

國立臺灣大學工學院機械工程學系  
碩士論文

Department of Mechanical Engineering  
College of Engineering  
National Taiwan University  
Master Thesis



在重複特徵環境下針對未知相對初始姿態之  
多機器人自主探索與地圖合併

Map Merging for Multi-Robot Exploration in Repetitive  
Environments with Unknown Initial Poses

賴重叡

Chung-Jui Lai

指導教授：詹魁元博士  
Advisor: Kuei-Yuan Chan, Ph.D.

中華民國 112 年 08 月

August, 2023

國立臺灣大學碩士學位論文  
口試委員會審定書

MASTER'S THESIS ACCEPTANCE CERTIFICATE  
NATIONAL TAIWAN UNIVERSITY

在重複特徵環境下針對未知相對初始姿態之  
多機器人自主探索與地圖合併

Map Merging for Multi-Robot Exploration in Repetitive  
Environments with Unknown Initial Poses

本論文係 賴重叡 (R10522614) 在國立臺灣大學機械工程學系完成之  
碩士學位論文，於民國 112 年 06 月 29 日承下列考試委員審查通過及  
口試及格，特此證明。

The undersigned, appointed by the Department of Mechanical Engineering on 29 June 2023 have  
examined a Master's thesis entitled above presented by Chung-Jui Lai (R10522614) candidate and  
hereby certify that it is worthy of acceptance.

口試委員 Oral examination committee:

(指導教授 Advisor)

蘇偉鴻

林昭廷

系主任/所長 Director: 林舜輝



---

## 誌謝

---

我衷心感謝指導教授詹魁元老師在整個研究過程中給予我們自由發揮的空間，並提供悉心指導和寶貴建議，您的學術洞察力對我的研究起到了重要的推動作用。特別感謝同屆同學提供的協助和支持。林易玄在碩士論文格式和離校流程方面給予了我許多指導和建議。林怡萱在修課方面給了我許多寶貴的建議。徐若瑄的協助對我的研究有著重要的幫助。李冠賢在口試投影片的修改上給予了我寶貴的意見。彭啟瑞在復興國中課程的場地布置幫助我許多。感謝博班學長江柏賢，你在實驗室中的健身經驗和鼓勵對我在健身方面提供了很多幫助。感謝博班學長黃彥智，你在研究相關的知識傳授和程式方面給予了我許多寶貴的指導和教學。同樣地感謝學長姊，郭冠成、鍾詔東、李亭宜、簡昱凡、陳昱霖的經驗分享和合作使我受益匪淺。而學弟妹們陳珮甄、蕭敬亭、謝鐘毅、柯琮祐、劉怡葶和張問蕖的支持讓我的研究生活更加豐富多彩。再次感謝親愛的父母的無私支持和鼓勵。你們是我生活中最重要的人，沒有你們的陪伴和支持，我將無法取得今天的成就。最後，衷心感謝口試委員林柏廷教授和蘇偉峻教授對我的研究提供寶貴的意見和建議。感謝所有在我研究過程中給予我支持和鼓勵的人們。因為你們的幫助，我能夠順利完成這篇論文。

賴重叡 謹識於  
國立臺灣大學 機械工程學系  
中華民國一百一十二年八月



---

## 摘要

---

高效的機器人導航需要預先建立的地圖，而建立地圖需要引導機器人自主探索未知區域。為了增加探索效率和範圍，通常會使用多台機器人進行協同探索，同步地建立環境地圖，以實現全面、高效的探索任務。然而，現有的多機器人自主探索，通常機器人間初始相對姿態是已知的，涉及具有未知初始相對姿態的多機器人自主探索非常罕見或未受關注。因此，本研究將重點放在未知初始相對姿態，因為在實際情況中，由於環境或通信限制，對機器人的狀態可能不完全了解。在初始相對姿態未知的情況下，地圖合併技術變得非常重要，因為此技術可以建立地圖之間的關聯。然而，目前的地圖合併技術在具有重複特徵的環境中面臨困難。因此，本研究提出了一種基於運動學分析和通信信號強度的新型地圖合併方法。最後，通過在具有重複特徵的環境中進行模擬實驗，證明了所提出的方法能夠有效地處理這些環境，同時在合理的時間內完成整個地圖的探索。

**關鍵字：**地圖合併、格點佔據地圖、機器人自主探索、重複特徵環境、多機器人系統



---

# Abstract

---

A predefined map is required for efficient robot navigation. This is created by directing the robot to explore unknown areas on its own. Multi-robot collaborative exploration is frequently used to improve exploration efficiency and coverage, but current strategies assume that the robots' initial relative poses are known. Situations involving robots with unknown initial relative poses received little to no attention, and this study focuses on them because, in practice, we may not have complete knowledge of robots due to environmental or communication constraints. When the initial relative poses are unknown, map merging becomes critical for establishing the relationship between maps. Current map merging techniques, however, face difficulties in environments with repetitive features. Based on kinematic analysis and communication signal strength, this study proposes a novel map merging method. Finally, simulations in environments with repetitive features are performed to demonstrate the proposed method's ability to handle these environments efficiently while exploring the entire map in a reasonable amount of time.

**Keywords:** Map merging, Occupancy grid map, Autonomous exploration, Repetitive features, Multi-robot systems

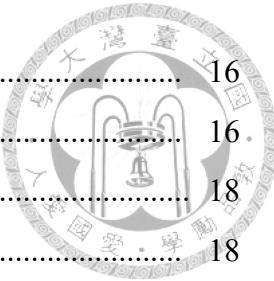


---

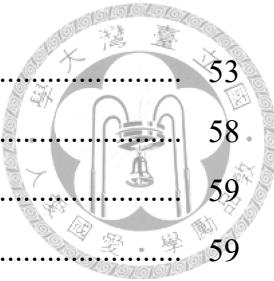
# 目錄

---

口試委員會審定書 .....	i
誌謝 .....	ii
摘要 .....	iii
Abstract .....	iv
目錄 .....	v
圖目錄 .....	viii
表目錄 .....	xi
第一章 緒論 .....	1
1.1 前言 .....	1
1.2 自主探索 .....	1
1.3 多機器人系統 .....	3
1.4 研究動機與目的 .....	3
1.5 本文架構 .....	4
第二章 文獻回顧 .....	6
2.1 機器人地圖資料型態 .....	6
2.2 多機器人自主探索 .....	8
2.2.1 基於邊界的探索演算法 .....	8
2.2.2 基於隨機運動規劃技術的探索演算法 .....	10
2.2.3 多機器人自主探索之小結 .....	11
2.3 格點佔據地圖合併方法 .....	11
2.3.1 直接類 .....	12
2.3.2 間接類 .....	13



2.3.3 格點佔據地圖合併方法之小結 .....	16
2.4 文獻回顧總結 .....	16.
<b>第三章 研究流程與架構</b> .....	<b>18</b>
3.1 研究假設 .....	18
3.2 問題定義 .....	19
3.3 研究方法與架構 .....	21
3.3.1 格點佔據地圖合併架構 .....	21
3.3.2 多機器人自主探索架構 .....	23
<b>第四章 未知初始相對姿態多機器人自主探索</b> .....	<b>27</b>
4.1 用語與符號定義 .....	28
4.2 基於 RRT 之邊界檢測器 .....	29
4.2.1 局部邊界檢測器 .....	29
4.2.2 全域邊界檢測器 .....	30
4.3 過濾器 .....	32
4.3.1 均值偏移聚類演算法 .....	32
4.3.2 刪除邊界 .....	33
4.4 單機器人任務分配 .....	34
4.5 多機器人任務分配 .....	36
4.5.1 避免重複探索之方法 .....	37
4.5.2 基於市場經濟的分配 .....	39
4.6 全域路徑規劃 .....	39
4.6.1 用語與符號定義 .....	41
4.6.2 RRT Star 演算法 .....	41
4.7 局部路徑規劃 .....	42
4.8 對現有自主探索架構的修改 .....	47
<b>第五章 提出之地圖合併方法</b> .....	<b>48</b>
5.1 信號強度預測距離之模型 .....	48
5.1.1 信號強度模型 .....	49
5.1.2 信號強度資料處理 .....	50
5.1.3 預測距離模型及修正 .....	52



5.2 運動學分析計算變換關係 .....	53
5.3 相對變換關係接受指標 .....	58.
5.4 修改之特徵地圖合併方法 .....	59
5.4.1 ORB 特徵擷取 .....	59
5.4.2 窮舉特徵匹配 .....	64
5.4.3 特徵修正變換關係 .....	65
5.5 地圖合併處理 .....	67
5.5.1 計算佔據機率 .....	67
5.5.2 熵濾波合併地圖 .....	69
5.6 本研究地圖合併演算法之虛擬碼 .....	71
<b>第六章 模擬環境測試 .....</b>	<b>73</b>
6.1 模擬設定 .....	73
6.2 現有間接地圖合併演算法之效用 .....	75
6.2.1 現有演算法特徵擷取與匹配測試 .....	75
6.2.2 現有演算法合併測試 .....	76
6.3 本研究地圖合併演算法之效用 .....	78
6.3.1 預測距離模型測試 .....	78
6.3.2 運動學分析合併測試 .....	79
6.3.3 修改之特徵地圖合併方法測試 .....	80
6.4 本研究多機器人自主探索測試與討論 .....	84
<b>第七章 總結與未來展望 .....</b>	<b>87</b>
7.1 結論 .....	87
7.2 研究建議與未來展望 .....	88
<b>附錄A 機器人自主探索測試數據 .....</b>	<b>89</b>
A.1 參考之單機器人自主探索方法結果 .....	89
A.2 參考之多機器人自主探索方法結果 .....	94
A.3 提出之多機器人自主探索方法結果 .....	99
<b>參考文獻 .....</b>	<b>105</b>

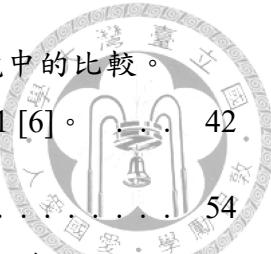


---

# 圖目錄

---

1.1	自主探索架構關係圖 [1]。 . . . . .	2
1.2	個體局部地圖之相對姿態關係示意圖， $x, y, \theta$ 表示個體的參考座標系， $\mathbf{T}_{ij}$ 表示兩個體的相對姿態關係。 . . . . .	4
2.1	格點佔據地圖示例 [2]。 . . . . .	7
2.2	特徵地圖示例 [2]。 . . . . .	7
2.3	拓樸地圖示例 [2]。 . . . . .	8
2.4	Yamauchi 邊界檢測：(a) 格點佔據地圖，(b) 邊界邊緣，(c) 邊界區域 [3]。 . . . . .	9
3.1	Ferrão et.al 所提出的特徵地圖合併方法 [4] 修改版本架構。 . . . . .	22
3.2	本文提出之地圖合併方法架構。 . . . . .	23
3.3	本文多機器人自主探索之參考方法 [1]。 . . . . .	24
3.4	本文單機器人自主探索之參考方法 [5]。 . . . . .	25
3.5	本文多機器人自主探索之研究方法，主要將單機器人及多機器人的自主探索架構進行整合，並加入了地圖合併。 . . . . .	26
4.1	局部 RRT 生長和檢測邊界的過程。 . . . . .	30
4.2	全域 RRT 生長和檢測邊界的過程。 . . . . .	31
4.3	邊界之導航成本 ( $N$ )。 . . . . .	34
4.4	邊界之信息增益區域。 . . . . .	35
4.5	機器人目標點 A 點，對邊界 B 的信息增益應扣除重疊區域 D。 . . . . .	38



4.6 (a) RRT 和 (b) RRT Star，在具有障礙物的模擬環境中的比較。 RRT 和 RRT Star 中最佳路徑的成本分別為 21.02 和 14.51 [6]。	42
5.1 以連桿機構進行地圖合併概念圖。 . . . . .	54
5.2 以局部地圖 $M_2$ 為基準，使用距離計算變換關係之符號定義。 . . . . .	55
5.3 高斯及高斯差分影像金字塔計算示意圖 [7]。 . . . . .	61
5.4 FAST 特徵檢測演算法之分割測試。 $p$ 是中心像素，半徑為 3 的圓由虛線表示，通過 16 個像素（突出顯示的正方形）。[8] . . . . .	61
6.1 本文研究測試用地圖。 . . . . .	73
6.2 每個案例構建的局部地圖。 . . . . .	74
6.3 兩個機器人的移動路徑。 . . . . .	74
6.4 SIFT 演算法之特徵擷取效用呈現，綠色圓圈表示 SIFT 特徵位置與主方向。 . . . . .	75
6.5 特徵匹配示意圖，紅框表示錯誤配對的特徵匹配。 . . . . .	76
6.6 現有演算法合併結果可視化示意圖，綠色區域為正確匹配，紅色區域為錯誤匹配。 . . . . .	77
6.7 預測距離模型所預測之兩機器人相對距離，粉色線為預測距離，藍色線為實際真實距離。 . . . . .	78
6.8 運動學分析合併結果可視化示意圖，綠色區域為正確匹配，紅色區域為錯誤匹配。 . . . . .	80
6.9 ORB 演算法之特徵擷取效用呈現，綠色圓圈表示 ORB 特徵位置與主方向。 . . . . .	81
6.10 特徵窮舉匹配示意圖。 . . . . .	82
6.11 特徵修正合併結果可視化示意圖，綠色區域為正確匹配，紅色區域為錯誤匹配。 . . . . .	84
6.12 各機器人自主探索之統計模擬行走距離結果，藍色為現有單機器人方法之結果 [5]，橘色為現有多機器人方法之結果 [1]，綠色為提出方法之結果。 . . . . .	85

6.13 多機器人自主探索之統計模擬花費時間結果，藍色為現有單機器人方法之結果 [5]，橘色為現有多機器人方法之結果 [1]，綠色為提出方法之結果。 . . . . .

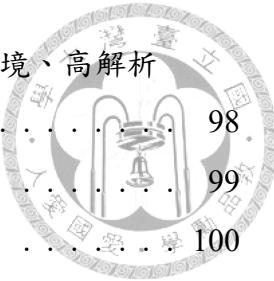


86



# 表目錄

5.1	路徑損耗模型參數數值表。	50
5.2	變換關係接受指標 $\omega$ 分類及處理關係表。	58
5.3	修正後變換關係接受指標 $\omega$ 分類及處理關係表。	67
5.4	格點佔據地圖之格點資訊關係表。	69
6.1	測試案例之地圖參數。	75
6.2	不同方法對於各個案例的測試結果整理。	85
A.1	參考單機器人自主探索方法之案例一結果（低重複特徵環境、低解析度）	89
A.2	參考單機器人自主探索方法之案例二結果（低重複特徵環境、高解析度）	90
A.3	參考單機器人自主探索方法之案例三結果（重複特徵環境、低解析度）	92
A.4	參考單機器人自主探索方法之案例四結果（重複特徵環境、高解析度）	93
A.5	參考多機器人自主探索方法之案例一結果（低重複特徵環境、低解析度）	94
A.6	參考多機器人自主探索方法之案例二結果（低重複特徵環境、高解析度）	95
A.7	參考多機器人自主探索方法之案例三結果（重複特徵環境、低解析度）	96



A.8 參考多機器人自主探索方法之案例四結果（重複特徵環境、高解析度）	98
A.9 提出方法之案例一結果（低重複特徵環境、低解析度）	99
A.10 提出方法之案例二結果（低重複特徵環境、高解析度）	100
A.11 提出方法之案例三結果（重複特徵環境、低解析度）	101
A.12 提出方法之案例四結果（重複特徵環境、高解析度）	103



# 緒論

## 1.1 前言

未知環境的自主探索一直是機器人領域的一個持續研究課題。雖然最初使用單個機器人進行自主探索，但近年來重點已轉向多機器人系統（Multi-Robot System, MRS），可以利用多個機器人實現高效率和高精度的地圖同時構建。但是，由於各個機器人的參考坐標系不同，即使機器人之間可以通信，也需要將採集到的數據進行整合，建立相對關係。這意味著每個機器人構建局部地圖後，需要一個演算法來合併每個機器人的局部地圖。隨著自主探索的不斷發展，了解多機器人在自主探索中遇到的問題至關重要。因此，接下來將分別介紹自主地圖探索和 MRS 技術，再探討同時應用兩種技術時可能面臨的挑戰和限制，並提出本研究的動機和目的。

## 1.2 自主探索

自主探索（Autonomous Exploration）是指探索未知環境以建立後續導航地圖的過程 [3]。這個過程必須高效進行，以在最短距離內獲取最大量的信息。探索過程包括避開已探索的區域和優先考慮未探索的區域。如圖 1.1 所示，探索過程的階層結構包括探索（Exploration）、路徑規劃（Path Planing）和機器人定位（Localization）。

層次結構中的頂層是探索，探索未知區域需要機器人在全域坐標系中的位置



和方向，且需要一條探索的路徑。探索包含探索策略及探索地圖建構，探索策略負責尋找探索的目標點，而探索地圖建構則負責更新地圖。探索可以讓機器人從未知的環境中獲取新的信息，提高機器人的自主性和適應性，因此，探索在自主地圖探索中被視為最重要的一步，是層次結構的頂層。

層次結構中的第二層是路徑規劃，探索策略在環境中尋找到目標點，機器人必須使用探索構建的地圖和機器人的當前位置和目標位置來計算最佳路徑，並規劃路徑以避開障礙物，到達目標點，同時使用感測器數據更新地圖，以確保地圖的準確性。

層次結構的底層為定位，即機器人需要具備知道當前姿態（Pose）的能力，也就是知道當前的位置和方向，以便進行路徑規劃和探索，目標點和當前姿態必須使用固定的全域框架來表示，以便統一距離傳感器讀數（例如來自光達（Light Detection And Ranging, LiDAR）的讀數），也就是從傳感器坐標系轉換為全域坐標系。此外，如果不使用絕對定位系統，地圖也可用於相對定位，這種方法被稱為同時定位與地圖構建（Simultaneous Localization And Mapping, SLAM），其通常使用濾波器（例如卡爾曼濾波器（Kalman Filter）[9]）等技術將感測器數據與地圖匹配，以計算機器人的姿態，將感測器雜訊和不確定性降到最低。

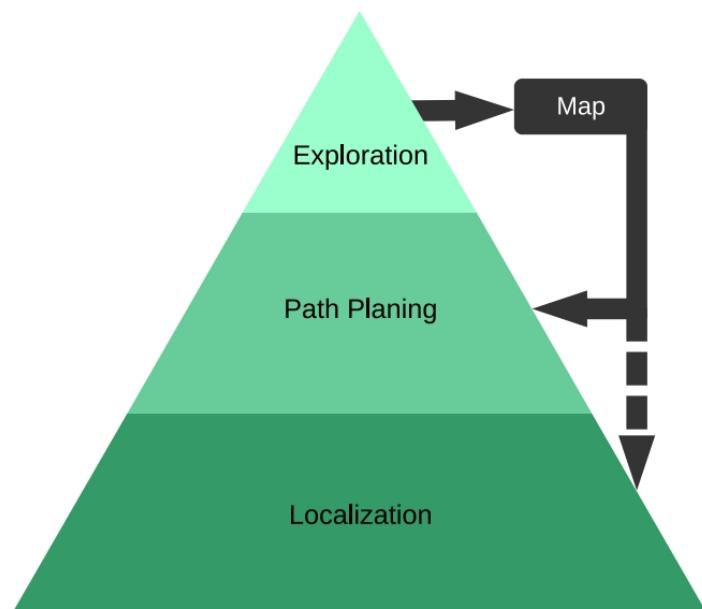


圖 1.1: 自主探索架構關係圖 [1]。



根據上述可以了解到，自主探索是一個包含多個問題的複雜領域，包括機器人探索策略、自我定位、地圖構建、路徑規劃等。而下一節將介紹多機器人系統，以及其如何提昇探索效率。

### 1.3 多機器人系統

在探索環境中，效率是一個關鍵問題，特別是當個體能力有限時。合作是提高效率的一種手段，這個概念在人類生活中和機器人領域都得到廣泛應用。MRS 涉及多個機器人的協作、共享信息，從而形成更多元化的感知能力和多種策略解決方案，大大提高了探索效率。在探索環境時，多個機器人的去中心化探索大大加速了覆蓋範圍。此外，MRS 提高了系統的可靠性，因為單個機器人的失敗不會導致任務失敗，其他機器人可以繼續進行操作，從而賦予系統強大的韌性。

在 MRS 中，通訊和協作至關重要。然而，隨著機器人數量的增加，感測資訊的大小也增加，使數據整合和相關處理變得具有挑戰性。多機器人協作技術包括集體決策、分工和協作、任務分配等活動。根據任務的不同，必須建立必要的主從關係或群體，進一步增加系統的複雜性。

儘管多機器人系統在工業、環境監測和救援等各個領域得到了廣泛應用（例如，在救援領域，它們可以承擔搜索和救援等任務），但仍然存在一些挑戰。因此，未來需要進行更深入的研究和探索，以全面解決這些問題，促進 MRS 在各種領域的廣泛應用。下一節將概述自主探索和 MRS 的整合，並說明本文研究目標及動機。

### 1.4 研究動機與目的

為了達成多台機器人同時自主探索環境的各項需求，必須將前兩節提及的機器人自主探索與 MRS 技術進行整合，並加入對各個機器人建構之地圖進行整合的技術，可稱作「多機器人自主探索（Autonomous Multi-Robot Exploration）」。多機器人自主探索中，機器人個體各自依據探索策略取得的目標點，以及利用通訊協定交換彼此感測資料並進行資料關聯的處理來更新地圖，以加快環境探索覆蓋率。但現有多機器人自主探索方法限制了機器人之間的初始相對姿態必須已知，也就是地圖整合技術只是負責整合局部地圖為全域地圖，並無計算地圖相對關係

(圖 1.2 中的  $T_{ij}$ )，也因此機器人必須擺放在一起，才容易量測取得相對姿態。然而，在多機器人自主探索中，並無法預期個體已知彼此初始相對姿態，例如在建築物救援探勘時，當有多個入口，且入口狹小無法擺放多台機器人時，機器人勢必需要分開擺放，這樣就無法輕易取得相對姿態，也就無法使用現有的方法，也因此必須有計算地圖相對關係並整合之技術，此技術又被稱為「地圖合併 (Map Merging)」。

不幸的是，現有的地圖合併技術在具有重複特徵的環境中面臨挑戰，因此，需要一種能夠應對不同環境特徵的地圖合併方法，以提高多機器人自主探索的效率和準確性。因此，本文希望探討機器人間未知初始相對姿態，且在重複特徵環境的情況下，如何僅藉由個體所具有的資訊使局部地圖進行合併，並進行多機器人自主探索。

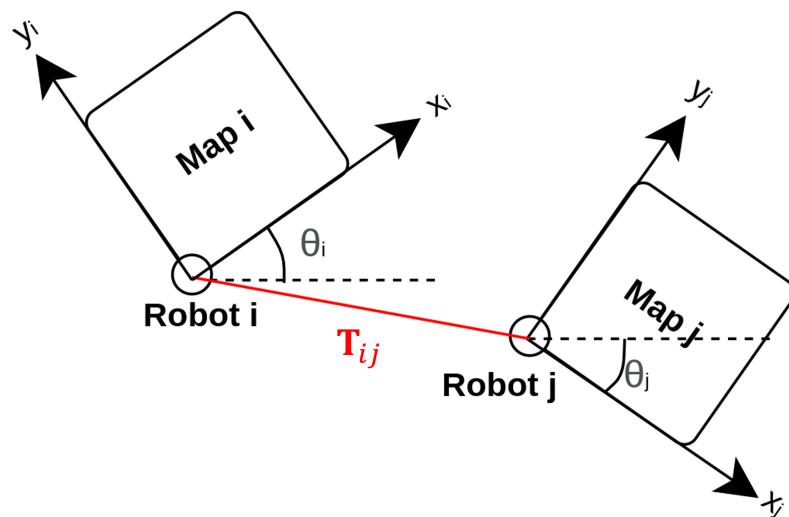


圖 1.2: 個體局部地圖之相對姿態關係示意圖， $x, y, \theta$  表示個體的參考座標系， $T_{ij}$  表示兩個體的相對姿態關係。

## 1.5 本文架構

本文一共分為七章，內容架構如下：

### 1. 第一章：緒論

介紹本研究主題的相關背景，說明研究動機與目的，並呈現全文架構。



## 2. 第二章：文獻回顧

探討本研究領域的文獻與相關方法。

## 3. 第三章：研究流程與架構

解釋本研究所使用方法的基本邏輯及相關流程。

## 4. 第四章：未知初始相對姿態多機器人自主探索

詳細說明本研究修改之多機器人自主探索的各項演算法。

## 5. 第五章：提出之地圖合併方法

說明本研究提出之地圖合併方法的各項演算法。

## 6. 第六章：模擬環境測試

以模擬環境測試本研究方法之效果，並加以分析。

## 7. 第七章：結論與討論

總結本研究內容，並提出未來研究可改進與發展之方向。



## 文獻回顧

在前一章中提到，為了有效率地實現大型環境的自主地圖建構，必須使用多機器人技術。然而，在多機器人自主探索中，由於無法預期個體已知彼此初始相對姿態，所以必須使用地圖合併之技術。本文的研究重點是多機器人自主探索及地圖合併問題，因此接下來會探討多機器人自主探索及地圖合併的相關文獻，但在討論問題之前，必須先了解機器人地圖的不同格式種類，本節將先介紹應用於機器人地圖建構的數種資料型態。

### 2.1 機器人地圖資料型態

根據不同的環境資訊處理方式，地圖可以分為三種類型：

1. 格點佔據地圖 (Occupancy Grid Map)：是一種將環境劃分為一系列格點的地圖。每個格點都有一個機率值，分為未知、自由和佔據區域三種 [10]，如圖 2.1 所示。格點佔據地圖的優點在於解析度易於調整 [11]，並且可以快速、簡單地應用到任何環境 [12, 13]。缺點在於在較大的場域或高解析度的情況下，地圖需要使用較多的格點，因此佔用較多的記憶體空間 [12]。

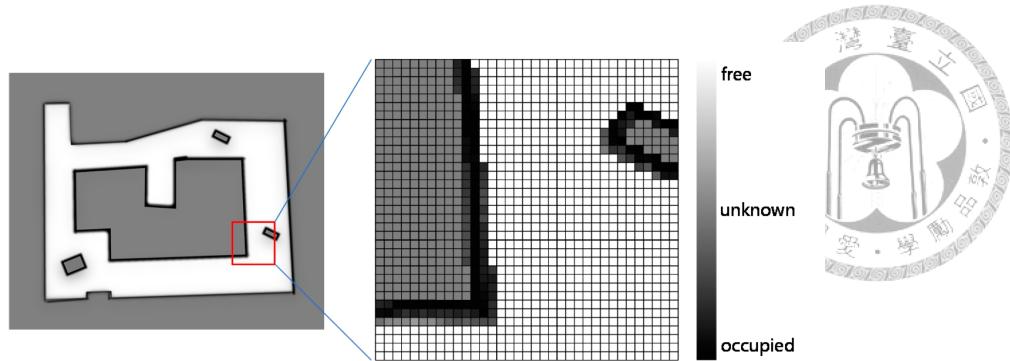


圖 2.1: 格點佔據地圖示例 [2]。

2. 特徵地圖 (Feature Map): 是一種基於收集到的感測數據提取和存儲特徵部分的地圖，如圖 2.2 所示。與網格佔用圖不同，特徵地圖只存儲部分感測資料，從而減少佔用的記憶體空間。然而，這張地圖需要大量的計算能力來過濾和擷取感測數據，因此計算成本相對較高。特徵地圖的實際應用例子包括從 2D LiDAR 數據中提取環境線特徵 [14] 和從深度相機獲得的點雲數據中提取牆平面特徵 [15]。

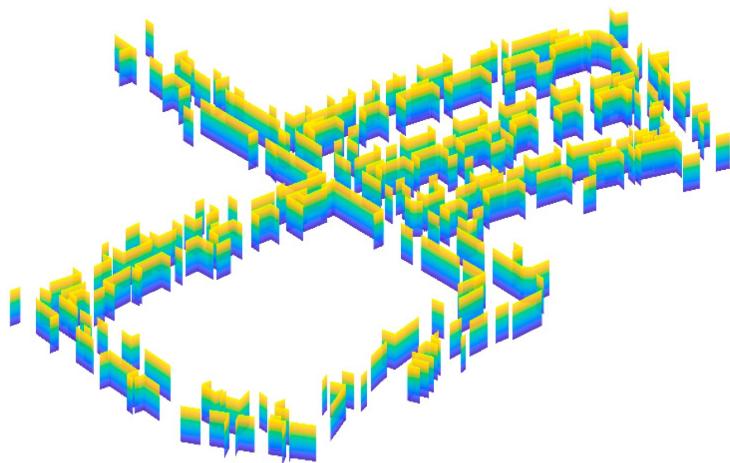


圖 2.2: 特徵地圖示例 [2]。

3. 拓撲地圖 (Topological Map): 是通過提取環境中突出的地標或特徵，存儲為節點 (Node)，計算節點之間的連接或路徑關係，存儲為邊 (Edge)，如圖所示 2.3。環境地標的網絡關係可以通過節點和邊的概念來構建。例如，在建築物的拓撲圖中，每個房間都可以表示為一個節點，連接房間或區域的門或走廊可以表示為邊。拓撲圖不像其他地圖那樣表示環境樣貌，而是表示

地標之間的相對位置和網絡關係。因此更常用於定位 [16]、導航 [17]、路徑規劃 [18] 等特定任務。

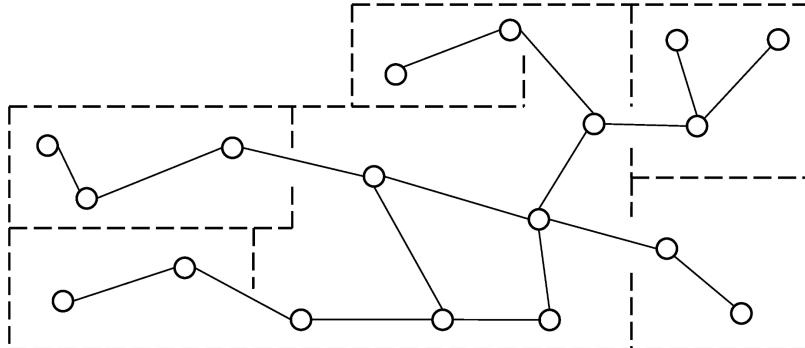


圖 2.3: 拓樸地圖示例 [2]。

在三種類型的地圖中，格點佔據地圖是最常用的類型，表達形式簡單。特徵地圖旨在通過減少數據量並僅保留用於各種任務的顯著特徵來改進格點佔據地圖。雖然拓樸地圖不能直接表示環境樣貌，但它具有對應用任務高效的節點連接關係。在本研究中，採以多數人使用的格點佔據地圖作為問題假設，因此接下來多機器人自主探索及地圖合併技術主要回顧格點佔據地圖的類型。

## 2.2 多機器人自主探索

多機器人自主探索雖然是涵蓋多個問題的複雜領域，但現有研究多聚焦於探索策略部分，因為探索策略是實現自主探索的核心，探索策略需要機器人在全域坐標系中的姿態以及地圖，目標是尋找探索的目標點，到達目標點可以讓機器人從未知的環境中獲取新的信息，提高機器人的自主性和適應性。對於格點佔據地圖的多機器人自主探索依據探索策略架構可以分為基於邊界 (frontier) 及基於隨機運動規劃技術，兩者主要思想都是確保機器人探索未知空間並避免已經探索過的空間，以下將分別介紹這兩種方法的背景和方法流程。

### 2.2.1 基於邊界的探索演算法

Yamauchi [3] 是最早使用基於邊界的探索演算法的研究人員之一。這種演算法可以讓機器人自主探索並建構複雜環境，例如一個堆滿傢俱和障礙物的辦公室。

該演算法基於將機器人引向邊界，所謂邊界是指“自由區域和未知區域之間的交界區域”[3]，直到整個區域探索完成。該演算法在一個真實的辦公室環境中用一個機器人進行了測試，圖 2.4 顯示了檢測邊界的過程，其中機器人將嘗試探索其最接近的區域。圖 2.4 (a) 中白色區域代表已知自由空間，大黑點代表已知佔據空間，小黑點區域代表未知空間，檢測到的三個邊界區域中心在圖 2.4 (c) 中用圓圈“+”標記。許多的自主探索沿襲 Yamauchi 的研究，因此出現了許多基於邊界的探索演算法。Yamauchi 在 [19] 的研究中，進一步將演算法應用於多機器人，每個機器人之間已知初始相對姿態，且每個機器人都有局部地圖，當機器人到達一個新的邊界之後才更新局部地圖，並把局部地圖共享給其他所有的機器人進行合併，並以協作、分散的方式探索，但該演算法可能會導致重複探索，因為每個機器人的探索獨立於其他機器人運行，並且邊界是在每個機器人的局部地圖計算的，因此兩個機器人最終可能會探索相同的邊界 [20]。

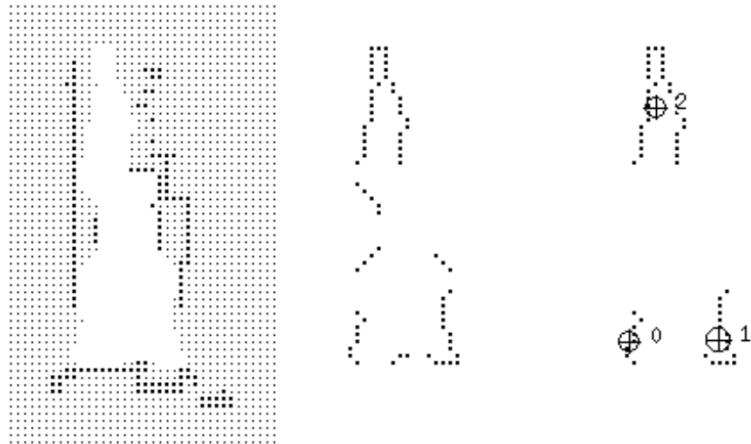


圖 2.4: Yamauchi 邊界檢測：(a) 格點佔據地圖，(b) 邊界邊緣，(c) 邊界區域 [3]。

Wang et al. [21] 的研究中，提出了一種解決 Yamauchi 演算法 [19] 在多機器人情況下存在的重複探索問題的方法。該解決方法基於將地圖劃分為子區域，其中每個機器人都不會嘗試進入另一個機器人的子區域，且每個機器人之間已知初始相對姿態。機器人將探索子區域直到沒有檢測到額外的邊界區域，即子區域被完全探索，機器人才會嘗試找到下一個要探索的子區域。

在 Yamauchi [19]，每當機器人到達目標點時都必須進行邊界檢測，然而這個過程消耗的計算資源相當高 [22]，因為每次運行都必須掃描整個地圖。為了節省計算資源，通常避免頻繁地計算邊界，因為在某些情況下這會導致不必要的冗餘



探索任務 [20]。

Yan et al. [23] 的研究中，使用了基於邊界的探索演算法，每個機器人之間已知初始相對姿態，並研究了機器人數量對探索效率的影響。該研究以探索所需時間和所有機器人的能量消耗為衡量標準，發現機器人數量越多，探索效率越高。

