國立交通大學

電機學院　電機與控制學程

碩士論文

以人類視覺系統為基礎之彩色影像縮放硬體實現

Hardware Implementation of Color Image Resizing Based on HVS Scheme

研究生：連明智

指導教授：林進燈　教授

中華民國九十八年七月

以人類視覺系統為基礎之彩色影像縮放硬體實現

Hardware Implementation of Color Image Resizing Based on HVS Scheme

研 究 生：連明智　　Student：Min-Jr Lian

指導教授：林進燈　　Advisor：Dr. Chin-Teng Lin

國 立 交 通 大 學

電機學院　電機與控制學程

碩 士 論 文

A Thesis

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

June 2009

Hsinchu, Taiwan, Republic of China

中華民國 九十八 年 七 月

以人類視覺系統為基礎之彩色影像縮放硬體實現

學生：連明智　　指導教授：林進燈 博士

國立交通大學　電機學院　電機與控制學程碩士班

中文摘要

目前市場上，在眾多的消費性電子產品中如：數位相機、數位電視、手機等在追求高品質的影像，取得較高解析度影像畫面的處理方法就更為格外的重要。一些相關的研究探討中，如何從低解析度的影像轉換到高解析度的影像來取得較佳的補插點，過去有提出很多相關的研究，例如雙線性補插或雙立方補插。這種補插方法無法滿足高畫質之需求，因為這些方法可能會使得影像產生模糊或邊緣鋸齒狀。此外，在高解析度彩色影像運算量非常大，因此需要使用硬體來加速演算法處理的速度。

本論文介紹針對彩色影像放大的硬體實現，從一般標準數位視訊訊號或一般儲存媒介(記憶體)中的彩色影像圖片輸入來執行影像的縮放。為提高人類的視覺效果，硬體實現有以下的特點：(1)可經由模糊決策模組能夠自動辨識輸入影像之特徵將其分類，來決定影像補插模組之使用。當輸入影像對肉眼不明顯時，選用雙線性補插方法以節省計算能耗，若輸入影像對肉眼明顯且具有方向性時，為了減少在影像邊緣所產生之鋸齒狀以及模糊狀，則選用邊緣適應性影像補插模組。(2)對於演算法所使用到的指數函數運算，在一個有限值的範圍內可以透過硬體查表的方式來實現，這不僅可以加快計算，並能減少計算錯誤，也不影響精確度。(3) 由於使用Line Buffer設計，使得硬體電路面積可以縮減。

彩色影像放大經由FPGA實現來達到影像縮放的功能，經由實驗的結果，此方法的PSNR與軟體實現方式來比較很相近，並能有效消除邊緣部份鋸齒和模糊的情形，以獲得更好的視覺效果。

Hardware Implementation of Color Image Resizing Based on HVS Scheme

Student：Min-Jr Lian Advisor : Dr. Chin-Teng Lin

Department of Electrical and Control Engineering

National Chiao Tung University

Abstract

At present the market, many consumer electronics products such as digital cameras, digital televisions, and mobile phones are all pursuing the high-quality image. High-resolution video image approach is even more particularly important. Some related researches have investigated how to get a better fill insertion point from a low resolution image to high-resolution image, for example, bilinear or two-cubic interpolation. This interpolation does not satisfy the requirement of a high-quality image due to jagged and blurry defects appearing in an edge. In addition, it is necessary to accelerate the computation with hardware design for the large amount of computation in high-resolution color image data.

This thesis presents the hardware implementation for color image resizing to perform color image scaling. The input source can be a standard video signal or a storage device. To enhance the human visual effect, the hardware implementation has the following features. (1) It uses fuzzy decision to automatically identify the characteristic of input image and to decide which interpolation module will be used. If input image is not sensitive to human eyes, the bilinear interpolation will be chosen to reduce computational power. Otherwise, the edge-adaptive image interpolation is chosen to reduce blurry and jagged defects in edge section. (2) For the computation of the exponential function, it is realized through the hardware look-up table in a limited range of values. This can not only speed up computation, but also reduce calculated errors in order to not influence the accuracy. (3) Due to line buffer design, the proposed hardware has smaller area.

The color image resizing is realized with FPGA to achieve a scalable image function. By the experimental results, the proposed hardware design and software simulation are approach compared with their PSNR. Simultaneously, this design can effectively eliminate the jagged and blurred part in the edge of an image in order to obtain better visual effect.

誌　　謝

經過兩年多的時間終於完成碩士學業，在這一段學習過程中，首先要感謝我的指導教授-林進燈教務長在論文上的指導上，無論是在模糊理論的研究或專業方面上也都獲得不少相關知識，也感謝老師提供一個很好的學習環境及支援，才能夠讓學生的論文順利成完，在這麼嚴謹教導中，對於未來在工作上的確會有不少的幫助。

另外在實驗室裡，鍾仁峰學長對於論文給予很多的指導，當在遇到研究的瓶頸時，都會不厭其煩地教導，而且在很多的實驗中授與更多的研究資源，使得學生在演算法及系統硬體電路設計上，能夠獲得更多的專業知識；再來是感謝范倫達教授在每一次報告中給予很多的論文的指導。還要感謝蒲鶴章博士對於演算法研究過程中的指導；另外要感謝義隆電子智慧型人機介面設計部的陳崇明博士及新唐科技工程師江明昌在專業實作上技術的支援及指導；在實驗室裡的夥伴還要感謝紹航學長及創連，在研究學習的這段時間裡給予很多的支持及鼓勵。

最後要感謝爸爸、媽媽、姐姐及妹妹，在生活中給予最大的支持，使得我能夠在交通大學順利完成碩士論文。

在人生的道路中我已經完成碩士的學業，對於未來更是充滿信心，也就是因為這樣對於家人、學長、同學及同事們除了感謝之外，也希望他們能夠和我一樣對於未來也能夠有更美好的將來。　~~

明智　2009/10/23

目　　錄

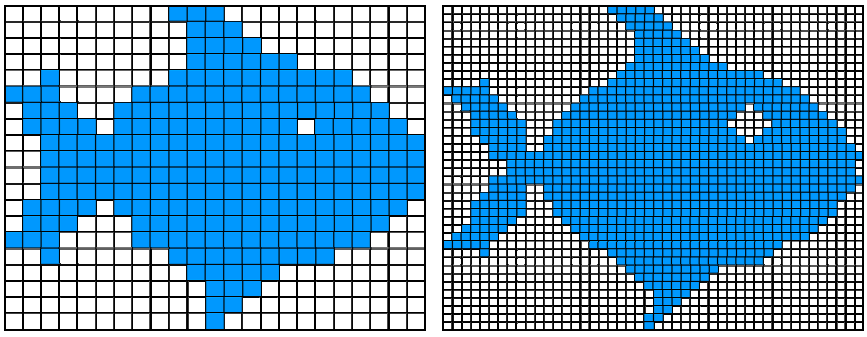
圖目錄

表目錄

一. 緒論

1.1 研究背景

人類視覺對於影像的辨別在輪廓上是很敏銳的，眼睛所看到的影像輪廓如果很清楚的話，才能夠讓人的大腦來判斷影像中物體的清晰度，所以在影像品質的優劣中，可以很簡單的分辨出來差異性。在視覺影像顯示技術的領域中，從標準畫質電視 (Standard Definition TV, SDTV) 的視訊影像規格轉換到高畫質電視(High Definition TV, HDTV)的視訊影像規格發展的很快，隨著電腦影像上更高的解析度需求，HDTV高解析度電視的盛行，高解析度影像(high resolution)的品質如圖1-1(b)，它可以提供更細緻的影像畫面，但絕大多數的影像應用，包含了電視系統 (NTSC or PAL) 在內，其顯示的來源影像內容並非皆為高解析度(720p/1080i/1080p)的影像品質，大多數還是屬於標準解析度(480i/576i)的影像如圖1-1(a)品質。根據這個理由，我們開始思考如何從低解析度的影像放大到高解析度的影像，其最主要的目的，能夠克服放大後的影像失真(artifacts)，例如模糊(Jagged)及鋸齒(Blur)效應等，因此影像品質的提升是很重要的一環。提高影像解析度也是在影像處理中一個很重要的技術。



(a) (b)

Fig. 1-1. 標準解析度影像及高解析度影像[23]

1.2 研究動機

由於數位化的影像是透過離散的數位資料所組合而成，如果要將影像解析度放大，必須在從相鄰的像素(Pixels)間，預測像素值並填入，而預測的準確性則直接影響了影像解析度放大後的品質，處理這些問題的相關技術稱之為數位變焦（Digital Zooming）、影像放大(Image Enlargement)或者是取樣技術（Resampling Technique）。

從過去所提出的高影像解析度的內插方法有很多，較常用的內插法中有最近相鄰內插法(Nearest Neighbor Interpolation)、雙線性內插法(Bilinear Interpolation)以及雙立方內插法(Bicubic Interpolation)等。在上述的內插法中屬於非可適性內插法(Non-Adaptive interpolation)，非可適性內插法在對每一個像素作內插運算時採用固定的模式計算的方式。此外在邊緣偵測內插法(Edge Directed Interpolation)是屬於可適性內插法(Adaptive interpolation)，在可適性內插法方法中可以偵測出附近相鄰像素的特徵，並且根據偵測的結果選擇最相似的相鄰像素作內插。一般來說，非可適性內插法運算較為簡單，但對於邊緣部份較無法處理;可適性內插法的運算較為複雜，但相對來看品質也比較好。

因此本論文結合了非可適性內插法(Non-adaptive interpolation)及可適性內插法(Adaptive interpolation)兩種特性能夠讓圖片提高解析度，在圖片中邊緣上的處理上會更為平滑。同時，將這種方法利用平行訊號處理的功能，來作為硬體的實現。

1. 對於像素的處理只能使用灰階對於視覺上的效果並不能滿足，所以在彩色架構中使用彩色影像的資料來處理。
2. 在灰階架構中因為在影像放大之後，圖片中會產生一些雜訊而且在邊緣的處理上會有些許的鋸齒狀。
3. 灰階架構的硬體電路中，所需要的面積較大。
4. 灰階架構中影像處理的單位以一整張圖(Frame)為一個單位，較浪費記憶體空間。

從上述的討論中可以知道灰階架構需要改善的缺點，所以在提出新的做法中會針對這些缺點來做修正。

1.3 論文架構

本論文中，在第二章會針對過去所提出的一些影像放大內插法的優缺點來做討論，從圖片的失真問題中邊緣所產生的一些鋸齒及模糊的情形來進行討論，最後會針對彩色像素格式及演算法的原理來做一個探討。第三章對於整個邊緣適應性演算法硬體實現，從彩色影像資料的擷取到演算法的運算後資料處理的流程討論，另外也會針對過去灰階架構中的一些問題來做一個改善。第四章部份則列出實驗後的結果及電路面積、效能的比較。第五章則是總論及未來展望。

二. 影像放大原理及相關研究

2.1 影像放大基本原理

在數位影像縮放技術中，影像放大原理是將一張低解析度的照片變成高解析度的照片。所以在圖片放大的過程中必須要在原來的點與點之間必須插入一個新的點，經由現有的資料插入資料以提高取樣(Up Sampling)，藉此求得較高解析度的影像，也就是所謂的內插法，內插(Interpolation)的數學意義就是將兩個函數做迴旋積分(Convolution)影響影像放大品質的主要關鍵是內插濾波器的不同，一個理想的濾波器函數是sinc如圖2-1所示，在實際上的濾波器不會跟sinc的函數一樣，所以在經過影像放大後的結果品質也會有所不同，一般來說，要放大低解析度的影像成為高解析度影像時，大部份的研究使用多項式內插法 (Polynomial Interpolation)。常用的多項式內插法包括最近相鄰內插法、雙線性內插法及雙立方內插法，而這些內插法中對於邊緣部份的處理會產生鋸齒或模糊的情況，為了解決此問題便有人提出邊緣方向偵測內插法 (Edge-Directed Interpolation)或等照度線內插法 (Isophote Interpolation)些類型主要目的是針對邊緣部份做有效的修正，使得放大後的圖片中的線條可以更平滑。

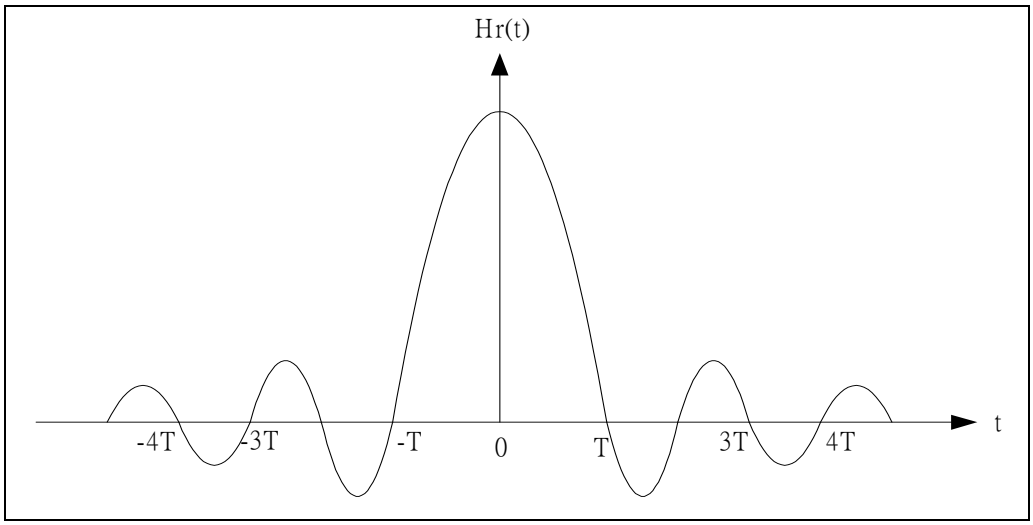


Fig. 2-1. 重建濾波器響應[24]

2.2 內插法的相關研究

(1) 最近相鄰內插法 (Nearest Neighbor Interpolation)

最近相鄰內插法[4]由A.H.Thiessen所提出的一種較為簡單的內插法，主要的精神是將四個相鄰近的像素點的值來做內插並求出新的補插點，這是所有內插法中最簡單的內插法，但所放大後的影像品質並不是那麼好，圖片通常會產生一些鋸齒狀(Jagged)的線條。

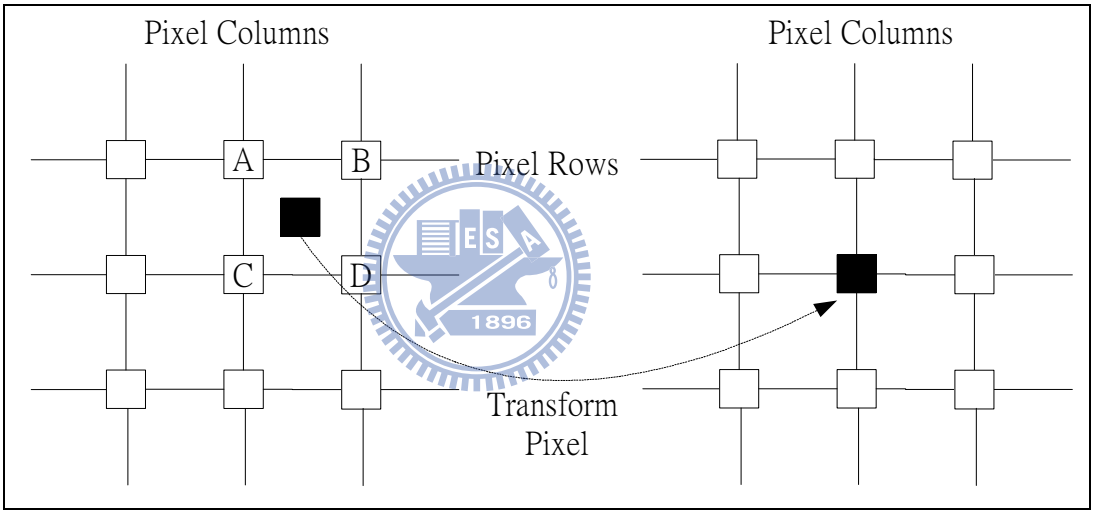


Fig. 2-2. 相鄰內插法說明圖

由圖2-2中可以得知，在目的影像中的像素，推回至原始影像，並不會落在整數值的像素上，而是落在像素A、B、C、D之間非整數倍率像素的位置上，最近相鄰內插法的作法就是將原始影像中的分數位置選擇一個最接近距離的整數位置的像素。以圖2-2的範例來說的話，假如C點為最近的距離，就將C點的像素值複製給目的影像這個像素。這種方法適合影像類型為影像內容非常銳化(例如方形)且顏色對比非常明顯，因為此內插法可以保持處理後的影像的色調與原始影像不變。

(2) 雙線性內插法 (Bilinear Interpolation)

雙線性內插法[2] 也是很簡單內插法之一，它是把未知像素色彩值周圍的同色像素，採用線性方式平均得到估算的像素值，如式(2.1)所示。

$$f(x,y) = (1-dx)(1-dy)P1 + dx(1-dy)P2 + (1-dx)dyP3 + dxdyP4 \qquad (2.1)$$

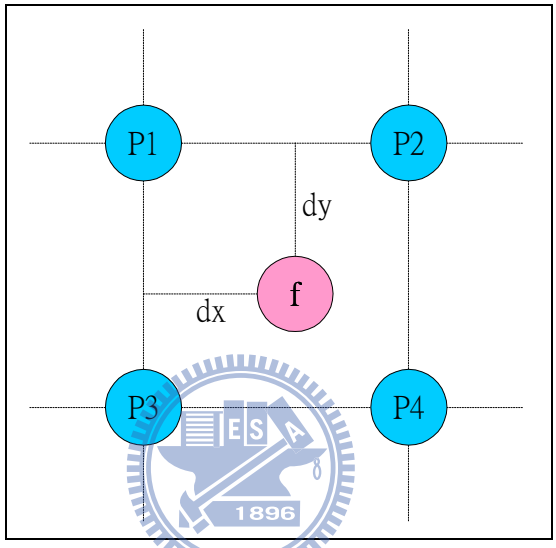


Fig. 2-3. 雙線性內插法說明圖

在圖 2-3 中，要計算的新像素值 $f(x, y)$補插點的位置介於四個整數值的像素點 P1、P2、P3、P4 之間，而且它們與的間距也是已知的，這個間距只要計算出跟像素點 P1 的間距，這些鄰近點若是離越近，表示對補插點越有貢獻，亦是表示此點的值對有較多的影響，反之，則對的影響力越小，使用雙線性內插法縮放大的影像，相鄰像素之間會比直接取整數點更有連續性，也就是更加平滑。

(3) 雙立方內插法 (Bicubic Interpolation)

雙立方內插法[5]是計算鄰近周圍16個像素，根據目標點與鄰近16點的距離不同而有不同的貢獻程度。雙立方內插法之補插點運算式如式(2.2)及雙立方內插法的核心(interpolation kernel)如式(2.3)：

$$P = \sum_{m=-1}^{2}\sum_{n=-1}^{2} S_{(i+m,j+n)} h_c(n-x) h_c(m-y) \tag{2.2}$$

$$h_c(x) = \begin{cases} 1-2|x|^2+|x|^3 & ,0 \le |x| < 1 \\ 4-8|x|+5|x|^2-|x|^3 & ,1 \le |x| < 2 \\ 0 & ,otherwise \end{cases} \tag{2.3}$$

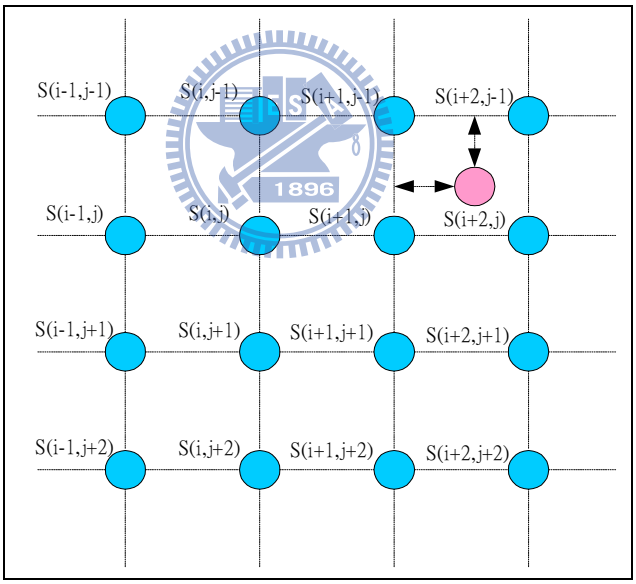


Fig. 2-4. 雙立方內插法說明圖

由圖2-4所示，若為一個內插的像素點，根據其周圍鄰近點的相關性，如式(2.3)可知雙立方內插法的方法是針對要內插的像素點即，根據周圍16個鄰近的像素點作迴旋積分的結果雙立方內插法的缺點主要在於效率較低，運算執行時間較長。由於此內插法於低通濾波表現較前述各插補法更趨近於Sinc函數，因此其影像插補品質較好，但在插補運算複雜度也相對的提高許多。

(4) 邊緣方向偵測內插法 (Edge-Directed Interpolation)

邊緣方向偵測內插法[9]，是經由找出在低解析度影像中的邊緣資訊，將它對應到高解析度上的相對位置，再藉由修正過的內插法及配合適當的濾波器，建構出高解析度影像如圖2-5所示。

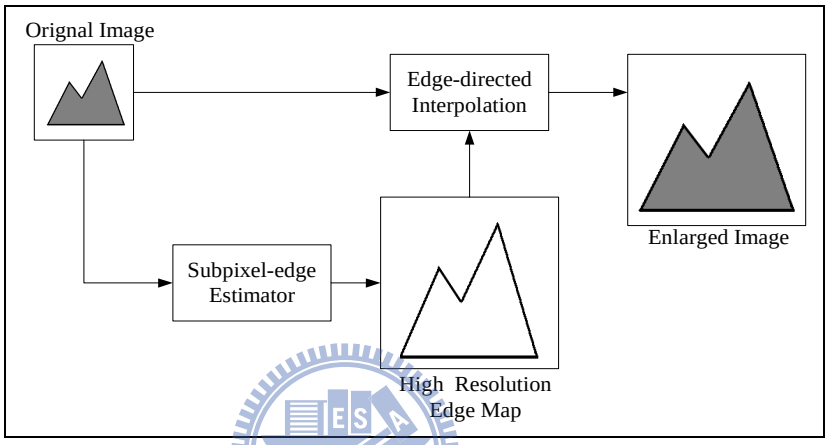


Fig. 2-5. 邊緣方向內插法架構[9]

