JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

FPGA BASED IMPLEMENTATION AND ANALYSIS OF MEGA FILTER FOR EFFECTIVE DENOISING OF IMAGES

[1]**Siddesh Kumar B** [2]**Shilpa S N**
[1]Lecturer, [2]Lecturer,
[1]Dept. of Electronics & Communication Engineering, Government Polytechnic, Channapatna, Ramanagara, Karnataka, India - 562 160
[2]Dept. of Electronics & Communication Engineering, Government Polytechnic, Ramanagara, Karnataka, India - 562 159

***Abstract* – MeGa filter design and implementation on FPGA has been proposed in the paper. The proposed MeGa filter is a hybrid filtering technique used to effectively suppress the combination of Gaussian and Impulse noise present in the images. The proposed filtering technique is a two-stage process. In the first level linear filtering through Gaussian filter of various masks is employed to eliminate the different intensity noises. At the second level a non-linear method of filtering the images using Median filter is implemented to filter impulse noise/salt and pepper noise. The size of the image used/data of the design proposed is 256*256 pixels. To achieve high speed of image processing, filter design is implemented on Field Programmable Gate Array (FPGA). The proposed work shows the implementation of MeGa filter design on FPGA and to run the design on Spartan 6E FPGA board.**

***Keywords:* Gaussian, Median, MeGa, FPGA, SAP.**

I. INTRODUCTION

Image processing is a wide area with tremendous application in our day to day life activities such as Biometric registrations, Industrial Automation, Automatic machine controlling [20]. Image processing is also being extensively used in the field of space, research and military applications [22].

In most of the real time images captured, different kinds of distortions such as noise may get added differences in the capturing hardware, limitations in the memory storages or error encountered during transmission or due various environmental factors. The variation in the intensity of the images might also reduce the quality of the images. The presence of noise and intensity variation in the image makes images difficult for further analysis and decision making .

Image noise may originate from various sources. While transferring images from one device to another through cable such as optical cable, the electrical to optical conversion is done at the transmitter side and optical to electrical conversion is performed at the receiver side. During this conversion and because of reflection of the signal in the cable while transferring, the noise may get added in the images [20]. Each step in conversion process is associated with random changes which contributes noise component to the image thereby altering the intensity of a given pixel [15]. As the image goes through multiple conversions before being processed noise gets added to the original image and the added noise may alter the original image content.

The main dominating noises in the digital image are Gaussian noise and SAP noise [23]. Because of the Gaussian noise, the edges of the objects in the image become dominant due to high frequency components [23]. The step of pre-processing is done to extract the image data and suppress the noises, such as Gaussian noise.

At present, most of the real time image processing systems are based on the standard median filter with little modifications [13,14]. The traditional method of Filtering has several shortcomings [10,11,12]. They can only filter one kind of noise of particular intensity level with certain or improved PSNR value. An effective filtering technique is needed to get rid of the combination of noises or varying noise intensity levels in images without reducing the quality. An effort has been made to design a hybrid or a combination of filtering technique to get rid of the combination of noises like Gaussian noise and the salt and pepper noise of varying noise intensity levels in images without reducing the quality.

The design implementation is carried out on FPGA Spartan 6E for large data to be processed.

To meet the real time implementation requirements such as filtering the images with multiple noises so that this technique is applicable in applications such as robot vision. To filter multiple noises of various intensity levels. Compared with the implementation in a PC based system, pipelined implementation on FPGA takes much less implementation time and can therefore be used for the mobile robot vision system which is very strict for the real-time performance of its vision system [3].

II PROPOSED METHODOLOGY AND BLOCK DIAGRAM OF MEGA FILTER DESIGN

Pre-processing is the first step in robotic vision. In pre-processing step the input image is filtered to remove noise. Noise consists of undesirable components which degrades image standard..

