International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 4 Issue 07, July-2015

# A High Performance Architecture for Elimination of Impulse Noise in Images

Sabeeh Ahammed M T
Dept. of Electronics,
Govt. Engineering College,
Kozhikode, India.

Jithesh C P
Dept. of Electronics,
Govt. Engineering College,
Kozhikode, India.

***Abstract*—Noise filtering and image enhancement are two active areas of research in signal processing. For the time of signal acquisition and transmission, impulse noise may corrupt the digital data. To overcome this, switching median filters are generally used which consist of impulse detection and noise filtering. In this paper, an efficient decision tree based denoising technique and a high performance architecture for FPGA prototyping is presented for the elimination of random valued impulse noise. To identify noise pixels, we propose a new tree based impulse noise detector. The proposed architecture make use of parallel computation by splitting the input image in to odd and even index pixel masks. An additional performance enhancement is achieved by introducing pipelining between processing stages. The design is implemented using XILINX ISE 14.1 for Virtex 5 ML507 FPGA platform which can operate at 106 MHz clock frequency. The proposed method outperforms the existing denoising techniques in terms of image quality as shown by the simulation results.**

***Keywords—Improved decision tree; Impulse noise; Image denoising***

## I. INTRODUCTION

There are many applications in image processing for areas such as face recognition, remote sensing, techniques for scanning and medical imaging. The images may get corrupted by impulse noise. This could happen during acquisition and transmission. So, efficient denoising technique plays an important role in the world of image processing. The distribution of noisy pixel intensities are different in many cases. They are broadly classified into the following, fixed valued impulse noise (salt and-pepper noise) and random valued impulse noise. In the case of random valued impulse noise, the distribution is uniform and because of the random nature, it is difficult to remove them. The focus of this work is to eliminate random valued impulse noise from the corrupted images. There are different methods available for this purpose [5], [6], [7],[8],[9]. Some methods make use of standard median filter [5] or its alterations [6], [7]. The approach here is to introduce a modification of both noisy and noise-free pixels, which blurs the image. The methods in [8], [9] use an efficient switching strategy so as to avoid the damage on the pixels that are noise free. The method of switching median filters work in two steps. First, impulse noise is detected and then noise is filtered. The detection is done by an impulse detector, and only the pixels which are detected are further processed. The decision tree is a powerful form of multivariable analysis [10]. A complex decision is divided into smaller simpler steps. There have been several decision tree based methods [4],[12]. A novel adaptive decision tree based noise detector is presented in [2]. Here, a decision tree based noise detector is followed by an edge preserving filter.

In this paper, we propose an improved decision tree based denoising method (IDTBDM) based on decision-trees for elimination of random-valued impulse noise. This method consists of an impulse noise detector based on improved decision tree. The experimental outcome ensures the better performance of the proposed technique, if we analyse the images with respect to the visual quality and quantitative evaluation when compared with the other denoising methods [5], [6],[7], [8], [9]. The requirement of an efficient architecture with high performance and low cost is very challenging in real-time applications. The architecture for proposed design is modelled using VHDL and implemented on Xilinx Virtex 5 ML507 FPGA. Parallel processing and pipelining are introduced in the implementation for performance enhancement. The hardware is cost effective as the design is implemented using low complexity computational blocks. Our design achieves a clock speed of 106 MHZ.

The rest of the paper is systematized as follows: Section II describes the proposed IDTBDM. Section III describes System architecture of IDTBDM. Section IV describes the implementation results. We conclude the paper in section IV.

