Pattern Recognition 42 (2009) 2194 – 2202

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

# 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.



## 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:

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

where $p$ is noise density.

* Corresponding author.
*E-mail address:* sswang@ttu.edu.tw (S.-S. Wang).

0031-3203/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.patcog.2009.01.022

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 $X_{ij}$ 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 $r_{ij}$ ) is used for impulse detection, which can be represented as

$$r_{ij} = \min\{X_{ij} \otimes K_p| : p = 1 \text{ to } 4\} \tag{2.2}$$

where $K_P$ is the $p$th kernel, and $\otimes$ denotes a convolution operation. We compare $r_{ij}$ with a threshold $T$ to determine whether a pixel is corrupted, i.e.,

$$\alpha_{ij} = \begin{cases} 1, & r_{ij} > T \\ 0, & r_{ij} \leqslant T \end{cases} \tag{2.3}$$

If $\alpha_{ij} = 1$, 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:

$$W = \begin{bmatrix} a_0 & a_5 & a_3 \\ a_6 & a_1 & a_7 \\ a_4 & a_8 & a_2 \end{bmatrix} = \begin{bmatrix} X_{i-1,j-1} & X_{i,j-1} & X_{i,j+1} \\ X_{i-1,j} & X_{ij} & X_{i+1,j} \\ X_{i-1,j+1} & X_{i,j+1} & X_{i+1,j+1} \end{bmatrix} \tag{2.4}$$

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

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

Case 2 : $X_{ij} = \overline{a_4} \neq 225$

Case 3 : $X_{ij} = \overline{a_0} \neq 0$

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

| | | | | |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 |
| -1 | -1 | 4 | -1 | -1 |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 |

| | | | | |
|---|---|---|---|---|
| 0 | 0 | -1 | 0 | 0 |
| 0 | 0 | -1 | 0 | 0 |
| 0 | 0 | 4 | 0 | 0 |
| 0 | 0 | -1 | 0 | 0 |
| 0 | 0 | -1 | 0 | 0 |

| | | | | |
|---|---|---|---|---|
| -1 | 0 | 0 | 0 | 0 |
| 0 | -1 | 0 | 0 | 0 |
| 0 | 0 | 4 | 0 | 0 |
| 0 | 0 | 0 | -1 | 0 |
| 0 | 0 | 0 | 0 | -1 |

| | | | | |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | -1 |
| 0 | 0 | 0 | -1 | 0 |
| 0 | 0 | 4 | 0 | 0 |
| 0 | -1 | 0 | 0 | 0 |
| -1 | 0 | 0 | 0 | 0 |

**Fig. 1.** Four 5×5 convolution kernels.

**Table 1**
PSNR (db) and computation time (seconds) for various algorithms for Lena image at different noise density.

| Noise density | Algorithm | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | SMF | | AMF | | CD-SMF | | DBA | | Proposed algorithm | |
| | PSNR | Time | PSNR | Time | PSNR | Time | PSNR | Time | PSNR | Time |
| 10% | 29.93 | 0.610 | 35.57 | 0.639 | 32.94 | 0.938 | 31.89 | 0.828 | 39.09 | 0.764 |
| 20% | 28.36 | 0.625 | 32.94 | 0.639 | 30.97 | 0.823 | 29.80 | 0.717 | 34.32 | 0.734 |
| 30% | 26.42 | 0.625 | 30.72 | 0.639 | 29.62 | 0.823 | 28.17 | 0.875 | 32.00 | 0.735 |
| 40% | 23.99 | 0.625 | 28.94 | 0.657 | 27.83 | 0.828 | 27.41 | 0.890 | 30.27 | 0.734 |
| 50% | 21.74 | 0.625 | 27.37 | 0.671 | 26.00 | 0.938 | 26.89 | 0.906 | 28.54 | 0.735 |
| 60% | 19.74 | 0.640 | 26.17 | 0.718 | 22.55 | 0.796 | 26.17 | 0.889 | 27.33 | 0.75 |
| 70% | 15.98 | 0.609 | 24.18 | 0.859 | 17.39 | 0.843 | 25.52 | 0.907 | 26.00 | 0.765 |
| 80% | 13.04 | 0.609 | 22.39 | 1.531 | 12.37 | 0.86 | 24.49 | 0.938 | 24.53 | 0.764 |
| 90% | 8.94 | 0.609 | 20.17 | 10.60 | 8.46 | 0.764 | 22.20 | 0.906 | 22.20 | 0.765 |

**Table 2**
PSNR (db) and computation time (seconds) for various algorithms for Girl image at different noise density.

| Noise density | Algorithm | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | SMF | | AMF | | CD-SMF | | DBA | | Proposed algorithm | |
| | PSNR | Time | PSNR | Time | PSNR | Time | PSNR | Time | PSNR | Time |
| 10% | 31.89 | 0.609 | 37.33 | 0.639 | 35.34 | 0.823 | 33.98 | 0.859 | 42.11 | 0.735 |
| 20% | 31.05 | 0.625 | 35.57 | 0.656 | 33.07 | 0.86 | 31.59 | 0.75 | 37.33 | 0.734 |
| 30% | 28.73 | 0.671 | 33.65 | 0.640 | 31.14 | 0.938 | 30.64 | 0.859 | 34.90 | 0.75 |
| 40% | 26.73 | 0.639 | 31.79 | 0.656 | 29.20 | 0.86 | 29.80 | 0.859 | 32.81 | 0.75 |
| 50% | 23.98 | 0.625 | 30.34 | 0.688 | 27.26 | 0.823 | 29.09 | 0.890 | 31.50 | 0.75 |
| 60% | 20.32 | 0.609 | 28.49 | 0.75 | 22.95 | 0.823 | 28.26 | 0.890 | 29.87 | 0.75 |
| 70% | 16.67 | 0.625 | 27.02 | 0.875 | 16.05 | 0.836 | 27.26 | 0.875 | 28.73 | 0.75 |
| 80% | 12.66 | 0.625 | 24.57 | 3.03 | 11.52 | 0.796 | 26.08 | 1.06 | 27.4 | 0.765 |
| 90% | 8.26 | 0.610 | 23.12 | 11.2 | 7.34 | 0.836 | 24.06 | 0.938 | 24.84 | 0.74 |