A new impulse detection and filtering method for removal of wide range impulse noises
Shuenn-Shyang Wang*, Cheng-Hao Wu

Department of Electrical Engineering, Tatung University, 40 Chungshan N. Road, 3rd Sec., Taipei, Taiwan, ROC

ARTICLE INFO

Article history:
Received 21 January 2008
Received in revised form 21 October 2008
Accepted 18 January 2009

Keywords:
Image processing
Impulse noise
Image filtering

ABSTRACT

A new impulse detection and filtering algorithm is proposed for restoration of images that are highly corrupted by impulse noise. It is based on the minimum absolute value of four convolutions obtained by one-dimensional Laplacian operators. The proposed algorithm can effectively remove the impulse noise with a wide range of noise density and produce better results in terms of the qualitative and quantitative measures of the images even at noise density as high as 90%. Extensive simulations show that the proposed algorithm provides better performance than many of the existing switching median filters in terms of noise suppression and detail preservation.

© 2009 Elsevier Ltd. All rights reserved.

1. Introduction

During image acquisition or transmission, digital images could be contaminated by impulse noise. Two common types of impulses are the salt-and-pepper noise and the random-valued noise [1,2]. For image corrupted by salt-and-pepper impulse noise, the noisy pixels can take only the maximum and the minimum value in the dynamic range. For 8-bit pixel, the maximum value is 255 and the minimum value is 0. In the literature, a large number of algorithms have been proposed to remove impulse noise while preserving image details [1–9]. One of the most popular and robust nonlinear filter is the standard median filter (SMF) [1], which exploits the rank-order information of pixel intensities within a filtering window and replaces the center pixel with the median value. Conventional median filtering approaches apply the median operation to each pixel without considering whether it is uncorrupted or corrupted, thus the impulse noise is removed at the expense of blurred and distorted feature. Improved filtering algorithms employ an impulse-noise detector to determine which pixels should be filtered; hence only those pixels identified as “corrupted” would undergo the filtering process, while those identified as “uncorrupted” would remain intact. The adaptive median filter (AMF) [2] ensures that most of the impulse noise can be detected even at a high noise level provided that the window size is large enough. But, it increased the computation complexity especially at high density impulse noise. The convolution-based impulse detector and switching median filter (CD-SMF) algorithm [3] distinguishes whether the interest pixel is noise or not depending on a threshold determined by computer simulations. Decision-based algorithm (DBA) [5] processes the corrupted image by first detecting the impulse noise and uses a fixed 3×3 window size to handle the corrupted pixel for removal of impulse noises. It is found that CD-SMF and DBA will exhibit serious image blurring for high density impulse noise [3,5]. In this paper, we propose a new impulse noise detection and filtering algorithm that can effectively remove a wide range impulse noise while preserving image details. The proposed algorithm is shown to achieve excellent performance across a wide range of noise densities varying from 10% to 90%.

The organization of the rest of this paper is as follows. In the next section, a new impulse noise detection and filtering algorithm is described in detail. In Section 3, some experimental results are presented with discussion. The concluding remarks are given in Section 4.

2. The proposed impulse detection and filtering algorithm

In this paper, noise is assumed to be salt and pepper impulse noise. Pixels are randomly corrupted by two fixed extreme values, 0 and 255 (for 8-bit monochrome images) generated with the same probability [6]. On the original image pixel at location (i,j) with intensity value \( S_{ij} \), the corresponding pixel of the noisy image will be \( X_{ij} \) with the probability density function:

\[
f(X_{ij}) = \begin{cases} 
p/2 & \text{for } X_{ij} = 0 \\
1 - p & \text{for } X_{ij} = S_{ij} \\
p/2 & \text{for } X_{ij} = 255 
\end{cases}
\] (2.1)

where \( p \) is noise density.
There are three steps (steps (a)–(c)) in our proposed algorithm for impulse detection and filtering. After classifying corrupted and uncorrupted pixels (see steps (a) and (b)), we replace the corrupted pixel by the suitable value of the sorted sequence of its neighborhood values (see step (c)). We repeat steps (a)–(c) for $K$ iterations to get the convergent recovery image.