## II. PROPOSED IDTBDM

In this paper, random valued impulse noise which is uniformly distributed is considered as in [5],[9] and [10]. We accept a 3x3 mask for image denoising. Let pixel which is to be denoised is at coordinate *(i, j).* The pixel is denoted as $p_{i,j}$ and its luminance value is noted as $f_{i,j}$ as shown in Fig. 1. In an image input sequence , we divide eight other adjacent pixel values into two groups: $W_{T\_Half}$ and $W_{B\_Half}$ . They are given as in equation (1) and (2).

$$W_{T\_Half} = \{a, b, c, d\} \quad (1)$$

| | j-1 | j | j+1 |
|---|---|---|---|
| i-1 | a | b | c |
| i | d | $f_{(i,j)}$ | e |
| i+1 | f | g | h |

Fig.1. A 3 x 3 mask centered on $p_{i,j}$

$$W_{B\_Half} = \{e, f, g, h\} \quad (2)$$

Our method termed as IDTBDM consists of two components: An improved impulse detector and an edge-preserving filter. The impulse detector decides $p_{i,j}$ is a noisy pixel or not, by using the improved decision tree and the relationship between pixel $p_{i,j}$ and its adjacent pixels. We use edge preserving image filter to generate the reconstructed value of noisy pixel, based on direction oriented. For noise free pixel, the value will be retained. The outline of the IDTBDM is depicted in Fig.2

### *A. Improved Decision Tree Based Impulse Detector*

In [2] contains three modules isolation module (IM), fringe module (FM), and similarity module (SM). We propose a more effective algorithm for isolation module than mentioned in [2]. Decision tree is made by considering three combined decision of these modules. The status of $p_{i,j}$ can be evaluated by using equations of different modules. we begin with isolation module to decide whether the pixel goes to noisy or noise free. If the decision is false, we confirm that the pixel is noise free. If not, the pixel might be a noisy pixel or an edge pixel.

To differentiate between edge and noise pixel, fringe module is used. Fringe module gives positive result, if the pixel is on an edge (noise free) or else it shows negative. The similarity module make the final decision wherein isolation and fringe module fails. Similarity module checks the likeness between each pixel and its neighbours. Accordingly we decide the pixel $p_{i,j}$ is a noisy pixel or not. Three modules are explained as below

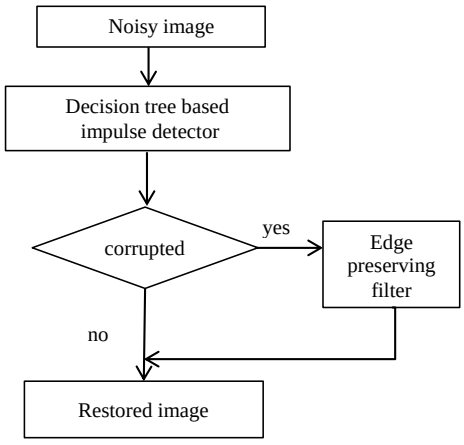


Fig.2. The dataflow of IDTBDM

*1*). Isolation module: If we consider a smooth region the pixel values present in that region shows only a gradual change or do not change at all. In that case, small differences exist between neighbouring pixel values. In a section, the values are distribute differently, if there exist noise or edges. The smoothness of the nearby pixels determine whether the current pixel is an isolation point or not. The pixels called as isolation point, if they have low similarity with the neighbouring pixels. Hence, difference between them is large. Following the above principles, firstly we identify the maximum and minimum intensity values in $W_{T\_Half}$ denoted as *T_H_max* and *T_H_min*, and compute the difference between them, denoted as *T_H_D*. To get *B_H_D* for $W_{B\_Half}$, we can use the similar idea mentioned above. In order to find out whether the nearby region belongs to a smooth area or not, the obtained differences are compared with a threshold *T_a* . The equations are as

$$T\_H\_D = T\_H\_\max - T\_H\_\min \quad (3)$$

$$B\_H\_D = B\_H\_\max - B\_H\_\min \quad (4)$$

$$Smooth = \begin{cases} true & if \quad (T\_H\_D \le T\_a) \\ & or \quad (B\_H\_D \le T\_a) \\ false & otherwise \end{cases} \quad (5)$$

If the nearby region does not belong to a smooth area, there also have a chance, that the considering pixel might be noisy. So next, we calculate mean of neighbouring pixel values named as *Mean_nbrs* and the difference between *Mean_nbrs* and $f_{i,j}$ . Then, if the difference is greater than threshold *T_b* , we decide the pixel may be noisy.

