VLSI Implementation of Image Segmentation with Resource Optimized Adaptive Median Filter

Prof. V. B. Baru.
Assistant Professor, E & TC,
Sinhgad College of Engineering, Pune, India.

Chetan S. Deokar.
P.G. Student (M.E. E &TC),
Sinhgad College of Engineering, Pune, India.

ABSTRACT
FPGA (Field Programmable Gate Array) is utilized to realize the image segmentation. We describe a newfangled method for segmentation of 2-D imagery that uses resource optimized adaptive median filter for image enhancement. A pipelined implementation on FPGA for this algorithm is designed. Sobel operator is used for edge detection. Processed results of test images are presented to illustrate the performance capabilities of the proposed method. The FPGA resource utilization for proposed architecture is 50% less compared to the Adaptive Median Filter and the variation in the final picture quality is only ± 1dB PSNR.

General Terms
Image enhancement, resource optimized implementation, edge-detection, Xilinx

Keywords: VLSI, FPGA, Adaptive Median Filter (AMF), Image segmentation, FPGA, Median, Sobel-operator, Resource Optimized Adaptive Median Filter (ROAMF).

1. INTRODUCTION
Implementing hardware design in FPGAs (Field Programmable Gate Arrays) is a formidable task. There is more than one way to implement the digital filter. Based on the design specification, careful choice of implementation method and tools can save a lot of time and work. Most of the algorithms are computationally intensive, so it is desirable to implement them in high performance reconfigurable systems. Recently, FPGA technology has become a viable target for the implementation of algorithms for image processing. Firstly, the algorithm is simulated in MATLAB, and then the same is implemented into VHDL with the help of Xilinx ISE and the ModelSim simulation results are verified with MATLAB results.

2. RELATED WORK
Conventionally, the Gaussian filter is used for the image enhancement purpose followed by first differentiation of it for the edge-detection. From simulation results, we observed that the AMF (Adaptive Median Filter) gives better results as compared to it. Therefore we propose to use the adaptive median filter instead of the Gaussian and median filter. We have proposed implementation of a resource optimized adaptive median filter. It is well known that the Sobel-edge operator gives expected result for edge detection; hence we propose to use the Sobel operator instead of first differentiation of the filter used in the first step.

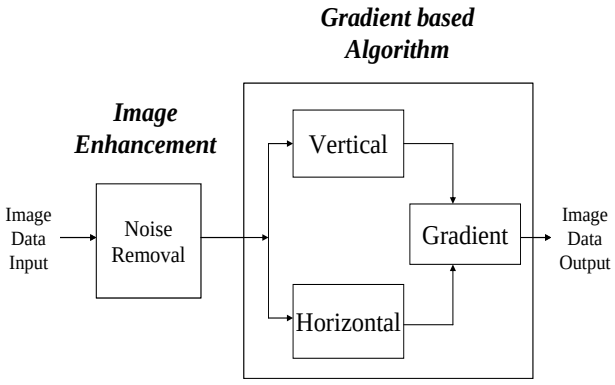

Fig. 1: Approach for FPGA Implementation

The approach for the proposed implementation is shown in the Fig. 1. The input image is first filtered for image enhancement. The result of this step is given to the edge-detection filter. The output of this filter gives the segmented image.

2.1 Why Adaptive Median Filter?
Traditionally, mean and median filter are used for image enhancement. The simulation results for mean filter, median and adaptive median filter are presented for comparison. The Fig. 2 shows the results with noise of level 0.4, whereas the Fig. 3 shows the results with a level 0.6.

