FPGA Based Area Efficient Median Filtering for Removal of Salt-Pepper and Impulse Noises

G. Neeraja¹, S. Deepika²

¹PG Scholar, Dept of ECE(ES&VD), Malla Reddy Institute of Technology & Science, Hyderabad, AP-INDIA, E-mail: gudiminchi_neeraja@yahoo.co.in.
²Asst Prof, Dept of ECE, Malla Reddy Institute of Technology & Science, Hyderabad, AP-INDIA, E-mail: sivalenkadeepika@gmail.com.

Abstract: In Real time signal and image processing applications, it is 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. A special kind of filter called Median filter is used 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 follow a histogram-based implementation. Different directional median filtering methods for FPGA are proposed. For each of these techniques, four directions are processed at the same time. All techniques aim to decrease processing time, while also reducing the hardware resource utilization. The 3x3 window technique is implemented to filter the noise from images. The processing time requirements for the proposed methods are dependent on the size of the window and number of filtering directions considered. Directional median filtering has been implemented using cumulative histogram of samples in several directions. The filtered images are smoothed along the direction of the filtering window. The proposed architecture is simulated using Modelsim 6.2c and synthesized using Xilinx ISE 9.2c and it will be implemented on XC3S500e Spartan 3E FPGA board for hardware implementation and testing. The Xilinx Chip scope tool will be used to test the FPGA inside results while the logic is running on FPGA.

Keywords: Directional Median Filtering, Cumulative Histogram, Image Processing, VHDL, FPGA.

1. INTRODUCTION

The handling of digital images has become a subject of widespread interest in different areas such as medical, technological applications and many others. There are lots of examples where image processing helps to analyze, infer and make decisions. The main objective of image processing is to improve the quality of the images for human interpretation, or the perception of the machines independently. This project focuses on processing an image pixel by pixel and in modification of pixel neighborhoods and the transformation that can be applied to the whole image or only a partial region. The need to process the image in real time, which is time consuming, leads to this implementation in hardware level, which offers parallelism, and thus significantly reduces the processing time. FPGAs are increasingly used in modern imaging applications, image filtering, medical imaging, image compression, and wireless communication.

An image is an array or a matrix of square pixels arranged in the form of rows and columns. Also an image is defined as a two-dimensional function f(x, y), where ‘x’ and ‘y’ are spatial coordinates, and the amplitude of ‘f’ at any pair of coordinates (x, y) is called the intensity or gray level of the image at that point. When x, y and the amplitude values of f are all finite discrete quantities, then the image is called as a digital image. Digital image is composed of a finite number of elements, each of which has a particular location and value. The elements are called pixels. The field of Digital Image Processing refers to processing digital image by means of a digital computer.

The continuum from image processing to computer vision can be broken up into three processes, they are

- Low-level
- Mid-level and
- High-level processes

Low-level process involves primitive operations such as image processing to reduce noise, contrast enhancement, compression and image sharpening. A low-level process is characterized by the fact that both its inputs and outputs are images. Mid-level process on images involves tasks such as
segmentation, description of those objects to reduce them to a form suitable for computer processing and classification of individual objects. A mid-level process is characterized by the fact that its inputs generally are images but its outputs are attributes extracted from those images. High-level processing involves “Making sense” of an ensemble of recognized objects, as in image analysis and at the far end of the continuum performing the cognitive functions normally associated with human vision.

<table>
<thead>
<tr>
<th>Low Level Process</th>
<th>Mid Level Process</th>
<th>High Level Process</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input:</strong> Image</td>
<td><strong>Input:</strong> Image</td>
<td><strong>Input:</strong> Attributes</td>
</tr>
<tr>
<td><strong>Output:</strong> Image</td>
<td><strong>Output:</strong> Attributes</td>
<td><strong>Output:</strong> Understanding</td>
</tr>
<tr>
<td><strong>Examples:</strong> Noise removal, image sharpening</td>
<td><strong>Examples:</strong> Object recognition, segmentation</td>
<td><strong>Examples:</strong> Scene understanding, autonomous navigation</td>
</tr>
</tbody>
</table>

Fig.1. Block diagram representing the various levels in image processing.

