INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH & DEVELOPMENT

ISSN 2278 – 0211 (Online)

Denoising Method for Removal of Impulse Noise Present in Images

D. Devasena
AP (Sr.G), Sri Ramakrishna Engineering College, Coimbatore, Tamil Nadu, India
A.Yuvaraj
Student, Sri Ramakrishna Engineering College, Coimbatore, Tamil Nadu, India
R.Buvnesh
Student, Sri Ramakrishna Engineering College, Coimbatore, Tamil Nadu, India
G. K. Shrikanth
Student, Sri Ramakrishna Engineering College, Coimbatore, Tamil Nadu, India

Abstract:
Images are often corrupted by impulse noise in the procedures of image acquisition and transmission. The efficient impulse noise detector is used to detect the noising pixels. Once the noisy pixel is identified, its value is replaced by a linear combination of its original value and median of its local window. Thus, a filter is used according to the signal and noise characteristics while preserving edges in images. In this project, an extension of a bilateral filter that can additionally remove impulse noise and an edge preserving filter to reconstruct the intensity values of noisy pixel. The algorithm involves mainly detection of the noise and classifying it based on an efficient detector. Based on the detection, the proposed filter is applied for noise removal. Especially, the proposed method can preserve edges very well while removing impulse noise. Since our algorithm is algorithmically simple, it is very suitable to be applied to many real-time applications. To achieve the goal of low cost, MATLAB tool is Proposed.

Keywords: *Image Denoising , image restoration , impulse detector, bilateral filter, edge preserving filter*

1. Introduction

Image processing is widely used in many fields, such as medical imaging, scanning techniques, printing skills, license plate recognition, face recognition, and so on. In general, images are often corrupted by impulse noise in the procedures of image acquisition and transmission. The noise may seriously affect the performance of image processing techniques.
Hence, an efficient denoising technique becomes a very important issue in
image processing. Removing noise may seem simple, but to preserve the image features along with noise removal is a tedious one. Various approaches are present for eliminating noise. Filters can classify into two types such that linear and non-linear. Linear filters are computationally easy to implement, but they often result in blurring of images. This disadvantage tends us to focus on non-linear filter though acknowledging the fact they are mathematically complex to implement.
There are many types of non-linear filters and each has its own operating mode. Here, I have considered the possibility of modifying only those pixels that are noise affected rather than changing the image as in linear filters.

2. Objective of the Project

The main objective of our project is image denoising, the project is MATLAB based denoise the impulse noise image. The MATLAB software finds major application in the field creating program. This helps the instrumentation engineer to understand the conditions of the process.
The main objective includes developing a program with functions which can be user friendly.

3. Need of Matlab

MATLAB is a very powerful, high level language. It is also very easy to use. It comes with a wealth of libraries and toolboxes that you can use directly, so that you don't need to program low level functions. It enables us to display very easily results on graphs and images. To get started with it, you need to understand how to manipulate and represent data, how to find information about the available functions and how to create scripts and functions to generate programs.

4. Software Description

4.1. Tool Box

Image Processing Toolbox is a collection of functions that extend the capability of the MATLAB numeric computing environment. The toolbox supports a wide range of image processing operations, including

1. Spatial image transformations
2. Morphological operations
3. Neighborhood and block operations
4. Linear filtering and filter design
5. Transforms
6. Image analysis and enhancement
7. Image registration
8. Deblurring
9. Region of interest operations

Many of the toolbox functions are MATLAB M-files, a series of MATLAB statements that implement specialized image processing algorithms.

In MATLAB tool, the capabilities of Image Processing Toolbox by writing your own M-files, or by using the toolbox in combination with other toolboxes, such as Signal Processing Toolbox and Wavelet Toolbox can be extended.

Image processing consists of a wide variety of techniques and mathematical tools to process an input image. An image is processed as soon as we start extracting data from it. The data of interest in object recognition systems are those related to the object under investigation. An image usually goes through some enhancement steps, in order to improve the extractability of interesting data and subside other data. Extensive research has been carried out in the area of image processing over the last 30 years.

Image processing has a wide area of applications. Some of the important areas of application are business, medicine, military, and automation. Image processing has been defined as a wide variety of techniques that includes coding, filtering, enhancement, restoration registration, and analysis.