The figure 2.1 shows the block diagram of a Modified MeGa Filter. The noisy image of different intensity is applied as input to the Gaussian filter of 3 different masks. There is block called noise intensity, where the noisy intensity level is measured and corresponding decision is made to which filter to operate. Which of the three Gaussian filters is used depends on the image noise intensity. Then the median filter to remove SAP noise present after filtering using Gaussian filter. Finally the filtered image output is obtained.

The proposed design is different from the design in the literature survey[22] in the way in which two level of filtering is done on noisy image where in which, at first level Gaussian filtering is done based on noise intensity followed by median filtering to remove SAP as against the paper[22]

where mean and median filters are used to filter same kind of noise at two levels on particular intensity level. Where as in traditional method single filter with modified algorithm is used for filtering.

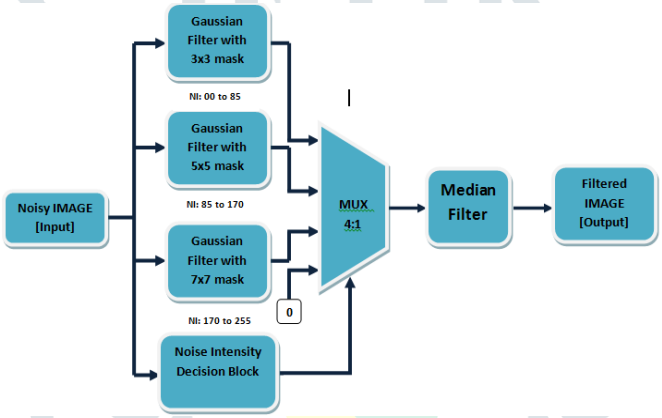


Fig.2.1 Block Diagram of proposed MeGa filter design

<u>Gaussian Filter Implementation and Method to find the Gaussian Kernel/Mask:</u>

The Gaussian filter is most suitable to remove Gaussian noise [15]. The one dimension Gaussian function is given by equation (1) [23] and the corresponding Impulse response curve is shown in figure 2.2.

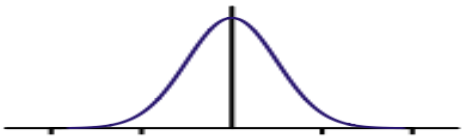


Fig 2.2 Impulse Response of Gaussian Filter

$$g(x,y) = \frac{1}{2\pi\sigma^2} e^{-\left(\frac{x^2+y^2}{2\sigma^2}\right)}$$ ----------Equation (1)

Where σ is standard deviation of Gaussian distribution

x is the distance from origin in the horizontal direction

y is the vertical distance from the origin

For σ=1, the impulse response is given in Equation (2).

$$g(x,y) = \frac{1}{2\pi} e^{-\left(\frac{x^2+y^2}{2}\right)}$$ ---------- Equation (2)

The values of x and y are ranging between -1 and 1 as shown in 3X3 matrix Table 2.1.

(-1,1)	(0,1)	(1,1)

(-1,0)	(0,0)	(1,0)
(-1,-1)	(0,-1)	(1,-1)

Table 2.1 x and y coordinates

The different combinations of x and y values of table 2.1 are substituted in equation (2) to obtain approximate numerical values yields Gaussian mask as given in table 2.2.[25]

g (-1,-1) = 0.0585≈0.0625=g(1,1) =g(-1,1)= g(1,-1)

g(0,1) = 0.096 ≈0.125=g(-1,0)= g(1,0)= g(0,-1)= g(0,1)

g (0,0) = 0.159 ≈ 0.25

(1/16)

1	2	1
2	4	2
1	2	1

1/273

1	4	7	4	1
4	16	25	16	4
7	26	41	26	7
4	16	26	16	4
1	4	7	4	1

1/1003

0	0	1	2	1	0	0
0	3	13	22	13	3	0
1	13	59	97	59	13	1
2	22	97	159	97	22	2
1	13	59	97	59	13	1
0	3	13	22	13	3	0
0	0	1	2	1	0	0

Table 2.2 Gaussian 3x3 mask,5x5 mask & 7x7 mask

To obtain Gaussian filtered image, first the image is divided into 3X3 overlapping window as shown in the figure 3.1. Where in which the block of nine pixels of original image are convolved with Gaussian kernel (table 2.2) of Size 3×3.

