

Decision Based Median Filter Algorithm Using Resource Optimized FPGA to Extract Impulse Noise

Rutuja N. Kulkarni^{1*}, P.C. Bhaskar²

¹Research Scholar, Department of Technology, Shivaji University, Kolhapur

²Professor, Department of Technology, Shivaji University, Kolhapur

*Corresponding author: kulkarni.rutuja@gmail.com

Received September 02, 2013; Revised March 20, 2014; Accepted March 22, 2014

Abstract Median filter is a non-linear filter used in image processing for impulse noise removal. It finds its typical application in the situations where edges are to be preserved for higher level operations like segmentation, object recognition etc. This paper presents accurate and efficient noise detection and filtering algorithm for impulse noise removal. The algorithm includes two stages: noise detection followed by noise filtering. The proposed algorithm replaces the noisy pixel by clipping median value when other pixel values, 0's or 255's are present in the selected window and when all the pixel values are 0's and 255's then the noise pixel is replaced by mean value of all the elements present in the selected window. This median filter was designed, simulated and synthesized on the Xilinx family of FPGAs (XC3S500E of Spartan-3E). The VHDL was used to design the above 2-D median filter using ISE (Xilinx) tool & tested & compared for different grayscale images.

Keywords: impulse, median filter, PSNR, salt & pepper, FPGA

Cite This Article: Rutuja N. Kulkarni, and P.C. Bhaskar, "Decision Based Median Filter Algorithm Using Resource Optimized FPGA to Extract Impulse Noise." *Journal of Embedded Systems*, vol. 2, no. 1 (2014): 18-22. doi: 10.12691/jes-2-1-4.

control that why FPGAs are often used as implementation platforms for image processing application.

1. Introduction

Salt & pepper noise is a type of impulse noise which is typically observed on images. It represents itself as randomly occurring white and black pixels. An image containing this type of noise will have dark pixels in bright regions and bright pixels in dark regions. This type of noise can be caused by dead pixels, analog-to-digital converter errors, bit errors in transmission, etc.

In past years, linear filters became the most popular filters in image signal processing. The reason of their popularity is caused by the existence of robust mathematical models which can be used for their analysis and design. However, there exist many areas in which the nonlinear filters provide significantly better results. The advantage of nonlinear filters lies in their ability to preserve edges and suppress the noise without loss of details. The success of nonlinear filters is caused by the fact that image signals as well as existing noise types are usually nonlinear. As salt and pepper noise is a random valued shot noise, it is very difficult to remove this type of noise using linear filters because they tend to smudge resulting images.

The proposed study was carried out to evaluate the performance on FPGA (XC3S500E of Spartan-3E) for image processing operations. Generally image processing system require a high speed and FPGA has fast executing speed, large memory and has capable of flexible logic

2. Related Work

Traditionally, the impulse noise is removed by a median filter which is the most popular nonlinear filter. However, the standard median filter [1] gives a poor performance for images corrupted by impulse noise with higher intensity. A simple median filter utilizing 3×3 or 5×5-pixel window is sufficient only when the noise intensity is less than approx. 10-20%. When the intensity of noise is increasing, a simple median filter remains many shots unfiltered. Adaptive Median Filter (AMF) [2] perform well at low noise densities. But at high noise densities the window size has to be increased which may lead to blurring the image. The Adaptive Rank Order Filter method in [8] provides better filtering properties than it is possible with Adaptive Median Filter (AMF). The AROF (Adaptive Rank Order Filter) adapts the window size itself when all pixels within the current window are noisy or when median itself is noisy. The AROF VLSI architectures developed is implemented on Xilinx Virtex XC2VP50- 7ff1152 FPGA device. The pipelining and parallel processing techniques have been adapted in order to speed up the filtering process. Paper [9] implement standard median filter and improve the computational speed of different image Enhancement Techniques on FPGA. The simulation and synthesis results were obtained and designs are successfully

validated using hardware simulation feature of using Cyclone III family chip type EP3C16F484C6 on development kit type DE0.

In switching median filter [3,4] the decision is based on a pre-defined threshold value. The major drawback of this method is that defining a robust decision is difficult. Also these filters will not take into account the local features as a result of which details and edges may not be recovered satisfactorily, especially when the noise level is high. To overcome this drawback, an improved decision based algorithm is proposed.

3. Proposed Algorithm

The proposed Decision Based Median Filter algorithm processes the corrupted images by first detecting the impulse noise. The processing pixel is checked whether it is noisy or noisy free. That is, if the processing pixel lies between maximum and minimum gray level values then it is noise free pixel, it is left unchanged. If the processing pixel takes the maximum or minimum gray level then it is noisy pixel which is processed by proposed algorithm.