(a) The input image $X_0$ is first convolved with a set of convolution kernels. Here, four one-dimension Laplacian operators shown in Fig. 1 are used, each of which is sensitive to edges in a different orientation [3]. Then, the minimum absolute value of these four convolutions (denoted as $r_j$) is used for impulse detection, which can be represented as

$$ r_j = \min(|X_{ij} \odot K_p| : p = 1 to 4) \quad (2.2) $$

where $K_p$ is the $p$th kernel, and $\odot$ denotes a convolution operation. We compare $r_j$ with a threshold $T$ to determine whether a pixel is corrupted, i.e.,

$$ x_{ij} = \begin{cases} 1, & r_j > T \\ 0, & r_j \leq T \end{cases} \quad (2.3) $$

If $x_{ij} = 1$, then the pixel $X_{ij}$ is marked as noise candidate; otherwise the pixel $X_{ij}$ is noise-free. A reasonable threshold $T$ can be determined using computer simulation.

(b) If the interesting pixel $X_{ij}$ is marked as noise candidate, we use a fixed 3×3 window $W$ shown in (2.4) for further processing:

$$ W = \left[ \begin{array}{ccc} a_0 & a_5 & a_3 \\ a_6 & a_1 & a_4 \\ a_8 & a_2 & a_9 \end{array} \right] = \left[ \begin{array}{ccc} X_{i-1,j-1} & X_{i,j-1} & X_{i,j+1} \\ X_{i-1,j} & X_{i,j} & X_{i+1,j} \\ X_{i-1,j+1} & X_{i,j+1} & X_{i+1,j+1} \end{array} \right] \quad (2.4) $$

By sorting five elements $a_0, a_1, a_2, a_3$ and $a_4$ in ascending order, we get a sorted sequence: $a_0 > a_1 > a_2 > a_3 > a_4$ where $a_0 < a_1 < a_2 < a_3 < a_4$. If $X_{ij}$ satisfies the following cases, the pixel will be considered a noise-free pixel and retain its value:

Case 1: $a_0 < X_{ij} < a_4$

Case 2: $X_{ij} = a_5 \neq 225$

Case 3: $X_{ij} = a_3 \neq 0$

After these procedures, we can find the corrupted pixels from the noisy image.

---

**Table 1**

<table>
<thead>
<tr>
<th>Noise density</th>
<th>Algorithm</th>
<th>PSNR (db)</th>
<th>Time (seconds)</th>
<th>PSNR (db)</th>
<th>Time (seconds)</th>
<th>PSNR (db)</th>
<th>Time (seconds)</th>
<th>PSNR (db)</th>
<th>Time (seconds)</th>
<th>PSNR (db)</th>
<th>Time (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10%</td>
<td>SMF</td>
<td>28.93</td>
<td>0.610</td>
<td>35.57</td>
<td>0.639</td>
<td>32.94</td>
<td>0.938</td>
<td>31.89</td>
<td>0.828</td>
<td>39.09</td>
<td>0.764</td>
</tr>
<tr>
<td>20%</td>
<td>AMF</td>
<td>28.36</td>
<td>0.625</td>
<td>32.94</td>
<td>0.639</td>
<td>30.97</td>
<td>0.823</td>
<td>29.80</td>
<td>0.717</td>
<td>34.32</td>
<td>0.734</td>
</tr>
<tr>
<td>30%</td>
<td>CD-SMF</td>
<td>26.42</td>
<td>0.625</td>
<td>30.72</td>
<td>0.639</td>
<td>29.62</td>
<td>0.823</td>
<td>28.17</td>
<td>0.875</td>
<td>32.00</td>
<td>0.735</td>
</tr>
<tr>
<td>40%</td>
<td>DBA</td>
<td>23.99</td>
<td>0.625</td>
<td>28.94</td>
<td>0.657</td>
<td>27.83</td>
<td>0.828</td>
<td>27.41</td>
<td>0.890</td>
<td>30.27</td>
<td>0.734</td>
</tr>
<tr>
<td>50%</td>
<td>Proposed algorithm</td>
<td>21.74</td>
<td>0.625</td>
<td>27.37</td>
<td>0.671</td>
<td>26.09</td>
<td>0.938</td>
<td>26.89</td>
<td>0.906</td>
<td>28.54</td>
<td>0.735</td>
</tr>
<tr>
<td>60%</td>
<td></td>
<td>19.74</td>
<td>0.640</td>
<td>26.17</td>
<td>0.718</td>
<td>22.55</td>
<td>0.796</td>
<td>26.17</td>
<td>0.889</td>
<td>27.33</td>
<td>0.75</td>
</tr>
<tr>
<td>70%</td>
<td></td>
<td>15.98</td>
<td>0.609</td>
<td>24.18</td>
<td>0.859</td>
<td>17.39</td>
<td>0.843</td>
<td>25.52</td>
<td>0.907</td>
<td>26.00</td>
<td>0.765</td>
</tr>
<tr>
<td>80%</td>
<td></td>
<td>13.04</td>
<td>0.609</td>
<td>22.39</td>
<td>1.531</td>
<td>12.37</td>
<td>0.86</td>
<td>24.49</td>
<td>0.938</td>
<td>24.53</td>
<td>0.764</td>
</tr>
<tr>
<td>90%</td>
<td></td>
<td>8.94</td>
<td>0.609</td>
<td>20.17</td>
<td>10.60</td>
<td>8.46</td>
<td>0.764</td>
<td>22.20</td>
<td>0.906</td>
<td>22.20</td>
<td>0.765</td>
</tr>
</tbody>
</table>

