具定位校正機制之群組機器人室內導航與控制系統之研究

彭君豪、簡宜興*、許陳鑑、王偉彥

國立臺灣師範大學電機工程學系

摘要

本論文提出具定位校正機制之群組機器人路徑追蹤以及路徑規劃演算法，透過融合里程計以及雷射距離感測器的設計，來解決里程計存在累積誤差的問題。在定位方面則是引進蒙地卡羅定位法，以粒子濾波器作為主要架構，藉由雷射資訊進行機器人自主定位，透過定位結果補償移動過程中里程計的誤差，來達到更好的控制效果。在路徑規劃方面，考慮到路徑會隨著機器人的移動而有所改變，基於 D*Lite 演算法的基礎，並將演算法推廣到群組，視其他機器人為障礙物，進行迴避。並交由 MySQL 伺服器進行資料統籌，減少機器人的運算負擔，讓機器人可以專心處理感測器資訊。在路徑追蹤方面，本文使用倒階控制法設計運動學控制器，將路徑規劃的結果當作參考信號，以蒙地卡羅定位以及里程計混合輸出結果取代過去所使用的里程計定位，以避免里程計的累計誤差。實驗結果表示此方法能使機器人準確運行於事先規劃好的軌跡之上，並且能夠發現群組中其他機器人的存在，並嘗試進行迴避。

關鍵字：群組機器人系統、D*Lite 演算法、蒙地卡羅定位、路徑追蹤控制。

Abstract

This paper presents path tracking algorithm incorporate localization correction mechanism based a planned path. The existed odometer cumulative error problems can be solved by fusing the odometer readings and localization results. For path planning, we apply the D* Lite algorithm to determine an optimal path and extend its use to multi-robot path planning, where other robots are viewed as obstacles to be avoided. A MySQL server is used to integrate information obtained from odometer readings and localization results by a Monte Carlo Localization (MCL) algorithm to reduce the loading of the robots so that they can concentrate on sensing the environment. As for path tracking, the paper uses backstopping method to design a kinematic controller where the planned path is considered as a reference signal. Experimental results show that the proposed method can control the robot to move along the planned path with good agreement.

Keywords: Multi-robot system, D* lite algorithm, Monte Carlo Localization (MCL), path tracking control.

壹、緒論

隨著機器人科技的持續發展，逐漸開始有人意識到多機器人可以同時多工執行任務，效率遠高於單一機器人獨立作業。群組機器人之間具備通訊與相互協作之功能，可以處理更複雜之問題。除此之外，當其中一個機器人故障時，其他機器人可取代故障機器人繼續工作，不影響整個團隊任務的進行，具備系統強健性。這些的優點，使得群組機器人成為機器人研究發展中熱門題目。

群組機器人導航是單一機器人導航的延伸研究，因此在群組機器人導航系統當中，依照任務性質可以區分為定位 (Li et al., 2014; Paull et al., 2014; Rashid et al., 2015; Rodríguez-Araújo et al., 2014; Shim & Cho, 2015)、路徑規劃 (Al-Mutib et al., 2011; Conkur, 2005; Ferguson & Stentz, 2005; Ge & Cui, 2002; Wang et al., 2015) 與控制 (Antonini et al., 2006; Chen et al., 2009; Chien et al., 2015; Chien et al., 2012; Martins et al., 2008;) 等三大主題。「定位」是為了讓機器人獲得當前位置，以提供控制與路徑規劃所需之資訊，廣義來說，可視為系統之回授，用以估測機器人之狀態。「路徑規劃」在傳統的導航任務當中，是負責規劃可通行之最佳路徑以提供機器人進行移動，而若要考慮到路徑追蹤控制，則除了規劃可通行的路徑之外，還需要設計機器人朝向角度以及抵達時間。「控制」泛指機器人路徑追蹤，即在取得定位回授以及路徑規劃結果之後，控制機器人的運行使機器人能精地行走於預定好的路徑。

機器人定位技術已發展很長一段時間，除了透過馬達速差取得里程計當前的位置 (Kao et al., 2013) 以及 GPS 透過衛星回傳當前位置 (Drawil et al., 2013) 之外。發展最多的定位演算法通常是透過距離傳感器，比如雷射測距儀、超音波或深度攝影機，在已知地圖進行定位估測。最早提出的卡爾曼濾波器 (Kalman, 1960)、貝氏濾波器 (Arulampalam et al., 2002; Ho & Lee, 2001) 以結合而成的馬可夫定位法(Markov localization) (Fox et al., 1999)，以及目前最普遍採用的蒙地卡羅定位法(Monte Carlo localization) (Dellaert et al., 1999)。蒙地卡羅定位法由 D. Fox 所提出，利用粒子濾波器為架構實現機器人定位法則，利用感測器融合後之障礙物資訊與環境地圖做對應，再利用貝式機率計算粒子與機器人之間的相似程度或是可信度，以此作為依據，逐漸淘汰不良的粒子，使粒子群逐漸收斂至同一區域，進而估測機器人在環境中移動時的位置。在定位演算法當中，主要的任務為全域定位(global localization)與位置追蹤

(position tracking)，全域定位的定義為在沒有預先知識(prior knowledge)下定位機器人位置，定位演算法必須在全域不確定性(global uncertainty)下估測出機器人位置，位置追蹤的目標是補償機器人在導航時所產生的里程計誤差，因此前提是必須要知道機器人的初始位置。

路徑規劃方面以 1968 年由 Peter Hart 等人所提出之 A* Algorithm (Hart et al., 1968) 應用最為廣泛，其精髓為啟發式搜索(heuristic)，並使搜尋具有方向性。A* Algorithm 的優點是演算法操作公式簡單且具有高計算效率，但由於這種方向性搜尋法，若起點與終點之間出現特殊障礙物，如凹型障礙物(concave obstacles)，則將會陷入"暫時性"的區域最佳解，造成過多的不必要搜尋，此外 A*演算法規劃出來的路徑會緊貼著障礙物，容易使機器人在移動時有安全上的顧慮，並且 A*演算法缺乏路徑更新的機制。在 2005 年，Sven Koenig 所提出的 D*Lite (Koenig & Likhachev, 2005)，提出了路徑更新的概念，在地圖發生變化時，能夠利用過去搜尋的結果進行路徑更新，減少更新路徑所需之時間。

路徑追蹤控制器，顧名思義是使其能夠行走於預先規劃好的軌跡之上。主要可分為兩個控制策略：第一個控制策略為運動學控制器，另一策略則針對機器人動態模型設計控制器。De La Cruz 以及 Carelli (De La Cruz & Carelli, 2006) 提供了一種考慮動態模型的控制方法，以速度和角速度作為控制器的輸入。然而，如果參數選取錯誤或隨著時間大幅度的變化，控制器的性能將會下降，因此如何調整參數就是一個重要的課題。文獻 (Chen et al., 2009) 提出了一個兩階段的路徑追蹤滑動模式控制器，其設計了一個滑動面。滑動模式控制可簡化輸入資訊，具備強健性，此文獻利用滑動模式控制用以消除系統的不確定性和外部干擾，但訊號抖震的現象會導致控制力在兩數值中不斷切換，造成系統的不穩定。文獻 (Li et al., 2010) 使用滑動 PID 控制器來解決外部干擾，透過幅狀基底函數類神經網路(RBFNN)與自適應調節來解決訊號抖震的問題，但因執行時間太冗長，且機器人須及時動作，因此並非為最佳的方法。Das 及 Kar (Das & Kar, 2006) 設計了一個適應性模糊控制器，其參數可以即時調整，而機器人的速度和角速度是利用類神經網路來控制 (Antonini et al., 2006)。然而，執行類神經網路需要很高的運算量。