Median Filter Implementation

Median Filter is a simple and powerful non-linear filter which is based order statistics. It is easy to implement method of smoothing images. Median filter is used for reducing the amount of intensity variation between one pixel and the other pixel. In this filter, we do not replace the pixel value of image with the mean of all neighbouring pixel values, we replaces it with the median value. Then the median is calculated by first sorting all the pixel values into ascending order and then replace the pixel being calculated with the middle pixel value.

The method of median filter is articulated using Sort optimization

a(x,y) =med b[(x-i), (y-j)], Ikε W (1)

b(x, y) and a(x,y) represents the input image and the filtered output image respectively. W represents a 2D format, usually it is 3 × 3 or 5 × 5 in size. Here is an example of 3 X 3 moving window for median filter.

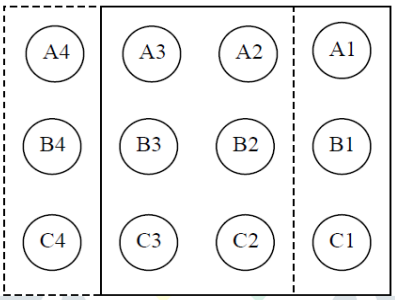


Fig.2.3 Considering the pixel values row-wise

According to the Median filter design sorting as to be performed on the 9 pixels under consideration and the middle value is used to replace the (2,2) pixel. Where in the algorithm of finding the median value reduces the number of comparisons to 9 thereby reducing the sorting time to 13 from 21 which greatly improves the systems efficiency[24].

The proposed algorithm is similar the image pre-processing block on FPGA. This algorithm works for a 3 X 3 filter structure. The lines n-1, n and n+1 saves the short term data. A three element FIFO memory structure can be used to store these three data. After processing of data present on line n , the data on n+1 is processed and continued so on.

III HARDWARE IMPLEMENTATION

The hardware implementation of the block diagram as shown in 2.1 is explained in detail in the following section.

Gaussian Filter implementation

In this section we have taken 3x3 moving window construction and the same idea is extended for 5x5 and 7x7 moving window. Next step is to construct a moving window, which results in 3x3 matrix of input image elements, so that this input matrix can be multiplied with the 3x3 Gaussian kernel. The calculation of the 3x3 Gaussian kernel is done in the following section.

Nine D-flip flops and two FIFOs are employed in the moving window. The D-flip flops stores the input pixels. The FIFO act as a dynamic memory array. The array size is 253 as depicted in figure 3.1. The image size is standard 256x256 is chosen for filtering.

The operation of the moving window is explained in detail below. Each pixel is 8 bits wide. The image consists of 256x256 pixels as shown in the below figure 3.1. The pixels are processed serially. The 8-bit pixel image is first fed to the flip flop D0. As the D-flip flop is a delay flip flop, it just passes the input with some pre-defined delay. In our work, the clock period is 10ns, i.e. each D- flip flop will pass the input after 10ns. The output of each D-flip flop can be stored as an intermediate signal, for further calculation. As there are D0 to D8 D-flip flops are there, S0 to S8 are the intermediate signals.