3.1. Algorithm

Step 1: Select 2-D window of size 3×3 . Assume that the pixel being processed is P_{ij}

Step 2: If centre pixel $0 < P_{ij} < 255$ then P_{ij} value is left unchanged.

Step 3: If $P_{ij} = 0$ or $P_{ij} = 255$ then check for next condition

Step 4: If processing pixel is 0 or 255 & also surrounding all elements has same value then processing element is an information instead of noise as there is high co-relation between neighboring pixels so pixel value should keep as it was. Otherwise check for next condition.

Step 5: P_{ij} is a corrupted pixel then two cases are possible as given in Case i) and ii).

Case i): If the selected window contain all the elements as 0's and 255's. Then replace P_{ij} with the mean of the element of window.

Case ii): If the selected window contains not all elements as 0's and 255's. Then eliminate 255's and 0's and find the median value of the remaining elements. Replace P_{ij} with the median value.

Step 6: If more than 50% of the selected pixels in a window are corrupted then there is a possibility to get corrupted median of a window. For this condition median can be replaced by nearest information pixel.

Step 7: Repeat steps 1 to 6 until all the pixels in the entire image are processed.

4. System Architecture

The architecture of the proposed system is shown in figure. The system is composed of PC, Universal asynchronous receiver and transmitter (UART), Random access memory (RAM) and improved decision based median filter. PC interfacing is optional and carried only for the testing purpose. In actual implementation, image from the acquisition system will be fed for processing.

Image data is taken from the PC and processed image is again sent back to PC.

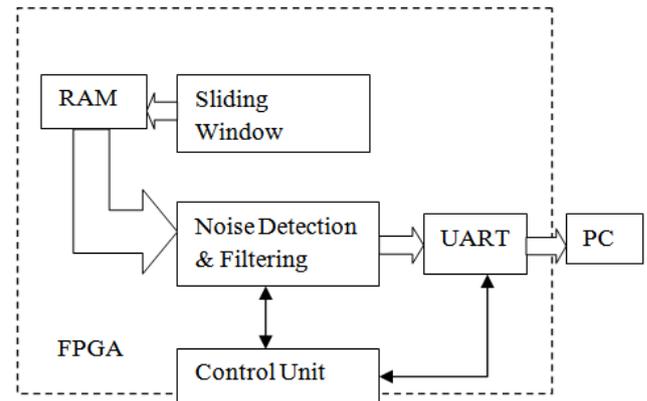


Figure 1. FPGA system architecture

4.1. Design Methods

Proposed system consist of following components

i) Proposed Median Filter

Median filter is nonlinear spatial filter. It is particularly effective in the presence of impulse noise also called salt and pepper noise. Simple median filter is improved by an algorithm stated above & is implemented on FPGA.

ii) UART

UART includes a transmitter and a receiver. The transmitter is essentially a special shift register that loads data in parallel and then shifts it out bit by bit at a specific rate. The receiver, on the other hand, shifts in data bit by bit and then reassembles the data. No clock information is conveyed through the serial line. Therefore before the transmission begins, the transmitter and receiver must agree on a set of parameters in advance, which include the baud rate, number of data bits, stop bits, and, parity bit. Since the voltage level defined in RS-232 is different from that of FPGA I/O, a voltage converter chip is needed between a serial port and an FPGA's I/O pins. In our system UART transmit the data from the FPGA to PC. The design is customized for a UART with a 9600 baud rate, 8 data bits, 1 stop bit, and no parity bit.

iii) RAM

RAM is used as a temporary storage for the image data before processing through the median filter block. For the experiments images size is chosen as 60 X 125 pixels. All Spartan-3E devices support block RAM, which is organized as configurable, synchronous 18 Kbit blocks.

5. Implementation

The median filter is implemented using window of size 3×3 , The proposed architecture for median filter was tested on the image 60 X 125 pixels. The image was transferred to the target FPGA Spartan-3E (XC3S500E) during configuration The median filtered image was transferred back to the PC for comparison purposes. MATLAB 7.8 is used to send start signal to FPGA through UART & filtered image pixels are collected in MATLAB for display. The implementation was carried out at the clock of 50 MHz. The percent utilization of the target device was estimated.

6. Experimental Results

Following figure shows the simulation results for the above implementation of a proposed decision based median filter of window size 3x3.