---

**Table 2**

<table>
<thead>
<tr>
<th>Noise density</th>
<th>Algorithm</th>
<th>PSNR (db)</th>
<th>Time (seconds)</th>
<th>PSNR (db)</th>
<th>Time (seconds)</th>
<th>PSNR (db)</th>
<th>Time (seconds)</th>
<th>PSNR (db)</th>
<th>Time (seconds)</th>
<th>PSNR (db)</th>
<th>Time (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10%</td>
<td>SMF</td>
<td>31.89</td>
<td>0.609</td>
<td>37.33</td>
<td>0.639</td>
<td>35.34</td>
<td>0.823</td>
<td>33.98</td>
<td>0.859</td>
<td>42.11</td>
<td>0.735</td>
</tr>
<tr>
<td>20%</td>
<td>AMF</td>
<td>31.05</td>
<td>0.625</td>
<td>35.57</td>
<td>0.656</td>
<td>33.07</td>
<td>0.86</td>
<td>31.59</td>
<td>0.75</td>
<td>37.33</td>
<td>0.734</td>
</tr>
<tr>
<td>30%</td>
<td>CD-SMF</td>
<td>28.73</td>
<td>0.671</td>
<td>31.65</td>
<td>0.640</td>
<td>31.14</td>
<td>0.938</td>
<td>30.64</td>
<td>0.859</td>
<td>34.90</td>
<td>0.75</td>
</tr>
<tr>
<td>40%</td>
<td>DBA</td>
<td>23.98</td>
<td>0.625</td>
<td>30.34</td>
<td>0.688</td>
<td>27.26</td>
<td>0.823</td>
<td>29.09</td>
<td>0.890</td>
<td>31.50</td>
<td>0.75</td>
</tr>
<tr>
<td>50%</td>
<td>Proposed algorithm</td>
<td>20.32</td>
<td>0.609</td>
<td>28.49</td>
<td>0.75</td>
<td>22.95</td>
<td>0.823</td>
<td>29.26</td>
<td>0.890</td>
<td>29.87</td>
<td>0.75</td>
</tr>
<tr>
<td>60%</td>
<td></td>
<td>16.67</td>
<td>0.625</td>
<td>27.02</td>
<td>0.875</td>
<td>16.05</td>
<td>0.836</td>
<td>27.26</td>
<td>0.875</td>
<td>28.73</td>
<td>0.75</td>
</tr>
<tr>
<td>70%</td>
<td></td>
<td>12.66</td>
<td>0.625</td>
<td>24.57</td>
<td>3.03</td>
<td>11.52</td>
<td>0.796</td>
<td>26.08</td>
<td>1.06</td>
<td>27.4</td>
<td>0.765</td>
</tr>
<tr>
<td>80%</td>
<td></td>
<td>8.26</td>
<td>0.610</td>
<td>23.12</td>
<td>11.2</td>
<td>7.34</td>
<td>0.836</td>
<td>24.06</td>
<td>0.938</td>
<td>24.84</td>
<td>0.74</td>
</tr>
</tbody>
</table>
Table 3
PSNR (db) and computation time (seconds) for various algorithms for Girl image at different noise density.

