Scientia Iranica D (2012) 19 (6), 1738–1745

Sharif University of Technology

Scientia Iranica

Transactions D: Computer Science & Engineering and Electrical Engineering

www.sciencedirect.com

Research note

New algorithms for recovering highly corrupted images with impulse noise

A. Jourabloo *, **A.H. Feghahati, M. Jamzad**

Department of Computer Engineering, Sharif University of Technology, Tehran, P.O. Box 11155-8639, Iran

Received 13 July 2011; revised 20 December 2011; accepted 17 July 2012

KEYWORDS

Noise removal;
Image processing;
Impulse noise;
Filtering;
Salt & pepper;
Restoring an image.

Abstract In this work, we present a new method of noise removal which is applied on images corrupted by impulse noise. This new algorithm has a good trade-off between quantitative and qualitative properties of the recovered image and the computation time. In this new method, the corrupted pixels are replaced by using a median filter or, they are estimated by their neighbors' values. Our proposed method shows better results especially in very high density noisy images than Standard Median Filter (SMF), Adaptive Median Filter (AMF) and some other well-known filters for removing impulse noise. Experimental results show the superiority of the proposed algorithm in measures of PSNR and SSIM, specifically when the image is corrupted with more than 90% impulse noise.

1. Introduction

Corruption of images by noise is a regular phenomenon in image processing field. One of the most common noise types which corrupt images during transmission or acquisition is impulse noise, also known as salt & pepper [1]. In this type of noise, a pixel gets the minimum or maximum value that can take in a dynamic range of available values.

It is not surprising that many methods have been developed to remove noisy data from a signal, especially for removing impulse noise because of its presence in many acquisition devices or noisy channels for transmission. Noise removal is a common pre-processing step in which the recovered signal, in this case the image, would have better visual properties or have more meaningful data, thus, it can be used as a proper input for other algorithms such as edge detectors.

It is known that if the noise is not additive, linear filtering fails [2], so most of the algorithms use a non-linear approach to achieve better results. In fact, median filtering, also known as Standard Median Filtering (SMF), is a good choice to achieve reasonable results, but, the problem arises when the ratio of the noise is higher than 50% in which there is a good chance that median is a corrupted pixel rather than a clean one. Another type of filtering is Adaptive Median Filtering (AMF) [3] in which the window size changes to get a non-noisy pixel as median, but this type of process is time-consuming and it propagates the error in terms that if an image is highly corrupted and we use AMF, we surely lose the edge information. To overcome this problem, symmetric and asymmetric trimmed median filters have been proposed. Although these filters provide better results, they fail when the image is corrupted with high ratio of noise such as 90% or more.

The rest of this paper is organized as follows. In Section 2, the main advantages and disadvantages of recently proposed algorithms for removing impulse noise are mentioned. In Section 3, the proposed algorithms are explained, in Section 4, the experimental results are discussed, and finally, the conclusion of our work is presented in Section 5.

2. Previous works

As mentioned earlier, many algorithms have been proposed to restore images which are corrupted by impulse noise [1–13]. In this section, we analyze some of the most well-known algorithms by discussing their advantages and disadvantages. We used the advantages of these algorithms to develop our new methods.

In AMF [3], the window size is repeatedly increased until it can find a non-noisy pixel as median or it reaches the maximum

* Corresponding author.
E-mail addresses: Jourabloo@ce.sharif.edu (A. Jourabloo), Feghahati@ce.sharif.edu (A.H. Feghahati), Jamzad@sharif.edu (M. Jamzad).
Peer review under responsibility of Sharif University of Technology.

Production and hosting by Elsevier

1026-3098
doi:10.1016/j.scient.2012.07.016

```
For two iterations
    For each pixel
        If the pixel is noisy Then
        Use an adaptive median filter with max window size of 9
End Iterations
    For the remaining noisy pixels
        Apply an adaptive median filter without a window size limit
```

Figure 1: Overall algorithm of the first proposed method.

N	N	N	240
N	◯	Δ	N
N	N	N	N
N	190	N	N

N	N	N	240
N	◯	240	N
N	N	N	N
N	190	N	N

Figure 2: Sample windows of a highly noisy images. Noisy pixels are shown with N, and we should estimate a value for the pixels shown with ◯ and Δ.

size of the window. As mentioned earlier, this method is very time-consuming due to increasing the window size and has poor results in high density noises because it replaces a pixel with another pixel which is in a far distance from it. Generally, in images, each small region has similar properties within itself, and there is little alternation of information in small neighboring regions [4]. Because of this property, an image can be seen as a soft picture. By replacing a far pixel with a noisy one, this smoothness will be lost, which mainly results in edge loss and blurring.