$$Mean\_nbrs = mean(a,b,c,d,e,f,g,h) \quad (6)$$

$$N\_Smooth = \begin{cases} true & if\left(\left| f_{i,j} - Mean\_nbrs \right| \ge T\_b\right) \\ false & otherwise \end{cases} \quad (7)$$

$$Decision1 = \begin{cases} true & if \quad (Smooth = true) \\ & or \quad (N\_Smooth = true) \\ false & otherwise \end{cases} \quad (8)$$

Next, we take $p_{i,j}$ into account. At first, we must calculate two values, the difference between $f_{i,j}$ and *T_H_max* , and the difference between $f_{i,j}$ and *T_H_min*. A threshold *T_b* is used to compare the obtained differences. As in the case of $W_{B\_Half}$ the similar method is worked. The equations are as

$$IM\_T\_H = \begin{cases} true & if \quad \left(\left| f_{i,j} - T\_H\_\max \right| \ge T\_b\right) \\ & or \quad \left(\left| f_{i,j} - T\_H\_\min \right| \ge T\_b\right) \\ false & otherwise \end{cases} \quad (9)$$

$$IM\_B\_H = \begin{cases} true & if \quad \left(\left| f_{i,j} - B\_H\_\max \right| \ge T\_b\right) \\ & or \quad \left(\left| f_{i,j} - B\_H\_\min \right| \ge T\_b\right) \\ false & otherwise \end{cases} \quad (10)$$

$$Decision2 = \begin{cases} true & if \quad (IM\_T\_H = true) \\ & or \quad (IM\_B\_H = true) \\ false & otherwise \end{cases} \quad (11)$$

At the end, we can make a transient judgement whether $p_{i,j}$ belongs to a probably noisy pixel or is noise free.

*2). Fringe module:* It is difficult to conclude the pixel is noisy or situated on an edge. For solving this difficulty, four directions from E1 to E4 as illustrated in Fig. 3 is defined. For example, consider direction E1, by manipulating the absolute difference between the two pixel values along the direction and $f_{i,j}$ , we can decide if it is edge or not. The complete equations are as

$$F\_E1 = \begin{cases} true & if \quad \left(|a - f_{i.j}| \geq T\_c\right) \\ & or \quad \left(|h - f_{i,j}| \geq T\_c\right) \\ & or \quad \left(|a - h| \geq T\_d\right) \\ false & otherwise \end{cases} \tag{12}$$

$$F\_E2 = \begin{cases} true & if \quad \left(|c - f_{i,j}| \geq T\_c\right) \\ & or \quad \left(|f - f_{i,j}| \geq T\_c\right) \\ & or \quad \left(|c - f| \geq T\_d\right) \\ false & otherwise \end{cases} \tag{13}$$

$$F\_E3 = \begin{cases} true & if \quad \left(|b - f_{i,j}| \geq T\_c|\right) \\ & or \quad \left(|g - f_{i,j}| \geq T\_c\right) \\ & or \quad \left(|b - g| \geq T\_d\right) \\ false & otherwise \end{cases} \tag{14}$$

$$F\_E4 = \begin{cases} true & if \quad \left(|d - f_{i,j}| \geq T\_c\right) \\ & or \quad \left(|e - f_{i,j}| \geq T\_c\right) \\ & or \quad \left(|d - e| \geq T\_d\right) \\ false & otherwise \end{cases} \tag{15}$$

$$Decision3 = \begin{cases} false & if \quad (F\_E1) \quad or \quad (F\_E2) \\ & or \quad (F\_E3) \quad or \quad (F\_E4) \\ true & otherwise \end{cases} \tag{16}$$

*3) Similarity module:*

The similarity module is the last step in impulse detection. The intensity values in mask W might be close when it is positioned in a noise free area. In the variational sequence, impulse is generally positioned near one of its ends and median is frequently placed in the centre. So, there is a chance for noisy signals if the variational series contains very large or small values. Based on this concept, we can obtain the values (fourth, fifth, and sixth) which are near to the median in mask W, by sorting the nine values in increasing order. The $4\_thinW_{i,j}$, $MedInW_{i,j}$ , and $6\_thinW_{i,j}$ are stands for fourth, fifth, and sixth values correspondingly. To perform the operation,