隨後，Xin Li 與Michael Orchard 又提出新邊緣方向內插法[10]，其主要的概念上是先計算原來低解析度影像的局部協方差係數 (local covariance coefficient)，再經由協方差計算出適合插補在較高解析度的影像。新邊緣方向內插法是一個4階的內插演算法，並透過最小均方差 (least mean squares)來計算出內插像素的灰階值，這種演算法會先判斷影像物體邊緣的部分，然後再沿著邊緣的部分去做內插，圖2-6為新邊緣方向內插示意圖。

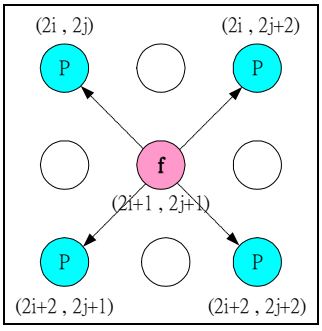


Fig. 2-6. 邊緣方向內插法之補插點[10]

在圖2-6所示，原來較低解析度影像P的4個像素(P0, P1, P2, P3)，在較高解析度影像中欲補插點的影像像素$F_{2i+1,2\ j+1}$，則是由低解析度影像P中的4個像素的灰階值加權所計算出來的。其數學式表示如式(2.4)。

$$\hat{Y}_{2i+1,2j+1} = \sum_{k=0}^{1}\sum_{l=0}^{1} P_{2k+l} F_{2(i+k),2(i+l)} \tag{2.4}$$

這個演算法雖然可以考慮到解決因影像放大所造成的一些失真現象，但由於協方差演算法的運算過程比較複雜。此方法只是將原先低解析度的影像，作較準確的放大，而並未考量到如果此低解析度的影像本身就已經遭受到模糊退化時，那麼影像放大之後，勢必仍然是模糊的影像。

(5) 等照度線內插法 (Isophote Interpolation)

等照度線內插法[14] (Isophote Interpolation)從最近的邊緣方向內插法中添加有限的幾何訊息（邊緣地圖）建立更準確視覺效果補插在影像中較關鍵的輪廓上。等照度線內插法是一種針對幾何的內插法適合於等照度線（強度級別曲線）在影像中所有的點，而不是只有特定的輪廓。此方法能使用現有插值技術作為一個初步近似(如雙立方內插法)，然後再反覆重建等照度線將邊緣做一個平滑約束，如圖 2-7 所示為 Isophote 重建誤差。

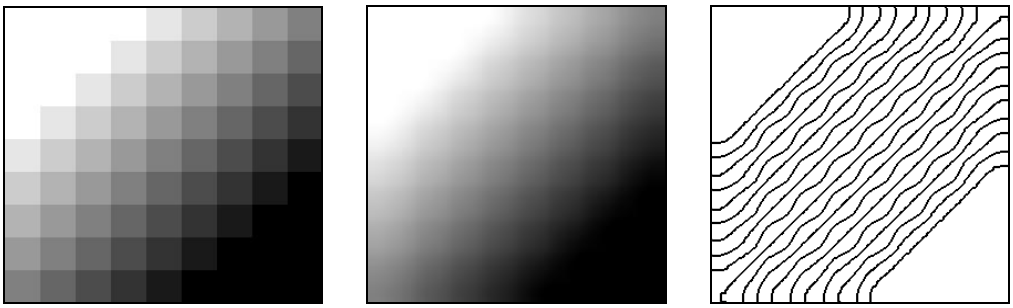


Fig. 2-7. Isophote 重建誤差[14]

2.3 影像失真問題

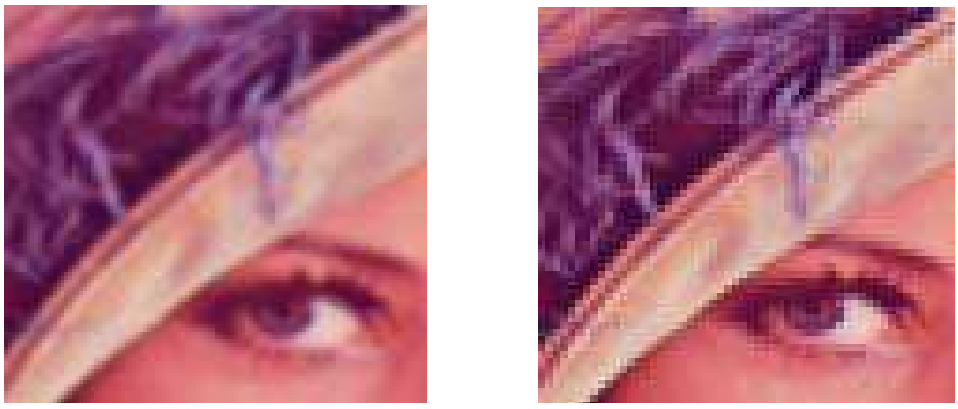
由於我們使用上述的各種不同的內插法，會導致影像失真，主要失真會有兩種就是模糊(Blur)及鋸齒(Jag)，以下針對圖片放大後會產生的兩種問題來討論。

(1) 模糊(Blurring)

當圖片在放大後發生模糊(Blurring)的情形時，人的眼睛對於放大後影像中的物體無法對焦與辨識，就會造成模糊(Blurring)主要原因是影像像素反覆地進行內插，通常重建濾波器為一個低通濾波器，因此會把高頻部份給濾除，便會出現模糊(Blurring)的效果，如果使用雙線性內插法如圖2-8(a)，就會發現有模糊(Blurring)的情形。

(2) 格狀及鋸齒(Block & Jag)

由於實際的內插函數在時間軸上並非如同sinc函數是無限延伸的，都是以有限數目的取樣像素來取樣，因此取樣點的像素都會被周圍的像素給影響到，會造成格狀的失真效果，而且在影像的邊緣處則出現鋸齒狀(Jag)的現象。通常內插函數在時間軸上，波形的下降程度越陡的話，其格子狀(Block)與鋸齒狀(Jag)的情況會更為嚴重。在典型的內插函數為最近相鄰內插法，可以從圖2-8(b)中可以看出放大後的影像會有出現格狀的現象。



(a) (b)

Fig. 2-8. (a)使用雙線性內插法產生模糊(Blurring)及(b)相鄰內插法後圖片產生格狀(Block)[31]

2.4 色彩處理相關研究

利用彩色圖片來做放大也可以利用上述的這些內插法來執行，在眾多彩色格式中以 RGB 及 YUV 為主，RGB 訴求於人眼對色彩的感應，YUV 則著重於視覺對於亮度的敏感程度，因為無論是數位電視、數位相機或電視遊樂器在輸入或輸出時所要求的格式是以這兩種為最多。

(1) RGB 格式:

RGB 三原色光顯示主要用於電視和計算機的顯示器，有陰極射線管顯示、液晶顯示和電漿顯示等方法，將三種原色光在每一像素中組合成從全黑色到全白色之間各種不同的顏色光，目前在計算機硬體中採取每一像素用 24 位元表示的方法，所以三種原色光各分到 8 位元，每一種原色的強度依照 8 位元的最高值 2^8 分為 256 個值。用這種方法可以組合出 1670 萬種顏色如圖 2-9 所示為 RGB 立方體平面，但人眼實際只能分辨出 1000 萬種顏色。每一個像素點為 24 位元編碼的 RGB 值，所使用表示為紅色、綠色和藍色強度的三個 8 位元無符號整數（0 到 255）。處理內插法必須注意到，在補點時的運算都要根據這三種顏色(RGB)的變化來做運算，所以在使用 RGB 格式過程內插法的運算都必須是獨立，這樣使得它的運算量或者是資料量的處理都會是一個很大的負擔。

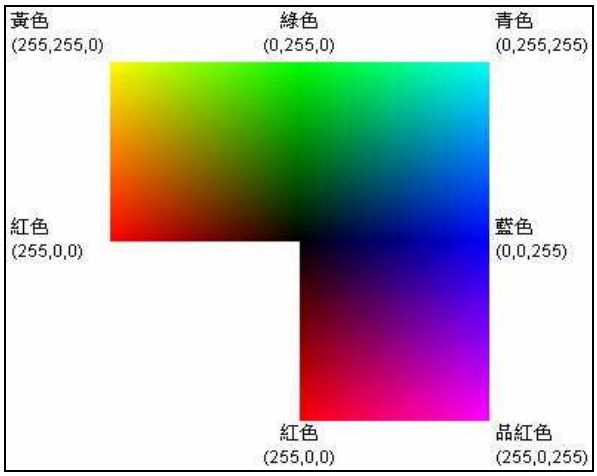
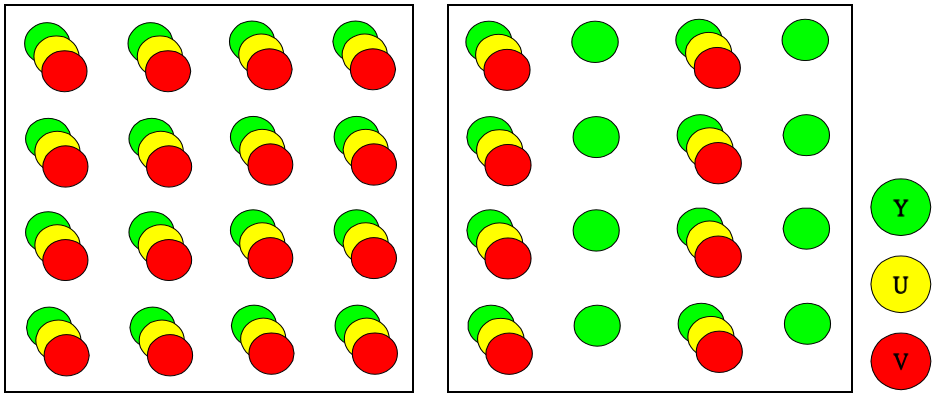


Fig. 2-9. RGB 立方體平面[25]

(2) YUV 格式:

YUV被解釋成True-color顏色空間（color space）的種類，YUV、YC_bC_r、YP_bP_r等專有名詞都可以統稱為YUV。「Y」表示明亮度（Luminance）如圖2-10(a)，「U」和「V」則是色差、濃度（Chrominance）如圖2-10(c)(d)，其中YUV和通常用來描述類比訊號，而相反的YCbCr則是用來描述數位的影像訊號。大多數YUV格式平均使用的每像素位數都少於24位元。如圖2-11(a)所示為YUV 4:4:4排列所有的像素組合中最逼真的格式，每一個像素（24 bits）都會有自己獨立的YUV元素，即每4個Y能夠分配4個U及4個V；如圖2-11(b)所示為YUV 4:2:2排列在不影響視覺上的判斷時可以將UV做減半，即每4個Y能夠分配2個U及2個V[20]。

Fig. 2-10. (a) YUV 4:4:4 組合[26]、(b)Y 像素部份、(c)U 像素、及(d)V 像素部份



(a) (b)

Fig. 2-11. YUV 排列 (a)4:4:4 排列, (b)4:2:2 排列

YUV的格式中分成兩個格式：

- 緊縮格式(Packed Format)：將Y、U、V儲存成Macro Pixels陣列如圖2-12緊縮格式是將YUV的元素混合在一起，為了節省資料量的處理，在本論文中所採用的格式為YUV 4:2:2緊縮格式，而排列的方式有YUYV或UYUV等。
- 平面格式(Planar Format)：將 Y、U、V 分別儲存成不同的矩陣中如圖 2-13 平面格式是指每 Y 份量，U 份量和 V 份量都是以獨立的平面組織的，也就是說所有的 U 份量必須在 Y 份量後面，而 V 份量在所有的 U 份量後面。

Y	U	Y	V	Y	U	Y	V	Y	U	Y	V	Y	U	Y	V
Y	U	Y	V	Y	U	Y	V	Y	U	Y	V	Y	U	Y	V
Y	U	Y	V	Y	U	Y	V	Y	U	Y	V	Y	U	Y	V
Y	U	Y	V	Y	U	Y	V	Y	U	Y	V	Y	U	Y	V

Fig. 2-12. YUV 緊縮格式排列

Y	Y	Y	Y	.....	U	U	U	U	.....	V	V	V	V	.....	
Y	Y	Y	Y		U	U	U	U		V	V	V	V		
Y	Y	Y	Y		U	U	U	U		V	V	V	V		
Y	Y	Y	Y		U	U	U	U		V	V	V	V		

Fig. 2-13. YUV 平面格式排列

2.5 根據人類視覺可適應性邊緣彩色影像縮放(HVS- based edge-adaptive Color image scaling)

2.5.1 簡介

在數位影像處理中，影像縮放技術的應用非常廣泛，但普遍存在的問題就是影像縮放後的失真現象。過去所使用的一些內插法中例如:最近相鄰內插法、雙線性內插法及雙立方內插法[2]-[6]這些內插法對於邊緣處理未能提供較好的效果。

由於人的眼睛對於圖片中的邊緣部份較為敏感所有很多針對邊緣處理的演算法已被提出[7]-[15]。這些理論可用來提高在視覺上對於邊緣的效果。

為了讓影像縮放後得到更好的邊緣效果，本篇論文採用根據人類視覺可適應性邊緣影像縮放(HVS-based edge-adaptive image scaling)[1]。在本篇論文中會使用到雙線性內插法(Bilinear Interpolation)和邊緣適應性影像內插法(Edge-Adaptive Interpolation)。在使用上述兩種內插法之前會使用到模糊決策系統(Fuzzy Decision System) 。來決定影像的是否為邊緣部份，經由此系統的判斷可以將影像分為人類視覺中的非感測區域(Non-Sensitive Regions)和感測區域(Sensitive Regions)。當偵測出是非感測區域也就是一般較為平面或人眼較無法判別的範圍，選擇使用雙線性內插法(Bilinear Interpolation)，如果偵測出是感測區域也就是邊緣線條部份，選擇則使用邊緣適應性影像內插法(Edge-Adaptive Interpolation)。

2.5.2 演算法架構

在人類視覺可適應性邊緣彩色影像縮放(HVS-based edge-adaptive Color image scaling)演算法架構中如圖 2-14 所示，包含模糊決策系統、角度計算(Angle Evaluation)、邊緣適應性影像內插法及雙線性內插法。

在處理影像之前，必須先把影像切割成 4×4 的滑動區塊如圖 2-15(a)，模糊決策系統會先將 4×4 的滑動區塊執行背景物件及邊緣的判斷，再根據模糊決策系統所輸出的結果，來決定使用邊緣適應性內插法或雙線性內插法；如果判斷輸出的結果是邊緣，將會執行邊緣適應性影像內插法，在取得適應性補插點之前，需對邊緣方向需先做角度的判定，故在影像 4×4 的滑動區塊中的 16 個像素在計算出角度的結果之後，演算法會根據每一個像素所輸出的角度做直方圖(Histogram)統計，最後再依據直方圖計算出來的結果選擇邊緣適應性影像內插法的權重(Weight)再將 16 個像素做乘加的計算，即可得到所要的補插點；另外模糊決策系輸出是非邊緣時，只需使用較簡單的雙線性內插法。

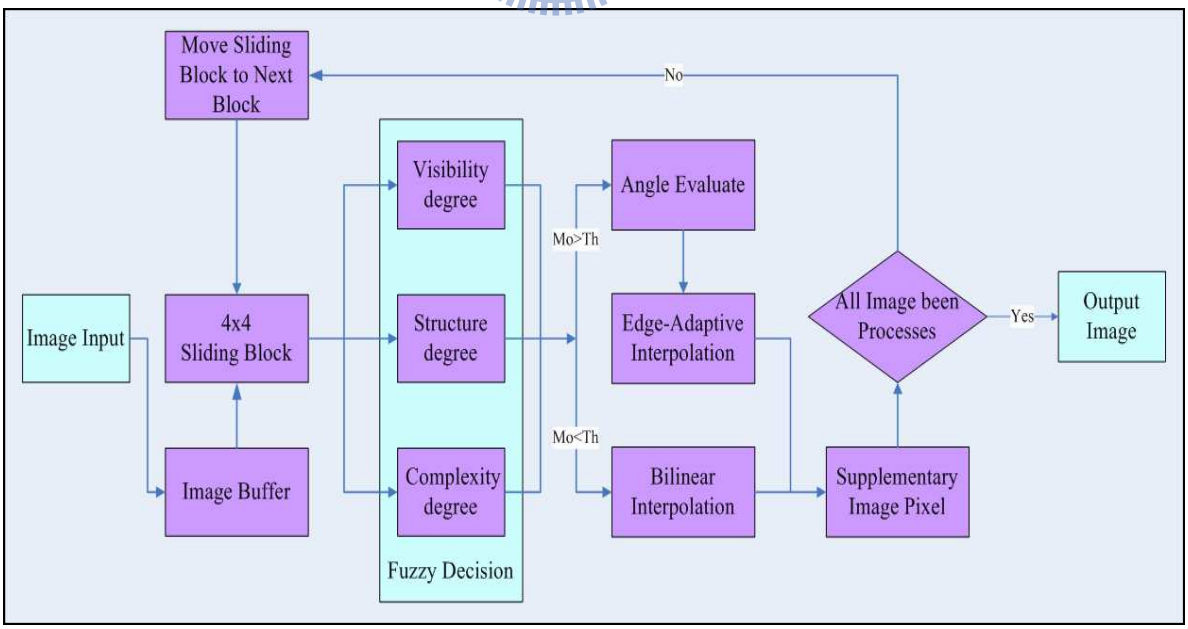
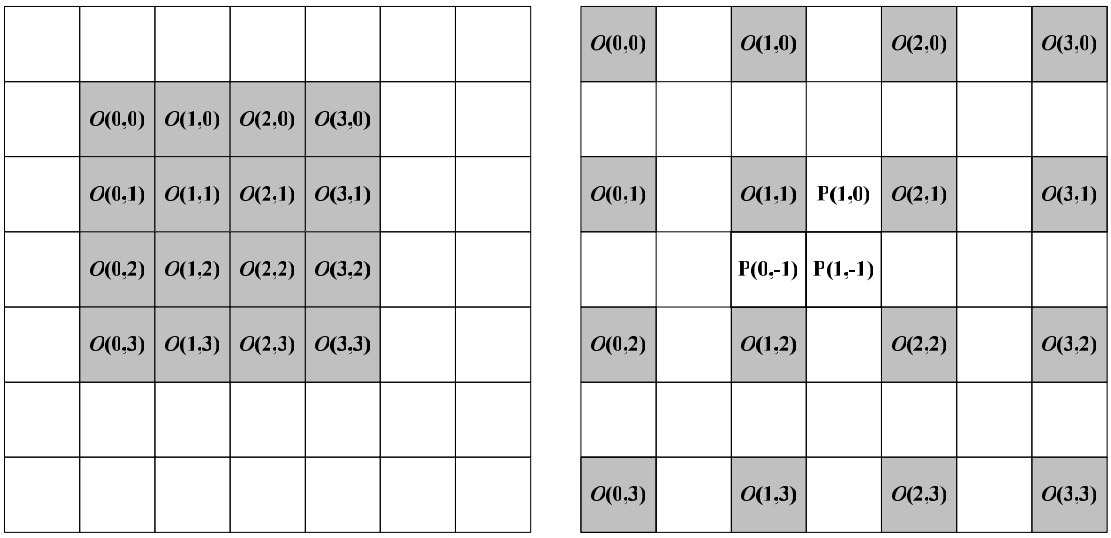


Fig. 2-14. 人類視覺可適應性邊緣彩色影像縮放演算法架構[1]



(a) (b)

Fig. 2-15. (a) 4 × 4 滑動原始影像, (b) 兩倍補插點方塊圖[1]

當一張原始影像輸入時，首先會切割成一個4×4滑動區塊，會切割為4×4的原因就是考慮到，如果將滑動區塊切割太小，如3×3將使得演算法可能無法判斷出邊緣的情形，反之如果將滑動區塊切割太大，如6×6將使得邊緣的向位無法單一，也就是在一個滑動區塊中邊緣可能會出現多個向位，這樣一來會使得演算法對於角度計算會較為複雜，故在演算法在運算之前只要取得4×4滑動區塊，而在一張圖片中會有很多重疊的滑動區塊，在每一次的滑動區塊計算完畢之後，4×4影像區塊就會以水平的方向移動一個像素的間距，直到整個圖片的水平的像素點處理完之後，再以垂直的方向移動一個像素的間距。如圖2-15(a)是一個被切割4×4滑動區塊這裡$O(i, j)$指的是每一原始圖中的像素，圖2-15(b)是一張兩倍補插點的方塊圖在這張圖會有P(1,0), P(0,-1), P(1,-1)需要經由適應性影像內插法或雙線性內插法來執行補插點。

2.5.3 演算法分析

1. 模糊決策系統(fuzzy decision system)

研究發現人類視覺系統(Human Visual System)對於亮度對比和在相同亮度比較之下亮度對比對於人類視覺較為敏感。人類眼睛的能力可以判別物件及背景之間的不同取決於背景亮度(Background Luminance (BL))的平均值。如圖 2-16 所示，主要是模擬人眼對於背景及物件的曲線圖，也就是能見度臨界值(Visibility Threshold)與背景亮度關係曲線，從曲線中得知當背景亮度的範圍介於 70 到 150，所需的能見度臨界值較低，就是指背景亮度在 70 到 150 時，只要能見度臨界值大約等於 3，就能夠分辨出背景及物件之間的關係，另外當背景亮度最暗(BL=0)時，所需的能見度臨界值大約等於 22，所以背景亮度處在最暗的情形之下必需要提高能見度臨界值，才能夠使得人眼能夠分辨出背景及物件，總結上述的解釋可以從式子(2.7)所計算出來的結果以關係曲線為分界點，位於曲線上方者表示可以分辨出背景及物件，但如果在曲線的下方者並不能夠分辨出背景及物件。

