Research Journal of Applied Sciences, Engineering and Technology 4(24): 5474-5479, 2012
ISSN: 2040-7467

Submitted: March 18, 2012 Accepted: April 14, 2012 Published: December 15, 2012

Vlsi Implementation of Edge Detection for Images

T. Mahalakshmi and R. Muthaiah
School of Computing, SASTRA University, Thanjavur-613402, Tamil Nadu, India

Abstract: Edge is the boundary between the image and its background. Edge detection in general is defined as the local maxima obtained from high pass filters, but an optimized edge detector should mark the edges with respect to luminance or brightness changes. It is easy to obtain them in software implementation but for hardware implementation there is an issue with percentage of accuracy and processing time. This study discusses various edge detection algorithms and proposes an optimized edge detector which provides the solution for mentioned above issue. Since FPGA provides practical solutions for most of the image processing problems, the proposed architecture has been developed using Matlab System generator. Experimental results show the accuracy of edge detected using proposed architecture.

Keywords: Edge detection, FPGA, image processing, Matlab (system generator), xilinx (IP core)

INTRODUCTION

Image processing is a type of signal processing in which the image or information regarding image is fed as an input signal and various operations are performed on it. The various operations performed on it can be used on a host of applications such as Image filtering, medical imaging, Wireless communication, image compression, computer vision etc. Some of the most common operations on an image that come under the canopy of image processing are image scaling, converting between various colour format, image rotation, removing noise, adding noise, filtering, blurring, edge detection and contour detection (Gonzalaz and Woods, 2002). Some combination of these algorithms is used in almost all image processing applications.

Edge detection is a primary tool in image processing that is used in cases of feature detection and feature extraction. It identifies points in the image at which there is a sharp or gradual change in brightness. These points are alone marked in the output image with a specified colour which is usually black and rest of the image is white. So the output image contains only the edges of the various components in the input image. There are a few algorithms used for edge detection out of which Prewitt filter and Sobel filter are commonly used ones. This study presents architecture for edge detection using System Generator, which is an extension of Simulink and consists of models called Xilinx blocks. It script file to produce synthesis in FPGA. The tool retains the hierarchy of Simulink when it is converted into VHDL/Verilog (System Generator Introduction, 2012).

As previously mentioned, XSG is configured to program the FPGA. The synthesis and implementation of the program are done subsequently. In real time implementation of edge detection on FPGA by Sudeep and JharnaMajumdar (2011) is done with Spartan3A DSP board but in present study Spartan3E starter kit is used to implement the design with least resource usage. The architecture implemented in this study is versatile for any edge detection operator unlike the study by Sami *et al*. (2010) which only deals with Sobel operators. Compared to vehicle image edge detection algorithm hardware implementation on FPGA by Zhang and Wang (2010) the resource usage by the proposed architecture is reduced by 50%.

METHODOLOGY

Edge detection algorithms: Two basic methods are used for detecting edges. They are search based and zero-crossing based. In the first method a first order derivative expression such as a gradient derivative is obtained from the image and then the local maxima or minima are searched for. The points of local maxima or minima are treated as edges and are mapped on to the final image. In the second method, the second order derivative expression of the image is obtained by zero-crossings of the Laplacian or the zero-crossings of a non-linear differential expression. These points of zero-crossings are treated as edges and are mapped on the final output image. The method of obtaining the first order derivative expression is by using a gradient map. Three common methods are Prewitt filter and Sobel operator: Allin *et al*. (2011).

Corresponding Author: T. Mahalakshmi, School of Computing, SASTRA University, Thanjavur-613402, Tamil Nadu, India

Prewitt edge detector:
Sobel edge detector:

$$G_x = \begin{bmatrix} -1 & 2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} G_y = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$$

$$G_x = \begin{bmatrix} -1 & 2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} G_x = \begin{bmatrix} -1 & 2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

Gx and Gy are the gradient along horizontal and vertical axis respectively which is moved over the input image in both horizontal and vertical direction. Total gradient obtained is, $G = \sqrt{G_x^2 + G_y^2}$. By using the threshold along with model, noises can be suppressed. All the pixels above a particular user-set threshold are considered as edges and all the other points are discarded. A high threshold value results in finding less number of edges. Low threshold value results in finding a large number of edges. For certain kinds of images a large threshold value is needed and for certain applications the vice versa is needed. The trick lies in choosing the correct threshold value and the correct algorithm for each application.

