

國立雲林科技大學機械工程系

碩士論文

Department of Mechanical Engineering

National Yunlin University of Science & Technology

Master Thesis

使用光學雷達與姿態量測系統之無人自走車導航研究

Autonomous Navigation of UGV Based on LiDAR and



指導教授：吳英正博士

Advisor: Ying-Jeng Wu, Ph.D.

中華民國 102 年 6 月

June, 2014

# 摘要

本論文提出一方法整合了全球衛星定位系統（GPS）、姿態量測系統（AHRS）與光學雷達（LiDAR），以達成無人自走車自動導航與避開導航路徑上之障礙物的功能。使用 GPS 和 AHRS，車輛可得知本身與目標點之間的相對關係和方向資訊；使用光學雷達，車輛便可在導航的過程中量測週遭環境之變化。而利用這些感測器所測量到的資訊，便可規劃出讓自走車在導航至目標位置的同時也能夠避開障礙物的路徑。本論文開發的 Yun-Trooper II 使用四驅模型搖控車作為底盤，並使用 ARM 處理器架構之 Linux 嵌入式系統作為運算核心，而軟體方面則使用 C++ 進行計算和控制車輛運動。

**關鍵字：**GPS、AHRS、LiDAR、Linux、無人自走車



# Abstract

The thesis presents a method for the navigation and obstacle avoidance of an Unmanned Ground Vehicle (UGV), which integrated the Global Positioning System (GPS), Attitude and Heading Refrence System (AHRS) and Light Detection and Ranging sensor (LiDAR). With GPS and AHRS sensor, position of the target relative to vehicle itself and heading of the vehicle could be determined. And with a LiDAR sensor, the vehicle could also perceive the environment. From the data collected by these sensors, navigation algorithm could planning a collision-free path to avoid obstacles and reach the target. The thesis has developed an UGV named ‘Yun-Trooper II’, which is based on a 4-wheel-driven remote-controlled model car. It is equipped with an ARM-Based embedded Linux computer to make calculation and control, and all the program were developed by C++ programming language.

**Keywords:** GPS, AHRS, LiDAR, Linux, UGV

# 誌謝

感謝...



# 目錄

摘要 . . . . .	i
Abstract . . . . .	ii
誌謝 . . . . .	iii
目錄 . . . . .	v
圖目錄 . . . . .	vi
表目錄 . . . . .	viii
<b>一、緒論 . . . . .</b>	<b>1</b>
1.1 研究動機與目的 . . . . .	1
1.2 研究方法與文獻回顧 . . . . .	2
1.2.1 導航方向計算 . . . . .	2
1.2.2 路徑規劃 . . . . .	3
1.3 論文架構 . . . . .	4
<b>二、Yun-Trooper II 硬體架構介紹 . . . . .</b>	<b>5</b>
2.1 硬體架構簡介 . . . . .	5
2.2 運算核心 . . . . .	5
2.3 感測元件 . . . . .	6
2.3.1 位置姿態感測器 . . . . .	6
2.3.2 掃描式雷射測距儀 . . . . .	8
2.4 驅動元件 . . . . .	9
2.4.1 轉向伺服機 . . . . .	9
2.4.2 直流馬達 . . . . .	9
2.4.3 直流馬達驅動器 . . . . .	10
2.5 電源供應元件 . . . . .	11
2.6 通訊元件 . . . . .	12
<b>三、路徑規劃與控制法則 . . . . .</b>	<b>14</b>
3.1 目標方向計算 . . . . .	14
3.1.1 地理座標系統 . . . . .	14
3.1.2 測地線 . . . . .	17

3.1.3	區域座標系統	18
3.2	避障演算法	19
3.2.1	常見演算法介紹	19
3.2.2	Vector Field Histogram Plus	25
3.2.3	問題討論與補償	33
3.3	速度演算法	35
3.3.1	障礙物密度	35
3.3.2	障礙物接近率	37
3.3.3	碰撞偵測	39
3.4	控制法則	40
	參考文獻	42



# 圖目錄

圖 1.1	Yun Trooper 比較 . . . . .	2
(a)	Yun Trooper . . . . .	2
(b)	Yun Trooper II . . . . .	2
圖 2.1	Yun-Trooper II 硬體架構圖 . . . . .	6
圖 2.2	BeagleBone Black 開發板 . . . . .	7
圖 2.3	Xsens MTi-G AHRS 位置姿態感測器 . . . . .	7
圖 2.4	HOKUYO URG-04LX-UG01 掃描式雷射測距儀 . . . . .	9
圖 2.5	DS1015 數位伺服機 . . . . .	10
圖 2.6	RS-550VC-7525 直流馬達 . . . . .	10
圖 2.7	Pololu MD01B 直流馬達驅動器 . . . . .	11
圖 2.8	DC-DC LM2596 可調穩壓模組 . . . . .	11
圖 2.9	XBee PRO 無線通訊模組 . . . . .	12
圖 2.10	來源 : digipak.org . . . . .	12
圖 2.11	XBee 轉 TTL 轉接板 . . . . .	12
圖 2.12	XBee Explorer USB 轉接板 . . . . .	13
圖 3.1	路徑規劃目標 . . . . .	15
圖 3.2	導航流程圖 . . . . .	15
圖 3.3	ECEF 座標系統 . . . . .	16
圖 3.4	橢球座標系 . . . . .	16
圖 3.5	測地緯度示意圖 . . . . .	17
圖 3.6	測地線 . . . . .	18
圖 3.7	區域切平面示意圖 . . . . .	19
圖 3.8	目標方向相對於車輛之角度 . . . . .	19
圖 3.9	位能場 . . . . .	20
圖 3.10	極座標直方圖 . . . . .	21
圖 3.11	CVM 下的機器人運動軌跡 . . . . .	22
圖 3.12	環境資訊轉換 . . . . .	24

(a) 實際環境資訊 . . . . .	24
(b) 速度空間中的環境資訊 . . . . .	24
圖 3.13 動態視窗示意圖 . . . . .	24
圖 3.14 光學雷達量測示意圖 . . . . .	26
圖 3.15 極座標直方圖 . . . . .	26
(a) $d_i$ . . . . .	26
(b) $P_i$ . . . . .	26
圖 3.16 過濾結果比較 . . . . .	28
(a) 單一閾值 . . . . .	28
(b) 雙閾值 . . . . .	28
圖 3.17 邊界縮減 . . . . .	28
圖 3.18 轉向角度限制 . . . . .	29
圖 3.19 偵測直方圖示意圖 . . . . .	30
圖 3.20 直方圖轉換 . . . . .	30
圖 3.21 邊界分類 . . . . .	31
(a) 交疊 . . . . .	31
(b) 狹窄 . . . . .	31
(c) 寬廣 . . . . .	31
圖 3.22 無候選轉向角之環境 . . . . .	33
(a) 實際情況 . . . . .	33
(b) 光學雷達量測結果 . . . . .	33
圖 3.23 安全空間邊界誤判之環境 . . . . .	34
(a) 實際情況 . . . . .	34
(b) 光學雷達量測結果 . . . . .	34
圖 3.24 環境直方圖 . . . . .	35
圖 3.25 邊界縮減後之方向 . . . . .	36
(a) 計算後的邊界方向 . . . . .	36
(b) 實際之邊界方向 . . . . .	36
圖 3.26 最近距離之邊界縮減 . . . . .	37
圖 3.27 障礙物接近率示意圖 . . . . .	38
圖 3.28 第一階段碰撞偵測示意圖 . . . . .	40
圖 3.29 第二階段碰撞偵測示意圖 . . . . .	41

# 表目錄

表 2.1	BeagleBone Black 規格 . . . . .	6
表 2.2	MTi-G Performance Specification . . . . .	8
表 2.3	URG-04LX-UG01 規格 . . . . .	8
表 2.4	DS1015 數位伺服機規格 . . . . .	9
表 2.5	RS-550VC-7525 規格 . . . . .	10
表 2.6	XBee PRO 無線通訊模組性能規格 . . . . .	13
表 3.1	WGS84 大地基準參數 . . . . .	17



# 一、緒論

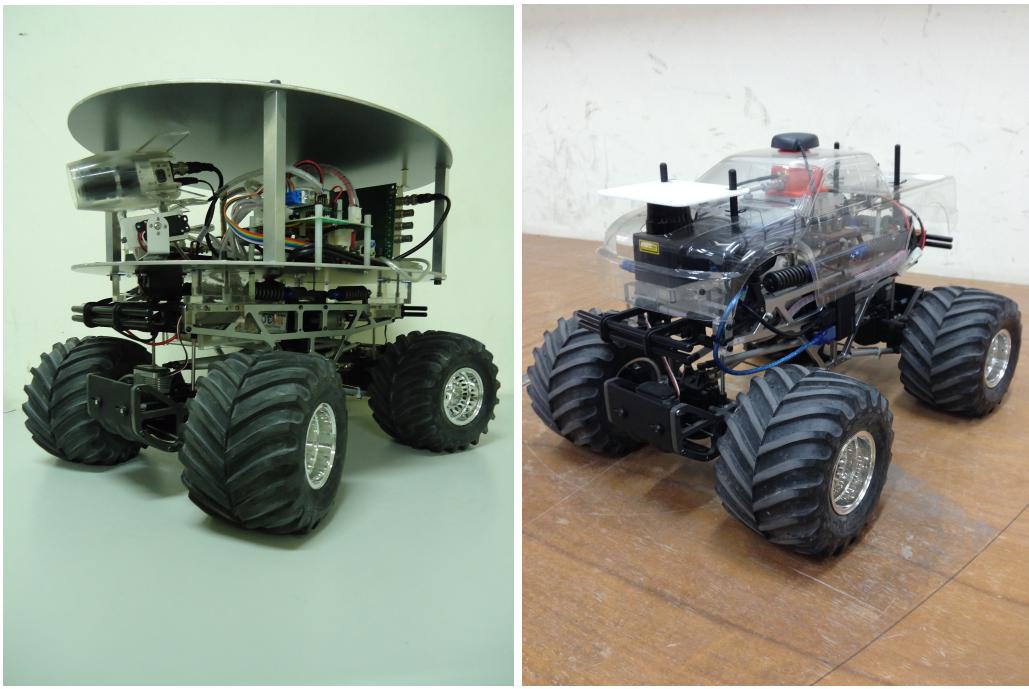
## 1.1 研究動機與目的

美國國防部高等研究計劃署（Defense Advanced Research Projects Agency，DARPA）分別於 2004、2005 與 2007 年舉辦了無人車輛競速大賽，各參賽隊伍運用各種感測器儀器，定位導航技術及影像處理技術等挑戰橫越沙漠和穿越城市。而 Google 公司也在 2011 年時也發表了自行研發的無人自走車，可見現今人工智慧的發展讓自主式無人載具成為目前廣為研究的課題，而基於這些影響，本實驗室也開始投入研究無人自走車的領域。

本實驗室目前已擁有一輛使用四驅越野搖控車作為底盤架構的 Yun-Trooper 實驗平台，先前也使用此平台完成了許多研究：陳維懋 [陳維懋 (2011)] 使用電腦視覺的方式來完成自動巡航，但由於電腦視覺容易受到環境干擾，呂明修 [呂明修 (2012)] 使用全球衛星定位系統 (GPS) 及慣性量測單元 (IMU) 取代電腦視覺系統來完成自動巡航。

第一代 Yun-Trooper，如圖 1.1a 所示，使用工業用電腦主機板搭配 Windows XP 作業系統作為決策中心，加上電腦視覺所需要的攝影機與影像擷取卡，其體積龐大且重量較重，造成巡航速度降低與耗電量增加。而本論文使用原先的底盤開發了第二代實驗平台 Yun-Trooper II，如圖 1.1b 所示，其使用嵌入式單板電腦搭配 Linux 作業系統，並且簡化了所需要的週邊配備，使得體積與重量大幅降低，開發成本也大幅下降。

本論文所開發的 Yun-Trooper II 裝置有無線通訊模組，因此使用者可在遠端以個人電腦或其它工具控制 Yun-Trooper II 的一切功能。使用者可以將導航目標點以經緯度的方式，直接手動輸入至電腦並傳送給 Yun-Trooper II；或是以搖控



(a) Yun Trooper

(b) Yun Trooper II

圖 1.1: Yun Trooper 比較

的方式將 Yun-Trooper II 事先移動至導航目標點，並使用車輛裝置的 GPS 記錄該點之經緯度，之後便可依導航目標點的傳送或記錄順序，利用姿態量測系統（AHRS）計算導航方向，依序導航至各個目標點。而導航過程中將使用光學雷達（LiDAR）量測環境的變化，同時規劃出安全的路徑以避開障礙物，避免碰撞。

## 1.2 研究方法與文獻回顧

本章將依 Yun-Trooper II 的研究方法及過程做出相關的文獻回顧。

### 1.2.1 導航方向計算

本論文提出的方法利用車輛本身位置與目標位置之間的相對關係，以及車輛的方位角，計算出相對距離及角度差，並做為車輛控制法則的參數。而 GPS 所量測到的經緯度是以 WGS84 大地基準（World Geodetic System 1984 Datum）為參考

座標系所得到的座標位置 [B.V.(2012)]。WGS84 為一橢球面 (Ellipsoid)，量測到的每一個經緯度都代表此橢球面上的一點 [El-Rabbany(2006)]，因此若要計算兩組經緯度之間的最短距離與方向，就必須考慮橢球面與圓球面之間的差異，才能得到正確的數值。這個最短距離就稱為測地線 (Geodesic) [Karney(2013)]。

自 19 世紀以來便有許多著名的數學家著手解決這個問題，例如勒讓得 (Legendre, A. M.)、貝索 (Bessel, F. W.) 及高斯 (Gauss, C. F.) 等。而由於現代電腦運算速度的提昇和數值方法的演進，Karney [Karney(2013)] 提出了適用於現代電腦的相關演算法。本論文使用這位作者所開發的 GeographicLib 函式庫 [Karney(2014)] 來做此方面的運算，增加運算效能以及縮短開發的時間。

### 1.2.2 路徑規劃



依路徑的規劃範圍可分為全域路徑規劃與部份路徑規劃 (Global/Local Path Planning)，有時也會以路徑規劃 (Path Planning) 與避障 (Obstacle Avoidance) 的名稱來區別 [Siegwart and Nourbakhsh(2004)]。全域路徑規劃是藉由地圖的輔助，根據不同的需求 (最短路徑、最少轉向、最容易通過等) 與地圖上已知的障礙物，像是建築物、走廊等，規劃出不同的路徑；部分路徑規劃則是利用感測器所量測到的環境資訊，對當下的環境變化做出反應，避開地圖上沒有出現或是隨時間改變的障礙物，例如隨機放置的紙箱、路人等等。而自走車必須要同時具備兩種路徑規劃演算法，才能夠得到最高的導航效率。

