HIGH PERFORMANCE AND AREA EFFICIENT FPGA IMPLEMENTATIOIN OF DISTRIBUTED CANNY EDGE ALGORITHM

[1]Vinayaka pallakki V, [2] Pushpalatha K N

[1]Student, MTech, Digital Electronics and Communication,[2]Associate Professor, Digital electronics and Communication
[1]Digital Electronics and communication Engineering, [2] Department of DEC,
[1&2] Dayananda Sagar College of Engineering, Bangalore, Karnataka, India

Abstract : Edge detection is one of the basic operations carried out in image processing and objects identification. Edge detection is an important role in image processing which needs optimized, accurate and less latency architecture. Edge detection is a key role in this project with Adaptive threshold technique. The efficient canny edge detector plays an important role in real time application. In this Project the Gaussian filter is applied as preprocessing block to remove the high frequency edges. The Canny coefficients are approximated to suite the hardware requirements with less LUT's and further the adaptive threshold technique is applied to obtain the finer details of edges. In this proposed method of edge computation using adaptive threshold method to reduce memory elements significantly, because it is takes less area takes less decision making time so it reduces delay and increased efficiency without effecting detection performance. The proposed edge detection architecture is implemented using Xilinx system generator tool on Spartan6 ATLYS (XC6SLX45) board. It is observed that the proposed method is better compared to existing architectures.

Index Terms: FPGA Implementation, Image Processing, Canny Edge Detection and System Generator.

I. INTRODUCTION

Edge detection is one of the significant sections of the image processing algorithms which have many applications like image morphing, pattern recognition, image segmentation and image extraction etc., [1]. As the edge is one of the major information contributors to any image, hence the edge detection is a very important step in many of the image processing algorithms. Edge detection is defined as it is a set of mathematical methods from those methods detecting the edges of the image, particularly at edges the pixel value changes more sharply and it has more discontinuities. It represents the contour of the image which could be helpful to recognize the image as an object with its detected edges. In the ideal case, by applying the edge detector to an image gives the different edges that are connected to form the outline of the object [2]. Edge detection detects all edges present in the image with various orientations.

Mohamed Nasir et al., [3] proposed the real time hardware image processing framework improvement. This depends on FPGA. The algorithm utilized on the image processing as a part of this work is edge location. Here Verilog HDL was utilized as the programming dialect for the real time picture edge identification. The resulted edge identification of a image is improved and designed on Altera Cyclone-II FPGA for the image processing. The extra step can be actualized on improvising the edge location should be possible by including high performed end FPGA. Guo et al., [4] proposes the image processing development and the improvement of Sobel edge discovery algorithms. The merit of this algorithm is that it will show the consequence of handling one pixel within one clock periods. The executed design can distinguish or recognize the edge of the gray picture effectively within minimum processing time. The Sobel operator utilized in the algorithm includes the orientation of kernels. Ferdous Hossain et al., [5] projected adaptative cagey edge recognition rule for edge recognition through active thresholds. The experimental outcome represent the sting recognized by the adaptative cagey operator has additional continuity, and better signal to noise proportion relating to to different leading edge techniques. Mina Asaduzzaman et al., [6] given adptive clever edge recognition formula is projected. adaptie clever formula is employed to expand the truth of output objects . In customary clever ought to set 2 threshold values physically, therefore there square measure some imperfections to varied photos but this paper advances a adaptational threshold values taking into consideration mean and median values.

II. PROPOSED ARCHITECTURE

The diagram of projected Canny Edge detection is shown in Fig.1. The input image is converted into suitable format for processing by preprocessing block which is then filtered by Gaussian filters. The Adaptive Threshold Calculation block calculates a specific threshold value for each image. A delay block is inserted before Gradient Calculation block for proper timing synchronization purpose. Next Thresholding block uses the output of Gradient and Adaptive Threshold Calculation block to generate proper edge.