本論文設計並整合多個導航功能相關的演算法並將其應用於群組機器人上。因此，我們需要考慮群組之間應該如何有效地避免碰撞，如何規劃出一條避開彼此的路徑，以及系統之

間該如何相互配合，以做出最妥善的任務安排。因此在路徑規劃方面，我們改良了傳統 D*Lite 演算法，將其服務對象由單機推廣到群組，並加入安全性機制，避免路徑貼牆，路徑追蹤控制方面，為了解決里程計長距離之累計誤差，我們引進雷射測距儀，並使用蒙地卡羅定位演算法來校正誤差，雖然增加了運算量，但也有效地降低了機器人的移動誤差，使得機器人能較準確地移動更遠的距離。

貳、具定位校正機制之路徑追蹤控制器設計

一、輪型機器人之運動學模型

輪型機器人最典型的模型如圖 1 (Fierro & Lewis, 1995) 所示，l 代表輪型機器人的車體寬度，車體左右兩邊有兩個驅動輪，而在車體的尾部有一個輔助輪。我們定義機器人實際位置 $\mathbf{q}$ 和參考位置 $\mathbf{q}_r$ 如下:

$$\mathbf{q}=\left[x, y, \theta\right]^T \in \Re^3, \mathbf{q}_r=\left[x_r, y_r, \theta_r\right]^T \in \Re^3 \tag{1}$$

其中 x、y 是橫軸與縱軸的實際位置，θ 是機器人的方向。x_r 、y_r 和 θ_r 是參考訊號。

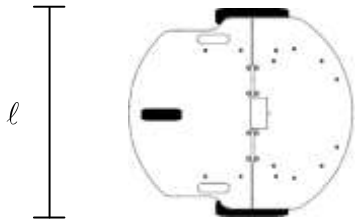


圖 1、輪型機器人外觀

機器人的運動學模型定義如下：

$$\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{\theta}\end{bmatrix}=\begin{bmatrix}\cos\theta & 0\\ \sin\theta & 0\\ 0 & 1\end{bmatrix}\mathbf{V} \tag{2}$$

其中 $\mathbf{V}=[v,\omega]^T$ ，v 代表機器人的移動速度，ω 代表機器人的角速度。

二、路徑追蹤控制器

前一小節中我們描述了輪型機器人的運動模型，發現機器人下一時刻的狀態，只與當前朝向角以及速度向量 $\mathbf{V}$ 有關，所以只要能求出欲抵達目的地的控制速度向量 $\mathbf{V}_c$，再將其轉換為左、右輪轉速，即能驅動輪型機器人前往目的地。接著，我們將探討如何進行誤差回授、透過倒階法 (Fierro & Lewis, 1995) 求得控制速度 $\mathbf{V}_c$，以及如何換算出左、右輪速。我們定義位置誤差向量 $\mathbf{q}_e$ 如(3)。圖 2 為機器人的位置誤差圖。

$$\mathbf{q}_e = \begin{bmatrix} q_{e1} \\ q_{e2} \\ q_{e3} \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_r - x \\ y_r - y \\ \theta_r - \theta \end{bmatrix} \tag{3}$$

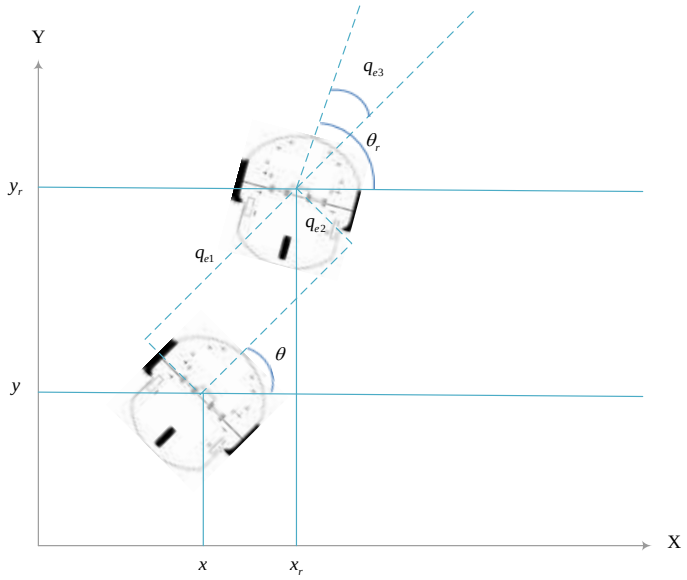


圖 2、機器人位置誤差圖

圖 2 中，我們定義了參考座標 $\mathbf{q}_r$，其代表的意義為欲前往的目的地。當前實際位置 $\mathbf{q}$ 與參考位置之間的關係，有旋轉量以及位移量兩個部分，所以誤差信號我們可以表示為旋轉矩陣的形式。有了誤差信號，我們就能夠透過倒階法來進行速度控制。首先經由(3)將 q_{e1} 做微分，則可得到(4)

$$\dot{q}_{e1} = (\dot{x}_r - \dot{x})\cos\theta + (\dot{y}_r - \dot{y})\sin\theta - (x_r - x)\dot{\theta}\sin\theta + (y_r - y)\dot{\theta}\cos\theta \tag{4}$$

將 $\dot{\theta} = \omega$ 和 $v = \dot{x}\cos\theta + \dot{y}\sin\theta$ 代入(4)，則可得到(5)

$$\dot{q}_{e1} = -v + \omega q_{e2} + v_r \cos q_{e3} \tag{5}$$

經由(3)將 q_{e2} 做微分，則可得到(6)

$$\dot{q}_{e2} = -\omega q_{e1} + v_r \sin q_{e3} \tag{6}$$

經由(3)將 q_{e3} 做微分，則可得到(7)

$$\dot{q}_{e3} = \dot{\theta}_r - \dot{\theta} = \omega_r - \omega \tag{7}$$

根據上列的計算，誤差方程式(3)可以寫成(8)。

$$\begin{bmatrix} \dot{q}_{e1} \\ \dot{q}_{e2} \\ \dot{q}_{e3} \end{bmatrix} = v \begin{bmatrix} -1 \\ 0 \\ 0 \end{bmatrix} + \omega \begin{bmatrix} q_{e2} \\ -q_{e1} \\ -1 \end{bmatrix} + \begin{bmatrix} v_{\mathrm{r}} \cos q_{e3} \\ v_{\mathrm{r}} \sin q_{e3} \\ \omega_{\mathrm{r}} \end{bmatrix} \tag{8}$$

其中 v_{r} 和 ω_{r} 是參考速度和參考角速度，且 $\mathbf{V}_r = [v_r, \omega_r]^{\mathrm{T}}$ 。

在文獻 (Chen et al., 2009) 中，作者透過倒階控制法去設計速度控制器，如下:

$$\begin{bmatrix} v_{\mathrm{c}} \\ \omega_{\mathrm{c}} \end{bmatrix} = \begin{bmatrix} v_{\mathrm{r}} \cos q_{e3} + \alpha_1 q_{e1} \\ \omega_{\mathrm{r}} + \alpha_2 v_r q_{e2} + \alpha_3 v_{\mathrm{r}} \sin q_{e3} \end{bmatrix} \tag{9}$$

其中 $\mathbf{V}_c = [v_c, \omega_c]^{\mathrm{T}}$ 是速度控制器。當 α_1 、 α_2 和 α_3 是正數時，可由定理 1 證明系統的穩定性。

定理 1: 如果運動學控制器設計如(9)，則當時間趨近無窮大時，機器人實際位置 $\mathbf{q}$ 將會趨近參考位置 $\mathbf{q}_{\mathrm{r}}$ 。

證明: 考慮一個 Lyapunov-like 方程式如(10)。

$$V(q_{e1}, q_{e2}, q_{e3}) = \frac{1}{2}({q_{e1}}^2 + {q_{e2}}^2) + \frac{1 - \cos q_{e3}}{\alpha_2} \tag{10}$$

