

國立交通大學

電機與控制工程學系

碩 士 論 文

相機陣列影像縫合之快速校正技術

Fast Calibration Techniques for Auto-Stitch of Camera Array  
Image

研 究 生：莊詠麟

指 導 教 授：董蘭榮 博士

中 華 民 國 九 十 七 年 九 月

相機陣列影像縫合之快速校正技術

Fast Calibration Techniques for Auto-Stitch of Camera Array  
Image

研究生：莊詠麟

Student: Yung-Lin Ghuang

指導教授：董蘭榮 博士

Advisor: Lan-Rong Dung

國立交通大學

電機與控制工程學系



A Thesis

Submitted to Department of Electrical and Control Engineering

September 2008

Hsinchu, Taiwan, Republic of China

中華民國九十七年九月

# 相機陣列影像縫合之快速校正技術

研究生：莊詠麟

指導教授：董蘭榮 博士

國立交通大學電機與控制工程學系

## 摘要

這篇論文的目標是發展一個使用影像縫合技術於相機校正設計，加快相機校正速度的演算法。近年來因為相機校正技術需要輔助校正物且機器的精確性誤差的關係，因此影像縫合技術在相機校正領域佔有越來越重要的地位。而在影像縫合的應用上，考慮縫合速度和影像品質的權衡是件非常重要的事。傳統的演算法由於為了達到夠好的影像品質，因此消耗了相當大的運算量，也因為如此，傳統的演算法不適用於硬體的實作。因此本篇論文提出一個快速的演算法，在擷取特徵點的時候由內外往外擴散慢慢刪除，並且使用適應性的方法讓擷取特徵點的步驟停止。接著我們在RANSAC的部份也使用特殊的技巧來擷取matching points做RANSAC，使擷取到的matching points容易呈現兩極化的現象，如此就不需要做大量的RANSAC，此兩個步驟都大大地節省CPU time，加快速度但卻可以得到一樣影像品質。最後我們也發現在多台Sensor的角度、位置、距離都已經固定的前提下，其影像彼此之間的轉換關係是固定的，因此我們只要跑過一次完整的演算法即可求出彼此之間的轉換關係，所以之後的校正只要透過其固定的轉換關係即可校正，如此的發現非常有利於我們在硬體實現的突破，因此校正的速度將可以大幅地改善。

# Fast Calibration Techniques for Auto-Stitch of Camera Array Image

Student : Yung-Lin Chuang

Advisor : Lan-Rong Dung

Institute of Electrical and Control Engineering National  
Chiao-Tung University

## Abstract

The objective of this thesis is to develop an algorithm of AutoStitch for Camera Calibration and to improve the speed of Camera Calibration. Because the technique of Camera Calibration needs some auxiliary patterns to do the calibration and the error correctness of the machines, Auto-Stitch has been played an important part in the field of Camera Calibration. In applications of Autostitch, it is imperative to consider the speed of stitch and performance trade-offs. Traditional methods spend a lot of execution times in order to get good performance. Because of this, it is not suitable to use traditional methods in hardware. This thesis develops an algorithm to improve the speed of Autostitch. In the step of Features Extraction, it deletes Feature Points from small search range to broad search range and uses a special adaptive method to stop this step. After this, we use adaptive method in RANSAC times and repeat proper times to accomplish the performance that we want. We don't need to do a lot of RANSAC times. These two steps economizing the CPU time, higher speed but we can almost get the same performance. Finally, we also find that if the angle, position, and the distance of the multi-sensor are fixed, the rule of transformation can be fetched through our algorithm just one time. After this, we can use the fixed transformation to

do other calibration. This method is useful to realize in hardware and the speed of calibration can be improved substantially.



## 誌謝

本篇論文得以順利完成，首先要感謝的是我的指導教授——董蘭榮教授，在碩士班的兩年間，董教授不厭其煩地指導我，當我陷入瓶頸時，董教授亦適時地指點我正確的方向，讓我不至於常常失焦，並且及時做出修正，並且提供非常豐富的資源，讓我能好好潛心於學習研究，讓我在這兩年間獲益良多。

同時，也感謝實驗室的學長——穎毅、盟淳、宗錫、騰轟、學之、文豪、俊衛、峻轍、志惟、仕捷、信承，在我的求學過程中給予指點與幫助，以及陪伴兩年的同學們——XT、Lander、小康、Kong，在課業與生活上的互相扶持、分擔紓解彼此的壓力，給了我一段美好的研究所時光。當然也要感謝實驗室可愛又嘴砲的學弟妹們和兩位正妹助理——邱哥、單哥、智聖、小嘉鴻、貞如、乃禎、惟茵，因為有了你們使得枯燥煩悶的實驗室常常很嗨。

最後要感謝我的家人的支持，有了你們的鼓勵，使我無後顧之憂，才能夠安心地完成碩士班學業。

謹將此論文獻給所有關心我的人，在此致上最深的謝意。

# 章節目錄

中文摘要	i
英文摘要	ii
誌謝	iv
章節目錄	v
圖目錄	vii
表目錄	x
<b>第一章 簡介</b> .....	<b>1</b>
1-1 相機校正(Camera Calibration)與影像縫合(AutoStitch)的應用....	1
1-2 相機校正(Camera Calibration)與影像縫合(AutoStitch)演的缺點	2
1-3 影像縫合與校正在效能上的考量.....	5
1-4 章節規劃.....	6
<b>第二章 背景</b> .....	<b>7</b>
2-1 影像縫合流程(The Flow of AutoStitch).....	7
2-2 特徵點擷取(Feature Extraction).....	7
2-2.1 特徵點定義.....	7
2-2.2 一般Harris Corner取法.....	9
2-2.3 適應性非最大化壓縮法(Adaptive Non-Maximal Suppression)	
.....	14
2-3 特徵點匹配(Feature Matching).....	17
2-3.1 最小平方差SSD(Sum of Squared Differences).....	17
2-3.2 次近鄰居法(Second-Closest Neighbor).....	18
2-4 剔除掉錯誤的匹配點>Delete The False Match).....	22

2-5	歪斜影像(Warp Image).....	26
2-6	混合影像(Blending).....	28
<b>第三章</b>	<b>使用BUMS與PRSC影像縫合快速演算法設計</b> .....	<b>33</b>
3-1	Bottom-Up Maxima Selection(BUMS).....	33
3-1.1	傳統適應性非最大化壓縮法之深入討論.....	33
3-1.2	BUMS流程(The flow of Bottom-Up Maxima Selection).....	39
3-1.3	特徵點均勻分佈特性驗證(Verification of the Uniform Distribution of the Feature Points).....	45
3-2	極化型隨機取樣程序(Polarized Random Sample Consensus).....	48
3-2.1	RANSAC執行次數之討論(Discussion of RANSAC Times).....	48
3-2.2	極化型隨機取樣程序流程(The Flow of Polarized Random Sample Consensus).....	51
<b>第四章</b>	<b>模擬結果之探討與比較</b> .....	<b>57</b>
4-1	BUMS運算效率模擬(Simulation of Computational Efficiency for Bottom-Up Maxima Selection).....	57
4-2	PRSC運算效率模擬(Simulation of Computational Efficiency for Polarized Random Sample Consensus).....	60
4-3	整體運算效率模擬(Simulation of Total Computational Efficiency for the Algorithm).....	62
4-4	轉換矩陣H的重覆使用性質(Reusability of Homography H).....	64
<b>第五章</b>	<b>結論與未來發展</b> .....	<b>68</b>
<b>參考文獻：</b> .....		<b>71</b>

## 圖目錄

圖 1-1.1 相機陣列校正技術範例.....	1
圖 1-2.1 360 度環場影像 1 .....	3
圖 1-2.2 360 度環場影像 2 .....	3
圖 1-2.3 全自動絕對座標縫合影像.....	4
圖 2-2.1 特徵點示意圖.....	8
圖 2-2.2 只取一般 Corner 範例一.....	10
圖 2-2.3 只取一般 Corner 範例二.....	11
圖 2-2.4 ni 分別取(a)200 (b)350 (c)500 的結果.....	11
圖 2-2.5 Corner Detector 顯示圖，場景為國立交通大學運動場.....	13
圖 2-2.6 Corner Detector 顯示圖，場景為國立交通大學運動場和宿舍.....	13
圖 2-2.7 適應性非最大化壓縮法 (ANMS) 範例一.....	15
圖 2-2.8 ANMS 流程圖.....	16
圖 2-2.9 The highest corner strength 法與 ANMS 比較圖範例一.....	16
圖 2-2.10 The highest corner strength 法與 ANMS 比較圖範例二.....	17
圖 2-3.1 SSD (Sum of Squared Differences) 示意圖.....	18
圖 2-3.2 次近鄰居法 (Second-Closest Neighbor) 的流程圖.....	19
圖 2-3.3 次近鄰居法 (Second-Closest Neighbor) 的成功機率圖表.....	20
圖 2-3.4 特徵點擷取與特徵點匹配的範例一.....	20
圖 2-3.5 特徵點擷取與特徵點匹配的範例二.....	21
圖 2-4.1 RANSAC 範例一，圖為交通大學工程五館.....	24
圖 2-4.2 RANSAC 範例二，圖為台北 101 .....	25
圖 2-4.3 RANSAC (RANDOM Sample Consensus) flow chart .....	25
圖 2-5.1 歪斜影像範例一.....	26

圖 2-5.2 歪斜影像範例二.....	26
圖 2-5.3 沒有使用 RANSAC 的歪斜影像.....	27
圖 2-5.4. 沒有使用 RANSAC 的縫合影像.....	27
圖 2-6.1 原始影像與歪斜影像的相對位置已經吻合.....	28
圖 2-6.2 沒有用特殊 Blending strategy 呈現的失真影像範例一.....	29
圖 2-6.3 沒有用特殊 Blending strategy 呈現的失真影像範例二.....	29
圖 2-6.4 Weight Function 示意圖.....	30
圖 2-6.5 歪斜影像對應的權重函數.....	30
圖 2-6.6 原始影像對應的權重函數.....	31
圖 2-6.7 兩張圖的相對位置與權重函數關係.....	31
圖 2-6.8 經過 Blending Strategy 所產生的縫合影像.....	32
圖 3-1.1 適應性非最大化壓縮法 (ANMS) 範例一.....	34
圖 3-1.2 ANMS 流程圖.....	35
圖 3-1.3 ANMS 與 Search Range $r$ 之深入分析範例一.....	36
圖 3-1.4 ANMS 與 Search Range $r$ 之深入分析範例二.....	38
圖 3-1.5 修正型 ANMS 演算法流程圖.....	40
圖 3-1.6 BUMS 與 Search Range $r$ 之深入分析範例一.....	41
圖 3-1.7 BUMS 與 Search Range $r$ 之深入分析範例二.....	43
圖 3-1.8 Bottom-Up Maxima Selection (BUMS) 流程圖.....	45
圖 3-1.9 特徵點較為集中的情況範例.....	46
圖 3-1.10 特徵點較為分散的情況範例.....	47
圖 3-2.1 失敗 RANSAC 與失敗的縫合影像.....	49
圖 3-2.2 執行不同次數 RANSAC 的內部點結果與縫合影像.....	50
圖 3-2.3 極化型隨機取樣程序 (PRSC) 流程圖.....	53
圖 3-2.4 PRSC 臨界值取高標的結果.....	55
圖 4-1.1 測試模擬的 12 組 Pattern.....	58

圖 4-1.2 縫合三張影像.....	60
圖 4-2.1 傳統 RANSAC 與 PRSC 運算量比較圖.....	62
圖 4-3.1 傳統演算法與快速演算法比較圖.....	63
圖 4-4.1 Multi-Sensor 相對位置固定情形下,轉換矩陣重複性的實驗步驟....	65
圖 4-4.2 轉換矩陣 H 重複性質的基底 Pattern.....	65
圖 4-4.3 使用相同的轉換矩陣 H 所校正的影像.....	66
圖 5-1.1 透視圖.....	68
圖 5-1.2 多 Sensor 平台範例.....	69
圖 5-1.3 多 Sensor 平台影像及縫合影像.....	70
圖 5-1.4 焦距過近的縫合影像.....	70



## 表目錄

---

---

表 3-1.1 SR 從無限大往內縮回時特徵點擷取數量與增加數量統計表範例一...	37
表 3-1.2 SR 從無限大往內縮回時特徵點擷取數量與增加數量統計表範例二...	39
表 3-1.3 SR 從 $r=1$ 往外擴張時特徵點擷取數量與剔除數量統計表範例一.....	42
表 3-1.4 SR 從 $r=1$ 往外擴張時特徵點擷取數量與剔除數量統計表範例二.....	44
表 3-2.1 匹配點經過 5000 次 PRSC 投票結果之關係表.....	52
表 3-2.2 執行 10 次 PRSC 低標和均標的成功機率比較.....	56
表 4-1.1 縫合兩張圖的傳統 ANMS 與 BUMS 的計算效率比較.....	59
表 4-1.2 縫合三張圖的傳統 ANMS 與 BUMS 的計算效率比較.....	59
表 4-2.1 縫合兩張圖的傳統 RANSAC 的 PRSC 的計算效率比較.....	61
表 4-2.2 縫合三張圖的傳統 RANSAC 的 PRSC 的計算效率比較.....	62
表 4-3.1 縫合兩張圖時傳統演算法與我們快速演算法的計算效率比較.....	63
表 4-3.2 縫合三張圖時傳統演算法與我們快速演算法的計算效率比較.....	63

# 第一章 簡介

## 1-1 相機校正(Camera Calibration)與影像縫合(AutoStitch)的應用

隨著相機校正和影像處理的發展，影像縫合(AutoStitch)近幾年在影像處理中已經佔有越來越重要的地位，其可應用的範圍也越來越廣，除了攝影愛好者原本受限於相機角度有限而無法拍攝廣角影像，因為 AutoStitch 的發展而一償宿願，其他還有如 Camera Motion[1]-[3]，Scene Motion[4]-[6]，Advanced Camera Modelling[7]，Photometric Modelling[8]-[10]，監視系統、軍事系統方面，如圖 1-1.1，AutoStitch 的應用也是越來越趨於廣泛。

在早期 AutoStitch 或 Camera Calibration 技術尚未成熟的時候，所有的影像要縫合或是校正，都需要人工徒手靠著繪圖軟體來接合或手動修正，這種人工方法相當麻煩，而且效果不盡理想。直到在相機校正領域[40]-[41]提出了用相機相對位置來做校正，[11]-[14]提出了利用 direct method 來做影像縫合校正，此技術才突破了人工徒手校正的瓶頸，之後[3]，[15]-[16]更進一步提出了 feature-based method，影像縫合與校正技術才慢慢趨近於成熟，feature-based method 比起 direct method，運算量更加節省，並且拒絕 outlier 的能力更強，其校正結果自然誤差更小更加可靠。



圖 1-1.1 相機陣列校正技術範例(本圖來源出自[44])

## 1-2 相機校正(Camera Calibration)與影像縫合(AutoStitch)演的缺點

通常一個好的相機校正方法需要考慮幾種因素，第一、自動化，即不需使用者給定一些參數的初始值或是校正的過程當中還需要使用者做適當地調整。第二、精確性，相機之間彼此的相對位置精確性是否足夠，校正過後的結果是否夠精確，當然直接影響校正的好壞。第三、計算效率，盡量避免去解高次的非線性方程式，特別是一些需要重複執行相當多次的地方。第四、具備彈性，使相機校正技術可適用於各種不同的環境當中。第五、適用性，能適用於各種不同的相機，而不只是限定在特殊的高檔攝影機。

在相機校正(Camera Calibration)的演算法當中，常常使用的方法是測試兩相機之間彼此的參數，並且適當地使用輔助校正物來做校正，但此種方法卻有幾種缺點，第一，多相機相對位置之間彼此的誤差，有可能希望彼此之間夾角是 30 度，但事實上卻有誤差變成 31 度，以致於校正上也有誤差。第二，需要輔助校正物來做校正，增加校正的複雜度，也增加成本，因此近年來在相機校正的領域都是使用彼此之間的對應點來做校正，即影像縫合(AutoStitch)技術。如此即使相機之間彼此有所誤差也會因為 AutoStitch 的技術而補償回來，並且此方法也不需要任何輔助校正物來做校正，只需一個固定的場景即可，因此影像縫合校正技術已成為近年來相機校正技術的核心，在本篇論文當中我們也將採取影像縫合的方式來做相機的校正技術並加以改良。

在要做影像縫合(AutoStitch)之前，我們需要先知道影像縫合在演算法上會遇到的瓶頸或缺點有哪些。第一，效能(performance)的判定:縫合影像結果的效能好壞，在目前的文獻裡還沒有較精確的準則來衡量，只能憑著肉眼的判斷來衡量縫合影像結果的好壞，且影像縫合的效能判定也會因需求的不同而有不同的判定，例如:360 度環場影像(360° Panorama Image)[17]-[20]、全自動絕對座標縫合影像(Automatically Absolute Coordinated Image Stitching)……等。360 度環場影像(360° Panorama Image)所希望達到的效果是可以看到前後左右所有

360 度的影像[17]-[20]，但是這樣的影像往往會造成扭曲的效果，與真實世界 (ground truth)不符合，如圖 1-2.1、圖 1-2.2。在強調 360 度環場影像(360° Panorama Image)的前提下，這兩張圖達到了我們的要求，即前後左右 360 度的影像皆可看到，但是我們也不難發現，這兩張圖很明顯地失真了，圖 1-2.1 的竿子應該是直的，卻被扭曲了，圖 1-2.2 的工作牆壁應該是平的，卻被扭曲了。為了達到 360 度環場影像的效果，卻失去了原本真實世界的內容(content)。全自動絕對座標縫合影像(Automatically Absolute Coordinated Image Stithing) [21]-[22]所希望是達到與真實世界(ground truth)內容一樣的結果，所以相對



圖 1-2.1 360 度環場影像 1(本圖 Pattern 來源出自[42])



圖 1-2.2 360 度環場影像 2



圖 1-2.3 全自動絕對座標縫合影像

位置也必須一樣，如圖 1-2.3。圖 1-2.3 縫合影像所顯示的影像內容與真實世界的內容(ground truth)完全一樣，連相對位置也一樣，但是很明顯地我們也可以發現為了達到相對位置一樣的效果，在縫合的過程必須也把影像做歪斜，以致於縫合後的結果越來越歪。此處的歪斜與前面 360 度環場影像的扭曲不一樣，360 度環場影像是為了達到 360 度環場效果所以導致相對位置不同以致失真。全自動絕對座標縫合影像[21]-[22]是為了達到絕對座標都與真實世界一樣所以造成影像的歪斜，因為沒有顯示出來的部份本來就沒有拍攝到，因此根本不會被縫合出來。因此，影像縫合的效能判定也會因需求的不同而有不同的判定，目前還沒有較一個精確的準則來衡量效能(performance)的判定。

第二，運算量過大:在 1-1 節我們有大概提到，早期所有的影像要縫合或是修正，都需要人工徒手靠著繪圖軟體來接合或手動修正，這種方法好處是完全不用機器來運算，但是人工方法相當麻煩，而且效果不盡理想。直到[11]-[14]提出了利用 direct method 來做影像縫合與校正，才突破了人工徒手校正的瓶頸，但是 direct method 因為是用整張影像來比較，所以運算量非常大，之後 [3], [15]-[16]更進一步提出了 feature-based method，運算量比較節省，但就整體影像縫合(AutoStitch)而言，運算量過大還是一個存在已久的問題，本篇論

文主要的目的之一就是希望可以節省傳統的演算法運算量，甚至在某些前提的假設下，運算量可以壓低到適合用於硬體實作上。

### 1-3 影像縫合與校正在效能上的考量

正如 1-2 所提到的，在影像縫合(AutoStitch)的領域裡，還沒有一套精確的準則來判斷縫合過後的影像品質好壞，但從整個影像縫合(AutoStitch)系統的角度來看，仍然可以大約把影像縫合系統(AutoStitch System)的效能分成以下幾方面來探討：

- (1) 縫合影像本身的品質好壞：縫合過後的影像清楚與否、是否有不合理的內容(圖片搭不起來)，首要決定了影像縫合系統效能的好壞。
- (2) 縫合影像本身與真實世界內容(ground truth)的誤差：不同於第一點，有些影像即使縫合出來效果清晰無誤，但是卻與真實世界內容(ground truth)有相當的誤差，正如 1-2 提到的，影像縫合的效能判定也會因需求的不同而有不同的判定，360 度環場影像(360° Panorama Image)[17]-[20]所縫合出來的影像，相當清楚無誤，但是卻和真實世界內容(ground truth)有段相當的誤差，所以在第二點上仍然視我們的需求而判定。
- (3) 縫合演算法的強健度：越強健的演算法自然能夠相容越多組的影像內容，即越強健的演算法可適用於各種不同的樣本影像(Pattern)，若是縫合演算法不夠強健，自然地某些較為 critical 的樣本影像(Pattern)即無法適用，在影像縫合系統(AutoStitch System)的效能考量中最為重要。
- (4) 運算量多寡：正如 1-2 所提到的，影像縫合的發展歷史當中，從 direct method 再到 feature-based method，節省了許多的運算量，但是就整個影像縫合系統(AutoStitch System)而言，運算量過大還是一個存在已久的問題。

綜合以上的分析以及討論，我們在(1)-(3)都與傳統演算法相同的情形下，提出一套屬於我們自己的快速演算法，使得縫合運算量(4)大大地再降低，甚至在某些前提的假設下，運算量可以壓低到適合用於硬體實作上。

## 1-4 章節規劃

本章節中，介紹了目前相機校正(Camera Calibration)與影像縫合(AutoStitch)的應用範圍以及在本論文裡所要探討的事情做個簡單的描述。接下來第二章中，會說明影像縫合(AutoStitch)的基本背景理論和演算法來做相機的校正。第三章我們就影像縫合(AutoStitch)在設計過程中所遭遇到的困難，和問題的解決方法，做進一步的了解與探討。並針對各種不同的狀況做比較，模擬結果列於第四章中。最後第五章再對這一連串的分析所發現與觀察到現象做個總結。



