ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol.2, Special Issue 4, September 2014

FPGA Implementation of Removal of Impulse Noise in Images by using MDBUT Technique

Y.Lilly Margaret[1], T.Vijaya Nirmala[2]

[1]M.Tech, Department of ECE**,** AITS, Kadapa, Andhra Pradesh, India.

[2]Assistant professor, Department of ECE, AITS, Kadapa, Andhra Pradesh, India

ABSTRACT: A novel idea of using median deviation parameter in estimating the noise in the images is proposed and successfully applied to gray level images. The median filter which is very popular in removing the salt and pepper noise from the images has undergone many changes in recent past. To this modified median filter the concept of mean deviation is added and used in estimating and removing the noise. The proposed method is implemented by developing a Graphical User Interface in MATLAB and also implemented using the Spartan 3E Filed Programmable Device. The results are found to be better than earlier methods and also robust in terms of preserving the contrast and fine details of the image even at high noise densities.

KEYWORDS: Median filter; Mean deviation; Salt and Pepper noise; MATLAB; Image processing

I. INTRODUCTION

During the last few years there has been innumerable number of research papers published in various journals on the application of median filters for removal of salt and pepper noise from the images, by various authors [1,2,3]. The success of median filters can be attributed to two intrinsic properties: edge preservation and efficient noise attenuation with robustness against impulsive type noise. Edge preservation is essential in image processing due to the nature of visual perception. Edges also occur in biomedical signals when the “system” moves from one state to another .The fact that some signals are invariant to median filtering offers interesting possibilities. In noise filtering, the basic idea is how to preserve some desired signal features while attenuating the noise. An optimal situation would arise if the filter could be designed **so** that the desired features were invariant to the filtering operation and only noise would be affected. As the superposition principle cannot be applied to non-linear filters, this can never be fully achieved. However, when a signal consists of constant areas and stepwise changes between these areas, a similar effect is achieved. Noise will be attenuated, but stepwise changes will remain [1]. In spite of this, the median filter is far from being a perfect filtering method since it may remove fine details, sharp corners and thin lines. The main reason is that the ordering process destroys any structural and spatial neighbourhood information. To overcome such problems many modifications were applied to median filters which resulted various new filters. For example, An adaptive median filter eliminated the above drawback, but owing to its increasing window size lead to blurring of images [2]. Switched median filters [3,4] were proposed. But these filters do not have a strong decision or does not consider the local statistics.

To elude the flaw, Decision based filter [5] was proposed. This filter identifies the processed pixel as noisy, if the pixel value is either 0 or 255; else it is considered as noiseless pixel. Under High noisy environment the DBA filter replaces the noisy pixel with neighbourhood pixel. In spite of repeated replacement of neighbourhood pixel results in streaks in restored image. To avoid streaks in images an improved DBA (DBUTMF) [6] is proposed with replacement of median of unsymmetrical trimmed output, but under high noise densities all the pixel inside the current would take all 0’s or all 255’s or combination of both 0 and 255. Replacement of trimmed median did not fare well for the above case. So, the modified decision based unsymmetric trimmed median filter (MDBUTMF) was proposed [7]. This 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.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol.2, Special Issue 4, September 2014

If the processing pixel takes the maximum or minimum gray level then it is noisy pixel which is processed by MDBUTMF. If the selected window contains salt/pepper noise as processing pixel (i.e., 255/0 pixel value) and neighbouring pixel values are either 0 or 255, then the median of the centre pixel will be either 0 or 255. So, to solve this problem the authors have taken the mean of the selected window and the centre pixel is replaced by the mean. But, it is our observation that a better alternative can be thought of by considering certain statistical parameters which will be a robust solution even at high salt and pepper noises. Also, some authors [8] proposed an idea that, when the median of the selected window is either 0 or 255, the centre pixel must be replaced by the top neighbour pixel value. But, here there is no convincing reason for this argument. So, to address this problem, the authors have used the statistics of the window elements. Instead of considering the mean an attempt is made to compute the mean deviation of the widow and the central pixel is replaced by this.