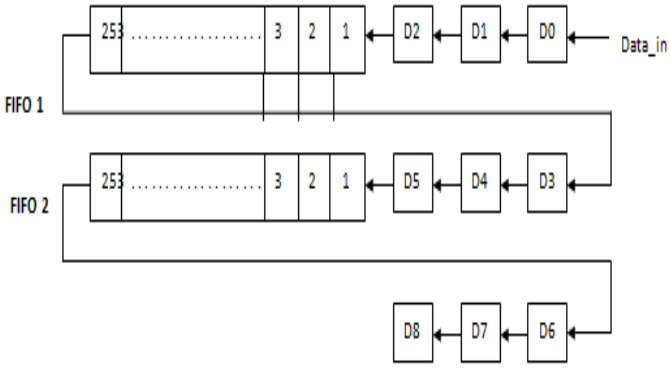


Fig 3.1. 3x3 Gaussian Moving Window Design

The figure 3.2 shown below is the hardware structure and the median filter model structure. The working operation is as follows: the initial step includes writing of data into the FIFO module of FPGA. when the last FIFO is in half full state, the median filter module working begins. The 3 FIFO data will be ordered in ordering comparator and fed to the next stage of comparators. Two D Flip flops are used to introduce two circle delays to distinguish input data order before being fed to second comparator. The output of the second comparator is fed to the median comparator to get the final output. The finally obtained median value is replaced for the (2,2)th pixel in the 3x3 window.

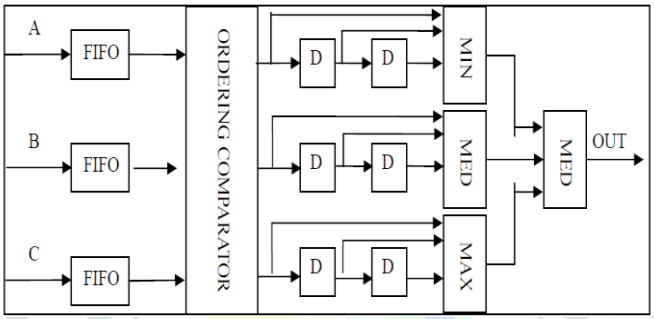


Fig 3.2. Hardware Structure of Median Filter Module

The maximum and minimum values of three data can be obtained by two comparisons and the median values is obtained by ordering followed by three comparisons. There are four different comparators available for ordering, finding minimum, finding maximum and finding median. The figure below shows hardware structure of the four comparators.

In the below figures 3.3a, 3.3b, 3.3c, 3.3d, the basic processing element is PE which compares the two data inputs. D flip-flop, introduces a single circle delay which is required for the process of synchronization. The proposed algorithm requires a two storage space to store temporary data and hence two flip flops are used. The figure shows the hardware structure of this.

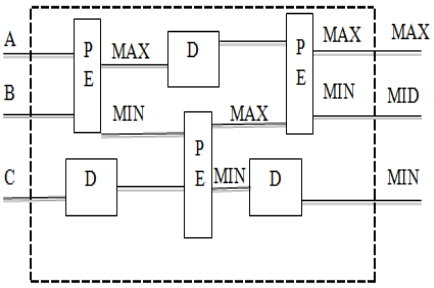


Fig 3.3a Ordering Comparator

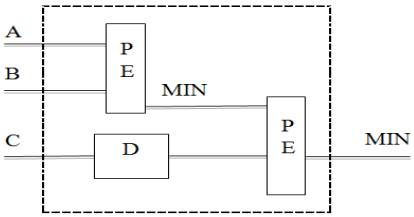


Fig 3.3b Minimum Comparator

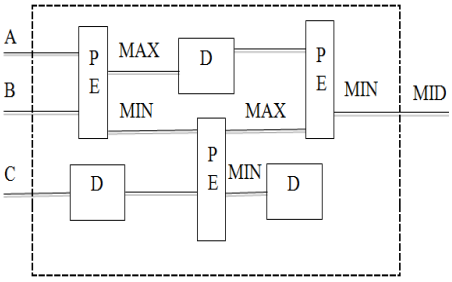


Fig 3.3c Median Comparator

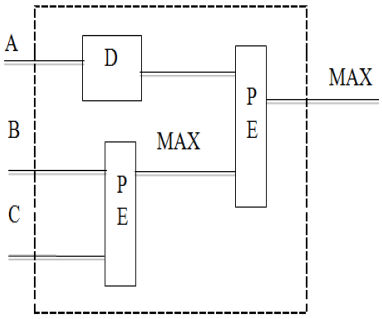


Fig 3.3d Maximum Comparator

Noise Intensity Module Implementation

First the Noisy Image is filtered by all the three types of Gaussian Filters and depending on the amount of noise intensity in the image , Image is calculated by decision block the MUX output is selected and further processed by median filter. The noise intensity is calculated by the following equation 3.2.1,

$$\mathbf{NI} = \frac{\sum_{i=0}^{m-1} Ai}{65536} \ldots\ldots \qquad (3.2.1.)$$

Where, NI is the Noise Intensity & Ai refers to the pixel value in the image.

