An Efficient and Simple Switching Filter for Removal of High Density Salt-and-Pepper Noise

Hani M. Ibrahim
Math and Computer Science dept., Faculty of science, Menufia university, Egypt.
hanimir78@yahoo.com

Abstract — This paper presents an efficient and simple adaptive method for high density salt-and-pepper noise removal. A noise detector is utilized to check whether the selected pixel is noisy or noise free. Noise pixels will then be subjected to the second stage of the filtering action, while the noise free pixels are left unaltered. Since not every pixel is filtered, undue distortion can be avoided. The noise free pixels are only considered in the filter operation for finding the value of the processed pixel. The window size is selected as 3 X 3 in the first step. If all pixels within the window are considered to be noise, then change the selected window size to 5 X 5. If all the pixels within the selected 5 x 5 window are considered to be noise, then the processing pixel is replaced by the previous resultant pixel. This technique requires one nonnoise original image as training image. The key point of the filter operation is based on the solution of the equations system X=A^{-1}B in the nonnoise original image. An algorithm to extract the data from the nonnoisy image and form it in the linear equation system is presented. Comparison of the given filter with other existing filters is provided in this paper. The results demonstrate that the proposed technique can obtain better performances than other existing denoising techniques. The proposed method works well for high-density salt & pepper noise even up to a noise density of 97%.

Index Terms — Salt – and – pepper noise, switching filter, noise reduction, image denoising.

I. INTRODUCTION

Digital images are corrupted by undesirable random variations in intensity values called noise [1]. During image acquisition or transmission, digital images could be contaminated by impulse noise, which is caused by malfunctioning pixels in camera sensors, faulty memory locations in hardware, or transmission in a noisy channel [2]. It’s important to eliminate noise before subsequent image processing tasks such as edge detection or segmentation is carried out. Salt-and-pepper noise is a special case of impulse noise where the noisy pixels can take only the maximum and minimum values in the dynamic range. Recently, many algorithms such as [2-6] have been proposed to carry out the salt-and-pepper noise suppression. Therefore, to efficiently remove noise from an image while preserving its features is a fundamental problem of image processing [7]. The standard median filter (SMF) is the most popular choice for removing the salt-and-pepper noise. However, the median filter and its derivations such as weighted median (WM) [8], center weighted median (CWM) [9], and recursive weighted median (RWM)[10] are typically implemented invariantly across an image. They tend to alter the noise free pixels as well as the noise pixels, exhibits blurring of filtered images. To avoid the damage on the noise free pixels, many image filters with an impulse detector are proposed such as in the literatures [4],[11-13].

In this paper, an efficient and simple method to remove salt-and-pepper noise is proposed. In this method, only the noise pixels are changed by filter operation, while the noise free pixels are kept unchanged. In addition, the window size is altered among 3X3 and 5X5 based on the noise and noise free pixels within the window. This technique requires only one original nonnoise image as data set. May be the median value or other values based on the median are not the correct choice for noise pixel replacement. Thus, the filter output is based on the solution of the system X=A-1B in the nonnoise original image which will be explained later. Further, this method does not require threshold parameters. This method can remove salt-and-pepper-noise with a noise level as high as 97%, and it’s relatively fast. The experimental results demonstrate that the proposed method is effective for removing noise and preserving fine details than other existing methods. The outline of this paper is organized as follows. A brief review of related work is given in section II. In section III, the proposed technique is introduced. The implementation results and comparison are provided in section IV. Finally, the conclusions are summed up in section V.

II. RELATED WORK

Many approaches have been proposed to remove salt-and-pepper noise. Chang-you et al[13] have proposed an adaptive weighted median filtering algorithm to remove salt-and-pepper noise. This algorithm first takes a decision whether the pixel under test is noise or not by comparing the block uniformity of the 3x3
An Efficient and Simple Switching Filter for Removal of High Density Salt-and-Pepper Noise

III. PROPOSED METHOD

An efficient method for removal high density salt-and-pepper noise with preserving image details is proposed. It is composed of three stages as: (1) Data extraction from the nonnoisy image. (2) Noise detection. (3) Noise cancellation. Figure 1 shows the core idea for the proposed method. The extracted data from the nonnoisy image, as Lena image, is applied in the filter equation to produce filtered image from its noisy image.