### 2.2.2 基於隨機運動規劃技術的探索演算法

使用隨機運動規劃技術代替基於邊界的方法，機器人可以被引導到未探索的空間，而無需檢測邊界區域。[24] 是最早使用基於隨機運動規劃技術的探索演算法的研究，其使用快速探索隨機樹（Rapidly-exploring Random Tree, RRT）演算法，形成的節點樹會偏向於往未探索空間擴展，這使其成為被動檢測探索邊界的理想選擇。Oriolo et al. [25] 的研究中，提出了一種基於感測器隨機樹（Sensor-based Random Tree, SRT）的探索策略，其為基於 RRT 概念的改進版本。相對於 RRT，SRT 在生成隨機目標點時限制這些目標點必須位於機器人的感測器範圍內，這是 SRT 和 RRT 的一個重要區別，也因此使得 SRT 更適合深度優先探索。然而，由於這種差異，SRT 的方法需要回溯，也就是當節點樹的分支停止增長時（例如，當機器人到達死路），機器人必須返回先前的位置。但是，在回溯過程中可能會重複探索同一個地方，這是 SRT 的一大缺陷。Franchi et al. 在 [26] 的研究中，將之前基於 SRT 的方法應用於多個機器人。其提出的基於 SRT 的多機器人探索方法是一種分散的方法，每個機器人之間已知初始相對姿態，並構建基於機器人個體初始姿態的 SRT。當機器人個體不能再擴展自己的節點樹後，會嘗試幫助其他機器人擴展節點樹，這就是所謂的“支持階段”，並且所有機器人個體都會擴展一顆節點樹。Franchi et al. 在 [27] 中進一步改進了這項研究，通過在節點樹上的可連接節點間引入直接快捷路徑，有利於支持階段的節點樹擴展。

Umari et al. [5] 的研究中，提出了一種基於使用多個 RRT 的自主探索。機器人個體同時具有一顆局部 RRT 與一顆全域 RRT。局部 RRT 趨向於探索機器人個體附近的邊界，全域 RRT 則趨向於探索整個地圖的邊界，從而實現高效的機器人探索。Umari et al. 在 [1] 的研究中，將此方法應用於多機器人，並採用基於市場經濟（Market Economy）的多機器人協作任務分配策略 [28]，每個機器人個體以最大化利潤的方式被分配任務，未探索區域信息的獲取就是收益，達到目標的預期距離就是成本，整體結果顯示該策略是一個有效的探索任務分配方式。同時模擬

和實驗結果表明，提出的策略可以成功地提取邊界，並在合理的時間內探索整個地圖，降低地圖探索成本。



### 2.2.3 多機器人自主探索之小結

基於邊界的探索演算法會透過掃描整個地圖的邊界來取得邊界資訊，雖然可以取得較多且完整的邊界，但執行時需要較多計算資源，速度較慢。而基於隨機運動規劃技術的探索演算法則會往未探索空間擴展隨機節點樹，透過找出邊界的方式來探索地圖，效率較高且不需要每次都掃描整個地圖，但找出的邊界是隨機且不完整的。而 [1] 使用多個 RRT 的方法可以改善基於隨機運動規劃技術的探索演算法找出的邊界不完整的問題，同時仍保有高效率的優點。應用於多機器人時，加入了協作任務分配策略。目前表現較佳的協作任務分配策略包括 [21] 所提出的基於將地圖劃分為子區域的方法，以及 [1] 所運用的基於市場經濟的方法 [28]。同時，可以發現多機器人自主探索皆是假設機器人之間已知初始相對姿態來進行，但在實際應用中這是無法預期的，因此需要使用地圖合併技術來取得機器人之間的相對姿態，因此下一節將對地圖合併技術進行詳細介紹。綜合上述，本研究選擇使用多個 RRT 的多機器人自主探索 [1] 進行修改，因為其具有高完整性邊界、高效率，以及表現良好的協作任務分配策略等優勢。

## 2.3 格點佔據地圖合併方法

地圖合併的關鍵問題在於獲取機器人局部地圖之間的相對關係，當關係已知時，合併機器人生成的局部地圖就變得相對簡單，也就是可以計算到全域參考系的變換，並在全域參考系中覆蓋轉換後之地圖。H.C Lee [29] 對格點佔據地圖合併技術進行了全面的調查，將演算法分為直接地圖合併和間接地圖合併。這兩種方法的邏輯架構有很大的不同，並且問題的發展也被劃分為早期和晚期，早期方法主要屬於直接類方法，而近年來提出的方法則主要屬於間接類方法。以下將分別介紹這兩種方法的背景和方法流程。



### 2.3.1 直接類

直接類方法主要是透過通訊找出個體感測資料的相關性，進而建立相對關係和地圖合併。這些感測資料的相關性可以根據觀察環境資訊來實現，這些環境資訊可分為兩類：

1. **觀測個體資訊的方法**：為了獲取不同機器人之間的相對關係，最簡單直接的方法是觀察其他機器人與自己的相對位置和關係。透過感測器，機器人可以決定通訊範圍，觀察周圍的可合作對象，並利用感測資料處理與其他機器人的相對關係。

Konolige [30] 的研究中，地圖合併被分為兩個步驟：建立假設、驗證假設。建立假設中，也就是當機器人觀察到彼此時，會開始建立通訊並開始接收其他機器人的感測數據，並嘗試建立兩者之間的相對關係，也就是地圖變換矩陣（Map Transformation Matrix，MTM）。在驗證假設中，機器人會利用前一階段建立的 MTM，嘗試於某個地點進行會面，若會面成功，則代表假設得到驗證，地圖進行合併，並且機器人開始進入協作。

Zhou [31] 預先設定好會合地點，並使用全景相機偵測安裝在其他機器人上的彩色圓柱體，來建立 MTM，求得相對關係以進行地圖合併。然而，由於矩陣存在較大的誤差，Zhou [31] 同時提出了一種修正地圖的方法，通過辨識環境特徵進行合併地圖的修正。

2. **辨識共同物件或區域的方法**：當機器人無法直接觀測到彼此時，可以透過參照相同物件來建立彼此之間的關係。透過參照相同物件，機器人可以形成三角關係並計算彼此的相對位置。這些相同物件可能是環境中的物體，也可能是辨識出的相同區域。

Lee et al. [32] 將機器人裝上向上的相機，用來觀測周圍的天花板影像，並從影像中提取和存儲影像特徵。當兩個機器人的地圖有重疊的部分時，這代表它們觀察到相同的影像特徵。因此可以使用最佳化演算法來搜索這些地圖特徵之間的對應關係，以計算出兩個機器人之間的相對姿態差異，並完成地圖合併。

Tungadi et al. [33] 使用全景相機和基於 Haar 小波的機率地區識別系統，來檢



測機器人當前位置是否與先前存取或運行的地圖相符。如果是前者，這種方法用於閉環檢測，有助於地圖的建立；如果是後者，它可以與存取的地圖合併。Tungadi 的方法中，Haar 小波系統可以計算出當前位置相對於參考原點的角度，提高了掃描匹配演算法驗證系統輸出結果的收斂速度和準確度，因此提高了方法的穩健性。

在 [34] 中，通過匹配相機識別的二維視覺對象來計算相對關係，當視覺對像被識別時，機器人之間的相對位置和方向被估計並用於計算 MTM。

### 2.3.2 間接類

間接類方法是基於地圖影像進行處理的方法。地圖可以看作是圖片影像的一種，利用格點像素的縮放、旋轉、平移等方式進行對齊並完成地圖合併。對齊方式主要可以分為兩種，一種是利用最大化演算法來搜尋地圖影像之間的相對關係，另一種是透過擷取與匹配地圖影像的特徵來計算相對關係。下面將介紹這兩種不同的對齊方式：

1. **最大化搜尋方法：**最大化搜尋方法中，必須先定義地圖像素相關的目標函數，並使用演算法進行求解。根據格點佔據地圖的分類特性，目標函數可定義為最大化兩地圖的重合程度，也就是最大化格點像素種類的正確配對或最小化錯誤配對。以 Carpin et al. [35] 的方法為例，其首先定義了地圖重疊函數如下：

假設局部地圖  $\mathbf{M}_1$  與  $\mathbf{M}_2$  屬於一大小為  $[N \times M]$  的全域地圖  $\mathbf{M}_{N \times M}$  之子集，其中  $N, M$  為兩個正實數。若座標系統一的情況下，Carpin et al. [10, 35] 將兩個局部地圖  $\mathbf{M}_1$  與  $\mathbf{M}_2$  之間的重疊定義如下：

$$\omega(\mathbf{M}_1, \mathbf{M}_2) = \sum_{i=0}^{N-1} \sum_{j=0}^{M-1} \text{Eq}(\mathbf{M}_1[i, j], \mathbf{M}_2[i, j]) \quad (2.1)$$

其中函數  $\text{Eq}(\mathbf{M}_1[i, j], \mathbf{M}_2[i, j])$  表示若格點  $[i, j]$  位置的兩像素值相同，則輸出 1，反之輸出 0，根據上述函數定義，則可將目標函數定義為：

$$\max_{x, y, \theta} \omega(\mathbf{M}_1, \mathbf{T}\mathbf{M}_2) \quad (2.2)$$

其中操作變數為  $x, y, \theta$ ， $\theta$  表示兩個圖之間的旋轉角度， $x$  和  $y$  表示兩個圖之間的縱向和橫向平移。地圖相對變換關係矩陣  $\mathbf{T}$  為

$$\mathbf{T} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & x \\ \sin(\theta) & \cos(\theta) & y \\ 0 & 0 & 1 \end{bmatrix} \quad (2.3)$$



地圖合併的目標函數並非凸函數，可行解區域中可能具有數個極值點，且局部最大、最小值點可能非常鄰近。現有的求解演算法研究包括 Locatelli et al. 所使用的模擬退火（Simulated Annealing, SA）[36] 和 Rocha et al. 所使用的多點爬山（Multi-Point Hill Climbing）[37]。

此外，Carpin et al. [38] 提出了適應性隨機漫步（Adaptive Random Walk, ARW）演算法進行求解。但 ARW 算法本質上是基於迭代窮舉搜索，計算量較大。為了克服這個缺點，Birk 和 Carpin [10] 使用啟發式函數來修正最佳化過程，有效地加快了搜索過程，此外 Birk 和 Carpin [10] 還提出了一種錯誤檢測機制，以有效避免錯誤的地圖合併。然而，如果兩個局部地圖之間的重疊程度很低，該演算法的合併結果會受到影響。

為了克服 ARW 演算法的計算量大的缺點，Li et al. [39] 提出使用基因演算法 [40] 進行求解。這種算法模仿自然進化過程，具有隨機全域搜索和並行性的特點。相對於 ARW 演算法，其更適合在大量可能性中尋找目標解，能夠更有效地實現兩幅圖之間的最優匹配。同時作者也提出相關的應用情境，在室外環境中，當車輛感測器被遮蔽造成無法得知環境資訊的情況。通過地圖合併可以獲得車輛之間的相對位置關係，從而從其他車輛中獲取被遮蔽區域的環境資訊，以有效提高車輛的感知能力和安全性能。

2. **特徵擷取方法：**此方法在合併機器人生成的局部地圖時，首先提取局部地圖的興趣點（即特徵），然後使用這些提取的特徵來合併佔據格點地圖 [41]。提取的特徵通常被認為是靜態的，不隨時間變化，確保提取的特徵可以在不同的局部地圖中重複出現 [42]。

格點像素的特徵十分多樣，常見的特徵類型包括點特徵（例如，尺度不變特

徵變換 (Scale Invariant Feature Transformation, SIFT) 特徵 [4, 13, 43]、加速穩健特徵 (Scale Invariant Feature Transformation, SURF) [44] 和 Harris [45])、線特徵 [46] 和幾何特徵 [47]，也可能是線段角度轉換的頻譜特徵 [48]。根據使用的特徵不同，其演算方式雖然具有差異，但基本邏輯架構相似，皆是透過提取的特徵進行匹配，找出局部地圖的相對關係，並驗證合併結果。

以 Wang et al. [43] 為例，其將格點佔據地圖視為圖像，使用尺度不變特徵轉換 (SIFT) [49] 演算法對格點佔據地圖進行特徵擷取。將不同地圖各自擷取特徵後，依照 SIFT 輸出的描述符，並使用 ICP 掃描匹配演算法 [50] 進行特徵匹配，藉由建立兩地圖間的特徵配對集合直接計算出 MTM 並合併地圖。此方法常見於影像處理的全景圖合併、圖像縫合問題 [51]，不同的是，一般影像圖片的紋理、明度、色彩特徵較多，可擷取出較多特徵；而格點佔據地圖的像素往往趨於三值（白、灰、黑），因此可擷取的特徵較少。

驗證合併結果的方式以 Carpin et al. [48] 為例，其提出接受指標 (Acceptance Index, AI)，對於格點佔據地圖的匹配，接受指標相當適合用於判斷地圖合併成功與否，概念與先前提及的區域種類的正確、錯誤配對類似，實際定義如下：

$$\omega(\mathbf{M}_1, \mathbf{M}_2) = \begin{cases} 0 & , \text{if } agr(\mathbf{M}_1, \mathbf{M}_2) = 0. \\ \frac{agr(\mathbf{M}_1, \mathbf{M}_2)}{agr(\mathbf{M}_1, \mathbf{M}_2) + dis(\mathbf{M}_1, \mathbf{M}_2)} & , \text{if } agr(\mathbf{M}_1, \mathbf{M}_2) \neq 0. \end{cases} \quad (2.4)$$

其中  $\omega(\mathbf{M}_1, \mathbf{M}_2)$  為接受指標， $agr(\mathbf{M}_1, \mathbf{M}_2)$  代表兩地圖  $\mathbf{M}_1$ 、 $\mathbf{M}_2$  的正確格點種類配對的個數， $dis(\mathbf{M}_1, \mathbf{M}_2)$  則代表錯誤配對的個數。

上述基於特徵的方法都沒有考慮到極端情況的特徵匹配錯誤。當不同地圖中兩個相似特徵被誤認為相同特徵時，就會發生這種特徵匹配錯誤 [45]。特徵匹配錯誤會導致嚴重的地圖合併錯誤，這種情況通常發生在具有重複特徵的環境中，例如對稱結構環境。為了解決這個問題，Durdu 和 Korkmaz [45] 提出了一種語義方法來合併具有對稱結構的環境，此方法通過評估特徵的有效性來刪減特徵，有效地避免對稱環境中的特徵匹配錯誤。但此方法的前提是重複特徵的環境仍有細微的差異，對於無法有細微差異的重複特徵環境，例如只有機台的工廠，仍無法有效的成功合併。



### 2.3.3 格點佔據地圖合併方法之小結

直接類方法的優勢在於簡單易行，計算複雜度相對較低且不受到環境特徵的影響，尤其適用於重複特徵環境。但是，由於觀察彼此位置要求機器人同時位於同一場域，限制了時間分配，同時因為必須要觀測到彼此，限制了活動範圍及靈活性，進而降低了探索效率。相比之下，辨識共同環境資訊則可放寬活動範圍的限制，只要個體探索範圍曾重疊過，就可以計算相對關係，這種概念與間接類方法相似，但是直接類方法需要儲存大量資料。以 [34] 的方法為例，需要儲存視覺對象的 RGB 影像，隨著時間的累積，所需儲存的圖片數量也越來越多，因此佔用的儲存空間很大。且不論那一個方法，準確度在很大程度上依賴於感測器的測量精度，受到硬體規格的限制。

間接類方法則利用地圖影像儲存的資料進行地圖合併，不受時間和探索範圍的限制，具有靈活性。同時，其通訊傳遞的格點佔據地圖影像的像素只趨於三值（白、灰、黑），檔案大小與 RGB 影像相比較小，可避免資料遺失。然而，在間接類方法中，最佳化搜尋方法需要良好的初始估計值，並且在大規模地圖合併時需要消耗大量計算資源，因此執行速度較慢，且無法保證全域最佳解的收斂，此外從目標函數的定義可以得知，在重複特徵環境中，存在多組最佳解，可能導致地圖合併錯誤；特徵擷取方法雖然能夠快速進行大規模地圖合併，因為其只需要計算影像中的關鍵點，而不需要處理整個影像，但特徵擷取方法可取得的特徵數量較一般 RGB 影像少，且在重複特徵環境中容易出現特徵匹配錯誤。

## 2.4 文獻回顧總結

回顧與分析多機器人自主探索及地圖合併的研究，可以發現現有的研究多著重在單一方面，缺乏將兩者結合在一起的探討，因為現有多機器人自主探索皆是假設機器人已知彼此的初始相對姿態。然而在實際應用中，無法預期各個機器人已知彼此的初始相對姿態，因此將探索和合併兩者結合在一起是必要的。同時如上述所提及，在面對重複特徵的環境時，格點佔據地圖使用間接類難以成功合併，而使用直接類則會需要儲存及處理大量資料，並限制了活動範圍及靈活性，進而降低了探索效率，這是解決地圖合併問題的一大挑戰。因此，本研究旨在探討如何在未知初始相對姿態的情況下，將多機器人協作探索策略和地圖合併結合



在一起，並解決重複特徵地圖的合併問題。在第三章中，首先會具體地針對本研究核心問題進行描述，給出完整數學定義，並可以看到本研究基於 [1,5] 的方法發展未知初始相對姿態多機器人自主探索。並在第四章詳細介紹本研究地圖合併以外的演算法，此部分是基於 [1,5] 內的演算法進行整合。而第五章將詳細介紹地圖合併的演算法，其將針對重複特徵地圖提出一個新方法。最後以模擬環境測試本研究方法，並進行討論與總結。



# 研究流程與架構

前兩章節已經闡述了本研究方法分為兩大部分：重複特徵的地圖合併、未知初始相對姿態多機器人自主探索。在實現感測、地圖建構、運動規劃的部分，本研究使用 MATLAB 中開源的專案來協助完成。而探索策略及地圖合併等其他部分，則是完全自行撰寫和實作，同樣使用 MATLAB 語言。本章節將針對這兩大部分進行更詳細的問題定義，並說明本研究使用的研究假設、問題定義及研究架構。

## 3.1 研究假設

- **環境假設:** 本研究所考慮的環境為平坦地面，以確保執行任務時不會有高低差所造成的量測誤差，同時環境是靜態的，除了機器人之外不會有任何其他移動物件。此外，環境是全域封閉的，代表機器人無法逃離可用的總空間，且環境是局部開放的，機器人不能被困在特定位置，例如房間。最後，環境中的障礙物均為水泥牆，這是因為本研究參考之通信強度模型 [52] 是在此環境進行擬合。
- **機器人假設:** 在本研究中，假設機器人里程計（Odometry）沒有誤差，因此不需要自我定位。此外，機器人配備的光達也沒有誤差，並且機器人個體之間使用 5G WiFi 持續進行通訊，以便快速交換資料，這是因為參考之通信強度模型 [52] 是以 5G Wi-Fi 來進行擬合。最後假設機器人建構之地圖解析度皆相同，以方便後續的資料處理和演算法實現。



## 3.2 問題定義

本研究有兩個目標，第一個是成功將格點佔據地圖合併，第二個是將未知環境地圖建構完成。為了完成上述目標，必須先定義格點佔據地圖。格點佔據地圖可視作灰階影像圖片的一種，因此數學上可將其定義為二維矩陣  $\mathbf{M}_i$ ，如式 (3.1)：

$$\mathbf{M}_i = \begin{bmatrix} m_{11} & \cdots & m_{1c_i} \\ \vdots & \ddots & \vdots \\ m_{r_i 1} & \cdots & m_{r_i c_i} \end{bmatrix} \quad (3.1)$$

其中，矩陣大小為  $r_i \times c_i$ ，矩陣  $\mathbf{M}_i$  的元素代表灰影像像素值，數值範圍如下：

$$\mathbf{M}_i(x, y) = m_{xy} \in [0, 255], \quad x = 1, \dots, r_i, \quad y = 1, \dots, c_i \quad (3.2)$$

影像像素值代表格點佔據地圖的區域分類資訊，因此主要有三個整數值，即 (1) 佔據區域像素值為 0、(2) 未知區域像素值為 128、(3) 自由區域像素值為 255。

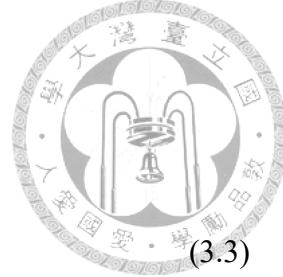
上述已經定義了格點佔據地圖，接下來將分別介紹如何定義成功地將格點佔據地圖合併，以及如何將未知環境地圖建構完成。以下是這兩個目標的問題描述：

### 1. 格點佔據地圖合併

地圖之間的 MTM 是能夠將一個地圖轉換成另一個地圖的數學變換關係。在機器人建構地圖時，由於機器人的參考原點、地圖解析度和比例尺可能不同，因此需要將地圖進行旋轉、縮放和平移等變換，以使不同地圖的座標系統和比例尺一致，而本研究假設地圖解析度相同，因此不需考慮縮放的變換。假設有兩個格點佔據地圖，分別為  $\mathbf{M}_1$  和  $\mathbf{M}_2$ ，且存在一個 MTM 為  $T^{12}$ ，它可以以地圖  $\mathbf{M}_2$  為參考，將地圖  $\mathbf{M}_1$  進行變換，使其座標系統與地圖  $\mathbf{M}_2$  對齊並統一，如式 (3.3)。其中， $\theta$  表示地圖  $\mathbf{M}_1$  與  $\mathbf{M}_2$  的旋轉角度差異，

$t_x$  和  $t_y$  分別表示兩者平移的水平和垂直距離差異。

$$\mathbf{T}^{12} = \begin{bmatrix} \cos \theta & -\sin \theta & t_x \\ \sin \theta & \cos \theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \quad (3.3)$$



這種相對變換關係是雙向的，也就是說，如果地圖  $\mathbf{M}_1$  可以轉換成地圖  $\mathbf{M}_2$ ，則地圖  $\mathbf{M}_2$  也可以轉換成地圖  $\mathbf{M}_1$ 。經過 MTM 變換後，地圖  $\mathbf{M}_1$  和  $\mathbf{M}_2$  已經統一比例與座標系關係，因此兩地圖會有部份重疊的格點。對於每一重疊的格點，像素值來自於地圖  $\mathbf{M}_1$  和  $\mathbf{M}_2$ ，因此可以根據這兩個像素值來區分重疊格點的種類。區分方式如同第二章提及的正確匹配、錯誤匹配：

- 兩像素值所代表的區域類型相同，且並非未知區域，則屬於正確匹配 (*agr*)。
- 兩像素值所代表的區域類型相異，且並非未知區域，則屬於錯誤匹配 (*dis*)。
- 任一像素值所代表的區域中為未知區域，則不予計算。

根據上述的正確、錯誤匹配計算，可以使用 Birk [10] 所提出的接受指標（式 (2.4)) 來評估地圖  $\mathbf{M}_1$  和  $\mathbf{M}_2$  在經過相對變換後的重疊結果。理論上，如果地圖  $\mathbf{M}_1$  經過正確的變換後，其格點應該與  $\mathbf{M}_2$  相同的區域類型重疊，也就是說正確匹配的數量應該增加，錯誤匹配的數量應該減少，這樣接受指標  $\omega$  就會越接近於 1。根據這個概念，可以設定一個閾值來評估是否要接受相對變換，若接受則把兩個地圖的重疊情況作為地圖合併的結果：

$$\omega > \omega_{thres} = 0.95 \quad (3.4)$$

在此設定閾值為 0.95 的原因是多項研究 [4, 10] 在成功合併地圖的實驗結果中，接受指標大多落在 0.95 至 0.98 之間，而失敗的結果則遠低於 0.90。因此，本研究採用了一個較低的標準作為閾值。

2. 未知環境地圖建構完成邊界是指自由區域和未知區域之間的交界區域。在地圖探索中，邊界被用來表示機器人需要繼續探索的區域，因為這些區域可能包含未知的信息。數學上，邊界可以被定義為：



$$\mathbf{M}_i(x, y) = 255 \text{ and } (\exists(a, b) \text{ s.t. } \mathbf{M}_i(a, b) = 128) \quad (3.5)$$

式 (3.5) 中， $(a, b)$  是  $(x, y)$  的鄰居格點，即  $(x-1, y)$ 、 $(x+1, y)$ 、 $(x, y-1)$ 、 $(x, y+1)$  四個方向上的格點。如果多機器人的所有局部地圖都找不到邊界（即所有邊界都已被探索），可以定義地圖已被探索完成。假設有  $k$  台機器人，每個機器人都可以感知到本身局部地圖上的所有格點。對於每個機器人  $i$ ，定義  $\mathcal{F}_i$  為機器人  $i$  的邊界集合，即該機器人局部地圖未被探索的邊界集合。所有機器人的邊界集合的聯集，也就是所有局部地圖未被探索的邊界集合，表示如式 (3.6)：

$$\mathcal{F} = \bigcup_{i=1}^k \mathcal{F}_i \quad (3.6)$$

如果所有局部地圖的邊界都已被探索，則代表地圖已被探索完成，數學表示如下：

$$\mathcal{F} = \emptyset \quad (3.7)$$

式 (3.7) 中， $\emptyset$  表示空集。

### 3.3 研究方法與架構

前一節已針對格點佔據地圖進行定義，同時也給定評判合併地圖結果的量化指標及探索完成的定義。接下來將介紹本文研究架構，並在此將簡介研究架構的基礎背景。

#### 3.3.1 格點佔據地圖合併架構

如第二章所述，當格點佔據地圖面臨重複特徵環境時，使用間接特徵擷取方法難以成功合併。格點佔據地圖作為影像進行處理時，其像素值主要為 3 個整數 (0、128、255)，相較於一般影像包含的 256 個整數像素值 (0~255)，像素值的變化較小，因此特徵擷取演算法可取得的特徵數量相對較少。同時，在重複特徵



環境中，提取的特徵無法分辨區別，可能導致特徵配對錯誤。然而，使用直接特徵擷取方法的辨識共同環境資訊方法，雖然可以不受重複特徵環境的影響，但需要儲存大量影像圖片，佔用大量儲存空間並消耗大量運算資源。

為了克服直接和間接地圖合併方法的缺點，本研究提出新的地圖合併方法，其利用基於通信信號強度模型 [52] 推導出的距離預測模型，來估計兩個機器人之間的相對距離，並通過運動學分析計算 MTM 來合併地圖。然而，由於信號雜訊，MTM 可能包含不準確的信息，本研究應用接受指標（式（3.4））檢測 MTM 的接受度，如果不可接受，則使用 Ferrão et.al 所提出的特徵地圖合併方法 [4] 修改版本來修正 MTM，修改內容包括將特徵唯一匹配改為窮舉匹配以及將 MTM 的計算限制旋轉角度及平移的限制，架構如圖 3.1 所示。接著再次使用接受指標判斷是否接受 MTM，若仍不接受，則回到距離預測模型進行距離修正。一旦過程中 MTM 是可接受的，就會透過濾波技術輸出合併地圖，若修正多次距離仍無法接受 MTM，則輸出原地圖。提出的地圖合併方法結構如圖 3.2 所示。

此外，信號強度資料相較於需要儲存大量圖片的直接類辨識共同環境資訊方法，儲存空間小，不需要消耗大量運算資源。因此透過本研究提出的方法，可以有效解決重複特徵環境的地圖合併問題。

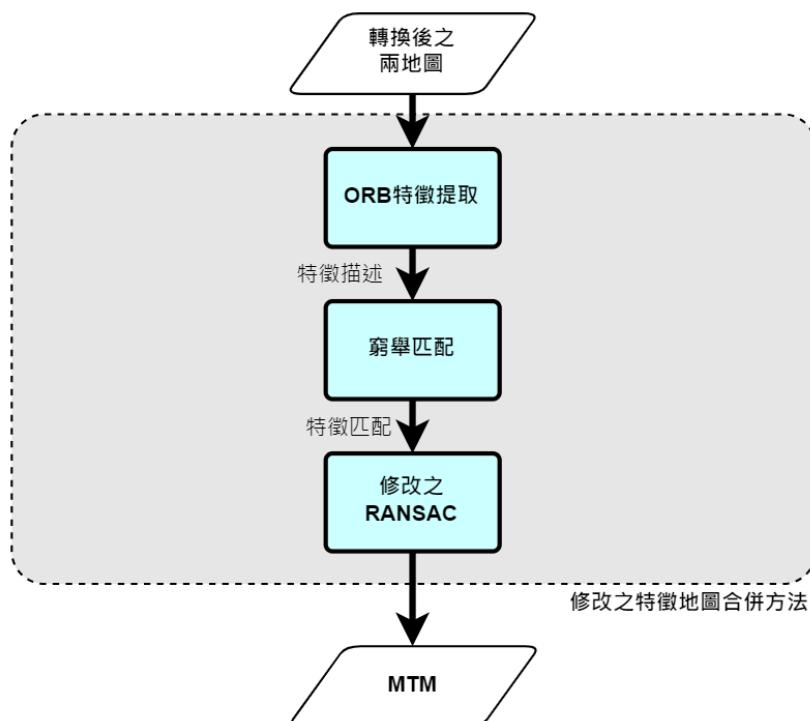


圖 3.1: Ferrão et.al 所提出的特徵地圖合併方法 [4] 修改版本架構。

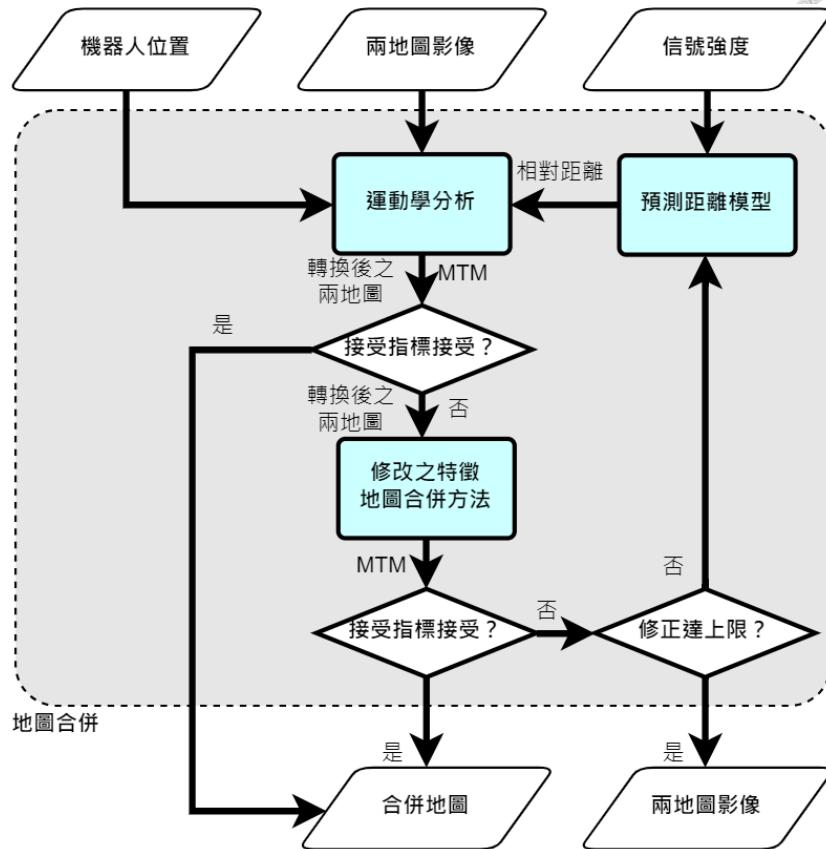


圖 3.2: 本文提出之地圖合併方法架構。

### 3.3.2 多機器人自主探索架構

本文多機器人自主探索之研究架構主要參考 Umari et al. 所提出的多機器人自主探索架構 [1] 及其所提出的單機器人自主探索架構 [5]，利用前一章所提出的地圖合併方法將兩者結合，以達成初始相對姿態未知的多機器人自主探索。

其中 [1] 的多器人自主探索架構如圖 3.3，其主要由路徑規劃、機器人探索策略、地圖整合、SLAM 所組成，首先每個機器人會使用 SLAM 及其光達掃描、里程計的資料來更新建構局部地圖及定位姿態，接著使用地圖整合將局部地圖整合為全域地圖，接著將全域地圖以及機器人姿態傳給多機器人探索策略，利用地圖來進行全域 RRT 的生長，其趨向於尋找遠離機器人的邊界，以及利用地圖以及機器人姿態來進行局部 RRT 的生長，其趨向於尋找機器人附近的邊界，接著將找到的全部邊界進行過濾，將過於接近的邊界進行聚類縮減資料，最後對剩下的邊界進行計算收益，最終透過市場機制，將多個邊界按照收益分配給多個機器人作為目標點，並利用地圖、機器人姿態及目標點位置進行全域路徑規劃，及利用光達



掃描的資料、地圖、機器人姿態來進行局部路徑規劃避障，直到地圖的邊界都已被探索，且找不到新的邊界（式 (3.7)）。

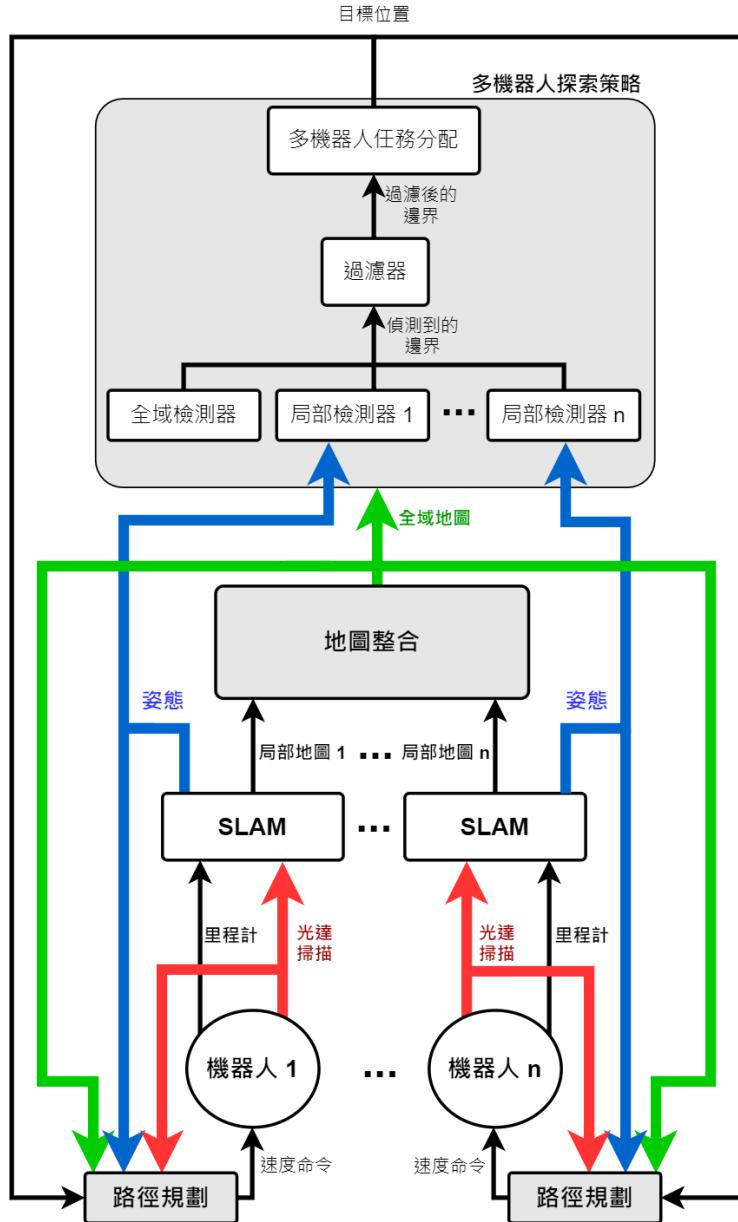


圖 3.3: 本文多機器人自主探索之參考方法 [1]。

而 [5] 的單機器人探索架構如圖，如圖 3.4，其架構與多器人自主探索架構相似，只差在不需使用地圖整合將局部地圖整合，且單機器人探索策略最終不需市場機制進行任務分配，只需將收益最高的邊界視為目標點，其停止條件一樣為地圖的邊界都已被探索，且找不到新的邊界（式 (3.7)）。