然而 Yun-Trooper II 並無法預先得知地圖資訊，因此在全域路徑規劃方面，如上一小節所述，只有計算兩點之間的直線距離與導航方向。在區域路徑規劃方面，大部分的演算法都是將感測器所得到的資訊，從實際的二維或三維空間轉換至各種不同的組態空間 (Configuration Space) 以簡化資訊量，並且從該空間透過不同的方法計算最佳路徑，以避開障礙物。

Artificial Potential Field [Khatib(1985)] 將環境資訊轉換為一虛擬力場，讓目標

對機器人施加吸引力，而障礙物則對其施加排斥力，而合力即為自走車應走的方向。Vector Field Histogram [Borenstein and Koren(1991)] 使用極座標的方式取得環境資訊並建立直方圖，找出足夠讓機器人通過的空間，並計算其轉向角度。Curvature Velocity Method [Simmons(1996)] 假設機器人的運動軌跡為圓弧，並將障礙物簡化為圓形，接著找出能夠讓機器人前進最長距離的一條路徑。Dynamic Window Approach [Fox et al.(1997)Fox, Burgard, and Thrun] 同樣假設機器人的運動軌跡為圓弧，並將環境資訊轉換至速度空間（Velocity Space），接著考慮動態拘束後找出可行方向，並計算最佳解。

由於 Yun-Trooper II 不具速度感測器，本論文採用增強後的 Vector Field Histogram 演算法：Vector Field Histogram Plus (VFH<sup>+</sup>) [Ulrich and Borenstein(1998)] 做為基礎，改進其功能使之能夠利用光學雷達得到的資訊。

### 1.3 論文架構



本論文共有六章，除本章外，第二章描述 Yun-Trooper II 所使用的硬體架構；第三章介紹導航所使用的路徑規劃演算法；第四章介紹 Yun-Trooper II 的控制法則；第五章為室外實際導航的實驗結果；第六章為結論與建議；附錄一為 MTi-G 的精度測試。

## 二、Yun-Trooper II 硬體架構介紹

### 2.1 硬體架構簡介

本論文之所開發的 Yun-Trooper II 是建置於 TAMIYA 公司的 1:10 四驅越野模型車 TXT-1 之上，並使用開源開發平台 BeagleBone Black 與 Linux 作業系統作為運算核心。而 Yun-Trooper II 的動力來源為兩顆直流馬達，並由兩顆伺服馬達同時控制前後輪之轉向機構。由於 BeagleBone Black 開發板可直接輸出 PWM 訊號，因此不需要伺服馬達控制器便可直接控制伺服馬達，而將輸出的 PWM 訊號利用馬達驅動器放大後也可直接控制直流馬達。BeagleBone Black 也可直接輸出 3.3V 準位的 IO 訊號，因此另外裝置了 2 顆白光 LED 做為狀態指示燈，以顯示車輛運作狀況。位置姿態感測器使用 Xsens 公司所生產的 MTi-G 位置姿態感測器，環境感測器則使用 HOKUYO 公司所生產的 URG-04LX-UG01 掃描式雷射測距儀 (LiDAR，光學雷達)，而兩者皆使用 USB 介面與運算核心連接。然而 BeagleBone Black 只具有單一 USB 埠且輸出電流只有 500mA，無法直接連接兩個感測器，因此需安裝一 USB 集線器且必須能夠外接電源，以提供兩顆感測器足夠的電流。Yun-Trooper II 的一切動作都是由遠端無線操控，由筆記型電腦搭配一遊戲控制器做控制，而無線通訊則是使用 Digi International 生產的 XBee PRO 無線通訊模組。硬體架構如圖 2.1 所示。

### 2.2 運算核心

為了降低體積與重量，Yun-Trooper II 使用 BeagleBone Black 開發板取代原先的工業用主機板作為運算核心。BeagleBone Black 為一開放原始碼之開發平台，具有低功耗、體積小、成本低等優點，搭配同為開放原始碼的 Linux 作業系統讓

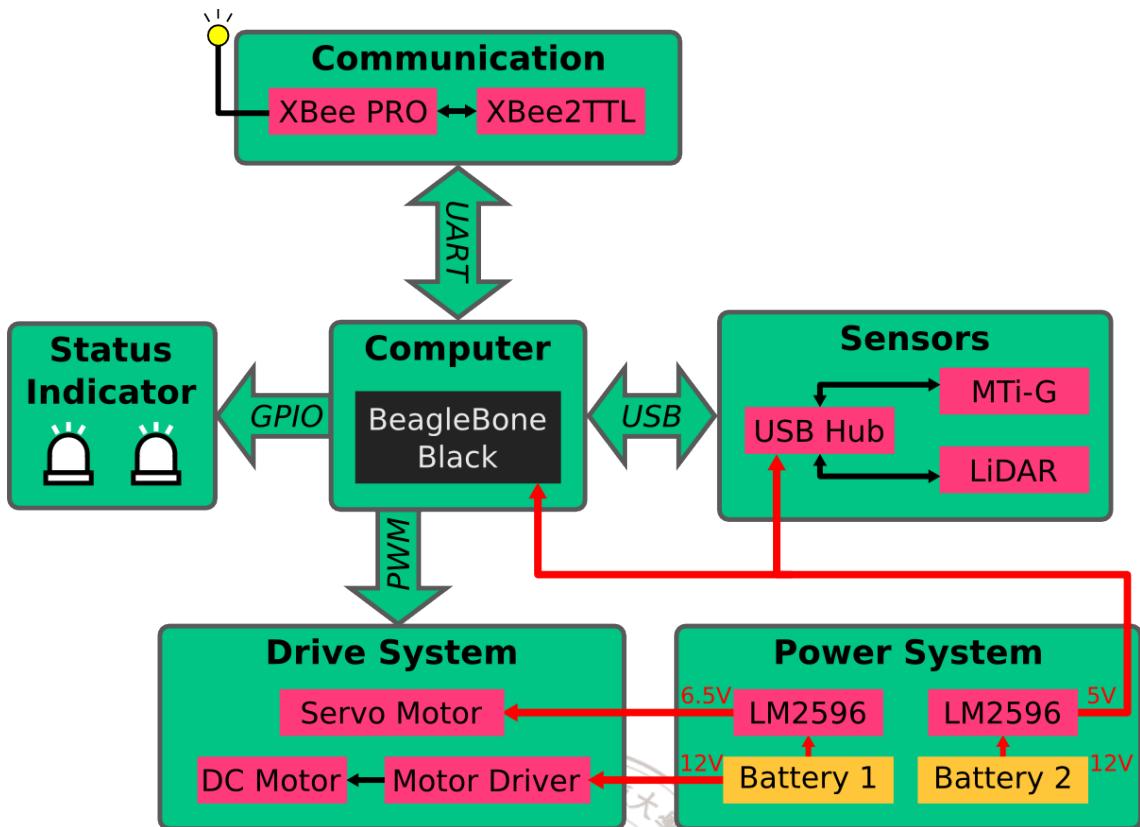


圖 2.1: Yun-Trooper II 硬體架構圖

開發成本降至最低。其規格如表 2.1 所示，外觀如圖 2.2 所示。

表 2.1: BeagleBone Black 規格

CPU	AM335x 1GHz ARM® Cortex-A8
Memory	512MB DDR3 RAM
Storage	2GB 8-bit eMMC on-board flash
IO	2 x 46 Pin Header
OS	Linux

## 2.3 感測元件

### 2.3.1 位置姿態感測器

位置姿態感測器使用 Xsens 公司所生產的 MTi-G 整合式感測器，它結合了 GPS、加速度計、角速度計、磁力計與壓力計，並內建 DSP 處理器即時利用

# BeagleBone Black

## 1 GHz performance ready to use for \$45

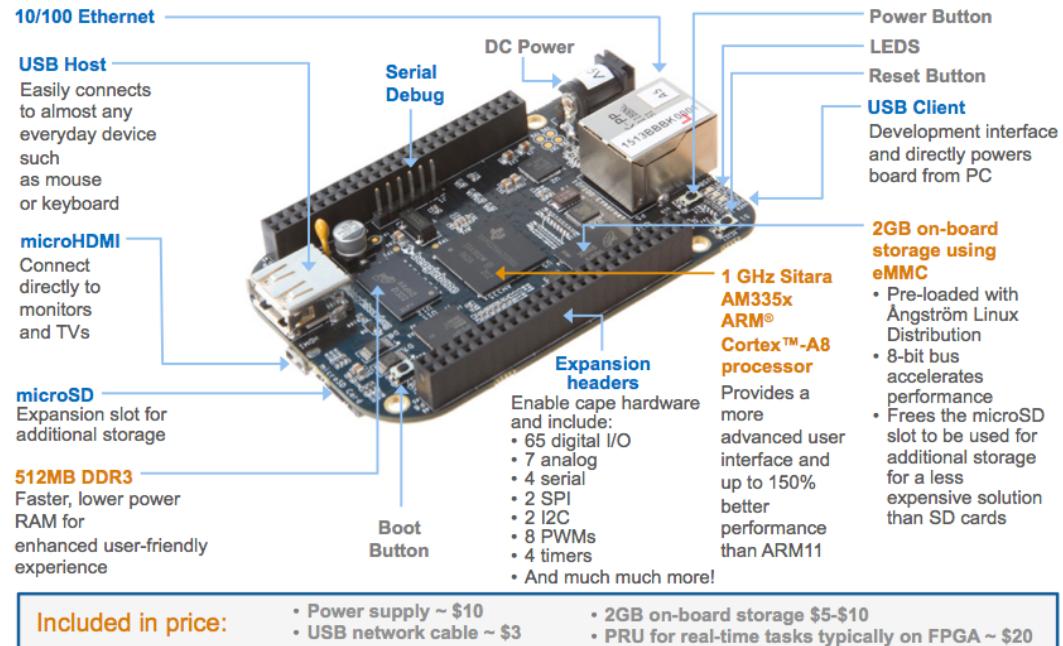


圖 2.2: BeagleBone Black 開發板

來源：[codeduino.com](http://codeduino.com)

Xsens 卡爾曼濾波器 (Xsens Kalman Filter) 來增進量測的準確度，比起傳統的 GPS 和姿態航向參考系統 (Attitude and Heading Reference System, AHRS) 更能提供穩定且低干擾的量測結果，外觀如圖 2.3 所示。其於特定條件下的性能如表 2.2 所示，請參考使用手冊 [B.V.(2012)]。



圖 2.3: Xsens MTi-G AHRS 位置姿態感測器

來源：[www.xsens.com](http://www.xsens.com)

表 2.2: MTi-G Performance Specification

GPS Receiver specification		Attitude and Heading from XKF-6G	
<b>Receiver Type<sup>12</sup>:</b>	50 channels GPS L1, C/A code GALILEO OpenService L1	<b>Dynamic Range:</b>	
<b>GPS Update Rate:</b>	4 Hz	<b>Pitch:</b>	$\pm 90^\circ$
<b>Pos/Vel Update Rate:</b>	120 Hz <sup>13</sup>	<b>Roll:</b>	$\pm 180^\circ$
<b>Accuracy Position SPS:</b>	2.5 m CEP	<b>Heading:</b>	$\pm 180^\circ$ (0...360°)
<b>SBAS:</b>	2.0 m CEP <sup>14</sup>	<b>Angular Resolution<sup>15</sup>:</b>	0.05 deg
<b>Start-up Time Cold start:</b>	29 s	<b>Static Accuracy:</b>	
<b>Re-acquisition:</b>	<1 s	<b>Roll/Pitch:</b>	<0.5 deg
<b>Tracking Sensitivity:</b>	-160 dBm	<b>Heading<sup>16</sup>:</b>	<1 deg
<b>Timing Accuracy:</b>	30 ns RMS	<b>Dynamic Accuracy<sup>17</sup>:</b>	
<b>Operational Limits:</b>		<b>Roll/Pitch:</b>	1 deg RMS
<b>Maximum Altitude:</b>	18 km	<b>Heading<sup>18</sup>:</b>	2 deg RMS
<b>Maximum Velocity:</b>	515 m/s	<b>Max update rate:</b> <b>Autonomously:</b>	120 Hz <sup>14</sup>
<b>Max dynamics GPS:</b>	4 g	<b>PC/raw data:</b>	512 Hz

來源：MTi-G User Manual and Technical Documentation[B.V.(2012)]

### 2.3.2 掃描式雷射測距儀

Yun-Trooper II 使用 HOKUYO 公司所生產的 URG-04LX-UG01 掃描式雷射測距儀，又稱光學雷達，來量測週遭環境的變化，規劃路徑以避開障礙物。其規格如表 2.3 所示，外觀如圖 2.4 所示

表 2.3: URG-04LX-UG01 規格

光源	半導體雷射 ( $\lambda = 785\text{nm}$ )
輸入電壓	5V DC $\pm 5\%$ (USB Power)
輸入電流	500mA (最大 800mA)
量測距離	20mm~4000mm
距離解析度	1mm
掃描範圍	$\pm 120^\circ$
角度解析度	0.36°
取樣頻率	10Hz



圖 2.4: HOKUYO URG-04LX-UG01 掃描式雷射測距儀

來源：[www.hokuyo-aut.jp](http://www.hokuyo-aut.jp)

## 2.4 驅動元件

### 2.4.1 轉向伺服機

為了縮小車輛的迴轉半徑，Yun-Trooper II 前後皆具有阿克曼轉向機構 (Ackermann Steering Mechanism)，各裝有一顆由 Thunder Tiger 公司所生產的 DS1015 數位伺服機做為轉向之動力來源。其規格如表 2.4 所示，外觀如圖 2.5 所示。

表 2.4: DS1015 數位伺服機規格

電壓範圍	4.8V ~ 6V
扭力 @ 6V	14.5kg-cm
重量	66g
速度 @ 6V	0.108 sec/60°
尺寸 (長 x 寬 x 高)	41.8mm x 20.6mm x 39.6mm
角度範圍	180°

### 2.4.2 直流馬達

Yun-Trooper II 使用 MABUCHI MOTOR 公司生產的 RS-550VC-7525 直流馬達做為車輛之動力來源，其規格如表 2.5 所示，外觀如圖 2.6 所示。



