International Journal of Scientific Engineering and Technology Research

ISSN 2319-8885
Vol.04,Issue.22,
July-2015,
Pages:4170-4177

www.ijsetr.com

A Novel Impulse Noise Removal Algorithm for High Speed Multimedia Applications

SHERIN NITASHA[1], NIMMAGADDA RAVALI[2]

[1]PG Scholar, Dept of ECE, Sridevi Women's Engineering College, JNTU, Hyderabad, Telangana, India.
[2]Asst Prof, Dept of ECE, Sridevi Women's Engineering College, JNTU, Hyderabad, Telangana, India.

Abstract: Images are often degraded by noise. Noise can occur during image capture, transmission etc. Noise removal is an important task in image processing. Efficient VLSI implementation is presented in this paper in order to remove random valued impulse noise. It employs decision based impulse detector to detect noisy pixels and an edge preserving filter to reconstruct the intensity values of noisy pixels . Pixels that are detected as noisy are filtered, other remain unchanged. The impulse detector consists of three modules- isolation module, fringe module and similarity module. Since these modules are operating in parallel , this technique is faster compared to the previous low complexity methods. The main objective of this paper is to reduce the impulse noise with less computation complexity while achieving high speed operation.

Keywords: Image Denoising, Impulse Noise, Impulse Detector, VLSI, Edge Filter, Architecture.

I. INTRODUCTION

Digital image processing is promising area of research in the fields of electronics and communication engineering ,consumer and entertainment electronics, control and instrumentation biomedical instrumentation, remote sensing, robotics etc. So for a meaningful and useful processing , the image must be free from noise. When an image gets corrupted with noise during the process of acquisition, transmission, storage and retrieval, it becomes necessary to suppress the noise effectively without distorting the edges and the fine details in the image so that the filtered image becomes more useful for display and/or further processing. According to the distribution of noisy pixel values, impulse noise can be classified into two categories: fixed valued impulse noise and random-valued impulse noise .The former is also known as salt-and-pepper noise because the pixel value of a noisy pixel is either minimum or maximum value in gray-scale images. The values of noisy pixels corrupted by random-valued impulse noise are uniformly distributed in the range of [0,255] for gray-scale images.

In this paper, the denoising method consists of an impulse detector and edge preserving filter. For an input grey scale image, the impulse detector detects whether the image is corrupted by noise or not. If noise is present, edge preserving filter is used to reconstruct the image. The impulse detector consists of three modules. They are isolation module, fringe module and similarity module. These modules are used for making decisions based on whether noise is present or not. Here we consider a 3-3 window at a time for noise detection. Only the pixels that are corrupted by noise are used for filtering, others remain unchanged. The three modules for making decisions are operated in parallel. As a result, this method is fast compared to the previous methods in which the modules are operated sequentially. Also in the previous filters like mean filters and median filters , the noise free pixels are also affected by filtering. As a result the image will be blurred. But in the case of edge preserving filter, only the noisy pixels are used for filtering. Other pixels that are noise free remain unchanged. That is it filters the noisy pixels rather than the whole pixels of an image to avoid causing the damage on noise-free pixels. Also it preserves minute structures like edges or corners.

II. THE DECISION BASED DENOISING METHOD

This method consists two components – impulse detector and edge preserving image filter. The detector determines whether the image is corrupted by noise. If noise is present, then an edge preserving image filter is used to reconstruct the intensity values of noisy pixels.

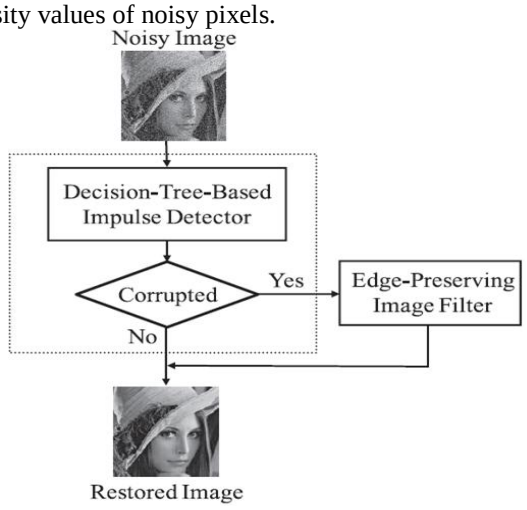


Fig1. Dataflow of Denoising Method.

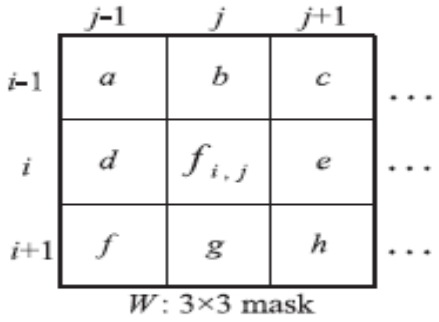


Fig2. A 3x3 Mask Centred on P(I,J).