The rest of the paper is organized as follows: Section II discusses the development details of proposed image processing algorithms, section III gives the illustration of the proposed algorithm, section IV gives the implementation of the proposed algorithms using the MATLAB GUI and FPGA and section V summarizes the results and conclusions.

II. PROPOSED ALGORITHM

In many practical cases of image processing, only a noisy image is available. This circumstance is known as the blind condition. Many denoising methods usually require the exact value of the noise distribution as an essential filter parameter. So, the noise estimation methods in the spatial domain, use the variance or standard deviation to estimate the actual added noise distribution. But it is found that the mean deviation provides better results than the variance or standard deviation to estimate the noise distribution. The advantage of this approach is that the mean deviation is actually more efficient than the standard deviation in practical situations [9].The standard deviation emphasizes a larger deviation; squaring the values makes each unit of distance from the mean exponentially (rather than additively) larger [10]. The larger deviation will cause overestimation or under estimation of the noise. So, we assume that use of the mean deviation may contribute to more accurate noise estimation. Keeping these points in view, the authors have used the mean deviation parameter in deciding the noise pixel and replaced the central pixel by its mean deviation instead of its mean. The steps in the proposed algorithm are given below.

Step 1: Select 2-D window of size 3x 3. Assume that the pixel being processed is Pij.

Step 2: If this pixel value lies between 0 and 255, 0< Pij < 255, this is considered as an uncorrupted pixel. So, no processing is required and its value is left unchanged.

Step 3: If Pij = 0 or 255, it indicates that the pixel is corrupted by salt and pepper noise. Here two cases are considered

Case i: The selected window contains few 0 or 255 elements and other elements lie between 0 and 255.Then the 0 and 255 elements are discarded and the median of the remaining elements is found. The Pij pixel is replaced with this median value.

Case ii: Suppose the window under consideration has all the elements either 0 or 255 .Then median of these elements may also be either 0 or 255 which is again a noisy element. Now , find the mean deviation or absolute mean deviation of the window which can never be 0 or 255.Replace the pixel Pij with this mean deviation value.

Step 4: Apply the steps 1 to 3 for all the pixels in the image for complete processing.

III. ILLUSTRATION OF THE PROPOSED ALGORITHM

This section explains the proposed algorithm with a flow chart and numerical examples. In the processing methodology the entire image must be checked for the presence of noise. The flow chart of the algorithm is shown in Fig.1

Let us first consider the Case ii. The 3x3 window under consideration has all the elements between 0 and 255 as shown below.

$$\begin{bmatrix} 76 & 48 & 125 \\ 69 & \mathbf{86} & 49 \\ 98 & 77 & 55 \end{bmatrix}$$

Here the central pixel Pij is 86 which is a noise free pixel .So, no further processing is required for this pixel.

Next, let us consider a 3x3 window which contain both 0 and 255 elements along with other elements as shown below.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol.2, Special Issue 4, September 2014

$$\begin{bmatrix} 76 & 48 & 125 \\ 69 & \mathbf{255} & 49 \\ 0 & 77 & 255 \end{bmatrix}$$

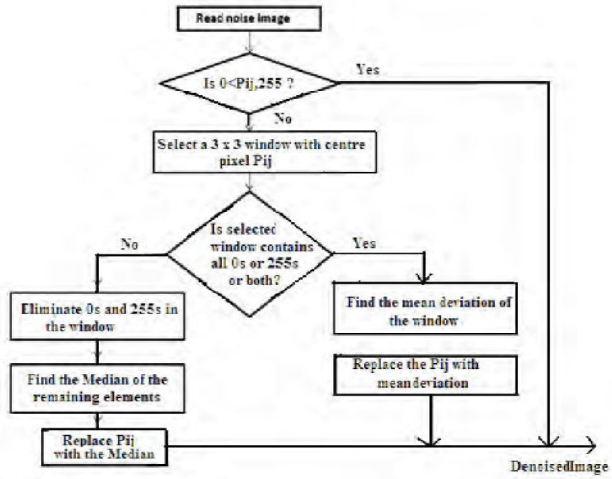

Figure 1. Flow chart of the proposed algorithm

Here the Pij is 255.To process this pixel, eliminate all the 0 and 255 elements and arrange the remaining elements in the ascending order. The ascending order after elimination is

