ISSN : 2348 - 9065 (Online)
ISSN : 2349 - 3143 (Print)
Vol. 2, Issue 4 Oct. - Dec. 2015
International Journal of Research in Electronics and Communication Technology (IJRECT 2015)

Detection of Traces by using Median Filtering on Reconfigurable Hardware Platform

[I]Heerakar Rajini, [II]J. Lingaiah, [III]Mohammad Sajeed Pasha
[I]M. Tech. VLSI, Arjun College of Technology and Sciences, Batasingaram village, Hayathnagar, R.R. Dist, Hyderabad, Telangana, India.
[II]Head of Dept. & Associate Professor, Dept. of Electronics and Communication Engineering in Arjun College of Technology and Sciences, Batasingaram Village, Hayathnagar, R.R. Dist, Telangana, India.
[III]Application Engineer in Unistring Tech Solutions Pvt Ltd, Hyderabad, Telangana, India.

Abstract
In Real time signal and image processing applications, it is often desirable to be able to perform some kind of noise reduction on an image or signal. Impulsive noise replaces the intensities associated to a certain percentage of pixels by the maximum or minimum possible intensity. This kind of noise is called salt and pepper noise. We go for a special kind of filter called Median filter [1] to reduce this noise.
The median filter is a non-linear filter which is commonly used to remove impulsive noise from images, while preserving edges and other details. The median of a given sequence can be found by sorting all values in the sequence and by choosing the middle value in the sorted sequence. The median filtering methods presented in this paper follow a histogram-based implementation.
In this project, three different directional median filtering methods for FPGA are proposed. For each of the techniques, four directions are processed at the same time. All techniques presented in this project aim to decrease processing time, while also reducing the hardware resource utilization. Three different directional median filtering methods are: a) By saving only the directional codes, b) Without saving directional codes c) By saving all codes of the window. The processing time requirements for the three proposed methods are dependent on the size of the window and number of filtering directions considered.

Keywords
Median filter, impulsive noise, histogram, FPGA.

I. Introduction

Digital images surrounds us everywhere now-a-days, with an increasing amount of devices capable of delivering, capturing and sharing multimedia sources. With increased interconnectivity it is easier to share experiences, often in form of an image or a video. Often this kind of media is shared on social network sites such as Facebook or Twitter. More devices are being able to connect to the Internet so media can be shared from anywhere, at anytime.
The devices used to share images is often times mobile phones, and their capturing device is in form of an integrated charge-coupled device (CCD). As keeping the size of mobile devices small is often essential, not much space is allowed for the image capturing device. With ever increasing resolutions being demanded of the CCD's the amount of light per pixel becomes diminishingly small. Mobile devices are often required to capture high mobility situations, which require high shutter speed, and/or dark scenes. Either situation making it difficult to capture enough light.

II. Median Filter

The median filter is a non-linear filter which is commonly used to remove impulsive noise from images. There are two types of filters: linear filters and non-linear filters. The median filter is a non-linear filter; it is a special case of rank order filters whose rank is half the length of the sequence. In image processing applications, median filter is used to remove impulsive noise from images while preserving the edges. The below figure 1 shows an example of median filtered image.

Fig. 1 Noisy image and median filtered image

Impulsive noise can be classified into two types:
(1) salt and pepper noise
(2) Random valued noise.
Salt and pepper noise pixels take only two values, either the minimum or the maximum possible value, for example, in a gray scale image, salt and pepper noise pixels will be either 0 or 255. The median of a given sequence is given by sorting the sequence and choosing the middle value from the sorted sequence.
In signal processing, it is often desirable to be able to perform some kind of noise reduction on an image or signal. The median filter is a nonlinear digital filtering technique, often used to remove noise. Such noise reduction is a typical pre-processing step to improve the results of later processing (for example, edge detection on an image). Median filtering is very widely used in digital image processing because, under certain conditions, it preserves edges while removing noise [4].