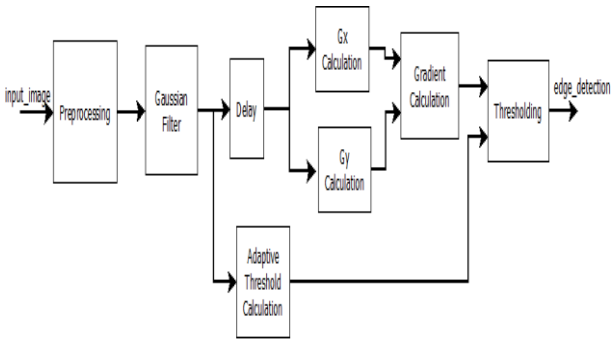

Fig. 1: Proposed Diagram of Canny Edge

2.1. Preprocessing

In this stage the input image is resized to a suitable size (256x256) and color image is converted to gray level for hardware optimization purpose. Also in this block the image matrix is converted into serial format.

2.2. Gaussian Filter

The Gaussian filter is a two dimensional convolution operator used to remove noise from the image. Here the kernel matrix used is represents the Gaussian shape that is it is having bell shaped representation. The kernel [7] matrix used is

$$\text{Gaussian Filter} = \frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix} * \begin{bmatrix}d_0 & d_1 & d_2\\ d_3 & d_4 & d_5\\ d_6 & d_7 & d_8\end{bmatrix}$$

Where, d_0 to d_8 are the 3x3 image sub-matrix pixel values.
This kernel is symmetric and as well as its total weight is 1 because to get the image gain as equal to original image. It acts as a low pass filter it will remove high frequency components. Its kernel values are non-negative everywhere. Convolution with a Gaussian matrix results a non-negative result, so these type of function maps non negative values to resultant non negative values. So it leads to another valid image as output.
The moving window architecture to implement 3x3 image sub-matrix is shown in Fig.3.

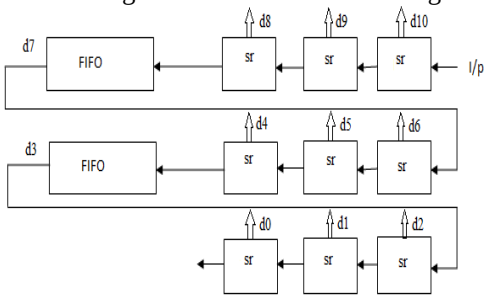

Fig 2: moving window architecture (3x3 pixel generation)

2.3. Adaptive Threshold

Image thresholding is the frequent task in many computer applications like computer and graphics. The objective of the image thresholding is to indicate pixels as 'dark' and 'bright'. This technique is used or it is suitable for video streaming. Based on the intensity value or pixel value the image is segmented as 'dark' and 'bright'. In most basic thresholding technique threshold value is fixed and it is compared with the image pixel values if the pixel value is greater than threshold value then that pixel value is considered for the edge detection it is called fixed thresholding. However this method may fail because of the variation in the intensity value in the image. Here the adaptive thresholding block is utilized to calculate threshold value from the image to detect edges in the image. If the pixel value is larger than the threshold value then that pixel value is considered for the edge detection otherwise it is treated as zero. To calculate adaptive thresholding value the equation [8] is as follows

$$S = \sum_{i=1}^{N} \frac{(A_i)^2}{8N}$$

Where S is the summation,
N is the total dimension of input image (N=256x256)
A_1, A_2…A_N are the actual image pixel intensity values after filtering.

2.4. Modified Canny Edge Detection

Edge detection is important step in object identification. It is the process of finding the drastic discontinuities. The drastic variation in the pixel intensity which shows boundaries of things in an image. So in short it can be summarized as finding the line drawing of an image which specify the feature extraction in image processing and it is utilized in computer vision algorithms such as recognition, tracking and in medical applications.
The edge detection method includes the use of operators like two dimensional filter. In an image an edge appears whenever the gradient value is more. There are so many operators are present to find out the edges in the image.
The Modified Canny operator uses two 3x3 kernel matrix one is horizontal gradient and one is vertical gradient [9-11]. They are as follows:

$$G_x = \begin{bmatrix}-\frac{1}{4} & 0 & \frac{1}{4}\\ -1 & 0 & 1\\ -\frac{1}{4} & 0 & \frac{1}{4}\end{bmatrix}$$

$$G_y = \begin{bmatrix} \frac{1}{4} & 1 & \frac{1}{4} \\ 0 & 0 & 0 \\ -\frac{1}{4} & -1 & -\frac{1}{4} \end{bmatrix}$$

The image is convolved with horizontal and vertical gradient. And the magnitude is calculated [12] from the below equation

$$Gradient\ (G) \approx |G_x| + |G_y|$$

From the moving window architecture pixel values are taken and those are convolved with the horizontal and vertical gradient filter. The hardware structure is implemented by using only shifters and adders/subtractors.

