

國立交通大學

電機學院 電機與控制學程

碩 士 論 文

嵌入式路口行人偵測系統設計與實現

The design and implementation of an embedded
pedestrian detection in intersection

研 究 生：劉光仁

指導教授：吳炳飛 教授

中 華 民 國 九 十 七 年 九 月

嵌入式路口行人偵測系統設計與實現

The design and implementation of an embedded pedestrian detection in intersection

研 究 生：劉光仁

Student : Kuang-Jen Liu

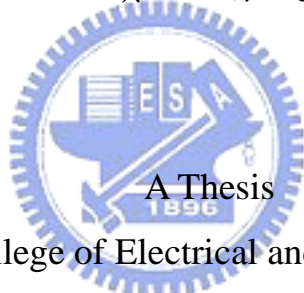
指 導 教 授：吳炳飛

Advisor : Dr. Bing-Fei Wu

國 立 交 通 大 學

電機學院 電機與控制學程

碩 士 論 文



Submitted to College of Electrical and Computer Engineering

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master of Science

In

Electrical and Control Engineering

Sep 2008

Hsinchu, Taiwan, Republic of China

中華民國九十七年九月

嵌入式路口行人偵測系統設計與實現

學生：劉光仁

指導教授：吳炳飛

國立交通大學 電機學院 電機與控制學程碩士班

摘 要

目前看到大部份的影像處理的系統，大多是用個人電腦的平台為基礎架構，透過攝影機及影像擷取卡，將影像傳入個人電腦，再加入所要的運算功能，達到所需的功能。這樣的系統是相當龐大及複雜的，且個人電腦體積龐大，不適合架設在各種監視的地方。個人電腦成本也相對的高，一旦裝設多組的監控系統所要花費的成本也相當可觀。為了改善這些因素，本論文利用德州儀器生產的數位信號處理 TMS320DM642 DSP 開發出一組體積小、成本低廉的嵌入式系統，將影像處理的功能在 DSP 平台上實現。開發階段從電路設計、元件選擇、佈局及走線，組裝測試到驗證完成。並利用 DSP 內部的 EDMA 功能和最佳化程式的方法來加速處理影像。在行人偵測上，首先利用背景相減法取出行人移動的區塊，並且用膨脹及侵蝕影像處理技術來處理影像，最後使用角點和邊緣的判斷，來判別是否為行人。在行人追蹤上，建立移動行人的直方圖特徵，利用直方圖比較法及向量預測來追蹤物體，並預測物體的移動方向。處理完畢的影像，透過 DSP 將畫面輸出到監控螢幕上，可以直接看到移動物體即時偵測與追蹤結果。最後實現這一套完整的行人偵測系統。

The design and implementation of an embedded pedestrian detection in intersection

Student : Kuang-Jen Liu

Advisors : Dr. Bing-Fei Wu

Degree Program of Electrical and Computer Engineering

National Chiao Tung University

ABSTRACT



Most of current digital image processing systems are built on camcorders and personal computers equipped with video capture cards. Required work is achieved by using the computers to do some mathematics calculations on video files captured from the camcorders. However, such systems can be very tremendous and complicated. For example, those computers are too large to fit in all kinds of environments which need to be monitored. Besides, the cost of multiple monitor systems is increased with the number of the computers of which each has high expenditure.

In order to solve these drawbacks, this thesis develops a set of prototype, which has advantages of tiny size and cheap cost, by using TMS320DM642 DSP of Texas Instruments Incorporated. It not only realizes the functionality of digital image processing from the DSP, but also makes use of internal EDMA of the DSP and utilizes programs to

accelerate the process.

The developing stages starts from circuit design through component selection, layout, trace, and package testing, to verification.

The first topic is pedestrian detection. Background extraction helps find out pedestrian moving blocks. Morphological operator, such as dilation and erosion, are also used to process video images. Whether an object is a pedestrian or not is determined by characteristics of the detected corner and edge.

The second topic is pedestrian tracking. A histogram of moving pedestrian is created at first. By means of the histogram comparative method and prediction by motion vector, the pedestrian can be tracked and its moving direction can also be forecasted.

Processed images are exported to the screens of monitor systems by the DSP, which the real-time pedestrian detection and tracking results are displayed. Finally, the embedded pedestrian detection system in intersection is completed.

誌 謝

首先，要感謝的就是我的指導教授－吳炳飛博士。在我人生的過程中，再一次拿起課本學習新知，在學校看到老師對學術研究的熱忱，以及對老師將學術研究與實際應用結合的不遺餘力，深感佩服及尊敬。也感謝老師能讓我進入到這個實驗室之中，努力學習到不同的技術領域，不論是在系統整合，程式撰寫，影像處理演算法等等，這些都讓我獲益良多，也充實了不少。

再來要感謝實驗室的阿誠(陳昭榮)學長、陳志忠學長。謝謝他們在研究之中對於影像處理及程式撰寫上給予很多協助及指導。從系統整合開始，提出許多有效的建議及想法，那邊可以縮小那邊可以功能減少，或是那些功能要能夠使用，讓我可以能夠成功的實現整個系統平台。再來是行人偵測上，感謝他們協助去架設交通路口攝影機，拍攝路口行人的影片可以做為研究分析使用，並且在模擬實驗上，給與許多寶貴的經驗以及不一樣的啟發思維，讓我可以每次遇到瓶頸的時候，給予最有效的幫助。

最後感謝家人的精神支持，感謝老婆，淑君，在我每天很忙碌的時候，可以將家中打點的很好，讓我在每天下完班之後能夠專心的繼續做研究。再來就是感謝我那個可愛的怡萱及奕廷，謝謝你們有時能夠懂事知道，爸爸忙於工作及學業，無暇陪你們，這個碩士是獻給你們的。希望你們以後長大也可以努力向上，伴演好自己的角色。

目錄

第一章 緒論.....	1
1.1 研究動機與目的.....	1
1.2 相關研究.....	2
1.3 論文架構.....	3
第二章 系統架構.....	4
2.1 硬體介紹.....	4
2.2 軟體架構.....	6
第三章 DSP 系統設計與線路規劃	9
3.1 設計目的.....	9
3.2 線路設計流程.....	9
3.2.1 功能定義.....	10
3.2.2 電路板規格.....	11
3.3 電路設計.....	14
3.4 佈局及走線.....	15
3.5 重要元件特性.....	16
3.5.1 TMS320DM642 DSP	16
3.5.2 MEMORY	17
3.5.3 CPLD	19
3.5.4 TV Decoder and Encoder.....	19
3.5.5 電源系統架構.....	20
第四章 行人偵測及追蹤.....	21
4.1 行人偵測.....	21
4.1.1 背景收斂.....	23
4.1.2 移動物件擷取.....	24
4.1.3 雜訊濾除 Noise Filter.....	25

4.1.4	邊緣強化 Edge Enhance.....	26
4.1.5	膨脹 Dilation	27
4.1.6	侵蝕 Erosion	28
4.1.7	角點偵測 Corner Detection.....	29
4.1.8	水平邊緣 Horizontal Edge	32
4.1.9	人頭偵測 Head Detection.....	33
4.2	行人追蹤.....	35
4.2.1	建立追蹤物件.....	35
4.2.2	行人判斷.....	38
4.2.3	向量預測.....	40
4.2.4	直方圖比對法.....	41
4.3	最佳化方法.....	43
4.3.1	記憶體規劃.....	44
4.3.2	EDMA 的加速.....	45
4.3.3	程式 Coding Style 的改善.....	47
第五章 實驗結果.....		50
5.1	實作成果.....	50
5.1.1	頭部角點偵測.....	50
5.1.2	行人追蹤預測.....	52
5.1.3	直方圖比較法.....	53
5.1.4	行人重疊.....	55
5.2	系統效率分析.....	58
5.2.1	偵測率及正確率.....	58
5.2.2	平台運作速度.....	58
5.3	錯誤分析.....	60

5.4	實驗結果分析.....	62
5.4.1	邊緣強化實驗.....	62
5.4.2	自適應閥值算法實驗.....	64
第六章	結論.....	68
6.1	總結.....	68
6.2	未來展望.....	69
文獻參考	70



圖目錄

圖 2-1 交通路口示意圖	4
圖 2-2 硬體設備連接裝置	5
圖 2-3 EVB 架構圖	6
圖 2-4 PC 模擬畫面	7
圖 2-5 系統執行流程圖	8
圖 3-1 線路設計流程	10
圖 3-2 模組高度示意圖	11
圖 3-3 實際 TMS320DM642 模組尺寸	12
圖 3-4 主板、電源板與連接板(一) 正面圖	13
圖 3-5 主板、電源板與連接板(二) 背面圖	13
圖 3-6 預留硬體線路介面圖	14
圖 3-7 ORCAD HIERARCHY	14
圖 3-8 電路板內層圖	16
圖 3-9 TMS320DM642 內部架構	17
圖 3-10 CLOCK BUFFER 與 SDRAM	18
圖 3-11 TMS320DM642 內部 SRAM 規劃圖	19
圖 3-12 VIDEO 輸入及輸出	20
圖 3-13 模組電源規劃架構	20
圖 4-1 行人候選人偵測流程	22
圖 4-2 實際路口圖	22
圖 4-3 背景收斂示意圖	24
圖 4-4 移動物體收斂圖	24
圖 4-5 背景相減圖	25
圖 4-6 背景相減二值化圖	25

圖 4-7 5x5 NOISE FILTER MATRIX	26
圖 4-8 經過雜訊濾除處理之後	26
圖 4-9 邊緣強化表示之像素	27
圖 4-10 邊緣強化處理	27
圖 4-11 侵蝕與膨脹 MATRIX	28
圖 4-12 經過膨脹的結果	28
圖 4-13 經過侵蝕的結果	29
圖 4-14 角點 TYPE 1 的種類	30
圖 4-15 角點 TYPE 1 的位置	30
圖 4-16 角點 TYPE 2 的種類	30
圖 4-17 角點 TYPE 2 的位置	31
圖 4-18 角點 TYPE 3 的種類	31
圖 4-19 角點 TYPE3 的位置	31
圖 4-20 角點 TYPE 4 的種類	32
圖 4-21 角點 TYPE4 的位置	32
圖 4-22 SOBEL MATRIX	33
圖 4-23 水平邊緣處理之後的圖	33
圖 4-24 行人候選人之圖	35
圖 4-25 建立追蹤物件流程	37
圖 4-26 行人物體重疊示意圖	38
圖 4-27 重疊區域面積表示圖	38
圖 4-28 行人行逕預測圖	40
圖 4-29 周圍搜尋比對法	41
圖 4-30 特徵圖	42
圖 4-31 移動特徵直方圖	43

圖 4-32 原始特徵直方圖	43
圖 4-33 實際規劃分配記憶體位置	45
圖 4-34 DMA 的處理流程	46
圖 4-35 CCSTDIO 的設定	47
圖 4-36 CODING 範例(1)	48
圖 4-37 CODING 範例(2)	49
圖 4-38 CODING 範例(3)	49
圖 5-1 角點偵測判斷正確圖	52
圖 5-2 正確追蹤畫面	53
圖 5-3 使用 PATTERN MATCHING 方法	54
圖 5-4 直方圖比較法	55
圖 5-5 行人重疊全區域比較法	57
圖 5-6 行人重疊部份區域比較法	57
圖 5-7 角點偵測多處判斷錯誤圖	60
圖 5-8 錯誤追蹤圖	62
圖 5-9 邊緣強化比較圖	64
圖 5-10 自適應閾值統計圖	64
圖 5-11 自適應閾值趨勢圖	65
圖 5-12 使用固定閾值和自適應閾值前景圖	66

表目錄

表 5-1 實驗人數統計	58
表 5-2 偵測率與正確率	58
表 5-3 TMS320DM642 一張畫面處理時間	59
表 5-4 自適應閾值設定表	65
表 5-5 閾值比較結果	67



第一章 緒論

1.1 研究動機與目的

交通安全一向都是人們非常重視的問題，不論是在路口還是一般道路上，安全第一是人人所想要的。路口行人的安全也是在交通安全中很重要的課題，行人比機車或是汽車脆弱許多，一旦發生了意外碰撞的話，行人容易造成很嚴重的受傷。路口行人的安全是不能夠忽視的，行人穿越路口除了經過天橋、地下道之外，最常經過的就是行人穿越道。有行人穿越道的地方就表示會有車子通過的地方。在交通安全上，行人穿越道是為了保障行人穿越的權利與安全，一旦車輛未禮讓行經路口行人，便會威脅行人安全。車讓人及行人優先的路權觀念，是為了維護路口行人的安全。相反的不守規矩的行人也是造成自己生命安全的危險。當遇上老弱婦孺或是身體不方便的人過馬路時，走在行人穿越道中間時發現已經紅燈的時候，進退兩難。如果有行人偵測系統來發現這樣的情況，延遲綠燈的時間，讓這些行人可以繼續的安全通過，可以減少不必要的危險發生。一旦發生有不守法的行人要闖紅燈，系統可以即時偵測到並發出語音警告，嚇阻違規的行人，避免因為這樣不守法產生出來的交通意外。發展這樣路口即時行人偵測系統，是一項很重要的課題，達到保護行人安全重要目的。

大部份發展行人偵測系統都是使用電腦來當作平台，利用電腦強大的運算能力及儲存能力來進行各項監控。由於這些以電腦為基礎的系統，體積過於龐大，消耗功率也很大，整體的系統費用也相當昂貴，一旦要廣範應用在實際生活上每一個角落，還是會有很大的落差。除了經費就是一個很大的問題之外，體積也是一個相對的問題，較難達到理想的安全監視系統。理想的系統是要能夠有速度快、體積小、成本低、功能強大具有獨立判斷能力的智慧型系統。於是本論文提出一個完整的解決方案並且去實現它，利用德州儀器 TMS320DM642 數位訊號處理器強大的運算功能、節省高成

本的應用系統、縮小整體系統的體積以及發展即時辨識的偵測系統。針對交通路口上，對於過往行走的行人影像給與偵測及辨識。在整個解決方案，設計出一套實用的數位訊號處理器公板，大小為 8 cm x 6.5 cm x 2.8 cm 的模組，這樣的公板大小比一般的皮夾還小，成本也是遠低於一般電腦的費用，適合方便架設於各種場所。並發展即時辨識的演算法，以實際的路口影像測試使用本論文設計的嵌入式系統，務求達到高辨識的正確性，實現整個行人偵測系統。

1.2 相關研究

嵌入式路口行人偵測系統設計與實現，主要分做兩個主軸，一個是實際的將系統電路設計完成，另一個方面就是行人偵測。在行人偵測系統主要有移動偵測、行人辨識、軌跡追蹤三個主要的工作。一般做移動物體的偵測可以分為背景相減法、連續影像相減及立體視覺方法。本論文所使用的是背景相減法[11,12]，主要是先要能收斂出一張背景圖，利用目前的畫面與背景圖互相減之後，取到移動的物體。在行人辨識的方面，有先將交通環境的物體先分為數類，如汽車，機車及行人等，將各種物體特徵擷取出來並加以歸納分類，然後在利用類神經網路(Neural Network)[4,5,6]訓導，記憶在網路內。大部份的論文都提出樣板比對(Template Matching)[7]方法找出行人特徵，或並用支援向量機 (Support Vector Machine, SVM)[1,2,3]，分類器來精確偵測出行人，並只需要單張的影像就可完成。另外有人提出以行人走路的節奏週期性，以及時間頻率的特性當作影像辨識特徵如文獻[8,9,10]。我們分析這些優點及缺點，發現很多演算法實際在數位訊號處理器上執行的話，無法達到即時的效果。於是我們提出利用一些基本的影像處理方法，侵蝕及膨脹法[14]，*Sobel* 邊緣[13]等，先作基本的處理。再來使用角點偵測[15,16,17]的想法，來判斷行人頭部的邊邊角角，證明是一個行人頭部的形體。再加一些判斷來偵測到的是一個行人，完成行人辨識。至於軌跡追蹤，使用簡單的向量預測及直方圖[25]比較法來偵測行人新的位置所在。完成整個行人偵測的架構想法

在硬體線路設計中，主要由原本德州儀器所提供的參考線路[20]，依照自己定義的規格去重新設計符合需求，並可在線路上預留未來擴充性的設計。在規劃系統的記憶體使用參考文獻[21][23]，將 DSP 內部的記憶體及外部的記憶體的容量，配合著程式流程的架構，規劃使用的區域。相對於視訊信號在數位訊號處理器輸入輸出設定方式，也要參考文獻[22]。一切硬體規劃完畢後，將所發展的演算法程式，放入到數位訊號處理器平台上，最佳化的工作也是很重要的，文獻[24]內提供一些，德儀數位訊號處理器如何去寫最佳化的方式，例 Direct Memory Access(DMA)的快速存取或是乒乓(Ping-Pong)Buffer 方法的使用。能將行人偵測演算法，放到數位訊號處理器上去實現如文獻[18][19]，將理論與實際結合在一起。

本論文完成一個結合理論與實務的研究，規劃一個實用性高、功能性強的系統平台，從線路設計、元件選定、佈局及走線、電路板製作、驗證平台到最後的演算法實現，完成一套行人偵測系統的方案。

1.3 論文架構



本論文內容主要分為六章：其內容的架構如下所敘述。第一章介紹論文的研究動機、過去相關研究與本論文架構。第二章介紹系統的架構，分成硬體線路的部份以及軟體程式的架構。第三章 DSP 系統設計與線路規劃，描述整體平台是如何從規劃、設計到完成。第四章行人偵測及追蹤，介紹行人的判別及追蹤的方法。第五章為實驗結果與討論。第六章則是對論文提出結論以及未來研究方向的討論。

第二章 系統架構

本論文發展之目標為建立一套完整的數位訊號處理器 (Digital Signal Processor, DSP) 的系統解決方案，包含硬體，軟體雙方面的發展。主要應用於交通路口上的行人偵測，如圖 2-1 所示。在紅綠燈的燈架上，架設所需要使用的攝影機及系統板，監視著對面路口行人穿越道上過往的行人。利用數位訊號處理器來實現路口行人偵測及追蹤系統，改善一般的偵測系統都使用個人電腦為基礎的架構，無法獨立作業的情況，且系統成本過高、體積過於龐大、安裝複雜的情況發生。於是本論文發展出一套成本低、體積小、功能強、安裝簡單，適合架設於各種地方的嵌入式模組。並利用數位訊號處理器強大的運算能力，研究出執行速度快及效率高、偵測效果好的軟體架構。完成一套真正可以完全應用於行人偵測的系統模組，為本論文最大的重點。

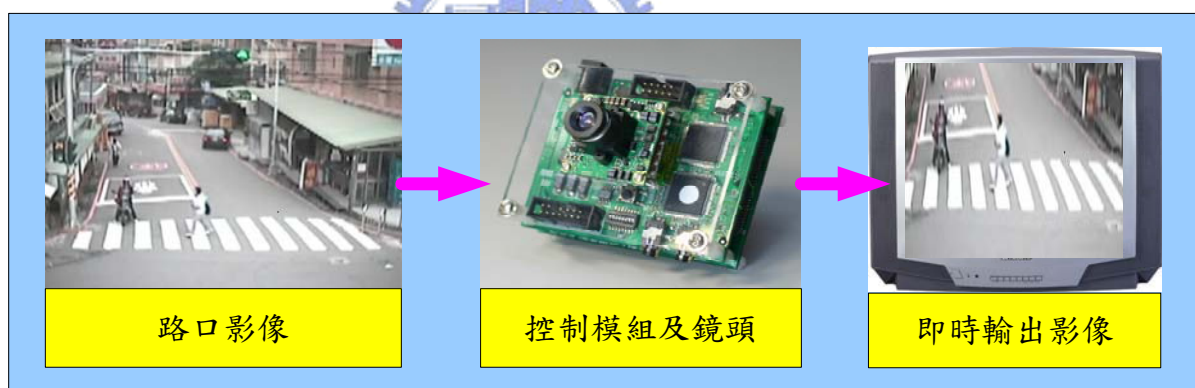


圖 2-1 交通路口示意圖

2.1 硬體介紹

本論文所有用的監控系統，是透過單支的彩色的攝影機，搭配 TI 德州儀器所出的 TMS320DM642 數位訊號處理器，以每秒 30 張畫面的速度及 CIF (352x288) 的影像來進行研究。如圖 2-2 所示為實際連接裝置圖，輸入視訊及音源有兩個通道，輸

出視訊及音源有一個通道，JTAG 的插槽，GPIO 的插槽，UART 的连接座，並有 Reset 的按鈕及指撥開關可以供系統設定。

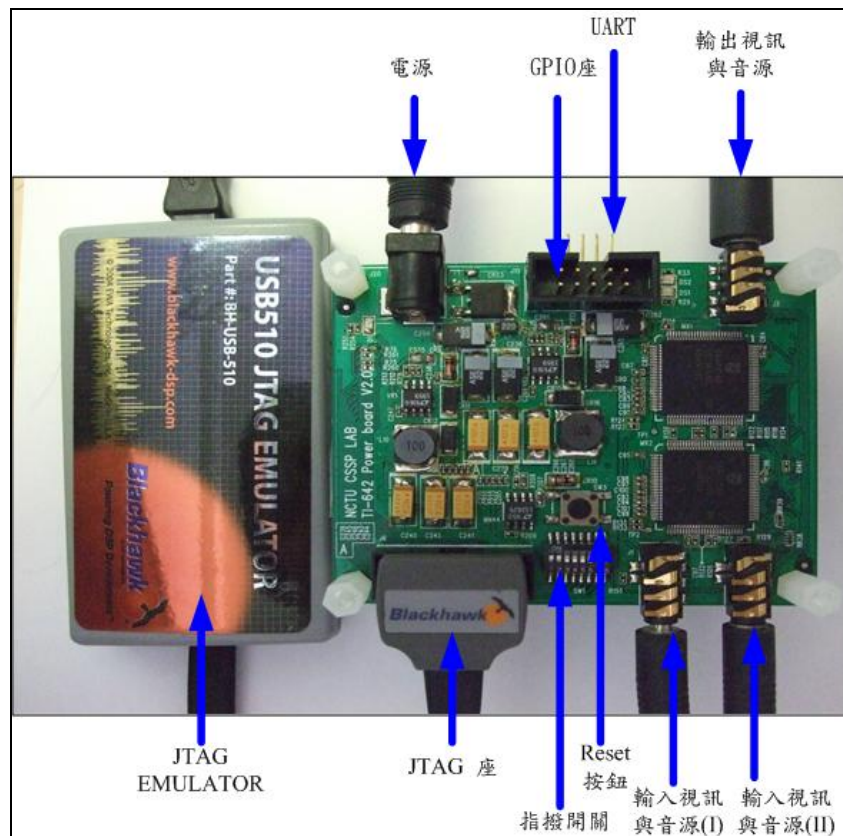


圖 2-2 硬體設備連接裝置

在評估模組板(Evaluation Board, EVB)線路中，如圖 2-3 所示，TMS320DM642 DSP 為主體的處理器，它有三個 Video Ports 可以做視訊輸入及輸出。在視訊輸入端可接兩個解碼器，視訊輸出端接上一個視訊的編碼器。在外部記憶體方面，透過外部記憶體介面(External Memory Interface ,EMIF)與的記憶體元件連接，有 64 bit 的 32MBytes SDRAM 外部記憶空間，並有 4 MBytes 的可燒錄儲存程式碼的快閃記憶體。複雜可編程邏輯器件(Complex Programmable Logic Device, CPLD)是提供系統中更多的訊號控制，協助位訊號處理器去掌握其它的周邊元件的狀況，並回報給數位訊號處理器。雙埠通用非同步收發機(Dual Port UART, DUART)元件，將系統的訊息利用 UART 傳出到電腦或是其它的周邊。

