OPTIMIZED IMPLEMENTATION OF MEDIAN FILTER ALGORITHM ON FPGA

1BHAVIK TRIVEDI, 2JAYESH POPAT, 3KISHAN GOVANI

1PG Student, Marwadi Education Foundation’s Group of Institutions, GTU, Rajkot.
2Asst. Prof., Marwadi Education Foundation’s Group of Institutions, GTU, Rajkot.
3Asst. Prof., Darshan Institute of Engineering & Technology, GTU, Rajkot.

1bhavik.trivedi45@gmail.com, 2jayesh.popat@marwadieducation.edu.in, 3govanikishan@gmail.com

ABSTRACT—The median filter is a mixed filter which removes spike noise (or an impulse noise) from a noisy image while preserving sharp edges. For this purpose a double derivative filter or Laplace filter can be used which acts as a noise detector. Both filter generally used median filter for performing operation. The median filter technique, the eight direct neighbors and centre point of a sliding 3-by-3 array are examined. Optimized approach can be consumes less hardware and power. Different logics for majority bit evaluation and bit sorting circuit can be applied and simulate in efficient manner. Optimized implementation of median filter algorithm analyzed for hardware (FPGA) resource utilization and power analysis.

Keywords—Selective; Median Filter; Mixed Filter; Impulse Noise; Double Derivative Filter; Laplace Filter; Majority Bit Evaluation; FPGA; VHDL.

I. INTRODUCTION
Image processing is a wide area with tremendous applications including our everyday life such as medicines, image authentication, automated industry inspection, automated control equipments and widely used in space exploration, military surveillance etc. In most of cases captured images from image sensors or camera are affected by various types of noises. The impulse noise is among one of the most frequently referred type of noise perceived in captured images.
This impulse noise, commonly also known as salt & pepper noise, is caused by malicious pixels in camera sensors and in images, absurd memory locations in hardware, or errors in the data transmission and in many other forms affected by environment conditions.
Median filtering is considered a suitable method to remove impulse noises from images. This non-linear technique is a good substitute to linear filtering as it can efficiently subdue impulse noise while preserving edge information. The median filter operates for each pixel of the image and ensures that it fits with the pixels around it. It filters out samples that are not representative of their surroundings; in other words the impulses. Therefore, it is very useful in filtering out missing or damaged pixels of the image.
Median filter has complexity to implement in hardware, as large amount of data involved in representing image information in digital format. General purpose processor as an implementation option is easier to implement on but not time-efficient due to additional constraints on memory, I/O bandwidth and other peripheral devices. Full custom hardware designs like Application Specific Integrated Circuits (ASICs) provide the highest speed to application but at the same time they have very less scope for flexibility [1].
The fast growing demand for System on Chip (SOC) design requires much more developed. There is a heavy demand for reducing power factors with same functioning capabilities. Most of these products include embedded microprocessors, Digital Signal Processors (DSPs) and ASICs.
Therefore low power design of any VLSI design is now a days a challenging task. So current need to design any application has low power consumption.
One more task according to current scenario is that design should be having less hardware with same functionality potential. It is possible due to advance research and hardware development in VLSI area.
FPGA has been chosen because of its various features are advantageous to design and develop any application. FPGAs are reconfigurable devices, which enables simplified debugging and verification and rapid prototyping. Its parallel processing characteristic increases the speed of implementation [4].

In next section, basic of median filter is given. Proposed architecture reviews are shown based on our work. Simulation results are discussed to justify work. Conclusion is drawn with reference of literature. Further future work is also depicted.

II. MEDIAN FILTER

Median filter is a spatial filtering operation, so it uses a 2-D mask that is applied to each pixel in the input image. The median value is determined by placing the brightness in ascending order and selecting the centre value [5].

The obtained median value will be the value for that pixel in the output image. Figure 1 shows an example [2].

![Fig. 1: Application of the median filter](image)

We use 3x3 window in median filter. Bigger size in window 5x5, 7x7, etc. also possible but usually give lesser sharp edge than of 3x3 window median filter.