Hardware architecture: The general hardware architecture for edge detection algorithm is shown Fig. 1. It includes four major blocks, First block Delay line is provided with the input image pixel values. It delays the pixel values and forwards it to next block which performs masking. Masking is done in

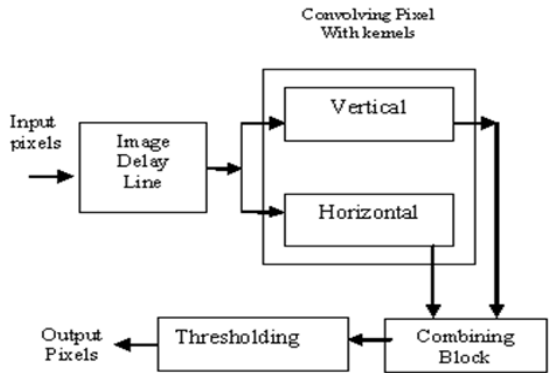

Fig. 1: Hardware architecture of edge detection

both horizontal and vertical direction individually by convoluting the pixel values with their corresponding kernels (changes depending on the algorithm chosen). Next block combines the output of previous blocks (both horizontal and vertical). Final block is for threshold, which reduces the noise.

Architecture using simulink xilinx block sets: Among number of edge detection algorithms, this study implements only Sobel, prewitt edge detection algorithms. Comparative study of these algorithms depending on their performance has also been done.

Colour image is converted into gray image in mat lab and the value is loaded in workspace. The workspace variable is fed to model through "From workspace" block. Image pixel values are fed as the input for delay line

