ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Special Issue 8, March 2018

HIGH SPEED IMPLEMENTATION OF IMAGE DETECTION USING GRADIENT METHODS IN FPGA

S.Hariprasath[1], S.Sharanya[2], G.Ponmozhi[3], R.Parimala[4] , V.Sivaranajanai[5]
Assistant Professor, Department of EEE, Saranathan College of Engineering, Trichy, India[1]
UG Students, Department of EEE, Saranathan College of Engineering, Trichy, India[2,3,4,5]

Abstract Inthis paper, High Speed implementation of image edge detection using gradient methods is described. In the proposed methods, for the gradient edge detection, the a new technique named as Modified Moving Window Detection (MMWD) is superimposed over the source image and edge points are computed. The architecture is designed to process source image and the givenkernel with resolution of 25X25 and 5×5 pixels respectively. In this work two techniques namely MMWD process architecture foredge pixel computation and an optimized binary tree adder are proposed. For every moving window, the sum of product between source image and kernelis calculated in parallel processing mode to compute the edge pixels within two five cycles. In order to achieve area reduction, the adder scheme is modified. This Proposed System is Implemented using Verilog HDL 2005 version. Simulation is carried out by using imulated by Modelsim 6.4ctool and Synthesized by Xilinx 9.2 tool. The proposed system is implemented Spartan3XC3S200 TQ144 FPGA. The proposed design is efficient in terms of speed of operation. Total delay achieved is 5.985ns (4.724ns logic, 1.261ns route) (78.9% logic, 21.1% route). With the help of proper routing , the routing delay can be minizmed. The achieved frequency of operation is 170 Mhz which is sufficient for real time feature extraction and detection.

Index Terms—**Edge detection, Gradient methods, FPGA design, kernel matching, sum of absolute difference**

I. INTRODUCTION

Edge detection is a renowned technique in digital image processing for finding small parts of an image at which there are variations in image brightness or presence of discontinuities. Generally edge detection is used as a part in manufacturing process as a tool for quality control, a way to navigate a mobile robot or as a method to detect edges in images. The points at which there are sharp variation in the brightness of the image pixels area agglomerated as a set of curved line segments. In order to identify the whether the pixel in current position of the source image is an edge pixel , a smaller matrix of known values denoted as kernel is defined and movethe input image (as similarto two dimensionalconvolution). By superimposing the kernel window over the source image (as patch of the input source image under the window), the sum of product of the values are estimated. In order to find the variations in the brightness over the length as well the breadth of the input patch, kernel windows are defined for both the directions.

After the estimation of the sum in both the directions, the resultant value is taken as sum of horizontal variations and vertical direction variations. The calculated value replaces the intensity value at the centre of the input image patch. If input image is of size (WxH) and template image is of size (wxh), output image will have a size of (W-w+1, H-h+1). The parameters considered here (W,H)are taken as width and height of the rectangle window that slides over the source image. The chosen rectangle is the region of patch of the image. In this work, Modified Moving Window Detection technique is chosen to detect the patch pixels, since the MMWD technique is a unpretentious operation and can be simplified as transposition operations and additions. In this proposed work, the objective is to design and implement the intensity edge detection algorithm in FPGA such that the resultant design is delay optimized and also equitably occupies smaller area. Thus, two techniques namely LUT based moving window detection and optimized binary tree adder are proposed in this work. The rest of the paper is structured in the following manner. The survey of the existing systems is presented in section II. The proposed system is described in section III. The MMWDtechnique is described in section IV. The results are discussed in section V. Finally the future enhancement of the proposed design is described in section VI.

II. EXISTING SYSTEMS

In the work titled as Design of Sobel Operator Based Image Edge Detection Algorithm on FPGA by Girish Chaple and R.D.Daruwala [1], the execution time gets reduced due to parallelism technique. In their work Xilinx ISE Design Suite 14 (VHDL Language) was used.The designed system consist of two modules namely pixel generation and sobel edge detection module. The first module contains two FIFOs and three Shift registers. Sobel

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Special Issue 8, March 2018

edge detection module contains convolution, addition, threshold block.