Based on the noise intensity(NI) value MUX input is selected and taken into its output for further process.

a. If the NI value is between 0 & 85 then "00" of the MUX input ie. 3x3 Gaussian filtered image is allowed at its output to feed to median filter.

b. If the NI value is between 85 & 170 then "01" of the MUX input input ie. 5x5 Gaussian filtered image is allowed at its output to feed to median filter.

c. If the NI value is between 170 & 255 then "10" of the MUX input input ie. 7x7 Gaussian filtered image is allowed at its output to feed to median filter.

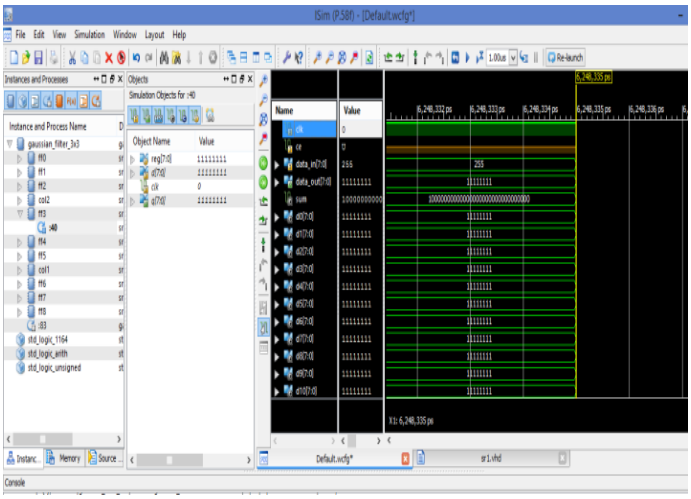
IV TESTING, RESULTS AND COMPARATIVE ANALYSIS

In the following section the various blocks of MeGa Filter are designed using VHDL language, tested for simulation results and also noisy images are filtered to get PSNR values for comparative analysis purpose. Then all the blocks are combined into a single Hybrid filter to obtain MeGa filter also the noisy images are filtered to obtain the PSNR values for comparative analysis.

SIMULATION RESULTS OF INDIVIDUAL BLOCKS

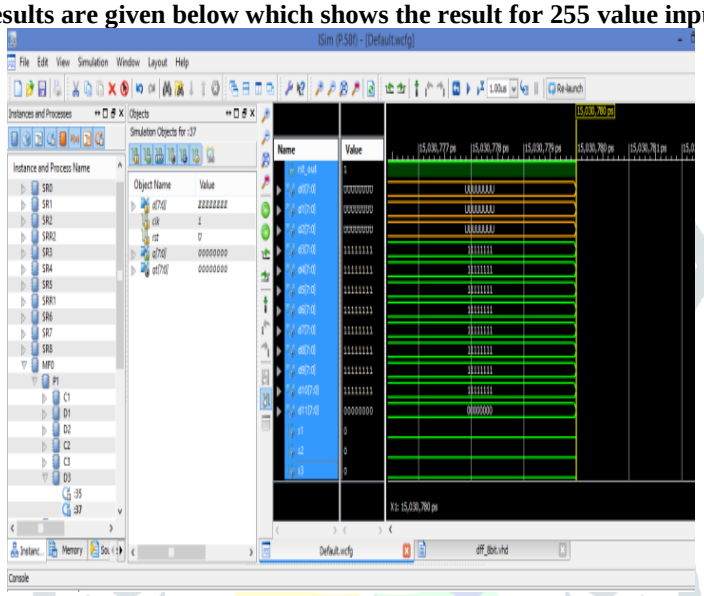
i. Gaussian Filter with 3x3 mask

Gaussian filter with 3x3 mask simulation results are given below which shows the result for 255 value input