If only processing element is faulty

If the selected window contains salt/pepper noise as processing pixel (i.e., 255/0 pixel value) and neighboring pixel values contain information, then median of selected window is calculated by sorting. Example is given below & also simulation result shows the result of median. For given ex. by sorting window elements '1B' is median which replaces processing element.

$$\begin{pmatrix} 1B & 19 & 1D \\ 0F & < FF > & 37 \\ 1F & 16 & 14 \end{pmatrix}$$

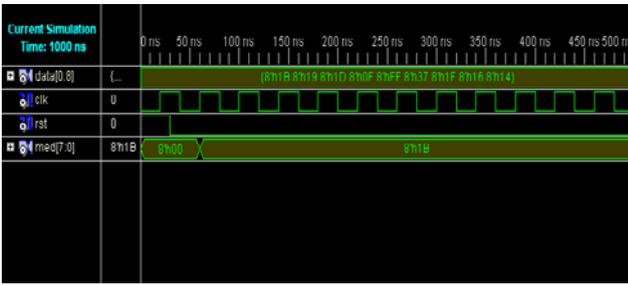


Figure 2. Example & result for case 1

If all the elements of window are '0'

If the selected window contains salt/pepper noise as processing pixel (i.e., 0 pixel value) and & also surrounding all elements has same value then processing element is an information instead of noise as there is high correlation between neighboring pixels so pixel value should keep as it was. Following figure shows simulation result

$$\begin{pmatrix} 0 & 0 & 0 \\ 0 & < 00 > & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

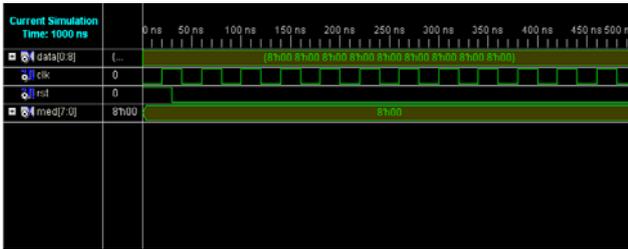


Figure 3. Example & result for case 2

If all the elements of window are '255'

If the selected window contains salt/pepper noise as processing pixel (i.e., 255 pixel value) and & also surrounding all elements has same value then processing element is an information instead of noise as there is high co-relation between neighboring pixels so pixel value should keep as it was. Following figure shows simulation result

$$\begin{pmatrix} FF & FF & FF \\ FF & < FF > & FF \\ FF & FF & FF \end{pmatrix}$$

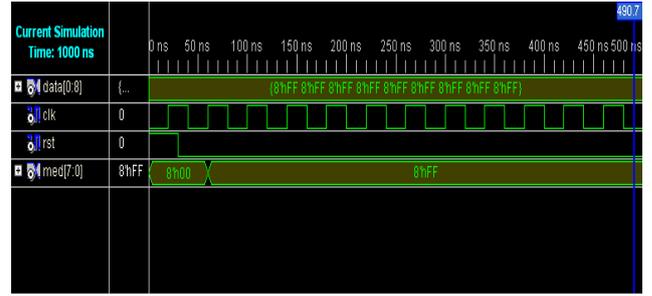


Figure 4. Example & result for case 3

If more than 50% of the window elements are faulty

If the selected window contains salt/pepper noise as processing pixel (i.e., 255 pixel value) and & also more than 50% of the surrounding elements has faulty intensities then replace the processing element by nearest neighboring information pixel value. Following example illustrates the case.

$$\begin{pmatrix} 1B & FF & FF \\ FF & < FF > & FF \\ 05 & 19 & 33 \end{pmatrix}$$

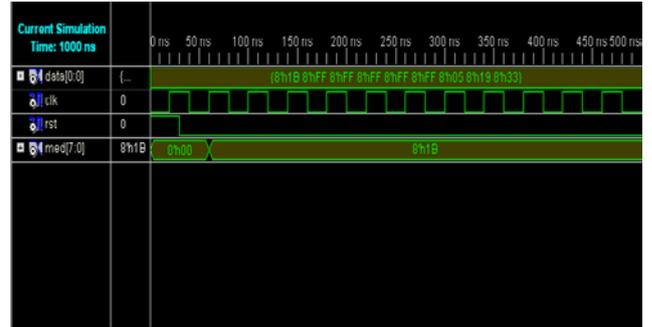


Figure 6. Example & result for case 4

If window elements are 0 or 255

If the selected window contains salt/pepper noise as processing pixel & other window elements are 0 or 255 then take mean of all window elements & replace the processing pixel by mean.

$$\begin{pmatrix} 0 & 0 & FF \\ FF & < 0 > & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

