以 FPGA 實現閥值參數化之 Sobel 影像邊緣偵測演算法

黃登淵
大葉大學電機工程學系-助理教授
e-mail:kevin@mail.dyu.edu.tw

王嘉宏
大葉大學電機工程學系-碩士班
e-mail:wanghouse2000@yahoo.com.tw

謝珍珍
大葉大學電機工程學系-碩士班
e-mail:R9503036@mail.dyu.edu.tw

摘要

本文以 FPGA 內部的區塊記憶體進行影像資料提取單元之設計，較傳統以移位暫存器所組成的 FIFO(First In First Out)記憶體的架構速度更快，且於邏輯閘數使用量平均約可降低 6.2 倍。而多數影像邊緣偵測的演算法，需利用一個閥值將物體邊緣提取出來，既有的演算法雖可估算出精確之閥值，但運算不僅複雜也十分耗時，本文提出一快速的閥值運算方法，其運算之結果較 Otsu 演算法更為優秀。本研究所設計之硬體架構配合了多層管線化、有限狀態機、及資源共享之設計方式，其硬體速度高達 183MHz，每秒可處理約 1706 張 256×256 大小之影像，已達到即時影像處理系統之需求。

關鍵詞：邊緣偵測、影像分割、現場可規劃邏輯閘陣列

Abstract

An efficient and quick algorithm to determine threshold value for edge detection, which has been implemented in FPGA, is proposed. Two different architectures, including block RAM-based and shift register-based, are used to design a moving window generator or elementary convolver to retrieve the pixel data in image. The hardware resource in FPGA gate counts when used block RAM-based architecture can be greatly reduced by 6.2 times with comparison to shift register-based architecture. Additionally, block RAM-based architecture also can achieve faster computation speed than that of shift register-based architecture. As well known, a threshold value is indispensable for most algorithms used in the edge detection of images. Although existing algorithms have already given user a good estimation of threshold value, they also suffer from the disadvantages with heavy computation-effort and great time-consuming. To solve the difficulties mentioned above, an efficient and quick algorithm to determine the threshold value is proposed. When further compared to Otsu's method, a more satisfactory result can be obtained by our method.

The system hardware architecture adapted in this research incorporates the techniques of pipeline, finite state machine, and resource sharing paradigm. The operation speed in our proposed system can reach up to 183MHz. The frame rate of processing is about 1706 frames per second, and it turns out that the realization of a real-time image processing system can be achieved without any difficulties.

Keywords: Edge detection, image segmentation, FPGA.

1. 前言

邊緣偵測(edge detection)在影像處理中的影像強化(image enhancement)、影像分割(image segmentation)及特徵擷取(feature extraction)技術都扮演著極為重要的角色[1]，由於影像中每一個物體的邊緣資訊都不相同，觀察其輪廓就能大致地了解物體的位置、大小、形狀與紋理，進一步就能進行物體之辨識。而絕大部分邊緣偵測的演算法，均是利用一個特定的遮罩(mask)與影像在空間域中進行迴旋積分(convolution integral)之運算，這也是影像濾波的關鍵技術，雖然運算簡單，但是當需處理的資料量較多時，便需要進行相當多次的運算，十分費時，因此將邊緣偵測的演算法硬體化的確有其必要性[2-8]。

其中 R.C.D.M. Tavares [2]以 FPGA 實現 Roberts 邊緣偵測演算法，將架構規劃為操作解碼單元、暫存器設定單元、位址單元、鄰域儲存單元、運算單元、以及控制單元，整體架構的完整性相當高，但運算單元的設計僅能針對 Roberts 邊緣偵測演算法(2×2 mask)進行處理，面對於絕大部分使用 3×3 mask 的演算法來說，所達成的實用性相當有限。

K. Benkrid and D. Crookes [3-4]對於影像資料提取單元之設計，以儲存少量的資料於晶片內部為原則，採用大量的暫存器作為影像資料的暫存區。文獻[7]則使用了兩塊外部記憶體晶片，一塊用來儲存自CMOS感測器的影像資料，而另一塊則作為FPGA處理後影像的儲存空間。 針對於此點，本研究使用FPGA內部雙埠區塊記憶體(Block RAM)進行設計，不僅可以儲存大量的資料，資料存取速度也較快。[6, 8]則提出了可變更影像大小的資料提取單元架構，使得硬體能處理不同大小的影像。