Figure 1. The Core Idea for the Proposed Method.

(1) Data extraction

Extracted the data from the nonnoisy image are used in the filter operation to produce filtered image from its noisy image. This is presented in this stage by achieving the following steps:-

1- Divide the nonnoisy image into non-overlapping four blocks.

2- Repeat the following for each block \(i\), \(\exists i = 1,4\)

2.1- Select 3X3 window and place it, randomly, in any area within the block \(i\).

2.2- Extract the center pixel value and put it in the first row in the vector \(iB\).

2.3- Extract two nearest neighbors and put them in the second row in the matrix \(iA\).

2.4- Move, randomly, the window within the block \(i\).

2.5- Extract the center pixel value and put it in the first row in the vector \(iB\).

2.6- Extract two nearest neighbors and put them in the second row in the matrix \(A\).

End

\[
A = \left( \sum_{i=1}^{4} A_i \right) / 4 \quad \text{and} \quad B = \left( \sum_{i=1}^{4} B_i \right) / 4
\]
4- Repeat the following for each block \( i \)
4.1- Select 3X3 window and place it randomly in any area within the block \( i \).
4.2- Extract the center pixel value and put it in the first row in the vector \( \overrightarrow{B_i} \).
4.3-Extract three nearest neighbors and put them in the first row in the matrix \( \overrightarrow{A_i} \).
4.4-Move randomly the window within the block \( i \).
4.5- Extract the center pixel value and put it in the second row in the vector \( \overrightarrow{B_i} \).
4.6-Extract three nearest neighbors and put them in the second row in the matrix \( \overrightarrow{A_i} \).
4.7-Move randomly the window within the block \( i \).
4.8- Extract the center pixel value and put it in the third row in the vector \( \overrightarrow{B_i} \).
4.9-Extract three nearest neighbors and put them in the third row in the matrix \( \overrightarrow{A_i} \).

End

5- Compute \( \overrightarrow{r} = \left( \frac{1}{4} \sum_{i=1}^{4} \overrightarrow{A_i} \right) / 4 \) and \( \overrightarrow{s} = \left( \frac{1}{4} \sum_{i=1}^{4} \overrightarrow{B_i} \right) / 4 \)

6- Repeat the following for each block \( i \)
6.1-Select 3X3 window and place it randomly in any area within the block \( i \).
6.2-Extract the center pixel value and put it in the first row in the vector \( \hat{\overrightarrow{B_i}} \).
6.3-Extract four nearest neighbors and put them in the first row in the matrix \( \hat{\overrightarrow{A_i}} \).
6.4-Move randomly the window within the block \( i \).
6.5-Extract the center pixel value and put it in the second row in the vector \( \hat{\overrightarrow{B_i}} \).
6.6-Extract four nearest neighbors and put them in the second row in the matrix \( \hat{\overrightarrow{A_i}} \).
6.7-Move randomly the window within the block \( i \).
6.8-Extract the center pixel value and put it in the third row in the vector \( \hat{\overrightarrow{B_i}} \).
6.9-Extract four nearest neighbors and put them in the third row in the matrix \( \hat{\overrightarrow{A_i}} \).
6.10-Move randomly the window within the block \( i \).
6.11- Extract the center pixel value and put it in the fourth row in the vector \( \hat{\overrightarrow{B_i}} \).
6.12-Extract four nearest neighbors and put them in the fourth row in the matrix \( \hat{\overrightarrow{A_i}} \).

End

\[ \hat{A} = \left( \frac{1}{4} \sum_{i=1}^{4} \overrightarrow{A_i} \right) / 4 \quad \hat{B} = \left( \frac{1}{4} \sum_{i=1}^{4} \overrightarrow{B_i} \right) / 4 \]

All the matrices and vectors that extracted in this stage are used in filter operation.

(2) Noise detection.

Noise detection is the first stage in this method. The salt-and-pepper noise pixels can take the maximum and minimum values in the dynamic range. Therefore, if the value of the processed pixel lies between the maximum and minimum values inside the window, then it kept unchanged. If the value does not lie within this range, then it is a noisy pixel and is replaced by the output of the filter operation.