## 第二章 背景

---

### 2-1 影像縫合流程(The Flow of AutoStitch)

在影像縫合(AutoStitch)的過程中，大致上可分成幾個步驟，第一，特徵點擷取(Feature Extraction)。為的是要讓在影像縫合的剛開始，並不需使用影像中的每個像素(pixel)來做運算，只需要找出幾個夠特殊的點(即特徵點)來做代表即可。第二，特徵點匹配(Feature Matching)。在初步擷取特徵點過後，我們也需要做初步的特徵點匹配，以確認兩張圖當中的特徵點哪些是彼此對應到的。第三，剔除掉錯誤的匹配點。承接第二步驟，因為在初步特徵點匹配的過程結束之後，仍然有可能會存在錯誤的匹配點，因此需要建立一套機制使得錯誤的特徵點可以被剔除掉。第四，歪斜影像(Warp Image)。正如 1-3 所提到的，不同需求的影像縫合在效能上的判定準則也不一樣，本篇論文以對座標縫合影像(Automatically Absolute Coordinated Image Stithing)來做基準，即相對位置也要與真實世界內容(Ground Truth)一樣。所以第四步驟我們需要把影像做歪斜使其可以跟另一張影像正確地縫合。第五，混合影像(Blending)。在把影像歪斜過後，最後一步就是要做適當地混合，不同的混合方法所呈現出來的最後縫合影像結果也不盡相同。接下來在第二章中我們將針對上列影像縫合的基本步驟做一個完整地背景介紹。

### 2-2 特徵點擷取(Feature Extraction)

#### 2-2.1 特徵點定義

正如 1-1 當中所提到的，在早期所有的影像要縫合或是修正，都需要人工徒手靠著繪圖軟體來接合或手動修正，這種人工方法相當麻煩，而且效果不盡理想。直到[11]-[14]提出了利用 direct method 來做影像縫合與校正，此技術才突破了人工徒手校正的瓶頸，之後[3], [15]-[16]更進一步提出了 feature-based method，即不需用整張影像的像素(pixel)來做運算，只需要找出幾個夠特殊的

點(即特徵點)來做代表即可，[23]-[32] 分別提出了許多不同的方法，其中[26] Harris Corner Detector 的作法，是把影像當中的角落(Corner)擷取出來。首先我們需要知道為什麼需要找角落來當特徵點，若我們使用影像中比較平坦(flat)的點來當特徵點，則往上下左右稍微移動一點點，所看到的東西仍會很像，所以這不是一個適合的特徵點。接著，若我們使用影像中邊界(edge)的點來當特徵點，則此點往某兩個方向稍微移動一點點，所看到的東西仍會很像，仍然不是一個適合的特徵點。最後，若我們使用影像中的角落(Corner)點來當特徵點，則此點往上下左右稍微移動一點點，所看到的東西仍會很不一樣，因此我們可以得知角落點(Corner)為我們所希望找到的特徵點，圖 2-2.1 即是一個最簡單的例子。

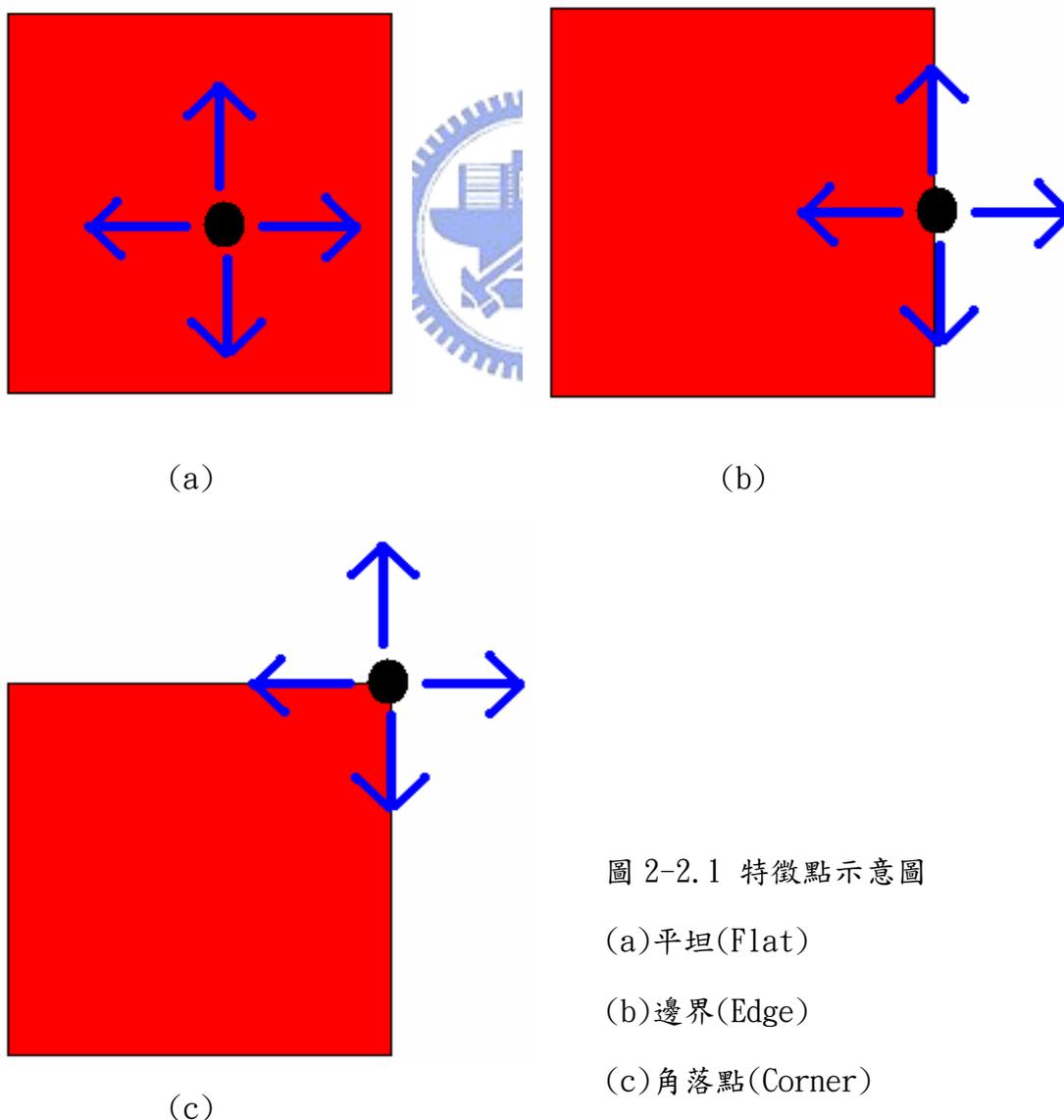


圖 2-2.1 特徵點示意圖

(a)平坦(Flat)

(b)邊界(Edge)

(c)角落點(Corner)

## 2-2.2 一般 Harris Corner 取法

接著我們要來討論如何找出影像的角落點以作為特徵點，[26]首先把每個輸入影像的點拿來做計算， $I_x, I_y, I_x \times I_y, I_x^2, I_y^2$  表示我們去計算每點在不同方向的微分，以及他們彼此相乘和各自的平方， $w$  為高斯函數，對於每個影像中的 Pixel，算出運算式 2-1 中矩陣的特徵值

$$A = \begin{bmatrix} (I_x^2) \otimes w & (I_x * I_y) \otimes w \\ (I_x * I_y) \otimes w & (I_y^2) \otimes w \end{bmatrix} \text{-----}(\text{式})2-1$$

當式 2-1 矩陣特徵值計算出來後，我們去觀察計算出來的特徵值 (EigenValue):

- (1) 如果這兩個特徵值都很小，表示此點與附近的關係為平坦的(flat)，不是我們想要的情況(我們想要的為角落)。
- (2) 如果兩個特徵值其中一個很大，另一個很小，表示此點與附近的關係為邊界(Edge)，也不是我們想要的情況(我們想要的為角落)。
- (3) 如果兩個特徵值都很大，表示此點與附近的關係為角落(Corner)。

會有上述的特質是因為我們都知道矩陣的特徵值代表它梯度(Gradient)的大小，所以當特徵值越大表示梯度越大，但是當只有一個特徵值很大，另一個很小時，表示只有一個方向的梯度(Gradient)很大，另一個很小，很明顯地此種狀況為邊界(Edge)。相對地，當兩個特徵值都很大時，表示兩個方向的梯度(Gradient)都很大，這就是我們所想要的角落點(Corner)。在此因為求特徵值需要花費較多的運算樣，[23]提供一個代替的方法，假設  $\lambda_1, \lambda_2$  為矩陣A的兩個特徵值，則我們只要去計算式 2-2 的反應函數(Response Function)即可知道此點是Corner的可能性有多大:

$$R = \lambda_1 \lambda_2 - k(\lambda_1 + \lambda_2)^2 = \det(A) - k \times \text{trace}^2(A) \text{-----}(\text{式})2-2$$

其中根據文獻參考的經驗法則， $k$  在 0.04~0.15 之間都可以有彈性的空間，不同

於式 2-1 的運算方法，在計算式 2-2 當中，只須計算加減法以及乘法，因此運算量將會比式 2-1 減少許多。圖 2-2.2、圖 2-2.3 即為一個 Corner Detector 的例子，藍色圓圈即為 Corner。但是從式 2-1 中或從圖 2-2.2、圖 2-2.3 我們不難發現  $I_x, I_y, I_x \times I_y, I_x^2, I_y^2$  都是由每個 Pixel 周圍 3\*3 大小範圍的值求得，也就是說此 Corner 找出來只相對於周圍的 3\*3 而言是 Corner，但是對於一張影像而言，3\*3 的視窗大小其實是一個非常小的範圍，以我們的兩種 Pattern Size 分別為 500\*752 和 640\*480 而言，若單就式 2-1 和式 2-2 所擷取出來的特徵點至少會有 4000~6000 個以上的特徵點，數量非常龐大，如圖 2-2.2，只用單純式 2-1 和式 2-2 的擷取特徵點法，左邊影像有高達 5018 個特徵點，右邊影像也有 4972 個特徵點，數量非常龐大，圖 2-2.3 也有類似的結果，但這些點的確都是我們所要的 Corner 特徵點，只是它所相對應的範圍只在周圍的 3\*3 大小而已。

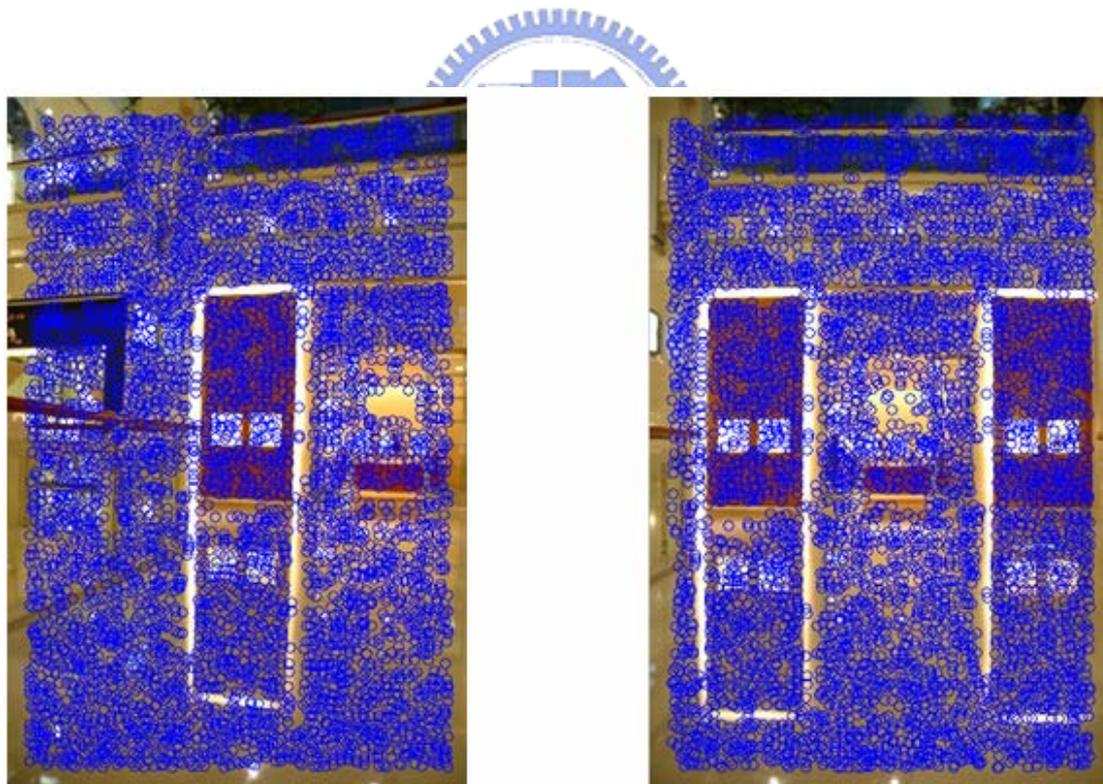


圖 2-2.2 只取一般 Corner 範例一，左邊影像有 5018 個特徵點，右邊影像有 4972 個特徵點

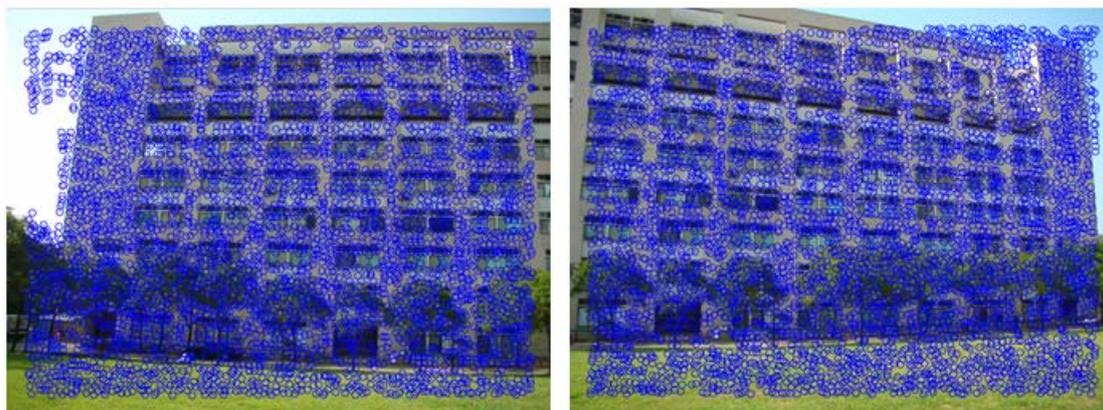


圖 2-2.3 只取一般 Corner 範例二

若是要用這些特徵點來接著執行接下來 AutoStitch 的步驟仍然是可行的，最後仍然會縫合成功，但是可想而知的在 Matching 部分和 RANSAC 部分，其運算量一定會非常地龐大進而拖垮了整個演算法的執行效率。因此通常會在 Response Function  $R$  定下一個臨界值  $T$ ，比  $T$  大的值就令為最後的特徵點，比  $T$  小的值就剔除掉，或是直接定下另一種臨界值  $n_i$ ，取 Response Function  $R$  裡面前  $n_i$  強的特徵點， $n_i$  由使用者自行決定，不同的  $n_i$  有不同的效果，圖 2-2.4 分別顯示  $n_i=200$ 、 $350$ 、 $500$  的效果，我們可以發現  $n_i=200$  的時候點數較少但是卻是比較強健的 Corner 點， $n_i=500$  的時候點數較多但是卻是比較沒這麼強健的 Corner 點，整體來說  $n_i$  取幾點並沒有一定的好壞，取 200 點雖然比較強健而且點數少，但是卻不能保證取 200 點到最後 match 和 RANSAC 之後點數還足夠可以縫合，取



(a)  $n_i=200$



(b)  $n_i=350$



(c)  $n_i=500$

圖 2-2.4  $n_i$  分別取(a)200 (b)350 (c)500 的結果

500 點雖然點數較多且較為不強健，但卻經過 match 和 RANSAC 之後比 200 有保障足夠的點數可以成功縫合，因此取  $n_i$  幾點並沒有一定的好壞，但為了保障最後要有一定足夠的點數可以縫合，[27]使用  $n_i=500$  來作臨界值，即取 Response Function R 裡面前 500 名的點進來，如圖 2-2.4(c)、圖 2-2.5、圖 2-2.6，我們會發現這幾張影像不同於圖 2-2.2、圖 2-2.3 當中的特徵點是所有  $3 \times 3$  大小的 Corner 都放進去，所以點數特別多，對於之後的演算法執行效率上非常不好，但圖 2-2.4(c)、圖 2-2.5、圖 2-2.6 只取這些 Response Function R 裡面特徵點當中前  $n_i=500$  名的，對之後的演算法執行效率有非常大的幫助。



圖 2-2.5 Corner Detector 顯示圖，場景為國立交通大學運動場



圖 2-2.6 Corner Detector 顯示圖，場景為國立交通大學運動場和宿舍

### 2-2.3 適應性非最大化壓縮法 (Adaptive Non-Maximal Suppression)

在使用影像縫合(AutoStitch)於相機校正的演算法中，運算消耗量與特徵點的多寡是呈現超線性(superlinear)的關係，因此有效地限制特徵點的數量是非常重要的，在 2-2.2 當中提到使用固定常數  $n_i=500$  來限制特徵點個數，但即便如此，我們還是需要考量到這種方法只考慮到周圍  $3 \times 3$  視窗大小的範圍(Search Range)，我們希望達到每個特徵點(Feature Points)可以影響的範圍達到比  $3 \times 3$  更大並且特徵點可以儘可能地平均分散在整張影像上，此特性非常重要，因為在縫合兩張影像的過程當中，可能兩張影像的重疊部分(overlap)非常少，或是剛好沒有特徵點存在於兩張影像重疊的部份，則縫合的結果會大大的受到影響，此部分在 3-1.3 我們會提出一個例子作為驗證，因為這樣的需求，[27]提出了適應性非最大化壓縮法(Adaptive Non-Maximal Suppression)簡稱 ANMS，ANMS 即是希望提出一套強健的策略，使得特徵點可以被壓縮在一定的數量，並且所擷取出來的特徵點也可以平均分散在整張影像當中。圖 2-2.7 即是一個例子，(a)(b)兩個圖使用 the highest corner strength 法，分別取所有候選特徵點(candidate points)當中反應函數(Response Function) $R$  的前 250 強和 500 強的 corner 點進來，(c)(d)使用 Adaptive Non-Maximal Suppression(ANMS)法，取點的數量同樣是分別取 250 個點和 500 個點，我們會發現(c)(d)的特徵點(Feature Points)比起(a)(b)的特徵點(Feature Points)在整張影像上的分佈均勻多了，這是一個非常好的特性，如圖 2-2.7(a)，我們可以清楚地觀察到大部分的特徵點(Feature Points)都集中在上半部份，下半部份特徵點非常少，因此若有另一張縫合影像的重疊部分(overlap)是與圖 2-2.7(a)的下半部分重疊，則這兩張圖的匹配點 Matching Points 會非常的少，因為下半部份存在的特徵點本來就很少，因此能夠被 Matching 到的點當然很少，如此的情況非常不利於我們的影像縫合。但是同樣的圖若換成是圖 2-2.7(c)，圖 2-2.7(c)或(d)當中，因為沒有此種情況，



(a) Strongest 250



(b) Strongest 500



(c) ANMS 250



(d) ANMS 500

圖 2-2.7 適應性非最大化壓縮法(ANMS)範例一

而且所有的特徵點較為均勻分佈，因此若有另一張縫合影像的重疊部分(overlap)是與圖 2-2.7(c)的下半部分重疊也不會造成太大的影像，因為本身下半部份就有許多的特徵點可以提供之後作 Matching 和縫合的動作，因此能有像圖 2-2.7(c)(d)這樣均勻分佈而且特徵點數目又控制在所希望的大小是一個非常不錯的特性。

接著我們要介紹 ANMS 在實作上的方法，我們知道若只用式 2-1 和式 2-2 求出來的特徵點只考量旁邊周圍  $3 \times 3$  視窗大小的範圍，若在  $5 \times 5$  的範圍中此點也是最大，則此特徵點的可靠度必然比  $3 \times 3$  的視窗大小更可靠，以此類推，當視窗大小為無限大時，找出來的特徵點必然最為強健，因此在實作上，[27]提出了針對式 2-2 的 Response Function  $R$  裡面的每一點，去比較此點周圍的視窗大小值，並且 Search Range  $r$  從無限大開始往內縮回，當某個點為此 Search Range 的最

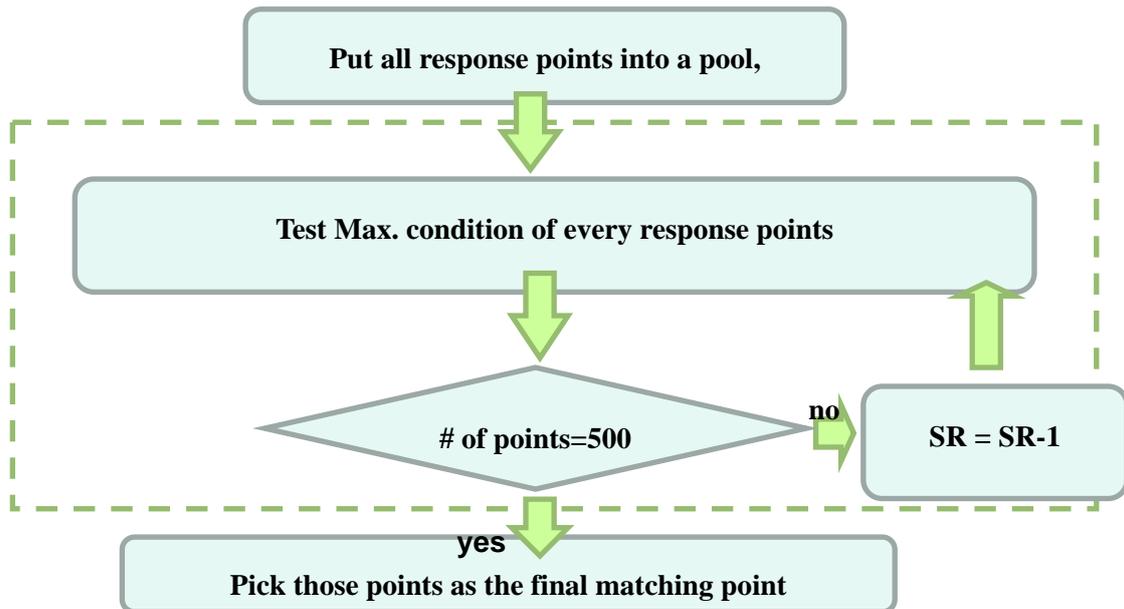


圖 2-2.8 ANMS 流程圖

大值時，就令此點為 ANMS 的特徵點。當 Search Range 從無限大往內縮回時，特徵點數目也會相對增加，當特徵點數目增加到我們想要的數目  $n_i$  時（[27] 令  $n_i=500$ ），ANMS 才停止擷取特徵點，這樣的推論是非常合理的，因為一旦在某個 Search Range  $r$  時出現了某個特徵點，那麼此點必定會繼續出現在  $r' < r$  的情形之下。圖 2-2.8 為 ANMS 的流程圖，圖 2-2.9 和圖 2-2.10 為 The highest corner strength 法與 ANMS 的兩個比較圖範例，由圖我們也可以發現 ANMS 比起 The highest corner strength 法，特徵點的分佈均勻多了。