In another paper [2] titled asArea Optimized Fpga-Based Implementation of The Sobel Compass Edge Detector, published by Sanjay Singh [2], the proposed architecture uses single processing element to compute the gradient for all directions. It was reported that it utilized less than forty percent of the FPGA resources compared to standard implementation presented in real time constraints. In the reported work, Sobel compass edge detector is used which increases the computational complexity. In the paper [3] titled Fpga Based Implementation of Image Edge Detection Using Canny Edge Detection Algorithm authored by K.Naresh, M.Mahender, the hardware device chosen was Vertex5 FPGA. In this work, first order derivative was preferred to second order derivative. Sobel operator was preferred over Prewitt operator and Robert operator due to simplicity. The designed hardware operated at 54.5MHz frequency with 23% resource utilization.

In the paper [4] titled as FPGA-based template matching using distance transform authored by S. Hazel, A. Kugel, R. Manner, and D. M. Gavrilaa high-performance FPGA solution to generic shape-based object detection in images is proposed. The underlying detection method involves representing the target object by binary templates containing positional and directional edge information. A particular scene image is pre-processed by edge segmentation, edge cleaning and distance transforms. Matching involves correlating the templates with the distance-transformed scene image and determining the locations where the mismatch is below a certain user-defined threshold. In this paper we present a step by step implementation of the components of such object detection systems, taking advantage of the data and logical parallelism opportunities offered by FPGA architecture. The realization of a pipelined calculation of the pre-processing and correlation on FPGA is presented in detail.

In the paper[5] titled as Using template matching for object recognition in infrared video sequences authored by the authors I. Pham, R. Jalovecky, and M. Polasekthe similarity criteria normalized cross correlation is described. The cross correlation is not invariant to changes in image intensity such as lighting conditions, and the range of correlation coefficient is dependent on the size of the feature, while we can normalize for the effect of changing intensity and template size by using normalized cross correlation. The basic principle of the algorithm is based on the assumption the object is selected at time t with the centre of mass T and the greatest relative velocity between the camera and the target vct is always known.

III.PROPOSED SYSTEM

Edge detection based on intensity difference with the neighbourhood pixels of the current pixel is a technique used for identifying variations in the intensity values of the pixels in the source image.Window matching needs two primary components: source image and Kernel image; and three sequential processing stages: gray scaling, binarization, and moving window computation. Gray scaling processes is called preprocessing stage. The first step in the proposed system is gray scaling. The input color image is converted into gray scale image by standard conversion procedure and the pixels are read as a text file. Matlab tool is utilized for this task. The Pixel Values of Template Image and Main Images are stored in Memory. After this process is finished, the pixels are read and MMWD process on the input pixels is performed to complete the sum of product process.MMWD is a calculation that sums the product of gray levelper pixel, between source and kernel image. LetI(x,y) is the source image andk(x,y) is the kernel image thenMMWDprocedure is described by (1). Here, x and y represent the position of pixel coordinates (x,y). If the the resultant sum value is greater than the maximum value of the of type of the image chosen then a value of one is stored in the center of the moving window and zero value is stored otherwise.The sum using MMWD process is implemented as the Processing element (PE) of each pixel.

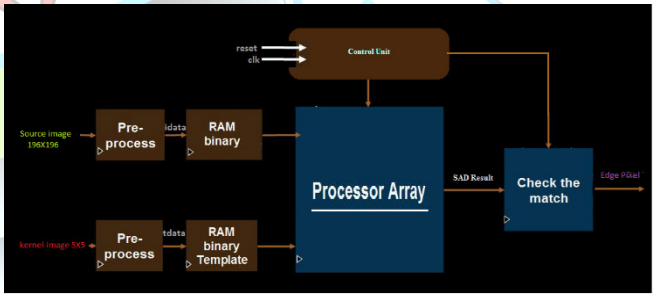

Fig 1: Proposed Template Matching System

The MMWD procedure is executed to find the sum of product between the present position of the image and kernel Images. If the computed pixel value is greater than the maximum value of the original source image, then the resultant computation produces one as the result, otherwise the calculated value is retained and stored as the pixel value in the source image. Based on repeating the aforementioned step continuously until the entire image is scanned by the

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Special Issue 8, March 2018

moving window, the scanning processing is completed. Then by comparing the computed values with the threshold, the edge pixel map of the image patch is estimated. The comparator section gives the co-ordinate values of thepixels where value one is written and for the rest of the pixel positions value zero is written.The co-ordinates are given as input to the binarizer module which produces the binary image. The binarizer output values are stored in the memory. After all the pixels are scanned by the moving window, the values are exported to Matlab tool to view the matched result. The complete flow the proposed work is depicted in figure 2.