圖 2.5: DS1015 數位伺服機

來源：Thunder Tiger 官方網站

表 2.5: RS-550VC-7525 規格

MODEL	VOLTAGE		NO LOAD		AT MAXIMUM EFFICIENCY					STALL		
	OPERATING RANGE	NOMINAL	SPEED	CURRENT	SPEED	CURRENT	TORQUE		OUTPUT	TORQUE	CURRENT	
			r/min	A	r/min	A	mN·m	g·cm	W	mN·m	g·cm	A
RS-550VC-7525 (*1)	6.0~14.4	12V CONSTANT	17600	1.20	15730	10.1	58.3	594	95.9	549	5596	85.0

來源：MABUCHI MOTOR 官方網站



圖 2.6: RS-550VC-7525 直流馬達

來源：MABUCHI 官方網站

### 2.4.3 直流馬達驅動器

由於 BeagleBone Black 輸出的 PWM 訊號為 3.3V，最大電流約只有 6mA，無法直接驅動直流馬達，所以使用 Pololu 公司所生產的 MD01B 直流馬達驅動器將訊號放大，其外觀如圖 2.7 所示。

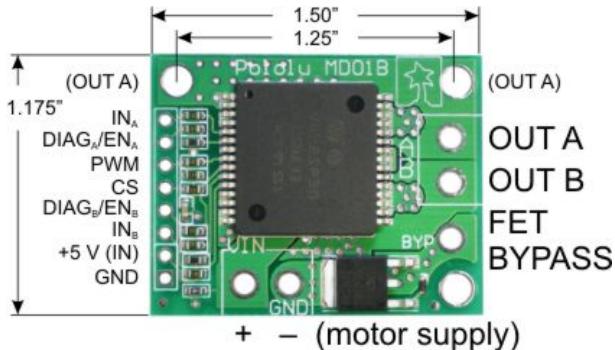


圖 2.7: Pololu MD01B 直流馬達驅動器

來源：Pololu 官方網站

## 2.5 電源供應元件

Yun-Trooper II 使用兩個 12V 的鋰電池做為電力來源，一顆供應動力系統（直流馬達與伺服機），另一顆供應運算與感測系統（運算核心與感測器）。其中伺服機需要 6.5V 的電壓源，運算核心與感測器則需 5V 電壓源，所以當中使用了兩個飄機器人公司所生產的 DC-DC LM2596 可調穩壓模組來供應不同的電壓源。其外觀如圖 2.8 所示。



圖 2.8: DC-DC LM2596 可調穩壓模組

來源：飄機器人官方網站

## 2.6 通訊元件

Yun-Trooper II 主要是由遠端透過無線通訊控制其功能，其使用 Digi International 公司所生產的 XBee-PRO 無線通訊模組做為通訊介面。XBee-PRO 簡單易用，只需在兩端設定相同的鮑率（Baud Rate）即可直接以串列埠（Serial Port）的方式通訊，不需其它設定。其性能規格如表 2.6 所示，外觀如圖 2.10 所示。XBee PRO 與 BeagleBone Black 的連接是使用飄機器人公司所生產的 XBee 轉 TTL 轉接板，如圖 2.11 所示；與個人電腦的連接則是使用同為飄機器人公司所生產的 XBee Explorer USB 轉接板，如圖 2.12 所示。



圖 2.9: XBee PRO 無線通訊模組

圖 2.10: 來源：[digipak.org](http://digipak.org)



圖 2.11: XBee 轉 TTL 轉接板

來源：飄機器人官方網站

表 2.6: XBee PRO 無線通訊模組性能規格

Specification	XBee-PRO
<b>Performance</b>	
Indoor/Urban Range	Up to 300 ft. (90 m), up to 200 ft (60 m) International variant
Outdoor RF line-of-sight Range	Up to 1 mile (1600 m), up to 2500 ft (750 m) international variant
Transmit Power Output (software selectable)	63mW (18dBm)* 10mW (10 dBm) for International variant
RF Data Rate	250,000 bps
Serial Interface Data Rate (software selectable)	1200 bps - 250 kbps (non-standard baud rates also supported)
Receiver Sensitivity	-100 dBm (1% packet error rate)
<b>Power Requirements</b>	
Supply Voltage	2.8 – 3.4 V
Transmit Current (typical)	250mA (@3.3 V) (150mA for international variant) RPSMA module only: 340mA (@3.3 V) (180mA for international variant)
Idle / Receive Current (typical)	55mA (@ 3.3 V)
Power-down Current	< 10 $\mu$ A

來源：Digi International 官方網站

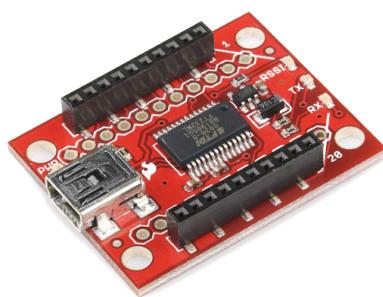


圖 2.12: XBee Explorer USB 轉接板

來源：[www.sparkfun.com](http://www.sparkfun.com)

### 三、路徑規劃與控制法則

本論文之路徑規劃係利用目標點位置與車輛位置之間的相對關係，以及車輛本身相對於地面的方位角  $\psi$ ，計算出目標點相對於車輛的角度  $\Theta_t$  與最短距離  $\sigma_t$ ，並使用這些參數將機器人依序導引至目標點，同時在導航的過程中必須要避開障礙物，如圖 3.1 所示。

由於位置單位為經緯度，3.1 節介紹了由經緯度計算  $\Theta_t$  與  $\sigma_t$  之方法。而利用  $\Theta_t$ 、 $\sigma_t$  與光學雷達所得到的環境資訊，本論文基於 VFH+ [Ulrich and Borenstein(1998)] 開發出一避障演算法計算出前進方向，以避開障礙物，3.2 節介紹其架構。

由於位置量測存在誤差，因此當目標點與車輛的距離  $\sigma_t$  小於一設定的距離  $R_p$  時，便判定車輛已到達該目標點，並開始導航至下一目標點。導航的完整流程圖如圖 3.2 所示：

#### 3.1 目標方向計算

##### 3.1.1 地理座標系統

地理座標系統（Geographic Coordinate System）是用來表示地球上某個位置的座標系統，這些座標系統可分為兩類：ECI（Earth Centered Inertial）與 ECEF（Earth Centered Earth Fixed），兩者之原點皆位於地球質心，但前者之座標系統不隨地球自轉而轉動，座標軸永遠指向固定的方向（相對於星星）；而後者之座標軸固定於地球上。前者一般使用於天文學，像是找尋衛星的軌道等；後者則多用來表示物體於地球上的位置，如 GPS 即使用 ECEF 系統。簡單的 ECEF 座標系統可以用三維卡式座標系來表示，其原點位於地球的質心，XY 平面與地球的赤道平面重合，X 軸指向經度  $0^\circ$ ，Y 軸則是指向東經  $90^\circ$ ，如圖 3.3 所示。

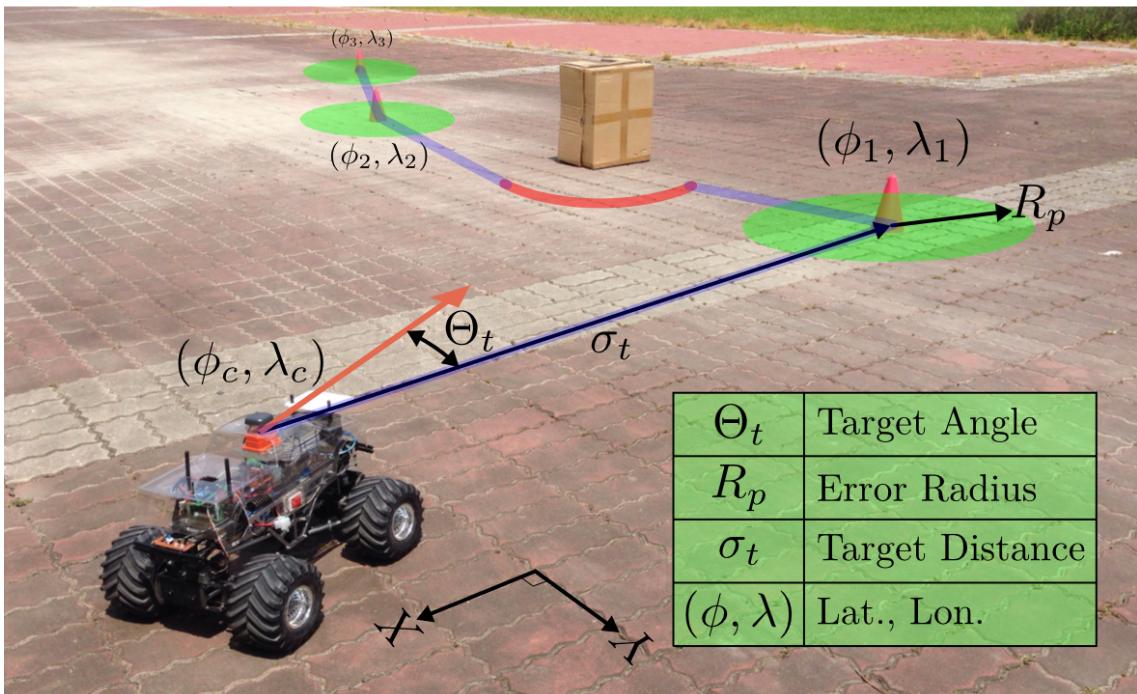


圖 3.1: 路徑規劃目標

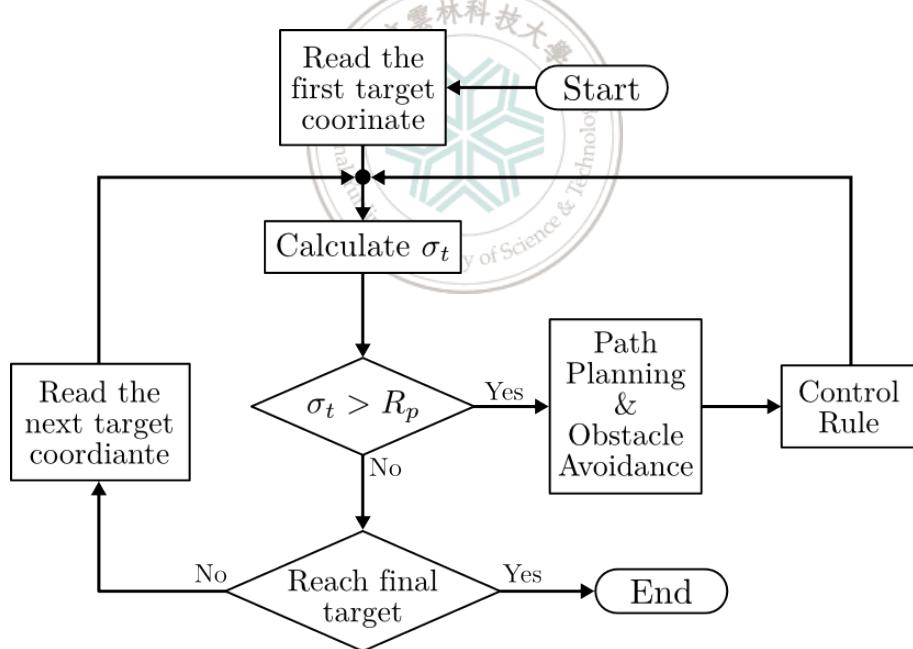


圖 3.2: 導航流程圖

雖然能夠使用各種座標系統來表示位置，但一般最常使用的是 ECEF 橢球座標系 (Ellipsoidal Coordinates)，使用緯度 (Latitude)  $\phi$ 、經度 (Longitude)  $\lambda$  與高度 (Altitude)  $h$  來表示三維空間中的點，如圖 3.4 所示，因為地球的形狀最接近橢球。此處的橢球為一雙軸橢球 (Biaxial Ellipsoid)，為一橢圓以短軸為旋轉軸旋轉

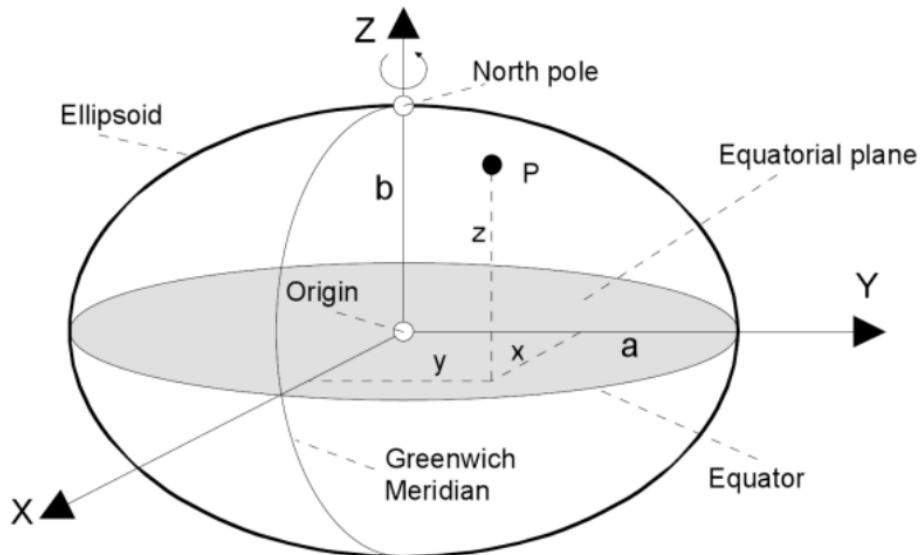


圖 3.3: ECEF 座標系統

來源：MTi-G Manual

一圈後得到的表面。

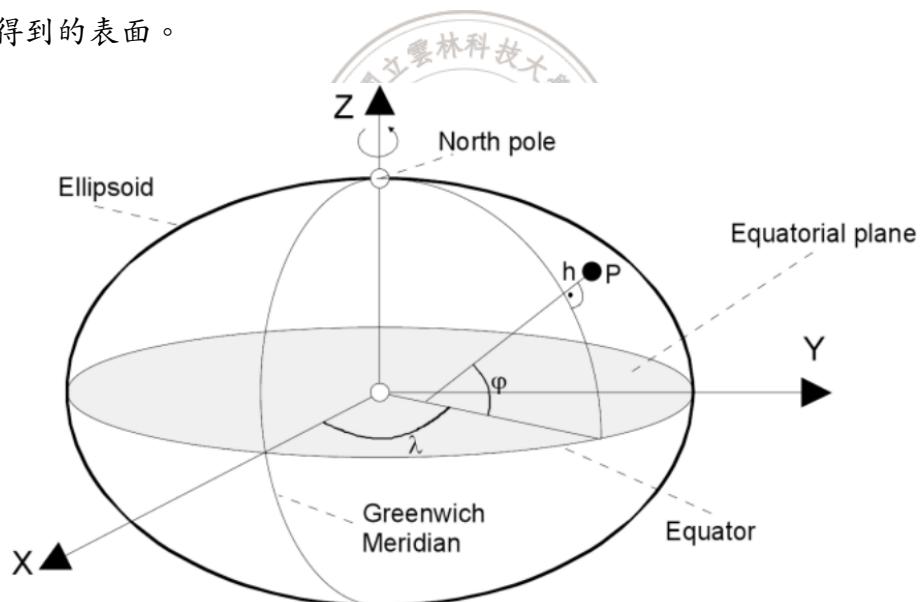


圖 3.4: 橢球座標系

來源：MTi-G Manual

橢球座標系會隨著定義的橢球不同而改變，而用來描述地球所定義的橢球就稱為大地基準（Datum）。MTi-G 位置感測器使用 WGS84 大地基準，而這個基準也是一般 GPS 所使用的標準座標系，如圖 3.3 所示，而參數如表 3.1 所示：