The noise considered in this paper is random-valued impulse noise with uniform distribution. Here, we adopt a 3 - 3 mask for image denoising. Assume the pixel to be denoised is located at coordinate (i,j) and denoted as p(i,j), and its luminance value is named as f(i,j). According to the input sequence of image denoising process, we can divide other eight pixel values into two sets: WTopHalf and WBottomHalf . They are given as

$$WTopHalf = (a, b, c, d) \quad (1)$$

$$WBottomHalf = (e, f, g, h) \quad (2)$$

Denoising method consists of two components:

1. Decision Based Impulse Detector
2. Edge-Preserving image filter

1.Decision Based Impulse Detector

In order to determine whether p(i,j)is a noisy pixel, the correlations among p(i,j)and its neighboring pixels are considered classify them into several ways

- Observing the degree of isolation at current pixel,
- Determining whether the current pixel is on a fringe
- Comparing the similarity between current pixel and its neighboring pixels.

Therefore, here we design three modules

1. Isolation Module (IM).
2. Fringe Module (FM).
3. Similarity Module (SM).

Based on the decisions of these modules we can determine the status of p(i,j)by using the different equations in different modules. We use Isolation Module to decide whether the pixel value is in a smooth region. If the result is negative, we conclude that the current pixel belongs to noisy free. Otherwise, if the result is affirmative, it means that the current pixel might be a noisy pixel or just situated on an edge. The Fringe Module is used to confirm the result. If the current pixel is situated on an edge, the result of fringe module will be negative (noisy free); otherwise, it will be noisy. If Isolation Module and fringe Module cannot determine whether current pixel belongs to noise free, then the decision of similarity module is used to decide the result. It compares the correspondence between current pixel and its neighboring pixels. If the result is positive, p(i,j)is a noisy pixel; otherwise, it is noise free. The luminance values in mask W located in a noisy-free area might be close.

2. Isolation Module

The pixel values in a even region should be close or locally slightly varying, as shown below.

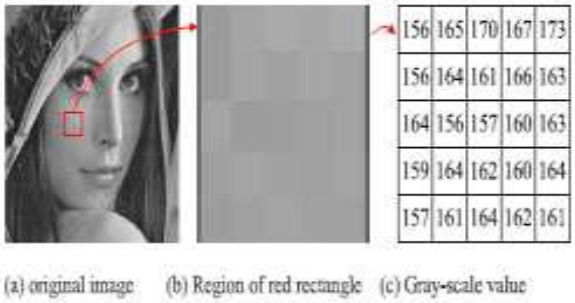


Fig3. A Smooth region in Lena

The differences between its neighboring pixel values are small. If there are noisy values, edges, or blocks in this region, the allocation of the values is different, as shown in the figure. Therefore, we determine whether current pixel is an isolation point by observing the smoothness of its surrounding pixels. The pixels with shadow suffering from noise have low similarity with the neighboring pixels and the so-called separation point.

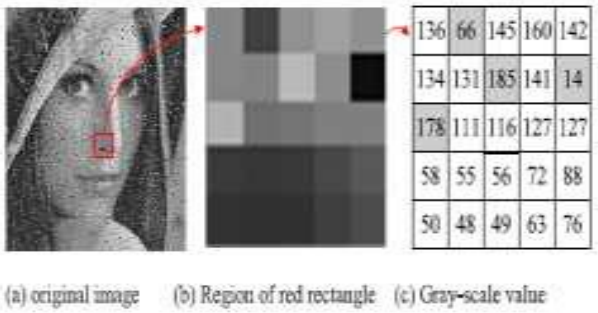


Fig4. The Difference between noisy & neighboring pixels in Lena.

The pixels with shadow suffering from noise have low similarity with the neighboring pixels and the so-called isolation point. The difference between it and its neighboring pixel value is large. According to the over concepts, we first detect the most and minimum luminance values in WTopHalf named as TopHalf_max,, TopHalf_ min, and compute the difference between them, named as TopHalf_diff.. For WBottomHalf , we apply the same idea to obtain BottomHalf_diff. Then the two values are compared with a threshold Th_IMa to decide whether the surrounding region belongs to a smooth area. The equations are as

$$TopHalf\_diff = TopHalf\_max - TopHalf\_min \quad (3)$$

$$BottomHalf\_diff = BottomHalf\_max - BottomHalf\_min \quad (4)$$

$$DecisionI = \begin{cases} true, & \text{if } (TopHalf\_diff \geq Th\_IM_a) \\ & \text{or } (BottomHalf\_diff \geq Th\_IM_a) \\ false, & \text{otherwise.} \end{cases} \quad (5)$$

Now taking p(i,j) into consideration. Two principles must be calculated.