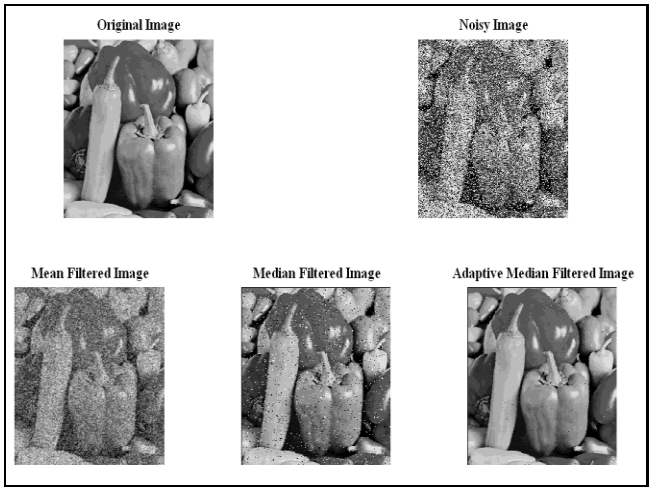

Fig. 2: Results with noise level 0.4

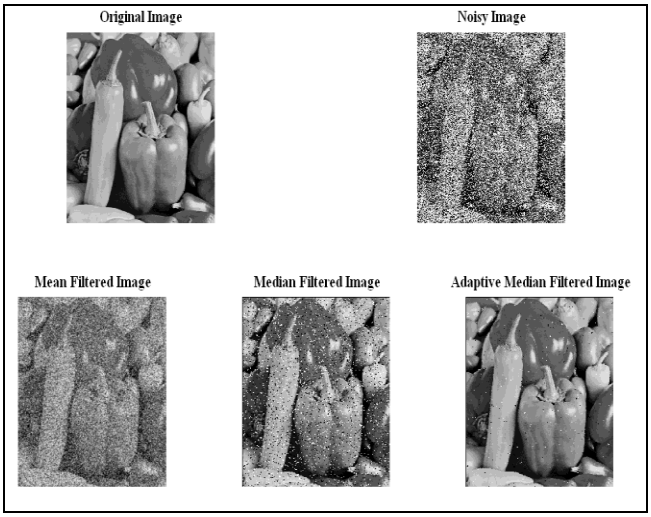

Fig. 3: Results with noise level 0.6

The careful examination shows that the adaptive median filter preserves the sharpness in a better way. Also, it removes the noise level more than mean as well as median filter.

2.2 FPGA Implementation

2.2.1 Median Filter:

The Algorithm for finding median of 3x3 mask is as follows:

1. Sort all the rows.
2. Sort all the columns.
3. Sort both the diagonals.
4. Middle value of step 3 is '*Median'*.

The steps are elaborated in the Fig. 4.

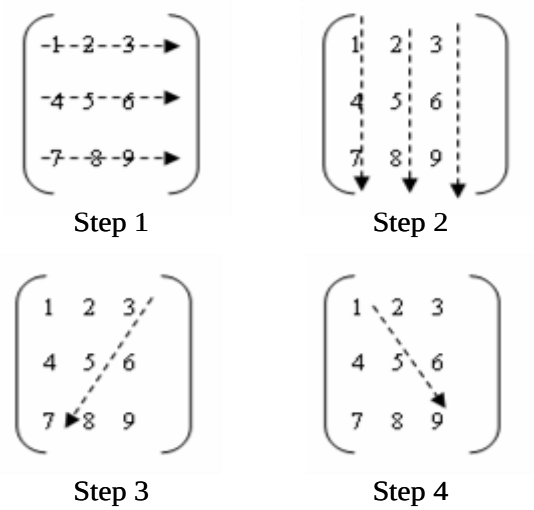

Fig. 4: FPGA Implementation of Median Filter

2.2.2 Adaptive Median Filter:

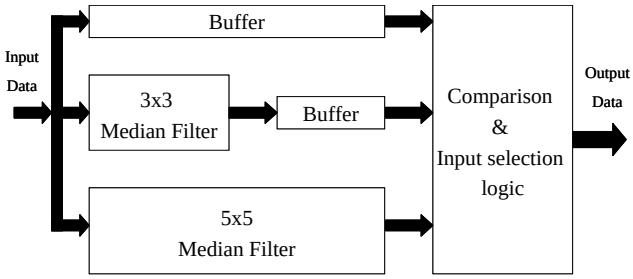

Fig. 5: FPGA Implementation of Adaptive Median Filter

The Fig. 5 shows the approach for FPGA implementation. Median filter with two masks are realized simultaneously. The output from either filter is selected depending on the conditions status. The buffers are used to compensate the latency introduced by median filters with different masks.