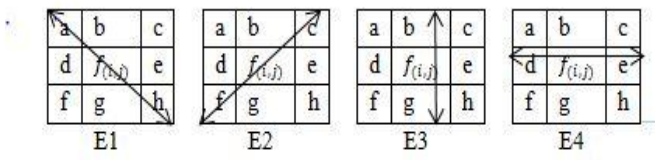


Fig.3. Four directions in IDTBDM.

we need to define the following variables $Maxm_{i,j}$ and $Minm_{i,j}$ as

$$Maxm_{i,j} = 6\_thinW_{i,j} + T\_e \tag{17}$$

$$Minm_{i,j} = 4\_thinW_{i,j} - T\_e \tag{18}$$

To evaluate the status of pixel $p_{i,j}$ , $Maxm_{i,j}$ and $Minm_{i,j}$ are used. We have to do some alterations in order to make the decision more perfect such as

$$P_{\max} = \begin{cases} Maxm_{i,j} & if \left(Maxm_{i,j} \leq MedianInW_{i,j} + T\_f\right) \\ MedianInW_{i,j} + T\_f & otherwise \end{cases} \tag{19}$$

$$P_{\min} = \begin{cases} Minm_{i,j} & if \left(Minm_{i,j} \leq MedianInW_{i,j} - T\_f\right) \\ MedianInW_{i,j} - T\_f & otherwise \end{cases} \tag{20}$$

At the end, we can conclude that $P_{i,j}$ is a noise pixel, if $f_{i,j}$ is not between $P_{max}$ and $P_{min}$. For creating restored value, edge preserving filter is used. For noise free pixel, we keep the same value $f_{i,j}$ will be the output. Equation is shown below

$$Decision4 = \begin{cases} true & if \left(f_{i,j} \geq P_{\max}\right) \quad or \quad \left(f_{i,j} \leq P_{\min}\right) \\ false & otherwise \end{cases} \tag{21}$$

It is clear that, in this proposed technique, the excellence of denoised images is being affected by the threshold. Take in to account our experimental analysis, the thresholds $T_a, T_b, T_c, T_d, T_e$ and $T_f$ are fix as 20, 25, 30, 60, 15, and 60 respectively.

### B. *Edge preserving image filter*

The Fig. 4. shows, the dataflow of our edge preserving image filter. The method works as follows. Consider eight directional differences from $D_a$ to $D_h$ , as illustrated in Fig. 5.

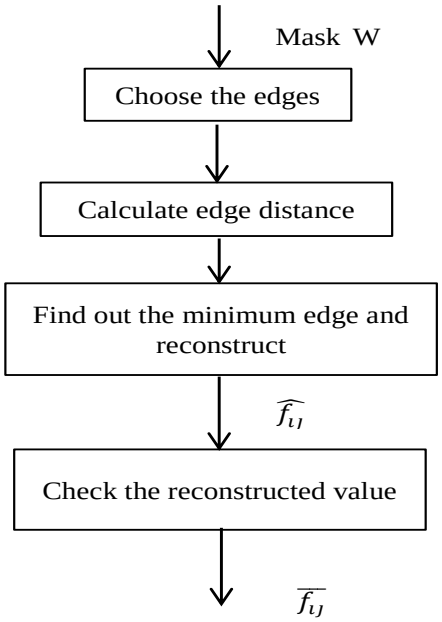


Fig.4. Dataflow of edge-preserving image filter.