1. The difference between f(i,j) and TopHalf_max.
2. The difference between f(i,j)and TopHalf_minqw.

After the subtraction, a threshold Th_IMb is used to compare these two differences. The identical method is used for WBottomHalf . The equations are as

$$IM\_TopHalf = \begin{cases} true, & \text{if } (|f_{i,j} - TopHalf\_max| \geq Th\_IM_b) \\ & or\ (|f_{i,j} - TopHalf\_min| \geq Th\_IM_b) \\ false, & \text{otherwise.} \end{cases} \quad (6)$$

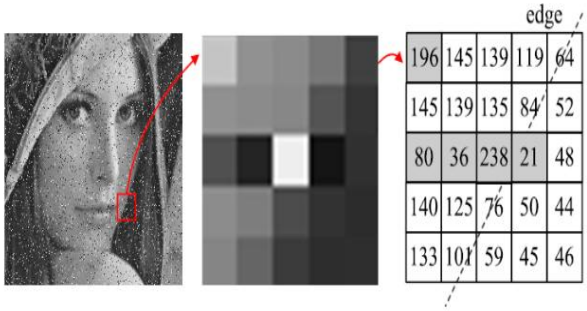
$$IM\_BottomHalf = \begin{cases} true, & \text{if } (|f_{i,j} - BottomHalf\_max| \geq Th\_IM_b) \\ & or\ (|f_{i,j} - BottomHalf\_min| \geq Th\_IM_b) \\ false, & \text{otherwise.} \end{cases} \quad (7)$$

$$DecisionII = \begin{cases} true, & \text{if } (IM\_TopHalf = true) \\ & \text{or } (IM\_BottomHalf = true) \\ false, & \text{otherwise.} \end{cases} \quad (8)$$

From this decisions we can make a temporary decision whether p(i,j)belongs to a suspected noisy pixel or is noisy free.

2. Fringe Module

If p(i,j)has a great discrepancy with neighboring pixels, it might be a noisy pixel or just situated on an edge, as shown in Figure.



(a) original image (b) Region of red rectangle (c) Gray-scale value

Fig5. The Edge region in lena.

To end that a pixel is noisy or situated on an edge is difficult we define four directions, from E1 to E4, as shown in Figure.

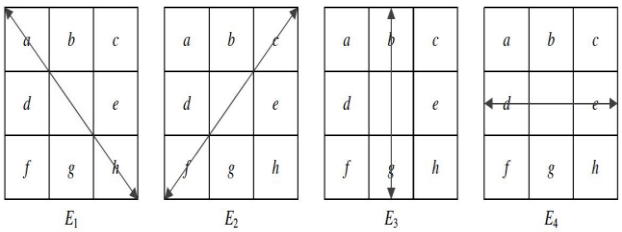


Fig6. Four directions in denoising method.

We take direction E1 for case. By calculating the absolute difference between f(i,j)and the other two pixel values along the same way, respectively, we can determine whether there is an edge or not. If above two module is not enough to find the noisy pixel, then we have to proceed similarity module.

$$FM\_E_1 = \begin{cases} false, & \text{if } (|a - f_{i,j}| \geq Th\_FM_a) \\ & \text{or } (|h - f_{i,j}| \geq Th\_FM_a) \\ & \text{or } (|a - h| \geq Th\_FM_b) \\ true, & \text{otherwise.} \end{cases} \quad (9)$$

$$FM\_E_2 = \begin{cases} false, & \text{if } (|c - f_{i,j}| \geq Th\_FM_a) \\ & \text{or } (|f - f_{i,j}| \geq Th\_FM_a) \\ & \text{or } (|c - f| \geq Th\_FM_b) \\ true, & \text{otherwise.} \end{cases} \quad (10)$$

$$FM\_E_3 = \begin{cases} false, & \text{if } (|b - f_{i,j}| \geq Th\_FM_a) \\ & \text{or } (|g - f_{i,j}| \geq Th\_FM_a) \\ & \text{or } (|b - g| \geq Th\_FM_b) \\ true, & \text{otherwise.} \end{cases} \quad (11)$$

$$FM\_E_4 = \begin{cases} false, & \text{if } (|d - f_{i,j}| \geq Th\_FM_a) \\ & \text{or } (|e - f_{i,j}| \geq Th\_FM_a) \\ & \text{or } (|d - e| \geq Th\_FM_b) \\ true, & \text{otherwise.} \end{cases} \quad (12)$$

$$Decision\ III = \begin{cases} false, & \text{if } (FM\_E_1) \text{ or } (FM\_E_2) \\ & \text{or } (FM\_E_3) \text{ or } (FM\_E_4) \\ \text{true}, & \text{otherwise.} \end{cases} \quad (13)$$

3. Similarity Module

The last module is likeness module. The luminance values in mask W located in a noisy-free area might be close. The median is always located in the center of the variational sequence, while the impulse is usually located near one of its trimmings. Hence, if there are extreme big orminute values that imply the possibility ofnoisy signals. Here, we sort nine values in ascending order and obtain the fourth, fifth, and sixth values which are close to the median in mask W. The fourth, fifth, and sixth values are represented as 4thinWi,j, MedianInWi,j, and 6thinWi,j. We define Maxi,j and Mini,j as

$$Max_{i,j} = 6_{th}inW_{i,j} + Th\_SM_a,$$
$$Min_{i,j} = 4_{th}inW_{i,j} - Th\_SM_a. \quad (14)$$