The adaptive median filter works as follows:

Z_{min} = Minimum gray level value in S_{xy}.
Z_{max} = Maximum gray level value in S_{xy}
Z_{med} = Median of gray levels in S_{xy}
Z_{xy} = gray level at coordinates (x, y)
W_{max} = Maximum allowed size of S_{xy}

Block 1:

If (($Z_{min} < Z_{xy}$) AND ($Z_{xy} < Z_{max}$))
    Output = Z_{xy}
Else If ($Z_{min} < Z_{med}$ AND ($Z_{med} < Z_{max}$)
    Output = Z_{med}
Else
    Increase the window size

If (window_size <= W_{max})
    Repeat Block 1

The implementation of 5x5 median filter introduces the problem of resources. It consumes enormous resources. To fit the design with device resource constraints, we have proposed a method which consumes very less resources as compared to the original 5x5 median filter.

The new algorithm is as follows:

1. Sort all the 5 columns.
2. Sort the median of all 5 columns.
3. Take the median of sorted elements from step 2.

This algorithm requires 150 comparisons for 5x5 mask while original 5x5 median filter requires 625 comparisons.
As the number of comparisons are reduced, the power consumption reduces.

Though the above algorithm does not provide the median of 25 elements, it works excellently when used as a module in adaptive median filter. The simulation results for the comparison of adaptive median filter (with original 5x5 median filter) and approximate adaptive median filter (with above algorithm) are shown in fig. 6.

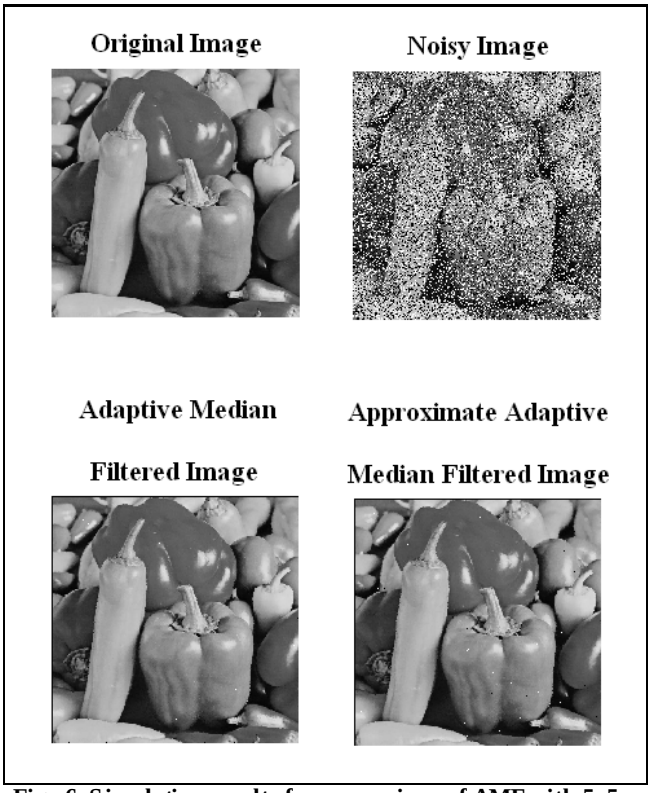

Fig. 6: Simulation results for comparison of AMF with 5x5 median filter & modified filter for 5x5 mask

Table below shows the comparison based on PSNR for the evaluation of adaptive median filter with modified filter for 5x5 mask.

Table 1: Comparison based of Adaptive Median Filter (AMF) and Resource Optimized Adaptive Median Filter (ROAMF) on PSNR

Noise Level	PSNR (dB)		Absolute difference	% difference
	AMF	ROAMF		
0.0	38.05	38.05	0.00	0.006570
0.1	37.58	37.58	0.00	0.007451
0.2	36.57	36.57	0.00	-0.004376
0.3	35.46	35.45	0.01	0.024537
0.4	34.39	34.37	0.02	0.060184
0.5	33.40	33.35	0.05	0.155989
0.6	32.38	32.33	0.05	0.161530
0.7	31.25	31.16	0.08	0.267862
0.8	30.06	29.94	0.11	0.381912
0.9	28.64	28.56	0.09	0.300228
1.0	27.07	27.08	-0.01	-0.046547

2.3 Edge-Detection

Gradient-based edge detection is the most common approach for detecting meaningful discontinuities in gray level. This leads to formalism in which "meaningful" transitions in gray levels can be measured. An ideal edge is a set of connected pixels, each of which is located at a step transition in gray level.