Gray scale image: A gray scale digital image is an image in which the value of each pixel is a single sample, that is, it carries only intensity information. Images of this sort are composed exclusively of shades of gray, varying from black at the weakest intensity to white at the strongest intensity. Gray scale images have many shades of gray in between. Image processing is a type of signal processing in which the image or information regarding image is fed as an input signal and various operations are performed on it. The various operations performed on it can be used on a host of applications such as Image filtering, medical imaging, Wireless communication, image compression, computer vision etc. Some of the most common operations on an image that come under the image processing are image scaling, converting between various color format, image rotation, removing noise, adding noise, filtering, blurring, edge detection and contour detection. Some combination of these algorithms is used in almost all image processing applications.

Modern digital technology has made it possible to manipulate multi-dimensional signals with systems that range from simple digital circuits to advanced parallel computers. The goal of this manipulation can be divided into three categories:
- Image Processing (image in -> image out)
- Image Analysis (image in -> measurements out)
- Image Understanding (image in -> high-level description out)

An image is an array, or a matrix, of square pixels (picture elements) arranged in Columns and rows. In a (8-bit) gray scale image each picture element has an assigned intensity that ranges from 0 to 255. A gray scale image is what people normally call a black and white image, but the name emphasizes that such an image will also include many shades of gray.

II. MEDIAN FILTERING

One or two-dimensional median filtering is a non-linear operation which is known for preserving sharp edges in signals or images. It is particularly effective in removing non-Gaussian, impulsive noise. The standard median filter is characterized by the following method: the output value of the median filter is that input sample value, which is located in the centre of the list of ordered samples. The sampling window is shifted through the full data window. The median filter is a non-linear tool, while the average filter is a linear one. In smooth, uniform areas of the image, the median and the average will differ by very little. The median filter removes noise, while the average filter just spreads it around evenly. The performance of median filter is particularly better for removing impulse noise than average filter.

Fig.2. Image - an array or a matrix of pixels arranged in columns and rows.

Image processing is referred to processing of a 2D picture by a computer. Basic definitions. An image defined in the “real world” is considered to be a function of two real variables, for example, a(x, y) with a as the amplitude (e.g. brightness) of the image at the real coordinate position (x, y).
FPGA Based Area Efficient Median Filtering for Removal of Salt-Pepper and Impulse Noises

As Fig.3 shown above are the original image and the same image after it has been corrupted by impulse noise at 10%. This means that 10% of its pixels were replaced by full white pixels. Also shown are the median filtering results using 3x3 and 5x5 windows; three (3) iterations of 3x3 median filter applied to the noisy image; and finally for comparison, the result when applying a 5x5 mean filter to the noisy image. In median filtering, the neighbouring pixels are ranked according to brightness (intensity) and the median value becomes the new value for the central pixel. Median filters can do an excellent job of rejecting certain types of noise, in particular, “shot” or impulse noise in which some individual pixels have extreme values. In the median filtering operation, the pixel values in the neighbourhood window are ranked according to intensity, and the middle value (the median) becomes the output value for the pixel under evaluation.

Median in statistic means literally the value in the middle (10,5,5,3,3,3,3,2,2 ) 9 items so half way down item 5 that’s the median. Most scanned images contain noise caused by the scanning method (sensor and its calibration -electrical components, radio frequency spikes) this noise may look like dots of black and white. Median filter helps us by erasing the black dots, called the Pepper, and it also fills in white holes in the image, called Salt. Median filter is better in preserving sharp edges. The median value is much like neighborhood pixels and will not affect the other pixels significantly.

As Fig.4 shown above is noise removal using Median filter.

Image processing is a very important field within industrial automation, and more concretely, in the automated visual inspection. For example, automatically analyzing predetermined features of manufactured parts on an assembly line to look for defects and process variations. In these applications, the main challenge normally is the requirement of real-time results. On the other hand, in many of these applications, the acquired images must pass through a stage of image pre-processing in order to remove distracting and useless information from the images. For example, the existence of impulsive noise in the images is one of the most habitual problems. 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. One of the disadvantages of linear filters, such as the moving average filter, when used to de-noise the data, is that they not only smooth the noise, but also smooth the sudden and sharp transitions that were present in the original data, such as edges in images. Moreover, they are not as efficient as the median filters in removing certain types of noise, such as impulsive noise. Although median filters do not blur the edges as much as the linear filters do, because they still possess smoothing characteristics, as the size of the filter increases, there may be significant image blurring.