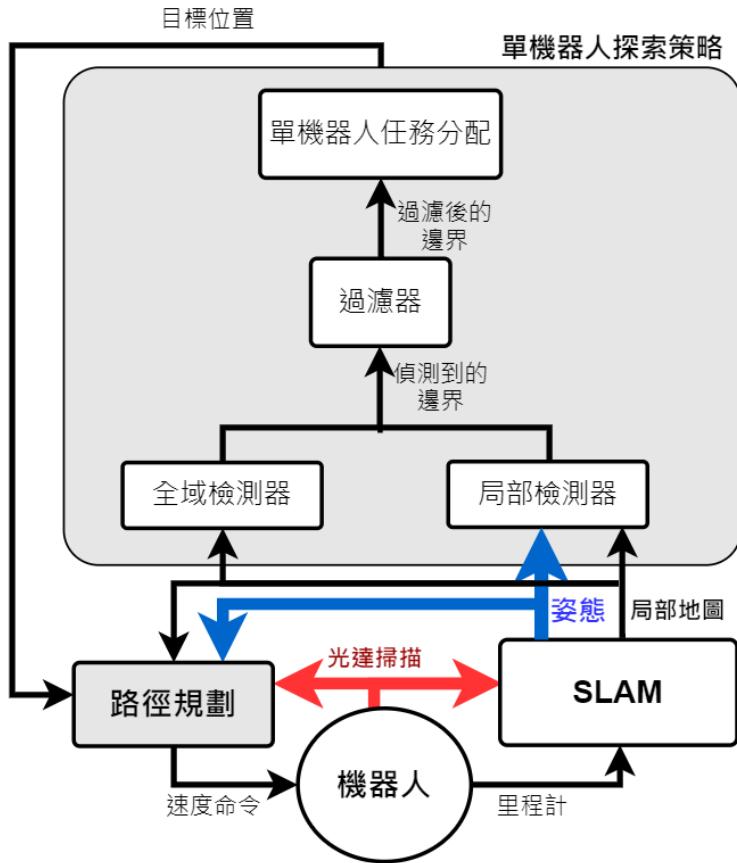


圖 3.4: 本文單機器人自主探索之參考方法 [5]。

綜合上述，本研究多機器人自主探索架構如圖 3.5，其主要添加地圖合併以整合兩個自主探索架構 [1, 5]，地圖合併方法如上一節所提，地圖成功合併將採用多機器人自主探索中的多機器人探索策略 [1]，地圖合併失敗將採用單機器人自主探索中的單機器人探索策略 [5]，同時，將架構進行些微簡化，假設里程計沒有誤差，也就是可以直接知道機器人個體在其局部地圖之姿態，不需要透過 SLAM 來定位姿態，因此 SLAM 簡化為建構地圖。並且機器人之間會傳遞通信信號強度，以處理本研究之地圖合併。

對於本研究之多機器人自主探索除了地圖合併之外的各個演算法，將於第四章進行詳細介紹。而本研究提出之地圖合併方法將於第五章進行詳細的說明。

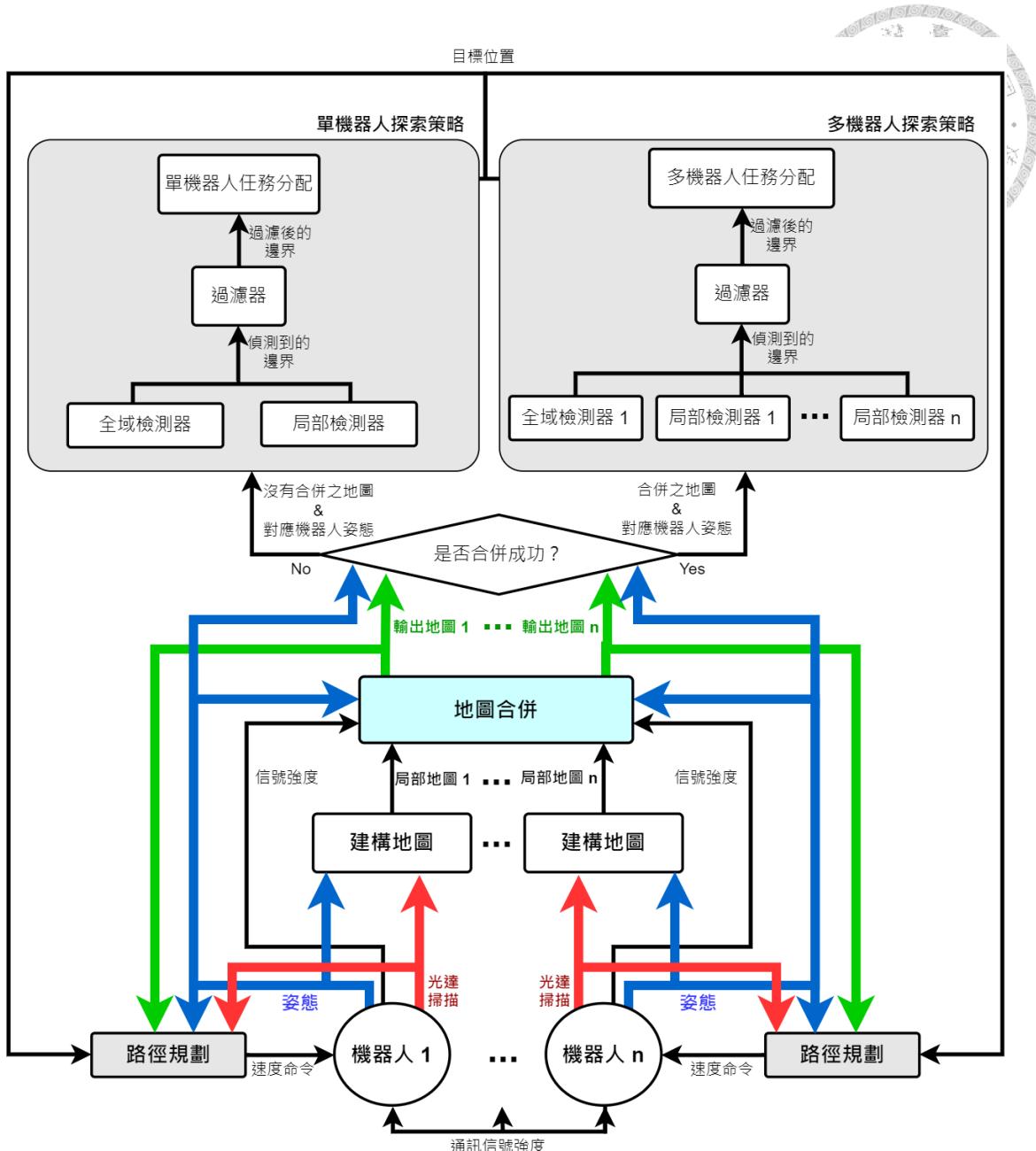


圖 3.5: 本文多機器人自主探索之研究方法，主要將單機器人及多機器人的自主探索架構進行整合，並加入了地圖合併。



## 未知初始相對姿態多機器人自主探索

第三章已對本研究之多機器人探索架構進行簡述，即使用 Umari et al. 所提出的多機器人自主探索架構 [1] 及其所提出的單機器人自主探索架構 [5]，利用所提出的地圖合併方法將兩者結合，以達到未知初始相對姿態之多機器人自主探索，因為未知初始相對姿態，所以無法直接將局部地圖整合為全域地圖，對於地圖尚未合併之機器人，採用 [5] 所提出的單機器人探索架構中的單機器人探索策略，而地圖合併成功之機器人，則採用 [1] 所提出的多機器人自主探索架構中的多機器人探索策略，而本研究提出之地圖合併技術將於第五章詳細介紹。本章將說明本研究多機器人探索（圖 3.5）除了地圖合併以外的演算法，即

- 用於檢測機器人附近邊界的局部邊界檢測器（Local Frontier Detector）
- 用於檢測遠離機器人邊界的全域邊界檢測器（Global frontier detector）
- 用於過濾縮減邊界數量的過濾器（Filter）
- 用於指派單機器人目標點的單機器人任務分配（Single-Robot Task Allocator）
- 用於指派多機器人目標點的多機器人任務分配（Multi-Robot Task Allocator）
- 用於規劃機器人路徑的全域路徑規劃（Local Path Planning）
- 用於規劃機器人避障的局部路徑規劃（Global Path Planning）



## 4.1 用語與符號定義

- $X$ ：包含佔據、自由和未知的空間（地圖）。
- $X_{free}$ ：未被障礙物佔據的已知空間。
- $\mathcal{V}$ ：節點為地圖上的座標點，這些節點通過邊相互連接，存儲在集合  $\mathcal{V}$  中。
- $\mathcal{E}$ ：邊是連接兩個節點的線，每條邊根據兩個節點的空間坐標存儲，邊存儲在集合  $\mathcal{E}$  中。
- $\mathcal{G}$ ：由所有的邊和節點構成的節點樹，也就是  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ 。
- $SampleFree$ ：一個函數，其輸出在未被障礙物佔據的空間中獨立且有共同分佈 (i.i.d) 的點。
- $Nearest(\mathcal{G} = (\mathcal{V}, \mathcal{E}), x \in X_{free})$ ：一個函數，其輸入節點樹  $\mathcal{G}$  和  $X_{free}$  中的點，並輸出距離輸入點最近的節點  $v \in \mathcal{V}$ ，也就是  $Nearest(\mathcal{G} = (\mathcal{V}, \mathcal{E}), x) = argmin_{v \in \mathcal{V}} \|x - v\|$ 。
- $Steer$ ：一個函數，其輸入兩個點  $x_1, x_2 \in X$ ，並輸出點  $x_{new}$ ，使得  $argmin_{x_{new}} \|x_{new} - x_2\|$ ，且  $\|x_{new} - x_1\| \leq \eta$ ，其中  $\eta$  是節點樹的邊的最大長度， $\eta > 0$ ， $atan2(x_{new}, x_1) = atan2(x_2, x_1)$ 。
- $GridCheck$ ：一個函數，其輸入地圖和兩個點  $x_1$  和  $x_2$ 。如果根據輸入的地圖，在輸入的兩點之間存在障礙物，將輸出 0。如果在輸入的兩點之間不存在障礙物，且  $x_2$  屬於未知區域或是邊界（邊界之定義如式 (3.5)），函數將輸出 -1。否則將輸出 1，表示點  $x_2$  不是邊界且在  $X_{free}$  中。
- $FrontierSteer$ ：一個函數，其輸入兩個點  $x_1, x_2 \in X$ ，並輸出點  $x_{new}$ ，使得  $argmin_{x_{new}} \|x_{new} - x_2\|$ ，且  $x_{new}$  屬於邊界， $atan2(x_{new}, x_1) = atan2(x_2, x_1)$ 。
- $PublishPoint$ ：一個函數，將檢測到的邊界發送到過濾器。



## 4.2 基於 RRT 之邊界檢測器

RRT 偏向於往未探索空間擴展 [24]，使得節點樹傾向於偵測邊界，這種特性可以通過沃羅諾伊圖來解釋。一個由節點和邊組成的圖形（RRT 節點樹），沃羅諾伊圖將圖形劃分為區域，每個區域只有一個節點。具有較大沃羅諾伊區域的節點靠近未知空間，由於選擇節點是基於尋找最近的鄰居節點，因此節點樹更有可能從這些區域擴展 [5]。同時，RRT 可以保證地圖將被完全探索 [6]。

基於 RRT 之邊界檢測器用於發現邊界，在本研究中，只要 RRT 節點樹擴張的節點屬於地圖未被障礙物佔據已知區域且周圍有任何一個屬於未知區域的格點，該點就被視為邊界。本研究修改 [5] 所提出的兩個邊界檢測器：(1) 局部邊界檢測器，和 (2) 全域邊界檢測器。在單機器人探索的情況，局部邊界檢測器和全域邊界檢測器都由該機器人運行。而在多機器人探索的情況，全域邊界檢測器、局部邊界檢測器由每個機器人運行。

### 4.2.1 局部邊界檢測器

流程如演算法 1 所示。局部邊界檢測器中的節點樹初始節點由機器人目前位置  $x_{init}$  開始，節點樹初始設定為  $\mathcal{V} = \{x_{init}\}$  和  $\mathcal{E} = \emptyset$ ，並且在每次迭代中對  $SampleFree$  進行隨機取樣  $x_{rand} \subset X_{free}$ 。接著找到最接近  $x_{rand}$  的節點，這個節點稱為  $x_{nearest} \subset \mathcal{V}$ 。然後，使用  $Steer$  函數生成一個點  $x_{new}$ ，並用  $GridCheck$  函數檢查  $x_{new}$  是否屬於邊界，如果是邊界就將  $x_{new}$  發送到過濾器，並刪除節點樹，節點樹的下一次迭代從機器人目前位置開始生長（即  $\mathcal{V} = \{x_{current}\}$ , and  $\mathcal{E} = \emptyset$ ）。如果在  $x_{new}$  沒有障礙物且在  $x_{new}$  和  $x_{nearest}$  之間的空間中沒有障礙物，則節點樹通過添加  $x_{new}$  作為一個新的節點，並在  $x_{new}$  和  $x_{nearest}$  之間創建邊。對於每個運行局部邊界檢測器的機器人，通過上述過程生成一棵節點樹。一旦發現邊界，就會標記邊界並重置樹，且這個過程發生在機器人的運動過程中，也就是機器人的運動不會因為節點樹的運算而停止。而局部邊界檢測器效用在於隨時快速檢測機器人附近的邊界，因為節點樹從機器人目前位置開始生長。圖 4.1 顯示了局部 RRT 生長和檢測邊界的過程。




---

**Algorithm 1 局部邊界檢測器**


---

```

1:  $\mathcal{V} \leftarrow x_{init}; \mathcal{E} \leftarrow \phi;$ 
2: while True do
3:    $x_{rand} \leftarrow SampleFree;$ 
4:    $x_{nearest} \leftarrow Nearest(\mathcal{G}(\mathcal{V}, \mathcal{E}), x_{rand});$ 
5:    $x_{new} \leftarrow Steer(x_{nearest}, x_{rand});$ 
6:   if  $GridCheck(x_{nearest}, x_{new}) = -1$  then            $\triangleright$  未知區域或邊界
7:      $x_{new} \leftarrow FrontierSteer(x_{nearest}, x_{new});$        $\triangleright$  邊界
8:      $PublishPoint(x_{new});$ 
9:      $\mathcal{V} \leftarrow x_{current}; \mathcal{E} \leftarrow \phi;$                    $\triangleright$  刪除並重置節點樹
10:    else if  $GridCheck(x_{nearest}, x_{new}) = 1$  then     $\triangleright$  未被障礙物佔據的已知空間
11:       $\mathcal{V} \leftarrow \mathcal{V} \cup \{x_{new}\}; \mathcal{E} \leftarrow \mathcal{E} \cup \{(x_{nearest}, x_{new})\};$ 
12:    end if
13: end while

```

---

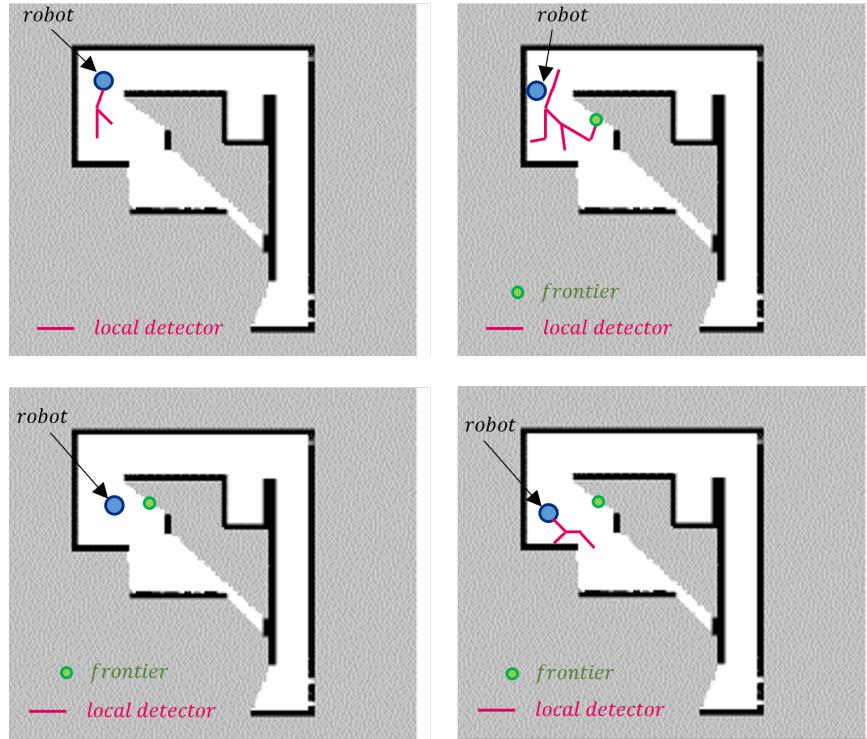


圖 4.1: 局部 RRT 生長和檢測邊界的過程。

#### 4.2.2 全域邊界檢測器

流程如演算法 2 所示。基本流程與局部邊界檢測器相同，唯一的區別在於節點樹不會重置，並在整個探索期間保持生長，直到地圖被完全探索。全域邊界檢測器效用在於通過整個地圖檢測遠離機器人的區域的邊界，因為機器人可能會錯



過地圖中的小角落。然而，隨著節點樹變大，也就是節點數量增加，全域邊界檢測器中節點樹的生長會變慢，這可以以沃羅諾伊圖來解釋：隨著節點數量的增加，空間被分解成越來越小的沃羅諾伊區域，其結果是新節點的邊越來越短，因此邊界的檢測也就會變慢，這也是需要局部邊界檢測器的原因。圖 4.2，顯示了全域邊界檢測器的 RRT 生長和檢測邊界的過程。

---

**Algorithm 2 全域邊界檢測器**


---

```

1:  $\mathcal{V} \leftarrow x_{init}; \mathcal{E} \leftarrow \phi;$ 
2: while True do
3:    $x_{rand} \leftarrow SampleFree;$ 
4:    $x_{nearest} \leftarrow Nearest(\mathcal{G}(\mathcal{V}, \mathcal{E}), x_{rand});$ 
5:    $x_{new} \leftarrow Steer(x_{nearest}, x_{rand});$ 
6:   if  $GridCheck(x_{nearest}, x_{new}) = -1$  then            $\triangleright$  未知區域或邊界
7:      $x_{new} \leftarrow FrontierSteer(x_{nearest}, x_{new});$            $\triangleright$  邊界
8:      $PublishPoint(x_{new});$ 
9:   else if  $GridCheck(x_{nearest}, x_{new}) = 1$  then       $\triangleright$  未被障礙物佔據的已知空間
10:     $\mathcal{V} \leftarrow \mathcal{V} \cup \{x_{new}\}; \mathcal{E} \leftarrow \mathcal{E} \cup \{(x_{nearest}, x_{new})\};$ 
11:   end if
12: end while

```

---

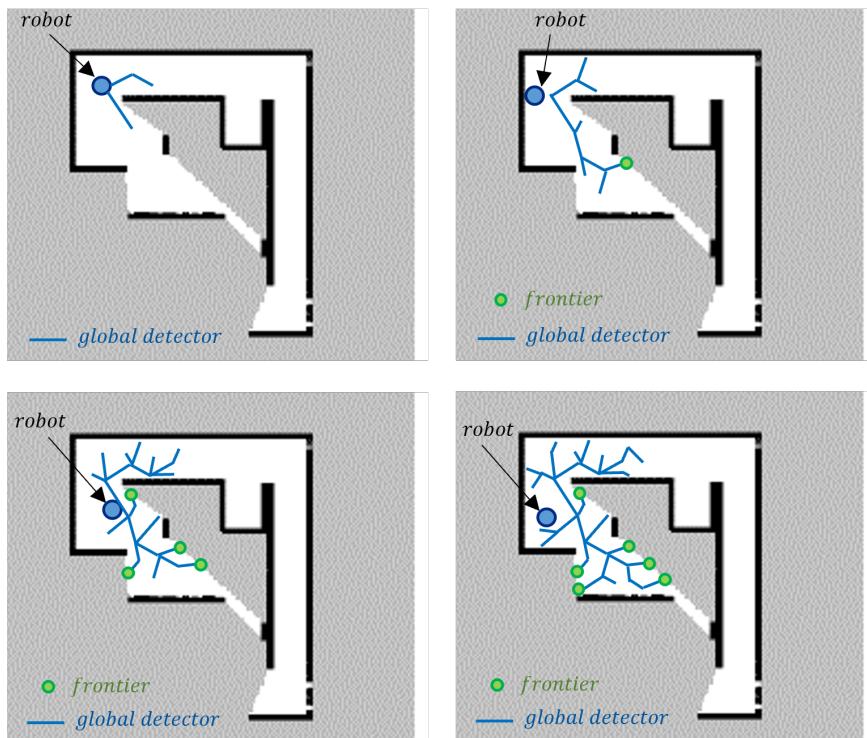


圖 4.2: 全域 RRT 生長和檢測邊界的過程。



## 4.3 過濾器

過濾器從所有局部邊界檢測器和全域邊界檢測器接收檢測到的邊界。每次接收到一個新的邊界，過濾器將其存儲在邊界數據中。首先，過濾器使用均值偏移聚類演算法 [53] 對存儲在邊界數據中的點進行聚類，並刪除一個集群中不是中心的邊界，同時刪除無效和舊的邊界來更新數據，然後將邊界數據發送到機器人任務分配。需要聚類和隨後的丟棄過程來減少邊界的數量，因為全域和局部邊界檢測器可以提供太多彼此非常接近的邊界，如果這些邊界被發送到機器人任務分配，就會有不必要的計算資源消耗，同時無法獲得更多的地圖信息，因為在同一區域的邊界對應於同一個邊界邊緣。

### 4.3.1 均值偏移聚類演算法

該演算法在探索策略的實現中用於減少計算負擔，用於檢測給定地圖上的邊界，如上述所說，檢測到的邊界數量可能非常大且非常靠近彼此，因此，必須將邊界聚類以丟棄冗餘的邊界。否則，計算大量的邊界會使探索變得緩慢，並且無法獲得更多地圖信息，因為在同一區域的邊界對應於同一個邊界邊緣。其優勢在於不需要預先指定聚類數量，只需要指定聚類的大小。

對於一組資料集給定的  $n$  個點  $\{x_i\}_{i=1}^n$  於  $d$  維空間  $\mathbb{R}^d$ ，均值偏移聚類演算法將這些點視為由一個概率密度函數 (probability density function, PDF)  $p_K(x)$  中抽取的樣本， $p_K(x)$  的局部最大值對應於每個群集的中心。均值偏移聚類演算法通過迭代的方式將每個點向  $p_K(x)$  的每個群集的中心方向平移，即可完成分群。

$p_K(x)$  通過使用特定的核函數 (kernel)  $K(x)$  進行機率密度估計獲得，核函數是以零為中心對稱的函數，如式 (4.2)，其中  $c_{k,d}$  是正規化參數，並利用  $k(x)$  使得核函數的積分值為 1，如式 (4.3)，PDF 與核函數詳細關係如式 (4.1)，並可以知道  $p_K(x)$  是獨立核函數的相加，其中參數  $h$  被稱為頻寬。

$$p_K(x) = \frac{1}{nh^d} \sum_{i=1}^n K\left(\frac{x - x_i}{h}\right) \quad (4.1)$$

$$K(x) = c_{k,d} k(\|x\|^2) \quad (4.2)$$



$$\int_{-\infty}^{+\infty} c_{k,d} k(\|x\|^2) dx = 1 \quad (4.3)$$

本研究使用使用平坦核函數的均值偏移，其中平坦核函數的數學形式如下：

$$K(x) = \frac{1}{2} \begin{cases} 1, & \text{if } \|x\| \leq h \\ 0, & \text{if } \|x\| > h \end{cases} \quad (4.4)$$

不同的  $h$  會對機率密度估計有很大的影響。太小的  $h$  會使得  $p_K(x)$  的最大值為資料集的所有點，也就是所有點自成一類；太大的  $h$  則會使  $p_K(x)$  的最大值只剩一個，也就是所有點分成一類。本研究  $h$  設定為 0.1。

每一個點  $x \in \{x_i\}_{i=1}^N$  都會向著距離不超過  $h$  的  $\{x_i\}_{i=1}^N$  中所有點的平均值  $m(x)$  的方向移動 [54]。這些點會重複被移動，直到停下並收斂到一個值。這個值就是該聚類的中心，所有收斂到同一個中心的點屬於同一個聚類。每次迭代中，對於每個數據點  $x \in \{x_i\}_{i=1}^N$ ，都會計算其對應的均值  $m(x)$  (式 (4.5))。將每個點  $x$  移位至其相應的均值 (即  $x \leftarrow m(x)$ )。其中  $m(x) - x$  稱為均值位移。

$$m(x) = \frac{\sum_{i=1}^N K(x_i - x)x}{\sum_{i=1}^N K(x_i - x)} \quad (4.5)$$

---

### Algorithm 3 均值偏移聚類演算法

---

- 1: Input : data set  $X = \{x_i\}_{i=1}^n$ , where  $n$  is number of data,  $n \geq 2$ , bandwidth  $h$ ;
  - 2: Output : clusters of PDF  $\mathcal{Y} = \{y_i^*\}_{i=1}^k$ , where  $k$  is number of clusters;
  - 3: **for**  $i = 1 \dots n$  **do**
  - 4:     Initialization :  $j \leftarrow 1; y_j \leftarrow x_i$  ▷ 初始自成一類
  - 5:     **while**  $y_j$  not converge **do**
  - 6:          $y_j \leftarrow m(y_j)$  ▷ 將點  $y_j$  移位至其相應的均值  $m(y_j)$ ，均值計算如式 (4.5)
  - 7:     **end while**
  - 8:      $y_i^* \leftarrow y_j$
  - 9: **end for**
- 

### 4.3.2 刪除邊界

通過均值偏移聚類演算法對輸入的邊界進行縮減之後（將一個集群中不是中心的邊界刪除），還必須要對剩餘的邊界進行過濾，將無效和舊邊界刪除來更新數組，其中舊邊界和無效邊界定義如下：



- 舊邊界：先前檢測到的邊界，但不再符合邊界定義。
- 無效邊界：機器人無法到達的邊界，即機器人的位置和給定的邊界之間不存在有效的路徑。

## 4.4 單機器人任務分配

此部分通過從過濾器接收到邊界位置數據並將它們分配給單機器人，單機器人任務分配參考自 [5]，在本研究中將其用於地圖尚未合併之機器人。假設接收到  $n$  個邊界位置數據為  $\{x_f^i\}_{i=1}^n \in R^2$ ，通過考慮以下因素來分配單機器人要探索的目標點：

- 導航成本 ( $N$ )：定義為機器人到達邊界所需行進的距離 [5]。本研究為了簡化計算，只通過考慮機器人當前位置  $x_r$  與邊界位置  $x_f^i$  之間的絕對距離來計算導航成本（式 (4.6))，如圖 4.3。

$$N(x_f^i, x_r) = \|x_r - x_f^i\| \quad (4.6)$$

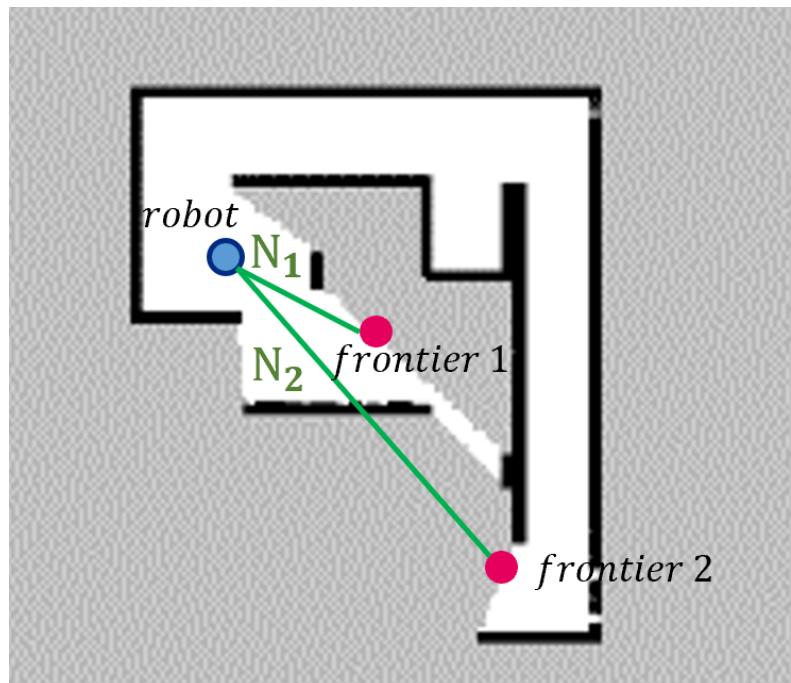


圖 4.3: 邊界之導航成本 ( $N$ )。



- 信息增益 ( $I$ )：定義為給定邊界期望探索的未知區域的面積 [5]。通過輸入目前地圖、邊界位置  $x_f^i$  與傳感器範圍  $r_{sensor}$ ，來計算當機器人到達邊界時，傳感器範圍內未知格點的數量  $N_{cell}(x_f^i, r_{sensor})$ ，如圖 4.4，然後再通過將此格點數量乘以每個格點的面積（根據地圖解析度  $r$  獲得）來計算面積（式 (4.7))。

$$I(x_f^i) = N_{cell}(x_f^i, r_{sensor}) \times r^2 \quad (4.7)$$

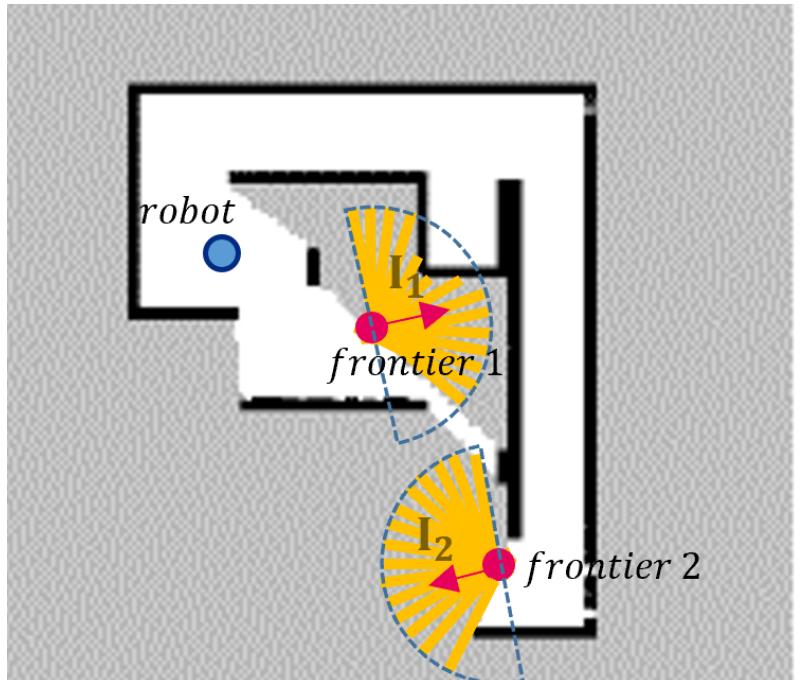


圖 4.4: 邊界之信息增益區域。

- 權重 ( $\lambda$ )：調整權重  $\lambda$  可以使分配分數更加重視信息增益，這也有助於使信息增益和導航成本具有相似的數量級。
- 遲滯增益 ( $h$ )：將機器人當前位置  $x_r$  之設定半徑  $r_h$  內的邊界的信息增益進行增益，也就是乘以增益  $h_{gain}$ ，其中  $h_{gain} > 1$ 。如果機器人已有目標點，目前正在執行探索，則將機器人當前目標點  $x_{target}$  之設定半徑  $r_h$  內的邊界之信息增益進行增益，如式 (4.8)。應用遲滯增益會使機器人附近的邊界的 information gain (I) 增加，這代表機器人會偏向繼續探索當前位置  $x_r$  周圍的區域，也就是除非完全探索當前區域，否則機器人不會切換到新區域。機器人在相距較遠距離之區域之間切換探索的過程是造成重複探索的主要原因，因

為在此過程中機器人會經過已探索的區域，這會造成消耗時間和提高機器人行進的總距離，使效率降低。

$$h(x_f^i, x_r) = \begin{cases} h_{gain}, & \text{if robot available and } \|x_r - x_f^i\| < r_h \\ h_{gain}, & \text{if robot busy and } \|x_{target} - x_f^i\| < r_h \\ 1, & \text{otherwise} \end{cases} \quad (4.8)$$

最終，每個邊界的收益（Revenue）透過輸入邊界位置  $x_f^i$  及機器人的位置  $x_r$ ，並考慮了上述的因素進行計算，如式 (4.9)，收益最高的邊界將分配給機器人進行探索，如演算法 4。

$$R(x_f^i, x_r) = \lambda h(x_f^i, x_r) I(x_f^i) - N(x_f^i, x_r) \quad (4.9)$$

---

**Algorithm 4** 單機器人任務分配

---

```

1: while True do
2:    $n_{frontiers} \leftarrow \text{length of Frontiers};$ 
3:   for  $i = 1, \dots, n_{frontiers}$  do
4:      $infoGains[i] \leftarrow I(x_f^i);$ 
5:     if robot available  $\wedge \|x_r - x_f^i\| < r_h$  then
6:        $h \leftarrow h_{gain};$ 
7:     else if robot busy  $\wedge \|x_{target} - x_f^i\| < r_h$  then
8:        $h \leftarrow h_{gain};$ 
9:     else
10:       $h \leftarrow 1;$ 
11:    end if
12:     $N[i] \leftarrow \|x_r - x_f^i\|;$ 
13:     $R[i] \leftarrow \lambda \times h \times infoGains[i] - N[i];$ 
14:  end for
15:   $index = Max(R);$ 
16:  Assign( $x_f^{index}$ )
17: end while
```

---

## 4.5 多機器人任務分配

此部分通過從過濾器接收邊界數據並將它們分配給多機器人，多機器人任務分配參考自 [5]，在本研究中將其用於地圖合併之機器人。假設接收到  $n$  個邊界位置數據為  $\{x_f^i\}_{i=1}^n \in R^2$ ，通過考慮重複探索來分配多機器人要探索的目標點，重



複探索必須考慮以下幾點，(1) 機器人個體不應探索目前分配給另一個機器人個體的目標點，或接近另一個機器人個體目前位置附近的目標點，(2) 機器人個體應傾向於探索位於分配的目標點周圍或個體目前位置周圍區域的邊界。為了達到這些考量，下列將詳細解釋使用到的方法。

#### 4.5.1 避免重複探索之方法

如同上述所言，為了減少多機器人重複探索區域，必須考慮到兩個面相，(1) 機器人個體不應探索目前分配給另一個機器人個體的目標點，或接近另一個機器人個體目前位置附近的目標點，(2) 機器人個體應傾向於探索位於分配的目標點周圍或個體目前位置周圍區域的邊界。為了解決上述問題，以下將詳細介紹解決方法：

- 導航成本 ( $N$ )：與單機器人任務分配相同，使用機器人目前位置  $x_r$  與邊界位置  $x_f^i$  之間的絕對距離來計算導航成本 (式 (4.6))，如圖 4.3。這解決了第 (1) 點中，機器人不應探索接近另一個機器人個體目前位置附近的目標點的問題。
- 信息增益 ( $I$ )：與單機器人任務分配相同，通過輸入目前地圖、邊界位置  $x_f^i$  與傳感器範圍  $r_{sensor}$ ，來計算當機器人到達邊界時，傳感器範圍內未知格點的數量  $N_{cell}(x_f^i, r_{sensor})$ ，如圖 4.4，然後再通過將此格點數量乘以每個格點的面積 (根據地圖解析度  $r$  獲得) 來計算面積 (式 (4.7))。
- 遲滯增益 ( $h$ )：大致與單機器人任務分配相同，對於沒有目標點的機器人，應用遲滯增益會使機器人附近的邊界的信息增益 ( $I$ ) 增加；不同在於，只有當所有機器人都已有目標點的情況，也就是都正在執行探索，才會應用遲滯增益將機器人目前目標點  $x_{target}$  附近的邊界的信息增益 ( $I$ ) 增加。這代表機器人會偏向繼續探索當前位置或目標點周圍的區域，也就是除非完全探索目前區域，否則機器人不會切換到新區域。這解決了第 (2) 點，達成機