A. Median filtering methods

Various algorithms have been developed to implement the median filtering on hardware. Median filtering techniques are usually based on the sorting network architectures; another approach to

ISSN : 2348 - 9065 (Online)
ISSN : 2349 - 3143 (Print)

implement the median filter is based on the histogram.

1. Sorting network based median filtering

Sorting network architectures may depend on bubble sorting, quick sorting, and insertion sorting to implement the median filter on FPGAs. These sorting networks use 6 Compare and delay units to implement the median filter. The incoming pixels are passed through a network of comparators and swapping units - the comparator units compare two to three incoming pixels at once and then the swapping unit sorts them accordingly.

The median value will be the middle value of the sorting network. To implements a 3x3 median filter using bubble sort, 41 compare-and-swap units are required and as the size of the window increases, the number of compare-and-swap units required for implementation will also increase. Some optimizations may lead to the reduction in the number of these units required as presented. The resources required to implement the sorting network architecture on a FPGA device increases with the size of the filtering window. However, the sorting network based algorithms are independent of the size of the image and depend only on the size of the window [5].

2. Histogram based median filtering

Histogram is a representation of the distribution of the intensities in an image, i.e., it shows how many pixels in an image take a particular intensity value. The histogram is calculated by incrementing the value of the bin representing the corresponding intensity level.

Every time a particular intensity is encountered, the value of the corresponding bin is increased by one. Similar to any software implementation, the hardware implementation of the histogram requires as many counters as the expected number of intensity levels in an image. Each counter is associated to an intensity level and the values of these counters are incremented according to the incoming pixel intensities.

The cumulative histogram is calculated by increasing the values of all the bins greater than or equal to the incoming pixel intensity. Then, after building the cumulative histogram, the 7 first bin which has a value greater than or equal to the median index is taken as the median value of the window [6].

B. Median filtering types

There are two types of filters: linear filters and non-linear filters. The median filter is a non-linear filter; it is a special case of rank order filters whose rank is half the length of the sequence. In image processing applications, median filter is used to remove impulsive noise from images while preserving the edges[7].

The median of a given sequence can be found by sorting all values in the sequence and by choosing the middle value in the sorted sequence. For instance, in a sequence of N numbers, where N is odd, the median value is the (N+1)/2-th ordered number in the sorted sequence. Since salt and pepper noise pixel intensities assume only extreme values, they are easily excluded by median filtering, since they are ranked as either at the beginning or at the end of the sorted sequence. Therefore, salt and pepper noise is, in general, easier to remove than random valued noise.

Median filtering is usually based on data sorting algorithms, including bubble sort, quick sort, and insertion sort. Several techniques based on these algorithms have been proposed in the literature for implementing median filters on hardware. In these sorting schemes, the incoming pixels pass through a network of comparators and swapping units. The comparators compare two to three incoming pixels at once, while the swapping unit sorts them accordingly[8].

III. Block Diagram

The histogram is implemented by instantiating an array of registers is called bin nodes. Bin nodes represent all the possible gray level intensities in an input image (for a gray scale image where each pixel is represented using 8-bits, 2^8= 256 bins are required). A bin node is basically a counter that keeps track of the number of times the bin is incremented.