Only the noise free pixels are taken into account to avoid possible false detection. Directions passing through the suspected pixels are avoided. The directional differences of the chosen directions which are calculated by the edge filter are used and the smallest among them is calculated. The equations are as follows:

$$\begin{aligned} D_a &= |d-h| + |a-e| \\ D_b &= |a-g| + |b-h| \\ D_c &= |b-g| \times 2 \\ D_d &= |b-f| + |c-g| \\ D_e &= |c-d| + |e-f| \\ D_f &= |d-e| \times 2 \\ D_g &= |a-h| \times 2 \\ D_h &= |c-f| \times 2 \end{aligned} \tag{22}$$

$$\hat{f}_{i,j} = \begin{cases} (d+h+a+e) \div 4 & if \quad (D_a = D_{\min}) \\ (a+g+b+h) \div 4 & if \quad (D_b = D_{\min}) \\ (b+g) \div 2 & if \quad (D_c = D_{\min}) \\ (b+f+c+g) \div 4 & if \quad (D_d = D_{\min}) \\ (c+d+e+f) \div 4 & if \quad (D_e = D_{\min}) \\ (d+e) \div 4 & if \quad (D_f = D_{\min}) \\ (a+h) \div 4 & if \quad (D_g = D_{\min}) \\ (c+f) \div 4 & if \quad (D_h = D_{\min}) \end{cases} \tag{23}$$

In the final block of Fig. 4, the smallest directional difference specifies that the spatial relation of $p_{i,j}$, with the pixels in the particular direction mentioned is the strongest. So there is a greater probability for the existence of an edge in that direction.. So $\overleftrightarrow{f}_{i,j}$, the mean of luminance values of the pixels which belonging to the smallest directional difference is determined.

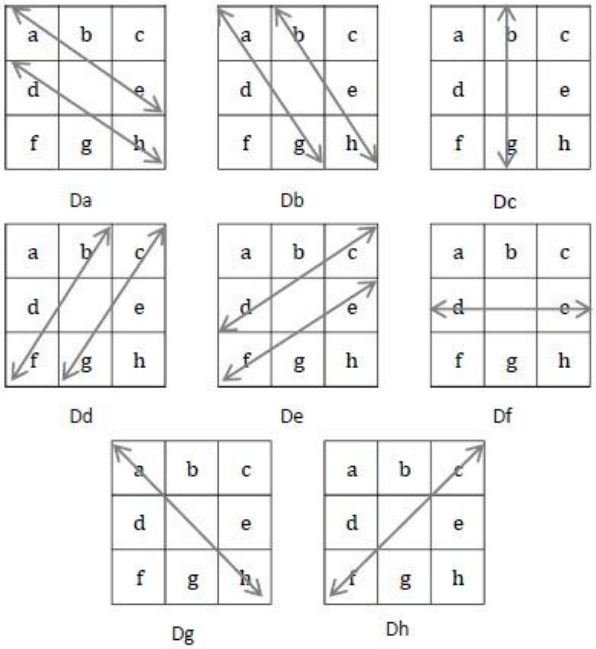


Fig.5. Eight directional differences of IDTBDM

To filter the bias edge, a tuning skill is required. If the mean calculated sits on an edge, it will be equal to the median of g, e, d and b because of the spatial relation. If not, the values of $f_{i,j}$ will be substituted by the median of g, e, d and b. Thus the algorithm preserves an edge in the image. We can express $\overline{f}_{i,j}$ as

$$\overline{f}_{i,j} = median \quad \left(\hat{f}_{i,j}, b, d, e, g\right) \tag{24}$$

## III. SYSTEM ARCHITECTURE OF IDTBDM

The block diagram of the system architecture for IDTBDM is shown in Fig.6. System architecture includes five modules: split module, isolation module, fringe module, similarity module and edge preserving filter.