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. Random valued noise pixels take any random value, which is more difficult to remove than the salt and pepper noise. If \( p \) and \( q \) are the probabilities of occurrences of 255 and 0, where \( 0 \leq p, q \leq 1 \) and \( p \) and \( q \) can be equal or different, a pixel may be replaced by 255 with a probability \( p \) and by 0 with a probability \( q \). The median of a given sequence is given by sorting the sequence and choosing the middle value from the sorted sequence. If there are odd numbers of elements in a sequence, then the median is the middle element in the sorted list. If there is even number of elements then the median is given by the arithmetic mean of the two middle elements in the sorted sequence.

In image processing, the 2D filtering operation is performed by sliding the window along all the rows and columns of the image until all the pixels are covered by the window. The filtering is done by sliding the window across the image, sorting all the pixels in the window, which consists of a centre pixel and the neighborhood pixels, and then replacing the central pixel with the median intensity of the window. Since salt and pepper noise pixels take only either the maximum or the minimum possible value and the result of a median filter excludes the extreme value, median filtering provides a good reduction of the salt and pepper noise. Median filter is the nonlinear filter more used to remove the impulsive noise from an image. Furthermore, it is a more robust method than the traditional linear filtering, because it preserves the sharp edges. Median filter is a spatial filtering operation, so it uses a 2-D mask that is applied to each pixel in the input image. To apply the mask means to centre it in a pixel, evaluating the covered pixel brightness and determining which brightness value is the median value. Figure presents the concept of spatial filtering based on a 3x3 mask, where I is the input image and O is the output image.

**A. Algorithm**

The main idea of the median filter is to run through the signal entry by entry, replacing each entry with the median of neighboring entries. The pattern of neighbors is called the “window”, which slides, entry by entry, over the entire signal. For 1D signal, the most obvious window is just the first few preceding and following entries, whereas for 2D (or higher-dimensional) signals such as images, more complex window patterns are possible (such as "box" or "cross" patterns). Note that if the window has an odd number of entries, then the median is simple to define: it is just the middle value after all the entries in the window are sorted numerically. For an even number of entries, there is more than one possible median.

**B. 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 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.

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 bins which has a value greater than or equal to the median index is taken as the median value of the window.

For example, the median of 3x3 windows using cumulative histogram method is found as follows: 156, 89, 75, 190, 204, 89, 89, 75, and 255. All the bins greater than and equal to the these intensities are increased by one every time the number is encountered, i.e., all the bins above 155th bin are increased by one, when the value 89 comes-in the values of all the bins above 88th bin are increased by 1(they are incremented 2 more times in this sequence) and so on for the rest of the sequence. The median index of a 3x3 window is 5, so counting the bin values to find the first bin which has the value equal to the median index gives 89 as the median value of the given sequence.

III. IMPLEMENTATION

A. Existing Method

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. The median filtering methods presented as follow a histogram-based implementation similar to the one proposed. Histogram represents the pixel intensity distribution in an image, and hence, requires many as bins as the number of gray levels in the image. For example, for a gray scale image having 256 levels of gray, 256 bins are required to build the histogram. The cumulative histogram indicates how many pixels with intensity equal to or less than the pixel intensity corresponding to a particular bin are present in an image.

If cumulative histogram is used, then the median is found at the bin where the corresponding cumulative histogram value is more than or equal to (N+1)/2, while the cumulative histogram at the immediately previous bin has a corresponding value less than (N+1)/2. There are also nonlinear neighborhood operations that can be performed for the purpose of noise reduction that can do a better job of preserving edges than simple smoothing filters. In median filtering, the neighboring pixels are ranked according to brightness (intensity) and the median value becomes the new value for the central pixel. Median filters can do an excellent job of rejecting certain types of noise, in particular, “shot” or impulse noise in which some individual pixels have extreme values. In the median filtering operation, the pixel values in the neighborhood window are ranked according to Intensity and the middle value (the median) becomes the output value for the pixel under evaluation. The histogram is implemented by instantiating an array of registers 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.