In many applications, such as the recognition of three-dimensional objects, image processing and pattern recognition are not separate disciplines. Pattern recognition has been defined as a process of extracting features and classifying objects. In every three-dimensional (3-D) object recognition system there are units for image processing and there are others for pattern recognition.

4.2. Image Analysis

Image analysis accepts a digital image as input and produces data or a report of some type. The produced data may be the features that represent the object or objects in the input image.

To produce such features, different processes must be performed that include segmentation, boundary extraction, silhouette extraction, and feature extraction. The produced features may be quantitative measures, such as moment invariants, and Fourier descriptors, or even symbols, such as regular geometrical primitives.

5. Image Denoising

A new framework for removing impulse noise from images is presented in which the nature of the filtering operation is conditioned on a state variable defined as the output of a classifier that operates on the differences between the input pixel and the remaining rank-ordered pixels in a sliding window. As part of this framework, several algorithms are examined, each of which is applicable to fixed and random-valued impulse noise models.

First, a simple two-state approach is described in which the algorithm switches between the output of an identity filter and a rank-ordered mean (ROM) filter. The technique achieves an excellent tradeoff between noise suppression and detail preservation with little increase in computational complexity over the simple median filter. For a small additional cost in memory, this simple strategy is easily generalized into a multistate approach using weighted combinations of the identity and ROM filter in which the weighting coefficients can be optimized using image training data.

Extensive simulations indicate that these methods perform significantly better in terms of noise suppression and detail preservation than a number of existing nonlinear techniques with as much as 40% impulse noise corruption. Moreover, the method can effectively restore images corrupted with Gaussian noise and mixed Gaussian and impulse noise.

5.1. Image Filters

Image filters are used to remove noise, sharpen contrast, highlight contours, detect edges and other uses. Image filters can be classified as linear or non linear. Linear filters are also know as convolution filters as they can be represented using a matrix multiplication. Thresholding and image equalization are examples of non-linear operations, as is the median filter.

5.2. Proposed Filters

1. Basic filters
2. Non-local means filters
3. Bilateral filters

4. Edge preserving smoothing filter

5.3. Basic Filters

1. Median filter
2. Average filter

5.4. Median Filter

Median filtering is a nonlinear method used to remove noise from images. It is widely used as it is very effective at removing noise while preserving edges. It is particularly effective at removing 'salt and pepper' type noise. The median filter works by moving through the image pixel by pixel, replacing each value with the median value of neighbouring pixels. The pattern of neighbours is called the "window", which slides, pixel by pixel over the entire image 2 pixel, image.

The median is calculated by first sorting all the pixel values from the window into numerical order, and then replacing the pixel being considered with the middle (median) pixel value.

5.5. Average Filter

Average (or mean) filtering is a method of 'smoothing' images by reducing the amount of intensity variation between neighboring pixels. The average filter works by moving through the image pixel by pixel, replacing each value with the average value of neighboring pixels, including itself.

There are some potential problems:

1. A single pixel with a very unrepresentative value can significantly affect the average value of all the pixels in its neighbourhood.
2. When the filter neighborhood straddles an edge, the filter will interpolate new values for pixels on the edge and so will blur that edge. This may be a problem if sharp edges are required in the output.

5.6. Non-Local Means Filter

The non-local means algorithm does not make the same assumptions about the image as other methods. Instead it assumes the image contains an extensive amount of self-similarity. Efros and Leung originally developed the concept of self-similarity for texture synthesis. An example of self-similarity is displayed in Figure 1 below. The figure shows three pixels *p*, *q1*, and *q2* and their respective neighborhoods. The neighborhoods of pixels *p* and *q1* are similar, but the neighborhoods of pixels *p* and *q2* are not similar. Adjacent pixels tend to have similar neighborhoods, but non-adjacent pixels will also have similar neighborhoods when there is structure in the image. For example, in Figure 2 most of the pixels in the same column as *p* will have similar neighborhoods to *p*'s neighborhood.

The self-similarity assumption can be exploited to denoise an image. Pixels with similar neighborhoods can be used to determine the denoised value of a pixel.