器人傾向於探索位於分配目標點周圍或機器人目前位置周圍區域的邊界。

$$h(x_f^i, x_r) = \begin{cases} h_{gain}, & \text{for each available robot } \|x_r - x_f^i\| \leq r_h \\ h_{gain}, & \text{if no available robots} \\ & \text{and for each robot } \|x_{target} - x_f^i\| < r_h \\ 1, & \text{otherwise} \end{cases} \quad (4.10)$$

- 扣除重疊: 為了避免將相同的邊界分配給多個機器人，使用了 *Discount* 函數。對於與先前分配的目標點具有重疊區域的每個邊界，應用 *Discount* 函數，從其信息增益中減去重疊區域。重疊區域與信息增益 ( $I$ ) 相似，是通過計算重疊格點的數量並將計數乘以每個格點的面積（取決於地圖解析度）來計算的。重疊格點是先前分配的目標點信息增益區域與目前邊界信息增益區域之間共有的格點，如圖 4.5 所示。扣除重疊減少了靠近目標點的邊界之信息增益，應用扣除重疊後，目前分配的目標點將具有零信息增益 ( $I = 0$ )，也因此，其收益會變小，其他機器人就不會分配到這個邊界，這解決了第 (1) 點中，機器人不應探索目前分配給另一個機器人的目標點的問題。

$$Discount(I(x_f^i)) = I(x_f^i) - (N_{cell}(x_f^i, r_{sensor}) \cup N_{cell}(x_{target}, r_{sensor})) \times r^2 \quad (4.11)$$

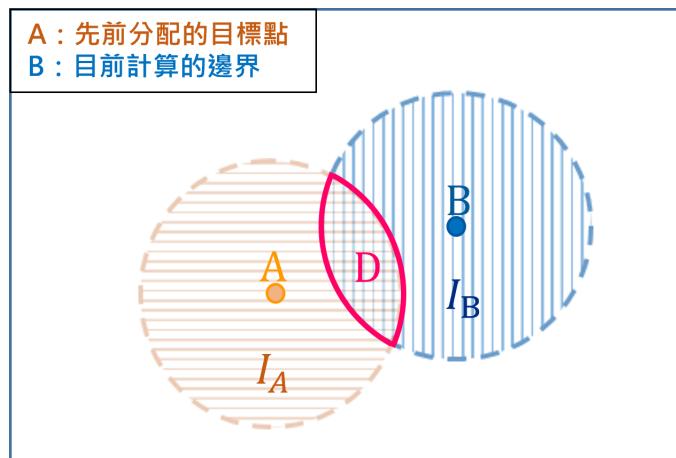


圖 4.5: 機器人目標點 A 點，對邊界 B 的信息增益應扣除重疊區域 D。



- 去除扣除重疊：多機器人任務分配優先分配可用的機器人，只有當所有機器人都忙碌時，任務分配才會將忙碌的機器人重新分配到具有更高預期收益的邊界。扣除重疊雖然解決了將同一個邊界分配給多個機器人的問題，但是，當所有機器人都忙碌時，如果不去除扣除重疊，對於機器人原本分配到的目標點，其扣除重疊後信息增益將變為零，這會造成多機器人任務分配一定會為機器人分配一個不同的目標點，即使這個目標點的收益較低，導致忙碌的機器人在到達目標點前就被重新分配不同的目標點。為了解決這個問題，當所有機器人都忙碌時，在每個接近機器人當前分配的目標點（在  $r_h$  內）的邊界上去除扣除重疊，如演算法 5 第 35 行所示，通過重新計算信息增益來去除扣除重疊。僅當所有機器人都忙碌時才應用去除扣除重疊，使忙碌的機器人偏向於繼續探索目前的目標點，除非存在一個收益更高的邊界。

#### 4.5.2 基於市場經濟的分配

透過基於市場經濟的分配可以為多機器人分配一個合適的目標點。其方法是每次迭代都會計算出投標列表，其中投標的出價等於預期收益。收益的計算與單機器人任務分配基本相同（式 (4.9)），但其考慮了更多的因素，詳細如 4.5.1 節。而對於每個機器人個體，出價最高的邊界被分配為目標點，投標列表由記錄組成，其中每個記錄是由邊界位置、機器人編號以及機器人探索該邊界的預期收益所組成。收益計算和機器人任務分配僅涉及可用的機器人（當前未執行探索的機器人）。但如果沒有可用的機器人（所有機器人都正在探索），則為所有機器人再創建一個投標列表，如果存在具有更高收益的邊界，其將會被分配為目標點，使效率提高。詳細過程如演算法 5 所示。

### 4.6 全域路徑規劃

在全域路徑規劃問題中，有一個機器人開始的初始點，以及機器人想要到達的目標點，這兩個點都位於一個有障礙物的佔據格點地圖中。本節介紹 RRT Star 演算法。RRT Star 用於快速尋找目標點，並找出機器人目前位置和目標點之間的最短路徑。




---

**Algorithm 5** 多機器人任務分配

```

1: while True do
2:    $n_{frontiers} \leftarrow$  length of Frontiers;
3:   for  $i = 1, \dots, n_{frontiers}$  do
4:      $infoGains[i] \leftarrow I(x_f^i)$ ;
5:   end for
6:   for each robot do
7:      $infoGains \leftarrow Discount(map, infoGains, x_f, x_{target})$ ;
8:   end for
9:    $Revenue \leftarrow \phi$ ;
10:   $Robot_{id} \leftarrow \phi$ ;
11:   $Frontier \leftarrow \phi$ ;
12:  for each available robot  $k$  do
13:    for  $i = 1, \dots, n_{frontiers}$  do
14:       $N \leftarrow \|x_r^k - x_f^i\|$ ;
15:      if  $\|x_r^k - x_f^i\| \leq r_h$  then
16:         $infoGains[i] \leftarrow infoGains[i] \times h_{gain}$ ;
17:      end if
18:       $R \leftarrow \lambda \times infoGains[i] - N$ ;
19:       $Revenue \leftarrow Revenue \cup R$ ;
20:       $Robot_{id} \leftarrow Robot_{id} \cup$  robot  $k$  ID;            $\triangleright$  機器人  $k$  的編號
21:       $Frontier \leftarrow Frontier \cup x_f^i$ ;
22:    end for
23:  end for
24:  if no available robots then
25:     $Revenue \leftarrow \phi$ ;
26:     $Robot_{id} \leftarrow \phi$ ;
27:     $Frontier \leftarrow \phi$ ;
28:    for each busy robot  $k$  do
29:      for  $i = 1, \dots, n_{frontiers}$  do
30:         $N \leftarrow \|x_r^k - x_f^i\|$ ;
31:        if  $\|x_r^k - x_f^i\| \leq r_h$  then
32:           $infoGains[i] \leftarrow infoGains[i] \times h_{gain}$ ;
33:        end if
34:        if  $\|x_{target}^k - x_f^i\| \leq r_h$  then
35:           $infoGains[i] \leftarrow I(x_f^i) \times h_{gain}$ ;
36:        end if
37:         $R \leftarrow \lambda \times infoGains[i] - N$ ;
38:         $Revenue \leftarrow Revenue \cup R$ ;
39:         $Robot_{id} \leftarrow Robot_{id} \cup$  robot  $k$  ID;
40:         $Frontier \leftarrow Frontier \cup x_f^i$ ;
41:      end for
42:    end for
43:  end if
44:   $index = Max(Revenue)$ ;
45:  Assign ( robot with ID =  $Robot_{id}[index]$ ,  $x_f^{index}$ )
46: end while

```

---



### 4.6.1 用語與符號定義

首先介紹除了4.1之外會使用到的新術語：

- *CollisionFree*：一個函數，輸入兩個點  $x_1, x_2 \in X$ ，如果  $x_1$  和  $x_2$  之間的線段位於  $X_{free}$  中，輸出 True，即  $[x_1, x_2] \subset X_{free}$ ，否則為 False。
- *Parent*：一個函數，輸入一個節點，輸出父節點。每個節點都有唯一的父節點，每個父節點可以有多個子節點。
- $Cost(v)$ ：一個函數，其將計算節點  $v \in \mathcal{V}$  到根節點  $v_0$  的路徑成本，而  $Cost(x_{inti}) = 0$ ，計算方式如式 (4.12)，其中  $Line(x_1, x_2)$  表示從  $x_1$  到  $x_2$  的線段， $c(Line(x_1, x_2))$  表示從  $x_1$  到  $x_2$  的線段距離。

$$Cost(v) = Cost(Parent(v)) + c(Line(Parent(v), v)) \quad (4.12)$$

- *Near*( $\mathcal{G} = (\mathcal{V}, \mathcal{E}), x \subset X_{free}, r$ )：一個函數，其輸入節點樹  $\mathcal{G}$ 、 $X_{free}$  中的點和一個半徑  $r$ ，並輸出在半徑  $r$  之內的節點  $v \subset \mathcal{V}$ 。其中半徑  $r$  之計算如式 (4.13)，其中  $card(\mathcal{V})$  可以得出  $\mathcal{V}$  的節點數量， $\eta$  是節點樹的邊的最大長度， $\gamma_{RRT^*} > \gamma_{RRT^*}^* = 2(1 + 1/d)^{1/d} (\mu(X_{free}) / \zeta_d)^{1/d}$ ， $d$  為空間  $X$  的維度， $\mu(X_{free})$  表示無障礙空間的體積， $\zeta_d$  是  $d$  維歐氏空間中單位球的體積。也就是說，半徑  $r$  隨著節點數量的增加而減小。

$$r(card(\mathcal{V})) = \min \left\{ \gamma_{RRT^*} (\log(card(\mathcal{V}))/card(\mathcal{V}))^{1/d}, \eta \right\} \quad (4.13)$$

### 4.6.2 RRT Star 演算法

RRT Star 是 Karaman 和 Frazzoli [6] 提出的演算法。其是基於 RRT 演算法的改進版本，RRT 演算法是由 Lavalle [24] 所提出的，其具有適應性強、可以調整搜尋方向和擴展範圍、易於實現和計算速度快等優點。相較其他路徑規劃演算法，計算速度較快，這是由於 RRT 演算法是基於隨機樣本來擴展搜尋空間的，因此可以自動調整搜尋節點的密度，從而更有效地探索搜尋空間，並且通常可以在處理較大的空間時不需要像 A 演算法一樣進行整個空間的搜尋。缺點在於不能保證找到的路徑是最短或最佳的，因為 RRT 演算法是一種隨機搜尋演算法，所以只能保



證找到一條可行的路徑。同時，RRT 生成之路徑非常不平滑，這也是因為隨機搜索新節點所導致。而 RRT Star 改進了上述之缺點，使路徑更加平滑，且可以找出較佳之路徑，兩者的差別如圖 4.6，儘管仍不如其他路徑規劃演算法，但其保有 RRT 之優勢，有著較快的計算速度。

RRT Star 演算法其流程如演算法 6，一開始與 RRT 相同，從隨機採樣開始，到尋找最近節點  $x_{nearest}$  和確定新節點  $x_{new}$ ，但接下來的步驟與 RRT 不同，在 RRT Star 演算法中，在選擇父節點時會有一個“重連”過程，也就是以  $x_{new}$  為圓心，半徑為  $r(card(\mathcal{V}))$  的區域內，找到與  $x_{new}$  連接後，從起點  $x_{init}$  移動到  $x_{new}$  的路程最短的節點  $x_{min}$ ，並將  $x_{min}$  作為  $x_{new}$  的父節點，而不是  $x_{nearest}$ 。接著，對新節點  $x_{new}$  半徑  $r$  內的節點  $x_{near} \in X_{near}$  進行重連，如果到  $x_{new}$  的路徑成本加上  $x_{new}$  到  $x_{near}$  的距離，低於  $x_{near}$  的路徑成本，則將  $x_{new}$  與  $x_{near}$  相連創建新邊，並刪除連接  $x_{near}$  到其父節點的邊，以保持節點樹結構。最終透過輸出之節點樹  $\mathcal{G}$ ，從節點  $x_{target}$  不斷往回尋找父節點，直到到達節點  $x_{init}$  即可得到路徑。

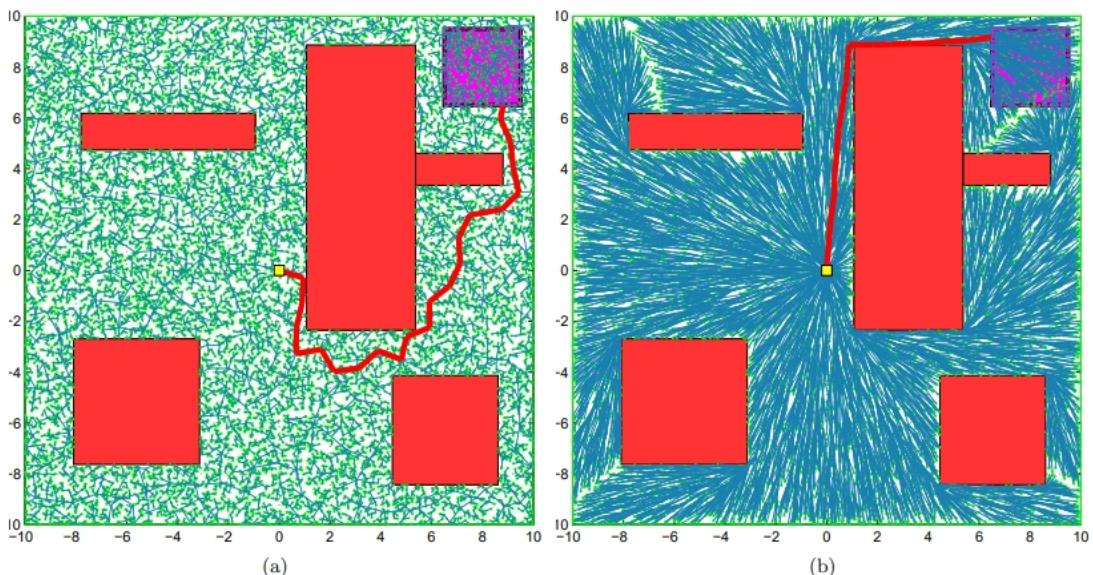


圖 4.6: (a) RRT 和 (b) RRT Star，在具有障礙物的模擬環境中的比較。RRT 和 RRT Star 中最佳路徑的成本分別為 21.02 和 14.51 [6]。

## 4.7 局部路徑規劃

局部路徑規劃，也被稱為局部避障，是指機器人在運動過程中遇到未知障礙物時，利用傳感器掃描的信息重新規劃路徑，繞過障礙物繼續朝目標點前進的過




---

**Algorithm 6** RRT Star 演算法

---

```

1:  $\mathcal{V} \leftarrow x_{init}; \mathcal{E} \leftarrow \phi;$ 
2: while  $x_{new} \neq x_{target}$  do
3:    $x_{rand} \leftarrow SampleFree;$ 
4:    $x_{nearest} \leftarrow Nearest(\mathcal{G}(\mathcal{V}, \mathcal{E}), x_{rand});$ 
5:    $x_{new} \leftarrow Steer(x_{nearest}, x_{rand});$ 
6:   if  $ObstacleFree(x_{nearest}, x_{new}) = -1$  then
7:      $X_{near} \leftarrow Near(\mathcal{G}(\mathcal{V}, \mathcal{E}), x_{new}, r(card(\mathcal{V})))$ ;
8:      $\mathcal{V} \leftarrow \mathcal{V} \cup x_{new};$ 
9:      $x_{min} \leftarrow x_{nearest};$ 
10:     $c_{min} \leftarrow Cost(x_{nearest}) + c(Line(x_{nearest}, x_{new}))$ ;
11:    for each  $x_{near} \in X_{near}$  do            $\triangleright$  找到 Cost 最小的節點作為父節點
12:      if  $CollisionFree(x_{near}, x_{new}) \wedge \dots$ 
13:         $Cost(x_{near}) + c(Line(x_{near}, x_{new})) < c_{min}$  then
14:           $x_{min} \leftarrow x_{near}; c_{min} \leftarrow Cost(x_{near}) + c(Line(x_{near}, x_{new}))$ ;
15:        end if
16:    end for
17:     $\mathcal{E} \leftarrow \mathcal{E} \cup \{(x_{min}, x_{new})\};$ 
18:    for each  $x_{near} \in X_{near}$  do            $\triangleright$  節點樹重連
19:      if  $CollisionFree(x_{new}, x_{near}) \wedge \dots$ 
20:         $Cost(x_{new}) + c(Line(x_{new}, x_{near})) < Cost(x_{near})$  then
21:           $x_{parent} \leftarrow Parent(x_{near});$ 
22:           $\mathcal{E} \leftarrow (\mathcal{E} \setminus \{(x_{parent}, x_{near})\}) \cup \{(x_{new}, x_{near})\};$ 
23:        end if
24:      end for
25:    end if
26:  end while
27: return  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ 

```

---

程。本研究為靜態地圖，在靜態地圖中，局部路徑規劃不是必需的，但其可以優化機器人的運動路徑，提高機器人的運動效率和安全性，避免碰撞和障礙物，因此，本研究有加入局部路徑規劃。本研究選擇使用 VFH+ 演算法 [55] 作為局部避障的演算法，VFH+ 是一種用於快速移動機器人的避障演算法，該演算法基於光達數據並使用直方圖來描述機器人周圍的環境，VFH+ 演算法具有高速、可靠、適用於複雜環境、對傳感器雜訊穩健性高等優點，並且可以應用於不同類型的移動機器人，但缺點是對地形變化和不同高度障礙物的處理能力較弱，並且在存在大量雜訊時，演算法的性能會下降。選擇 VFH+ 主要原因是其在 MATLAB 已有現成套件，以下將詳細說明 VFH+ 演算法的詳細流程。

VFH+ 主要有 4 個步驟：

1. 創建以機器人為中心的活動區域格點佔據地圖  $C_a$ ，並根據光達數據賦予  $C_a$  的格點機率值，其是一個直徑為  $w$  的圓形區域。活動區域中每個格點都被創建障礙向量，向量方向為  $\beta_{i,j}$ ，定義為機器人中心點  $(x_0, y_0)$  到格點  $(x_i, y_i)$  的方向：

$$\beta_{i,j} = \tan^{-1} \frac{y_j - y_0}{x_i - x_0} \quad (4.14)$$

而障礙向量大小則為：

$$m_{i,j} = (c_{i,j})^2 (a - bd_{i,j}^2) \quad (4.15)$$

其中， $c_{i,j}$  為  $C_a(i, j)$  的值， $d_{i,j}$  為格點  $C_a(i, j)$  到機器人中心點  $(x_0, y_0)$  的距離。 $a, b$  應當滿足  $a - b \left( \frac{w-1}{2} \right)^2 = 1$ ，這是為了使  $m_{i,j}$  在活動區域邊界處等於零，因為這些點遠離機器人的工作空間，所以這些位置的障礙物對機器人的影響應該為零。

在此假設機器人為圓形，地圖中的障礙物以機器人半徑  $r_r$  膨脹。為了更進一步的安全，障礙物格點膨脹半徑為  $r_{r+s} = r_r + d_s$ ，其中  $d_s$  是機器人與障礙物格點之間的最小距離，也就是說對於每個障礙物格點，放大了角度  $\gamma_{i,j} = \arcsin \frac{r_{r+s}}{d_{i,j}}$ 。將圓形活動區域依據設定之解析度  $\alpha$  劃分為 ( $k = 360/\alpha$ )



個扇形區，對於每個扇形區  $k$ ，極坐標直方圖密度計算如下：

$$H_k^p = \sum_{i,j \in C_a} m_{i,j} \cdot h'_{i,j}, \text{ where } h'_{i,j} = \begin{cases} 1, & \text{if } k \cdot \alpha \in [\beta_{i,j} - \gamma_{i,j}, \beta_{i,j} + \gamma_{i,j}] \\ 0, & \text{otherwise} \end{cases} \quad (4.16)$$

2. 利用遲滯雙閾值，即  $\tau_{\text{low}}$  和  $\tau_{\text{high}}$ ，將極坐標直方圖轉換為用 0 (自由)、1 (佔據) 表示的二元極坐標直方圖  $H^b$ ，可表示哪些方向是自由的。在時間  $n$ ，二元極坐標直方圖用以下規則更新：

$$H_{k,n}^b = \begin{cases} 1, & \text{if } H_{k,n}^p > \tau_{\text{high}} \\ 0, & \text{if } H_{k,n}^p < \tau_{\text{low}} \\ H_{k,n-1}^b & \text{otherwise} \end{cases} \quad (4.17)$$

3. 將機器人兩側的轉向半徑定義為  $r$ ，左右轉向軌跡中心相對於目前機器人位置的位置定義為：

$$\Delta x_r = r \cdot \sin \theta, \Delta y_r = r \cdot \cos \theta \quad (4.18)$$

$$\Delta x_l = -r \cdot \sin \theta, \Delta y_l = -r \cdot \cos \theta$$

從障礙物格點  $(x_{\text{obstacle}}, y_{\text{obstacle}})$  到左右轉向軌跡中心的距離由下式給出：

$$\begin{aligned} d_r^2 &= (\Delta x_r - x_{\text{obstacle}})^2 + (\Delta y_r - y_{\text{obstacle}})^2 \\ d_l^2 &= (\Delta x_l - x_{\text{obstacle}})^2 + (\Delta y_l - y_{\text{obstacle}})^2 \end{aligned} \quad (4.19)$$

如果出現以下情況，障礙物格點將阻擋前進方向：

$$\begin{aligned} \text{條件一 : } d_r^2 &< (r + r_{r+s})^2 \\ \text{條件二 : } d_l^2 &< (r + r_{r+s})^2 \end{aligned} \quad (4.20)$$

接著計算兩個運動方向極限角度  $\varphi_r$  和  $\varphi_i$ ，首先定義  $\theta$  為目前運動方向， $\varphi_b = \theta + \pi$  為與目前運動方向相反的方向。一開始  $\varphi_r$  和  $\varphi_l$  初始值設定為  $\varphi_b$ ，接著對活動區域  $C_a$  中屬於障礙物的每個格點進行：(1) 如果  $\beta_{i,j}$  在  $\theta$

右邊和  $\varphi_r$  左邊，檢查式 (4.20) 條件一，如果滿足則  $\varphi_r$  等於  $\beta_{i,j}$ ；(2) 如果  $\beta_{i,j}$  在  $\theta$  左邊和  $\varphi_l$  右邊，檢查式 (4.20) 條件二，如果滿足則  $\varphi_l$  等於  $\beta_{i,j}$ 。最終可以得到  $\varphi_r$  和  $\varphi_b$ ，並透過  $\varphi_r$ 、 $\varphi_b$  和二元極坐標直方圖，用以下規則構建遮蔽極坐標直方圖：

$$H_k^m = \begin{cases} 0, & \text{if } H_k^b = 0 \text{ and } (k \cdot \alpha) \in \{[\varphi_r, \theta], [\theta, \varphi_l]\} \\ 1, & \text{otherwise} \end{cases} \quad (4.21)$$

4. 首先確定遮蔽極坐標直方圖中，所有開口的左右邊界  $k_r$  和  $k_l$ ，並區分開口類型。如果一個開口的兩個邊界之間的差異大於  $s_{\text{mux}}$  個扇形區，則為寬開口，否則為窄開口。對於窄開口，只有一個候選方向可以讓機器人通過相應障礙物之間的間隙中心，如式 (4.22)。

$$c_d = \frac{k_r + k_l}{2} \quad \text{centered direction} \quad (4.22)$$

對於寬開口，至少有兩個候選方向， $c_r$  到開口的右側， $c_l$  到開口的左側， $c_r$  和  $c_l$  使機器人在安全距離內跟隨障礙物輪廓。如果目標方向  $c_t$  位於其他兩個方向之間，則可列為其中一個候選方向，引導機器人朝向目標方向。詳細如式 (4.23)。

$$\begin{aligned} c_r &= k_r + s_{\text{max}}/2 && \text{towards the right side} \\ c_l &= k_l - s_{\text{max}}/2 && \text{towards the left side} \\ c_t &= k_t && \text{if } k_t \in [c_r, c_l] \end{aligned} \quad (4.23)$$

接下來定義一個合適的成本函數  $g(c)$ ，讓機器人選擇最合適的候選方向  $c$  作為其新的運動方向  $\varphi_d$ ：

$$g(c) = \mu_1 \cdot \Delta(c, k_t) + \mu_2 \cdot \Delta\left(c, \frac{\theta_n}{\alpha}\right) + \mu_3 \cdot \Delta(c, k_{d,n-1}) \quad (4.24)$$

其中  $\Delta(c_1, c_2)$  是計算兩個扇形區  $c_1$  和  $c_2$  之間的絕對角度差的函數。

成本函數  $g(c)$  的第一項表示候選方向和目標方向的差值相關成本， $\mu_1$  越高，

機器人的行為就越以目標為導向。第二項表示候選方向和機器人目前方向的差值相關成本， $\mu_2$  越高，機器人越會嘗試以最小的運動方向變化執行平滑路徑。第三項表示候選方向和先前選擇的運動方向的差值相關成本， $\mu_3$  越高，機器人嘗試朝先前選擇的方向前進的次數增加，轉向指令就越平滑。對於目標導向的行為，必須滿足以下條件：

$$\mu_1 > \mu_2 + \mu_3 \quad (4.25)$$

最終即可輸出新的最佳運動方向  $\varphi_d$ 。

## 4.8 對現有自主探索架構的修改

主要修改自 Umari et al. 所提出的單機器人及多機器人自主探索架構 [1, 5]。主要修改如下：

1. 添加地圖合併以整合兩個自主探索架構 [1, 5]，地圖合併將會於下一章進行說明。成功的地圖合併將採用多機器人探索策略，失敗的地圖合併將採用單機器人探索策略。
2. 將全域路徑規劃從 RRT 更改為 RRT star，因為 RRT star 具有更平滑的路徑。將局部路徑規劃改為 VFH+，因為其較為安全穩健。
3. 機器人之間會持續傳遞通信信號強度，以處理本研究提出之地圖合併。
4. SLAM 簡化為建構地圖，因為本文假設里程計沒有誤差。



## 提出之地圖合併方法

第四章已說明本研究多機器人自主探索（圖 3.5）除了地圖合併以外的演算法，而第三章已簡述本研究之地圖合併架構，因此本章將詳細說明本研究之地圖合併。為了解決特徵方法難以成功合併，而直接類則會需要儲存及處理大量資料，降低探索效率的問題，本研究將提出一個使用信號強度預測距離，並使用運動學分析來合併地圖的方法，再加入修改之 Ferrão [4] 的特徵地圖合併方法進行修正。本章將說明本研究地圖合併（圖 3.2）的演算法，即

- 用於信號強度預測距離之模型（Distance Prediction Model）
- 用於透過預測距離計算相對變換關係的運動學分析（Kinematic Analysis）
- 用於評判重疊情況的接受指標（Acceptance Index）
- 用於修正相對變換關係的修改之特徵地圖合併方法（Modification of Indirect Method）
- 用於合併地圖影像的熵濾波演算法（Entropy Filter）

### 5.1 信號強度預測距離之模型

根據假設，本研究使用 5 GHz WiFi (IEEE 802.11ac) 進行通訊。IEEE 802.11 是一個由 IEEE (Institute of Electrical and Electronics Engineers) 制定的無線網路標準，也稱為 WiFi。它定義了一系列的技術規範，包括無線網路的協定、頻率、速

率、功率管理和安全等方面。而 802.11ac 是 IEEE 802.11 標準中的一種無線網路技術，也被稱為 Gigabit WiFi 或者 5G WiFi，被設計用來提供更高的傳輸速度、更大的容量和更好的覆蓋範圍。802.11ac 標準支持 5GHz 頻段，相較於 2.4GHz 頻段，它提供更少的干擾和更大的頻寬。本研究之機器人會持續利用 5 GHz WiFi 來通信，也因此信號源為機器人，而信號強度預測距離之模型是基於信號強度模型所得出，因此首先介紹信號強度模型。

### 5.1.1 信號強度模型

信號強度基本模型如式(5.1)，也稱為 RSSI (Received Signal Strength Indicator) 模型，其中， $SS$  表示信號強度，單位為 dB； $P$  表示發射功率，單位為 dBm； $L$  表示路徑損耗，單位為 dB。

$$SS = P - L \quad (5.1)$$

其中的路徑損耗，T. Adame et al. [52] 針對 5 GHz WiFi (IEEE 802.11ac) 進行路徑損耗模型的擬合與驗證，其硬體使用 AP TP-Link Archer C7 C1750 V4 進行測，此路由器支持 IEEE 802.11ac 標準，可提供高達 1.75 Gbps 的無線數據傳輸速率，可以在 5 GHz 頻段上實現高達 1300 Mbps 的無線速度。

而本研究之通信強度模型採用 [52] 中，均方根誤差 (Root-Mean-Square Error, RMSE) 最低之模型，也就是基於牆壁因素之路徑損耗模型 (Wall Factor Path Loss Model)，如式 (5.2)。

$$L = L_{d_0} + 10 \times \gamma \times \log_{10} \left( \frac{d_m}{d_0} \right) + nW \times WAF \quad (5.2)$$

其中  $L_{d_0}$  是參考距離  $d_0$  量測出的信號衰減， $d_0$  通常為 1m， $\gamma$  是路徑損失率， $d_m$  是距離， $nW$  是牆壁的數量， $WAF$  是牆壁衰減係數。參數設置如下表：

而 Wi-Fi 的發射功率  $P$  是由各國相關部門和組織制定的法規和標準來規定的，通常是在不同頻段、使用場景和環境下設置不同的發射功率限制。例如，在美國，聯邦通訊委員會 (Federal Communications Commission, FCC) 規定了 Wi-Fi 設備的發射功率不得超過 20 dBm [56]，也因此本研究之發射功率  $P$  就設定為 20 dBm。

接著根據 [52] 可以知道基於牆壁因素之路徑損耗模型相對於真實數據之 RMSE 為 4.8237，因此本研究之真實路徑損耗模擬會考慮此 RMSE 作為雜訊，也



表 5.1: 路徑損耗模型參數數值表。

路徑損耗模型參數	數值
$d_0$	1(m)
$L_{d_0}$	54.1200(dB)
$\gamma$	2.06067
$WAF$	5.25(dB)

就是從一個平均為 0，標準差為 4.8237 的高斯分佈中隨機生成一個值作為雜訊，得出雜訊路徑損耗模型，如式 (5.3)。

$$L_{noise} = L + \mathcal{N}(0, 4.8237^2) \quad (5.3)$$

其中， $\mathcal{N}(0, 4.8237^2)$  表示平均為 0，標準差為 4.8237 的高斯分佈。

最終可以根據式 (5.1~5.3)，得出帶有雜訊之模擬信號方程式 (式 (5.4))，並模擬出帶有雜訊之信號強度。

$$SS = P - L_{d_0} - 10 \times \gamma \times \log_{10} \left( \frac{d_m}{d_0} \right) - nW \times WAF - \mathcal{N}(0, 4.8237^2) \quad (5.4)$$

### 5.1.2 信號強度資料處理

模擬出帶有雜訊之信號強度後必須進行資料處理。兩台機器人相互通訊時，它們之間的相對距離相同，因此在理論上應該具有相同的信號強度。然而，由於存在雜訊的干擾，兩台機器人得到的信號強度可能會有所不同，這表示在同一時間內，機器人之間的信號強度可能會有兩個值。而將多次測量平均可以減少隨機誤差，提高測量的精確度和可靠性。因此，為了降低雜訊所造成的影响，本研究對於同一時間點上獲得的兩個信號強度值進行平均。

接著將離群值 (Outlier) 去除並替換，離群值是指在數據集中與其他數據值明顯不同的極端值。離群值會對數據分析結果造成影響，甚至影響決策的準確性和可靠性。因此，需要將其從數據集中去除或進行處理。本研究使用移動窗中位數平滑法及線性插值替換法去除離群值，詳細方法如下：



對於一個長度為  $n$  的數據序列  $x_1, x_2, \dots, x_n$ ，移動窗中位數平滑法可以表示為：

- 紿定一個窗口大小  $k$ （本研究設為 3），移動窗口遍歷整個數據序列。
- 在每個窗口中取出數據點  $x_i, x_{i+1}, \dots, x_{i+k-1}$ ，並將其按大小排序。
- 取排序後的中位數  $med$ ，將其作為窗口中心的新值  $x_{i+\lfloor k/2 \rfloor}$ 。
- 重複上述過程直到窗口遍歷完整個數據序列。

對於一個長度為  $n$  的數據序列  $x_1, x_2, \dots, x_n$ ，移動窗中位數離群值使用線性插值替換法可以表示為：

- 紿定一個窗口大小  $k$ （本研究設為 3），移動窗口遍歷整個數據序列。
- 在每個窗口中取出數據點  $x_i, x_{i+1}, \dots, x_{i+k-1}$ ，組成一個長度為  $k$  的子序列。
- 計算子序列中的中位數  $med = median(x_i, x_{i+1}, \dots, x_{i+k-1})$ ，其中  $median$  表示中位數運算符號。
- 定義移動窗口內的平均絕對偏差為  $MAD_i = c \times median(|(x_i, x_{i+1}, \dots, x_{i+k-1}) - med|)$ ，其中， $c = -1/(\sqrt{2} \times erfcinv(3/2))$ ， $erfcinv$  表示誤差函數的反函數。
- 如果  $x_i$  與中位數的偏差大於  $3 \times MAD_i$ ，就認為  $x_i$  是離群值，並使用線性插值法，將其替換為相鄰兩個數據點的平均值。

最後對資料進行斷層處理，由於牆壁造成之信號衰減是不連續的，從式 (5.4) 可以知道每個牆壁會造成值為  $WAF$  的信號衰減值，因此只要斜率足夠大的區域即代表此處可能改變了牆壁數量或是誤差造成影響，在此定義為斷層區；而斷層區以外的區域定義為非斷層區，非斷層區代表此區域的牆壁數量是一樣的，沒有改變牆壁數量。接著將每一個非斷層區的數據進行平滑處理，用於降低資料的雜訊和變異性，而不將斷層區考慮進去平滑處理，是因為斷層區被平滑後會失去其斜率足夠大的特性，平滑方式採用 S-G 濾波器（Savitzky-Golay Filter）[57] 進行處理，詳細方法如下：

對於一個長度為  $n$  的數據序列  $x_1, x_2, \dots, x_n$ ，S-G 濾波器平滑處理可以表示為：



- 給定一個窗口大小  $k = 2m + 1$  (本研究設為 5)，移動窗口遍歷整個數據序列。
- 在每個窗口中取出數據點  $\{x_t\} = \{x_{-m}, x_{-m+1}, \dots, x_0, \dots, x_{m-1}, x_m\}$ ，組成一個長度為  $k$  的子序列。
- 假設擬合二次多項式為  $x(t) = a_0 + a_1 * t + a_2 * t^2$ ，其中  $t$  表示待擬合之數據， $a_i$  為待求解之參數， $x(t)$  為擬合後的多項式。
- 將數據點  $\{x_t\}$  帶入擬合二次多項式，其中  $\varepsilon$  是誤差

$$\begin{pmatrix} x_{-m} \\ \vdots \\ x_0 \\ \vdots \\ x_m \end{pmatrix} = \begin{pmatrix} 1 & t_{-m} & t_{-m}^2 \\ \vdots & \vdots & \vdots \\ 1 & t_0 & t_0^2 \\ \vdots & \vdots & \vdots \\ 1 & t_m & t_m^2 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix} + \begin{pmatrix} \varepsilon_{t_{-m}} \\ \vdots \\ \varepsilon_{t_0} \\ \vdots \\ \varepsilon_{t_m} \end{pmatrix} \quad (5.5)$$

- 上述方程式可以簡化為  $\mathbf{X}_{(2m+1) \times 1} = \mathbf{T}_{(2m+1) \times 3} \mathbf{A}_{3 \times 1} + \mathbf{E}_{(2m+1) \times 1}$ ，接著使用最小二乘法進行求解得出最佳  $\mathbf{A}_{best}$  的解為  $\mathbf{A}_{best} = (\mathbf{T}^T \cdot \mathbf{T})^{-1} \cdot \mathbf{T}^T \cdot \mathbf{X}$
- 最終得出平滑處理的數據為  $\mathbf{X}_{smooth} = \mathbf{T} + \mathbf{A}_{best}$