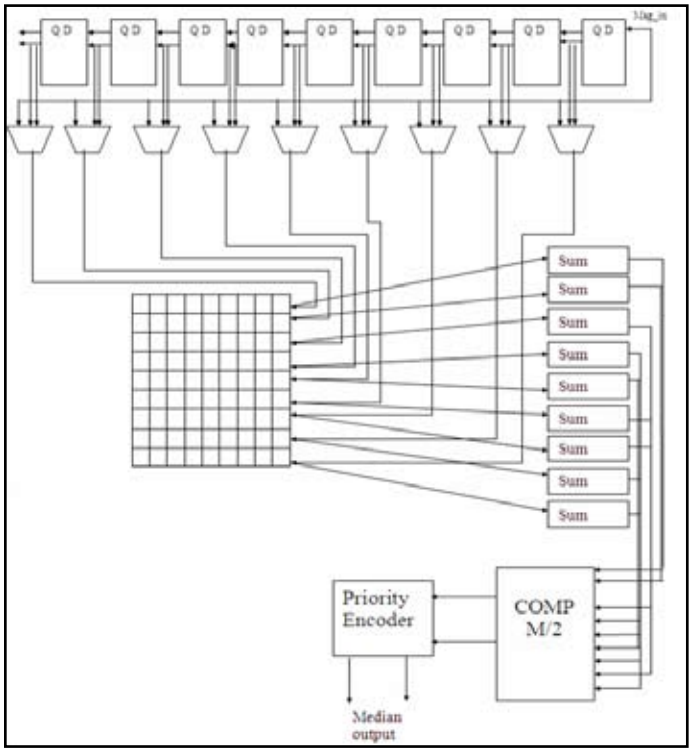

Fig.2 : Block diagram of median filter

The main block diagram of median filter is shown in figure 1 which consists of the modules:

1. PIPO shift registers (Accumulator)

The accumulator is a register in which the received data are temporarily stored. It has one input signal that stores an eight-bit word at every LD pulse and ten output signals: FULL-FLAG and the others nine lines for data. Once the accumulator is full, the FULL-FLAG signal is enabled what releases all data and the capture process remains in stand-by mode until the next LD pulse.

The PIPO shift registers which are of Parallel-in to Parallel-out Shift Registers, also act as a temporary storage device or as a time delay device is shown in figure 3. The DATA is presented in a parallel format to the parallel input pins PA to PD and then shifts it to the corresponding output pins QA to QD when the registers are clocked.

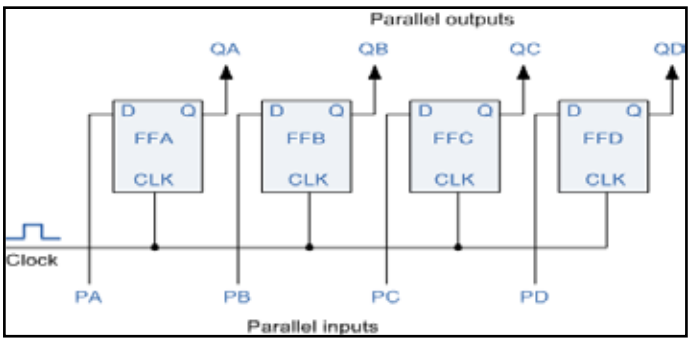

Fig.3 : PIPO shift registers

2. Sorting Network

The sorting network block is a nine-inputs/one-output combinational module with a data word length of eight bits. It is in fact, the kernel of the median filter architecture and is constituted by an array of seven blocks of three-data comparator modules as The topology of the three-data comparator blocks interconnections is directly related to the median algorithm. As it can be seen, it is a process divided in three stages: the first one is the column sorter, the second is the row sorter and finally the last one is the diagonal sorter. The connection scheme can also be seen in the format of the so called sorting network.

3. Comparators

A digital comparator or magnitude comparator is a hardware electronic device that takes two numbers as input in binary form and determines whether one number is greater than, less than or equal to the other number. Comparators are used in central processing units (CPUs) and microcontrollers (MCUs). Examples of digital comparator include the CMOS 4063 and 4585 and the TTL 7485 and 74682-'89.

It can be observed that if an incoming pixel has a value n, then the histogram at the nth bin is incremented. As a result, all bins with index greater than or equal to n in the cumulative histogram should be incremented. Thus, the cumulative histogram can be updated directly every time a new pixel comes in, by incrementing by "1" all counters in the cumulative histogram which correspond to the successive bins with index greater than or equal to the pixel value. The three-data comparator module is shown in figure 4.