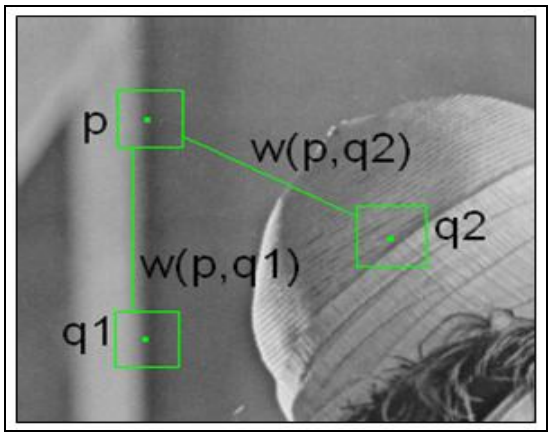

Figure 1: Example of self-similarity in an image. Pixels p and q1 have similar neighborhoods, but pixels p and q2 do not have similar neighborhoods. Because of this, pixel q1 will have a stronger influence on the denoised value of p than q2.

5.7. Bilateral Filter

The bilateral filter is a non-linear technique that can blur an image while respecting strong edges. Its ability to decompose an image into different scales without causing haloes after modification has made it ubiquitous in computational photography applications such as tone mapping, style transfer, relighting, and denoising. This text provides a graphical, intuitive introduction to bilateral filtering, a practical guide for efficient implementation and an overview of its numerous applications, as well as mathematical analysis.

The bilateral filter has several qualities that explain its success:

1. Its formulation is simple: each pixel is replaced by a weighted average of its neighbors. This aspect is important because it makes it easy to acquire intuition about its behavior, to adapt it to application-specific requirements, and to implement it.

2. It depends only on two parameters that indicate the size and contrast of the features to preserve.
3. It can be used in a non-iterative manner. This makes the parameters easy to set since their effect is not cumulative over several iterations.
4. It can be computed at interactive speed even on large images, thanks to efficient numerical schemes, and even in real time if graphics hardware is available.

Figure 2: The bilateral filter converts any input image (a) to a smoothed version
(b) It removes most texture, noise, and fine details, but preserves large sharp edges without blurring.

Bilateral filtering is a technique to smooth images while preserving edges. It can be traced back to 1995 with the work of Aurich and Weule on nonlinear Gaussian filters. It was later rediscovered by Smith and Brady as part of their SUSAN framework, and Tomasi and Manduchi who gave it its current name.
Since then, the use of bilateral filtering has grown rapidly and is now ubiquitous in image processing applications Figure 2. It has been used in various contexts such as denoising, texture editing and relighting, tone management, demosaicking, stylization, and optical-flow estimation.

6. Edge Preserving Filter
To locate the edge existing in the current W, a simple edge-preserving technique, which can be realized easily. The dataflow of our edge-preserving image filter are shown in Fig 4.5 respectively.
Here, we consider eight directional differences, from D1 to D8, to reconstruct the noisy pixel value, as shown in Figure 3. Only those composed of noise-free pixels are taken into account to avoid possible misdetection. Directions passing through the suspected pixels are discarded to reduce misdetection.
Therefore, we use Maxij and Minij, defined in similarity module, to determine whether the values of d, e, f, g, and h are likely corrupted, respectively. If the pixel is likely being corrupted by noise, we don't consider the direction including the suspected pixel.
In the second block, if d, e, f, g, and h are all suspected to be noisy pixels, and no edge can be processed, so ^ fij (the estimated value of pij) is equal to the weighted average of luminance values of three previously denoised pixels and calculated as (a + b * 2 + c)/4.
In other conditions, the edge filter calculates the directional differences of the chosen directions and locates the smallest one ðDminÞ among them in the third block.