將(10)做微分，則可得到(11)

$$\dot{V} = q_{e1}\dot{q}_{e1} + q_{e2}\dot{q}_{e2} + \frac{\sin q_{e3}}{\alpha_2}\dot{q}_{e3} \tag{11}$$

將(8)代入(11)，則可得到(12)

$$\dot{V} = q_{e1}(-v_c + \omega_c q_{e2} + v_r \cos q_{e3}) + q_{e2}(-\omega_c q_{e1} + v_r \sin q_{e3}) + \frac{\sin q_{e3}}{\alpha_2}(\omega_r - \omega_c) \tag{12}$$

將(12)整理過後，則可得到(13)

$$\dot{V} = -v_c q_{e1} + v_r q_{e1} \cos q_{e3} + v_r q_{e2} \sin q_{e3} + \frac{\sin q_{e3}}{\alpha_2}(\omega_r - \omega_c) \tag{13}$$

再將(9)式帶入(13)，則可得到(14)

$$\begin{aligned} \dot{V} &= -\alpha_1 q_{e1}^2 + v_r q_{e2} \sin q_{e3} + \frac{\sin q_{e3}}{\alpha_2}(-\alpha_2 v_r q_{e2} - \alpha_3 v_r \sin q_{e3}) \\ &= -\alpha_1 q_{e1}^2 - \frac{\alpha_3 v_r}{\alpha_2} \sin^2 q_{e3} \end{aligned} \tag{14}$$

由於α_1、α_2、α_3及v_r為設計參數且皆為正值，則$\dot{V} \le 0$。根據 Barbalat lemma (Wang & Mendel, 1992)，我們可得到當時間趨近於無窮大時，位置誤差向量$\mathbf{q}_e$會趨近於零。

三、定位校正機制

本節將介紹利用蒙地卡羅定位演算法來校正里程計的過程。首先，輪型機器人回傳當前狀態X_t進入系統，系統將檢查資料類別當中蒙地卡羅定位是否完成，檢查的方法是察看暫存器內的定位結果是否與上一筆相異，如果蒙地卡羅定位完成定位則回傳定位的結果，如果定位沒有完成則直接將輪型機器人當前狀態X_t直接輸出成為下一狀態$\hat{X}_t$。

由於蒙地卡羅定位演算法是建立在已知環境地形的基礎下進行定位，描述定位的方式與地圖的解析度息息相關，而路徑追蹤控制器通常需要以釐米作為單位這樣做才足夠精細，所以在雷射校正機制當中需要進行座標系的映射，將定位結果映射到釐米座標系當中，除了比例上的調整之外，里程計本身是一個以起點當作原點紀錄相對座標的感測器，所以我們也需要將里程計映射到絕對座標，轉換的方式不外乎旋轉以及平移，我們可以使用齊次矩陣來描述如(15)所示，其中x_0, y_0, θ_0為蒙地卡羅定位初始的起點位置。$x_{MCL}, y_{MCL}, \theta_{MCL}$則為當前時刻的定位結果。$R$代表單位相素對應到現實世界中多少釐米的比例常數。$x, y, \theta$則為經定位校正過之下一狀態$\hat{X}_t$。

$$\begin{bmatrix} x \\ y \\ \theta \end{bmatrix} = R \begin{bmatrix} \cos\theta_0 & -\sin\theta_0 & 0 & -x_0 \\ \sin\theta_0 & \cos\theta_0 & 0 & -y_0 \\ 0 & 0 & 1 & -\theta_0 \end{bmatrix} \begin{bmatrix} x_{MCL} \\ y_{MCL} \\ \theta_{MCL} \end{bmatrix} \tag{15}$$

在經過座標轉換函數的映射之後，蒙地卡羅定位的結果與里程計回傳結果可映射在同一座標平面當中。接著我們將討論時間延遲的問題，里程計是讀取馬達編碼器速差，得知當前位置的感測器，取樣時間可以控制在 10 毫秒內，但蒙地卡羅定位乃是一個迭代演算法，每一筆定位結果都需經歷預測、權重計算以及重新取樣迭代才能取得，定位時間隨粒子數的多寡、地圖的複雜度以及擾動的範圍也有所不同，不過總體來說還是以秒為單位給予回授。這也代表收到雷射開始，直到蒙地卡羅定位計算完畢，定位的結果會有所延遲，以下使用圖 3 來進行說明。

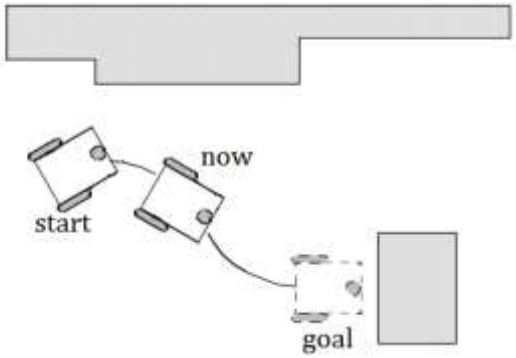


圖 3、定位延遲示意圖

輪型機器人一開始位於 start 的位置，欲前往 goal，在 start 的位置時接收雷射資料，當機器人走到 now 時定位完成，但此時並非位於收取雷射資訊的 start 而是位於 now，定位的回傳結果是由早些時間收取的雷射資訊計算而得，所得定位結果勢必與真實位置有所不同。為了解決此一問題，決定透過里程計補償定位延遲過程中的位移量，來達到定位延遲校正的作用，如(16)所示。其中 P 為蒙地卡羅定位的結果，ΔP_{Od} 為定位延遲過程中里程計的位移量，P' 則經定位延遲校正後之蒙地卡羅定位結果。

$$P' = P + \Delta P_{Od} \tag{16}$$

接著我們將討論長廊問題，由於蒙地卡羅定位演算法對於相似程度太高的地圖環境來說，時常會陷入區域最佳解，為了避免路徑追蹤控制器失控，本文採取濾除不合理的定位校正結果，來避免錯誤的校正導致錯誤的控制。在路徑追蹤控制器中，速度與角速度都是在系統的控制之下，因此如果定位的結果與控制的結果不相符，那麼此次定位的結果，有非常高的可能性是陷入區域最佳解。圖 4 為室內地形當中很容易出現的情況，圖中綠色圓圈代表機器人所在位置，紅色線段代表雷射，可以發現雖然朝向角不同，但是所收到的雷射資訊是完全相同，在輸入資訊完全無差異的情況下，蒙地卡羅定位將陷入區域最佳解，自然也不具備校正的能力。

圖 4、相似地形導致的朝向角大反轉

為了避免上述情況的發生，定位出來機器人朝向角與前一次定位的機器人朝向角加上取樣時間內的角位移數值大小超過門檻值的話，此次結果我們將不予採用。此外，在位移量方面，我們透過所記錄的前一時刻位置加上位移期間的移動量，作為本次定位成功與否的判斷標準，如果本次定位結果超過我們所設定的門檻值，本次定位將不採用。另外，如果定位結果位於牆內，本次的定位結果一樣不採計。

參、群組機器人導航系統

群組機器人系統可以共用機器人彼此的感測器資訊，處理更複雜的問題。在系統調度方面，也擁有更高的彈性，不會因單一機器人的故障，導致系統停止運作。感測器資訊共享的對策，對群組機器人來說是很重要的一個環節，畢竟群組本身就是一個複雜的系統。當考慮的層面更深、則應用的範圍越廣，原先單一機器人導航演算法，將不足以應付群組導航的任務。本章節將探討群組之間如何通力合作，以及提出導航演算法來執行群組機器人的導航任務。