$$[48\ 49\ 69\ 76\ 77\ ]$$

The median of the window now is 69. So, the central pixel 255 is replaced by 69. As a last illustration let us consider the Case ii: Let us consider the 3x3 window shown below which contains all the 0 or 255 elements.

$$\begin{bmatrix} 255 & 0 & 255 \\ 0 & \mathbf{255} & 0 \\ 255 & 0 & 255 \end{bmatrix}$$

For this window the central pixel Pij is equal to 255.The median of the window is either 0 or 255.Replacing the Pij with this value is of no use. So, find the mean deviation of the window. The mean deviation of the window is

$$\frac{\sum |x - \bar{x}|}{n}$$

Where x is the element of the window, x is the mean of the window elements and n is the total number of elements. So, for the above window the mean deviation is 126. So, the central pixel 255 is replaced with the value 126.

IV. EXPERIMENTAL SETUP

A. Xilinx Platform Studio

The Xilinx Platform Studio (XPS) is the development environment or GUI used for designing the hardware portion of your embedded processor system. The block diagram of the proposed system is given by the following diagram.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol.2, Special Issue 4, September 2014

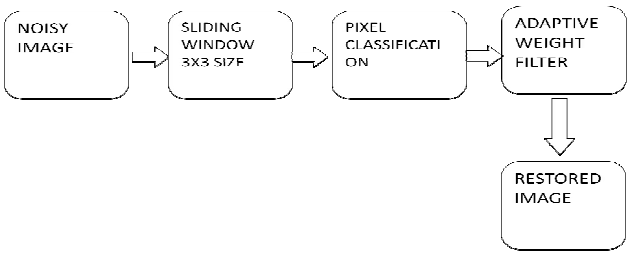

Figure 2: Block Diagram of Impulse noise Removal

B. Embedded Development Kit

Xilinx Embedded Development Kit (EDK) is an integrated software tool suite for developing embedded systems with Xilinx MicroBlaze and PowerPC CPUs. EDK includes a variety of tools and applications to assist the designer to develop an embedded system right from the hardware creation to final implementation of the system on an FPGA. System design consists of the creation of the hardware and software components of the embedded processor system and the creation of a verification component is optional. A typical embedded system design project involves: hardware platform creation, hardware platform verification (simulation), software platform creation, software application creation, and software verification. Base System Builder is the wizard that is used to automatically generate a hardware platform according to the user specifications that is defined by the MHS (Microprocessor Hardware Specification) file. The MHS file defines the system architecture, peripherals and embedded processors]. The Platform Generation tool creates the hardware platform using the MHS file as input. The software platform is defined by MSS (Microprocessor Software Specification) file which defines driver and library customization parameters for peripherals, processor customization parameters, standard 110 devices, interrupt handler routines, and other software related routines. The MSS file is an input to the Library Generator tool for customization of drivers, libraries and interrupts handlers.
The creation of the verification platform is optional and is based on the hardware platform. The MHS file is taken as an input by the Simgen tool to create simulation files for a specific simulator.