最後計算斷層區最大及最小的信號值之差值，當此差值接近  $WAF$  及其倍數時，代表此斷層是牆壁造成之信號衰減，將斷層區之差值調整為  $WAF$  或其倍數且斜率為  $WAF/\Delta t$ ，否則視為雜訊造成之誤差，不進行處理。

### 5.1.3 預測距離模型及修正

根據式 (5.1~5.3) 進行預測距離公式推導，最終可以得到預測距離公式如式 (5.6)，其中，由於牆壁數量  $nW$  是未知的，因此首先假設為 0，接著再進行修正。

$$d_m = d_0 \times 10^{(P - SS - L_{d_0} - nW \times WAF)/(10 \times \gamma)} \quad (5.6)$$

修正方法與前一小節中的斷層處理相同，將斜率足夠大的區域定義為斷層區，而斷層區以外的區域則被定義為非斷層區。在同一個非斷層區內的數據代表相同的牆壁數量  $n_W$ ，而不同非斷層區之間則代表不同的牆壁數量  $n_W$ 。

接下來，可以從式 (5.6) 得知增加牆壁數量  $n_W$  會導致預測距離  $d_m$  的降低。因此，本研究可以將非斷層區中數值最低的一層作為底層，首先假設底層牆壁數量為 0，並使用增加牆壁數量  $n_W$  的方式將其他非斷層區的預測距離數據修正為與底層對齊。修正公式如式 (5.7) 所示，該公式是根據式 (5.1) 到式 (5.3) 推導得出的，詳細的修正流程如下：

1. 將非斷層區中數值最低的一層作為底層，假設  $n_W = 0$ 。
2. 對於其他每個非斷層區，增加牆壁數量  $n_W = n_W + 1$ 。
3. 對於其他每個非斷層區中的預測距離數據  $d_m$ ，以式 (5.7) 修正為  $d_m^{correction}$ 。

$$d_m^{correction} = 10^{(\log_{10} d_m) - (nW \times WAF)/(10 \times \gamma)} \quad (5.7)$$

4. 重複步驟 2~3，到底層之外的非斷層區的預測距離數據都與底層對齊。

因為假設底層牆壁數量為 0，因此若是底層數量不為零，就會導致後續計算相對關係錯誤，因此本研究流程最後一步判斷相對關係錯誤，就會回到本小節來修正底層牆壁，也就是底層牆壁數量加一，並將預測距離數據修正為與底層對齊，再重新接續執行本研究流程，到底層牆壁數達到上限，本研究上限設為 5，才會判斷無法合併地圖。

最後與信號資料去除離群值相同，將修正預測距離數據的離群值去除並替換，同樣使用移動窗中位數平滑法及線性插值替換法去除離群值，詳細流程可見 5.1.2 節。處理離群值之後，最後對資料進行平滑處理，以降低資料的雜訊和變異性，在此採用與上一章相同的 S-G 濾波器進行處理，詳細流程可見 5.1.2 節，最終仍然會有誤差存在，誤差部分將仰賴後續演算法進行排除。

## 5.2 運動學分析計算變換關係

本節以機動學的運動學分析為發想，利用前一章得出之時間點  $t_i$  兩個機器人個體的相對距離  $\{d_i\}_{i=1,2,\dots,n}$ ，以及時間點  $t_i$  兩個機器人個體在其局部地圖  $\mathbf{M}_1$ 、

$\mathbf{M}_2$  的位置，即  $(x_i^{\mathbf{M}_1}, y_i^{\mathbf{M}_1})_{i=1,2,\dots,n}$  和  $(x_i^{\mathbf{M}_2}, y_i^{\mathbf{M}_2})_{i=1,2,\dots,N}$ ，將兩個局部地圖視為桿件，在局部地圖的位置視為局部地圖桿件之旋轉對，相對距離視為連接桿件，透過多個連接桿件即可將兩個局部地圖桿件進行固定，達到地圖合併的效果，如圖 5.1，也就可以得出局部地圖間的相對變換關係  $\mathbf{T}^{12}$ 。而因為此方法與地圖特徵無關，所以可以處理重複特徵環境。

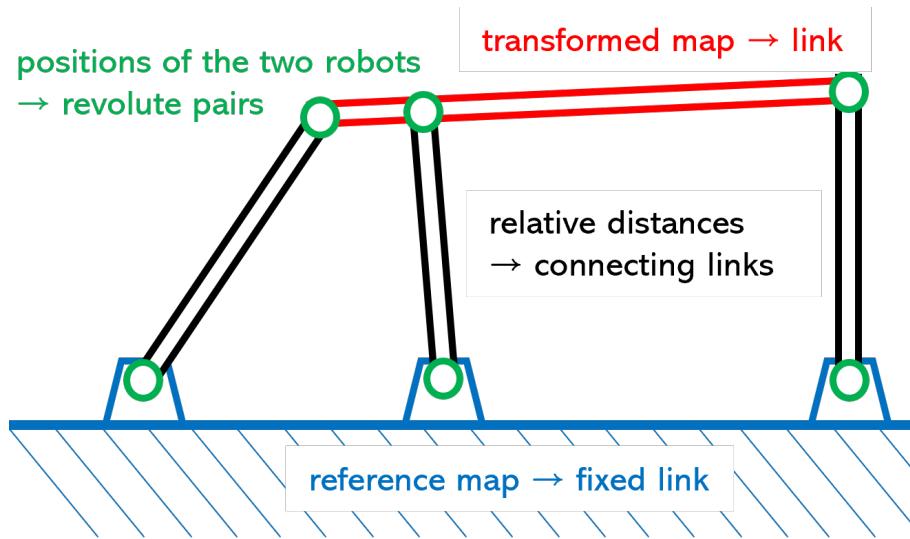


圖 5.1: 以連桿機構進行地圖合併概念圖。

一個具有  $N$  個桿件之平面機構的自由度，可用 Grubler-Kutzbach criteria 判別式 [58] 來求出，只有旋轉對之機構的 Grubler-Kutzbach criteria 判別式為：

$$F = 3(N - 1) - 2 \times J_r \quad (5.8)$$

其中  $F$  為機構之自由度， $N$  為所有桿件數目， $J_r$  為旋轉對數目。而本研究中，桿件數目為兩個地圖桿件加上  $n$  個時間點的相對距離，也就是  $N = 2 + n$ ；旋轉對數目為  $n$  個時間點的兩個局部地圖的位置，也就是  $J_r = 2 \times n$ 。經過代換可以得到式 (5.9)。

$$F = 3 - n \quad (5.9)$$

要使兩個局部地圖桿件固定，也就代表桿件間無相對運動，Grubler-Kutzbach criteria 判別式得出之值必須符合下列條件：

$$F \leq 0 \quad (5.10)$$



從式(5.9)及式(5.10)可以知道，要使兩個局部地圖桿件固定，必須要取至少3個時間點，也就是 $n \geq 3$ 。但此條件只是使桿件固定而已，並不能確保固定的方式為唯一，因此為了確保唯一解，必須使約束條件增加，也就代表所取的時間點數量必須大於3。

接著進行符號定義，以局部地圖 $\mathbf{M}_2$ 為基準，找出局部地圖 $\mathbf{M}_1$ 與 $\mathbf{M}_2$ 的相對關係，詳細圖示如圖5.2，下列將詳細說明每個符號定義：

- $t_i$ ：第*i*個時間點。
- $L_{i,j}^{\mathbf{M}_k}$ ：在局部地圖 $\mathbf{M}_k$ ， $t_i$ 及 $t_j$ 時間點的距離。
- $\alpha_{i,j}^{\mathbf{M}_k}$ ：在局部地圖 $\mathbf{M}_k$ ， $t_i$ 及 $t_j$ 時間點的位置連線與水平線的夾角。
- $d_i$ ：在 $t_i$ ，兩個機器人個體的相對距離。
- $\theta_i$ ： $d_i$ 與水平線的夾角。
- $\varphi$ ：局部地圖 $\mathbf{M}_2$ 與 $\mathbf{M}_1$ 合併所需旋轉的角度。
- $[t_x, t_y]$ ：局部地圖 $\mathbf{M}_2$ 與 $\mathbf{M}_1$ 合併所需移動的距離。

除了 $\{\theta_i\}_{i=1,2,\dots,n}, \varphi, [t_x, t_y]$ ，其餘作為輸入參數皆是已知。

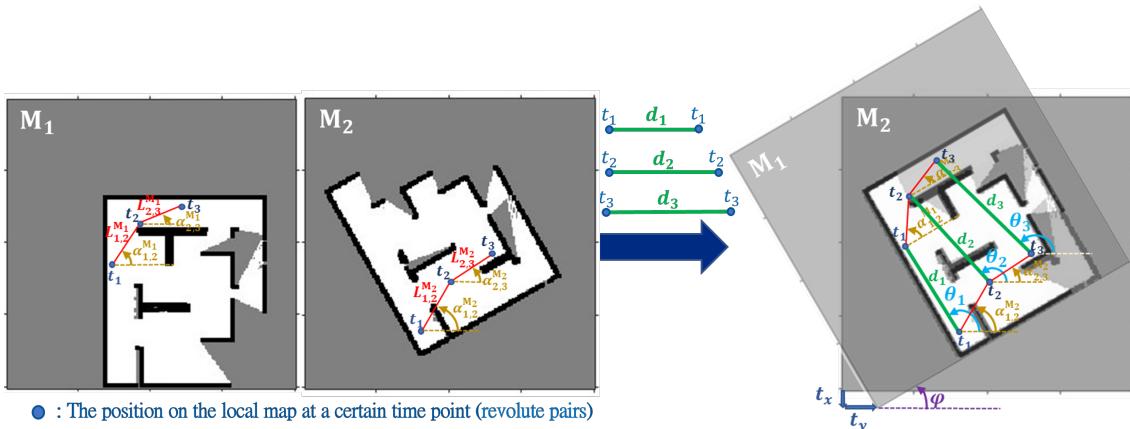
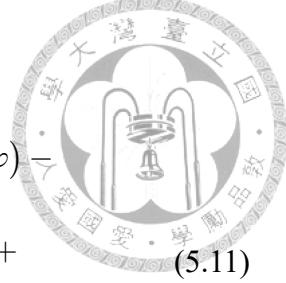


圖5.2: 以局部地圖 $\mathbf{M}_2$ 為基準，使用距離計算變換關係之符號定義。

在進行計算模型的詳細步驟之前，先說明會使用到之方程式。當有*n*個時間



點在  $\mathbf{M}_1$  跟  $\mathbf{M}_2$  時，可以利用向量迴圈的概念推導出式 (5.11)。

$$\begin{aligned}
 L(\varphi, \{\theta_i\}_{i=1,2,\dots,n}) = 0 = & \sum_{i=1}^n d_i \sin \theta_i + L_{i,i+1}^{\mathbf{M}_1} \sin (\alpha_{i,i+1}^{\mathbf{M}_1} + \varphi) - \\
 & d_{i+1} \sin \theta_{i+1} - L_{i,i+1}^{\mathbf{M}_2} \sin \alpha_{i,i+1}^{\mathbf{M}_2} + \\
 & d_i \cos \theta_i + L_{i,i+1}^{\mathbf{M}_1} \cos (\alpha_{i,i+1}^{\mathbf{M}_1} + \varphi) - \\
 & d_{i+1} \cos \theta_{i+1} - L_{i,i+1}^{\mathbf{M}_2} \cos \alpha_{i,i+1}^{\mathbf{M}_2}
 \end{aligned} \tag{5.11}$$

若是預測之距離沒有誤差，則可以直接透過式 (5.11) 得出未知參數，但由於預測之距離是有誤差的，因此必須將式 (5.11) 視為目標函數，並使用最佳化的方式得出未知參數  $\varphi, \{\theta_i\}_{i=1,2,\dots,n}$ ，如式 (5.12)。在此使用 LM 演算法 (Levenberg-Marquardt Algorithm) 來進行最佳化求解，其為一種常用的非線性最小二乘最佳化方法，對於擬合曲線或非線性參數估計非常有效 [59]。

$$\arg \min_{\varphi, \{\theta_i\}_{i=1,\dots,n}} L(\varphi, \{\theta_i\}_{i=1,2,\dots,n}) \tag{5.12}$$

利用式 (5.12) 得出  $\varphi$  之後，即可利用  $\varphi$  及任意時間點  $t_i$  兩個機器人個體在其局部地圖  $\mathbf{M}_1$ 、 $\mathbf{M}_2$  的位置： $(x_i^{\mathbf{M}_1}, y_i^{\mathbf{M}_1})_{i=1,2,\dots,n}$  和  $(x_i^{\mathbf{M}_2}, y_i^{\mathbf{M}_2})_{i=1,2,\dots,N}$ ，計算出平移量  $t_x, t_y$ ，如式 (5.13)。

$$\begin{bmatrix} t_x \\ t_y \end{bmatrix} = \begin{bmatrix} x_i^{\mathbf{M}_2} + d_i * \cos \theta_i \\ y_i^{\mathbf{M}_2} + d_i * \sin \theta_i \end{bmatrix} - \begin{bmatrix} \cos \varphi & -\sin \varphi \\ \sin \varphi & \cos \varphi \end{bmatrix} \begin{bmatrix} x_i^{\mathbf{M}_1} \\ y_i^{\mathbf{M}_1} \end{bmatrix} \tag{5.13}$$

接著進行計算模型的詳細步驟說明，首先令上述之資料作為本演算法的輸入資料集合  $\mathcal{X}$ ，即  $\mathcal{X} = \{d_{t_i}, (x_{t_i}^{\mathbf{M}_1}, y_{t_i}^{\mathbf{M}_1}), (x_{t_i}^{\mathbf{M}_2}, y_{t_i}^{\mathbf{M}_2})\}_{t_i=1,2,\dots,N}$ ，接著可計算兩地圖的相對變換關係  $\mathbf{T}^{12}$  (式 (3.3))。計算模型的詳細步驟如下：



1. 首先根據式 (3.3) 定義猜測模型  $\mathbf{T}^{12,guess}$  :

$$\mathbf{T}^{12,guess} = \begin{bmatrix} \cos \varphi & -\sin \varphi & t_x \\ \sin \varphi & \cos \varphi & t_y \\ 0 & 0 & 1 \end{bmatrix} \quad (5.14)$$

接著從時間序中任意取出  $n$  個時間點，但由於預測距離誤差，若是時間點間隔距離太小，雜訊造成的距離預測誤差就等同被放大，因此時間點的選取必須符合式 (5.15)，也就是在兩張地圖的時間點  $t_i$  與下一個時間點  $t_{i+1}$  之距離，必須大於閥值 ( $d_{Threshold}$ )。

$$\begin{aligned} \sqrt{(x_{i+1}^{\mathbf{M}_1} - x_i^{\mathbf{M}_1})^2 + (y_{i+1}^{\mathbf{M}_1} - y_i^{\mathbf{M}_1})^2} &> d_{Threshold} \\ \sqrt{(x_{i+1}^{\mathbf{M}_2} - x_i^{\mathbf{M}_2})^2 + (y_{i+1}^{\mathbf{M}_2} - y_i^{\mathbf{M}_2})^2} &> d_{Threshold} \end{aligned} \quad (5.15)$$

最後根據式 (5.11~5.13) 計算猜測模型的三個參數  $\varphi$ 、 $t_x$ 、 $t_y$ 。

2. 根據猜測模型，測試該模型的相對變換關係接受指標（詳細可見 5.3 節），根據式 (2.4)，即可得到該模型之相對變換關係接受指標  $\omega$ 。記錄猜測模型的相對變換關係接受指標  $\omega$ ，若  $\omega$  超過先前的記錄，則更新目前猜測模型為最佳模型  $\mathbf{T}^{12,best}$ 。
3. 若是最佳模型  $\mathbf{T}^{12,best}$  之  $\omega$  符合相對變換接受條件（式 (3.4)），則輸出最佳模型  $\mathbf{T}^{12,best}$ 。若否，則重複步驟 1、2，直到到達迭代上限  $k$  次，輸出接受指標  $\omega$  紀錄最高之最佳模型  $\mathbf{T}^{12,best}$ 。
4. 使用  $\mathbf{T}^{12,best}$  對  $\mathbf{M}_1$  轉換後得到  $\mathbf{M}'_1$ ，接著根據最終輸出之最佳模型  $\mathbf{T}^{12,best}$  相對變換關係接受指標  $\omega$  跟符合相對變換接受條件（式 (3.4)），來進行分類及處理，如表5.2。

表 5.2: 變換關係接受指標  $\omega$  分類及處理關係表。

$\omega$ 範圍	分類	處理
$\omega \geq 0.95$	不須修正	直接使用 5.5 節的方法輸出合併地圖
$\omega < 0.95$	須修正	使用 $\mathbf{M}'_1$ 及 $\mathbf{M}_2$ 透過特徵進行修正



### 5.3 相對變換關係接受指標

如同 2.3.2 節所介紹，若有  $\mathbf{M}_1$  與  $\mathbf{M}_2$  兩張地圖的相對關係模型  $\mathbf{T}^{12}$ ，即可將地圖  $\mathbf{M}_1$  進行變換：

$$\mathbf{M}'_1 = \mathbf{T}^{12}\mathbf{M}_1 \quad (5.16)$$

根據式 (5.16) 將地圖  $\mathbf{M}_1$  的座標系轉為地圖  $\mathbf{M}_2$  的坐標系。經變換後的地圖  $\mathbf{M}'_1$  與  $\mathbf{M}_2$  部分重疊，這些重疊的格點組成一個新的地圖  $\mathbf{M}_3$ ，其被定義為重疊地圖。在  $\mathbf{M}_3$  中，每個格點都包含了來自  $\mathbf{M}'_1$  和  $\mathbf{M}_2$  的像素值，即

$$\mathbf{M}_3(x, y) = \{m'_{1,xy}, m_{2,xy}\} \quad (5.17)$$

依據兩個像素值分別所代表的區域，可定義重疊地圖  $\mathbf{M}_3$  的格點為正確匹配 ( $agr$ ) 或錯誤匹配 ( $dis$ )：

$$\mathbf{M}_3(x, y) = \begin{cases} agr & , \text{if } m'_{1,xy} = m_{2,xy} = 0. \\ agr & , \text{if } m'_{1,xy} = m_{2,xy} = 255. \\ dis & , \text{if } m'_{1,xy} = 0, m_{2,xy} = 255. \\ dis & , \text{if } m'_{1,xy} = 255, m_{2,xy} = 0. \end{cases} \quad (5.18)$$

定義完地圖  $\mathbf{M}_3$  的格點後可根據式 (2.4) 計算目前重疊情況的指標分數，令  $agr(\mathbf{M}_1, \mathbf{M}_2)$  為地圖  $\mathbf{M}_3$  中  $agr$  格點的總數， $dis(\mathbf{M}_1, \mathbf{M}_2)$  為地圖  $\mathbf{M}_3$  中  $dis$  格點的總數，則指標分數為：

$$\omega(\mathbf{M}_1, \mathbf{M}_2) = \begin{cases} 0 & , \text{if } agr(\mathbf{M}_1, \mathbf{M}_2) = 0 \\ \frac{agr(\mathbf{M}_1, \mathbf{M}_2)}{agr(\mathbf{M}_1, \mathbf{M}_2) + dis(\mathbf{M}_1, \mathbf{M}_2)} & , \text{if } agr(\mathbf{M}_1, \mathbf{M}_2) \neq 0 \end{cases} \quad (2.4)$$



## 5.4 修改之特徵地圖合併方法

若是判斷  $T^{12,best}$  需修正的話，就會使用地圖  $M'_1$  及  $M_2$  進入本小節開始的特徵修正流程，本小節將介紹本研究修改之特徵地圖合併方法（圖 3.1）的流程及其中所使用的各項演算法，即

**步驟 1** 利用 ORB (Oriented FAST and Rotated BRIEF) 演算法對影像進行特徵擷取。

**步驟 2** 利用歐氏距離 (Euclidean Distance) 對 ORB 擷取的特徵進行窮舉匹配。

**步驟 3** 利用修改之隨機採樣共識演算法 (RANdom SAmple Consensus, RANSAC) 計算地圖相對變換關係。

### 5.4.1 ORB 特徵擷取

本小節將詳細說明 ORB 演算法的原理與流程。本研究選擇 ORB 演算法來替代現有地圖合併演算法 Ferrão [4] 使用的 SIFT，是因為其相較於 SIFT，執行的速度快了許多，同時保有旋轉不變性、尺度不變性和穩定性好等特性。

ORB 於 2011 年由 Ethan Rublee et al. [60] 提出，其通過使用金字塔技術對圖像進行不同尺度的縮放，在不同尺度下尋找特徵點，計算該特徵點周遭像素的梯度向量作為主方向，在生成特徵描述時，將特徵點周圍的像素按照特徵點的主方向旋轉，並作為該特徵點的特徵向量，使得所有特徵點的主方向都指向同一個方向。其具有旋轉不變性和尺度不變性，可以在旋轉和縮放的情況下對圖像進行有效的匹配。其主要特點包括旋轉不變性、尺度不變性、速度快和穩定性好。下列將介紹 ORB 演算法的流程，主要可分為三個步驟：

1. 不同尺度空間之 FAST 特徵檢測
2. ORB 特徵點方向計算
3. 具旋轉不變性之 BRIEF 特徵點描述



## 不同尺度空間之 FAST 特徵檢測

為了在不同尺度空間中找出穩定的特徵點，需要對影像進行模糊處理，以便在不同尺度空間上獲取影像信息。根據 Koenderink [61] 和 Lindeberg [62] 的合理假設，高斯函數是尺度空間中唯一可用的方法。因此，在檢測特徵點位置之前，必須對影像進行高斯模糊處理，其表示函數如下：

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y) \quad (5.19)$$

其中  $I(x, y)$  表示原始影像，運算子  $*$  表示卷積操作，而  $G(x, y, \sigma)$  如式 (5.20) 表示高斯函數。

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \quad (5.20)$$

高斯影像金字塔以層級 (Octave) 進行分層，層數根據影像大小而定，每一層包含多個不同尺度的高斯模糊影像。例如，最底層 (First Octave) 是原始影像大小，包含多個尺度空間函數  $L(x, y, \sigma), L(x, y, k\sigma), L(x, y, k^2\sigma), \dots$ ，其中  $\sigma$  為高斯函數的標準差， $k$  為常數。每個上一層級 (Next Octave)，原始影像進行降採樣，並重複進行高斯模糊處理，標準差改為前一層級的 2 倍，即  $L(x, y, 2\sigma), L(x, y, 2k\sigma), L(x, y, 2k^2\sigma), \dots$ 。

在建立高斯影像金字塔之後，為實現尺度不變性，需要使用具有參數  $\sigma^2$  的歸一化拉普拉斯算子 [62]，而此算子也可以使用高斯差分函數 (Difference of Gaussian, DOG) 進行近似。在特徵點檢測時，Lowe [49] 使用高斯差分函數建立高斯差分金字塔，其運算只需要將兩個不同尺度空間的影像進行相減，計算方式如下：

$$\begin{aligned} D(x, y, \sigma) &= (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) \\ &= L(x, y, k\sigma) - L(x, y, \sigma) \end{aligned} \quad (5.21)$$

高斯及高斯差分影像金字塔的關係可見於圖 5.3。

對於每一個高斯差分影像，都會使用 FAST 特徵檢測演算法來檢測該影像的特徵點。FAST 是一種高效的特徵檢測演算法，用於快速檢測影像中的角點特徵。該演算法將像素點  $p$  做分割測試來判斷是否為特徵點，該測試考慮以  $p$  為中心，半徑為 3 的圓所經過的 16 個像素（可調整半徑改變考慮像素數量），如圖 5.4，如果在圓內存在  $n$  個以上像素比  $p$  的像素值高或低於一個閾值  $\varepsilon_d$ ，就判斷為特徵

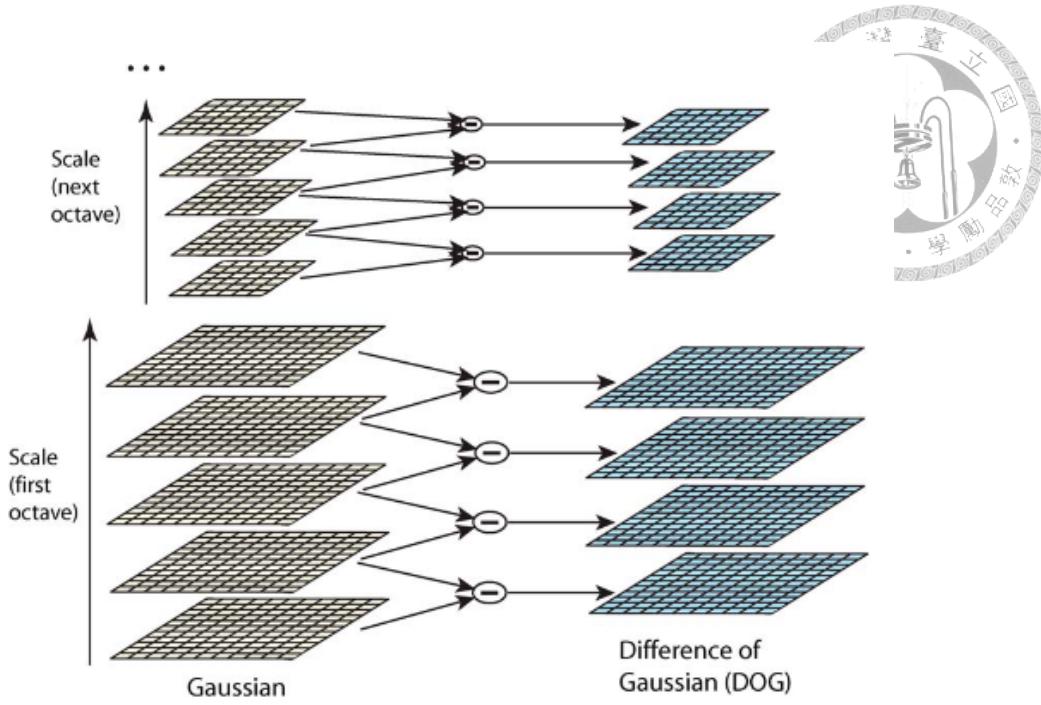


圖 5.3: 高斯及高斯差分影像金字塔計算示意圖 [7]。

點，如式 (5.22) 所示，通常  $n$  為 9。

$$\sum_{p_i \forall (\text{circle}(p))} (|D(p_i) - D(p)| > \varepsilon_d) \geq n \quad (5.22)$$

$D(p)$  表示像素  $p$  的高斯差分影像像素值

由於擷取的特徵點可能包含受到高斯差分函數增強邊緣響應而較不穩定的特徵點。ORB 演算法在捨棄不穩定特徵點時，是通過計算每個特徵點的 Harris 響應值來實現的，然後選擇具有較高響應值的特徵點。以下是 ORB 算法中選擇穩定特

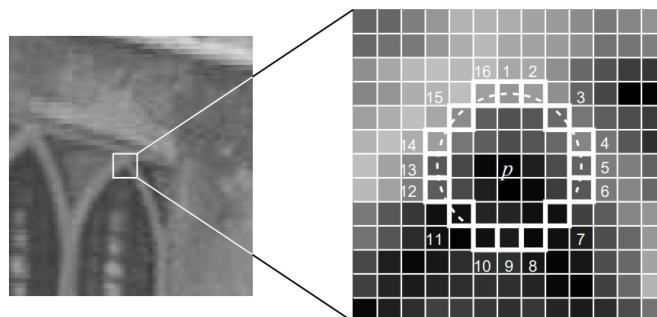


圖 5.4: FAST 特徵檢測演算法之分割測試。 $p$  是中心像素，半徑為 3 的圓由虛線表示，通過 16 個像素（突出顯示的正方形）。[8]



徵點的數學描述：

- 首先，對於每個特徵點  $p$  位於座標  $(i, j)$ ，計算其  $x$  和  $y$  方向的梯度  $g_x$  和  $g_y$ ：

$$\begin{aligned} g_x(i, j) &= D(i + 1, j) - D(i - 1, j) \\ g_y(i, j) &= D(i, j + 1) - D(i, j - 1) \end{aligned} \quad (5.23)$$

其中， $D(i, j)$  表示像素坐標的高斯差分影像像素值。

- 當存在不穩定的邊緣特徵點時，其高斯差分函數在跨越邊緣時會呈現較大的曲率，在垂直方向上則較小。透過比較曲率比例的關係，可以決定是否應該丟棄位於邊緣的特徵點。由於曲率比例與該特徵點位置的 Hessian 矩陣特徵值比例相同，因此需要計算每個特徵點  $p$  的 Hessian 矩陣：

$$\mathbf{H} = \begin{bmatrix} g_x(i, j)^2 & g_x(i, j) * g_y(i, j) \\ g_x(i, j) * g_y(i, j) & g_y(i, j)^2 \end{bmatrix} \quad (5.24)$$

- 假設特徵點  $p$  的 Hessian 矩陣的特徵值為  $\lambda_1$  與  $\lambda_2$ ，其中  $\lambda_1 > \lambda_2$ ，並可以透過計算每個特徵點  $p$  的 Harris 響應值  $R_p$ ，來量化 Hessian 矩陣的特徵值：

$$R_p = \text{Det}(\mathbf{H}) - k(\text{Tr}(\mathbf{H}))^2 \quad (5.25)$$

其中， $\text{Det}(\mathbf{H})$  和  $\text{Tr}(\mathbf{H})$  分別表示  $\mathbf{H}$  矩陣的行列式和迹，如式 (5.26)， $k$  是一個常數，通常取值為 0.04 到 0.06。

$$\begin{aligned} \text{Tr}(\mathbf{H}) &= g_x(i, j)^2 + g_y(i, j)^2 = \lambda_1 + \lambda_2 \\ \text{Det}(\mathbf{H}) &= g_x(i, j)^2 g_y(i, j)^2 - (g_x(i, j) * g_y(i, j))^2 = \lambda_1 \lambda_2 \end{aligned} \quad (5.26)$$

- 最後，ORB 演算法會選擇具有較高 Harris 響應值的特徵點作為穩定特徵點。換句話說，ORB 演算法會設置一個 Harris 響應值閾值，當一個特徵點的 Harris 響應值小於該閾值時，就會從特徵點中刪除。



## ORB 特徵點方向計算

得到特徵點後，根據周圍影像資訊計算該特徵點方向，使其具有旋轉不變性。影像資訊來自於該關鍵點同一尺度空間的模糊影像，以便維持尺度不變性。是以模糊影像  $L$  中各特徵點來計算，以該點為圓心，在指定半徑內做圓，圓內的像素值作為質量，計算出該圓質心，圓心指向質心的向量即為主方向。詳細計算如下：

1. 在半徑為  $R$  的圓形影像區域內，沿著兩個坐標軸  $x, y$  方向計算圖像矩：

$$\begin{aligned} m_{10} &= \sum_{x=-R}^R \sum_{y=-R}^R xL(x, y) \\ m_{01} &= \sum_{x=-R}^R \sum_{y=-R}^R yL(x, y) \end{aligned} \quad (5.27)$$

2. 圓形區域內所有像素值總和為：

$$m_{00} = \sum_{x=-R}^R \sum_{y=-R}^R L(x, y) \quad (5.28)$$

3. 圖像的重心為：

$$C = (c_x, c_y) = \left( \frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}} \right) \quad (5.29)$$

4. 特徵點  $L(x, y)$  主方向為：

$$\theta = \text{atan2}(m_{01}/m_{10}) \quad (5.30)$$

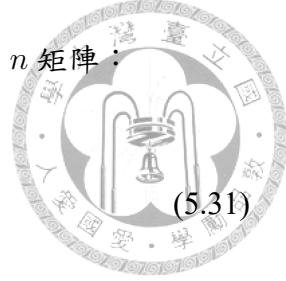
## 具旋轉不變性之 BRIEF 特徵描述

求得特徵點及主方向資訊後，接著需以一組向量描述特徵點，稱為描述符 (Descriptor)，使特徵點可去除亮度、視角變化造成的影响。

BRIEF 描述符是一組由強度測試構建的二進制編碼描述符，以特徵點為中心，取固定大小的窗口，窗口大小為  $31 \times 31$  像素 [60]，接著將窗口進行平滑處理，平滑方式為使用  $5 \times 5$  的 kernel 進行高斯平滑處理 [60]，最後從平滑窗口中隨機選擇

$n$  對像素  $x_i$  和  $y_i$ ，本研究  $n$  為 128，並將  $n$  對像素，定義成  $2 \times n$  矩陣：

$$\mathbf{S} = \begin{pmatrix} x_1, \dots, x_n \\ y_1, \dots, y_n \end{pmatrix}$$



接著使用主方向  $\theta$  和對應的旋轉矩陣  $\mathbf{R}_\theta$ ，將  $\mathbf{S}$  旋轉為  $\mathbf{S}_\theta$ ：

$$\mathbf{S}_\theta = \mathbf{R}_\theta \mathbf{S}, \quad (5.32)$$

進行強度測試， $(x_i, y_i) \in \mathbf{S}_\theta$ ，如果  $x_i$  的像素值大於等於  $y_i$ ，則記為 0，小於則記為 1。考慮一個平滑窗口  $p$ ，強度測試  $\tau$  定義為：

$$\tau(p; x, y) := \begin{cases} 1 & : p(x) < p(y) \\ 0 & : p(x) \geq p(y) \end{cases}, \quad (5.33)$$

其中  $p(x)$  是  $p$  在  $x$  的像素值。 $n$  維的二進制編碼描述符：

$$g_n(p, \theta) := \sum_{1 \leq i \leq n} 2^{i-1} \tau(p; x_i, y_i) \mid (x_i, y_i) \in \mathbf{S}_\theta \quad (5.34)$$

### 5.4.2 窮舉特徵匹配

前一小節已介紹了利用 ORB 演算法對影像特徵點進行描述，假設地圖影像  $\mathbf{M}_1$  所擷取出的特徵點集合為  $\mathcal{P} = \{p_1, p_2, \dots, p_n\}$ ，地圖影像  $\mathbf{M}_2$  所擷取出的特徵點集合為  $\mathcal{Q} = \{q_1, q_2, \dots, q_m\}$ 。在兩個特徵點集合中，每一個特徵點皆具有 128 維的描述符向量，因此可利用歐氏距離比對特徵點向量的相似度：

$$d(x, y) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \dots + (x_n - y_n)^2} \quad (5.35)$$

式 (5.35) 為歐式距離的計算，代數  $x$ 、 $y$  為欲計算相似度的兩個向量，有下標的代數  $x_i$ 、 $y_i$  分別表示向量  $x$ 、 $y$  的分量。

接著利用式 (5.35) 對集合  $\mathcal{P}$  以及集合  $\mathcal{Q}$  的各特徵點兩兩計算歐式距離，進行窮舉匹配，也就是會有  $n \times m$  個匹配。接著按照歐式距離的大小進行排序，並



將距離超過閥值的匹配進行剔除，其餘視為配對成功。由於兩組特徵點匹配中可能具有相同的配對關係，因此需要剔除重複的部份，即  $\{p_i, q_i\}_{i=j} = \{p_j, q_j\}_{j=i}$ 。最後將剩餘部份統整，可得出特徵點匹配  $\{p_i, q_i\}_{i=1, \dots, N}$ ，可用於下一節的相對變換關係計算。本研究選擇窮舉特徵匹配來替代現有地圖合併演算法 Ferrão [4] 使用的唯一特徵匹配，是因為在重複特徵環境中，特徵的描述皆十分相似，唯一匹配會造成正確的匹配被剔除。