一、群組機器人系統架構

群組機器人導航系統是利用群組中每一台機器人身上的感測器資訊來幫助完成任務，首要工作就是需要避免碰撞。碰撞的結果輕則朝向角發生偏移，嚴重一點可能導致設備掉落，甚至導致機器人故障。當前的機器人技術還無法做到全故障排除，因此應極力避免碰撞的發

生。因此，建立群組機器人之間互相溝通的管道十分重要，唯有將每一台機器人位置資訊通報給群組中的每個單位知曉，才能避免這種情況的發生。本論文提出的群組機器人伺服器端與客戶端的關係，伺服器作為機器人之間資訊傳遞的媒介並統整機器人所接收的地圖環境資訊，其角色等同於軍隊當中的司令官，能夠減少機器人客戶端的運算量，避免客戶端花費額外的運算量來等待其他機器人回傳資訊。

對於群組中每台輪型機器人，他們都擁有定位、路徑規劃、路徑追蹤以及更新資料庫的能力。演算法彼此的溝通透過發布/訂閱(publish/subscribe)架構進行設計，演算法運算完畢的結果無須傳送給特定的訂閱者，而是將資料彙整起來，分成一個或多個類別，有需要的演算法自行訂閱感興趣的資料。這個架構大大減低了演算法之間的相依性，方便進行抽換，程式的維護也更為單純。此外每個演算法分別使用一個執行緒進行平行處理，也避免的互相等待，造成控制延遲的問題。

二、群組機器人定位演算法

傳統的蒙地卡羅定位演算法，並沒有考量到多台機器人同時定位的問題，由於缺乏溝通協調的管道，機器人之間也無法分享彼此所獲得的資訊，進而導致難以完成任務。在本文所提出的系統架構中，使用 MySQL 作為資料儲存以及資訊交換的中繼站，將會記錄每一台機器人當前的位置，以及其移動的路徑，作為協調處理的資料儲存中心。機器人之間有了溝通協調之管道，就能夠分享彼此的資訊。

此外，使用傳統的蒙地卡羅定位演算法來做機器人定位，在單一機器人定位問題當中表現出眾，但是在群組機器人定位問題當中，部分雷射資訊有可能遭到其他機器人阻擋，如圖 5 所示，紅色及藍色圓圈代表機器人，橘色線段代表雷射資訊。因為沒有足夠的雷射資訊，定位演算法將無法準確地進行有效定位。為了解決雷射遭到阻擋，而無法正確定位的問題，本小節將提出改良型蒙地卡羅定位演算法以解決此問題。

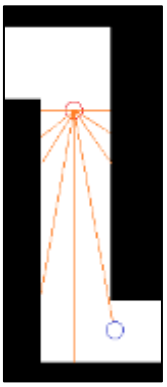


圖 5、機器人雷射遭阻擋示意圖

以下將介紹本論文提出的群組機器人定位演算法的步驟與說明。

步驟一: 將 m 個粒子分布在環境地圖中，每個粒子姿態需與機器人相同，因此各粒子狀態須包含位置與朝向角，表示如下：

$$p_0^i = (x_0^i, y_0^i, \theta_0^i), i = 1, 2, \cdots, m \tag{17}$$

其中，p_0^i 代表粒子 i，(x_0^i, y_0^i) 代表粒子所處座標，θ_0^i 代表粒子朝向角度。

步驟二: 根據機器人所接收之指令，改變粒子位置與朝向角，如(18)。

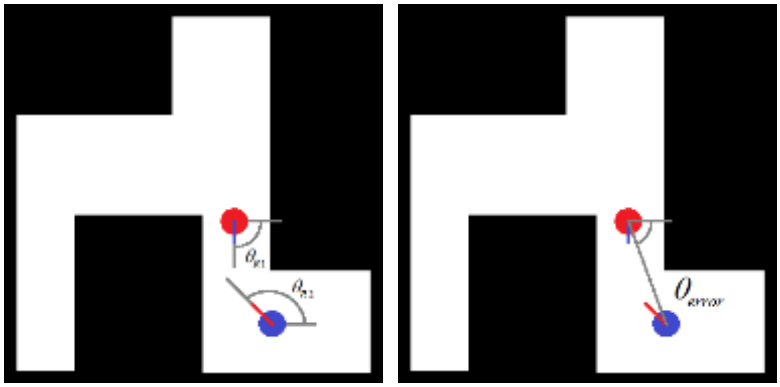
$$p_t^i \sim bel(p_t^i \mid p_{t-1}^i, u_t), i = 1, 2, \cdots, m \tag{18}$$

其中 p_t^i 代表 t 時刻第 i 個粒子，bel 代表貝式機率，為 t 時刻控制指令。u_t 為 t 時刻的控制力。

步驟三: 經由無線網路連結 MySQL 資料庫，讀取每一台機器人當前的方位。

步驟四: 找出錯誤雷射資訊，如圖 6 所示。我們以紅色機器人角度作為觀測主體，其位置為 $p(x_{R1}, y_{R1}, \theta_{R1})$，藍色機器人為觀測客體其位置為 $p(x_{R2}, y_{R2}, \theta_{R2})$，$\theta_{error}$ 為我們欲求被藍色機器人阻擋之雷射角度，其計算的方程式如(19)。接著，我們將移除此角度的雷射距離資訊。

$$\theta_{error} = \tan^{-1}\left(\frac{y_{R2} - y_{R1}}{x_{R2} - x_{R1}}\right) \tag{19}$$



(a) (b)

圖 6、找出錯誤雷射資訊示意圖 (a) 兩台機器人各自朝向角 θ_{R1}, θ_{R2}

(b) 遭阻擋之雷射角度 θ_{error}

步驟五: 各粒子依據所在之位置計算模擬感測資訊，依據模擬感測資訊與機器人實際獲得之感測資訊進行比對，計算出各粒子所擁有之權重值，並進行正規化，使其總合為 1。

$$w_t^i = w(o_t \mid x_t^i, \hat{o}_t^i), i = 1, 2, \cdots, m \tag{20}$$

$$\hat{w}_t^i = \frac{w_t^i}{\sum_{i=1}^{m} w_t^i} \tag{21}$$

其中，w_t^i 代表 t 時刻 i 粒子之權重值，o_t 為機器人在 t 時刻所獲得之感測資訊，x_t^i 代表 t 時刻 i 粒子之座標位置，$\hat{o}_t^i$ 代表 t 時刻 i 粒子之模擬感測資訊，(21)為正規化處理，$\hat{w}_t^i$ 代表 t 時刻正規化後 i 粒子之權重值。

步驟六: 在重新取樣部分，使用了隨機通用取樣法 (Pencheva et al., 2009)，或稱作輪盤法，其作法是將粒子權重值進行正規化，正規化後之粒子權重值總和為 1，再依照每個粒子在族群權重值所佔比例切割出該粒子的面積，權重值越高之粒子所切割出的面積越大，再依據粒子所佔面積從 0 開始給予數值範圍，面積越大者獲得之範圍越大。取樣時，隨機產生一個 0 至 1 的數值，依據產生之數值去對應擁有該數值之粒子，將此粒子放入下一代粒子群體中，再重新產生數值去取出粒子加入下一代粒子群體中，直到下一代粒子總數與上一代粒子總數相同為止。

三、群組機器人路徑規劃演算法

由於群組機器人的路徑規劃需考慮其他機器人的方位(不同的機器人將視為障礙物)，因此相較於單一機器人的路徑規劃，我們要考慮多個動態障礙物的情況。在本論文中，群組機器人會不斷地透過雲端資料庫來更新自身機器人的當前位置資訊以及獲知其他機器人(動態障礙物)的位置資訊，當某一機器人執行路徑規劃時，即可更新障礙物資訊，如果參考點路徑發生變化，則只需局部地更新障礙物周圍節點，因此可避免過長的更新路徑時間成本。以下將介紹本論文提出的群組機器人路徑規劃演算法的步驟與說明。