指撥開關(DIPSW)的功能是要設定位訊號處理器的初始狀態使用。JTAG 是系統

模擬做重要的地方，利用它可以模擬軟體的情況，加速完成系統的驗證及發展。剩下最重要的地方是電源轉換(Power Regulation)的地方，它必須將外部輸入的直流電壓，轉換出內部所有元件所需求的直流電壓值，及能提供每一個元件所需的電流量，達到系統的電源的穩定。

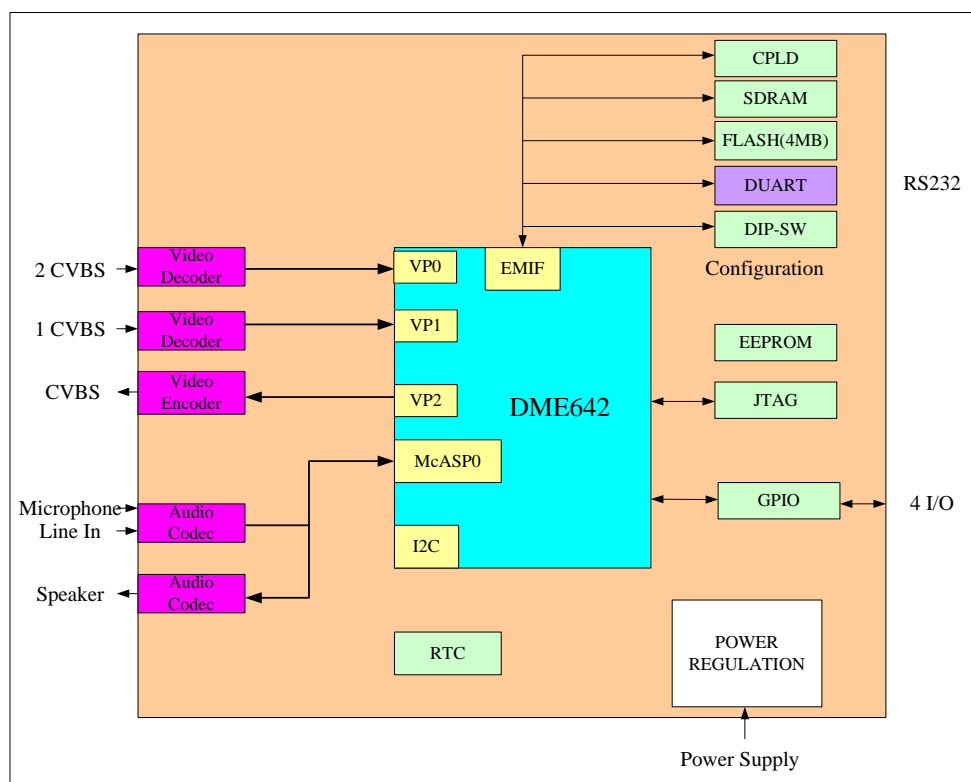


圖 2-3 EVB 架構圖

2.2 軟體架構

發展初期利用電腦平台來模擬系統的，將影像轉換成點陣圖(Bitmap, BMP)格式，利用電腦上的 Visual C 的開發環境下來發展演算法，透過 C 程式將影像的 BMP 檔圖檔，讀到程式中將我們開發的程式結合，並在電腦螢幕上顯示出處理的結果，以符合我們的需求，如圖 2-4。



圖 2-4 PC 模擬畫面

依照整個各個影像處理的技術，整個系統的軟體執行流程如圖 2-5，首先要先做背景的收斂，取出背景圖之後，然後在將新進的畫面與背景相減取出移動的物件。再將畫面做一些後續的影像處理例如：雜訊去除，邊緣強化，侵蝕與膨脹等。接下來作角點偵測及物體判斷，最後就是物體的追蹤，細節將於後面章節介紹。

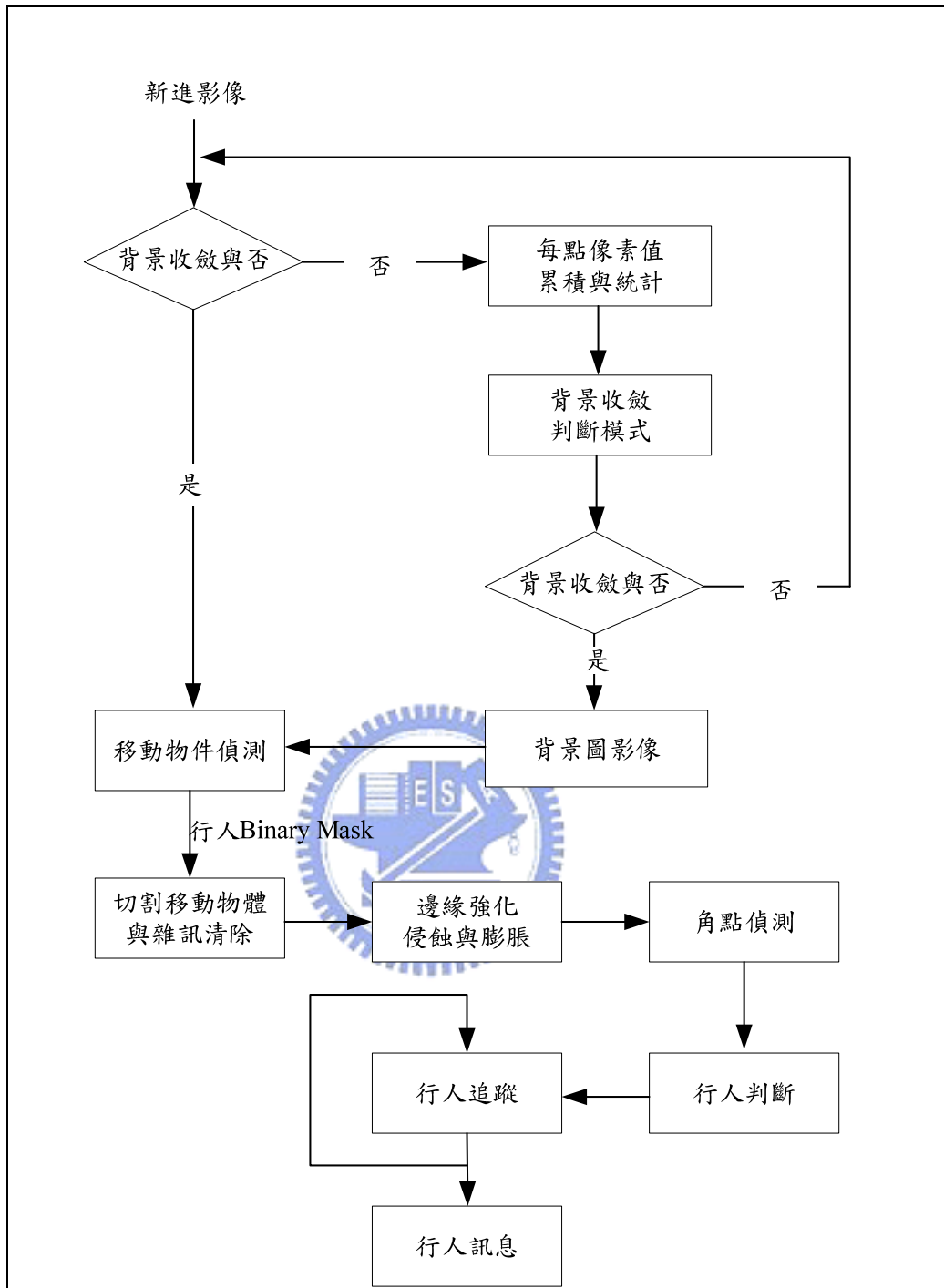


圖 2-5 系統執行流程圖

第三章 DSP 系統設計與線路規劃

3.1 設計目的

研究初期使用跟 ATEME 公司買的數位訊號處理器 DMEK642 系統實驗板，板子的面積為 30cm X 10cm X 8cm. 對於架設在車上的前後監視系統或是一般道路路口攝影機的位置，都嫌過大，無助於提升整體應用的功能性。且公板上的所有功能，也不完全會被拿來廣泛使用。針對以上兩個主要的想法，於是就思考要如何將公板的板子面積縮小，功能規格達到符合影像處理應用的需求。

3.2 線路設計流程

在線路設計開始之前，針對以下幾項步驟循序漸進的評估及考量，如圖 3-1。首先最重要的是定義清楚這套模組所要的功能及優點，再定義出所要呈現出來的外觀尺寸。整個定義清楚之後，接著尋找所需要的電子元件及整體的電路設計就要開始同時並行及完成。進入到佈局及走線的階段，也就是電路板(Print Circuit Board, PCB)上的零件擺放及訊號線的走向，此時是比較煩雜的時候，需特別注意特殊信號的走線及位置，如類比信號例如：視訊信號、高速頻率的信號例如：SDRAM Clock 等等，這些是需要特別小心的。接下來就是將相關檔案送到電路板廠來製作所需要的電路板，等完成之後就開始焊接及組裝。最後就是測試及驗證這套系統，完成製作符合本系統需求的 TMS320DM642 平台。在繪製電路上，我們使用 ORCAD 的軟體來幫我們，繪出這套系統的線路。在佈局及走線上，使用 POWER-PCB 軟體來製作電路板中，實際線路的走法及零件位置，再利用 POWER-PCB 轉換出電路板廠需要的 GERBER 檔。

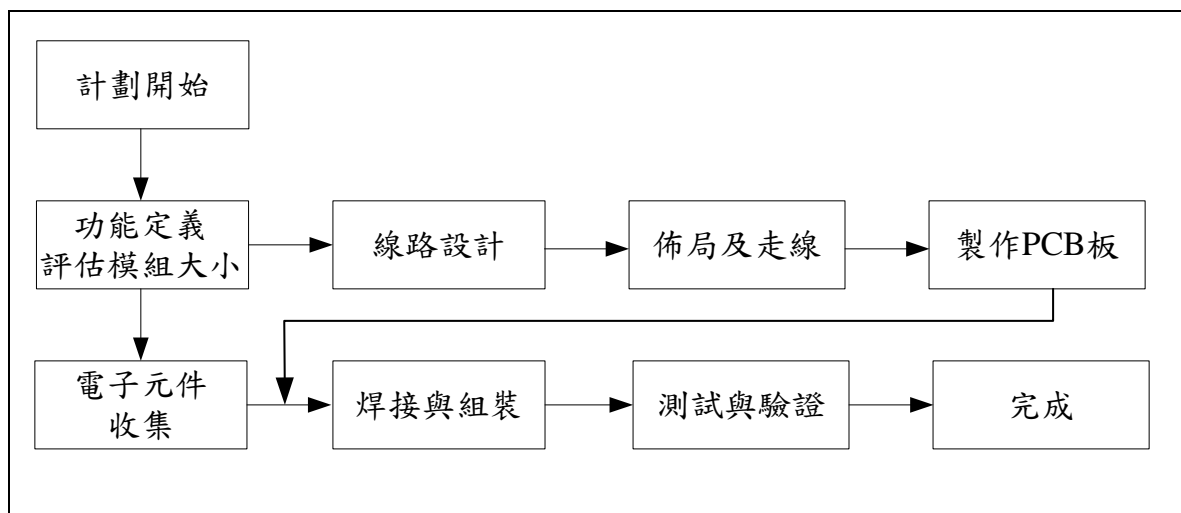


圖 3-1 線路設計流程

3.2.1 功能定義

針對製作 DSP 平台，討論及規劃出會應用的功能，捨去不會使用的功能線路部份。達到符合屬於研究室使用的平台。以下是所列為定義的規格：

- 內建 2 組 類比影像擷取通道(NTSC or PAL)
- 內建 1 組 類比影像輸出(NTSC or PAL)
- 兩組通用型非同步串列埠(RS-232)與電腦聯接或是平台對接
- 提供立體聲及數位聲道的輸出及輸入
- 4MB 快閃記憶體 NOR-Flash Memory 與 32MB 的同步動態記憶體
(Synchronous Dynamic RAM, SDRAM)
- 通用輸入/輸出埠(General Purpose I/O, GPIO)
- 電路測試介面(Joint Test Action Group, JTAG) 的介面
- 直流 12V 的輸入，可使用在汽車電池的供應
- 預留擴充性的介面

NTSC 及 PAL 是彩色電視的廣播標準。NTSC 制式又簡稱為 N 制，是 1952 年 12 月由美國國家電視標準委員會 (National Television System Committee，縮寫為

NTSC) 制定的彩色電視廣播標準，台灣也是使用 NTSC 制。至於 PAL，是一逐行倒相視訊輸出標準(Phase Alternation Line)，為德國人 *Walter Bruch* 於 1967 年提出，為廣播電視中色彩編碼的一種方法，指每行掃描線的彩色訊號會跟上一行倒相，以自動改正在傳播中可能出現的錯相；其標準用於歐洲、香港及中東一帶。

3.2.2 電路板規格

在電路板的整體規劃上，遇到幾個重要考慮的因素在思考：

- 考量實驗室經費不足，只有能投入 10pcs 成本考量，不容許失敗率太高的情形發生。
- 達到能夠在既定的日期完成驗證，不消耗太多的人力及時間。
- 完成規格上的功能使用，以及預留硬體功能擴充性的機會。
- 達到 PCB Size 小於一般皮夾的面積。

於是我們的想法是將 TMS320DM642 系統分作兩個電路板，結合成一套 TMS320DM642 平台。上面的板子稱做為電源板(Power Board)，下面的板子稱做為主板(Main Board)。上下板子之間用 2 個 60pin 1.27mm pitch 的连接器連接如圖 3-2. 所示 電路板面積為 8 cm x 6.5 cm 如圖 3-3，兩板子之間的上下距離為 1cm。

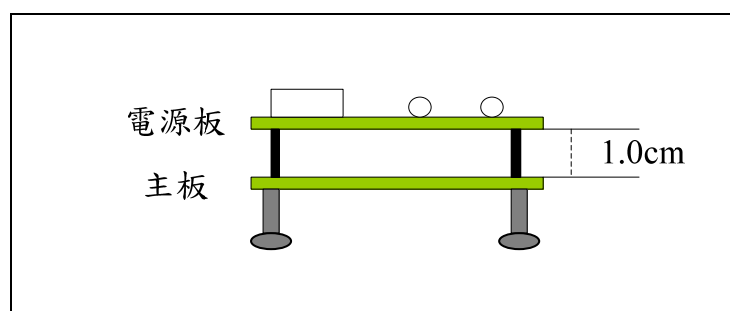


圖 3-2 模組高度示意圖

為了達到電路板尺寸越小越好，於是我們在幾個元件上的使用有特別的想法：一般電阻電容被動元件，採用手機業者都在使用的 0402(電阻尺寸規格)，也就是一顆電阻面積為 0.04 inch x 0.02 inch，換成公制 1.0 mm x 0.5 mm。視訊輸入及輸出部份採

用表面粘著元件(Surface Mount Device, SMD)的耳機座(Phone Jack)，此為 3.5 Φ 的口徑，擁有三段的訊號使用，一段為 Video 使用，另外兩段為 Audio 使用。在 Reset Switch、指撥開關、LED 及 兩個板之間的聯接器也採用了 SMD 的元件，增加電路板內層訊號線，拉線空間。

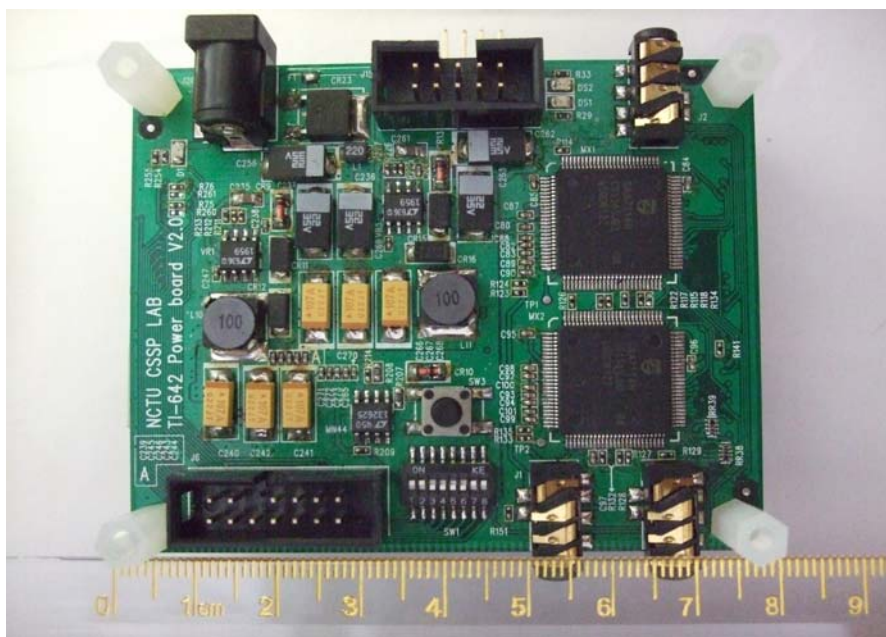


圖 3-3 實際 TMS320DM642 模組尺寸

將電路分為主板及電源板有幾個原因如下：

- (1) 擔心電壓轉換電路(DC/DC)錯誤的電壓輸出，造成在驗證時燒毀內部元件，於是希望能有所隔離，避免這樣的情況導致良率太低。並將電源板規劃為 4 層電路板，主板規劃為 6 層的電路板。
- (2) 分成兩片板子可以同時的進行驗證，達到縮短驗證所消耗的時間。並另外製作連接板，連接電源板及主板，方便驗證。如圖 3-4 與圖 3-5 所示



圖 3-4 主板、電源板與連接板(一) 正面圖



圖 3-5 主板、電源板與連接板(二) 背面圖

- (3) 在系統板子預留了兩組連接器的埠，當未來有需求其它功能時(例如：網路 Ethernet)，只需要製作其功能的板子，再與這兩片結合就可以迅速完成如圖 3-6 所示。

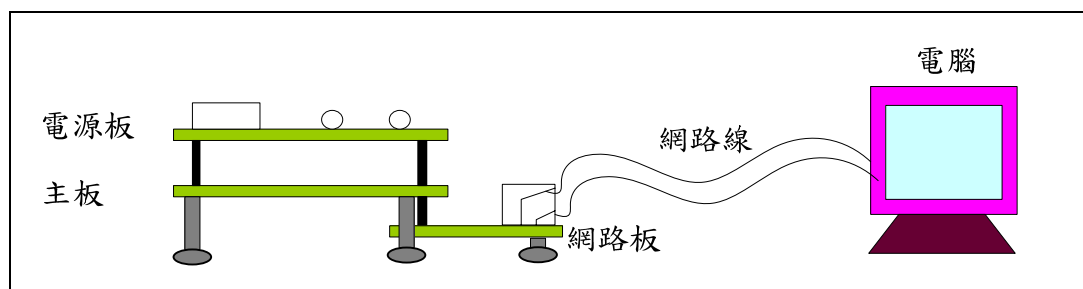


圖 3-6 預留硬體線路介面圖

3.3 電路設計

我們是使用 ORCAD 軟體來製作電路設計，首先必須要將所有使用到的電子元件在 ORCAD 上重新建立 Library 做為繪圖使用之圖庫，製作 Library 的用意是為了將元件的訊號腳位順序定義清楚，以供給繪圖使用的零件不會發生錯誤腳位。接下來是使用 Hierarchy 階層方式如圖 3-7，去製作出所要產生的功能，方便其它使用者容易了解系統線路上的路徑及控制方式。

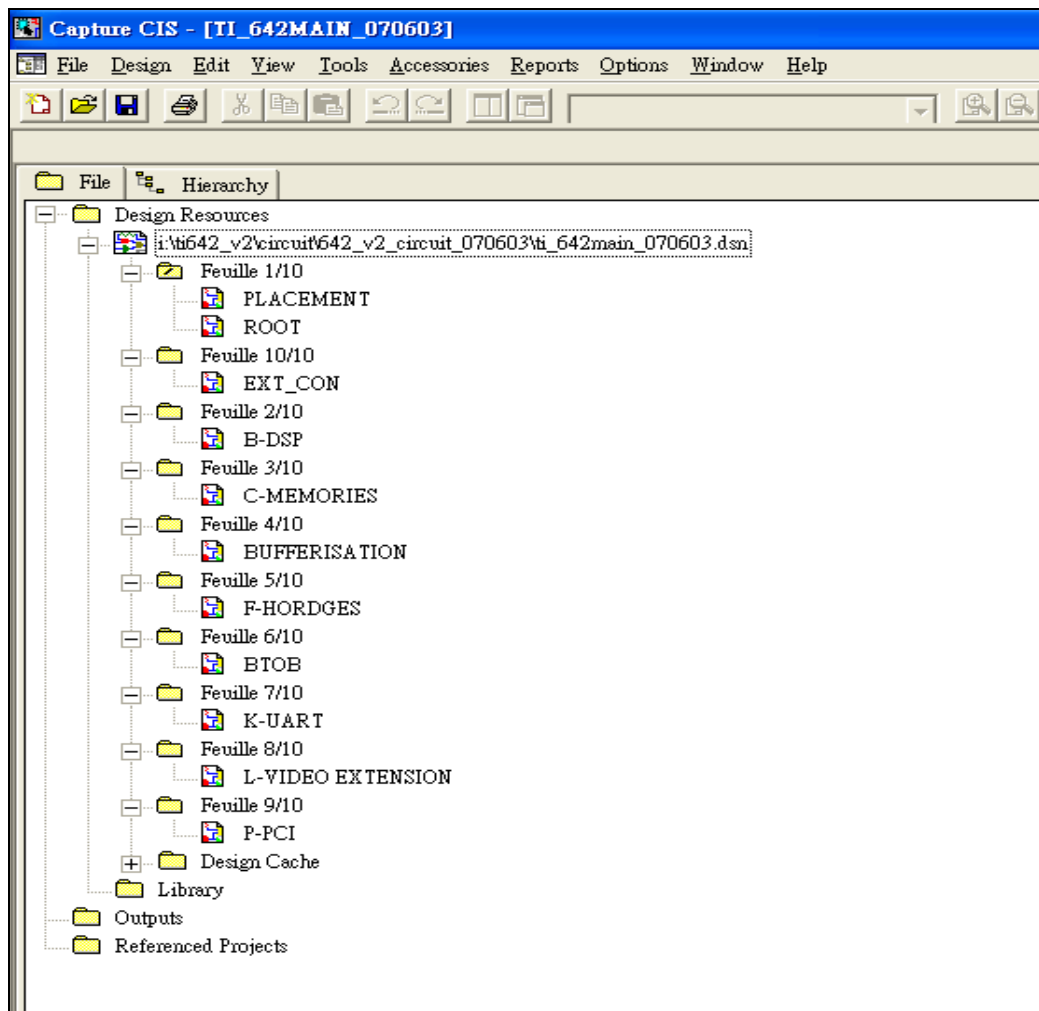


圖 3-7 ORCAD Hierarchy

在線路設計上，以下幾個重點在設計：

- (1) 每一個元件的主要功能及腳位所代表的意義，是被控制還是控制其它元件必需

了解。

- (2) 線路上所用的電阻電容值是否符合線路的特性，是否過大或過小。
- (3) 線路完成後需要跑 DRC (Design Rule Check)，看是否有線路上邏輯錯誤。
- (4) 每一個元件對應到 Layout 時，用的 Footprint 需要特別注意，需要確認是否相對應在 Layout 用的軟體(Power-PCB)是同樣的 Footprint。
- (5) 為了方便之後除錯的速度，增加重要的測試點是很重要的。
- (6) 電容的耐壓是要特別的注意，有些特殊的電容耐壓值需要實際電壓的 2~3 倍，才不會有突然被燒燬的情況。