<table>
<thead>
<tr>
<th>Noise density</th>
<th>SMF</th>
<th>AMF</th>
<th>CD-SMF</th>
<th>DBA</th>
<th>Proposed algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PSNR</td>
<td>Time</td>
<td>PSNR</td>
<td>Time</td>
<td>PSNR</td>
</tr>
<tr>
<td>10%</td>
<td>19.94</td>
<td>0.609</td>
<td>23.33</td>
<td>0.640</td>
<td>20.61</td>
</tr>
<tr>
<td>20%</td>
<td>19.75</td>
<td>0.640</td>
<td>22.82</td>
<td>0.656</td>
<td>20.24</td>
</tr>
<tr>
<td>30%</td>
<td>19.42</td>
<td>0.639</td>
<td>21.87</td>
<td>0.686</td>
<td>19.87</td>
</tr>
<tr>
<td>40%</td>
<td>18.91</td>
<td>0.640</td>
<td>21.00</td>
<td>0.671</td>
<td>19.45</td>
</tr>
<tr>
<td>50%</td>
<td>18.09</td>
<td>0.656</td>
<td>19.94</td>
<td>0.717</td>
<td>18.91</td>
</tr>
<tr>
<td>60%</td>
<td>17.06</td>
<td>0.639</td>
<td>19.14</td>
<td>0.781</td>
<td>17.50</td>
</tr>
<tr>
<td>70%</td>
<td>14.94</td>
<td>0.640</td>
<td>18.15</td>
<td>0.921</td>
<td>15.36</td>
</tr>
<tr>
<td>80%</td>
<td>12.15</td>
<td>0.640</td>
<td>17.27</td>
<td>2.75</td>
<td>11.61</td>
</tr>
<tr>
<td>90%</td>
<td>8.68</td>
<td>0.640</td>
<td>16.14</td>
<td>13.03</td>
<td>8.11</td>
</tr>
</tbody>
</table>

Fig. 2. The original image: Lena, Girl and Baboon.

Remarks. 1. Case 1 indicates that $X_{ij}$ is not an extreme value. So, the $X_{ij}$ will not be a noise candidate and it will be retained on original value.
2. Case 2 indicates that $X_{ij}$ is not a salt impulse noise.
3. Case 3 indicates that $X_{ij}$ is not a pepper impulse noise.
4. If $X_{ij}$ does not satisfy Cases 1–3, the pixel will be regarded as a noisy pixel.

(c) As stated in the above remarks, if $X_{ij}$ does not satisfy Cases 1–3, the pixel will be regarded as a noisy pixel. For the noisy pixels we detected, we use a median filter for filtering. We retain the pixel value if the pixel is not a corrupted pixel. If $X_{ij}$ is corrupted, we use a median filter for filtering. We retain the pixel value if the pixel is not a corrupted pixel. If $X_{ij}$ is not a noise candidate and it will be retained on original value.