Maxi,j and Mini,j are used to determine the status of pixel p(i,j). However, in order to make the decision more precisely, we do some modifications as

$$N_{\max} = \begin{cases} Max_{i,j}, & \text{if } (Max_{i,j} \leq MedianInW_{i,j} + Th\_SM_b) \\ MedianInW_{i,j} + Th\_SM_b, & \text{otherwise.} \end{cases}$$

$$N_{\min} = \begin{cases} Min_{i,j}, & \text{if } (Min_{i,j} \geq MedianInW_{i,j} - Th\_SM_b) \\ MedianInW_{i,j} - Th\_SM_b, & \text{otherwise.} \end{cases} \quad (15)$$

Finally, if f(i,j)is not between Nmax and Nmin, we conclude thatp(i,j) is a noise pixel. Edge-preserving image filter will be used to build the reconstructed value. Otherwise, the original value f(i,j)will be the output. The equation is as:

$$Decision\ IV = \begin{cases} true, & \text{if } (f_{i,j} \geq N_{max}) \text{ or } (f_{i,j} \leq N_{min}) \\ false, & \text{otherwise.} \end{cases} \quad (16)$$

Obviously, the threshold affects the quality of denoised images of the proposed method. A more appropriate threshold contributes to achieve a better detection result.The thresholds Th_IMa, Th_IMb, Th_FMa, Th_FMb, Th_SMa, and Th_SMb are all predefined values and set as 20, 25, 40, 80, 15, and 60, respectively.

B.Edge-Preserving Image Filter

To locate the edge existing in the current W, a simple edge preserving technique this can be adopted. The dataflow of our edge-preserving image filter is as shown below. Here, we consider eight directional differences, from D1 to D8, to reconstruct the noisy pixel value, as shown in Figure. . Only those composed of noise-free pixels are taken into account to avoid possible misdetection. Directions passing through the suspected pixels are dis-carded to reduce misdetection.

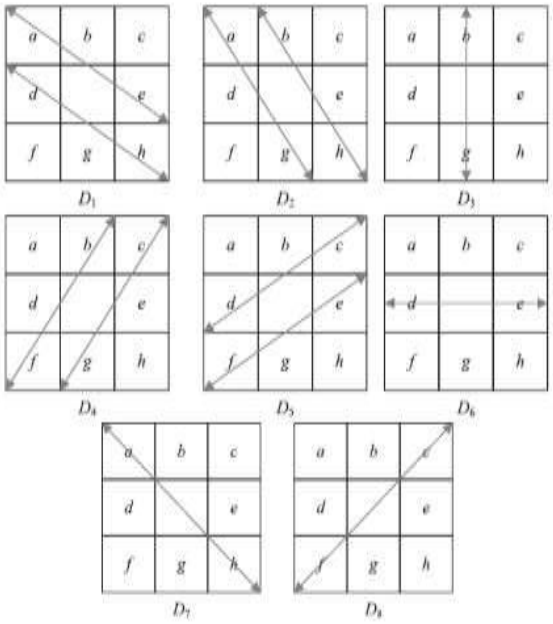


Fig7. Eight directional differences of Denoisng method

Only those composed of noise-free pixels are taken into account to avoid possible misdetection. Directions passing through the suspected pixels are discarded to reduce misdetection. Therefore, we use Max(i;j) and Min(i;j), defined in similarity module, to determine whether the values of d, e, f, g, and h are likely corrupted, respectively. If the pixel is likely being corrupted by noise, we don't consider the direction including the suspected pixel. In the second block, if d, e, f, g, and h are all suspected to be noisy pixels, and no edge can be processed, so f(i,j) the estimated value of p(i,j) is equal to the weighted average of luminance values of three previously denoised pixels and calculated as (a+b*2+c)/4.. In other conditions, the edge filter calculates the directional differences of the chosen directions and locates the smallest one (Dmin) among the m in the third block. The equations are as follows

$$\begin{aligned} D_1 &= |d-h| + |a-e| \\ D_2 &= |a-g| + |b-h| \\ D_3 &= |b-g| \times 2 \\ D_4 &= |b-f| + |c-g| \\ D_5 &= |c-d| + |e-f| \\ D_6 &= |d-e| \times 2 \\ D_7 &= |a-h| \times 2 \\ D_8 &= |c-f| \times 2, \end{aligned} \quad (17)$$

$$\hat{f}_{i,j} = \begin{cases} (a+d+e+h)/4, & \text{if } D_{\min} = D_1, \\ (a+b+g+h)/4, & \text{if } D_{\min} = D_2, \\ (b+g)/2, & \text{if } D_{\min} = D_3, \\ (b+c+f+g)/4, & \text{if } D_{\min} = D_4, \\ (c+d+e+f)/4, & \text{if } D_{\min} = D_5, \\ (d+e)/2, & \text{if } D_{\min} = D_6, \\ (a+h)/2, & \text{if } D_{\min} = D_7, \\ (c+f)/2, & \text{if } D_{\min} = D_8. \end{cases} \quad (18)$$

