01\執行檔\104502518_劉冠聲_HW1_V01_Data

- *程式介面說明*

1. 執行 104502518_劉冠聲_HW_V01.exe 後顯示如下視窗



選項設置為

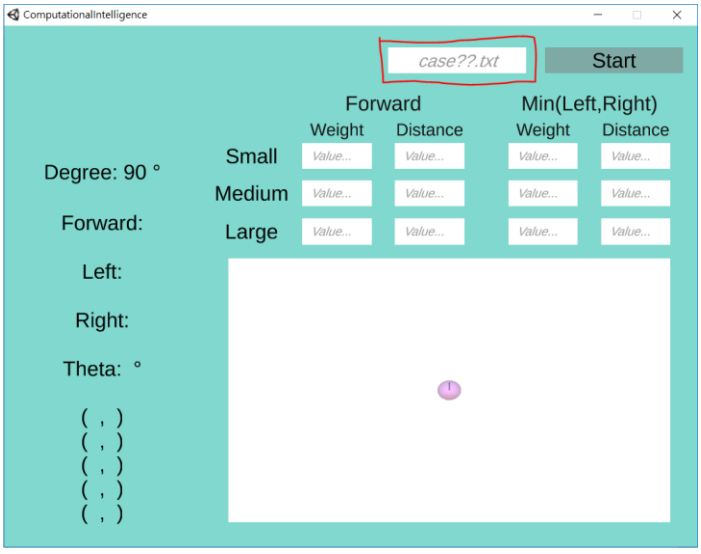
Screen : 1024*768

Windowed : checked

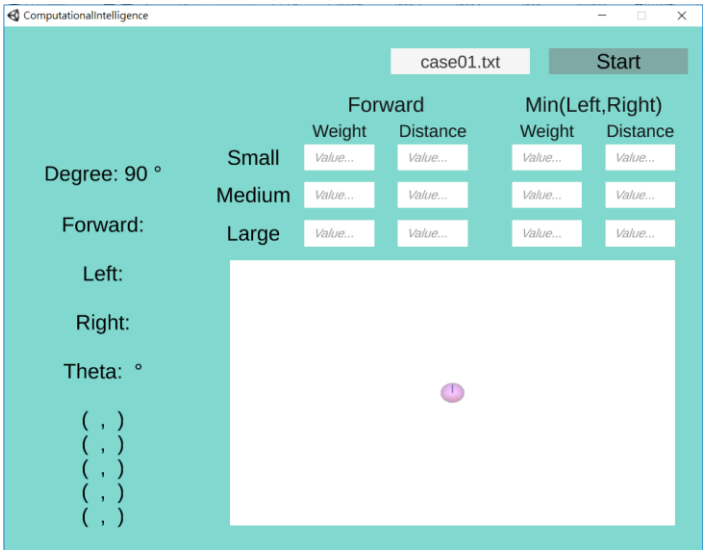
Graphics quality : Ultra

Select monitor : Display 1

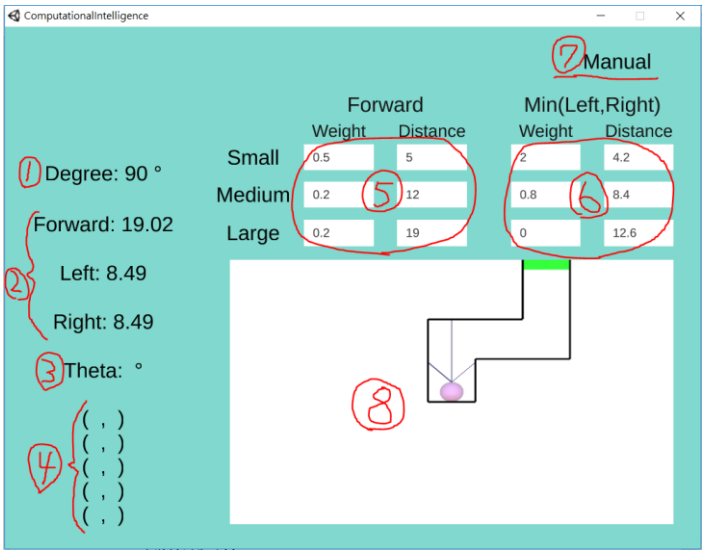
2. 點擊 Play!按鈕後經過場動畫後顯示如下視窗



請在紅色框框處輸入測試資料的檔名(EX : case01.txt)