3.4 佈局及走線

使用 Power PCB 來製作電路板的佈局及走線，先將所有的元件建立 Library，建立的 Library 主要是把元件的包裝大小及腳位間距設定加上焊接的距離製作成零件庫。之後匯入由 ORCAD 轉出的 Net-List 檔案，確認每一個零件是否都是正確無誤的。就開始進入電路板的佈局及走線。在開始前有幾個設計重點，需要特別的處理：

(1)在電路板的規劃上，我們將每一層定義清楚如圖 3-8 所示：

(1-1) 4 層的電路板。

- 第一層及第四層為零件擺放層、和訊號線使用層。
- 第二層為接地(Ground)層，用於將所有接地的訊號集中在此層。
- 第三層為電源(Power)層，用於將所有電源電壓的訊號集中在此層，因為會有不同的電壓(例如：1.4V、3.3V、5V 等)，所以這層需要另外再切割幾個區塊，給各組電壓當作自己電源平面(Power Plane)。

(1-2) 6 層的電路板。

- 第一層及第六層為零件擺放層、和訊號線使用層。
- 第二層為接地層，用於將所有接地的訊號集中在此層。
- 第五層為電源層，用於將所有電源電壓的訊號集中在此層。

- 第三層及第四層為訊號線使用層。

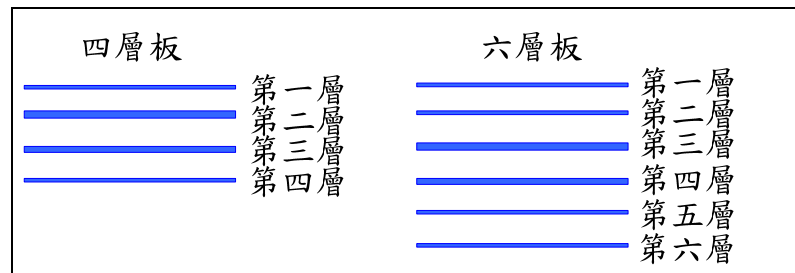


圖 3-8 電路板內層圖

- (2) TMS320DM642 的 CPU 頻率為 600MHZ，相當的快速，相對應的產生出來的熱能會很大，避免影響到其它周邊元件，所以將其位置放在主板與電源板接的另一邊，一旦未來可能發生運作時溫度過熱，可以在上面加上散熱片或是風扇，也不會與板子有所干涉。
- (3)SDRAM 所使用的頻率為 133MHz，線路上是使用 4 顆 16bit Data 長度 SDRAM 並接完成 642 Memory Bus 為 64bit，所以在 Layout 上要求這 64bit 的線路必需在走線的時候，走相同的距離到 642 Chip，避免發生資料傳輸錯誤。這樣的檢查可以用軟體上的工具來實現。
- (4)在板上有 Video 及 Audio 類比信號的輸入／輸出，在走線的時候，不能將數位信號經過這些類比的信號，以防止串音(Crosstalk)的發生。
- (5)在走線的時候，線路上的資料匯流排(Data Bus) 及時脈(Clock)要先完成，剩下的信號之後再完成。資料匯流排及時脈的信號都是走線的距離越小越好，減少線路電感效應。

3.5 重要元件特性

3.5.1 TMS320DM642 DSP

TMS320DM642 處理器主要包含 CPU、內部程式記憶體、內部資料記憶體、DMA 控制器、外部記憶體控制器、周邊裝置等等。CPU 的架構是採用先進的超長指令設

計，可提供 8/16/32/64 位元的資料格式，並有六個數學邏輯單元及兩個乘法器及許多暫存器，負責指令之執行，具有高速的運算能力，一個時脈週期可以同時執行八個指令，而最高的時脈速度可達 600MHz。圖 3-9 為 642 內部的架構圖。

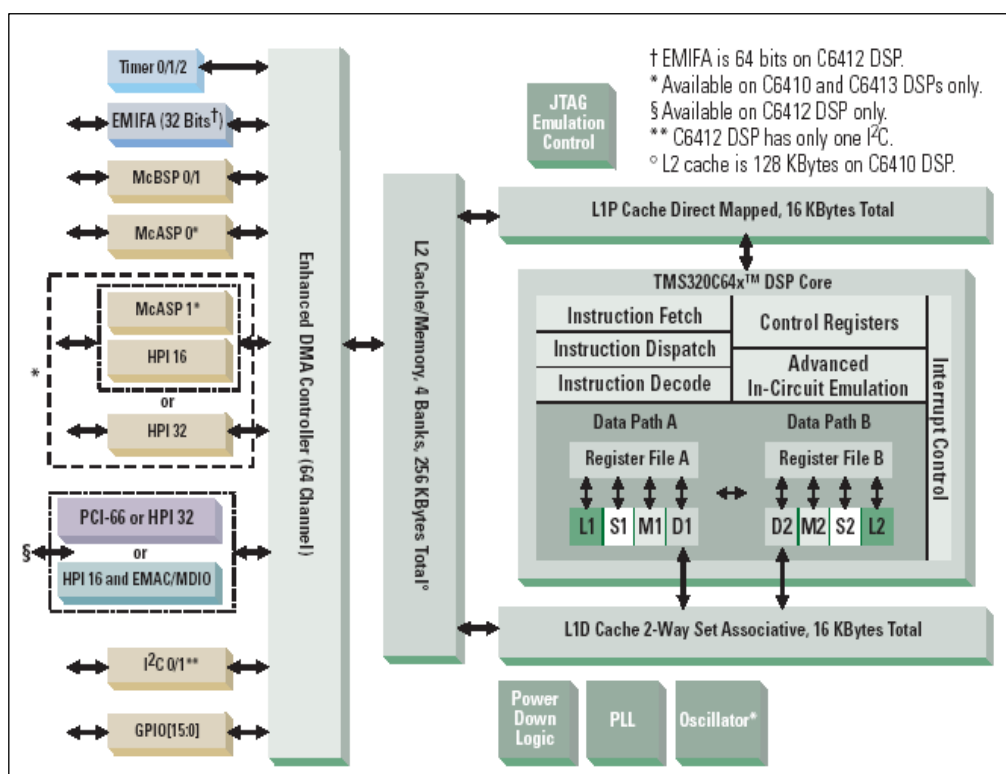


圖 3-9 TMS320DM642 內部架構

3.5.2 MEMORY

TMS320DM642 系統中所使用的記憶體有分成內部及外部使用，外部的有 NOR-Flash 及 SDRAM，內部有 Internal Cache。在系統中使用的 SDRAM 型號為 48LCHM16A2 內部為 1Meg x16 x 4Banks，表示 SDRAM 為 16bit 的 Data 長度，4 個 Bank。因為 TMS320DM642 的 EMIF 為 64bit，所以這個系統使用 4 個 SDRAM 並聯接一起，且 SDRAM 的工作頻率為 133MHz。因為是用 4 顆 SDRAM 所組成一個 SDRAM Data Base，所以在電路設計上使用 Clock Buffer 來為了讓 SDRAM 的 Clock 能同時推動 4 顆，避免發生時脈相位(Clock Phase) 到元件的時間不同，所以使用 Clock Buffer 來完成，型號 CY2305。如圖 3-10 所示

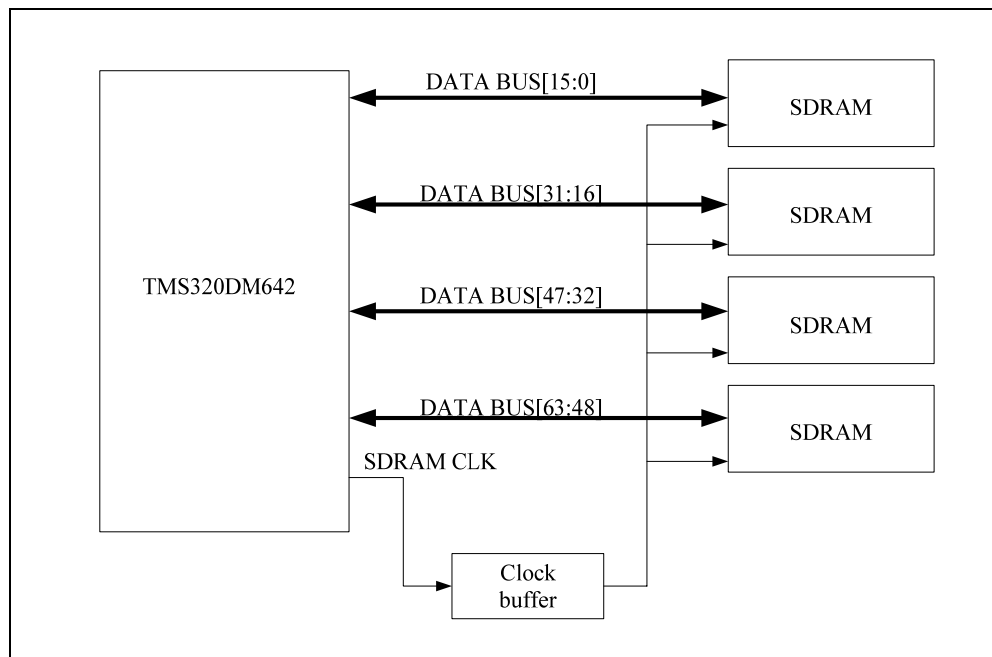


圖 3-10 Clock buffer 與 SDRAM

至於 NOR-Flash 主要是把程式碼放入在此，使用的型號為 S29AL032D901F100，32Mbit (4M x 8bit). 在規劃上，除了放主程式外，另外放入 Boot-loader 的程式放入自此，當系統 Reset 之後就會開始執行 NOR-Flash 的程式碼。

在 TMS320DM642 內部有 256Kbyte 的 SRAM，俗稱為 Cache。於是利用 SRAM 來加速整體運算的速度是件很重要處理的事，加快影像處理速度以減少時間，圖 3-11 為我們規劃使用的情況。

Boot loader 的部份是留著當 Stand alone 的狀況的變數使用區，Cache 是給原本的程式中的變數，堆疊區所使用，剩下的區塊就是拿來加速影像處理的記憶體區塊，利用內部增強型直接記憶體存取(Enhance Direct Memory Access, EDMA)的功能將影像資料搬入到內部 SRAM 來處理。本次規劃有 5 個區塊，主要配置有存放背景圖、背景相減圖、水平邊緣處理完之圖及剩餘的兩塊記憶空間，就是拿來當作暫時處理區的地方。這樣的好處可以減少在整個程式執行的時候，需要一直使用 EDMA 來搬資料的時間。這樣的技巧只需要每次視訊進入 DSP 的時候，利用 EDMA 搬動一次到 SRAM，處理完之後，再用 EDMA 搬到視訊輸出的位置，減少程式執行所需要多耗的時間。

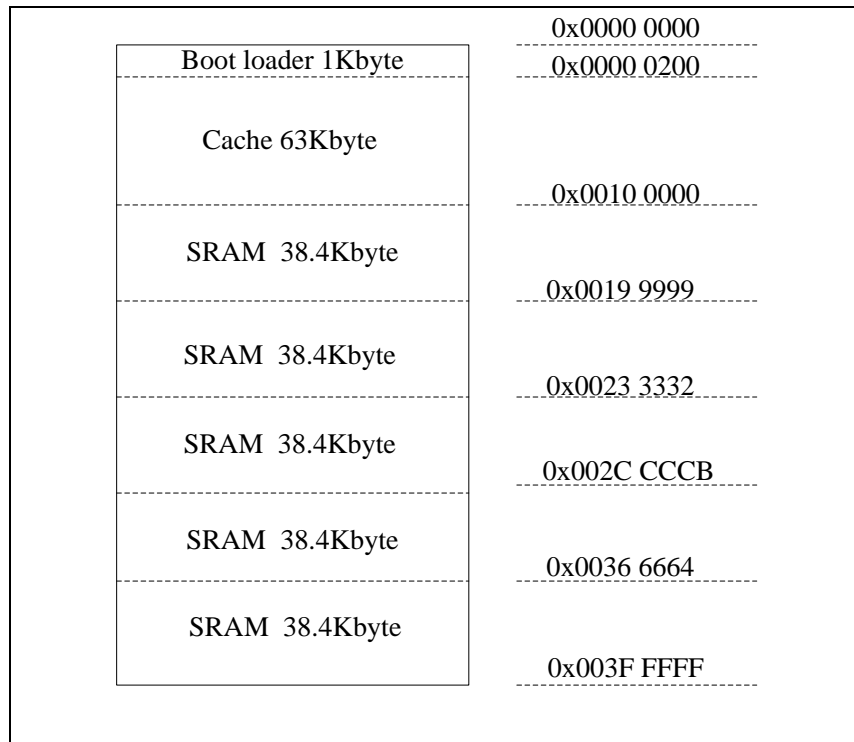


圖 3-11 TMS320DM642 內部 SRAM 規劃圖

3.5.3 CPLD

在 TMS320DM642 系統中所加入的這個 CPLD，主要因為 TMS320DM642 本身能提供的 GPIO 腳位不足，加上系統上有許多周邊元件，它是可以增加系統的 GPIO 控制數量，以及控制 NOR-Flash 的位置 A20，A21，這兩點較高的位置線。主要原因是 TMS320DM642 原本提供的位置線只有 20 條也就是 A0~A19，這樣是儲存 1MByte 的容量。當利用 CPLD 多控制兩個高位置，可以多存四倍較大的容量，也就是 4MByte。

3.5.4 TV Decoder and Encoder

TMS320DM642 系統中所使用的 TV Decoder, 型號為 SAA7114H 含有標準的 NTSC /PAL Composite 訊號輸入, 以及標準 Digital CCIR 601/656 輸出的介面。另外內部還有 Video 的 Scale Engine. 可以輸出不同大小的影像格式。控制 TV Decoder 的介面為積體電路間通訊(Integrated Circuit , I²C)。

TMS320DM642 系統中所使用的 TV Encoder, 型號為 SAA7121H 含有標準的 PAL/NTSC Composite 複合訊號輸出, 以及標準 Digital CCIR 601/656 輸入的介面。控制 TV Encoder 的控制介面為 I^2C 。如圖 3-12 所示:

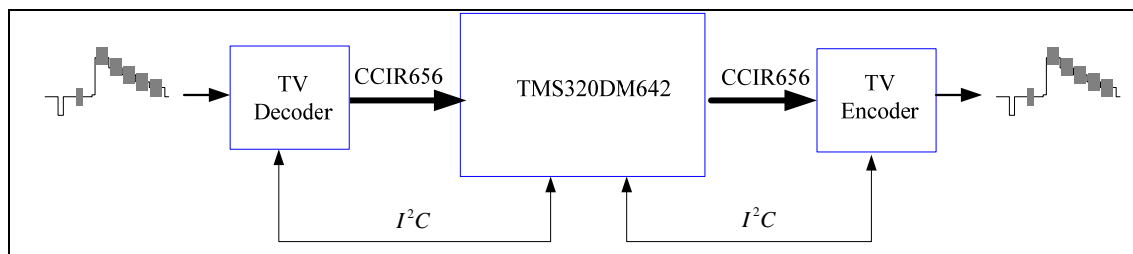


圖 3-12 Video 輸入及輸出

3.5.5 電源系統架構

在 TMS320DM642 平台系統中, 主要外部輸入電壓為 DC 12V, 內部電壓主要為 3.3V 及 1.4V 規劃線路上的電壓轉換, 採用了 PWM(Pulse Width Modulation) IC 及線性穩壓器 LDO (Low Drop Out Linear) 來產生所需的電壓, 如圖 3-13 所示:

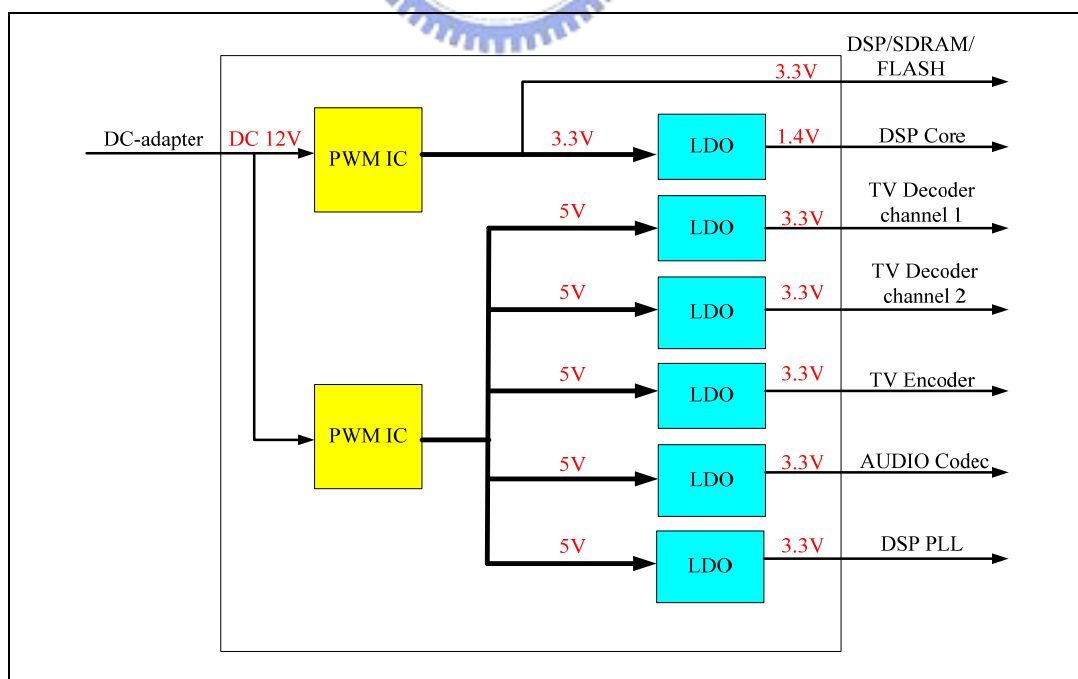


圖 3-13 模組電源規劃架構

第四章 行人偵測及追蹤

本論文另外一個發展的重心，就是將行人偵測的方法，實際呈現在 DSP 的平台上，且利用最佳化的方式，進行行人偵測並達到即時處理。發展初期，先利用 PC 上的平台來模擬視訊輸入的畫面及輸出的畫面，使用 Visual C++ 程式語言來發展行人偵測的演算法。首先是利用背景相減法將畫面上的移動物體區分出來，並利用常見的影像處理的方法如：邊緣強化、雜訊濾波器、侵蝕及膨脹等等，將畫面處理到一定的程度，之後利用角點的偵測來判斷是否為行人的頭部。最後確認一些行人的特徵，這樣可以完成行人候選人的選取。下一個步驟就是行人追蹤的方式，發現了行人能持續的跟蹤，直到離開畫面中的選取範圍(Region of Interesting, ROI)位置，這是一項很重要的處理。當兩個行人在畫面中突然的重疊及離開，程式應該要有最佳的預測及處理，達到追蹤良好的情況。在電腦上發展完演算法之後，最重要的是要將發展完成的程式移植到 DSP 平台上，要克服 DSP 上的記憶體配置，因為它是一個有限的容量，比起電腦來看是容量是小了，所以要能最佳化程式的內容，縮短處理的時間，減少程式使用太多記憶體的位置，這樣就可以達到高效率及即時的處理結果，完成本論文要達到的理想境界。

4.1 行人偵測

在影像處理中，要分析那些是影像為需要辨識的範圍，而行人影像便是屬於移動的影像，必須要被辨識出的資料。在一般常見的移動影像偵測的方法是背景相減法，利用一張固定未移動的影像當作背景圖，與目前的影像相減，便可分割出移動影像的輪廓。本章節主要是探討如何在單張影像，利用基本的影像處理方法，如邊緣強化、雜訊濾波器、侵蝕及膨脹等等，加上頭部、圓密度、邊緣運算來完成行人候選的偵測，如圖 4-1。

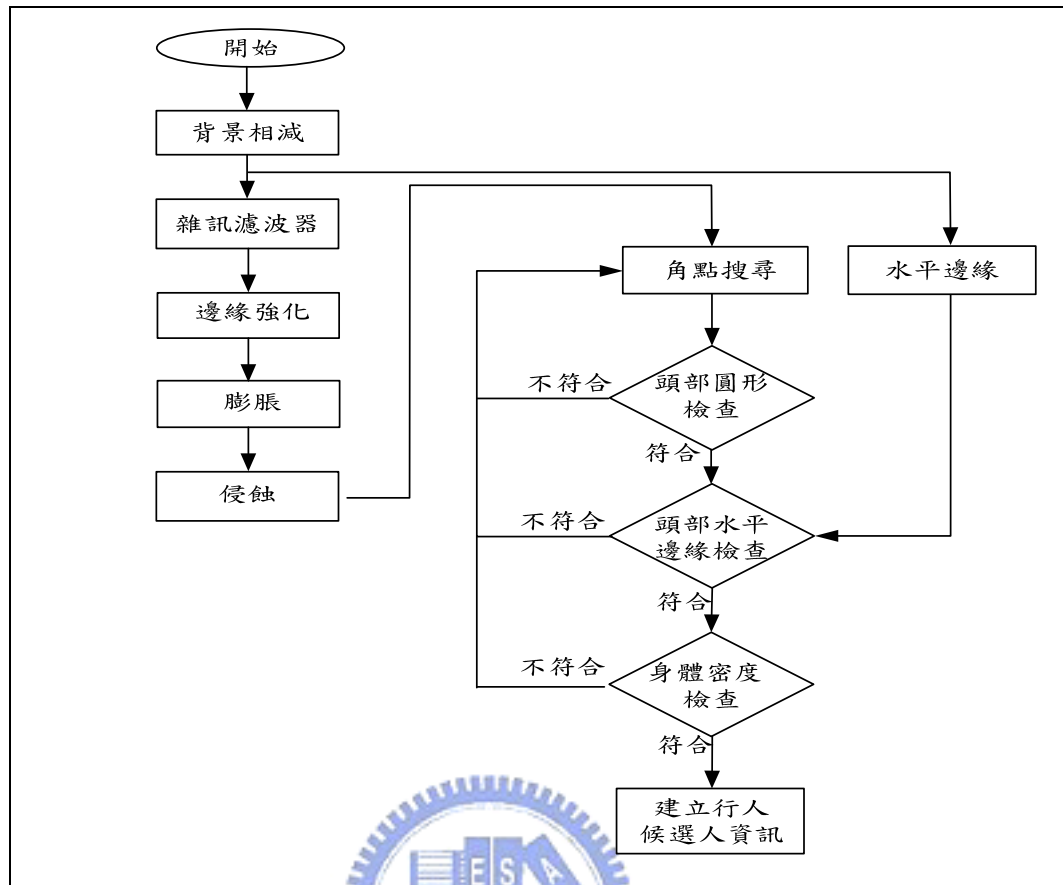


圖 4-1 行人候選人偵測流程

本論文所偵測的路口中，是一個倒 T 字型路口，主要的偵測目標是由左至右或
是由右至左的方向，圖中綠色包圍的方框為本論文所定義偵測的範圍，如圖 4-2 實
際路口圖：