The smallest directional difference implies that it has the strongest spatial relation with p(i,j) and probably there exists an edge in its direction. Hence, the mean of luminance values of the pixels which possess the smallest directional difference is treated as f(i,j). After f(i,j)is determined, a tuning skill is used to filter the bias edge. If f(i,j)obtain the correct edge, it will situate at the median of b, d, e, and g because of the spatial relation and the characteristic of edge preserving. Otherwise, the values the values of f(i,j) will be replaced by the median of four neighboring pixels (b, d, e, and g). We can express $\bar{f}_{i,j}$ as

$$\bar{f}_{i,j} = \text{Median}(\hat{f}_{i,j}, b, d, e, g). \quad (19)$$

The three modules isolation , fringe and similarity modules are operated in parallel in order to reduce the time delay for making decisions .As a result, it is fast compared to the previous sequential operation of modules. Hence the proposed method is suitable for high speed multimedia applications.

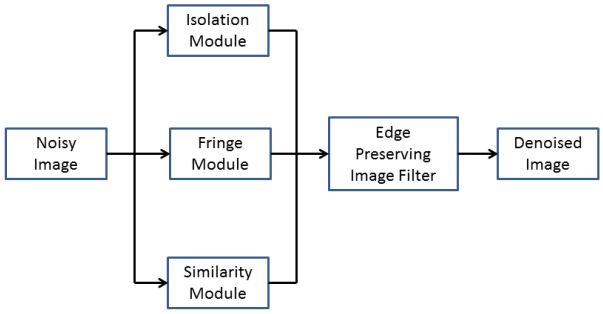


Fig8. Block diagram for parallel operation of modules.

III. VLSI IMPLEMENTATION OF DENOISING METHOD

The architecture adopts an adaptive technology and consists of five main blocks: line buffer, register bank, decision tree- based impulse detector, edge-preserving image filter and controller.

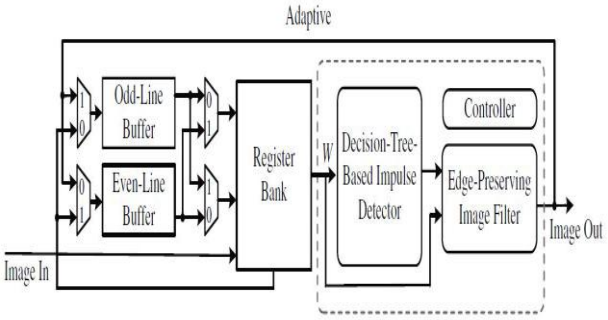


Fig9. Block Diagram of VLSI Architecture of DM

A. Adaptive Technology

As an advanced method compared with standard median filtering, the Adaptive edge Filter performs spatial processing to preserve detail and smooth non-impulsive noise. A prime benefit to this adaptive approach to median filtering is that repeated applications of this Adaptive Median Filter do not erode away edges or other small structure in the image.The reconstructed pixels are adaptively written or stored into theline buffers. The current pixel to be denoised is located at coordinate (i, j) . The adaptive points located at coordinate (i-1, j-1), (i-1, j), and (i-1, j+1) are already denoised at the previous denoising process. Although only three adaptive points (less than half) of nine input points are available, the reconstructed value is significantly affected by the adaptive points, since each adaptive point is written back to influence the next point continuously.

B. Line Buffer (Ping-Pong Buffer)

Since we adopts a 3 - 3 mask, so three scanning lines are needed. If p(i,j)are processed, three pixels from row(i-1), row(i), and row(i+1) , are needed to perform the denoising process. Here, we use the concept of ping-pong arrangement. With the help of four crossover multiplexers we realize three scanning lines with two line buffers. Odd-Line Buffer and Even-Line Buffer are designed to store the pixels at odd and even rows, respectively.

C. Register Bank

The register bank, consisting of nine registers, is used to store the 3 - 3 pixel values of the current mask W. Fig. 10 shows its architecture where each three registers are connected serially in a chain to provide three pixel values of a row in W, and the keeps the luminance value f(i,j) of the current pixel to be denoised. Obviously, the denoising process for p(i,j)doesn't start until f (i+1;j+1) enters from the input device. The nine values stored in RB are then used simultaneously by subsequent data detector and noise filter for denoising. Once the denoising process for p(i,j) is completed, the reconstructed pixel value f(i,j)generated by the edge-preserving filter is outputted and written into the line buffer storing row(i) to replace f(i,j). When the denoising process shifts from p(i,j)to pi;jþ1, only three new values f(i-1,j+2); f (i,j+2) and f (i+1,j+2) are needed to be read into RB (Reg2, Reg5, and Reg8, respectively) and other six pixel values are shifted to each one's proper register. At the same time, the previous input value from the input device, f(i+1,j+1) , is written back to the line buffer storing row (i-1) for subsequent denoising process from the input device, , is written back to the line buffer storing row(i-1) for subsequent denoising process. The selection signals of the four multiplexers are all set to 1 or 0 for denoising the odd or the even rows, respectively.

