圖書館自動 RFID 書籍盤點與自走車定位技術開發

The location finding of an autonomous cart for library RFID-book inventory

張鐘仁、蔡承祐、潘宗諭、侯凱育

國立聯合大學 資訊工程學系

苗栗市南勢里聯大 2 號

{U0524017,U0524003,U0524034,U0524039}@smail.nuu.edu.tw

摘要

圖書館管理員到書櫃前盤點書籍確認擺放狀況，過程費時且費力，若有自動盤點書籍的自走車，必能減少人力與時間，因此我們與機械、電機與資管合力推出一套自動盤點書籍的自走車系統。

為了讓自走車掌握目前位置，使用光學雷達做室內定位，搭配路徑規劃走到指定書櫃。走進書櫃之間前，用相機取相做影像校正，安全地走進去後，由超音波測距接管，保持左右間距防止撞到書櫃。書櫃底下貼上存有書櫃編號的 QR code，當 QR code 掃描器讀到時將重新定位，若到達指定編號將進行書籍盤點。

進入盤點程序後，控制升降梯將 RFID 掃描器升降至指定高度，配合自走車來回移動，將書本編號讀回並傳送到資料庫核對，確認書本擺放狀況。

透過上述系統能節省盤點書籍時間，幫助管理員快速確認書本擺放狀況。

關鍵詞：書籍盤點、自走車、定位、影像校正。

一、緒論

傳統盤點書本的方法是管理員親自走到書櫃前透過肉眼確認書本編號，檢查有無擺放錯誤的情況，由於書本編號字體小的關係，這樣的做法除了費時外，也不利於視力不好的人處理。隨著進入工業 4.0 的時代，在各式各樣的產業裡都能看見無人車技術的身影，無人車透過模擬人類的思考模式與行為，漸漸取代人類完成簡單的作業，如：掃地機器人、無人搬運車…等，為人們帶來更為便利的生活。如果將自走車技術引進到書櫃，協助管理員進行書籍盤點，必定能減少許多人力在盤點上。

圖書館登記借閱書籍時使用的是 RFID 掃描器，掃描器讀取書本編號後透過資料庫系統處理快速得到相關資訊。如果將這個技術搭配無人載物車技術的話，就能實現讓無人載物車自動盤點書籍，為管理員省時省力。

二、系統架構

2.1 系統架構

自走車運作時的主控中心，在樹莓派上接收與發送訊號：

A. Dijkstra Node 所計算的車子需要的轉動訊號。

B. LIDAR Node 所推估的 LIDAR 定位座標。

C. QR code Node 得到 QR code 資訊。

D. Ultra Node 計算後的超音波測距數據。

E. Camera Node 計影像處理後的校正資訊。

F. 請求 Control Node 執行升降梯控制與掃書程序，並等待接收完成訊號。

在自走車出發前，主控程式會接收到 Dijkstra Node 所發送的轉動訊號，將轉動訊號透過 UART 的方式傳給控制自走的 Arduino，開始自走車的移動。移動過程中主要是接收來自 LIDAR Node 的定位座標，判斷是否到達終點，若成功到達將會進入下一個動作；否則代表走錯，需要重新規劃路徑後再走到終點。

自走車在進入車道前(兩書櫃之間)，透過 Camera Node 進行影像處理後得知歪斜情況，主控程式再慢慢修正到自走車以最正的方向走入車道。而進入車道後，為了防止自走車走歪而撞向書櫃，Ultra Node 會計算與車道間的距離，傳送應修正的方向訊號給主控程式來修正自走車行走方向。

當 QR code Node 讀到 QR code 時，會將 QR code 的資訊傳給主控中心做判斷，若已經走偏會重新規畫路徑。理論上 QR code 的位置是固定不變的，因此每當讀取到 QR code 資訊主控程式就會對現在座標做一次確認，若座標位置差別很多，將會重新修正現在座標。

當主控程式判斷為正確的 QR code 位置時，將會傳送訊號給 Control Node 進行升降梯、伸

縮臂控制與掃書的程序。當升降梯、伸縮臂與 RFID 準備就緒時 Control Node 回傳動作完成，主控程式控制自走車在車道間來回走動，直到掃書程序結束收到來自 Control Node 傳送的結束訊號，主控程式控制自走車停下來後計算前往下一個 QR code 的路徑。