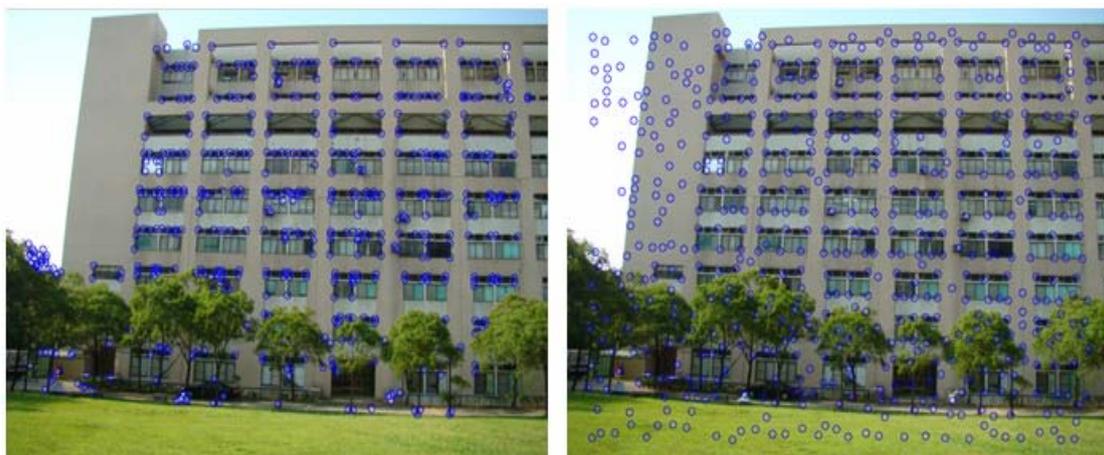


圖 2-2.9 The highest corner strength 法與 ANMS 比較圖範例一

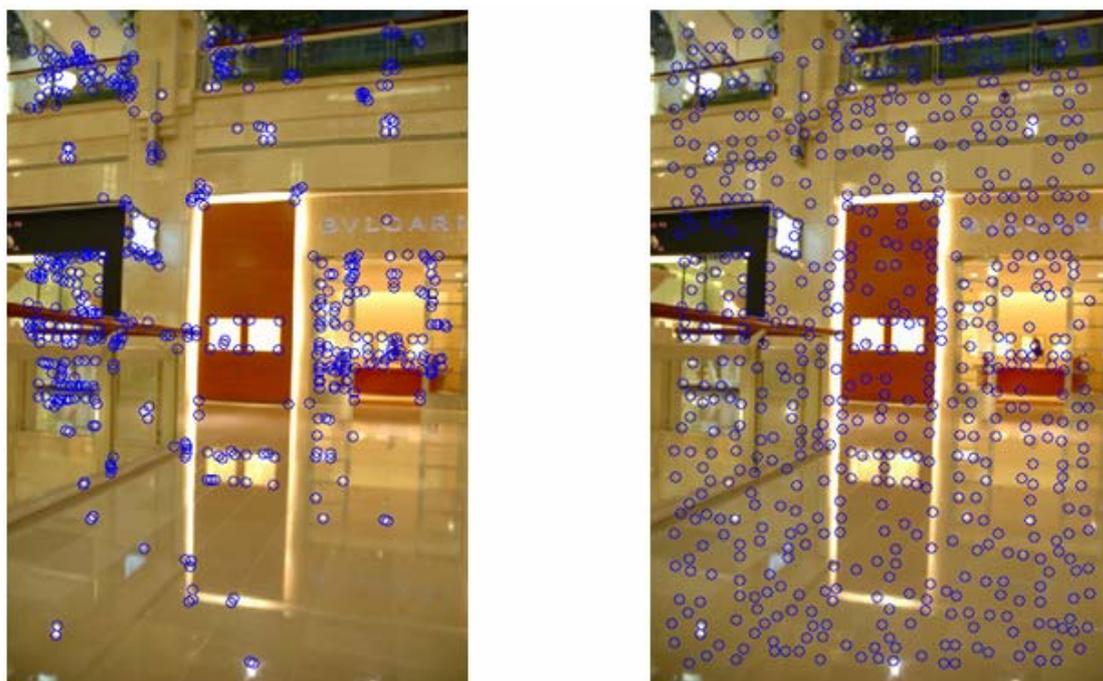


圖 2-2.10 The highest corner strength 法與 ANMS 比較圖範例二

## 2-3 特徵點匹配 (Feature Matching)

### 2-3.1 最小平方差 SSD (Sum of Squared Differences)

當擷取完特徵點 (Feature points) 之後，接下來需要把兩張圖的特徵點彼此之間做匹配 (Matching)，看看 A 圖裡面的所有特徵點在 B 圖裡面是對應到哪些點，最簡單的方法是直接去計算兩個特徵點周圍的一個小範圍的 SSD (Sum of Squared Differences)，並定下一個臨界值 (Threshold) 為門檻，因為當兩個點彼此是對應點，則其附近的 Pixel 值會非常接近，所以當 SSD 越小，自然表示兩個 A 圖的  $(x_1, y_1)$  點很有可能是對應到 B 圖的  $(x_2, y_2)$  點，如圖 2-3.1 和式 2-3，此方法又有人稱為 Nearest Neighbor 法或 Closest Neighbor 法，這種方法最為簡單和直觀，但卻有許多缺點，在稍後我們會提到。

$$S = \sum_{i=-2}^2 \sum_{j=-2}^2 [F_1(x_1 + i, y_1 + j) - F_2(x_2 + i, y_2 + j)]^2 \quad \text{--(式)2-3}$$

$F1(x1-2,y1-2)$	$F1(x1-1, y1-2)$	$F1(x1,y1-2)$	$F1(x1+1,y1-2)$	$F1(x1+2,y1-2)$
$F1(x1-2,y1-1)$	$F1(x1-1, y1-1)$	$F1(x1,y1-1)$	$F1(x1+1,y1-1)$	$F1(x1+2,y1-1)$
$F1(x1-2,y1)$	$F1(x1-1,y1)$	$F1(x1,y1)$	$F1(x1+1,y1)$	$F1(x1+2,y1)$
$F1(x1-2,y1+1)$	$F1(x1-1,y1+1)$	$F1(x1,y1+1)$	$F1(x1+1,y1+1)$	$F1(x1+2,y1+1)$
$F1(x1-2,y1+2)$	$F1(x1-1,y1+2)$	$F1(x1,y1+2)$	$F1(x1+1,y1+2)$	$F1(x1+2,y1+2)$

$F2(x2-2,y2-2)$	$F2(x2-1, y2-2)$	$F2(x2,y2-2)$	$F2(x2+1,y2-2)$	$F2(x2+2,y2-2)$
$F2(x2-2,y2-1)$	$F2(x2-1, y2-1)$	$F2(x2,y2-1)$	$F2(x2+1,y2-1)$	$F2(x2+2,y2-1)$
$F2(x2-2,y2)$	$F2(x2-1,y2)$	$F2(x2,y2)$	$F2(x2+1,y2)$	$F2(x2+2,y2)$
$F2(x2-2,y2+1)$	$F2(x2-1,y2+1)$	$F2(x2,y2+1)$	$F2(x2+1,y2+1)$	$F2(x2+2,y2+1)$
$F2(x2-2,y2+2)$	$F2(x2-1,y2+2)$	$F2(x2,y2+2)$	$F2(x2+1,y2+2)$	$F2(x2+2,y2+2)$

圖 2-3.1 SSD(Sum of Squared Differences)示意圖

### 2-3.2 次近鄰居法(Second-Closest Neighbor)

在 2-3.1 的 SSD 方法(或稱 Closest Neighbor 法)是一種最直觀最簡便的方法，但是在影像中的特徵點可能因為背景的模糊或是對應點根本不在所對應的影像當中，所以理論上來說應當不會產生好的 Match 在所對應的影像上，因此 SSD 並不是很好的方法，[28]提出了次近鄰居法(Second-Closest Neighbor)來有效地剔除掉這種不該有好 Match 的特徵點。

次近鄰居法(Second-Closest Neighbor)的構想是每一對好的匹配點(Matching Points)相對於其他對匹配點，都應該要夠獨特並且有區別性(discriminative)，因此我們對於某個特徵點 Matching 時，除了計算最近的距離(Closest Neighbor)  $d1$ ，也去計算次近的距離(Second-Closest Neighbor)  $d2$ ，若是這兩個的比例  $D=d1/d2$  大於 0.8，則表示此組的 Matching 太過類似( $d1/d2$  接近 1)，則我們把這組 Match 剔除掉並稱他們為錯誤的 Matching Points。相反

地若是  $D=d1/d2$  小於 0.8，則表示此組的 Matching 夠獨特有區別性 (discriminative)，我們稱這兩個特徵點為一組好的 Matching Points。圖 2-3.2 為次近鄰居法 (Second-Closest Neighbor) 的流程圖。圖 2-3.3 為使用 40000 的特徵點 (Feature points) 來作次近鄰居法，水平軸為  $d1/d2$  的比例，垂直軸為 false match 的機率，由圖我們可以發現當  $d1/d2$  比例很低時，表示此組的 Matching 夠獨特有區別性，圖形也顯示 false match 的比例也非常低，但是當  $d1/d2$  比例很高時 (大於 0.8)，表示此組的 Matching 太過類似，圖形也顯示 false match 的比例也非常高，應該要剔除掉，[28] 聲稱此種作法可替除掉 90% 的 false match，但只會剔除掉 5% 的 correct match，比起 SSD 作法，算是相當值得。圖 2-3.4、圖 2-3.5 為次近鄰居法的範例，(a)-(b) 為特徵點擷取，(c)-(d) 為特徵點匹配，我們會發現 (c)(d) 的許多點都成功對應到了。

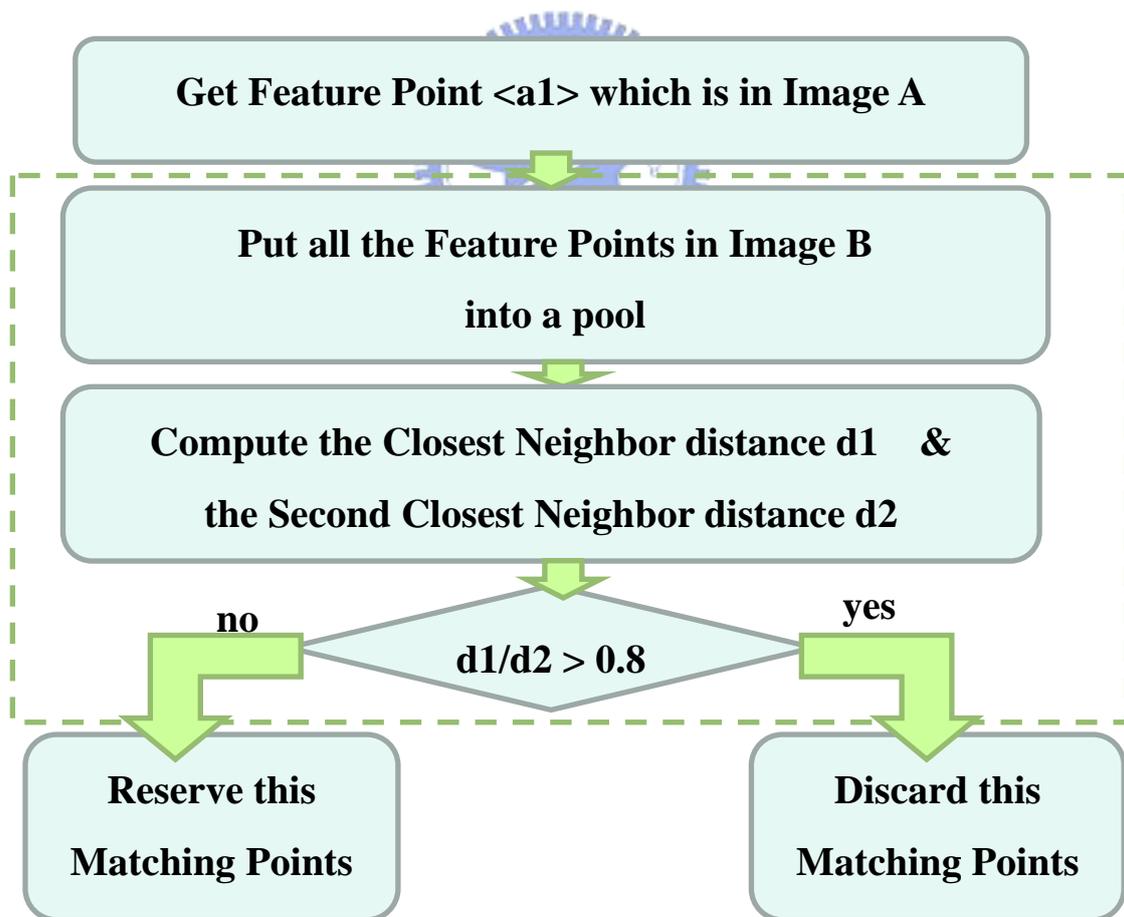


圖 2-3.2 次近鄰居法 (Second-Closest Neighbor) 的流程

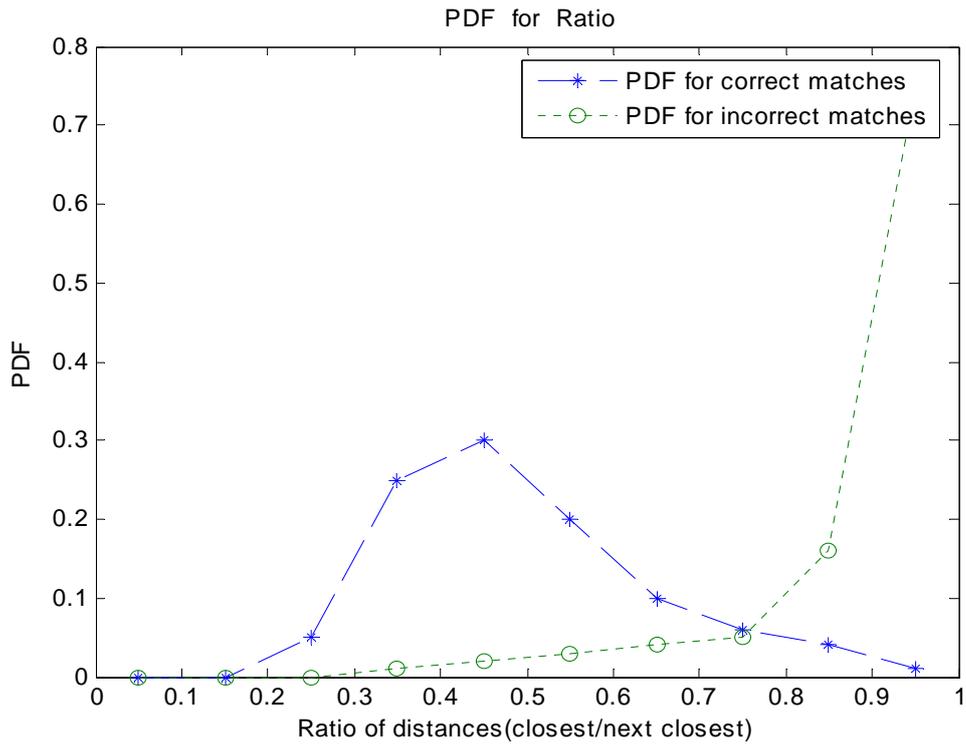


圖 2-3.3 次近鄰居法(Second-Closest Neighbor)的成功機率圖表



(a)

(b)



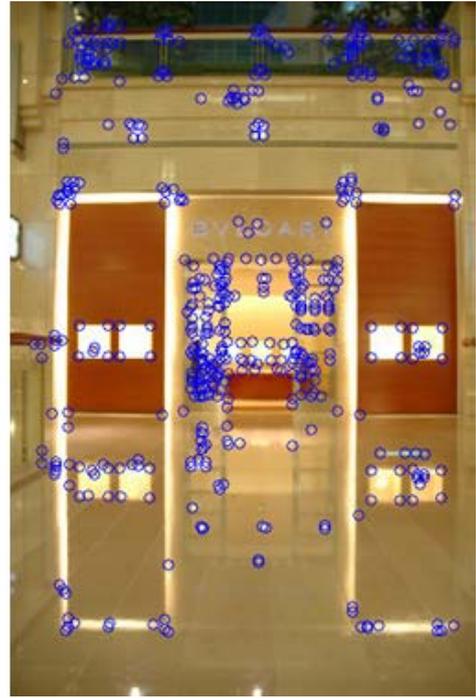
(c)

(d)

圖 2-3.4 特徵點擷取與特徵點匹配的範例一



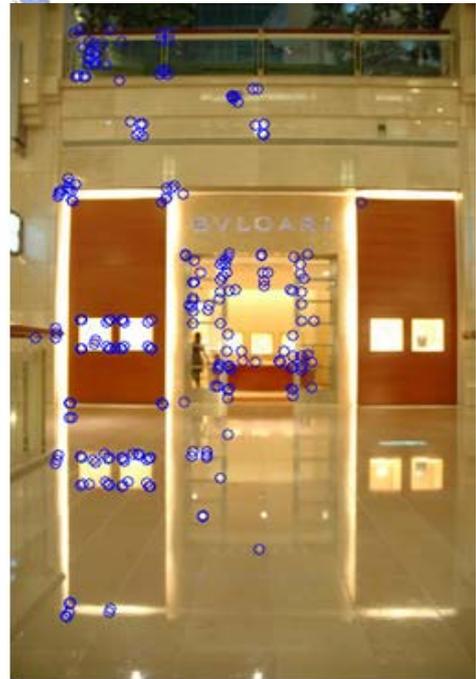
(a)



(b)



(c)



(d)

圖 2-3.5 特徵點擷取與特徵點匹配的範例二

## 2-4 剔除掉錯誤的匹配點(Delete The False Match)

經過特徵點匹配(Feature Matching)後，我們會發現有許多的特徵點都彼此對應到了，而且大部分皆為正確的匹配(correct Match)，但是由圖 2-3.4 和圖 2-3.5 當中我們也不難發現仍然有少數的特徵點彼此的對應是錯誤的(即 False Match)，所以在特徵點匹配之後的下一個動作，就是要把錯誤的匹配點剔除掉(Delete The False Match)，[33]-[34]提出了幾種方法來建立一個正確的模型，再利用此模型來剔除掉錯誤的資料(data)，這些方法不僅可以適用於影像上，也可以適用於許多統計資料的處理。而在[36]-[37]都使用[33]的作法來建立正確的模型以剔除掉錯誤的特徵點。[33]使用隨機取樣程序(RANdOm SAmpLe Consensus)來建立模型剔除掉錯誤的資料，簡稱 RANSAC。RANSAC(RANdOm SAmpLe Consensus)為一套使用隨機取樣來估計影像轉換參數的強健估計機制，在一般的例子裡，RANSAC 會隨機取  $n$  個 data 來建立一個模型，並且用剩餘的  $(N-n)$  個 data 來測驗此模型是否正確，但是並非每次取的  $n$  個 data 都是好的 data，自然地經過測試過後會顯示不是一個好的模型，所以通常 RANSAC 都要執行一定的次數才能擷取出最後我們所要的正確 data。至於要重複多少次，則完全看使用者和問題的不同而定，當然，RANSAC 是在大部分的資料都是正確只有少部分資料是錯誤的情況下才能使用，但是在 2-3 我們已經經過初步的特徵點匹配(Feature Matching)來淘汰大部分錯誤的匹配點，因此剩下來的匹配點大部分都是正確的(由圖圖 2-3.4、圖 2-3.5 我們也可以看到)，因此 RANSAC 在這個地方是絕對適用的，以下為 RANASC 步驟的流程：

- (1)隨機地取  $n$  個樣本資料進來。
- (2)使用這  $n$  個樣本資料來建立一個模型  $\Phi$ 。
- (3)把剩下的  $(N-n)$  個樣本資料代入  $\Phi$  模型，測試此模型的建立是否正確。
- (4)重複執行  $K$  次。
- (5)把最大聯集的內部點 (inlier points) 取出來做最後的結果。

由上面的RANSAC 流程我們並不難發現RANSAC 流程當中有許多參數都是由使用者自己決定的，例如：如何定義樣本資料的數目  $n$ 、如何定義重複執行次數、如何定義何為內部點 (inlier points)，這些都沒有一定的答案，而是根據我們遇到的問題來決定。[36]-[37]當中使用樣本資料數目  $n=4$ ，重覆執行次數  $k=500$ ，定義內部點為 500 次當中最大對應成功次數的樣本資料，之所以會這樣定義的原因，我們先看式 2-4 和式 2-5。式 2-4 為兩個 2D 座標互相轉換的關係矩陣，式 2-5 為是 2-4 的矩陣型式。

$$P' = HP \text{ ----- (式)2-4}$$

$$\begin{bmatrix} w x' \\ w y' \\ w \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \text{ -----(式)2-5}$$

將式 2-5 做簡單的展開，如式 2-6。

$$\begin{aligned} w x' &= a x + b y + c \cdots (1) \\ w y' &= d x + e y + f \cdots (2) \\ w &= g x + h y + 1 \cdots (3) \end{aligned} \text{ ---- (式)2-6}$$

將式 2-6 的(1)-(3) $\times x'$  和(2)-(3) $\times y'$  得到式 2-7

$$\begin{aligned} a x + b y + c - g x x' - h y x' - x' &= 0 \\ d x + e y + f - g x y' - h y y' - y' &= 0 \end{aligned} \text{ -----(式)2-7}$$

再將式 2-7 展開成矩陣的型式得到式 2-8，

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1x_1' & -y_1x_1' & -x_1' \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1y_1' & -y_1y_1' & -y_1' \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -x_2x_2' & -y_2x_2' & -x_2' \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -x_2y_2' & -y_2y_2' & -y_2' \\ \cdot & \cdot \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \\ g \\ h \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad \text{-(式)2-8}$$

注意式 2-8 是可以往下無限延伸的，由式 2-8 我們可以發現有 a~h 總共八個參數，每個樣本資料點可產生兩個方程式，所以至少需要 4 個樣本資料點以產生八個方程式，以提供我們解出這 8 個未知參數，這也就是為什麼 [36]-[37] 使用的 RANSAC 每次取 4 個樣本資料點來建立模型，因為資料點取越多越容易取到錯誤的點，以致於建立錯誤的模型，但是又必須取一定的點數來解答八個未知參數，所以使用 n=4。並且 [36]-[37] 也用簡單的機率模型來粗略證明當重複執行次數 k=500 時，模型錯誤的機率已經小於  $10^{-14}$ ，因此使用 k=500，最後在內部點 (inlier points) 的部分求出來之後，再把所有內部點代入式 2-8 並使用 [35] direct linear transformation (DLT) 以得到最後最精確的模型 H。圖 2-4.1、圖 2-4.2 為 RANSAC 的兩個範例，藍色圈為經過初步 Matching 的結果，紅色又為經過 RANSAC 的結果，我們會發現所有 Match 都正確對應到了。圖 2-4.3 為 RANSAC 的流程圖。