2.5 Thresholding

After gradient calculation there are many edges exist due to light intensity variations. To remove those unwanted edges we use thresholding. The equation for thresholding [12] is given as

$$Thresholding = \begin{cases} input\ value; if\ input\ data \geq adaptive\ threshold \\ 0; Otherwise \end{cases}$$

III . FPGA IMPLEMENTATION

The proposed architecture is implemented on Digilent ATLYS FPGA board. For coding standard VHDL language is used and synthesized those codes using Xilinx System Generator tool.

3.1 Hardware Utilizations

The hardware utilizations of the proposed architecture is given in Table 1.

Table 1: Hardware Utilization of Total Module

Device Utilization	
Logic Utilization	**Used**
No of Slice Registers	526
No of Slice LUTs	5490
No of fully used LUT-FF pairs	331
No of bonded IOBs	28
No of BUFG/BUFGCTRLs	2
No of DSP48Es	1
Frequency (MHz)	158.295

3.2 Implementations

The designed architecture in the System Generator tool is given in Fig. 3.

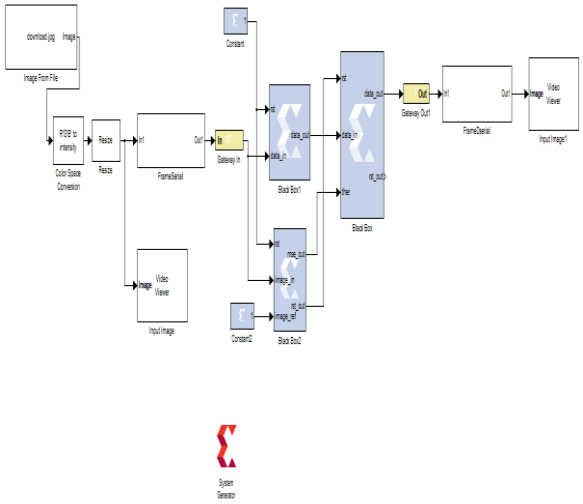

Fig. 3: System Generator Architecture of Canny Edge

The corresponding edge detected image is given in Fig.4

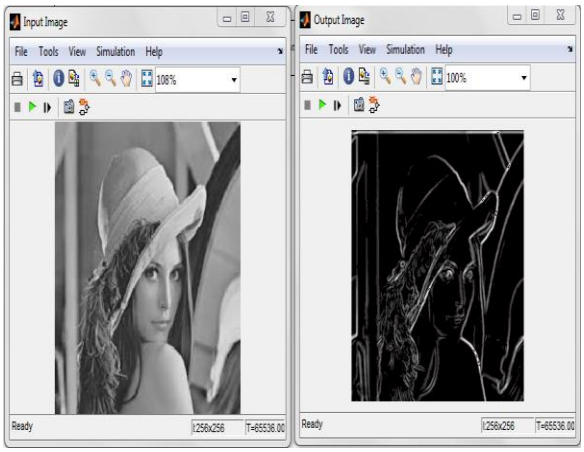

Fig. 4. Input and Output image in Software model of System Generator

IV. COMPARISONS WITH EXISTING TEXHNIQUE

In this section, the proposed technique is compared with existing technique in-terms of hardware and image quality to check the superiority of the design.

4.1 Hardware Comparisons

The frequency comparison of proposed canny edge detection and existing method is given in table 3. The architecture proposed by Qian Xu et al., [13] operated at maximum frequency of 133 MHz. But the proposed architecture can be able to operate at maximum frequency of 213 MHz. The proposed architecture working at higher frequency because there are no fractional part present in the modified equation and all multiplications and divisions are present in the equation in such a way that we can easily replace them by shifting.