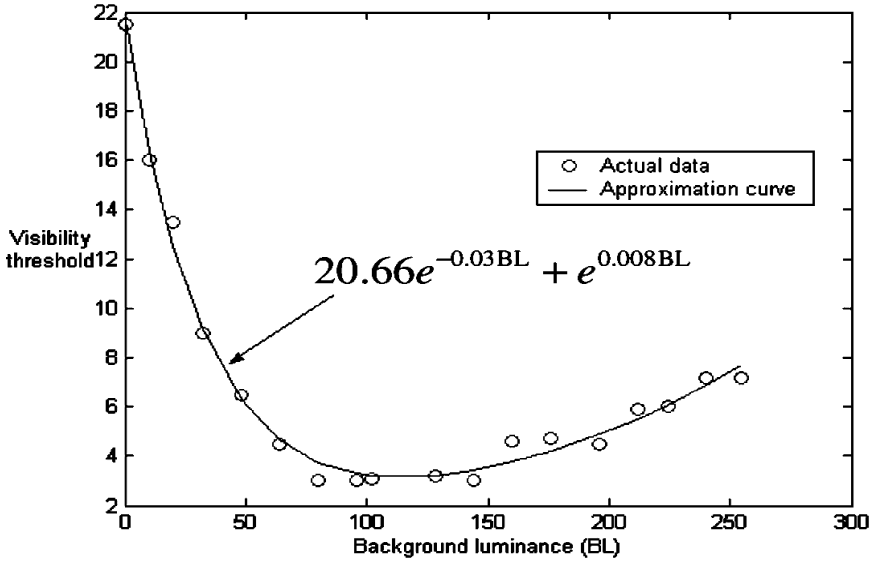


Fig. 2-16. Visibility thresholds 與背景亮度關係曲線[1]

為了使影像在補插點很能夠使得影像能夠有較好的品質，本論文中所提到的模糊決策系統能夠將輸入的影像分類人類視覺中的非感測區域(Non-Sensitive Regions)和感測區域(Sensitive Regions)。

針對非感測區域，可以使用雙線性內插法能夠減少計算成本對於硬體的實現也較為簡單，針對感測區域中，使用邊緣適應性影像內插法可以達到較好的視覺品質。

在模糊決策系統中會有三種輸入變量，為VD (Visibility Degree)、SD (Structure Degree)及CD (Complexity Degree)。VD主要是檢測背景及物件之間的關係，SD為邊緣或雜訊的一個判斷指標，而CD能夠對邊緣或紋理做一個偵測處理。從上述這三種變化中最後會得到一個結果，來做為是否為邊緣適應性影像內插法或是雙線性內插法的一個指標，接下來就是對於模糊決策系統的詳細介紹：

(1) VD (Visibility Degree)

VD的主要目的是將4×4的滑動區塊執行背景及物件的一個分析，每一個滑動區塊中的像素資訊都可能包含背景及物件的一個關係，從圖2-16所示可以了解當背景亮的不同呈現所需的能見度臨界值就會有不一樣的變化，故在取得VD的資訊之前需取得背景亮度如式(2.5)及4×4的滑動區塊中16個像素最亮像素及最暗像素的差(D)如式(2.6)，而背景亮度的計算可以透過16個像素再和 $B(i, j)$ 矩陣相乘後再相加所得到的結果再取平均即可得到平均背景亮度。

$$BL = \frac{1}{23}\sum_{i=0}^{3}\sum_{j=0}^{3} O(i, j) \times B(i, j), \qquad B(i, j) = \begin{bmatrix} 2 & 2 & 2 & 1 \\ 2 & 0 & 2 & 1 \\ 2 & 2 & 2 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \tag{2.5}$$

$$D = \max(O(i, j)) - \min(O(i, j)). \tag{2.6}$$

從關係曲線如式(2.7)中，可以將已計算出的背景亮度帶入能見度臨界值和背景亮度的非線性函數中，即可求得V(BL)值。

$$V(BL) = 20.66e^{-0.03BL} + e^{0.008BL} \tag{2.7}$$

在得到背景亮度、D和非線性函數V(BL)之後，經由式子(2.8)將D和非線性函數V(BL)的相減，將能夠計算出VD的值。

$$VD = D - V(BL) \tag{2.8}$$

在取得VD的值之後，如果是正值(VD>0)表示物體及背景之間的差別超過能見度臨界值表示已偵測出背件及物件的關係，如果是負的表示無法偵測出背景及物件，可以直接使用雙線性內插法。

(2) SD (Structure Degree)

SD的主要目的是將4×4的滑動區塊執行邊緣及雜訊的判斷，SD對於4×4滑動區塊中是否有高對比區域，而且影像區塊中的像素可以分成兩種類型如圖2-17，SD運算如式(2.9)

$$SD = \frac{\left|\max(O(i,j)) - mean(O(i,j)) - \left[mean(O(i,j)) - \min(O(i,j))\right]\right|}{\max(O(i,j)) - \min(O(i,j))} \tag{2.9}$$

$$mean(O(i,j)) = \frac{1}{16}\sum_{i=o}^{3}\sum_{j=0}^{3} O(i,j). \tag{2.10}$$

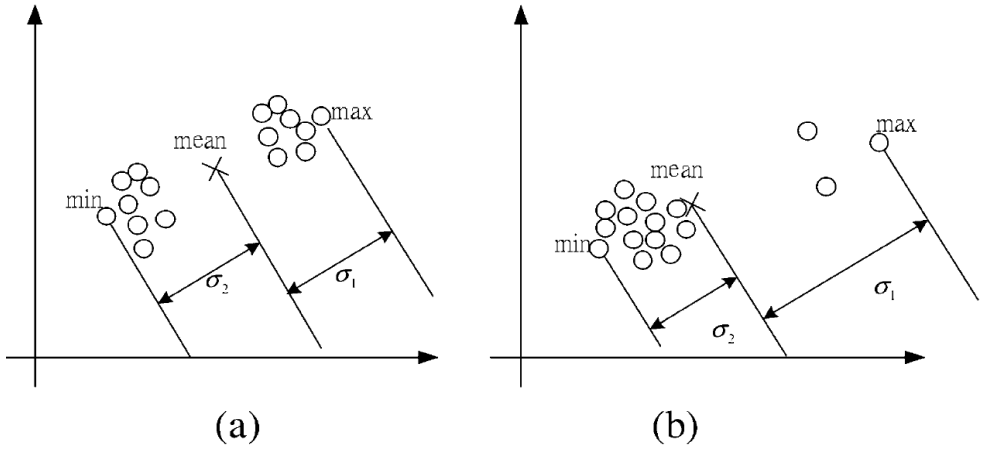


Fig. 2-17. 4×4 的滑動區塊像素在 SD 參數分佈圖[1]

SD的值介於0和1之間，如果SD的值比較小(接近0)表示σ_1和σ_2是很接近的如圖2-17(a)，4×4的滑動區塊中分出兩種類型表示影像區塊可能包含邊緣或紋理結構。反之如果SD的值比較大($|\sigma_1 - \sigma_2| >> 0$)如圖2-17(b)表示影像區塊中的像素的只有一種類型，因此在影像區塊中可能包含雜訊。

(3) CD (Complexity Degree)

在得SD的值之後，如果從中SD判斷出影像區塊為邊緣或紋理時，必需使用CD的計算，CD又稱Laplacian運算，為一個全方向性的邊緣強化處理運算，主要是對像素點的灰階斜度變化率作運算，在灰度值轉變的區域的值都會通過0點如圖2-18，也就是二次微分的特性如式(2.11)，從圖2-19所示中心點4和周圍的-1值的範圍之間可以對應到式子(2.12)就是CD的運算式子，由於Laplacian運算對於任何正值或負值的斜率效應都會加以強化但對雜訊相當敏感，因為Laplacian對邊緣方向具有不變性，所以無法偵測出邊緣的方向，故在本論文會利用Laplacian運算特性來取得邊緣或紋理的偵測。

$$\nabla^2 f(x,y) = [\frac{1}{\Delta x}]^2 [f(x+\Delta x, y) + f(x-\Delta x, y) + f(f(x, y+\Delta y) + f(x, y-\Delta y) - 4f(x,y)] \quad (2.11)$$

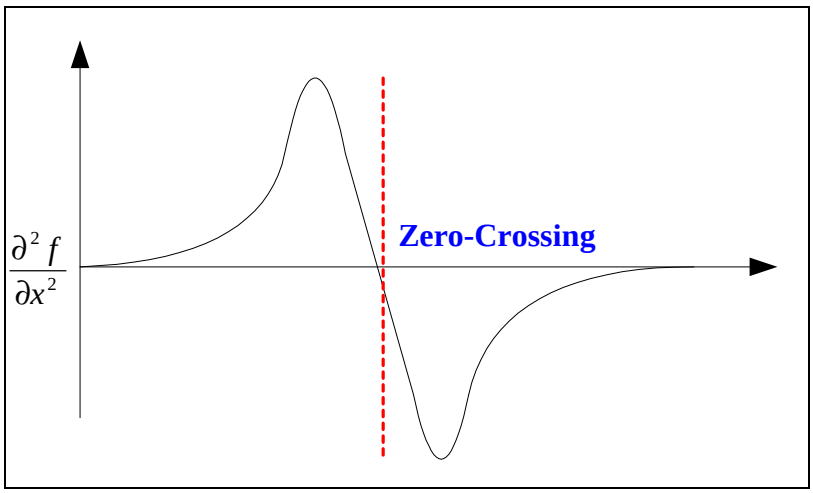


Fig. 2-18. 二階微分曲線

$$CD = \sum_{i=0}^{3}\sum_{j=0}^{3}\left|4O'(i,j) - [O'(i+1,j) + O'(i-1,j) + O'(i,j+1) + O'(i,j-1)]\right| \tag{2.12}$$

$$O'(i,j) = \{O(i,j) \le or > mean\} \tag{2.13}$$

	-1	
-1	4	-1
	-1	

Fig. 2-19. Laplacian 運算遮罩

$O'(i,j)$ 是經由4×4的滑動區塊中像素和16個像素計算出的平均值如式(2.10)做比較來取得適應性二值化後的結果如式(2.13)。在每一個4×4的滑動區塊中的像素的四個方向的邊(上下左右)都會有四個鄰近的像素，並從CD的式子(2.12)中計算出16個梯度值(Gradient Values)再將全部的梯度值加總後就會得到CD的值。如果CD值比較大表示影像區塊會是一個紋理的結構。反之，如果CD值較小影像區塊可能包含描繪出邊緣的結構。

(4) Fuzzy output

VD的變化量會有兩組模糊集合(Fuzzy Set)，N(negative) 和P(positive)，SD的變化量會輸出三組模糊集合，S(small), M(medium) 和B(big)，CD的變化量也會有三組模糊集合，S(small), M(medium) 和B(big)。歸屬函數(Membership Function)所對應的VD、SD和CD的關係如圖2-20 (a)–(c)，經由模糊決策系統中會有7種不同的結果如下:

1. If VD is N then Mo is BL.
2. If SD is B then Mo is BL.
3. If CD is B then Mo is BL.
4. If VD is P and SD is S and CD is S then Mo is NN.
5. If VD is P and SD is S and CD is M then Mo is BL.
6. If VD is P and SD is M and CD is S then Mo is NN.
7. If VD is P and SD is M and CD is M then Mo is BL.

當模糊決策系統計算輸出是NN，系統將選擇邊緣適應性影像內插法，反之選擇雙線性內插法。

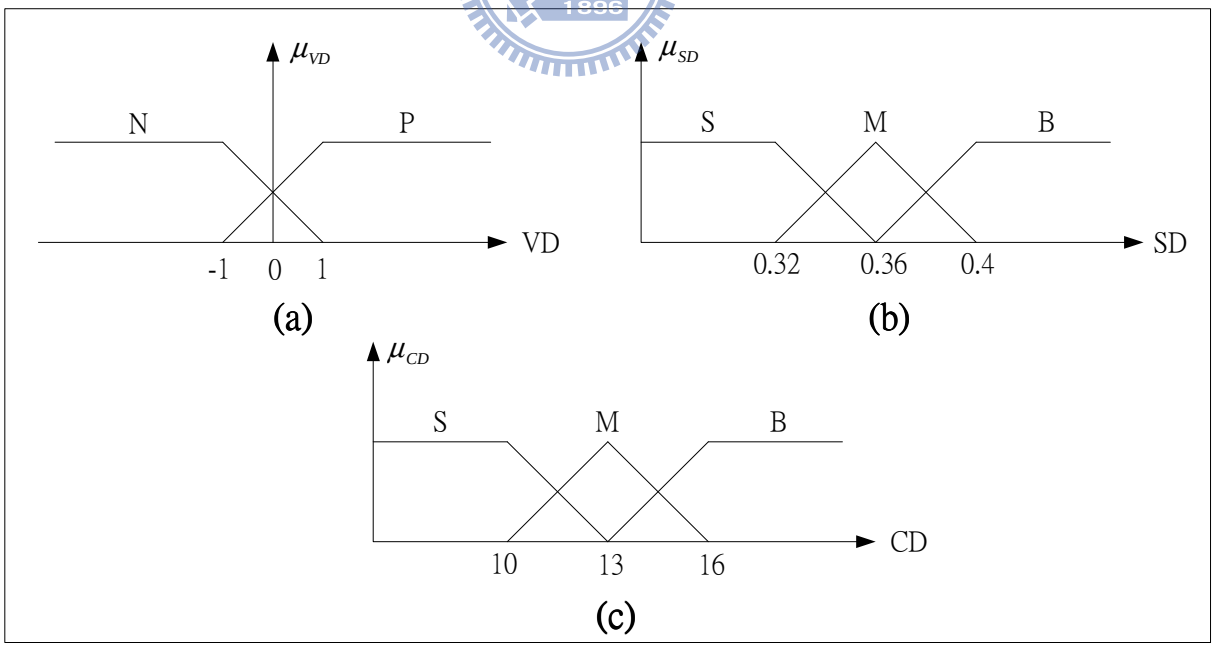


Fig. 2-20. (a)VD,(b)SD 和(c)CD 於隸屬函數的各種模糊集合[1]

2. Angle Evaluation

當模糊決策系統偵測出邊緣時，需先執行角度計算來決定影像區塊中的邊緣定位，角度計算的流程如圖 2-22 所示，在 4×4 的滑動區塊的每一個像素都會有鄰近八個像素如圖 2-21 所示，在取得反正切函數(Arc tan)之前，必須要先將原點 $O(i, j)$ 鄰近的八個像素做 $D_x(i , j)$及 $D_y(i , j)$ Soble 運算如式(2.14)及(2.15) ； 索貝爾運算（Sobel operator）主要用來對邊緣做檢測，所以在本論文中將索貝爾運算的結果代入反正切函數中來求得角度就能夠得到邊緣的方向，索貝爾運算共有兩種，一個為 X 方向，另一個方向為 Y，如圖 2-21 所示對原點像素的周圍做 3×3 的旋積計算，再將像素 $O(i, j)$ 的定位角度 $A(i, j)$ 經由反正切函數來計算出來如式(2.16)。

$$D_y(i, j) = O(i-1, j-1) + 2O(i, j-1) + O(i+1, j-1) - (O(i-1, j+1) + 2O(i, j+1) + O(i+1, j+1)) \tag{2.14}$$

$$D_x(i, j) = O(i-1, j-1) + 2O(i-1, j) + O(i-1, j+1) - (O(i+1, j-1) + 2O(i+1, j) + O(i+1, j+1)) \tag{2.15}$$

$$A(i, j) = -\frac{180}{\pi}[\tan^{-1}(\frac{D_y(i, j)}{D_x(i, j)})] \qquad 0 \le i \le 3 \quad 0 \le j \le 3 \tag{2.16}$$

O(i-1, j -1)		*-O(i-1, j +1)*
O(i, j -1) x2	*Oignal Pixel O(i , j)*	*-O(i, j +1) x2*
O(i+1, j -1)		*-O(i+1, j +1)*

Dy

O(i-1, j -1)	*O(i-1, j) x2*	*O(i-1, j +1)*
	Oignal Pixel O(i , j)	
-O(i+1, j -1)	*-O(i+1, j) x2*	*-O(i+1, j +1)*

Dx

Fig. 2-21. D_y 和 D_x 移動區塊

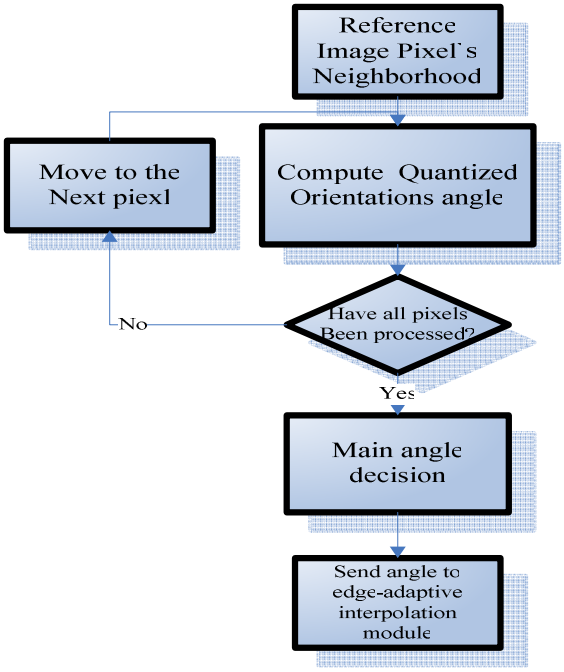


Fig. 2-22. 角度計算流程圖[21]

在獲得4×4影像區塊16個像素定位角度之後，演算法會針對這16次的角度做直方圖的計算，而直方圖的範圍共分成8階，就是角度會有八種不同的範圍如$\theta = 22.5 \times k$，k=0, 1, …, 7，而演算法會針對這16個像素來計算每一個像素所代表的角度範圍，如圖2-23所示為每一個像素角度位於直方圖中的範圍，從直方圖可以得知這16個像素出現在那一個角度範圍的次數，以出現頻率最多次(k=0,1,2…,7)來執行邊緣適應性影像內插法詳細內容請參考下一小節。

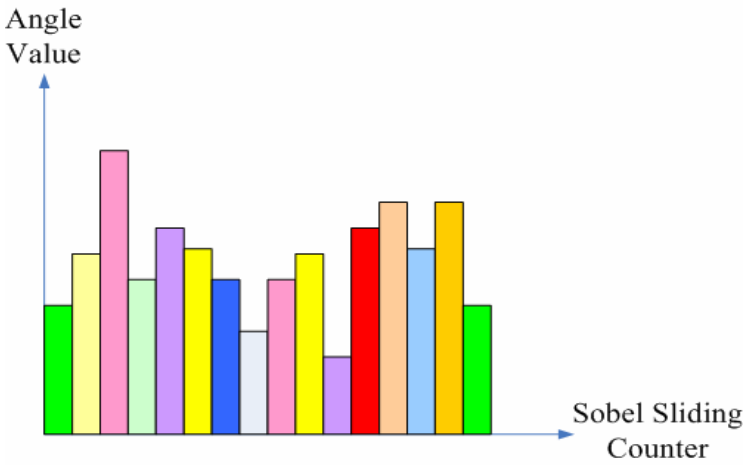


Fig. 2-23. 角度於直方圖中的分佈

2.5.4 內插法分析

如果模糊決策系統輸出是BL，表示輸入的影像不是邊緣的結構，對於人的眼睛較不敏感，所以只需使用雙線性內插法來對影像做內插。反之，當模糊決策系統輸出是NN，表示目前影像是邊緣的結構，並對影像做邊緣適應性內插法。

因為在直方圖中會有8種不同角度的範圍，所以演算法中會有8種不同的權重(Weight)來提供補插點的計算，在角度的計算及取得直方圖之後，演算法會將直方圖中取得出現機率最高者的資訊來對這8種權重做一個選擇。如式(2.17)，權重是根據輸入影像的定位來做分析，權重會根據影像的定位的不同來做內插，權重矩陣$W_{\theta,m,n}(i,j)$如下:

$$W_{\theta,m,n}(i,j)=\begin{bmatrix} W_{00} & W_{01} & W_{02} & W_{03} \\ W_{10} & W_{11} & W_{12} & W_{13} \\ W_{20} & W_{21} & W_{22} & W_{22} \\ W_{30} & W_{31} & W_{32} & W_{33} \end{bmatrix}. \tag{2.17}$$

在影像的放大中每一個原點會有三個補插點如圖2-24，每一個補插點都會有個別獨立的權重，而每一組權重都會有16個權重參數，所以三個補插點總共會有48個權重參數，4×4影像區塊的16個像素，只要將它們所對應的補插點位置的權重做乘加的運算就能得到所要的結果如式(2.18)。

$$P(i,j)=\sum_{i=0}^{3}\sum_{j=0}^{3}O(i,j)W_{\theta,m,n}(i,j) \tag{2.18}$$

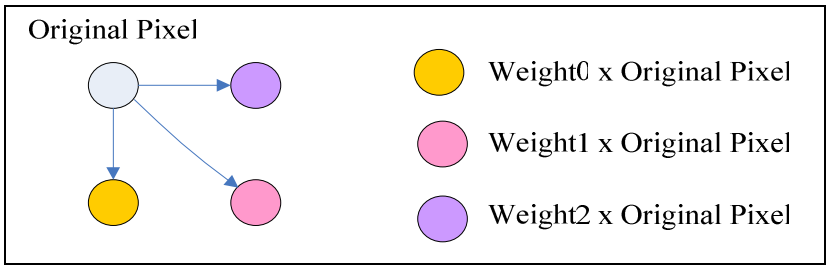


Fig. 2-24. 邊緣適應性內插法

三. 邊緣可適應性彩色影像縮放硬體實現

3.1 簡介

為了可以在硬體電路實現邊緣可適應性彩色影像縮放，首先必需考慮到原始影像來源，因為所要處理的資料是彩色的資料非常的大圖片的大小是720×240因為在彩色的像素中所使用YUV格式每一個像素除了Y像素(亮度或稱灰階)之外，還包含U像素及V像素(色差)，硬體對於大量彩色影像資料的處理，必須要有外部記憶體來做為存放資料的一個地方，所以在硬體的架構上就要規劃一個能跟外部記憶體溝通的一個介面(SSRAM interface)，另外也必需考慮到的影像處理的一個即時性，所採用的記憶體是同步靜態隨機記憶體(SSRAM)，因為它的存取速度非常的快最快可達到200MHz，所以才能夠在處理演算法的同時，並立即得到所要處理的資料。