圖 4-2 實際路口圖

4.1.1 背景收斂

為了達到能移動影像輪廓的完整性，並減少前景留下的雜訊。本論文所使用的收斂的方式為一個以統計的方式來建立背景圖。主要的觀念是以當取一段連續影像如圖 4-3，在相同 x 與 y 座標位置的像素上，統計出像素值出現最多次的當作是背景影像。收斂背景的時候，本論文將移動中的像素如圖 4-4 所示，不列入在統計的方法。主要方法將前一張及後一張畫面去做彼此相減，然後取絕對值如公式 4-1， $P_{x,y}^{t-1}$ 為前一張畫面的像素值， $P_{x,y}^t$ 為這一時刻畫面的像素值， $D_{x,y}$ 為相減之後取絕對值的結果。當 $D_{x,y}$ 計算出來的絕對值大於設定的臨界值，如公式 4-2 $M_{x,y}$ 為 1 的像素，我們就認定這個像素為移動的像素，不將它列入背景影像統計之中。 $M_{x,y}$ 為 0，再將像素值累積至統計區 $h_{x,y}$ 。

$$D_{x,y} = |P_{x,y}^{t-1} - P_{x,y}^t| \quad (4-1)$$

$$M_{x,y} = \begin{cases} 1, & \text{if } |D_{x,y}| > \delta; \\ 0, & \text{otherwise.} \end{cases} \quad (4-2)$$

$$P_{x,y} = \max_{0 \leq l \leq L} h_{x,y}(l) \quad (4-3)$$

在建立背景的計算中，會將影片中的每一個像素的值利用直方圖儲存起來。在一段時間內，在相同位置的像素，經過每張畫面統計像素值之後，將重覆最多的像素值當作背景影像。公式 4-3 中， L 表示有 256 階的值，就是每一個像素要去找這 256 階之中，找出那一階是統計出來最多的值，這個像素 $P_{x,y}$ 的背景值就是這一階。在計算移動像素的時候，我們將 δ 固定為 20，做為計算的依據。

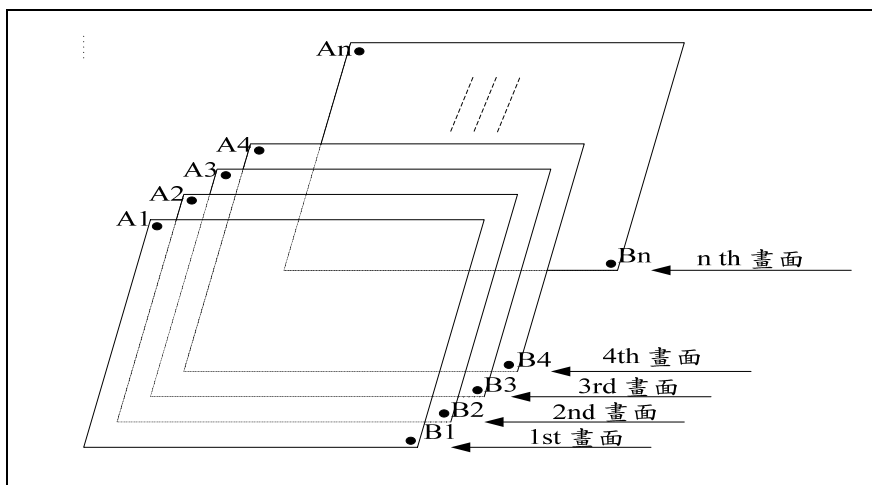


圖 4-3 背景收斂示意圖



圖 4-4 移動物體收斂圖

4.1.2 移動物件擷取

應用在影像追蹤上，在背景不變的情況下，在連續影像中的移動目標可以由影像相減比較出移動方向與位置。

$$P_j(x, y) = \begin{cases} 1, & \text{if } |I_j(x, y) - B(x, y)| > T_{hd}; \\ 0, & \text{otherwise.} \end{cases} \quad (4-4)$$

在公式(4-4)中， $I_j(x, y)$ 表示第 j 張影像 (x, y) 位置像素值， x 為水平方向座標，

y 為垂直方向座標， B 為背景的參考影像， I 為目前影像的像素值，將兩張影像相減之後取絕對值如圖 4-5 所示，可以看到有人的形體產生。在取絕對值後，當差值大於閾值 T_{hd} 的標記為 1，否則標記為 0。其中 T_{hd} 的選擇為經驗值，在本論文中 T_{hd} 是為 30，二值化後的影像如圖 4-6 所示。人體的整個區域基本上都顯示出來了。



圖 4-5 背景相減圖

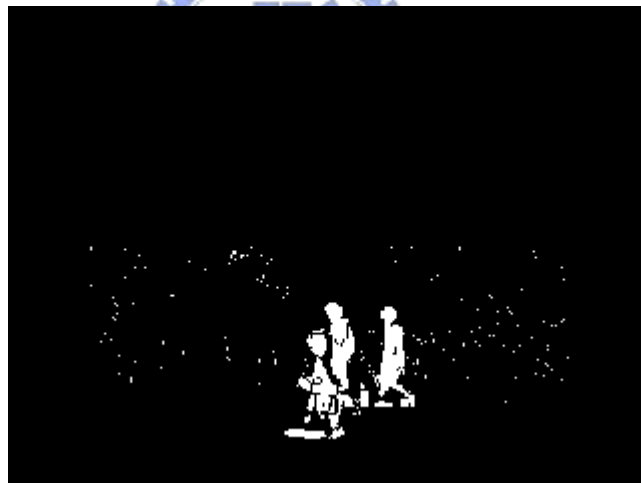


圖 4-6 背景相減二值化圖

4.1.3 雜訊濾除 Noise Filter

在取像的過程中，影像多少會有雜訊的產生，過量的雜訊存在，會影響所要辨識的圖形，辨識的結果可能會因此出錯。所以在開始處理影像的時，將雜訊降到最低，是影像處理前重要的工作。在本論文中，我們使用的 5x5 的 Filter 來處理雜訊，圖 4-7

表示 Noise Filter Matrix，當掃瞄到 P_{13} 位置的時候，依照公式 4-5 來判斷 P_{13} 的值為何。當 P_{13} 為 0 的時候，輸出為 0。當 P_{13} 為 1 的時候， $P_1 \sim P_{25}$ 的像素要有 9 (T_{hd}) 個以上的像素為 1，則輸出為 1，反之為 0。處理完之結果如圖 4-8 所呈現的結果，大部份的雜訊都去除了。

P_1	P_2	P_3	P_4	P_5
P_6	P_7	P_8	P_9	P_{10}
P_{11}	P_{12}	P_{13}	P_{14}	P_{15}
P_{16}	P_{17}	P_{18}	P_{19}	P_{20}
P_{21}	P_{22}	P_{23}	P_{24}	P_{25}

圖 4-7 5x5 Noise Filter Matrix

$$P = \begin{cases} 1, & \text{if } \left(\sum_{z=1}^{25} P_z \right) > Thd; \\ 0, & \text{otherwise.} \end{cases} \quad (4-5)$$



圖 4-8 經過雜訊濾除處理之後

4.1.4 邊緣強化 Edge Enhance

使用邊緣強化是把邊緣的特徵加強，也就是原本在邊緣上一個像素為單位的圖，並成以 4 個像素為單位的圖。如圖 4-9 為左邊 P 是原本圖的像素，右邊 O 為經過邊緣強化所產生出來的像素。做法如公式 4-6 所表示，讓 2x2 為單位的 Block Base，從影像 ROI 最左上開始，有左至右，有上而下，每 2x2 的 Block 就計算一次。假如

在原本 2x2 的 Block 中只要有一個像素為 1 的話，輸出 O 的四個像素就全部為 1，如果原本 2x2 內部全部為 0，輸出 O 的四個像素就全部為 0，影像處理完之後的圖如圖 4-10 所示。

將經過雜訊濾波器的影像做邊緣強化，主要的原因是為了在接下來要做的影像中角點偵測的部份，能夠清楚找到影像中的角點，並加以分類及判斷，所以邊緣強化與角點偵測很重要。

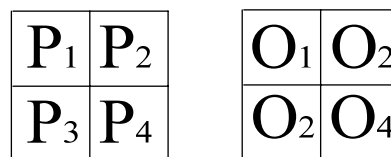


圖 4-9 邊緣強化表示之像素

$$O_{1\sim 4} = \begin{cases} 1, & \text{if } \left(\sum_{z=1}^4 P_z \right) \geq 1; \\ 0, & \text{otherwise.} \end{cases} \quad (4-6)$$

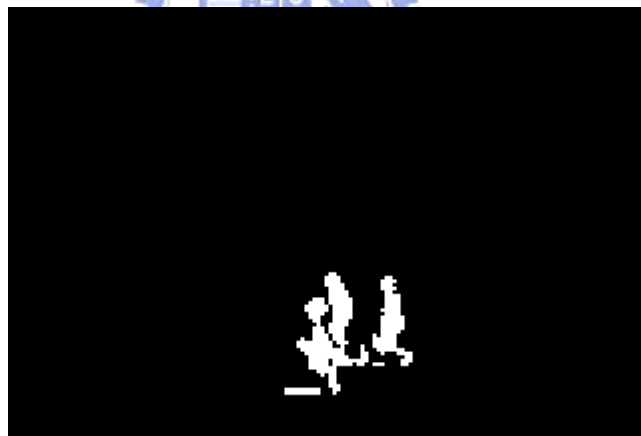


圖 4-10 邊緣強化處理

4.1.5 膨脹 Dilation

在影像重整最常使用的是膨脹(Dilation)與侵蝕(Erosion)兩種方法，在數位影像處理上稱為數學形態學(Mathematical Morphology)，侵蝕的方法是將物體可能多出來的邊緣或因為二質化後多出來的點將其消除，或者是物體在經過膨脹後，為使分析特徵

值時較為精確，而必須再膨脹後給予侵蝕一次，這樣就可以避免物體膨脹過大，使得分析時不太準確。所以本論文所使用的方法是先膨脹處理，再做侵蝕，這樣為數學形態中所產生出 Closing 的效果。

如果圖形經過二質化後，呈現邊緣有些崎嶇不平或有凹洞時，可以利用此方法將圖形膨脹，看看是否可以改善邊緣或凹洞的情形。圖 4-11 是表示拿來做膨脹的矩陣，利用公式 4-7 來當運算 P_5 位置像素的時候，在二值化影像中 P_5 是否為 1，符合的話需再判斷周圍的 8 個點($P_1 \sim P_9$)是否有值皆為 1，若有其中有為 1， P_5 就為 1，反之($P_1 \sim P_9$)都為 0 的時候， P_5 為 0。經過膨脹處理的影像結果如圖 4-12。

$$P_5 = P_1 \cup P_2 \cup P_3 \cup P_4 \cup P_6 \cup P_7 \cup P_8 \cup P_9 \quad \cup \text{表示OR運算} \quad (4-7)$$

P_1	P_2	P_3
P_4	P_5	P_6
P_7	P_8	P_9

圖 4-11 侵蝕與膨脹 Matrix

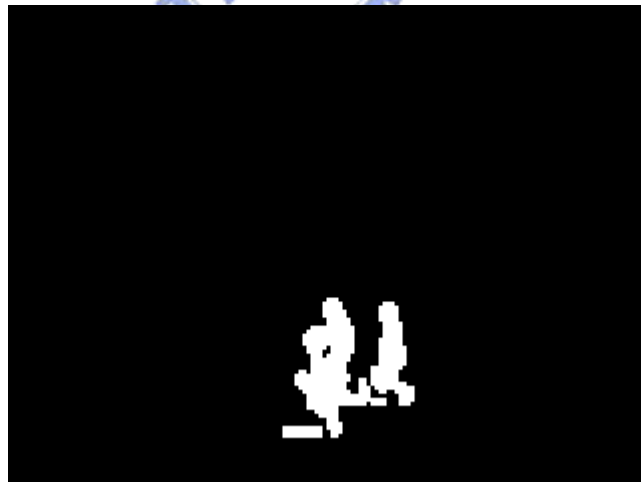


圖 4-12 經過膨脹的結果

4.1.6 侵蝕 Erosion

因為剛剛用過膨脹的方法，所以使用侵蝕來平衡前景的圖。在侵蝕的運算中用

的 matrix 如圖 4-11 所示，利用公式 4-6 來計算 P_5 產生出來新的像素值。首先判斷 P_5 在二值化影像中是否為 1，符合的話需再判斷周圍的 8 個點($P_1 \sim P_9$)是否所有值皆為 1，皆為 1 後， P_5 才為 1，反之 P_5 為 0，經過侵蝕公式 4-8 處理後的影像結果如圖 4-13。

$$p_5 = p_1 \cap p_2 \cap p_3 \cap p_4 \cap p_6 \cap p_7 \cap p_8 \cap p_9 \quad \cap \text{表示AND運算} \quad (4-8)$$



圖 4-13 經過侵蝕的結果

4.1.7 角點偵測 Corner Detection

現在影像處理技術應用廣泛，而邊緣偵測也是一個重要的課題，在其中邊角的偵測(Corner Detection)更為有用。本論文利用角點來檢知可能為行人的頭部邊緣的地方，找出可能的地方，再加上確認物件二值化之後的密度，水平邊緣處理之後等，可以辨識出行人頭部。我們使用很簡單的方式來判斷角點的種類，主要分成四種角點如下面所敘述：

Corner Type1 為物體左上的角邊，我們定義當影像出現如圖 4-14 一樣情況的區塊就為 Type1，它所呈現出來的角邊是朝著左上角的位置，有可能是物體最左上角的地方，如圖 4-15 所示這是一個經過二值化之後的人頭可以看到做左上角的情形就跟我們定義的 Corner1 為一樣的情形。

0	1	1
1	1	1
1	1	1

0	0	0
0	1	1
0	1	1

圖 4-14 角點 TYPE 1 的種類

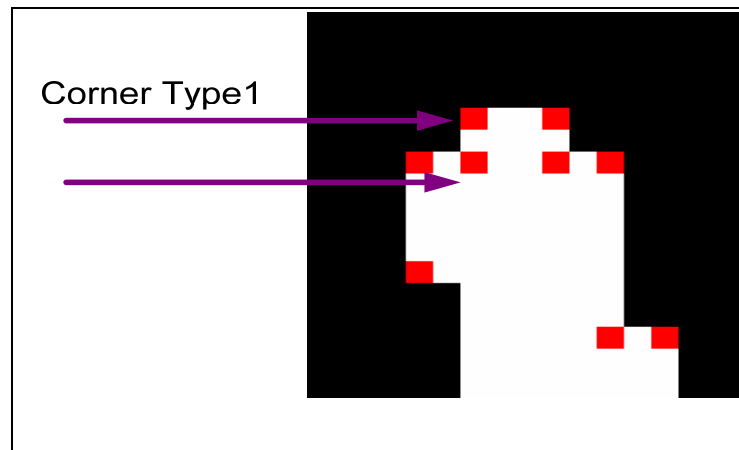


圖 4-15 角點 TYPE 1 的位置

Corner Type2 為物體右上的角邊，我們定義當影像出現如圖 4-16，一樣情況的區塊就為 Type2，它所呈現出來的角邊是朝著右上角的位置，有可能是物體最右上角的地方，如圖 4-17 所示這是一個經過二值化之後的人頭可以看到做右上角的情形就跟我們定義的 Corner Type2 為一樣的情形。

1	1	0
1	1	1
1	1	1

0	0	0
1	1	0
1	1	0

圖 4-16 角點 TYPE 2 的種類

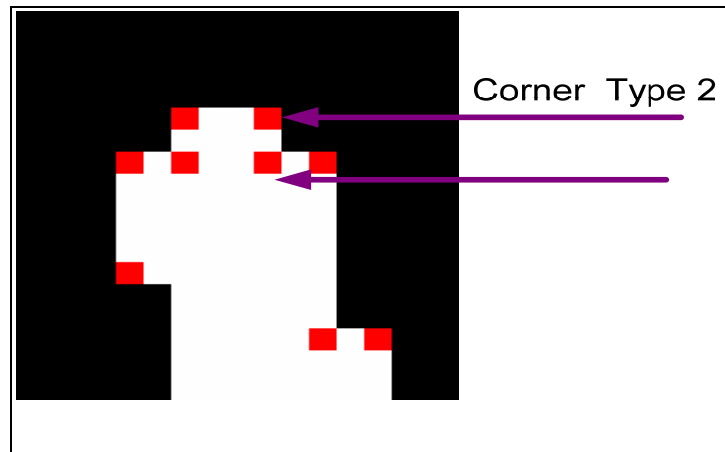


圖 4-17 角點 TYPE 2 的位置

Corner Type 3 為物體左下的角邊，我們定義當影像出現物體如圖 4-18，一樣情況的區塊就為 Type3，所呈現出來的角邊是朝著左下角的位置，有可能是物體最左下角的地方，如圖 4-19 所示這是一個經過二值化之後的人頭可以看到做左下角的情形就跟我們定義的 Corner Type 3 為一樣的情形。

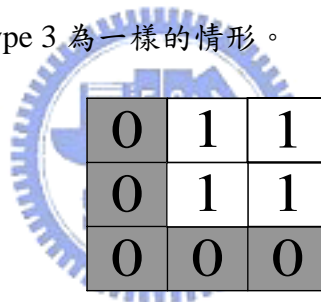


圖 4-18 角點 TYPE 3 的種類

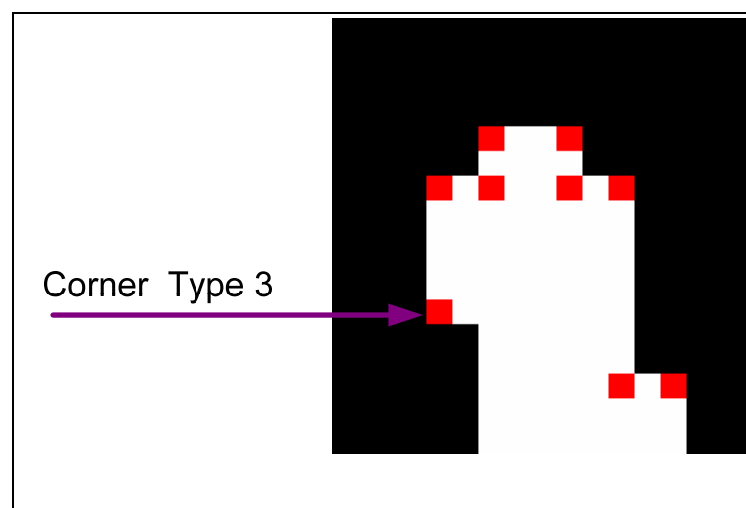


圖 4-19 角點 TYPE3 的位置

Corner Type 4 為物體右下的角邊，我們定義當影像出現物體如圖 4-20，一樣情況的區塊就為 Type4，所呈現出來的角邊是朝著左下角的位置，有可能是物體最左下角的地方，如圖 4-21 所示這是一個經過二值化之後的人頭可以看到做左下角的情形就跟我們定義的 Corner Type 4 為一樣的情形。

1	1	0
1	1	0
0	0	0

圖 4-20 角點 TYPE 4 的種類

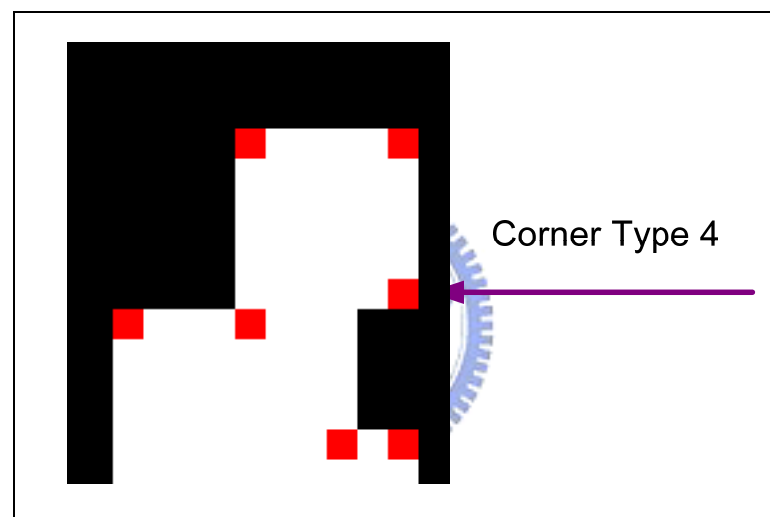


圖 4-21 角點 TYPE4 的位置

4.1.8 水平邊緣 Horizontal Edge

以行人的外型輪廓當作辨識特徵，透過線條化的抽取，去除大部份的像素及獨立點，只取出物體重要的邊緣線條。在取出的線條化又可分成垂直線條與水平線條兩類，本論文只用到的為水平的 *Sobel* 方式，原始圖以 3x3 的遮罩如圖 4-22 由左至右，由上而下的方式與 *Sobel* 的濾波器的乘積，每一點的像素都去需要與旁邊像素做係數相乘之後再加減為新的像素值。如公式 4-9 它是一個水平的 *Sobel* 公式，將 $Z_7 \sim Z_9$ 的部份先相加，再減去 $Z_1 \sim Z_3$ 的總合後取絕對值，其結果值 h_5 將取一個零臨界點當作分

界，小於門檻值的為 0，大於的為 255，將計算出來的像素值 e_{z5} ，把它填回新的影像區的中間位置。

$$h_5 = \left| (z_7 + 2 \times z_8 + z_9) - (z_1 + 2 \times z_2 + z_3) \right|$$

$$e_{z5} = \begin{cases} 255, & \text{if } h_5 > \delta; \\ 0, & \text{otherwise.} \end{cases} \quad (4-9)$$

在圖 4-22 左邊的 Matrix 中 Z_i 表示在 3x3 影像範圍內的像素 i 之灰階值， δ 為設定二值化的臨界值參數(Threshold Value Parameter)。中間的 Matrix 是表示水平方向做邊緣需要使用的常數，右邊的 Matrix 是垂直方向做邊緣所需要使用的常數。圖 4-23 為經過水平邊緣處理過後之影像。