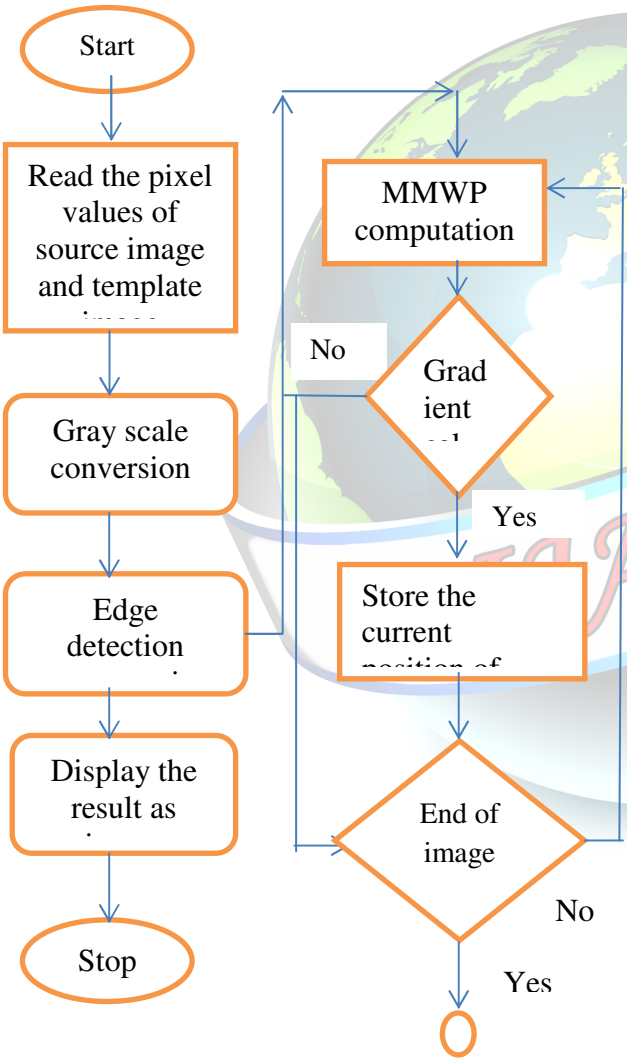

Figure 2 flowchart of the proposed method

IV. PROPOSED SAD ALGORITHM

In this work, a 2-D intra-level architecture called the Propagate Partial SAD is proposed as MMWD process. The architecture is composed of PE arrays with a 1-D adder tree in the vertical direction. Current pixels are stored in each PE, and two sets of continuous reference pixels in a row are broadcasted to PE arrays at the same time. In each PE array with a 1-D adder tree, distortions are computed and summed by a 1-D adder tree to generate one-row SAD. The row SADs are accumulated and propagated with propagation registers in the vertical direction The reference data of searching candidates in the even and odd columns are inputted by Ref. Pixels 0 and Ref. Pixels 1, respectively. After initial cycles, the SAD of the first searching candidate in the zero th column is generated, and the SADs of the other searching candidates are sequentially generated in the following cycles. When computing the last searching candidates in each column, the reference data of searching candidates in the next columns begin to be applied as input values through another reference input. Then, the hardware utilization is 100% except the initial latency. In Propagate Partial SAD, by broadcasting reference pixel rows and propagating partial-row SADs in the vertical direction, it provides the advantages of fewer reference pixel registers and a shorter critical path. The specific PE in Fig. 3 Estimates the absolute difference between the Cur_pixel of the search area and the Ref_pixel of the current macroblock. Thus, by utilizing PEs, SAD shown in as follows, in a macroblock with NxN size of can be evaluated:

$$SAD=\sum_{i=0}^{N-1}\sum_{j=0}^{N-1} x_{ij} - y_{ij} \quad \ldots\ldots (1)$$

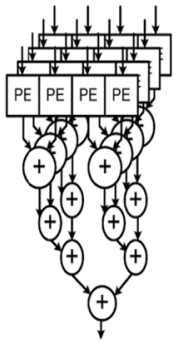

Figure 3. SAD Tree Architecture

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Special Issue 8, March 2018

The pixel example is shown in figure 4.