邊緣偵測演算法於[2-8]的研究中均無包含閥值參數的運算設計，主要由於閥值之運算較為耗時且需先進行計算才可開始進行邊緣偵測之運算，使用既有的演算法進行設計將難以取得硬體效能及資源使用上的優勢。因此本研究除了實現Sobel邊緣偵測演算法之外，亦提出一嶄新的閥值運算演算法，並實現於硬體電路中，運算速度遠超過即時影像處理系統之需求。

2. 系統架構

本研究所設計之系統架構如圖 1 所示，一開始先將影像暫存於 FPGA 內部的區塊記憶體，緊接著開始進行直方圖統計，以計算出合適之影像閥值。求出閥值後便由影像資料提取單元(moving window generator)開始蒐集 3×3 鄰域像素點的資料，以供運算單元進行計算，當運算單元計算完成後，將計算之結果儲存於另一個區塊記憶體中，即處理後的邊緣影像。而透過 VGA 顯示單元便將原影像與處理過的邊緣影像同時輸出到 LCD 螢幕，便能驗證系統執行的正確性。

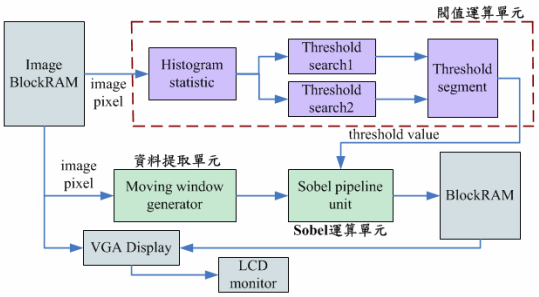


圖 1 閥值參數化之邊緣偵測系統架構

2.1 影像資料提取單元

進行影像邊緣偵測的演算法，第一步需要先擷取 3×3 鄰域中的像素值，接著利用一特定之遮罩由上而下、由左而右對整張影像進行迴旋積分的運算，如圖 2 所示。而晶片內部的記憶體容量有限，不見得能儲存整張影像的資料，因此必須採取儲存部分影像的策略，只將

目前所要進行運算的影像資料暫存在晶片中，以節省所需的硬體資源。

圖 3 為目前最為常見的資料提取單元之設計架構[9, 10]，其中 Line Buffer (FIFO A, B) 用來儲存近一整列的影像資料，多數的研究於 Line Buffer 的設計多採移位暫存器的設計方式，需耗費許多的硬體資源[3, 4, 6, 8-10]。以一張 8 位元 256×256 的灰階影像來說，使用 3×3 的大小的影像遮罩時，暫存器總共需要 (256×2+3)×8 = 4120 個，數量相當驚人，若遮罩或影像都需使用更大一些的時候，暫存器的數量將急遽上升，而消耗掉晶片上許多的資源。其中[6, 8]藉由設定 Line Buffer 中的資料量，可適用於多種不同尺寸大小之影像。

而本文針對於圖 3 中 Line Buffer 的部份，改以 FPGA 內部的區塊記憶體進行設計，以節省所需的硬體資源。其中[11-13]針對此單元之設計亦採用區塊記憶體進行設計，但與本文所設計之架構(圖 3)有所不同，也沒有針對兩種不同元件之設計方式進行比較。本文則以表 1 進行這兩種不同元件設計在資源及效能上之比較，明顯地以區塊記憶體進行設計不論在速度上、資源的使用上皆優於移位暫存器之設計方式。

而圖 4、圖 5 則是使用不同尺寸大小之影像下，兩種不同設計方式於資源與效能之分析圖。由此可知當影像尺寸愈大時，以區塊記憶體之設計於資源使用上僅會有些微上升，在速度上會開始有下滑之現象。而對於以移位暫存器之設計來說，隨著影像尺寸愈來愈大，所需的資源會急遽上升，平均要高出前例 6.2 倍之多，而速度上也較為緩慢，約降低 13%，見表 2。其中 speed ratio 與 gate count ratio 之計算均以移位暫存器之設計除以區塊記憶體之設計來進行計算。由表 2 可知本文採用區塊記憶體來設計資料提取單元，不但能大幅降低資源使用量，且於效能上亦能獲得提升。