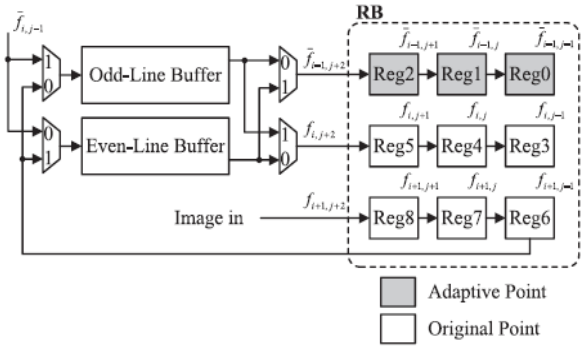


fig10. Architecture of register bank

D. Decision-Based Impulse Detector

The decision-tree-based impulse detector is composed of three modules

1. Isolation module
2. Fringe module
3. Similarity module

1. Isolation Module

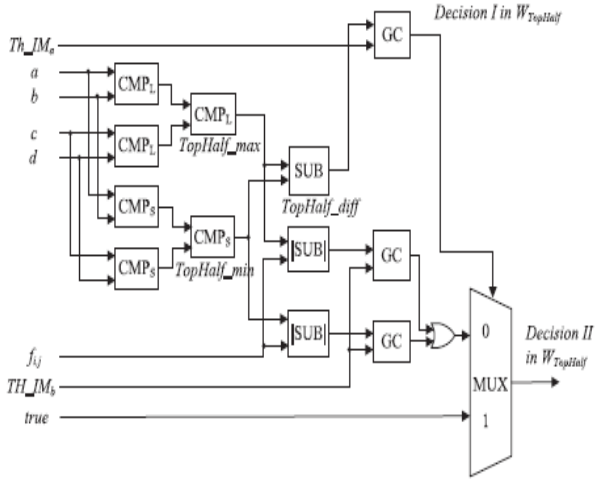


Fig11: Architecture of Isolation Module.

Fig.11 shows the architecture of IM (we take WTopHalf for example). The comparator CMPL is used to output the larger value from the two input values while the comparator CMPS is used to output the smaller value from the two input values. The first two-level comparators are used to find TopHalf_max and TopHalf_min. The SUB unit is used to output the difference which is subtracted the lower input (TopHalf_min) from the upper one (TopHalf_max), and the SUB unit is used to output the absolute value of difference of two inputs. The GC is the greater comparator that will output logic 1 if the upper input value is greater than the lower one. The OR gate is employed to generate the binary result for IM_TopHalf. Finally, if the result of Decision II is positive, p(i,j)might be a noisy pixel or situate on an edge.

2. Fringe Module

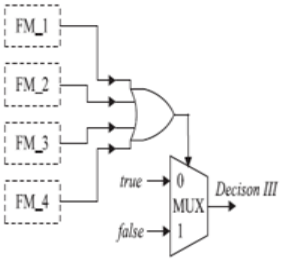


Fig12. Architecture of FM

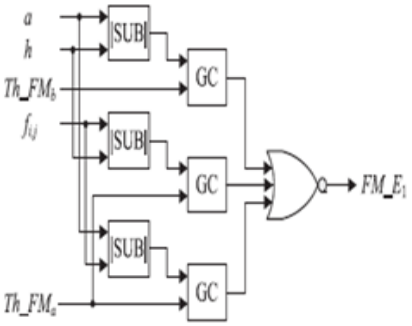


Fig13. Architecture of Fm_1 Module.

Fig.12 shows the architecture of FM. The FM is composed of four small modules, from FM_1 to FM_4, and each of them is used to determine its direction. Fig.13 is a detailed implementation of FM_1. Since E1 is the direction from a to h, the relation between a, h, and f(i,j)must be referenced. The three SUB units are used to determine the absolute differences between them. The NOR gate is used to generate the result of FM E1. If the result is positive, we consider that f(i,j)is on the edge E1 and regard it as noise free.

E. Similarity Module

If isolation and fringe can't determine whether f(i,j) belongs to a noise free value or not, SM is used to confirm the result. Fig.14 shows the architecture that is designed to accelerate the sorting speed to obtain the fourth value in mask W. The detailed implementation of module M0 is shown in Fig. 5.8. If a is greater than b, C01 is set to 1; otherwise, C01 is set to 0. The eight GC units are used to determine the values from C01 to C08. After comparing, a combined unit is used to combine the results of each comparator to obtain a number between 0 and 8. The number indicates the order of value in mask W. If a is the smallest value in mask W, the output of the M0 module is 0; if a is the biggest value in mask W, the output is 8. The architectures of other modules (M1 to M8) are almost the same as M0, with only little difference

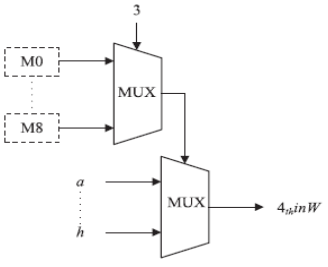


Fig14. Architecture of sorting.