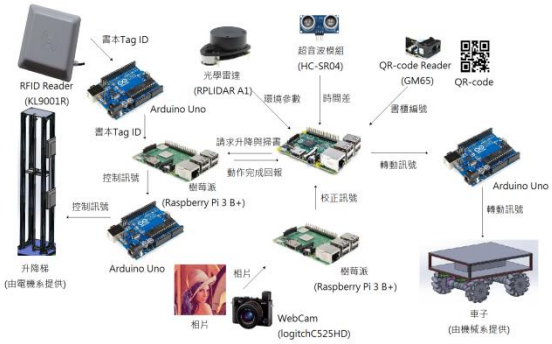

圖 1：硬體架構圖

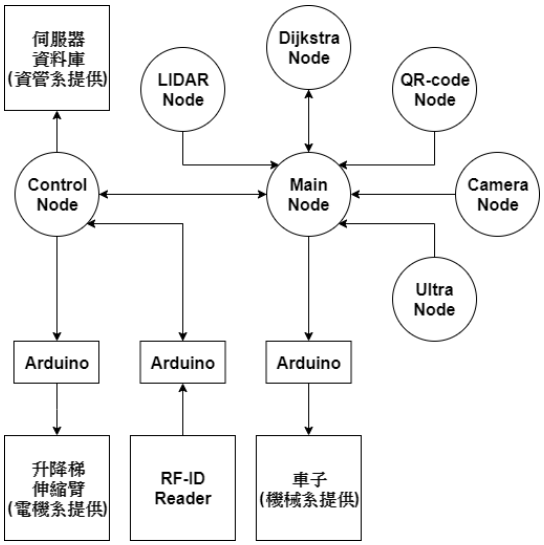

圖 2：軟體架構圖

2.2 硬體設備

A. 樹莓派(Raspberry Pi 3 B+)：基於 Linux 的單晶片電腦，Wi-Fi、藍芽功能。為自動導航的主控中心。

B. Arduino Uno：基於 ATmega328P 的開發板。用於車體輪子正反轉、轉速控制。

C. 光學雷達(RPLIDAR A1)：用雷射三角測距對周圍環境的 3 全方位掃描測距，從而獲得環境的輪廓圖。用於室內定位與地圖建置。

D. 超音波模組(HC-SR04)：是由超音波發射器、接收器和控制電路所組成。當它被觸發的時候，聲波從離它最近的物體接收回音。用於車體行走間的防撞。

E. QR code Reader(GM65)：接收 QR code 資訊回傳給 Raspberry Pi。

F. RFID Reader(KL9001R)：UHF 超高頻電子標籤一體機，掃描書本編號透過 Arduino 傳給 Raspberry Pi。

G. WebCam (Logitech HD 網路攝影機 C525)：屬閉路電視的一種，一般具有藉由鏡頭採集圖像後，然後藉由 USB，輸入到電腦後由軟體再進行圖像還原。

2.3 開發環境與工具

A. ROS：機器人作業系統(Robot Operating System)：開源的作業系統，它也提供一些工具和庫用於獲取、建立、編寫和執行多機融合的程序。[1]

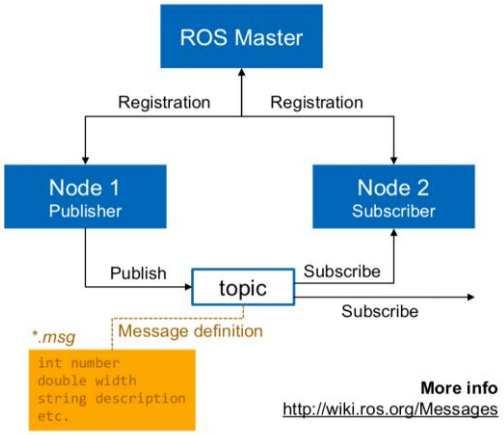

圖 3：ROS 環境下 Node 之間的溝通示意圖

B. OpenCV ： OpenCV(Open Source Computer Vision Library)跨平台的電腦，可以在商業和研究領域中免費使用。可用於開發即時的圖像處理、電腦視覺以及圖型識別程式。

C. Arduino Software IDE：以 Java 編寫的跨平台應用軟體。

D. C++：是一種使用廣泛的程式設計語言，支援多重程式設計模式，為當今主流程式設計語言中最複雜的一員。

E. Python：直譯式、進階編程、通用型程式語言，設計哲學強調程式碼的可讀性和簡潔的語法。相比於 C++或 Java，Python 讓開發者能夠用更少的代碼表達想法。

三、技術探討與執行方式

3.1 研究方法

(1) 自走車控制

A. 路徑規劃

使用 Dijkstra 演算法裡面的最短路徑規劃，將自走車現在座標當作起始點與目標書櫃的 QR code 座標當作終點在地圖上規劃出一條路線，再將路徑轉為自走車移動訊號。[2]