步驟一：首先計算整張地圖的 $h(s,s_{start})$ 值，計算當前格點 s 與起點的距離，最常使用的方法是切比雪夫距離。此值的用意是評估起點移動到此進行啟發式搜索(heuristic)的成本，啟發式搜索即錯誤嘗試，利用 $h(s)$值的大小來評估起點移動到這個格點的成本，成本越高，即最短路徑的機會越小，所以只要持續的從成本最低的節點向外擴張，則找到最短路經的速度越快。啟發式搜索以 Key 值的方式作呈現，同時存放 $\min(g(s), rhs(s)) + h(s,s_{start})$ 值以及 $rhs(s)$。heuristic 值和 Key 值的計算方程式如(22)及(23)。

$$h(\mathrm{A},\mathrm{B})=\max(|\mathrm{B}_x-\mathrm{A}_x|, |\mathrm{B}_y-\mathrm{A}_y|) \tag{22}$$

$$\mathrm{Key}=[\min(g(s), rhs(s))+h(s,s_{start}); rhs(s)] \tag{23}$$

接著，將地圖上所有節點的 *rhs* 和 *g* 值設成無限大，同時清空開放列表。然後將終點加入開放列表，並將 *g* 值設為無限大、*rhs* 值設為零。以終點作為路徑規劃的起始節點 u ，目標是找到起點。

步驟二：找尋開放列表當中 Key 值最小者做為當前節點 u ，如果最小 Key 值節點不只一個，則任意選一，此時也代表路徑很有可能不只一組解。由於此時只有一個終點座標，所以當前節點即為終點座標。接著找尋當前節點 u 的後繼節點加入開放列表 s'，並更新後繼節點 s'的 *g* 值與 *rhs* 值。更新 *g* 值的方法為: 當 $g=rhs$，則這個節點為 consistent，令 $g=\infty$。當 $g>rhs$，則這個節點為 over-consistent，令 $g=rhs$。當 $g<rhs$，則這個節點為 under-consistent，令 $g=\infty$。

D*Lite 演算法存在路徑貼牆的問題，如果規劃出來的路徑貼牆，會導致機器人與牆壁發生碰撞，因此讓機器人與牆壁保持安全距離，亦是一件重要的事情。在本文當中，我們考慮了地形成本函數，當位置 s 位於障礙物周圍時，移動成本將乘上一個係數 $T(s)$，更新 *rhs* 值的

方法如(24)。

$$rhs(s)=\begin{cases}0, \text{if } s=s_{start}\\ \min\limits_{s'\in Pred(s)}(g(s')+T(s)*c(s',s)), \text{otherwise}\end{cases} \tag{24}$$

步驟三: 如果機器人行走時偵測到障礙物，由於機器人無法通過障礙物，所以須將障礙物格點 *rhs* 值設成無限大，並且放進開放列表。因為路徑發生變化，需檢查障礙物後周圍節點的 *rhs* 值，如果障礙物周圍節點 *g* 值與 *rhs* 值不相等，則需放入開放列表。

重複上述步驟二及步驟三，直到機器人抵達目的地或開放列表中沒有任何節點。如果開放列表中沒有任何節點，代表機器人與目的地之間不存在可通行的路徑。從起點方向往 *rhs* 值最小的節點走，即可抵達目的地。

肆、實驗結果

本實驗使用兩台 Pioneer 3-DX 輪型機器人搭配雷射測距儀 LMS-100，其外觀如圖 7 所示，此機器人離地高度為 21.5 公分，寬度為 38 公分，迴轉半徑為 26 公分，配有里程計、超音波以及電子羅盤。其鋁製的車身可減輕機器人的重量，最多可以同時配備 3 顆鉛蓄電池，在只裝一顆電池的情況下，車體重量為 9 公斤，機器人在兩側裝有驅動輪並在車體的後方加裝萬向輪，以增加機器人移動的穩定度，藉由車體前方的 8 顆超音波，機器人可以感測與物體之間的距離。

圖 7、群組機器人外觀顯示圖

本實驗所使用的機器人(Pioneer 3-DX)內部含有編碼器，經由定義機器人開始放置的座標為原點，編碼器可以由馬達的轉速來計算出機器人目前座標與機器人目前角度，將輪型機器人目前的位置與原點的座標相減，可得到相對於起始位置的橫坐標、縱座標及角度。接著，將得到的機器人座標資訊及角度資訊傳送至微電腦控制板進行處理，再透過本論文所提出的控制器來控制機器人左右輪轉速以抵達目的地。

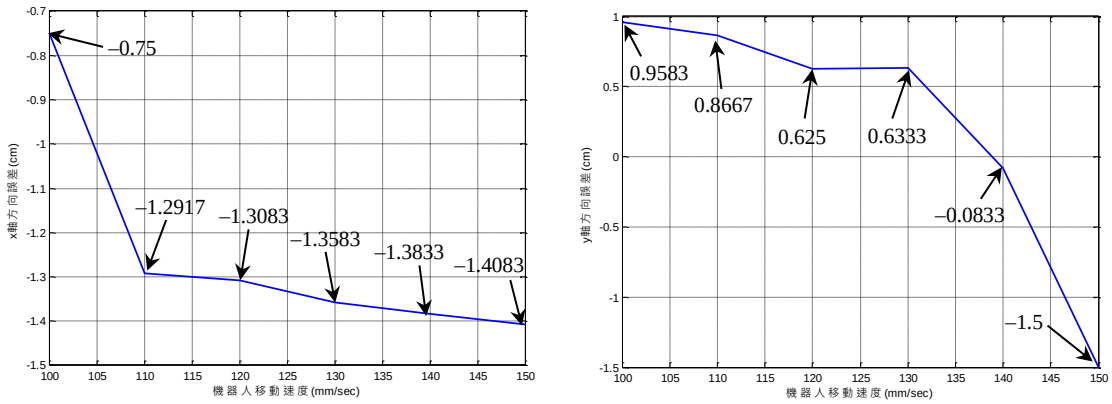
在進行整合型實驗之前，我們先實驗單獨只使用里程計，其機器人的定位效果。表 1 及表 2 為機器人移動 2.5 公尺時，機器人移動速度 v 與 x 軸/y 軸方向的誤差數據表。圖 8 為經由 12 次實驗，我們獲得機器人移動速度與其對應的 x 軸/y 軸方向平均誤差值。我們可以發現累積誤差會因為機器人移動速度而增加，因此若單獨只使用里程計來做定位，則機器人將會在到達終點前即會撞上牆壁亦或是碰撞到其他障礙物。

表 1、機器人移動速度 v 與 x 軸方向誤差數據表

	移動時驗					
速度序次	100 mm/s	110 mm/s	120 mm/s	130 mm/s	140 mm/s	150 mm/s
1	-2.1 cm	-0.9 cm	0 cm	-1.2 cm	0 cm	-2 cm
2	2.2 cm	-1.1 cm	-2.3 cm	-2 cm	-2 cm	1 cm
3	2.5 cm	-1.2 cm	-2.1 cm	-2 cm	-1 cm	-3 cm
4	-3 cm	-1 cm	-1.1 cm	-1 cm	-1 cm	-2.8 cm
5	0.5 cm	-2 cm	-0.5 cm	-1.8 cm	-5 cm	0 cm
6	-0.9 cm	0 cm	-1 cm	0 cm	-1.9 cm	-2.7 cm
7	-1.7 cm	-1.5 cm	-1.2 cm	0 cm	-0.5 cm	0 cm
8	0 cm	0 cm	-2 cm	-0.8 cm	-1.5 cm	-1.3 cm
9	-2.1 cm	-2.4 cm	-1.3 cm	-3 cm	-2.2 cm	-0.5 cm
10	-1.2 cm	-1.7 cm	-1.3 cm	-0.5 cm	1 cm	-1.7 cm
11	-2.5 cm	-2.2 cm	-1.9 cm	-2 cm	1 cm	-0.9 cm
12	-1 cm	-1.5 cm	-1 cm	-2 cm	-3.5 cm	-3 cm