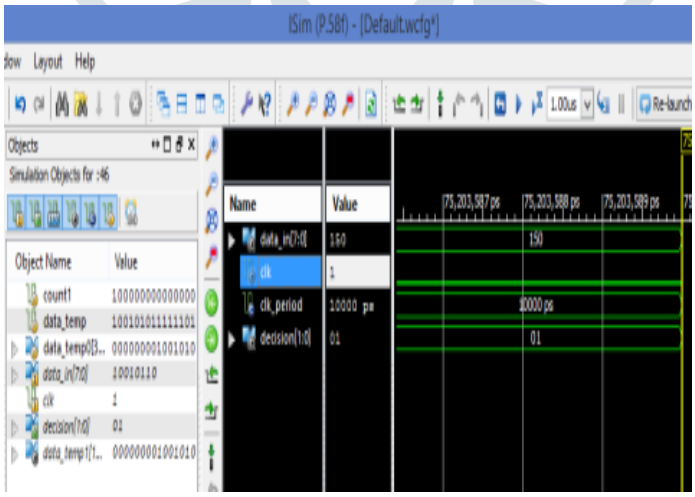
ii. Median Filter

Median filter simulation results are given below which shows the result for 255 value input



iii. Decision Block

Decision block simulation results are given below which is test for intensity level of 150 so option 2 is selected ie 5x5 mask Gaussian filter is selected



FILTERED IMAGE OUTPUT OF MeGa FILTER

COMPARATIVE ANALYSIS

In the following section PSNR values of Median filter, 3x3 Gaussian Filter and MeGa Filter are considered for comparison. The Table 4.1 shown below gives the PSNR values of filtered output gray scale images. For which input gray scale images having different noise intensity levels are considered for this comparative analysis. The analysis is carried on the hardware model designed. In the below Table 4.1 column 1 represents the images with different noise intensity levels.

i) 'SP01' represents gray scale image added with 01% Salt n Pepper noise .Similarly 'GN01' represents gray scale image added with 01% of Gaussian Noise. Also 'GNSP01' represents gray scale image added with 01% of each Gaussian noise followed by Salt n Pepper noise.

ii) 'SP04' represents gray scale image added with 04% Salt n Pepper noise .Similarly 'GN04' represents gray scale image added with 04% of Gaussian Noise. Also 'GNSP041' represents gray scale image added with 04% of each Gaussian noise followed by Salt n Pepper noise.

iii) 'SP10' represents gray scale image added with 10% Salt n Pepper noise .Similarly 'GN10'represents gray scale image added with 10% of Gaussian Noise. Also 'GNSP10' represents gray scale image added with 10% of each Gaussian noise followed by Salt n Pepper noise.

Comparison Based on PSNR values

Gray Scale images of varying Noise Intensity	PSNR VALUES		
	3x3 mask GAUSSIAN filter	MEDIAN Filter	MeGa Filter
SP01	68.52	60.98	**60.67**
GN01	68.74	61.12	**60.68**
GNSP01	68.49	61.51	**60.88**
SP04	67.57	61.13	**60.76**
GN04	68.39	61.1	**60.92**
GNSP04	65.12	60.62	**60.73**
SP10	66.34	61.03	**60.86**
GN10	66.62	60.64	**61.06**
GNSP10	64.39	61.37	**61.73**

Table 4.1 PSNR values obtained by 3 three filter under comparison for hardware model