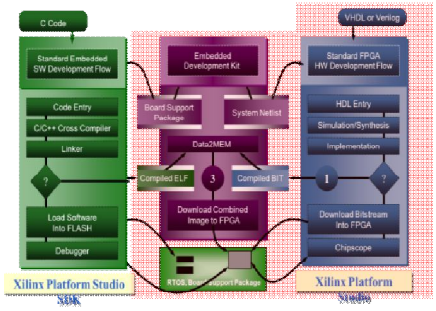

Figure 3: Embedded Development Kit Design Flow

Three types of simulation models can be generated by the Simgen tool: behavioural, structural and timing models. Some other useful tools available in EDK are Platform Studio which provides the GUI for creating the MHS and MSS files. Create / Import IP Wizard which allows the creation of the designer's own peripheral and import them into EDK projects. Platform Generator customizes and generates the processor system in the form of hardware netlists. Library Generator tool configures libraries, device drivers, file systems and interrupt handlers for embedded processor system. Bitstream Initializer tool initializes the instruction memory of processors on the FPGA shown in figure 3. GNU Compiler tools are used for compiling and linking application executables for each processor in the system [6]. There are two options available for debugging the application created using EDK namely: Xilinx Microprocessor Debug (XMD) for debugging the application software using a Microprocessor Debug Module (MDM) in the embedded processor system, and Software Debugger that invokes the software debugger corresponding to the compiler being used for the processor.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol.2, Special Issue 4, September 2014

C. **Software Development Kit:**

Xilinx Platform Studio Software Development Kit (SDK) is an integrated development environment, complimentary to XPS, that is used for C/C++ embedded software application creation and verification. SDK is built on the Eclipse open source framework. Soft Development Kit (SDK) is a suite of tools that enables you to design a software application for selected Soft IP Cores in the Xilinx Embedded Development Kit (EDK).The software application can be written in a "C or C++" then the complete embedded processor system for user application will be completed, else debug & download the bit file into FPGA. Then FPGA behaves like processor implemented on it in a Xilinx Field Programmable Gate Array (FPGA) device.

V. RESULTS

The Algorithm is implemented in Microblaze Processor and the results are furnished in the figures below.

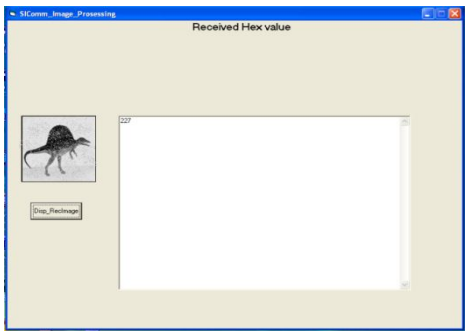

Figure 4: Noisy Image

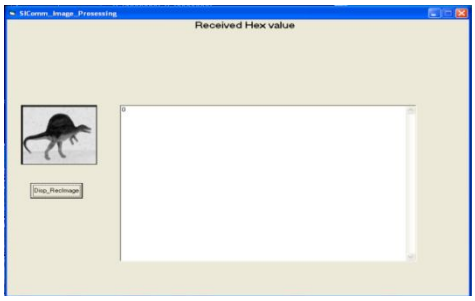

Figure 5: Restored Image

The synthesis report of the proposed algorithm is as shown in figure 6.

```
Design Summary:
Number of errors:      0
Number of warnings:    3
Logic Utilization:
  Number of Slice Flip Flops:         976 out of   3,840   25%
  Number of 4 input LUTs:           1,560 out of   3,840   40%
Logic Distribution:
  Number of occupied Slices:                      1,174 out of   1,920   61%
    Number of Slices containing only related logic:   1,174 out of   1,174  100%
    Number of Slices containing unrelated logic:          0 out of   1,174    0%
      *See NOTES below for an explanation of the effects of unrelated logic
Total Number 4 input LUTs:          1,953 out of   3,840   50%
  Number used as logic:             1,560
  Number used as a route-thru:          7
  Number used for Dual Port RAMs:     256
    (Two LUTs used per Dual Port RAM)
  Number used as Shift registers:     130
  Number of bonded IOBs:               62 out of      97   63%
    IOB Flip Flops:                   122
  Number of Block RAMs:                4 out of      12   33%
  Number of MULT18X18s:                3 out of      12   25%
  Number of GCLKs:                     2 out of       8   25%
  Number of DCMs:                      1 out of       4   25%
  Number of BSCANs:                    1 out of       1  100%

   Number of RPM macros:           3
Total equivalent gate count for design:  325,662
```