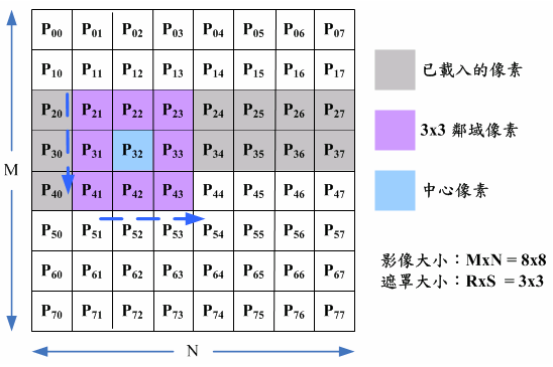


圖 2 影像迴旋積分運算示意圖

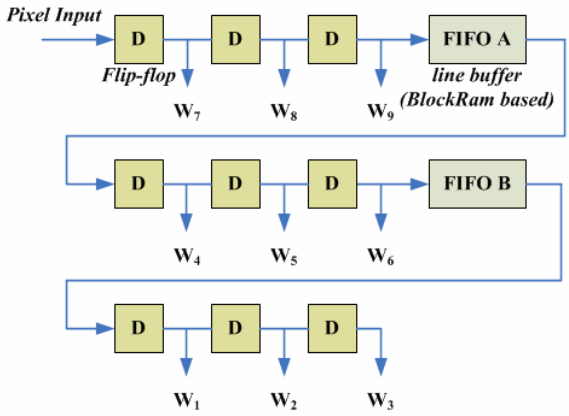


圖 3 資料提取單元之架構

表 1 資料提取單元晶片資源及效能之比較

(Device : XC4vlx25-ff668-10)		
Resource Utilization	Block RAM based (proposed)	Shift Register based [3, 4, 6, 8-10]
Flip-Flops	155	170
4 input LUTs	200	554
BlockRAMs	**2**	0
Speed (MHz)	**232.89**	**214.01**
Power (mW)	338	430
Gate count	**2640**	**20036**

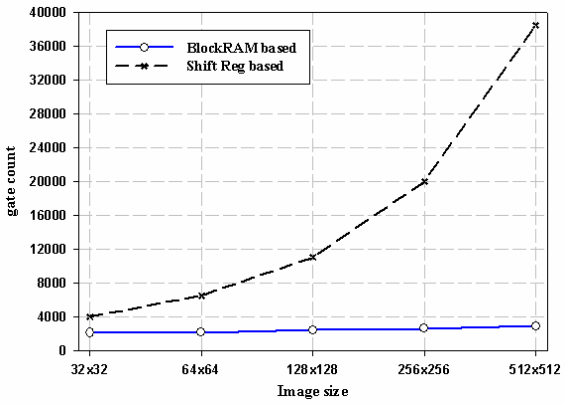


圖 4 不同影像尺寸所需邏輯閘數之分析圖

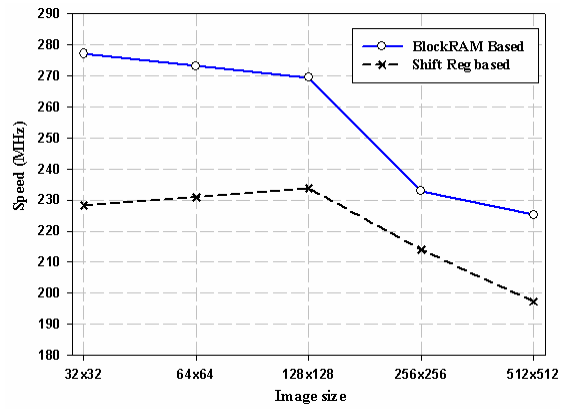


圖 5 不同影像尺寸所達效能之分析圖

表 2 資料提取單元數據分析

Image size	Speed ratio	Gate count ratio
32×32	0.82	1.90
64×64	0.85	3.01
128×128	0.87	5.50
256×256	0.92	7.59
512×512	0.87	13.04
Average	**0.87**	**6.21**

2.2 Sobel 運算單元

Sobel邊緣偵測運算子採用的是一階微分之影像遮罩，如圖 6 所示[1]，可用來偵測影像中水平及垂直之邊緣資訊。依據圖 6 其運算式可表示為(1)、(2)式，其中兩式之運算複雜度皆相同，也為避免偵測出一些不必要的邊緣資訊，因此本文僅實現水平邊緣偵測的部份，即G_x之運算。

$$G_x = (w_7 + 2w_8 + w_9) - (w_1 + 2w_2 + w_3) \quad (1)$$

$$G_y = (w_3 + 2w_6 + w_9) - (w_1 + 2w_4 + w_7) \quad (2)$$

圖 7 即為本文所設計之 Sobel 運算單元之架構，以四層管線進行設計，以提高資料處理的速度，此架構之運算速度可高達 238.2 MHz，已逼近於晶片之最高速限。