The gradient of an image *f(x, y)* at location *(x, y)* is defined as vector. The gradient vector points in the direction of maximum rate of change of f at coordinates *(x, y)*. An important quantity in the edge detection is magnitude of this vector, denoted as f, where

$$f = \sqrt{G_x^2 + G_y^2} \quad (1)$$

where G_x = Horizontal gradient
G_y = Vertical gradient

The direction of the gradient vector is also important quantity. Let $\theta(x, y)$ represent the direction angle of vector *f* at *(x, y)*. Then, from vector analysis,

$$\theta(x, y) = \tan^{-1}\left(\frac{G_y}{G_x}\right) \quad (2)$$

where the angle is measured with respect to the x-axis. The direction of an edge at *(x, y)* is perpendicular to the direction of the gradient vector at that point. Computation of the gradient of an image is based on obtaining the partial derivative ($\partial f/\partial x$) and ($\partial f/\partial y$) at every pixel location.

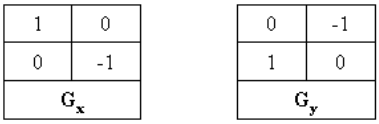

Fig. 7: Robert-cross operator

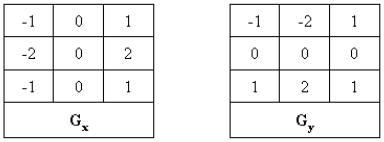

Fig. 8: Sobel-operator

Though masks of size 2x2 require fewer computations, they are awkward to implement because they don't have a clear center. An approach using masks of size 3 x 3 is preferred. The equations required to be implemented are shown below:

$$G_x = (I_7 + 2*I_8 + I_9) - (I_1 + 2*I_2 + I_3) \quad (3)$$

&

$$G_y = (I_3 + 2*I_6 + I_9) - (I_1 + 2*I_4 + I_7) \quad (4)$$

$$f = \sqrt{G_x^2 + G_y^2} \quad (5)$$

For calculation of square-root, cordic core available in Xilinx ISE can be used. It consumes a lot of resources. Alternatively, equation of approximate square-root calculation is used which requires less amount of hardware.

2.3.1 Alpha max plus beta min algorithm:

The alpha max plus beta min algorithm is a high-speed approximation of the square root of the sum of two squares.

$$|Y| = \sqrt{A^2 + B^2} \quad (6)$$

The algorithm avoids the necessity of performing the square and square-root operations and instead uses simple operations such as comparison, multiplication and addition. Specific choices of α and β coefficients of the algorithm allow the multiplication operation to be reduced to a simple shift of binary digits that is particularly well suited to implementation in high-speed digital circuitry.

The approximation is expressed as:

$$Y = \alpha * Max + \beta * Min \quad (7)$$

Where ***Max*** is the maximum absolute value of A and B and ***Min*** is the minimum absolute value of A and B.

For the closest approximation, the optimum values for α and β are 0.96 and 0.39. For FPGA implementation, the value of α should be adjusted such that β becomes 0.5. It allows reduction of one multiplier. After statistical analysis, the new value of α is 0.93. The variation of error with modified equation of alpha max plus beta min algorithm is shown in Fig. 9. The analysis evidences that the maximum error between exact square-root and approximate square-root is 9%.

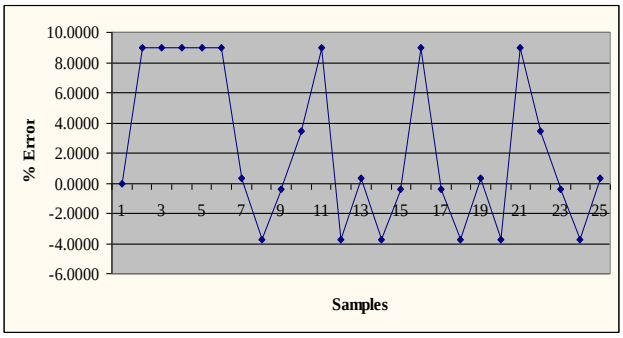

Fig. 9: Analysis of alpha max plus beta min algorithm with proposed coefficients

3. SIMULATION WAVEFORMS

The ModelSim simulation waveforms for the modules performing sorting are shown in Fig. 10 and Fig. 11.