在演算法的硬體架構上，會使用一顆FPGA的平台來實現演算法，在設計演算法電路的過程中，必須要考慮在有限的資源中減少電路的面積，並且在整個系統上可以讓影像資料處理的速度可以提升；在影像的來源中會有兩種，一個是從 TV-Decode 透過ITU-R.656標準輸入彩色視訊的資料，輸入的彩色影像是720×240 YUV 4:2:2格式的像素(pixels)，另一個來源是從Flash Memory輸入彩色影像；演算法處理完之後的資料會寫入外部同步靜態隨機記憶體(SSRAM)中，並輸出到320×240的TFT LCD Panel；當放大後的影像是1440×480 從TFT LCD Panel上只能看到圖片的四分之一，所以硬體電路中也加上了捲軸(Scroll)的功能如圖3-1所示，在每一次經由捲軸的功能就可以看到這張放大之後的影像；在FPGA硬體中有放入邊緣適應性影像內插法及雙線性內插法，當彩色影像從外部輸入到這兩種演算法硬體中，即可分辨出處理過後邊緣的差異性。

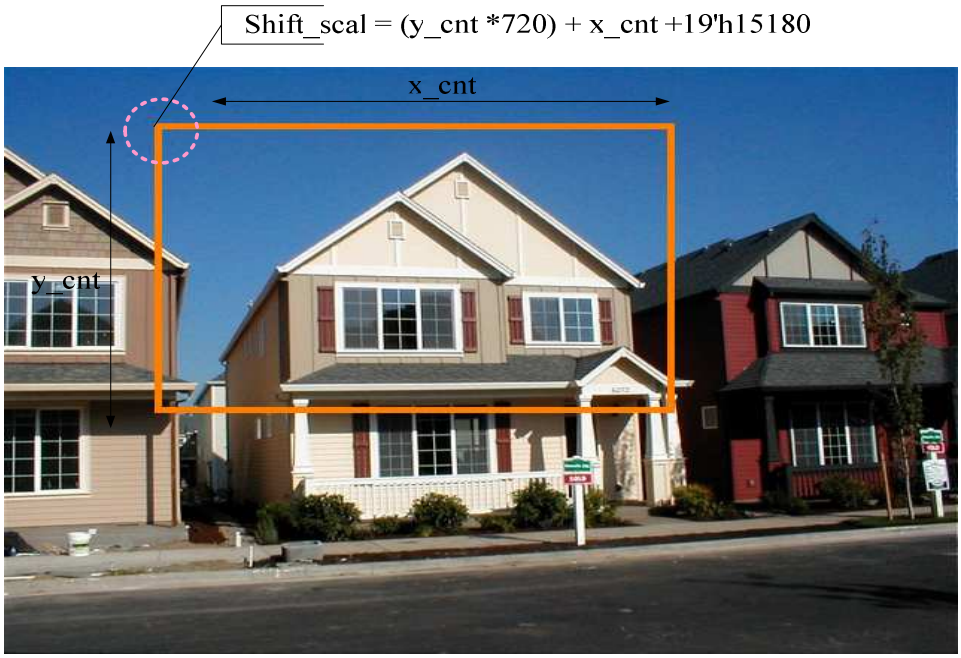


Fig. 3-1. 圖片放大於 TFT LCD 捲軸[28]

在邊緣可適應性彩色影像縮放硬體實現中，可以將擷取到的彩色圖片執行放大(Zoom in)或縮小(Zoom out)的功能，如圖3-2可以看出當硬體電路在執行放大時經由演算法，可以將原圖放大為2倍，也可縮回原圖。

Fig. 3-2. 彩色影像經由硬體電路做縮放[27]

3.2 硬體架構

如圖3-3，所示為整個邊緣可適應性彩色影像縮放的整個硬體架構，硬體的開發平台是採用Altera FPGA開發板，當在處理演算法中必需要有一張原始的影像為了能夠得從外部的周邊可以得到所要的彩色的影像，所以此系統會提供兩個影像來源，所以FPGA發展板中提供兩組的視訊介面，它們個別接到視頻解碼晶片(ADV7180)中，這一顆視頻解碼的晶片(ADV7180)針對輸入的類比的訊號如NTSC、PAL的標準訊號，再經由視頻解碼晶片(ADV7180)轉換出ITU-R.656標準數位影像格式，本系統是採用非交錯式NTSC格式來取得YUV彩色影像；另一個影像的來源是透過開發板中的Flash memory來取得彩色影像，可以透過FPGA開發板的USB介面將彩色影像(YUV 4:2:2)的資料從PC載入到Flash memory中；從上述的兩個影像來源可以透過多工器硬體電路來選擇要從那一端來源輸入，當資料一旦輸入都會先將影像資料透過高速同步靜態隨機記憶體(SSRAM)控制電路寫入SSRAM中，FPGA在顯示圖片或演算法的處理都會到SSRAM中讀取所需的影像資料。

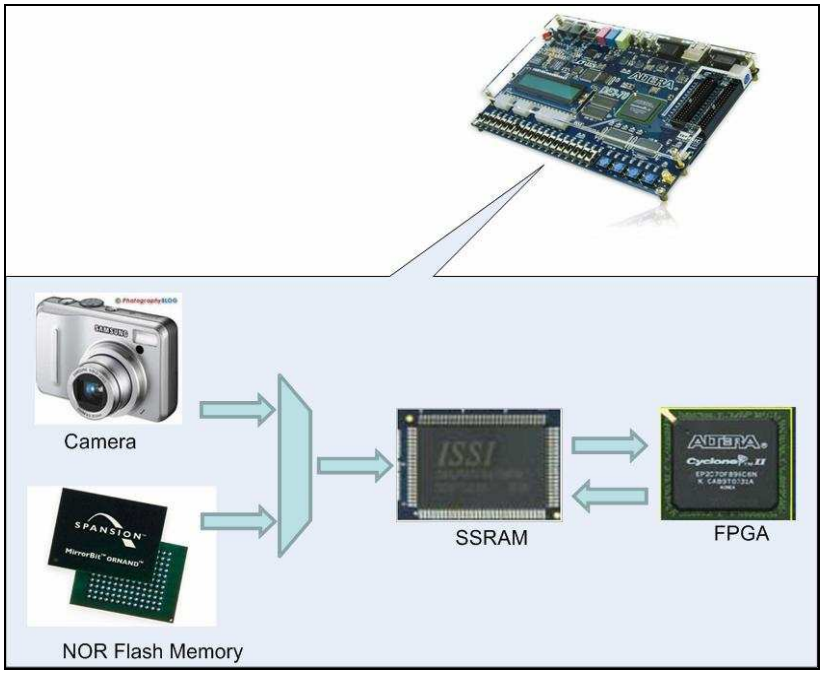


Fig. 3-3. FPGA 開發板及系統架構圖

如圖 3-4 所示為影像放大的硬體系統的架構圖，此系統架構圖詳細描述如下:

(1) 影像來源有 Flash 記憶體來存取從電腦所下載的圖片以及視訊解碼(ADV7180)的類比影像輸入。

(2) SSRAM 控制電路可控制高速外部 SSRAM 從 Flash 記憶體或 TV-Decode 取得所要的彩色影像資料，另外也可以提供 Fuzzy 演算法資料運算的讀取或 TFT LCD 顯示的資料。

(3) TFT LCD Controller 能夠控制 TFT LCD 顯示器來執行圖片的顯示。

(4) Fuzzy 演算法電路是此系統的核心，當系統要執行影像放大時，有限狀態機制(FSM Finite State Machine)會立即執行演算法的運算。

(5) 有限狀態機制(FSM Finite State Machine)經由一些輸入的判斷及輸出訊號來控制整個周邊控制電路及演算法的流程。

(6) Clock System 經由外部頻率的輸入並經由鎖向迴路(PLL)電路倍頻的輸出頻率來提供周邊控制電路及演算法的頻率。

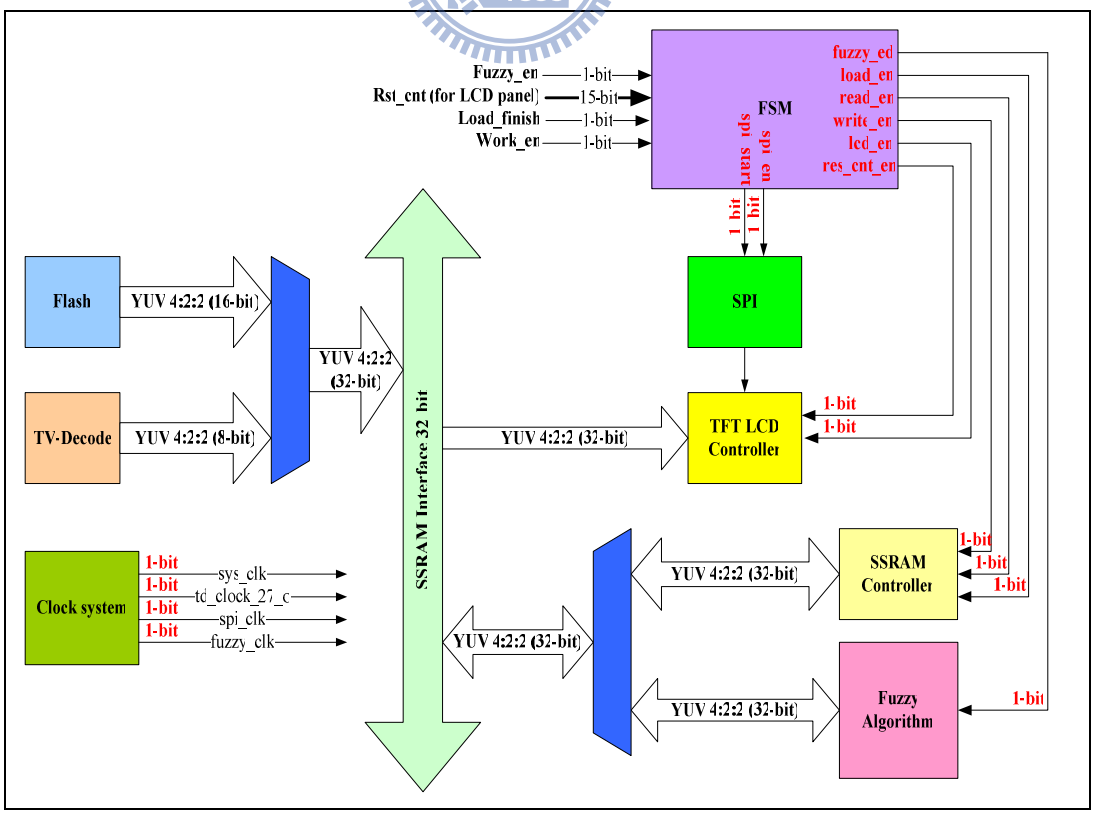


Fig. 3-4. 系統硬體方塊圖

如圖 3-5 所示為周邊電路的一個有限狀態機制(FSM Finite State Machine)從周邊的初始化開始來確保每一個控制電路能夠重置，當每一個周邊電路都做完初始之後，高速同步靜態隨機記憶體(SSRAM)控制電路，可以開始對資料做讀取或寫入，當演算法電路被啟動時，它會立刻對高速同步靜態隨機記憶體讀取影像資料到 FPGA 內部的記憶體中做暫時的存放，影像資料的讀取是以 Line(720×6)為一個單位，每一次的運算完的資料都會立即寫回 SSRAM，當演算法執行完之後，TFT LCD 控制電路可以透過高速同步靜態隨機記憶體控制電路的讀取方式將資料輸出到 TFT LCD 上。

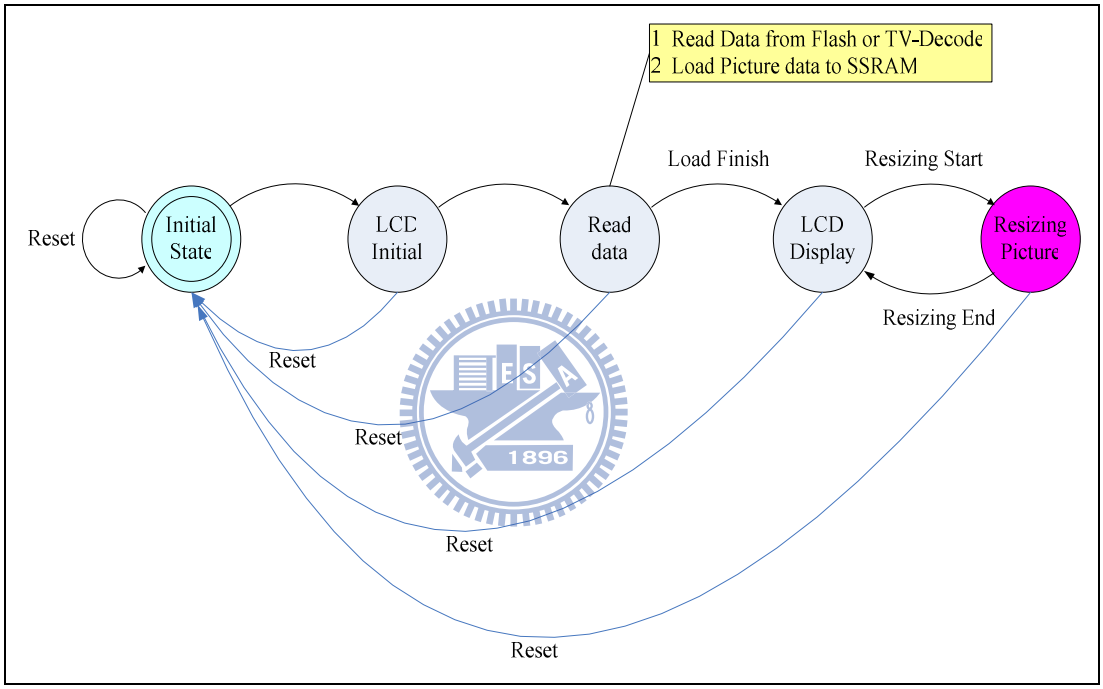


Fig. 3-5. 有限狀態機制 FSM (Finite State Machine)

在整個系統中的每一個周邊電路及演算法電路都會需要 Clock 來同步硬體電路，所以會有一個 Clock 控制系統如圖 3-6 所示，主要是透過外部的一個 50MHz 的石英振盪器輸入到 FPGA 的內部的 PLL 產生所需的頻率(130MHz)，再將 PLL 所輸出的頻率傳送到一個 8-bit 的除頻電路中，最後再依每一個周邊電路的需求來輸出所需的頻率，因為在系統中所使用到 TFT LCD 的頻率 27MHz 是比較特殊的頻率所以會另外從外部來提供另一顆 27MHz 的石英振盪器來供給，另外彩色影像來源會有兩組，所以 SSRAM 的 Clock 會有一組多工器來選擇頻率的來源，當選擇影像來源是選擇從視訊晶片那麼 Clock 會選

擇視訊晶片所輸出的 27MHz 頻率，反之當影像來源是選擇 Flah 記憶體時多工器就會選擇除頻器所輸出的頻率，在每一個周邊所需的頻率如下:

1. SPI Circuit 需要的頻率為 130MHz/8 = 16.25MHz
2. Fuzzy Algorithm 需要的頻率為 130MHz/2 = 65MHz
3. Flash Controller 需要的頻率為 130MHz/2 = 65MHz
4. SSRAM Controller 需要的頻率為 130MHz/2 = 65MHz (影像來源為 Flash Memory)
5. SSRAM Controller 需要的頻率為 27MHz (影像來源為 TV-Decode)

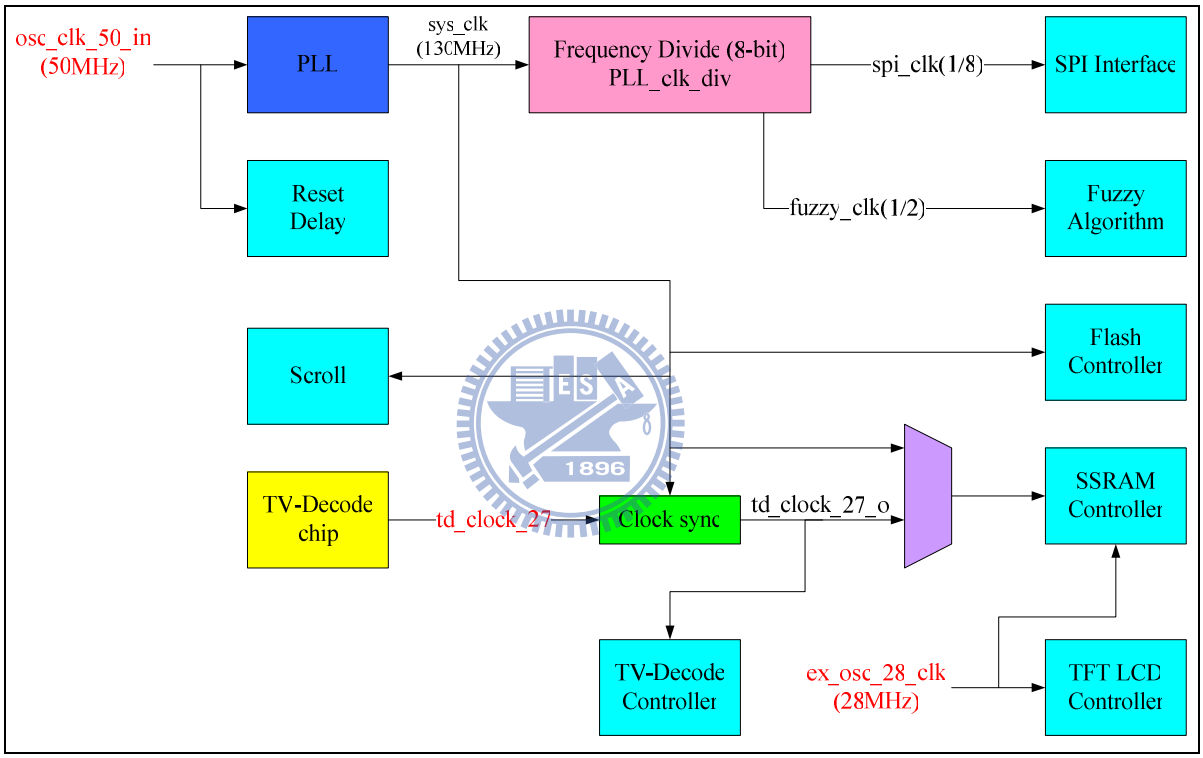


Fig. 3-6. Clock 系統方塊圖

3.3 彩色像素資料流程控制

當影像資料無論從Flash記憶體或視訊解碼晶片(ADV7180)擷取到的影像資料，都會立即寫入SSRAM，對於輸入資料的格式中上述的這兩種的影像來源都會有些許的不同，因為SSRAM的資料匯流排是32-bit，而Flash記憶體資料匯流排是16-bit，所以在存取操作中必需要對Data Buffer寫入兩次，另外一個來源是經由視訊解碼晶片所輸入的資料是8-bit，當在對SSRAM做寫入時必需對Data Buffer寫入四次，在第四次寫完之後才能一起寫入SSRAM，如圖3-7所示可知彩色影像從Flash記憶體或視訊解碼晶片(ADV7180)輸入，都會經由一個多工器來選擇那一個影像來源，硬體控制電路可以針對這兩種不同的來源，選擇那一種SSRAM的寫入方式(針對Data Buffer寫入兩次或四次)，另外在SSRAM的讀取部份是將4-byte資料一次輸出到Data Buffer，再由YUV資料的分配控制電路將像素分配到Y Pixels內部RAM、U Pixels內部RAM及V Pixels內部RAM，影像資料的操作的詳細流程在下一小節中會說明。

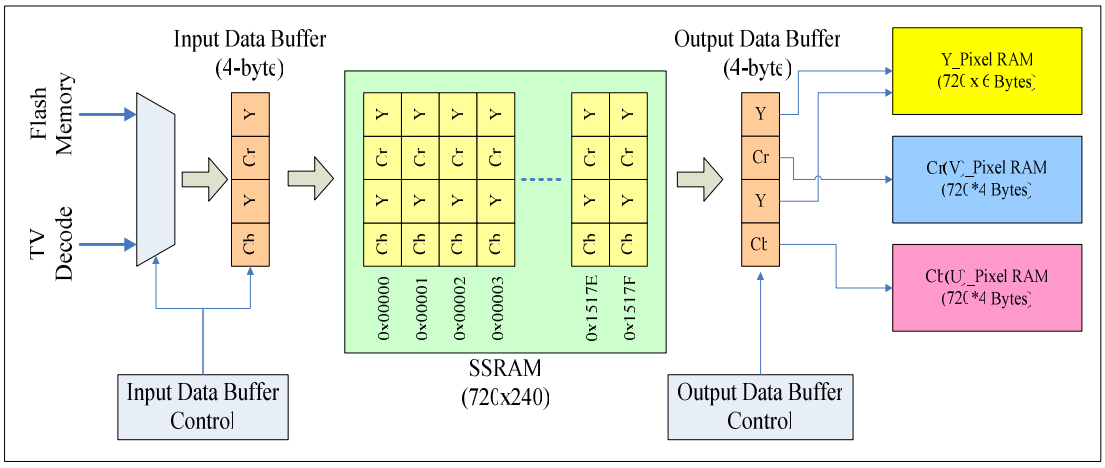


Fig. 3-7. 影像資料處理架構圖

3.3.1 Flash 記憶體影像來源操作

操作流程如圖3-8所示，就是Flash記憶體的操作流程圖，詳細的操作流程如下:

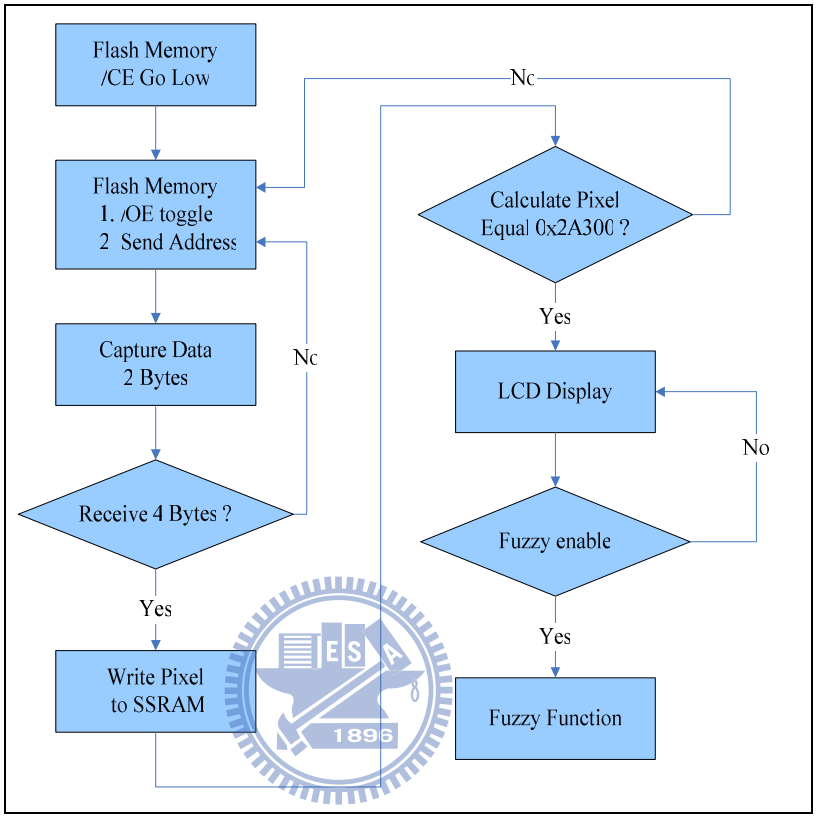


Fig. 3-8. Flash 記憶體流程圖

1. 在系統始初化之後Flash記憶體的晶片致能控制信號(CE)會先啟動(1→0)。
2. Flash控制電路送出位址(Address) Flash記憶體，最後由輸出致能控制信號啟動(1→0)。
3. 當16-bit像素(Pixels)資料從Flash記憶體輸出時，會先被寫入到內部的32-bit暫存器，接下來重複**2.**的做法再將16-bit像素資料的寫入32-bit暫存器。
4. 如果已接收到32-bit的像素資料後可以寫入SSRAM中，SSRAM控制電路提供一組位址計數器，每寫一筆32-bit的像素資料就會計數一次，如果還有資料未寫完電路的動作會回到**2.**繼續讀取Flash記憶體的資料。
5. 720 × 240的影像資料寫完之後，TFT LCD控制電路以32-bit為單位將像素資料從

SSRAM中讀取出來，最後再以8-bit為單位寫入TFT LCD顯示畫面。

6. 如果影像資料已寫入SSRAM後，只要啟動演算法硬體電路，就能夠執行影像縮放的功能。

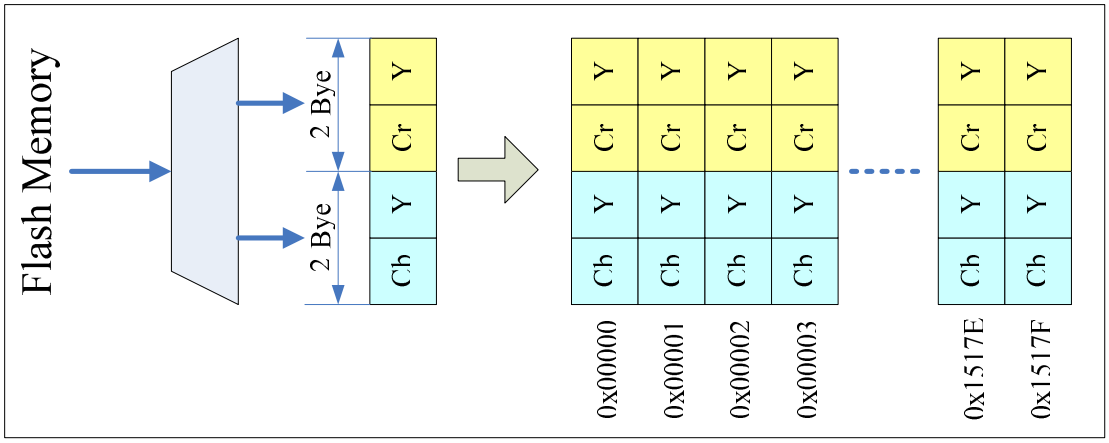


Fig. 3-9. Flash 記憶體資料及 SSRAM 資料配置

3.3.2 視訊解碼晶片影像來源操作

另一個影像來源是經由視訊解碼晶片(ADV1780)所輸入的影像資料，因為輸入的影像資料為 720 × 240 YUV 4:2:2 格式，而且在資料的輸入是以 8-bit 為一個單位，所以在資料處理上必需要對 SSRAM 的 Data buffer 做四次的寫入，操作流程如圖 3-10 所示，為視訊解碼的資料擷取操作流程圖，詳細的操作流程如下:

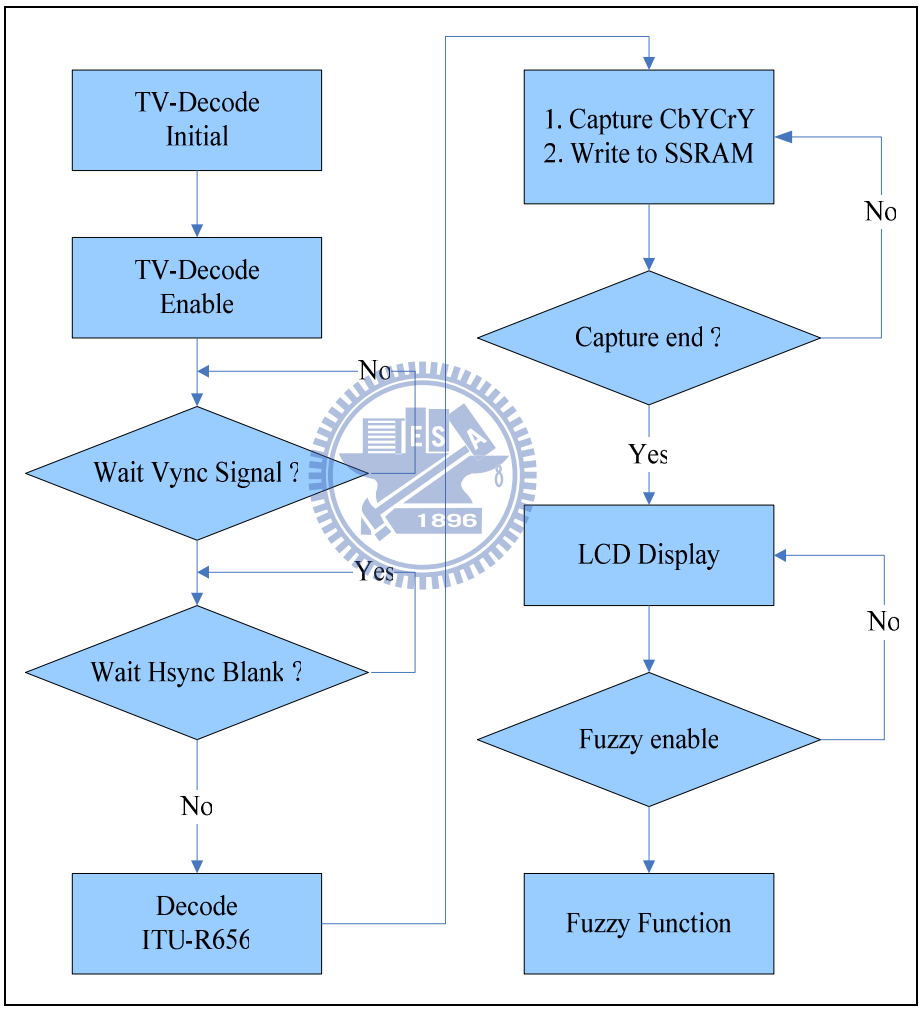


Fig. 3-10. 視訊解碼的資料擷取操作流程圖

1. 在系統始初化及I2C的介面對視訊解碼晶片做設定之後，控制電路會先等待視訊解碼晶片(TV-Decode)的垂直同步信號。
2. 當取得垂直同步信號之後，接下來等待水平同步信號及Blank，接下來就是對視訊解碼晶片(TV-Decode)所輸出的ITU-R656格式做解碼。

3. 接下來開始擷取YUV的影像資料，因為從視訊解碼晶片(TV-Decode)中，所接收的資料是以8-bit為一個單位，在寫入SSRAM之前必需先將32-Bit的Data buffer先填滿四次，再寫入SSRAM中。
4. SSRAM介面電路提供一組位址計數器，每寫一筆32-bit的像素(Pixels)資料就會計數一次，如果還有資料未寫完電路的動作會回到**3.**繼續擷取的視訊解碼晶片(TV-Decode)輸入的資料。
5. 720×240的影像資料寫完之後，TFT LCD控制電路以32-bit為單位將像素資料從SSRAM中讀取出來，最後再以8-bit為單位寫入TFT LCD顯示畫面。
6. 如果影像資料已寫入SSRAM後，只要啟動演算法硬體電路，就能夠執行影像縮放的功能。

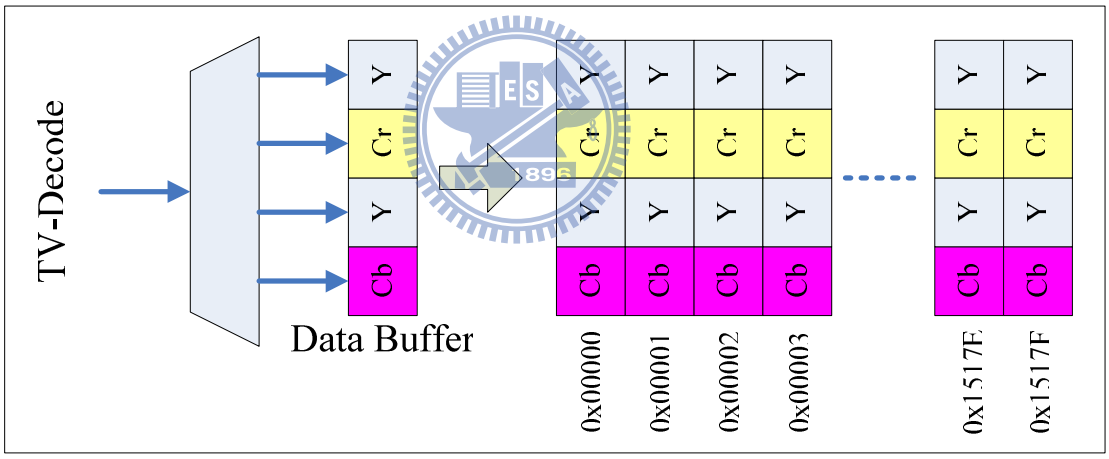


Fig. 3-11. 視訊解碼晶片(TV-Decode)輸出資料及 SSRAM 資料配置

3.3.3 SSRAM 存取操作

在演算法的處理或 TFT LCD 顯示圖片資料的立即性成為一個重要的問題，為了能夠使得大量的資料可以快速的獲得，所以在系統中有提供了一顆高速同步靜態隨機記憶體(SSRAM)，外部的影像來源所擷取到的影像都會先暫時存放在這一顆高速同步靜態隨機記憶體(SSRAM)中，在 SSRAM 控制電路中會需要位址匯流排(Address Bus)、資料匯流排(Data Bus)及一些控制信號(Control Signal)如圖 3-12 所示為 SSRAM 介面電路方塊圖，硬體主要是由 SSRAM 位址計數器電路來輸出位址(Address)，資料的讀寫主要是由資料方向電路(Data Direction Circuit)以及 SSRAM 控制(SSRAM control)來決定要寫入資料到 SSRAM 或從 SSRAM 讀取資料，因為對 SSRAM 存取時會需一些控制信號，所以也會有一組控制信號電路(Control Signal Circuit)來對 SSRAM 做一些讀寫以及致能信號的控制。

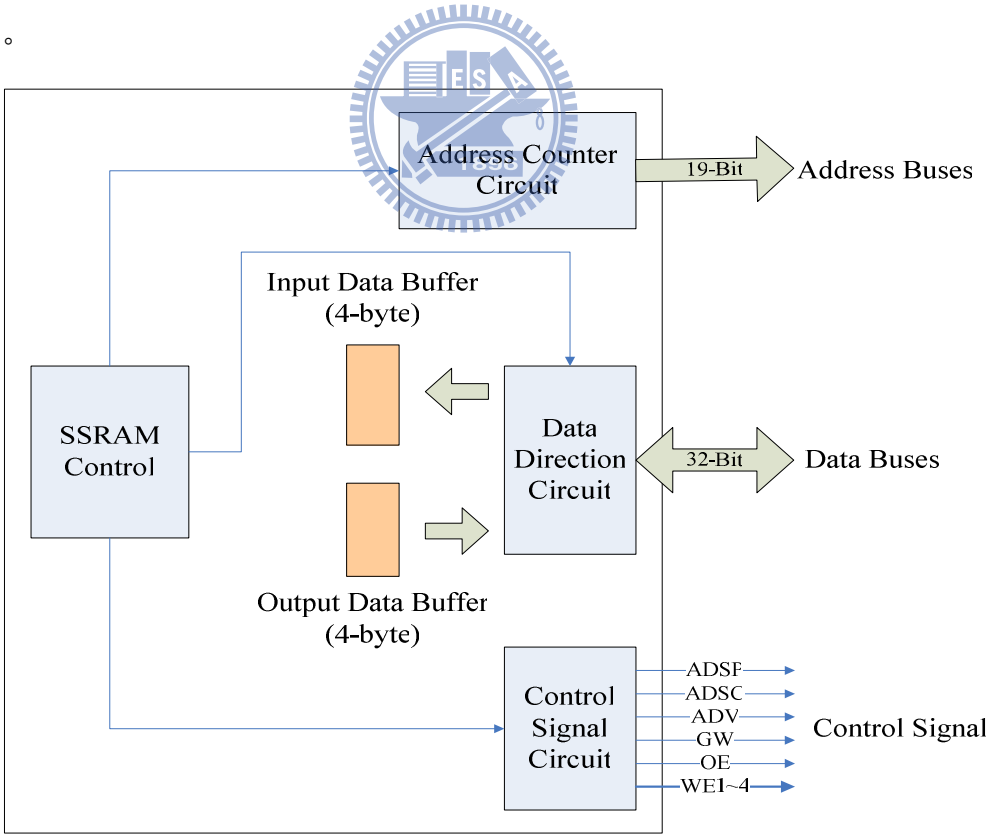


Fig. 3-12. SSRAM 介面電路方塊圖

在影像的立即資料的取得從兩個影像來源的寫入到讀取 TFT LCD 控制電路的讀取可以透過如圖 3-13 所示，可以了解 SSRAM 的寫入及讀取的流程。

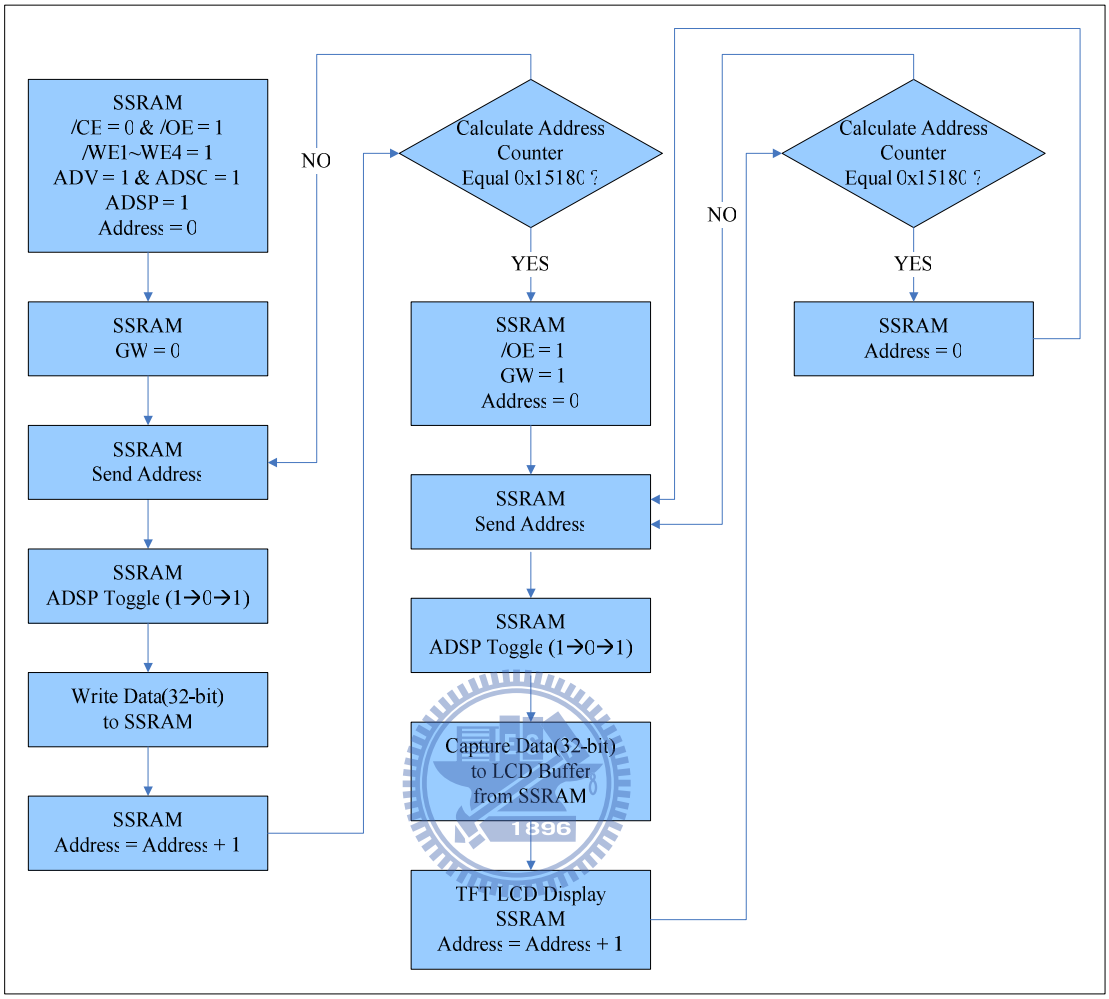


Fig. 3-13. SSRAM 存取資料操作流程

1. 在系統始初化的同時，SSRAM介面電路會對SSRAM所需的一些控制信號做設定，並且位址(Address)會先清除為零。
2. 當要對SSRAM寫入資料時必需先將/GW控制信號設定為零，接下來送出位址(Address)信號，最後對ADSP送出1→0→1的信號，告知SSRAM已經把位址送出。
3. 在送完位址(Address)信號及設定完ADSP信號之後，可以立即對SSRAM寫入資料，接下來再對位址(Address)加一，並且判斷是否已寫完整張圖片的資料(720 × 240)，如果還沒有全部寫入重複**2.**的做法。
4. 如果圖片(720 × 240)已全部寫入SSRAM之後，TFT LCD控制電路會對SSRAM讀

取所要顯示的影像資料，SSRAM介面電路會將/GW設為1，並將/OE設為0(讀取控制信號)。

5. TFT LCD控制電路以32-bit為單位將像素(Pixels)資料從SSRAM中讀取出來，最後再以8-bit為單位寫入TFT LCD顯示畫面，從SSRAM讀取32-bit的資料之後，SSRAM的位址加一並判斷是否已顯示完整張圖片的資料(720 × 240) 如果還沒有全部寫入重複**5.**的做法。
6. 只要TFT LCD顯示影像資料，就會一直不斷向SSRAM讀取資料。

3.3.4 Line Buffer 資料處理操作

從圖 3-14 所示，像素的處理中從外部記憶體所讀取到的資料必需存放在由 FPGA 內部所提供的內部記體中，稱之為 Line Buffer。因為所要影像處理的像素為 YUV 的格式，所以 FPGA 也針對了這個需求提供三組獨立的 RAM 來暫時存放 Y(亮度)、U 及 V(色差)，因為從 SSRAM 中所讀取到的資料一次為 4-byte，所以 SSRAM 的讀取電路會提供一個 4-byte 的輸出 Data buffer，再經由一個 YUV 的分配控制電路來做分配，將 YUV 的像素分配到自己所屬的內部記憶體中，在資料存取的同時內部記憶體(Line Buffer)的位址也會自動計數。

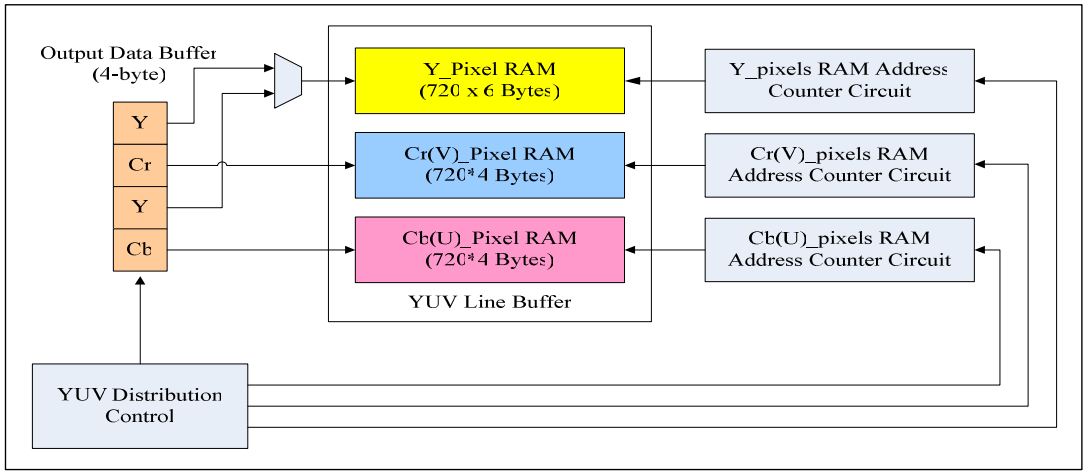


Fig. 3-14. YUV 像素資料分配 RAM(Line Buffer)

