

國立交通大學
電機與控制工程研究所

碩士論文

無校正視覺伺服系統之
影像 Jacobian 矩陣估測研究

The Estimation of Image Jacobian Matrix
for Image-based Uncalibrated Visual Servoing

研究生：許育綸

指導教授：胡竹生 博士

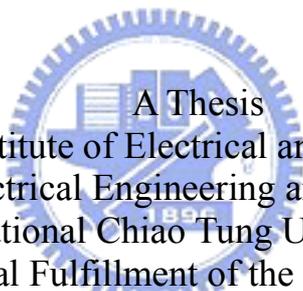
中華民國九十八年六月

無校正視覺伺服系統之
影像 Jacobian 矩陣估測研究

The Estimation of Image Jacobian Matrix
for Image-based Uncalibrated Visual Servoing

研究生：許育綸 Student：Yu-Lun Hsu
指導教授：胡竹生 博士 Advisor：Jwu-Sheng Hu

國立交通大學
電機與控制工程學系
碩士論文



A Thesis
Submitted to Institute of Electrical and Control Engineering
College of Electrical Engineering and Computer Science
National Chiao Tung University
in partial Fulfillment of the Requirements
for the Degree of Master
in

Electrical and Control Engineering

June 2009

Hsinchu, Taiwan, Republic of China

中華民國九十八年六月

無校正視覺伺服系統之 影像 Jacobian 矩陣估測研究

研究生：許 育 綸

指導教授：胡 竹 生 博士

國立交通大學電機與控制工程研究所碩士班

摘要

本文建立一個以影像為基礎的視覺回授系統，此系統使用不需要校正的相機，估測並更新影像 Jacobian 矩陣，進而控制機器手臂到達所需的位置。本文將視覺回授系統分成兩大部分。第一部份是影像特徵的擷取，我們把電腦視覺的平均位移法，應用在特徵擷取上，以提高特徵追蹤的準確率。平均位移法是以空間中的色彩分佈為特徵，並且以相似函數來鎖定追蹤的對象，在複雜的環境，也能即時準確的擷取特徵。第二部分是視覺回授的控制，我們利用簡單的演算法，估測相機與機器手臂末端的關係。相較於其他文獻，此方法易於實現。

在模擬的部分，本文透過仿真的相機與機器手臂，可以有效的模擬。而實作的部份，本文僅使用一台電腦，來完成所有的軟體開發，包括對機器手臂的傳輸與雙相機的處理，以解決全域與區域的 Jacobian 問題。最後，在實驗的部份，我們選擇使用球具的球類運動，加入了運動偵測的演算法，展示機器手臂連續擊球的任務，並將結果加以分析與討論。

The Estimation of Image Jacobian Matrix for Image-based Uncalibrated Visual Servoing

Student : Yu-Lun Hsu

Advisor : Prof. Jwu-Sheng Hu

Institute of Electrical and Control Engineering

Abstract

This thesis proposes an image-based visual servo system, which can estimate and update Image Jacobian Matrix to control a robot arm without camera calibration. There are two parts in the visual servo system. The first one is image feature extraction. We apply the Mean-Shift algorithm in order to improve the performance of feature tracking. Mean-Shift algorithm, which takes the color distribution as a model, is based on the similarity measure function to decide a tracking candidate. This algorithm works well in a real-time and complicated environment. The second part is the control of a manipulator to track visual objects. An easy-to-implement algorithm is proposed to find the relationship between the camera and the manipulator. It can be carried out easily as compared with other methods.

We use the camera and the manipulator emulator to simulate the system effectively. In the implementation, we use only one computer to develop all the software, including the communication to the manipulator and the processing of two cameras' images to solve the local and the global Image Jacobian matrices. Ball-hitting experiments such as the juggling task are presented to analyze the real-time performance of the proposed algorithms.

誌謝

本文能順利完成，首先要感謝我的指導教授—胡竹生教授、口試委員楊谷洋教授與蔡文祥教授的提醒，其中楊教授的鼓勵與蔡教授的親切讓我終生難忘。胡教授是我在大學時專題的指導教授，教授的博學多聞、態度熱誠等等，開拓了我的視野，激發了我的潛能，在這裡真的要好好謝謝他。

接著，我要感謝嵌入式系統實驗室的好伙伴們，例如幫我校正英文的嘟嘟、助教合作愉快的 judo、跟我搶著用 DV 的肉鬆、與我同一屆畢業的文棋、營隊高手的 gum、常跟我練習英文的活多福、屢屢幫我突破瓶頸的阿吉、一起去深圳比賽的興哥、帶我們拿第一名的永融、幫忙維修網站的大師兄、數學超強的學姊鏗元、讓我沉住氣的唐哥、已經畢業還常被我們諮詢的 papa 與 hcy、從總務退休還要被我煩的俊宇與宗敏、傳承電影給我的劉大、幫實驗室打點很多事的淑伶、常陪我聊天的小蔡、用功的塞門、很靦腆的阿 him、超像馬拉桑的沛錡、低調的聖翔、復出的阿法等等。

此外，也要感謝與我同名的黃育綸教授，與她的實驗室那群陪我玩樂的同學們，阿彬、道哥、雕兄、宗勳、小狼、小叮嚀，還有那些還與我有聯絡的朋友們，與那些默默關心我的朋友們，真誠的感謝你們。最後，我要感謝扶養我的老爸、教育我的老媽、一同長大的老哥老弟、每年團圓一次的家人們與在天堂的奶奶、每一位喜歡過我、愛過我的人們。

在交大的六年，讓我從搖滾青年，蛻變成宅宅工程師。我曾經對未來迷惘、對社會失望，而年輕只是一種過程，最終我們還是必須面對這世界。除了念書，抽空回家、聚餐約會、辦營隊、玩樂團、弄社團、搞梅竹等等，這之間夾雜的友情、愛情與稍縱即逝的親情，讓我學著如何應對關於成長這回事。如果這真的是一種過程，我要感謝至今陪我兩年的雷子曦與她的家人，他們讓我重新相信了一些事情。如果沒有她就沒有現在的我，或是說沒有她的話，我無法想像現在的我是什麼樣子。

目錄

摘要.....	i
Abstract.....	ii
誌謝.....	iii
目錄.....	iv
表目錄.....	v
圖目錄.....	vi
一、緒論.....	1
1.1 研究背景動機.....	1
1.2 相關架構分類.....	1
1.3 文獻回顧探討.....	4
1.4 論文成果貢獻.....	6
1.5 章節概要.....	6
二、相機系統.....	7
2.1 系統簡介.....	7
2.2 相機模型.....	7
2.3 相機參數.....	9
2.4 其他模型.....	10
2.5 相機校正.....	11
三、特徵擷取.....	15
3.1 系統簡介.....	15
3.2 方法分析.....	16
3.3 特徵追蹤.....	18
3.4 實驗模擬.....	22
四、視覺回授.....	26
4.1 系統簡介.....	27
4.2 方法分析.....	27
4.3 影像回授.....	31
4.4 實驗模擬.....	37
五、模擬實作.....	42
5.1 任務介紹.....	43
5.2 模擬簡介.....	44
5.3 模擬結果.....	47
5.4 實作簡介.....	50
5.5 實作結果.....	57
六、結論.....	65
七、相關文獻.....	66

表目錄

表格 1. 相機校正後實際的內部參數表.....	14
表格 2. 平均位移法常用的 kernel 函數表.....	19
表格 3. Logitech Quick-cam 規格.....	52
表格 4. Mitsubishi RV-M2 手臂規格.....	53
表格 5. Mitsubishi RV-M2 控制器規格.....	53
表格 6. Visual C++與 MFC 版本對照表.....	55
表格 7. 球棒與球的打擊實驗表.....	63
表格 8. 圖形處理的時間測試表.....	64



圖目錄

圖 1. 位置基礎視覺觀察運動架構圖.....	2
圖 2. 位置基礎視覺伺服控制架構圖.....	2
圖 3. 影像基礎視覺觀察運動架構圖.....	3
圖 4. 影像基礎視覺伺服控制架構圖.....	3
圖 5. 相機幾何座標與透鏡失真圖.....	8
圖 6. 針孔相機模型圖解.....	10
圖 7. 相機校正實驗圖解.....	12
圖 8. 實際校正步驟圖.....	13
圖 9. 校正前後比較圖.....	14
圖 10. 影像空間濾波器圖解.....	17
圖 11. 機率分布的函數比較圖.....	19
圖 12. 平均位移演算法流程圖.....	21
圖 13. 主要成分分析圖.....	22
圖 14. 平均演算法的追蹤結果示意圖.....	23
圖 15. 平均位移法的誤差圖與遞迴次數圖.....	24
圖 16. SONY EVI-D70 PTZ 攝影機與 EVERFOCUS 彩色 TFT LCD 螢幕.....	25
圖 17. 平均位移法的即時追蹤結果圖.....	25
圖 18. 視覺回授示意圖.....	28
圖 19. 遞迴影像 Jacobian 矩陣即時估測法流程圖.....	35
圖 20. 排除異常值之遞迴即時估測法流程圖.....	37
圖 21. 遞迴即時估測法模擬示意圖.....	38
圖 22. 遞迴即時估測法模擬誤差圖.....	39
圖 23. 排除異常值之遞迴即時估測法模擬示意圖.....	40
圖 24. 排除異常值之遞迴即時估測法模擬誤差圖.....	40
圖 25. 視覺回授硬體軟體分配架構圖.....	42
圖 26. MATLAB 機器手臂工具箱原始使用介面圖示.....	45
圖 27. MATLAB 機器手臂工具箱新增顯示函數圖示.....	45
圖 28. MATLAB 世界座標系與相機座標系模擬圖.....	46
圖 29. MATLAB 模擬程式流程圖.....	47
圖 30. 相機影像誤差造成的位置誤差模擬圖.....	49
圖 31. 手臂位置誤差造成的位置誤差模擬圖.....	50
圖 32. 影像基礎視覺觀察運動實作架構圖.....	51
圖 33. 實作使用的相機系統.....	52
圖 34. 實作使用的機器手臂.....	54
圖 35. 機器手臂的工作範圍.....	54
圖 36. 機器手臂五軸示意圖.....	54
圖 37. MFC 與 OpenCV 實際程式流程圖.....	56

圖 38. 棒球與手套的位置判斷實驗.....	58
圖 39. 球拍與球的位置判斷實驗.....	59
圖 40. 運動偵測程式流程圖.....	60
圖 41. 球棒與球的打擊實驗前半段.....	61
圖 42. 球棒與球的打擊實驗後半段.....	62
圖 43. 球棒與球的打擊實驗狀況一.....	62
圖 44. 球棒與球的打擊實驗狀況二.....	63



一、緒論

1.1 研究背景動機

在眾多的機器人感測器之中，視覺感測有許多優點。第一、機器人視覺感測相似於人類的視覺感測[1]，比起其他感官，人類的視覺行為較容易撰寫成演算法。第二、視覺感測不需要直接的觸碰環境[1]，也能有相對較佳的精確度。

機器人的視覺研究類似於電腦視覺，能作到追蹤、偵測、定位等工作[2]，若是機器人利用視覺感測的資訊來控制機器手臂，進而影響感測的環境，就可以稱之為視覺回授(visual feedback)[1][3]。此概念由 Shirai 與 Inoue[4]所提出，是為了增加機器人動作的準確度，整合了機器手臂控制與視覺感測，這方面的研究包含控制理論、機器人運動學、動力學、相機校正、高速影像處理與即時運算。簡單來說，機器人不再只是被動觀察環境，而是可以主動改變觀察的內容，甚至可以改變觀察角度。對於機器人來說，如何善用視覺感測，是相當重要的議題。

基於上述的需求與應用，過去已經有不少此方面的研究成果。由於這些研究是屬於跨領域的範圍，因此在各領域下，也有更多更深入的探討，而整體的研究必須在各領域發展到一定的程度才能突破。幸運的是，近年來機器人技術與電腦的升級，有助於此項研究的推進[1]。過去關鍵的問題，譬如視覺感測器、計算量等，已經能夠從軟硬體方面改善，以往機器人比較難做到的任務也能完成。在這些知識基底上，我們能夠改善視覺回授的表現，來解決更多的課題與任務。

1.2 相關架構分類

Shirai 與 Inoue[4]在 1973 年整合機器手臂控制與視覺感測，描述視覺回授的概念，也就是先觀察再移動(looking then moving)，如此一來結果將由兩者各自的準確度來決定。在 1979 年，Hill 與 Park[5]提出視覺伺服(visual servo)的觀念，建立新的系統。隔年 Sanderson 與 Weiss[6]整理分析出四種系統架構，依照兩個問題來分類，第一、機器手臂是否有各關節角度的回授。第二、得到視覺感測器

的資訊後，是否直接使用影像特徵來定位(image-based)，或是算出空間中的位置來定位(position-based)。以下我們依序介紹這四種系統架構[1][3][7]：

1. 位置基礎視覺觀察運動架構(Dynamic position-based look-and-move structure)

此架構以位置為基準，先利用理論算出各關節角度，再由控制器移動手臂，最後用視覺感測器來擷取影像特徵，估算出手臂的位置。

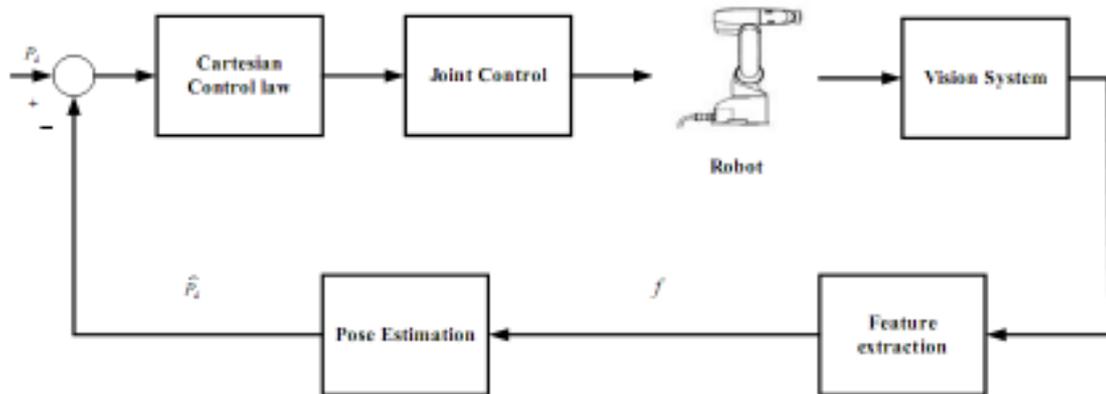


圖 1. 位置基礎視覺觀察運動架構圖

2. 位置基礎視覺伺服控制架構(PBVS, Position-based visual servo structure)

此架構以位置為基準，先利用理論算出各關節的角度之後，直接控制手臂，最後用視覺感測器來擷取影像特徵，估算出手臂的位置。

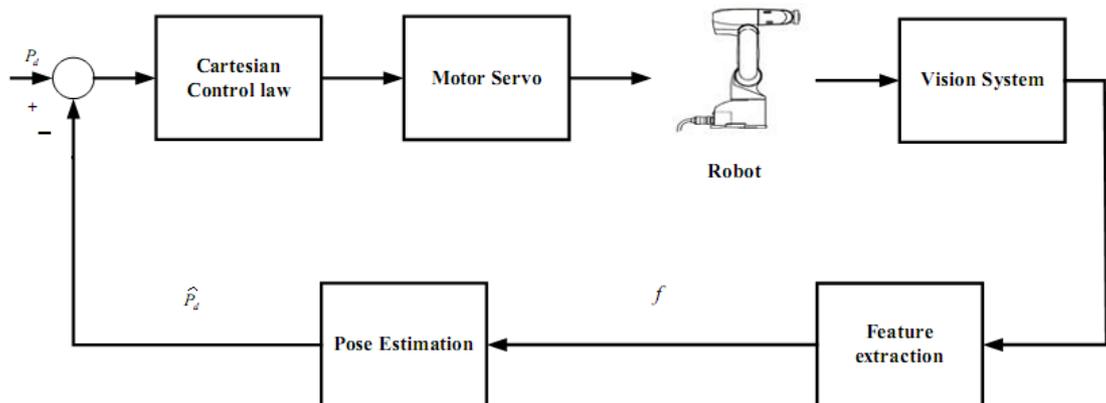


圖 2. 位置基礎視覺伺服控制架構圖

3. 影像基礎視覺觀察運動架構(Dynamic image-based look-and-move structure)

此架構以影像為基準，先利用理論算出各關節角度，再由控制器移動手臂，

最後用視覺感測器來擷取影像特徵，比對影像特徵的變化。

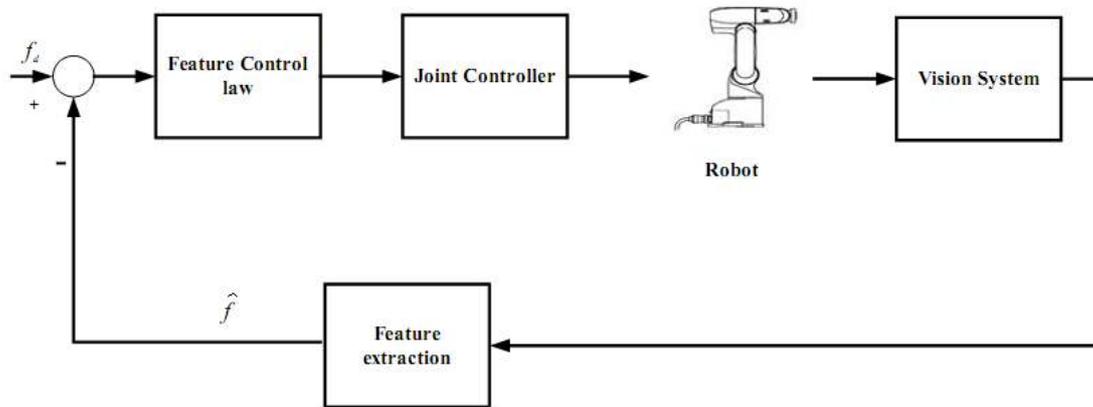


圖 3. 影像基礎視覺觀察運動架構圖

4. 影像基礎視覺伺服控制架構(IBVS, Image-based visual servo structure)

此架構以影像為基準，先利用理論算出各關節的角度之後，直接控制手臂，最後用視覺感測器來擷取影像特徵，比對影像特徵的變化。

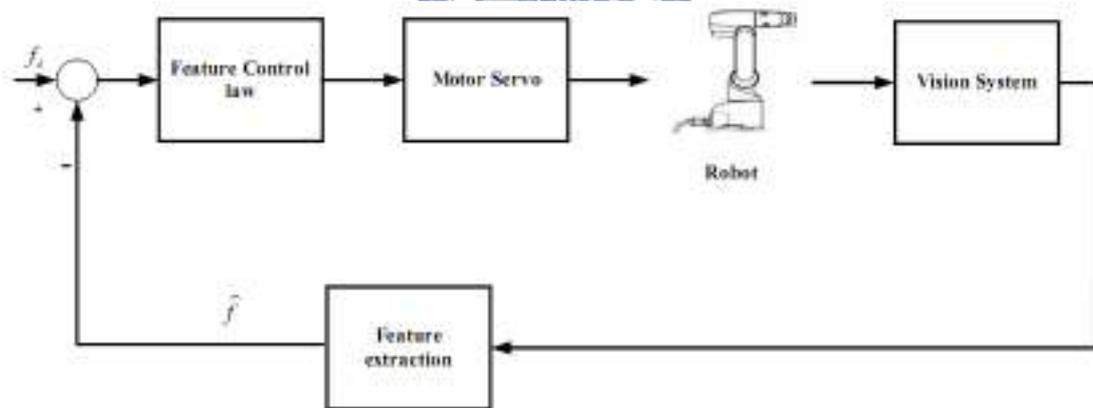


圖 4. 影像基礎視覺伺服控制架構圖

依照不同的軟硬體設計，可以選擇不同的系統架構，也有不同的優缺點。以位置為基礎的話，優點是卡式座標系的轉換比較容易，而缺點是必須校正相機的參數，對於校正參數的誤差也非常敏感[1]。相機校正的參數可以分成外部參數 (external parameters) 與內部參數 (intrinsic parameters)，外部參數包含相機的位置與角度，內部參數包含焦距、透鏡失真、影像中心、影像傾斜，這方面的研究目前已經相當的完整，譬如 R. Y. Tsai [8] 的校正技術或是 Zhang [9] 的校正模型。以

影像為基礎的話，優點是校正參數的誤差比較不影響結果，缺點是特徵擷取變得非常重要，這裡可以參考 Jang 與 Bien[10]的一般影像特徵擷取的方法。

除了上述的分類方法，根據視覺感測器的架設位置，可以分成 eye-in-hand 與 eye-to-hand 兩種[1][3]。eye-in-hand，顧名思義是將感測器裝在機器手臂上，故手臂與感測器的相對位置是已知的。而 eye-to-hand 是將感測器裝在工作環境中，感測器與手臂移動是各自獨立的。如何搭配視覺感測器的架設，來選擇適當的系統架構，可以參考 Hutchinson et al.[1]的指導文獻與 Corke[7]的回顧文獻。

理論上，因為採用以位置為基礎的架構需要相機校正，所以通常將視覺感測器固定，而不論感測器是否移動，都可以選用以影像為基礎的架構。然而，無論感測器是藉由機器手臂移動[11]，或是本身有轉動機制來改變觀察的內容[12]，皆可稱之為主動式視覺(active vision)，部份文獻將這個領域獨立出來討論，本文不將重點放在這個部份，這邊不再深入撰寫，更多的細節可參考[11-13]。

1.3 文獻回顧探討



引用[1][3][7]的定義，我們利用系統架構，將文獻分門別類，可發現大部分是採用以位置為基礎的系統，再搭配上相機校正的技術[8][9]，此種方法優點是系統簡單容易實現，缺點是對機器人來說，使用雙眼立體視覺重建環境時，校正誤差將嚴重影響結果[1]。若是以影像為基礎，雖然不必將影像換算回卡式座標系，但必須將影像移動與物體移動的關連性，利用其他方式來表示。在1987年，Sanderson et al.[14]提出了 Image Jacobian 的概念。此方法在 Papanikolopoulos 與 Khosla[15][16]與 Espiau et al.[17]的文獻中雖改用不同的名詞，但基本觀念是相同的。接著，許多文獻開始研究決定 Jacobian 的方法，譬如區域線性[17]、估測法[18-20]、預測法[21]、實驗法[22]等等，各種方法有各自的使用的條件，其中[18][19]不仔細考慮相機與機器人的模型，而是觀察靜態的目標來估測，相反的，[23][24]是依照其模型來觀察動態的目標。最後，Piepmeier et al.[25][26]結合以上的優點，

在不根據模型的情況下，利用靜態相機，追蹤動態目標的 visual servoing。基本上，靜態的 eye-to-hand 能完成大範圍的任務，而精確度比 eye-in-hand 低。

在 1994 年，Yoshimi et al.[11] 提出不需要校正的主動式視覺伺服概念(Active, Uncalibrated Visual Servoing)，之後 N. Hollinghurst et al.[27]加入立體視覺系統，Hager et al.[28]則是修正了估計誤差。除此之外，Chaumette et al.[29]、Jang 與 Bien[10]、Papanikolopoulos 與 Khosla[15][16]延續 eye-in-hand 的高解析度優點，然而 eye-in-hand 的系統容易被兩種狀況限制住，一、相機或機器人模型。二、需要靜態的目標。在[31]，Asada 與 Hosoda 擴展到移動的目標。最後，[32]在不根據模型的狀況下，解決不需要靜態的參考目標，來完成 visual servoing 的系統。理論上，動態的 eye-in-hand 雖然有比較好的解析度，但缺乏全域的資訊。

在實驗的評估方面，Wijesoma et al.[33]展示了兩個自由度的實現，Chan et al.[34]則是作到 peg-in-hole 的動作，在 1997 年，Jagersand 在[35]有深入的整理，朝向更多自由度、更通用的演算法來推進。事實上，以往的研究，偏向解決區域性的問題，在 2007 年，Jagersand 在[36][37]將區域推廣到全域，使用 K-Nearest Neighborhood (K-NN)演算法來處理。另外，也可以利用兩個相機來解決，譬如在[19]的實驗展示，還有[38][39]所提出的系統架構，這方面的研究不屬於雙相機的立體視覺，而是綜合 eye-in-hand 與 eye-to-hand 的合作[40]，結合兩種架構的優點，混合全域與區域的資訊，增加了任務的可執行度與準確度。

以上整理了以影像為基礎的架構，探討重點在研究 Jacobian 估測法的文獻。不論在 eye-to-hand 或 eye-in-hand，都已經可以作到追蹤動態目標，而不用事先知道模型的資訊，而相關理論的證明、穩定度的討論、控制器的選用、實作等等也有足夠的文獻可以支持[37]。至於全域 Jacobian 的問題，除了上述的全域演算法之外，也可以改變硬體架構，譬如用雙相機合作來解決。值得注意的是，上述的機器手臂系統，其輸入可能是手臂末端的位置，或是各關節角度[32]，故如何針對各種硬體，撰寫適合的軟體，是此研究的關鍵，也是文獻最值得參考的地方。

除此之外，系統估測的演算法已經十分繁多，而且針對任務的種類，將系統

特化的研究也趨於完善，而對於一般性的任務，卻較少有詳細的探討。這些方法無論使用伺服控制架構，或是觀察運動架構，基本骨架都是估測出影像特徵移動與物體移動的關連，故影像處理與任務也較簡單，像是 peg-in-hole、pick-and-place 等等。實際上，利用改善估測法與擷取有效的特徵，來確保低運算量與高強健度，才是最根本的問題。多數的文獻將資源分配在後端，前端只有簡單的影像處理。在這邊我們可以活用電腦視覺研究的知識基底[41-44]，發揮在影像處理上，這樣即可在更複雜的環境，用機器手臂執行更艱難的任務，並且達到即時的反應。

1.4 論文成果貢獻

本文利用 Image Jacobian 矩陣，估測無校正的相機與機器手臂末端的關係，達成視覺回授，同時也利用雙相機合作，解決全域與區域的問題。在演算法部分，相機影像的特徵擷取可以在複雜的環境下，達到即時追蹤，而視覺回授可以透過 RS-232 的傳輸，從電腦端即時命令機器手臂的動作。在模擬的部份，本文介紹仿真的機器手臂軟體，達到有效的模擬。而實作時，我們只用一台電腦完成所有的軟體開發，滿足即時的運算。最後在實驗的部份，我們選擇了使用球具的球類運動來展示，像是追球、擊球、挑球等等，完成整個視覺回授的系統研究。