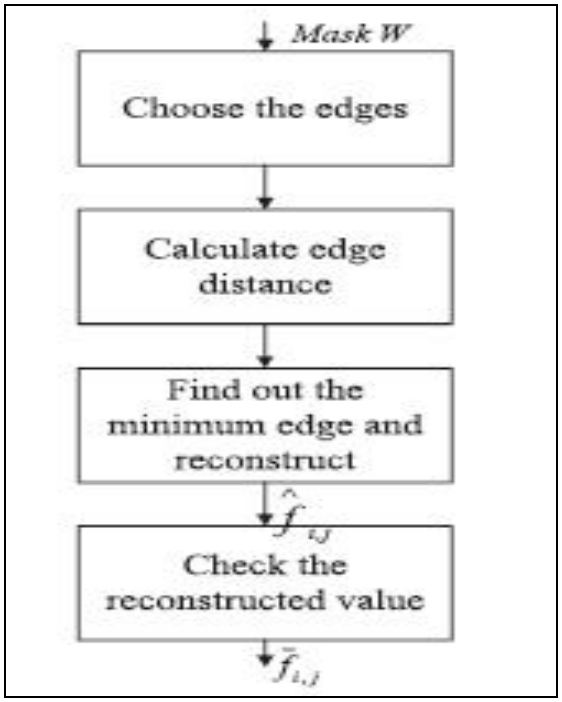

Figure 3: Data flow of edge-preserving image filter

The standard deviation intensity parameter specified in the same unit as z values determining the intensity domain contribution to kernel waiting's. No data values in the input image are ignored during filtering when the neighborhood around the grid cell extent beyond the edge of the grid no data values are assigned to these sites. The output raster is of the float data type and continuous data scale.

In the last block of Fig. 4.4, the smallest directional difference implies that it has the strongest spatial relation with pi;j, and probably there exists anedge in its direction. Hence, the mean of luminance values of the pixels which possess the smallest directional difference is treated as ^ fij. After ^ fij is determined, a tuning skill is used to filter the bias edge. If ^ fij obtain the correct edge, it will situate at the median of

The equations are as follows:

$$
\begin{aligned}
D_1 &= |d-h| + |a-e| \\
D_2 &= |a-g| + |b-h| \\
D_3 &= |b-g| \times 2 \\
D_4 &= |b-f| + |c-g| \\
D_5 &= |c-d| + |e-f| \\
D_6 &= |d-e| \times 2 \\
D_7 &= |a-h| \times 2 \\
D_8 &= |c-f| \times 2,
\end{aligned}
$$

$$
\hat{f}_{i,j} = \begin{cases}
(a+d+e+h)/4, & \text{if } D_{\min} = D_1, \\
(a+b+g+h)/4, & \text{if } D_{\min} = D_2, \\
(b+g)/2, & \text{if } D_{\min} = D_3, \\
(b+c+f+g)/4, & \text{if } D_{\min} = D_4, \\
(c+d+e+f)/4, & \text{if } D_{\min} = D_5, \\
(d+e)/2, & \text{if } D_{\min} = D_6, \\
(a+h)/2, & \text{if } D_{\min} = D_7, \\
(c+f)/2, & \text{if } D_{\min} = D_8.
\end{cases}
$$

b, d, e, and gin the equation (1,2) because of the spatial relation and the characteristic of edge preserving.
Otherwise, the values of fij will be replaced by the median of four neighboring pixels (b, d, e, and g).

7. Simulation Results

7.1. Average Filter Result

In the Fig.4 average filtering method is been processed. This filter is used for 'smoothing' images by reducing the amount of intensity variation between neighbouring pixels.

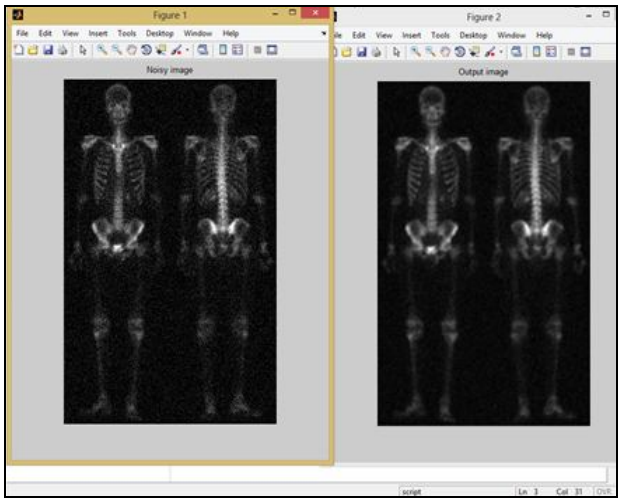

Figure 4: Average filter simulation result

The average filter works by moving through the image pixel by pixel, replacing each value with the average value of neighbouring pixels, including itself.

7.2. Median Filter Result

In the Fig. 5 median filtering is simulated. Here nonlinear method is used to remove noise from images. It is widely used, as it is very effective at removing noise while preserving edges.