圖 2-4.1 RANSAC 範例一，圖為交通大學工程五館

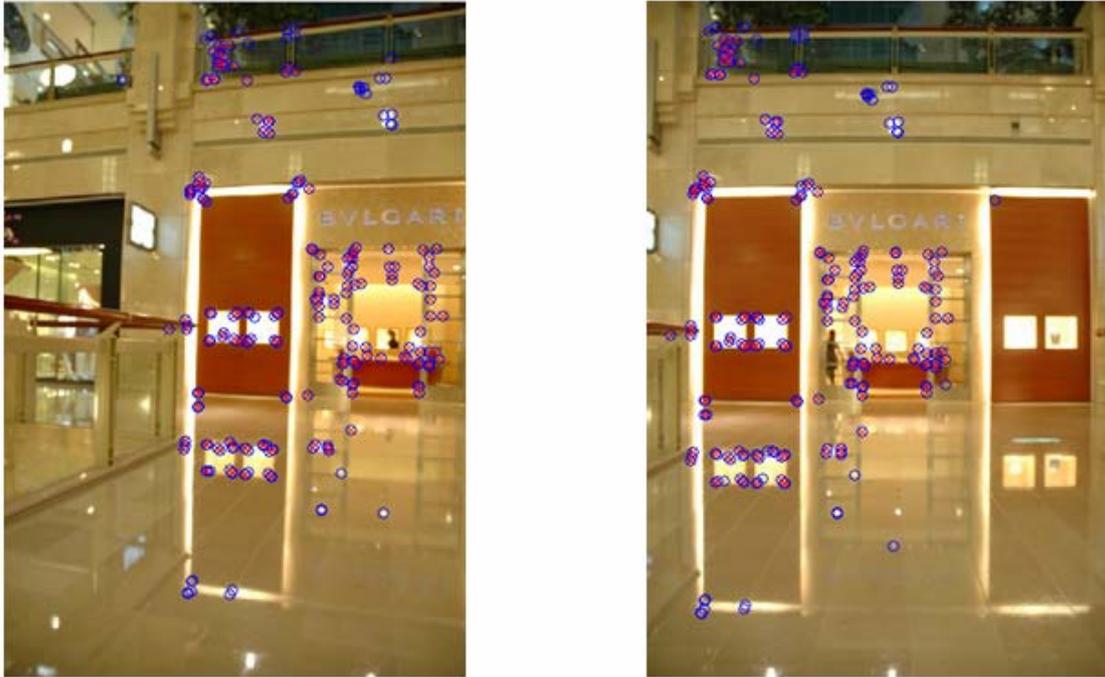


圖 2-4.2 RANSAC 範例二，圖為台北 101

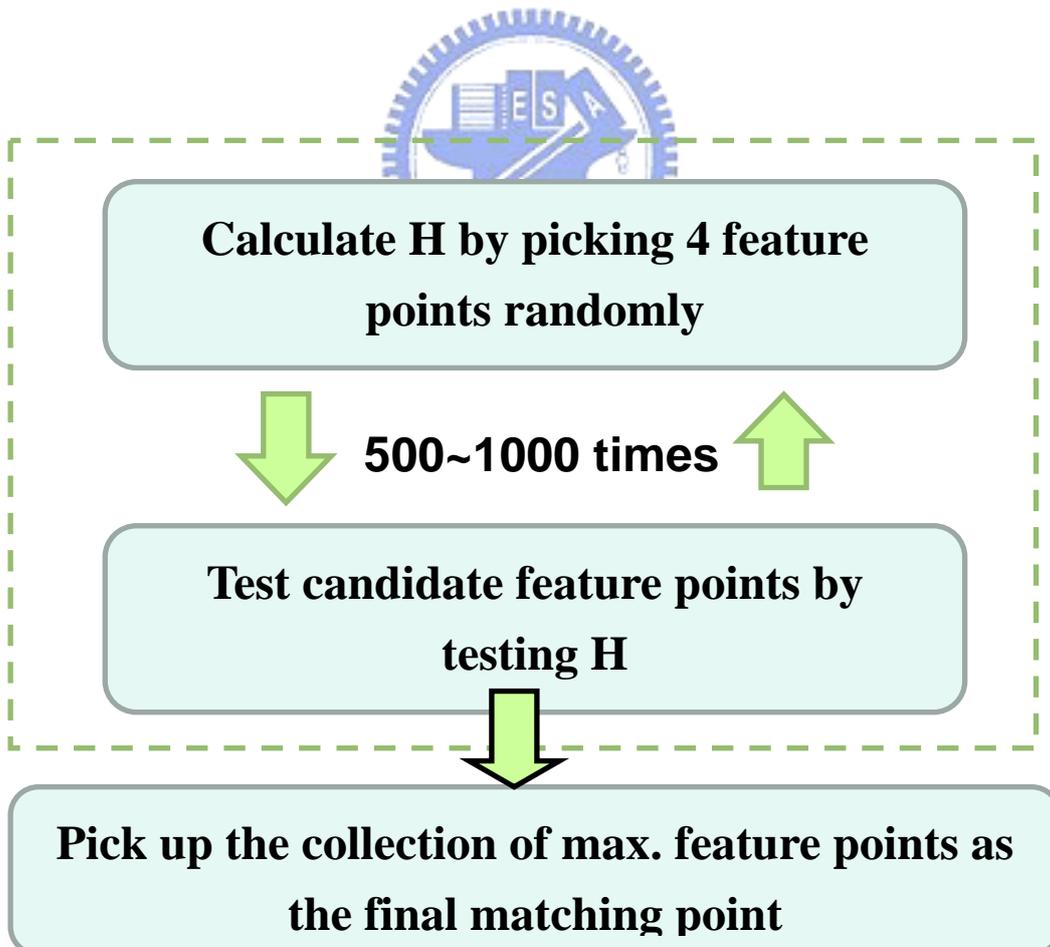


圖 2-4.3 RANSAC(RANDom SAmple Consensus) flow chart

## 2-5 歪斜影像(Warp Image)

在 2-4 經過 RANSAC 剔除掉錯誤的匹配點，同時也建立了正確的模型  $H$  之後，下一步就是要使影像做歪斜(Warp Image)，在 1-2 曾經提到我們所使用的前提是全自動絕對座標縫合影像(Automatically Absolute Coordinated Image Stithing)，希望是達到與真實世界(ground truth)內容一樣的結果，所以相對位置也必須一樣，假設我們在 Sensor 皆為水平移動的前提下(本篇論文皆如此假設)，要縫合兩張影像一定要把其中一張影像做歪斜，因為他們彼此的關係不是水平移動就可縫合的，必定有某種投影的關係，我們稱為透視投影(Perspective Projection)，其關係是即為式 2-5，因此我們只要把需要做歪斜的影像座標經過式 2-5 即可得到新的歪斜影像，如圖 2-5.1、圖 2-5.1，左邊的影像經過  $H$  做歪斜即為右邊的影像。



圖 2-5.1 歪斜影像範例一



圖 2-5.2 歪斜影像範例二

到這裡我們也可以從另一個角度來看如果沒有作 RANSAC 會有什麼樣的結果，圖 2-5.3 和圖 2-5.4 顯示沒有作 RANSAC 步驟的歪斜影像(Warp Image)和縫合影像，我們可以看到這兩張影像完全失真了，會有這樣的結果是因為沒有使用 RANSAC 來剔除掉錯誤的匹配點，進而使用了錯誤的匹配點求得錯誤的模型 H，自然地歪斜影像和縫合結果就不正確了。



圖 2-5.3 沒有使用 RANSAC 的歪斜影像



圖 2-5.4 沒有使用 RANSAC 的縫合影像

## 2-6 混合影像(Blending)

經過 2-5 歪斜影像的步驟後，兩張圖的相對位置已經吻合(見圖 2-6.1)，最後一步就是要把兩張影像作混合(Blending)，理想情況若是在拍攝影像時，光源皆沒有任何改變，並且每一張圖的受光度都一樣(表示每張圖的明暗程度都一樣)，理論上應該是可以直接將兩張圖放在一起而不用作任何的動作，但事實上並非如此，在擷取影像的過程當中，即使是 multi-sensor 並且同步擷取影像，

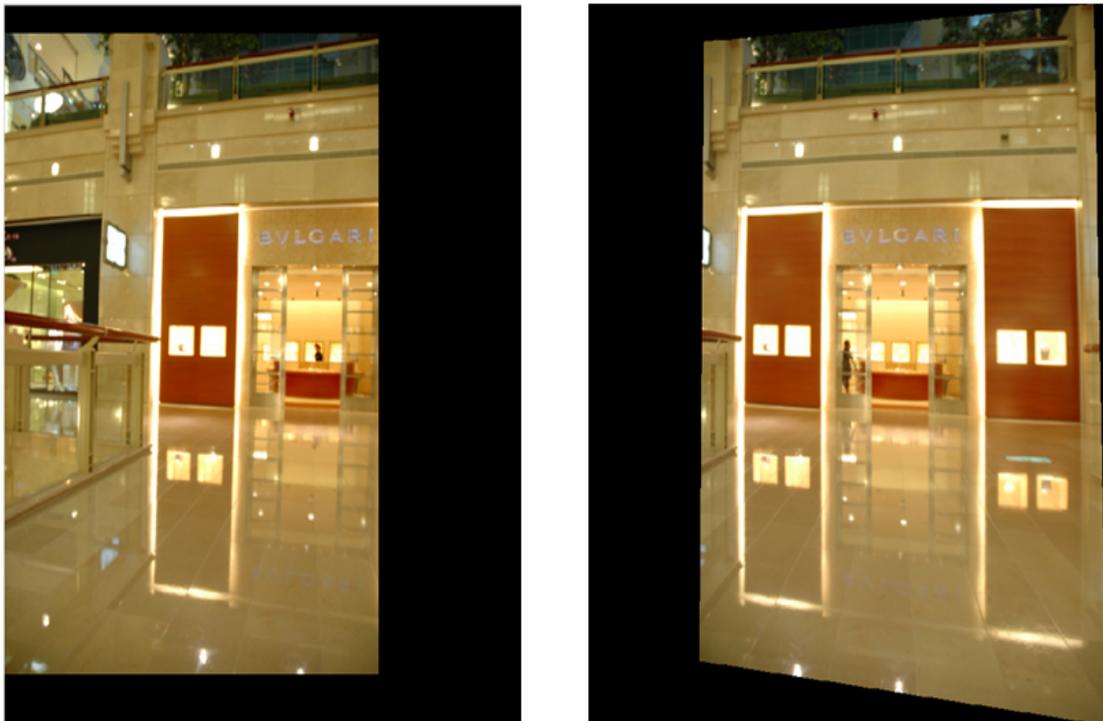


圖 2-6.1 原始影像與歪斜影像的相對位置已經吻合

仍然會因為角度的不同而在明暗度上有些落差，如圖 2-6.1 我們就可大約觀察出右圖比左圖稍微亮一點，若是沒有經過特別的混合策略就縫合，會造成影像的失真或是我們不希望看到的邊界層次的情況產生，如圖 2-6.2、圖 2-6.2。我們可以發現因為影像受光度的不同，若是沒有聽過特殊的混合策略來縫合，所會造成的邊界層次失真情況產生，因此有一個好的混合策略也是非常重要的。

在此我們使用常見的線性混合(Linear Blending)來做縫合，[39]提出的線性混合(Linear Blending)理論使用距離的遠近為權重函數(Weight function)來解決邊界線不均勻的問題，使兩張影像在縫合時可以有平均的動作，圖 2-6.4、



圖 2-6.2 沒有用特殊 Blending strategy 呈現的失真影像範例一



圖 2-6.3 沒有用特殊 Blending strategy 呈現的失真影像範例二

為其示意圖， $H_l$  為左邊影像的權重， $H_r$  為右邊影像的權重，越靠近本身影像的中心則權重越高，越靠近另一張影像則權重越低，圖 2-6.4 中間重疊的部份都介於 0~1 之間，是為方便之後作平均的動作，圖 2-6.5、圖 2-6.6 為兩張圖分別對應的權重(Weighting)，我們會發現兩張權重圖是不一樣的，歪斜影像的權重圖也是歪斜的，原始影像的權重圖卻是長方形的，原因即是因為他們是分別由不同的圖產生的權重函數，當 Pixel 的距離離影像中心點越近則權重越大，距離越遠則權重越小，且所有權重皆介於 0~1 之間，因此產生了圖 2-6.5、圖 2-6.6，我們也可發現兩張權重圖越往中心點越亮，此即表示他們的權重越大。

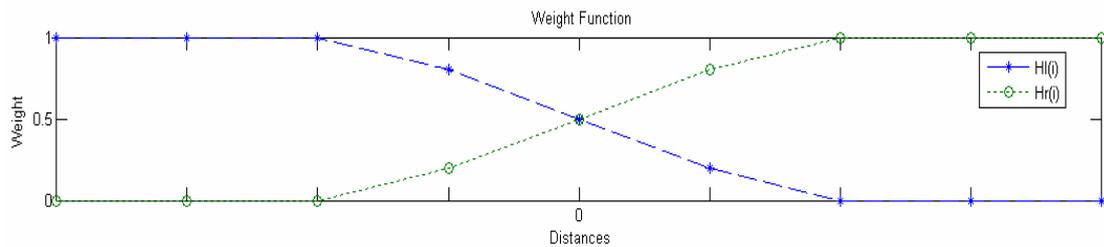


圖 2-6.4 Weight Function 示意圖

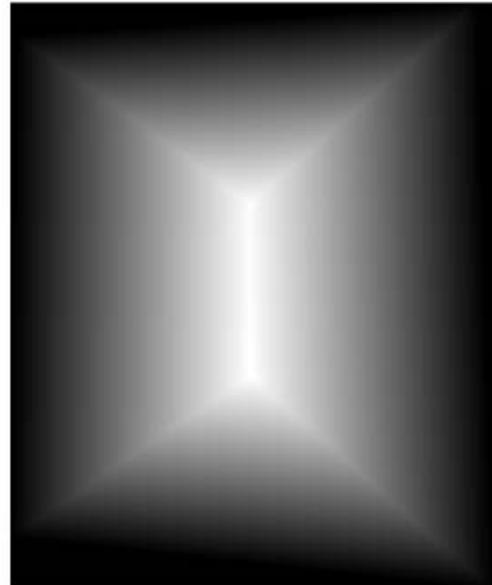


圖 2-6.5 歪斜影像對應的權重函數



圖 2-6.6 原始影像對應的權重函數

產生權重函數之後，最後只要利用簡單的平均關係即可將影像來作縫合，根據式 2-9 即可得到每個重疊地方的 Pixel 值，進而消除圖 2-6.2、圖 2-6.3 因為光線強度不同所造成不均勻失真的效果。

$$I^{linear} = \frac{\sum_{i=1}^n I^i \times W^i}{\sum_{i=1}^n W^i} \text{-----(式)2-9}$$

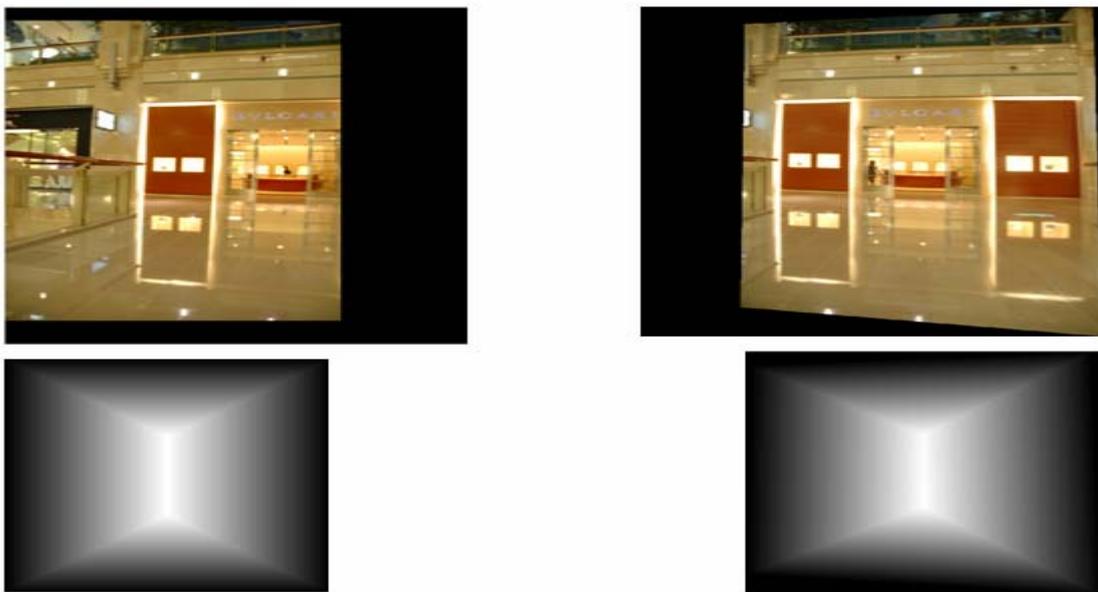


圖 2-6.7 兩張圖的相對位置與權重函數關係



圖 2-6.8 經過 Blending Strategy 所產生的縫合影像

圖 2-6.7 為兩張圖的相對位置與權重函數關係，由圖我們可以看到不僅兩張圖相對位置已經吻合，其權重函數(Weight Function)也已經確定，因此只要經過式 2-9 即可得到每個重疊地方的 Pixel 值，圖 2-6.8 為經過 Blending Strategy 所產生的最後縫合影像，我們可以明顯地看到它改善了圖 2-6.2 所產生的失真效應。

## 第三章 使用 BUMS 與 PRSC 影像縫合快速演算法設計

---

在第二章介紹了一些關於影像縫合的基本理論之後，接下來本篇論文所要談的，是關於影像縫合演算法在設計或執行過程當中可能會遭遇到的問題，針對此來做說明以及尋求解決的辦法。

### 3-1 Bottom-Up Maxima Selection(BUMS)

#### 3-1.1 傳統適應性非最大化壓縮法之深入討論

在 2-2.3 當中我們介紹了適應性非最大化壓縮法(Adaptive Non-Maximal Suppression)的演算法，本節我們要繼續 ANMS 對於 AutoStitch 的優缺點來做深入的探訪與分析。

讓我們先回顧一下在特徵點擷取(Feature Extraction)的步驟使用 ANMS 的優點，之所以會從 the highest corner strength 進展到 ANMS，主要的原因有兩個：第一、ANMS 所擷取的特徵點比起 the highest corner strength 可以平均分散在整張影像當中，這是一個非常好的特性，因為在縫合兩張影像的過程當中，可能兩張影像的重疊部分(overlap)非常少，或是剛好沒有特徵點存在於兩張影像重疊的部份，則縫合的結果會大大的受到影響。圖 3-1.1(a)(b) 大部分的特徵點都集中在上半部份，下半部份特徵點非常少，因此若有另一張縫合影像的重疊部分(overlap)是與圖 3-1.1(a)的下半部分重疊，則這兩張圖的 Matching Points 會非常的少，因為下半部份存在的特徵點本來就很少，如此的情況非常不利於我們的影像縫合。但是同樣的圖若換成是圖 3-1.1(c)，因為所有的特徵點較為均勻分佈，因此若有另一張縫合影像的重疊部分(overlap)是與圖 3-1.1(c)的下半部分重疊也不會造成太大的影響，因為本身下半部份就有許多的特徵點可以提供之後作 Matching 和縫合的動作，因此能有像圖 3-1.1(c)(d)這樣均勻分



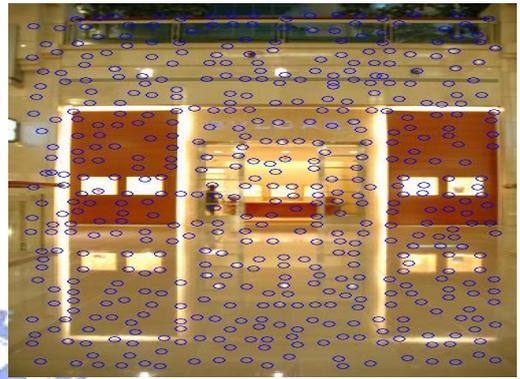
(a) Strongest 250



(b) Strongest 500



(c) ANMS 250



(d) ANMS 500

圖 3-1.1 適應性非最大化壓縮法(ANMS)範例一

佈而且特徵點數目又控制在所希望的大小是一個非常不錯的特性。第二、特徵點個數可以壓制在一定的數目以下。運算消耗量與特徵點的多寡是呈現超線性 (superlinear) 的關係，因此 ANMS 同樣也把特徵點個數壓制在  $n_i=500$ ，使得特徵點數目不至於過多進而拖垮整個演算法的運算效率。因為同時具備這兩個優點，ANMS 在擷取特徵點的使用相當普遍。

接著我們要從另一個角度來看適應性非最大化壓縮法(ANMS)的缺點，在 2-2.3 當中我們有提到 ANMS 為了達到整張影像平均分布的效果，因此把反應函數 Response Function  $R$  的搜尋範圍  $r$  從無限大往內縮回，當某個點為此 Search Range 的最大值時，就令此點為 ANMS 的特徵點。當 Search Range 從無限大往內縮回時，特徵點數目也會相對增加，當特徵點數目增加到我們想要的數目  $n_i$  時 ([27] 令  $n_i=500$ )，ANMS 才停止擷取特徵點。如圖 3-1.2。