1.5 章節概要

本文可以分成四大部分：第一章的概括介紹、第二章至第四章的系統分析、第五章的系統實現、第六章的結論。其中第二章到第四章是個別分析系統的內部架構，而第五章是討論整個架構如何模擬與實作。這兩大部份的介紹如下：

第二章：系統硬體的模型，我們省略機器手臂模型，只分析相機模型。

第三章：特徵擷取的推導與模擬，包含演算法討論與其他方法的比較。

第四章：視覺回授的推導與模擬，包含演算法討論與其他方法的分析。

第五章：系統的模擬實作，選擇相關硬體後，用軟體實現演算法來完成實驗。

二、相機系統

本文強調的是以影像為基礎的架構，而以位置為基礎的研究可以參考[1]。為了方便描述，我們將整個系統切割成各個子系統，接著分析每個子系統，最後再將它們組合起來，探討整體的架構。依照研究領域與前面分類的系統方塊圖，我們可以將整個系統略分成四大塊，分別是機器手臂系統、相機系統、特徵擷取系統、視覺回授系統，其中前兩者以硬體為主，後兩者是以軟體為主。由於每篇文獻都有各自不同的機器手臂系統，因此我們先不限定系統是採用伺服控制架構 (visual servo structure) 或觀察運動架構 (dynamic look-and-move structure)，讓整個討論更加的通用，接著我們依照順序介紹各個子系統，並且將它們方程式化。

2.1 系統簡介

相機系統的輸入是觀察的內容，輸出是數位的影像。一般來說，CCD 內部的光電轉換工作晶片，將光強度訊號轉成類比訊號後，經由影像擷取卡或者 A/D 卡，將類比影像資訊轉換成數位資訊。不同的相機有不同的晶片，其輸出的格式也略有不同，處理方式也有所不同[3]。相同的是，整個相機成像可以分成四個步驟，由這些步驟定義了相機的外部與內部參數，此為 R. Y. Tsai 在 [8] 提出的模型，而 Zhang [9] 將此模型以另一種方式表示，定義了同樣的參數。以下我們採用最常見的針孔模型 (pinhole model)，依序介紹兩種相機模型。

2.2 相機模型

R. Y. Tsai 的模型是由四種步驟構成，下圖是前三步驟的相機幾何模型。

1. 由世界座標系轉換成相機座標系

在 3D 的世界座標系當中， (x_w, y_w, z_w) 是質點 P 的座標值。而在 3D 的相機座標系當中， (x, y, z) 也是質點 P 的座標值。依照幾何座標系轉換，我們可以將 (x_w, y_w, z_w) 與 (x, y, z) 表示成下列的關係，其中 \mathbf{R} 是旋轉矩陣， \mathbf{T} 是平移矩陣。

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \mathbf{R} \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} + \mathbf{T}, \text{ where } \mathbf{R} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \text{ and } \mathbf{T} = \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} \quad (1)$$

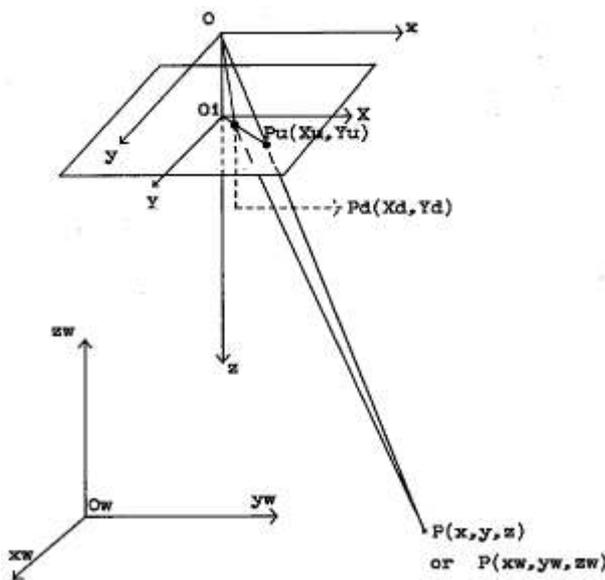


圖 5. 相機幾何座標與透鏡失真圖

2. 由相機座標系轉換成理想影像座標系

本步驟用針孔模型，以透視投影(perspective projection)來成像，如上圖所示， (X_u, Y_u) 是理想影像座標系的座標，其中 O_1 為影像平面的中心， O_1 與 O 的距離 f 稱之為焦距(effective focal length)， (x, y, z) 與 (X_u, Y_u) 的關係如下：

$$X_u = f \frac{x}{z}, Y_u = f \frac{y}{z} \quad (2)$$

3. 由理想影像座標系轉換成透鏡失真座標系

上圖我們可以發現相機實際成像的位置 (X_d, Y_d) 與理想狀況 (X_u, Y_u) 不同，此現象被稱為透鏡失真，其中失真係數可以用 κ_1 與 κ_2 表示，它們的關係如下：

$$\begin{cases} X_d = X_u - D_x \\ Y_d = Y_u - D_y \end{cases} \quad (3)$$

$$\begin{cases} D_x = X_d(\kappa_1 r^2 + \kappa_2 r^4) \\ D_y = Y_d(\kappa_1 r^2 + \kappa_2 r^4) \end{cases} \text{ and } r = \sqrt{X_d^2 + Y_d^2}$$

由[8]，我們可以知道透鏡失真分成 radial 與 tangential 兩種，然而在實驗中我們發現，只需要考慮 κ_1 的 radial 失真，其他項影響甚小，可以忽略不計。

4. 由透鏡失真座標系轉換成實際影像座標系

最後， (X_d, Y_d) 經由運算後，可以轉換成實際的影像 pixel (X_f, Y_f) ，其中 (C_x, C_y) 是中心影像 pixel，而 s_x 是比例參數，其他則是參閱文件的設計參數。

$$\begin{cases} X_f = s_x d_x^{-1} X_d + C_x \\ Y_f = d_y^{-1} Y_d + C_y \end{cases}, \text{ where } d_x' = d_x \frac{N_{cx}}{N_{fx}} \quad (4)$$

2.3 相機參數

上面四個步驟，定義了六個外部參數，五個內部參數，合計十一個校正參數。其中 \mathbf{T} 矩陣有三個參數， \mathbf{R} 矩陣雖然有九個元素，但可以用 Euler angles yaw θ 、pitch ϕ 、tilt ψ 來簡化成三個參數，其中的關係如下：

$$\mathbf{R} = \begin{bmatrix} \cos \psi \cos \theta & \sin \psi \cos \theta & -\sin \theta \\ -\sin \psi \cos \phi + \cos \psi \sin \theta \cos \phi & \cos \psi \cos \phi + \sin \psi \sin \theta \sin \phi & \cos \theta \sin \phi \\ \sin \psi \sin \phi + \cos \psi \sin \theta \cos \phi & -\cos \psi \sin \phi + \sin \psi \sin \theta \cos \phi & \cos \theta \cos \phi \end{bmatrix} \quad (5)$$

而五個內部參數，分別是上述的 f 、 κ_1 、 s_x 、 C_x 、 C_y ，其中 f 為 focal length， κ_1 為失真係數， s_x 是 scale factor， (C_x, C_y) 是中心影像的 pixel 值。方便起見，將各個步驟代入後，求得最後 (X_f, Y_f) 的與最初 (x_w, y_w, z_w) 的關係：

$$\begin{cases} s_x^{-1} d_x^{-1} X_f + s_x^{-1} d_x^{-1} X_f \kappa_1 r^2 = f \frac{r_1 x_w + r_2 y_w + r_3 z_w + T_x}{r_7 x_w + r_8 y_w + r_9 z_w + T_z} \\ d_y^{-1} Y_f + d_y^{-1} Y_f \kappa_1 r^2 = f \frac{r_4 x_w + r_5 y_w + r_6 z_w + T_y}{r_7 x_w + r_8 y_w + r_9 z_w + T_z} \end{cases} \quad (6)$$

上式右側外部參數，左側為內部參數。往後我們校正這些參數的時候，可以用已知的 (x_w, y_w, z_w) 與相機得到的 (X_f, Y_f) ，來估算出這些參數的值。

2.4 其他模型

對以位置為基礎的架構來說，要將機器手臂的末端移動到想要的位置，校正這些參數是絕對必要的，因為校正誤差對系統影響非常大，所以校正結果越準確越好。以影像為基礎的話，因為是將機器手臂末端移動到想要的影像位置，所以誤差影響較少。實作上，R. Y. Tsai [8] 的模型雖然觀念容易想像，比較好理解，但實際操作比較煩瑣，方程式較多，要寫成演算法較複雜。幸運的，Zhang [9] 將此模型重新寫成如下的矩陣形式，對於我們用軟體來校正有相當大的助力。

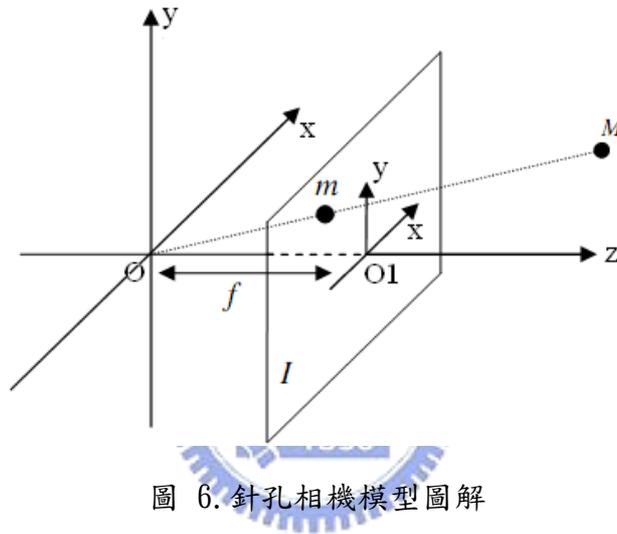


圖 6. 針孔相機模型圖解

上圖座標定義與之前相同， O 為光學中心(optical center)， O_1 為主點(principal point)， f 為焦距(focal length)。空間中一點 M 投影至影像平面 I 為 m ，其中 M 為 (x, y, z) ， m 為 (u, v) ，其關係可以用齊次座標系(homogeneous coordinate)表示成：

$$\text{Let } \tilde{M} = \begin{bmatrix} M \\ 1 \end{bmatrix}, m = \begin{bmatrix} \frac{fX}{Z} \\ \frac{fY}{Z} \\ Z \end{bmatrix}, \tilde{m} = \begin{bmatrix} fX \\ fY \\ Z \end{bmatrix} \quad (7)$$

$$\Rightarrow \tilde{m} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = P\tilde{M}$$

上式假設影像平面 I 的原點是在 O_1 ，而實際上原點卻在 (p_x, p_y) ，故改寫成：

$$m = \begin{bmatrix} \frac{fX}{Z} + p_x \\ Z \\ \frac{fY}{Z} + p_y \end{bmatrix} \Rightarrow \tilde{m} = \begin{bmatrix} fX + Zp_x \\ fY + Zp_y \\ Z \end{bmatrix} = \begin{bmatrix} f & 0 & p_x & 0 \\ 0 & f & p_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = P\tilde{M} \quad (8)$$

為了方便，我們寫成定義了相機校正矩陣 \mathbf{K} (camera calibration matrix)：

$$\mathbf{K} = \begin{bmatrix} f & \gamma & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} \Rightarrow \tilde{m} = \mathbf{K}[\mathbf{I}|\mathbf{O}]\tilde{M} \quad (9)$$

其中 γ 是歪斜參數(skew factor)，是考慮到若影像兩軸發生不平行的狀況。

最後我們考慮比例參數 (u_x, u_y) 的問題，即可定義出相機的內部參數矩陣為：

$$\mathbf{K} = \begin{bmatrix} fu_x & \gamma & p_x \\ 0 & fu_y & p_y \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \alpha_x & \gamma & p_x \\ 0 & \alpha_y & p_y \\ 0 & 0 & 1 \end{bmatrix} \Rightarrow \tilde{m} = \mathbf{K}[\mathbf{I}|\mathbf{O}]\tilde{M} \quad (10)$$

接著我們只要乘上旋轉矩陣 \mathbf{R} 與平移矩陣 \mathbf{T} ，即完成了模型的定義：

$$\tilde{m} = \mathbf{K}[\mathbf{R}|\mathbf{T}]\tilde{M} \quad (11)$$

上面的矩陣，也定義了六個外部參數，五個內部參數，合計十一個校正參數。

其中 \mathbf{T} 與 \mathbf{R} 與上述相同，而 (p_x, p_y) 即是 (C_x, C_y) ，而 f 與 s_x 的參數合併成 α_x 與

α_y ，而新參數 γ 是 R. Y. Tsai 沒考慮到的參數。至於失真係數 κ_1 ，Zhang 在[9]的

作法是將得到的影像資訊作運算，利用最大相似函數，來估測 κ_1 與 κ_2 的值。

2.5 相機校正

相機校正，是為了求出相機的外部參數與內部參數。在[8]，R. Y. Tsai 提出了四項觀察來校正，而 Zhang 在 [9] 也提出類似的限制條件，應用於校正技術當中。由於校正相機需要空間的特徵點，因此通常是選用已知特徵的圖形，譬如下圖黑白格狀的圖形，利用軟體抓出該特徵點在影像上的位置，就可以開始校正。

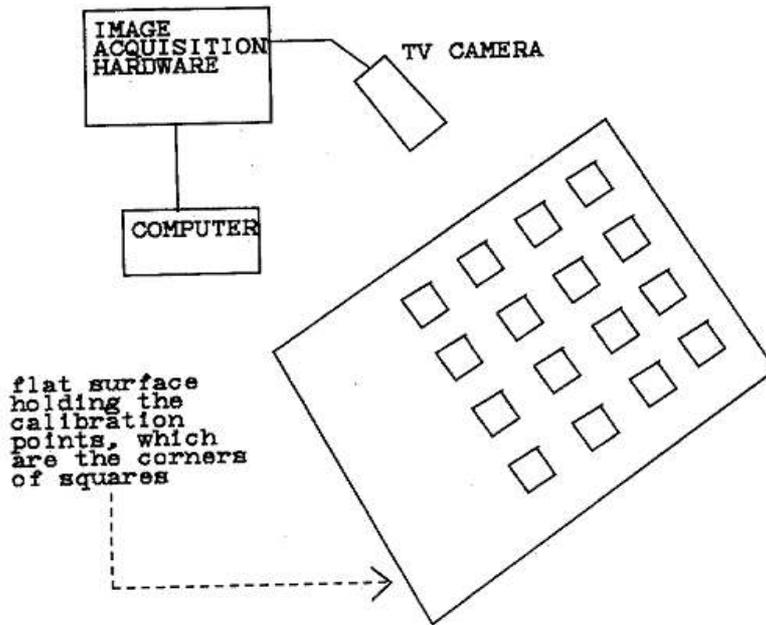


圖 7. 相機校正實驗圖解

R. Y. Tsai 的作法是使用將校正分成兩步，第一步是利用相機的文件查出硬體設計參數 N_{cx} 、 N_{fx} 、 d_x ，算出 d_x ，接著依照 [8] 的指導先估算出 (C_x, C_y) 與 s_x ，有了這些資訊即可由 (X_f, Y_f) 換算成 (X_d, Y_d) 。因為我們已知圖形的特徵，所以我們可以估算出相對的 κ_1 ，用 (X_d, Y_d) 來求 (X_u, Y_u) 。然後先上下相除，把焦距 f 約分掉，如此一來我們得到許多的特徵點 (x_w, y_w, z_w) 與其相對應的 (X_u, Y_u) ，這邊可以使用最佳化的理論來估算出外部參數 $r_1 \dots r_9$ 與 \mathbf{T} 矩陣，而 $r_1 \dots r_9$ 即是 \mathbf{R} 矩陣的元素。最後再回去求其他的參數，其相關的推導與證明可以參考[8]的敘述，Zhang 的作法是利用已知的特徵點 (x_w, y_w, z_w) 與影像上相對應的 (X_f, Y_f) ，用最佳化的方法，一次估算內部與外部參數，最後用最小平方法解決失真的問題[9]。

以上兩種校正技術，都已經成功的被軟體實現，現今的研究已經進展到如何用最少的特徵點來校正。對於以影像為基礎的話，原則上是不需要任何校正的，然而在一些文獻提到，事實上必須考量各自的系統狀況來決定，假設相機的失真過於嚴重，還是建議略為校正參數，才能增加系統穩定度[11]。除此之外，對於 eye-in-hand 的系統，因為相機會跟隨著機器手臂移動，校正外部參數是沒有意義

的，所以一般系統不需要精確的校正，有需要的話則校正內部參數即可。

目前相機校正已經被模組化。Jean-Yves 在 2008 年公佈了 MATLAB 的相機校正工具箱，適用於 MATLAB 5.0、6.0、7.0，在 Windows、Unix、Linux 等作業系統皆可使用。其原理是用各種角度的圖形，再結合 R. Y. Tsai 與 Zhang 的校正演算法，估算出相機的參數，以下我們介紹此工具箱的使用方法，如下圖：

1. 將相機所拍攝的靜態圖形建立成資料庫

一般我們存取的檔案格式為 TIF，圖形數量建議為 20 張至 25 張。

2. 輸入拍攝時參考圖形的幾何特徵

譬如我們使用黑白格狀的圖形，故輸入格狀的邊長與大小。

3. 讀取資料庫後手動選取圖形特徵點

如下圖所示，我們依序點選黑白格狀的特徵點，誤差容忍範圍在 5 pixel 內。

此步驟是為了避免軟體自動偵測特徵的錯誤，縮小之後校正的誤差。

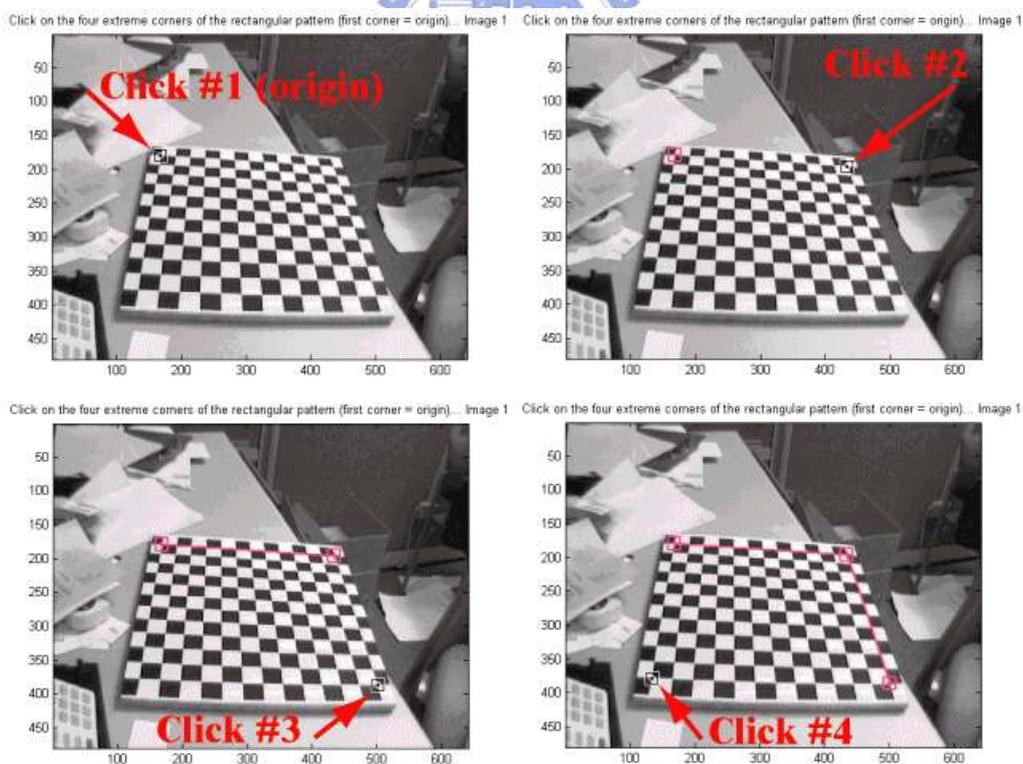


圖 8. 實際校正步驟圖

4.將所有圖形的特徵點選完成後自動算出所有參數值

在點選的過程中，若是失真過於嚴重，可以先手動猜測失真係數，方便點選特徵。此步驟完成後，軟體將顯示出校正後的所有參數與誤差範圍。

我們按照上面的步驟，用 20 張圖形來校正內部的參數，將校正後的結果用下表列出，並且用此參數來校正回原來的圖形，來比較校正前後的差別。下圖是校正比較圖，用肉眼就可以發現失真的狀況明顯的改善了，如箭頭所示。



圖 9. 校正前後比較圖

表格 1. 相機校正後實際的內部參數表

Parameter	Calibration results after optimization	Uncertainties
Focal Length	[661.67001 662.82858]	[± 1.17913 1.26567]
Principal Point	[306.09590 240.78987]	[± 2.38443 2.17481]
Skew Factor	[0.00000]	[± 0.00000]
Distortion Coefficient	[-0.26425]	[± 0.00934]

Note: The numerical errors are approximately three times the standard deviations.

了解相機模型之後，對於控制系統的設計也會有所幫助。以位置為主的控制架構，必須以繁瑣的相機校正為基礎，由外部參數的相機位置來反推物體位置，而校正誤差嚴重影響其結果。以影像為主的話，則不需要以上冗長的步驟。本文為了比較兩種架構的優劣，特地介紹相機模型與校正，以供其他論文參考。

三、特徵擷取

觀察的環境內容通過相機系統後，將類比的訊號轉換成數位影像，然而得到資料不等同於得到資訊，必須再經過資料處理等手續，整理過後，才能得到真正的資訊。雖然相機系統的演進，現今的相機每秒鐘至少能輸出數十張數位影像，甚至還有更高速的相機能捕捉更多的資料，但是與其增加資料量，讓機器人看得更細，不如改善資料處理的演算法，讓機器人看得更精。即使這樣在單位時間內還要考慮演算法的時間，透過特徵擷取也能更有效率的分析資料。

特徵擷取是屬於前處理(preprocessing)，能降低輸入資料的維度，克服維度災難(the curse of dimensionality)，同時能減少資料處理的數量，避免資料累積。對於之後的系統處理來說，特徵擷取提供了更有意義的輸入，也讓它變得更快且較少花費，更能補償前面擷取特徵時所耗費的資源，增進整個系統的表現。

擷取特徵有相當多種方法，很多文獻採用設定特徵的門檻(threshold)，大於此門檻的資料表示擁有某資訊，小於此門檻的資料則沒有某資訊。多維度的門檻我們稱之為 decision boundary，譬如線分割或是平面切割。其它方法像是 Principal Components Analysis(PCA)等等。值得一提的是，如何選取適當的特徵，是重要的問題。在選擇的時候，必須考慮演算法的假設與所需時間等等，才能事半功倍。

3.1 系統簡介

在電腦視覺的文獻中，影像特徵的定義是從影像測量出來的任何結構關係，譬如像是影像的輪廓或是物體的邊角。典型的影像特徵是物體特徵投影到影像上的對應[1]。在這種定義下，一個好的影像特徵點，必須是從任何角度投影下，都是不容易混淆的，像是[17]提到的 contrived pattern。而 Jang et al.定義影像特徵的參數，是適用於單張或多張的影像中，都能夠運算出來的實數[7]。藉由這些參數，我們即可應用到以影像為基礎架構的控制，包含在影像上的影像平面座標點[15-17]、直線、橢圓[17]等等結果。簡單來說，假設影像特徵維度為 k ，我們

可以定義影像特徵的參數向量為 $\vec{f} = [f_1 \ \dots \ f_k]^T$ ，其中 f_i 是實數，又可以寫成 $\vec{f} = [f_1 \ \dots \ f_k]^T \in \mathbf{F} \subseteq \mathbf{R}^k$ ，其中 \mathbf{F} 為影像特徵的參數空間[1]。此子系統的輸入是數位影像，輸出是參數向量 \vec{f} 。如果將機器手臂末端的移動空間，定義為 \mathbf{T} 的話，則整個系統的對應關係，就可以寫成 $\mathbf{T} \rightarrow \mathbf{F}$ ，而通常 $\mathbf{T} \subseteq \mathbf{R}^3$ 。

3.2 方法分析

影像特徵選取的方法，通常是依照後端的設計需求來決定參數。部分的文獻簡化了這個問題，採用簡單的物體與特定的工作環境，部份則是使用特定任務的線索，譬如 Allen[23]利用移動偵測來定位物體是否被抓住，或是利用鮮豔顏色來完成撿拾水果的系統，更多例子可以參考回顧文獻[7]。然而，擷取整張影像的特徵需要高速的運算，而並非整張影像的所有像素，都是我們有興趣的。如果我們能針對特徵出現機率較高的區域進行搜索，勢必能減少運算的時間。此概念稱之為 window-based 法[1]，優點是運算簡單、適用性廣、不需要其他硬體實現，缺點是必須知道初始的區域位置。對以影像為基礎的架構來說，我們能有技巧的控制機器手臂從起始位置開始移動，這部份的問題可以在實驗中克服。

window-based 分為兩種。在非限定的工作環境裡，可以從影像中清晰的劇烈對比來建立特徵，像是輪廓或邊角等等，可以判斷物體在影像中的位置，這種叫 feature-based 法，屬於影像處理的應用[1]。而如果利用區域內的資訊，像是色彩等等，來建立特徵，稱為 area-based 法，屬於電腦視覺的應用[1]。在這邊我們將引入影像追蹤的觀念，在時間花費與表現成果的折衷下，應用在適合我們的系統上。在其他領域中，有更多關於物體辨識的文獻，譬如類神經網路、建立資料庫等等。可惜這些系統太過於龐大，不適合應用在現階段的系統當中。

1. feature-based 法

此方法是影像的空間濾波器處理，以濾波器覆蓋的中心為基準點，鄰近的點為運算範圍，將它們的關係求出後，填回基準點。這裡我們介紹的是微分濾波器，

如下圖的 Roberts operator、Prewitt operator、Sobel operator，在變化比較劇烈的地方，通過濾波器之後，會有較大的斜率，這些點對應到的是物體的輪廓或邊角，可以作為特徵。若是覆蓋區域太大，或是要對整張圖形作濾波，非常耗費時間。在[1]提出了 window-based 法，強健性卻不高，必須再觀察其他的特徵，來增加辨識度[1]。對我們來說，為了提升穩定度與強健性，必須選取更有效率的特徵。