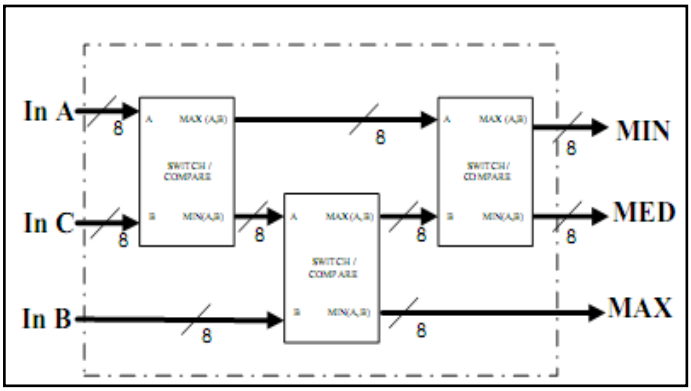

Fig. 4 : Block diagram of the three-data comparator module

The three data comparator module included in the sorting network. The three-data comparator Module which consists of three eight-bit word length inputs described as: In A, In B and In C can be identified. Also, three switch/compare blocks are internally connected what makes possible to collect always the median datum (piece of information) in the internal bus denoted as medium and the minimum and maximum data into the external buses.

Each one of the switch/compare blocks operates as follows: firstly, it takes two eight-bit word length input data, then both data are compared and finally two multiplexed outputs are enabled by directing the maximum value datum at the upper path MAX(A,B) and the minimum at the lower path MIN(A,B). Note that it includes a magnitude detector composed by four full adders (FA) and two eight-bit multiplexer.

4. 2-D matrixes

A Data Matrix code is a two-dimensional matrix barcode consisting of black and white "cells" or modules arranged in either a square or rectangular pattern. The information to be encoded can be text or numeric data. Usual data size is from a few bytes up to 1556 bytes. The length of the encoded data depends on the number of cells in the matrix. Error correction codes are often used to increase reliability: even if one or more cells are damaged so it is unreadable, the message can still be read. A Data Matrix symbol can store up to 2,335 alphanumeric characters. Here all the information gathered and sorted.

5. Priority encoder

A priority encoder is a circuit or algorithm that compresses multiple binary inputs into a smaller number of outputs. The output of a priority encoder is the binary representation of the ordinal number starting from zero of the most significant input bit. They are often used to control interrupt requests by acting on the highest priority request.

Here in priority encoder we observed the comp_vec_sum_match with the input of the mag_val_array which is mag_value as input. Finally got median value which is 3x3 windows as middle value after comparing and sorting.

IV. Simulation And Chipscope Results

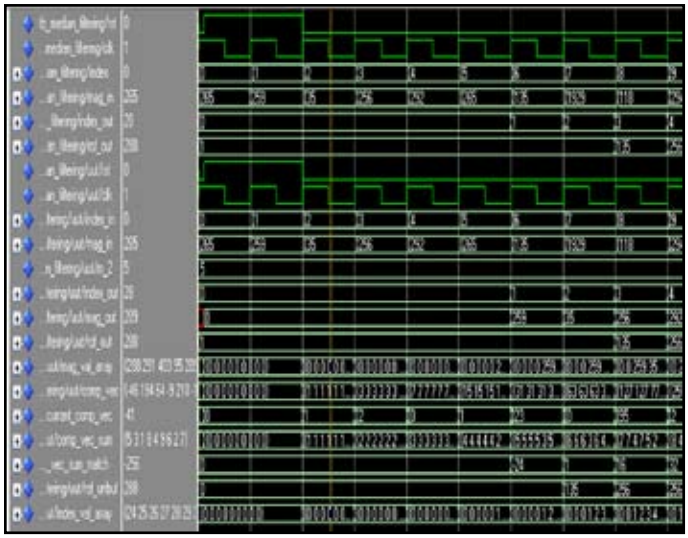

Fig. 5 : Simulation results of top module

ISSN : 2348 - 9065 (Online)
ISSN : 2349 - 3143 (Print)