在 YUV 的格式中以 Y 像素(亮度)的成份對於人類視覺最為敏感，所以演算法中的模糊決策系統只針對 Y 像素來做計算，而最後的補插點 Y 像素經由可適應內插法或雙線性內插法來做補點，而 U 像素及 V 像素透過雙線性內插法來做補點；當彩色影像資料個別寫入 YUV Line Buffer 中以後，演算法的硬體電路會開始對 YUV Line Buffer 中的資料做讀取，其中因為視覺對於 Y 像素(亮度)的敏感程度比起 U 或 V 更為重要，所以在演算法中的模糊決策系統只針對 Y 像素做運算，每一次的計算中內部記憶體的控制電路會對 Y Line Buffer 讀取 6×6 的影像區塊資料，因為在演算法的運算過程中是以 4×4 的影像區塊為一個單位，而在這個區塊的每一個像素都會使用到角度的計算，故每一個原點像素都必需取得它相鄰邊的像素來做運算，故每一次的讀取都需要 36 個像素，當在每一次的補插點運算完畢之後，要處理下一個補插點的運算時，整個 6 × 6 的影像區塊資料會向右水平的方向移動一個像素的間格，再繼續執行演算法的計算如圖 3-15 所示為 Y 像素的影像區塊資料的水平移動。

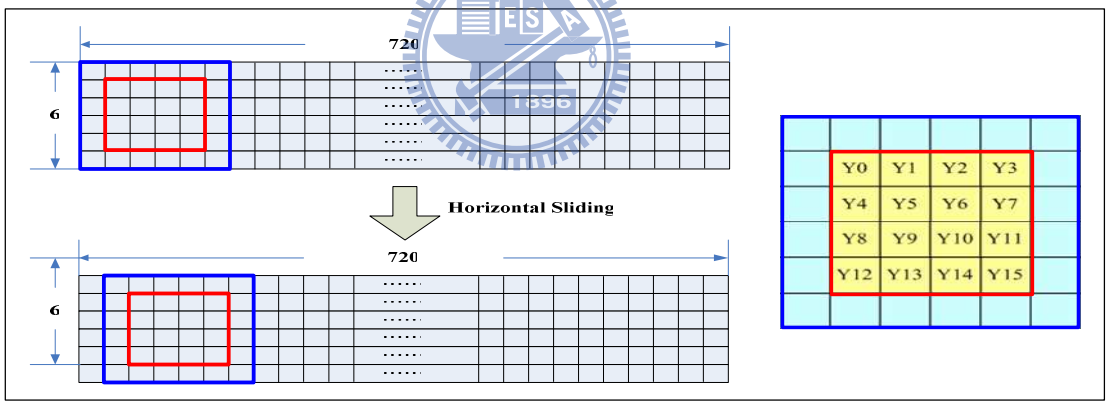


Fig. 3-15. Y 像素 Line Buffer 執行水平移動

此外從圖 3-16 所示在 Line Buffer 的二維平面上可以看出在每一次水平的移動(Horizontal Sliding)在完成 715 次的補插點之後，會對垂直方向移動一條 Line 的間格，所以每一次的垂直移動之前必需計算出下一個水平移動的起始點位置，故在硬體的實現中，彩色像素是存放於外部的 SSRAM 中，故需將二維的方式轉換為一維來計算每一次的垂直移動量，可以從圖 3-17 所知，每一次的垂直的移動硬體電路都會根據下一條 Line 的補插點來計算出一個垂直的起始的移動量。

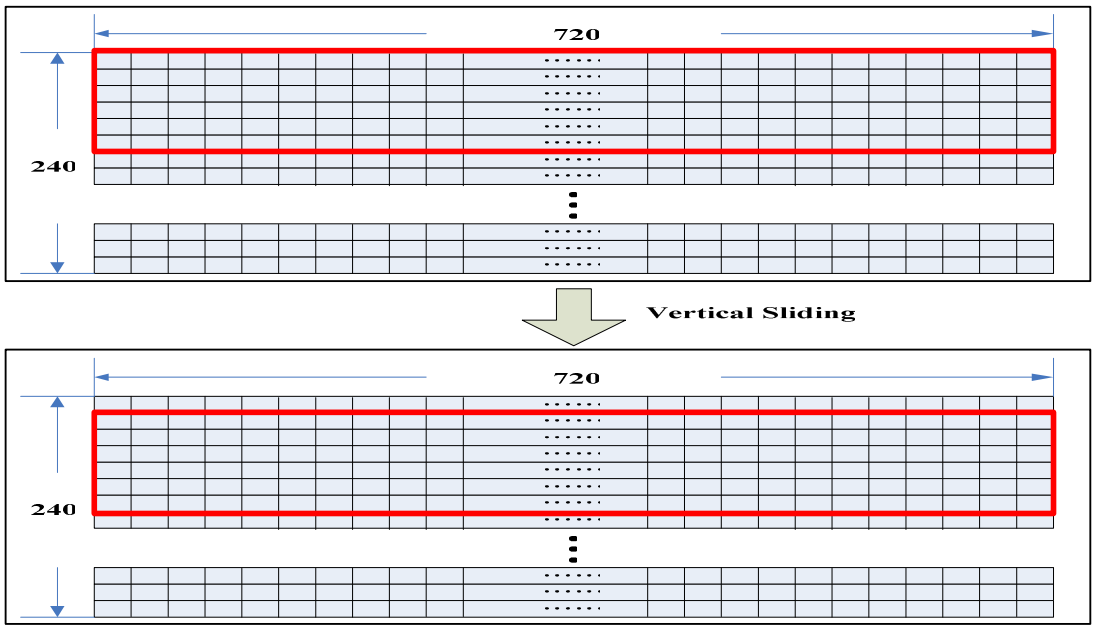


Fig. 3-16. Y 像素執行垂直移動

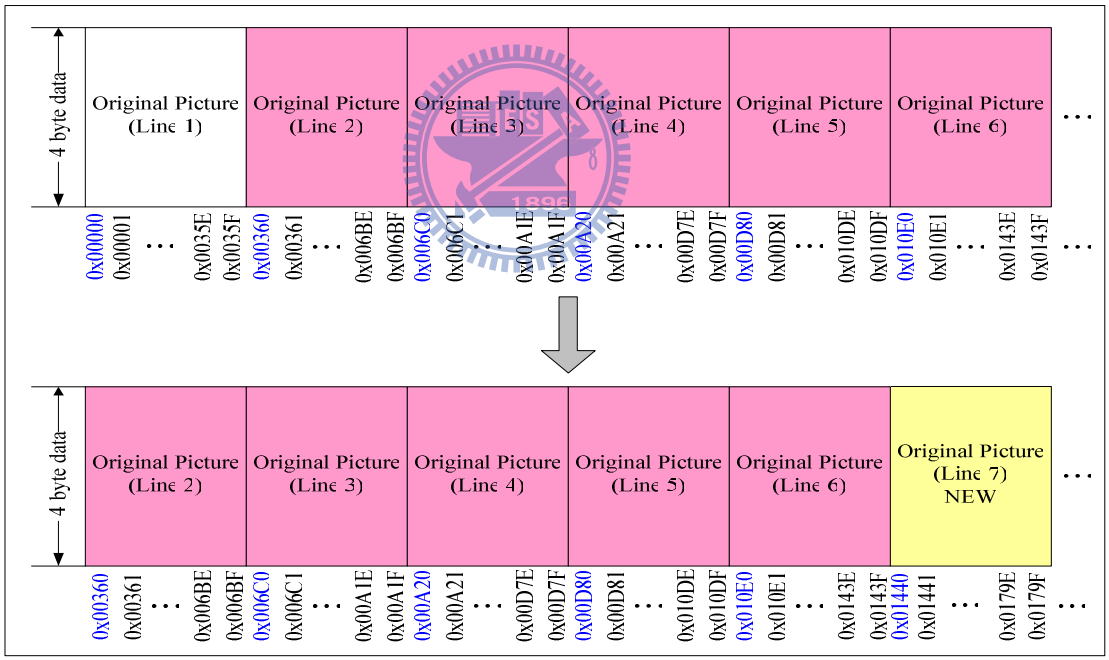


Fig. 3-17. 垂直移動於一維 SSRAM 記憶體配置

3.4 演算法實現

3.4.1 演算法硬體架構

從圖3-18的演算法架構方塊圖中可以得知，從影像的來源端輸入YUV(4:2:2)的彩色影像，硬體電路將Y像素、U像素及V像素存入各自的Buffer中，模糊決策系統硬體電路會對Y像素做運算，演算法電路會根據運算出來的結果來判斷是否執行雙線性內插法或邊緣適應性影像內插法，因為在模糊決策系統中的三種運算(VD、SD、CD)的輸出都關係到4×4的滑動區塊最後補插點結果，運算出來的結果，如果是人類眼睛較敏感的部份，演算法電路會更進一步對邊緣的方向執行角度的運算，演算法硬體實現中角度的取得是採用CORDIC硬體電路實現，從CORDIC硬體中得到角度輸出後，接下來是做直方圖，在4×4的滑動區塊中的16個Y像素都會有自己的一個角度範圍($k=0$~7)，硬體電路取得角度範圍所累計最高的值做為權重查表的位址，經由Y像素及權重可以乘加計算出所要的補插點，反之如果模糊決策系統輸出的結果是人類眼睛較不敏感的部份，只需要使用雙線性內插法即可；另外U像素及V像素部份也只需要雙線性內插法。

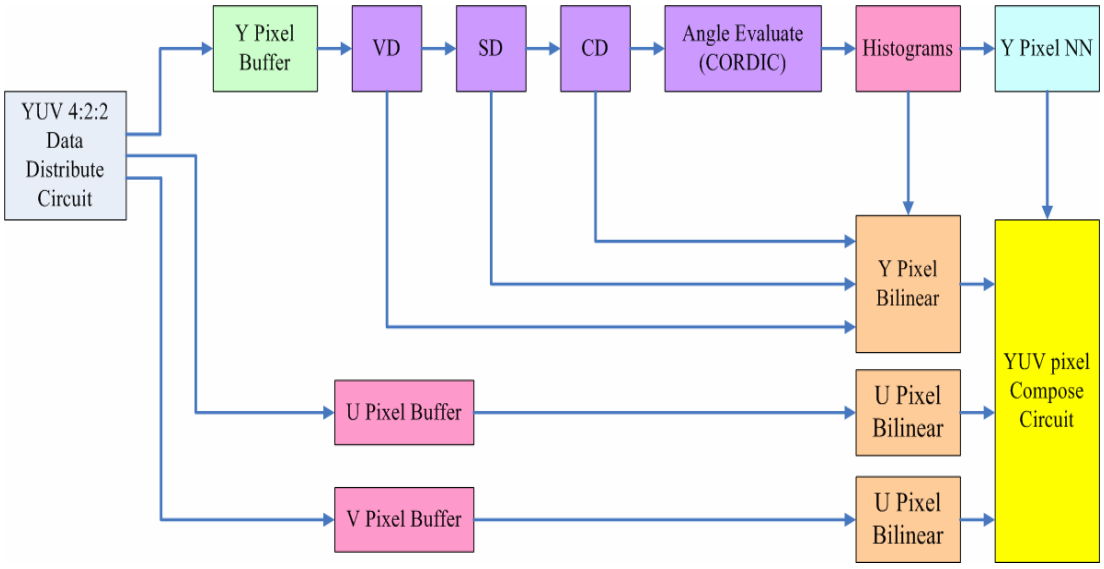


Fig. 3-18. 演算法架構方塊圖.

3.4.2 Fuzzy Decision 硬體實現

有模糊決策系統的系統會有三種針對4×4的滑動區塊的演算法模組，有Visibility Degree（VD）、Structure Degree（SD）和Complexity Degree (CD)。硬體詳細的描述如下：

(1) Visibility Degree模組

Visibility Degree（VD）的計算式子如(3.1) (3.2) (3.3)，

$$VD = D - V(BL), \tag{3.1}$$

$$D = \max(O(i,j)) - \min(O(i,j)) \tag{3.2}$$

$$V(BL) = 20.66e^{-0.03BL} + e^{0.008BL}. \tag{3.3}$$

從過去灰階架構中設計可以看出VD硬體的實現方式，是將4×4的影像區塊中的16個像素找多最亮像素點及最暗像素點相減並取得差值(D)，原來式(3.3)的V(BL) Visibility Threshold的數值不再以指數的方式來表示，而是透過演算法實驗後的測試結果來取得一個固定的臨界值，再和差值(D)做相減所得到VD值如式(3.4)，此種做法可能會造成在邊緣的處理上出現些許的鋸齒情形，並不能將邊緣的部份處理的更為平滑，圖3-19所示，為灰階架構中的VD硬體方塊圖。

$$VD = D - Threshold, \tag{3.4}$$

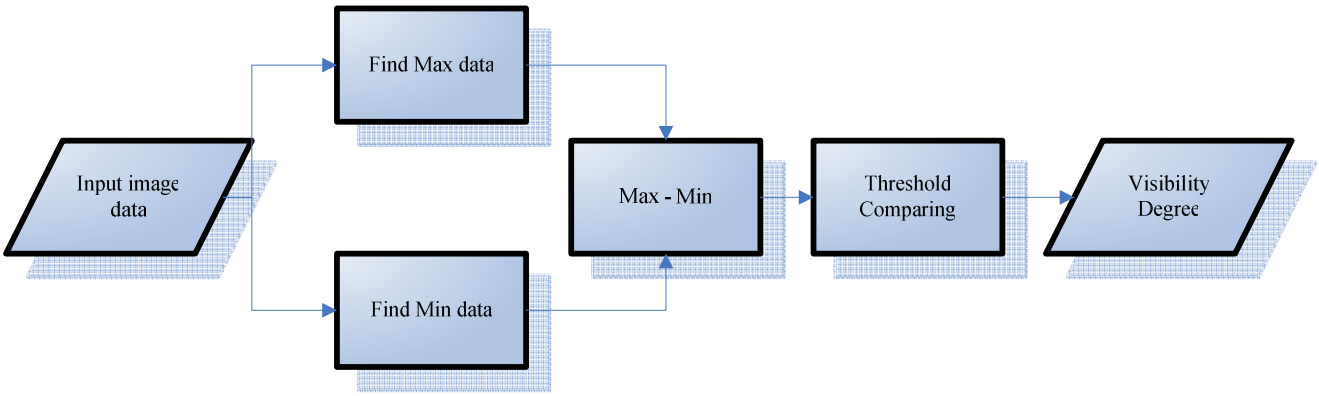


Fig. 3-19. 灰階架構的 VD 硬體架構方塊圖 [21]

在彩色架構中VD做法所使用的做法是在從背景及Visibility Threshold的不同來辨別物件於背景上的辨識能見度，所以透過式(3.3)來求得V(BL)的函數，因為在V(BL)的式子中會使用指數函數，在一般硬體電路的實現上較為困難，所以在背景亮度(Background Luminance BL)的有限數值範圍中可以透過查表的方式來實現，背景亮度的範圍是BL=0~255，所以V(BL)的輸出會有256種不同的可能性，而將浮點數轉為定點數所需的精確度是採用8-bit是方式來處理，所需的記憶體容量為256 × 8 bits，在得到V(BL)的輸出後再與式(3.2)最亮點及最暗點做差值的相減如式(3.1)就會得VD的結果，如圖3-20所示為彩色架構的VD硬體方塊圖。

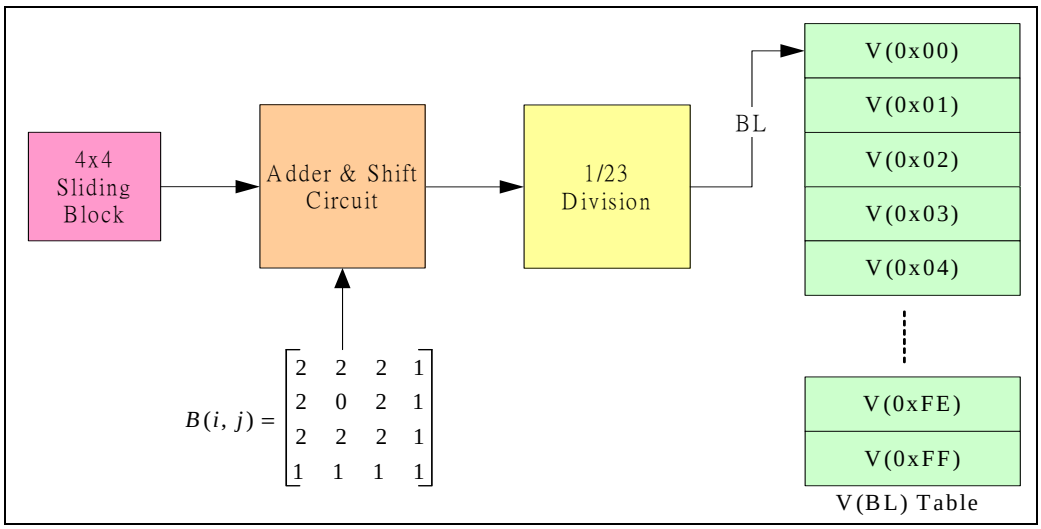


Fig. 3-20. 彩色架構中的 VD 硬體方塊圖

因為在V(BL)函數中如式(3.3)會必須先求得背景亮度，如式(2.5)，主要是從4×4的影像區塊中的16個像素及背景亮度的參數矩陣做乘加的運算，並在最後做除23的平均，所以在硬體的實現除法器會利用加法器來處理商數的累加及減法器執行除法的功能，在這個除法電路中會使用到一組除數比較表每一組的除數比較值都會對應一組商數如表3-1。

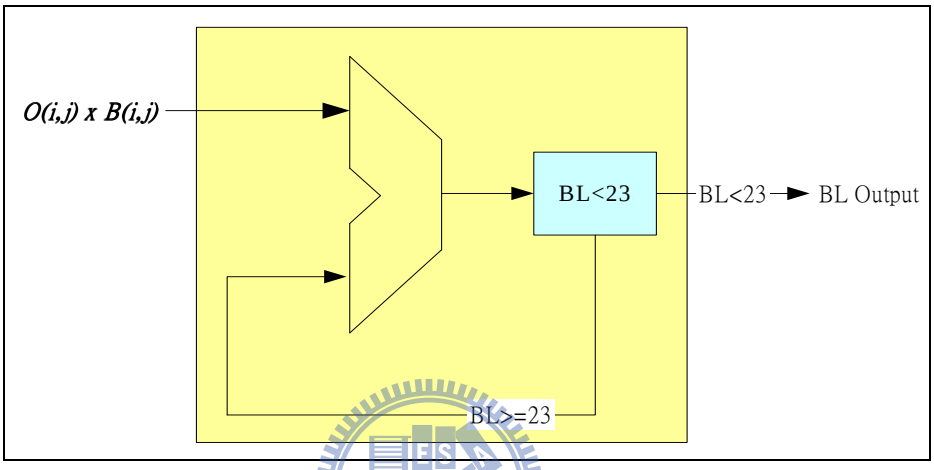


Fig. 3-21. 背景亮度(BL)硬體方塊圖

Table 3-1.除數及商數查表

Divider 23 Table		
	除數比較表	商數
23*128	2944	128
23*64	1472	64
23*32	736	32
23*16	368	16
23*8	184	8
23*4	92	4
23*2	46	2
23	23	1

(2) Structure Degree Module

Structure Degree（SD）的計算式子如(3.5)，

$$SD = \frac{\left|\max(O(i,j)) - mean(O(i,j)) - \left[mean(O(i,j)) - \min(O(i,j))\right]\right|}{\max(O(i,j)) - \min(O(i,j))} \tag{3.5}$$

在式（3.5）中，從軟體的實驗中得知在SD所計算出來的結果只有兩種情形，就是SD≥1或SD<1，如果小於1繼續執行CD，否則執行雙線性內插法，所以在硬體的實現中不需對小數點做處理，硬體的實現如圖3-22所示從4×4的滑動區塊中會取出16個像素中的最大值及最小值，並將16個像素透過加法器執行加法的運算最後將16個像素的總和取平均，輸出結果是經由一個比較器電路判斷如式(3.5)中分子和分母的比較來取得SD。

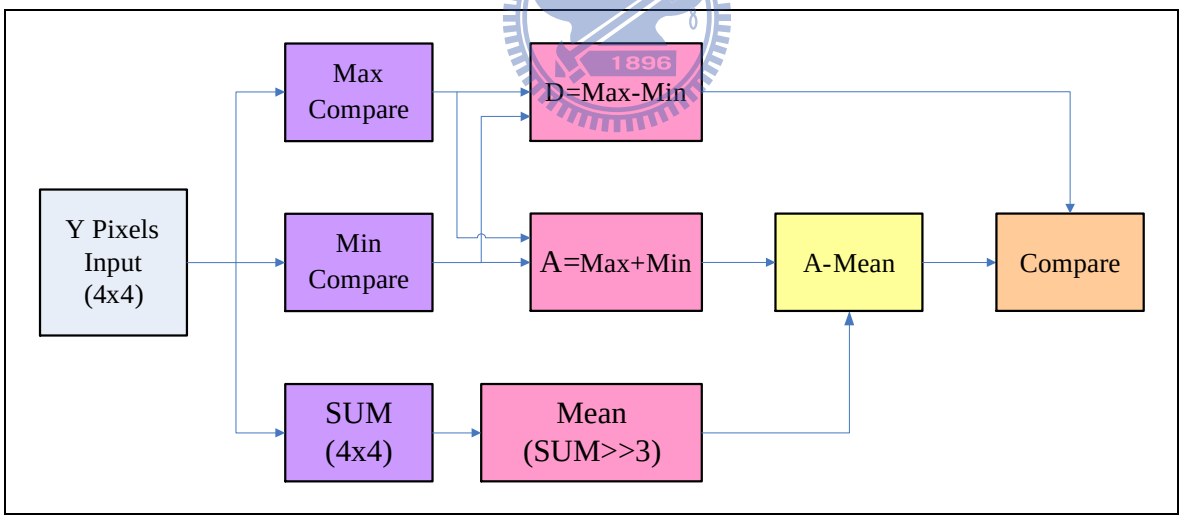


Fig. 3-22. SD 硬體實現方塊圖

(3) Complexity Degree Module

Complexity Degree計算是整數運算所以沒有小數點的問題在硬體的實現上較為簡單，CD計算的開始會先將4×4的滑動區塊中的16個像素$O(i, j)$做適應性二值化$O'(i, j)$，因為影像在取得適應性二值化的結果之前，會需要適應性臨界值(Adaptive Threshold)來做比較，所以將16個像素的和再取平均就是所要的適應性臨界值(Adaptive Threshold)，當像素的值≧適應性臨界值(Adaptive Threshold)輸出的值就為1，反之輸出為0。如果16個像素做適應性二值化$O'(i, j)$結束後，就執行Laplacian運算如式(3.6)，而CD運算的硬體實現方式是利用一些互斥或邏輯及加法器來完成如圖3-23為CD硬體實現方塊圖。

$$CD = \sum_{i=0}^{3}\sum_{j=0}^{3}\left|4O'(i, j) - [O'(i+1, j) + O'(i-1, j) + O'(i, j+1) + O'(i, j-1)]\right| \tag{3.6}$$