3. 點擊右側的 Start 按鈕後顯示如下，為手動模式(Manual)



①目前車子的角度

②分別為前方距離、左方距離、右方距離(為了顯示出牆壁的寬度導致有一點誤差，但不影響模糊規則的應用)

③經模糊規則計算後得出的方向盤角度

④近 5 次的車子座標(由於 TimeScale 設定為 10，故每秒會更新 10 次)

⑤前方距離在 Small、Medium、Large 代表的權重和距離值

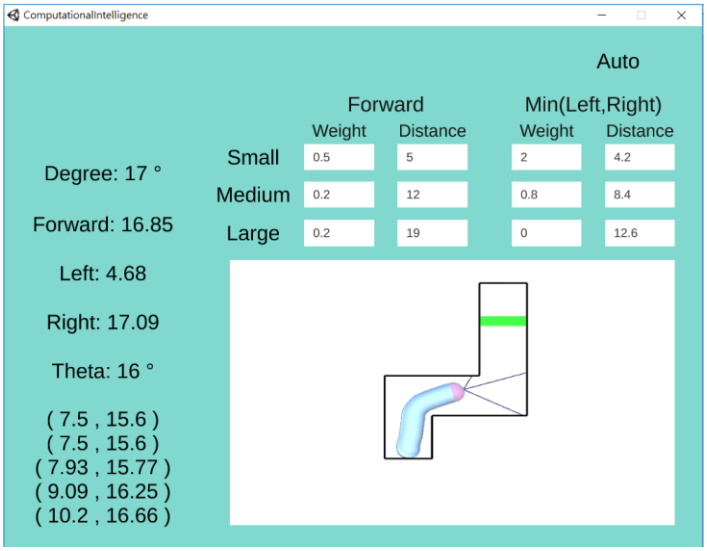
⑥左方距離和右方距離間較短者在 Small、Medium、Large 代表的權重和值

⑦手動模式(Manual)和自動模式(Auto)的切換按鈕，若要中途更新權重或距離值請先切換至手動模式，輸入完後在切換至自動模式，便會以更新後的設定繼續運作

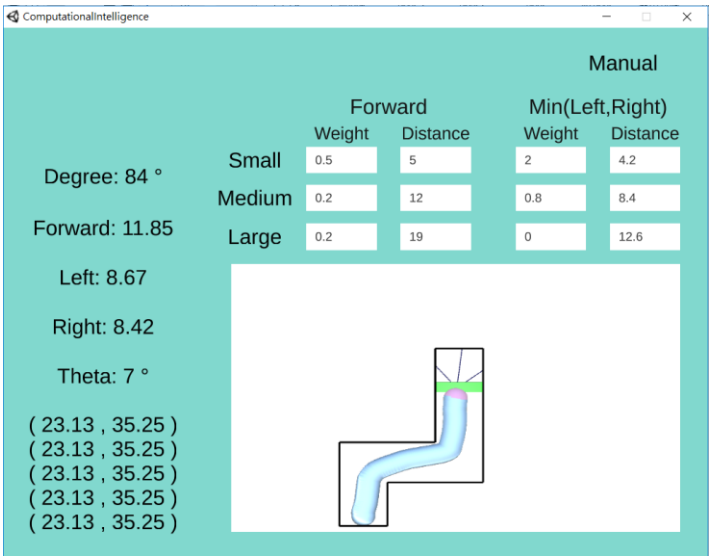
(手動模式可以利用箭頭上左右鍵手動控制)

⑧運行畫面

4. 各欄位初始值為實驗找到最適合的參數，點擊 Manual 切換至自動模式開始運作，後面的淺藍色為車子行走的軌道，可隨時點擊右上角之 Auto 暫停運作，切換為 Manual



5. 當車子中心成功進入綠色終點區或是撞到牆壁時，回到一開始輸入檔名的畫面



6. 若是成功走到終點，會在 104502518_劉冠聲_HW1_V01\執行檔\104502518_劉冠聲_HW1_V01_Data 產生 train6D.txt 記錄每秒車子的座標(因為 TimeScale=10，故每秒會有 10 筆紀錄)。
若是失敗，train6D.txt 則記錄 Failed.