The median is calculated by first sorting all the pixel values from the window into numerical order, and then replacing the pixel being considered with the middle pixel value

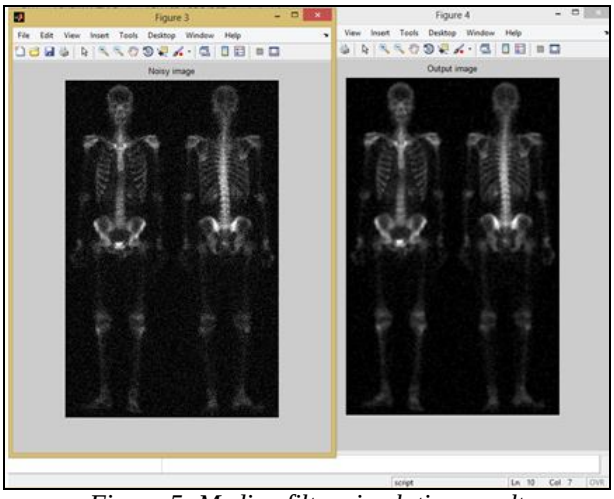

Figure 5: Median filter simulation result

7.3. Bilateral Filter Result

In the Fig. 6 bilateral filter is simulated. It is a non-linear technique that can blur an image while respecting strong edges. After modification has made it ubiquitous in computational photography applications such as tone mapping, style transfer, relighting, and denoising.

It has been used in various contexts such as denoising, texture editing and relighting, tone management, demosaicking, stylization, and optical-flow estimation.

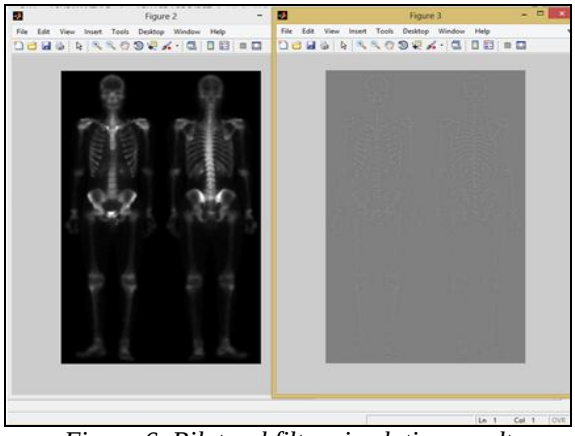

Figure 6: Bilateral filter simulation result

7.4. Non-Local Means Filter Result

In the Fig. 7 NLM method is simulated. It does not make the same assumptions about the image as other methods. Instead it assumes the image contains an extensive amount of self-similarity.

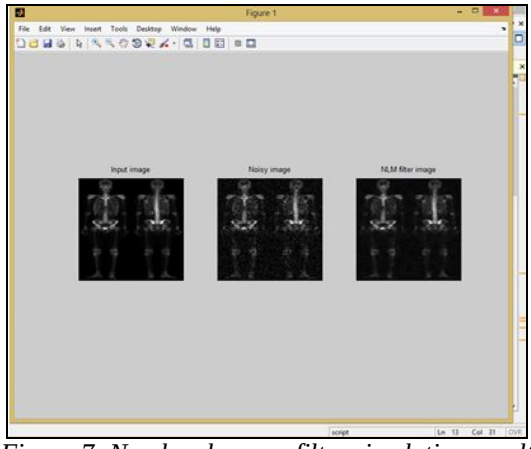

Figure 7: Non local means filter simulation result

Adjacent pixels tend to have similar neighborhoods, but non-adjacent pixels will also have similar neighborhoods when there is structure in the image.

7.5. Edge Preserving Filter Result

In the Fig. 8 edge preserve filter is simulated. It is used to preserves the edges in the image. It takes some time for simulation process.

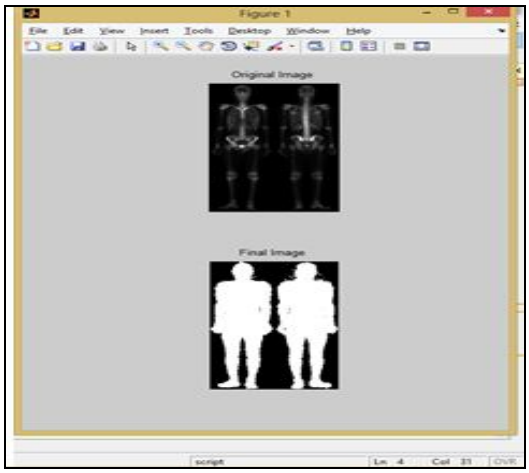

Figure 8: Edge preserve filter simulation result

It is an image processing technique that smooths away textures whilst retaining sharp edges.