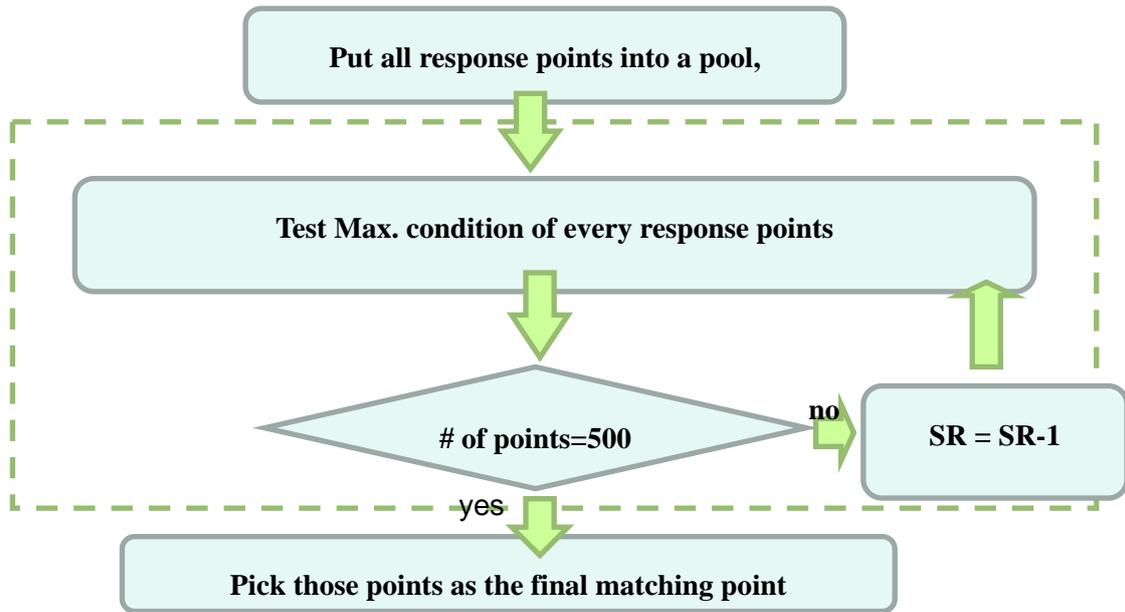
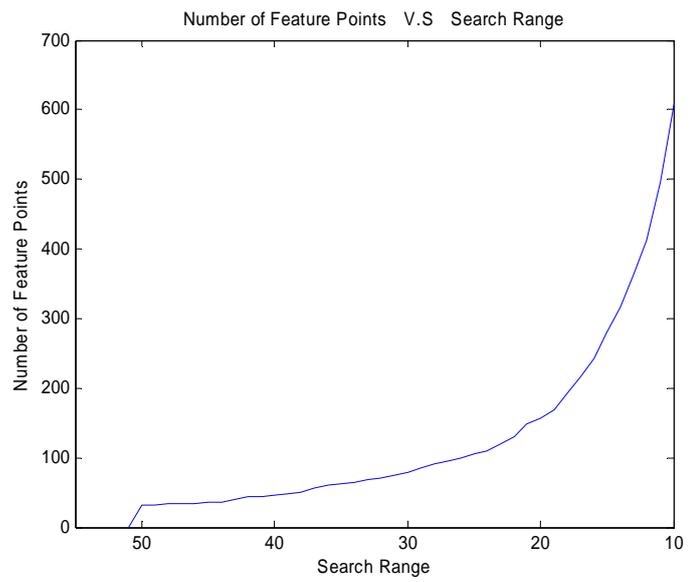


圖 3-1.2 ANMS 流程圖

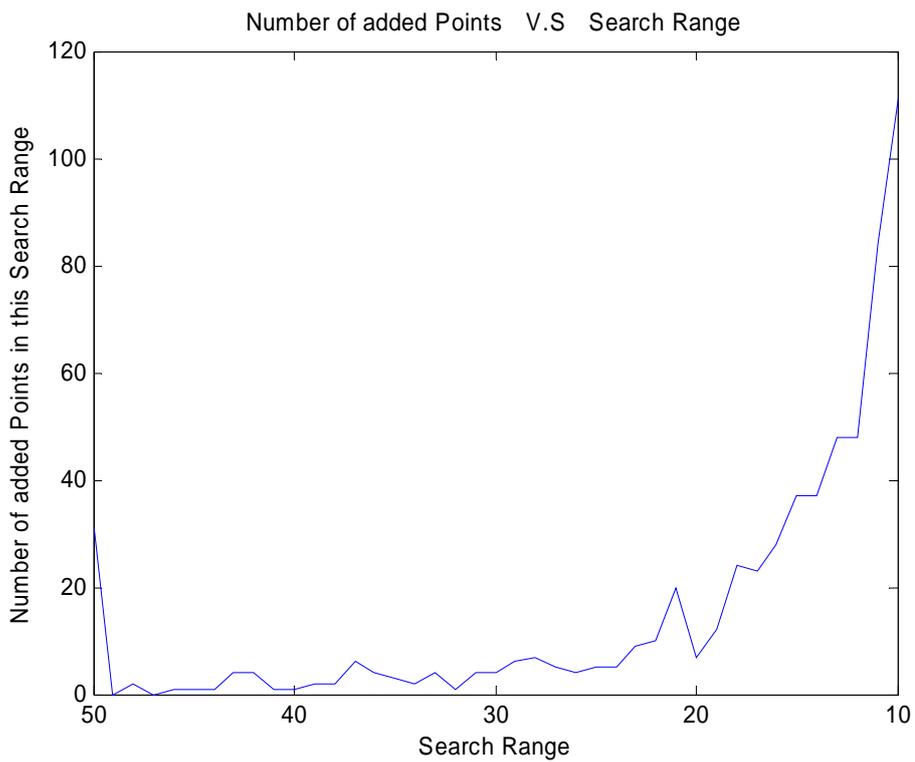
但是在這裡因為 Search Range  $r$  一開始就從無限大搜尋，但是特徵點卻很少，之後慢慢往內縮，特徵點數目才慢慢增加。也就是說一開始 Search Range  $r$  很大時，運算量消耗非常多，但是能擷取出來的特徵點卻非常少（因為  $r$  很大表示限制嚴格，能擷取進來的特徵點較少），這不是一個好的現象，正常來說當演算法某部份消耗很大運算量時，則希望這部份能發揮較大的效果，但是 ANMS 這裡卻剛好相反，運算量消耗非常多，但是能擷取出來的特徵點卻非常少。並且根據統計，整個 AutoStitch 流程當中，特徵點擷取佔最大的運算量，也就是 ANMS 為了達到平均分佈的效果，使得  $r$  從無限大往內縮，但是在這裡卻因此佔了縫合影像流程當中最 Critical 的部份，拖垮了整個影像縫合演算法的效率。如圖 3-1.3，圖 3-1.3(a)顯示 ANMS 的結果，圖 3-1.3(b)顯示 Search Range  $r$  從無限大縮回時，特徵點數量與 Search Range  $r$  的對應關係圖，我們會發現一開始  $r$  很大時，特徵點累積數量很少，到後面  $r$  慢慢縮小到 20 以下，特徵點數量才開始慢慢變多，但是在剛開始因為  $r$  很大，所以要花上相當大的運算量，在運算效率上相當不划算。（在此我們定  $r$ =影像長寬的最小值除以 10 代替  $r$ =無限大，因



(a)



(b)



(c)

圖 3-1.3 ANMS 與 Search Range  $r$  之深入分析範例一

(a)ANMS 的結果

(b)Search Range 從無限大往內縮回時的特徵點擷取數量累積圖

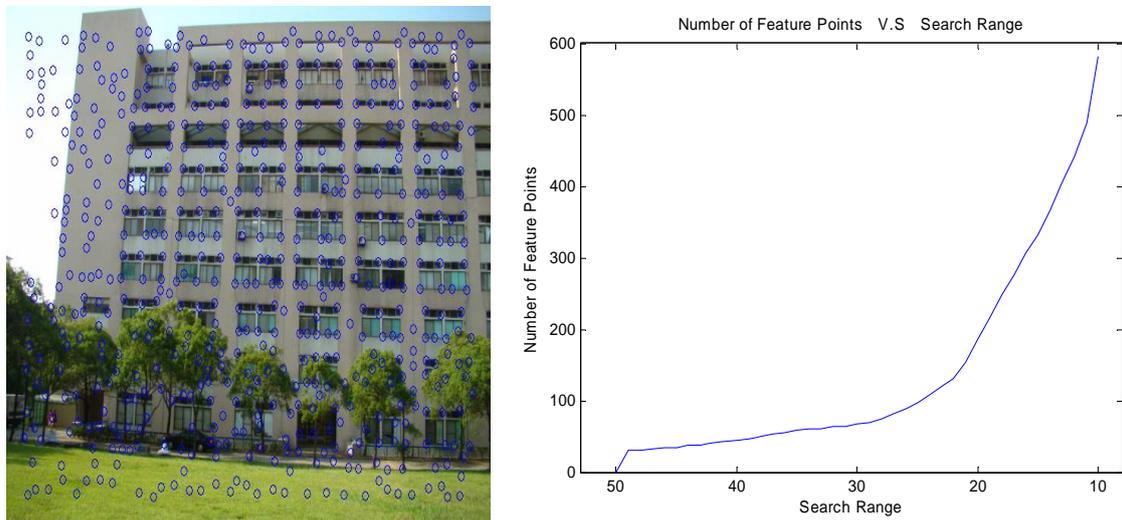
(c)Search Range 從無限大往內縮回時的特徵點增加數量圖

表 3-1.1 SR 從無限大往內縮回時特徵點擷取數量與增加數量統計表範例一

SR	>50	50	49	48	47	46	45	44	43	42	41
Points	0	31	31	33	33	34	35	36	40	44	45
add	31	0	2	0	1	1	1	4	4	1	1
	<b>40</b>	<b>39</b>	<b>38</b>	<b>37</b>	<b>36</b>	<b>35</b>	<b>34</b>	<b>33</b>	<b>32</b>	<b>31</b>	<b>30</b>
	46	48	50	56	60	63	65	69	70	74	78
	2	2	6	4	3	2	4	1	4	4	6
	<b>29</b>	<b>28</b>	<b>27</b>	<b>26</b>	<b>25</b>	<b>24</b>	<b>23</b>	<b>22</b>	<b>21</b>	<b>20</b>	<b>19</b>
	84	91	96	100	105	110	119	129	149	156	168
	7	5	4	5	5	9	10	20	7	12	24
	<b>20</b>	<b>19</b>	<b>18</b>	<b>17</b>	<b>16</b>	<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>10</b>	
	156	168	192	215	243	280	317	365	413	608	
	12	24	23	28	37	37	48	48	84	0	

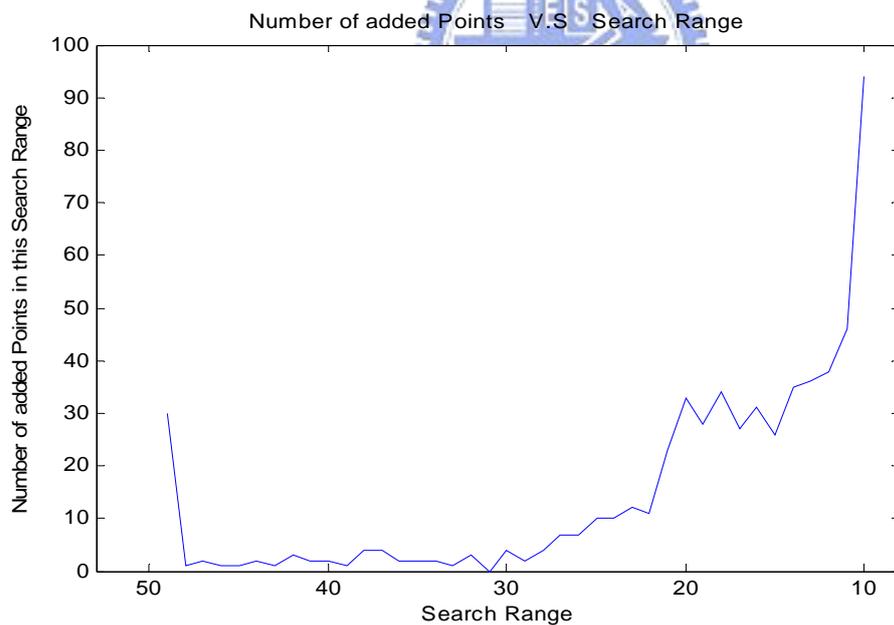
為使用  $r$ =無限大時，真實運算量將會非常驚人，我們將此作為簡化並模擬，另一方面也方便做 Worse Case 的考量)。圖 3-1.3(c)顯示 Search Range  $r$  從無限大縮回時，特徵點增加數量與 Search Range  $r$  的對應關係圖。我們也可以更清楚地觀察到當  $r$  很大時，特徵點增加的數量非常少，一直到  $r$  慢慢縮小到 20 以下，特徵點增加的數量才慢慢增加，如此的狀況實在相當不符合運算效率。表 3-1.1 更進一步說明了 ANMS 的統計情況，因我們假設  $r$  從影像長寬的最小值除以 10 開始往內縮，此 Pattern 大小為 752\*500，所以  $r$  設定為從 50 開始往內縮，大於 50 先假設為 0，我們會發現當  $r$  從無限大往內縮回時，特徵點增加的數量都非常少，在  $r$  縮小到 19 之前，特徵點增加的數量沒有超過 20 的，這是一個非常不好的現象，以  $r=49$  為例，表示我們去搜尋影像中每個 pixel 附近的 99\*99 範圍的視窗大小，卻只讓我們增加兩個特徵點，實在是相當不划算。圖 3-1.4 與表 3-1.2 使用了另外一個 Pattern 也有一樣的現象，在我們的研究當中總共測試

了 12 組的 Pattern 都有一樣的現象，這現象是相當合理的，因為當  $r$  很大，表示限制較嚴格，因此能擷取到的特徵點自然較少，當  $r$  慢慢變小，能擷取到的特徵點自然慢慢變多。



(a)

(b)



(c)

圖 3-1.4 ANMS 與 Search Range  $r$  之深入分析範例二

(a)ANMS 的結果

(b)Search Range 從無限大往內縮回時的特徵點擷取數量累積圖

(c)Search Range 從無限大往內縮回時的特徵點增加數量圖

表 3-1.2 SR 從無限大往內縮回時特徵點擷取數量與增加數量統計表範例二

<b>SR</b>	<b>&gt;48</b>	<b>48</b>	<b>47</b>	<b>46</b>	<b>45</b>	<b>44</b>	<b>43</b>	<b>42</b>	<b>41</b>	<b>40</b>	<b>39</b>
<b>Points</b>	0	30	31	33	34	35	37	38	41	43	45
<b>add</b>	30	1	2	1	1	2	1	3	2	2	1
	<b>38</b>	<b>37</b>	<b>36</b>	<b>35</b>	<b>34</b>	<b>33</b>	<b>32</b>	<b>31</b>	<b>30</b>	<b>29</b>	<b>28</b>
	46	50	54	56	58	60	61	64	64	68	70
	4	4	2	2	2	1	3	0	4	2	4
	<b>27</b>	<b>26</b>	<b>25</b>	<b>24</b>	<b>23</b>	<b>22</b>	<b>21</b>	<b>20</b>	<b>19</b>	<b>18</b>	<b>17</b>
	74	81	88	98	108	120	131	154	187	215	249
	7	7	10	10	12	11	23	33	28	34	27
	<b>16</b>	<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>11</b>	<b>10</b>	<b>9</b>			
	276	307	333	368	404	442	488	582			
	31	26	35	36	38	46	94	0			

### 3-1.2 BUMS 流程(The flow of Bottom-Up Maxima Selection)

經過 3-1.1 傳統 ANMS 的深入分析之後，我們不難發現傳統 ANMS 演算法雖然有平均分佈於整張影像的優點，但卻大大地犧牲了運算量，因此我們提出一套快速的演算法，稱為 Bottom-Up Maxima Selection，簡稱 BUMS。BUMS 比起傳統 ANMS 演算法，可以達到一樣的目的，但運算量卻大大地降低。

傳統 ANMS 演算法因為剛開始 Search Range  $r$  為無限大，但是能擷取到的特徵點數目又很少，不符合運算效益，因此我們很直接地想到應該把 Search Range  $r$  從內往外擴張，即  $r=1$  開始往外擴張，當  $r$  越大時，表示擷取出來的特徵點越可靠，當然也越少，當特徵點數目剔除掉剩下 500 個點時，演算法即停止，注意，到這裡只是初步構想，最後的 BUMS 尚未成型。目前暫時稱為修正型 ANMS，

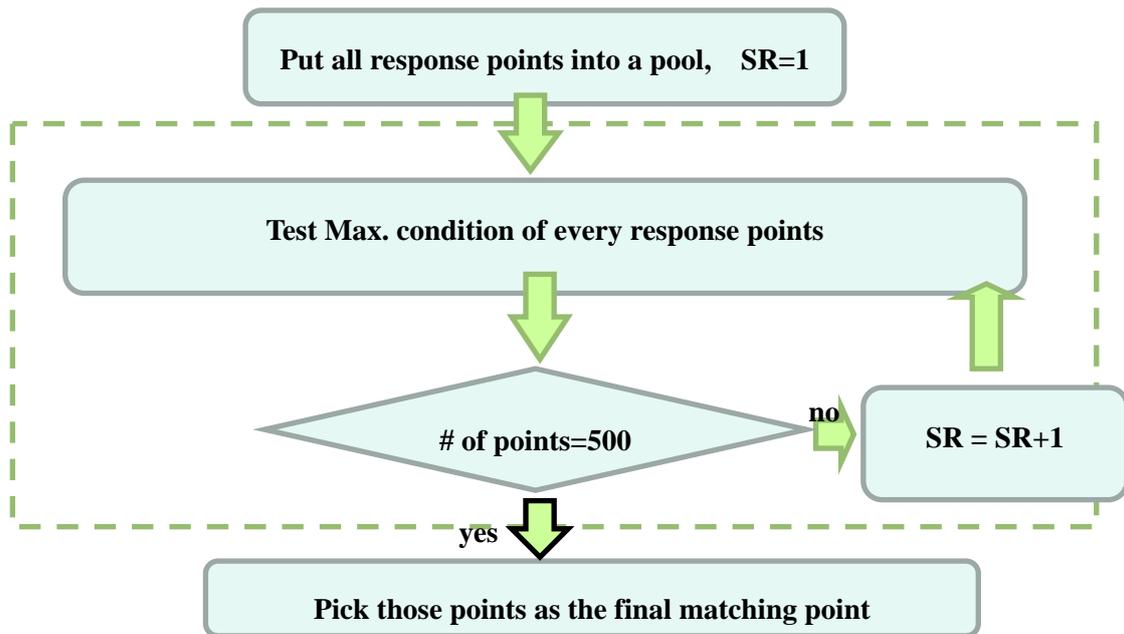
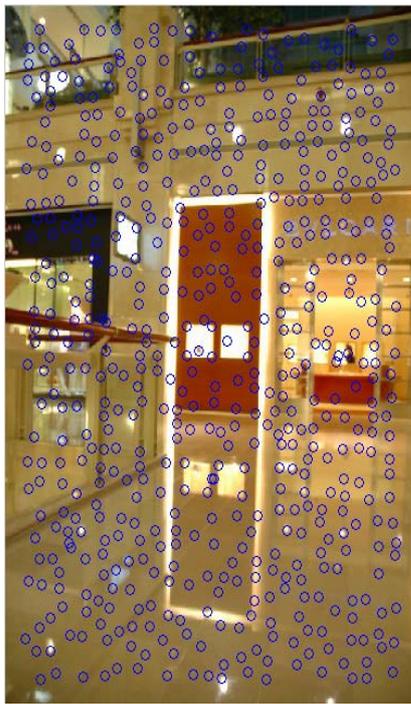


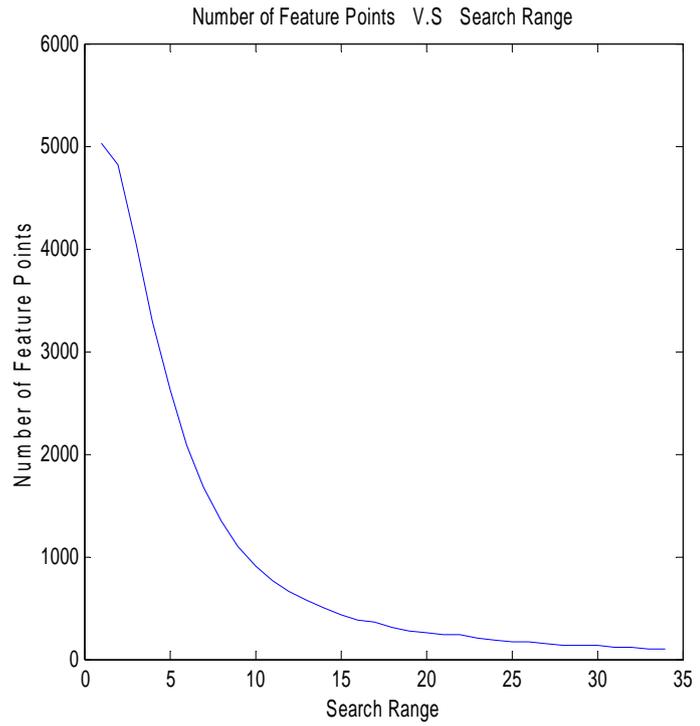
圖 3-1.5 修正型 ANMS 演算法流程圖

圖 3-1.5 為修正型 ANMS 的演算法流程圖，我們可以看出傳統 ANMS 從 Search Range  $r$  為無限大開始往內縮回，相當消耗運算量，但修正型 ANMS 演算法從  $r=1$  往外擴張，比起傳統 ANMS 當然節省相當多運算量。

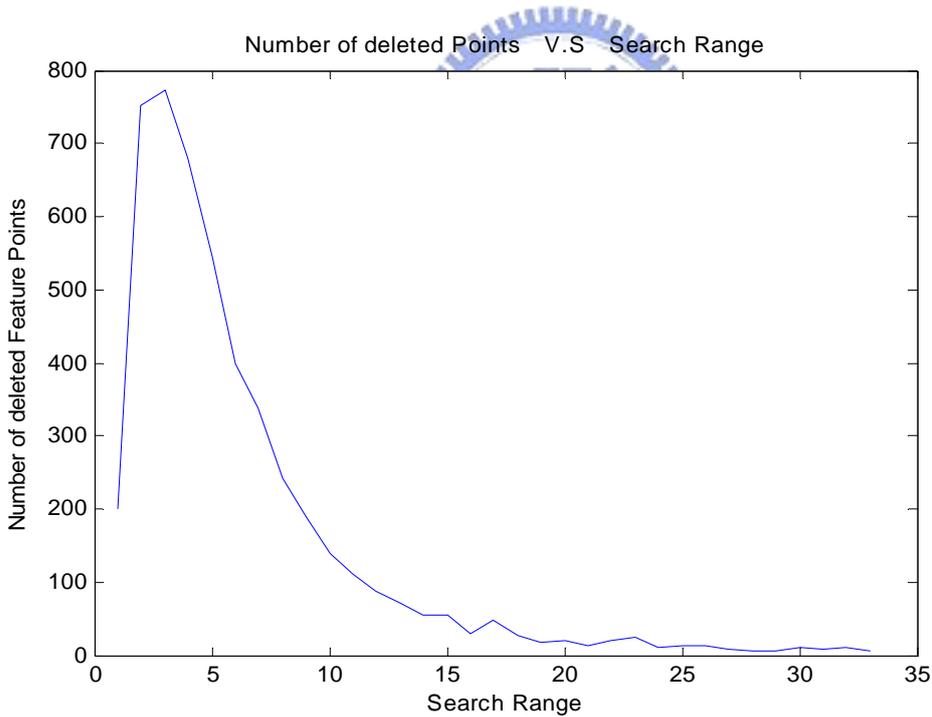
接著修正型 ANMS 之後，我們想到因為修正型 ANMS 從  $r=1$  往外擴張，但在擴張的過程當中，會有一些重複的點是不用繼續測試它的最大值情況，只需測試上一個  $r$  當中所擷取出來的候選特徵點(candidate points)即可，因為若在  $r=A$  的情況都無法成為可能的特徵點，那麼在  $r=A' > A$  的情況就更不可能成為特徵點了，傳統 ANMS 之所以需要重複判斷是因為  $r$  由外往內縮回，所以當  $r=B$  不是特徵點時並不一定代表  $r=B' < B$  一定不是特徵點，所以需要重複判斷。另外我們也考慮到是否一定要取  $n_i=500$  做為臨界值，若是  $n_i < 500$  可以節省後面的運算量，或是  $n_i > 500$  一定足夠做縫合嗎？因此我們試著仿照 3-1.1 的做法去觀察當 Search Range  $r$  從  $r=1$  往外擴增時候的特徵點擷取數量和特徵點剔除數量與  $r$  的對應關係圖，如圖 3-1.6。圖 3-1.6(b) 表示  $r=1$  往外擴張時候的特徵點擷取數量與  $r$  的對應關係圖，我們會發現剛開始當  $r=1$  時，擷取到的特徵點非常多，之後當  $r$  往外擴張，特徵點數量便急速下降，再往外擴大到約超過 25 之後，特徵點