表 2、機器人移動速度 *v* 與 y 軸方向誤差數據表

速度序次	移動實驗					
	100 mm/s	110 mm/s	120 mm/s	130 mm/s	140 mm/s	150 mm/s
1	1 cm	-0.3 cm	1 cm	0.3 cm	0 cm	0 cm
2	1 cm	0.7 cm	0.5 cm	1.3 cm	1 cm	-1 cm
3	0 cm	0.7 cm	0 cm	1.3 cm	0 cm	-1.5 cm
4	1 cm	-0.3 cm	1 cm	0.3 cm	-1 cm	-4 cm
5	1 cm	0.7 cm	0 cm	-0.7 cm	-1 cm	-1 cm
6	1 cm	1.7 cm	0 cm	0.3 cm	-4 cm	-2 cm
7	1 cm	1.7 cm	0.5 cm	-0.7	1 cm	-2 cm
8	0 cm	1.7 cm	2 cm	1.3 cm	1 cm	-1 cm
9	2 cm	0.7 cm	1.5 cm	1.3 cm	0 cm	-1 cm
10	1.5 cm	1.7 cm	0 cm	0.3 cm	0.5 cm	-2 cm
11	1 cm	0.7 cm	-1 cm	1.3 cm	1.5 cm	-1 cm
12	1 cm	0.7 cm	2 cm	1.3 cm	0 cm	-1.5 cm



(a) (b)

圖 8、機器人移動速度與 x/y 軸方向平均誤差值 (a) x 軸方向平均誤差 (b) y 軸方向平均誤差

在群組機器人實驗中，機器人會將自身之外的機器人視為障礙物，透過本論文提出的基於 D*Lite 演算法來規劃行走軌跡，讓機器人沿著軌跡前往終點。圖 9 為群組機器人的導航實驗截圖。

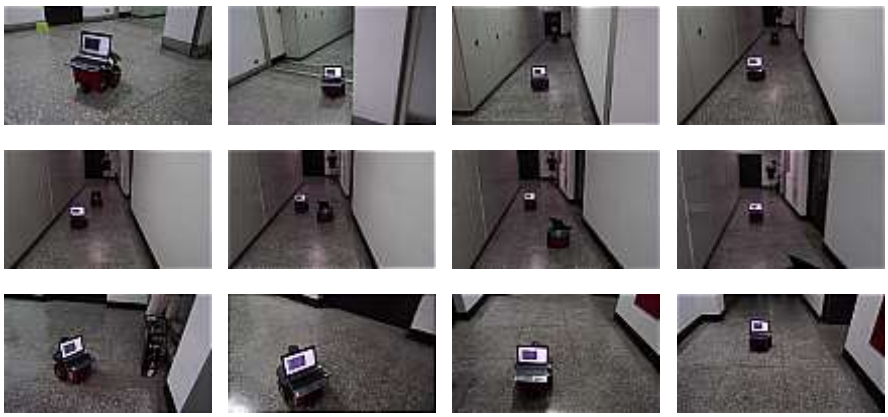


圖 9、群組機器人的導航實驗截圖

本實驗中所使用的參數包含路徑規劃中的安全距離$\delta = 5\,\text{pixel}$，機器人定位所用到的粒子數$\eta = 2000$以及擾動半徑$\gamma = 5\,\text{pixel}$，路徑追蹤控制器中地圖上每單位 pixel 所對應的距離$\rho = 55.24mm$，倒階控制法中的α依據轉彎與否分為兩組，直線型選取$\alpha_1 = 5.7928$、$\alpha_2 = 30.0028$、$\alpha_3 = 51.0172$，轉彎型選取$\alpha_1 = 5.2602$、$\alpha_2 = 0.0885$、$\alpha_3 = 56.2783$以及輪速限制$V_{Max} = 450\,mm/s$。圖 10 為群組機器人移動軌跡顯示圖。圖 11 及圖 13 分別顯示 Robot #1 及 Robot #2 的實際移動方位與參考移動方位。由於 Robot#1 與 Robot#2 一開始在追蹤參考軌跡時，為了減少因快速移動所造成的震盪現象以維持機器人移動的穩定度，因而導致一開始的誤差較大。另外，Robot#2 目前並沒有搭載電子羅盤，只能依靠輪速差以及定位演算法來判斷方位，因而間接地導致移動軌跡比 Robot#1 更加緩慢地收斂。在未來改善方面，我們除了裝設電子羅盤以改善機器人定位與追蹤參考軌跡等功能外，亦可提出更有智慧地調整系統參數，以提升整體運作效能。圖 12 及圖 14 分別顯示 Robot #1 及 Robot #2 的移動速度與角速度。

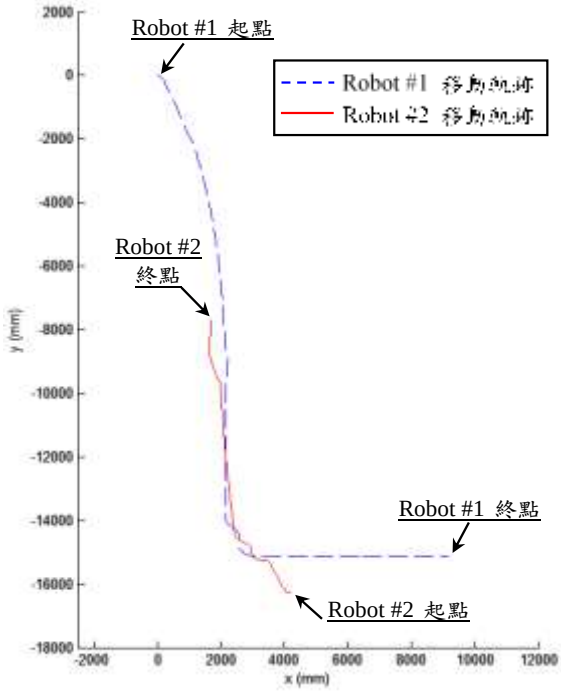
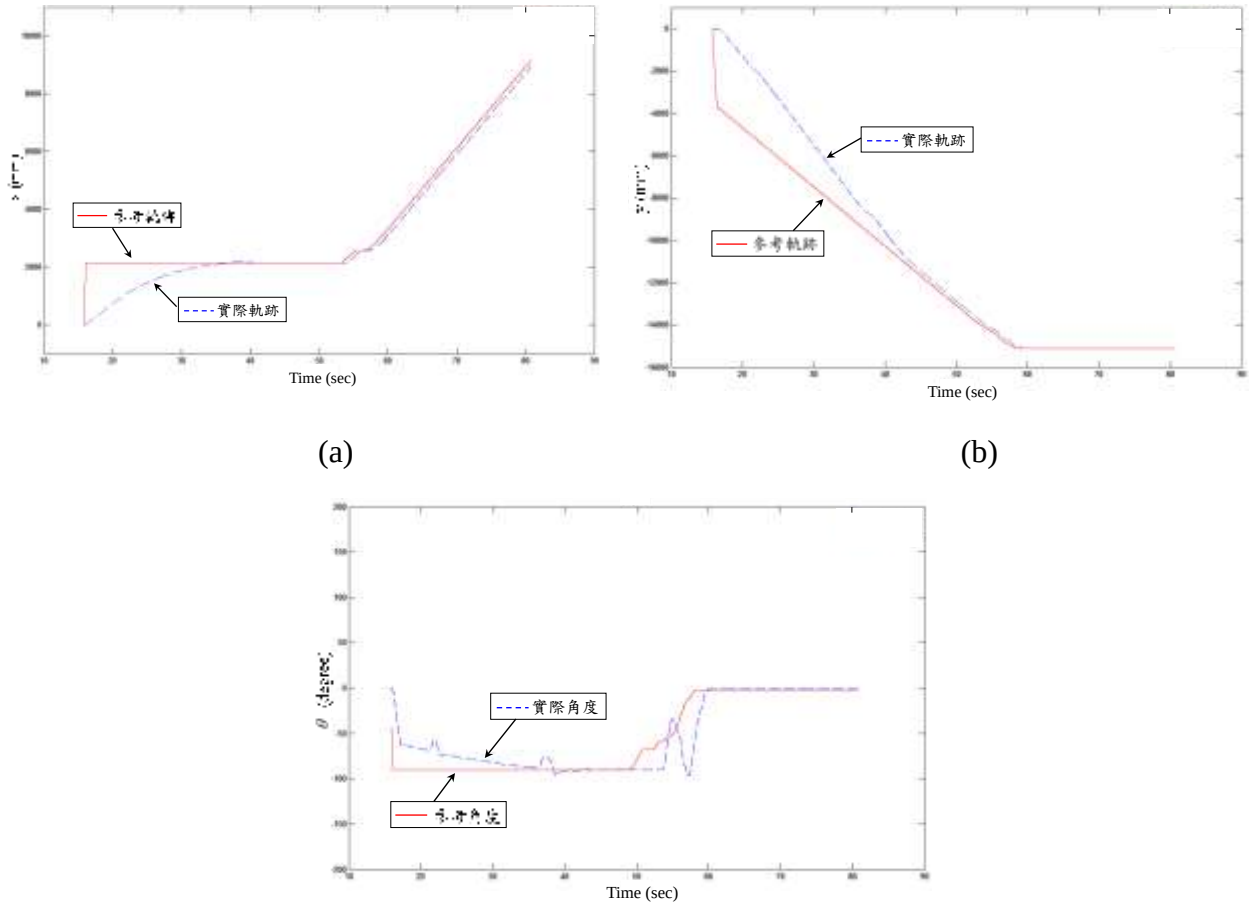