B. 超音波

利用超音波測距，其執行程序如下：取得距離資料後，將其資料格式轉換成「公分」度量，每秒可以取得 1000 筆資料，由於距離資料變化非常大，將利用濾波器，消除雜訊資料，目前採用每 30 筆資料做平均運算，而利用超音波資料判斷，進而控制自走車的方向，現行架設四個超音波接收器。

自控車移動時，會造成接收時訊號不穩，參考圖 6，利用濾波讓超音波訊號偏差減少。在進行平均前，判斷接收到的訊號是否為錯誤訊號，判斷方法為單筆訊號出現大幅度改變如圖 6 紅色圈所示則為錯誤訊號，如果為錯誤訊號便不會採用，每 30 筆資料做平均運算，平均運算可以解決小幅度變化如圖 6 黃色圈所示的雜訊資料，以此兩中方式作用於慮波已降低距離資料帶來的偏差。[3]

現行架設四個超音波，其距離判斷規則如下：

a. 一開始太靠左牆壁(1,2 小於 20 公分)：順時針旋轉，ROS 代號 1

b. 一開始太靠右牆壁(0,3 小於 20 公分)：逆時針旋轉，ROS 代號 2

c. 車頭偏左(1,3 小於 20 公分，0,2 大於 25 公分)： 順時針旋轉，ROS 代號 1

d. 車頭偏右(0,2 小於 20 公分，1,3 大於 25 公分)： 逆時針旋轉，ROS 代號 2

e. 車頭正常(0,1,2,3 介於 20 公分~25 公分內)：直走，ROS 代號 0

(備註：如果只有一邊牆壁，另一邊超音波感測器會自動忽略，只使用兩個感測器進行判斷，目前情況兩個感測器便能直走，Arduino 需設 delay 讓調整更為順暢)

圖 4：超音波節點程序

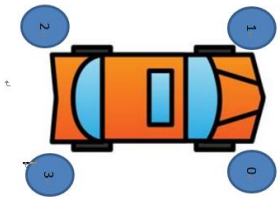

圖 5：超音波感應器裝設示意圖

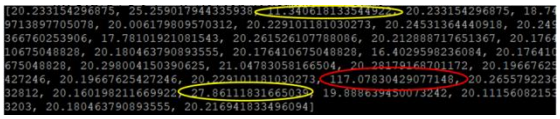

圖 6：超音波偏差訊號

C. 邊緣檢測

取像後第一件事，就是作邊緣檢測，先將圖片轉成灰階用高斯模糊(Gaussian Blur)過濾掉雜訊(降噪作用)，先後取得圖片每個像素的梯度值和方向，再找有可能的邊緣及確定的邊緣，再來考慮這些 edge 哪些能作為判斷的特徵點，方便做下一步的影像處理。[4][5][6]

D. SURF

由於 SURF 演算法在可以輕易的捕捉稜角分明或材質特殊的物體作為特來作比對，因此決定以此種方式來進行特徵點的捕捉。在經過 webcam 畫面上的實際特徵點與資料庫原圖作比對，計算出目前位置與資料庫的角度差異，校正後，安全的進入車道。目前影像處理是在樹莓派上進行運算，再以 rxtx 的方式傳至中控端做溝通，判斷自走車該如何校正。[7]