### 5.4.3 特徵修正變換關係

前幾節中，已透過通信強度預測距離得出一個相對關係模型  $\mathbf{T}^{12,best}$ ，並根據最後的評斷指標判斷是否要再進行修正，若需修正，則透過前一節取得兩地圖影像的特徵點配對  $\{p_i, q_i\}_{i=1,2,\dots,N}$  並利用本小節的修改之 RANSAC 進行修正，其中  $p_i$ 、 $q_i$  分別表示  $\mathbf{M}_1'$ 、 $\mathbf{M}_2$  的第  $i$  個已配對特徵。修正方法是使用旋轉角度及平移限制之 RANSAC 演算法，令特徵點配對作為輸入 RANSAC 演算法的資料集合  $\mathcal{X}$ ，即  $\mathcal{X} = \{p_i, q_i\}_{i=1,2,\dots,N}$ ，接著限制旋轉角度及平移，最終計算出兩地圖影像的相對變換關係  $\mathbf{T}_{correct}^{12}$  (式 (3.3))。此相對變換關係  $\mathbf{T}_{correct}^{12}$  含有三個未知變數，即旋轉  $\varphi'$ 、平移  $[t'_x, t'_y]$ ，因此每次需從資料集合  $\mathcal{X}$  中採樣至少 2 組點對 (可建立 4 條方程式)。計算模型的詳細步驟如下：

- 首先根據式 (3.3) 定義猜測模型  $\mathbf{T}_{correct}^{12, guess}$ ：

$$\mathbf{T}_{correct}^{12, guess} = \begin{bmatrix} \cos \varphi' & -\sin \varphi' & t'_x \\ \sin \varphi' & \cos \varphi' & t'_y \\ 0 & 0 & 1 \end{bmatrix} \quad (5.36)$$

接著計算猜測模型的三個參數  $\varphi'$ 、 $t'_x$ 、 $t'_y$ 。假設隨機選取 2 組特徵點對，點對與對應的二維座標為

$$\begin{aligned} &\{p_a : (x_{p_a}, y_{p_a}), q_a : (x_{q_a}, y_{q_a})\} \\ &\{p_b : (x_{p_b}, y_{p_b}), q_b : (x_{q_b}, y_{q_b})\} \end{aligned} \quad (5.37)$$



角度參數  $\varphi'$  的計算為

$$\varphi' = \tan^{-1} \frac{y_{q_b} - y_{q_a}}{x_{q_b} - x_{q_a}} - \tan^{-1} \frac{y_{p_b} - y_{p_a}}{x_{p_b} - x_{p_a}} \quad (5.38)$$

兩個位移參數  $t_x$  與  $t_y$  為

$$\begin{bmatrix} t'_x \\ t'_y \end{bmatrix} = \begin{bmatrix} x_{q_a} \\ y_{q_a} \end{bmatrix} - \begin{bmatrix} \cos \varphi' & -\sin \varphi' \\ \sin \varphi' & \cos \varphi' \end{bmatrix} \begin{bmatrix} x_{p_a} \\ y_{p_a} \end{bmatrix} \quad (5.39)$$

若  $\varphi' > \varphi_{th}$  或是  $[t'_x, t'_y] > [t_{th}, t_{th}]$ ，也就是旋轉角度或位移量大於閥值，則直接重新步驟 1，小於閥值才會進入下一步。

2. 根據猜測模型，測試所有特徵點對是否符合該模型，令第  $i$  個特徵點對及其二維座標為：

$$\{p_i = (x_{p_i}, y_{p_i}), q_i = (x_{q_i}, y_{q_i})\}_{i=1,2,\dots,N} \quad (5.40)$$

根據式 (5.38、5.39)，可將每一點對進行變換：

$$\begin{aligned} \mathbf{v}_{q_i} &= \begin{bmatrix} x_{q_i} \\ y_{q_i} \end{bmatrix} \\ \mathbf{v}'_{p_i} &= \begin{bmatrix} \cos \varphi' & -\sin \varphi' \\ \sin \varphi' & \cos \varphi' \end{bmatrix} \begin{bmatrix} x_{p_i} \\ y_{p_i} \end{bmatrix} + \begin{bmatrix} t'_x \\ t'_y \end{bmatrix} \end{aligned} \quad (5.41)$$

變換後，若符合

$$|\mathbf{v}_{q_i} - \mathbf{v}'_{p_i}| < \epsilon \quad (5.42)$$

則視該特徵點對符合猜測模型，其中  $\epsilon$  為一調整閥值，本研究設為 0.1。

記錄符合模型的特徵點對數量，若數量超過先前記錄的最高數量，則更新目前模型為最佳模型。

3. 重複  $k$  次步驟 1、2，輸出符合模型的特徵點對數量最高的最佳模型  $\mathbf{T}_{correct}^{12,best}$ 。
4. 根據最終輸出最佳模型  $\mathbf{T}_{correct}^{12,best}$  使用式 (2.4) 計算相對變換關係接受指標  $\omega'$ ，

並根據相對變換關係接受指標  $\omega'$  跟符合相對變換接受條件（式 (3.4)），來進行分類及處理，如表 5.3。

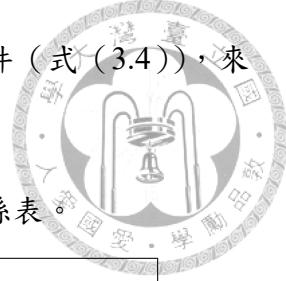


表 5.3: 修正後變換關係接受指標  $\omega$  分類及處理關係表。

$\omega'$ 範圍	分類	處理
$\omega' \geq 0.95$	合併正確	使用 5.5 節的方法輸出合併地圖
$\omega' < 0.95$	合併錯誤	回到 5.1.3 節增加底層牆壁數量，若底層牆壁數量達上限則輸出原地圖

## 5.5 地圖合併處理

在評判相對變換關係中已定義出重疊地圖  $\mathbf{M}_3$ ，重疊地圖  $\mathbf{M}_3$  不只可以用於判斷是否接受相對變換關係，同時也是  $\mathbf{M}_1$ 、 $\mathbf{M}_2$  兩個地圖需進行合併的部份。本研究的合併方法採用熵濾波處理（Entropy Filter）[63]，透過格點佔據地圖計算佔據機率，再透過佔據機率計算對應熵值，再篩選出該重疊格點的最終區域種類。

### 5.5.1 計算佔據機率

格點佔據地圖以機率區分每個格點的區域種類，格點佔據地圖  $\mathbf{M}$  其每個格點  $\mathbf{M}(x, y)$  皆含有兩個佔據機率  $\{p(s = 0), p(s = 1)\}$ ，前者代表該格點屬於自由區域的機率，後者則代表屬於佔據區域的機率，兩者相加為一，如式 (5.43)。

$$p(s = 0) + p(s = 1) = 1 \quad (5.43)$$

根據式 (5.43) 將每個格點的兩個機率透過 (5.44) 化為機率比值  $Odd(s)$  進行表示。

$$Odd(s) = \frac{p(s = 1)}{p(s = 0)} \quad (5.44)$$

當感測器取得格點新的感測資訊時，將機率比值  $Odd(s)$  更新為式 (5.45)。

$$Odd(s|z) = \frac{p(s = 1|z)}{p(s = 0|z)} \quad (5.45)$$



其中  $z$  表示新的感測資訊。根據貝氏定理， $p(s = 0|z)$  與  $p(s = 1|z)$  可等同於：

$$p(s = 1|z) = \frac{p(z|s = 1)p(s = 1)}{p(z)}$$

$$p(s = 0|z) = \frac{p(z|s = 0)p(s = 0)}{p(z)}$$
(5.46)

根據式 (5.46)，式 (5.45) 可重新改寫為

$$\begin{aligned} Odd(s|z) &= \frac{p(s = 1|z)}{p(s = 0|z)} = \frac{\frac{p(z|s=1)p(s=1)}{p(z)}}{\frac{p(z|s=0)p(s=0)}{p(z)}} \\ &= \frac{p(z|s = 1)}{p(z|s = 0)} Odd(s) \end{aligned} \quad (5.47)$$

將式 (5.47) 等號兩邊取對數  $\log$  後，得

$$\log Odd(s|z) = \underbrace{\log \frac{p(z|s = 1)}{p(z|s = 0)}}_{\text{lomeas}} + \log Odd(s) \quad (5.48)$$

從式 (5.48) 可看出，概率比值的更新只需將原本的  $\log Odd(s)$  與  $\log \frac{p(z|s=1)}{p(z|s=0)}$  相加。在格點佔據地圖中， $\log \frac{p(z|s=1)}{p(z|s=0)}$  為一測量模型，稱為測量值 (lomeas)。

測量值是一個固定值，具有兩個數值，即 lofree 和 looccu，分別表示當接收格點的感測資訊為自由區域與佔據區域的情況，可定義為：

$$\begin{aligned} \text{lofree} &= \log \frac{p(z = 0|s = 1)}{p(z = 0|s = 0)} \\ \text{looccu} &= \log \frac{p(z = 1|s = 1)}{p(z = 1|s = 0)} \end{aligned} \quad (5.49)$$

因此，只需要設定感測器的測量值 (Lomeas) 中的 lofree 和 looccu 數值，就可以快速更新機率比值  $Odd(s|z)$  (式 (5.48))。更新後，可以根據式 (5.43) 和式 (5.44) 將其轉換為佔據機率  $p(s)$ ，然後根據格點區域閾值輸出格點的區域種類。式 (5.50) 中， $p_{occupied}$  表示佔據閾值 (Occupied Threshold)， $p_{free}$  表示自由閾值 (Free Threshold)，根據兩個閾值可將格點區分為佔據 (Occupied)、自由 (Free)、

未知 (Unknown) 區域，以像素值表示則分別為 0、255、128 三個整數值。

$$\mathbf{M}(x, y) = \begin{cases} 0 & (\text{occupied}) & , \text{if } p(s=1) > p_{\text{occupied}} \\ 255 & (\text{free}) & , \text{if } p(s=1) < p_{\text{free}} \\ 128 & (\text{unknown}) & , \text{else} \end{cases} \quad (5.50)$$



當格點佔據地圖  $\mathbf{M}$  作為影像時，每一個格點皆含有「區域種類、像素值、佔據機率」的資訊，表 5.4 建立三者的相互關係。

表 5.4: 格點佔據地圖之格點資訊關係表。

區域種類	像素值	佔據機率
佔據	0	$p(s=1) \in (p_{\text{occupied}}, 1]$
自由	255	$p(s=1) \in [0, p_{\text{free}})$
未知	128	$p(s=1) \in [p_{\text{free}}, p_{\text{occupied}}]$

### 5.5.2 熵濾波合併地圖

地圖合併時，熵濾波處理考量的是機率而非像素值代表的區域種類，因此必須改為使用佔據機率值。重疊地圖  $\mathbf{M}_3$  的格點皆可取得來自地圖  $\mathbf{M}_1$ 、 $\mathbf{M}_2$  的 2 個計算出之佔據機率，先依照上面所提的計算佔據機率方法計算出合併機率，再將原先 2 個機率與合併機率帶入熵濾波進行以下處理：

1. 假設重疊地圖  $\mathbf{M}_3$  中，每個格點對應的兩個佔據機率值：

$$p(s=1)_{\mathbf{M}_3(x,y)} = \{p(s=1)_{\mathbf{M}'_1(x,y)}, p(s=1)_{\mathbf{M}'_2(x,y)}\} \quad (5.51)$$

2. 將佔據機率轉換為機率比值，對數形式表示如下：

$$\begin{aligned} Odd(s=1)_{\mathbf{M}'_1(x,y)} &= \frac{p(s=1)_{\mathbf{M}'_1(x,y)}}{p(s=0)_{\mathbf{M}'_1(x,y)}} = \frac{p(s=1)_{\mathbf{M}'_1(x,y)}}{1 - p(s=1)_{\mathbf{M}'_1(x,y)}} \\ Odd(s=1)_{\mathbf{M}'_2(x,y)} &= \frac{p(s=1)_{\mathbf{M}'_2(x,y)}}{p(s=0)_{\mathbf{M}'_2(x,y)}} = \frac{p(s=1)_{\mathbf{M}'_2(x,y)}}{1 - p(s=1)_{\mathbf{M}'_2(x,y)}} \end{aligned} \quad (5.52)$$

$$\log Odd(s=1)_{\mathbf{M}_3(x,y)} = \{\log Odd(s=1)_{\mathbf{M}'_1(x,y)}, \log Odd(s=1)_{\mathbf{M}'_2(x,y)}\}$$



3. 轉換為對數機率比值  $\log Odd(s = 1)$  後，可直接以相加減合併對數機率比值，因此合併後的對數機率比值為：

$$\log Odd(s = 1)_{\mathbf{M}_{combined}(x,y)} = \log Odd(s = 1)_{\mathbf{M}'_1(x,y)} + \log Odd(s = 1)_{\mathbf{M}_2(x,y)} \quad (5.53)$$

4. 將式 (5.53) 的結果轉換回機率形式，可得整合後的佔據機率為：

$$p(s = 1)_{\mathbf{M}_{combined}(x,y)} = \frac{Odd(s = 1)_{\mathbf{M}_{combined}(x,y)}}{1 + Odd(s = 1)_{\mathbf{M}_{combined}(x,y)}} \quad (5.54)$$

經過上述的流程，對於重疊地圖的每一格點可以得到 3 個機率值：

$$p(s = 1)_{\mathbf{M}_3(x,y)} = \{p(s = 1)_{\mathbf{M}'_1(x,y)}, p(s = 1)_{\mathbf{M}_2(x,y)}, p(s = 1)_{\mathbf{M}_{combined}(x,y)}\} \quad (5.55)$$

熵濾波處理中，熵值與機率的關係定義如式 (5.56)，其中  $X$  為事件隨機變數， $p(X = i)$  為某一情況  $i$  下的機率， $H(X)$  則表示事件對應的熵值。熵值越高，表示事件不確定性越大 [63]。

$$H(X) = - \sum_{i=1}^n p(X = i) \log p(X = i) \quad (5.56)$$

將格點佔據地圖的機率套用至式 (5.56)，事件隨機變數被格點佔據的可能性 ( $X = s$ )，其包含兩種情況：格點為佔據的可能性 ( $s = 1$ ) 與格點為非佔據的可能性 ( $s = 0$ )。並根據式 (5.57) 計算對應熵值。

$$\begin{aligned} H(s) &= -p(s = 1) \log p(s = 1) - p(s = 0) \log p(s = 0) \\ &= -p(s = 1) \log p(s = 1) - (1 - p(s = 1))(\log 1 - p(s = 1)) \end{aligned} \quad (5.57)$$

在格點佔據地圖中，屬於不確定性較低的自由區域與佔據區域，其原先的佔據機率亦分別接近 0 與 1，而不確定性較高的未知區域，原先的佔據機率則接近中間值 0.5。因此，機率在熵濾波的處理中與格點佔據地圖的概念相符合，因此可使用熵濾波進行佔據機率的篩選。

接著，將重疊地圖  $\mathbf{M}_3$  的每一個格點的 3 個佔據機率值（從式 (5.55) 得出），

帶入熵濾波進行篩選，並分別使用式（5.57）計算對應熵值：

(5.58)

$$\begin{aligned}
 H(s)_{\mathbf{M}'_1(x,y)} &= - p(s=1)_{\mathbf{M}'_1(x,y)} \log p(s=1)_{\mathbf{M}'_1(x,y)} \\
 &\quad - p(s=0)_{\mathbf{M}'_1(x,y)} \log p(s=0)_{\mathbf{M}'_1(x,y)} \\
 &= - p(s=1)_{\mathbf{M}'_1(x,y)} \log p(s=1)_{\mathbf{M}'_1(x,y)} \\
 &\quad - (1 - p(s=1)_{\mathbf{M}'_1(x,y)}) \log (1 - p(s=1)_{\mathbf{M}'_1(x,y)}) \\
 H(s)_{\mathbf{M}_2(x,y)} &= - p(s=1)_{\mathbf{M}_2(x,y)} \log p(s=1)_{\mathbf{M}_2(x,y)} \\
 &\quad - p(s=0)_{\mathbf{M}_2(x,y)} \log p(s=0)_{\mathbf{M}_2(x,y)} \\
 &= - p(s=1)_{\mathbf{M}_2(x,y)} \log p(s=1)_{\mathbf{M}_2(x,y)} \\
 &\quad - (1 - p(s=1)_{\mathbf{M}_2(x,y)}) \log (1 - p(s=1)_{\mathbf{M}_2(x,y)}) \\
 H(s)_{\mathbf{M}_{combined}(x,y)} &= - p(s=1)_{\mathbf{M}_{combined}(x,y)} \log p(s=1)_{\mathbf{M}_{combined}(x,y)} \\
 &\quad - p(s=0)_{\mathbf{M}_{combined}(x,y)} \log p(s=0)_{\mathbf{M}_{combined}(x,y)} \\
 &= - p(s=1)_{\mathbf{M}_{combined}(x,y)} \log p(s=1)_{\mathbf{M}_{combined}(x,y)} \\
 &\quad - (1 - p(s=1)_{\mathbf{M}_{combined}(x,y)}) \log (1 - p(s=1)_{\mathbf{M}_{combined}(x,y)})
 \end{aligned}$$

根據式（5.58）計算完成熵值後，選擇對應熵值較小的機率作為重疊格點的佔據機率，並將此機率根據佔據、自由閥值輸出重疊格點  $\mathbf{M}_3(x,y)$  的區域種類（式（5.50）），進而完成地圖合併。

## 5.6 本研究地圖合併演算法之虛擬碼

下方為本研究地圖合併演算法之虛擬碼，透過該流程來輸出合併地圖：




---

**Algorithm 7** 本研究出之地圖合併演算法

---

```

1: Input : Signal Strength Dataset  $SS$ , Local Maps  $\mathbf{M}_1, \mathbf{M}_2$ , Pose Dataset  $P_{\mathbf{M}_1}, P_{\mathbf{M}_2}$ ;
2:  $SS \leftarrow RemoveNoise(SS)$ ;            $\triangleright$  將通信強度  $SS$  去除雜訊，詳細見 5.1.2 節
3:  $BaseN_w \leftarrow 0$ ;                    $\triangleright$  信號強度底層之牆壁數
4: while  $BaseN_w < 3$  do
5:    $\varphi' \leftarrow 0$ ;
6:    $t'_x \leftarrow 0$ ;
7:    $t'_y \leftarrow 0$ ;
8:    $d_m \leftarrow PredictiveDistanceModel(SS, BaseN_w)$ ;  $\triangleright$  預測兩機器人相對距離，  
詳細見 5.1.3 節
9:    $[\varphi, t_x, t_y] \leftarrow KinematicAnalysis(\mathbf{M}_1, \mathbf{M}_2, d_m, P_{\mathbf{M}_1}, P_{\mathbf{M}_2})$ ;
10:   $\mathbf{M}'_1 \leftarrow MTM(\varphi, t_x, t_y) \times \mathbf{M}_1$ ;
11:   $\omega \leftarrow \omega(\mathbf{M}'_1, \mathbf{M}_2)$ ;
12:  if  $\omega \geq 0.95$  then
13:    break;
14:  else                                 $\triangleright$  修正  $MTM$ 
15:     $P \leftarrow detectORBFeatures(\mathbf{M}'_1)$ ;       $\triangleright$  提取  $\mathbf{M}'_1$  特徵
16:     $Q \leftarrow detectORBFeatures(\mathbf{M}_2)$ ;       $\triangleright$  提取  $\mathbf{M}_2$  特徵
17:     $X \leftarrow matchFeatures(P, Q)$ ;           $\triangleright$  匹配特徵  $P, Q$ 
18:     $[\varphi', t'_x, t'_y] \leftarrow ConstrainRANSAC(X, \varphi_{th}, t_{th})$ ;  $\triangleright$  計算相對關係
19:     $\mathbf{M}''_1 \leftarrow MTM(\varphi', t'_x, t'_y) \times \mathbf{M}'_1$ ;
20:     $\omega \leftarrow \omega(\mathbf{M}''_1, \mathbf{M}_2)$ ;
21:    if  $\omega \geq 0.95$  then
22:      break;
23:    else
24:       $BaseN_w \leftarrow BaseN_w + 1$ ;
25:    end if
26:  end if
27: end while
28: if  $BaseN_w < 3$  then
29:    $MergeMap \leftarrow EntropyFilterMerge(\mathbf{M}_1, \mathbf{M}_2, MTM(\varphi + \varphi', t_x + t'_x, t_y + t'_y))$ ;
30:   Output :  $MergeMap$ ;
31: else
32:   Output : Local Maps  $\mathbf{M}_1, \mathbf{M}_2$ ;
33: end if

```

---



## 模擬環境測試

為了測試所提出的方法，本研究在兩個不同的特徵的環境中運行了模擬，分別為低重複特徵環境（如圖 6.1(a) 所示）和重複特徵環境（如圖 6.1(b) 所示），兩者面積相同，大約為  $600\text{ m}^2$ 。目標是評估所提出方法在各種環境中的有效性。

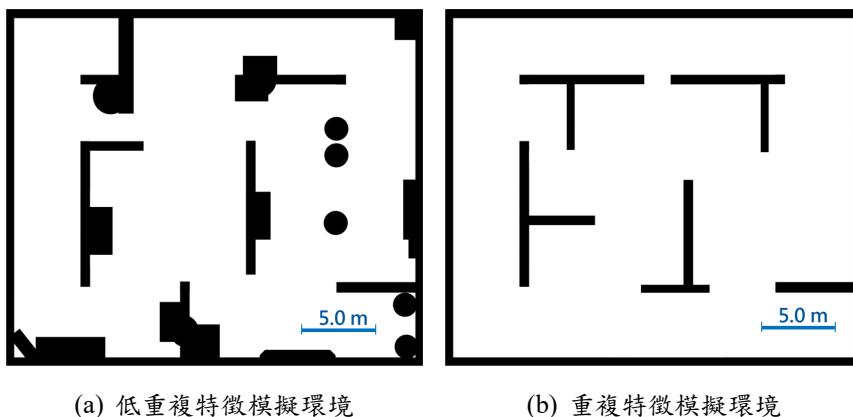


圖 6.1: 本文研究測試用地圖。

### 6.1 模擬設定

本研究使用 Matlab 進行模擬，其提供機器人運動、物理引擎、傳感器數據生成和路徑規劃等功能。Lidar 掃描範圍設置為 25 公尺，機器人為差速機器人，其半徑為 0.1 公尺。為了評估本研究提出的地圖合併方法在具有不同分辨率和環境的地圖上的有效性，本研究進行了各種案例之測試，包括四種構建的局部地圖組合案例。為了評估本研究提出的地圖合併方法和多機器人自主探索方法在具有



不同解析度和環境的地圖上的有效性，本研究進行了四種案例之測試，詳細的參數將條列於表6.1。並包括由四種案例構建之局部地圖的組合（圖 6.2），圖 6.2(a)～6.2(b) 皆於低重複特徵模擬環境並使用圖 6.3(a) 中的路徑進行建構，而圖 6.2(c)～6.2(d) 則皆於重複特徵模擬環境並使用圖 6.3(b) 中的路徑進行建構。

並且在本研究中，使用兩個機器人來模擬多機器人自主探索。在參考之多機器人自主探索方法的模擬中，兩台機器人的初始相對距離限制在 1 公尺左右，以確保了解初始相對姿態。在所提出的多機器人自主探索方法，兩台機器人的初始相對距離雖然沒有限制，但本研究模擬設定兩台機器人初始相對距離必須超過 20 公尺，以確保使用所提出的方法的必要性。

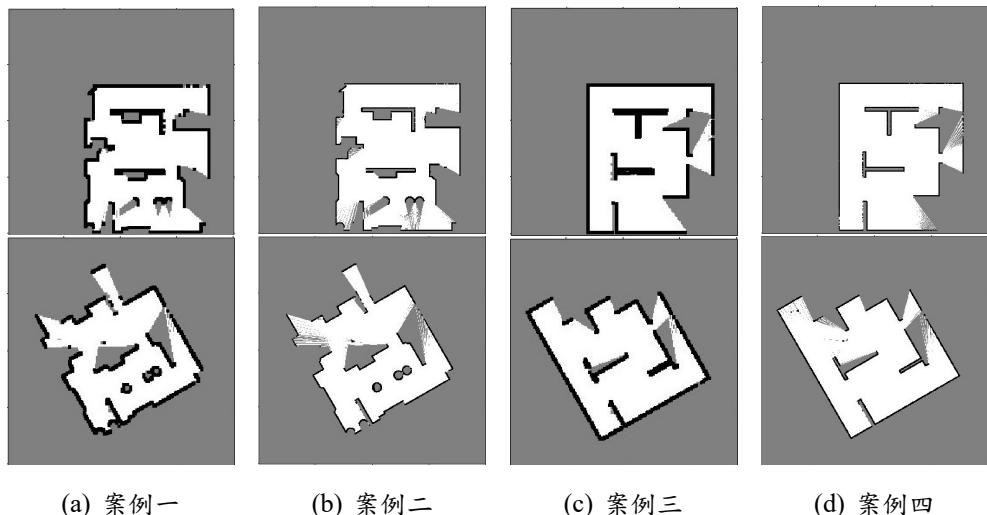


圖 6.2: 每個案例構建的局部地圖。

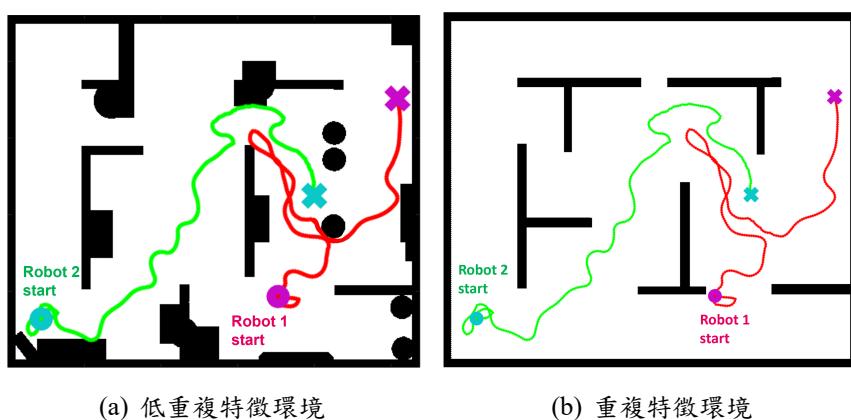


圖 6.3: 兩個機器人的移動路徑。

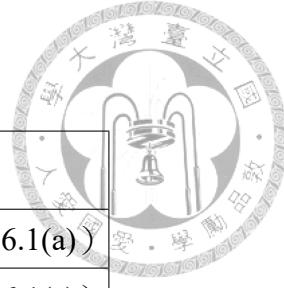


表 6.1: 測試案例之地圖參數。

測試案例	解析度 (比例尺) (格點數/公尺)	環境
案例一	5	低重複特徵環境 (圖 6.1(a))
案例二	20	低重複特徵環境 (圖 6.1(a))
案例三	5	重複特徵環境 (圖 6.1(b))
案例四	20	重複特徵環境 (圖 6.1(b))

## 6.2 現有間接地圖合併演算法之效用

以圖 6.2 的測試案例進行 Ferrão et.al 所提出的特徵地圖合併方法 [4] 測試，以說明現有地圖合併效用。其流程主要是將兩張格點佔據地圖視為影像進行處理，使用 SIFT 演算法進行特徵的擷取，接著使用唯一特徵匹配對擷取的特徵進行匹配，最終使用 RANSAC 演算法計算出 MTM，再使用 MTM 將兩地圖進行合併。

### 6.2.1 現有演算法特徵擷取與匹配測試

首先，本研究針對測試案例使用 SIFT 演算法進行特徵的擷取，其結果如圖 6.4。從圖中可看出，每一張地圖影像皆擷取出數個以綠色圓圈表示的特徵點，圓圈中的線段表示該特徵主方向。

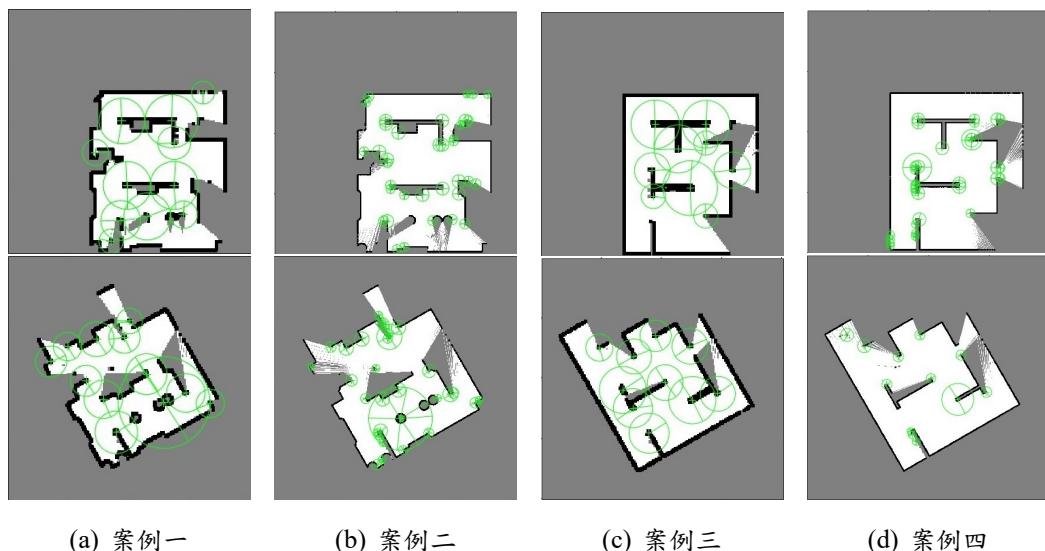


圖 6.4: SIFT 演算法之特徵擷取效用呈現，綠色圓圈表示 SIFT 特徵位置與主方向。

根據圖 6.4 的結果，接著使用地圖的特徵點描述符向量進行特徵唯一匹配，建立特徵點之間的關係。由於格點佔據地圖的像素值差異過少（僅 3 種數值），因此計算描述符向量並匹配時可能產生錯誤配對，特別是對於重複特徵環境，因為其特徵都十分相似，無法使用描述符判別差異，導致錯誤配對，如圖 6.5(c)、6.5(d)，部份配對連線的特徵點明顯來自不同區域（紅框），若以此進行後續 RANSAC 演算法計算相對關係，會導致錯誤甚至無法計算。因此將在下一小節中討論各案例的合併結果。

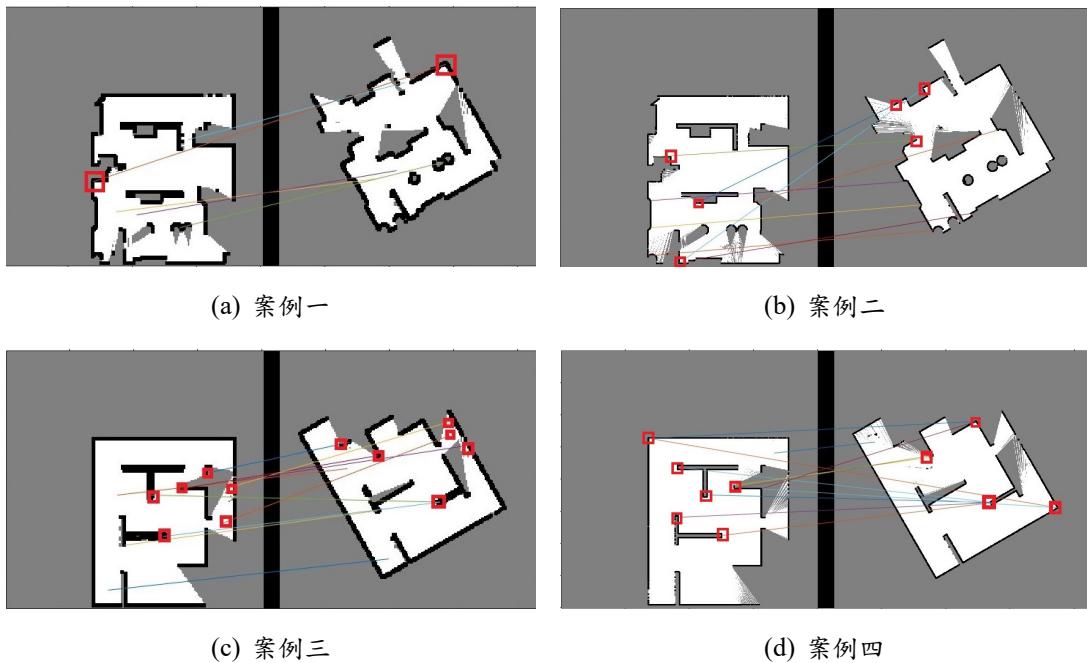


圖 6.5: 特徵匹配示意圖，紅框表示錯誤配對的特徵匹配。

### 6.2.2 現有演算法合併測試

本小節將實測於圖 6.2 的每個案例，每個案例皆具有旋轉、平移差異，因此必須考量 MTM 的三個參數 ( $\theta$ 、 $t_x$ 、 $t_y$ )，使用 RANSAC 演算法計算出 MTM 後進行合併。接著討論各案例的測試合併結果，測試結果不使用熵濾波的輸出圖作為呈現，而是另外使用其他色彩來表示格點合併的正確匹配 (*agr*, 綠色)、錯誤匹配 (*dis*, 紅色) 關係。結果如圖 6.6(a)、6.6(b)，分別為案例一跟二，而案例三及案例四，其結果是 RANSAC 無法找到解，無法合併輸出合併地圖。

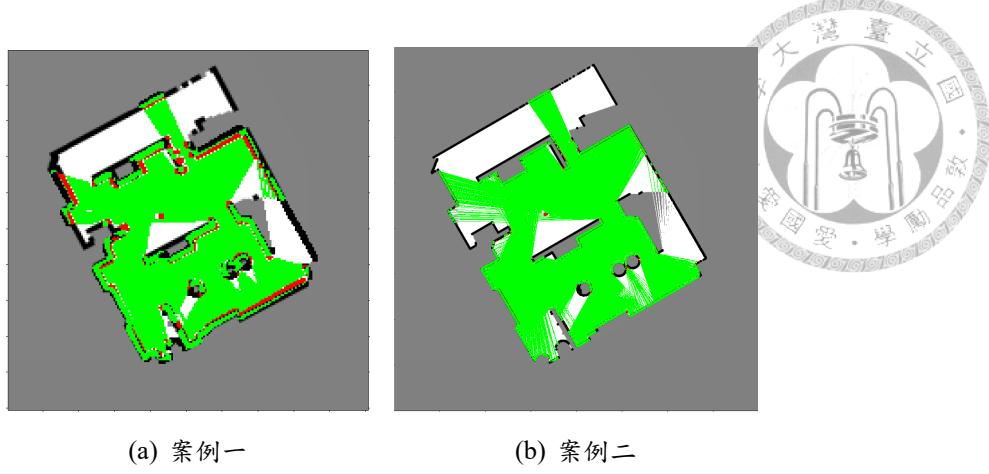


圖 6.6: 現有演算法合併結果可視化示意圖，綠色區域為正確匹配，紅色區域為錯誤匹配。

接著判斷地圖合併的成功與否，以案例一為例，從圖 6.6(a) 以人眼主觀判斷，可看出該合併結果與圖 6.1(a) 的模擬環境大部分相符，但仍有少部分錯誤，也就是少部份線段顯示為紅色，而此情況下的接受指標為 0.9262 (如式 (6.1))，本研究設定合併成功的閥值為 0.95 (原因如 3.2 節所述)，固可確定為合併失敗 (下標 1 表示案例一)。

$$\omega_1 = \frac{agr_1}{agr_1 + dis_1} = 0.9262 < \omega_{thres} = 0.95 \quad (6.1)$$

而案例二如圖 6.6(b)，圖中可見重疊區域為綠色，少部份線段顯示為紅色，表示大部分格點為正確匹配，僅有少量錯誤匹配。若以人眼主觀判斷，可看出該合併結果與圖 6.1(a) 的模擬環境相符，故先以主觀判斷為成功合併，而此情況下的接受指標為 0.9922 (如式 (6.2))，本研究設定合併成功的閥值為 0.95 (原因如 3.2 節所述)，固可確定為合併成功 (下標 2 表示案例二)。

$$\omega_2 = \frac{agr_2}{agr_2 + dis_2} = 0.9922 > \omega_{thres} = 0.95 \quad (6.2)$$