Figure 6: Synthesis Report

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol.2, Special Issue 4, September 2014

VI. CONCLUSION

The proposed algorithm is tested using the MATLAB and FPGA hardware. From the observation of figure 5 and figure 6, one can come to the conclusion that the present method is showing reasonably good performance at high noise density levels. Also it is clear that the fine details of the image and the contrast levels are much better in the case of the proposed algorithm. This confirms the validity of the proposed algorithm for denoising the high density salt and pepper noise from the images.

In fact every image processing algorithm cannot be implemented effectively in hardware. Most of the image processing algorithms are inherently computationally intensive and may require vast computing power if strict time-constraints are posed. The spreading of parallel image processing techniques and systems has been driven not only by the afore-mentioned need, but the inherent parallel nature of many image processing algorithms has also eased this evolution. The execution characteristics of a certain parallel algorithm on a given architecture heavily depends on the 'mutual conformance' of the mentioned algorithm and the architecture pair. Two algorithms with similar sequential performance may behave very differently in a parallel environment. In sequential algorithms the complexity is expressed in terms of operations and storage. In parallel environments these terms are not adequate for characterizing the computing efficiency - fewer operations does not directly mean shorter execution time since there is a definite overhead involved due to availability of resources and communication between processors [11].

REFERENCES

[1] T.A. Nodes and N.C. Gallagher, Jr., "The output distribution of median type filters," IEEE Trans Communication., 32(5): 532-541,1984.
[2] H. Hwang and R. A. Hadded, "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] K. S. Srinivasan and D. Ebenezer, "A New Fast and Efficient Decision- Based Algorithm for Removal of High-Density Impulse Noises", IEEE Signal Processing Letters, Vol.14, No.6, and March 2007.
[6] V. Jayaraj and D. Ebenezer, "A new switching-based median filtering scheme and algorithm for removal of high-density salt and pepper noise in image**,"** *EURASIP J. Adv. Signal Process.*, 2010.
[7] S. Esakkirajan , T. Veerakumar , Adabala N. Subramanyam and C.H. Prem Chand, "Removal of high density Salt and pepper noise through modified decision based Unsymmetrical trimmed median filter." IEEE Signal processing letters, Vol. 18,no.5, May 2011.
[8] R. Pushpavalli, G.Sivaradje," Switching Median Filter for Image Enhancement", International Journal of Scientific & Engineering Research, Volume 3, Issue 2, February.2012.
[9] Suhaila Sari, Hazli Roslan, Tetsuya Shimamura," Noise Estimation by Utilizing Mean Deviation of Smooth Region in Noisy Image", Fourth International Conference on Computational Intelligence, Modelling and Simulation,2012.
[10] S. Gorard, "Revisiting a 90-year-old debate: the advantages of the mean deviation," British Journal of Educational Studies, vol. 53, no. 4, pp. 417–430, Dec. 2005.
[11] Donald G Bailey , Christopher T Johnston," Algorithm Transformation for FPGA Implementation", Fifth IEEE International Symposium on Electronic Design, Test & Applications,2010.

BIOGRAPHY

Y. LILLY MARGARET, born in kadapa, A.P., India in 1991. She received her B.Tech Degree in Electronics and Communication Engineering from J.N.T University Anantapur, India. Presently pursuing M.Tech (VLSI System Design) from Annamacharya Institute of Technology and Sciences, Kadapa, A.P., India. Her research interests include VLSI, Digital Image Processing and Digital Signal Processing.

T. VIJAYA NIRMALA, has received her M.Tech degree from JNTU Anantapur. She is working as Assistant Professor in the Department of Electronics & Communication Engineering, Annamacharya Inst of Technology & Science, Kadapa, A.P, India. Her areas of interests are VLSI, Digital Image Processing and Communication Systems.