圖 7：影像處理後結果

E. 麥克輪正反轉控制。

關於自走車基礎控制方面原為機械系提供，包括前進、後退、左平移...等指令，但這些動作的配合都需要靠 Arduino 還有附設的遙控器手動進行控制(圖 8)。

而本研究中我們則使用 rxtx 傳輸樹莓派與 Arduino 之間的控制，Arduino 部分則設定指令，控制四輪轉速與行走方式，讓樹梅派能以指令控制來調整自走車的行走(圖 9)，不再需要倚賴遙控器作控制。細節方面，透過四輪的轉速不同的配置差異，車體就能完成原先需要依靠遙控器手動控制的動作，方便以程式控制自走車行進。

圖 8：自走車遙控器(由聯合大學機械系提供)

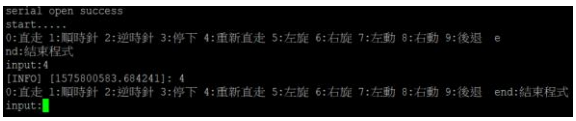

圖 9：樹莓派控制程式介面

(2) 定位系統

A. LIADA

使用開源的 Hector Slam 演算法將數據轉換成空間座標並存到佇列裡面進行濾波，轉換完成後透過 ROS 的 RViz 將周圍環境顯示在螢幕上，以灰色為空曠的地方，黑色為長時間不動的物體，如：牆面、障礙物...等。使用 tf 功能就會在 RViz 上顯示現在 LIDAR 在地圖上的定位位置。透過接收 /slam_out_pose 的 topic 數值算出在地圖上的相對座標。[8][9]

B. QR code

本次使用的 QR code 掃描器型號是 GM65，利用 USB 與樹莓派連接，使用手冊進行環境設定。 [10]

利用 QR code 掃描器讀取印製好的 QR code，QR code 以書櫃代號印製，利用掃描器讀取進 Python 做處理，讀取到 QR code 停下(自控車行走訊號 3)，再利用 ROS 平台傳遞資料，建立 QR code 節點，將字串的代號發佈到 topic，Subscriber 讀取到代號後，會利用代號轉換成陣列來確定目前位子，把讀取到的櫃號先轉成 String，再利用 ROS 平台傳送。

C. WiFi

使用位置指紋進行定位通常有兩個階段：離線階段和在線階段。離線階段，為了採集各個位置上的指紋，構建一個數據庫，需要在指定的區域進行繁瑣的勘測，採集好的數據有時也稱為訓練集。在線階段，系統將估計待定位的移動設備的位置。而現在所使用的指紋為 RSS，利用接收器接收各點 AP 的 WiFi 訊號強度，與位置建立數據庫。[11]

離線階段：位置指紋關係通常在離線階段建立而成，一個區域被矩形網格所覆蓋，有 2 個 AP 點，如圖 10，每個網格經過一段時間收集來自 AP 的 RSS 樣本均值，把這些網格點做標與指紋組成一個數據庫，如圖 11，之後就能以數據庫推估移動設備的位置。

在線階段：一個移動設備處於一個區域中，但我們無法具體知道他的位置，有時甚至不會剛好在網格點上。先測量來自各個 AP 的 RSS，想要確定設備的位置就要找出與數據庫最匹配的指紋。

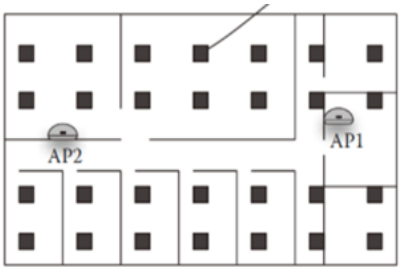

圖 10：網格與 AP 示意圖

	A	B	C	D
1		CSIE_404	nuu	local
2	0	-46	-62	0
3	1	-46	-64	0
4	2	-46	-64	0
5	3	-46	-62	0
6	4	-48	-62	0
7	5	-48	-64	0
8	6	-48	-64	0
9	7	-48	-62	0