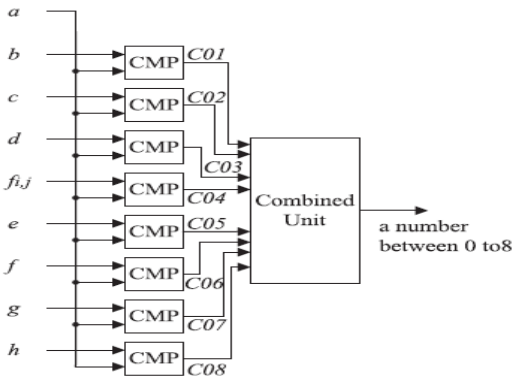


Fig15. Architecture of M0 Module.

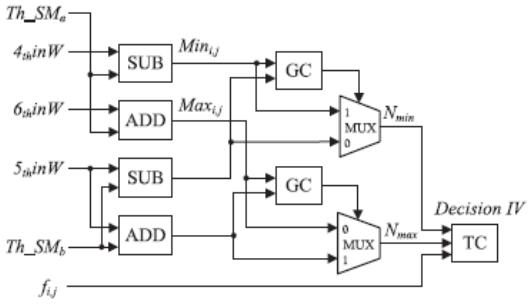


Fig16. Architecture of SM.

Fig.16 shows the architecture of SM after we obtain the fourth, fifth, and sixth values in mask W. The SUB and ADD units are used to calculate the value of Maxi;j and Mini;j. The Two GC and MUX units are used to determine the Nmax and Nmin. The TC unit is a triple input comparator which can output the logic 1 if the lowest input is not between the upper two inputs.

F. Edge-Preserving Image Filter

The Edge-Preserving Image Filter is composed of two modules, minED generator and average generator (AG). If noise is detected from any of the modules, edge preserving filter is used to reconstruct the intensity values of noisy pixels. Min ED generator is used to calculate the smallest directional difference and average generator is used to calculate the mean possess the smallest directional difference.

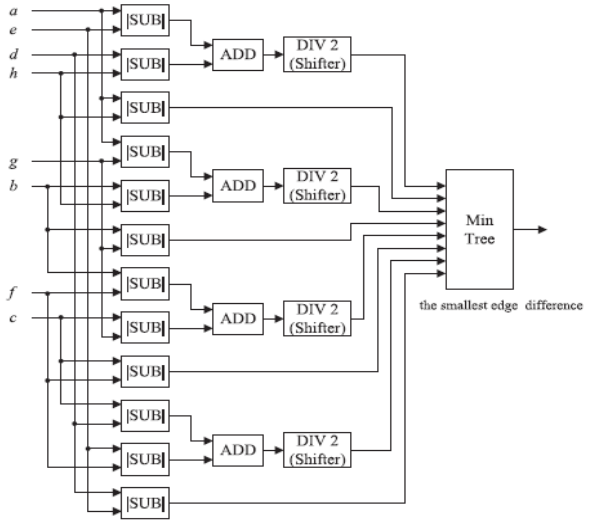


Fig17. Architecture of Mined Generator

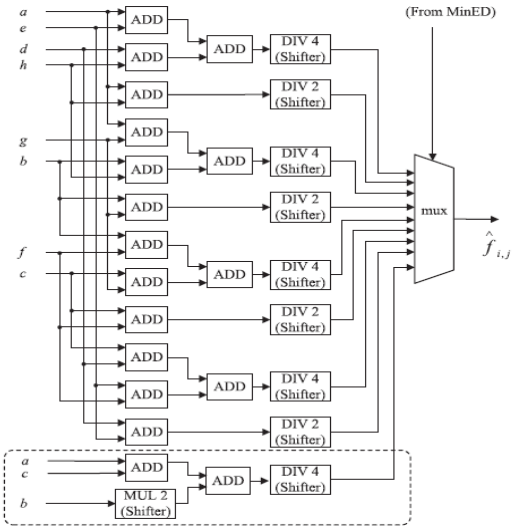


Fig18. Architecture Of Average Generator

Fig 17 shows the architecture of the minED generator which is used to determine the edge that has the smallest difference. Eight directional differences are calculated with twelve SUB, four ADD, and four shifter units. Then, the smallest one is determined by using the Min Tree unit. Min Tree is made up of a series of comparators. After that, the mean of luminance values of the pixels which possess the smallest directional difference Dmin can be obtained from the average generator. If p(i;j-1); p(i;j+1); p(i+1;j-1); p(i+1,j) and p(i+1;j+1) are all suspected to be noisy pixels, the final MUX will output a+b*2+c/4. Otherwise, the MUX will output the mean of the pixel values which possess Dmin. Some directional differences are determined according to four pixel values, so its reconstructive values also need four pixel values. . After above computations, we sort b, d, e, and g in order. The reconstructed value f(i,j) obtained from edge-preserving filter will be compared with the second and third values, named as SortFour2, SortFour3, and the final value f(i,j)is obtained from the equation as

$$\bar{f}_{i,j} = \begin{cases} SortFour2, & \text{if } (SortFour2 > \hat{f}_{i,j}) \\ SortFour3, & \text{if } (SortFour3 < \hat{f}_{i,j}) \\ \hat{f}_{i,j}, & \text{otherwise.} \end{cases} \quad (20)$$

G. Controller

Controller sends signals to control pipelining and timing statuses of the proposed circuits. It also sends control signals to schedule reading and writing statuses of the data that are stored in register bank or in line buffers. The realization of the controller is based on the concept of finite state machine (FSM). By the controller design, the proposed circuit can automatically receive stream-in data of original images and produce stream-out results of reconstructed image.