B. Proposed Method

The main block diagram of median filter which consists of the modules:

- Pipo shift registers (Accumulator).
- Sorting Network.
Comparators.
- 2-D matrix.
- Priority encoder.

Each node consists of a register, an incremented, and a comparator. Each node has two inputs and one output: an enable input, a median index input, and the comparator output. The value of the bin after every increment operation is stored in register, while the comparator checks whether the register value is equal to or greater than the median index input provided. The enable input to bin node determines whether the register value has to be incremented or not. A bin value is incremented if the enable input is "1".

![Fig.7. Block Diagram of Median Filter.](image)

The median computation is performed by the mask at each one of the pixels of matrix IR. It is done in three steps: firstly, the pixels of the mask are sorted in a column by column sequence, then row by row and finally along to the diagonal elements. After that sorting task is achieved, the central element (median) of the mask is picked out of IR and stored in the matrix IF (the filtered image). An illustrative description for the median algorithm.

![Fig.8. Graphical Description for the Median Algorithm](image)

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. 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.9. PIPO Shift Registers](image)
Today, high speed bi-directional universal type Shift Registers such as the TTL 74LS194, 74LS195 or the CMOS 4035 are available as a 4-bit multi-function devices that can be used in serial-serial, shift left, shift right, serial-parallel, parallel-serial, and as a parallel-parallel Data Registers, hence the name "Universal". Shift registers are a type of sequential logic circuit, mainly for storage of digital data. They are a group of flip-flops connected in a chain so that the output from one flip-flop becomes the input of the next flip-flop. Most of the registers possess no characteristic internal sequence of states. All flip-flops are driven by a common clock, and all are set or reset simultaneously. Here we have nine PIPO shift registers which are storing the data like D FLIPFLOP.

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 (Digital Comparator)
   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. DC compares only the logic levels of one number's binary bits while the later compares two potential differences each with respect to a common ground potential. DC made from the digital logic gates while the later made from an operation amplifier. Find whether these are equal or one greater than other or vice versa.

4. 2-D Matrices
   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. Median filtering is usually based on sorting algorithms. Utilizes the cumulative histogram. Histogram-based median filtering requires as many counters as the number of intensity levels in an image. On an FPGA, the histogram is implemented by instantiating an array of registers, 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. Each node consists of a register, an incremented and a comparator. Each node has two inputs and one output: an enable input, a median index input, and the comparator output.

   The value of the bin after every increment operation is stored in register, while the comparator checks whether the register value is equal to or greater than the median index input provided. The enable input to bin node determines whether the register value has to be incremented or not. A bin value is incremented if the enable input is 1. Otherwise the bin value is not incremented. It can be observed that if an incoming pixel has a value \(n\), then the histogram at the \(n\)-th 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. Here uses \(256 \times 256\)-bit codes saved in 8 BRAMS, for updating the cumulative histogram. Since the maximum bus width of a FPGA block RAM is 36-bits, without considering the parity bits, only 9Kb of the 18Kb space available in each BRAM is used. To reduce the BRAM utilization, only 32 \(\times\) 32-bit codes are stored in the BRAM. This requires only 1 BRAM. The remaining 224-bits are saved in registers and are assigned according to the first 3-bits of input sample. In another approach, 32 ‘1’s are stored in registers and then all 32 bits are right-shifted at once by the value equal to last 5 bits of the 8-bit input pixel.

   This eliminates the use of BRAM for saving bit codes. The remaining 224-bits are obtained in a similar. These implementations simultaneously calculate several medians using several directional windows, while minimizing the resource utilization and the number of clock cycles required for processing. It has been shown in the literature that using several directions for median filtering could be more effective in identifying and removing impulsive noise compared to traditional median filtering. In this project in 3x3 matrixes we are finding the median for the sequence and stored in Bram like as rof_buffer.

5. Priority Encoder
   A priority Encoder is a circuit or algorithm that compresses multiple binary into smaller number of outputs. The output of 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 the median value which is middle value of 3x3 windows obtained after comparing and sorting.