Z_1	Z_2	Z_3	-1	-2	-1	-1	0	1
Z_4	Z_5	Z_6	0	0	0	-2	0	2
Z_7	Z_8	Z_9	1	2	1	-1	0	1

圖 4-22 Sobel Matrix

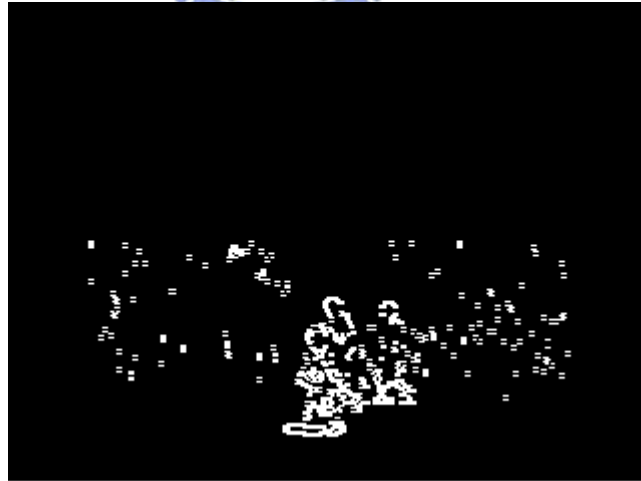


圖 4-23 水平邊緣處理之後的圖

4.1.9 人頭偵測 Head Detection

有了前幾個階段的影像處理，接下來就是要來判斷是否為行人候選人，在圖 4-1

流程之中，我們已經可以確認在圖中有產生出角點的地方，於是我們利用角點做一個簡單的判斷，來證明是否為一個行人候選人的頭部。對抓到的前景區域做搜尋找 Corner Type 1，以 Type 1 為基準點，由以下幾點來判斷：

- (1) 向左 4 個像素，向下 4 個像素做偵測找 Type 1，為最左邊的點
- (2) 向右 20 個像素，向上 4 個像素到向下 4 個像素做偵測找 Type 2，為最右邊的點。
- (3) 向右 6 個像素，向上 4 個像素做偵測找 Type 1，為最上方的點
- (4) 如果左到右符合大小 6 到 15 個像素，以左右和上基準點，再向下找 10 個像素，由最左邊點一直到最右邊點的距離，最上方的點開始往下約 10 像素的整體位置，處理這樣的水平邊緣的圖去做掃描，計算出這個面積的邊緣的點數，如果點數大於左右寬度的兩倍以上，就判斷可能是頭部。
- (5) 接下來再利用頭幾乎看起來像是圓的輪廓，於是我們在既有的地區內畫上一與頭大小差不多 10x10 像素的圓，然後確認這個範圍內 Y 平面範圍為 10~128 是否有超過一半以上，這樣我們判定這可能是個頭。
- (6) 當找到頭之後，我們將區域往下來尋找，也就是從剛剛角點最上方的那一個點開始往下約 46 個像素，左右的距離拉到約 16 個像素的地方，於是我們計算這個區域是否密度是否有到一個臨界值以上，如果是的話，我們就把它當作是一個行人的候選人，如果不是的話就可能表示這樣的區域所看頭的位置，實際上不是一個行人的頭，而可能是其它東西所造成的誤判。圖 4-24 為行人候選人的影像。

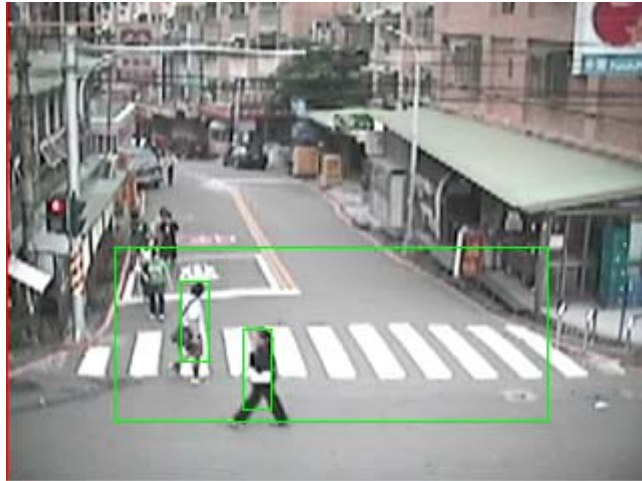


圖 4-24 行人候選人之圖

4.2 行人追蹤

當有前面所介紹的方法，於是我們判斷出行人候選人。接下來就是要如何判斷出是一個真正的行人，並加以追蹤行人的動向，是否一直在所觀察的畫面之中，並將其結果記錄下來。於是將整個處理行人偵測的想法，將其實現在 DSP 平台之上，要能夠真正跑在這樣的平台之上，達到相當好的效能，程式的最佳化是一個很重要的步驟。於是將如何把程式最佳化也是一個很重要的課題。

4.2.1 建立追蹤物件

當產生出行人的追蹤物件，於是我們要能夠開始在之後的視訊畫面能夠順利的跟蹤這個行人物件直到它離開了這個監視範圍。於是我們定義了建立及追蹤物件的流程，如圖 4-25 所示，首先當進入到這個流程，我們會先去比較目前這張畫面上的所有行人候選人與前一時刻追蹤到所有的行人，彼此去找出相關連比較，在行人候選人以重疊率及直方圖表較法先找出有與行人候選人相關的行人。如果有找到的話就將行人候選人的資訊更新在這個追蹤行人的資訊，包含目前位置上所累積的直方圖結果，更新行人的動態特徵。如果行人候選人這時候都沒有找到的話相關連的追蹤行人，我們將會把這張畫面的剩下行人候選人與上一張畫面的行人候選人進一步找出是否有

重疊率高的物件，如果有的話，我們會將它當作是一個新的追蹤的行人，這時候我們會儲存目前這個行人所在的區域累積出來的直方圖結果，做原始特徵及動態特徵，這些特徵為追蹤行人比較的依據。當比較完這張畫面上所有的行人的候選人之後，表示有可能是新的行人物體已經處理完畢了，接下來要看的是上一張畫面所追蹤到的行人是否在這一張都沒有被更新的資訊的情況，如果有的話，可能是被其它物體檔到或是這張畫面找不到相關的行人候選人，只要是沒有離開這個畫面的區域，就只是暫時找不到這個行人。於是我們可以利用之前的軌跡來預測這個行人的位置，並在附近的區域利用直方圖比對法找尋更精確的位置資訊，如發現預測的時候會跟其他行人有相互重疊的情況發生，我們會將重疊的區域不考慮，只針對沒有重疊的區域，將它使用直方圖比對，找出好的預測結果。預測找尋完之後，也更新目前的資訊以便下一張畫面的追蹤。當做完以上行人更新或是預測新的位置之後，我們還會再確認每一個被追蹤的行人是否在目前的畫面中所包含的身體的密度有無達到一定的臨界值，以及是否與原始特徵不相近，如果密度小於臨界值的話，或是與特徵不相近的話，表示我們的追蹤結果是錯的，我們會取消這個人的資訊，重新再出新的行人。當行人離開 ROI 之後，我們也會將取消個這個人的資訊，繼續處理其它行人的影像。

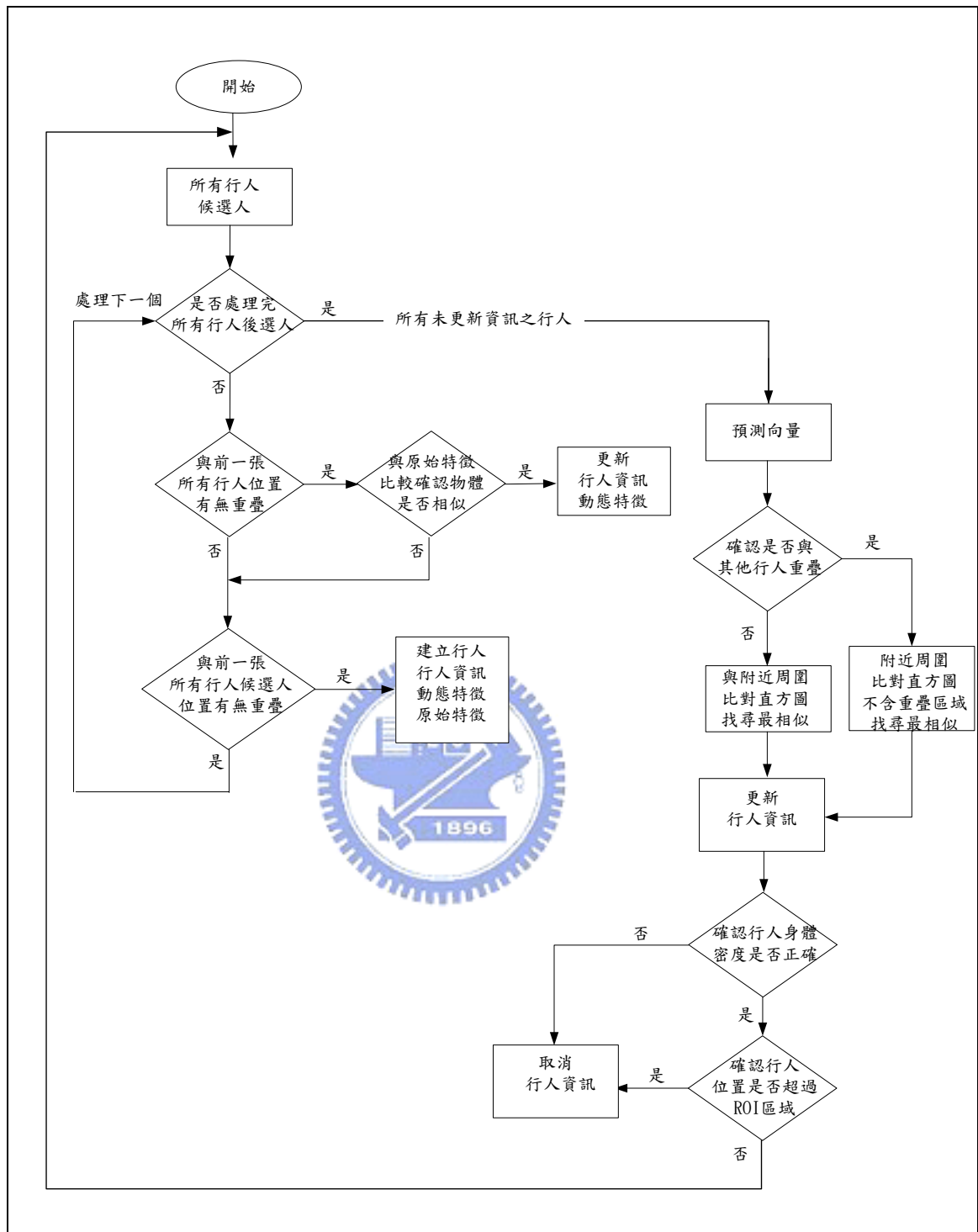


圖 4-25 建立追蹤物件流程

4.2.2 行人判斷

當判斷出頭的特徵時候，這時後我們稱所框到的為“行人候選人”，接下來就是要能真正判斷為“行人”，並開始追蹤行人，看行人是否往左還是往右行走。要能確定追蹤到的物件為行人時，我們定義的方式為需要在連續兩張的畫面中，有行人候選人經過一張畫面之後的距離很近，如圖 4-26 所示。我們以交疊比例來確認行人候選人是否為我們所要追蹤的行人，如果是的話，將它標示為行人，並且從此開始追蹤，直到離開畫面監視的區域。

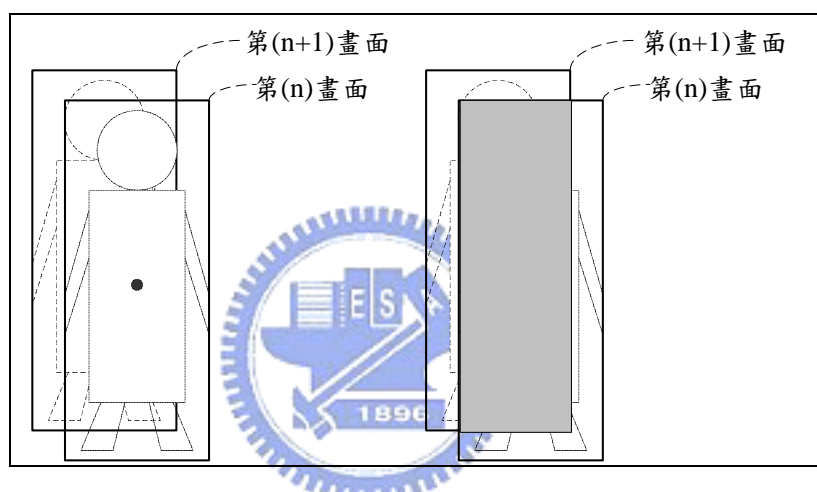


圖 4-26 行人物體重疊示意圖

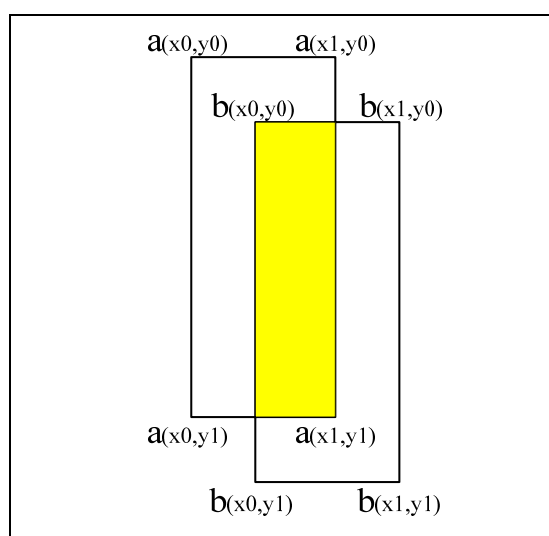


圖 4-27 重疊區域面積表示圖

計算交疊比例的方法：

如圖 4-27 所示：有一個 a 的物件與一個 b 的物件產生出重疊，利用彼此之間的座標及寬高計算出重疊率。方法如下，其中 a, b 的 X,Y 座標如圖 4-27 內標示。

(1)首先計算出 a 及 b 的面積，利用以知的坐標來計算，如公式 4-10 及 4-11。 a_{area}

為 a 的面積， b_{area} 為 b 的面積，

$$a_{area} = (a_{x1} - a_{x0}) \times (a_{y1} - a_{y0}) \quad (4-10)$$

$$b_{area} = (b_{x1} - b_{x0}) \times (b_{y1} - b_{y0}) \quad (4-11)$$

(2)接下來找出最這兩區域的最邊邊點，也就是最右邊的點 r_{max} 、最左邊的點 l_{min} 、最上面的點 t_{min} 及最下面的點 b_{max} ，如公式 4-12，公式 4-13，公式 4-14 及公式 4-15。

$$r_{max} = \max \{a_{x1}, b_{x1}\} \quad (4-12)$$

$$l_{min} = \min \{a_{x0}, b_{x0}\} \quad (4-13)$$

$$b_{max} = \max \{a_{y1}, b_{y1}\} \quad (4-14)$$

$$t_{min} = \min \{a_{y0}, b_{y0}\} \quad (4-15)$$



(3)再來計算出在水平寬度及垂直高度的地方重疊的區域為多少，如公式 4-16 及公式 4-17。 w 為在水平重疊的區域， h 為在垂直重疊的區域。

$$w = ((a_{x1} - a_{x0}) + (b_{x1} - b_{x0})) - (r_{max} - l_{min}) \quad (4-16)$$

$$h = ((a_{y1} - a_{y0}) + (b_{y1} - b_{y0})) - (b_{max} - t_{min}) \quad (4-17)$$

(4) 找出 a 及 b 最小的面積，如公式 4-18。 a_{min} 為兩者最小的面積。

$$a_{min} = \min \{a_{area}, b_{area}\} \quad (4-18)$$

(5)將重疊的水平寬度乘上重疊的垂直高度算出來的面積除上最小的 a 或 b 的面積， c 為計算出來兩個區塊重疊的比率如公式 4-19， c 與 0 判斷誰是最大值，最後的值就是表示兩個區塊重疊的比率，如公式 4-20。 $rate$ 為計算出來的重疊比率。

$$c = \frac{w \times h}{a_{\min}} \quad (4-19)$$

$$rate = \max\{0, c\} \quad (4-20)$$

4.2.3 向量預測

向量預測就是當在發生在這張畫面找不到相關的行人候選人時，利用預測的方法來達到有效的追蹤行人的目的。於是我們提出以下的方式來方便我們預測行人新的位置，並可以大量減少 DSP 的運算量。我們是參考前四張畫面的時候，這個行人所在的位置如圖 4-28 所示，也就是第(n-4)畫面的位置所在。有了前四張的位置，於是我們使用 x 及 y 方向分開計算的方式來預測新的位置。有 x 方向來說，如公式 4-21 所示目前這張畫面(n)的座標 $x_{(n)}$ 減去上一張畫面的座標 $x_{(n-1)}$ ，產生出來的間距，然後由此再往後計算每一張畫面的間距。於是將所有的間距值加總之後除 4 做平均。平均出來的值加上目前的所在的座標，這樣就可以產生出新 x 方向的座標。x 方向的做法如公式 4-22 與 y 方向一樣，參考前四張畫面的座標關係來猜測新的 y 座標。

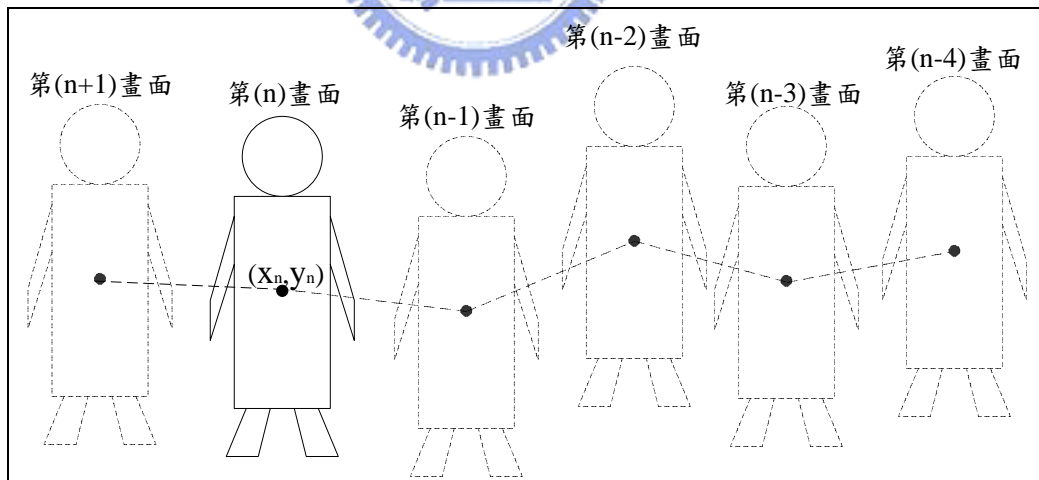


圖 4-28 行人行選預測圖

$$x(n+1) = \frac{1}{4} \left(\sum_{k=0}^3 (x(n-k) - x(n-k-1)) \right) + x(n) \quad (4-21)$$

$$y(n+1) = \frac{1}{4} \left(\sum_{k=0}^3 (y(n-k) - y(n-k-1)) \right) + y(n) \quad (4-22)$$

n 為目前的畫面， k 為前四張畫面的索引，一旦預測出來這個座標之後，我們還

會在這個座標上的位置附近上下左右各多三個像素的位置範圍內如圖 4-29，利用直方圖比較法來比較找出最接近原始圖的方式，這樣可以找出最正確的預測座標。

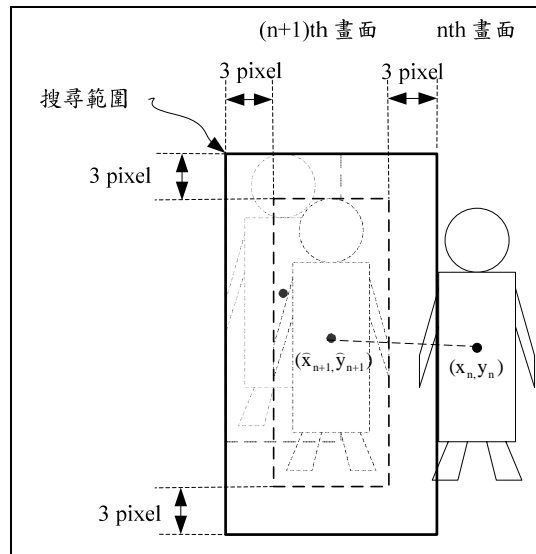


圖 4-29 周圍搜尋比對法

4.2.4 直方圖比對法

做完向量預測之後，要確保行人追蹤上得到好的座標結果，我們加上了直方圖的比較法，因為在預測向量上可能會預測的有些偏差，於是本論文在預測向量之後，會在預測點的位置找尋，更接近物體位置所在，以防止預測上所帶來的嚴重錯誤。於是將預測向量之後產生出來的位置上，在它四周範圍 3 個像素的距離差，統計出行人大小的直方圖資訊與原本一開始所產生出來的行人直方圖資訊，兩者用直方圖去做比較，找出最接近行人所在的位置。一旦之後每次都有判斷到行人候選人，我們會把移動特徵一直更新，讓直方圖比較法拿移動特徵去與目前的影像做比較。當更新完行人的資訊後，系統會比較原始的特徵與移動特徵，如果兩者差異過大，表示所追蹤的行人已經不是原本所追蹤的行人。如圖 4-30 所示，在圖的左上方有顯示出原始的 Y 平面的特徵，以及移動時候的特徵。這些都是拿來用直方圖比較的依據。

將行人位置上 Y 平面的像素值當作直方圖的統計，但行人在影像中的大小約 16x40 像素不會很大，為避免做出來的直方圖看不出變化，於是我們將原本 8bit 256

階的直方圖，縮成 6bit 64 階的直方圖來做比較，8bit 轉換成 6bit 的直方圖，我們做法是直接捨去最小的 2bit 來運算。可以看到圖 4-31 為移動特徵直方圖，圖 4-32 為原始特徵直方圖。比對的公式為 4-23，其中 H_g 要比對之區域特徵直方圖， H_i 為前一時刻的行人特徵直方圖， m 為 64 筆， j 表是每一階， $H_g(j)$ 為每一階比對之區域直方圖所表示的值， $H_i(j)$ 前一時刻的行人特徵直方圖所表示的值， $f(g,i)$ 為比較出來的結果。首先把 H_g 及 H_i 的直方圖從 1 階到 64 階每一階都去比誰的值比較小，再將這些小的值累加起來，與原本 H_i 直方圖中 64 階每一階的值加總的合，兩者相除。結果會是 $0 < f(g,i) < 1$ ，當 $f(g,i)$ 最接近為 1 時，此特徵為最像的區域。

$$f(g,i) = \frac{\sum_{j=1}^m \min\{H_g(j), H_i(j)\}}{\sum_{k=1}^m H_i(k)} \quad (4-23)$$