Table 3: Frequency Comparisons of Proposed and Existing Canny Edge Detection

Authors	Frequency (MHz)
Qian Xu et al., [13]	133
Proposed	158.295

4.2 Image Comparisons

The comparison of modified canny edge detection in-terms of the output image of existing canny edge detection is shown in the Fig. 5. The structure proposed by Rong et al., [10] detect some unwanted edges along with required edges whereas the proposed structure minimizes those unwanted edges.

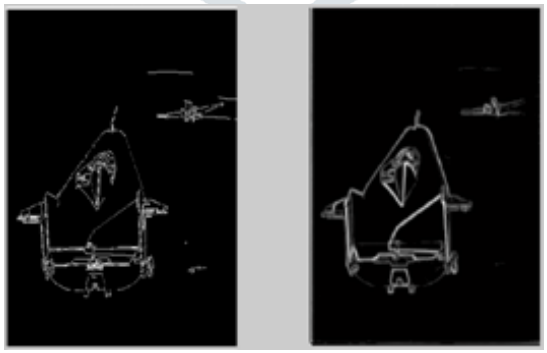

Fig. 5: Output Image Comparison for existing and proposed Canny Edge Detection

CONCLUSION

This paper describes the “FPGA implementation of Canny Edge Detection using adaptive technique” that used to detect the edge of any image as a complete image without dividing it into blocks. The proposed Block level Canny Edge detector has overcome the limitation of existing edge detection algorithms by reducing the delay and area. The design of Block Level Canny Edge Detector is coded in VHDL language. The simulation and synthesis of the design is carried out using Xilinx ISE 14.5 tool. The proposed method takes less area and less computational time result of this decreases latency and increases throughput. In Future, it can be possible to propose

dynamic based edge detection algorithm which can adapt for different variations of lighting conditions in image and also can be extended to video processing in detection of real time edges required for broadcasting.

REFERENCES

[1]. [Online]https://en.wikipedia.org/wiki/Canny_edge_detector
[2]. Rafael C Gonzalez and Richard E Woods, "Digital Image Processing", 3rd Edition, Prentice Hall, 2008.
[3]. Mohamed Nasir Bin Mohamed Shukor, Lo Hai Hiung, Patrick Sebastian, "Implementation of Real-time Simple Edge Detection on FPGA", International Conference on Intelligent and Advanced Systems, 2007.
[4]. Zhengyang Guo, Wenbo Xu and Zhilei Chai, "Image Edge Detection Based on FPGA", Ninth International Symposium on Distributed Computing and Applications to Business Engineering and Science, 2010.
[5]. Ferdous Hossain, Mithun Kumar P.K. and Mohammad Abu Yousuf, "Hardware Design and Implementation of Adaptive Canny Edge Detection Algorithm", International Journal of Computer Applications, Vol. 128, No. 9, pp. 31-38, 2015.
[6]. Mina Asaduzzaman, Md. Armanur Rahman, Mohammad Abu Yousuf and Ferdous Hossain, "Dynamic Thresholding based Adaptive Canny Edge Detection", International Journal of Computer Applications, Vol. 135, No. 4, pp. 37-41, 2016.
[7]. [Online]http://en.wikipedia.org/wiki/Kernel_%28image_processing%29.
[8].
[9]. Neethu P. R, "Cancer cell detection using distributed canny edge detector", International Research Journal of Engineering and Technology, Vol. 2, pp. 1224-1226, 2015.
[10]. R. Tourki, T. Saidani, M. Atri, D. Dia and W. Elhamzi, "Hardware Co-simulation For Video Processing Using Xilinx System Generator", Proceedings of the World Congress on Engineering, 2009.
[11]. Weibin Rong, Zhanjing Li, Wei Zhang and Lining Sun, "An Improved Canny Edge Detection Algorithm", International Conference on Mechatronics and Automation, pp. 577-782, 2014.
[12]. Rafael C. Gonzalez, Richard E. Woods and Steven L. Eddins, "Digital Image Processing", *Pearson Education*, 2004.
[13]. Qian Xu, Srenivas Varadarajan, Chaitali Chakrabarti and Lina J. Karam, "A Distributed Canny Edge Detector: Algorithm and FPGA Implementation", *IEEE Transaction on Image Processing*, Vol. 23, no. 7, pp.-2944-2960, July 2014.