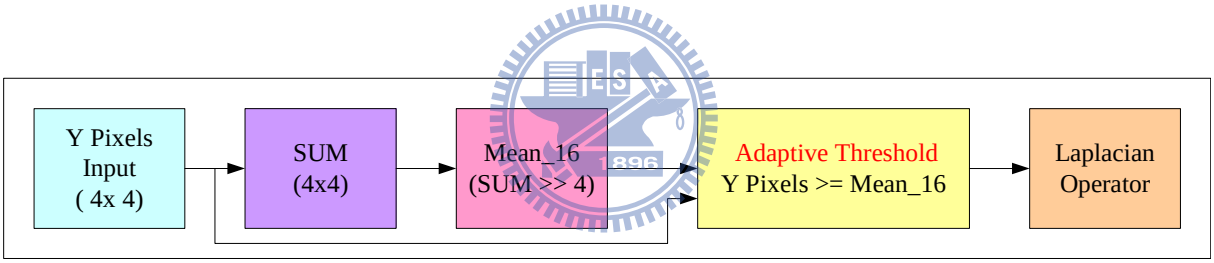


Fig. 3-23. CD 硬體實現方塊圖

3.4.3 Angle Evaluation 硬體實現

角度的計算的開始會先將4×4的滑動區塊中的16個像素做索貝爾運算如式(3.7)及(3.8)，在每一每像素的周圍都有自己相鄰近的點，可以藉由原點像素和鄰近像素執行X方向及Y方向的計算，最後將D_x及D_y的索貝爾運算輸入到反正切函數(Arc tangent)中，來求得邊緣的方向角度$A(i, j)$如式(3.9)及圖3-24所示，每一次D_x及D_y輸出經由鄰邊(X軸)及對邊(Y軸)的關係來求得夾角。

$$\begin{aligned} D_y(i, j) = {} & O(i-1, j-1) + 2O(i, j-1) + O(i+1, j-1) \\ & -(O(i-1, j+1) + 2O(i, j+1) + O(i+1, j+1)) \end{aligned} \tag{3.7}$$

$$\begin{aligned} D_x(i, j) = {} & O(i-1, j-1) + 2O(i-1, j) + O(i-1, j+1) \\ & -(O(i+1, j-1) + 2O(i+1, j) + O(i+1, j+1)) \end{aligned} \tag{3.8}$$

$$A(i, j) = \frac{180}{\pi}[\tan^{-1}(\frac{Dy(i, j)}{Dx(i, j)})] \tag{3.9}$$

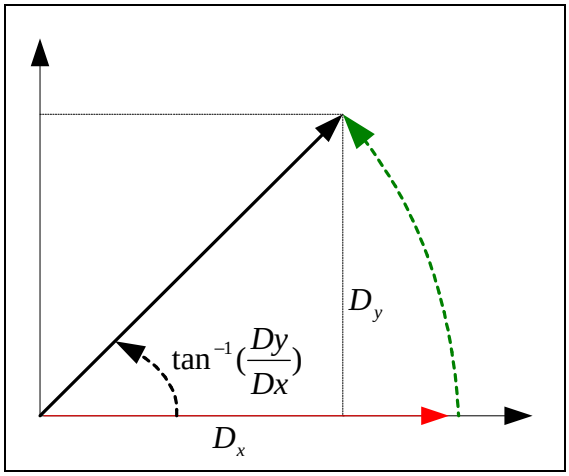


Fig. 3-24. $A(i, j)$ 及 D_x 及 D_y 關係曲線

(1) CORDIC Module

反正切函數可以藉由 CORDIC(Coordinate Rotation Digital Computer)硬體電路來實現，CORDIC(Coordinate Rotation Digital Computer)[22]演算法，是由 J. Volder 於 1959 年提出，首先用於導航系統，使得向量的旋轉和定向運算不需要做查三角函數表、乘法、開方及反三角函數等複雜運算。它的概念主要是利用一組簡單的查表、加減法器及移位電路所組成，CORDIC 是經由這三種簡單電路以階層計算的方式來組合而成，每一階層都會有一組固定的角度參數、加減法器及移位電路，首先將索貝爾運算如式(3.7)及(3.8)的運算結果 D_x 及 D_y 輸入到 CORDIC 的第一層如圖 3-25 所示得知，輸入的 D_x 及 D_y 都會做數值的移位及加減法的運算，控制電路會針對 D_x 及 D_y 的最高位元的正負符號來選擇加減法的運算，如果是正號選擇加法器，反之使用減法器，另外角度的查表規則也是根據 D_x 及 D_y 最高位元的正負符號來選擇角度為正角度或負角度，當所輸入的 D_x 及 D_y 值經過每一階層的如上述的運算後到了最後一層就會得到所要的輸出角度；在反三角函數中的計算結果會有小數點精確度的問題，若希望 CORDIC 硬體電路輸出的角度可以和實際值的誤差可以更小，所需的計算階層就必需更多，所以演算法的電路在不影響電路面積的情況之下，所使用的階層數為九階(i = 0~8)，如式(3.10)所示為 CORDIC 的數學運算式 z_{i+1} 代表每一階層的角度輸出，而 $D_{x(i+1)}$ 及 $D_{y(i+1)}$ 代表每一階層的 D_x 及 D_y 的輸出，d_i 代表每一階層 D_x 輸出的最高位元的符號數，i 代表階層數。

$$\begin{aligned} D_{x(i+1)} &= K_i(D_{x(i)} - (D_{y_{(i)}} d_i 2^{-i})) \\ D_{y_{(i+1)}} &= K_i(D_{y_{(i)}} + (D_{x(i)} d_i 2^{-i})), \qquad d_i = \begin{cases} +1 & if\ y_i < 0 \\ -1 & otherwise. \end{cases} \\ z_{i+1} &= z_i - d_i \tan^{-1}(2^{-i}) \end{aligned} \tag{3.10}$$

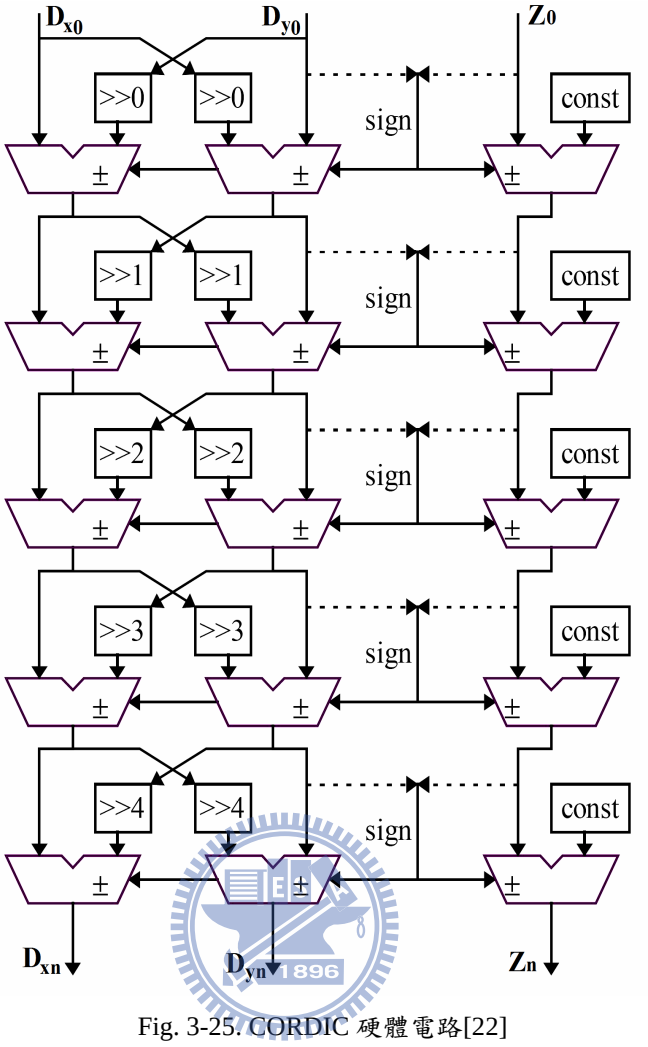


Fig. 3-25. CORDIC 硬體電路[22]

在CORDIC硬體電路中，會需要一組固定的角度參數(Zi)如表3-2，來提供每一階層輸出所要的查表，而每一次所查到的角度參數(Zi)，硬體電路都會依$D_{y(i)}$所輸出的最高位元的符號數來取得角度參數(Zi)的正負向位做角度參數(Zi)的相加或相減，因為在軟體的模擬中，角度參數(Zi)的建立是以浮點數為主，為了保留小數點後的解析度在硬體上的實現會採用16-bit的定點方式(1)，所以在CORDIC的九階的角度查表中，每一組的浮點數角度必需乘上2^{16}將小數點向右邊移動16-bit，因為所求的角度的範圍為$0^{\circ} \sim 180^{\circ}$故需再除上180；在CORDIC硬體中已計算出角度後，如果輸出的角度為 $A(i, j) \geq 180^{\circ}$，CORDIC控制電路會執行$180^{\circ} - A(i, j)$才會得到實際的夾角。

Table 3-2.角度參數(Zi)查表

Float Point(SW)	Fixed Point(HW)
45 Degree	0x4000
26.565 Degree	0x25C8
14.036 Degree	0x13F6
7.125 Degree	0x0A22
3.576 Degree	0x0516
1.7899 Degree	0x028B
0.8951 Degree	0x0145
0.4476 Degree	0x00A2
0.2238 Degree	0x0051

(1) (14.036 Degree x 65536)/180 = 0x13F6

從表3-3所示為硬體計算後之結果與實際值的比較表，從實驗的結果得知RMSE (**Root Mean Square Error**)範圍在0.19384。

Table 3-3.角度計算使用硬體及實際值誤差比較

Software Result	Hardware Result	Error of Result
17.76479 degree	17.55889 (18F9H)	0.2059
12.41738 degree	12.64251 (11FBH)	0.22513
28.0202 degree	27.83064 (278BH)	0.18956
81.52882 degree	81.37847 (73BDH)	0.15035
25.307923 degree	25.11749 (23B9H)	0.190433

(2) 直方圖(Histogram)

在16個像素經由角度的計算完之後，直方圖會累計這16個角度出現機會最多者，角度的範圍為$\theta = (22.5\times k), 0 \leqq k \leqq 7$共有8個範圍，所以電路會針對像素角度的範圍做判斷，舉例來說：其中的一個像素角度為$\theta = 59$ degree它的範圍在22.5×2，所以電路會針對*k = 2(His_Point2 Counter)*這個計算器加一，如圖3-26所示為直方圖硬體電路方塊圖，在每一個像素的角度輸出會經由直方圖的控制電路以及一組解多工器來選擇角度的範圍，在這8個直方圖計數器會針對每一次角度的不同來做計數，當16個像素全部計數完之後，直方圖的比較電路會比較出這8個直方圖計數器中所計數出來最大值為多少，硬體電路再依計數器的編號(*His_Pointk Counter k=0~7*)來決定使用那一組權重，在每一次4×4的滑動區塊的計算之前必需要將這8個計數清除為零。

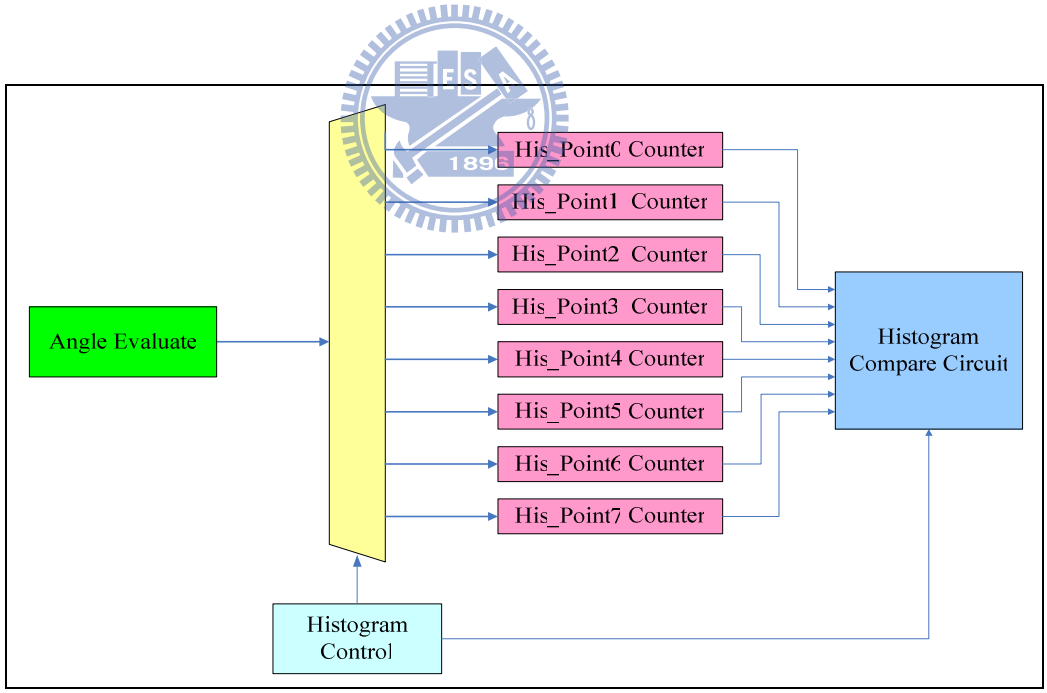


Fig. 3-26. 直方圖硬體電路方塊圖

3.4.4 影像內插法硬體實現

論文有會提到到雙線性內插法及邊緣適應性內插法，補插點要選擇使用那一種，取決於模糊決策系統運算及角度運算的結果。

(1) 雙線性內插法(Bilinear interpolation)

如果模糊決策系統的輸出結果是人眼較無法辨識的結果，可使用雙線性內插法，從式(3.13)為各個補插點的運算式。

$$\begin{aligned} P(1,0) &= (O(1,1) + O(2,1)) >> 1 \\ P(0,-1) &= (O(1,1) + O(1,2)) >> 1 \\ P(1,-1) &= (O(1,1) + O(2,1) + O(1,2) + O(2,2)) >> 2 \end{aligned} \tag{3.13}$$

此外使用的彩色影像格式是採用YUV(4:2:2)，所以U及V各佔一半無法使用邊緣適應性內插法，U像素及V像素也是一律採用雙線性內插法來處理補插點，從圖3-27 U像素及V像素的補插點方法中可得知，在U3的補插點中可以看出因為它的鄰近沒有U像素，所以必需要找較遠的U1及U5來做為它鄰近參考像素來做為雙線性內插法的補插點。

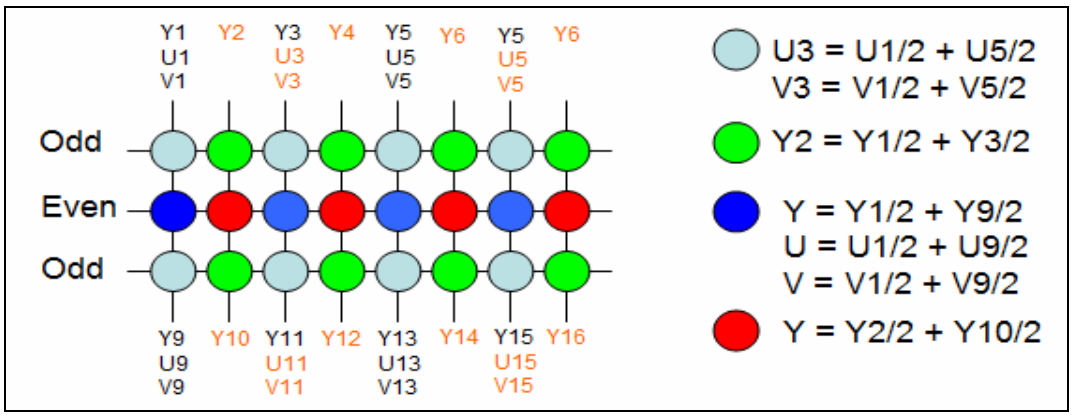


Fig. 3-27. U 像素及 V 像素的補插點

(2) 邊緣適應性內插法(Edge-Adaptive Interpolation)

如果模糊決策系統的輸出結果是人眼較為敏感的結果，使用邊緣適應性內插法，如式(3.14)所示，邊緣適應性內插法的權重參數$W_{\theta,m,n}(i,j)$，將根據角度及直方圖計算的結果來決定使用何種權重。權重矩陣如式(3.15)。

$$P(m,n)=\sum_{i=0}^{3}\sum_{j=0}^{3}O(i,j)W_{\theta,m,n}(i,j) \tag{3.14}$$

$$W_{\theta,m,n}(i,j)=\begin{bmatrix} W_{00} & W_{01} & W_{02} & W_{03} \\ W_{10} & W_{11} & W_{12} & W_{13} \\ W_{20} & W_{21} & W_{22} & W_{22} \\ W_{30} & W_{31} & W_{32} & W_{33} \end{bmatrix} \tag{3.15}$$

在軟體模擬中，所有的權重參數精確度的範圍為 0.000000001 ，但如果我們使用和軟體模擬相同準確權重的參數，將會增加計算時間和計算電路面積的問題，所以採用 8-bit 的定點近似的方式，將每一個權重以 2 進制將小數點往右保留 8-bit (2^8) ，並將每一個權重的最高位元保留再做 8-bit 延伸，目的是將高位元的正負號做保留，以確保像素和權重在乘加的過程中不會產生錯誤，故使用 16-bit 之乘法器如圖 2-28。從 8-bit 定點方式中可以發現，當權重精確度升級從 1/128 到 1/256 之間的差異，將會造成所計算出來的補插點誤差大約從 12 到 5。另外，當權重的精確度升級從 1/256 到 1/512，所計算出來的範圍差異的減少是有限的，所以決定使用精度重量為 1 / 256。

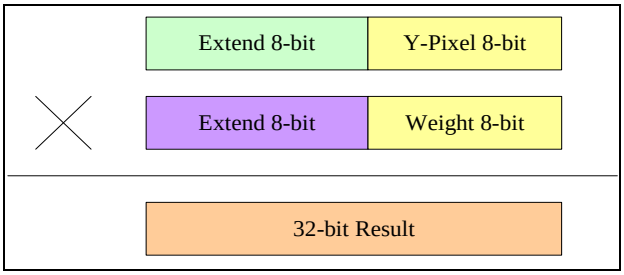
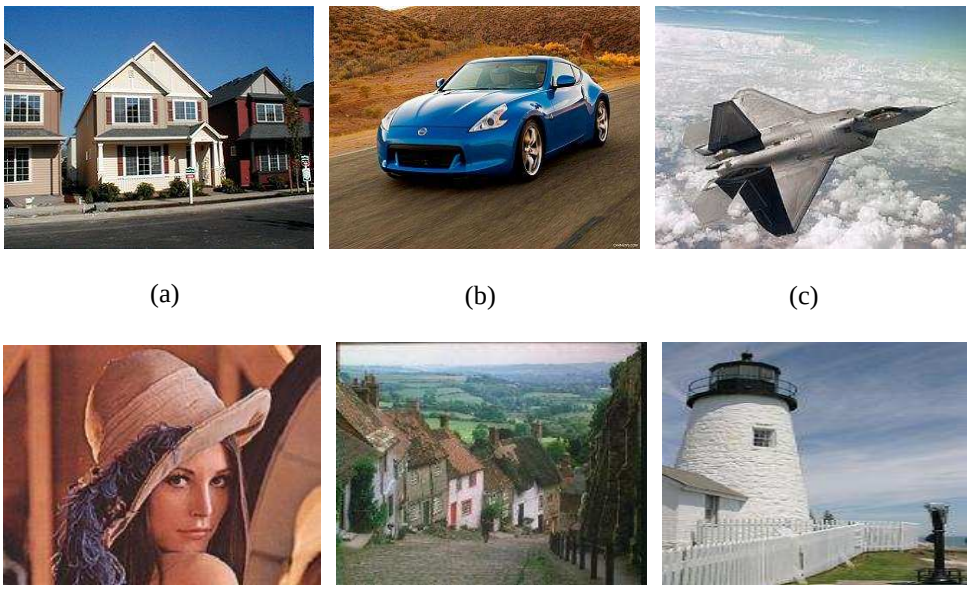


Fig. 3-28. Y 像素和權重之乘法

四. 實驗結果

4.1 HVS 硬體實驗結果

經由前一章結的討論後，可以針對硬體的實現提出兩個實驗結果，第一可以和灰階架構[21]所輸出的結果做一個比較，如圖 4-2(b)所示可以看出在灰階架構中的輸出，可以看出房屋的屋簷邊會有些許的鋸齒情形，再對照彩色架構所輸出的結果已將這鋸齒問題給解決，第二我們也選擇了 6 張彩色圖片輸入到硬體電路上來執行如圖 4-1 所示，和雙線性內插法來做比較在邊緣的處理都較為平滑，另外也選了其中的 3 張圖片如圖 4-6 ~ 4-8 和過去的 HVS 使用的軟體模擬做 PSNR 的比較如表 4-1，由於 PSNR 並不是百分之百的準確性，主要是 PSNR 的大部份是使用在影像壓縮後結果來和原圖做相比，但對於影像補插的計算，PSNR 的對照表主要是做為和其它內插法的實驗的結果來做為一個比較。