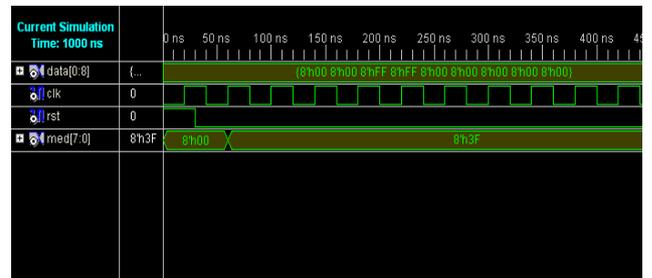


Figure 7. Example & result for case 5

Simulation result shows that median of a window is available at immediately next rising clock edge after reset goes low. This shows time requirement for the calculation of median of a window. Thus maximum time required for calculation of median for a window is one clock cycle. For 50 MHz clock frequency, this time requirement become

20 ns. Time required to scan & filter Image of 60 X 125 pixels is = 20 X 7500 = 150 ms. On general purpose processor, by experiment, same operation takes approx. 1.2 s. Thus FPGA implementation is more beneficial & approaches to a real time.

Following figure 8 shows visual results taken from FPGA.

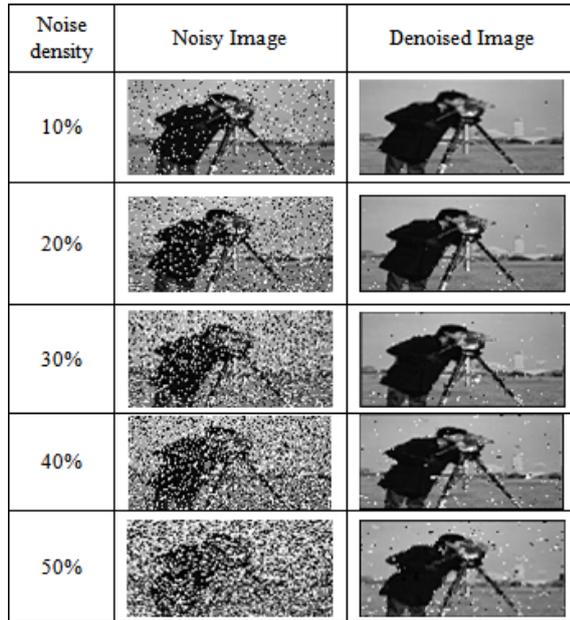


Figure 8. visual results from FPGA

The performance of above algorithm is tested for various levels of noise corruption. Each time the test image is corrupted by salt and pepper noise of different

density ranging from 10 to 90 with an increment of 10 will be applied to the filter. In addition to the visual quality, the performance of the proposed algorithm is quantitatively measured by the following parameters such as peak signal-to-noise ratio (PSNR) & Mean square error (MSE).

$$PSNR = 10 \log_{10} \left(\frac{255^2}{MSE} \right)$$

$$MSE = \frac{\sum_i \sum_j (Y(i, j) - \hat{Y}(i, j))^2}{M \times N} \quad [8]$$

where MSE stands for mean square error, Y represents the original image, \hat{Y} denotes the denoised image and M x N is the size of the image. Filter is implemented in MATLAB 7.8 and filtering window used for experiments is of size 3x3.

Table 1 shows simulation results for different images of same size. From this table it is clear that values of PSNR decreases with increase in noise density. Table 2 compares hardware & simulation results of PSNR for different image resolution. Values shows that PSNR for Hardware & software are equivalent. For the image of size 128 X 128 & 50 X 50, the value of PSNR decreases largely as increase in noise density than for image of size 60 X 125. For large noise densities hardware implementation gives better results of PSNR than MATLAB simulation.

Figure 9 & Figure 10 shows device utilization for different image resolution. Thus resource utilization for proposed algorithm is independent of image size.

Table 1. shows values of PSNR for different images of size 60 X 125

Image	Noise Density								
	10%	20%	30%	40%	50%	60%	70%	80%	90%
Cameraman	42.69	39.42	37.40	36.74	34.84	33.86	32.43	30.43	27.91
Rice	40.16	36.88	35.33	34.09	32.96	31.81	30.60	29.17	27.02
Eight	44.60	41.87	39.25	38.49	36.87	34.99	31.18	30.69	27.61

Table 2. Comparison of PSNR for simulation & Hardware Results for Image 'Cameraman'