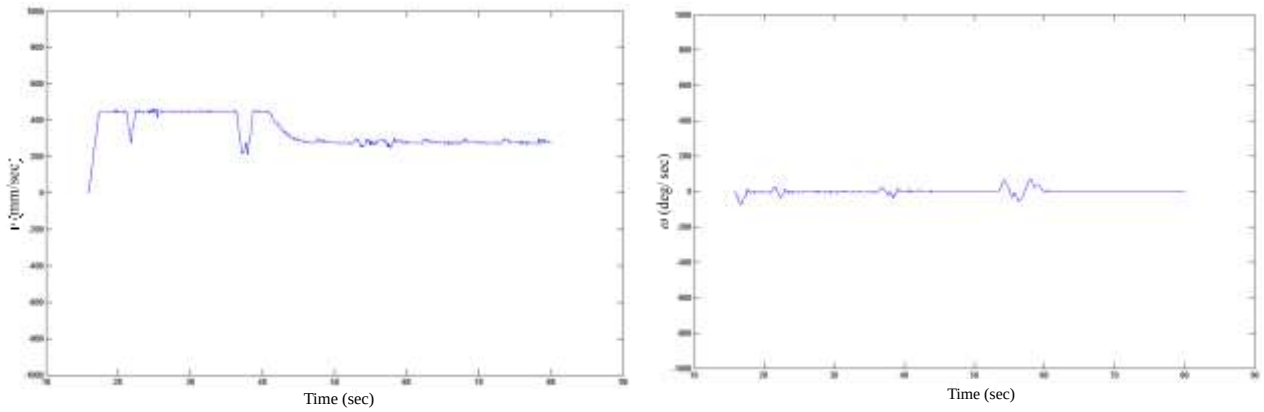
圖 10、群組機器人移動軌跡



(a) (b)

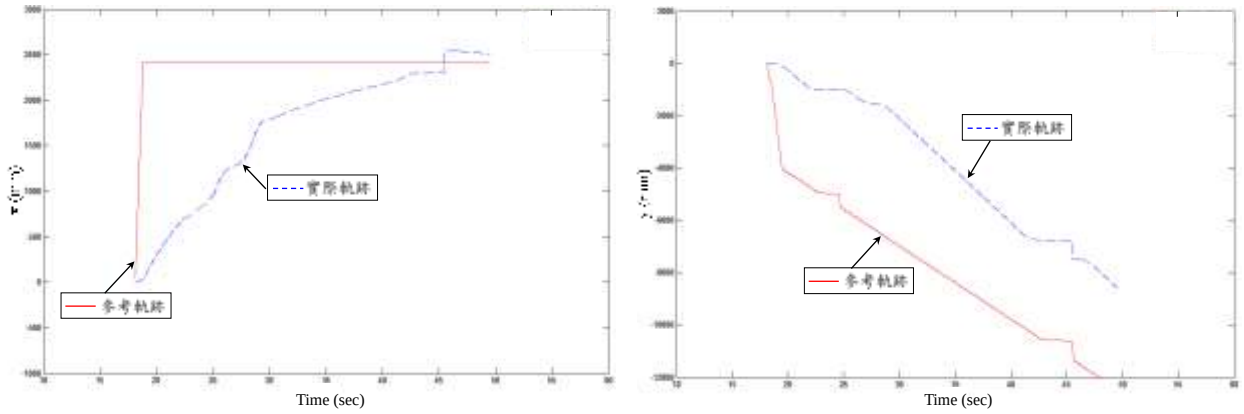
(c)

圖 11、Robot #1 的實際移動方位與參考移動方位 (a) x 軸移動軌跡 (b) y 軸移動軌跡 (c) 移動角度

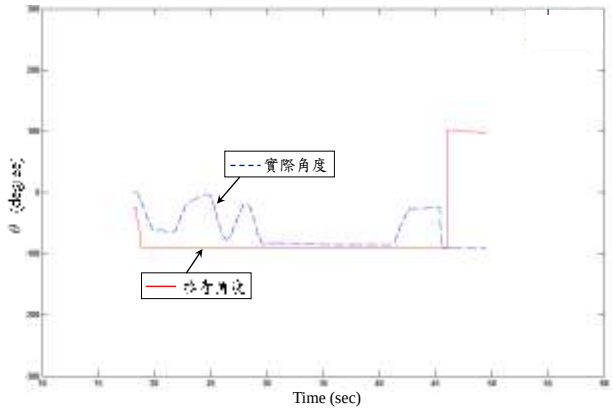


(a) (b)

圖 12、Robot #1 的移動速度與角速度 (a) 速度 v (b) 角速度 ω



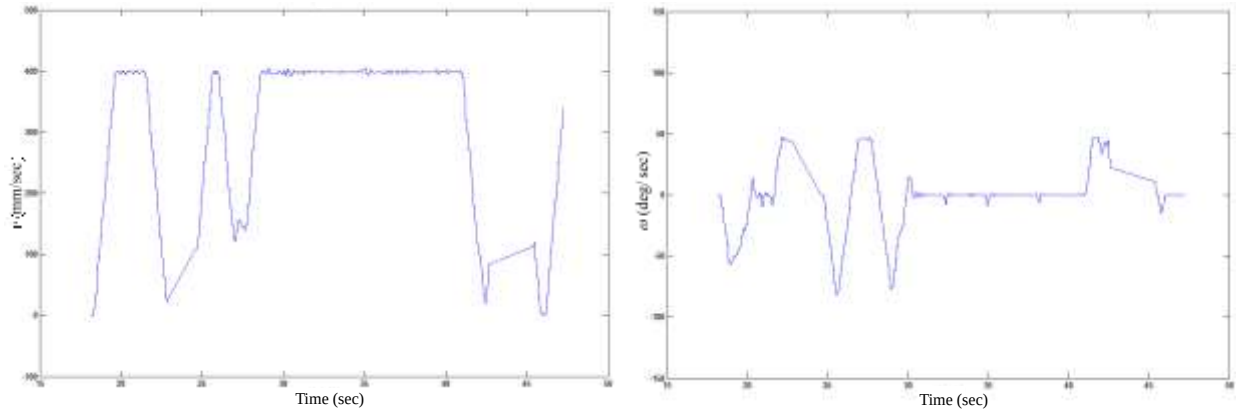
(a) (b)