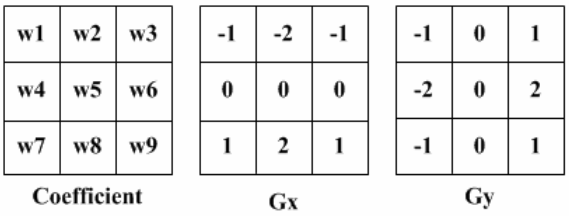


圖 6 Sobel 影像遮罩

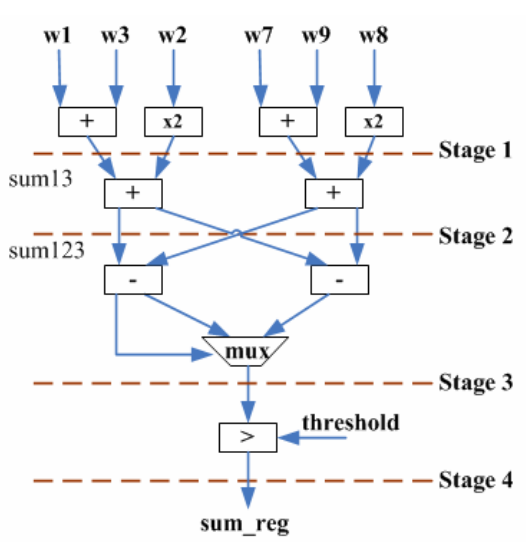


圖 7 Sobel 運算單元

3. 閥值運算演算法

影像邊緣偵測需利用一個閥值，以判定該點是否為有效之邊緣，而這個閥值對整體演算法的正確性有著關鍵性的影響。其中[14]以

直方圖統計配合斜率計算之方式，先找出斜率最大的兩個點，接著在此兩點中間，尋找一最低點作為影像之閥值，雖然方法簡單，但當有一灰階值斜率較大時，閥值就無法進行計算。而[15]則利用 Fuzzy 的歸屬函數，對差值直方圖(Difference Histogram)進行歸屬函數之區域配置，以進行閥值之計算。雖然[14, 15]之方式已是較為精簡的演算法，但對於硬體之設計上，[14]斜率之運算、及[15]差值直方圖之求取，於設計上並無法有效進行實現。而近年來多數的研究為了達到更高的精準度，便不再以直方圖來求取閥值，其中[16]便利用影像之熵值以求取最佳化之閥值(fast entropic thresholding)，但此方法運算較為費時，且需計算影像出現之機率，更加難以設計到硬體電路中。因此本文專為硬體設計一閥值運算之演算法，以使硬體電路能擁有自行運算參數之能力。

本文所提出之方法首先需進行直方圖之統計，將影像灰階值以 16 個計數器(C0~C15)進行統計，如圖 8。舉例來說：當灰階值為 0~15 時，C0 的計數值將增加 1，計數器索引值則為 0；若灰階值為 16~31 時，換成 C1 的計數值增加 1，此時計數器的索引值為 1，其餘依此類推。接著由計數器 C7 往 C0 出發，尋找最大計數值的兩個索引值(max1_1, max1_2)；另一邊則由計數器 C8 往 C15 的方向尋找出最大計數值的兩個索引值(max2_1, max2_2)。接著利用上述四個最大計數值與相對應之索引值，來求出參數 th1 及 th2 等兩個索引值，其中閥值即為(th1+th2)×8。

而為了使閥值的移動範圍擴大，以能有效應用在各種不同亮度的影像上，針對參數 th1 及 th2 之決定方式分為以下四種情況討論：

(1) 當max1_1不等於7且max2_1不等於8，則th1等於max1_1、th2等於max2_1。

(2) 當max1_1等於7且max2_1等於8，則th1等於max1_2、th2等於max2_2。

(3) 當max1_1不等於7且max2_1等於8，則th1等於max1_1。接著需進一步決定th2之值，當計數器max2_1之值不為0或max1_2大於max1_1時，th2等於max2_1，否則th2等於一極小值1。