Roberts operator		Prewitt operator			Sobel operator								
1	0	-1	-1	-1	-1	0	1	-1	-2	-1	-1	0	1
0	-1	0	0	0	-1	0	1	0	0	0	-2	0	2
		1	1	1	-1	0	1	1	2	1	-1	0	1

圖 10. 影像空間濾波器圖解

2. area-based 法

此方法適用於區域內的圖形識別比對，首先將初始區域運算出其特徵，接著以此特徵為參考，以初始區域為中心，往鄰近的所有區域搜尋，直到找到發現與此特徵最相似為止，而下次的搜尋，則是以最相似的此點為起點開始。事實上，由於搜尋的區域越大，越耗費時間，這表示如果物體在兩張影像內移動太遠，在犧牲運算時間下，還是可以找到的，因此實作時，必須在這兩者取得平衡[1]。

以上兩種方法都需要對特徵作起始化。feature-based 的方法是先求出輪廓或邊角特徵後，接著尋找附近最相似的關係。此方法優點是十分簡單，缺點是不夠強健。area-based 的方法是先得到區域內的色彩分布，接著也是尋找附近最相似的關係。此方法的優點是比較通用，缺點是在光影變化下容易失常。在文獻回顧後，我們發現過去的研究都將重點放在 Jacobian 的估測，之前的影像只作簡單的處理。若是影像這邊處理品質不佳，之後的估測當然會不準確，故只要改善特徵的擷取系統，即可明顯的增加系統強健性，改善以往的瓶頸。在電腦視覺的研究中，提供了很多資源來解決這方面的問題，我們可以引進影像追蹤的技術，應用在以影像為基礎的架構上。讓整個的系統更加平衡，也更加的完善。

3.3 特徵追蹤

影像的物體追蹤技術已經發展多年，依照其方式可以分成非常多種類，像是骨架的分析、模型的建立等，由於影像追蹤擁有相當多種不同的技術，因此我們這邊不詳細的舉例，而是找出最適合我們的系統應用的即時技術。在 2003 年，Comaniciu et al.[41]利用 kernel 函數計算權重，將平均位移法成功的實現在物體追蹤上。平均位移法(mean shift)與上述的 area-based 法相似，其中的技巧就是將區域內的色彩用分布圖表示，而搜索的範圍也是用分布圖表示，兩者用相似函數比較之後，就可以觀察出物體移動的位移。我們將一開始作為參考的起始區域叫目標模型(target model)，搜索的結果叫目標候選(target candidate)，它們的分布圖用機率分布函數(probability density function)來表示，稱為 q 與 p 。不失一般性，我們令目標模型的中心點為 0，而目標候選的中心點為 \bar{y} ，我們把分布圖看作是成切割後 m -bin 的向量。 m 越小，計算越慢，而結果越準確，是可調整的參數。

$$\begin{cases} \bar{q} = \{q_u\}_{u=1, \dots, m} & \sum_{u=1}^m q_u = 1 \\ \bar{p}(\bar{y}) = \{p_u(\bar{y})\}_{u=1, \dots, m} & \sum_{u=1}^m p_u = 1 \end{cases} \quad (12)$$

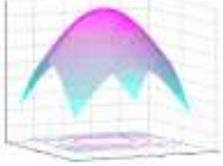
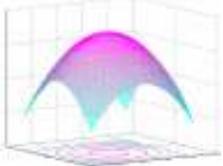
接下來計算 bin 的值。假設 $\{\bar{x}_i^*\}_{i=1, \dots, n}$ 是目標模型標準化過後 pixel 的位置，而標準化是為了取 norm。 $\{\bar{x}_i\}_{i=1, \dots, n_h}$ 是目標候選標準化的 pixel 位置。我們利用 kernel 函數 $k(x): [0, \infty) \rightarrow \mathbf{R}$ 來表示權重，而函數 $b(\bar{x}): \mathbf{R}^2 \rightarrow \{1, \dots, m\}$ 則表示在位置為 \bar{x} 的 pixel 所屬於 bin 的編號， δ 是 Kronecker delta function。經由這些定義，我們寫出：

$$\begin{cases} q_u = C \sum_{i=1}^n k(\|\bar{x}_i^*\|^2) \delta[b(\bar{x}_i^*) - u] & C = \frac{1}{\sum_{i=1}^n k(\|\bar{x}_i^*\|^2)} \\ p_u(\bar{y}) = C_h \sum_{i=1}^{n_h} k\left(\left\|\frac{\bar{y} - \bar{x}_i}{h}\right\|^2\right) \delta[b(\bar{x}_i) - u] & C_h = \frac{1}{\sum_{i=1}^{n_h} k\left(\left\|\frac{\bar{y} - \bar{x}_i}{h}\right\|^2\right)} \end{cases} \quad (13)$$

其中 C 與 C_h 是為了滿足總和為 1 的係數， h 為用 kernel 函數的頻寬。kernel 函數是 convex 的單調遞減函數，越遠離中心的權重越低，這與現實追蹤的觀念

相同，背景的權重必須小於中心的權重。這裡我們列出兩個常用的 kernel 函數：

表格 2. 平均位移法常用的 kernel 函數表

Name	Definition	Sketch with $d = 2$
Gaussian Function	$k_N(\bar{x}) = \begin{cases} \frac{1}{(2\pi)^{d/2}} \exp(-\frac{1}{2}\ \bar{x}\ ^2), & \text{if } \ \bar{x}\ < 1 \\ 0, & \text{otherwise.} \end{cases}$	
Epanechnikov Function	$k_E(\bar{x}) = \begin{cases} \frac{1}{2C_d} (d+2)(1-\ \bar{x}\ ^2), & \text{if } \ \bar{x}\ < 1 \\ 0, & \text{otherwise.} \end{cases}$	
d : dimension of space (in our 2D image case, $d = 2$) C_d : the volume of the unit d-Dimension sphere (in our 2D image case, $C_d = \pi$)		

我們用下圖的 Bhattacharyya 係數 ρ 來定義相似函數 $d(\bar{y})$ 。考慮兩個維度 m 的單位向量 $(\sqrt{q_1}, \dots, \sqrt{q_m})^T$ 與 $(\sqrt{p_1(\bar{y})}, \dots, \sqrt{p_m(\bar{y})})^T$ 。我們將單位化的元素，相乘後開根號相加，如果越相似，則 ρ 越大。注意 ρ 是 \bar{y} 的函數，可以寫成下列的關係：

$$\begin{cases} \rho(\bar{y}) \equiv \rho[p(\bar{y}), q] = \sum_{u=1}^m \sqrt{p_u(\bar{y})q_u} \\ d(\bar{y}) = \sqrt{1 - \rho[p(\bar{y}), q]} \end{cases} \quad (14)$$

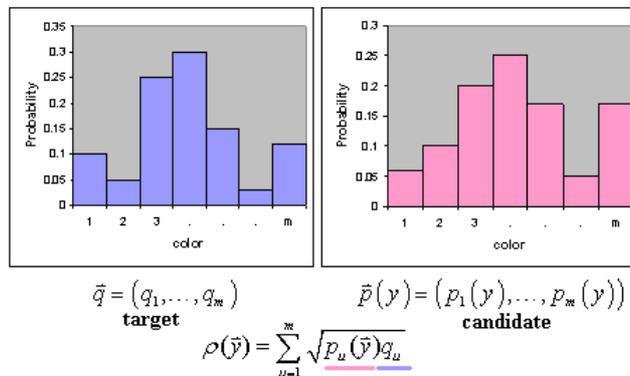


圖 11. 機率分布的函數比較圖

接著我們用泰勒展開式，對 \bar{y}_0 展開，其中 \bar{y}_0 是上一個 frame 的目標中心點。

$$\begin{aligned}\rho[p(\bar{y}), q] &= \sum_{u=1}^m \sqrt{p_u(\bar{y})q_u} \\ &= \sum_{u=1}^m \sqrt{p_u(\bar{y})q_u} \Big|_{\bar{y}=\bar{y}_0} + \sum_{u=1}^m \frac{1}{2\sqrt{p_u(\bar{y})q_u}} \Big|_{\bar{y}=\bar{y}_0} \cdot q_u \cdot [p_u(\bar{y}) - p_u(\bar{y}_0)] + \dots \quad (15) \\ &\approx \frac{1}{2} \sum_{u=1}^m \sqrt{p_u(\bar{y}_0)q_u} + \frac{1}{2} \sum_{u=1}^m p_u(\bar{y}) \sqrt{\frac{q_u}{p_u(\bar{y}_0)}}\end{aligned}$$

我們將後項省略，將之前的定義代入上式，可以得到下式：

$$\begin{aligned}\rho[p(\bar{y}), q] &\approx \frac{1}{2} \sum_{u=1}^m \sqrt{p_u(\bar{y}_0)q_u} + \frac{1}{2} \sum_{u=1}^m C_h \sum_{i=1}^{n_h} k\left(\left\|\frac{\bar{y} - \bar{x}_i}{h}\right\|^2\right) \delta[b(\bar{x}_i) - u] \sqrt{\frac{q_u}{p_u(\bar{y}_0)}} \\ &= \frac{1}{2} \sum_{u=1}^m \sqrt{p_u(\bar{y}_0)q_u} + \frac{C_h}{2} \sum_{i=1}^{n_h} w_i k\left(\left\|\frac{\bar{y} - \bar{x}_i}{h}\right\|^2\right) \quad (16) \\ \text{where } w_i &= \sum_{u=1}^m \sqrt{\frac{q_u}{p_u(\bar{y}_0)}} \delta[b(\bar{x}_i) - u]\end{aligned}$$

接著我們要對 ρ 求最大值，於是對 ρ 求微分為零的關係式如下：

$$\begin{aligned}\nabla \rho[p(\bar{y}), q] &= \frac{C_h}{2} \sum_{i=1}^{n_h} w_i k'\left(\left\|\frac{\bar{y} - \bar{x}_i}{h}\right\|^2\right) \cdot \frac{(\bar{y} - \bar{x}_i)}{h^2} \quad (17) \\ &= \frac{C_h}{2h^2} \sum_{i=1}^{n_h} (\bar{y} - \bar{x}_i) w_i k'\left(\left\|\frac{\bar{y} - \bar{x}_i}{h}\right\|^2\right) = 0\end{aligned}$$

令 $g(x) = -k'(x)$ ，我們可以整理出下列的方程式：

$$\begin{aligned}\nabla \rho[p(\bar{y}), q] &= \frac{C_h}{2h^2} \sum_{i=1}^{n_h} (\bar{x}_i - \bar{y}) w_i g\left(\left\|\frac{\bar{y} - \bar{x}_i}{h}\right\|^2\right) \\ &= \frac{C_h}{2h^2} \left[\sum_{i=1}^{n_h} \bar{x}_i w_i g\left(\left\|\frac{\bar{y} - \bar{x}_i}{h}\right\|^2\right) - \bar{y} \sum_{i=1}^{n_h} w_i g\left(\left\|\frac{\bar{y} - \bar{x}_i}{h}\right\|^2\right) \right] = 0 \quad (18) \\ \Rightarrow \bar{y} &= \frac{\sum_{i=1}^{n_h} \bar{x}_i w_i g\left(\left\|\frac{\bar{y} - \bar{x}_i}{h}\right\|^2\right)}{\sum_{i=1}^{n_h} w_i g\left(\left\|\frac{\bar{y} - \bar{x}_i}{h}\right\|^2\right)}\end{aligned}$$

此方法稱為平均位移法， \bar{y} 即是相似區域的位移量。我們將方程式寫成遞迴演算法，若是位移小於我們設定的遞迴參數 ε ，則遞迴結束。下圖是流程圖：

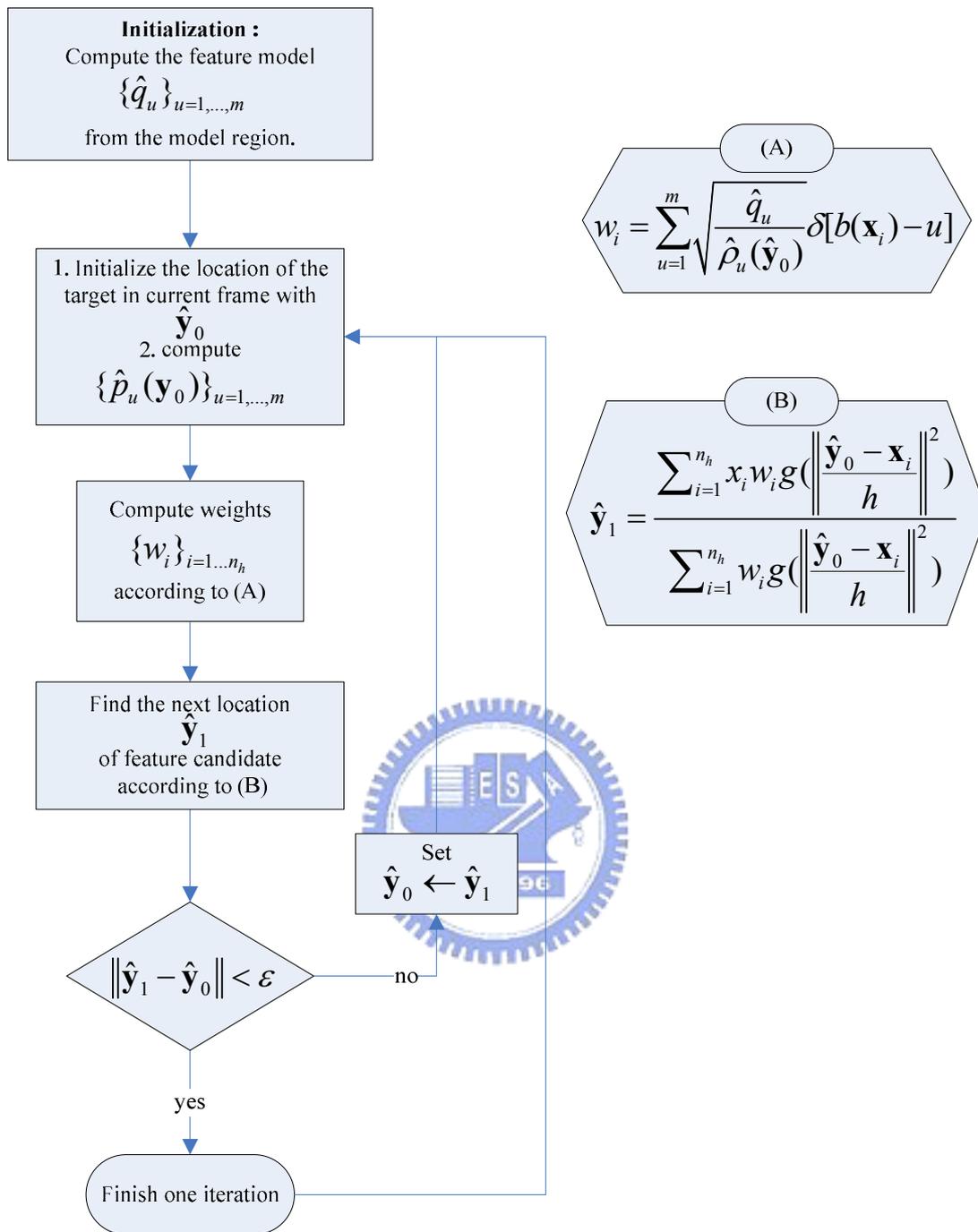


圖 12. 平均位移演算法流程圖

上述的演算法，被稱為傳統的平均位移法，此方法是利用色彩分布圖(color histogram)當作目標，而找出與此目標最相似的位移。比起原始的特徵擷取法，此方法優點是雖然增加運算時間，但強健度卻改善非常多。在硬體上，對於目前個人電腦來說，已經能夠即時的反應。在軟體上，如果遞迴次數太多，表示可能找不到目標，我們可以技巧的先結束這次的遞迴，下次的遞迴再繼續即可。而此

方法缺點是為了減少運算量，只採用目標的色彩資訊，捨棄了目標的空間資訊。S. Birchfield 與 S. Rangarajan[42]提出了空間分布圖的概念(spatial histogram, or spatiogram)，也就是多計算出區域中的 mean vector 與 covariance matrix，當作是目標。除此之外，C. Yang et al.[43]改寫了方程式，定義了新的相似函數與縮放的參數，來解決追蹤區域縮放(scaling)的問題。而追蹤區域轉向(orientation)的問題，可以用下圖的 PCA 技術來克服[44]，讓平均位移法能適用於一般的狀況。

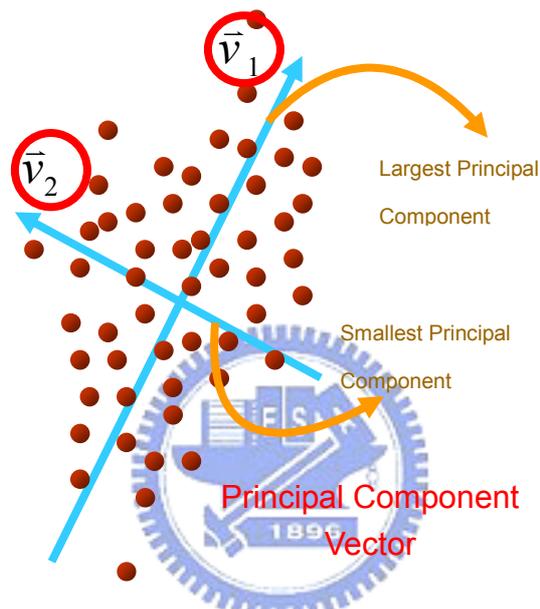


圖 13. 主要成分分析圖

3.4 實驗模擬

以上我們引進了物體追蹤的觀念，將系統數位影像的輸入，先轉換成追蹤的目標模型(target model)，再定義目標候選(target candidate)，算出與目標最相似的位移。此位移可以視為追蹤的目標中心，即可以當作是系統的輸出。以往的系統僅用簡單的背景與單純的物體，依照狀況設定門檻，將輸入處理成二元的影像，再找尋影像的中心後輸出，把節省下來的資源分配到後端。這樣的優點是可以用較繁瑣的估測法增加準確度，缺點是工作環境遭受限制。一旦影像處理出了錯，所付出的代價極大。故我們將資源重新分配，只要前端的影像計算準確，後端的系統就可以改用較低運算量的估測，也可以在未知的環境，還能有不錯的表現。

電腦視覺在物體追蹤的研究有非常多文獻可以引用，只要資源許可，[42-44]的方法都可以應用，端看工作環境與工作任務的複雜程度。平均位移法的好處是耗費的時間較少，資源佔用不多，而表現卻是比較好的演算法，不僅在個人電腦上，用解析度較差的視訊都能夠實現，連在嵌入式系統，用 CCD(charge-coupled device)的影像輸入也能即時處理，以下是我們的模擬與實驗的結果。

1.MATLAB 模擬

本模擬是在個人電腦完成，CPU 規格為 AMD Athlon(tm) 64 Processor 3000+ 1.81 GHz，RAM 為 1.87GB，OS 是 Microsoft Windows XP Professional Version 2002 SP3，軟體是 MATLAB version 7.0.4.365(R14) SP2。我們先用 DV 預錄一段格式為 AVI 的影片，再用 MATLAB 讀取後，代入演算法。最後將運算後的結果存成 JPG 檔圖形，來觀察整體的表現。下圖是以杯子為追蹤目標的結果，我們擷取 frame 編號為 1、50、100、150、200、250、300、350、400 的圖形。追蹤的結果用黃色框框表示，可以發現平均位移法在較複雜的背景也能適用[44]。

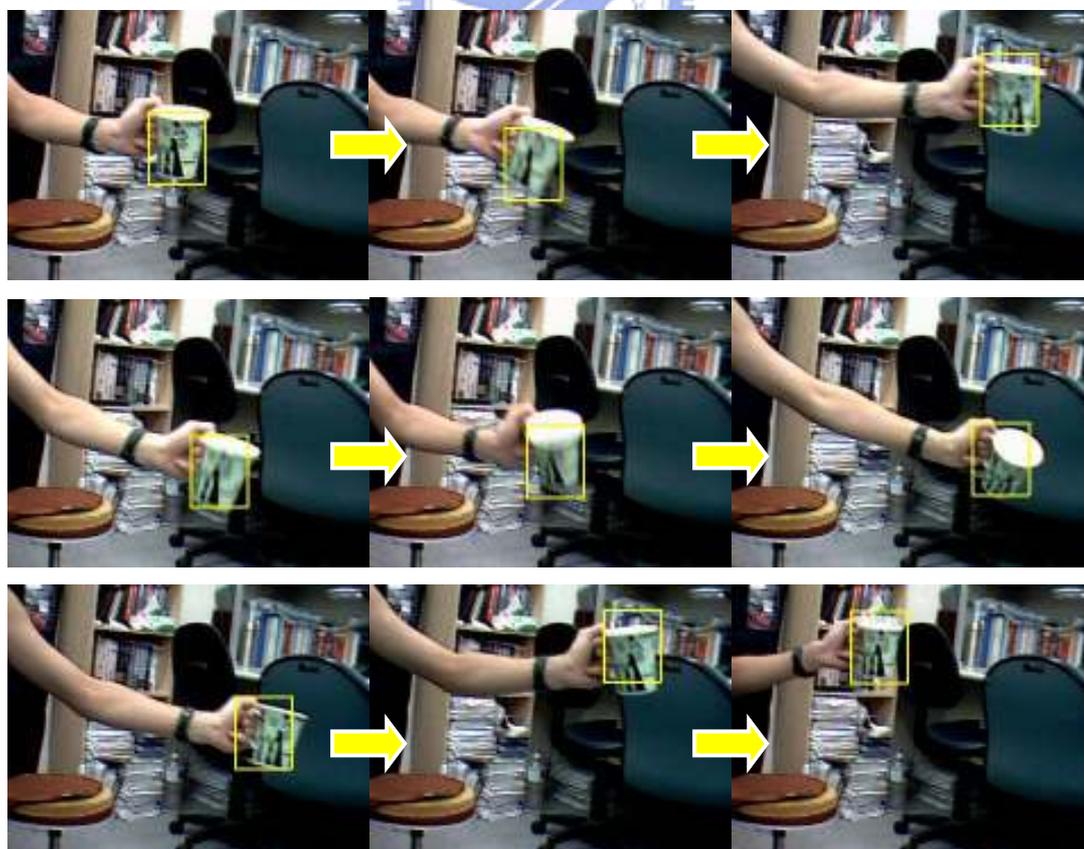


圖 14. 平均演算法的追蹤結果示意圖

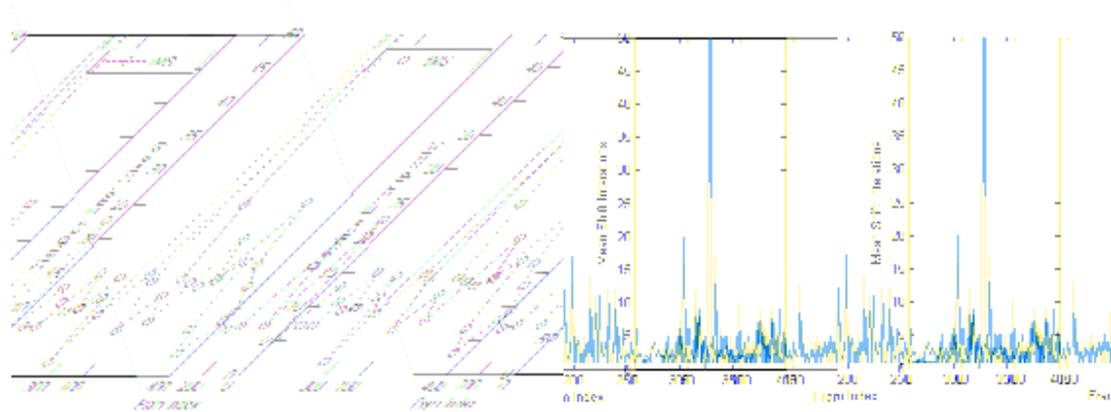


圖 15. 平均位移法的誤差圖與遞迴次數圖

上左圖是平均位移法的誤差圖，橫軸是 frame 的編號，縱軸是追蹤的中心與實際中心的誤差距離。我們發現在編號為 71 與 310 的地方誤差比較大，原因是因為杯子的翻轉，使得色彩分布圖差異較大，所以略有誤差，而之後的追蹤結果又馬上回到可接受的範圍。上右圖是遞迴次數圖，橫軸是 frame 的編號，縱軸是遞迴的次數，可以發現在編號為 71 的地方遞迴數較多，我們在程式中設定遞迴次數超過五十次，就跳出這次的遞迴，以便在下次的遞迴再繼續尋找。

2. 系統實驗

為了測試即時的追蹤狀況，我們使用德州儀器(TI, Texas Instrument)所發行的 DM6446 達文西平台，利用其中的 RISC CPU ARM926 來實現。此平台作業系統是 MONTAVISTA Linux V4.0，具有影像處理的功能，可以將影像的輸入暫存，方便我們程式讀取。開發時我們使用 arm_v5t_le-gcc 作為轉換編譯器，將程式碼先在 PC 編譯，確定無誤後，再轉譯存入到平台。在 PC 的 Linux 上，我們安裝 MONTAVISTA Linux V4.0 System Tools，內含相關系統開發套件，包括了編譯器與平台的 Linux kernel 與 file system 的重製工具。簡單來說，我們整個開發過程是在 PC 完成，只是將結果放入平台上。使用嵌入式系統的優點，是能在最少的資源下測試演算法的結果，而且周邊硬體齊全。我們用下圖攝影機輸入影像後，在平台上代入演算法，最後輸出到螢幕。下下圖是以杯子為目標的結果，以綠色框框表示。取樣時間為一秒[45]，可以發現平均演算法在即時的互動也能支援。