(c)

圖 13、Robot #2 的實際移動方位與參考移動方位 (a) x 軸移動軌跡 (b) y 軸移動軌跡

(c) 移動角度



(a) (b)

圖 14、Robot #2 的移動速度與角速度 (a) 速度 v (b) 角速度 ω

伍、結論

在本篇論文當中，我們整合了機器人導航功能中的定位、路徑規劃以及路徑追蹤控制演算法，並且透過融合蒙地卡羅定位與里程計資訊的方法，解決了里程計回授產生的累計誤差，讓機器人能夠進行更長距離的移動。此外，我們運用自行架設的 MySQL 伺服器，作為群組機器人資訊的交換中心，以此得知機器人之間彼此的位置，群組中的每一台機器人皆視其他機器人為障礙物，規劃一條能避開彼此碰撞的路徑，讓機器人能沿著規劃好的路徑進行移動。在實驗方面，我們使用兩台輪型機器人同時執行導航任務。當兩台機器人互相接近時，機器人會透過 MySQL 伺服器得知彼此的位置，並規劃出避開碰撞的路徑。在未來的研究方面，我們將可透過多台單板電腦來平行處理定位、路徑規劃以及路徑追蹤控制演算法，讓每台單板電腦的任務較為單純，使得執行效率能夠大幅提升。

參考文獻

Al-Mutib, K., AlSulaiman, M., & Mattar, E. (2011). D* lite based real-time multi-agent path planning in dynamic environments. *2011 Third International Conference on Computational Intelligence*, 170-174.

Antonini, P., Ippoliti, G., & Longhi, S. (2006). Learning control of mobile robots using a multiprocessor system. *Control Engineering Practice*, 14, 1279-1295.

Arulampalam, M. S., Maskell, S., Gordon, N., & Clapp, T. (2002). A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Trans. on Signal Processing*, 50(2), 174-188.

Chen, C.-Y., Li, T.-H., Yeh, Y.-C., & Chang, C.-C. (2009). Design and implementation of an adaptive sliding-mode dynamic controller for wheeled mobile robots. *Mechatronics*, 19(2), 156-166.

Chien, Y.-H., Wang, W.-Y., & Leu, Y.-G. (2015). On-line hybrid intelligent tracking control for a class of nonaffine multivariable systems. *International Journal of Fuzzy Systems*, 17(1), 39-52.

Chien, Y.-H., Wang, W.-Y., Li, I-H., Lian, K.-Y., & Lee, T.-T. (2012). Hybrid intelligent output-feedback control for trajectory tracking of uncertain nonlinear multivariable dynamical systems. *International Journal of Fuzzy Systems,* 14(1), 141-153.

Conkur, E. S. (2005). Path planning using potential fields for highly redundant manipulators. *Robotics and Autonomous Systems*, 52(2-3), 209-228.

Das T., & Kar, I. N. (2006). Design and implementation of an adaptive fuzzy logic based controller for wheeled mobile robots. *IEEE Trans. on Control Systems Technology*, 14(3), 501-510.

De La Cruz C. C., & Carelli, R., (2006). Dynamic modeling and centralized formation control of mobile robots. *32th Annual Conference of the IEEE Industrial Electronics Society*, 3880-3885.

Dellaert, F., Fox, D., Burgard, W., & Thrun, S. (1999). Monte Carlo localization for mobile robots. *IEEE International Conference on Robotics and Automation*, 1322-1328.

Drawil, N. M., Amar, H. M., & Basir, O. A. (2013). GPS localization accuracy classification: A context-based approach. *IEEE Trans. on Intelligent Transportation Systems*, 14(1), 262-273.

Ferguson, D., & Stentz, A. (2005). The delayed D* algorithm for efficient path replanning. *2005 IEEE International Conference on Robotics and Automation*, 2045-2050.

Fierro, R., & Lewis, F. L. (1995). Control of a nonholonomic mobile robot: Backstepping kinematics into dynamics. *34th Conference on Decision and Control*, 3805-3810.

Fox, D., Burgard, W., & Thrun, S. (1999). Markov localization for mobile robots in dynamic environments. *Journal of Artificial Intelligence Research*, 11, 391-427.

Ge, S. S., & Cui, Y. J. (2002). Dynamic motion planning for mobile robots using potential field method. *Autonomous Robots*, 13(3), 207-222.

Hart, P. E., Nilsson, N. J., & Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. *IEEE Trans. on Systems Science and Cybernetics*, 4(2), 100-107.

Ho, Y. C., & Lee, R. (2001). A Bayesian approach to problems in stochastic estimation and control. *IEEE Trans. on Automatic Control*, 9(4), 333-339.

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1), 35-45.

Kao, Y.-F., Chien, Y.-H., Li, I-H., Wang, W.-Y., & Lee, T.-T. (2013). Design and implementation of adaptive dynamic controllers for wheeled mobile robots. *IEEE International Conference on System Science and Engineering*, 195-199.

Koenig, S., & Likhachev, M. (2005). Fast replanning for navigation in unknown terrain. *IEEE Trans. on Robotics*, 21(3), 354-363.

Li, I-H., Chen, M.-C., Wang, W.-Y., Su, S.-F., & Lai, T.-W. (2014). Mobile robot self-localization system using single webcam distance measurement technology in indoor environments. *Sensors*, 14(2), 2089-2109.

Li, Y., Wang, Z., & Zhu, L. (2010). Adaptive neural network PID sliding mode dynamic control of nonholonomic mobile robot. *2010 IEEE International Conference on Information and Automation*, 753-757.

Martins, F. N., Celeste, W. C., Carelli, R., Sarcinelli-Filho, M., & Bastos-Filho, T. F. (2008). An adaptive dynamic controller for autonomous mobile robot trajectory tracking. *Control Engineering Practice*, 16, 1354-1363.

Paull, L., Saeedi, S., Seto, M., & Li, H. (2014). AUV navigation and localization: A review. *IEEE Journal of Oceanic Engineering*, 39(1), 131-149.

Pencheva, T., Atanassov, K., & Shannon, A. (2009). Modelling of a stochastic universal sampling selection operator in genetic algorithms using generalized nets. *10th Int. Workshop on Generalized Nets*, 1-7.

Rashid, A. T., Frasca, M., Ali, A. A., Rizzo, A., & Fortuna, L. (2015). Multi-robot localization and orientation estimation using robotic cluster matching algorithm. *Robotics and Autonomous Systems*, 63(1), 108-121.

Rodríguez-Araújo, J., Rodríguez-Andina, J. J., Fariña, J., & Chow, M.-Y. (2014). Field-programmable system-on-chip for localization of UGVs in an indoor ispace. *IEEE Trans. on Industrial Informatics*, 10(2), 1033-1043.

Shim, J.-H., & Cho, Y.-I. (2015). A mobile robot localization using external surveillance cameras at indoor. *Procedia Computer Science*, 56, 502-507.

Wang, C., et al. (2015). Path planning of automated guided vehicles based on improved A-star algorithm. *2015 IEEE International Conference on Information and Automation*, 2071-2076.

Wang, L. X., & Mendel, J. M. (1992). Fuzzy basis functions, universal approximation, and orthogonal least squares learning. *IEEE Trans. on Neural Networks*, 3(5), 807-814.