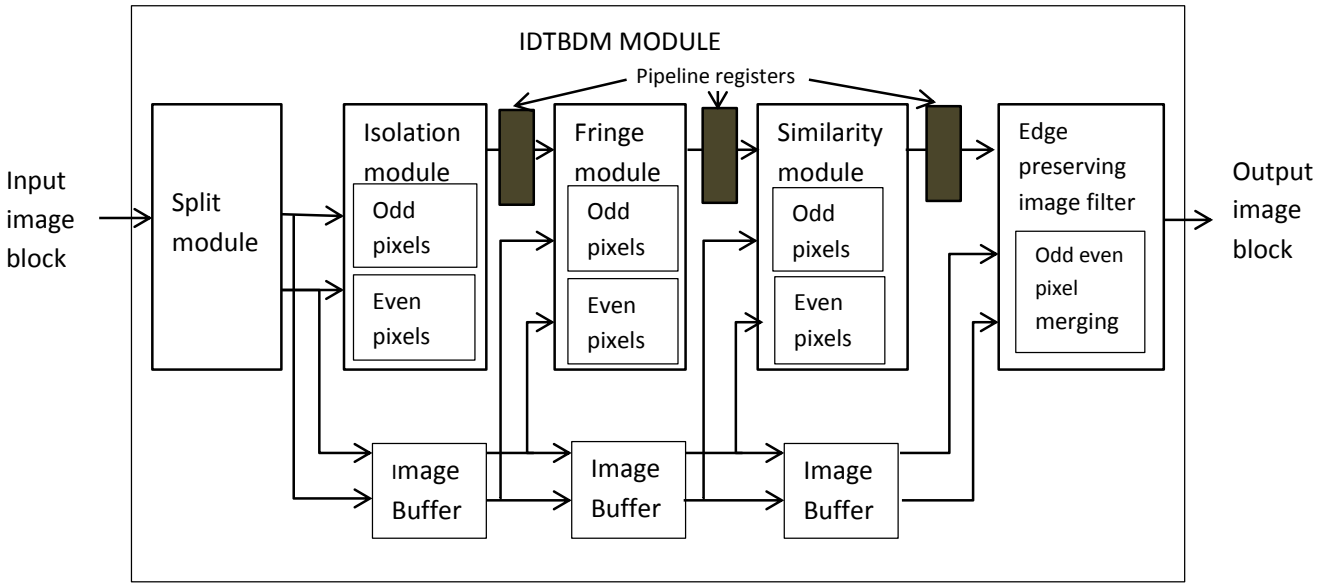


Fig.6. Block diagram of system architecture for IDTBDM

TABLE I
DEVICE UTILIZATION AND TIMING SUMMARY OF IDTBDM SYSTEM

| | | |
|---|---|---|
| Slice logic utilization | | |
| Number of slice registers | 3158 out of 44800 | 2% |
| Number of slice LUTs | 26482 out of 44800 | 59% |
| Number used as Logic | 26410 out of 44800 | 58% |
| IO utilization | | |
| Number of Ios | 579 | |
| Number of bonded IOBs | 515 out of 640 | 58% |
| Specific Feature Utilization | | |
| Number of BUFG/BUFGCTRLs | 2 out of 32 | 6% |
| Clock period | 9.39ns | |
| Clock frequency | 106.48 MHz | |

To improve the performance of architecture two strategies are implemented: parallel processing and pipelining. To provide concurrent operation, parallel processing is implemented in each module by splitting the input image block in to odd and even pixels. For this, split module is introduced in the architecture as the first block which divides the input image in to odd and even index pixels. This facilitates the concurrent operation in each of the following modules. To improve clock speed, pipeline registers are placed in between modules. Image buffers are used to save the previous cycle input image block. This design not only provides clock improvement but also streams the output in every clock cycle.

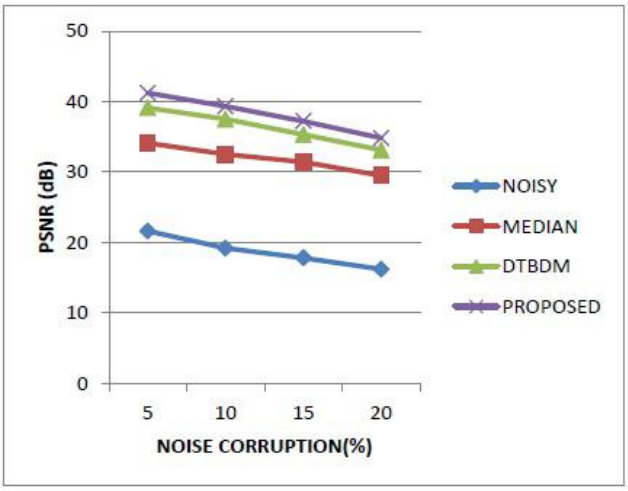


Fig.7. Noise corruption (%) vs PSNR (dB).