圖 16. SONY EVI-D70 PTZ 攝影機與 EVERFOCUS 彩色 TFT LCD 螢幕

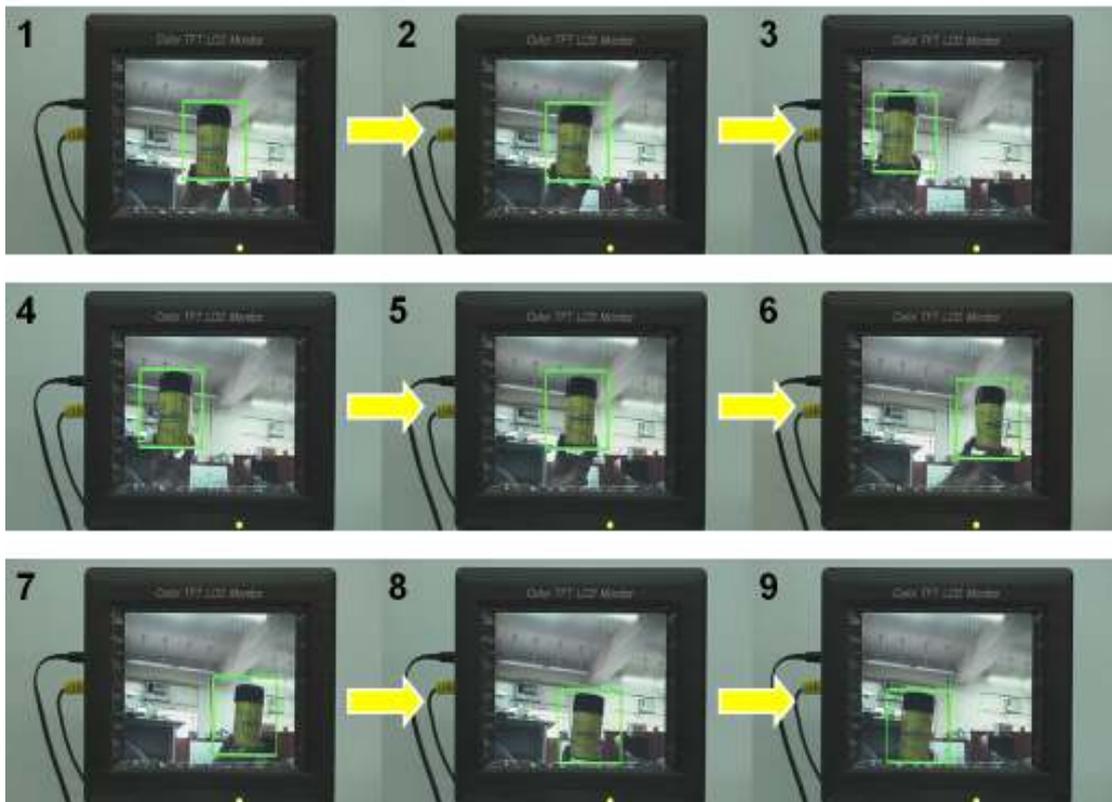


圖 17. 平均位移法的即時追蹤結果圖

在這個階段中，我們引用了平均位移法，來提升特徵擷取的效能，與過去的子系統相比，我們更能準確的得到物體中心的位置，而且在複雜的環境與移動的狀況下，也能有穩健的表現。其他技術還有立體視覺可以應用，不過該方法需要雙相機，而且受限於雙眼的距離，並不適合。最後，本文建議設計時，必須考慮硬體的負荷。若是在此階段耗費太多資源，在下個子系統中，將嚴格受到考驗。

四、視覺回授

數位影像通過特徵擷取系統之後，將未經處理的資料(raw data)轉換成所需的資訊。在我們的系統內，此資訊被定義為特徵，是為了得知物體在影像中的位置。而在其他研究領域中，特徵被廣泛定義為輪廓、邊角、色彩等容易被識別的特殊內容。以位置為基礎架構的缺點，除了之前提到的對於校正誤差比較敏感，還有計算的問題。使用影像特徵來定位的優點是不需要校正，不需要算出空間中相機與物體的相對位置，缺點是不論選用伺服控制，或是觀察運動架構，系統設計都比較複雜。下面我們先從文獻回顧開始，分析與探討兩種控制系統的架構。

在 1980 年，Sanderson 與 Weiss[6]介紹了系統架構分類法，其中分辨方法是依照手臂得到的指令種類決定。更仔細的說，影像特徵與我們想要的特徵彼此相減，可以算出特徵誤差，此誤差經過控制法則來驅動機器手臂。如果這個法則是直接用特徵誤差來得到角度輸出，分配給各個關節的話，我們稱之為伺服控制架構。如果這個法則是用特徵誤差判斷如何移動，而讓手臂自行計算各個關節，這我們稱為觀察運動架構。在定義上[1]，視覺伺服(visual servo)一般已經被定義為用任何種類的視覺回授(visual feedback)來控制機器手臂，而定義伺服控制架構為 direct visual servo structure，觀察運動架構為 dynamic look-and-move structure。

通常，選擇伺服控制架構的原因是考慮到機器手臂的輸入比較簡單，而且此方法可以運用到相同自由度下的任何機器手臂，也就是不限定模型、比較一般的方法，而運算的速度由此演算法來決定。然而，選擇觀察運動系統的原因有三種，一、此系統有內部回授，二、目前手臂都能接受此種移動指令。三、避免 kinematic singularities 等等問題，因為能接受這種指令的手臂，通常會有最佳化過的 inverse kinematics，將末端位置換算回關節角度，所以運算的速度是由該演算法決定。對於現今的發展來說，不論是角度的輸入或是端點的輸入都很常見。總而言之，輸入角度移動量的方法比較直接、通用，而輸入端點移動量的方法必須考慮手臂是否接受，其它文獻較少討論兩種架構的優劣，本文建議依照硬體規格選擇即可。

4.1 系統簡介

視覺回授的子系統分成兩部分，前面是軟體演算法的計算，後面是機器手臂的移動。基本上硬體的作業系統、通訊協定、輸入介面為不可更動的固定規格，我們只能依照不同的硬體來設計不同的演算法，而分成伺服控制與觀察運動兩種架構。兩者輸入都是特徵誤差，而前者的輸出是各關節角度，後者的輸出是位移。以理論的角度來看，由於兩者的輸出是類似的，只有變數的維度不同，因此我們在分析方法時，可以不限定輸出變數的維度與內容，完成不失一般性的推導。

在前面的文獻回顧探討中，提到了估測 Jacobian 的方法，Hosoda 與 Asada[18] 是用 exponentially weighted recursive least-squares (RLS) 來更新方程式，Jagersand [19] 是用 nonlinear least-squares optimization method 與 Broyden estimation，而兩者是觀察靜態目標，Piepmeier et al.[25][26] 則是觀察動態的目標，完成了靜態相機視覺回授，推導了收斂的性質，而不考慮精確的相機校正模型與機器手臂模型。以上是靜態相機的討論，而動態相機方面，Asada 與 Hosoda[31] 延伸了他們的方法，以靜態參考點來觀察動態目標，而 Piepmeier[32] 在不根據模型的狀況下，解決了靜態參考點的問題。2007 年，Farahmand et al.[36] 試圖解決全域性的問題，在方程式中不限定變數的維度，可能是各關節的角度或是位移。以下我們與[36] 相同，綜合上述兩種架構，配合 eye-in-hand 與 eye-to-hand，以求推導的一般性。

4.2 方法分析

關於視覺回授系統，我們參考[1]的定義，是影像特徵誤差 e 的函數。不論是 eye-in-hand 與 eye-to-hand，機器手臂的移動造成了影像特徵的移動，此移動結果與我們想要的影像特徵 f_d 相減，得到特徵誤差 e ，我們的目的是讓誤差 e 為零。我們可以想像成嬰兒的學習，當他們看到有興趣的物體，會試圖抓取。由於他們不太能控制自己的動作，將會發現手的位置與物體的位置不同，因此觀察到這個現象後，修正幾次即可抓到物體。我們把眼睛看到的物體，稱為想要的影像位置

特徵 f_d ，與眼睛看到手的位置相減，得到特徵誤差 e ，再去修正這個誤差。

為了完成這個目的，我們必須將影像的變化，對應到機器手臂的變化，這個相對應的概念，稱之為 Jacobian 矩陣。很多文獻探討著如何決定 Jacobian，各自有各自的名稱。在這邊我們統一使用 Jagersand[35]的變數名稱，與 Farahmand et al.[36]的變數定義，我們定義關節角度(joint angles)或位移(displacements)為變數 $\bar{x} \in \mathbf{R}^n$ ，而影像特徵(feature points)為 $\bar{y} \in \mathbf{R}^m$ 。當變數 \bar{x} 改變的時候，機器手臂也改變其位置，此時由影像回授，擷取改變的特徵 \bar{y} ，此函數關係可以寫成：

$$\bar{y} = f(\bar{x}) \quad (19)$$

因為函數 f 與手臂的運動學模型(kinematic model)與相機模型(camera model)有關，所以是非線性函數，Jagersand 稱之為 visual-motor function[35]。如果起始狀態為 \bar{x}_0 與 \bar{y}_0 ，我們想得到的最終狀態是 \bar{y}^* 。簡單來說，我們必須想辦法得到 $\Delta\bar{x}_k$ ，而使得 $f(\bar{x}_0 + \sum \Delta\bar{x}_k) = \bar{y}^*$ 。下圖是簡單的示意圖，其中 f^{-1} 是 f 的反函數。

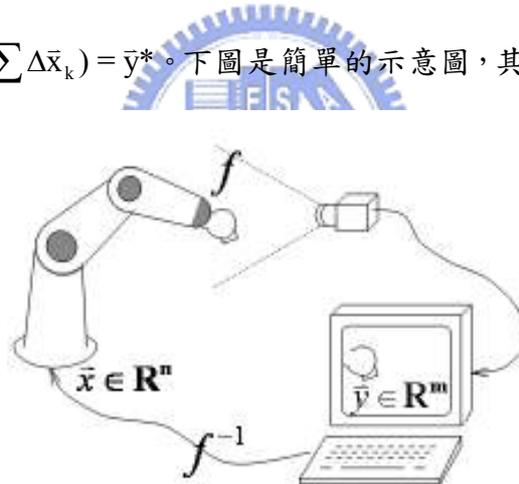


圖 18. 視覺回授示意圖

傳統上的藉由校正，我們可以得到兩種函數，第一是手臂運動學 g ，第二是相機校正 h ，故我們可以把 f 看成 g 與 h 的合成函數，可以寫成：

$$\bar{y} = f(\bar{x}) = h(g(\bar{x})) \quad (20)$$

由於此為非線性函數，我們用分段線性(piecewise linear)的方法來分析，在某個區間 k 時假設可微分，因此我們可以用一階微分的模型，將 $\bar{y} = f(\bar{x})$ 寫成：

$$f(\bar{x}) \approx f(\bar{x}_k) + \mathbf{J}(\bar{x}_k)(\bar{x} - \bar{x}_k) \quad (21)$$

此模型在 \bar{x}_k 附近是有效的，而 \mathbf{J} 矩陣是 Jacobian 矩陣，Jagersand 稱此矩陣為 visual-motor Jacobian[35]，最初稱為 feature sensitivity matrix[14]，如下式定義：

$$\mathbf{J} = \frac{\partial(y_1, \dots, y_m)}{\partial(x_1, \dots, x_n)} = \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \frac{\partial y_1}{\partial x_2} & \dots & \frac{\partial y_1}{\partial x_n} \\ \frac{\partial y_2}{\partial x_1} & \frac{\partial y_2}{\partial x_2} & & \frac{\partial y_2}{\partial x_n} \\ \vdots & & \ddots & \\ \frac{\partial y_m}{\partial x_1} & \frac{\partial y_m}{\partial x_2} & & \frac{\partial y_m}{\partial x_n} \end{bmatrix} \quad (22)$$

簡單來說， f 是 $\mathbf{R}^n \rightarrow \mathbf{R}^m$ 的函數，此函數的偏微分是 $m \times n$ 的 Jacobian 矩陣，意義是將手臂移動變化，對應到影像的移動變化。我們把方程式改寫成其他形式：

$$\bar{y}_{k+1} = \mathbf{J}\Delta\bar{x} + \bar{y}_k \Rightarrow \Delta\bar{y} = \mathbf{J}\Delta\bar{x} \quad (23)$$

我們知道 Jacobian 矩陣 \mathbf{J} 之後，必須想辦法得到 \mathbf{J}^{-1} ，而 \mathbf{J}^{-1} 的意義是將影像的移動變化，對應到手臂移動變化。有了 \mathbf{J}^{-1} ，才能利用視覺回授，來控制手臂。

$$\Delta\bar{y} = \mathbf{J}\Delta\bar{x} \Rightarrow \Delta\bar{x} = \mathbf{J}^{-1}\Delta\bar{y} \quad (24)$$

為了得知 \mathbf{J}^{-1} ，我們考慮以下兩種情況，第一種是當 \mathbf{J} 為 rank deficient 時，因為 \mathbf{J}^{-1} 不存在，所以用奇異值分解(SVD, singular value decomposition)來分解 \mathbf{J} ：

$$\mathbf{J} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \begin{cases} \mathbf{U} : m \times m \\ \mathbf{\Sigma} : m \times n \\ \mathbf{V} : n \times n \end{cases} \quad (25)$$

$\mathbf{\Sigma}$ 是 $m \times n$ 的對角矩陣(diagonal matrix)，假設 $\mathbf{J}^T\mathbf{J}$ 的特徵值為 $\lambda_1 \geq \dots \geq \lambda_r > 0$ 而且 $\lambda_{r+1} = \dots = \lambda_n = 0$ ，其中 r 是 $\mathbf{J}^T\mathbf{J}$ 的 rank 值，則 $\mathbf{\Sigma}$ 矩陣的定義如下列方程式：

$$\mathbf{\Sigma} = \text{diag}(\sqrt{\lambda_1}, \dots, \sqrt{\lambda_n}) = \text{diag}(\sigma_1, \dots, \sigma_n) \in \mathbf{R}^{m \times n} \quad (26)$$

由於 $\lambda_{r+1} = \dots = \lambda_n = 0$ ，因此我們可以改寫上式，得到 $\mathbf{\Sigma}$ 矩陣的定義如下式：

$$\mathbf{\Sigma} = \text{diag}(\sigma_1, \dots, \sigma_r) \in \mathbf{R}^{n \times n} \quad (27)$$

而 $\mathbf{V} = \{\bar{v}_1, \dots, \bar{v}_n\}$ 矩陣則是 $\mathbf{n} \times \mathbf{n}$ 的正交矩陣(orthogonal matrix)， \bar{v}_i 的定義為：

$$\mathbf{J}^T \mathbf{J} \bar{v}_i = \lambda_i \bar{v}_i \quad (28)$$

最後， $\mathbf{U} = \{\bar{u}_1, \dots, \bar{u}_m\}$ 是 $\mathbf{m} \times \mathbf{m}$ 的正交矩陣(orthogonal matrix)， \bar{u}_i 的定義為：

$$\mathbf{J} \mathbf{J}^T \bar{u}_i = \lambda_i \bar{u}_i \quad (29)$$

由於 $\lambda_{r+1} = \dots = \lambda_n = 0$ ，因此我們可以改寫 \mathbf{U} 矩陣的定義，得到簡化的下式：

$$\mathbf{J} = \mathbf{U} \mathbf{\Sigma} \mathbf{V}^T \begin{cases} \mathbf{U} : \mathbf{m} \times \mathbf{n} \\ \mathbf{\Sigma} : \mathbf{n} \times \mathbf{n} \\ \mathbf{V} : \mathbf{n} \times \mathbf{n} \end{cases} \quad (30)$$

當 \mathbf{J} 為 rank deficient 時，我們可以用 \mathbf{J} 的 pseudo-inverse， \mathbf{J}^+ 來計算 \mathbf{J}^{-1} ：

$$\mathbf{J}^+ = \mathbf{V} \mathbf{\Sigma}^+ \mathbf{U}^T \quad (31)$$

如此即可得到其對應關係。更多細節可以參考[1]與 Farahmand et al.的[36]。

$$\Delta \bar{x} = \mathbf{J}^+ \Delta \bar{y} \quad (32)$$

第二種是當 \mathbf{J} 為 full rank 時，可分成： $\mathbf{m} = \mathbf{n}$ 、 $\mathbf{m} > \mathbf{n}$ 、 $\mathbf{m} < \mathbf{n}$ 三種來討論：

1. $\mathbf{m} = \mathbf{n}$

Feddema[39]使用的方法，當 $\mathbf{m} = \mathbf{n}$ 且 \mathbf{J} 為可逆時，則 \mathbf{J}^{-1} 存在且 $\Delta \bar{x} = \mathbf{J}^{-1} \Delta \bar{y}$ 。

2. $\mathbf{m} > \mathbf{n}$

Jang[10]使用的方法，此為 over-determined 的狀況，當 \mathbf{J} 為滿秩(full rank)，也就是 $\text{rank}(\mathbf{J}^{-1}) = n$ 的話，可用 \mathbf{J} 的 pseudo-inverse， \mathbf{J}^+ 來計算，其關係如下：

$$\mathbf{J}^+ = (\mathbf{J}^T \mathbf{J})^{-1} \mathbf{J}^T \Rightarrow \Delta \bar{x} = \mathbf{J}^+ \Delta \bar{y} \quad (33)$$

3. $\mathbf{m} < \mathbf{n}$

此為 under-determined 的狀況，我們可用 \mathbf{J} 的 pseudo-inverse， \mathbf{J}^+ 來計算：

$$\mathbf{J}^+ = \mathbf{J}^T (\mathbf{J} \mathbf{J}^T)^{-1} \quad (34)$$

此狀況較少發生，原因是 f 為 $\mathbf{R}^n \rightarrow \mathbf{R}^m$ 的函數， $\mathbf{m} < \mathbf{n}$ 的意義是特徵空間的維度，比機器手臂空間的維度還要少，這樣方程式將會變成 under-determined 的狀況，所得到的解是近似解，誤差很大。一般來說我們會避免這種狀況發生。

以上我們討論了各種估測 Jacobian 的方法。由上面的推導我們得知，怎樣的手臂變化 $\Delta\bar{x}$ ，會對應到怎樣的影像變化 $\Delta\bar{y}$ ，而在 \mathbf{J}^{-1} 的幫忙下，我們可以知道怎樣的影像變化 $\Delta\bar{y}$ ，會得到怎樣的手臂變化 $\Delta\bar{x}$ 。在計算出這些關係之後，我們接著要送訊號 \bar{u} 給機器手臂，在[1]的教導下，目前大部分文獻的訊號設計如下：

$$\bar{u} = \mathbf{KJ}^{-1}(\bar{f}_d - \bar{f}) \quad (35)$$

其中 \mathbf{K} 是增益矩陣，在[36]也有類似的應用。更多內容可以參考[14][15]。

4.3 影像回授

以上我們推導了通用狀況 Jacobian 矩陣的方程式，方程式中不限制輸入維度 \mathbf{R}^n 與輸出維度 \mathbf{R}^m 的情況下，我們所得到的 f 與 f^{-1} ，可以適用於伺服控制架構 (direct visual servo structure) 與觀察運動架構 (dynamic look-and-move structure)，而這兩種架構相異的地方有二，第一、前者輸出是各關節角度，一般的機器手臂有六個自由度可以定位，屬於 \mathbf{R}^6 空間。後者的輸出是相對的位移，通常是三度空間，屬於 \mathbf{R}^3 空間。第二、由於前者是將六軸關節，先對應到三度空間，接著由三度空間對應到影像，因此我們更嚴謹的定義，將這裡的 Jacobian 矩陣稱為合成 Jacobian (composite Jacobian) 矩陣，是由運動 Jacobian (kinematic Jacobian) 與影像 Jacobian (image Jacobian) 組成，而後者是直接從三度空間對應到影像，我們稱之為影像 Jacobian (image Jacobian)，故上一節所推導的通用 Jacobian 矩陣，對伺服控制架構是合成 Jacobian 的推導，對觀察運動架構是影像 Jacobian 的推導。

本文採用的是後者，也就是觀察運動架構，來估測影像 Jacobian 矩陣。主要有兩個原因，第一、在輸入影像特徵的誤差後，前者必須花費大量計算時間算出各軸角度，而後者所計算的矩陣維度與運算量遠小於前者。第二、前者算出各軸關節角度後，由於還要多花時間找出最小移動量的解，因此這方面的研究主要是用模擬完成，實作比較難達到即時的反應。相較之下，後者算出是三度空間座標值，因為目前的機器手臂皆有最佳化的演算法，可以算出最少的關節移動量，來

抵達所需要的座標點，所以實作上比較容易。在前面的文獻回顧也可以發現，大部分的文獻是以估測影像 Jacobian 矩陣為主，原因也是大致如上列所述。

以下我們介紹本文決定影像 Jacobian 矩陣的方法，在這邊我們採用的是即時估測法。此方法靈感於來自[1]的例子，而採用 Feddema[39]的 $\mathbf{m} = \mathbf{n}$ 與 \mathbf{J} 為 full rank 的假設，我們將其方程式加以簡化後[30]，完成即時估測，最後加以實現。在這邊我們考慮單眼相機(monocular camera)視覺回授的狀況。因為未校正相機並不知道其確切位置的座標與角度，所以為了符合常理，假設在執行任務時，物體與機器手臂不會超出影像的感測範圍。接著，我們定義手臂的輸入指令值，都在指令的定義域(command domain)內，此定義的原因是並非整個三度空間機器手臂皆可以抵達，故此定義域可以限制指令不會超出手臂伸展空間與相機範圍。寫成數學式的話，我們將輸入指令稱為 $\bar{\mathbf{p}}(t)$ ，投影到影像的平面為 $\bar{\mathbf{s}}(t)$ ，可以寫成：

$$\left\{ \begin{array}{l} \bar{\mathbf{p}}(t) = \begin{bmatrix} x_b(t) \\ y_b(t) \end{bmatrix} \\ \bar{\mathbf{s}}(t) = \begin{bmatrix} u(t) \\ v(t) \end{bmatrix} \end{array} \right. \quad (36)$$

兩者皆是時間的變數，我們對其微分，可以得到影像 Jacobian 矩陣的關係式：

$$\dot{\bar{\mathbf{s}}}(t) = \mathbf{J}(t) \dot{\bar{\mathbf{p}}}(t) \text{ where } \mathbf{J}(t) = \begin{bmatrix} \frac{\partial u}{\partial x_b} & \frac{\partial u}{\partial y_b} \\ \frac{\partial v}{\partial x_b} & \frac{\partial v}{\partial y_b} \end{bmatrix} \quad (37)$$

我們將指令 $\bar{\mathbf{p}}(t)$ 輸入到手臂，內容為世界座標系的 $[x_b \ y_b \ Z]^T$ ，其中 Z 為任意值。由於採用觀察運動架構，因此手臂有內建最佳化的反向運動學(inverse kinematics)演算法，將三度空間座標值換算出最少的關節移動量後，驅動末端到此座標，再投影到相機座標系的 $[x_c \ y_c \ z_c]^T$ ，兩者的座標值轉換的關係為：

$$\begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = \mathbf{R} \begin{bmatrix} x_b \\ y_b \\ Z \end{bmatrix} + \mathbf{T} \quad (38)$$

因為採用固定相機的架構，所以 \mathbf{R} 與 \mathbf{T} 為定值，其中 \mathbf{R} 是旋轉矩陣， \mathbf{T} 是平移矩陣，在未校正的情況下，是未知的相機外部參數。矩陣的元素為：

$$\mathbf{R} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \text{ and } \mathbf{T} = \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} \quad (39)$$

經過座標轉換到相機座標系之後，其座標值為 $[x_c \ y_c \ z_c]^T$ 。由相機模型的投影關係，我們可以知道如果相機的焦距為 f ，其座標值與 $[u \ v]^T$ 的關係為：

$$\begin{cases} u = f \frac{x_c}{z_c} \\ v = f \frac{y_c}{z_c} \end{cases} \quad (40)$$

我們對上式微分後，整理成矩陣的形式，再將上面的關係式代入，可以得到：

$$\dot{\hat{s}} = \begin{bmatrix} \dot{u} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} \frac{f}{z_c} & 0 & -\frac{u}{z_c} \\ 0 & \frac{f}{z_c} & -\frac{v}{z_c} \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = \begin{bmatrix} \frac{f}{z_c} & 0 & -\frac{u}{z_c} \\ 0 & \frac{f}{z_c} & -\frac{v}{z_c} \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} \\ r_{21} & r_{22} \\ r_{31} & r_{32} \end{bmatrix} \dot{\hat{p}} \quad (41)$$

對照其關係，我們發現影像 Jacobian 矩陣即是上式矩陣相乘的結果，如下：

$$\mathbf{J} = \begin{bmatrix} \frac{f}{z_c} & 0 & -\frac{u}{z_c} \\ 0 & \frac{f}{z_c} & -\frac{v}{z_c} \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} \\ r_{21} & r_{22} \\ r_{31} & r_{32} \end{bmatrix} = \frac{1}{z_c} \begin{bmatrix} fr_{11} - ur_{31} & fr_{12} - ur_{32} \\ fr_{21} - vr_{31} & fr_{22} - vr_{32} \end{bmatrix} \quad (42)$$

在此方程式中，我們得知影像 Jacobian 矩陣是由 u 、 v 、 z_c 決定，然而相機的外部參數 \mathbf{R} 與 \mathbf{T} 、內部參數 f 不隨時間改變，必須從校正來決定，故我們採用遞迴的估測法，即時估測影像 Jacobian 矩陣，並且加以更新。此遞迴方程式為：

$$\mathbf{J}(k) = \mathbf{J}(k-1) + \Delta\mathbf{J}(k) \quad (43)$$

其中在時間 k 的 Jacobian，是由時間 $k-1$ 的 Jacobian 與更新項 $\Delta\mathbf{J}(k)$ 來決定：

$$\Delta\mathbf{J}(k) = \frac{1}{z_c(k)} \begin{bmatrix} fr_{11} - u(k)r_{31} & fr_{12} - u(k)r_{32} \\ fr_{21} - v(k)r_{31} & fr_{22} - v(k)r_{32} \end{bmatrix} - \frac{1}{z_c(k-1)} \begin{bmatrix} fr_{11} - u(k-1)r_{31} & fr_{12} - u(k-1)r_{32} \\ fr_{21} - v(k-1)r_{31} & fr_{22} - v(k-1)r_{32} \end{bmatrix} \quad (44)$$

我們假設此相機觀察全域的資訊，因為其觀察對象較遠， $z_c(k-1)$ 與 $z_c(k)$ 的相差甚小，所以我們可以簡化 $\Delta\mathbf{J}(k)$ 的運算，並且可以為了除去非線性的分母項：

$$\Delta\mathbf{J}(k) \approx \frac{1}{z_c(k)} \begin{bmatrix} u(k-1) - u(k) \\ v(k-1) - v(k) \end{bmatrix} \begin{bmatrix} r_{31} & r_{32} \end{bmatrix} = \Delta\bar{s}(k) \bar{w}(k)^T \quad (45)$$