- *程式碼說明*

1. ComplexControl.cs：將前方距離和左右距離短者套入模糊規則運算，去模糊化後得出計算 θ 的權重 afa 和 beta，並用 θ 計算下次車子的 x,y, Φ

2. CoordinateLog.cs : 顯示車子的座標
3. DegreeLog.cs : 顯示車子角度
4. DistanceLog.cs : 顯示前方距離、左方距離、右方距離
5. ThetaLog.cs : 顯示經模糊計算後得出的θ
6. FileManager.cs : 讀取檔案繪製地圖、設定車子和終點位置，將成功抵達終點之車子每秒座標或失敗紀錄寫入 train6D
7. GoalReach.cs : 偵測車子中心進入終點
8. LineSensor.cs : 計算前方距離、左方距離、右方距離
9. SimpleControl.cs : 簡單手動控制、繪製車子行走軌道、偵測撞到牆壁
10. Switch.cs : 切換手動模式和自動模式，若是從手動模式切換至自動模式，便以各欄位數值更新模糊規則設定

- *模糊規則設計*

函數規則為 θ = 40 * afa * beta

afa 為前方距離透過模糊規則計算後去模糊化而得

beta 為左右距離短者透過模糊規則計算後去模糊化而得

詳細如下↓↓↓

afa : 歸屬函數使用高斯函數

$$\exp\left(\frac{-1*(x-10)^2}{2*3^2}\right)$$

Small : 中心 5　afa = 0.5

Medium : 中心 12　afa = 0.2

Large : 中心 19　afa = 0.2

使用權重式平均法去模糊化

---

beta : 歸屬函數使用高斯函數

$$\exp\left(\frac{-1*(\mathrm{x}-10)^2}{2*1^2}\right)$$

Small : 中心 4.2　beta = 2

Medium : 中心 8.4　beta = 0.8

Large : 中心 12.6　beta = 0

使用權重式平均法去模糊化

- *實驗結果與分析*

之前有把 case01 抵達終點各欄位數值記在紙上，但後來找不到那張紙…

一開始很直覺得就決定使用函數式來設計模糊規則，因為方向盤最大轉的角度為 40 度，便設定了θ = 40 * afa * beta，其中 afa * beta 不會大於 1。

第一次的策略是左右距離一有差，就馬上開始轉方向盤，轉的幅度以左右差的大小為主，前方距離為輔，一開始不大能拿捏前方距離和左右差的 Small、Medium、Large 的範圍要怎麼抓還有各規則對應到的 afa、beta 應該要多少，調整了很久，大多都是在第一個彎沒辦法轉成功，參數調太小直接撞牆，調太大又會直接卡轉角，好不容易參數適中，第一個彎轉過了，但因為緊接著第二個彎使左減右變大，馬上開始左轉卡到第二個轉角，後來決定改策略。

第二次的策略是以前方距離為主，左右距離相減的大小為輔，在越靠近牆壁時方向盤轉的角度越大，預期狀況是在靠近牆壁時能夠將方向盤轉到近 40 度，以利快速轉彎，但一開始遇到的狀況是，第一個轉角成功轉彎，車子角度仍然有偏，撞上第二個轉角，不過多次修改參數後，對 Small、Medium、Large 的範圍大概比較會抓了之後，有成功抵達 case01 終點過，我有記在紙上…
後來跟同學借了他設計的其他地圖卻無限撞牆，因為這次的參數很多是專門為 case01 設計的，再繼續修正後，突然想到更好的第三次的策略。

第三次策略首先先把 Small、Medium、Large 的範圍和高斯函數的延伸程度透過道路寬度通常為 12 這點確定了，然後我把左右差這個變數改成了左右距離較小者，也就是 Min(左方距離,右方距離)，既能控制左轉右轉也能判斷是否快要撞到左側或右側的牆，所以我以左右距離較小者為主，前方距離為輔，而預想狀況還是沿用第二次策略的甩尾模式，最後成功讓車子在轉角能連續大幅度轉彎，並維持在路的正中間行駛，即使接近了兩側的牆壁，也因為以左右距離較小者為主，成功迅速迴轉避免撞牆，而前方距離很近時，有兩種狀況，第一種是垂直往牆前進，左右距離一樣，較小者便是擇一(我是擇左)，越前進左右距離也會越小，故成功轉彎，第二種是左右不一樣，就是一般轉彎的狀況，成功。