Noise Density(%)	FPGA Results			MATLAB Simulation Results		
	Image Size 128 X 128	Image Size 60 X 125	Image Size 50 X 50	Image Size 128 X 128	Image Size 60 X 125	Image Size 50 X 50
10	43.01	39.83	33.74	43.59	42.69	42.95
20	40.33	39.08	33.38	40.47	39.42	38.83
30	38.57	38.57	33.10	38.23	37.40	36.52
40	37.10	37.98	32.45	36.63	36.74	35.43
50	35.86	37.48	32.14	35.36	34.84	34.35
60	34.62	36.49	31.76	34.19	33.86	32.88
70	33.11	35.54	31.04	32.62	32.43	31.74
80	31.17	34.09	30.24	30.64	30.43	30.33
90	29.36	32.19	28.75	27.98	27.91	27.77

Device Utilization Summary				
Logic Utilization	Used	Available	Utilization	
Number of Slice Flip Flops	1,306	9,312	14%	
Number of 4 input LUTs	5,101	9,312	54%	
Logic Distribution				
Number of occupied Slices	2,724	4,656	58%	
Number of Slices containing only related logic	2,724	2,724	100%	
Number of Slices containing unrelated logic	0	2,724	0%	
Total Number of 4 input LUTs	5,271	9,312	56%	
Number used as logic	5,101			
Number used as a route-thru	170			
Number of bonded IOBs	5	232	2%	
Number of RAMB16s	16	20	80%	
Number of BUFGMUXs	2	24	8%	

Figure 9. Resource Utilization for Image of size 128 X 128

Device Utilization Summary				
Logic Utilization	Used	Available	Utilization	Note(s)
Number of Slice Flip Flops	1,297	9,312	13%	
Number of 4 input LUTs	5,079	9,312	54%	
Logic Distribution				
Number of occupied Slices	2,704	4,656	58%	
Number of Slices containing only related logic	2,704	2,704	100%	
Number of Slices containing unrelated logic	0	2,704	0%	
Total Number of 4 input LUTs	5,251	9,312	56%	
Number used as logic	5,079			
Number used as a route-thru	172			
Number of bonded IOBs	5	232	2%	
Number of RAMB16s	8	20	40%	
Number of BUFGMUXs	2	24	8%	

Figure 10. Resource Utilization for Image of size 60 X 125

7. Conclusion

A new algorithm is proposed which gives better performance in terms of PSNR. The performance of the algorithm has been tested at low, medium and high noise densities for gray-scale images on FPGA XC3S500E with clock frequency 50 MHz. The implementation of RAM, UART & proposed median filter takes 58% of chip resources. The maximum time required for the filtered image to send from FPGA to MATLAB through UART is 6.9 seconds. Even at high noise density levels the algorithm gives better results in comparison with other existing algorithms. Both visual and quantitative results are demonstrated.

References

- [1] Dagu Venkateshwar Rao, Shruti Patil, Naveen Anne Babu and V Muthukumar "Implementation and Evaluation of Image Processing Algorithms on Reconfigurable Architecture using C-based Hardware Descriptive Languages" International Journal of Theoretical and Applied Computer Sciences Volume 1 Number 1 (2006) pp. 9-34.
- [2] H. Hwang and R. A. Haddad, "Adaptive median filter: New algorithms and results," *IEEE Trans. Image Process.*, vol. 4, no. 4, pp. 499-502, Apr. 1995.

- [3] S. Zhang and M. A. Karim, "A new impulse detector for switching median filters," *IEEE Signal Process. Lett.*, vol. 9, no. 11, pp. 360-363, Nov. 2002.
- [4] P. E. Ng and K. K. Ma, "A switching median filter with boundary discriminative noise detection for extremely corrupted images," *IEEE Trans. Image Process.*, vol. 15, no. 6, pp. 1506-1516, Jun. 2006.
- [5] W. Zhou and Z. David, "Progressive switching median filter for the removal of impulse noise from highly corrupted images," *IEEE Trans On Circuits and Systems: Analog and Digital Signal Processing*, vol. 46, no. 1, pp. 78-80, 1999.
- [6] D. R. K. Brownrigg, "The weighted median filter," *Commun. ACM*, vol. 27, no. 8, pp. 807-818, 1984.
- [7] S. Marshall, "New direct design method for weighted order statistic filters," *VISP*, vol. 151, no. 1, pp. 1-8, February 2004.
- [8] M. C Hanumantharaju, M. Ravishankar, D. R Rameshbabu and S. B Satish "An Efficient VLSI Architecture for Adaptive Rank Order Filter for Image Noise Removal" International Journal of Information and Electronics Engineering, Vol. 1, No. 1, July 2011.
- [9] Tarek M. Bittibssi, Gouda I. Salama, Yehia Z. Mehaseb and Adel E. Henawy "Image Enhancement Algorithms using FPGA" International Journal of Computer Science & Communication Networks, Vol. 2 (4), 536-542.