表 3.1: WGS84 大地基準參數

長軸 $a$	$6378137m$
短軸 $b$	$6356752.3142m$
扁率 $f$	$= (a - b)/a = 1/298.257223563$

另外要注意的是，在橢球上緯度的定義有三種：地心（Geocentric）、修化（Reduced）與測地（Geodetic）緯度 [Jekeli(2006)]。而最常見的緯度定義，包括 WGS84，都是定義為測地緯度，其定義為：若  $P$  為橢球座標系上的一點，則能夠找出一過該點的經度平面（Meridian Plane），而測地緯度  $\phi$  為在此平面上，過  $P$  點且與該橢圓垂直的直線與長軸的夾角，如圖 3.5 所示。

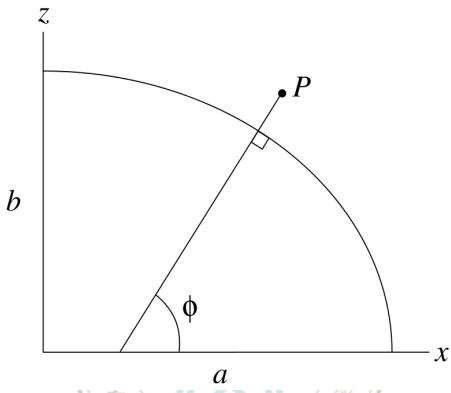


圖 3.5: 測地緯度示意圖

### 3.1.2 測地線

測地線（Geodesic）在微分幾何學上有嚴謹的定義，而在橢球曲面上可視為兩點之間的最短距離 [Karney(2013)]。與其相關的問題可分為兩種：Direct 與 Inverse，前者為給定起點  $A(\phi_1, \lambda_1)$ 、方位角（azimuth） $\alpha_1$  與距離  $s_{12}$  後計算終點位置  $B$ ；後者則是給定起點  $A(\phi_1, \lambda_1)$  與終點  $B(\phi_2, \lambda_2)$ ，計算兩者之間的方位角  $\alpha_1$  與最短距離  $s_{12}$ ，如圖 3.6 所示 [Jekeli(2006)]。

本論文需要計算兩個位置之間（車輛與目標點）的方位角與距離，因此必須要解決上述的 Inverse 問題。而測地線是微分幾何學上一個重要的研究對象，當曲面較為複雜時需要非常繁瑣的計算才能得到精準值，詳細解法可參

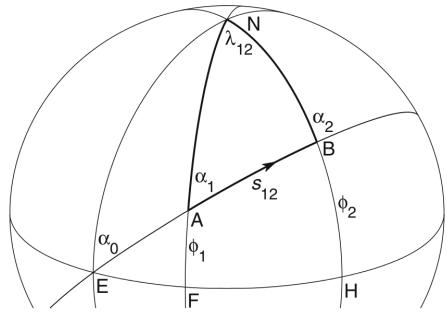


圖 3.6: 測地線

照 [Karney(2013), Jekeli(2006)]。然而，由於電腦計算能力大幅增加，已經可以使用數值方法來解決 Inverse 問題：假設  $\alpha_1$  為已知，因此利用已知的  $\phi_1$ 、 $\phi_2$  及  $\alpha_1$ ，可以計算出相應的  $\lambda_{12} = \lambda_2 - \lambda_1$ ，因此便可利用牛頓法迭代  $\alpha_1$  以得到正確的  $\lambda_{12}$ ，此時的  $\alpha_1$  就是正確的解，同時也可計算出相應的  $s_{12}$ ，也就是車輛與目標點之間的距離  $\sigma_t$  [Karney(2013)]。本論文使用 GeographicLib 函式庫 [Karney(2014)] 處理這個問題。

### 3.1.3 區域座標系統

區域切平面 (Local Tangent Plane) 為姿態量測系統的參考座標系，其 X 軸指向正北方且與橢球相切，如圖 3.7 所示。其量測的姿態角即為感測器座標系統相對於此座標系統之 Cardan Angle，即航空學上常用的 Roll( $\phi$ )、Pitch( $\theta$ ) 與 Yaw( $\psi$ ) 角。

上一節所述之方位角  $\alpha_1$  為使用地理座標系統 **G**，相對於北方順時針方向所量測的角度；Yaw 角度  $\psi$  為使用區域座標系統 **L**，相對於北方逆時針方向量測之角度。因此，目標方向  $\Theta_t$  於區域座標系統 **L**，相對於車輛之角度可依下式計算：

$$[\Theta_t]_L = -[\alpha_1]_G - [\psi]_L \quad (3.1)$$

若  $\Theta_t$  為負值代表目標在車輛右側，若為正值則於車輛左側，如圖 3.8 所示。

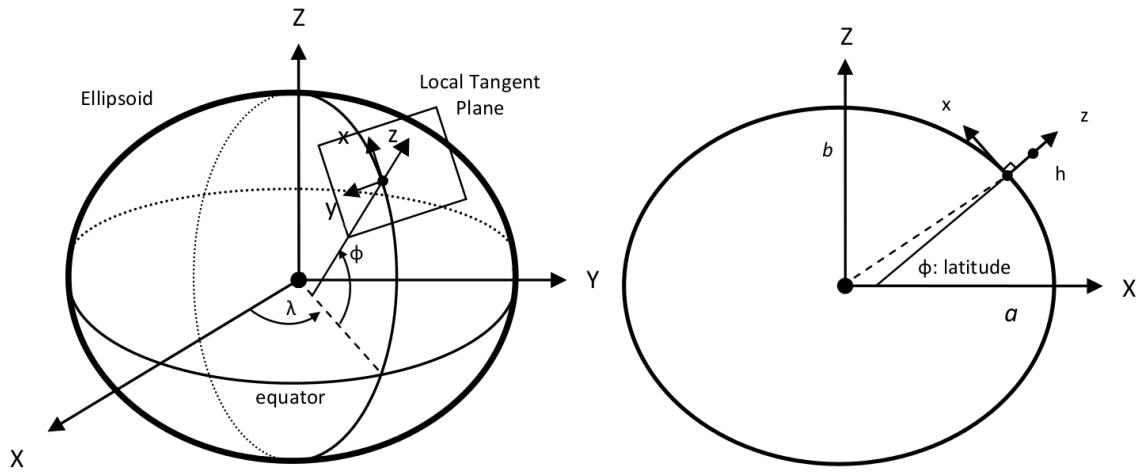


圖 3.7: 區域切平面示意圖

來源：MTi-G Manual

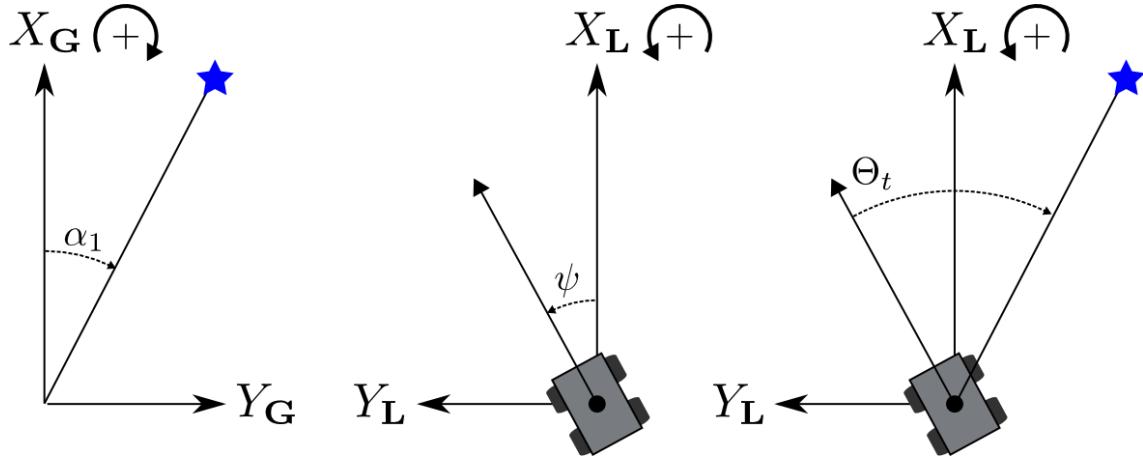


圖 3.8: 目標方向相對於車輛之角度

## 3.2 避障演算法

### 3.2.1 常見演算法介紹

#### Artificial Potential Field

Artificial Potential Field [Khatib(1985)] 使用環境資訊建立一虛擬力場，讓障礙物對機器人施加排斥力，目標點則對其施加吸引力，兩者之合力即為機器人需要

前進的方向。藉由已知的地圖或是感測器所得到的資訊，能夠計算出一位能場  $U(x, y)$ ，其中障礙物具有較高的位能，而目標點具有較低的位能，如圖 3.9 所示。接著對這個位能場做梯度運算，即可得到一虛擬力場  $F(x, y)$ ：

$$F(x, y) = -\nabla U(x, y) \quad (3.2)$$

利用此虛擬力場，便能夠計算機器人在每個位置所需要的導航方向，指引機器人遠離障礙物並接近目標點。

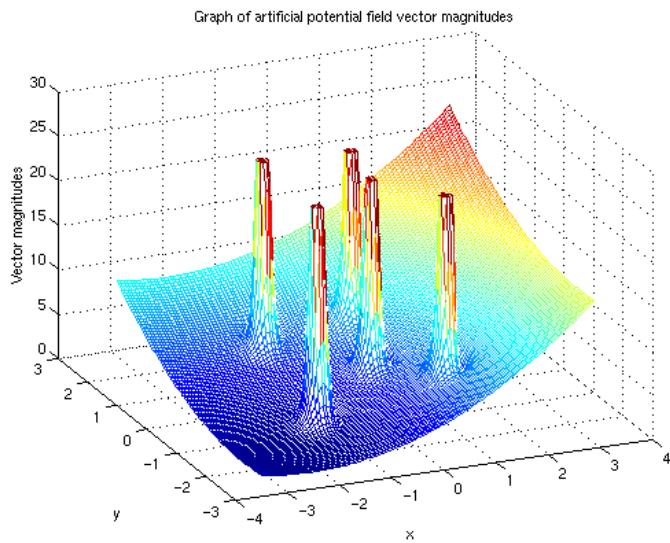


圖 3.9: 位能場

來源：[people.csail.mit.edu](http://people.csail.mit.edu)

此演算法不只能夠做障礙物迴避，只要有地圖資訊也同時具有全域路徑規劃的能力，而且計算效率高。然而此演算法假設機器人為單一質點，忽略機器人的動態拘束（最大加速度、機構拘束等）及幾何尺寸，且於狹窄的空間中表現較差。

### Vector Field Histogram

Vector Field Histogram (VFH) [Borenstein and Koren(1991)] 將環境資訊以極座標直方圖 (Polar Histogram)  $D$  的方式表示，橫軸為障礙之角度  $\theta$ ，縱軸為其距離

$d$ , 接著找出可供機器人通過的空間，並計算其相對應的轉向角度，如圖 3.10 所示。

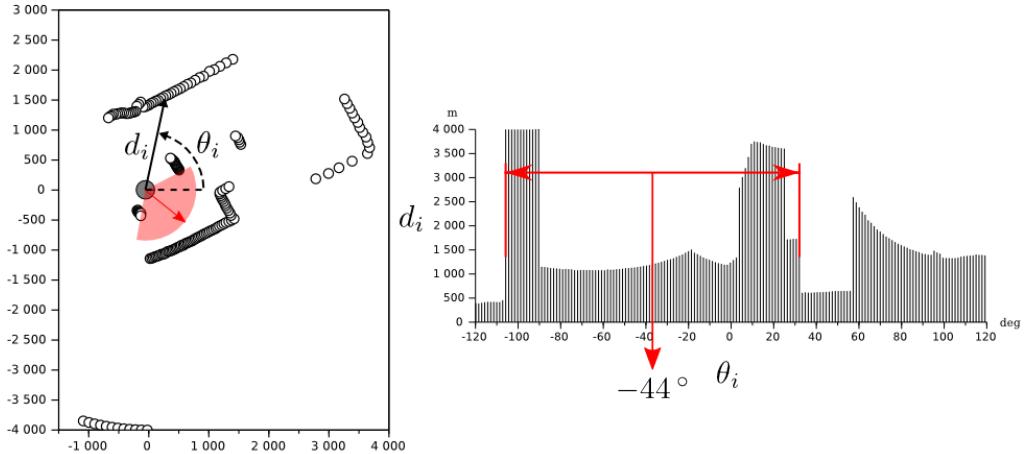


圖 3.10: 極座標直方圖

經過此計算後可能會出現多個可通過的候選轉向角度  $\alpha_n$ ，此時可以使用成本函數（Cost Function） $G$  來計算每個轉向角度所要花費的「成本」：

$$G = \mu_1 \delta_1 + \mu_2 \delta_2 + \mu_3 \delta_3 \quad (3.3)$$

其中  $\delta_1$  為目標方向與  $\alpha_n$  的差異； $\delta_2$  為目前車輛方位角與  $\alpha_n$  的差異； $\delta_3$  為前一次計算得到的轉向角度與  $\alpha_n$  的差異，而  $\mu_1$ 、 $\mu_2$  與  $\mu_3$  代表的是各個差異值的權重系數。 $G$  所計算出的值代表選擇該  $\alpha_n$  所需要耗費的成本，藉由調整權重系數也能改變機器人導航時的趨勢。