III. MATLAB code algorithm:

Step 1: Read an input image

Step 2: Define 3x3 array for 3x3 window to take 9 corresponding pixels from input image.

Step 3: In this step, all three rows scan in 3x3 window will find maximum, median and minimum values from each rows and will arrange as given sequence.

Step 4: Similarly all three columns then scan for finding maximum, median and minimum values and arrangement.

Step 5: Now window will find same 3 values from all scanned maximum, median and minimum value. Principle diagonal of 3x3 square window to find median from those three values.

Step 6: Similarly in other diagonal all three values scan except a centre value in that diagonal replace by result of previous step.

Step 7: Now this 3x3 window continues to scan input image until all pixels of image scanned.

Step 8: Finally, we get output image without impulse noise by using median filter.

Figure 2 & 3 shows flowchart of our MATLAB code. It gives us one of the efficient ways to get median value from 3x3 window.
IV. Flowchart of Developed MATLAB code for median filter

In Figure 4, steps of scanning input image by 3x3 window of median filter is shown. Here, (d) shows that one of the value being scanned is output of (c), that means output of (c) (median value from n1, n5 & n9), will scan with n7 & n9. Here in (a), (b) & (c), it is assumed that after sorting values of nine pixel values, all values’ positions are unchanged, for illustration purpose.

V. SIMULATION RESULTS
We get following result of simulation from our developed MATLAB code, as shown in Figures 5 & 6.
Figure 5 shows distorted image with 70% noise density of salt & pepper noise. Noisy image used to simulate design using MATLAB code. Figure 6 shows output image which is smooth and noise free as our design goal.

VI. VHDL IMPLEMENTATION OF DEVELOPED MATLAB CODE

We developed design in VHDL platform as per algorithm implemented in MATLAB code. A Hardware synthesis carried out on Spartan6 development board -XC6SLX16 device.

Hardware utilization Results are as shown below:

<table>
<thead>
<tr>
<th>Sr. No.</th>
<th>FPGA Resource</th>
<th>Available</th>
<th>Used</th>
<th>Utilization</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Number of slice registers</td>
<td>18224</td>
<td>680</td>
<td>3%</td>
</tr>
<tr>
<td>2</td>
<td>Fully used LUT-FF pairs</td>
<td>829</td>
<td>450</td>
<td>54%</td>
</tr>
<tr>
<td>3</td>
<td>Number of Slice LUTs</td>
<td>9112</td>
<td>599</td>
<td>6%</td>
</tr>
<tr>
<td>4</td>
<td>Number of bonded IOBs</td>
<td>232</td>
<td>20</td>
<td>8%</td>
</tr>
</tbody>
</table>

Table 1 – FPGA (Hardware) resource utilization of optimized implementation

<table>
<thead>
<tr>
<th>Sr. No.</th>
<th>FPGA Resource</th>
<th>Available</th>
<th>Used</th>
<th>Utilization</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Number of slice registers</td>
<td>54576</td>
<td>3874</td>
<td>7%</td>
</tr>
<tr>
<td>2</td>
<td>Fully used LUT-FF pairs</td>
<td>18,224</td>
<td>3965</td>
<td>21%</td>
</tr>
<tr>
<td>3</td>
<td>Number of Slice LUTs</td>
<td>1710</td>
<td>1567</td>
<td>91%</td>
</tr>
<tr>
<td>4</td>
<td>Number of bonded IOBs</td>
<td>218</td>
<td>112</td>
<td>51%</td>
</tr>
</tbody>
</table>

Table 2- FPGA (Hardware) resource utilization of reference design

Table-1 & Table-2 show FPGA implementation result. Comparative analysis on conclude better results of optimized implementation of median filter.

VII. CONCLUSION

As work shown in section 5 & 6 optimized implementation of median filter proves better result in simulation part on MATLAB and better in hardware (FPGA) resource utilization point of view which is also better for less power consumption.

REFERENCES