	0	1	2	3
0	128	128	64	255
1	128	64	255	64
2	64	255	64	128
3	255	64	128	128

Cur_pixel

	0	1	2	3
0	1	1	2	3
1	1	2	3	4
2	2	3	4	5
3	3	4	5	5

Ref_pixel

Figure 4. Pixel example for SAD computation

The concept of the proposed SAD Tree architecture. The proposed SAD Tree is a 2-D intra-level architecture and consists of a 2-D PE array and one 2-D adder tree with propagation registers Current pixels are stored in each PE, and reference pixels are stored in propagation registers for data reuse. In each cycle, current and reference pixels are inputted to PEs. Simultaneously, continuous reference pixels in a row are inputted into propagation registers to update reference pixels. In propagation registers, reference pixels are propagated in the vertical direction row by row. In SAD Tree architecture, all distortions of a searching candidate are generated in the same cycle, and by an adder tree, distortions are accumulated to derive the SAD in one cycle. In order to provide a high utilization and data reuse, the snake scan is adopted and reconfigurable data path propagation registers are developed in the proposed SAD Tree, which consists of five basic steps from A to E. The first step, A, fetches pixels in a row and the shift direction of propagation registers is downward. When calculating the last candidates in a column, one extra reference pixel is required to be inputted, that is, step B. When finishing the computation of one column, the

When finishing the computation of one column, the reference pixels in the propagation registers are shifted left in step C. Because the reference data have already been stored in the propagation registers, the SAD can be directly calculated. The next two steps, D and E, are the same as steps A and B except that the shift direction is upward. After finishing the computation of one column in the search range, we execute step C and then go back to step A.

V.RESULT AND DISCUSSION

PROPOSED MULTIPLIER RESULTS

Device utilization summary of the proposed system is given in table 1.

Table 1. Device utilization summary

Selected Device: 3s500efg320-4

S.no	Components	Used	Available	% of utilization
1	Number of Slices	184	4656	3
2	Number of Slice Flip Flops	226	9312	2
3	Number of 4 input LUTs	181	9312	1
4	Number of IOs	39	232	16%
5	Number of bonded IOBs	39	232	16%
6	Number of GCLKs:	1	24	4%

SYNTHESIS REPORT:

Final Results
RTL Top Level Output File Name: sobel3x3_pixel_pipeline.ngr
Top Level Output File Name: sobel3x3_pixel_pipeline
Output Format: NGC
Optimization Goal: Speed
Keep Hierarchy : No
Timing Summary:

Speed Grade: -4
Minimum period: 4.292ns (Maximum Frequency: 232.992MHz)
Minimum input arrival time before clock: 5.985ns
Maximum output required time after clock: 11.119ns
Maximum combinational path delay: 4.937ns
Total 5.985ns (4.724ns logic, 1.261ns route)
(78.9% logic, 21.1% route)

RTL synthesis outputs are shown in figure 5,6,7 respectively.