圖 4-30 特徵圖

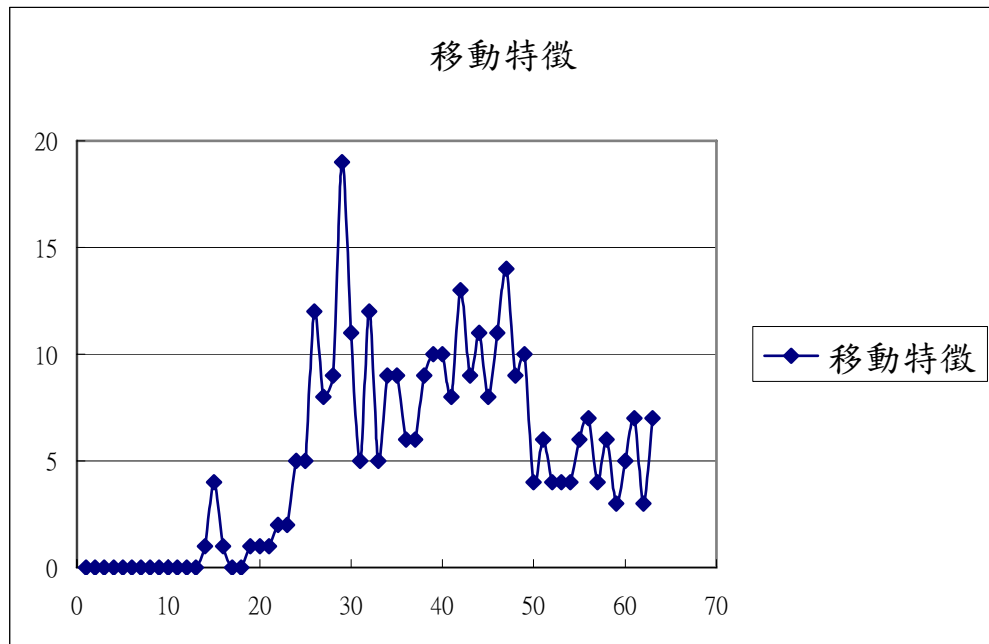


圖 4-31 移動特徵直方圖

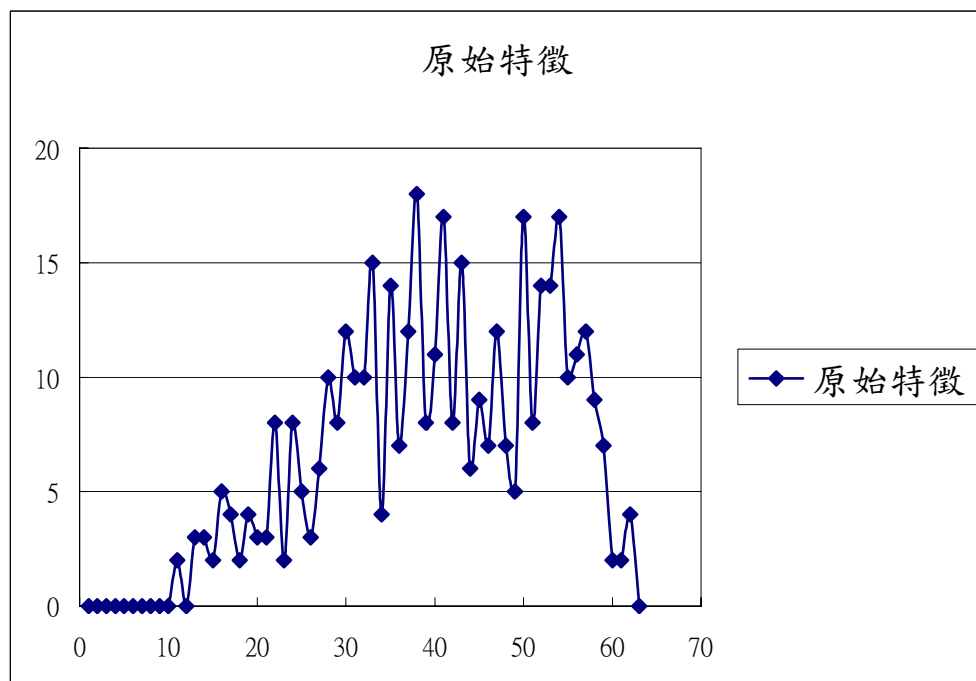



圖 4-32 原始特徵直方圖

4.3 最佳化方法

當在 PC 發展完演算法之後，最重要的是將功能完整的在 DSP 平台去實現，畢

竟 DSP 平台上與 PC 平台還是有很大的差異。於是我們為了能夠順利達成目標，於是針對了幾個重要的步驟來處理，首先重新規劃 DSP 記憶體的位置，將記憶體管理搭配整個程式流程，進而達到良好的記憶體使用。再來使用內部的 EDMA (Enhance Direct Memory Access) 功能，協助當 CPU 在忙碌的時候，快速的搬移記憶體中的資料，減少 CPU 在處理資料所浪費的時間。系統中的 Task 任務安排，除了一般 Suspend 或 Idle 權限較高之外，要能設定程式使用的 Task 權限也是一樣很重要的事，這樣可以加速 CPU 能。最後程式的 Coding Style 是一個很重要的因素，在 PC 平台上可能不要考慮這樣的因素，可是在 DSP 平台同樣的結果不同的程式寫法，經過 Compiler 之後，所執行出來的速度就會大不相同，於是 Optima 本身程式的寫法是一個很重要的課題。

4.3.1 記憶體規劃



在規劃 TMS320DM642 的內部記憶體，我們研究了圖 4-1 整個程式流程的部份，於是我們需要五個主要快速處理的緩衝區(Buffer)。因為如果一般影像的資料放在外部的 SDRAM 的話，做一個水平邊緣或是其它的影像處理，所需要的時間是把資料放在內部的 SRAM 來使用慢上 3~4 倍。於是第一個想法是能把要處理的影像資料放入 SRAM，整體的運算速度可以提升好，可是在 SRAM 的規格只有小小的 256KBbytes，要如何能好好應用是很重要的。在這個 5 個區塊大小的為每一個都是 38.4KBytes 左右如圖 4-33 所示，因為在整個影像之中，我們只針對 ROI 的路口去做影像處理，於是 38.4KBytes 可以放入 352 x 112 大小的影像。這樣接近實際畫面的 1/2，也比規劃的 ROI 大不少。

這 5 個區塊主要為存放背景影像、背景相減後之圖、水平邊緣處理之後的圖及剩下兩個緩衝區為暫存記憶區。如圖 4-33 所示。由此這樣的規劃之後，所有的影像處理的資料都會放 SRAM。處理的速度可以加速不少，不需要太多的 DMA 的動作使用。

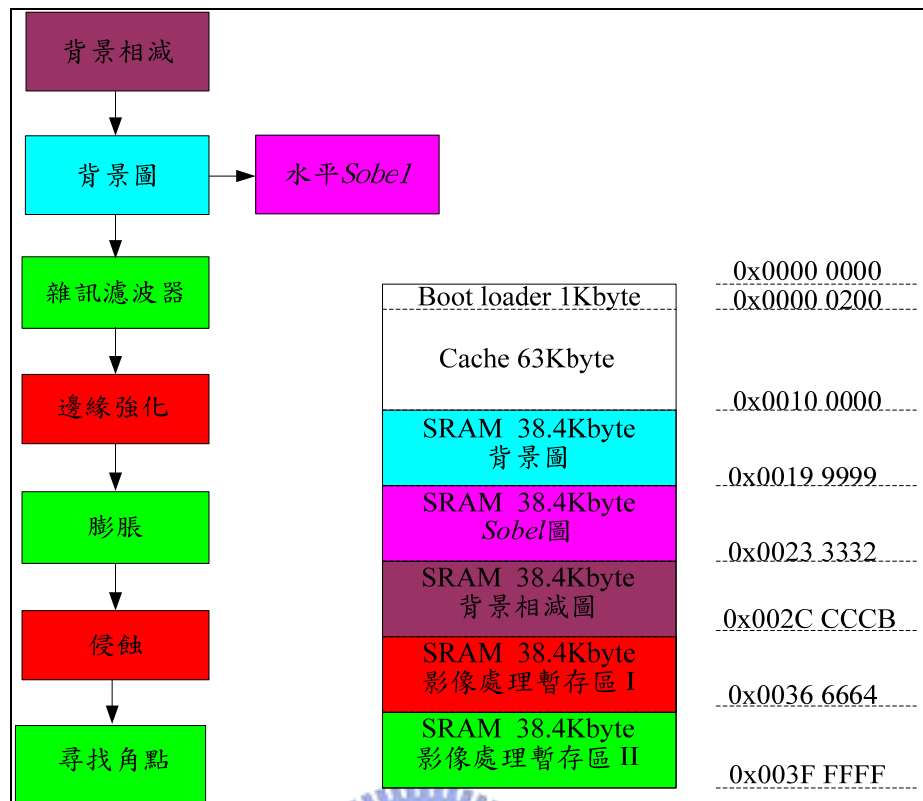


圖 4-33 實際規劃分配記憶體位置

4.3.2 EDMA 的加速

在 TMS320DM642 的內部中，EDMA 有 64 通道(Channel)可以使用，在這部份我們利用其中的幾個通道的 EDMA 功能來加速整體的處理效能。回想圖 3-12 的地方，原本 TMS320DM642 的平台是由外部的視訊利用 Decoder 轉成 CCIR 的介面傳入到 TMS320DM642，傳到 TMS320DM642 之後的資料會用 EDMA 放在外部的 SDRAM 之中，這個緩衝區稱為“Video-In Buffer”。等到這張畫面處理完畢之後，會再利用 EDMA 將處理完之後的畫面放到 SDRAM 中，這個稱為“Video-Out Buffer”之內。然後 TMS320DM642 會用 EDMA 從 SDRAM 搬出，再利用 CCIR 的介面傳給 Encoder 轉成 Composite 的信號給最終的螢幕。

在

圖 4-34 之中，顯示將一張畫面處理的過程是如何 EDMA 的搬移來加速處理。圖中步驟 1、2 的地方，表示視訊進入 VP0 之後，每一段資料都透過 EDMA 傳送到

SDRAM 儲存。當畫面整張儲存完畢之後，進入到步驟 3、4 的地方，也就是把要處理的影像搬到 SRAM 內部處理。經過在內部處理完畢之後，透過 5、6 CPU 的方式將所得到行人資訊寫入到原本 Video In 的地方例如：行人位置、追蹤軌跡或是行人特徵等等。這時候表示所有影像處理的事已經做完了接下來就是要把資料送到 Encoder。TMS320DM642 會將 Video-In Buffer 的資料透過 EDMA 也就是步驟 3、7 的地方，寫到 SDRAM Video-Out Buffer 的位置。最經由步驟 8、9，我們可以從 SDRAM 將處理完的資料經由 EDMA 的快速搬移資料，送到 VP2 的介面，再轉成 CCIR 的訊號給 Encoder，就可接到螢幕輸出了。這就是 TMS320DM642 EDMA 如何幫助整體的執行速度。

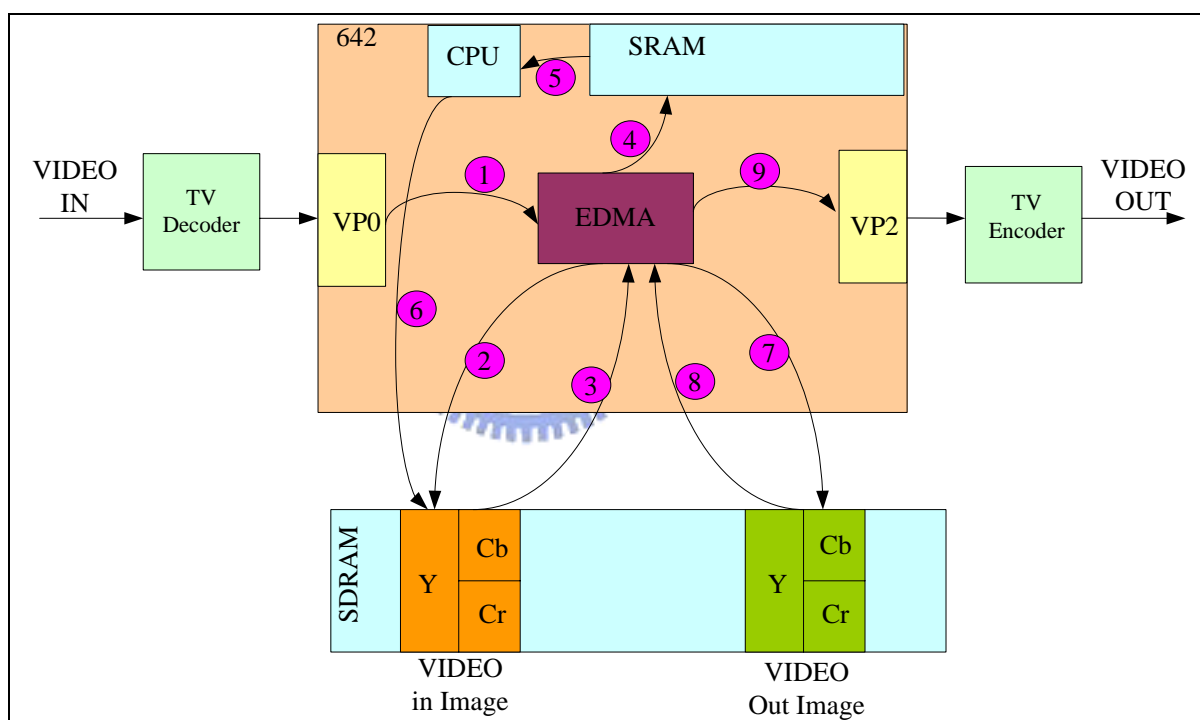


圖 4-34 DMA 的處理流程

在編輯 TMS320DM642 程式中，原本就會提供一個 TSK_main() 的 TASK，讓我們將編輯的程式都會放在其中，另外有 Tsk_idle() 的 task 來放入要進入 Idle state 的程式。為了能加速系統處理 Tsk_main() 的優先順序另外，我們修改了系統中的.CDB 檔。這是一個很重要的檔案，一個可以改變 TMS320DM642 內部相關的設定檔，

可以設定記憶體的管理，發生的 Event 管理及 Task 權限的管理等等。在 TMS320DM642 中，總共有 16 階 Priority 的可以設定。在 TI 提供的 CCStudio 的編輯軟體，可以編輯設定這個檔如

圖 4-35，可以設定 Task 的 Priority，於是我們就將 Tsk_main() 的地方設定到 Priority 15 的地方，讓系統加快的處理 Tsk_main()。另外之前所提出的記憶體配置的設定，也是利用這個 CCStudio 的 Tool 來設定.CDB 檔。讓內部的效率能完整提升。

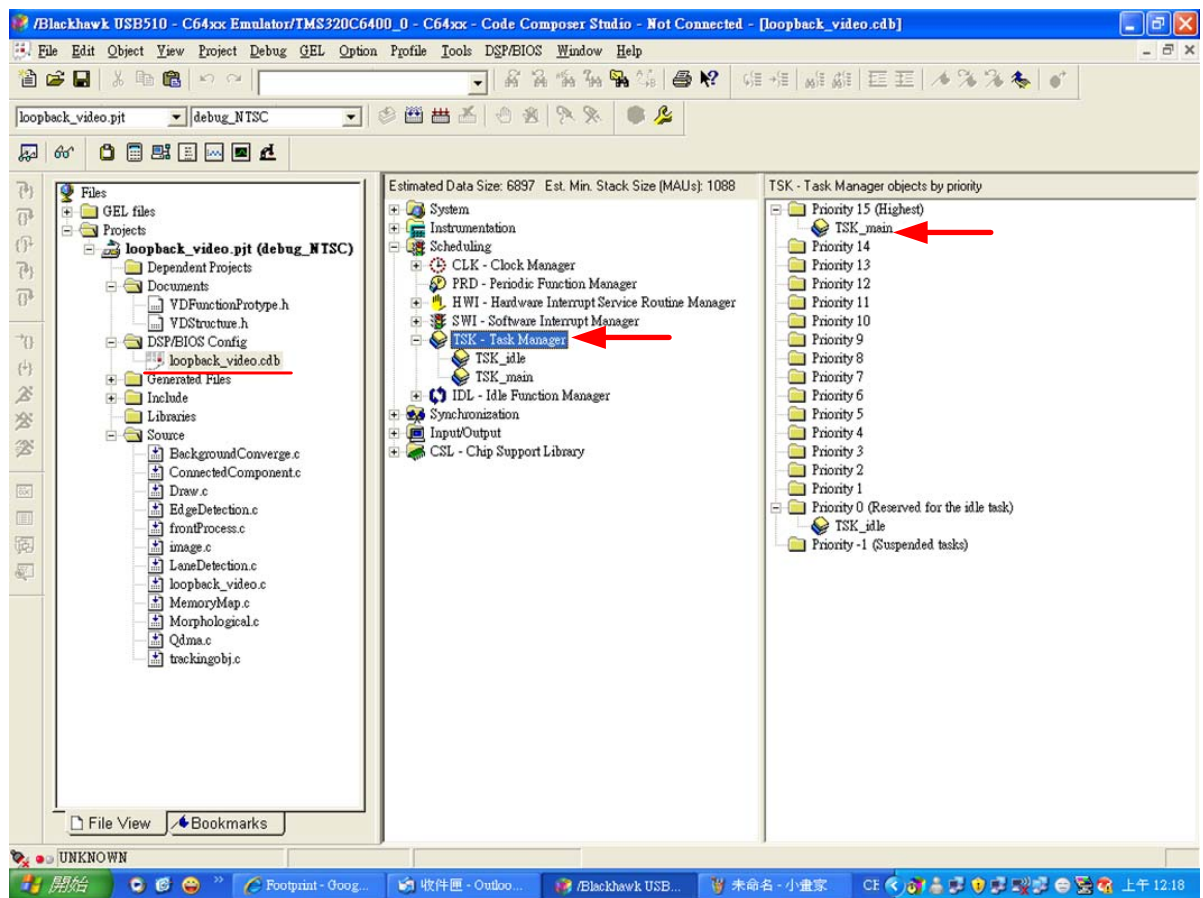


圖 4-35 CCStudio 的設定

4.3.3 程式 Coding Style 的改善

當 PC 上完成的程式要轉換到 DSP 平台的時候，雖然都是 C 的用法，一些看似簡單的寫法，執行之後所耗的時間不見的是最快的，因為經過 Compiler 也是屬於直

譯的方法去變成組合語言。所以當要改變原本的 Coding Style 的話，必需要有寫“組合語言”的精神來撰寫，這樣寫出來的程式在 DSP 上執行出來的結果會有快速的執行結果。在此提出幾個地方也是本論文改進程式撰寫的方法。

首先提出以圖 4-36 的例子，在寫程式之中，常常會用到乘除的功能，如果有乘除一個 2 的幕次方的常數，這時候把程式的寫法就改用 Shifter 的方式來使用。這樣可以加速 CPU 運算的時間。下一個例子如圖 4-37 是常常使用在掃描一個記憶體裡全部像素的值，當裡面的值等於某個值之後，再給與條件式判斷處理。一般會先做完一條水平的再換下一條水平的值。於是當發現這個現象時，我們就將同一條上的像素的位置，只要計算一次，剩下的部份就每次加 1 就可，減少 CPU 計算乘法的運算。最後提出一個在影像處理常見的部份。以圖 4-38 來看，每一個像素必須要參考其它幾個像素值，才會等於本身新算出來的值，像之前所提到的角點運算也是利用這樣的想法來找出那種角點，於是在寫這樣的程式，一般我們會如左邊原本寫法的方式來寫這樣的程式，這樣會比較好看也比較不會出錯。一旦在 CPU 執行的時候，會看到這樣的效率是很慢的。於是我們將程式改寫成右邊的修正寫法，這樣的方式減少 CPU 在做條件式判斷花費的時間，系統處理上會更加的有效率。

原本寫法	修正寫法
$A = C + B / 2 ;$	$A = C + (B \gg 1) ;$
$A = C + B * 2 ;$	$A = C + (B \ll 1) ;$

圖 4-36 Coding 範例(1)

原本寫法	修正寫法
<pre> For(j= ROIY0 ; j< ROIY1 ; j++) { For(i= ROIX0 ; j< ROIX1 ; j++) { A = *(pin+ j * ImageHsize + i); if(A==0x255) { } } } </pre>	<pre> For(j= ROIY0 ; j< ROIY1 ; j++) { jSize = j * ImageHsize; For(i= ROIX0 ; j< ROIX1 ; j++) { A = *(pin+ jSize + i); if(A==0x255) { } } } </pre>

圖 4-37 Coding 範例(2)

```

原本寫法

if(( A==0 ) && (B==0) && (C==255)&&(D==255))
{ w=100;}
if(( A==255) && (B==0) && (C==255)&&(D==0 ))
{ w=101;}
if(( A==0 ) && (B==0) && (C==255)&&(D==0))
{ w=102;}

修正寫法

if( (B==0) && (C==255) )
{
    if(( A==0) )
    {
        if (D==0)
        { w=102;          }
        else
        { w=100;          }
    }
    else
    {
        if ( D==0)
        { w=101;          }
    }
}

```

圖 4-38 Coding 範例(3)

以上這些 Coding 的方法，雖然看起來寫法沒有改變很多，實際幫助會很大的，以範例(2)來說，原本要做一張 352x288 圖的計算，以原本的寫法需要做上 101376 次的乘法，使用修正的寫法只要做上 288 次的乘法。整體處理時間可以少上 101024 次的乘法如公式 4-24，假設以 133MHz 來計算，每一個乘法需要 2 個週期來完成於是可由公式 4-25 計算出可以節省的運算時間有 1.52mSec 左右。程式中少做乘法的運算可以加快影像處理的速度，由此得知 Coding Style 是能夠讓 DSP 執行效率提高很重要的因素。

$$\text{原本寫法次數} - \text{修正後寫法的次數} = (352 * 288) - 288 = 101024 \quad (4-24)$$

$$\begin{aligned} \text{節省之後的運算時間} &= (\text{SRAM 時脈周期}) * 2 * (\text{增加乘法次數}) \\ &= (1/600\text{MHz}) * 101024 * 2 = 0.032 \text{ mSec} \end{aligned} \quad (4-25)$$

第五章 實驗結果

本實驗主要是在中華大學的校門口路口，在此路口架設攝影系統，以此作為實驗場景，利用我們所開發的系統架構，來測試本系統。並記錄實驗結果。

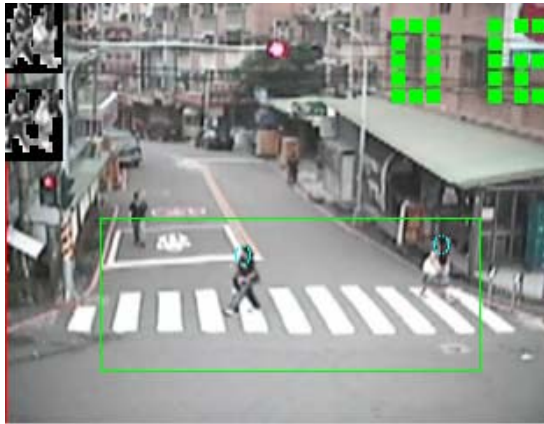
5.1 實作成果

本論文實驗的部份主要分成四個階段，第一部份是頭部的角點偵測，此實驗將驗證所提出的方式，是否可以利用角點的特性來偵測到行人頭部。第二部份是行人追蹤實驗，比較不同預測的方式，當目前畫面沒有行人候選人，前一張畫面有行人追蹤的資訊，要如何能正確預估行人所在新的位置是本部份的重點。第三部份是直方圖比較法，也就是當預估行人新的位置之後出，能更精確的找出正確行人位置。第四部份主要是提出驗證當兩個行人在畫面中突然有重疊情況，是否能夠持續追蹤判斷行人的路線。