我們將與影像有關的向量，整理成 $\Delta\bar{s}(k)$ ，其他向量整理成 $\bar{w}(k)$ ，代表權重：

$$\Delta\bar{s}(k) = \begin{bmatrix} u(k) - u(k-1) \\ v(k) - v(k-1) \end{bmatrix} \text{ and } \bar{w}(k) = \begin{bmatrix} w_1(k) \\ w_2(k) \end{bmatrix} = \begin{bmatrix} -\frac{r_{31}}{z_c(k)} \\ \frac{r_{32}}{z_c(k)} \end{bmatrix} \quad (46)$$

其中 $\Delta\bar{s}(k)$ 可以從影像的變化得知，而 $\bar{w}(k)$ 必須想辦法估測。假設在時間為 $k-1$ 時，我們已經估測出 Jacobian 為 $\hat{\mathbf{J}}(k-1)$ ，故在時間 k 時，我們必須得到 $\hat{w}(k)$ ：

$$\hat{\mathbf{J}}(k) = \hat{\mathbf{J}}(k-1) + \Delta\hat{\mathbf{J}}(k) = \hat{\mathbf{J}}(k-1) + \Delta\bar{s}(k) \hat{w}(k)^T \quad (47)$$

$\mathbf{J}(t)$ 的定義，是將 $\dot{p}(t)$ 對應到 $\dot{s}(k)$ ，而 $\mathbf{J}(k)$ 的定義，是將 $\Delta\bar{p}(k)$ 對應到 $\Delta\bar{s}(k)$ ：

$$\Delta\bar{s}(k) = \mathbf{J}(k) \Delta\bar{p}(k) \text{ where } \Delta\bar{p}(k) = \begin{bmatrix} x_b(k) - x_b(k-1) \\ y_b(k) - y_b(k-1) \end{bmatrix} \quad (48)$$

其中 $\Delta\bar{p}(k)$ 是指令變化， $\Delta\bar{s}(k)$ 是相對應的影像變化，接著遞迴方程式代入：

$$\begin{aligned} \Delta\bar{s}(k) &= (\hat{\mathbf{J}}(k-1) + \Delta\hat{\mathbf{J}}(k)) \Delta\bar{p}(k) \\ &= (\hat{\mathbf{J}}(k-1) + \Delta\bar{s}(k) \hat{w}(k)^T) \Delta\bar{p}(k) \end{aligned} \quad (49)$$

由於 $\hat{\mathbf{J}}(k-1)$ 已知，其他項在時間 k 時可知，因此利用投影法可得 $\hat{w}(k)$ 為：

$$\hat{w}(k) = \frac{\Delta\bar{p}(k) \Delta\bar{s}(k)^T (\Delta\bar{s}(k) - \hat{\mathbf{J}}(k-1) \Delta\bar{p}(k))}{\|\Delta\bar{s}(k)\|^2 \|\Delta\bar{p}(k)\|^2} \quad (50)$$

最後，我們必須知道起始狀態的影像 Jacobian 矩陣，才可以遞迴的求出之後狀態的 Jacobian。我們的作法是在執行任務前，先任意給機器手臂兩個探索指令 Δp_1 與 Δp_2 ，他們分別對應到影像的變化為 Δs_1 與 Δs_2 ，如此一來可以得到 $\hat{\mathbf{J}}(0)$ 為：

$$\hat{\mathbf{J}}(0) = [\Delta s_1 \ \Delta s_2] [\Delta p_1 \ \Delta p_2]^{-1} \quad (51)$$

此方法稱為遞迴影像 Jacobian 矩陣即時估測法，採用的是固定式相機的觀察運動架構。我們作了三個假設，第一、機器手臂的輸入指令都在定義域內，使得手臂不會超出伸展範圍。第二、因為相機觀察的是全域的內容，所以此方法適合 eye-to-hand 的架構。第三、 $\hat{w}(k)$ 是在時間 k 時，所估測的相機參數，如果更新的權重小於設定的誤差參數 ε ，可視為雜訊，而跳過此次的更新。下圖是流程圖：

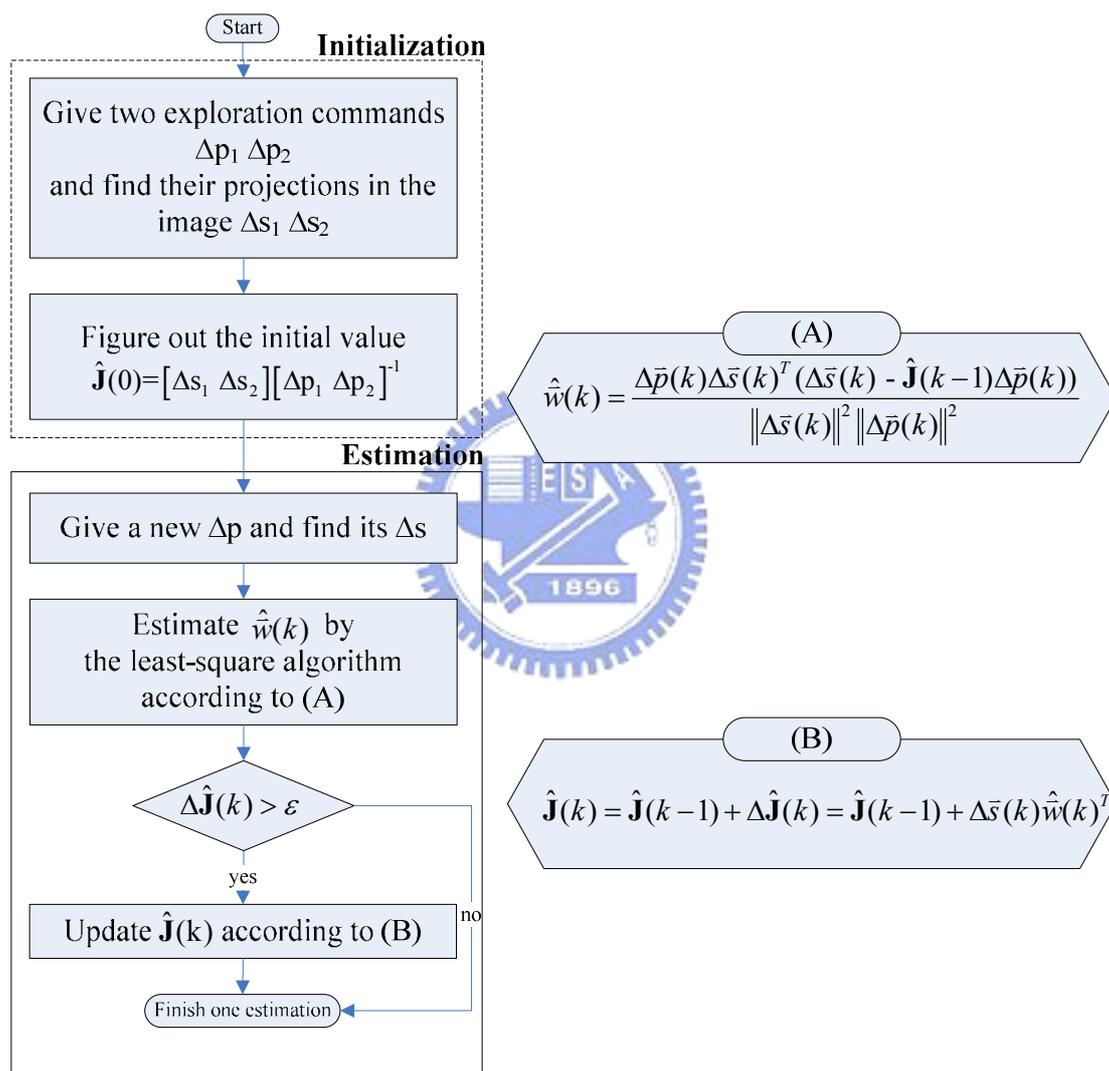


圖 19. 遞迴影像 Jacobian 矩陣即時估測法流程圖

上述的方法最大的特色是運算量少，而且容易實現。與其它的文獻相比較，此估測法的推導過程相當簡單，包含影像 Jacobian 矩陣的估測、更新，這麼做的優點是機器手臂能即時的反應，缺點是手臂必須要有起始狀態的探索動作。

事實上，從方程式中我們知道， $\hat{\mathbf{J}}(k)$ 是來自於 $\hat{\mathbf{J}}(k-1)$ 與更新項，而 $\hat{\mathbf{J}}(k-1)$ 又來自於 $\hat{\mathbf{J}}(k-2)$ 與當時的更新項，如此遞迴下去我們發現 $\mathbf{J}(k)$ 是由 $\hat{\mathbf{J}}(0)$ 與每一個時間的更新項所組合而成，這樣的話只要某次的誤差極大，會影響之後的更新。為了改善這個問題，本文推薦使用異常值偵測(outlier detection)的方法。在一群資料中，與其它筆資料明顯呈現反常的狀態，就可以稱為異常值。在資料處理的領域有非常多異常值偵測的理論，一般來說，此偵測的形式可以寫成下列形式：

$$|x^n - x^0| > k\xi \Rightarrow x^n \text{ is outlier} \quad (52)$$

其中 x^n 是某個第 n 筆的資料， x^0 是正常資料參考值， ξ 是正常資料變異量，而 k 是資料門檻的參數。這些理論就是在決定這些參數如何選取才能順利偵測。以下我們僅介紹兩種常用的方法，第一是平均數法，第二是中位數法。

1. 平均數法

此方法是高斯分佈的函數中，距離平均值越遠的值機率越小，因此：

$$x^0 = \bar{x}, k = 3, \xi = \sigma \quad (53)$$

2. 中位數法

此方法是由中位數與中位數絕對偏差 MAD(median absolute deviation)決定：

$$x^0 = x^+, k = 3, \xi = MAD \quad (54)$$

除此之外，為了 $\hat{\mathbf{J}}(0)$ 避免被誤差所影像，我們可以在起始狀態時，多給探索的指令，產生適當數量的 $\hat{\mathbf{J}}(0)$ 後，先偵測是否有異常值，有的話將之踢除，然後將剩下的資料取平均來決定 $\hat{\mathbf{J}}(0)$ 。在更新狀態也是，我們不急著將每一個時間的更新項加入 $\hat{\mathbf{J}}(k-1)$ ，而是先紀錄適當數量的 $\hat{w}(k)$ ，我們稱這個階段為訓練模式。之後稱為測試模式，此時我們已經有足夠數量來判斷接下來的 $\hat{w}(k)$ 是否為異常值。整個流程分成起始狀態、訓練模式、測試模式，其流程圖如下圖所示：

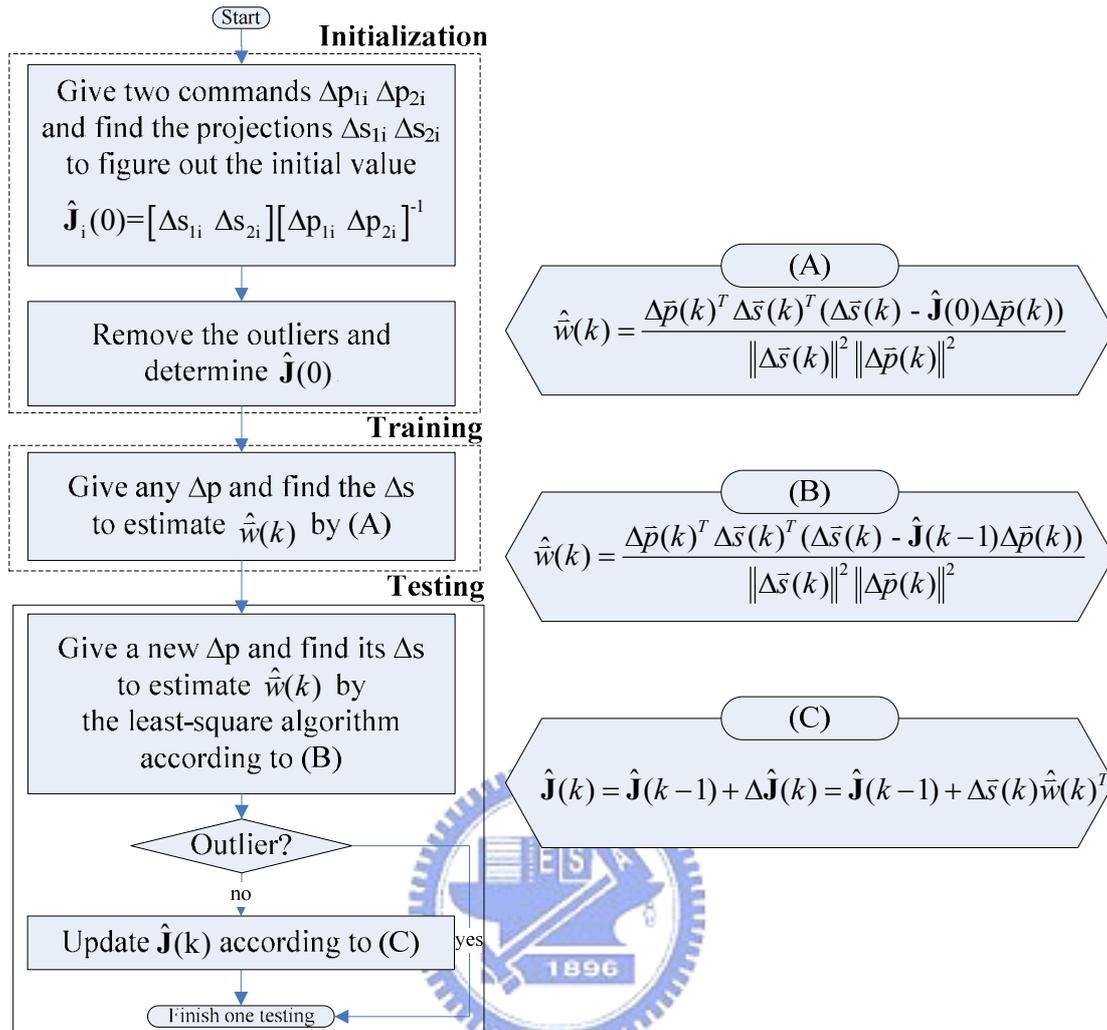


圖 20. 排除異常值之遞迴即時估測法流程圖

至此，我們完成了影像 Jacobian 矩陣的估測。此方法是採用遞迴形式即時的更新，而且方程式容易改寫成程式碼，運算量少、誤差忍受度高。然而為了抵抗誤差的影響，在測試之前，必須經過起始狀態與訓練模式，算是這個方法的缺點。

4.4 實驗模擬

本模擬是在筆記型電腦完成，CPU 規格為 Genuine Intel T2300 1.66 GHz，而 RAM 是 2GB，OS 則是 Microsoft Windows XP Home Edition Version 2002 SP3，軟體是 MATLAB version 7.0.1.24704(R14) SP1。此模擬主要是驗證即時估測法的可行性，由於 MATLAB 沒有內建與機器手臂有關的函數，因此我們在 MATLAB

Central 網站，找到了 Juan L. J. Bascones 在 2001 年公佈的運動模擬範例程式。此程式雖然只有兩個函數，一個是將世界座標系的三度空間座標值，轉換成手臂的六軸關節值，另一個是將此值送給虛擬的手臂，並且畫出其結果，但對我們來說已經足夠來模擬手臂的運動。此虛擬手臂有六個自由度，不考慮其重量與摩擦力等參數，我們不更改手臂長度與關節角度的設定，開始下面的實驗模擬。

首先，我們將機器手臂放在世界座標系的原點，而相機位置可以任意擺放。在模擬中我們擺放在世界座標系的 $[5 \ 5 \ 10]^T$ ，此為平移矩陣 \mathbf{T} ，而旋轉矩陣 \mathbf{R} 為以 X 軸旋轉 180 度，焦距 f 假設為 0.3。我們設定初始狀態的探索指令為 Δp_1 與 Δp_2 ，也就是手臂從 A 點經過 B 點，最後到達 C 點，其中 A 到 B 的指令差為 Δp_1 ，B 到 C 為 Δp_2 ，為了方便我們設定 A、B、C 三點的是以 $[1 \ 1 \ 0.1]^T$ 為中心，故 A 為 $[5 \ 1 \ 0.1]^T$ ，B 為 $[1 \ 5 \ 0.1]^T$ ，C 為 $[-3 \ 1 \ 0.1]^T$ ，我們最後可以命令手臂前往 D 點 $[-3 \ -3 \ 0.1]^T$ ，再看其中的誤差，來判斷即時估測法的準確度與誤差容忍度。

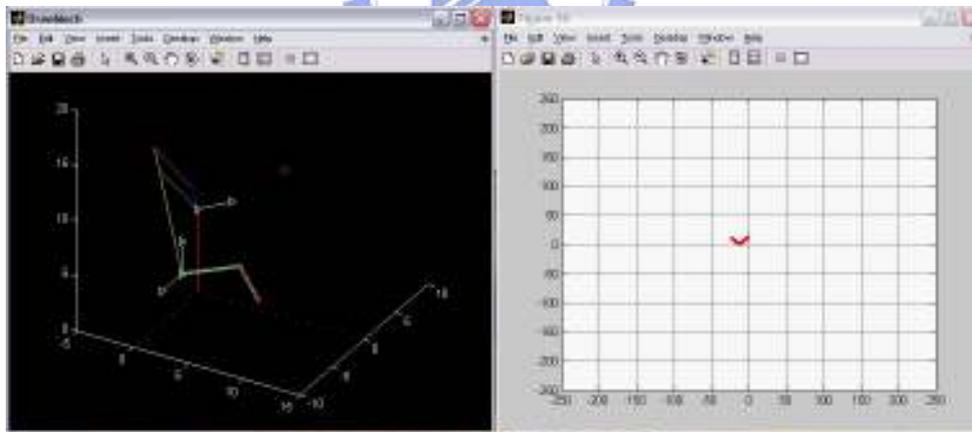


圖 21. 遞迴即時估測法模擬示意圖

上面左圖為探索指令為 Δp_1 、 Δp_2 的軌跡圖，紅點表示相機的位置。右圖是相對應的 Δs_1 與 Δs_2 ，由這些關係我們可以算出 $\hat{\mathbf{J}}(0)$ ，接下來每下一次的指令，就可以得到新的 $\Delta \bar{p}(k)$ 與 $\Delta \bar{s}(k)$ ，再由 $\hat{w}(k)$ 來求得 $\Delta \hat{\mathbf{J}}(k)$ ，最後得到 $\hat{\mathbf{J}}(k)$ 。下圖是機器手臂在探索行動後，我們命令手臂前往亂數產生的 D 點，在這邊我們假設

影像特徵已經擷取完成，而在影像加上不同的誤差，誤差的平均值是零，變異量從 0.01 到 10，故縱軸是每次估測的 $\|\hat{w}(k)\|$ 值，橫軸是實驗次數，總共四百次：

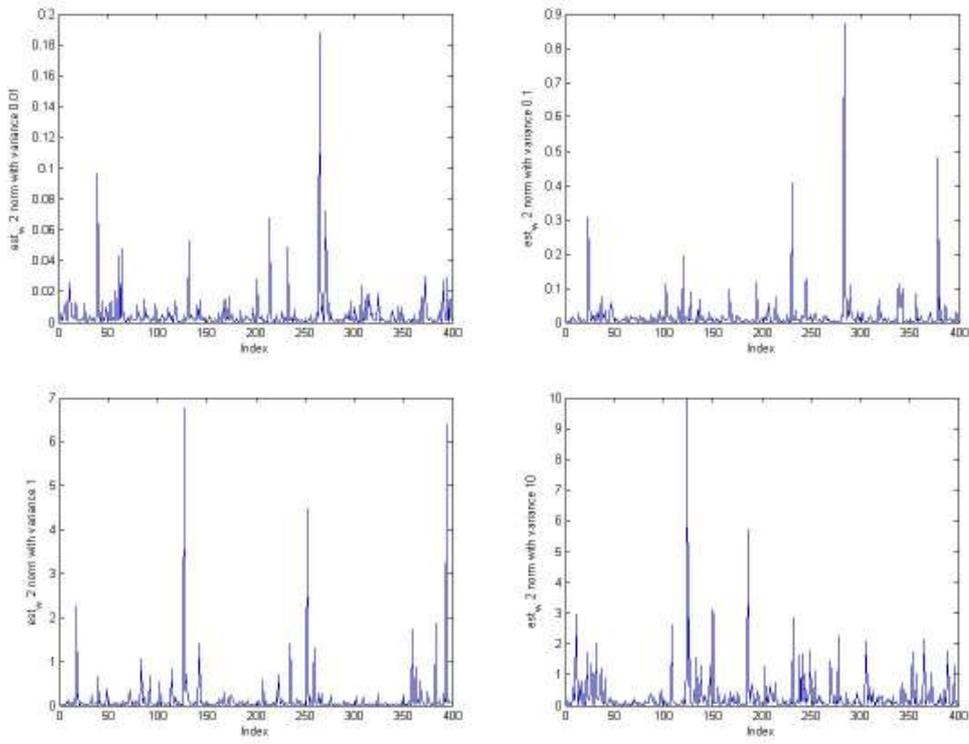


圖 22. 遞迴即時估測法模擬誤差圖

理論上，相機的外部與內部參數都不是時變函數，而且在推導中，我們假設深度變化不大甚小，故如果影像中沒有雜訊干擾，更新項應該是非常小的，然而我們為了模擬現實環境，在影像中加入誤差，此誤差包含兩種意義，一是擷取的誤差，二是影像的雜訊。模擬中我們得知，此誤差會反應在估測上，誤差越大，估測的值越大，影像 Jacobian 矩陣的更新項也就越大。總而言之，此法是在當下的時間找出最適合的 $\hat{\mathbf{J}}(k)$ ，再把 $\hat{\mathbf{J}}(k)$ 給下一個時間作為參考，而持續更新下去。

為了解決誤差帶來的影響，我們多花了一點時間在起始狀態與訓練模式。在起始狀態時作一些探索的動作，在訓練模式時，則是先紀錄適當數量的更新項，最後在測試模式時，與訓練模式的資料相比對，若發現異常值則不更新，若不是異常值就正常的更新，下圖是一開始的起始狀態探索動作，總共作了十次：

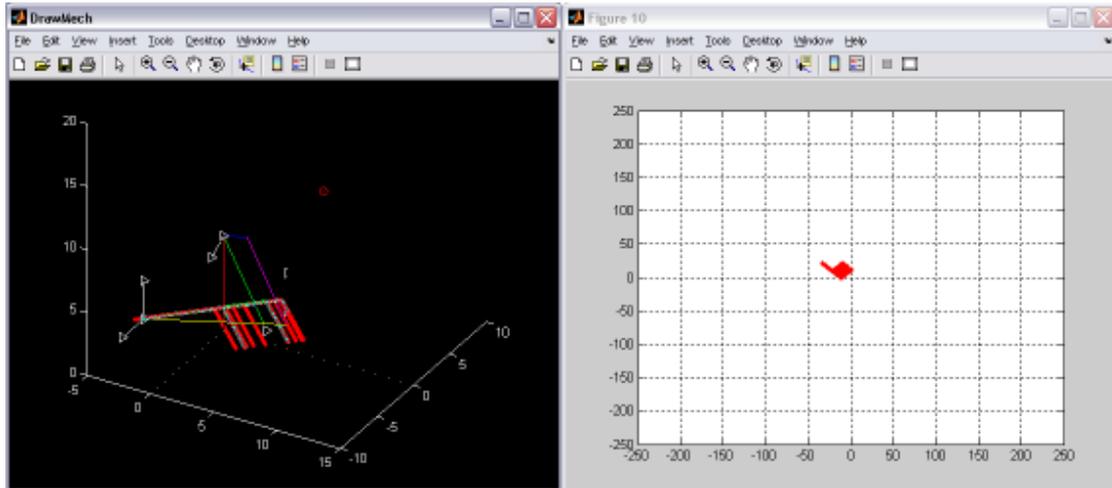


圖 23. 排除異常值之遞迴即時估測法模擬示意圖

起始狀態結束後是訓練模式。我們紀錄五十筆資料，算出平均值與變異量，要注意資料不能太少，原因是如果其中有異常值也不會嚴重影響。最後進入測試模式，與前面相同，加入平均值是零、變異量從 0.01 到 10 的誤差。橫軸是實驗次數，總共四百次，縱軸是用平均數法估測的權重值。在每次得到新的更新項後，我們會判斷是否為異常值，如果是的話就不需要更新，而讓該次的更新項為零。