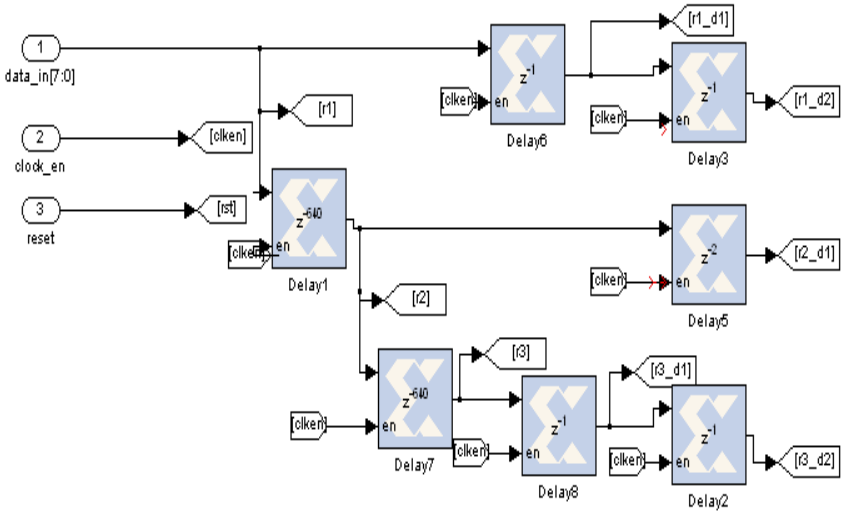

Fig. 2: Image delay

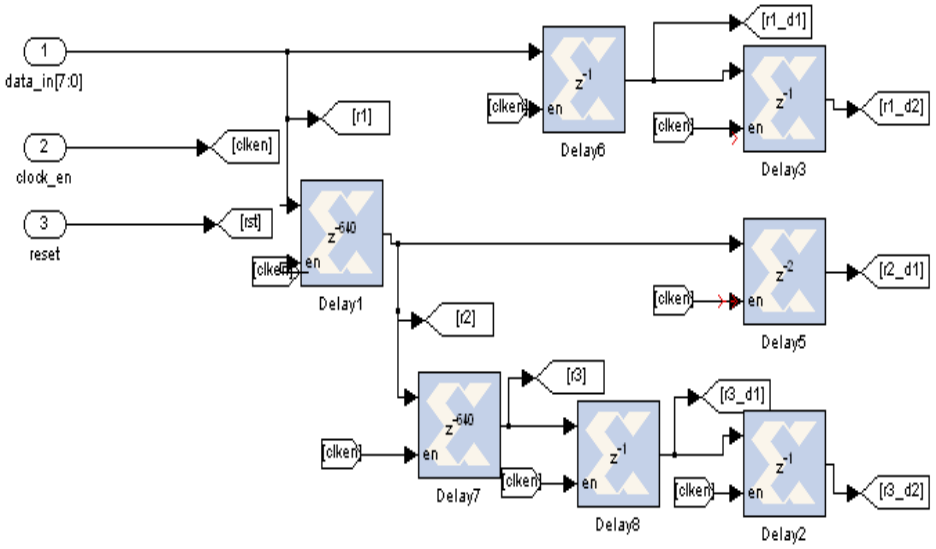

Fig. 3: Horizontal masking

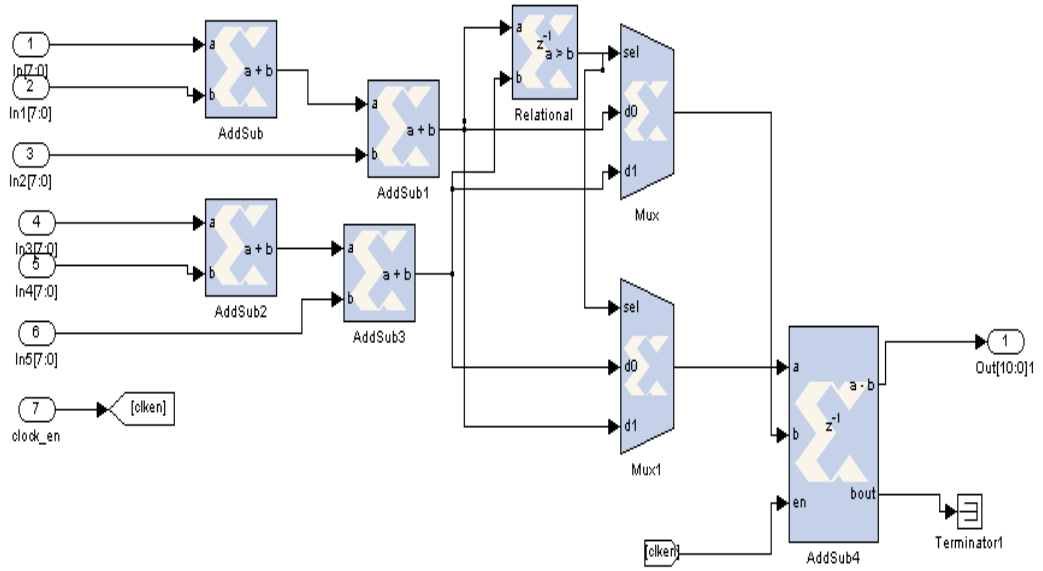

Fig. 4: Vertical masking

block shown in Fig. 2; which acts as buffer, that is it delay's the input pixel values and forwards it to the horizontal and vertical convolution blocks.

Figure 3 and 4 performs horizontal and vertical masking with their corresponding kernels. It performs convolution of pixels values from delay line block with kernels. Prewitt Edge detection kernel has only 1,-1 and 0 values, so there are no multipliers required as used in: Sudeep and JharnaMajumdar (2011). Vertical kernel is simply 90° rotation of horizontal kernel. So there is no major change in masking architecture. In Sobel Edge detection kernel has 2, -2 and 0 values. Multiplication by two can be simply achieved by right shifting the value. Slight modification in this block gives masking unit for Sobel detector.

Gradient magnitude is calculated after both horizontal and vertical masking using Fig. 5. Then the values are combined using combinational block which is designed using adder. Threshold reduces the noise, so the final output is taken after this block shown in Fig. 6.