(3) Noise cancellation

Each element in the vectors \( \overrightarrow{B}, \overrightarrow{B} \) and \( \hat{\overrightarrow{B}} \) is the desired value for its corresponding row (its neighbors) in the matrices \( \overrightarrow{A}, \overrightarrow{A} \) and \( \hat{\overrightarrow{A}} \), respectively. The goal is to estimate parameters which are obtained the desired values, in vectors, from their corresponding rows in the matrices. The linear systems \( A \overrightarrow{X} = \overrightarrow{B}, \overrightarrow{AX} = \overrightarrow{B} \) and \( \hat{A} \hat{\overrightarrow{X}} = \hat{\overrightarrow{B}} \) are used to describe the relation between the vectors and their corresponding matrices such that \( \overrightarrow{X} = \overrightarrow{A}^{-1} \overrightarrow{B}, \overrightarrow{AX} = \overrightarrow{A}^{-1} \overrightarrow{B} \) and \( \hat{\overrightarrow{X}} = \hat{\overrightarrow{A}}^{-1} \hat{\overrightarrow{B}} \). The vectors \( \overrightarrow{X}, \overrightarrow{X} \) and \( \hat{\overrightarrow{X}} \) represent the estimation parameters that are used in the filter operation. Switching between 3X3 and 5X5 window size is used based on the number of noisy pixels where only the noise free pixels are considered in the filter operation. But, if all the pixels inside the 5X5 window are noisy, then replace the processed pixel by the previous resultant pixel. The noise cancellation stage for each pixel location \((i,j)\) to produce filter image \( \overrightarrow{R} \) can be achieved by the following steps.

1- Select a 2-D window of size 3X3.
2- Eliminate the noisy pixels within the window.
3- Insert the noise free pixels into vector \( \overrightarrow{R} \).
4- Assign the number of pixels in the \( \overrightarrow{R} \) to \( L \).
5- If \( L < 1 \), go to step 10.
6- If \( L = 1 \)
   6.1 Let \( g(i,j) \) equal to the pixel value in \( L \)
   6.2 Return to step 1 for the next \((i,j)\)
7- If \( L = 2 \)
   7.1 \( g(i,j) = L(1) \cdot \overrightarrow{X}(1) + L(2) \cdot \overrightarrow{X}(2) \)
   7.2 Return to step 1 for the next \((i,j)\)
8- If \( L = 3 \)
   8.1 \( g(i,j) = L(1) \cdot \overrightarrow{X}(1) + L(2) \cdot \overrightarrow{X}(2) + L(3) \cdot \overrightarrow{X}(3) \)
An Efficient and Simple Switching Filter for Removal of High Density Salt-and-Pepper Noise

8.2 Return to step 1 for the next \((i, j)\)

9. If \((L \geq 4)\)

\[
g(i, j) = L(1)\hat{X}(1) + L(2)\hat{X}(2) + L(3)\hat{X}(3) + L(4)\hat{X}(4)
\]

9.2 Return to step 1 for the next \((i, j)\)

10. Increase the window by 5x5.

11. Eliminate the noisy pixels within the window.

12. Insert the noise free pixels into vector \(S\).

13. Assign the number of pixels in the \(S\) to \(P\).

14. If \((P < 1)\), go to step 19

15. If \((P = 1)\)

15.1 Let \(g(i, j)\) equal to the pixel value in \(P\)

15.2 Return to step 1 for the next \((i, j)\)

16. If \((P = 2)\)

16.1 \(g(i, j) = P(1)\hat{X}(1) + P(2)\hat{X}(2)\)

16.2 Return to step 1 for the next \((i, j)\)

17. If \((P = 3)\)

17.1 \(g(i, j) = P(1)\hat{X}(1) + P(2)\hat{X}(2) + P(3)\hat{X}(3)\)

17.2 Return to step 1 for the next \((i, j)\)

18. If \((P \geq 4)\)

\[
g(i, j) = P(1)\hat{X}(1) + P(2)\hat{X}(2) + P(3)\hat{X}(3) + P(4)\hat{X}(4)
\]