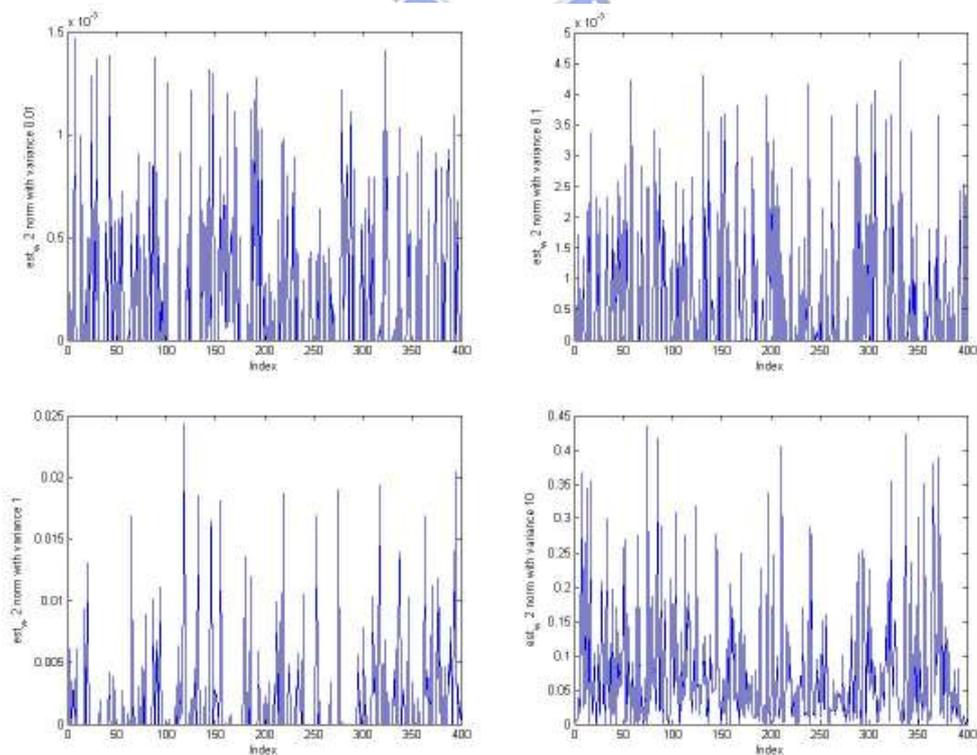


圖 24. 排除異常值之遞迴即時估測法模擬誤差圖

以上我們介紹兩種遞迴影像 Jacobian 矩陣即時估測法，第一種分成起始狀態與估測狀態。起始狀態得到 $\hat{\mathbf{J}}(0)$ 後，估測狀態在每段時間用 $\Delta\hat{\mathbf{J}}(k)$ 來更新 $\hat{\mathbf{J}}(k)$ ，為了減少運算量，我們可以設定若 $\hat{w}(k)$ 小於參數 ε 就不需更新。此方法容易遭受異常值的影響，故我們提出第二種方法，分成起始狀態、訓練模式、測試模式。起始狀態利用平均的方法消除雜訊得到 $\hat{\mathbf{J}}(0)$ ，訓練模式則是得到 $\hat{w}(k)$ 的平均值與變異量，最後在測試模式中，即可偵測 $\hat{w}(k)$ 是否屬於異常值，如果不是的話才算出 $\Delta\hat{\mathbf{J}}(k)$ 之後，更新 $\hat{\mathbf{J}}(k)$ 。從上面的圖形我們發現排除異常項之後，更新項變得較合理，原因是每段時間的 $\hat{\mathbf{J}}(k-1)$ 異常值不會影響到後面的 $\hat{\mathbf{J}}(k)$ 。

整體來看，第一種方法運算量較少，然而承受的風險較大，第二種方法花費的時間較多，而誤差忍受程度較大。在這邊我們強調，此誤差有可能是特徵擷取造成的誤差，或是影像本身的雜訊。本章節我們假設特徵已經擷取完成，而使用高斯分佈來模擬影像雜訊。在之後的章節中，我們會使用平均分布來模擬擷取的誤差，也會更好的模擬工具，模擬機器手臂移動時，有可能產生的位置誤差。

事實上，影像 Jacobian 矩陣最大的特色，不只找出怎樣的手臂變化 $\Delta\bar{x}$ ，會對應到怎樣的影像變化 $\Delta\bar{y}$ ，只要影像 Jacobian 矩陣的逆矩陣存在，而在 \mathbf{J}^{-1} 的幫忙下，我們可以知道怎樣的影像變化 $\Delta\bar{y}$ ，會得到怎樣的手臂變化 $\Delta\bar{x}$ 。本文提出的遞迴影像 Jacobian 矩陣即時估測法，是使用 $\mathbf{R}^{2 \times 2}$ 的矩陣。一般來說，只要矩陣的行列式不等於零，它的逆矩陣就會存在。如果存在的話，就可以任意操縱機器手臂，命令它去影像中的某個像素，這樣即可以達到視覺回授的效果。在程式中，我們很快能判斷影像 Jacobian 矩陣的逆矩陣是否存在，故此方法最大特色是容易實現，而且反應速度快。既可以擺脫以往僅能由跑模擬來測試的窘況，也可以在實際的在機器手臂上做實驗，更可以結合相機系統，利用特徵擷取技術，來完成整個系統。接下來我們開始將前面提出的理論與演算法，應用在真實的相機系統與機器手臂，再藉由模擬、測試、實驗等等方式，來驗證我們的想法。

五、模擬實作

本文至此介紹了視覺回授的架構，此系統分成四個子系統：機器手臂系統、相機系統、特徵擷取系統、視覺回授系統，其中前兩者以硬體為主，後兩者是以演算法為主。在硬體方面，機器手臂通常都有動作相關的指令，有些手臂單純的由微控器接受命令後執行，有些則是在上面寫好程式來執行。而相機系統的話，除了要注意檔案格式的問題之外，有些只要用電腦本身的硬體與加裝相關的驅動程式，即可讀取相機資訊，有些卻還要透過影像擷取卡，才能輸入給電腦。

一般文獻在實作上由於各硬體的種類不同，採取的方式也不同，本文用機器手臂是否有作業系統(Operating System)來分類。若是手臂有自己的 OS，第一種作法是在此 OS 安裝影像擷取卡的驅動程式，直接在手臂接收相機影像，處理後命令手臂動作。此作法缺點是必須在此 OS 跑演算法，並且還要考慮影像擷取卡的相容與支援性。第二種是在別台電腦安裝好影像擷取卡之後，將影像傳給手臂的 OS，接著處理影像與命令手臂。此作法缺點是影像傳輸十分耗時。第三種是在別台電腦擷取特徵、處理分析，最後傳命令給手臂。此作法缺點是在手臂的 OS 必須撰寫命令接受的程式，而且必要時還要回傳錯誤訊息，以方便控制。

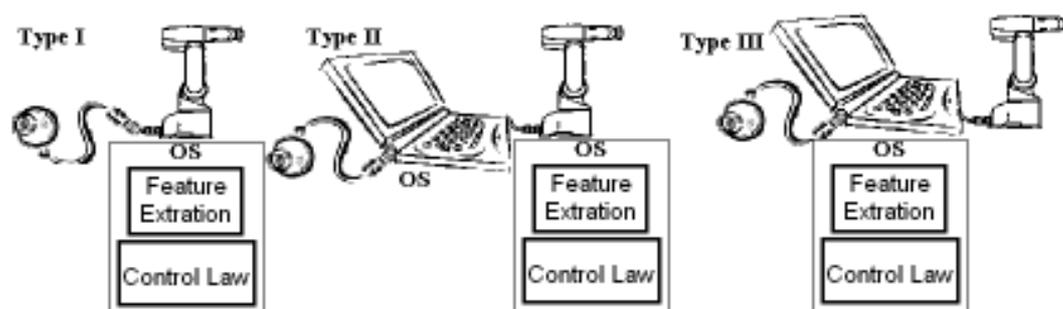


圖 25. 視覺回授硬體軟體分配架構圖

若是機器手臂沒有 OS，會有內定的通訊介面程式，來接受命令，因此要在別台電腦跑完演算法，最後傳命令給手臂。此作法的缺點，是電腦必須要與手臂傳輸通訊。幸運的是，機器手臂通常採取標準通訊規格，像是串列傳輸 RS-232、並列傳輸 Centronics、網路傳輸 TCP/IP 等等。為了適用於所有手臂的狀況，本文

推薦上述的第三種方法，也就是不論手臂本身是否有 OS，建議在別台電腦完成所有演算法，而手臂單純的接受命令，原因有四個：第一、如果要在不同的手臂作實驗，只要更改通訊協定即可。第二、可以自由選擇與相機系統相容的電腦與通用的影像擷取卡。第三、雖然在單位時間內，演算法送出許多命令，但長度很短，不會造成傳輸負擔。第四、驅動程式與軟體函數庫的選用，不會被手臂 OS 所限制。如果為了分散運算，讓手臂執行程式，反而會造成軟體移植更不方便。

以上是硬體的系統，而演算法方面，主要是實現特徵擷取與視覺回授。基本上，軟體方面都是用程式語言撰寫，比較不受硬體限制，然而特徵擷取的演算法，是把未經處理的影像資料，轉換成影像的特徵。這邊的影像處理牽涉到應用程式介面(Application Programming Interface)的軟體函數庫，此介面方便使用者有效率的實現演算法，此函數庫有些受到作業系統的限制，有些要另外安裝其他的工具箱。本文建議除了選擇與影像擷取卡相容的電腦之外，也可以選擇支援函數庫或工具箱的作業系統，這樣能夠增加軟體開發的速度，運算上也會更有效率。

而視覺回授的演算法，就是利用特徵得到機器手臂的命令，這邊的程式碼就比較單純，用基本敘述就可以達成。在程式中我們可以寫在相同的專案，意思是在特徵擷取之後，直接繼續運算結果，而其中的矩陣運算可以用函數處理，或是技巧性的撰寫來增加運算速度。最後程式中，還要包括電腦端與機器手臂之間的通訊介面傳輸協定，這邊必須依照各種協定，來個別設定與處理。

整體來說，在演算法方面，我們結合了平均位移法與遞迴即時估測法，加入了傳輸協定(protocol)。在硬體方面，我們使用本身實驗室既有的機器手臂與相機系統，解決硬體相容性與軟體支援的問題。在模擬中，我們想辦法用工具來模擬現實的狀況，在實作中，我們試圖克服問題來完成任務。以下是本文視覺回授的系統模擬與系統實作，本文所選用的任務是球類運動，下面是此運動詳細的介紹。

5.1 任務介紹

球類運動可以分成兩種，第一種是用手、腳或身體控制球，譬如籃球、足球、排球、手球、保齡球等等，第二種是用球具控制球，譬如棒球、曲棍球、桌球、羽毛球、網球、撞球等等。無論是哪一種，這些運動都需要兩種能力，第一種是位置的判斷，譬如球在哪裡、目標在哪裡等等，第二種是時機的判斷，譬如球速多少、什麼時候到達等等。現今對於機器人的球類運動研究算是非常熱門，因為球類運動是十分複雜而且具有挑戰性的問題。有些方法，是利用立體視覺來判斷位置，將球類運動方程式化，來判斷時機。有些則是使用學習理論、模糊控制等等有非常多的方式。本文提出用視覺的回授，來解決這方面的問題。以往的視覺回授所呈現的任務也比較簡單，例如 peg-in-hole、pick-and-place 等等。本文藉由改善即時估測法、擷取穩定特徵，能在更複雜的環境，執行更艱難的任務。

由於機器手臂的前端是可以加裝手掌，我們可以裝上兩指的手掌夾住球拍，或是裝上五指的手掌來拍球或是投球，而球類運動的範圍又太過廣泛，為了縮小範圍，因此我們針對使用球具的運動來分析，此運動最大的困難在於球具與球的接觸位置、速度、時機。我們首先必須先解決擊球位置的問題，模擬之中，我們先鎖定挑球，方法是用球拍，將球連續往上擊球的熱身動作。挑球是培養球感的動作，像高爾夫、桌球等等運動中，可以用這種動作，來增加球具的手感。以下我們應用視覺回授的技術，來討論擊球位置的準確度與誤差容忍度的問題。

5.2 模擬簡介

為了能有更逼真的模擬，我們在這邊使用 P.I. Corke 在 2008 年公佈的第八版 MATLAB 的機器手臂工具箱。此工具箱適用於 MATLAB 5.0 以上，提供了許多常用的運動學函數，也提供一些知名的機器手臂模組，比如 Puma 560 與 Stanford Arm。在[46]中，有此工具箱的基本介紹，在此工具箱內，也有函數的介紹文件。我們先將 MATLAB 的路徑設好後，測試它的範例程式。在程式中模擬了一般的機器手臂介面，有圖形顯示與手動操作的關節、位置移動，其畫面如下圖所示：

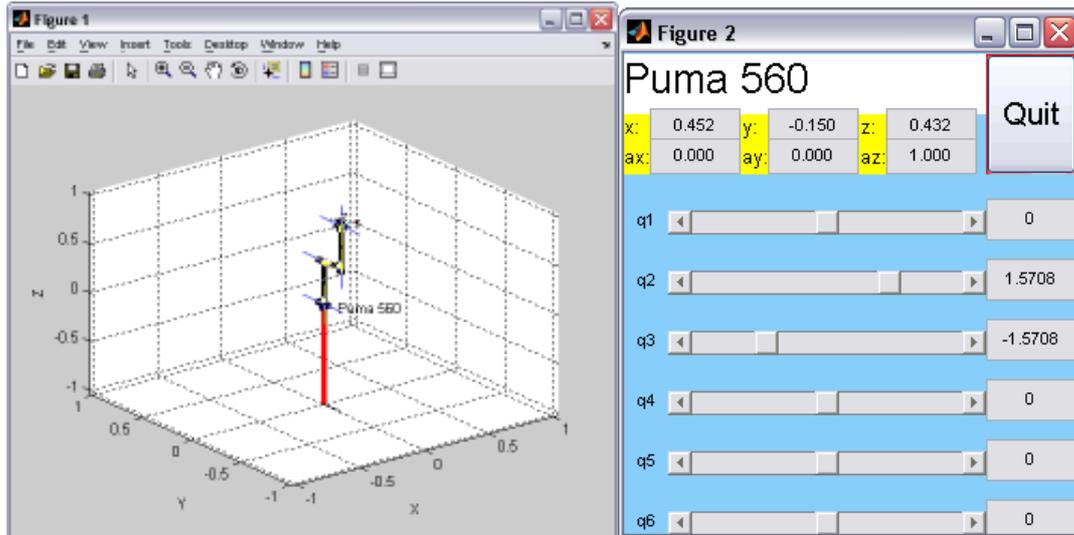


圖 26. MATLAB 機器手臂工具箱原始使用介面圖示

由於我們只需要上左圖的手臂顯示，而不需要上右圖的手動操作，因此我們從頭撰寫了一個新的程式，程式中呼叫其手臂顯示的函數即可。然而此顯示函數不包含球拍的顯示，我們新增了一個長寬各為 0.2m 的球拍，如下圖所示：

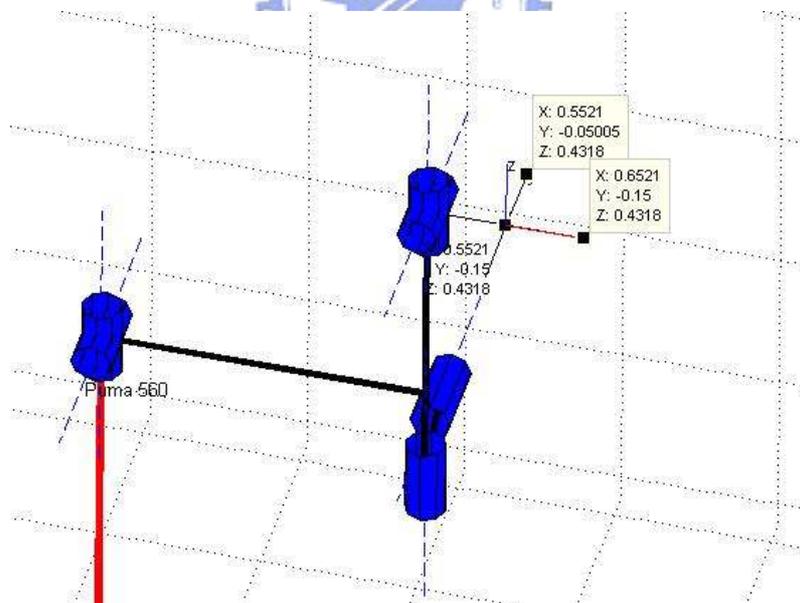


圖 27. MATLAB 機器手臂工具箱新增顯示函數圖示

我們使用六軸的 Puma 560 來模擬，原因是此工具箱有最佳化過的 Puma 560 反運動學函數，與現實的機器手臂較為相似。至於手臂質量、長度等等其他參數，

呼叫該模組後會自動設定，我們不加以更動。而相機的部份，此工具箱也有逼真的相機系統能使用，我們設定影像兩軸平行，歪斜參數 γ 為 0，影像長與寬都為 512 像素，故相機的原點 (p_x, p_y) 為 (256,256)，而 (α_x, α_y) 為 (640,640)。

有了這些硬體模組，我們在 MATLAB 中建立一個沒有空氣摩擦阻力、重力與地球相同的物理慣性座標系。為了方便觀察，我們先假設球拍與球的碰撞恢復係數為 1，意思是球以多少速度打到球拍，就以多少速度反彈。我們以機器手臂的原點當作是世界座標系的原點，球的位置在 $(-0.4, -0.4, 0.5)$ ，相機的位置設定在 $(0, 0, 3)$ ，下左圖是世界座標系，紅點為球，下右圖的右邊空心圈為目前手臂末端在影像的位置，左邊空心圈為目前球在影像的位置，如下圖所示：

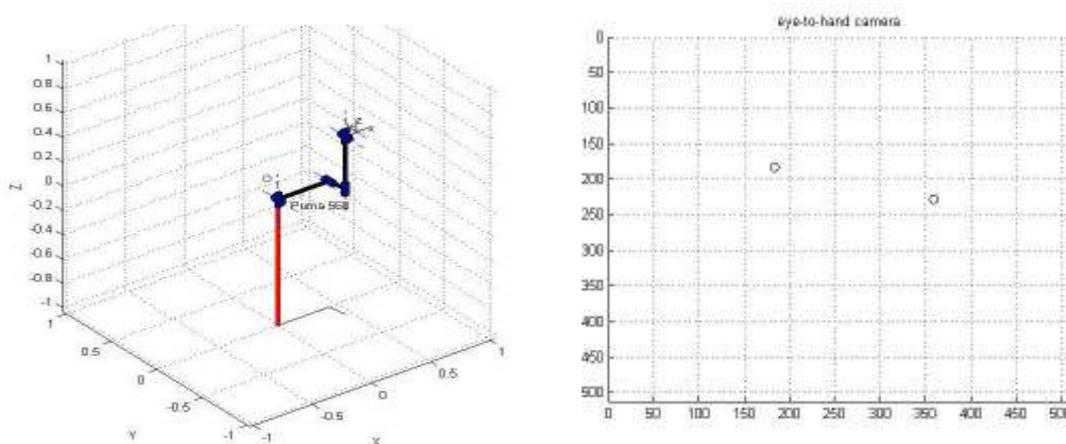


圖 28. MATLAB 世界座標系與相機座標系模擬圖

架構機器手臂與相機系統之後，我們開始模擬的實驗。在這邊我們假設球與球拍是空間中的質點，投影在相機的影像平面。有了這些影像特徵，再代入我們的遞迴即時估測法，其中球的在影像平面的位置，是機器手臂想要到達的位置。更詳細的說，遞迴即時估測法在起始狀態，得到影像 Jacobian 矩陣，此矩陣找出怎樣的手臂變化 $\Delta \bar{x}$ ，會對應到怎樣的影像變化 $\Delta \bar{y}$ 。之後在估測狀態，只要它的逆矩陣存在，由於知道利用影像變化 $\Delta \bar{y}$ ，來得到手臂指令變化 $\Delta \bar{x}$ ，因此在每段單位時間內，我們算出球在影像中新的位置變化，反推出該用什麼指令，來移動手臂到該點，再觀察手臂是否有誤差。整個模擬的流程可以用下圖來呈現：

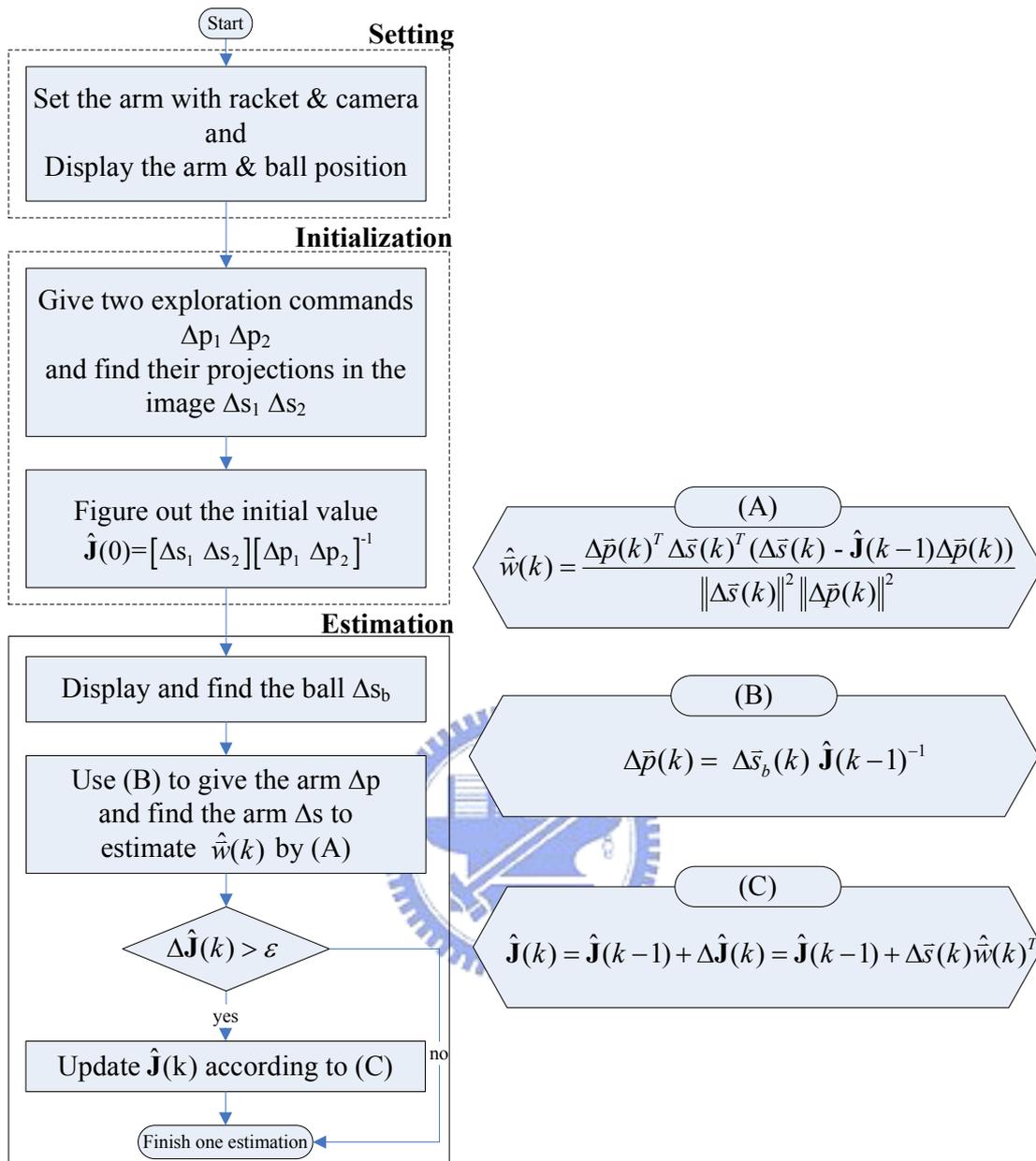


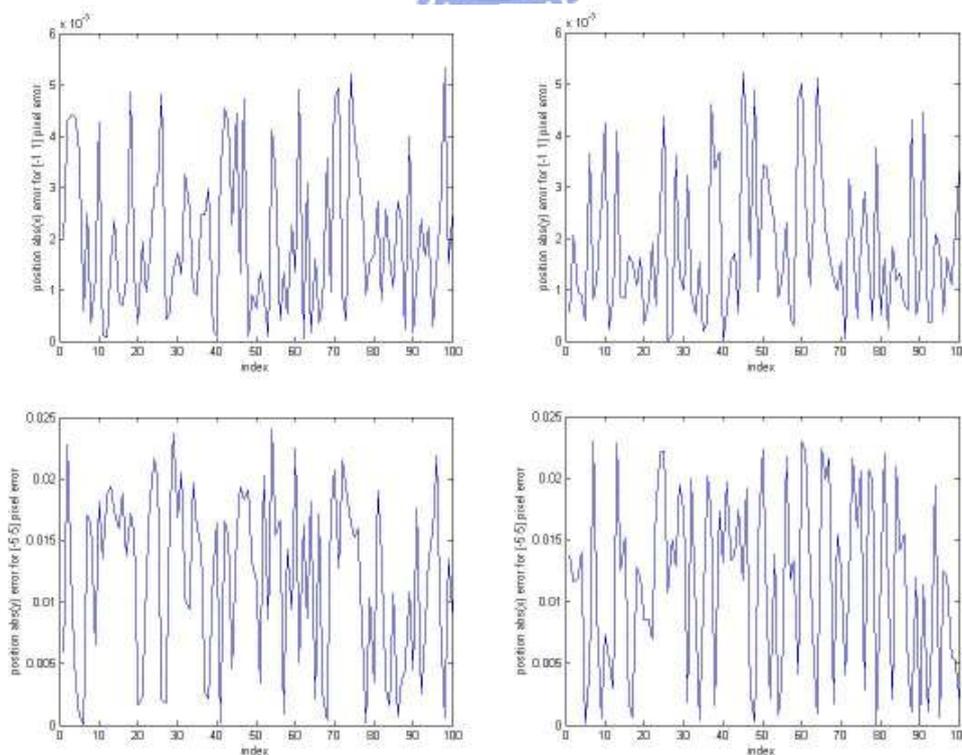
圖 29. MATLAB 模擬程式流程圖

5.3 模擬結果

本模擬是在筆記型電腦完成，CPU 規格為 Genuine Intel T2300 1.66 GHz，而 RAM 是 2GB，OS 則是 Microsoft Windows XP Home Edition Version 2002 SP3，MATLAB 版本是 7.0.1.24704(R14) SP1。由於將來實作時發現誤差極少出現異常值，而使用遞迴即時估測法比較符合現實，因此我們不用排除異常值的方法，在

模擬時同樣也使用遞迴即時估測法來估測影像 Jacobian 矩陣。與前面相同，我們在每段時間任意命令手臂，用 $\Delta\bar{p}(k)$ 來得到手臂 $\Delta\bar{s}(k)$ ，接著用 $\hat{w}(k)$ 來算出 $\Delta\hat{\mathbf{J}}(k)$ 即可更新 $\hat{\mathbf{J}}(k)$ 。而不同的是，在這邊手臂的 $\Delta\bar{p}(k)$ 是用球的 $\Delta\bar{s}(k)$ 來得到，作法是先用手球的 $\Delta\bar{s}(k)$ 乘上 $\hat{\mathbf{J}}(k-1)^{-1}$ 得到手臂的 $\Delta\bar{p}(k)$ ，故手臂拿著球拍移動後，再用前面的方法更新 $\hat{\mathbf{J}}(k)$ 。此作法如果機器手臂的準確度很高的話，球拍移動後的影像位置會與球的影像位置非常相近，故 $\hat{w}(k)$ 會很小，可以不更新 $\hat{\mathbf{J}}(k)$ 。值得注意的是，必須同時擷取球與球拍的特徵，演算法要能作到多物體特徵追蹤。