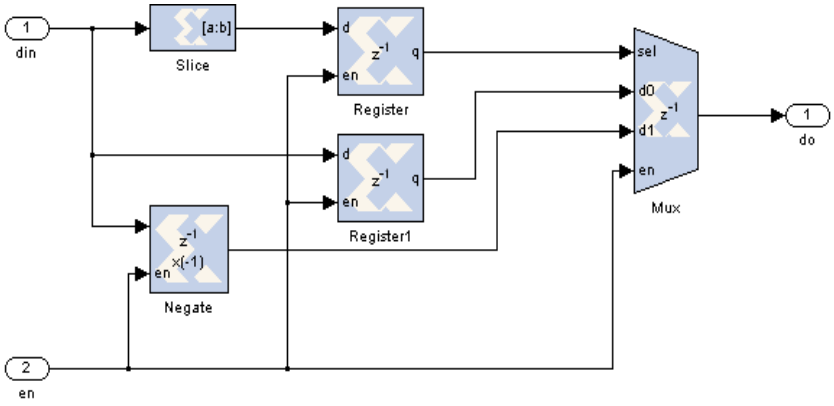

Fig. 5: Magnitude calculation

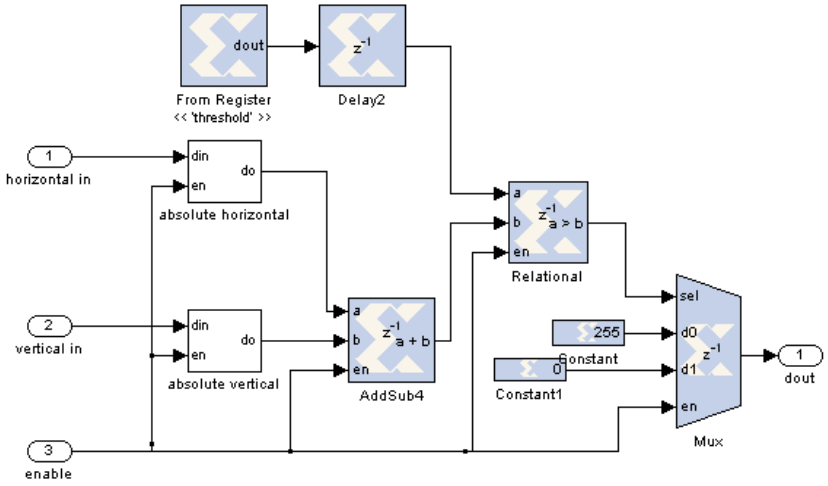

Fig. 6: Combining and threshold making

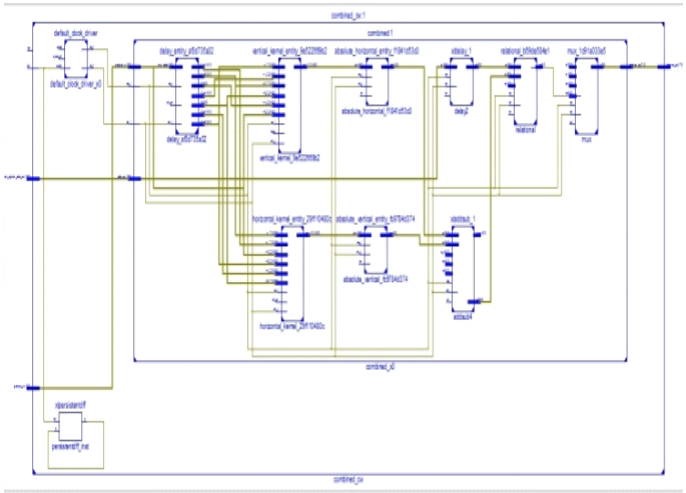

Fig. 7: Synthesized RTL view

Implementing designed model to FPGA: The model is designed with Xilinx block sets of Matlab tool. The blocks are converted into VHDL synthesizable file to implement the design in Spartan 3A DSP FPGA (Spartan 3A DSP FPGA Family Datasheet, 2012). Net- list files are generated and synthesized using Xilinx. The synthesized RTL view of the proposed architecture is show in Fig. 7.

RESULTS

The results shown below (Fig. 8, 9, 10 and 11) are the edges acquired by changing threshold value. The accuracy of edges detected changes with respect to both threshold and algorithm (Sobel and Prewitt) used. Table 1 shows the resource utilization of implemented design.