18.2 Return to step 1 for the next \((i, j)\)

19. Replace \(g(i, j)\) with the nearest noise free pixel.

20. Return to step 1 for the next \((i, j)\)

IV. SIMULATION RESULTS

To check the efficiency of the proposed technique, a variety of well-known test images with 1024x1024: Bridge, Boat, Goldhill, and Peppers are used. A wide range of high noise ratios varied from 50% to 97% is tested. Haidi at el method [14], DBA [18], MDBUTMF [19] and standard median filter (SMF) are used in the comparison with the existing algorithms by varying the noise density from 50% to 97%. Thus, it is observed that the proposed method gives better PSNR than other existing algorithms. Figure 2 shows a plot of PSNR versus noise densities for Bridge, Boat, Goldhill, and Peppers images. The visual result of the proposed method for Boat image corrupted by 50%, Goldhill image corrupted by 80% and Peppers image corrupted by 97% are shown in Figure 3, Figure 4 and Figure 5, respectively.

![Comparison graph of PSNR at different noise densities for Bridge image.](image1)

![Comparison graph of PSNR at different noise densities for Boat image.](image2)
An Efficient and Simple Switching Filter for Removal of High Density Salt-and-Pepper Noise

(c) Comparison graph of PSNR at different noise densities for Goldhill image.

(d) Comparison graph of PSNR at different noise densities for Peppers image.

Figure 2: Comparison Graph of PSNR at Different Noise Densities for Bridge, Boat, Goldhill and Peppers Images

Table 1: COMPARISON OF PROCESSING RESULTS IN PSNR (dB) FOR BRIDGE, BOAT, GOLDHILL AND PEPPERS IMAGES.

<table>
<thead>
<tr>
<th></th>
<th>Bridge</th>
<th>Goldhill</th>
<th>Peppers</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>50%</td>
<td>60%</td>
<td>70%</td>
<td>80%</td>
</tr>
<tr>
<td>SMF</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PA</td>
<td>33.0957</td>
<td>31.2863</td>
<td>30.4471</td>
<td>29.6731</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMF</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>15.4695</td>
<td>12.5225</td>
<td>10.1506</td>
<td>8.2364</td>
</tr>
<tr>
<td>PA</td>
<td>33.6924</td>
<td>31.8605</td>
<td>31.0683</td>
<td>28.5087</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMF</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>15.3552</td>
<td>12.3908</td>
<td>10.0116</td>
<td>8.0830</td>
</tr>
<tr>
<td>PA</td>
<td>36.2110</td>
<td>34.3507</td>
<td>33.5131</td>
<td>32.5638</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMF</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>15.3275</td>
<td>12.3661</td>
<td>9.9594</td>
<td>8.0579</td>
</tr>
<tr>
<td>PA</td>
<td>35.3872</td>
<td>33.4932</td>
<td>32.6362</td>
<td>31.7422</td>
</tr>
</tbody>
</table>

Copyright © 2013 MECS

LIJ. Image, Graphics and Signal Processing, 2013, 12, 1-8
An Efficient and Simple Switching Filter for Removal of High Density Salt-and-Pepper Noise

Figure 3. The Result of PA for Corrupted Boat Image

Figure 4. The Result of PA for Corrupted Goldhill Image
CONCLUSIONS AND PROSPECTS

A new denoising method for removing salt-and-pepper noise is introduced in this paper. This method is simple in implementation where no complex operations are required. The proposed method framework outperforms other existing denoising method for removal of salt- and -pepper noise with preserving image details as it is clearly seen in visual and quantitative results. Moreover, the suggested method presents a quite stable performance over a wide variety of images. In addition, the presented method does not need large data set for training like artificial neural networks and genetic programming, which consuming time for training. Indeed one image is enough for extracting the coefficients, which uses in the filter action. The main advantage of this method is no adjustment of threshold parameters, which influences the behavior of the system.

REFERENCES


An Efficient and Simple Switching Filter for Removal of High Density Salt-and-Pepper Noise


Hani M. Ibrahim was born in Egypt on Septamper 12th1978. He received the M.S. and Ph.D degrees in Computer Science at the University of Menoufyia, Egypt in 2004 and 2008, respectively. His research interests lies in the areas of image processing. Currently he is a lecture of Computer Science in the Faculty of Science, at the University of Menoufyia, Egypt.