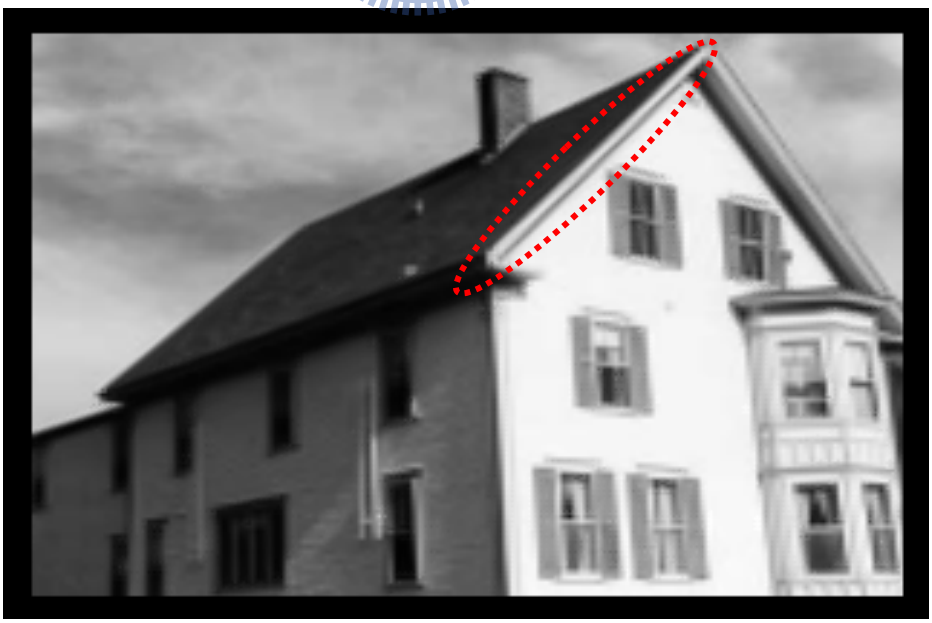
(a) (b) (c)

(d) (e) (f)

Fig. 4-1.（a）房屋[28]（b）跑車[29]（c）戰鬥機[30]（d）Lena[31]（e）街道[32]（f）燈塔[33]

(a)

(b)



I

Fig. 4-2. (a) 房屋原始圖片 (b) 灰階架構放大圖片 (c) 彩色架構放大圖片

(a)

(b) (c)

Fig. 4-3. (a) 房屋彩色原始圖片 (b) 使用雙線性內插法放大 (c) 使用 HVS 可適性放大

(a) (b)

(c)

Fig. 4-4. (a) 跑車彩色原始圖片 (b) 使用雙線性內插法放大 (c) 使用 HVS 可適性放大

(a)

(b)

(c)

Fig. 4-5. (a) 戰鬥機彩色原始圖片 (b) 使用雙線性內插法放大 (c) 使用 HVS 可適性放大

(a)

(b)

(c)

Fig. 4-6. (a) Lena彩色原始圖片 (b) 使用雙線性內插法放大 (c) 使用HVS可適性放大

(a)

(b) (c)

Fig. 4-7. (a) 街道彩色原始圖片 (b) 使用雙線性內插法放大 (c) 使用HVS可適性放大

(a) (b) (c)

Fig. 4-8. (a) 燈塔彩色原始圖片 (b) 使用雙線性內插法放大 (c) 使用HVS可適性放大

經由實驗的結果將圖4-6，4-7，4-8做PSNR計算如表4-1所示從過去一些其它內插法相比較來看PSNR都較為高，和HVS軟體模擬的結果比較誤差為RMSE (**Root Mean Square Error**) ≈ 0.11497，另外和過去灰階架構[21]相比較PSNR也較高。

Table 4-1.各種內插法 PSNR 之比較

		Bilinear [2]	Bicubic [3]	NN-Based [4]	Edge-Directed [5]	Aqua [11]	HVS [1]	Gray HW [21]	This work
Lena	**MSE**	62.85	64.38	50.47	77.36	27.17	19.99	28.76	**20.98**
	PSNR	30.15	30.04	31.10	29.24	33.78	35.12	33.54	**34.91**
Goldhill	**MSE**	94.46	102.24	80.28	53.09	55.51	51.58	61.72	**53.46**
	PSNR	28.37	28.02	29.09	30.88	30.68	31.01	30.22	**30.85**
Lighthouse	**MSE**	251.26	264.22	225.43	353.78	234.11	222.71	259.3	**225.59**
	PSNR	24.15	23.91	24.6	22.64	24.43	24.65	23.99	**24.60**

4.2 FPGA 硬體實現

彩色影像縮放硬體實現是根據DE2-70 FPGA硬體開發平台來完成演算法及整個系統的設計，透過這塊開發平台所提供的硬體資源如，表4-2所示核心的部份有一顆Altera Cyclone II EP2C70 FPGA晶片做為演算法硬體的實現，介面部份有SSRAM可以高速的存取影像資料能夠暫時存放彩色影像資料，將設計好的電路可透過USB介面從PC端將RTL code燒錄到DE2-70 FPGA中的Cyclone II EP2C70 FPGA晶片，在系統中有規劃兩個影像來源，其中以ADV7180視頻解碼器如圖4-9所示從TV端子如所輸入NTSC信號轉換成數位ITU-R.656數位影像格式，將彩色影像資料先暫時存放在高速SSRAM中，最後再送到FPGA晶片做演算法運算。另一個影像來源是預先將彩色影像資料存放於外部Flash記憶體中如圖4-10所示，經由Flash記憶體控制電路來讀取影像資料，並將影資料先暫時存放在高速SSRAM中，最後再傳送到FPGA晶片做演算法運算。

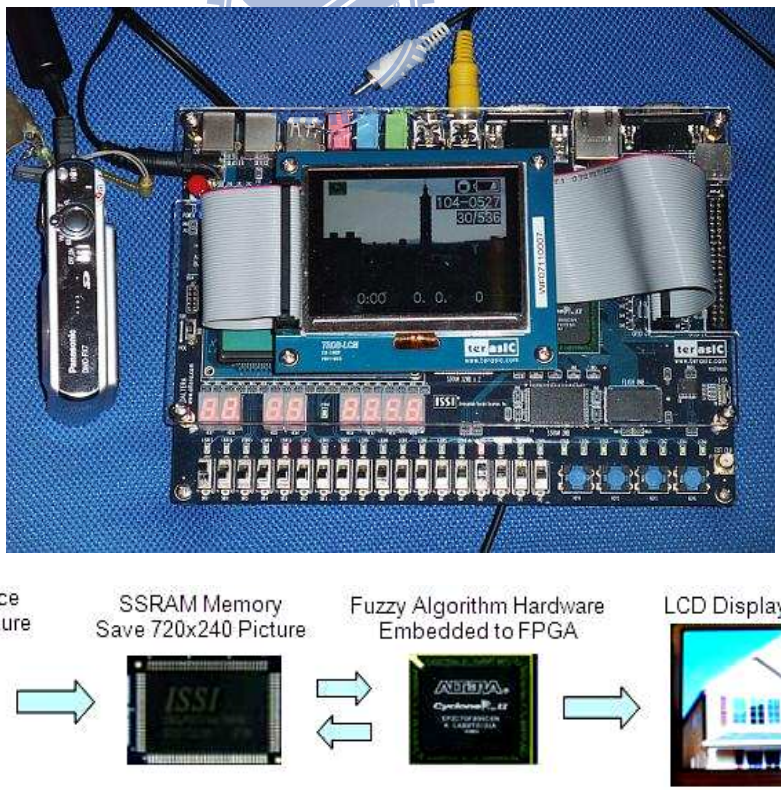


Fig. 4-9. 系統平台影像來源為視頻解碼晶片

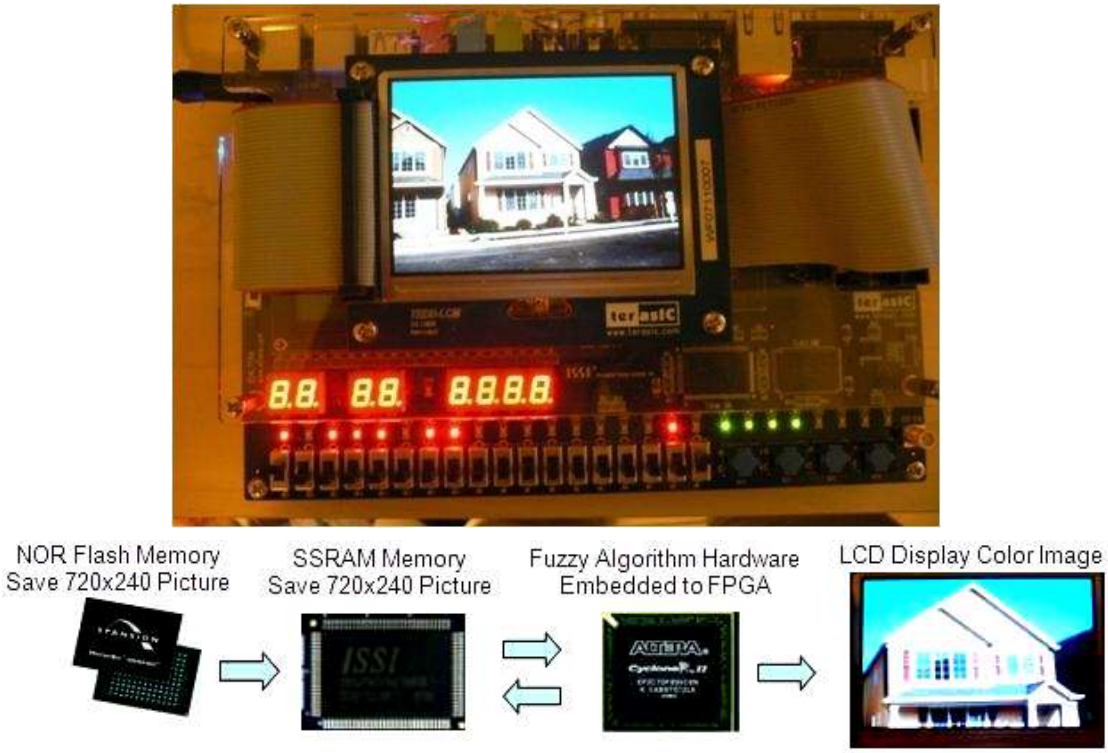


Fig. 4-10. 系統平台影像來源為 Flash 記憶體

Table 4-2. DE2-70 FPGA 板硬體資源

項目	硬體功能
FPGA (Cyclone II EP2C70)	演算法核心及周邊控制電路
SSRAM (IS61LPS51236A)	高速存取記憶體可暫時存放彩色影像資料
FLASH(S29GL064A)	存放彩色影像資料
視訊解碼晶片(ADV7180)	可經由視訊端子輸入 NTSC 訊號再將類比訊號轉換為數位訊號
TFT LCD 介面(GPIO)	顯示縮放圖片

4.2.1 電路面積效能及架構設計的比較

從表4-3中可以得知彩色架構的電路面積比灰階架構少了一半以上，因為採用Line Buffer資料處理，內部記憶體所使用的面積也少了很多;在效能的提升上系統頻率可以達到130 MHz如表4-4所示。

Table 4-3.架構及電路面積比較表

		Gray Architecture	This work
Algorithm	**VD**	No	Yes
	SD	Yes	Yes
	CD	Yes	Yes
Processes Memory		160 × 120 × 2 For Input 320 × 240 × 2 For Output Total : 187.5 k Byte	Y Pixel 720 × 6 U & V Pixel 360 × 4 × 2 Total : 7 k Byte
Processes Method		Frame Buffer Processes	Line Buffer Processes
Input Image source		TV decode	TV decode & NOR Flash
Input/Output Pixel		Gray Pixel (8-Bit 0~255)	Color Pixel (YUV 4:2:2)
Total Logic Element		20,017 / 68,416(29%)	9,064 / 68,416(13%)
Logic Register		2,343/68416(3%)	1,235/68416(1%)
Total Memory bits		775,760/1,152,000(67%)	86,400/1,152,000(8%)
PLL		2 sets	1 sets

Table 4-4.效能比較

	Old Performance	New Performance
System Frequency (max)	42MHz	130MHz
Data Load Algorithm	5,210,551ns	2,275,813ns

4.2.3 軟體及硬體效能比較

從表4-5所示在軟體模擬在PC上的Cycle數大於硬體的Cycle數，比較條件如下所示：

(1) 輸入影像解析度: 160×120 (Gray Level)

(2) 輸出影像解析度: 320×240

(3) 放大倍數: 2 倍 軟體語言: C++ Language PC 速度為 3GHz

(6) 硬體效能上角度及直方圖電路為平行處理

Table 4-5.軟體及硬體效能比較表

Executing Cycles	S/W performance	H/W performance	Ratio
VD	11,701,733	18,369	**637**
SD	8,760,938	15,997	**547**
CD	4,595,220	8,820	**521**
Angle	156,255,100	141,120	**1697**
Histogram	83,276,307		
Total	**264,589,298**	**184,306**	**1436**(1)

(1) SW_Cycle/HW_Cycle = **264,589,298/184,306 = 1436**

由表 4-5 可知，硬體的效能比軟體的執行效能快 1436 倍，因為硬體在處理資料的運算中是平行而且在模糊決策系統(VD,SD,CD)中的演算法及角度的計算不需花費太多的週期(Cycle)，而在軟體上雖然在 PC 上執行的速度非常快，但對於每一個演算法中的指令所需的週期(Cycle)數確實大於硬體，故在硬體上的執行比較能夠得到較好的效能。

4.2.4 影像放大處理效能

經由估算的結果，硬體在處理完一張彩色圖片時，所所需的時間如表4-6所示：輸入圖像的大小是720 × 240，放大倍數為2，輸出圖片解析度為1440 × 480，經由影像資料的載入到演算法的運算法以及到資料的輸出，所花費的時間為**14.98 ms**。

Table 4-6.彩色圖片放大處理效能

	Cycles	Times	Total cycles	Total time
Load Buffer (YUV Pixels)	2160	235	507600	3.9 ms
Load_reg_36_y Load_reg_18_u Load_reg_18_v	3	168025	504075	3.87 ms
VD	1	149152	149152	1.147 ms
SD	1	128810	128810	0.99 ms
CD	1	77374	77374	0.59 ms
Angle	16	77374	77374	0.59 ms
Data_out	3	168025	504075	3.87 ms
Total				**14.98 ms**

五. 結論及未來展望

在本論文，以人類視覺系統為基礎之彩色影像縮放硬體實現。在這項設計中採用模糊決策系統對於輸入彩色影像(YUV 4:2:2)中人眼較為敏感的 Y 像素特徵，來決定何種內插法將被使用。如果輸入影像對於人類眼睛較不敏感，為減少計算能力，故採用雙線性內插法。否則，使用邊緣適應性內插法，以減少模糊或鋸齒邊緣部分缺陷。在模糊決策系統演算法中的 VD，因為使用到背景亮度參數(BL)需使用到除法器，在硬體的設計上採用少許的加法器、減法器以及除數和商數的查表做為硬體的一個實現，而 V(BL)中的指數函數建立一個查表，並透過背景亮度參數(BL)可以快速的找到所要的 V(BL)函數，SD 及 CD 可以利用較為簡單的組合邏輯電路來完成，透過 CORDIC 電路的實現取代反正切函數，用於計算輸入影像中的方向。此種方法在經由 16-bit 的定點轉換可以取得較精確的值使得誤差值可以降到最小；在 U 像素及 V 像素的補插點中所採用是雙線性內插法，因為人類的眼睛在色差部份較不敏感，對於電路面積也較為節省。本論文的硬體設計對於過去的硬體架構缺點也做了一些改善詳細說明如下:

(1) 從過去架構中只能以灰階的方式來處理及顯示，彩色架構採用彩色影像可以滿足人類視覺。

(2) 在彩色架構中雖然使用大量彩色影像資料，並不會影響電路的面積而且使得電路面積比灰階架構少了一半以上。

(3) 在彩色架構中在影像放大之後，已消除灰階架構中所產生一些雜訊問題，並使邊緣變得更為平滑。

(4) 採用Line Buffer處理可以使得FPGA內部的記憶體更為減少。

儘管這一篇論文已將彩色圖片的縮放實現於硬體上，但對於未來仍需要改善的一些問題如下:

1. 不同的放大比例

   在影像縮放的硬體可能需要更大的比例，對於在輸出影像中可以看到更高解析的影像。

2. 周邊電路的模組化

   在設計硬體對於一些周邊控制電路的設計，如果能夠規劃出一組標準規格的介面每一個周邊控制電路都根據這一組規格的介面來設計可以減少設計的時間。

3. 演算法參數設定

   如果要將此論文中的硬體規劃成ASIC，可經由一些外部周邊介面(I2C)來設定演算法內部所需的變數設定如放大倍數。

參考文件

[1] C. T. Lin, K. W. Fan, H. C. Pu, S. M. Lu, and S. F. Liang, "An HVS-Directed Neural-Network-Based Image Resolution Enhancement Scheme for Image Resizing," *IEEE Transactions on Fuzzy Systems*, vol.15, no.4, pp.605-615, Aug. 2007.

[2] C. C. Wei and C. H. Chen, "Generalized Bilinear Interpolation of Motion Vectors for Quad-Tree Mesh," *International Conference on Intelligent Information Hiding and Multimedia Signal Processing* 15-17 pp.635-638 Aug. 2008.

[3] Z. Xiang, X. Zou, and Z. Liu, "An High Quality Image Scaling Engine for Large-scale LCD," *International Conference on Signal Processing*, vol. 1, pp.16-20 Nov. 2006

[4] F. M. Candocia and J. C. Principe, "Super-resolution of images based on local correlations," *IEEE Transactions on Neural Network*, vol. 10, no. 2, pp. 372–380, 1999.

[5] Z. Shi, S. Yao, and Y. Zhao "A novel video image scaling algorithm based on morphological edge interpolation," *International Conference on Neural Networks and Signal Processing* 7-11 pp. 388 – 391, June 2008.

[6] B. K. Gunturk, *Student Member, IEEE*, Y. Altunbasak, *Senior Member, IEEE*, and R. M. Mersereau, *Fellow, IEEE* "Color Plane Interpolation Using Alternating Projections" *IEEE Transactions on Image Process*, vol. 11, no. 9, September 2002.

[7] Amanatiadis, I. Andreadis, and A. Gasteratos, "A Log-Polar Interpolation Applied to Image Scaling" *IEEE International Workshop on Imaging Systems and Techniques*, 5-5 pp. 1-5, May, 2007

[8] I. Andreadis and A. Amanatiadis, "Digital Image Scaling," *Instrumentation and Measurement Technology Conference, 2005. Proceedings of the IEEE*, vol. 3, 16-19 pp. 2028–2032, May 2005.

[9] J. Allebach and P. W. Wong, "Edge-directed interpolation," *International Conference on Image Processing*, vol.3, 16-19 pp.707-710, Sept 1996.

[10] X. Li and M. T. Orchard, "New edge-directed interpolation," *IEEE Transactions on Image Processing*, vol.10, no.10, pp.1521-1527, Oct 2001.

[11] D. D. Muresan and T. W. Parks, "Adaptively quadratic (Aqua) image interpolation," *IEEE Transactions on Image Processing*, vol.13, no.5, pp. 690-698, May 2004.

[12] S. W. Lee and J. K. Paik, "Image interpolation using adaptive fast b-spline filtering," *International Conference on Acoust., Speech, Signal Process*, vol. 5, pp. 177-180 vol.5, 8-13 Mar 1993.

[13] M. M. L, S. D. G and S. R, "An image resizing algorithm for binary maps," *Digital Object Identifier* pp.126 – 132 2004

[14] B. S. Morse and D. Schwartzwald, "Isophote-based interpolation," *International*

Conference on Image Processing. 4-7 vol.3, pp.227-231 Oct 1998.

[15] K. Ratakonda and N. Ahuja, "POCS based adaptive image magnification," *International Conference on Image Processing*, 4-7 vol.3, pp.203-207 Oct 1998.

[16] C. T. Lin and C. S. G. Lee, "Neural Fuzzy Systems: A Neuro-Fuzzy Synergism to Intelligent Systems. Englewood Cliffs", NJ: Prentice-Hall, 1996.

[17] C. T. Lin, Y. C. Lee, and H. C. Pu, "Satellite sensor image classification using cascaded architecture of neural fuzzy network," *IEEE Transactions on Geoscience and Remote Sensing*, vol.38, no.2, pp.1033-1043, Mar 2000.

[18] C. H. Chou and Y. C. Li, "A perceptually tuned subband image coder based on the measure of just-noticeable-distortion profile," *IEEE Transactions on Circuits and Systems for Video Technology*, vol.5, no.6, pp.467-476, Dec 1995.

[19] A. A. A. Amanatiadis, I. I. Andreadis, and K. K. Konstantinidis, "Design and Implementation of a Fuzzy Area-Based Image-Scaling Technique," *IEEE Transactions on Instrumentation and Measurement*, vol. 57, no 8, pp. 1504- 1513 Aug. 2008.

[20] K. Jack, "YcbCr to RGB Considerations" *For information regarding Intersil Corporation and its products, website available at http://www.intersil.com* March 1997

[21] Y. C. Lo, "FPGA Implementation of Edge-Adaptive Interpolation Hardware for Real-Time Digital Image Resizing," *Master thesis, National Chiao Tung University in Electrical and Control Engineering*, July 2008.

[22] R. Andraka "A survey of CORDIC algorithm for FPGA based computers" Andraka Consulting Group, Inc 1998 ACM

[23] High-definition television - Wikipedia, the free encyclopedia

[24] Sinc function - Wikipedia, the free encyclopedia

[25] RGB color model - Wikipedia, the free encyclopedia

[26] http://imajr.com/New2.aspx?1=red-house-1551128.jpg

[27] http://imajr.com/Original.aspx?Id=normal-496f1793a2eb9-0000.bmp-1551133

[28] http://imajr.com/Original.aspx?Id=house2.bmp-1551131

[29] http://www.0755car.com/n/xw/xcsd/gw/08-12-31/89775.html

[30] LIFE Lockheed Martin F-22 fighter plane flyin... - Hosted by Google

[31] http://www.hlevkin.com/TestImages/lenna.bmp

[32] http://www.hlevkin.com/TestImages/goldhill.bmp

[33] http://www.blaisdell.org/Pemaquid%20Lighthouse..htm