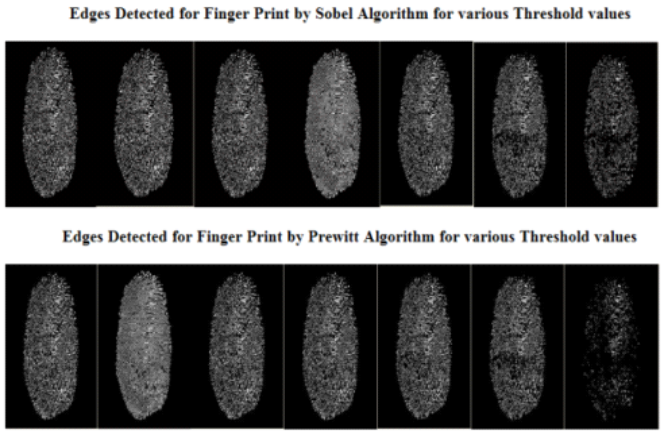

Fig. 8: Edges detected for finger print

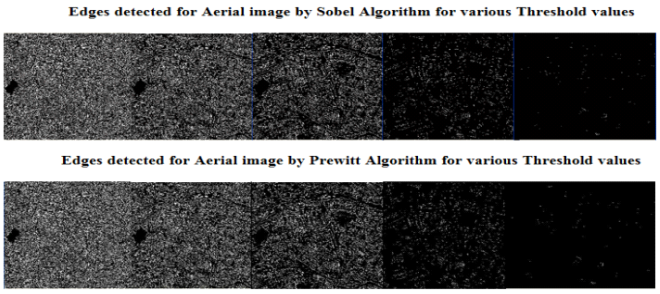

Fig. 9: Edges detected for aerial image

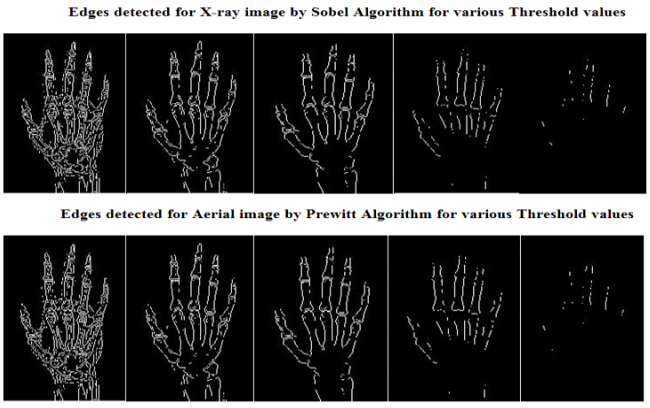

Fig. 10: Edges detected for X-ray image

Fig. 11: Edges detected for text

Table 1: Logic utilization

Logic Utilization	Utilization (%)
Slices	3
Flip – Flops	2
4 Input LUTs	2
Bonded IOBs	8
CLKs	4

Table 1 shows the resource utilization of FPGA.

CONCLUSION

In this study two different methods of edge detection algorithms are implemented in the FPGA. The main advantage is low resource requirement which makes this architecture more reliable. System generator blocks are optimized for both speed and area to make it more efficient and ubiquitous to embed the system with real time appliances.

REFERENCES

Allin, S.C., M. Vignesh and A. Kandaswamy, 2011. An efficient fpga implementation of mri image filtering and tumour characterization using xilinx system generator. Int. J. VLSI Desig. Commun. Syst. (VLSICS), 1(2).

Gonzalaz, R. and R. Woods, 2002. Digital Image Processing. Prentice-Hall, Upper Saddle River, NJ, Chapter-10.

Sami, H., Y. Alex and B. Said, 2010. Performance efficient FPGA implementation of parallel 2-D MRI image filtering algorithms using xilinx system generator. 7th International Symposium on Communication Systems Networks and Digital Signal Processing (CSNDSP), 21-23 July, Newcastle upon Tyne, UK, pp: 765-769.

Sudeep, K.C. and M. Jharna 2011. A novel architecture for real time implementation edge detectors on FPGA. Int. J. Comput. Sci. Issues, 8(1): 193-202.

System Generator Introduction, 2012. http://www.xilinx.com/tools/dsp.html

Spartan 3A DSP FPGA Family Datasheet, 2012. WWW.Xilinx.com/

Zhang, S. and X. Wang 2010. Vehicle Image Edge detection algorithm hardware implementation on FPGA. International Conference on Computer Application and System Modeling (ICCASM), 22-24 Oct., Beijing, China, 14: 184-188.