## IV. IMPLEMENTATION RESULTS

The proposed architecture is modelled using VHDL and implemented on Xilinx Virtex 5 ML507 FPGA. The device utilization and timing summary is shown in Table 1. To examine the distinguishing features and the quality of denoised images of different denoising algorithms, various simulations are carried out on eight-bit gray scale images having size 256x256: Lena, Blackbuck and Peppers. The corrupted image versions are generated in MATLAB environment adding random-valued impulse noise at various noise densities. Then,

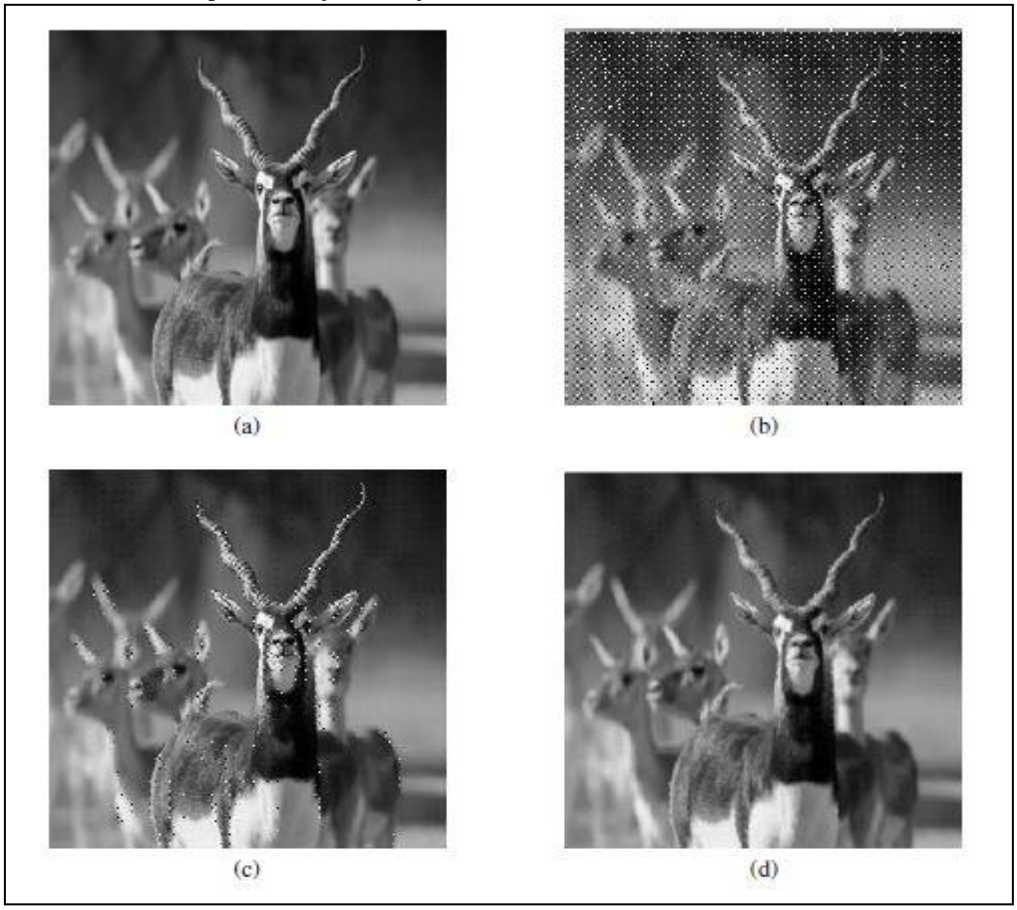


Fig.8. Results of different methods in restoring 10 percent corrupted image Blackbuck. (a) Original image (b) Noisy image (c) DTBDM (d) IDTBDM