3. Experiment results

In this experiment, we choose some typical images to assess the performance of our algorithm. The performance is tested with different gray scale images such as Lena, Girl, and Babbon with size 256x256. In the simulation, images are corrupted by “salt” (with value 255) and “pepper” (with value 0) noise with equal probability. The noise levels are widely varied from 10% to 90% with increments of 10%, and the restoration performance are quantitative measured by peak signal-to-noise ratio (PSNR). The main concept of PSNR is to make comparison of the difference between original images and resulted images, which is defined as follows:

$$\text{PSNR} = 10 \times \log_{10} \frac{255^2}{\text{MSE}}$$

where

$$\text{MSE} = \frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} (S_{ij} - y_{ij})^2$$

where $S_{ij}$ is the pixel of the original image $S$, $y_{ij}$ is the pixel of the restored image $y$ at position $(ij)$. $S_{ij}$ is the mean square error, $m$ is the height of the image, $n$ is the width of the image, and 255 is the maximum pixel value for 8-bit image. Borland C++ builder 6.0 on a PC equipped with 1.73 GHz CPU and 1 GB RAM memory has been employed for the evaluation of computation time of all algorithms. The threshold $T$ that we used in our algorithm is 90.

The PSNR and computation time in seconds are evaluated for simulation, and a comparison of performance with various algorithms including SMF, AMF, CD-SMF, and DBA are presented in Tables 1–3.

It is observed that the proposed algorithm can provide better performance in terms of image quality and computation time. For every noise level, our algorithm achieves better PSNR than SMF, AMF, CD-SMF and DBA. Though the computation time of SMF and AMF is shorter than our algorithm, but their PSNR performance is not good. Extensive simulations show that our algorithm converges with $K = 1$ iteration for noise density below 30%, with $K = 3$ iterations for noise...
Fig. 3. Image restoration results of the Lena image. (a) 20% noise corrupted, (b) SMF, (c) AMF, (d) CD-SMF, (e) DBA and (f) proposed algorithm.

Fig. 2 shows the original test images. Figs. 3–5 show the subjective visual qualities of the filtered images using various algorithms for the image “Lena” corrupted by 20%, 50% and 80% impulse noise. It can be seen that for different noise density the SMF and the DBA suppress the impulses but introduce the obvious blur effect. The DBA provide stable restored image quality in wide noise density, but it causes some blurring effect in low noise density too. Our algorithm can achieve better PSNR visual quality while preserving image detail very well for a wide range of noise density. Moreover, from Fig. 5 we can observe some impulse spots remained in the resultant image for applying other algorithms in high noise density. Figs. 6 and 7 show the restoration results for the Girl image and the Baboon image corrupted by 50% noise, respectively. These results also reveal that the proposed algorithm exhibits better visual quality.

4. Conclusions

In this paper, a new algorithm of impulse detection and filtering is proposed. The proposed algorithm can not only achieve better image quality, but also have shorter computation time. Extensive simulations reveal that the proposed algorithm provides better performance than many of the existing switching median filters in terms of noise suppression and detail preservation. The proposed algorithm shows stable performance across a wide range of noise densities.
Fig. 4. Image restoration results of the Lena image. (a) 50% noise corrupted, (b) SMF, (c) AMF, (d) CD-SMF, (e) DBA and (f) proposed algorithm.
Fig. 5. Image restoration results of the Lena image. (a) 80% noise corrupted, (b) SMF, (c) AMF, (d) CD-SMF, (e) DBA and (f) proposed algorithm.
Fig. 6. Image restoration results of the Girl image. (a) 50% noise corrupted, (b) SMF, (c) AMF, (d) CD-SMF, (e) DBA and (f) proposed algorithm.
Fig. 7. Image restoration results of the Baboon image. (a) 50% noise corrupted, (b) SMF, (c) AMF, (d) CD-SMF, (e) DBA and (f) proposed algorithm.
varying from 10% to 90%, and is suitable for real-time implementation since it uses a fixed 3 × 3 window for filtering processing.

References


About the Author—SHUENN-SHYANG WANG received the M.S. and Ph.D. degree in Electrical Engineering from Tatung Institute of Technology, Taipei, in 1985 and 1987, respectively. He was an Instructor at Military Police School, Taipei, during 1987–1989. Since 1989, he has been working with the Department of Electrical Engineering, Tatung University, Taipei, where he is currently a Professor. His research interests include digital signal processing, image/video processing, VLSI implementation for DSP and network security.