(4) 當max1_1等於7且max2_1不等於8，則th2等於max2_1。接著需進一步決定th1之值，當計數器max1_1之值不為0或max2_2小於max2_1時，th1等於max1_1，否則th1等於一極大值14。換言之，以演算法來表示即為(3)~(6)式：

$$\textit{when}\ \max1\_1 \neq 7\ \textit{and}\ \max2\_1 \neq 8 \Rightarrow th1 = \max1\_1\ \textit{and}\ th2 = \max2\_1 \quad (3)$$

$$\text{when}\ \max1\_1 = 7\ \textit{and}\ \max2\_1 = 8 \Rightarrow th1 = \max1\_2\ \textit{and}\ th2 = \max2\_2 \quad (4)$$

$$\text{when}\ \max1\_1 \neq 7\ \textit{and}\ \max2\_1 = 8 \Rightarrow th1 = \max1\_1$$
$$\textit{if}\ C[\max2\_1] \neq 0\ \textit{or}\ \max1\_2 > \max1\_1 \Rightarrow th2 = \max2\_1\ \textit{else}\ th2 = 1 \quad (5)$$

$$\text{when}\ \max1\_1 = 7\ \textit{and}\ \max2\_1 \neq 8 \Rightarrow th2 = \max2\_1$$
$$\textit{if}\ C[\max1\_1] \neq 0\ \textit{or}\ \max2\_2 < \max2\_1 \Rightarrow th1 = \max1\_1\ \textit{else}\ th1 = 14 \quad (6)$$

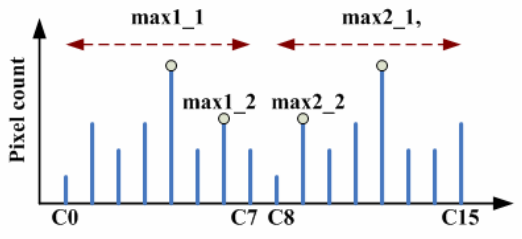
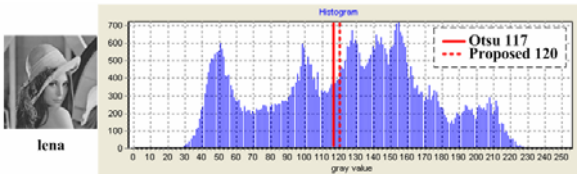


圖 8 影像直方圖統計示意圖

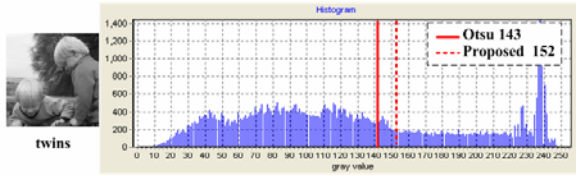
3.1 演算法比較與分析

依據前一節參數搜尋之方式，便能順利尋求到一個閥值。Sankur[17]對 40 多種的閥值計算方法進行了比較，結果指出 Otsu[18]的演算法所估測之閥值準確度相當高，本文也將與 Otsu 的演算法之結果進行比較。本文所使用的測試檔分為自然影像及人造影像兩種，其中人造影像取自[19]，該影像主要是作為直方圖模型分析之用。

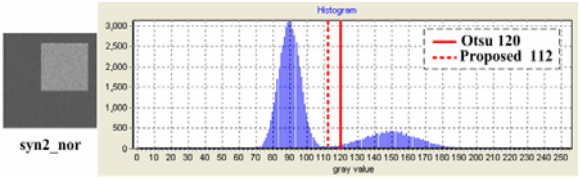
圖 9(a)~(f)為部份的測試影像，其中(a)(b)為自然影像，於(b)的直方圖可知本文所提出之方法較 Otsu 更逼近於山峰間的最低點。而圖(c)的部份，Otsu 法明顯已切割到物體之部分像素，而本文提出之演算法依然可找出最佳的閥值。(d)為多物體影像之測試，兩個演算法均能找到正確的閥值。而為了進一步確認演算法之效果，本文使用一張較暗(e)與較亮(f)之影像進行測試，結果證實的確本文所提出之演算法可運用於各種不同的影像上。較 Otsu 演算法計算快，且能找尋到更佳之影像閥值。