TABLE II
COMPARISON OF PSNR (dB) OF IMAGES CORRUPTED BY 10 % IMPULSES

| Method/Images | Lena | Blackbuck | Peppers |
|---|---|---|---|
| Noisy | 19.18 | 19.62 | 20.12 |
| Median | 32.51 | 33.12 | 33.64 |
| DTBDM | 37.51 | 37.92 | 38.12 |
| IDTBDM | 39.34 | 39.87 | 40.12 |

we use different methods to find out impulse noise and restore the corrupted image. The proposed IDTBDM is compared with DTBDM by comparing the two parameters. The quantitative quality of the restored images attained from different method are analysed by observing the peak signal-to-noise ratio (PSNR). Table.2 list the restoration results in PSNR (dB) of different images corrupted by 10 percent impulses. Graphical representation of previous PSNR values and proposed PSNR values is shown in Fig.7. Comparing with those experimental results, the quantitative qualities of proposed IDTBDM are better than DTBDM. Also the experimental results demonstrate the excellent performance of IDTBDM in terms of image quality shown in Fig.8.

## V. CONCLUSION

In this paper, a high performance architecture for FPGA prototyping is proposed for the removal of random valued impulse noise. The main advantage of this architecture is that it is of a very low complexity. Our design achieves clock speed of 106 MHZ. It provides better results in terms of PSNR and image quality.

## REFERENCES

[1] R. C. Gonzalez and R. E. Woods, Digital Image Processing. Pearson Education, Upper Saddle River, New Jersey, 2007

[2] Chih-Yuan Lien, Chien-Chuan Huang, Pei-Yin Chen, Member, IEEE, and Yi-Fan Lin "An Efficient Denoising Architecture for Removal of Impulse Noise in Images," IEEE transactions on computers, vol. 62, no. 4, april 2013.

[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] T. Nodes and N. Gallagher, "Median Filters: Some Modifications and Their Properties," IEEE Trans. Acoustics, Speech, Signal Processing, vol. ASSP-30, no. 5, pp. 739-746, Oct. 1982.

[6] S.-J. Ko and Y.-H. Lee, "Center Weighted Median Filters and Their Applications to Image Enhancement," IEEE Trans. Circuits Systems, vol.38, no. 9, pp. 984-993, Sept. 1991.

[7] T. Sun and Y. Neuvo, "Detail-Preserving Median Based Filters in Image Processing," Pattern Recognition Letters, vol. 15, pp. 341-347, Apr. 1994.

[8] E. Abreu, M. Lightstone, S.K. Mitra, and K. Arakawa, "A New Efficient Approach for the Removal of Impulse Noise from Highly Corrupted Images," IEEE Trans. Image Processing, vol. 5, no. 6, pp. 1012-1025, June 1996.

[9] T. Chen and H.R. Wu, "Adaptive Impulse Detection Using Center-Weighted Median Filters," IEEE Signal Processing Letters, vol. 8, no.1, pp. 1-3, Jan. 2001.

[10] B. De Ville, "Decision Trees for Business Intelligence and Data Mining," SAS Publishing, 2007.

[11] S. Rasoul Safavian and D. Landgrebe, "A Survey of Decision Tree Classifier Methodology," IEEE Trans. Systems Man, Cybernetics, vol. 21, no. 3, pp 660-674, May 1991.

[12] H.-H. Tsai, X.-P. Lin, and B.-M. Chang, "An Image Filter with a Hybrid Impulse Detector Based on Decision Tree and Particle Swarm Optimization," Proc. IEEE Intl Conf. Machine Learning and Cybernetics,July 2009.

[13] H.-L. Eng and K.-K. Ma, "Noise Adaptive Soft-Switching Median Filter," IEEE Trans. Image Processing, vol. 10, no. 2, pp. 242-251, Feb. 2001.