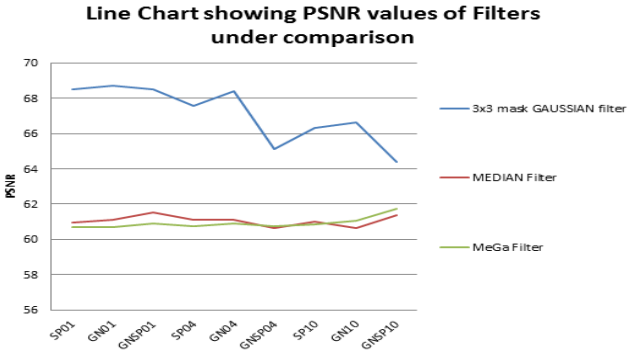


Fig 4.1. Line chart for plot of PSNR values for 3 different filters

Following are the observations made after the comparative analysis:

Gaussian filter PSNR values are good for salt n pepper, Gaussian noise and combination of both over Median filter and MeGa filter. However its performance reduces with the increase in the percentage of noise intensity levels. Also its performance degrades with the increase in noise intensity level for combined noises.

Even though Median filter is designed to remove salt n pepper noise effectively its PSNR values are less over Gaussian filter but comparatively more over MeGa filter. But its performance it randomly varying over different kind of noises and their related intensity levels. MeGa filter PSNR values are less compared to other two filter under comparison but its performance is consistently increasing with increased noise intensity levels and also with the combination of noises. Figure 4.1 represents the line graph and line graph of comparison of performance of MeGa filter over 3x3 Gaussian and Median filters.

Comparison Based on Synthesis Results

This section compares the synthesis report of MeGa filter module and the results of [13] and [14].

Tables 4.2 FPGA (Hardware) resource utilization of proposed MeGa filter

Sr No	Logic Utilization	Used	Available	Utilization
01	Number of Slice registers	1799	54576	3%
02	Number of fully used LUT-FF pairs	387	4418	8%
03	Number of slice LUTs	3006	27288	11%

MeGa Filter Synthesis Report

Sr. No.	FPGA Resource	Available	Used	Utilization
1.	Number of slice registers	18224	680	3%
2.	Fully used LUT-FF pairs	829	450	54%
3.	Number of Slice LUTs	9112	599	6%

Synthesis Report from reference 13 (Single median filter Implementation)

Table 4.2 Synthesis report of MeGa filter & Other filter under comparison

By observing above table 4.2 the number of fully used LUT –FF pairs is reduced at the cost 1% number slice LUTs are used more. The table tabulates resource utilization of the design on the FPGA board. Thus we can conclude that the proposed MeGa filter is uses less area compared to the existing median filters.

V CONCLUSION AND FUTURE WORK

The MeGa filter using Gaussian filter with three different masks and in combination with median filtering technique, accounts for the elimination of Gaussian and salt and pepper noise or combination of both. Depending on the amount of noise intensity, any one of the Gaussian masks that is that is 3x3, 5x5 and 7x7 is selected. The proposed algorithm is computationally efficient in removing salt n pepper noise from images. A new MeGa filter is useful for image restoration and elimination of salt and pepper noise is done while maintaining the information in the image.The proposed model has optimized area at the cost of speed.

The proposed MeGa filter can be extended in various directions. Improved results can be obtained by implementing various noise suppressing techniques for various kinds of noises, such as the random-valued impulse noise or combination three or more kind of noises also with higher levels of noise intensity. The proposed filter can be used in the preprocessing step of biometric recognition techniques like iris and face recognition. That improves performance of the biometric recognition.

REFERENCES

[1] J. J. Koenderink, "The structure of images," Biological Cybernetics, vol. 50, no. 5, pp. 363–370, August 1984.

[2] T. Lindeberg, "Scale-space for discrete signals," IEEE Transactions PAMI, vol. 12, no. 3, pp. 234–254, 1990. Scale-Space Theory in Computer Vision. Kluwer, 1994.

[3] Tony Lindeberg , "Feature detection with automatic scale selection," International Journal of Computer Vision, vol. 30, no. 2, pp. 79–116, 1998.