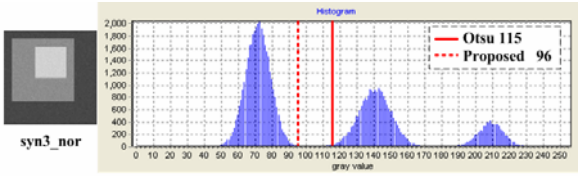
(a) 測試影像 Lena



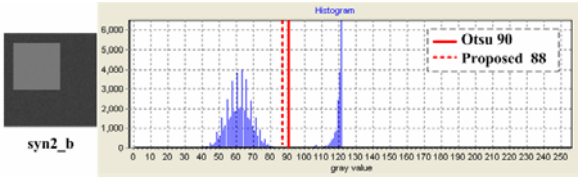
(b) 測試影像 twins



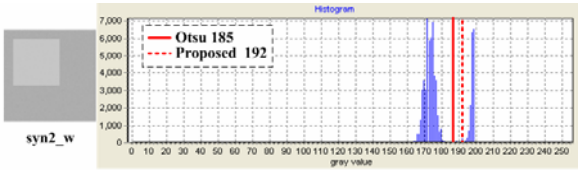
(c) 兩物體影像之測試



(d) 多物體影像之測試



(e) 低亮度影像之測試



(f) 高亮度影像之測試

圖 9 影像閥值之比較

3.2 閥值運算單元設計

依據本文所提出之演算法，其電路之設計架構如圖 10，將演算法切割為三部份進行設計。其中第一部份由 Histogram statistic 模組進行直方圖統計，當統計完成之後，便輸出 16 組計數器之數值到下一個模組。

Threshold search 模組在接收此 16 個影像統計之數值後，便會開始找尋前兩大的數值，接著便傳送到 Threshold segment 模組進行閥值之運算。

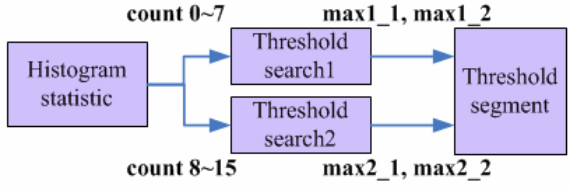


圖 10 閥值運算單元架構

4. 實驗結果

本研究將閥值運算單元、資料提取單元、Sobel 運算單元、以及其他資料驗證單元進行整合，並以 Xilinx Vertex4 系列晶片 XC4vlx25 進行電路合成，總邏輯閘數需 20199 個，其系統操作頻率可高達 183.69 MHz，相當於每秒可處理 1706 張 256×256 大小之影像。其中圖 11 為系統測試時，以 VGA 顯示模組驅動 LCD 液晶螢幕所顯示之影像，可達到大量資料的即時驗證，經過多次實驗資料的比對後，證實與軟體程式運算的資料完全符合，使本系統達到硬體實現演算法零誤差之成果。而表 3 為實現邊緣偵測演算法各個研究之效能比較(並非實現同一種演算法)，本文所設計之架構速度都在其他設計的 2 倍以上，效能極高已達到即時影像處理系統之需求。

圖 11 系統運作情形

5. 結論

本研究針對影像資料提取單元所使用之元件進行了效能與硬體資源的比較，採用區塊記憶體的設計方式，將能對此單元降低 6.2 倍的邏輯閘數，且能提升 13%的速度。對於閥值運算演算法本文亦提出一新的方法，並實現於 FPGA，此閥值運算電路時脈高達 188MHz，能快速而準確地進行閥值之計算，因此本系統可應用在光線改變量大之環境下。而進一步可再針對所檢測出的邊緣資訊進行物體的辨識與追蹤，藉由 FPGA 內部平行化之架構，將可再繼續擴充其他的演算法，達到即時影像處理系統單一晶片化之目的。

表 3 系統效能比較

Architecture	Image size	Operation speed	Frames in sec
Proposed	256×256	**183 MHz**	**1706**
文獻[4]	256×256	**75 MHz**	104
文獻[5]	256×256	40 MHz	610
文獻[6]	256×256	47.9 MHz	**730**
文獻[7]	640×480	13.2 MHz	43
文獻[8]	512×512	73.6 MHz	280

參考文獻

[1] R. C. Gonzalez, R. E. Woods, Digital image processing, 2nd ed., Prentice Hall 2002.