(a)



(b)



(c)

圖 3-1.6 BUMS 與 Search Range  $r$  之深入分析範例一

(a)BUMS 的結果

(b)Search Range 從  $r=1$  往外擴張時的特徵點擷取數量圖

(c)Search Range 從  $r=1$  往外擴張時的特徵點剔除數量圖

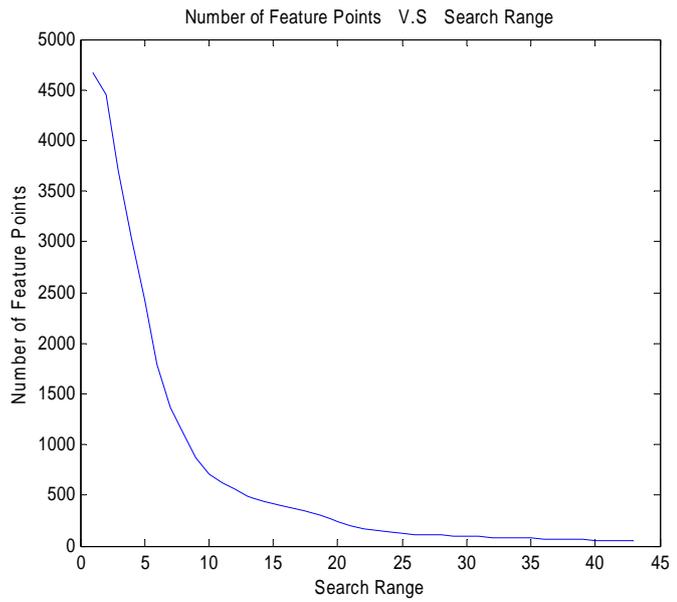
表 3-1.3 SR 從 r=1 往外擴張時特徵點擷取數量與剔除數量統計表範例一

SR	1	2	3	4	5	6	7	8	9	10
Points	5018	4819	4068	3296	2616	2071	1673	1336	1093	902
delete	199	751	772	680	545	398	337	243	191	139
	11	12	13	14	15	16	17	18	19	20
	763	653	566	494	438	383	354	305	279	262
	110	87	72	56	55	29	49	26	17	20
	21	22	23	24	25	26	27	28	29	30
	242	229	208	184	174	161	147	138	132	126
	13	21	24	10	13	14	9	6	6	10
	31	32	33	34						
	116	108	98	93						
	8	10	5							

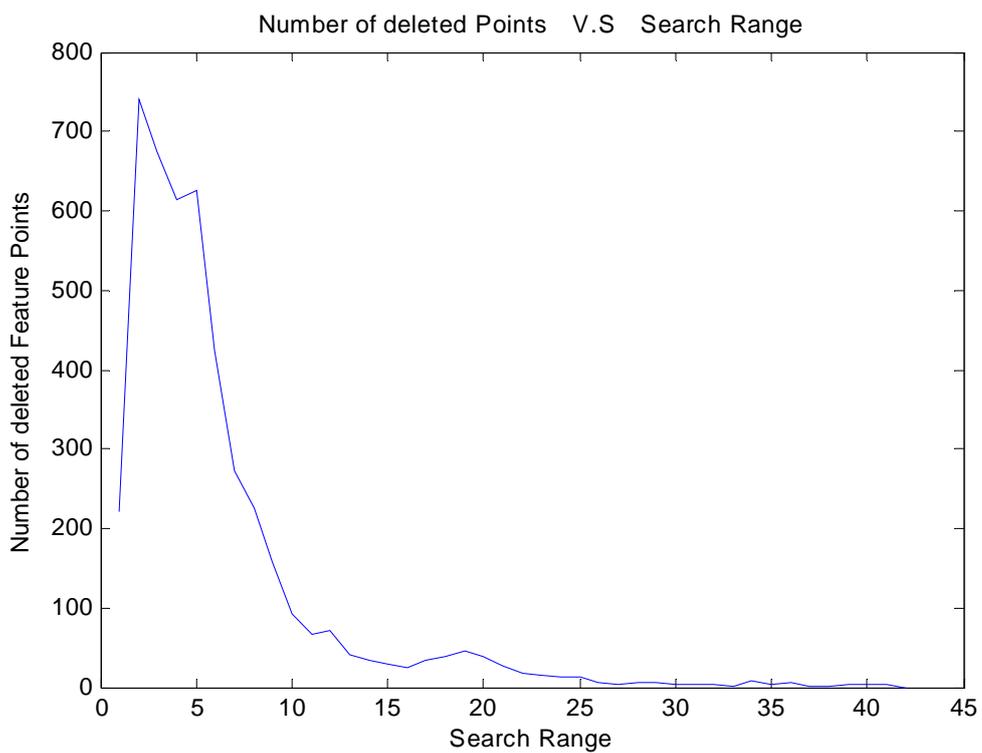
下降的數量才慢慢趨近於緩和。圖 3-1.6(c)顯示當 r=1 往外擴張時，特徵點剔除掉的數量與 Search Range r 的對應關係，我們可以清楚看到剛開始 r=1 往外擴張到約 r=15 時，剔除掉的特徵點數量都非常多，之後 r 超過 15 之後，剔除掉的特徵點數量才趨近於緩和。到了這裡我們已經可以非常清楚了解到為什麼 BUMS 可以比傳統 ANMS 節省相當多的運算量，因為 BUMS 剛開始 r 很小時就擷取出相當多的特徵點，運算效率相當高，但是傳統 ANMS 剛開始 r 很大時擷取到的特徵點數量卻相當的少，需要繼續做相當多的運算，因此 BUMS 自然可以比傳統 ANMS 節省相當多的運算量。圖 3-1.7 和表 3-1.4 使用另一個 Pattern 也顯示出相同的結果。在我們的研究當中總共測試了 12 組的 Pattern 都有一樣的現象，原因是因為剛開始 r 很小往外擴張時，因為特徵點很多，所以容易碰到其他的特徵點並做比較，經過比較之後此點不是此 r\*r 範圍視窗裡面最大的值就會被剔除掉，因此剛開始往外擴張時容易剔除掉很多的特徵點，但是隨著特徵點越來越少，能夠碰



(a)



(b)



(c)

圖 3-1.7 BUMS 與 Search Range  $r$  之深入分析範例二

(a) BUMS 的結果

(b) Search Range 從  $r=1$  往外擴張時的特徵點擷取數量圖

(c) Search Range 從  $r=1$  往外擴張時的特徵點剔除數量圖

表 3-1.4 SR 從 r=1 往外擴張時特徵點擷取數量與剔除數量統計表範例二

SR	1	2	3	4	5	6	7	8	9	10
Points	4672	4450	3710	3035	2420	1795	1371	1098	873	716
delete	222	740	675	615	625	424	273	225	157	92
	11	12	13	14	15	16	17	18	19	20
	624	558	486	445	411	382	357	323	284	238
	66	72	41	34	29	25	34	39	46	38
	21	22	23	24	25	26	27	28	29	30
	200	174	156	140	126	114	109	105	98	92
	26	18	16	14	12	5	4	7	6	3
	31	32	33	34	35	36	37	38	39	40
	89	86	83	82	74	70	65	63	61	57
	3	3	1	8	4	5	2	2	4	4
	41	42								
	53	50	50							
	3	0								

到其他特徵點並做比較的機率就越低，因此能夠刪除特徵點的機會就會低，所以到最後越往外面擴張，特徵點擷取數目與特徵點剔除數目就越趨近於飽和。

綜合以上，在 Bottom-Up Maxima Selection (BUMS) 當中，我們的做法是把 Search Range  $r$  從 1 開始往外擴張，並且在擴張的過程當中只針對候選特徵點 (Candidate Points) 來做擴張，不是候選特徵點的就不重複做，另外我們也不使用  $n_i=500$  做最後的特徵點，而是當某個 Search Range  $r$  剔除掉的點小於 50，BUMS 即停止，會這樣做的原因是因為當特徵點剔除數量  $< 50$  時，整個特徵點已經趨近於收斂，如表 3-1.3、表 3-1.4 都可以清楚地觀察到此特性。例如當表 3-1.4

的  $r=13$  時，特徵點剔除的數量只剩下 41 個，之後都越來越趨近飽和，因此若  $r$  繼續往外擴張，特徵點剔除的數量會越來越少，Search Range 變大但是能剔除的特徵點數量卻越來越少，不符合運算效率，因此我們定當特徵點剔除數量小於 50 時，BUMS 即停止，且如此的做法可以適應性地選擇特徵點的數量，若是特徵點數量小於 500 點，對之後的 Matching 也有很大的幫助，如圖 3-1.8。

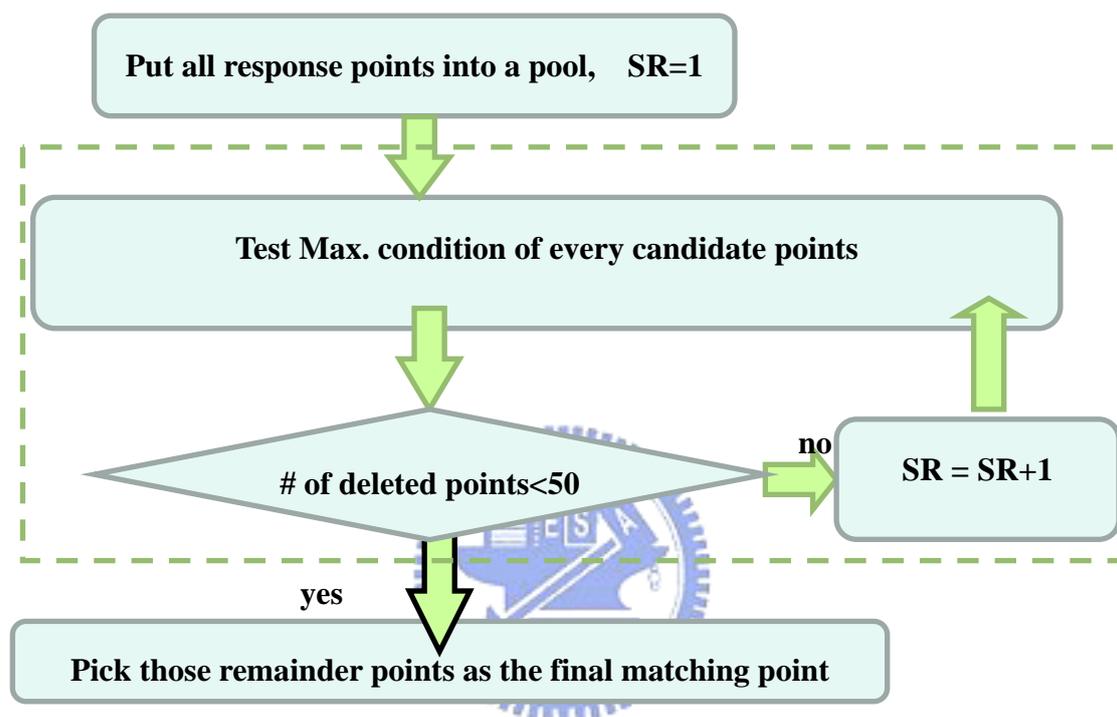


圖 3-1.8 Bottom-Up Maxima Selection(BUMS)流程圖

### 3-1.3 特徵點均勻分佈特性驗證(Verification of the Uniform Distribution of the Feature Points)

在前面幾節當中，我們曾經不只一次談到特徵點的均勻分佈將會對影像的縫合有很大的益處，因此本節將簡單地提出一個例子來並用標準差的方式來做驗證，圖 3-1.9 顯示特徵點擷取較為集中的情況，影像中的內部點(Inlier Points)總共有 14 個，並且我們去計算兩張影像特徵點的 X 軸與 Y 軸標準差，圖 3-1.9



(a) 匹配點(Matching Points)與內部點(Inlier Points)



(b) 歪斜影像

(c) 縫合影像

圖 3-1.9 特徵點較為集中的情況範例

(a) 藍色圈為匹配點(Matching Points)，紅色又為內部點(Inlier Points)

(b) 歪斜影像

(c) 縫合影像



(a) 匹配點(Matching Points)與內部點(Inlier Points)



(b) 歪斜影像



(c) 縫合影像

圖 3-1.10 特徵點較為分散的情況範例

(a) 藍色圈為匹配點(Matching Points)，紅色又為內部點(Inlier Points)

(b) 歪斜影像

(c) 縫合影像

的  $std_x$  分別為 67.493 與 67.723， $std_y$  分別為 39.431 與 38.896，我們會發現 Y 軸的標準差較小，即 y 軸分佈較不均勻，所以圖 3-1.9(b)和(c)顯示的歪斜影像和縫合影像顯示了因為特徵點不均勻分佈所產生的影像縫合失真。同樣的方法來看圖 3-1.10，影像中的內部點(Inlier Points)總共有 15 個，並且我們去計算兩張影像特徵點的 X 軸與 Y 軸標準差，圖 3-1.9 的  $std_x$  分別為 72.914 與 72.642， $std_y$  分別為 104.67 與 101.31，我們會發現 X 軸和 Y 軸的標準差都變大了，特別是 Y 軸標準差從 35~40 增加到 100~105 之間，落差非常大，即圖 3-1.10 特徵點分佈較為均勻，我們會發現圖 3-1.10(b)和(c)顯示的歪斜影像和縫合影像可以成功地縫合。由這個例子我們便可以清楚地知道特徵點若能均勻的分佈在整張影像上是一個非常好的特性，而 BUMS 既能把特徵點均勻分佈在整張影像上，比起 ANMS 又能大大節省運算量，因此 BUMS 是一個非常好的做法。

## 3-2 極化型隨機取樣程序(Polarized Random Sample Consensus)

### 3-2.1 RANSAC 執行次數之討論(Discussion of RANSAC Times)

在 3-1 當中我們提出了 BUMS 快速演算法，除了使特徵點可以均勻分佈於整張影像之外，另一個好處就是可以使特徵點不用受限於 500 點，只要符合我們的情況，即使特徵點數目不到 500 點也可以停止，因此從這裡也給了我們新的靈感，[36]-[37] 當中使用樣本資料數目  $n=4$ ，重覆執行次數  $k=500$ ，定義內部點為 500 次當中最大對應成功次數的樣本資料。此種做法讓我們想到是否可以不需要做這麼多次的 RANSAC？事實證明是可以的，[36]-[37] 提出執行次數  $k=500$  只是為要讓所有的錯誤匹配點經過 RANSAC 之後全部被剔除掉，因為若經過 RANSAC 之後沒有辦法剔除掉，最後的影像縫合一定會失敗。如圖 3-2.1，圖 3-2.1(a)(b)顯示經過 RANSAC 之後仍有兩個 false match 的結果，圖 3-2.1(c)顯示因為 false match 所造成的影像失真。

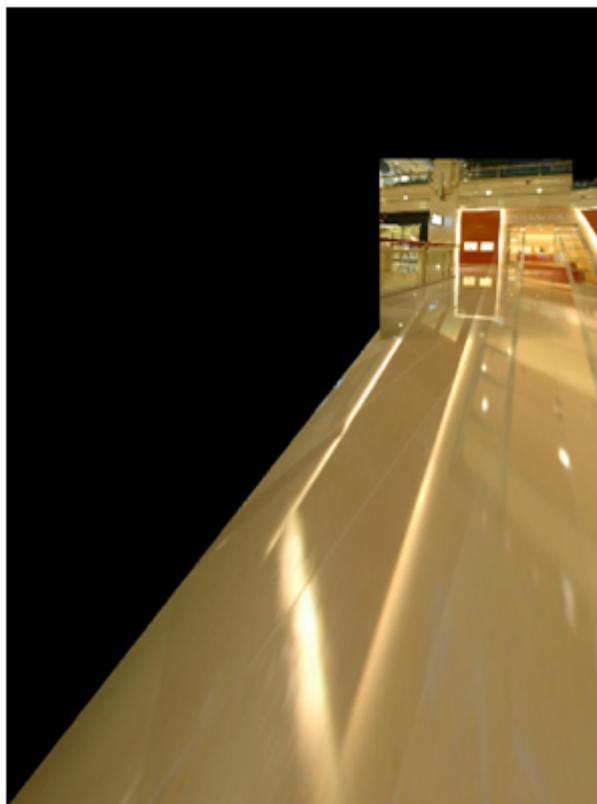
我們首先嘗試把 RANSAC 次數從 500 次降到 50 次，圖 3-2.2(a)顯示執行 500 次 RANSAC 得內部點，圖 3-2.2(b)顯示執行 50 次 RANSAC 得內部點，我們會



(a)



(b)

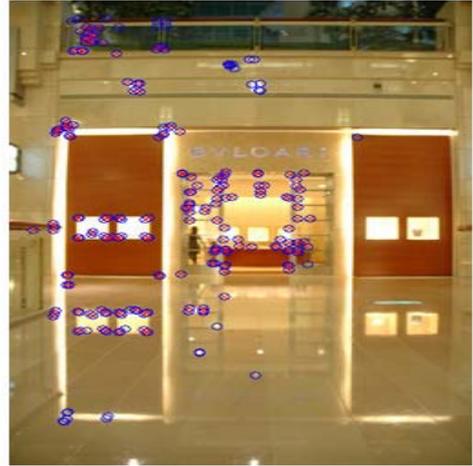


(a)

圖 3-2.1 失敗 RANSAC 與失敗的縫合影像

(a)(b) 經過 RANSAC 之後仍有兩個 false match 的結果

(c) 因為 false match 所造成的影像失真



(a) 執行 500 次 RANSAC 之後的內部點(inlier points)



(b) 執行 50 次 RANSAC 之後的內部點(inlier points)



(c) 執行 500 次 RANSAC 的縫合影像

(d) 執行 500 次 RANSAC 的縫合影像

圖 3-2.2 執行不同次數 RANSAC 的內部點結果與縫合影像

發現兩組影像完全一樣，因此縫合影像也完全一樣，從這裡我們就可以發現不一定需要 500 次的 RANSAC 才能成功地縫合影像。

### 3-2.2 極化型隨機取樣程序流程(The Flow of Polarized Random Sample Consensus)

[36]-[37] 當中使用樣本資料數目  $n=4$ ，重覆執行次數  $k=500$ ，定義內部點為 500 次當中最大對應成功次數的樣本資料。執行 500 次的原因最重要的是要讓最後擷取正確特徵點的機會提高，因此 RANSAC 次數  $k$  當然是越高越好，次數越低錯誤的機會越高。因此不同於 [36]-[37] 的做法，我們提出極化型隨機取樣程序 (PRSC)，PRSC 首先使用投票機制 (Voting) 來做我們最後取內部點的判別，如表 3-2.1，表 3-2.1 顯示直接做 5000 次 RANSAC 並使用投票機制的結果，如此做我們只是要先觀察使用 Voting 的好處與現象，表 3-2.1 左邊兩行 Match X1 與 Match Y1 為某組 Pattern 裡面其中一張影像的匹配點座標，Voting5000 表示每個座標點經過 5000 次 RANSAC 之後每個點被 Voting 到的次數。從表 3-2.1 當中我們可以清楚地看到這 24 個匹配點經過 5000 次的 RANSAC 之後，有 4 個點被 Voting 到的次數非常少，有一個座標點甚至只被投票到 22 次，剩下 20 個座標點被 Voting 到的次數都非常高，最少都有 1441 次，也就是所有的匹配點呈現兩極化 (Polarized) 的現象。如果是正確的匹配點 (Correct Match)，則經過一定次數的 RANSAC 之後被 Voting 到的次數會非常多，相對地，如果是錯誤的匹配點 (False Match)，經過一定次數的 RANSAC 之後被 Voting 到的次數會非常少，這是一個非常自然也可以預知的現象，在式 2-5 中我們曾經說明歪斜影像的轉換關係，此運算式也是判斷投票與否的判定式子，當我們建立了一個正確的模型  $H$ ，並把所有點套進  $H$  當中去測試此點是否為正確的匹配點，若測試成功就會在 Voting 上面加上 1，若失敗則不加，因此正確的匹配點經過一定次數的 RANSAC 之後被 Voting 到的次數會非常多，錯誤的匹配點被 Voting 到的次數會非常少。之所以錯誤的匹配點仍有少數的 Voting，原因在於若是我們建立的模型  $H$  是錯誤的，則錯誤

表 3-2.1 匹配點經過 5000 次 PRSC 投票結果之關係表

Number	Match X1	Match Y1	Voting5000
1	80	84	1638
2	81	84	1598
3	82	84	1590
4	238	101	<b>254</b>
5	106	134	1450
6	129	134	1443
7	130	134	1441
8	269	280	1878
9	269	281	2056
10	267	283	1861
11	228	289	1975
12	228	290	1991
13	336	301	<b>22</b>
14	200	311	2154
15	201	311	2183
16	302	317	<b>412</b>
17	301	318	<b>438</b>
18	193	336	2074
19	193	337	2053
20	193	338	2025
21	94	373	1549
22	91	379	1717
23	92	379	1733
24	94	380	1640

的匹配點也會被誤認會是正確的匹配點，因此仍然有少數的 Voting。當我們知道這個現象之後，我們就可以得到，當 RANSAC 次數到達一定程度，使得所有匹配點的 Voting 值呈現兩極化(Polarized)的現象(如表 3-2.1 的 Voting 兩極化)，RANSAC 即可停止，不需要執行 RANSAC 到 500 次才停止，因為若出現此情況，表示正確的匹配點已經幾乎知道，不需要再消耗運算量來做多餘的 RANSAC 了。因此我們在這裡定一個臨界值 0.3，當所有匹配點 Voting 的標準差已經大