C. Median Filter Implementation

By observing the above flow we can understand the median filter block. In our project the median filtering is used to reduce the noise of the image. Mainly the salt and pepper noise will be removed. When we know about the different noises occurred in an image salt and pepper noise is the critical one compared to the random noise. So here we are concentrating on the salt and pepper noise of the image. If we remove this noise we can get more clarity in the image.

IV. RESULTS

In this section, we present our simulation results to evaluate the proposed Median filter. MATLAB is used to view the image. Following completion of these steps, a simulation executable will be created which will allow you to run the simulation in the Model Sim GUI. In this simulation step we will launch the Model Sim Graphical User Interface by running the simulation executable which was generated by the MATLAB in the previous section, “Building the Simulation Executable”. After this step is complete, you will be able to use the MATLAB, MODEL Sim GUI to explore the design which is described in Appendix. Link for Model Sim offers flexibility in how we start and control an HDL model test bench or component session with MATLAB. Chip scope is logic analyzer used to verify the signals running inside FPGA. Input is original coin image, then “salt and pepper” of noise density 0.01 is added to the image using command. The noise in image is filtered by using 3median filter. The output image obtained is noise free image.

A. Input Images

Fig.10. Original Image

Fig.11. Salt and Pepper Noisy Image

B. Output Image

Fig.12. Noisy Image filtered by 3 by 3 Averaging Filter

Fig.13. Noisy Image filtered by 3 by 3 Median Filter

C. Simulation Waveform for Sorting

The Model Sim simulation waveforms for the modules performing sorting and calculating median value are shown below.
1. Waveform Description
Rst is used to clear data that means data has to start from initial state. All signals starts from next rising edge of the clock.
Clk is used for the synchronization.
Index_in is the input signal passing by using a counter. The counter length is 32 Bit.
Mag_in is the input signal by passing a counter. The data is randomly passing the image pixel values. An image pixel has the gray levels and having the values. Here we have to find the median of every pixel in the image. We have to remove the noises like salt and pepper noise.
M_2 is the signal having the 9 bit length but as per median finding we have to set to five only.
Index_out is the output of the indexes like counter. It has the address like counter of 5 bits length.
Mag_val_array is the signal which is the array of vector storing the signals in vector format. Here the matrix form of the 3x3 is updating.
Mag_out is the signal which is the median output of the mag_val_array signal. Here finding the median and middle value is selected that is comparing the comp_vector with the median value.
Comp_vec is the signal which is comparator vector. Here we are comparing the input signal.
Rof_out is the final output of the median filter here we are removing the maximum and minimum intensity numbers like salt and pepper noises.
Finally removed the salt and pepper noise from the image intensity levels by using directional median filtering.
V. CONCLUSION AND FUTURESCOPE

This project work deals with extending an existing cumulative histogram based median filtering technique to directional median filtering. All implementations were designed for Spartan 3E FPGA. The Spartan 3E device runs at 50 MHz. FPGA based implementation results in high speed processing hence the proposed architecture can work for wideband signals. The window technique implemented for processing of pixels results in less processing time and area. For all implementations it has been assumed that memory can be accessed only once during one clock cycle. This assumption is made to ensure that the bandwidth for reading and storing image data is kept low. At the same time, this may also be suitable for on-chip designs through utilization of BRAM, as opposed to flash or other external memory, for storing intermediate processing results, such as histogram values. The processing time requirements for the proposed methods are dependent on the size of the window and number of filtering directions considered. Spartan 3E the resource utilization was less than 30%. Moreover, since the number of BRAMs in Spartan 3E is 20, smaller images were considered for implementation. An approach for reconfigurable hardware and implemented a median filter on Xilinx FPGA XC3S500E. The implementation was tested with images ranging from 4x4 up to 128x128. The design offers implementation of median filter with various window sizes. The results indicate improvement and speeds compare to offline methods.

Future Scope:
Number of clock cycles per pixel can be reduced further by implementing all methods of directional median filtering within the same FPGA chip. The other image processing operations like edge filtering and segmentation can be integrated in same software which offers a real time image processor chip for medical applications.

VI. REFERENCES