為了測試此想法，在球的影像加入了誤差。之前我們強調，影像誤差有可能是特徵擷取造成的誤差，或是影像本身的雜訊，而在前面的模擬中，我們用高斯分佈，來模擬影像雜訊，故這邊我們用平均分布來模擬擷取的誤差。下圖是模擬的結果，左邊是球拍與球在 X 方向的絕對值差，右邊是球拍與球在 Y 的絕對值差，第一排是平均分布在區域 $[-1,1]$ 之間的影像誤差，第二排是 $[-5,5]$ ，而第三是 $[-10,10]$ ，第四是 $[-15,15]$ ，我們可以注意影像誤差造成的位置誤差有多大：



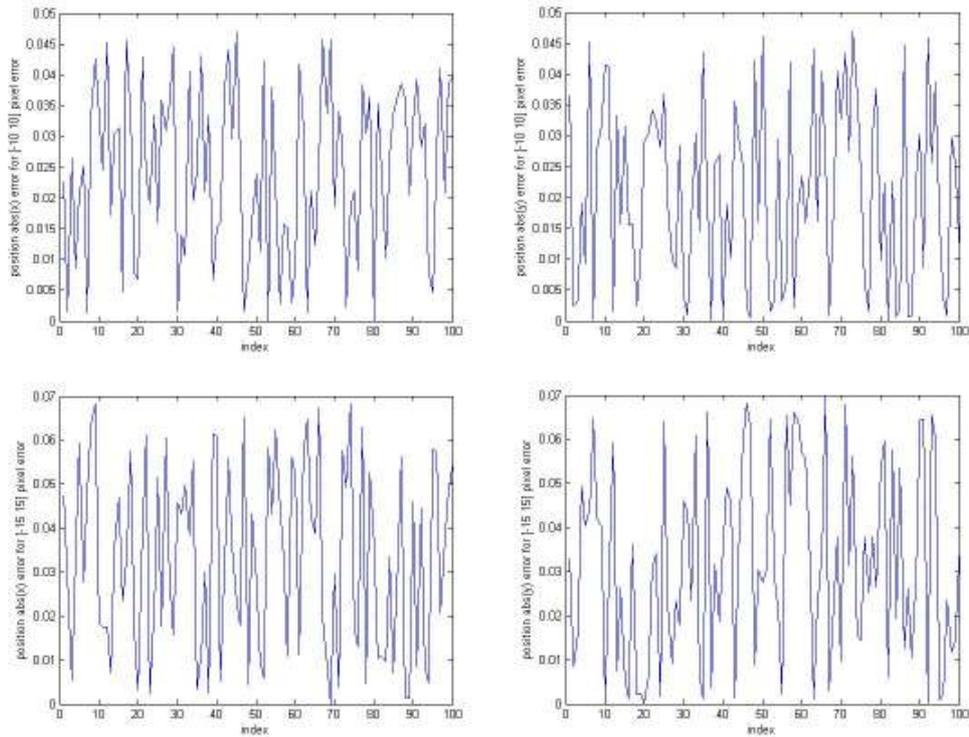
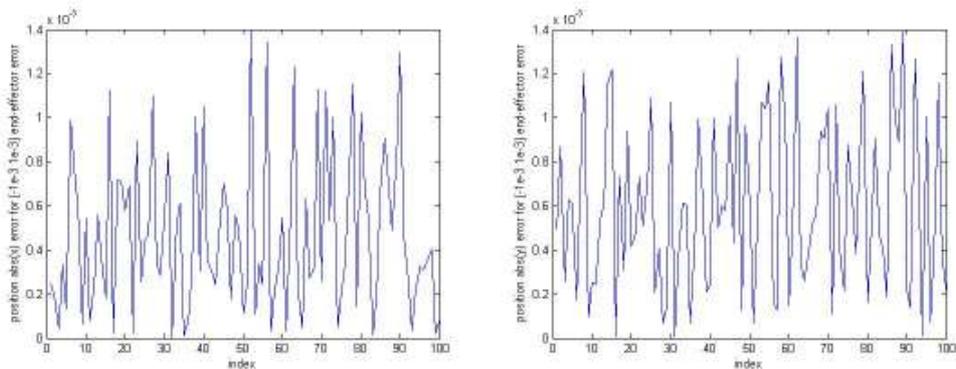


圖 30. 相機影像誤差造成的位置誤差模擬圖

我們發現當影像有平均分布在區域 $[-15, 15]$ 之間的影像誤差時，會造成球拍中心與球有 $0.07m$ 的誤差，而我們球拍的大小是 $0.2m$ ，故如果誤差大於 $0.1 m$ ，則球拍將打不到球，此任務宣告失敗。事實上如果影像有 $[-15, 15]$ 以上的誤差時，就表示特徵擷取已經不夠準確，可推論前面的演算法已經失效。除了影像的誤差會造成擊球失敗，還有可能是機器手臂精確度不夠，這表示我們原本命令末端去空間中的某座標，而手臂卻無法到達該點，故此誤差也是應該考慮的。下圖左邊是球拍與球在 X 方向的絕對值差，右邊是球拍與球在 Y 的絕對值差：



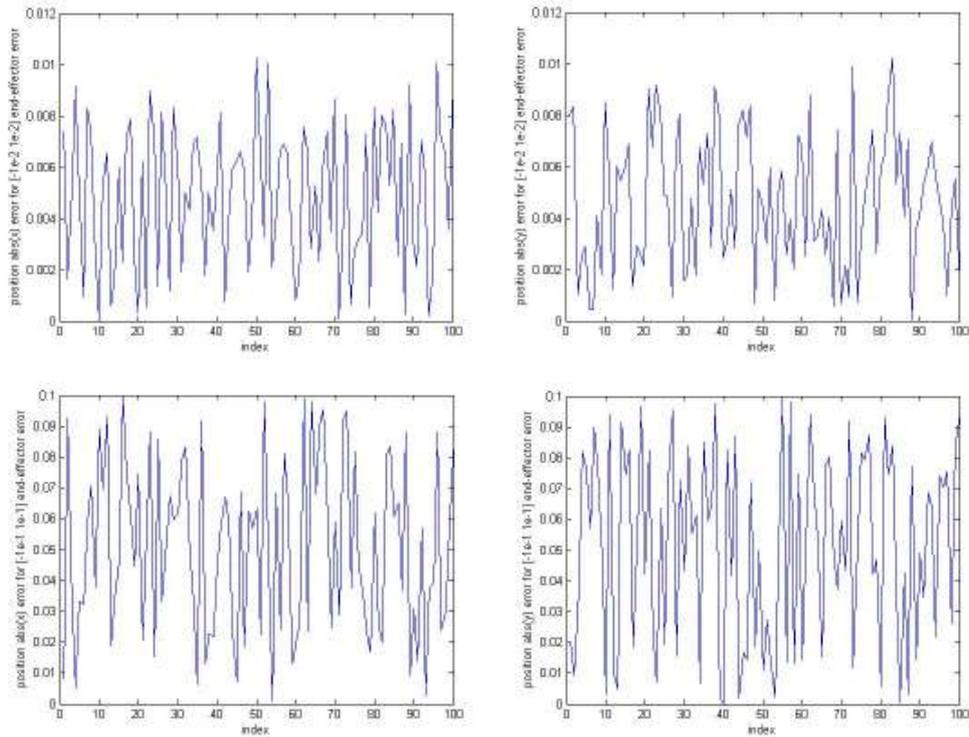


圖 31. 手臂位置誤差造成的位置誤差模擬圖

上圖第一排是平均分布在區域 $[-1e-3, 1e-3]$ 之間的機器手臂位置誤差，第二排是 $[-1e-2, 1e-2]$ ，而第三是 $[-1e-1, 1e-1]$ 。此球拍大小是 0.2m，故如果誤差大於 0.1 m，則任務將失敗，我們發現手臂位置誤差嚴重影響任務的表現，故在實作時要特別注意機器手臂的狀況，觀察是否能準確的移動末端到達所指定的位置。

上面的模擬我們用機器手臂呈現球類運動的任務，其中使用球拍擊球，最大的困難在於球具與球的接觸位置、速度、時機。在模擬中，實際的擊球點是未知的，意思是機器手臂不知道手臂末端與球拍之間的距離，而我們的演算法是藉由影像回授，來估測球拍與球在影像的位置，故可以解決擊球位置的問題。而擊球速度與擊球時機方面，模擬時我們假設是彈性碰撞，省略了這部份。在實作中，我們可以有技巧的用手臂來解決速度與時機的問題，完成球類運動的任務。

5.4 實作簡介

本章一開始有提到硬體、軟體的選擇問題，像是機器手臂是否有內建 OS、

影像擷取卡是否能相容、函數庫是否能支援等等。經過這些分析之後，本文決定選擇第三種架構，也就是在電腦完成所有演算法後，傳輸命令給機器手臂。這個架構的優點是能自由選擇電腦的 OS 與配備，不會有相容與支援的問題。雖然在手臂端只有命令的接收與執行，但單位時間內其實有很多命令等待被執行，故不適合應用分散式運算的觀念。下圖是我們實作的架構圖與考慮誤差的狀況：

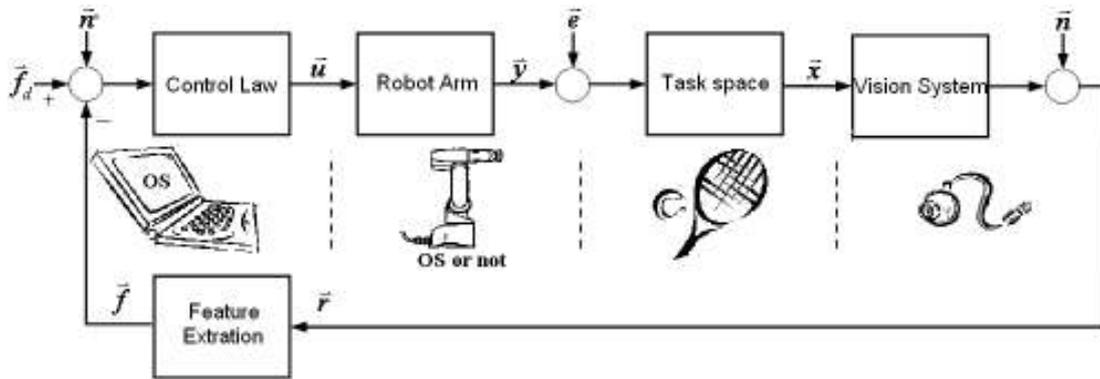


圖 32. 影像基礎視覺觀察運動實作架構圖

上圖中我們在電腦跑完擷取特徵與視覺回授的演算法，其中 \bar{n} 是影像擷取造成的誤差，是我們可以控制的，而 \bar{f} 是機器手臂末端球拍的影像特徵， \bar{f}_d 是球的影像特徵。在演算法之後，我們通過通訊協定命令手臂移動， \bar{e} 是機器手臂末端的移動誤差。手臂移動後，球拍與球投影到相機，其中 \bar{n} 是影像的雜訊，而影像最後回授給電腦。有了這個架構，接下來選擇各種軟硬體，依序介紹如下：

1. 電腦系統

電腦系統有個人電腦與筆記型電腦能選擇，而個人電腦的主機板與影像擷取卡的相容性比較高。由於在成本的考量之下，我們不增加影像擷取卡的硬體，而尋找一般電腦都能使用的影像擷取方法，像是用 USB 傳輸的相機系統，因此不限制個人電腦或筆記型電腦的選擇。在作業系統的選擇上，Microsoft Windows XP 支援較多的驅動程式，也支援許多應用程式介面的軟體函數庫，故本文使用作業系統是 Microsoft Windows XP Home Edition Version 2002 SP3，硬體是 TOSHIBA 的筆記型電腦 M100，CPU 為 Genuine Intel T2300 1.66 GHz，而 RAM 為 2GB。

2.相機系統

我們選擇不需要影像擷取卡、一般電腦都能使用的相機系統，故使用 USB 網路攝影機。USB 介面是主機板與作業系統通用的介面，只要在 OS 內安裝驅動程式即可處理影像。在這邊我們選取的是 30 萬像素的 CMOS Logitech Quick-cam Connect，它適用於 Windows XP 與 Windows Vista，每秒最多 30 張畫面。

表格 3. Logitech Quick-cam 規格

Logitech Quick-cam connect	
Quality	VGA resolution Provides clear, vibrant images at 640 x 480 pixel resolution
System Requirements	Windows® XP (SP2) Pentium III 700MHz or AMD Athlon 2.2 GHz processor 256 MB, RAM 250 MB (on hard disk), CD-ROM drive
Port	Available USB 1.1 or 2.0 USB
Frame rate	Up to 30 frames per second
Focus	Manual focus
LCD	Universal monitor clip for LCD, CRT, or notebooks
Limit	2-year limited warranty



圖 33. 實作使用的相機系統

3.機器手臂

機器手臂基本上非常昂貴，我們使用的是實驗室既有的 Mitsubishi RV-M2，此機器手臂分成控制器、教導器、五軸關節的手臂。其相關的規格如下表：

表格 4. Mitsubishi RV-M2 手臂規格

MOVEMASTER RV-M2		
Construction	Vertical, articulated	
Degrees of freedom	5 (not including hand)	
Electrical drive system	DC servo motors	
Reach	250 + 200mm (120mm offset)	
Operation range	Waist rotation	300°(max. 140°/s)
	Shoulder rotation	130°(max. 79°/s)
	Elbow rotation	120°(max. 140°/s)
	Wrist pitch	±110°(max. 163°/s)
	Wrist roll	±180°(max. 223°/s)
Maximum path velocity	1,500mm/s (PTP at wrist tool plate)	
Lifting capacity	2.0kg incl. hand	
Position repeatability	±0.1mm at wrist tool plate	
Installation position	Horizontal	
Ambient temperature	5°C - 40°C	
Weight	28kg	

表格 5. Mitsubishi RV-M2 控制器規格

Drive Unit	
Teaching method	Teaching and MDI (manual data input)
Control system	PTP and CP
Speed levels	21
Position memory	999 positions (max.)
Position memory	3,584 lines (max.)
Interface	Serial (RS-232C), parallel (Centronics)
External I/Os	16 inputs/outputs, EMERGENCY STOP (3 inputs configurable for interrupts)
Ambient temperature	5°C - 40°C
Power supply	120V AC, single phase, 50/60Hz, 500VA
Dimensions	380mm x 331mm x 246mm (W, H, D)
Weight	25 kg

我們發現此手臂是支援串列傳輸 RS-232C，故可以在電腦跑完演算法之後，透過 RS-232C 將指令傳給機器手臂，也可以從 RS-232C 傳回錯誤的訊息來處理。



圖 34. 實作使用的機器手臂

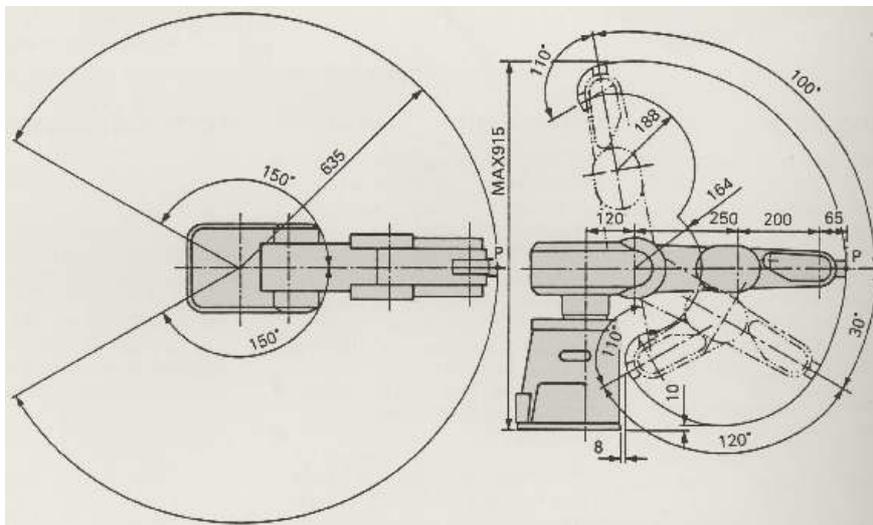


圖 35. 機器手臂的工作範圍

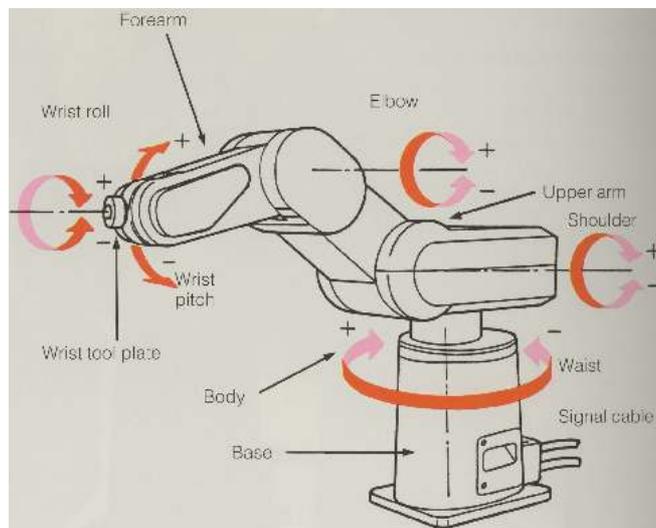


圖 36. 機器手臂五軸示意圖

4.開發軟體

由於我們選用 Windows XP 為作業系統，因此可以找到很多可支援函數庫來協助。本文使用 Microsoft Visual C++ 為開發環境，函數庫的部分則是使用了 MFC 與 OpenCV。MFC 全名是 Microsoft Foundation Classes，是微軟公司提供的類別庫(class libraries)，以 C++ class 的形式，封裝了 Windows 的 API，並且包含一個應用程式框架，以減少應用程式開發人員的工作量，其中 class 包含大量 Windows 控制代碼封裝 class，與很多 Windows 的內建控制項和零件的封裝 class。

MFC 是在 1992 年隨著微軟 C/C++ 7.0 編譯器發布，用於 16 位元 Windows 的軟體開發。MFC 的主要優點，是提供應用程式開發的便捷，我們可以用物件導向的方法，來呼叫 Windows API。缺點是雖然 MFC 的原始碼對使用者是完全開放的，但是 MFC 的一些封裝過程過於複雜，導致新使用者很難迅速掌握 MFC 的應用程式框架，以及在偵測錯誤中定位問題的位置。況且 MFC 物件在執行時，不保證線程(thread)的安全，故在線程之間存取物件時，需要額外的程式。



表格 6. Visual C++與 MFC 版本對照表

Product version	MFC version	Year
Microsoft C/C++ 7.0	MFC 1.0	1992
Visual C++ 1.0	MFC 2.0	
Visual C++ 2.0	MFC 3.0	
Visual C++ 4.0	MFC 4.0 (mfc40.dll included with Windows 95)	1995
Visual C++ 5.0	MFC 4.21 (mfc42.dll), major upgrade MFC 4.2.	
Visual C++ 6.0	MFC 6.0 (mfc60.dll)	1998

OpenCV 的全名是 Open Source Computer Vision Library，是跨平台的計算機視覺庫。OpenCV 是由英特爾公司(Intel)發起並參與開發，以 BSD 許可證(license)授權發行。OpenCV 可以在商業和研究領域中免費使用，適用於開發即時的圖像處理、計算機視覺以及模式識別程序。OpenCV 在 2006 年發布了第一版，至今仍然在更新中。OpenCV 優點是免費的跨平台函數庫，效率高、移植性高，非常

強大，而缺點是 OpenCV 雖然對影像處理的矩陣運算支援度高，但對於其他應用程式卻沒有相關的函數可以使用。我們在撰寫程式碼時發現，如果只使用 MFC 的函數，對機器手臂的通訊相當方便，卻不能處理兩個相機以上的輸入，而如果只使用 OpenCV 的話，對影像處理十分有利，卻無法處理傳輸介面。

本文各取 MFC 與 OpenCV 兩種函數庫的優點，在特徵擷取方面，我們呼叫 OpenCV 的函數，利用強力的影像處理函數，迅速的處理兩個相機。在通訊協定方面，我們呼叫 MFC 的函數，建立電腦與機器手臂傳輸，整個程式流程如下：

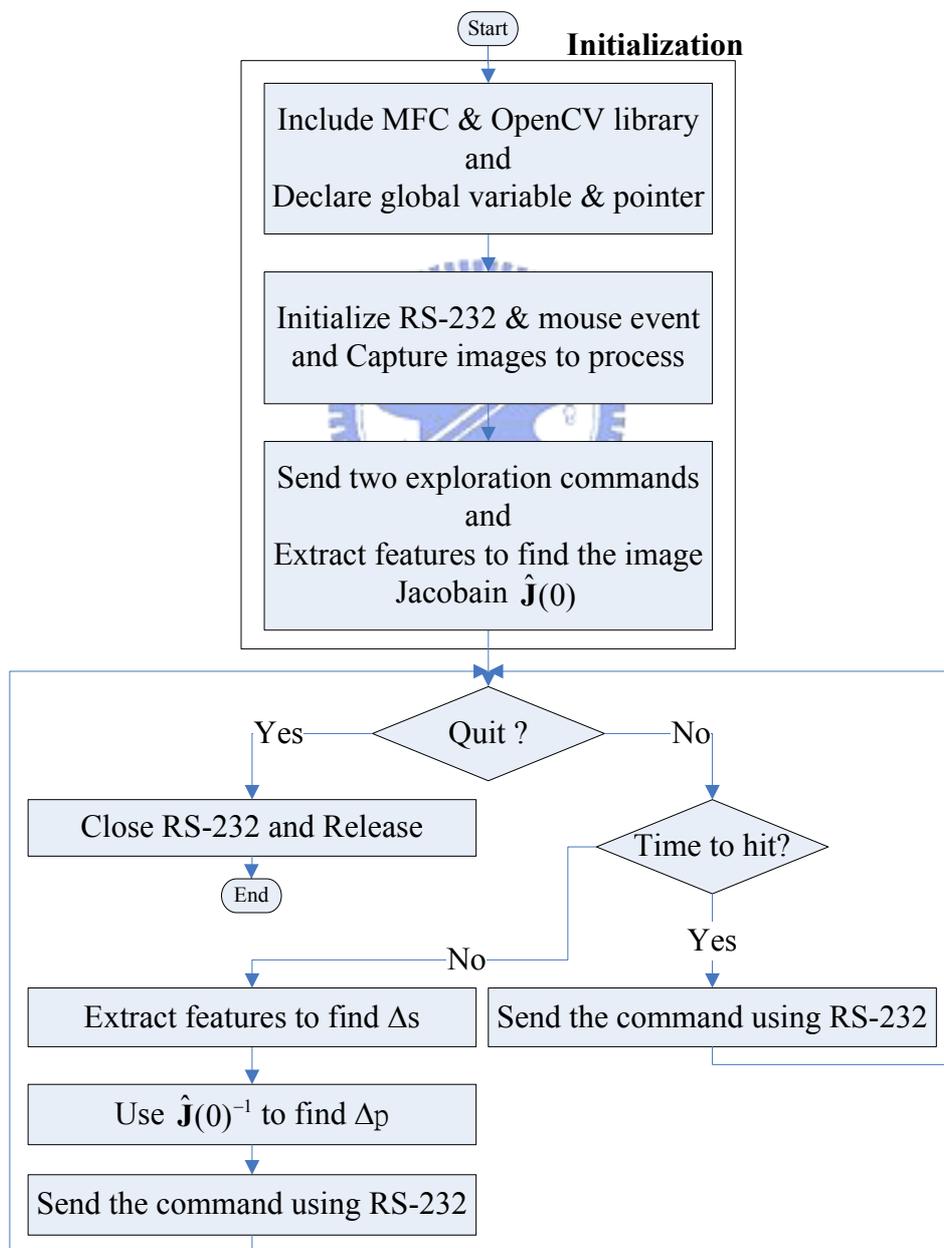


圖 37. MFC 與 OpenCV 實際程式流程圖

5.5 實作結果

在前面我們提到，使用球具的運動牽涉到手掌控制，需要更多的自由度，故我們省略這部份，主要針對使用球具的運動來分析。在模擬與實作時，我們首先在機器手臂末端上固定球拍，故末端與球拍沒有相對的移動。接著安裝了相機，由於相機沒有校正，因此我們不知道其確切位置，不過我們假設球拍與球都不會超出相機感測的範圍，故在程式中不用增加影像邊緣的物體偵測功能。最後啟動機器手臂，先下達探索指令，估測起始影像 Jacobian 矩陣。我們發現手臂末端的實際位置，與命令的位置誤差 \bar{e} 極低，故我們不必更新矩陣。在這邊我們強調，理論上遞迴即時估測法可以更新矩陣，讓影像與手臂造成的誤差達到最小，然而在實際上影像的雜訊 \bar{n} 很小，平均位移法又可以降低特徵擷取的誤差 \bar{n}' ，再加上手臂準確度很高，才能節省更新的運算步驟，把資源運用在影像處理，如此一來就不需要多物體的特徵擷取，省下了更多影像處理的記憶體空間與運算時間。

因為上述的原因，所以手臂探索之後，特徵擷取的目標從原本的球拍轉移到球上面。為了讓球拍擊中球，必須達成三個條件，第一個是球的位置判斷，第二是球到達的時機速度判斷，第三是決定球拍速度。在模擬之中，我們鎖定挑球的運動，用視覺回授的觀念，揮動球拍連續往上打球，而時機與速度是假設為完全碰撞。在實作中，我們是用一台相機視覺回授，判斷球的位置，至於擊球時機與速度，是用另一台相機來判斷球的移動，這邊使用的演算法稍後會介紹。我們將實作分成兩個階段，第一階段是單純的位置判斷，第二是加上位置與時機判斷。

1. 球的位置判斷實驗

本實驗是用一台相機視覺回授，不需要多線程，故使用 MFC 即可。我們一開始用白色框架顯示手臂末端手套的位置，為下圖的前兩張圖。接著手臂作探索的命令，移動手套估測影像 Jacobian 矩陣，為下圖的第三、四、五張圖。之後再切換模式，讓影像特徵追蹤球，而移動手套到球的位置，為下圖後四張圖。我們擷取影片中編號為 20、40、160、250、290、300、520、590、630 的 frame 如下：