圖 11：指紋與網格座標數據庫

(3) 書籍盤點

A. RFID

透過此型號 KL9001R 的手冊，運用命令數據塊的傳遞進行模式的切換，Arduino 將命令數據塊傳送到 RFID，RFID 判斷為正確命令時會回傳相對應的響應數據塊回饋，這樣就完成一次通訊。

將模式切換成自動掃描模式，可以保持掃描狀態，並且在 Arduino 進行對響應數據塊的字串處理，可以得到完整的掃書號，並 print 出，完成 Arduino 與 RFID 的溝通。

B. RFID 與升降機構的控制

運用樹梅派與 Arduino 的溝通達到硬體的控制。

RFID 的溝通：使用 USB 燒路線，運用樹梅派內的 tty 端口連接，程式內部使用 RS232 連接方式達成與 Arduino 資料的傳輸，再進資料的處理，使用 http 的溝通模式傳送到資料庫。

升降梯的溝通：使用 GPIO 腳位與 Arduino 的 rxtx 端進行溝通，透過樹梅派的字串命令與 Arduino 的字串回傳達到控制模式的轉換與狀態的確認，完成升降梯控制的部分。

四、結論

這次我們分別與聯合大學機械系(提供自走車)、電機系(提供升降機構)以及資管系(提供資料庫)合作，此次專題讓我們獲益良多，不過在專題完成性上面，跟原本預想的狀況還是有點落差，例如：QR code 會因為車速過快導致無法掃到，或因為自走車剛好在進行超音波校正，造成少概率沒掃到 QR code 的狀況；超音波校正會因為負重的增加，會使車體校正出現問題，或因為路面不平與距離牆壁過近，導致自走車的車體來不及校正回正軌，QR code 與超音波校正的配合也需要多次的嘗試才能找到符合車體負重與地形的適當車速，逐漸才能達到自走車的需求；影像處裡的部分，因為在環境方面因為跟圖書館的實際環境不同，原本預計要拍一張圖片就能直接做比對，但有時候一張照片的資訊量不夠時，車子可能就要停留的比較久，因為要補拍其他照，補充資訊量，比原先想的所花的時間還要更多；RFID 掃書部分在完整度方面遇到一些問題，像是 20 本書最後掃到的數量可能只有 10 本；LIDAR 會因為啟動時每次面向的都會有微小的誤差，長遠行走後誤差逐漸變大甚至影響定位系統判斷，雖然遇到這些問題，但是自走車是一塊很廣的領域，未來還有更多學習的地方。

參考文獻

[1] ROS wiki：http：//wiki.ROS.org/

[2] 路徑規劃 Dijkstra 演算法：https://zh.m.wikipedia.org/zh-tw/戴克斯特拉算法

[3] 曹永忠、許智誠、蔡英德，Arduino 超音波測距機設計，渥瑪數位有限公司，2014 年。

[4] John Canny , “A Computational Approach to Edge Detection,” “*IEEE Transactions on Pattern Analysis and Machine Intelligence*, ” vol. 8, no. 6, pp. 679-698, 1986.

[5] Tony Lindeberg, “*Edge detection and ridge detection with utomatic scale selection*,” vol. 30, no. 2, pp. 117-154, 1998.

[6] Richard O. Duda and Peter E. Hart, “*Use of the Hough Transformation To Detect Lines and Curves in Pictures*,” 1972

[7] 陳劭芃, 徐勝斌, 韓欽銓, 方志鵬, and 張陽郎, “巡邏機器人路徑規劃：以感應器資料與影像特徵比對方法,” 2013 全國計算機會議論文集, Dec.

[8] Hector Slam(2D 地圖建置)：https：//hollyqood.wordpress.com/2015/12/01/ROS-slam-2-hector-slam-2d 地圖建置/

[9] Stefan Kohlbrecher, Oskar von Stryk, Johannes Meyer and Uwe Klingauf, “*A flexible and scalable SLAM system with full 3D motion estimation*,” “*IEEE International Symposium on Safety, Security, and Rescue Robotics*,” pp1-5 , 2011.

[10] 杭州城章科技有限公司，GM65 條碼識讀模塊用戶設置手冊，2016 年 9 月。

[11] Hassan A. Karimi, “*Advanced Location-Based Technologies and Services*,”Published October 27, 2017.