VFH 的極座標直方圖可直接套用光學雷達所得到的資訊，計算速度也相當快，而且藉由成本函數也能夠調整機器人的導航特性。然而 VFH 並沒有考慮機器人的動態拘束與幾何尺寸，而且 VFH 所計算的轉向角度是由可通過的空間所決定的，並非目標方向。這是一個相當重要的特性，在狹窄的室內空間中這不會造成太大的影響，然而在較為寬廣、障礙物較少的室外空間中，若光學雷達沒有偵測到障礙物，此時演算法只會找到一個可通過的空間，以圖 3.10 來說就是從  $-120^\circ$  到  $120^\circ$  的範圍，所以機器人只會朝著唯一的方向—正前方前進，直到偵測到障礙

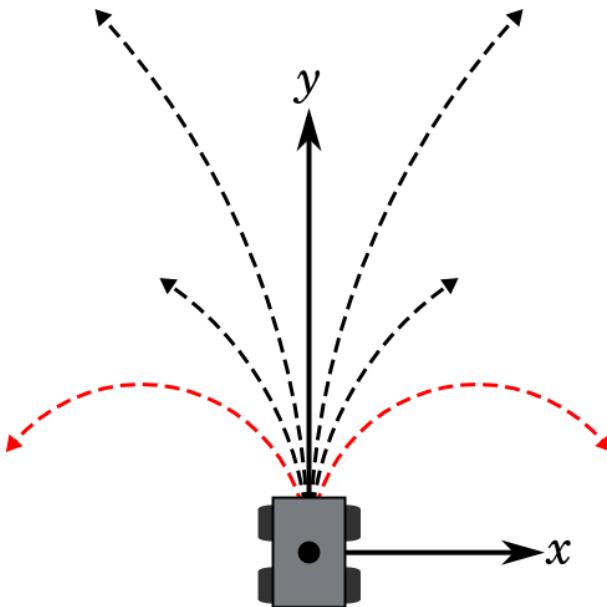


圖 3.11: CVM 下的機器人運動軌跡

物，因此 VFH 並不適用於本論文。

### Curvature Velocity Method

Curvature Velocity Method (CVM) [Simmons(1996)] 考慮機器人的動態拘束，假設機器人的運動軌跡為曲率  $c = \omega/v$  的圓弧，其中  $\omega$  代表機器人的旋轉速度 (Rotational Velocity)， $v$  則是直線速度 (Translational Velocity)，如圖 3.11 所示。因此，CVM 使用速度空間  $(v, \omega)$  來做路徑規劃，而非卡式座標空間。

為了將障礙物轉換至速度空間，設定一距離函數  $D(c, OBS)$  為沿著曲率  $c$  行走直到與障礙物集合  $OBS$  接觸的距離。接著設定一最大行走距離  $L$ ，表示機器人所能感測到的最大距離，此時距離函數將成為  $D_{limit}$ ：

$$D_{limit}(c, OBS) = \min(L, D(c, OBS)) \quad (3.4)$$

為了計算方便，CVM 將障礙物簡化為圓形，因此便能快速的將障礙物資訊轉換至速度空間。

CVM 同樣使用一目標函數（Objective Function） $f$  計算最佳的  $\nu$  與  $\omega$ ：

$$f(\nu, \omega) = \mu_1 \cdot speed(\nu) + \mu_2 \cdot dist(\nu, \omega) + \mu_3 \cdot head(\omega) \quad (3.5)$$

其中：

$$speed(\nu) = \nu / \nu_{max}$$

$$dist(\nu, \omega) = D_{limit} \left( \frac{\omega}{\nu}, OBS \right) / L$$

$$head(\omega) = 1 - |\theta_{target} - \omega \cdot T_c| / \pi$$

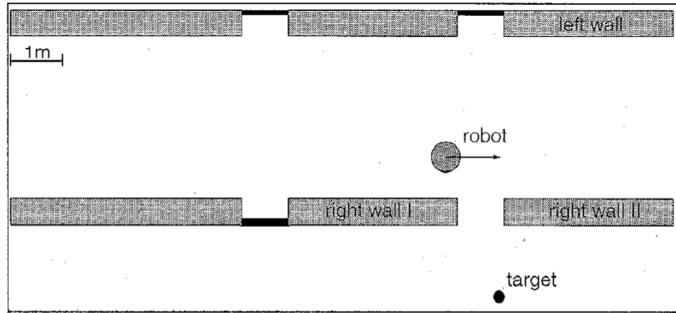
$\nu_{max}$  為最大直線速度， $\theta_{target}$  為目標方向， $T_c$  為一時間常數。此函數與先前的成本函數相反，產生最大值的  $(\nu, \omega)$  才是最佳值。

CVM 藉由速度空間設定動態拘束，可設定最大速度與最大加速度來限制其運動狀態。幾何拘束也可利用放大障礙物的尺寸來調整。然而過於簡化的障礙物是其限制，而且必須具備速度感測器才能使用此演算法，因此不適用於本論文。

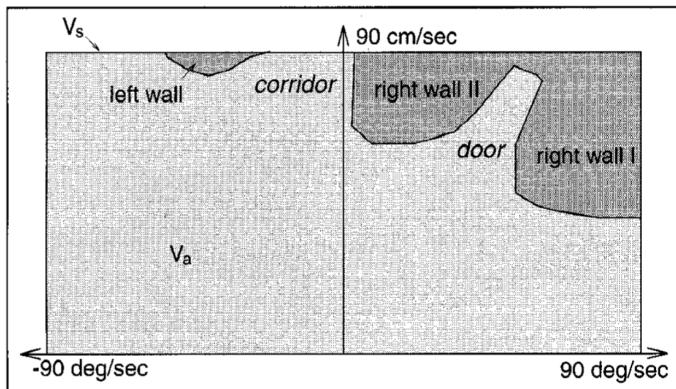
### Dynamic Window Approach

Dynamic Window Approach (DW) [Fox et al.(1997)Fox, Burgard, and Thrun] 與 CVM 相同，假設機器人之運動軌跡為曲率  $c = \omega/\nu$  的圓弧，利用最大直線速度與最大旋轉速度建立一速度空間，並將量測到的環境轉換至此空間中，如圖 3.12a 及 3.12b 所示。在圖 3.12b 中橫軸為旋轉速度，縱軸為直線速度；較暗的部分為被障礙物擋住的部分。

在速度空間中，根據機器人目前的速度以及最大加速度，可在速度空間中找出一動態視窗 (Dynamic Window)  $V_d$ ，此視窗根據最大加速度限制了機器人所能達到的速度 (視窗大小)，隨著機器人目前的速度不同此視窗的位置也會不斷改



(a) 實際環境資訊



(b) 速度空間中的環境資訊

圖 3.12: 環境資訊轉換

來源：The Dynamic Window Approach to Collision Avoidance

變，如圖 3.13 所示。此視窗中的所有速度( $\nu, \omega$ )代表了機器人所有可能達到的速度，

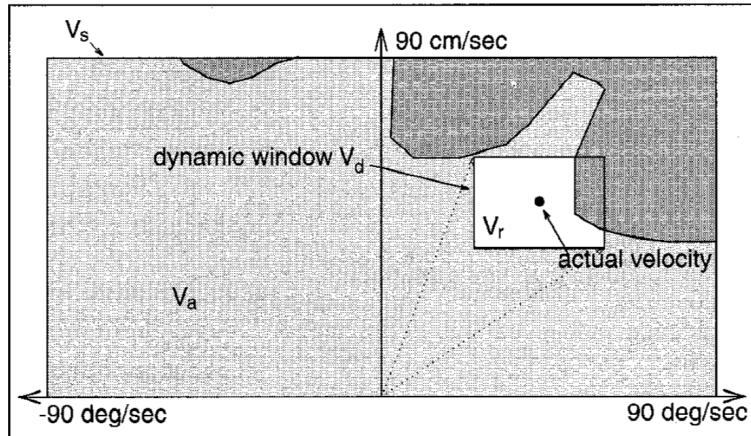


圖 3.13: 動態視窗示意圖

來源：The Dynamic Window Approach to Collision Avoidance

度，而為了在此視窗中找出最佳解，DW 同樣使用目標函數來做最佳化。

藉由改變速度空間的形狀，DW 能夠設定機器人的機構運動拘束，而動態視窗則考慮了機器人的動態拘束。然而 DW 的計算較為複雜，且與 CVM 相同，都必須裝設速度感測器才能使用此演算法，因此並不適用於本論文。

### 3.2.2 Vector Field Histogram Plus

Vector Field Histogram Plus (VFH+) [Ulrich and Borenstein(1998)] 改進了許多 VFH 的缺點，將機器人的幾何限制和運動拘束也考慮在內。

原先的 VFH+ 使用四個階段的計算逐一減少資訊量並找出轉向角度，而為了使用在光學雷達上，本論文一樣使用四個階段的計算，但修改其計算方式與順序以便使用於光學雷達。前面三個階段著重於根據機器人的拘束找出可通過的方向，最後一個階段則是計算最佳方向。

階段一、極座標直方圖

光學雷達所量測到的資訊可表示為  $d_i$  與  $\theta_i$ ， $d_i$  為第  $i$  個量測到的距離， $\theta_i$  則為  $d_i$  所對應的角度，如圖 3.14 所示。

與 VFH 相同，VFH+ 首先使用光學雷達所得到的資訊建立一極座標直方圖  $P_i$ ：

$$P_i = a - b \cdot d_i \quad (3.6)$$

$a$  與  $b$  皆為正值。藉由調整此處的  $a$  與  $b$ ，使用者可調整 VFH+ 所要偵測與計算的範圍。圖 3.15 為  $a = 1200$ 、 $b = 1$  的設定下，圖 3.14 所量測到之環境產生的  $d_i$  與  $P_i$  示意圖。

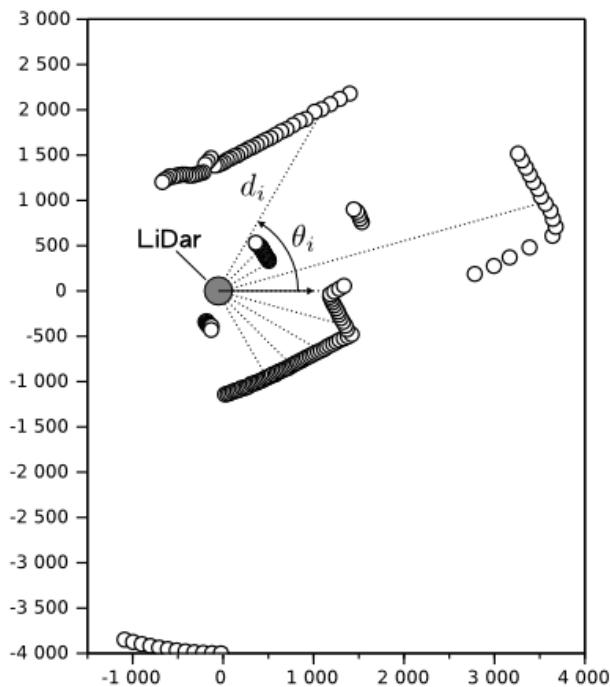
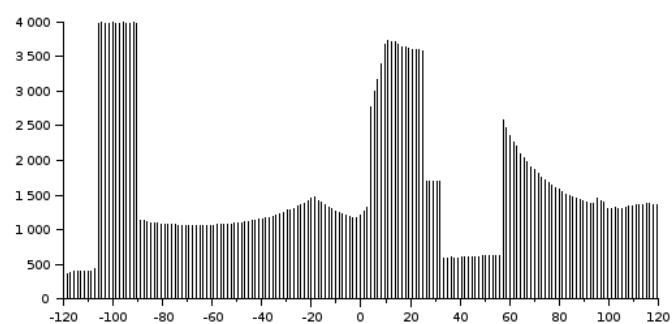
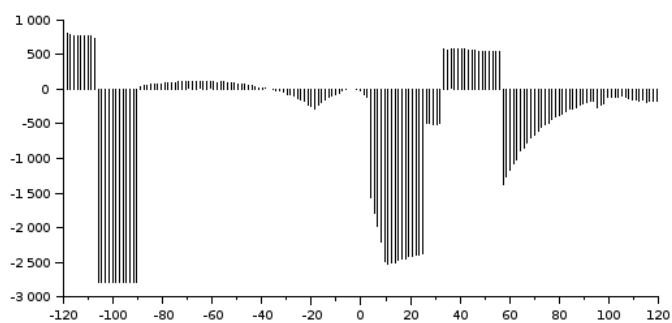


圖 3.14: 光學雷達量測示意圖



(a)  $d_i$



(b)  $P_i$

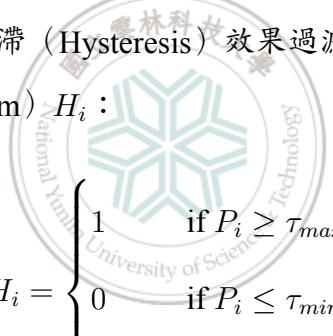
圖 3.15: 極座標直方圖

## 階段二、安全空間

為了找出可供機器人通過的空間，VFH 藉由一閾值  $\tau$  過濾出所有視為安全的距離，而一段連續的安全距離就代表一個可通過的安全空間  $V_j$ ，由一組邊界向量  $(\mathbf{B}_L, \mathbf{B}_R)_j$  定義，定義其左邊界與右邊界的角  $\theta$  與障礙物的距離  $d$ ，視為機器人所能通過的方向：

$$\begin{aligned}\mathbf{B}_L &= \begin{bmatrix} \theta_l & d_l \end{bmatrix} \\ \mathbf{B}_R &= \begin{bmatrix} \theta_r & d_r \end{bmatrix}\end{aligned}\quad (3.7)$$

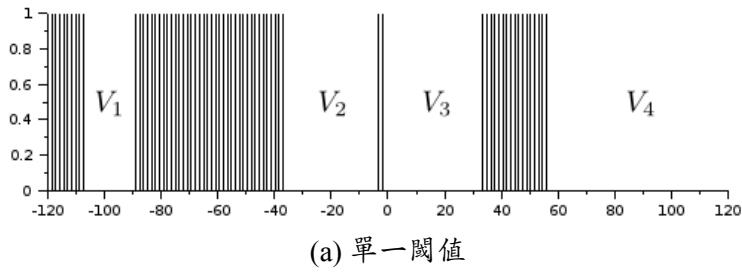
然而 VFH 只使用單一閾值可能會過濾出許多不連續的空間，造成許多不必要的方向選擇出現，讓機器人在決定方向時產生左右搖擺的現象。因此，VFH+ 使用兩個閾值  $\tau_{max}$  與  $\tau_{min}$ ，利用遲滯（Hysteresis）效果過濾掉這些不必要的空間，產生二元直方圖（Binary Histogram） $H_i$ ：