8. Comparison Chart of Filters

8.1. PSNR Comparison

Impulse noise (In percentage)	Average Filter	Median Filter	Bilateral filter	Non local means filter	Edge preserving filter
0.2	35.72578	32.43742	31.38461	35.49587	45.37099
0.1	37.10239	33.80766	32.98984	35.65442	42.70781
0.01	41.38394	39.05001	38.17624	36.18831	38.10124
0.001	44.25679	43.41799	42.64631	36.25277	32.79337
0.0001	45.04705	45.30492	44.86371	36.29652	31.33673

Table 1: Comparison Chart of PSNR

8.2. MSE Comparison

Impulse noise (In percentage)	Average filter	Median Filter	Bilateral filter	Non local means filter	Edge preserving filter
0.2	0.19234	0.06584	0.19327	0.30461	0.10532
0.1	0.12894	0.03375	0.12977	0.16324	0.05256
0.01	0.63982	0.00319	0.06375	0.08525	0.00499
0.001	0.05756	0.00044	0.5757	0.08043	0.00057
0.0001	0.05707	0.00018	0.5706	0.07857	0.00015

Table 2: Comparison chart of MSE

8.3. RMSE Comparison

Impulse noise (In percentage)	Average filter	Median Filter	Bilateral filter	Non local means filter	Edge preserving filter
0.2	0.43856	0.25660	0.43962	0.55192	0.32453
0.1	0.35908	0.18372	0.36024	0.40403	0.22925
0.01	0.25295	0.05645	0.25249	0.29198	0.07065
0.001	0.23992	0.02093	0.23993	0.28360	0.02394
0.0001	0.23888	0.01355	0.23886	0.28030	0.01214

Table 3: Comparison chart of RMSE

9. Conclusion

It is an efficient method for removal of random-valued impulse noise is proposed in this project. The approach uses the some particular filters to remove the noisy pixel and employs an effective design to locate the edge. With adaptive skill, the quality of the reconstructed images is notable improved. Our extensive experimental results demonstrate that the performance of our proposed technique is better than the previous lower complexity methods and is comparable to the higher complexity methods in terms of both quantitative evaluation and visual quality. It requires only low computational complexity and two line memory buffers. Therefore, it is very suitable to be applied to many real-time applications.

10. References

1. Bar. L, Sochen. N, and Kiryati. N.(2006), 'Image deblurring in the presenceof impulsive noise, 'International. J. Computer. Visual., vol. 70, no. 3,pp. 279–298,2006.
2. Chan.R.H, Ho.C.W, and Nikolova.M.(2005), 'Salt-and-pepper noise removal by median-type noise detectors and detail-preserving regularization,'IEEE Transactions. Image Processing., vol. 14, no. 10, pp. 1479–1485.
3. Hwang .Hand Haddad. R.A. (2001), 'Adaptive median filters: Newalgorithmsand results,'IEEE Transactions. Signal Process. vol. 4, no. 4, pp. 499–502.
4. Ko.S.Jand Lee.Y.H.(2000), 'Center weighted median filters and their application to image enhancement,'IEEE Transactions. Circuits Systems. vol. 38,no. 9, pp. 984–993.
5. Longhand Willson.A.N.(2001), 'Median filter with adaptive length', IEEE Transactions. Circuits Systems., vol. 35, no. 6, pp. 675–690.
6. Liu.X, Zhao.D, Xiong.R, Ma.S, Gao.W, and Sun.H.(2011), 'Image interpolation via regularized local linear regression,'IEEE Transactions. ImageProcess., vol. 20, no. 12, pp. 3455–3469.
7. Mallat.S.G. (2008), A Wavelet Tour of Signal Processing, the Sparse Way,3rd edition. Amsterdam, The Netherlands: Elsevier.
8. Ng.P.Eand Ma.K.K.(2006), 'A switching median filter with boundary discriminative noise detection for extremely corrupted images, 'IEEE Transactions. Image Processing. vol. 15, no. 6, pp. 1506–1516.
9. Sun. Tand Neuvo. Y. (2008), 'Detail-preserving median based filters in image processing, 'Pattern Recognition., vol. 15, no. 4, pp. 341–347.
10. Zhang.Sand Karim.M.A.(2002), 'A new impulse detector for switching median filters, 'IEEE Signal Processing., vol. 9, no. 11, pp. 360–363.