[4] A. Witkin, D. Terzopoulos, and M. Kass, "Signal matching through scale space," International Journal of Computer Vision, pp. 133–144, 1987.

[5] J. F. Canny, "Finding edges and lines in images," MIT. AI Lab., Tech. Rep. TR-720, 1983, masters Thesis.

[6] R. Deriche, "Fast algorithms for low-level vision," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 12, pp. 78–87, 1990.

[7] I. T. Young and L. J. van Vliet, "Recursive implementation of the gaussian filter," Signal Processing, vol. 44, pp. 139–151, 1995.

[8] R. A. Haddad and A. N.Akansu, "A class of fast gaussian binomial filters for speech and image processing," IEEE Transactions on Signal Processing, vol. 39, no. 3, pp. 723–727, 1991.

[9] J.-M. Geusebroek, A. W. M. Smeulders, and J. van de Weijer, "Fast anisotropic gauss filtering," IEEE Transactions on Image Processing, vol. 12, no. 8, pp. 938–943, 2003.

[10] F. Crow, "Summed-area tables for texture mapping," in SIGGRAPH '84: Proceedings of the 11th annual conference on Computer graphics and interactive techniques, 1984, pp. 207–212.

[11] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool, "SURF: Speeded Up Robust Features," Computer Vision and Image Understanding, vol. 110, no. 3, pp. 346–359, 2008.

[12] HassenSeddik ,Ezzedine Ben Braiek, "Efficient Noise Removing Based Optimized Smart Dynamic Guassian Filter", International Journel Of Computer Applications (0975-8887) Volume 51- No. 5, August 2012.

[13] S.S. Tavse, P.M. Jadhav, M.R. Ingle, "Optimized median filter implementation on FPGA including oft processor." International Journal of Emerging Technology and Advanced Engineering, Vol. 2, Issue 8, pp. 236- 239, August 2012.

[14] Bhavik Trivedi, Jayesh Popat and Kishan Govani "Optimized Implementation Of Median Filter Algorithm On Fpga", International Journal Of Emerging Technologies And Applications In Engineering, Technology And Sciences (Ij-Eta-Ets) Jan 14 – June 14.

[15] Gaussian filtering Technique, The University of Auckland, New zealand.

[16] Ian kuon and Jonathan Rose, "Measuring the gap between FPGAs and ASICS", IEEE transcation on Computer-Aided of Integrated Circuits and systems, 26(2): 203-215,2007

[17] Xilinx incorporated , virtex-6 Family overview, 2009.

[18] Matlab user guide, help navigator.2007.

[19] Tien-Ying kuo and Lin-ying chuong, "Fast global motion-compensated frame interpolator for very low-bit rate quality enhancement", IEEE International conference on Acoustics, Speech, and Signal processing, vol 3, page 111-113-16, April 2003.

[20] P. Viola and M. Jones, "Robust real-time object detection," in Second International Workshop on Statistical and Computational Theories of Vision Modeling, Learning, Computing, and Sampling, 2001, vancouver.

[21] Swamy TN ,Rashmi KM, Dr P Cyril prasanna Raj, Dr S L pinjare," FPGS implementation of efficient algorithm of image splitting for vedio streaming data", International journal of Engineering Research and Application(IJERA), vol 2 , issue 5, sep 2012.

[22] Sunkari Vinod Reddy and V. Srinivas "An fpga-based implementation of a mdbut filter for removal of noice in images". International Conference On Emerging Trends in Science Technology Engineering and Management 09th & 10th, October 2015.

[23] Priyanka Kamboj and Versha Rani "a brief study of various noise model and filtering techniques" Journal of Global Research in Computer Science, Volume 4, No. 4, April 2013.

[24] Sort Optimization Algorithm of Median Filtering Based on FPGA" Yan Lu, Ming Dai International Conference on Machine Vision and Human-machine Interface,2010.

[25]Digital Image Processing , Prof. P. K. Biswas, Department of Electronic & Electrical Communication Engineering, Indian Institute of Technology, Kharagpur.