5.1.1 頭部角點偵測

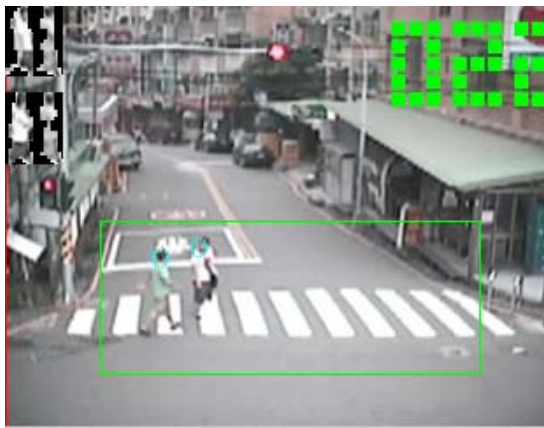
頭部角點偵測的實驗，主是在於連續影像上，先利用背景相減法，取出前景，並且二值化結果。接下來影像經過邊緣強化、侵蝕及膨脹的影像處理，最後搜尋方法以之前定義出角點搜尋方法來找出頭部的相對位置，來確認是有否找到影像中行人的頭部特徵。而且加上水平邊緣及身體密度的檢視，可以讓我們可以清楚找出行人頭部的地方。如果只有角點的搜尋方法，有可能會找到不是行人頭部的位置。當加上水平邊緣判斷及身體密度檢查，把一些會發生不必要的錯誤判斷加以處理，達成正確判斷行人的方法，可以看到結果如圖 5-1 (a)。圖 5-1(b)是(a)的放大圖，有此圖來看我們可以清楚的將行人頭部找到並且給與記號表示。圖 5-1(c)(e)(g)也是表示不同的行人，利用角點偵測的方法也可以正確的找出行人頭部位置。



(a)角點偵測判斷正確圖(一)



(b) 圖(a)局部放大圖



(c)角點偵測判斷正確圖(二)



(d) 圖(c)局部放大圖



(e)角點偵測判斷正確圖(三)



(f) 圖(e)局部放大圖



(g)角點偵測判斷正確圖(四)

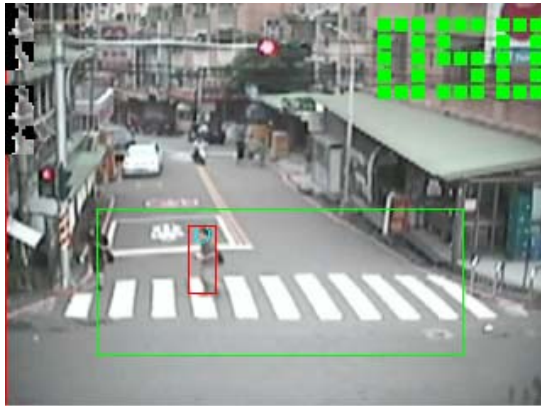


(h) 圖(g)局部放大圖

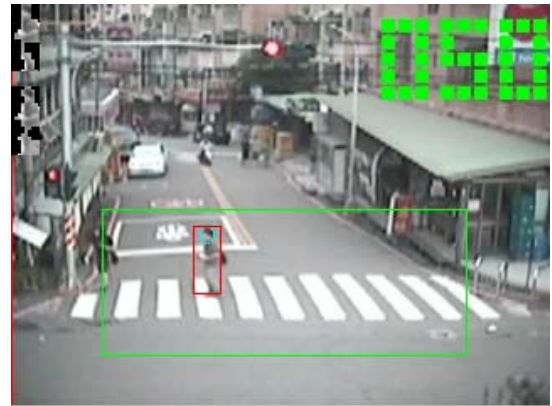
圖 5-1 角點偵測判斷正確圖

5.1.2 行人追蹤預測

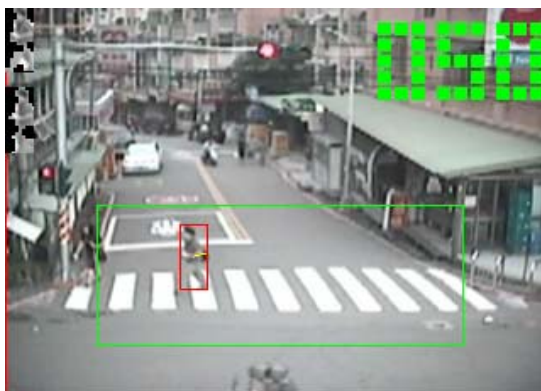
在行人追蹤實驗中，在每一個影像畫面不見的可以判斷出行人的所在位置，可能是被其它物體稍微遮到，或是身體的顏色與背景顏色很相近。但由前一張畫面的資訊可以得知行人還是在畫面的監視區，並沒有突然消失，預測行人所在的位置。預測行人軌跡就是很重要的工作，如何能夠確保所預測是很精確的，我們提出的方法是參考前四張畫面行人的所在位置，利用這幾張的統計位置相差的距離。來預測下一張畫面行人應該所在的位置。如圖 5-2(a)(b)來看，這時候系統有判斷出畫面上有偵測到行人候選人，根據之前追蹤的法則，此行人候選人的位置就是目前這個行人這張畫面的所在位置。接下來圖 5-2(c)(d)畫面，這時候畫面上無法找到正確的行人候選人，系統就會使用預測的方式來預測行人新所在的位置。當到了圖 5-2(e)(f)畫面的時候，行人追蹤還是可以保持正確的結果，所以使用四張畫面的參考資訊，可以讓預測能夠更加順利的進行，較少的機會錯誤。於是我們就使用這樣的方法來做為行人路徑預測的方法。



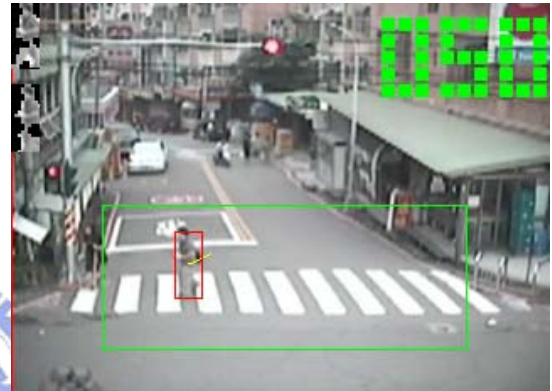
(a) 正確追蹤第 1033 張畫面



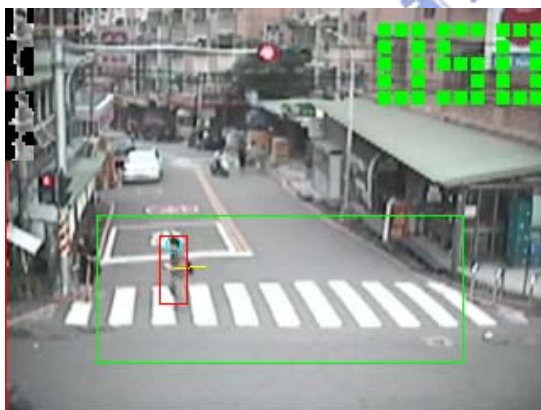
(b) 正確追蹤第 1035 張畫面



(c) 正確追蹤第 1041 張畫面



(d) 正確追蹤第 1046 張畫面



(e) 正確追蹤第 1048 張畫面



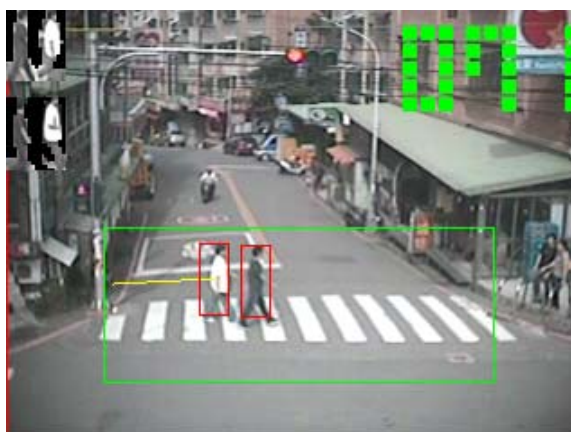
(f) 正確追蹤第 1049 張畫面

圖 5-2 正確追蹤畫面

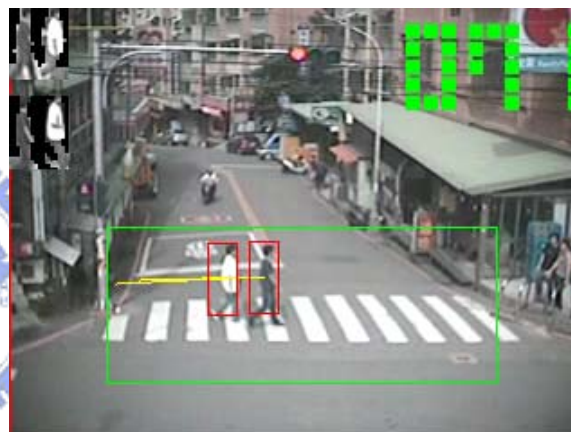
5.1.3 直方圖比較法

在整個行人追蹤中，使用直方圖比較法是一個很重要判斷影像是否相似的方

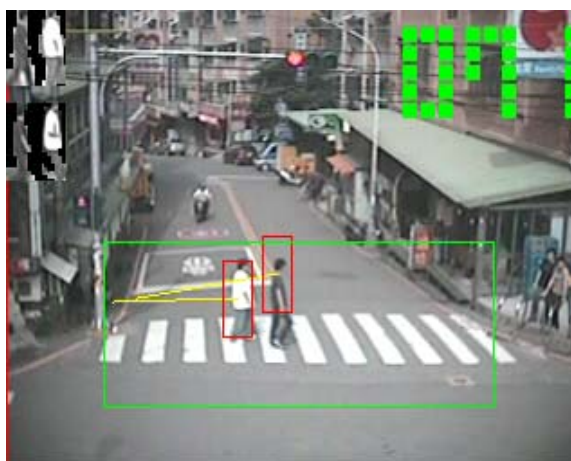
法。以本論文提出的行人追蹤方法，當行人追蹤預測路徑之外，在預測之後的位置附近找尋更相似的被追蹤行人的位置，是讓整體的正確性能夠更加提升。在一開始實驗中我們使用了 Pattern Matching 的演算法來找尋最接近位置，它是拿來比較兩個區塊的相似度。實驗出來的結果如圖 5-3(a)到(d)，當使用 Pattern Matching 的時候，它所運算出來的位置會偏離原本行人所走的路徑，造成追蹤上的錯誤。於是我們發現當要比較行人相似的方法，使用 Pattern Matching 方法易受到外界雜訊影響而追蹤錯誤。於是我們改用了直方圖的比較方法，找出相似行人的地方。如圖 5-4(a)到(d)所示，使用直方圖來比較影像相似的方法，可以幫助追蹤行人得到一個很好的結果，並不會找到不接近的位置。



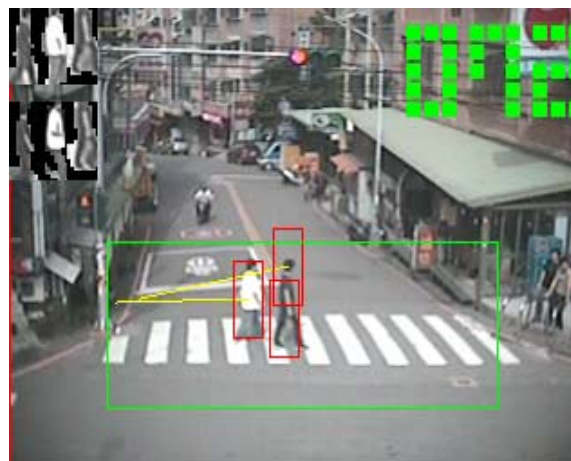
(a) Pattern Matching 方法第 20138 張



(b) Pattern Matching 方法第 20141 張

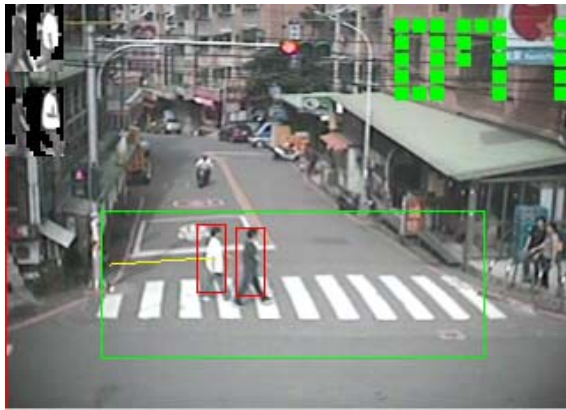


(c) Pattern Matching 方法第 20150 張

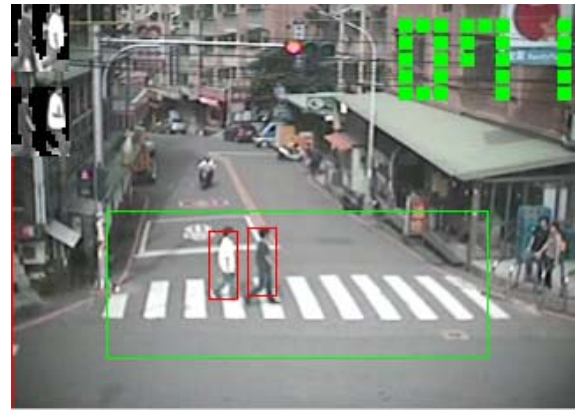


(d) Pattern Matching 方法第 20154 張

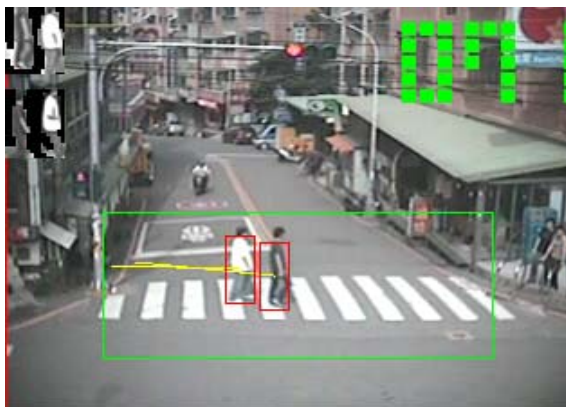
圖 5-3 使用 Pattern Matching 方法



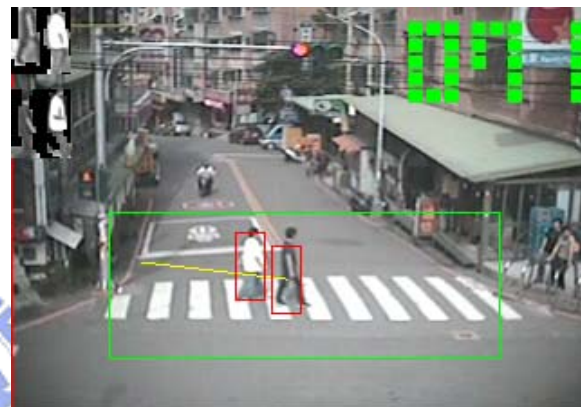
(a) 直方圖方法第 20138 張



(b) 直方圖方法第 20141 張



(c) 直方圖方法第 20150 張



(d) 直方圖方法第 20154 張

圖 5-4 直方圖比較法

5.1.4 行人重疊

在行人偵測追蹤，常常會看到在視訊畫面中，行人彼此之間會有交疊的現象。可能是某一個人的腳與某一個行人的頭部疊在一起。或是彼此之間走的比較近，這樣都會是讓我們處理影像上會比較麻煩的事情。在實驗中為了能夠好好處理這樣的問題，如之前所提到的使用直方圖來找尋相似度高的地方。但是當重疊的時候，我們還是使用直方圖來做判斷，但把重疊的區域不要加入計算行列，這樣剩下的區域就是跟原本彼此的圖很相近的地方。以為例子當兩個人一個由右往左走，另一個由左往右走，當交會的時候其中一個人的頭部就會被遮住。於是部份區域直方圖比較會是一個很有效的方法。以圖 5-5 來看，圖 5-5(a)(b)畫面出現兩個人，一人往左一人往右。

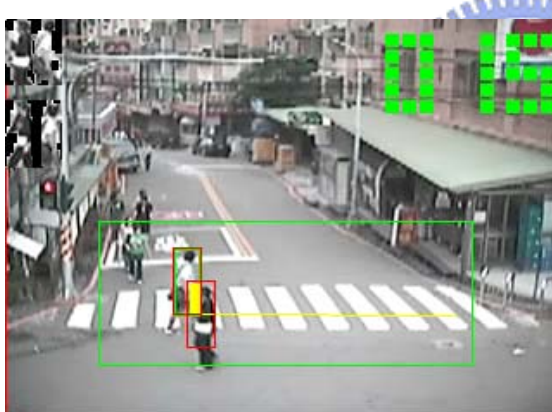
當在圖 5-5(c)的時候開始交疊，經過了圖 5-5(d)(e)之後。在圖 5-5(f)兩者分開之後，追蹤就跟不上原本的行人了。圖 5-6 (c)(d)上顯示，當交疊發生的時候，我們特別用黃色區塊表示。在直方圖比較法上把黃色重疊區域忽略不計，產生出來的結果如圖 5-6 (f)所看到的，系統程式還是可以持續的追蹤行人，直到離開監視的範圍。



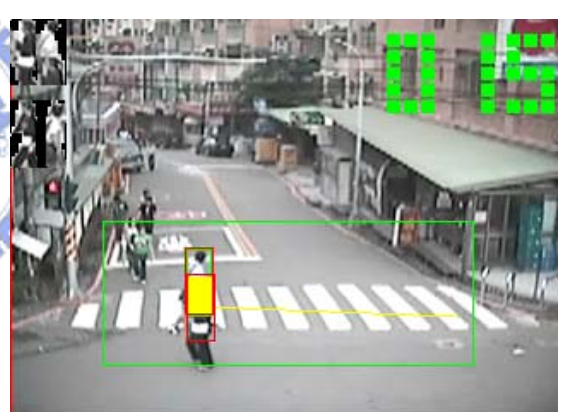
(a)行人重疊第 1104 張



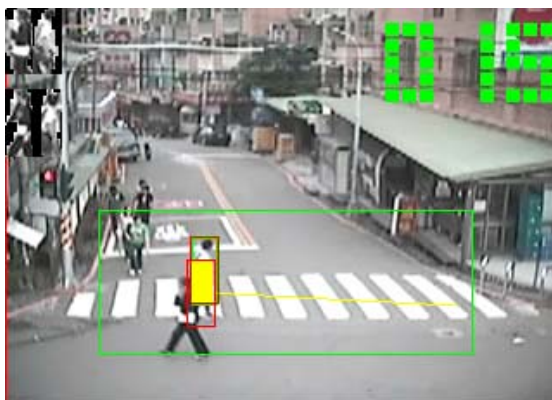
(b) 行人重疊第 1125 張



(c)行人重疊第 1130 張



(d) 行人重疊第 1133 張



(e)行人重疊第 1136 張

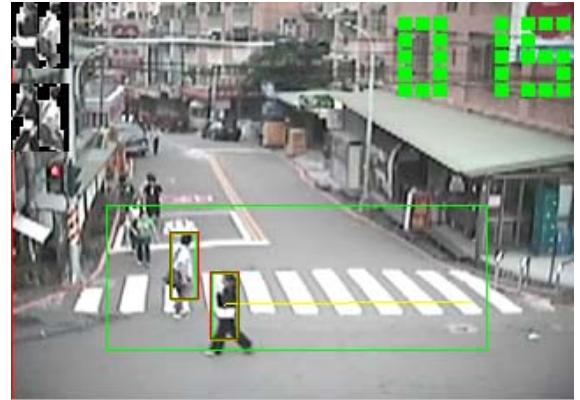


(f) 行人重疊第 1139 張

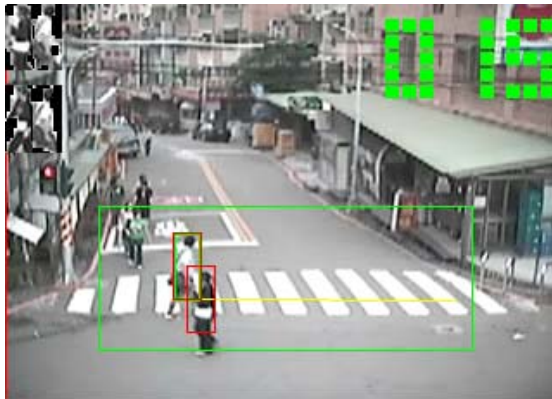
圖 5-5 行人重疊全區域比較法



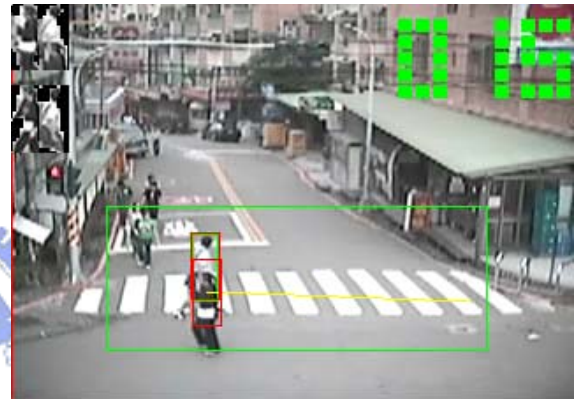
(a)行人重疊第 1104 張



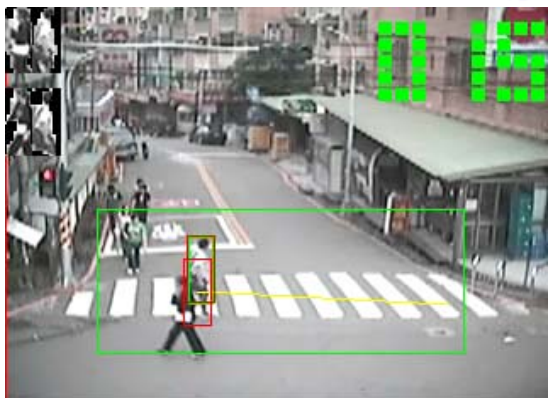
(b) 行人重疊第 1125 張



(c)行人重疊第 1130 張



(d) 行人重疊第 1133 張



(e)行人重疊第 1136 張



(f) 行人重疊第 1139 張

圖 5-6 行人重疊部份區域比較法

5.2 系統效率分析

5.2.1 偵測率及正確率

本論文實驗使用的影像為 352x288 CIF 的視訊影像，以約六分鐘的實際路口影像，輸入到所製作的 TMS320DM642 DSP 平台，偵測行人並且追蹤行人的路徑。將所看到的結果統計如表 5-1，畫面總共出現了 71 人，利用本論文提出的方法，可以偵測到行人的數量為 62 人，沒有偵測到的人數有 9 人。在正確判斷人數上，62 人之中我們可以判斷出正確的 50 人，錯誤判斷有 12 人，可能是跟蹤失敗或是車子判斷成車子。依照我們所提出的實驗數據，我們計算整個系統所產生出來的偵測率及正確率如公式(5-1)(5-2)。計算結果如表 5-2 偵測率為 87.3%，正確率有達到 80.4%，兩者都有達到 80%的水準。

表 5-1 實驗人數統計

	偵測到人數	沒偵測到人數	正確判斷人數	錯誤判斷人數	總人數
人數	62	9	50	12	71

$$\text{偵測率} = (\text{偵測到人數}) / (\text{總人數}) \quad (5-1)$$

$$\text{正確率} = (\text{正確判斷人數}) / (\text{偵測到人數}) \quad (5-2)$$

表 5-2 偵測率與正確率

	偵測率	正確率
比率	87.3%	80.4%

5.2.2 平台運作速度

平台運作速度是一個很重要的關卡，當系統處理影像的速度太慢的話，會造成影像處理完的結果，所呈現出來的畫面會有一格一格的跳動，人眼會覺得畫面很不流暢，無法達到即時的效果。加快平台速度是本論文另一個的目標，當我們將在原本電