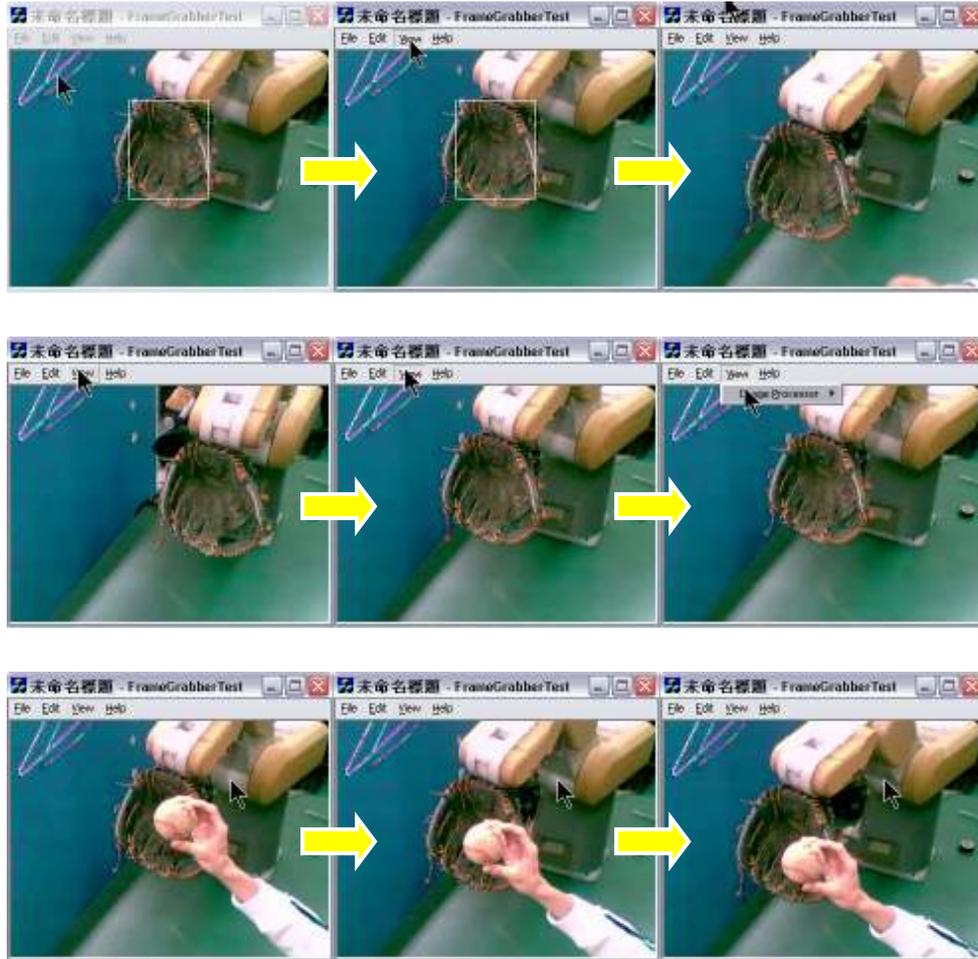


圖 38. 棒球與手套的位置判斷實驗

我們由視覺回授得知手臂上手套與相機位置的關係。為了測試平均位移法的準確度，我們選擇相似的背景與前景，挑戰比較不好處理的影像狀況，其中白色框架表示要追蹤的起始影像特徵，白色十字顯示目前追蹤的影像特徵。下圖前七張圖是探索的命令之後，估測影像 Jacobian 矩陣。接著切換模式，追蹤球的影像特徵，移動球拍到球的位置。我們從零秒開始，每兩秒擷取影片，圖形依序如下：





圖 39. 球拍與球的位置判斷實驗

2. 擊球實驗

本實驗是機器手臂搭配兩台相機完成，因為我們知道 MFC 比較難處理多台相機，所以影像部分換成 OpenCV 來處理，而 MFC 保留了與手臂通訊的部份。兩次實驗相同的是追蹤的起始特徵，是由滑鼠來點選，原因是避免自動偵測造成的錯誤。不同的是除了有上個實驗的位置判斷，還加入了擊球的速度時機判斷。

我們使用的球棒直徑約為 0.08m，總長約為 0.35m，而球是使用一般的乒乓球，直徑約為 0.04m。由於我們投球的位置距離機器手臂約為 2m，而投出球速約為 2m/s，因此從球出手到打擊的時間間隔不到一秒。除此之外，我們每次從不同的角度、不同的位置出手，故此運動必須判斷球的位置、擊球的速度與時機，需要兩台相機來處理，其中一台相機負責視覺回授，另一台相機負責運動偵測。運動偵測(motion detection)在電腦視覺中，是影像處理常見的偵測。此偵測可針對在影像當中，是否有正在移動的物體。傳統運動偵測的方式，是將觀察的環境建立成模型，而把之後觀察的內容與此模型相比，就可以知道是否有改變，可是這樣缺點是容易受雜訊影響。我們的作法是動態的更新模型，再加入低通濾波器減少模型的雜訊，此濾波器選用的是附近區域的中位數低通濾波器。流程圖如下：

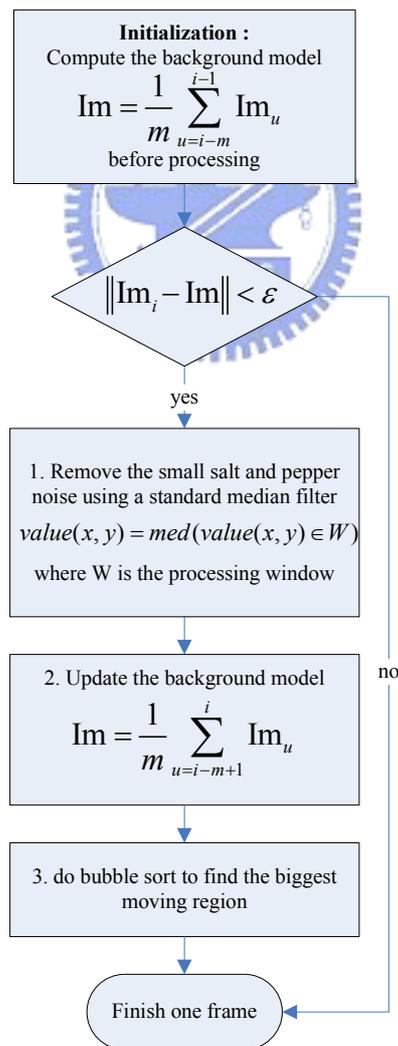


圖 40. 運動偵測程式流程圖

我們作法如下，如果以機器手臂為世界座標系的原點，第一台相機位置大約在(0m,1.8m,0.2m)地方，而第二台相機大約在(0m,1.2m,-0.6m)。第一台觀察投手的球，負責判斷球的位置。第二台則是觀察投手的動作，判斷擊球的速度與時機。一開始與前面相同，我們讓手臂拿著球棒，先往左邊探索，接著回原來位置，再往上探索，如此一來可以估測出第一台相機與球棒的位置對應關係，此關係就是影像 Jacobian 矩陣。下面是我們擷取影片中第 10、15、18、20 秒的圖，可以發現手臂是先往左邊探索，接著回原來位置，再往上面探索後回原來位置：



圖 41. 球棒與球的打擊實驗前半段

接下來影像擷取的追蹤對象，從球棒變成投手的球。在視覺回授的幫助下，投手可以不再定點投球，此時我們可以拿著球往右邊移動，手臂便會握著球棒往右邊移動，調整最佳的打擊位置。同樣的若是投手比較高，手臂也會調整球棒來打擊，此為第一台相機的功能。而第二台相機負責觀察球的運動，當投手投出球之後，運動偵測演算法會偵測到此狀況，即會控制手臂揮棒，完成擊球的運動。以下擷取同一段影片的第 30、35、36、37 秒，來觀察打擊實驗的結果：



圖 42. 球棒與球的打擊實驗後半段



上圖是一次成功的打擊實驗，而如果出棒判斷太慢則會有下面的結果：

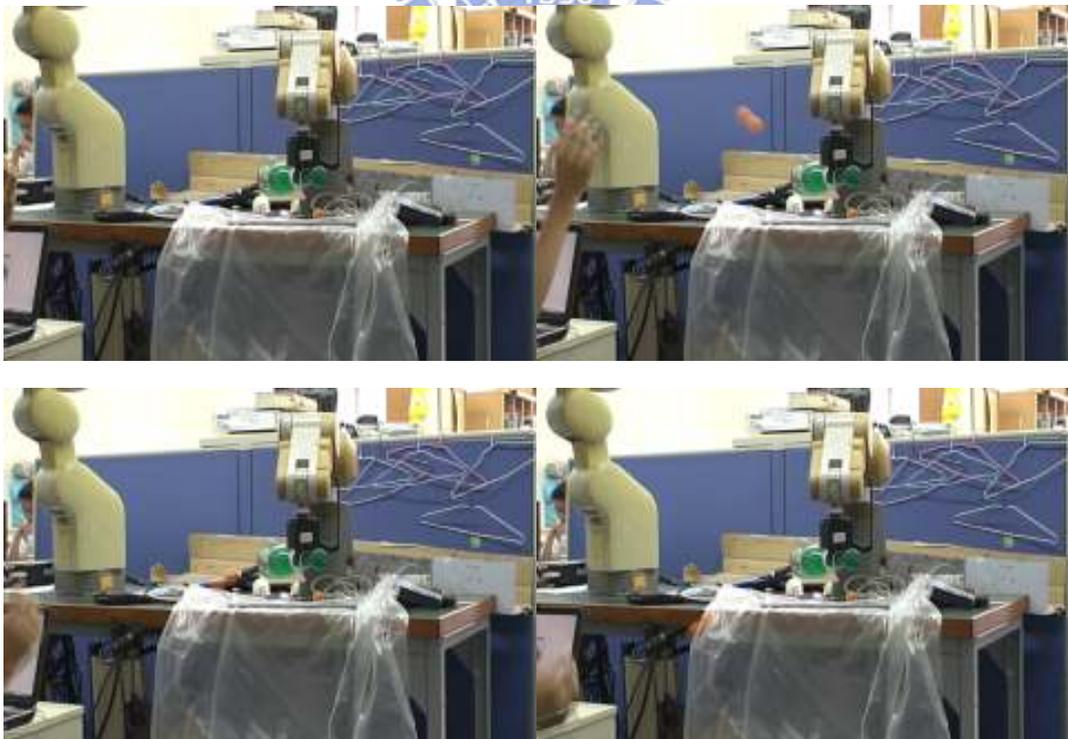


圖 43. 球棒與球的打擊實驗狀況一

上圖是一次失敗的打擊實驗，原因是運動偵測失靈，導致出棒的速度太慢。而還有一個狀況會造成揮棒落空，下圖是因為位置判斷失誤，導致打不到球：

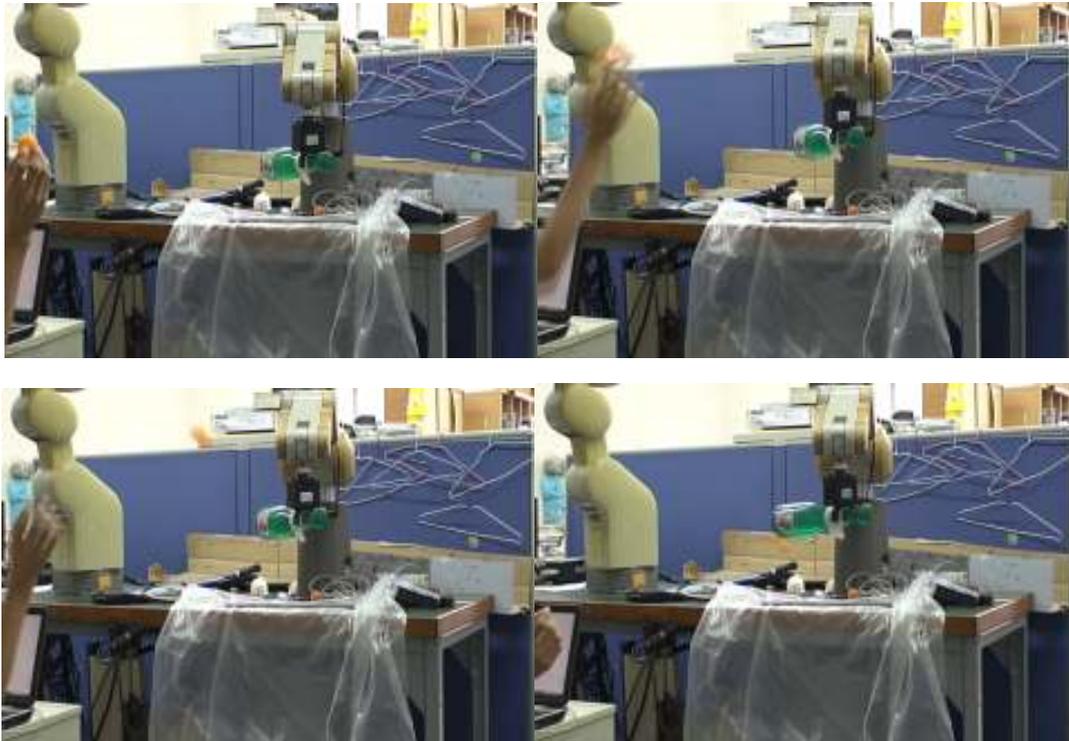


圖 44. 球棒與球的打擊實驗狀況二

上圖也是一次失敗的打擊實驗，原因是球棒依照變化球的軌跡，往上移動，揮棒後才發現跟不上球下墜，而揮棒落空。為了測驗打擊率，我們開始更多實驗。

表格 7. 球棒與球的打擊實驗表

Index	Hit Result	Reasons	Index	Hit Result	Reasons
1	○		21	○	
2	×	A & B	22	○	
3	×	A	23	○	
4	○		24	×	A and B
5	○		25	×	B
6	×	B	26	×	B
7	○		27	×	A and B
8	○		28	×	A
9	○		29	×	A
10	×	A	30	○	

11	○		31	○		
12	×	A	32	×	B	
13	○		33	×	B	
14	×	A	34	×	B	
15	×	A	35	○		
16	○		36	×	A	
17	×	B	37	○		
18	○		38	○		
19	×	A	39	×	B	
20	○		40	×	A	
Reason A: Position Fault			Reason B: Motion Fault		AVG	47.5%

以上實驗分別拍攝成四段影片來觀察，我們發現打擊率沒有想像中高，推測原因可能是運動偵測還不夠敏感，可以藉由調整參數來改善。此參數對偵測結果很重要，一旦參數設定不佳，可能會造成運動偵測錯誤。另外，我們又發現球拍有時會跟不上球的位置，故我們利用時間函數，在上面的程式之中加入顯示圖形運算時間的程式碼，算出處理時間的平均值約為 0.06 秒，表示一秒內能夠處理 15 至 20 張圖形，其中測試的 CPU 規格為 1.66GHz。下表列出其部份的顯示內容：

表格 8. 圖形處理的時間測試表

Index	Time(s.)	4	0.0592	8	0.0584	12	0.0641
1	0.0626	5	0.0612	9	0.0676	13	0.0502
2	0.0493	6	0.0543	10	0.0538	14	0.0602
3	0.0510	7	0.0567	11	0.0556	15	0.0606

相較之下，我們發現機器手臂每秒鐘卻只能接收 5 個命令，故此手臂處理的速度小於影像處理的速度。這個現象表示，即使影像特徵的擷取追蹤雖然夠快，然而手臂卻跟不上此速度。當大幅度變化球出現時，就會揮棒落空。現階段不易解決此問題，除非更改機器手臂的規格，增加每秒鐘能接收的命令數，才能改善。除此之外，此手臂在特殊模式之下才能高速移動，而目前的工作速度，只能做到擊球、追球等等動作，無法滿足其他像是挑球、丟球等等難度比較高的球類運動。

六、結論

在即時的現實環境之中，如何選擇快速、穩定的擷取技術，再搭配運算量少、可靠的回授系統，是本文想要解決的議題，故我們採用了適合的演算法，來實現即時運算。在影像方面，本文以無校正雙相機為架構，解決全域與區域的問題，模擬與實驗證實了平均位移法可以應用在此系統中，在複雜的環境也能有較佳的表現。在回授方面，此系統容易實現，移植性高而且擴充方便。模擬與實驗展示了低運算量的特性，使得反應時間比以往的系統都要短，故能處理更艱難的任務。

另外，在實作上，過去的系統有非常多硬體與軟體的限制，並且要使用許多電腦與配備來處理。本文所提出的架構突破了這些瓶頸，僅用一台筆記型電腦與兩台視訊來完成所有的軟硬體，其中本文仔細研究分析、探討各種方法，並致力於實現開發，改良原本高成本、長時間的研究，變成單電腦少設備的架構。相信此架構將有助於提升這方面的研究，而提升到更多應用、實用的層面。

在未來，我們可以利用 TCP/IP 的網路協定，加速與手臂之間的傳輸。目前手臂使用的 RS-232 通訊介面，其傳輸速率約為 9600 bps 或 115200 bps，而將來的機器手臂基本上都會支援網路傳輸，故未來能利用網路加快電腦端與機器手臂端之間的傳輸，使得手臂能更快的收到命令，進而移動手臂來達成任務。

以上我們討論了系統實驗結果的未來發展。在展示的運動方面，本文選擇了使用球具的運動，在未知球具長度的挑戰下，機器手臂成功的完成用手套接球、用球棒擊球、用球拍挑球等等運動。本文結合了影像處理與手臂控制，影像處理使用了擷取追蹤與運動偵測的技巧，手臂控制善用了機器手臂的介面來完成即時控制。最後，我們利用各種工具模擬了逼真的狀況，並且實現軟硬體各個系統，更在軟體的開發有所貢獻，展示機器手臂的運動與互動，呈現了完整的研究。

七、相關文獻

- [1] S. Hutchinson, G. D. Hager and P. I. Corke, "A tutorial on visual servo control," IEEE Trans. on Robotics and Automation, vol. 12, no.5, pp.651-670, October 1996.
- [2] B. K. P. Horn, Robot Vision, MIT Press, Cambridge, MA, 1986.
- [3] 劉詠龍, 「適應性視覺伺服控制之機械手臂應用」, 國立中央大學, 碩士論文, 民國92年。
- [4] Y. Shirai and H. Inoue, "Guiding a robot by visual feedback in assembling tasks," Pattern Recognition, vol.5, pp.99-108, 1973.
- [5] J. Hill and W. T. Park, "Real time control of a robot with a mobile camera," in Proc. 9th ISIR, pp.233-246, Washington, D.C., 1979.
- [6] A. C. Sanderson and L. E. Weiss, "Image-based visual servo control using relational graph error signals," Proc. IEEE, pp.1074-1077, 1980.
- [7] P. Corke, "Visual control of robot manipulators-A review," in Visual Servoing K.Hashimoto. Ed. Singapore: World Scientific, pp.1-31, 1993.
- [8] R. Y. Tsai, "A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses," IEEE Journal of Robotics and Automation, vol.3, pp. 323-344, August 1987.
- [9] Zhengyou Zhang, "A flexible new technique for camera calibration," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, no.11, pp.1330-1334, 2000.
- [10] W. Jang and Z. Bien, "Feature-based visual servoing of an eye-in-hand robot with improved tracking performance," in Proc. IEEE Int. Conf. on Robotics and Automation, pp.2254-2260, 1991.
- [11] B. Yoshimi and P. K. Allen, "Active, uncalibrated visual servoing," in Proc. IEEE Int. Conf. on Robotics and Automation, pp.156-161, San Diego, CA, May 1994.
- [12] W. Hong, "Robotic Catching and Manipulation Using Active Vision," M.S. Thesis, Department of Mechanical Engineering, MIT, Cambridge, MA, 1995.
- [13] A. Blake and A. Yuille, Active Vision, MIT Press, 1992.
- [14] A. C. Sanderson, L. E. Weiss, and C. P. Neuman, "Dynamic sensor-based control of robots with visual feedback," IEEE Trans. on Robot Automation, vol. RA-3, pp.404-417, October 1987.
- [15] N. P. Papanikolopoulos and P. K. Khosla, "Adaptive robot visual tracking: Theory and experiments," IEEE Trans. Automat. Contr., vol. 38, no.3, pp.429-445, 1993.
- [16] N. P. Papanikolopoulos, P. K. Khosla, and T. Kanade, "Visual tracking of a moving target by a camera mounted on a robot: A combination of vision and control," IEEE Trans. on Robot Automation, vol.9, no. 1, pp.14-35, 1993.
- [17] B. Espiau, F. Chaumette, and P. Rives, "A new approach to visual servoing in robotics," IEEE Trans. Robot Automation, vol.8, pp.313-326, 1992.

- [18] K. Hosoda and M. Asada, "Versatile visual servoing without knowledge of true Jacobian," In IEEE/RSJ/GI International Conference on Intelligent Robots and Systems, pp.186-193, 1994.
- [19] M. Jagersand, "Visual servoing using trust region methods and estimation of the full coupled visual-motor Jacobian," In IASTED Applications of Robotics and Control, 1996.
- [20] N. Papanikolopouliss and P. Khosla, "Robotic visual servoing around a static target: An example of controlled active vision," In Proceedings of the 1992 American Control Conference, pp.1489-1494, 1992.
- [21] R. Sharma, J-Y. Herve, and Peter Cucka, "Dynamic robot manipulation using visual tracking," In Proceedings of the 1992 IEEE International Conference On Robotics and Automation, pp.1844-1849, Nice, France, 1992.
- [22] D. Bennett, D.Geiger, and J. M. Hollerbach, "Autonomous robot calibration for hand-eye coordination," The International Journal of Robotics Research, vol.10, no.5, pp.550-559, October 1991.
- [23] P. Allen, A. Timcenko, B. Yoshimi, and P. Michelman, "Automated tracking and grasping of a moving object with a robotic hand-eye system," IEEE Transactions on Robotics and Automation, vol.2, no.2, pp.152-165, April 1993.
- [24] F. Chaumette and A. Santos, "Tracking a moving object by visual servoing," In IFAC 12th Triennial World Congress, pages 643-648, Sydney, Australia, July 1993.
- [25] J. A. Piepmeier, G. McMurray, and H. Lipkin, "Tracking a moving target with model independent visual servoing: A predictive estimation approach," In IEEE International Conference on Robotics and Automation, Leuven, Belgium, May 1998.
- [26] J. A. Piepmeier, G. V. McMurray, and H. Lipkin, "Uncalibrated dynamic visual servoing," IEEE Trans. on Robot and Automation, vol. 20, no. 1, pp.143-147, Jan. 2004.
- [27] N. Hollinghurst and R. Cipolla, "Uncalibrated stereo hand-eye coordination," Image and Vision Computing, vol.12, no.3, pp.187-192, 1994.
- [28] G. D. Hager, W.-C. Chang, and A. S. Morse, "Robot hand-eye coordination based on stereo vision," IEEE Control System Mag., vol.15, pp.30-39, Feb 1995.
- [29] F. Chaumette, P. Rives, and B. Espiau, "Positioning of a robot with respect to an object, tracking it, and estimating its velocity by visual servoing," in IEEE International Conference on Robotics and Automation, pp.2248-2253, April 1991.
- [30] Jianbo Su, Yanjun Zhang and Zhiwei Luo, "Online estimation of Image Jacobian Matrix for uncalibrated dynamic hand-eye coordination," Int. J. Systems, Control and Communications, vol.1, no.1, pp.31-52, 2008.
- [31] T. T. M. Asada and K. Hosoda, "Visual tracking of unknown moving object by adaptive binocular visual servoing," in IEEE International Conference on Multi-sensor Fusion and Integration for Intelligent System, pp.249-254, Taipei, Taiwan, August 1999.

- [32] J. A. Piepmeier, "Experimental results for uncalibrated eye-in-hand visual servoing," Proceedings of the 35th Southeastern Symposium on System Theory, pp.335-339, Morgantown, 2003.
- [33] S. W. Wijesoma, D. F. H. Wolfe and R. J. Richards, "Eye-to-Hand Coordination for vision guided Robot Control Applications," Int. Journal of Robotics Research, vol.12, no.1, 1993.
- [34] W. Z. Chen, U. A. Korde, and S. B. Skaar, "Position Control Experiments Using Vision," Int. Journal of Robotics Research, vol.13, no.3, pp.199-208, 1994.
- [35] M. Jagersand, O. Fuentes, and R. Nelson, "Experimental evaluation of uncalibrated visual servoing for precision manipulation," in IEEE International Conf. Robotics and Automation, vol.4, pp.2874-2880, April 1997.
- [36] A. M. Farahmand, A. Shademan, and M. Jagersand, "Global visual-motor estimation for uncalibrated visual servoing," in Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems, San Diego, CA, USA, October 2007.
- [37] S. Leonard and M. Jagersand, "On with the visuomotor function: a 6DOF adaptive approach for modeling image-based variations and visual servoing," in 2007 IEEE International Conference on Robotics and Automation, Roma, Italy, April 2007.
- [38] E. Marchand and G. D. Hager, "Dynamic sensor planning in visual servoing," in IEEE Int. Conf. on Robotics and Automation, vol.3, pp.1988-1993, Belgium, May 1998.
- [39] J. T. Feddema, C. S. G. Lee and O.R. Mitchell, "Model-based visual feedback control for a hand-eye coordinated robotic system," in IEEE Computer, vol.25, pp.21-33, 1992.
- [40] G. Flandin, F. Chaumette and E. Marchand, "Eye-in-hand/Eye-to-hand cooperation for visual servoing," in IEEE Int. Conf. Robotics and Automation, San Francisco, 2000.
- [41] D. Comaniciu, V. Ramesh, and P. Meer, "Kernel-based object tracking," IEEE Trans. on Pattern Analysis and Machine Intelligence, vol.25, no.5, pp.564-577, May 2003.
- [42] S. Birchfield and S. Rangarajan, "Spatiograms versus histograms for region-based tracking," Proc. IEEE Conf. on Computer Vision and Pattern Recognition, 2005.
- [43] C. Yang, R. Duraiswami, and L. Davis, "Efficient mean-shift tracking via a new similarity measure," IEEE Conf. on Computer Vision and Pattern Recognition, vol. 1, pp.176-183, 2005.
- [44] 阮崇維, 「使用空間與顏色特徵的平均移動演算法於物件大小與方位追蹤」, 國立交通大學, 碩士論文, 民國 96 年。
- [45] 黃啟揚, 「以雙核心平台實現即時影音追蹤與語音純化系統」, 國立交通大學, 碩士論文, 民國 97 年。
- [46] P.I. Corke, "A Robotics Toolbox for MATLAB", IEEE Robotics and Automation Magazine, Volume 3(1), March 1996, pp. 24-32.