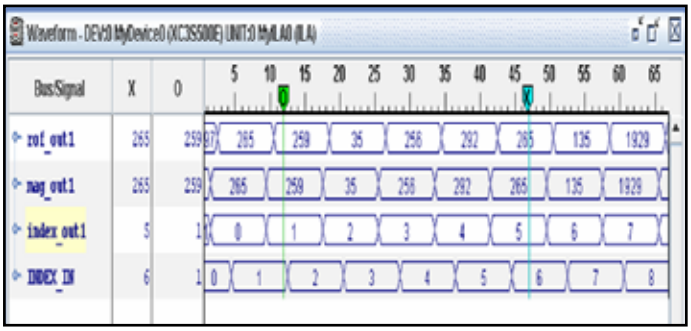

Fig. 6 : Chipscope results of top module

Table 1:

Logic utilization	used	available	utilization
Number of slice flipflops	698	9312	7%
Number of 4 input LUT's	467	9312	5%
Number of occupied slices	658	4656	14%
Number of bonded IOB's	76	232	32%
Number of block RAM's	6	20	30%

V. Conclusion

In this paper, a non linear median filtering was implemented to remove the impulsive noise by finding the median values of the pixels of the image by taking 9 pixels at a time of 3 by 3 window of an image.

The concept of median filtering developed on hardware reconfigurable device Xilinx Spartan 3E to improve the speed and lower the time consumption in find median value.

References

[1]. *H. T. Sencar and N. Memon, "Overview of state-of-the-artin digital image forensics," in Algorithms, Architectures and Information Systems Security, B. B. Bhattacharya, S. Sur-Kolay, S. C. Nandy, and A. Bagchi, eds., Statistical Science and Interdisciplinary Research 3, ch. 15, pp. 325–348, World Scientific Press, 2008.*

[2]. *H. Farid, "Image forgery detection," IEEE Signal Processing Magazine 26(2), pp. 16–25, 2009.*

[3]. *I. J. Cox, M. L. Miller, J. A. Bloom, J. Fridrich, and T. Kalker, Digital Watermarking and Steganography, Second edition, Morgan Kaufmann, New York, 2008.*

[4]. *R. B¨ohme, F. Freiling, T. Gloe, and M. Kirchner, "Multimedia forensics is not computer forensics," in Computational Forensics, Third International Workshop, IWCF 2009, The Hague, Netherlands, August 2009, Proceedings, Z. J. Geradts, K. Y. Franke, and C. J. Veenman, eds., Lecture Notes in Computer Science LNCS 5718, pp. 90–103, Springer Verlag, (Berlin, Heidelberg), 2009.*

[5]. *R. Neelamani, R. de Queiroz, Z. Fan, S. Dash, and R. G. Baraniuk, "JPEG compression history estimation for color images," IEEE Transactions on Image Processing 15(6), pp. 1365–1378, 2006.*

[6]. *M. Stamm and K. J. R. Liu, "Blind forensics of contrast enhancement in digital images," in Proceedings of the 15th IEEE International Conference on Image Processing (ICIP 2008), pp. 3112–3115, 2008.*

[7]. *G. Cao, Y. Zhao, and R. Ni, "Detection of image sharpening based on histogram aberration and ringing artifacts," in Proceedings of the 2009 IEEE International Conference on Multimedia and Expo (ICME 2009), pp. 1026–1029, 2009.*

[8]. *A. D. Ker and R. B¨ohme, "Revisiting weighted stego-image steganalysis," in Proceedings of SPIE-IS&T Electronic Imaging: Security, Forensics, Steganography, and Watermarking of Multimedia Contents X, E. J.Delp, P. W. Wong, J. Dittmann, and N. Memon, eds., 6819, p. 681905, 2008.*

Author's Profile

Heerakar Rajini is presently pursuing final semester M. Tech in VLSI at Arjun College of Technology and Sciences, Batasingaram village, Hayathnagar, R.R. Dist, Hyderabad, Telangana, India.

J. LINGAIAH is presently working as Head of Department & Associate Professor in the department of Electronics and Communication Engineering in Arjun College of Technology and Sciences, Batasingaram Village, Hayathnagar, R.R. Dist, Telangana, India.

MOHAMMAD SAJEED PASHA is presently working as Application Engineer in Unistring Tech Solutions Pvt Ltd, Hyderabad, Telangana, India.