腦上的演算法程式，移植到 DSP 平台上來執行。平台上的記憶體位置需要規劃除了 DSP 的時脈與 PC 的 CPU 低之外，加上因為 DSP 系統上的記憶體空間比電腦上的記憶體空間少了許多，一旦沒有好好的規劃，系統跑起來的速度會很慢，程式的最佳化於本嵌入式偵測系統是一件很重要的工作。以表 5-3 所示，在還沒有最佳化前一張畫面的處理，經過背景相減、雜訊濾除、邊緣強化、水平邊緣、膨脹侵蝕、角點偵測、人頭偵測、輸入影像傳送到輸出影像及最後的行人追蹤。整張畫面經過這些流程需要跑上 83mSec 的時間，經過前一章節所提到的最佳化方法。這樣一張畫面所需要的處理時間可以降到 39.6mS 的時間，也就是說平均一秒鐘有約 25.2 張的影像畫面可以處理，在人眼看起來也不會感到有延遲的現象。

表 5-3 TMS320DM642 一張畫面處理時間

	最佳化前	最佳化後	效能提升
背景相減	4	2.9	27.50%
雜訊濾除	4	2.7	32.50%
邊緣強化	6	2.9	51.67%
水平邊緣	17.8	4.9	72.47%
膨脹	6.9	2.6	62.32%
侵蝕	3.9	2.6	33.33%
角點偵測	25.6	4	84.38%
人頭偵測	7.8	5	35.90%
輸入視訊影像 傳至輸出視訊	7	5	28.57%
行人追蹤	7	6.2	11.43%
全部總和 (mS)	83	39.6	52.29%

在實驗之中，本論文提出的最佳化的方式之中，改善效果最多的是將程式的 Coding Style 改善，這樣可以達到最有效的結果，大約可以為整體提高效能方法的 80%，剩下

EDMA 及記憶體配置約改善了 20% 的效能方法。

5.3 錯誤分析

在實驗中，會看到一些錯誤的情況。當發生這些情況的時候，也就是發展的演算法不夠完整所造的。一開始當要發展偵測到行人的時候，我們提出以角點的方式來去實驗，實驗之中是可以發現行人頭部可以被程式所找到，但也發現程式也找到一些不相干的位置如圖 5-7 所示，畫面上可能有很多程式誤認為行人的頭部的地方。於是我們發現這樣的問題，在判斷程式中加上了水平邊緣、身體密度的判斷，增加程式判斷的可靠度。



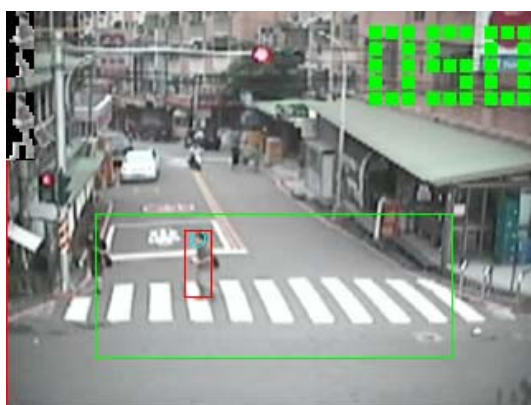
(a) 角點偵測多處判斷錯誤圖(一)

(b) 角點偵測多處判斷錯誤圖(二)

圖 5-7 角點偵測多處判斷錯誤圖

在偵測上能夠確認程式已經找行人的時候，相對於追蹤也是一個很重要的課題，這樣就可以掌握行人的移動方向。在追蹤上，如果這時候的畫面有找到行人候選人，利用直方圖及重疊比較法來確認是否跟目前追蹤的行人有很大的關聯，是的話就把相關位置資訊更新。如找不到合適行人候選人的時候，預測行人新的位置是一個重要環結，實驗之中我們首先參考前兩張畫面的行人位置差，來做為預測新的位置。圖 5-8 中(a)的畫面在行人中有藍色的圈表示程式有判斷到行人的頭部，有紅色的框表示目前正在持續追蹤這個行人。在圖 5-8(b)的畫面也持續有判斷到行人的頭部，也持續在追蹤。等到圖 5-8 (c)的時候，沒有抓到行人的頭部，也就是要使用預測的方式，

這時候看起來還好紅色框的位置沒有差很遠。到了圖 5-8 (d)(e)的時候預測的軌跡開始往別的方向了，紅色的框也會跑錯地方。最後到了圖 5-8 (d)同一個行人又被偵測到了，原本的追蹤紅色框也會繼續預測，於是這張畫面就產生出兩個行人追蹤的紅色框。當發生這樣的情況，我們在實驗上將原本預測方法改成參考前四張畫面行人位置差，因為行人走路的時候不見的都是會直直的走，也有可能一下快一下慢。除了增加為四張畫面的處理之外，我們並增加了直方圖比較法來找出更精確行人所在的位置，這樣的改進可以讓程式追蹤行人上可以更加正確。



(a) 錯誤追蹤第 1033 張畫面



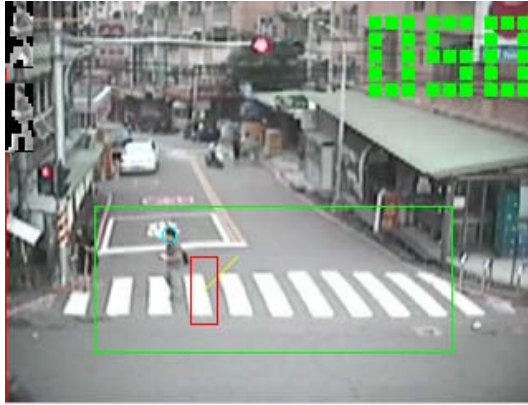
(b) 錯誤追蹤第 1035 張畫面



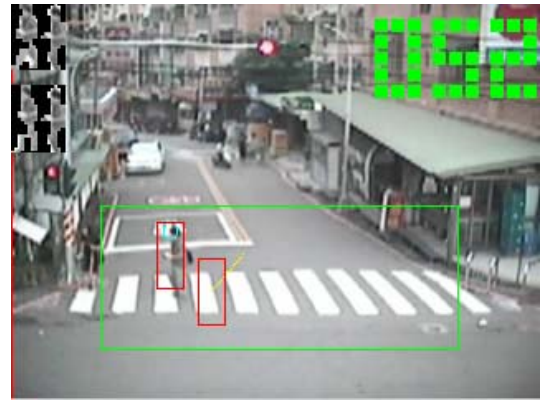
(c) 錯誤追蹤第 1041 張畫面



(d) 錯誤追蹤第 1046 張畫面



(f) 錯誤追蹤第 1048 張畫面



(f) 錯誤追蹤第 1049 張畫面

圖 5-8 錯誤追蹤圖

5.4 實驗結果分析

針對論文提出的處理流程，我們做了其它相關理論的驗證。證明本論文提出來的方法的可行性是較高以及產生出來的結果也比較好。於是第一個部份所做的實驗，是將在行人偵測流程之中，邊緣強化做一個比較，比較的方法就是整個行人偵測流程有去執行邊緣強化，另外一個方法就是沒有執行邊緣強化，兩者之間去做比較。另外一個實驗是當背景收斂的方法，本論文提出的是一個統計方法採用相同臨界值在整張畫面上處理，並針對使用自適應閾值算法(Adaptive Threshold)的方法彼此之間的差異。

5.4.1 邊緣強化實驗

實驗中，我們將圖 4-1 的偵測流程更改變動，原本做完雜訊清除之後，會做邊緣強化，然後再做膨脹。我們比較整個偵測流程中，執行邊緣強化與不執行的結果，分析邊緣強化的好處以及它可以得到的優點。以實驗的影像發現，我們將原本 4 個像素的點，只要一個為 1 就能自動填滿 4 個像素皆為 1，這樣強化之後可以將頭部特徵明顯的表示，從圖來看可以看到容易找到行人頭部的左上右上角的角點。

沒有做過邊緣強化的圖，頭部的特徵沒有做過邊緣強化的如此明顯是一個完整頭部的大小，且沒有經過邊緣強化的圖形並不容易可以找到正確的頭部左上及右上角的角

點位置。

邊緣強化處理

無邊緣強化處理



(a)邊緣強化第 3385 張畫面



(b)無邊緣強化第 3385 張畫面



(c)邊緣強化第 3984 張畫面



(d) 無邊緣強化第 3984 張畫面



(e)邊緣強化第 4091 張畫面



(f) 無邊緣強化第 4091 張畫面

圖 5-9 邊緣強化比較圖

5.4.2 自適應閾值算法實驗

在本論文中提到的前景與背景相減產生出來的物件的方式，我們是採取一個固定的閾值，計算出移動的物體。在實驗中背景中會有一些亮度較低的區域，當在這些區域使用固定閾值所分離出來的物件會比較沒有這麼明顯。於是我們對這樣的結果，採用了自適應閾值算法統計，來改善目前遇到有時前景減不出來的情况。在實驗之中，我們考量這樣的想法及理論是否可以在 DSP 上運作，於是我們將 Y 平面劃分成 16 個區段，當收斂完背景之後，我們採取 6000 張的前景與背景相減統計值計，每一段 Y 都統計前景與背景相減之值的直方圖。當統計出來之後，我們將數據描繪成圖如圖 5-10。因為大部份的畫面都是存在的是背景的情况，於是這樣統計出來的值，大部份都是背景圖的情况，於是我們定 90% 的地方為一個統計標準界限。Y 由 0 開始累積出來的值為 90% 對應的 Y 值，於是我們就可訂出這一段 Y 的閾值，其設定值如表 5-4 自適應閾值設定表。相對於其它 15 段，也是這樣完成。於是我們就有整段適應閾值，相對於在背景比較暗的地方，經過閾值的計算，也可以精確的把分離移動物體。

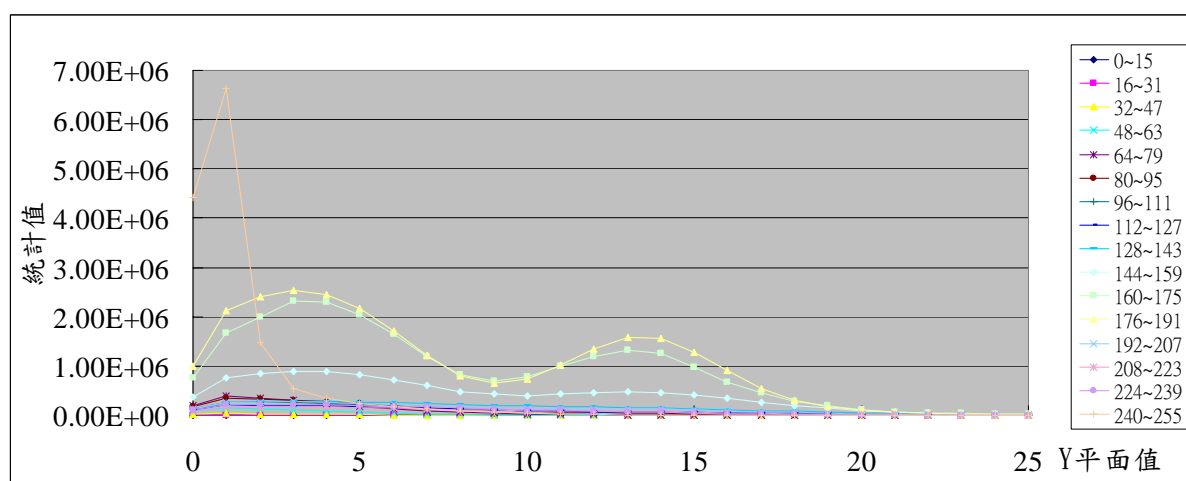


圖 5-10 自適應閾值統計圖

表 5-4 自適應閾值設定表

灰階區域範圍	閾值設定值	灰階區域範圍	閾值設定值
0~15	0	128~143	17
16~31	6	144~159	19
32~47	9	160~175	16
48~63	7	176~181	15
64~79	8	182~197	17
80~95	12	198~213	17
96~111	16	214~239	15
112~127	18	240~255	4

在表 5-4 之中，位於 0~15 灰階範圍，設定值為 0，主要原因是在背景並沒有灰階為 0~15 的像素才會這樣的結果。在圖 5-11 之中，大部份背景像素灰階都集中分佈於 100~200，灰階比較小就是畫面中比較陰暗的角落，所以相減差值都比較小，因此累計到 90% 所得到的閾值會比較小，是我們所期望的也是比較合理的。而分佈於中間的灰階也是畫面中路面部份，因為行人和汽機車通過所相減出的差值較大，所以對應的閾值也較大，白色班馬線因為灰階亮度都趨近飽和，所以相減差值小，得到的閾值也較小。

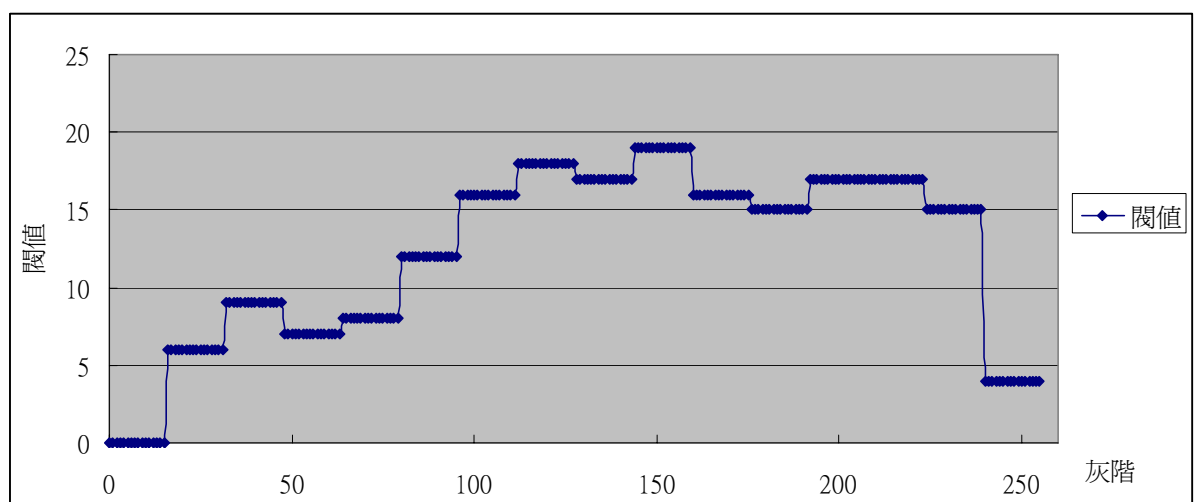


圖 5-11 自適應閾值趨勢圖

然後我們去做個比較性的實驗，分別使用固定閾值及自適應閾值的設定。在圖 5-12 之中，可以看到如果使用固定閾值所減出來的前景，行人的輪廓在某些部份會比較不明顯，使用自適應閾值的方法，減出來的前景，行人整體的輪廓會更明顯變認。於是我們在相同的影像畫面，比較及記錄那一種閾值方式在路口行人偵測上可以得到比較好的效果。如

表 5-5 所示，我們統計了一段時間，總共 67 人通過這個路口，由結果看出使用自適應閾值會有得到比較好的結果。

使用固定閾值

使用自適應閾值



(a)固定閾值第 2298 張畫面



(b)自適應閾值第 2298 張畫面



(c)固定閾值第 6230 張畫面



(d)自適應閾值第 6230 張畫面

圖 5-12 使用固定閾值和自適應閾值前景圖

表 5-5 閾值比較結果

	偵測到人數	沒偵測到人數	正確判斷人數	總人數
固定閾值	58	9	47	67
自適應閾值	59	8	51	67



第六章 結論

6.1 總結

在實驗觀察中，當畫面上出現多人以上的情形，這時候利用提出來的的方法，可以擷取出來這些行人的位置，亦發現了當行人重疊現象會影響到我們的追蹤行人的判斷，所以也提出利用直方圖比較法來解決了遮蔽的問題，實驗證明，我們的方法有相當高的準確性。然而在實驗中我們的方法之中，我們考慮行人重疊的情況，主要是兩個行人一開始進入畫面是沒有重疊，因為行人行走的速度不同，而會產生出重疊現象。對於一開始出現兩個行人就已經在畫面上重疊的情形，並且重頭到尾的畫面都是重疊情形，目前本論文提出的方法還無法解決這樣的問題；還有我們是以角點來辨識的方式來辨識行人，當在路口上通過的車輛在某些情況，也會造成角點判斷錯誤的時候，我們就沒有加以辨識，區分車子與行人，這點也有待加強改進。在場景之中，我們所使用的鏡頭是固定的，不會有伸縮鏡頭的效果，且場景是為固定的，所以當行人頭部角點的偵測上，可以維持固定的偵測比例，達到本論文偵測行人的結果。當變動場景的時候，需要將頭部特徵及行人特徵比例上最修改，可以完成不同場景所偵測行人的效果。

本論文發展出行人偵測系統模組，架設在路口穿越道上擷取路口行人影像做為實驗方式。結合著整體電路上設計與制作，力求模組成本低廉，體積微小，功能強大，適合放置各種場地及空間。並努力將系統的程式最佳化，內外記憶體理想的配置，建立一套快速處理影像的平台。讓人們的視覺感到影像有即時處理的效果，並且提出行人頭部的角點特性來偵測路口上之行人，所用的方法為行人在空間上的特點。並配合其它簡單的影像處理方式，讓 DSP 執行程式上，能夠快速且準確地將畫面中的行人擷取出來，不失去影像即時的效果。行人追蹤上，採用直方圖的比較法及路徑的預測，

讓追蹤行人達到基本的正確性。整體來看，本論文發展出來的行人偵測系統模組，在執行路口行人偵測的效率和結果都令人覺得很滿意，並證明了本論文實現了一套完整的行人偵測系統模組方案。

6.2 未來展望

雖然在實驗中偵測的效能得到不錯的成果，但正確率也是未來將試著改進的地方。未來可以增加影像辨識的技巧如像樣本比對方法等等，以彌補本系統判斷錯誤的不足之處。並試著找出 DSP 系統平台在整體規劃上，能否找出更好的執行效率的方法。這樣平台上能使用更複雜的演算法，來提升整體行人偵測的正確率。在系統平台上，目前聲音功能的部份，我們沒有去整合聲音與影像的功能，因為影像處理是目前比較重要的課題，等待某個程度之後。可以增加有關語音與行人偵測相關的應用。另外如網路傳遞即時偵測結果影像等也是可以增加整體系統的應用範圍，提升行人偵測的價值。本平台除了應用在交通路口上，也可以移植到其它場所來使用，如平交道等，可以為行人安全上多一份保護，也是本論文所要完成的目標。

文獻參考

- [1]C. Papageorgiou, and T. Poggio, “Trainable pedestrian detection,” *Proc. of ICIP*, pp. 25-28, 1999.
- [2]M. Oren, C. Papageorgiou, P. Sinha, E. Osuna, and T. Poggio, “Pedestrian detection using wavelet templates,” *IEEE Proc. on Computer Vision and Pattern Recognition*, pp. 193-99, 1997.
- [3]M. A. Sotelo. I.Parra, D. Femandez. E. Naranjo “Pedestrian Detection using SVM and Muti-feature Combination” *IEEE Trans. on Intelligent Transportation System Conference*, MA4.2, no.103-108,2006.
- [4]C. Wöhler, and J. K. Anlauf, “Real-time object recognition on image sequences with the adaptable time delay neural network algorithm – applications for autonomous vehicles,” *Image and Vision Computing*, vol. 19, no. 9-10, pp.593-618, 2001.
- [5]L. Zhao, and C. E. Thorpe, “Stereo- and neural network-based pedestrian detection,” *IEEE Intelligent Transportation Systems*, vol. 1, no. 3, pp.148-154, 2000.
- [6]U. Franke, D. Gavrila, S. Görzig, F. Lindner, F. Paetzold, and C. Wöhler, “Autonomous driving goes downtown,” *IEEE Trans. on Intelligent Transportation System*, vol. 13, pp. 40-48, 1998.
- [7]L. C. Fu, and C. Y. Liu, “Computer vision based object detection and recognition for vehicle driving,” *IEEE Proc. on Robotics & Automation*, pp. 2634-2641, 2001.
- [8]C. Curio, J. Edelbrunner, T. Kalinke, C. Tzomakas, and W. V. Seelen, “Walking pedestrian recognition,” *IEEE Trans. on Intelligent Transportation System*, vol. 1, no. 3, pp. 155-163, 2000.

- [9]H. Mori, N. M. Charkari, and T. Matsushita, "On- line vehicle and pedestrian detections based on sign pattern," *IEEE Trans. on Industrial Electronics*, vol. 41, no. 4, pp. 384-391, 1994.
- [10]S. Yasutomi, H. Mori, and S. Kotani, "Finding pedestrians by estimating temporal-frequency and spatial-period of the moving objects," *Robotics and Autonomous Systems*, vol.17, pp. 25-34, 1996.
- [11]I. Haritaoglu, D. Harwood, and L.S. Davis, "Hydra: Multiple People Detection and Tracking Using Silhouettes,"*Proceedings of the Second IEEE Workshop on Visual Surveillance*, pp. 6-13, June 26,1999.
- [12]C.R. Wren, A. Azarbayejani, T. Darrell, and A. Pentland, "Pfindex: Realtime Tracking of the Human Body," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol 19,780-785,1997.
- [13] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, Addison-Wesley Publishing Company , 1989.
- [14]R.C. Gonzalez and R. E. Woods, *Digital Image Processing*, Upper Saddle River,Nj: Prentice Hall,2002
- [15]Freeman, H. and Davis, L.S. "A Corner-Finding Algorithm for Chain-Coded Curves," *IEEE Transactions on Computer*, C-26(2), 297(1977).
- [16]C. Harris and M.J. Stephens. "A combined corner and edge detector. " *In Alvey Vision Conference*, pages 147–152, 1988.*Vision, Graphics and Image Processing*, 20(1),58(1982).
- [17] Rosenfeld, A. and Weszka, J.S., "An Improved Method of Angle Detection on Digital Curves," *IEEE Transactions on Computers*, C-24(9), 940(1985).
- [18]林文祥, "以 dsp 實現移動物體影像的即時偵測與追蹤系統" 國立台灣科技大學, 電機工程系, 2006.

- [19]顏睿余， ” 高性能 DSP Based 影像擷取辨識系統” 國立中山大學，電機工程學系，2002
- [20] Texas Instruments Inc, ”TMS320DM642 Technical Overview”
- [21]Texas Instruments Inc, ”TMS320C64x DSP Two-level Internal Memory Reference Guide”
- [22] Texas Instruments Inc, ”TMS320C64x DSP Video Port/VCXO Interpolated Control Port Reference Guide”
- [23] Texas Instruments Inc, ”TMS320C64x DSP Enhance Direct Memory Access Controller Reference Guide”
- [24] Texas Instruments Inc, ”TMS320C6000 Optimizing C Compiler Tutorial”
- [25] MJ.Swain and D.H. Ballard, “Colour Indexing,” *International Journal of Computer Vision* vol.7,pp. 11-32, 1999.