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Special Issue 8, March 2018

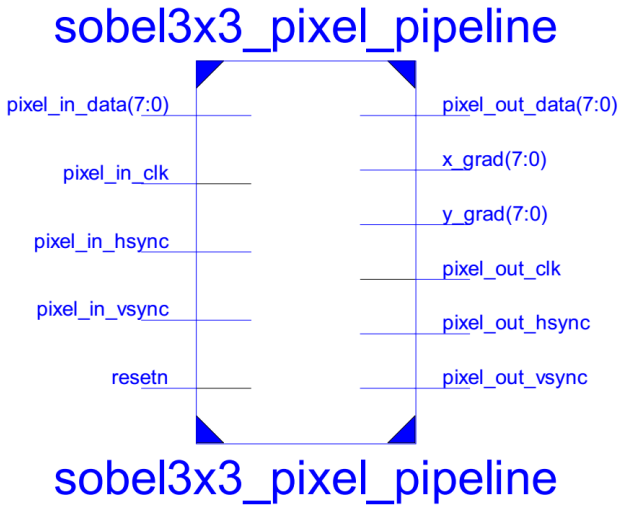

Figure 5. Main view

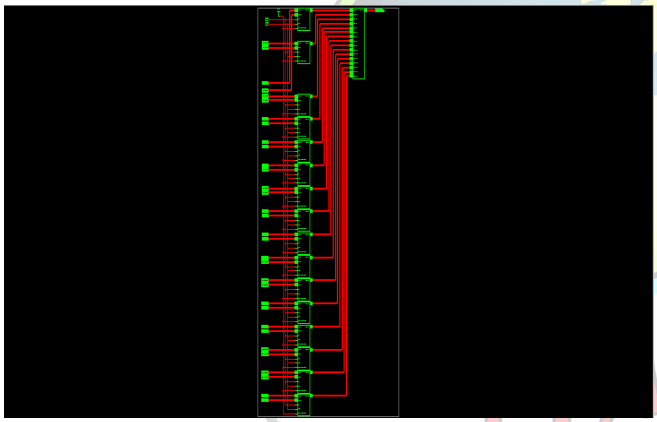

Figure 6. Inner view

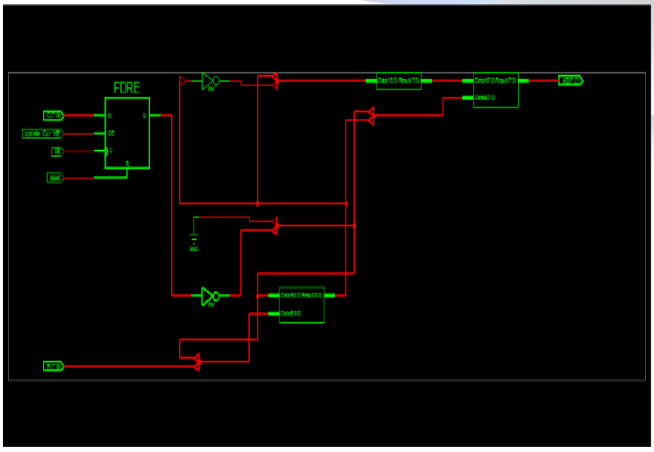

Figure 7. PE view

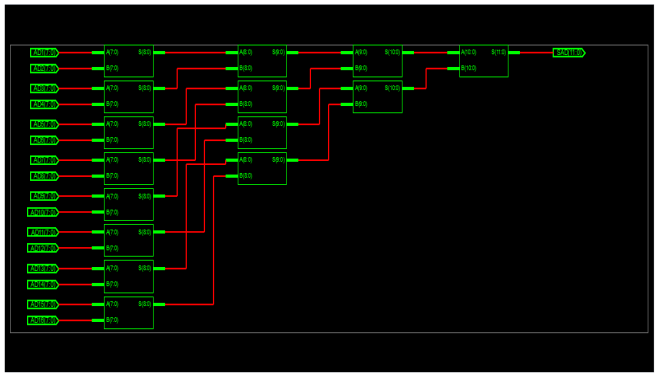

Figure 8. MMWD adder Tree

VI.CONCLUSION

The proposed PE array for MMWD processor occupies only 12% of the Spartan 3E chip. The delay achieved is 5.85ns. The designed system is operating at fairly high speed for real time applications. By incorporating speed optimization techniques such as double pipelining mechanism, the speed can be further increased.

REFERENCES

[1] Girish Chaple, R.D.Daruwala"Design Of Sobel Operator Based Image Edge Detection Algorithm On Fpga" in proceedings of International Conference on Communication and Signal Processing ,April 3-5,2014.

[2]Sanjay Singh "Area Optimized Fpga-Based Implementation Of The Sobel Compass Edge Detector"in Proc. of 23rd Signal Processing and Communications Applications on 4th February, 2015

[3]K.Naresh,M.Mahender"FpgaBased implementation of Image Edge Detection Using Canny Edge Detection Algorithm" in International Journal of Scientific Engineering and TechnologyResearch , 2016.

[4] I. Pham, R. Jalovecky, and M. Polasek, "Using template matching for object recognition in infrared video sequences," in Proc. Of 2015 IEEE/AIAA 34th Digital Avionics Systems Conf., pp. 8C5-1–8C5-9, September 2015.

[5] S. Hazel, A. Kugel, R. Manner, and D. M. Gavrila, "FPGA-based template matching using distance transform," in Proc. of 10th Annual IEEE Symp. on Field-Programmable Custom Computing Machines, pp. 89-97, April 2002.

[6] T. Adiono, M. D. Adhinata, N. Prihatiningrum, R. Disastra, R. V. W. Putra, and A. H. Salman, ''An architecture design of SAD based template matching for fast queue counter in FPGA,'' in Proc. of The 2016 Int. Symp. on Intelligent Signal Processing and Communication Systems, pp. 1-4, October 2016. (accepted)