於最大 Voting 值的 0.3 倍，表示 Voting 值呈現兩極化的現象，RANSAC 即停止，不需要再消耗運算量來做多餘的 RANSAC 了。

除了投票機制之外，PRSC 還有許多修正的地方，傳統 RANSAC 演算法在每次隨機取樣的時候都從所有匹配點當中做取樣，但我們發現這樣不是一個好的做法，因為在很多時候已經知道哪些可能是正確的匹配點或錯誤的匹配點，因此應該可以訂出一套機制使得隨機取樣程序更精準。我們預設一個臨界值 0.3，當每做一次 RANSAC 結束後，若是對應到的匹配點數目達到所有匹配點數目的 0.3 倍，表示測試成功的樣本座標點很多，則我們稱這次的 RANSAC 為成功，而且下次的 RANSAC 從不再重新隨機取樣，而是從上一次的 RANSAC 當中測試成功的樣本座標點來做隨機取樣，如此的做法可以大大的提升擷取正確內部點的機率，比起單純做 Voting 標準差法還可以更節省 RANSAC 次數，在第四章我們會提出模擬結果來證明。圖 3-2.3 為極化型隨機取樣程序(PRSC)的流程圖。

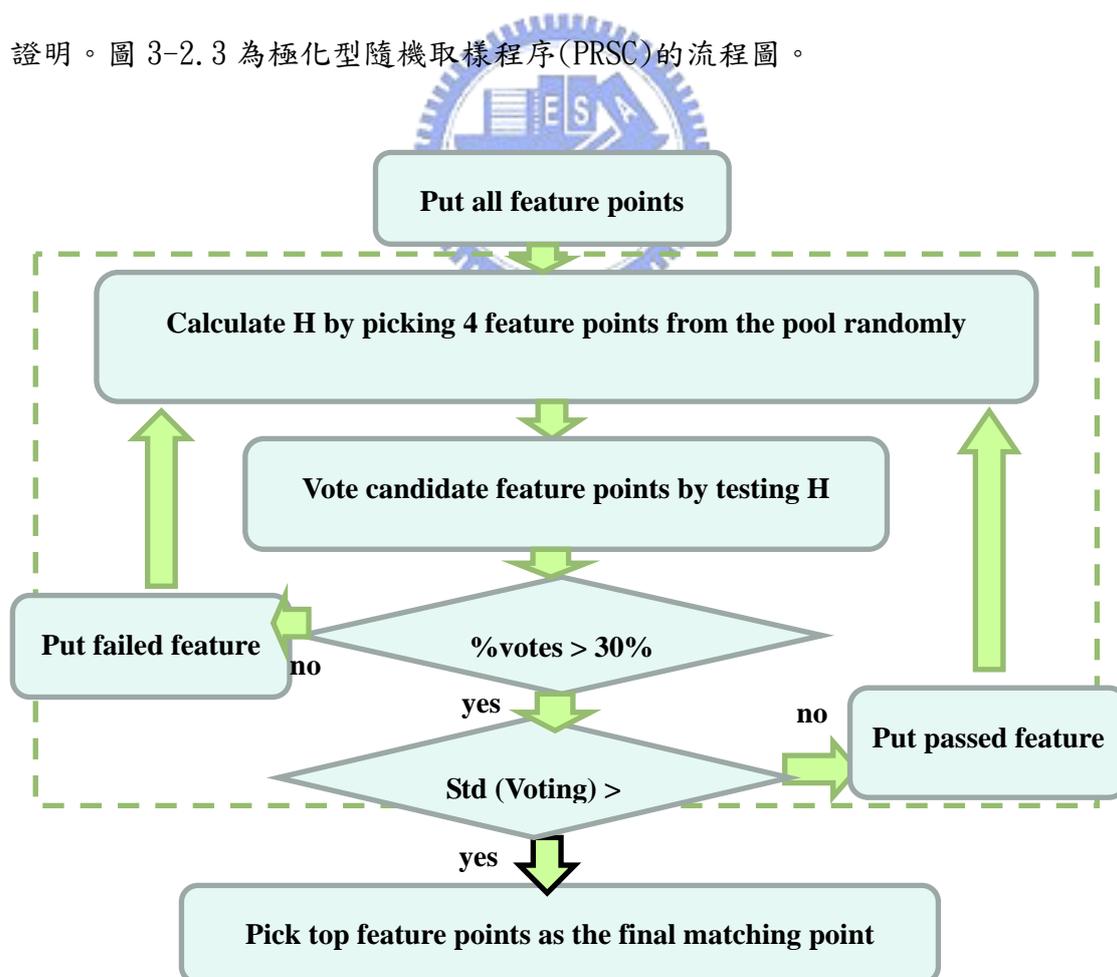


圖 3-2.3 極化型隨機取樣程序(PRSC)流程圖

PRSC 在最後階段擷取最後的正確匹配點或稱內部點(Inlier Points)時，我們是採取所有匹配點的低標來當臨界值，當 Voting 超過所有匹配點的低標值則稱為內部點(Inlier Points)，超過則視為外部點(Outlier Points)，傳統 RANSAC 定義內部點為 500 次當中最大對應成功次數的樣本點，因為它沒有做投票機制的動作所以無法取高低標或平均，但 PRSC 取低標當最後臨界值，原因是因為我們是採取適當的投票機制，幾次之後發現標準差拉開了，即 Voting 值呈現兩極化的現象，一組兩極化的資料，且大部分的資料值很大，少數的資料值很小，我們希望把大部分的資料值擷取出來，在統計上以低標最理想的臨界值。若是取高標當臨界值，雖然擷取出來的點都是 Voting 值很高的，表示非常可靠，但如此的取法卻容易使得特徵點數目不夠作之後的縫合，因為取高標會把多數正確的內部點也剔除掉，這不是好的現象，如圖 3-2.4，圖 3-2.4(a)(b)為臨界值取高標的匹配點與內部點結果，我們會發現取到的內部點非常少，大部分的正確內部點都因為臨界值取高標而被剔除掉了，因此圖 3-2.4(c)和圖 3-2.4(d)自然地也就因為內部點點數太少而失真了。若是我們取均標當臨界值，在大部分的情況仍然可行，但是在 PRSC 執行次數上比起低標當臨界值需要更多次的執行次數，



(a)(b)匹配點與內部點



(c) 歪斜影像



(d) 縫合影像

圖 3-2.4 PRSC 臨界值取高標的結果

原因與取高標當臨界值一樣，都是因為把部分正確的內部點剔除掉了，只是取均標比取高標強健，因為剔除掉的正確內部點沒這麼多，所以若是取均標仍然可行，但是相較於取後標當臨界值需要較多次的 PRSC 執行次數。若是取後標當臨界值，看似會比低標效果好，但是此種情況會有把外部點也誤認為內部點的機會，進而影響到了之後縫合的效果，因此取後標不是一個好的方法。表 3-2.2 證明在 PRSC 當中使用低標比均標成功的機率高，我們把 PRSC 大幅降低到只做 10 次並取觀察他們分別的結果，由表 3-2.2 當中我們可以看到在執行 10 次 PRSC 時使用低標當臨界值比均標當臨界值成功機率高上許多，如此低的執行次數卻已經有 98% 的成功縫合機率，由此也可以間接證明傳統 RANSAC 演算法的執行次數事實上是不需要執行到 500 次的，因為若我們使用投票機制來做 RANSAC，當執行次數到了一定次數之後(即匹配點呈現兩極化時)，正確匹配點(Correct Match)或稱內部點(Inlier)的正確性就已經顯明出來，RANSAC 可以立即停止，但若沒有使用投票機制，正確匹配點(Correct Match)或稱內部點(Inlier)的正確性就無法顯明出來，只能大幅增加 RANSAC 次數以確保當次數增加到非常多，最後擷取到正確匹配點的機會才會萬無一失，但 PRSC 卻不需要如此做，因此從這裡我

們終於可以一次清楚明白使用投票機制使得匹配點呈現兩極化現象與 PRSC 臨界值和取低標當最後門檻的原因。在第四章當中我們也會繼續針對使用 PRSC 與傳統 RANSAC 做更進一步的分析。

表 3-2.2 執行 10 次 PRSC 低標和均標的成功機率比較

	PRSC (average filtering)	PRSC (lower-average filtering)
#Correct trials	78	98
#False trials	22	2
Correctness	78%	98%



## 第四章 模擬結果之探討與比較

---

上一章節中，我們了解了有關使用 BUMS(Bottom-Up Maxima Selection)與 PRSC(Polarized Random Sample Consensus)的影像縫合之快速校正技術設計，所以本章當中我們就針對這兩種方法與傳統的演算法來分別運算效率上的比較，當然，我們也會把我們的快速校正演算法與傳統的演算法在整體運算效率上來做比較，以證明就整體運算效率而言我們仍然可以大大地改善。另外在 4-3 當中我們也會驗證在多個 Sensor 相對位置都固定的前提下，轉換矩陣 H 的重覆使用性質，以致於只要執行一次我們的快速校正演算法即可，剩下的影像校正只要經過轉換矩陣 H 做簡單的轉換即可，如此的特性將對相機陣列的校正技術更為方便，並且適用於硬體實作當中。

### 4-1 BUMS 運算效率模擬(Simulation of Computational Efficiency for Bottom-Up Maxima Selection)

在第三章當中我們介紹了 BUMS(Bottom-Up Maxima Selection)的方法，與傳統演算法的不同主要在於傳統 ANMS 把特徵點的搜尋範圍從無限大往內縮回，但是剛開始搜尋範圍  $r$  很大時，特徵點的數目會非常少，因此消耗了運算量，且根據統計傳統 ANMS 演算法因為搜尋範圍剛開始為無限大，因此佔了整個校正技術大部分的時間，因此拖垮了整個運算效率。BUMS(Bottom-Up Maxima Selection)把特徵點的搜尋範圍從內往外擴張，因此剛開始的搜尋範圍很小，但卻可以找到很多候選特徵點，並且最後也採用適應性的方法，當某個搜尋半徑  $r$  的特徵點剔除數目小於 50 點時，BUMS 即停止。如此的作法不僅可以使特徵點不一定在 500 點，若是小於 500 點，則對之後的校正也有很大的幫助，若是多於 500 點，因為我們的停止機制是採用適應性的方式，因此這些的點數會比傳統 AMNS 更加可靠。圖 4-1.1 的 12 組影像為我們模擬的 12 組 Pattern，每組 Pattern 有三張影像。並且在傳統 AMNS 剛開始搜尋範圍為無限大的地方，我們設定  $r$  為影像長寬

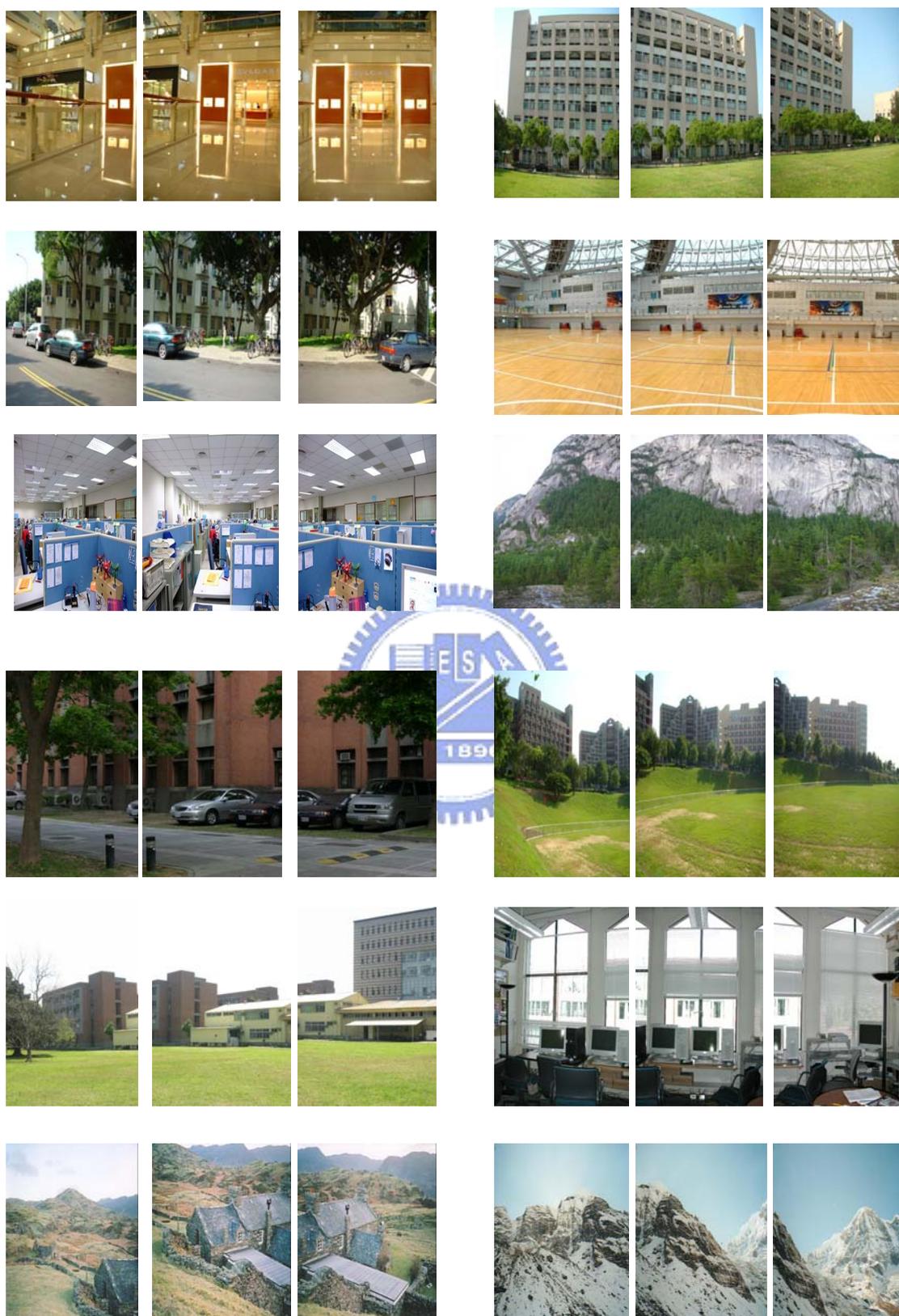


圖 4-1.1 測試模擬的 12 組 Pattern

表 4-1.1 縫合兩張圖的傳統 ANMS 與 BUMS 的計算效率比較

Pattern	1	2	3	4	5	6	7
BUMS/ANMS	0.25	0.25	0.27	0.24	0.51	0.18	0.21
	8	9	10	11	12	Average	
	0.23	0.20	0.32	0.36	0.33	0.28	

的最小值除以 10，如此做是為了增加 BUMS 的可靠度與 Worse Case 的考量。表 4-1.1 顯示這 12 組 Pattern 在傳統 ANMS 與我們 BUMS 的運算效率比較表。我們會發現這 12 組的 Pattern 至少都有將近 4 倍的改善效果，平均起來運算量也只剩下 0.28，顯示 BUMS 比起傳統 ANMS 的確省下相當多運算量，這也是相當合理的結果，因為傳統 ANMS 搜尋半徑  $r$  從外往內縮回，運算量自然較大，我們 BUMS 搜尋半徑  $r$  從內往外擴張並且使用適應性的停止機制，運算量自然較節省。表 4-1.2 顯示縫合三章影像時 ANMS 與 BUMS 的運算效率比較。我們會發現縫合三張圖對於運算效率平均上又更加節省一些了，而其中我們也發現有少數的 Pattern 在縫合三張影像時節省的效率反而比兩張還稍微差一點點，這是因為 BUMS 是使用適應性的停止機制來停止特徵點的擷取，因此當此步驟還沒達到停止機制時，BUMS 就不會停止，因此有些少數的 Pattern 可能在縫合第三張影像的時候需要花比較多的運算量來達到停止機制，但是平均起來還是比縫合兩張影像節省一些運算效率，當然，比起傳統 ANMS 演算法而言一定節省相當多運算效率。圖 4-1.2 為縫合三張影像的範例。

表 4-1.2 縫合三張圖傳統 ANMS 與 BUMS 的計算效率比較

Pattern	1	2	3	4	5	6	7
BUMS/ANMS	0.21	0.22	0.21	0.23	0.41	0.17	0.20
	8	9	10	11	12	Average	
	0.26	0.17	0.36	0.34	0.41	0.267	



圖 4-1.2 縫合三張影像

#### 4-2 PRSC 運算效率模擬 (Simulation of Computational Efficiency for Polarized Random Sample Consensus)

在前面幾章當中我們也介紹了傳統 RANSAC 的方法與我們加強型的 PRSC 方法，與傳統 RANSAC 方法不同的地方主要在於我們使用投票機制，因為當執行次數到了一定程度的時候，因為使用投票機制會使得所有的匹配點標準差拉開呈現

兩極化(Polarized)的現象，哪些是正確的匹配點已經幾乎可以知道，因此我們只要加入適應性的臨界值，並且簡單地取後標來當最後的門檻把內部點取進來即可，但是傳統的 RANSAC 演算法因為沒有使用投票機制，因此當執行次數到了一定的程度仍然無法知道哪些很有可能是最後正確的內部點，因此只能無奈地將 RANSAC 運算次數 k 提升到 500 次以把錯誤的機會減到最低，可見傳統 RANSAC 不是一個積極的好方法。表 4-2.1 顯示縫合兩張圖時使用傳統 RANSAC 與我們 PRSC 的運算效率比較表，我們會發現在此部分平均節省到剩下 0.43 倍的運算量，注意在此部分的模擬結果之所以標準差會比較大的原因是因為我們使用適應性的停止機制來停止 PRSC，因此當投票機制還沒有滿足我們適應性機制時就會繼續運算，因此此部份完全是依照影像內容而定，但是我們會發現到每一個 Pattern 在此步驟還是都會節省相當多的運算量。表 4-2.2 顯示縫合三張圖時在傳統 RANSAC 與我們 PRSC 的運算效率比較表。(使用的 Pattern 仍然是圖 4-1.1 當中的影像)我們會發現有些數據居然比縫合兩張還高，主要的原因也跟縫合兩張時一樣，因為此部分完全是依照影像內容而定，當還沒有滿足適應性的停止機制時就會繼續執行 PRSC，因此有時候反而沒有縫合兩張影像效果來的好，但就單純跟傳統 RANSAC 來做比較仍然是大幅地節省了運算量，由圖 4-2.1 我們也可以發現傳統方法執行了 500 次 RANSAC，但 PRSC 只執行了 99 次 RANSAC，因此運算量自然比較節省。

表 4-2.1 縫合兩張圖的傳統 RANSAC 的 PRSC 的計算效率比較

Pattern	1	2	3	4	5	6	7
PRSC/RANSAC	0.33	0.08	0.17	0.08	0.10	0.60	0.74
	8	9	10	11	12	Average	
	0.32	0.40	0.93	0.56	0.82	0.43	

表 4-2.2 縫合三張圖的傳統 RANSAC 的 PRSC 的計算效率比較

Pattern	1	2	3	4	5	6	7
PRSC/RANSAC	0.42	0.09	0.51	0.09	0.08	0.37	0.46
	8	9	10	11	12	Average	
	0.26	0.13	0.45	0.63	0.67	0.35	

Filename	File Type	Calls	Total Time	% Time	Time Plot
<a href="#">numerics&gt;mtimes</a>	Builtin-function	51501	0.156 s	21.3%	■
<a href="#">computeH</a>	M-function	501	0.063 s	8.5%	■
<a href="#">randperm</a>	M-function	500	0.031 s	4.3%	■
<a href="#">numerics&gt;inv</a>	Builtin-function	500	0.016 s	2.1%	■
<a href="#">numerics&gt;sum</a>	Builtin-function	2	0 s	0%	
<a href="#">numerics&gt;sort</a>	Builtin-function	1	0 s	0%	
<a href="#">mean</a>	M-function	1	0 s	0%	
<a href="#">numerics&gt;eye</a>	Builtin-function	1	0 s	0%	
Self time (built-ins, overhead, etc.)			0.469 s	63.8%	■
Totals			0.734 s	100%	

Filename	File Type	Calls	Total Time	% Time	Time Plot
<a href="#">computeH</a>	M-function	99	0.031 s	10.5%	■
<a href="#">numerics&gt;sum</a>	Builtin-function	392	0.031 s	10.5%	■
<a href="#">std</a>	M-function	98	0.031 s	10.5%	■
<a href="#">numerics&gt;size</a>	Builtin-function	91	0 s	0%	
<a href="#">libmwbuiltins&gt;_colonobj</a>	Builtin-function	1	0 s	0%	
<a href="#">randperm</a>	M-function	99	0 s	0%	
<a href="#">numerics&gt;inv</a>	Builtin-function	98	0 s	0%	
<a href="#">numerics&gt;mtimes</a>	Builtin-function	8429	0 s	0%	
<a href="#">numerics&gt;length</a>	Builtin-function	98	0 s	0%	
<a href="#">numerics&gt;max</a>	Builtin-function	98	0 s	0%	
<a href="#">numerics&gt;sort</a>	Builtin-function	1	0 s	0%	
<a href="#">mean</a>	M-function	1	0 s	0%	
<a href="#">numerics&gt;eye</a>	Builtin-function	1	0 s	0%	
Self time (built-ins, overhead, etc.)			0.203 s	68.4%	■
Totals			0.297 s	100%	

(a)傳統 RANSAC

(b)PRSC

圖 4-2.1 傳統 RANSAC 與 PRSC 運算量比較圖

### 4-3 整體運算效率模擬 (Simulation of Total Computational Efficiency for the Algorithm)

在最後我們當然要來比較一下傳統演算法與我們快速校正演算法在整體的運算量上來做比較，表 4-3.1 顯示縫合兩張圖時傳統演算法與我們快速校正演算法的計算效率比較，我們會發現至少都節省到剩下 0.445 倍的運算量，且這還只是 Worse Case，因為在 4-1 我們曾談到我們假設傳統搜尋範圍為影像長寬的最小值除以 10，因此比起傳統演算法真的使用無限大來做開始的搜尋，我們的快速校正演算法又會節省更多的運算量。表 4-3.2 顯示顯示縫合三張圖時傳統演算法與我們快速校正演算法的計算效率比較，我們會有些 Pattern 甚至多消耗了運

算量，原因是寫入三張影像的運算量平均上來講需要比寫入兩張影像的運算量多出 1.5 倍，但整體而言比起傳統演算法仍然是有一定程度的改善，除非到最後逢合相當多當影像，以致於我們改善的 BUMS 與 PRSC 的步驟都已經達到飽和，那麼演算法改善程度才會趨近於飽和，圖 4-3.1 為傳統演算法與快速演算法比較圖。