根據現有間接地圖合併方法的測試，本研究認為失敗主要來自於以下幾個原因：

1. 格點佔據地圖的像素值差異過少 (僅 3 種數值)，因此計算描述符向量並匹配時可能產生錯誤配對，特別是對於重複特徵環境，因為其特徵都十分相似，無法使用描述符判別差異，導致錯誤配對，如圖 6.5(a)、6.5(b)，部份配

對連線的特徵點明顯來自不同區域（紅框）。若以此進行匹配，則可能產生錯誤的特徵配對，使得後續計算相對關係錯誤甚至無法計算。



2. 建構的格點佔據地圖若是地圖解析度過低，會導致特徵提取時的位置不準確，使得後續計算相對關係錯誤。

根據上述幾個原因，本研究認為若要單純使用特徵擷取方法來解決地圖合併問題，必須具備具有可辨識特徵的環境及高解析的地圖，才能取得良好的特徵匹配。而對於重複特徵的環境，單純使用特徵擷取方法是無法輕易解決的，勢必要加入額外資訊，而本研究將以資料量小之信號強度作為額外資訊加入特徵擷取方法中，來改善使用直接法的資料量過大的問題，也解決重複特徵環境的問題。下一節將進行本研究提出的地圖合併方法測試，以證明其合併效果。

## 6.3 本研究地圖合併演算法之效用

本節將以圖 6.2 的測試案例進行本研究地圖合併方法的合併測試，並說明演算法效用。

### 6.3.1 預測距離模型測試

兩機器人的信號強度是使用圖 6.3 中兩機器人的路徑所得到，並透過章節 5.1 的預測距離及修正，可以得到預測距離如圖 6.7(a)、6.7(b)，結果大致符合實際真實相對距離，但仍然有誤差存在，誤差部分將仰賴後續演算法進行排除。

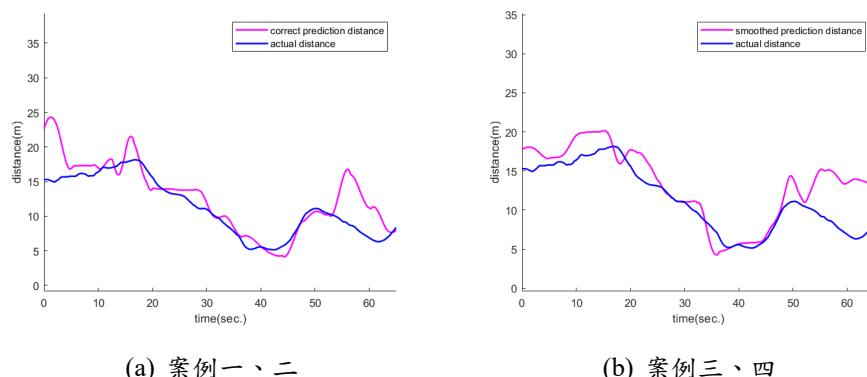


圖 6.7: 預測距離模型所預測之兩機器人相對距離，粉色線為預測距離，藍色線為實際真實距離。



### 6.3.2 運動學分析合併測試

本小節將透過前一小節得到之預測距離，實測於各個案例，每個案例皆具有旋轉、平移差異，因此必須考量 MTM 的三個參數 ( $\varphi$ 、 $t_x$ 、 $t_y$ )。最後也以合併可視化示意圖作為結果呈現，也就是使用其他色彩來表示格點合併的正確匹配 ( $agr$ ，綠色)、錯誤匹配 ( $dis$ ，紅色) 關係。各案例測試結果如下：

1. 案例一之合併結果如圖 6.8(a)，該示意圖與模擬環境（圖 6.1(a)）不相符，此時的接受指標為（下標 1 表示案例一，上標 L 表示透過運動學分析合併）

$$\omega_1^L = \frac{agr_1^L}{agr_1^L + dis_1^L} = 0.8096 < \omega_{thres} = 0.95 \quad (6.3)$$

案例一的結果不符合低重複特徵模擬環境，也低於接受指標的閾值，因此屬於合併失敗，需再進行後續修正。

2. 案例二之合併結果如圖 6.8(b)，該示意圖與模擬環境（圖 6.1(a)）不相符，此時的接受指標為（下標 2 表示案例二，上標 L 表示透過運動學分析合併）

$$\omega_2^L = \frac{agr_2^L}{agr_2^L + dis_2^L} = 0.8948 < \omega_{thres} = 0.95 \quad (6.4)$$

案例二的結果不符合低重複特徵模擬環境，也低於接受指標的閾值，因此屬於合併失敗，需再進行後續修正。

3. 案例三之合併結果如圖 6.8(c)，該示意圖與模擬環境（圖 6.1(b)）不相符，此時的接受指標為（下標 3 表示案例三，上標 L 表示透過運動學分析合併）

$$\omega_3^L = \frac{agr_3^L}{agr_3^L + dis_3^L} = 0.7989 < \omega_{thres} = 0.95 \quad (6.5)$$

案例三的結果不符合重複特徵模擬環境，也低於接受指標的閾值，因此屬於合併失敗，需再進行後續修正。

4. 案例四之合併結果如圖 6.8(d)，該示意圖與模擬環境（圖 6.1(b)）不相符，此時的接受指標為（下標 4 表示案例四，上標 L 表示透過運動學分析合併）

$$\omega_4^L = \frac{agr_4^L}{agr_4^L + dis_4^L} = 0.9099 < \omega_{thres} = 0.95 \quad (6.6)$$



案例四的結果不符合重複特徵模擬環境（圖 6.1(b)），也低於接受指標的閾值，因此屬於合併失敗，需再進行後續修正。

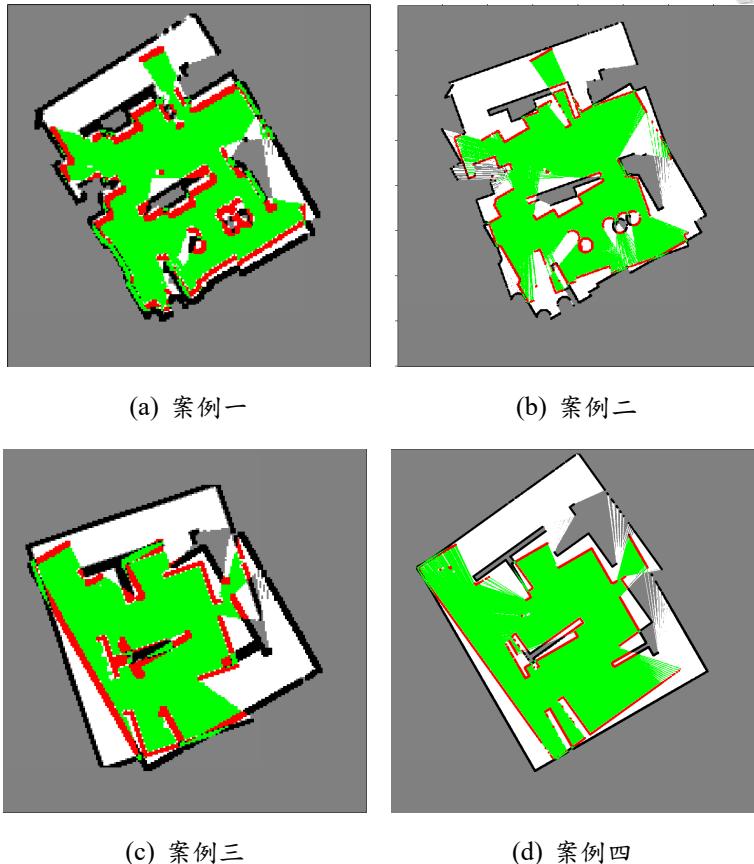


圖 6.8: 運動學分析合併結果可視化示意圖，綠色區域為正確匹配，紅色區域為錯誤匹配。

上述各個案例的測試結果，皆低於接受指標的閾值，因此屬於合併失敗，所以全部都需要再進行後續的特徵修正。

### 6.3.3 修改之特徵地圖合併方法測試

由於各個案例的運動學分析結果皆為需要再進行後續的特徵修正，因此接著使用 Ferrão et.al 所提出的特徵地圖合併方法 [4] 修改版本來修正 MTM，修改內容包括將特徵擷取改為 ORB 演算法、特徵唯一匹配改為窮舉匹配以及將 MTM 的計算限制旋轉角度及平移的限制。



## 特徵擷取與匹配測試

本研究針對測試案例使用 ORB 演算法進行特徵的擷取，其結果如圖 6.9。從圖中可看出，每一張地圖影像皆擷取出數個以綠色圓圈表示的特徵點，圓圈中的線段表示該特徵主方向。

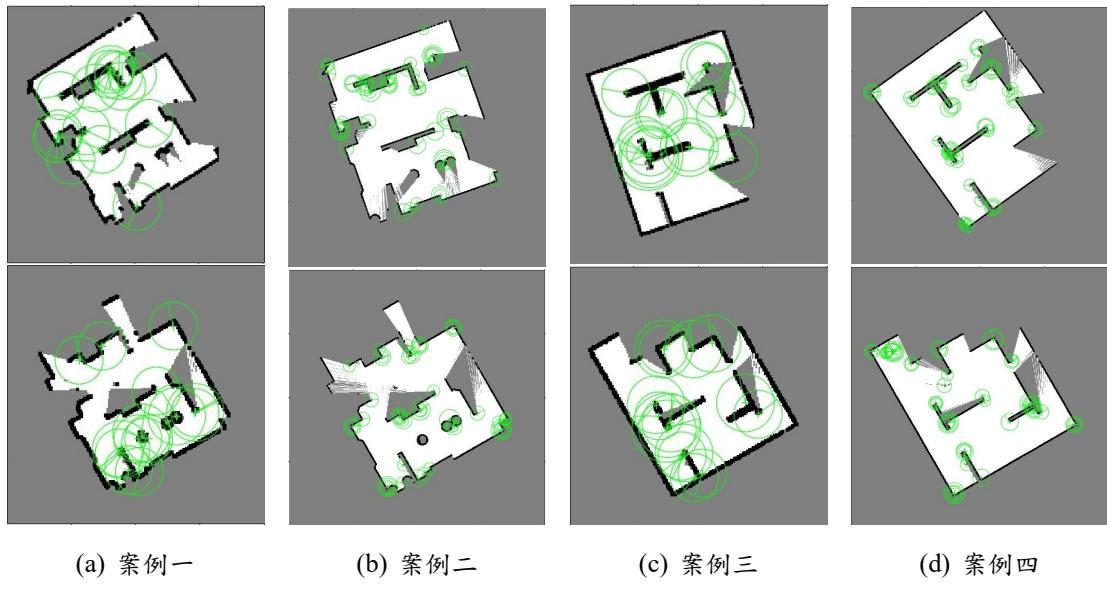


圖 6.9: ORB 演算法之特徵擷取效用呈現，綠色圓圈表示 ORB 特徵位置與主方向。

根據圖 6.9 的結果，接著使用地圖的特徵點描述符向量進行窮舉匹配，建立特徵點之間的關係。匹配過程已在 5.4.2 節說明，錯誤特徵配對則作為離群值，並仰賴 RANSAC 演算法進行排除。

## 特徵修正合併測試

接著使用限制旋轉角度及平移的 RANSAC 演算法再次計算各個案例的 MTM。同樣地，每個案例皆具有旋轉、平移差異，因此必須考量 MTM 的三個參數 ( $\theta$ 、 $t_x$ 、 $t_y$ )。經過前一小節說明的特徵點匹配結果，接著輸入至修改後的 RANSAC 來修正地圖間的相對變換關係，最後也以合併可視化示意圖作為結果呈現，也就是使用其他色彩來表示格點合併的正確匹配 ( $agr$ ，綠色)、錯誤匹配 ( $dis$ ，紅色) 關係。各案例測試結果如下：

1. 案例一之合併結果如圖 6.11(a)，該示意圖與模擬環境（圖 6.1(a)）相符，此

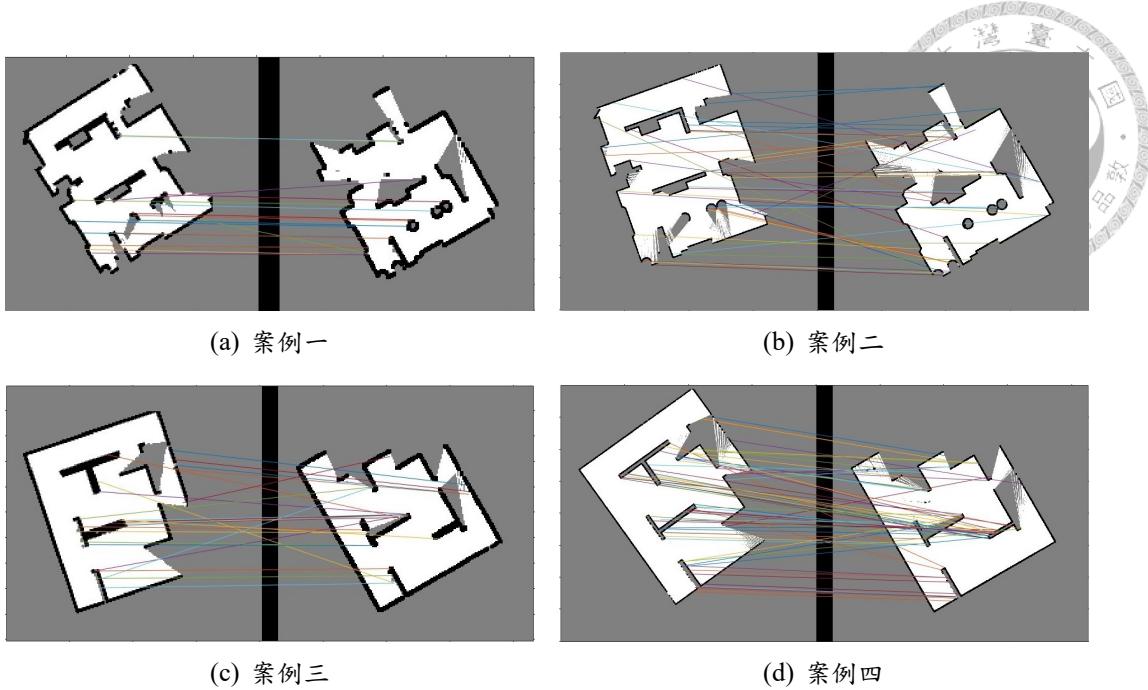


圖 6.10: 特徵窮舉匹配示意圖。

時的接受指標為（下標 1 表示案例一，上標' 表示經過特徵修正處理）

$$\omega'_1 = \frac{agr'_1}{agr'_1 + dis'_1} = 0.9713 > \omega_{thres} = 0.95 \quad (6.7)$$

案例一的結果符合低重複特徵模擬環境，也大於接受指標的閾值，因此屬於成功合併。與現有方法相比（圖 6.6(a)，接受指標  $\omega_1 = 0.9262$ ）提升了 4.869% 的分數，如式 (6.8)。

$$\frac{\omega'_1 - \omega_1}{\omega_1} = \frac{0.9713 - 0.9262}{0.9262} \approx 0.04869 \quad (6.8)$$

2. 案例二之合併結果如圖 6.11(b)，該示意圖與模擬環境（圖 6.1(a)）相符，接受指標分數為（下標 2 表示案例二，上標' 表示經過影像修正處理）

$$\omega'_2 = \frac{agr'_2}{agr'_2 + dis'_2} = 0.9923 > \omega_{thres} = 0.95 \quad (6.9)$$

案例二的結果符合低重複特徵模擬環境，也大於接受指標的閾值，因此屬於成功合併。與現有方法的結果相比（圖 6.6(b)，接受指標  $\omega_2 = 0.9922$ ），我



們提升了 0.01% 的分數，如式 (6.10)。

$$\frac{\omega'_2 - \omega_2}{\omega_2} = \frac{0.9923 - 0.9922}{0.9922} \approx 0.0001 \quad (6.10)$$

3. 案例三之合併結果如圖 6.11(c)，該示意圖與模擬環境（圖 6.1(b)）相符，此時的接受指標為（下標 3 表示案例三，上標' 表示經過特徵修正處理）

$$\omega'_3 = \frac{agr'_3}{agr'_3 + dis'_3} = 0.9736 > \omega_{thres} = 0.95 \quad (6.11)$$

模擬環境與先前兩個案例不相同，為重複特徵環境，相較於現有方法的無法合併，本研究之結果符合成功合併的判斷標準（大於 0.95）。

4. 案例四之合併結果如圖 6.11(d)，該示意圖與模擬環境（圖 6.1(b)）相符。此案例的接受指標則為（下標 4 表示案例四，上標' 表示經過特徵修正處理）

$$\omega'_4 = \frac{agr'_4}{agr'_4 + dis'_4} = 0.9928 > \omega_{thres} = 0.95 \quad (6.12)$$

亦符合正確合併的標準。相較於現有方法的無法合併，本研究之結果符合成功合併的判斷標準（大於 0.95）。

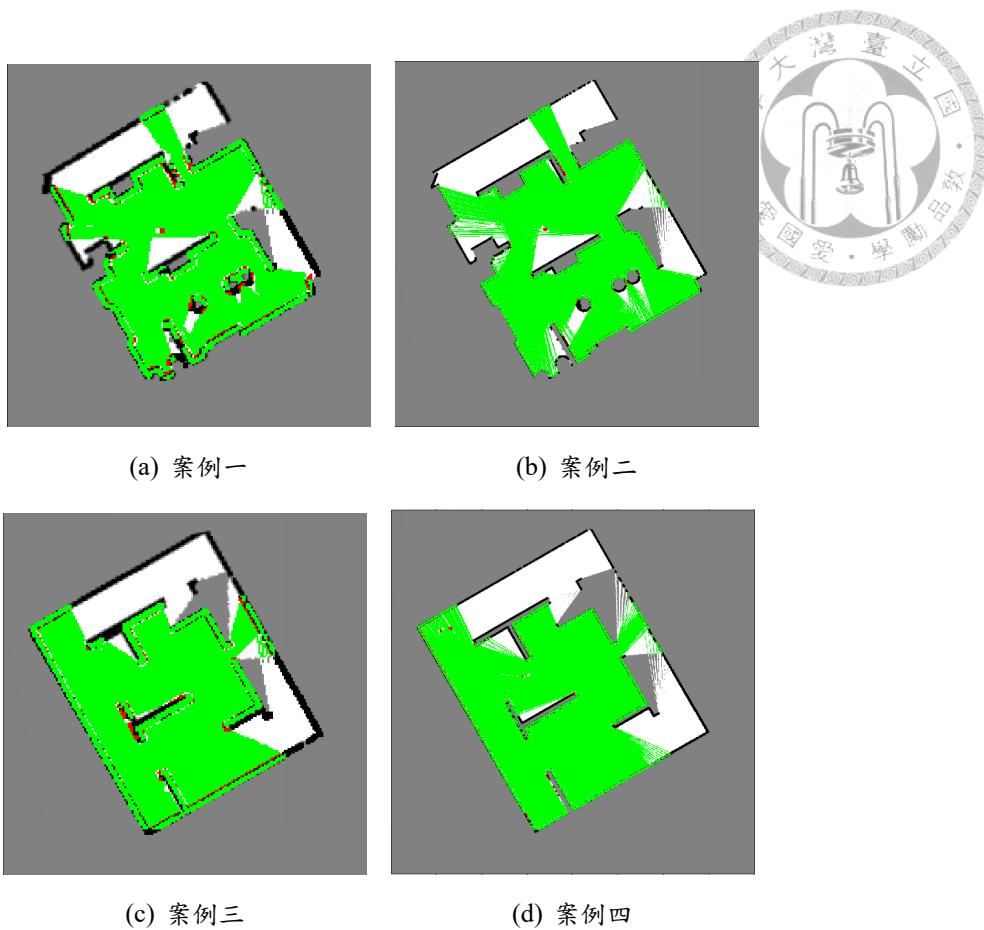


圖 6.11: 特徵修正合併結果可視化示意圖，綠色區域為正確匹配，紅色區域為錯誤匹配。

上述的測試結果，一併整理於表6.2中。經過不同案例的測試，本研究方法不僅可有效的完成格點佔據地圖的合併任務，在合併效果上除了現有方法合併分數就很高的案例三之外，合併分數有顯著的提昇，很明顯在準確性和處理重複特徵環境的能力方面優於現有方法。綜合來看，驗證了所提出的地圖合併方法在不同環境中合併地圖的有效性。

## 6.4 本研究多機器人自主探索測試與討論

本節將對每個案例，分別對參考之單及多機器人自主探索方法 [1, 5]，以及提出之多機器人自主探索執行了 20 次探索運行，在每次探索運行結束時，記錄機器人在探索期間所行走的總距離和花費時間，詳細數據如附錄A。所有探索運行模擬的統計結果如圖 6.12 跟 6.13 所示。



表 6.2: 不同方法對於各個案例的測試結果整理。

	現有方法之測試結果	本文方法之測試結果	接受指標提昇百分比
案例一	接受指標 $\omega_1 = 0.9262$ 圖 6.6(a) (失敗)	接受指標 $\omega'_1 = 0.9713$ 圖 6.11(a) (成功)	+4.87%
案例二	接受指標 $\omega_2 = 0.9922$ 圖 6.6(b) (成功)	接受指標 $\omega'_2 = 0.9923$ 圖 6.11(b) (成功)	+0.01%
案例三	無法合併 (失敗)	接受指標 $\omega'_3 = 0.9736$ 圖 6.11(c) (成功)	-
案例四	無法合併 (失敗)	接受指標 $\omega'_4 = 0.9928$ 圖 6.11(d) (成功)	-

圖 6.12 展示了各機器人自主探索之行走距離結果，顯示出所提出的方法在總行走距離方面略劣於現有的兩種方法。這是因為在本研究中，兩個機器人首先使用單機器人自主探索方法，然後在中期地圖合併之後才開始進行多機器人自主探索，這可能導致某些區域被多次探索，增加了總行走距離。

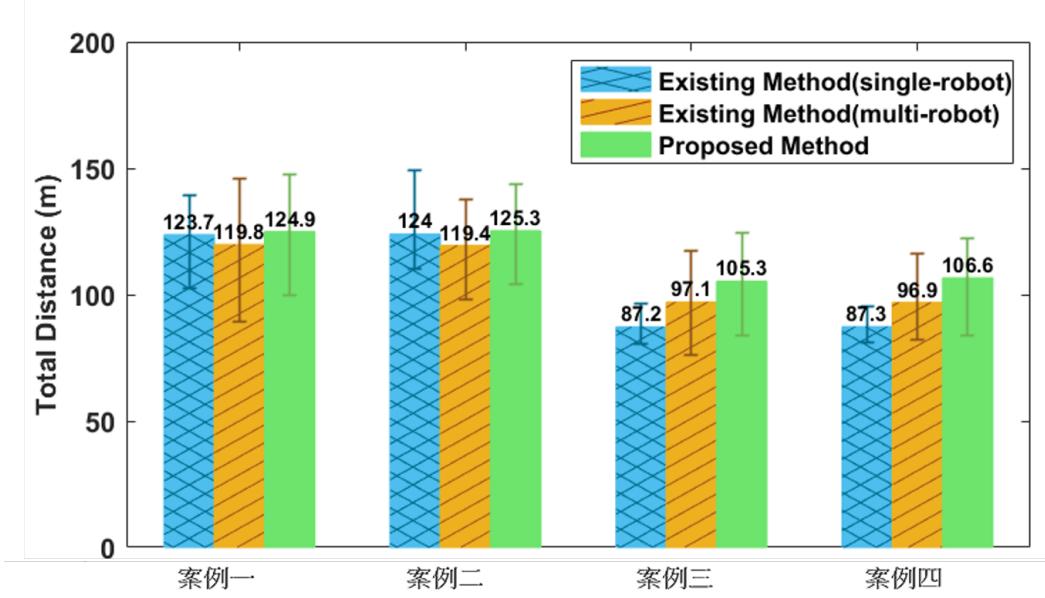


圖 6.12: 各機器人自主探索之統計模擬行走距離結果，藍色為現有單機器人方法之結果 [5]，橘色為現有多機器人方法之結果 [1]，綠色為提出方法之結果。

而圖 6.13展示了各機器人自主探索之花費時間結果，結果表明與現有方法相比，在花費時間上與現有多機器人自主探索相當，具體的結果會受到環境條件的影響，而造成總行走距離增加，但花費時間相差不多之原因在於，現有多機器人自主探索方法使用較為崎嶇的路徑規劃 (RRT)，導致在追蹤路徑點時行走速度較



慢，但整體而言明顯優於現有單機器人自主探索。

這些結果驗證了所提出方法與現有技術相比，可以解除必須已知初始相對姿態之限制，同時以合理的效率完成整個地圖的探索。

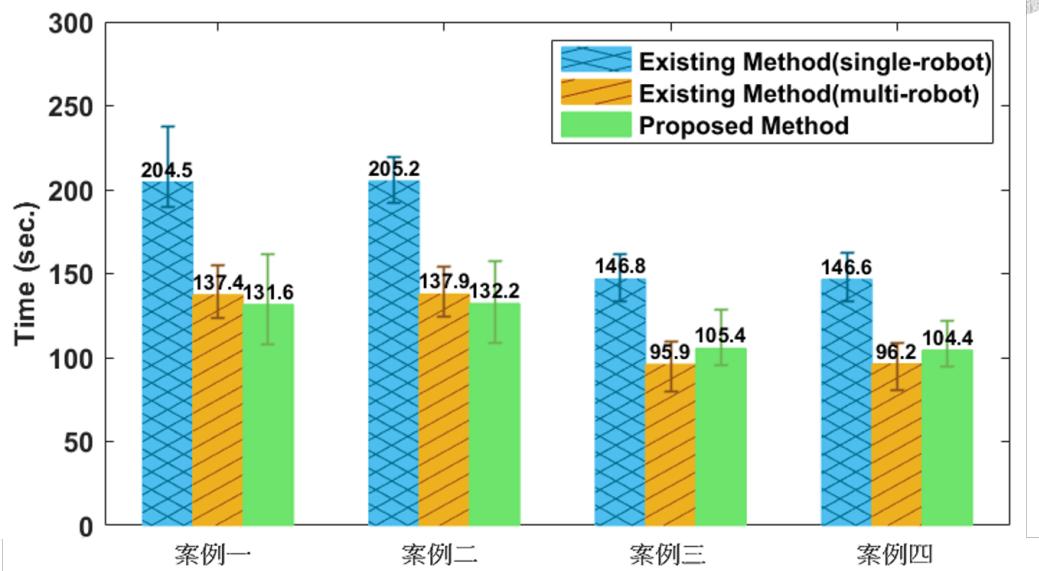


圖 6.13: 多機器人自主探索之統計模擬花費時間結果，藍色為現有單機器人方法之結果 [5]，橘色為現有多機器人方法之結果 [1]，綠色為提出方法之結果。



# 總結與未來展望

在前面的章節中，介紹了多機器人自主探索問題的背景（第一章）以及相關解決方法的回顧及挑戰（第二章），同時本研究也針對該問題進行了詳細的問題定義並呈現研究架構（第三章）。針對本文要解決的多機器人自主探索問題，本研究首先說明除了地圖合併以外所需使用到之演算法，其主要參考自 Umari et al. 所提出之單及多機器人自主探索 [1,5]（第四章）。接著說明本研究提出之地圖合併方法，其主要針對重複特徵環境來進行，同時加入 Ferrão et al. 提出之特徵地圖合併方法 [4] 的修改版本來修正合併結果（第五章）。最後對各種不同環境進行模擬測試，可以發現在同樣的案例下，本研究地圖合併方法可得到更好的結果，並且在本研究多機器人自主探索方法可以以合理的成本探索完環境（第六章）。最後，在本章節中，將對本研究的方法進行了總結，並探討未來持續研究的方向。

## 7.1 結論

本研究提出了一種多機器人自主探索的新方法，其是通過結合本研究所提出的地圖合併技術改進現有多機器人自主探索方法 [1]，使得多機器人自主探索技術可以在沒有初始相對姿態的情況下執行。結果表明，與現有多機器人自主探索方法相比，該策略可以以合理的時間成本成功探索整個地圖。而所提出的地圖合併技術還可以解決重複特徵環境的挑戰，提高地圖合併的準確性，同時減少處理數據的大小，該方法利用通信信號強度來估計機器人之間的相對距離，並通過運動學分析計算 MTM 來合併兩地圖。然而，信號雜訊可能會導致計算出之 MTM 不

準確，因此本研究使用接受指標來檢測錯誤，並在必要時使用 Ferrão et al. 提出之特徵地圖合併方法 [4] 的修改版本來修正 MTM，此修改包括將唯一特徵匹配更改為窮舉特徵匹配，並在計算 MTM 時添加旋轉跟平移約束。



## 7.2 研究建議與未來展望

本研究以不同模擬案例證實了提出方法之可行性，但仍有許多未來值得持續研究與改善的方向，以下將詳列進行說明：

### 1. 實際環境之測試

本研究的各案例皆是模擬環境，雖然模擬可調整各種參數誤差，但實際環境與模擬環境相比，有更多的不確定性需要考慮，例如通信有可能會中斷。因此，仍需進行實際環境的測試，才能確保本研究方法的穩健性。

### 2. 參數自動調整

本研究的地圖合併方法中，涉及了許多參數的調整，例如透過運動學分析計算 MTM 時要選取的時間點數量及距離閥值。不同環境可能使用不同的參數會達到更好的效果，而參數通常需要反覆進行調整才能達到此效果。因此，對於本研究地圖合併方法，建立參數自動調整更能穩健地進行。

### 3. 多機器人自主探索方法之完善

由於本研究多機器人自主探索方法是假設機器人里程計沒有誤差，因此是沒有加入 SLAM 進行自我定位的，但實際情況是里程計有誤差，因此加入 SLAM 於本研究多機器人自主探索方法，更能確保本研究方法的穩健性。



## 附錄 A

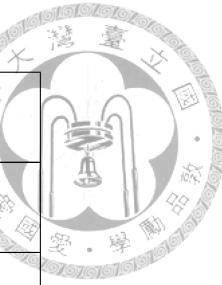
# 機器人自主探索測試數據

本附錄紀錄參考之機器人自主探索方法 [1, 5] 及提出之多機器人自主探索執行了 20 次探索運行的結果，初始姿態  $[x, y, \theta]$  代表其在全域地圖（圖 6.1(a)、6.1(b)）的座標及朝向，座標的單位為公尺。

## A.1 參考之單機器人自主探索方法結果

表 A.1: 參考單機器人自主探索方法之案例一結果（低重複特徵環境、低解析度）

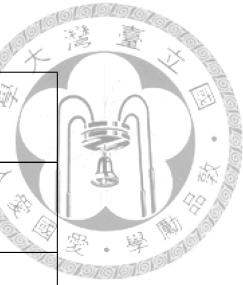
run	robot 初始姿態	總行走距離 (m)	花費時間 (sec.)
1	$[2.2 ; 2 ; \text{pi}/2]$	118.81	195.6
2	$[2 ; 3 ; \text{pi}/4]$	139.34	223.6
3	$[7.2 ; 2 ; \text{pi}/2]$	139.43	238.0
4	$[6 ; 2 ; \text{pi}/4]$	131.69	215.0
5	$[15 ; 2 ; \text{pi}/2]$	118.15	194.7
6	$[14 ; 2 ; \text{pi}/4]$	132.35	217.3



7	$[22.2 ; 2 ; \pi/2]$	125.67	210.0
8	$[23 ; 2 ; \pi/4]$	128.36	203.0
9	$[2.2 ; 12 ; \pi/2]$	130.55	213.6
10	$[2 ; 13 ; \pi/4]$	117.21	191.1
11	$[12.2 ; 12 ; \pi/2]$	102.3	190.1
12	$[12 ; 13 ; \pi/4]$	127.4	203.7
13	$[22.2 ; 12 ; \pi/2]$	117.54	190.7
14	$[23 ; 12 ; \pi/4]$	112.47	193.2
15	$[3.2 ; 21 ; \pi/2]$	122.15	202.8
16	$[4 ; 21 ; \pi/4]$	119.92	200.0
17	$[12.2 ; 22 ; \pi/2]$	125.88	203.6
18	$[12 ; 21 ; \pi/4]$	121.61	202.5
19	$[22.2 ; 22 ; \pi/2]$	122.10	202.3
20	$[23 ; 21 ; \pi/4]$	121.14	200.0

表 A.2: 參考單機器人自主探索方法之案例二結果（低重複特徵環境、高解析度）

run	robot 初始姿態	總行走距離 (m)	花費時間 (sec.)
1	$[2.2 ; 2 ; \pi/2]$	122.15	206.9
2	$[2 ; 3 ; \pi/4]$	125.35	204.8



3	$[7.2 ; 2 ; \pi/2]$	127.52	211.5
4	$[6 ; 2 ; \pi/4]$	148.94	219.4
5	$[15 ; 2 ; \pi/2]$	134.72	215.1
6	$[14 ; 2 ; \pi/4]$	134.72	212.7
7	$[22.2 ; 2 ; \pi/2]$	109.97	191.8
8	$[23 ; 2 ; \pi/4]$	120.48	200.8
9	$[2.2 ; 12 ; \pi/2]$	119.22	200.0
10	$[2 ; 13 ; \pi/4]$	122.52	205.0
11	$[12.2 ; 12 ; \pi/2]$	125.33	206.7
12	$[12 ; 13 ; \pi/4]$	114.84	198.1
13	$[22.2 ; 12 ; \pi/2]$	125.72	206.9
14	$[23 ; 12 ; \pi/4]$	121.30	203.3
15	$[3.2 ; 21 ; \pi/2]$	137.39	218.5
16	$[4 ; 21 ; \pi/4]$	112.13	199.0
17	$[12.2 ; 22 ; \pi/2]$	117.33	198.0
18	$[12 ; 21 ; \pi/4]$	125.07	206.0
19	$[22.2 ; 22 ; \pi/2]$	116.86	198.9
20	$[23 ; 21 ; \pi/4]$	117.85	199.8

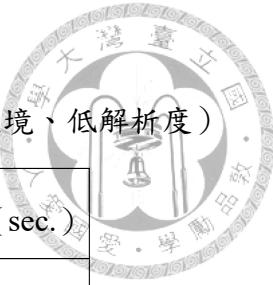
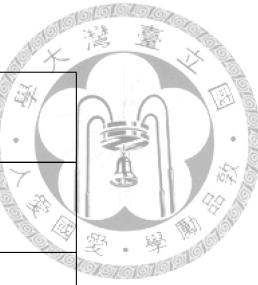


表 A.3: 參考單機器人自主探索方法之案例三結果（重複特徵環境、低解析度）

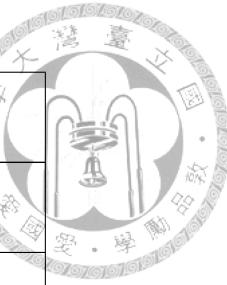
run	robot 初始姿態	總行走距離 (m)	花費時間 (sec.)
1	[2.2 ; 2 ; pi/2]	84.37	136.8
2	[2 ; 3 ; pi/4]	89	150.8
3	[7.2 ; 2 ; pi/2]	95.10	160.8
4	[6 ; 2 ; pi/4]	82.66	140.8
5	[15 ; 2 ; pi/2]	88.06	146.8
6	[14 ; 2 ; pi/4]	81.50	135.7
7	[22.2 ; 2 ; pi/2]	89.44	152.2
8	[23 ; 2 ; pi/4]	86.90	145.3
9	[2.2 ; 12 ; pi/2]	83.85	138.0
10	[2 ; 13 ; pi/4]	83.47	137.3
11	[12.2 ; 12 ; pi/2]	89.50	159.9
12	[12 ; 13 ; pi/4]	90.37	157.8
13	[22.2 ; 12 ; pi/2]	84.01	138.6
14	[23 ; 12 ; pi/4]	87.81	146.6
15	[3.2 ; 21 ; pi/2]	91.74	160.6
16	[4 ; 21 ; pi/4]	96.27	161.2
17	[12.2 ; 22 ; pi/2]	81.66	136.4



18	[12 ; 21 ; pi/4]	89.00	147.2
19	[22.2 ; 22 ; pi/2]	88.79	150.2
20	[23 ; 21 ; pi/4]	80.40	133.2