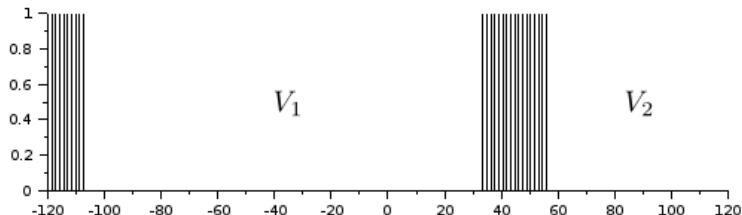
$$H_i = \begin{cases} 1 & \text{if } P_i \geq \tau_{max} \\ 0 & \text{if } P_i \leq \tau_{min} \\ H_{i-1} & \text{otherwise} \end{cases} \quad (3.8)$$

圖 3.16a 為使用單一閾值  $\tau = 0$  的過濾結果，圖 3.16b 則為使用雙閾值  $\tau_{min} = 0, \tau_{max} = 450$  的過濾結果，值為 0 的區域代表可通過的安全空間。圖中可看到單閾值過濾產生了四個區域，而雙閾值則只有二個。

原先的 VFH 忽略了機器人本身的幾何尺寸，而本論文使用縮小安全空間  $V_j$  邊界的方式來增加幾何拘束，此時便可將機器人視為一質點。假設機器人之尺寸為半徑  $w_s$  的圓，則將  $V_j$  之邊界同樣縮小  $w_s$  後便可將機器人視為一點，如



(a) 單一閾值



(b) 雙閾值

圖 3.16: 過濾結果比較

圖 3.17 所示。縮減後的安全空間  $\hat{V}_j = (\hat{\mathbf{B}}_{\mathbf{L}}, \hat{\mathbf{B}}_{\mathbf{R}})_j$  可由式 3.9 計算：

$$\begin{aligned}\hat{\mathbf{B}}_{\mathbf{L}} &= \left[ \theta_l - \delta_l \quad d_l \cos \delta_l \right], \delta_l = \arcsin\left(\frac{w_s}{d_l}\right) \\ \hat{\mathbf{B}}_{\mathbf{R}} &= \left[ \theta_r + \delta_r \quad d_r \cos \delta_r \right], \delta_r = \arcsin\left(\frac{w_s}{d_r}\right)\end{aligned}\quad (3.9)$$

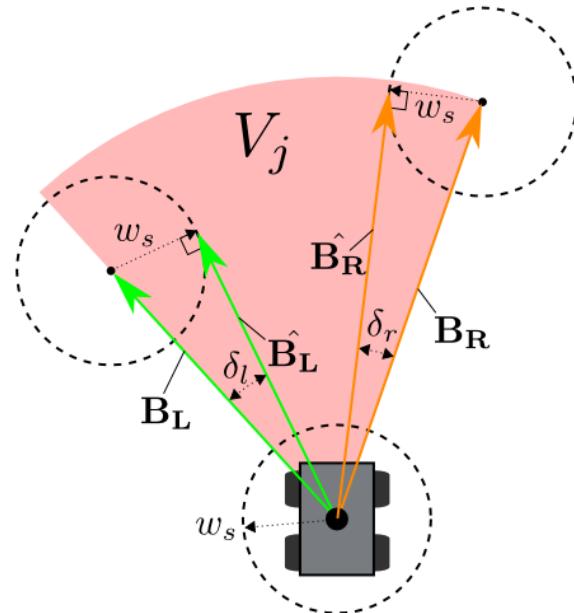


圖 3.17: 邊界縮減

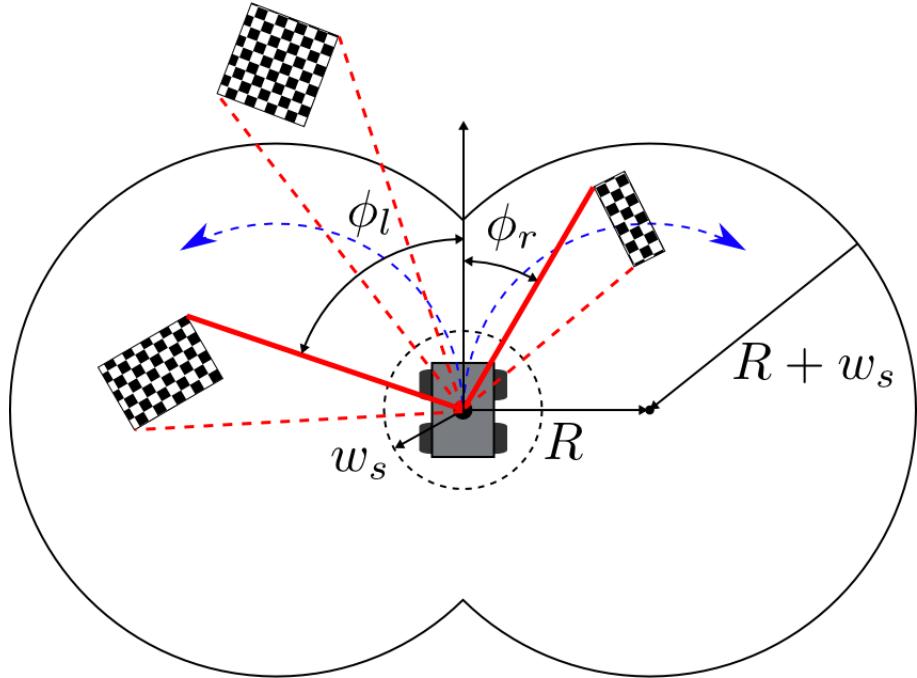


圖 3.18: 轉向角度限制

### 階段三、Blocked Directions

VFH 並沒有將機器人的動態拘束納入考慮，允許所有轉向角度的存在，然而這對某些轉向機構來說是不可能的。因此 VFH+ 使用機器人的最小迴轉半徑  $R$  與機器人尺寸  $w_s$ ，計算出被障礙物所限制住的轉向角度  $(\phi_l, \phi_r)$ 。而原先的 VFH+ 僅使用迴轉半徑  $R$  來計算，而本論文將機器人尺寸  $w_s$  加入計算，形成與迴轉半徑同心但半徑為  $R + w_s$  的圓，如圖 3.18 所示。

為了找出  $(\phi_l, \phi_r)$ ，首先使用與光學雷達相同的  $\theta_i$  建立一偵測用的直方圖  $D_i$ ，偵測障礙物是否阻擋了轉向角度，如圖 3.19 所示。其可由式 3.10 計算。

$$D_i = |R \sin \theta_i| + \sqrt{R^2 \sin^2 \theta_i + w_s^2 + 2Rw_s} \quad (3.10)$$

接著將光學雷達測得的  $d_i$  與  $D_i$  相減，所得到的直方圖稱為  $M_i$

$$M_i = d_i - D_i \quad (3.11)$$

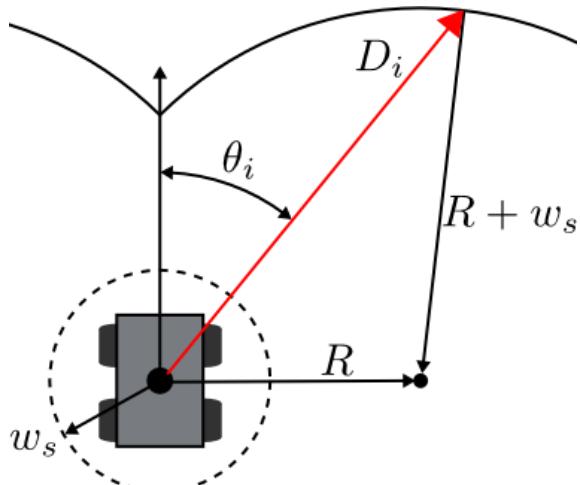


圖 3.19: 偵測直方圖示意圖

若  $M_i$  中對應於  $\theta_i$  之值小於 0，則代表  $\theta_i$  有障礙物位於機器人之最小迴轉範圍內，必須改變  $(\phi_l, \phi_r)$  限制轉向角度以避開障礙物。環境與這些直方圖之間的轉換如圖 3.20 所示。

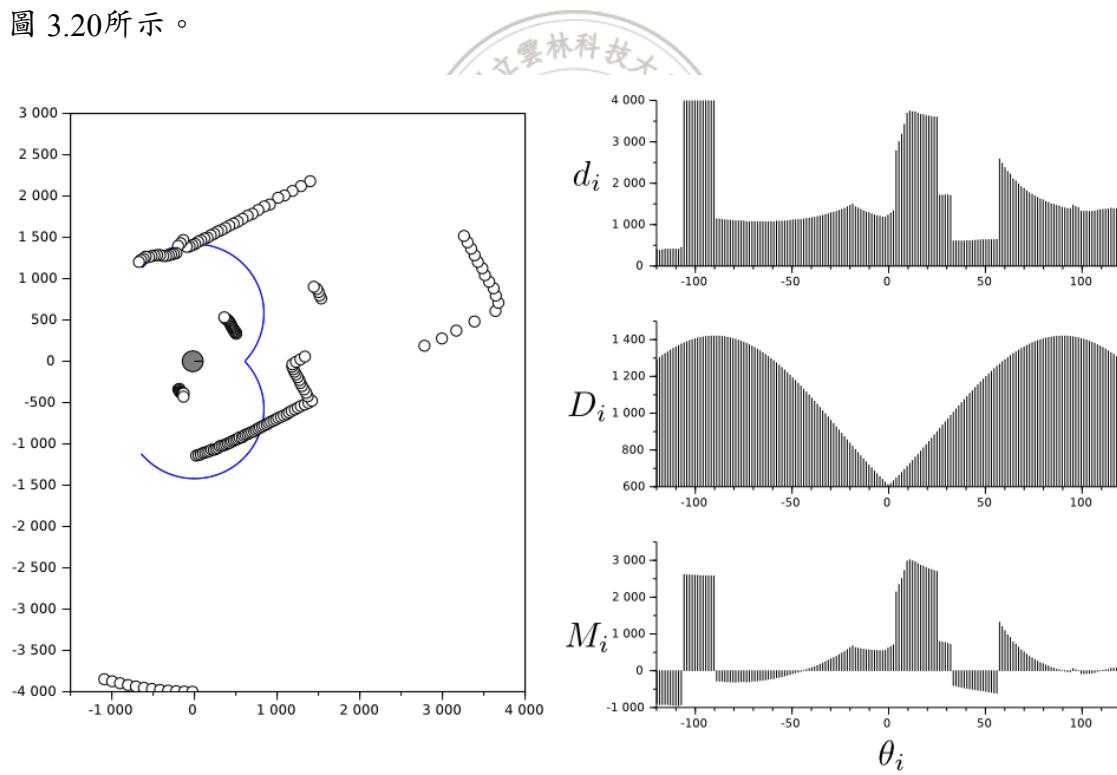


圖 3.20: 直方圖轉換

利用  $M_i$ ,  $(\phi_l, \phi_r)$  可以非常快速的被計算出來：

1. 首先設  $\phi_l = \pi$  、 $\phi_r = -\pi$
2. 對所有  $i$ ，若  $M_i < 0$ ：
  - (a) 若  $\theta_i < 0$  且  $\theta_i > \phi_r$ ，則將  $\phi_r$  設為  $\theta_i$
  - (b) 若  $\theta_i > 0$  且  $\theta_i < \phi_l$ ，則將  $\phi_l$  設為  $\theta_i$

#### 階段四、Selection of Steering Direction

根據安全空間  $\hat{V}_j$  的寬度，可以從每個  $\hat{V}_j$  中找到一個或多個候選方向  $c$ ，接著使用成本函數於這些候選方向中找出最佳解。對於安全空間的寬度，本論文使用其邊界向量之間的角度差  $\epsilon = \theta_l - \theta_r$  做為判斷標準，使用一角度閾值  $\tau_a$  與邊界縮減的情況將安全空間分為三種：交疊、狹窄與寬廣，如圖 3.21 所示。

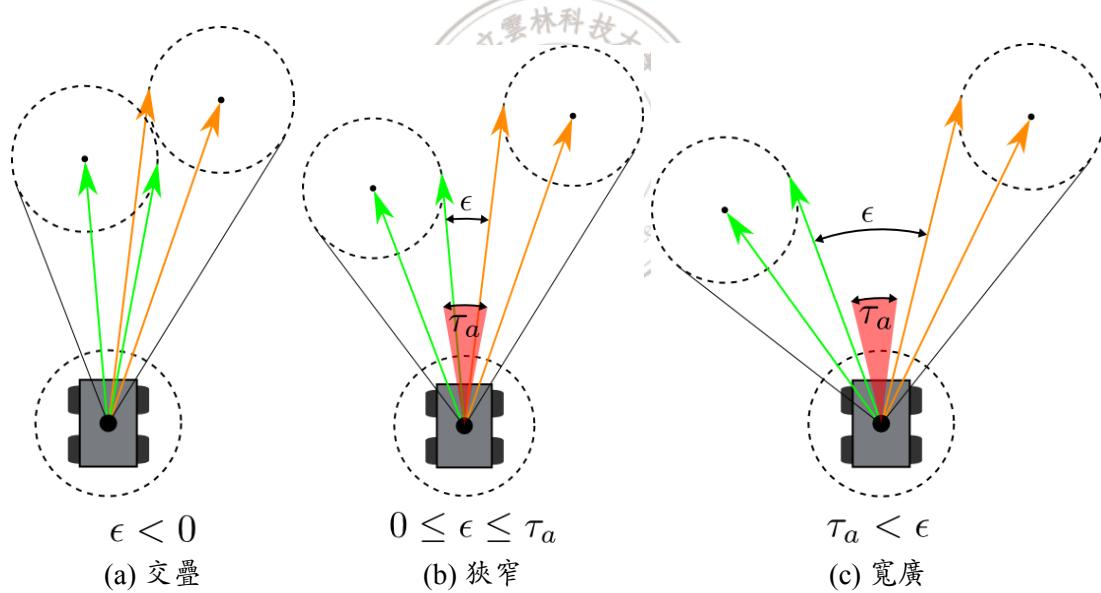


圖 3.21: 邊界分類

由於光學雷達在量測時是逆時針方向從  $-120^\circ$  掃描至  $120^\circ$ ，在搜尋安全空間時也是同一方向，因此每個安全空間的右邊界角度一定小於左邊界角度。因此，若是右邊界角度大於左邊界代表發生了邊界交疊，此時該安全空間將會被捨棄，不會產生任何候選方向，如圖 3.21a 所示。