IV. RESULTS

The proposed denoising method is implemented in VHDL, synthesized using Xilinx ISE and simulated in modelsim. It is implemented on 256 x 256 8-bit gray scale images. The noisy input and denoised output viewed in MATLAB is shown below.

Fig19: Noisy and denoised image.

In sequential operation of the three modules isolation, fringe and similarity, five clock cycles are needed to produce output for a3 -3 window. The results are shown below.

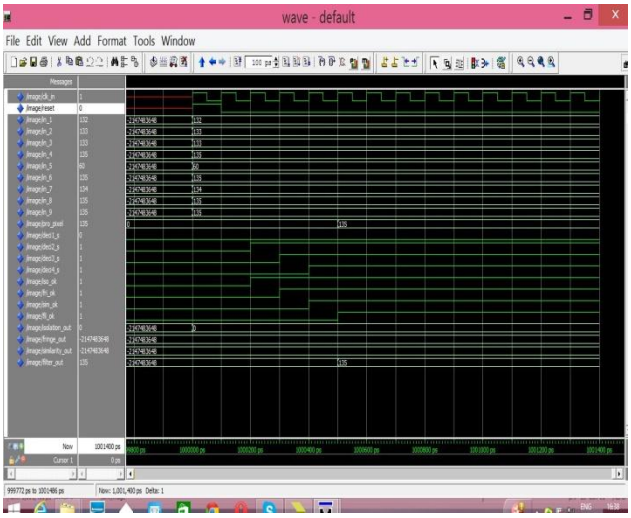


Fig20.Simulated results of a 3-3 window when modules are operated in series

In parallel operation of modules , the three modules are operated at the same time. As a result, only three clock cycles are required to produce output for a window. The results are shown below.

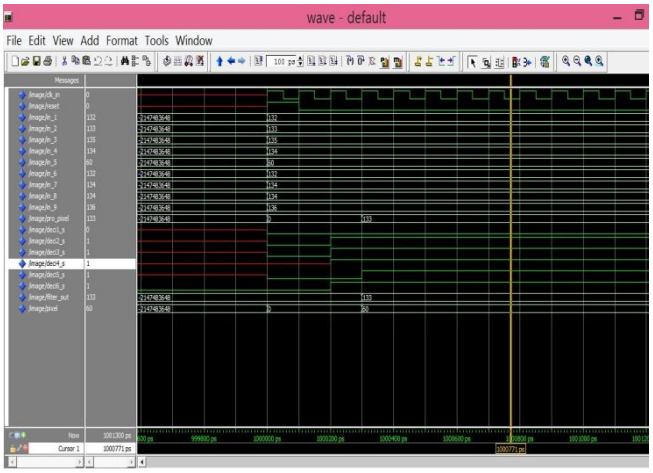


Fig21. Simulated results when modules are operated in parallel for a window.

PSNR of the noisy input and the denoised output are estimated. For noisy input, PSNR is found to be 12% and for the denoised output, PSNR is 24%.

V. CONCLUSION

A fast denoising method for the removal of random valued impulse noise is proposed in this paper. The approach used an impulse detector and an edge preserving filter. The impulse detector consists of three modules-isolation, fringe, and similarity, which are operated in parallel. Hence this method is fast compared to the previous methods using sequential operation of modules. The impulse detector detects whether the image is corrupted by noise or not. The edge preserving filter is found to be quite effective in eliminating impulse noise. The filtering operation is performing only on corrupted pixels, uncorrupted pixels are undisturbed. As a result the restored images can preserve perceptual details and edges in the image while effectively removing impulse noise.

VI. REFERENCES

[1] R.C. Gonzalez and R.E. Woods, Digital Image Processing. Pearson Education, 2007.

[2] W.K. Pratt, Digital Image Processing.

[3] H. Hwang and R.A. Haddad, "Adaptive Median Filters: New Algorithms and Results," IEEE Trans. Image Processing, vol. 4, no. 4, pp. 499-502, Apr. 1995.

[4] S. Zhang and M.A. Karim, "A New Impulse Detector for Switching Median Filter," IEEE Signal Processing Letters, vol. 9, no. 11, pp. 360-363, Nov. 2002.

[5] R.H. Chan, C.W. Ho, and M. Nikolova, "Salt-and-Pepper Noise Removal by Median-Type Noise Detectors and Detail-Preserving Regularization," IEEE Trans. Image Processing, vol. 14, no. 10, pp. 1479-1485, Oct. 2005.