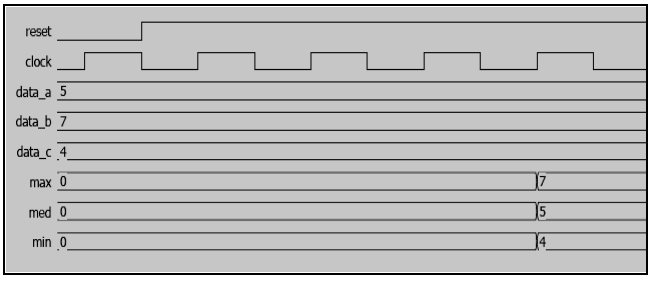

Fig. 10: Sorting of 3 data elements

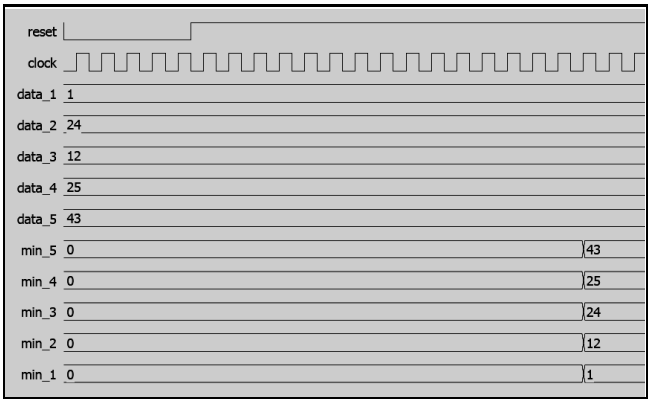

Fig. 11: Sorting of 5 data elements

4. FPGA IMPLEMENTATION RESULTS

Device : XC3S400

1. Median Filter with 3x3 mask

Power Consumption	:	0.056 W	
Number of RAMB16s	:	3	(18 %)
Number of Slices	:	395	(11 %)

2. Modified Filter for 5x5 mask

Power Consumption	:	0.056 W	
Number of RAMB16s	:	5	(31 %)
Number of Slices	:	1480	(41 %)

3. Sobel Edge-detection

Power Consumption	:	0.056 W	
Number of RAMB16s	:	3	(18 %)
Number of Slices	:	133	(3 %)

5. CONCLUSION

The integral components of the implementation are image smoothening by adaptive median filter and edge-detection with Sobel-operator. Resource optimized hardware implementation of adaptive median filter is presented. The resource utilization for proposed architecture is 50% less compared to the Adaptive Median Filter and the variation in the final picture quality is only ± 1dB PSNR.

Further, a version of the alpha max plus beta min algorithm is devised to find the edge-strength in edge-detection module. This contribution presented reduction in one multiplier for implementation in FPGA hardware mainly to solve resource issue. The proposed strategy has been conducted to implement FPGA based square-root of sum of squares successfully. The results have shown that proposed method has maximum 9% deviation as compared to exact value.

6. FUTURE SCOPE

The integration of multiple algorithms for image segmentation in addition to Sobel-edge detection and binary image segmentation can be considered.

7. ACKNOWLEDGMENT

We would like to thank Aranyeshwar Consultancy & Testing Services (R&D Engineers), Nashik, India for their technical and financial support during our research.

8. REFERENCES

[1] Zdenek Vasicek, Lukas Sekanina, "Novel Hardware Implementation of Adaptive Median Filters", 978-1-4244-2277-7.

[2] Miguel A. Vega-Rodríguez, Juan M. Sánchez-Pérez, Juan A. Gómez-Pulido, "An FPGA-based Implementation for Median Filter Meeting Real-time Requirements of Automated Visual Inspection Systems", 10th Mediterranean Conference on Control and Automation - MED2002 Lisbon, Portugal, July 9-12, 2002.

[3] Hong Shan Neoh, Asher Hazanchuk, "Adaptive Edge Detection for Real-Time Video Processing using FPGAs, Altera Corporation, Innovation Drive.

[4] Chetan Deokar, Dhiraj M. Dhane, "FPGA based Implementation of Real-time Image Segmentation", Journal of Software Engineering and Technology, International Sciences Press. ISSN: 0975-6159, vol.2 No.1 Jan-June 2010.

[5] Chetan Deokar, V. B. Baru., "VLSI based Implementation of Real-time Image Segmentation", Proc. International Conference on Industrial Electronics, Control and Robotics (IECR10), NIT, Rourkela, Orissa, India, Dec.27-29, 2010.