圖 3.21b 為狹窄 ( $0 < \epsilon < \tau_a$ ) 的安全空間，此時只有正中央的方向是唯一的候選方向：

$$c_n = \frac{\theta_l + \theta_r}{2} \quad (3.12)$$

圖 3.21c 則為寬廣 ( $\epsilon > \tau_a$ ) 的安全空間。於此空間中候選角度有二個或三個，分別為兩邊界向量所對應的角度，以及當目標方向  $\Theta_t$  落在此範圍中，則  $\Theta_t$  也是候選角度之一。

$$c_r = \theta_r$$

$$c_l = \theta_l$$

$$c_T = \Theta_t, \text{ if } \theta_l < \Theta_t < \theta_r \quad (3.13)$$

最後使用一成本函數  $G$  於這些候選轉向角中找出最佳解：

$$G(c) = \mu_1 \cdot (|c - \Theta_t|) + \mu_2 \cdot (|c|) + \mu_3 \cdot (|c - c_{t-1}|) \quad (3.14)$$

在式 3.14 中，第一項 ( $|c - \Theta_t|$ ) 代表候選方向與目標方向之間的差距。差距越大表示該候選角會將機器人帶離目標方向，所以成本將會增加。

第二項 ( $|c|$ ) 代表候選方向與目前車輛方向的差距。這些候選角都是以車身座標系做為參考，所以車輛本身方位角相對於此座標系將永遠是 0，而候選方向越大代表機器人將會偏離目前的方向越多，進而增加成本。

第三項 ( $|c - c_{t-1}|$ ) 則代表候選角與前一次選擇的轉向角之間的差距。這個差距越大代表機器人的越容易偏離目前的航向，造成擺盪的現象。

$\mu_1, \mu_2, \mu_3$  則是相對應的權重係數，藉由調整這三個係數之間的相對大小，機器人的導航特性就能夠被調整，而於成本函數中產生最小值的候選方向即為最佳方向  $c_t$ 。

### 3.2.3 問題討論與補償

VFH+ 演算法在大部分的情況下都能夠成功找出正確的轉向角度以迴避障礙物，然而有兩個情況會導致計算錯誤，以下將討論這兩種情況。

#### 無候選方向

VFH+ 對機器人的轉向有許多的限制，因此有時會發生無候選方向的情況。這種情況時常發生於狹窄的空間，同時前方已無路可走且機器人之機構限制無法迴轉離開時。然而，有時是因為 VFH+ 演算法的特性造成誤判，將可通過空間判定為無法通過，圖 3.22 即為一誤判的例子。在圖 3.22b 中，藍色虛線為機器人的最小迴轉半徑，可看到右前方的確擁有足夠的空間供機器人通過，然而此時 VFH+ 會因為邊界交疊的關係將此安全空間捨棄，造成無候選方向的情形，進而讓導航停止。此問題將會使用速度控制與碰撞偵測補償，於 3.3 節詳加介紹。

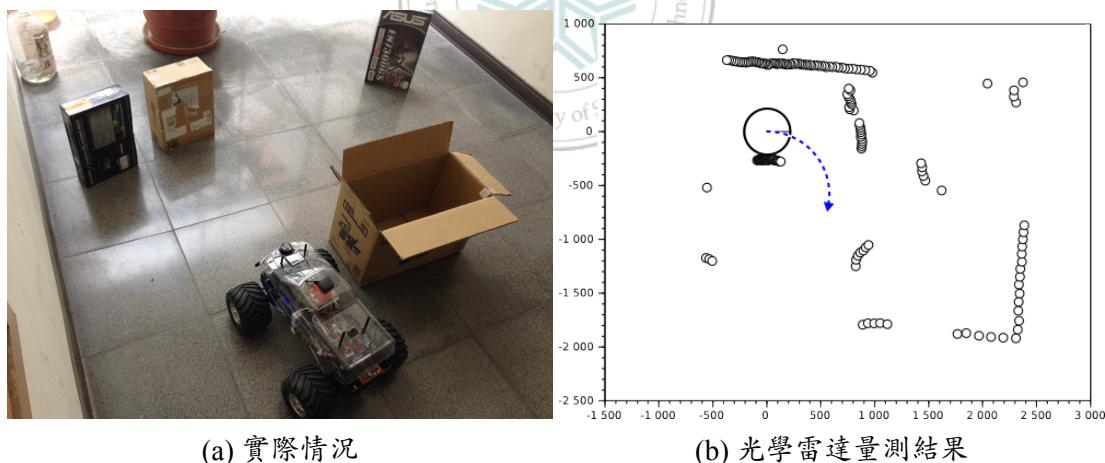


圖 3.22: 無候選轉向角之環境

#### 安全空間邊界誤判

為了將 VFH+ 使用在光學雷達上，本論文使用安全空間取代了原始計算方法，以增加計算效率。此方法在大部分情況下都能快速且有效的找出安全空間

之邊界向量，然而在某些特殊情況下，此方法會得到不正確的邊界向量，進而計算出錯誤的轉向角度，圖 3.23 即為一範例，而依圖 3.23 之環境產生之各直方圖如圖 3.24 所示。

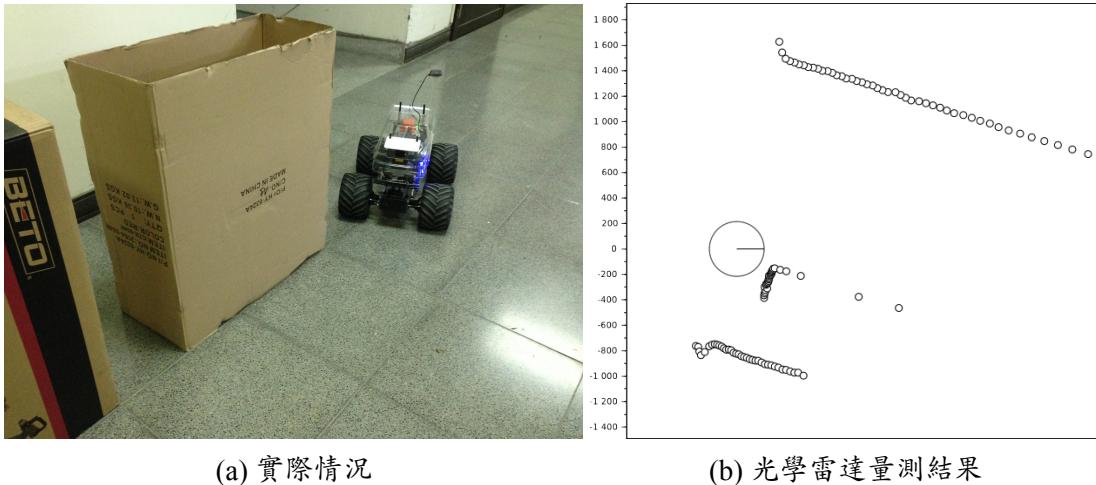


圖 3.23: 安全空間邊界誤判之環境

由於找尋安全空間的方式是從二元直方圖  $H_i$  中找出連續為 0 的區域，並根據此區域的邊界於  $d_i$  中找出相對應的邊界向量 ( $\mathbf{B}_L, \mathbf{B}_R$ )，因此邊界向量會是離該空間最近的兩個向量，如圖 3.24 所示。根據此計算方式，於圖 3.23 的環境下所找出的空間邊界與進行邊界縮減後的方向如圖 3.25a 所示。而此時量測到的邊界會造成縮減角度不足，計算出錯誤的轉向角度，即為發生了邊界誤判。

由圖 3.23 可觀察到，真正的邊界應為障礙物之頂角，如圖 3.25b 所示。因為距離較近，此邊界能夠提供機器人足夠的縮減角度，讓機器人能夠安全通過該空間。

為了解決此問題，可以對每一個量測值  $d_i$  與其角度  $\theta_i$  依前述的邊界縮減方式，計算所有量測值對機器人轉向角度的限制，如此便可找出真正的邊界向量。然而此計算相當費時，而且對轉向角限制最大的通常是距離最近的障礙物，因此本論文僅針對距離機器人最近的量測值  $d_{min}$  與其角度  $\theta_{min}$  計算邊界縮減值，找出其對機器人轉向角的限制範圍 ( $R_l, R_r$ )，如圖 3.26 所示。若是邊界縮減後的安全空間之右邊界角度落在此範圍內，則將該邊界之角度設為  $R_l$ ；若縮減後的左邊界角

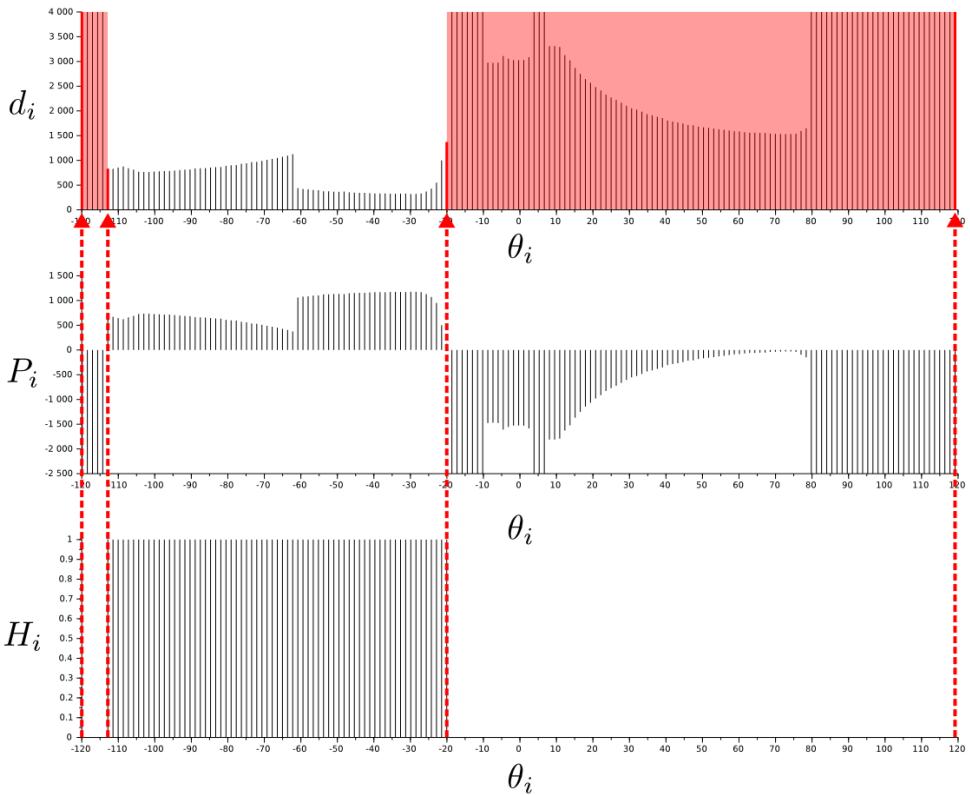


圖 3.24: 環境直方圖

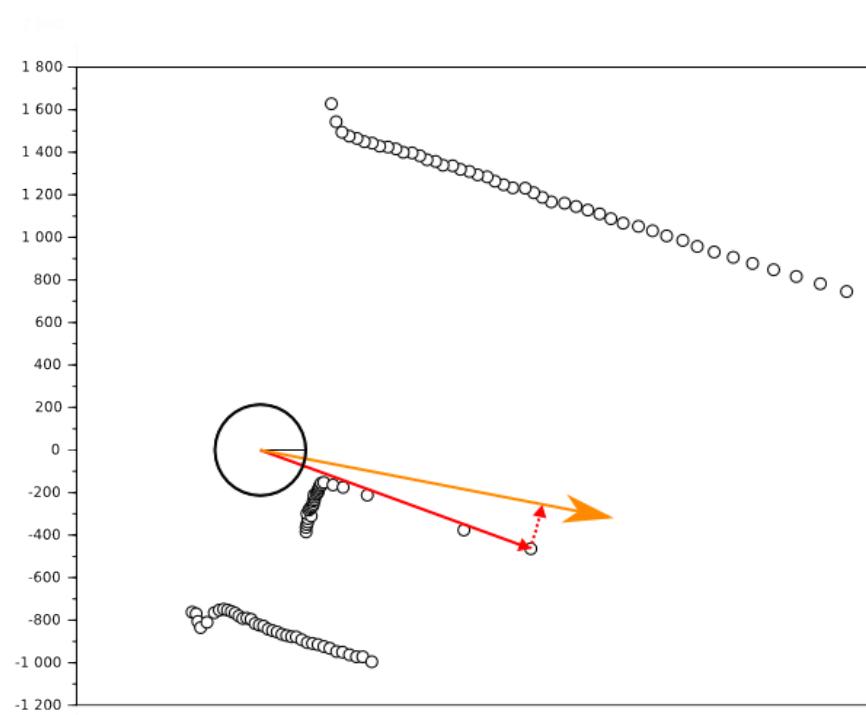
度落在此範圍內，則將該邊界之角度設為  $R_r$ 。如此，便可補償邊界誤判的問題，同時不致於降低計算效率。