[2] R.C.D.M. Tavares, C.J.N. Jr. Coelho, A.D.A. Araujo, A.O. Fernandez, “Implementation of an Edge Detection Algorithm in a Reconfigurable Computing System”, Proceedings of the 11th Brazilian Symposium on Integrated circuit design, Rio de Janeiro, Brazil, pp.38-41, 1998.

[3] A. Bouridane, D. Crookes, P. Donachy, K. Alotaibi, K. Benkrid, “A high level FPGA-based abstract machine for image processing”, Journal of Systems Architecture, Vol. 45, No.10, pp.809-824, 1999.

[4] K. Benkrid , D. Crookes, A. Benkrid, “Towards a general framework for FPGA based image processing using hardware skeletons”, Journal of Parallel Computing, Vol. 28, No. 7-8, pp.1141-1154, 2002.

[5] Li Xue, Zhao Rongchun, Wang Qing, “FPGA based Sobel algorithm as vehicle edge detector in VCAS”, IEEE International Conference Neural Networks and Signal Processing Nanjing, China, Vol.2, pp.1139-1142, 2003.

[6] 華春和，“影像邊緣偵測之參數化FPGA架構設計”，國立台灣師範大學工業教育學系碩士論文， 2003。

[7] R.L. Rosas, A. de Luca, F.B. Santillan, “SIMD architecture for image segmentation using Sobel operators implemented in FPGA technology”, 2nd International Conference on Electrical and Electronics Engineering (ICEEE) and XI Conference on Electrical Engineering, Mexico City, Mexico, pp.77-80, 2005.

[8] P.-Y. Hsiao, C.-H. Chen, H. Wen and S.-J. Chen, “Real-time realisation of noise-immune gradient-based edge detector”, IEE Proceedings-Computers and Digital Techniques, Vol.153, No.4, pp.261-269, 2006.

[9] A. Benedetti, A. Prati, N. Scarabottolo, “Image convolution on FPGAs: the implementation of a multi-FPGA FIFO structure”, Proceedings of the 24th Euromicro Conference, Vasteras, Sweden, Vol.1, pp.123-130, 1998.

[10] B. Bosi, G. Bois, Y. Savaria, “Reconfigurable pipelined 2-D convolvers for fast digital signal processing”, IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Vol.7, No. 3, pp.299-308, 1999.

[11] V. Muthukumar, D. V. Rao, “Image processing algorithms on reconfigurable architecture using HandelC”, Proceedings of the Euromicro Symposium on Digital System Design, Rennes, France, pp.218-226, 2004.

[12] M. O'Nils, B. Thörnberg, H. Norell, “A Comparison between Local and Global Memory Allocation for FPGA Implementation of Real-Time Video Processing Systems”, Proceedings of the IEEE International Conference on Signals and Electronic Systems, 2004.

[13] M. O'Nils, B. Thörnberg, H. Norell, “Address generation for FPGA RAMS for efficient implementation of real-time video processing systems”, Proceedings of the International Conference on Field Programmable Logic and Applications, Tampere, Finland, pp.136-141, 2005.

[14] C.K. Lee, F.W. Choy, H.C. Lam, “Real-time thresholding using histogram concavity”, Proceedings of the IEEE International Symposium on Industrial Electronics, Xian, China, Vol.1, pp.500-503, 1992.

[15] S.E. El-Khamy, M. Lotfy, N. El-Yamany, “A modified Fuzzy Sobel edge detector”, Seventeenth National Radio Science Conference, Minufiya, Egypt, pp.C32 1-9, 2000.

[16] Jianping Fan, Walid G. Aref, Mohand-Said Hacid, Ahmed K. Elmagarmid, “An improved automatic isotropic color edge detection technique”, Journal of Pattern Recognition Letters, Vol.22, No.13, pp.1419-1429, 2001.

[17] B. Sankur, M. Sezgin, “Survey over image thresholding techniques and quantitative performance evaluation”, Journal of Electronic Imaging, Vol.13, No1, pp.146-165, 2004.

[18] N. Otsu, “A threshold selection method from gray-level histogram,” IEEE Transactions on System Man Cybernetics, Vol. SMC-9, No. 1, pp. 62-66, 1979.

[19] Laura Frost, “Information Theoretic Image Thresholding”, Computer Science Department of Monash University, Thesis, 2002. (http://www.csse.monash.edu.au/hons/projects/2002/Laura.Frost/index.html)