表 A.4: 參考單機器人自主探索方法之案例四結果（重複特徵環境、高解析度）

run	robot 初始姿態	總行走距離 (m)	花費時間 (sec.)
1	[2.2 ; 2 ; pi/2]	81.89	133.3
2	[2 ; 3 ; pi/4]	95.19	162.0
3	[7.2 ; 2 ; pi/2]	80.81	134.3
4	[6 ; 2 ; pi/4]	84.04	136.4
5	[15 ; 2 ; pi/2]	86.18	145.0
6	[14 ; 2 ; pi/4]	82.04	134.7
7	[22.2 ; 2 ; pi/2]	89.16	160.0
8	[23 ; 2 ; pi/4]	88.19	148.6
9	[2.2 ; 12 ; pi/2]	93.60	157.3
10	[2 ; 13 ; pi/4]	87.75	146.1
11	[12.2 ; 12 ; pi/2]	93.37	158.8
12	[12 ; 13 ; pi/4]	83.61	136.3
13	[22.2 ; 12 ; pi/2]	84.74	139.2

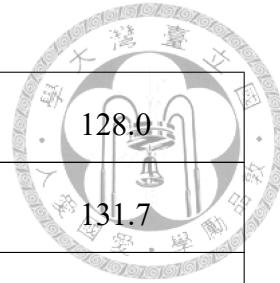


14	[23 ; 12 ; pi/4]	85.03	139.7
15	[3.2 ; 21 ; pi/2]	91.02	153.5
16	[4 ; 21 ; pi/4]	88.66	158.6
17	[12.2 ; 22 ; pi/2]	84.71	138.7
18	[12 ; 21 ; pi/4]	88.73	154.3
19	[22.2 ; 22 ; pi/2]	92.85	158.7
20	[23 ; 21 ; pi/4]	83.90	136.4

## A.2 參考之多機器人自主探索方法結果

表 A.5: 參考多機器人自主探索方法之案例一結果（低重複特徵環境、低解析度）

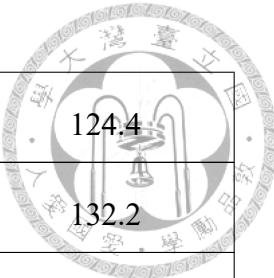
run	robot 1 初始姿態	robot 2 初始姿態	總行走距離 (m)	花費時間 (sec.)
1	[2.2 ; 2 ; pi/2]	[2 ; 3 ; pi/3]	130.55	151.0
2	[2.2 ; 2 ; pi/1.5]	[2 ; 3 ; pi/4]	89.4	123.6
3	[7.2 ; 2 ; pi/2]	[6 ; 2 ; pi/3]	134.85	154.8
4	[7.2 ; 2 ; pi/1.5]	[6 ; 2 ; pi/4]	111	127.6
5	[15 ; 2 ; pi/2]	[14 ; 2 ; pi/3]	121.35	139.2
6	[15 ; 2 ; pi/1.5]	[14 ; 2 ; pi/4]	114.9	130.3
7	[22.2 ; 2 ; pi/2]	[23 ; 2 ; pi/3]	120.15	137.4
8	[22.2 ; 2 ; pi/1.5]	[23 ; 2 ; pi/4]	114	129.1



9	[2.2 ; 12 ; pi/2]	[2 ; 13 ; pi/3]	111	128.0
10	[2.2 ; 12 ; pi/1.5]	[2 ; 13 ; pi/4]	116	131.7
11	[12.2 ; 12 ; pi/2]	[12 ; 13 ; pi/3]	123.15	145.2
12	[12.2 ; 12 ; pi/1.5]	[12 ; 13 ; pi/4]	131.9	152.8
13	[22.2 ; 12 ; pi/2]	[23 ; 12 ; pi/3]	126.15	151.3
14	[22.2 ; 12 ; pi/1.5]	[23 ; 12 ; pi/4]	119.5	132.2
15	[3.2 ; 21 ; pi/2]	[4 ; 21 ; pi/3]	146.15	153.0
16	[3.2 ; 21 ; pi/1.5]	[4 ; 21 ; pi/4]	110.5	125.4
17	[12.2 ; 22 ; pi/2]	[12 ; 21 ; pi/3]	123.9	132.5
18	[12.2 ; 22 ; pi/1.5]	[12 ; 21 ; pi/4]	126	147.4
19	[22.2 ; 22 ; pi/2]	[23 ; 21 ; pi/3]	107.5	125.4
20	[22.2 ; 22 ; pi/1.5]	[23 ; 21 ; pi/4]	117.5	130.0

表 A.6: 參考多機器人自主探索方法之案例二結果（低重複特徵環境、高解析度）

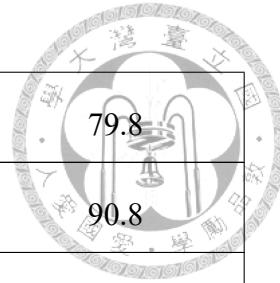
run	robot 1 初始姿態	robot 2 初始姿態	總行走距離 (m)	花費時間 (sec.)
1	[2.2 ; 2 ; pi/2]	[2 ; 3 ; pi/3]	129.9	143.9
2	[2.2 ; 2 ; pi/1.5]	[2 ; 3 ; pi/4]	110.95	131.8
3	[7.2 ; 2 ; pi/2]	[6 ; 2 ; pi/3]	109.95	130.7
4	[7.2 ; 2 ; pi/1.5]	[6 ; 2 ; pi/4]	110	130.3



5	[15 ; 2 ; pi/2]	[14 ; 2 ; pi/3]	98.35	124.4
6	[15 ; 2 ; pi/1.5]	[14 ; 2 ; pi/4]	113.55	132.2
7	[22.2 ; 2 ; pi/2]	[23 ; 2 ; pi/3]	124.2	146.3
8	[22.2 ; 2 ; pi/1.5]	[23 ; 2 ; pi/4]	119.7	137.4
9	[2.2 ; 12 ; pi/2]	[2 ; 13 ; pi/3]	122.7	141.4
10	[2.2 ; 12 ; pi/1.5]	[2 ; 13 ; pi/4]	106.35	126.2
11	[12.2 ; 12 ; pi/2]	[12 ; 13 ; pi/3]	123.3	137.9
12	[12.2 ; 12 ; pi/1.5]	[12 ; 13 ; pi/4]	117	132.7
13	[22.2 ; 12 ; pi/2]	[23 ; 12 ; pi/3]	133.5	154.1
14	[22.2 ; 12 ; pi/1.5]	[23 ; 12 ; pi/4]	137.7	152.5
15	[3.2 ; 21 ; pi/2]	[4 ; 21 ; pi/3]	111.55	132.0
16	[3.2 ; 21 ; pi/1.5]	[4 ; 21 ; pi/4]	119.3	135.1
17	[12.2 ; 22 ; pi/2]	[12 ; 21 ; pi/3]	131.5	149.3
18	[12.2 ; 22 ; pi/1.5]	[12 ; 21 ; pi/4]	122.7	138.5
19	[22.2 ; 22 ; pi/2]	[23 ; 21 ; pi/3]	131.4	147.5
20	[22.2 ; 22 ; pi/1.5]	[23 ; 21 ; pi/4]	115.2	132.9

表 A.7: 參考多機器人自主探索方法之案例三結果（重複特徵環境、低解析度）

run	robot 1 初始姿態	robot 2 初始姿態	總行走距離 (m)	花費時間 (sec.)



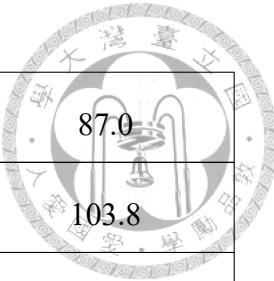
1	[2.2 ; 2 ; pi/2]	[2 ; 3 ; pi/3]	76.2	79.8
2	[2.2 ; 2 ; pi/1.5]	[2 ; 3 ; pi/4]	93.45	90.8
3	[7.2 ; 2 ; pi/2]	[6 ; 2 ; pi/3]	94.25	99.0
4	[7.2 ; 2 ; pi/1.5]	[6 ; 2 ; pi/4]	108.05	105.6
5	[15 ; 2 ; pi/2]	[14 ; 2 ; pi/3]	104.7	106.9
6	[15 ; 2 ; pi/1.5]	[14 ; 2 ; pi/4]	101.15	100.0
7	[22.2 ; 2 ; pi/2]	[23 ; 2 ; pi/3]	117.3	108.8
8	[22.2 ; 2 ; pi/1.5]	[23 ; 2 ; pi/4]	111.6	109.0
9	[2.2 ; 12 ; pi/2]	[2 ; 13 ; pi/3]	92.45	94.5
10	[2.2 ; 12 ; pi/1.5]	[2 ; 13 ; pi/4]	93.05	98.1
11	[12.2 ; 12 ; pi/2]	[12 ; 13 ; pi/3]	92.25	82.7
12	[12.2 ; 12 ; pi/1.5]	[12 ; 13 ; pi/4]	92.25	87.5
13	[22.2 ; 12 ; pi/2]	[23 ; 12 ; pi/3]	110.7	108.6
14	[22.2 ; 12 ; pi/1.5]	[23 ; 12 ; pi/4]	78.45	80.3
15	[3.2 ; 21 ; pi/2]	[4 ; 21 ; pi/3]	78.6	82.6
16	[3.2 ; 21 ; pi/1.5]	[4 ; 21 ; pi/4]	117.45	109.6
17	[12.2 ; 22 ; pi/2]	[12 ; 21 ; pi/3]	91.5	81.5
18	[12.2 ; 22 ; pi/1.5]	[12 ; 21 ; pi/4]	78.45	81.5
19	[22.2 ; 22 ; pi/2]	[23 ; 21 ; pi/3]	103.8	104.7



20	[22.2 ; 22 ; pi/1.5]	[23 ; 21 ; pi/4]	105.45	107.4
----	----------------------	------------------	--------	-------

表 A.8: 參考多機器人自主探索方法之案例四結果（重複特徵環境、高解析度）

run	robot 1 初始姿態	robot 2 初始姿態	總行走距離 (m)	花費時間 (sec.)
1	[2.2 ; 2 ; pi/2]	[2 ; 3 ; pi/3]	108	103.7
2	[2.2 ; 2 ; pi/1.5]	[2 ; 3 ; pi/4]	110.1	105.7
3	[7.2 ; 2 ; pi/2]	[6 ; 2 ; pi/3]	83.25	83.9
4	[7.2 ; 2 ; pi/1.5]	[6 ; 2 ; pi/4]	90.25	95.4
5	[15 ; 2 ; pi/2]	[14 ; 2 ; pi/3]	96.25	98.3
6	[15 ; 2 ; pi/1.5]	[14 ; 2 ; pi/4]	89.25	93.4
7	[22.2 ; 2 ; pi/2]	[23 ; 2 ; pi/3]	86.6	86.7
8	[22.2 ; 2 ; pi/1.5]	[23 ; 2 ; pi/4]	111	108.7
9	[2.2 ; 12 ; pi/2]	[2 ; 13 ; pi/3]	101.45	102.4
10	[2.2 ; 12 ; pi/1.5]	[2 ; 13 ; pi/4]	88.5	98.1
11	[12.2 ; 12 ; pi/2]	[12 ; 13 ; pi/3]	82.35	80.6
12	[12.2 ; 12 ; pi/1.5]	[12 ; 13 ; pi/4]	110.1	105.6
13	[22.2 ; 12 ; pi/2]	[23 ; 12 ; pi/3]	82.6	81.0
14	[22.2 ; 12 ; pi/1.5]	[23 ; 12 ; pi/4]	104.7	99.1
15	[3.2 ; 21 ; pi/2]	[4 ; 21 ; pi/3]	116.1	107.2

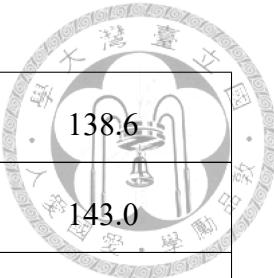


16	[3.2 ; 21 ; pi/1.5]	[4 ; 21 ; pi/4]	87.45	87.0
17	[12.2 ; 22 ; pi/2]	[12 ; 21 ; pi/3]	110	103.8
18	[12.2 ; 22 ; pi/1.5]	[12 ; 21 ; pi/4]	109.1	105.4
19	[22.2 ; 22 ; pi/2]	[23 ; 21 ; pi/3]	88.35	92.9
20	[22.2 ; 22 ; pi/1.5]	[23 ; 21 ; pi/4]	82.5	86.2

### A.3 提出之多機器人自主探索方法結果

表 A.9: 提出方法之案例一結果（低重複特徵環境、低解析度）

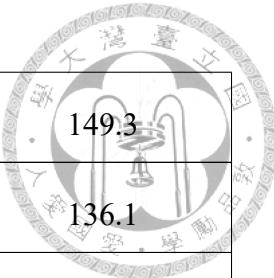
run	robot 1 初始姿態	robot 2 初始姿態	總行走距離 (m)	花費時間 (sec.)
1	[2.2 ; 2 ; pi/2]	[22 ; 22 ; pi/2.2]	107.2	112.0
2	[2.2 ; 2 ; pi/1.5]	[22 ; 22 ; pi/4]	125.4	125.6
3	[2.2 ; 2 ; pi]	[22 ; 22 ; pi/1.5]	99.75	108.0
4	[2.2 ; 2 ; pi/0.5]	[22 ; 22 ; pi]	119.9	120.4
5	[9.2 ; 4 ; pi/2]	[12 ; 22 ; pi/2.2]	125.7	132.2
6	[9.2 ; 4 ; pi/1.5]	[12 ; 22 ; pi/4]	123.9	124.7
7	[9.2 ; 4 ; pi/3]	[12 ; 22 ; pi/0.5]	126.8	136.7
8	[14.2 ; 2 ; pi/2]	[9 ; 22 ; pi/2.2]	147.45	161.8
9	[14.2 ; 2 ; pi/1.5]	[9 ; 22 ; pi/4]	124.1	122.4
10	[14.2 ; 2 ; pi/3]	[9 ; 22 ; pi/0.5]	126	137.4



11	[2 ; 10.2 ; pi/2]	[22 ; 12 ; pi/2.2]	127	138.6
12	[2 ; 10.2 ; pi/1.5]	[22 ; 12 ; pi/4]	128.52	143.0
13	[2 ; 10.2 ; pi/3]	[22 ; 12 ; pi/0.5]	137.7	147.9
14	[2 ; 14.2 ; pi/2]	[22 ; 8 ; pi/2.2]	136.3	156.1
15	[2 ; 14.2 ; pi/1.5]	[22 ; 8 ; pi/4]	135.45	146.7
16	[2 ; 14.2 ; pi/3]	[22 ; 8 ; pi/0.5]	133.05	149.3
17	[22 ; 22 ; pi/2]	[2.2 ; 2.4 ; pi/2.2]	116.45	117.6
18	[22 ; 22 ; pi/1.5]	[2.2 ; 2.4 ; pi/4]	119.3	114.4
19	[22 ; 22 ; pi]	[2.2 ; 2.4 ; pi/1.5]	114.7	114.0
20	[22 ; 22 ; pi/0.5]	[2.2 ; 2.4 ; pi]	123.75	122.6

表 A.10: 提出方法之案例二結果（低重複特徵環境、高解析度）

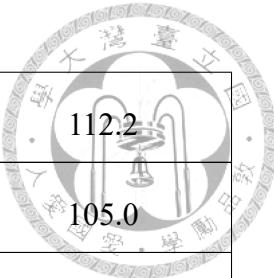
run	robot 1 初始姿態	robot 2 初始姿態	總行走距離 (m)	花費時間 (sec.)
1	[2.2 ; 2 ; pi/2]	[22 ; 22 ; pi/2.2]	114.1	122.0
2	[2.2 ; 2 ; pi/1.5]	[22 ; 22 ; pi/4]	115.4	121.1
3	[2.2 ; 2 ; pi]	[22 ; 22 ; pi/1.5]	125.5	125.1
4	[2.2 ; 2 ; pi/0.5]	[22 ; 22 ; pi]	107.05	113.9
5	[9.2 ; 4 ; pi/2]	[12 ; 22 ; pi/2.2]	143.72	157.5
6	[9.2 ; 4 ; pi/1.5]	[12 ; 22 ; pi/4]	136.6	147.2



7	[9.2 ; 4 ; pi/3]	[12 ; 22 ; pi/0.5]	139.5	149.3
8	[14.2 ; 2 ; pi/2]	[9 ; 22 ; pi/2.2]	126.4	136.1
9	[14.2 ; 2 ; pi/1.5]	[9 ; 22 ; pi/4]	129.5	135.1
10	[14.2 ; 2 ; pi/3]	[9 ; 22 ; pi/0.5]	123.35	125.0
11	[2 ; 10.2 ; pi/2]	[22 ; 12 ; pi/2.2]	140.3	151.4
12	[2 ; 10.2 ; pi/1.5]	[22 ; 12 ; pi/4]	116.5	122.6
13	[2 ; 10.2 ; pi/3]	[22 ; 12 ; pi/0.5]	133.15	145.3
14	[2 ; 14.2 ; pi/2]	[22 ; 8 ; pi/2.2]	134.72	142.7
15	[2 ; 14.2 ; pi/1.5]	[22 ; 8 ; pi/4]	130.9	139.9
16	[2 ; 14.2 ; pi/3]	[22 ; 8 ; pi/0.5]	128.55	136.1
17	[22 ; 22 ; pi/2]	[2.2 ; 2.4 ; pi/2.2]	104.25	117.1
18	[22 ; 22 ; pi/1.5]	[2.2 ; 2.4 ; pi/4]	111.75	109.1
19	[22 ; 22 ; pi]	[2.2 ; 2.4 ; pi/1.5]	118.5	122.8
20	[22 ; 22 ; pi/0.5]	[2.2 ; 2.4 ; pi]	125.6	125.6

表 A.11: 提出方法之案例三結果（重複特徵環境、低解析度）

run	robot 1 初始姿態	robot 2 初始姿態	總行走距離 (m)	花費時間 (sec.)
1	[2.2 ; 2 ; pi/2]	[22 ; 22 ; pi/2.2]	115.5	106.8
2	[2.2 ; 2 ; pi/1.5]	[22 ; 22 ; pi/4]	91.3	100.2



3	[2.2 ; 2 ; pi]	[22 ; 22 ; pi/1.5]	122.7	112.2
4	[2.2 ; 2 ; pi/0.5]	[22 ; 22 ; pi]	105.7	105.0
5	[10.2 ; 2 ; pi/2]	[12 ; 22 ; pi/2.2]	121.35	116.2
6	[10.2 ; 2 ; pi/1.5]	[12 ; 22 ; pi/4]	103.95	106.2
7	[10.2 ; 2 ; pi/3]	[12 ; 22 ; pi/0.5]	86.65	96.8
8	[14.2 ; 2 ; pi/2]	[9 ; 22 ; pi/2.2]	94.65	100.6
9	[14.2 ; 2 ; pi/1.5]	[9 ; 22 ; pi/4]	112.65	106.6
10	[14.2 ; 2 ; pi/3]	[9 ; 22 ; pi/0.5]	108.4	104.1
11	[2 ; 10.2 ; pi/2]	[22 ; 12 ; pi/2.2]	96.3	102.7
12	[2 ; 10.2 ; pi/1.5]	[22 ; 12 ; pi/4]	120.6	111.1
13	[2 ; 10.2 ; pi/3]	[22 ; 12 ; pi/0.5]	115.95	107.2
14	[2 ; 14.2 ; pi/2]	[22 ; 9 ; pi/2.2]	83.7	97.7
15	[2 ; 14.2 ; pi/1.5]	[22 ; 9 ; pi/4]	98.25	102.7
16	[2 ; 14.2 ; pi/3]	[22 ; 9 ; pi/0.5]	89.95	95.2
17	[22 ; 22 ; pi/2]	[2.2 ; 2 ; pi/2.2]	90.8	97.8
18	[22 ; 22 ; pi/1.5]	[2.2 ; 2 ; pi/4]	104.95	104.2
19	[22 ; 22 ; pi]	[2.2 ; 2 ; pi/1.5]	118.05	107.4
20	[22 ; 22 ; pi/0.5]	[2.2 ; 2 ; pi]	124.65	128.2



表 A.12: 提出方法之案例四結果（重複特徵環境、高解析度）

run	robot 1 初始姿態	robot 2 初始姿態	總行走距離 (m)	花費時間 (sec.)
1	[2.2 ; 2 ; pi/2]	[22 ; 22 ; pi/2.2]	112.7	108.6
2	[2.2 ; 2 ; pi/1.5]	[22 ; 22 ; pi/4]	115.55	106.8
3	[2.2 ; 2 ; pi]	[22 ; 22 ; pi/1.5]	93.3	96.6
4	[2.2 ; 2 ; pi/0.5]	[22 ; 22 ; pi]	103.95	102.6
5	[10.2 ; 2 ; pi/2]	[12 ; 22 ; pi/2.2]	96.6	95.7
6	[10.2 ; 2 ; pi/1.5]	[12 ; 22 ; pi/4]	94	96.0
7	[10.2 ; 2 ; pi/3]	[12 ; 22 ; pi/0.5]	110.55	105.0
8	[14.2 ; 2 ; pi/2]	[9 ; 22 ; pi/2.2]	104.75	103.2
9	[14.2 ; 2 ; pi/1.5]	[9 ; 22 ; pi/4]	116.25	110.7
10	[14.2 ; 2 ; pi/3]	[9 ; 22 ; pi/0.5]	103.45	103.6
11	[2 ; 10.2 ; pi/2]	[22 ; 12 ; pi/2.2]	106.2	100.2
12	[2 ; 10.2 ; pi/1.5]	[22 ; 12 ; pi/4]	107.7	104.4
13	[2 ; 10.2 ; pi/3]	[22 ; 12 ; pi/0.5]	112.7	107.6
14	[2 ; 14.2 ; pi/2]	[22 ; 9 ; pi/2.2]	116.1	110.5
15	[2 ; 14.2 ; pi/1.5]	[22 ; 9 ; pi/4]	83.7	94.4
16	[2 ; 14.2 ; pi/3]	[22 ; 9 ; pi/0.5]	97.95	96.9
17	[22 ; 22 ; pi/2]	[2.2 ; 2 ; pi/2.2]	102.15	98.4

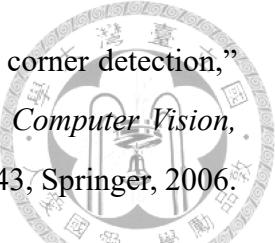


18	[22 ; 22 ; pi/1.5]	[2.2 ; 2 ; pi/4]	116.7	122.2
19	[22 ; 22 ; pi]	[2.2 ; 2 ; pi/1.5]	115.56	110.4
20	[22 ; 22 ; pi/0.5]	[2.2 ; 2 ; pi]	122	114.4

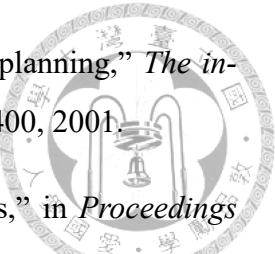


## 參考文獻

- [1] H. Umari, *Multi-robot map exploration based on multiple rapidly-exploring randomized trees*. PhD thesis, American University of Sharjah, 2017.
- [2] S. Yu, C. Fu, A. K. Gostar, and M. Hu, “A review on map-merging methods for typical map types in multiple-ground-robot slam solutions,” *Sensors*, vol. 20, no. 23, 2020.
- [3] B. Yamauchi, “A frontier-based approach for autonomous exploration,” in *Proceedings 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation CIRA'97. 'Towards New Computational Principles for Robotics and Automation'*, pp. 146–151, 1997.
- [4] V. T. Ferrão, C. D. N. Vinhal, and G. da Cruz, “An occupancy grid map merging algorithm invariant to scale, rotation and translation,” in *2017 Brazilian Conference on Intelligent Systems (BRACIS)*, pp. 246–251, IEEE, 2017.
- [5] H. Umari and S. Mukhopadhyay, “Autonomous robotic exploration based on multiple rapidly-exploring randomized trees,” in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1396–1402, IEEE, 2017.
- [6] S. Karaman and E. Frazzoli, “Sampling-based algorithms for optimal motion planning,” *The international journal of robotics research*, vol. 30, no. 7, pp. 846–894, 2011.
- [7] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.



- [8] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *Computer Vision–ECCV 2006: 9th European Conference on Computer Vision, Graz, Austria, May 7–13, 2006. Proceedings, Part I* 9, pp. 430–443, Springer, 2006.
- [9] S. Wang, Z. Wu, and W. Zhang, "An overview of slam," in *Proceedings of 2018 Chinese Intelligent Systems Conference: Volume I*, pp. 673–681, Springer, 2019.
- [10] A. Birk and S. Carpin, "Merging occupancy grid maps from multiple robots," *Proceedings of the IEEE*, vol. 94, no. 7, pp. 1384–1397, 2006.
- [11] S. Yu, C. Fu, A. K. Gostar, and M. Hu, "A review on map-merging methods for typical map types in multiple-ground-robot slam solutions," *Sensors*, vol. 20, no. 23, p. 6988, 2020.
- [12] Z. Jiang, J. Zhu, Y. Li, J. Wang, Z. Li, and H. Lu, "Simultaneous merging multiple grid maps using the robust motion averaging," *Journal of Intelligent & Robotic Systems*, vol. 94, no. 3, pp. 655–668, 2019.
- [13] Z. Jiang, J. Zhu, C. Jin, S. Xu, Y. Zhou, and S. Pang, "Simultaneously merging multi-robot grid maps at different resolutions," *Multimedia Tools and Applications*, vol. 79, no. 21, pp. 14553–14572, 2020.
- [14] F. Amigoni, S. Gasparini, and M. Gini, "Building segment-based maps without pose information," *Proceedings of the IEEE*, vol. 94, no. 7, pp. 1340–1359, 2006.
- [15] J. Weingarten and R. Siegwart, "3d slam using planar segments," in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3062–3067, IEEE, 2006.
- [16] H. Choset and K. Nagatani, "Topological simultaneous localization and mapping (slam): toward exact localization without explicit localization," *IEEE Transactions on robotics and automation*, vol. 17, no. 2, pp. 125–137, 2001.
- [17] O. Booij, B. Terwijn, Z. Zivkovic, and B. Kroese, "Navigation using an appearance based topological map," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pp. 3927–3932, IEEE, 2007.



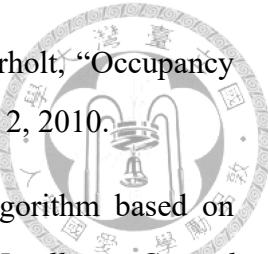
- [18] S. M. LaValle and J. J. Kuffner Jr, “Randomized kinodynamic planning,” *The international journal of robotics research*, vol. 20, no. 5, pp. 378–400, 2001.
- [19] B. Yamauchi, “Frontier-based exploration using multiple robots,” in *Proceedings 1998 Second International Conference on Autonomous Agents AGENTS'98.*, pp. 47–53, 1998.
- [20] M. Keidar and G. A. Kaminka, “Robot exploration with fast frontier detection: Theory and experiments,” in *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*, pp. 113–120, 2012.
- [21] Y. Wang, A. Liang, and H. Guan, “Frontier-based multi-robot map exploration using particle swarm optimization,” in *2011 IEEE Symposium on Swarm Intelligence*, pp. 1–6, 2011.
- [22] P. Senarathne, D. Wang, Z. Wang, and Q. Chen, “Efficient frontier detection and management for robot exploration,” in *2013 IEEE International Conference on Cyber Technology in Automation, Control and Intelligent Systems*, pp. 114–119, 2013.
- [23] Z. Yan, L. Fabresse, J. Laval, and N. Bouraqadi, “Team size optimization for multi-robot exploration,” in *Simulation, Modeling, and Programming for Autonomous Robots* (D. Brugali, J. F. Broenink, T. Kroeger, and B. A. MacDonald, eds.), (Cham), pp. 438–449, Springer International Publishing, 2014.
- [24] S. M. LaValle, “Rapidly-exploring random trees : a new tool for path planning,” *The annual research report*, 1998.
- [25] G. Oriolo, M. Vendittelli, L. Freda, and G. Troso, “The srt method: randomized strategies for exploration,” in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, vol. 5, pp. 4688–4694 Vol.5, 2004.
- [26] A. Franchi, L. Freda, G. Oriolo, and M. Vendittelli, “A randomized strategy for cooperative robot exploration,” in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pp. 768–774, IEEE, 2007.



- [27] A. Franchi, G. Oriolo, L. Freda, and M. Vendittelli, “A decentralized strategy for cooperative robot exploration,” in *First International Conference on Robot Communication and Coordination (ROBOCOMM 2007)*, pp. 1–8, ICST, 2007.
- [28] R. Zlot, A. Stentz, M. B. Dias, and S. Thayer, “Multi-robot exploration controlled by a market economy,” in *Proceedings 2002 IEEE international conference on robotics and automation (Cat. No. 02CH37292)*, vol. 3, pp. 3016–3023, IEEE, 2002.
- [29] H.-C. Lee, S.-H. Lee, T.-S. Lee, D.-J. Kim, and B.-H. Lee, “A survey of map merging techniques for cooperative-slam,” in *2012 9th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)*, pp. 285–287, IEEE, 2012.
- [30] K. Konolige, D. Fox, B. Limketkai, J. Ko, and B. Stewart, “Map merging for distributed robot navigation,” in *Proceedings 2003 IEEE/RSJ international conference on intelligent robots and systems (IROS 2003) (Cat. No. 03CH37453)*, vol. 1, pp. 212–217, IEEE, 2003.
- [31] X. S. Zhou and S. I. Roumeliotis, “Multi-robot slam with unknown initial correspondence: The robot rendezvous case,” in *2006 IEEE/RSJ international conference on intelligent robots and systems*, pp. 1785–1792, IEEE, 2006.
- [32] H. S. Lee and K. M. Lee, “Multi-robot slam using ceiling vision,” in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 912–917, IEEE, 2009.
- [33] F. Tungadi, W. L. D. Lui, L. Kleeman, and R. Jarvis, “Robust online map merging system using laser scan matching and omnidirectional vision,” in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 7–14, IEEE, 2010.
- [34] H. Lee *et al.*, “Implementation of a network-based robot system for cooperative recognition and localization of multiple objects,” in *Proc. IEEK Summer Conf.*, pp. 2008–2010, 2012.
- [35] S. Carpin, A. Birk, and V. Jucikas, “On map merging,” *Robotics and Autonomous Systems*, vol. 53, no. 1, pp. 1–14, 2005.



- [36] M. Locatelli, “Simulated annealing algorithms for continuous global optimization,” *Handbook of Global Optimization: Volume 2*, pp. 179–229, 2002.
- [37] R. Rocha, F. Ferreira, and J. Dias, “Multi-robot complete exploration using hill climbing and topological recovery,” in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1884–1889, IEEE, 2008.
- [38] S. Carpin and G. Pillonetto, “Motion planning using adaptive random walks,” *IEEE Transactions on Robotics*, vol. 21, no. 1, pp. 129–136, 2005.
- [39] H. Li, M. Tsukada, F. Nashashibi, and M. Parent, “Multivehicle cooperative local mapping: A methodology based on occupancy grid map merging,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 5, pp. 2089–2100, 2014.
- [40] K.-F. Man, K. S. Tang, and S. Kwong, *Genetic algorithms: concepts and designs*. Springer Science & Business Media, 2001.
- [41] S. Saeedi, L. Paull, M. Trentini, and H. Li, “Multiple robot simultaneous localization and mapping,” in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 853–858, IEEE, 2011.
- [42] J.-L. Blanco, J. González-Jiménez, and J.-A. Fernández-Madrigal, “A robust, multi-hypothesis approach to matching occupancy grid maps,” *Robotica*, vol. 31, no. 5, pp. 687–701, 2013.
- [43] K. Wang, S. Jia, Y. Li, X. Li, and B. Guo, “Research on map merging for multi-robotic system based on rtm,” in *2012 IEEE International Conference on Information and Automation*, pp. 156–161, IEEE, 2012.
- [44] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, “Speeded-up robust features (surf),” *Computer vision and image understanding*, vol. 110, no. 3, pp. 346–359, 2008.
- [45] A. Durdu and M. Korkmaz, “A novel map-merging technique for occupancy grid-based maps using multiple robots: A semantic approach,” *Turkish Journal of Electrical Engineering and Computer Sciences*, vol. 27, no. 5, pp. 3980–3993, 2019.



- [46] Y. Alnounou, M. J. Paulik, M. Krishnan, G. Hudas, and J. Overholt, “Occupancy grid map merging using feature maps,” *Journal Article*, vol. 1, p. 2, 2010.
- [47] Y. Sun, R. Sun, S. Yu, and Y. Peng, “A grid map fusion algorithm based on maximum common subgraph,” in *2018 13th World Congress on Intelligent Control and Automation (WCICA)*, pp. 58–63, IEEE, 2018.
- [48] S. Carpin, “Fast and accurate map merging for multi-robot systems,” *Autonomous robots*, vol. 25, no. 3, pp. 305–316, 2008.
- [49] D. G. Lowe, “Object recognition from local scale-invariant features,” in *Proceedings of the seventh IEEE international conference on computer vision*, vol. 2, pp. 1150–1157, Ieee, 1999.
- [50] J. M. Phillips, R. Liu, and C. Tomasi, “Outlier robust icp for minimizing fractional rmsd,” in *Sixth International Conference on 3-D Digital Imaging and Modeling (3DIM 2007)*, pp. 427–434, IEEE, 2007.
- [51] S. Arya, “A review on image stitching and its different methods,” *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 5, no. 5, pp. 299–303, 2015.
- [52] T. Adame, M. Carrascosa, and B. Bellalta, “The tmb path loss model for 5 ghz indoor wifi scenarios: On the empirical relationship between rssi, mcs, and spatial streams,” in *2019 Wireless Days (WD)*, pp. 1–8, IEEE, 2019.
- [53] D. Comaniciu and P. Meer, “Mean shift: A robust approach toward feature space analysis,” *IEEE Transactions on pattern analysis and machine intelligence*, vol. 24, no. 5, pp. 603–619, 2002.
- [54] Y. Cheng, “Mean shift, mode seeking, and clustering,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 17, no. 8, pp. 790–799, 1995.
- [55] I. Ulrich and J. Borenstein, “Vfh+: Reliable obstacle avoidance for fast mobile robots,” in *Proceedings. 1998 IEEE international conference on robotics and automation (Cat. No. 98CH36146)*, vol. 2, pp. 1572–1577, IEEE, 1998.

- [56] F. R. Part, “15.247: Operation within the bands 902-928 mhz, 2400-2483.5 mhz, and 5725-5850 mhz,” *Part15: Radio Frequency Devices*, 1997.
- [57] A. Savitzky and M. J. Golay, “Smoothing and differentiation of data by simplified least squares procedures.,” *Analytical chemistry*, vol. 36, no. 8, pp. 1627–1639, 1964.
- [58] J. J. Uicker, G. R. Pennock, J. E. Shigley, and J. M. McCarthy, *Theory of machines and mechanisms*, vol. 768. Oxford University Press New York, 2003.
- [59] A. Ranganathan, “The levenberg-marquardt algorithm,” *Tutorial on LM algorithm*, vol. 11, no. 1, pp. 101–110, 2004.
- [60] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, “Orb: An efficient alternative to sift or surf,” in *2011 International conference on computer vision*, pp. 2564–2571, Ieee, 2011.
- [61] J. J. Koenderink, “The structure of images,” *Biological cybernetics*, vol. 50, no. 5, pp. 363–370, 1984.
- [62] T. Lindeberg, “Scale-space theory: A basic tool for analyzing structures at different scales,” *Journal of applied statistics*, vol. 21, no. 1-2, pp. 225–270, 1994.
- [63] D. Fox, W. Burgard, S. Thrun, and A. B. Cremers, “Position estimation for mobile robots in dynamic environments,” *AAAI/IAAI*, vol. 1998, pp. 983–988, 1998.