### 3.3 速度演算法

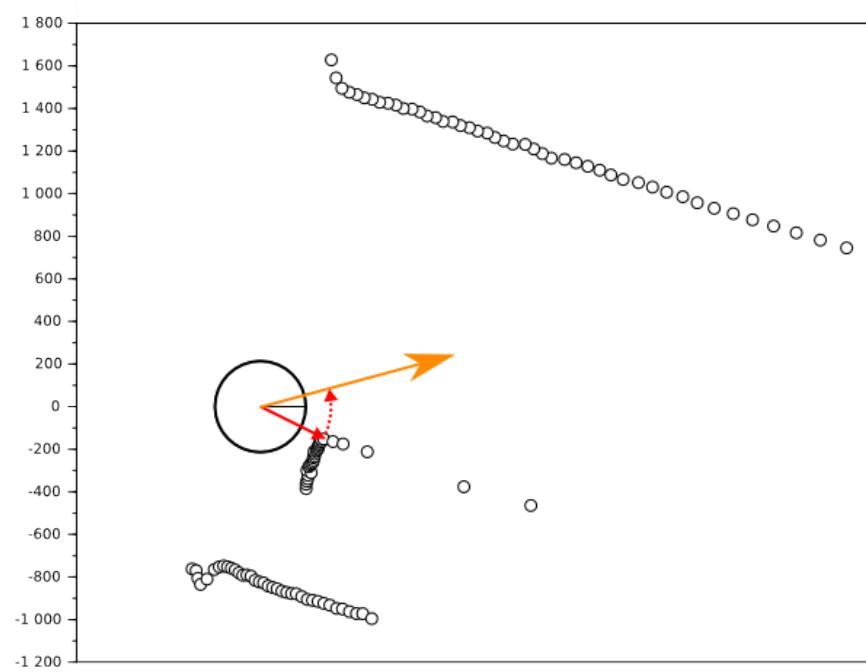
#### 3.3.1 障礙物密度

原先的 VFH+ 使用障礙物密度函數  $D$  來計算機器人於該環境狀態下的速度，而此函數可使用光學雷達的量測結果  $d_i$  與最大量測距離  $d_{max}$  計算：

$$D(d_i) = 1 - \frac{1}{N} \sum_{i=1}^N \frac{d_i}{d_{max}} \quad (3.15)$$



(a) 計算後的邊界方向



(b) 實際之邊界方向

圖 3.25: 邊界縮減後之方向

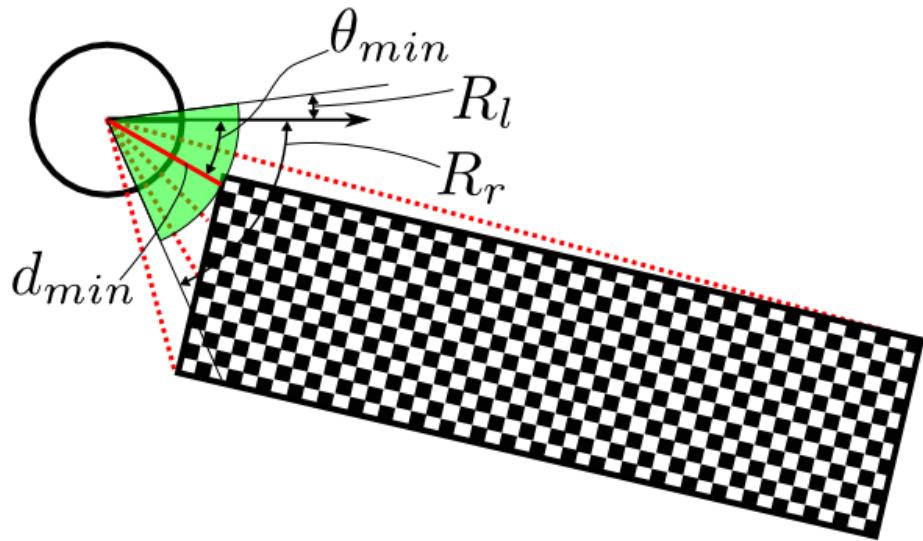


圖 3.26: 最近距離之邊界縮減

此函數所計算之值將會落在 0 與 1 之間，越接近 1 代表環境中的障礙物距離越近或數量較多（密度較高）；反之越接近 0 代表障礙物距離越遠或數量較少（密度較低）。因此，若設定機器人之最大速度  $v_{max}$  與最小速度  $v_{min}$ ，其速度  $v$  可依下式計算：

$$v = v_{min} + (1 - D(d_i)) \cdot (v_{max} - v_{min}) \quad (3.16)$$

雖然使用障礙物密度來計算速度是相當有效的方式，但此密度僅使用當時所處的環境進行計算，並沒有將機器人當下的速度加入考慮。然而在進行路徑規劃時，若是機器人的速度過快，則機器人可能無法成功進行迴避，來不及做出轉向反應，因此機器人當下的速度也是一個相當重要的因素。

### 3.3.2 障礙物接近率

由於本論文使用的實驗平台 Yun-Trooper II 並不具備速度感測器，無法直接獲得機器人的速度。因此，本論文在原先的障礙物密度中增加一項障礙物接近率（Obstacle Approaching Rate） $\delta$ ，做為目前機器人速度的補償值。

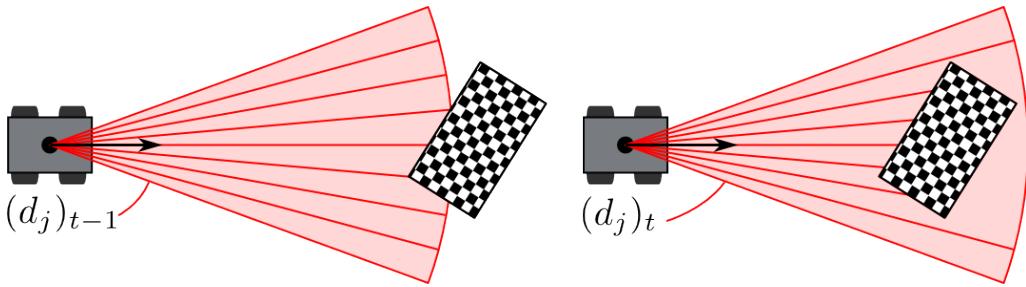


圖 3.27: 障礙物接近率示意圖

障礙物接近率  $\delta$  的概念為，假設機器人位於一靜態環境中（障礙物的位置不隨時時間而改變），則若是量測到的環境變化相當大，代表機器人的速度較高；反之則代表速度較低。而環境變化又以正前方的變化最為重要，因為這代表機器人接近正前方障礙物的速度，如圖 3.27 所示。因此，設定一角度範圍，例如正前方左右各  $20^\circ$ ，對所有位於此範圍內的光學雷達量測值  $d_j$ ， $\delta$  可依下式計算：

$$\delta = -\frac{1}{M} \sum_j \frac{(d_j)_t - (d_j)_{t-1}}{T} \quad (3.17)$$

其中  $(d_j)_t$  代表當下的環境量測值； $(d_j)_{t-1}$  代表上一次的環境量測值； $T$  代表兩次量測的時間間隔； $M$  代表範圍內的量測值個數。

對路徑規劃來說，最重要的是以高速接近障礙物時能夠成功的減速，對加速的要求則相對較低。因此，在計算環境變化率時，只將變化率  $d_t - d_{t-1}$  為負值（障礙物接近機器人）的方向納入計算，將變化率為正值的排除在外，得  $\delta_a$ ：

$$\delta_a = -\frac{1}{M} \sum_j \frac{\Delta((d_j)_t - (d_j)_{t-1})}{T} \quad (3.18)$$

其中

$$\Delta(d) = \begin{cases} d & \text{if } d < 0 \\ 0 & \text{otherwise} \end{cases}$$

為了讓障礙物接近率與環境密度能夠一起計算，必須讓障礙物接近率的範圍落在 0 與 1 之間。因此，將上式算出的  $\delta_a$  除以機器人的最大速度  $v_{max}$  便可得到

落在 0 與 1 之間的障礙物接近率  $\delta_n$ ：

$$\delta_n = -\frac{1}{M \cdot v_{max}} \sum_j \frac{\Delta((d_j)_t - (d_j)_{t-1})}{T} \quad (3.19)$$

$\delta_n$  之值越接近 1 代表障礙物以較高的速度接近機器人；反之則代表障礙物以較低的速度接近。

最後，將環境變化率  $\delta$  列入考慮後，速度  $v$  可依下式計算：

$$v = v_{min} + (1 - (D(d_i) + \delta)) \cdot (v_{max} - v_{min}) \quad (3.20)$$

### 3.3.3 碰撞偵測

先前於 3.2.3 中提到，因為避障演算法在某些情況下會發生誤判的情形，造成導航停止，所以本論文在機器人的導航過程中不斷偵測碰撞是否即將發生，使用碰撞偵測做為導航停止與否的指標，而非轉向方向。

碰撞偵測可分為兩階段：第一階段使用設定的機器人尺寸，偵測障礙物是否已經發生碰撞；第二階段是偵測機器人的行進方向是否有障礙物存在，偵測碰撞是否即將發生。只要其中一階段偵測到碰撞即代表碰撞發生，機器人將會停止前進。

#### 第一階段

在設定機器人尺寸時，為了安全起見，通常會讓設定的尺寸比真實尺寸還大。因此，第一階段的「已經發生碰撞」實際上代表的是，障礙物已經進入了設定的尺寸範圍內，而非真的發生了碰撞。而由於在計算時已經將機器人的尺寸簡化為半徑為  $w_s$  的圓形，因此只要判斷光學雷達的量測值  $d_i$  是否小於  $w_s$ ，即可判定碰撞是否發生，如圖 3.28 所示。

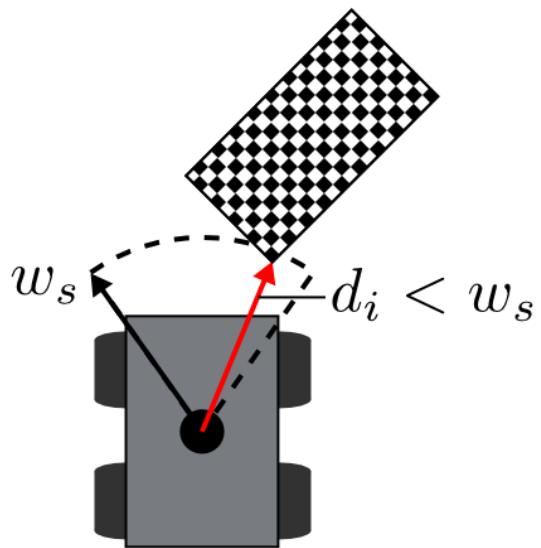


圖 3.28: 第一階段碰撞偵測示意圖

## 第二階段

若單純只使用機器人尺寸來偵測碰撞，在機器人速度較快時可能無法及時停止，讓機器人與障礙物實際發生碰撞，造成損壞。因此，第二階段設定一距離  $x$ ，利用車身尺寸  $w_s$  與轉向角度  $c_t$  建立一碰撞偵測範圍  $\theta$ ，預先偵測行進路線上的障礙物，以防止上述情況發生，如圖 3.29與式 3.21所示。

$$\theta = \arctan \frac{w_s}{x} \quad (3.21)$$

得到  $\theta$  後，可得到一角度範圍  $\theta_r : c_t + \theta \leq \theta_r \leq c_t - \theta$ 。在此角度範圍內的光學雷達量測值  $d_i$  若小於預先設定的距離  $x$ ，如圖 3.29所示，代表即將發生碰撞。

## 3.4 控制法則

綜合以上之演算法，機器人的轉向與速度控制法則使用虛擬碼表示如下：

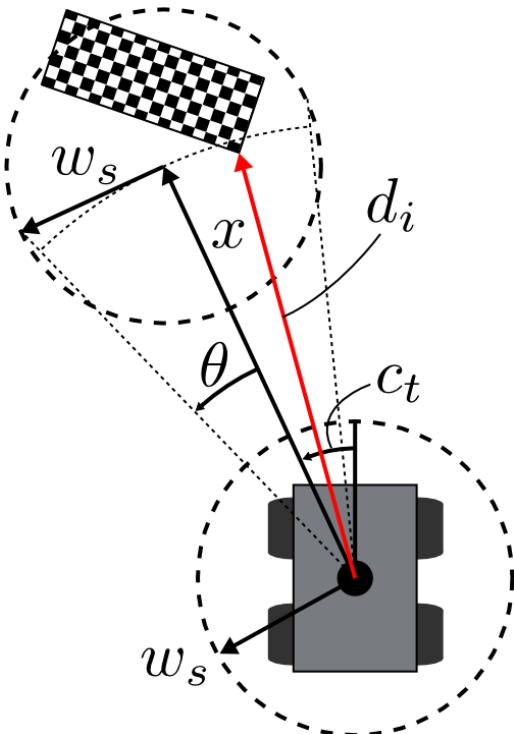


圖 3.29: 第二階段碰撞偵測示意圖

```

if  $c_t$  is available then
    steer  $\leftarrow K \cdot c_t$ 
    speed  $\leftarrow v$ 
else
    steer  $\leftarrow K \cdot c_{t-1}$ 
    speed  $\leftarrow v_{min}$ 
end if
if collision detected then
    speed  $\leftarrow 0$ 
end if
setCommand_Steer(steer)
setCommand_Speed(speed)

```

# 參考文獻

- [Borenstein and Koren(1991)] Borenstein, J. and Y. Koren, 1991: The Vector Field histogram - Fast Obstacle Avoidance for Mobile Robots. *IEEE Journal of Robotics and Automation*, **7 (3)**, 278–288.
- [B.V.(2012)] B.V., X. T., 2012: *MTi-G User Manual and Technical Documentation*.
- [El-Rabbany(2006)] El-Rabbany, A., 2006: *Introduction to GPS: The Global Positioning System*. 2d ed., Artech House.
- [Fox et al.(1997)] Fox, Burgard, and Thrun] Fox, D., W. Burgard, and S. Thrun, 1997: The Dynamic Window Approach to Collision Avoidance. *IEEE Robotics and Automation Magazine*, **4**, 23–33.
- [Jekeli(2006)] Jekeli, C., 2006: Geometric Reference Systems in Geodesy.
- [Karney(2013)] Karney, C. F. F., 2013: Algorithms for Geodesics. *Journal of Geodesy*, **87**, 43–55.
- [Karney(2014)] Karney, C. F. F., 2014: Geographiclib library. <http://geographiclib.sourceforge.net/html/index.html>.
- [Khatib(1985)] Khatib, O., 1985: Real-time Obstacle Avoidance for Manipulators and Mobile Robots. *IEEE Intl. Conf. on Robotics and Automation*.
- [Siegwart and Nourbakhsh(2004)] Siegwart, R. and I. R. Nourbakhsh, 2004: *Introduction to Autonomous Mobile Robots*. 1st ed., The MIT Press.
- [Simmons(1996)] Simmons, R., 1996: The Curvature-Velocity Method for Local Obstacle Avoidance. *IEEE Intl. Conf. on Robotics and Automation*.

[Ulrich and Borenstein(1998)] Ulrich, I. and J. Borenstein, 1998: VFH+: Reliable Obstacle Avoidance for Fast Mobile Robots. *IEEE Intl. Conf. on Robotics and Automation*, 1572–1577.

[呂明修 (2012)] 呂明修, 2012: GPS 與 IMU 應用於自主式四驅越野車輛之研究. M.S. thesis, 國立雲林科技大學.

[陳維懋 (2011)] 陳維懋, 2011: 自主式四驅越野車輛之開發. M.S. thesis, 國立雲林科技大学.