表 4-3.1 縫合兩張圖時傳統演算法與我們快速演算法的計算效率比較

Pattern	1	2	3	4	5	6	7
Ours/Tradition	0.45	0.47	0.48	0.43	0.62	0.34	0.39
	8	9	10	11	12	Average	
	0.42	0.42	0.56	0.74	0.65	0.455	

表 4-3.2 縫合三張圖時傳統演算法與我們快速演算法的計算效率比較

Pattern	1	2	3	4	5	6	7
Ours/Tradition	0.42	0.44	0.43	0.45	0.59	0.38	0.40
	8	9	10	11	12	Average	
	0.51	0.40	0.5	0.79	0.75	0.50	

Function name	Calls	Total Time	Self Time*	Total Time Plot (dark band = self time)
<a href="#">stitch</a>	1	323.516 s	9.203 s	
<a href="#">Extract_Feat</a>	1	251.964 s	237.734 s	
<a href="#">takesample</a>	342576	36.953 s	36.953 s	
<a href="#">subsampling</a>	6	35.094 s	4.594 s	
<a href="#">compositelimages</a>	1	21.313 s	21.141 s	
<a href="#">Axl</a>	29021	4.625 s	2.266 s	
<a href="#">imfilter</a>	70	2.359 s	0.078 s	
<a href="#">warpimage</a>	1	2.047 s	0.703 s	
<a href="#">images\private\imfilter_mex (MEX-function)</a>	70	1.813 s	1.813 s	
<a href="#">numerics&gt;norm (Builtin-function)</a>	440895	1.734 s	1.734 s	
<a href="#">interp2</a>	4	0.906 s	0.031 s	
<a href="#">numerics&gt;inv (Builtin-function)</a>	32560	0.906 s	0.906 s	
<a href="#">numerics&gt;_times_transpose (Builtin-function)</a>	53098	0.891 s	0.688 s	
<a href="#">numerics&gt;mtimes (Builtin-function)</a>	257204	0.813 s	0.813 s	
<a href="#">m_interpreter&gt;feval (Builtin-function)</a>	1049	0.797 s	0.031 s	
<a href="#">Ransac</a>	1	0.734 s	0.469 s	
<a href="#">libmwbuiltins&gt;_colonobj (Builtin-function)</a>	154863	0.578 s	0.578 s	
<a href="#">images\private&gt;writejpg</a>	1	0.531 s	0.031 s	

(a)傳統演算法

Function name	Calls	Total Time	Self Time*	Total Time Plot (dark band = self time)
<a href="#">stitch</a>	1	137.016 s	5.797 s	
<a href="#">Extract_Feat</a>	1	50.391 s	38.203 s	
<a href="#">takesample</a>	342576	38.234 s	38.234 s	
<a href="#">subsampling</a>	6	37.438 s	5.328 s	
<a href="#">compositelimages</a>	1	33.625 s	33.406 s	
<a href="#">warpimage</a>	1	7.188 s	3.328 s	
<a href="#">Axl</a>	16966	2.891 s	1.641 s	
<a href="#">imfilter</a>	70	2.156 s	0.047 s	
<a href="#">interp2</a>	4	2.109 s	0.031 s	
<a href="#">interp2\linear</a>	4	1.766 s	1.625 s	
<a href="#">numerics&gt;mtimes (Builtin-function)</a>	531047	1.719 s	1.719 s	
<a href="#">images\private\imfilter_mex (MEX-function)</a>	70	1.641 s	1.641 s	
<a href="#">m_interpreter&gt;feval (Builtin-function)</a>	1049	0.859 s	0.031 s	
<a href="#">numerics&gt;norm (Builtin-function)</a>	273406	0.828 s	0.828 s	
<a href="#">numerics&gt;sum (Builtin-function)</a>	150031	0.719 s	0.719 s	
<a href="#">imshow</a>	1	0.641 s	0.000 s	
<a href="#">images\private&gt;writejpg</a>	1	0.625 s	0.000 s	
<a href="#">bwfist</a>	3	0.516 s	0.000 s	
<a href="#">images\private&gt;euclid2 (MEX-function)</a>	3	0.469 s	0.469 s	
<a href="#">numerics&gt;_times_transpose (Builtin-function)</a>	29244	0.453 s	0.375 s	

(b)快速演算法

圖 4-3.1 傳統演算法與快速演算法比較圖

#### 4-4 轉換矩陣 H 的重覆使用性質(Reusability of Homography H)

前面三節介紹了傳統演算法與我們快速校正技術的運算效率比較，本節我們將驗證影像之間轉換矩陣(Homography)的重覆使用性質。2-4、2-5 說明了要縫合兩張影像，一定要經過特徵點的擷取與匹配並經過 RANSAC 來剔除掉錯誤的匹配點，最後在經過兩張影像的轉換關係，把某一張影像做歪斜並與另一張影像縫合，此轉換關係即為轉換矩陣(Homography)H，如式 4-1、式 4-2。而此轉換矩陣 H 即把兩張影像在做校正的關係式，我們覺得每次都要從頭到尾執行一次快速演算法求得轉換矩陣 H 再做校正太麻煩了，因此我們假定若在多 Sensor 並且 Sensor 相對位置彼此固定的情形下，轉換矩陣 H 是否可重複使用?若可以，表示在此前提之下，我們的快速校正演算法只要執行一次求得轉換矩陣 H，之後的校正就直接使用此轉換矩陣 H 來做校正，因為在 Sensor 相對位置彼此固定的情形下，理論上影像之間彼此的校正關係 H 也應該一樣。

$$P' = HP \quad \text{----- (式)4-1}$$

$$\begin{bmatrix} wx' \\ wy' \\ w \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad \text{----- (式)4-2}$$

而我們也做了簡單的實驗來驗證，如圖 4-4.1，圖 4-4.1(b)我們先把相機腳架的位子定死，並且在木板上作上記號，之後再將腳架水平移動一定的角度(腳架相對位置仍然不動)，在木板上作上第二個記號，如圖 4-4.1(a)，因此之後拍照時，只要把腳架擺在第一組記號當中，拍第一張照片，再將腳架水平移動到第二組記號當中，拍攝第二張照片，如圖 4-4.1(c)(d)，如此我們就可以簡單地模擬 Multi-Sensor 在相對位置固定的情形下，影像之間彼此的關係式，因為每組影像彼此之間的移動關係都是固定的。

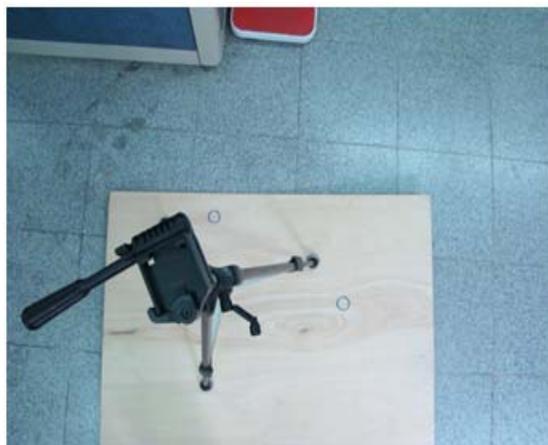
接著我們先對其中一組 Pattern 執行我們的快速校正技術，如圖 4-4.2，得到其轉換矩陣關係式為式 4-3，之後在其他組的 Pattern 校正當中，我們就不再



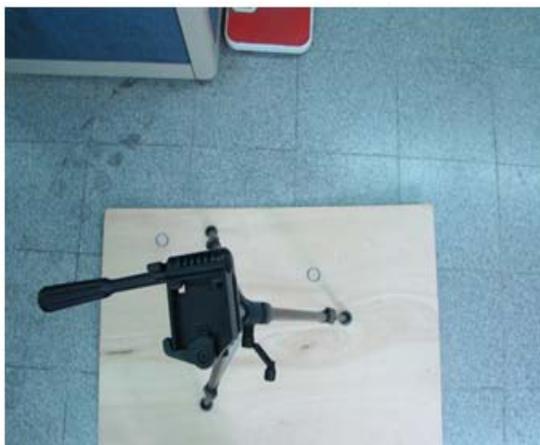
(a) 作上記號的木板



(b) 絕對位置已定死的腳架



(c) 拍攝第一張影像的位置



(d) 移動腳架到另一組記號中

圖 4-4.1 Multi-Sensor 相對位置固定情形下，轉換矩陣重複性的實驗步驟



圖 4-4.2 轉換矩陣  $H$  重複性質的基底 Pattern

從頭到尾執行快速演算法了，而是全部使用式 4-3 的轉換關係式來套用在每組的 Pattern 當中，在我們的實驗過程當中總共拍攝了 15 組的影像，圖 4-4.3 為其中的兩組影像校正的結果，注意在此步驟我們已經完全省略了前面特徵點擷取、

$$H = \begin{bmatrix} 0.8098 & 0.0548 & 204.0689 \\ -0.0939 & 0.9509 & 9.0957 \\ -0.0003 & 0.0000 & 1.0000 \end{bmatrix} \text{-----}(式)4-3$$

特徵點匹配與 RANSAC 剔除錯誤匹配點的步驟，而是直接使用式 4-3 的轉換矩陣 H(Homography)來做之後的校正，因為彼此之間校正轉換的關係已經知道，所以我們只要做傳統演算法的歪斜影像(Warp Image)與混合(Blending)即可，這樣的方法可以大幅地節省運算量，並且適用於硬體實作當中。由圖 4-4.3 當中我們不難發現，在相機相對位置彼此固定的情形下，即使不用執行前面特徵點擷取、特徵點匹配與 RANSAC 剔除錯誤匹配點的步驟，只是單純使用式 4-3 的轉換矩陣 H(Homography)來做校正，仍然可以校正出相當好的結果，如圖 4-4.3(a)(b)，幾乎都只有 1~3 個 Pixel 的模糊差距，此現象可能是因為水平面的些微誤差所造成，因此由此實驗我們可以得知當多 Sensor 並且 Sensor 相對位置彼此固定的情形下，轉換矩陣 H 的關係式是固定的。



(a)使用相同的轉換矩陣 H 所校正的影像範例一



(b) 使用相同的轉換矩陣  $H$  所校正的影像範例二

圖 4-4.3 使用相同的轉換矩陣  $H$  所校正的影像



## 第五章 結論與未來發展

近年來相機校正技術因為需要輔助校正物且機器的精確性誤差的關係，因此影像縫合技術在相機校正領域佔有越來越重要的地位。於本篇論文所談到的BUMS(Bottom-Up Maxima Selection)和 PRSC(Polarized Random Sample Consensus)快速演算法，因為在擷取特徵點的時候由內外往外擴散慢慢刪除，並且使用適應性的方法讓擷取特徵點的步驟停止，並且在剔除特徵點的步驟也使用投票特殊機制，使得匹配點容易呈現兩極化(Polarized)的現象，此兩個步驟都大大地節省 CPU time，加快速度但卻可以得到一樣的影像品質。對於我們在影像縫合(Auto-Stitch)的校正技術上，將有相當大的幫助。另外我們也發現在多台 Sensor 的角度、位置、距離都已經固定的前提下，其影像彼此之間的轉換關係是固定的，因此我們只要跑過一次完整的演算法即可求出彼此之間的轉換關係，所以之後的校正只要透過其固定的轉換關係即可校正，如此的發現非常有利於我們在硬體實現的突破。

在未來發展方面，運用快速影像縫合演算法於相機校正技術可運用於監視系統、軍事系統、Scene Motion、透視圖等等，如圖 5-1。如圖 5-1(a)顯示利用相機陣列監視一片樹叢，圖 5-1(b)顯示使用相機陣列校正技術透視後面的影像，此種技術非常適用於軍事作戰與人質救援。另外此種技術比起傳統演算法運算量大大節省，因此也適用於硬體實作，未來我們也將利用此演算法於硬體實作上來加以突破。



(a)



(b)

圖 5-1.1 透視圖(本圖來源出自[44])

圖 5-1.2 為一個多 sensor 的平台範例，每個刻度彼此之間相隔 30 度，並且在每個刻度之間嵌入一個 Sensor，透過這種方式使得彼此之間的關係固定，達到我們想要的固定 H 關係，但因為 Sensor 彼此的相對高度上也會有有些微誤差，因此此平台仍然有改良的空間。圖 5-1.3 為用此平台所攝影的其中一組相片及其縫合影像。最後，在我們的演算法中仍然有少數 Pattern 是會縫合失敗的，如圖 5-1.4。我們會發現兩張影像的縫合結果是成功的，但是三張影像的縫合結果是失敗的，會造成失敗的原因可能是因為相機與影像的焦距過近所導致，因為焦距很近時，把相機移動一點點就會使得影像的 Pixel 偏移很多，原本被擷取到的特徵點在相機移動後可能就變得不是特徵點了，這部份也是值得我們未來繼續探討的。

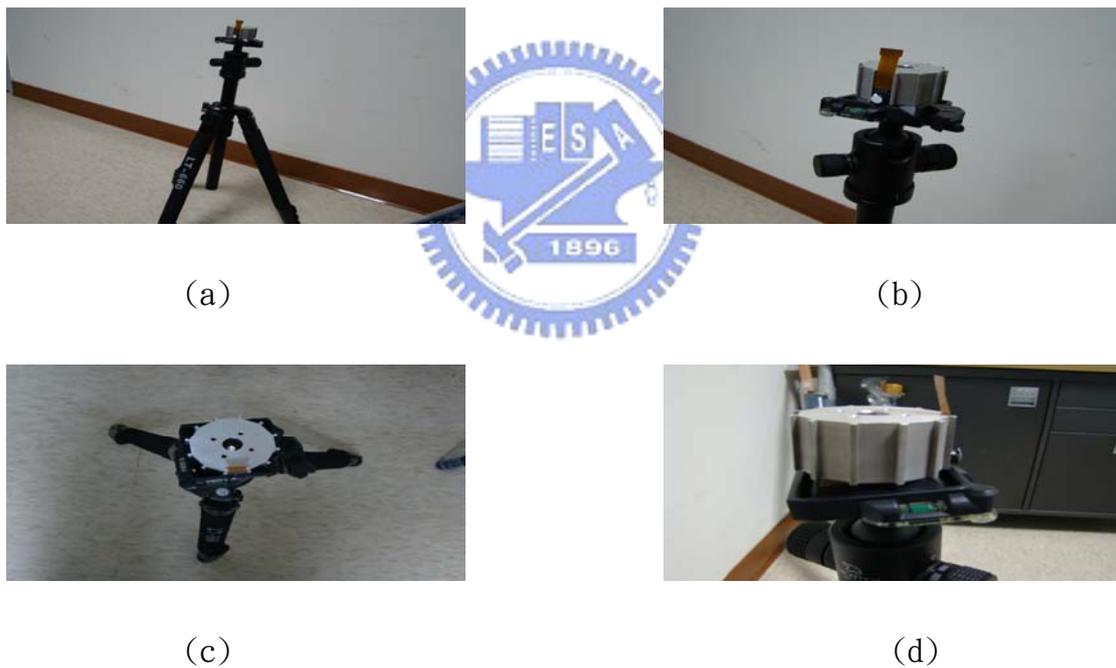


圖 5-1.2 多 Sensor 平台範例





(c)



(d)

圖 5-1.3 多 Sensor 平台影像及縫合影像



(a)影像一



(b)影像二



(c)影像三



(d)成功縫合兩張影像



(e)失敗地縫合三張影像

圖 5-1.4 焦距過近的縫合影像

## 參考文獻：

- [1] C. Rother and S. Carlsson. Linear multi view reconstruction and camera recovery using a reference plane. *International Journal of Computer Vision*, 49(2/3):117–141, 2002.
- [2] B. Bascle, A. Blake, and A. Zisserman. Motion deblurring and super-resolution from an image sequence. In *Proceedings of the 4th European Conference on Computer Vision (ECCV96)*, pages 312–320. Springer-Verlag, 1996.
- [3] D. Capel and A. Zisserman. Automated mosaicing with super-resolution zoom. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition (CVPR98)*, pages 885–891, June 1998.
- [4] J. Davis. Mosaics of scenes with moving objects. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition (CVPR98)*, pages 354–360, 1998.
- [5] M. Uyttendaele, A. Eden, and R. Szeliski. Eliminating ghosting and exposure artifacts in image mosaics. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition (CVPR01)*, volume 2, pages 509–516, Kauai, Hawaii, December 2001.
- [6] A. Agarwala, M. Dontcheva, M. Agarwala, S. Drucker, A. Colburn, B. Curless, D. Salesin, and M. Cohen. Interactive digital photomontage. In *ACM Transactions on Graphics (SIGGRAPH' 04)*, 2004.
- [7] D. Brown. Close-range camera calibration. *Photogrammetric Engineering*, 37(8):855–866, 1971.
- [8] D.B. Goldman and J.H. Chen. Vignette and exposure calibration and compensation. In *Proceedings of the 10th International Conference on Computer Vision (ICCV05)*, pages I: 899–906, 2005.
- [9] P. Debevec and J. Malik. Recovering high dynamic range radiance maps from

- photographs. *Computer Graphics*, 31:369–378, 1997.
- [10] H. Seetzen, W. Heidrich, W. Stuerzlinger, G. Ward, L. Whitehead, M. Trentacoste, A. Ghosh, and A. Vorozcovs. High dynamic range display systems. In *ACM Transactions on Graphics (SIGGRAPH'04)*, 2004.
- [11] R. Szeliski and S. Kang. Direct methods for visual scene reconstruction. In *IEEE Workshop on Representations of Visual Scenes*, pages 26–33, Cambridge, MA, 1995.
- [12] M. Irani and P. Anandan. About direct methods. In B. Triggs, A. Zisserman, and R. Szeliski, editors, *Vision Algorithms: Theory and Practice*, number 1883 in LNCS, pages 267–277. Springer-Verlag, Corfu, Greece, September 1999.
- [13] H. Sawhney and R. Kumar. True multi-image alignment and its application to mosaicing and lens distortion correction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(3):235–243, 1999.
- [14] H. Shum and R. Szeliski. Construction of panoramic mosaics with global and local alignment. *International Journal of Computer Vision*, 36(2):101–130, February 2000.
- [15] I. Zoghlami, O. Faugeras, and R. Deriche. Using geometric corners to build a 2D mosaic from a set of images. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition, Puerto Rico*. IEEE, June 1997.
- [16] P. McLauchlan and A. Jaenicke. Image mosaicing using sequential bundle adjustment. *Image and Vision Computing*, 20(9-10):751–759, August 2002.
- [17] R. Szeliski. Image Alignment and Stitching: A Tutorial1 Preliminary draft, September 27, 2004 Technical Report. MSR-TR-2004-92.
- [18] Chen, S. E. QuickTime VR – an image-based approach to virtual environment navigation. *Computer Graphics (SIGGRAPH'95)*, , 29–38. 1995.
- [19] Szeliski, R. Image mosaicing for tele-reality applications. In *IEEE Workshop on*

- Applications of Computer Vision (WACV'94)*, pages 44–53, IEEE Computer Society, Sarasota. 1994.
- [20] Szeliski, R. Creating Full View Panoramic Image Mosaics and Environment Maps. Microsoft Research.
- [21] irahu <http://www.iraful.com/workzone/stitch/index.php>
- [22] P. Heckbert. Projective Mappings for Image Warping Rendering, 15-862 .26 Sept 1995.
- [23] C. Harris and M.J. Stephens. A combined corner and edge detector. In Alvey Vision Conference, pages 147–152, 1988.
- [24] H. Moravec. Obstacle avoidance and navigation in the real world by a seeing robot rover. Technical Report CMU-RI-TR-3, Carnegie-Mellon University, Robotics Institute, 1980.
- [25] C. Schmid, R. Mohr, and C. Bauckhage. Evaluation of interest point detectors. *International Journal of Computer Vision*, 37(2):151–172, June 2000.
- [26] G. Derpanis. The Harris Corner Detector, October 27, 2004.
- [27] M. Brown, R. Szeliski, and S. Winder. Multiimage matching using multi-scale oriented patches. In *Proceedings of the Interational Conference on Computer Vision and Pattern Recognition (CVPR05)*, San Diego, June 2005.
- [28] D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.
- [29] J. Shi and C. Tomasi. Good features to track. In *Proceedings of the Interational Conference on Computer Vision and Pattern Recognition (CVPR94)*, Seattle, June 1994.
- [30] Y. Ke1, R. Sukthankar. PCA-SIFT: A More Distinctive Representation for Local Image Descriptors. Intel Research Pittsburgh In *Proceedings of the Interational Conference on Computer Vision and Pattern Recognition*

- (CVPR04).
- [31] D. Lowe, "Object recognition from local scale-invariant features," *International Conference on Computer Vision*, Corfu, Greece (September 1999), pp. 1150-1157.
- [32] D. Lowe, "Local feature view clustering for 3D object recognition," *IEEE Conference on Computer Vision and Pattern Recognition*, Kauai, Hawaii (December 2001), pp. 682-688.
- [33] M. Fischler and R. Bolles. Random sample consensus: A paradigm for model fitting with application to image analysis and automated cartography. *Communications of the ACM*, 24:381–395, 1981.
- [34] P. Torr. Bayesian model estimation and selection for epipolar geometry and generic manifold fitting. *International Journal of Computer Vision*, 50(1):35–61, 2002.
- [35] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004.
- [36] M. Brown, D. Lowe. Automatic Panoramic Image Stitching using Invariant Features. *International Journal of Computer Vision (ijcv)*, 2007.
- [37] M. Brown and D. Lowe. Recognising panoramas. In *Proceedings of the 9th International Conference on Computer Vision (ICCV03)*, volume 2, pages 1218–1225, Nice, October 2003.
- [38] J. Beis and D. Lowe. Shape indexing using approximate nearest-neighbor search in highdimensional spaces. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition (CVPR97)*, pages 1000–1006, 1997.
- [39] P. Burt and E. Adelson. A multiresolution spline with application to image mosaics. *ACM Transactions on Graphics*, 2(4):217–236, 1983.
- [40] ROGER Y. TSAI. A Versatile Camera Calibration Technique for 3D Machine

Vision. IEEE J. Robotics & Automation, RA-3, No. 4, August 1987, pp. 323-344.

[41] Zhengyou Zhang. A Flexible New Technique for Camera Calibration. December 2, 1998. Technical Report.MSR-TR-98-71.

[42] [http://mpac.ee.ntu.edu.tw/~ck/project\\_panorama/#SIFTTutorial](http://mpac.ee.ntu.edu.tw/~ck/project_panorama/#SIFTTutorial)

[43] <http://www.csie.ntu.edu.tw/~cyu/courses/vfx/05spring/assignments/proj2/>

[44] B. Wilburn and N. Joshi. High Performance Imaging Using Large Camera Arrays.In ACM Transactions on Graphics, Vol 24, No 3, July 2005, pp. 765-776