In [2], a decision-based algorithm, known as DBA, has been introduced. The authors' method acts like a trimmed median filter and whenever the method cannot detect a non-noisy pixel, it uses the last processed pixel as a substitution of current pixel. The advantage of this approach is that the algorithm will surely replace the noisy pixel with another non-noisy pixel. However, when there is a dense noisy region in the image, there is a high possibility that a non-noisy (i.e. clean) pixel cannot be found, so in this situation a noisy pixel must be replaced by the last processed clean pixel. As the algorithm proceeds in a dense area, the noisy pixels are replaced with the last clean one. In this approach of replacement, the error is propagated, i.e. the smoothness and fine details get lost as the noise pixels are replaced with clean ones located in a relatively far distance. Despite this drawback, DBA has a constant and stable runtime.

In [5], the combination of rand-ordered absolute difference has been used. It has good results in images that have many edges, but it needs to set up some external values. It also fails in high density noisy images because it cannot choose a none-noisy pixel due to its use of standard median filter for obtaining a reference image.

In [6,7], a decision-based algorithm is proposed. For each pixel, it decides if it is low, medium or high noisy pixel. It has designed a weighted filter for low noisy pixels, a median filter for medium noisy pixels and a mean filter for high density noisy pixels. Because of the mean filter and non-iterative approach, it does not produce fair results for high density noises.

In [5,8], the algorithm has to choose some parameters such as β by simulation; thus, the algorithm must run multiple times to tune with the best parameters, and therefore, have the ultimate performance.

In [9], the authors tried to estimate the proper value of the noisy pixel with respect to the preserving edges. They use a derivative approach to find the best direction and to estimate the value, but, the drawback of their work is in highly density images; when there is no meaningful data in a window, their method replaces a noisy pixel with 0 or 127 or 255.

In [10], the authors use an iterative approach which means that they try to replace a noisy pixel by its median. If the median is noisy too, they neither increase nor adapt the size of window but proceed to the next pixel. This action is performed on the entire image. The advantage of this method is that in each sequence it does not check pixels too far from the window center, so it does not lose the fine details or edges. Its disadvantage is that it is very time-consuming and in high ratio of noises such as 97%, it fails because of the fact that a pixel is estimated by its neighbor and its neighbor estimated by its neighbor and so forth. In this approach, because many pixels are estimated by their far neighbors it gives a poor result in highly noisy images.

In [11], a method which tends to preserve details and edges is presented. The presented method works well with noise ratio less than 50%, but it has problem with higher noise ratios. Their algorithm has problem to discriminate noise pixels and real ones in candidate selection phase.

The authors of [12] used a switching median filter and used predefined window sizes for different ration of noise presence. This method achieves good results, but like other mentioned methods, it fails on highly corrupted images. This method has to repeatedly increase window size in highly noisy images, thus, it loses fine details.

In [13], a novel, fast and accurate method is presented. This method is tuned for recovering images from high ratio of noise although it has good results with low ratio noises as well. They proposed a novel method for candidate selection based on counting the maxima and minima pixels. The speed of their method comes from the fact that they use different fixed window sizes based on the noise ratio. Their results are the most competitive ones compared to ours.

In [14], we have presented a brief version of the following proposed method. In the current work, we present a comprehensive description on the methods in [14] and also conduct more experimental results and compare our methods with the most recently published papers.

3. Proposed methods

In this work we try to combine the advantages of the method in [10] and adaptive median filter in order to achieve better results in terms of computation time and visual quality. In salt & pepper noise, a pixel can get its highest value — salt noise — or its lowest value — pepper noise — by some probability. Eq. (1) shows how to create salt & pepper noise on an image.

$$f\left(X_{ij}\right) = f\left(x\right) = \begin{cases} \dfrac{p}{2} & X_{ij} = 0 \\ 1-p & X_{ij} = I_{ij} \\ \dfrac{p}{2} & X_{ij} = 255. \end{cases} \tag{1}$$

In step one, the adaptive median filter approach is used with some modification. First of all, the value of each pixel is examined. If it falls between 1 and 254, inclusively, it remains unchanged. Otherwise, the pixels with intensity value of 0 or 255 drop, and then the adaptive median filter will be used. It is a reasonable assumption that the pixels with values of 0 or

Table 1: Results of PSNR and SSIM for different filters for Baboon.

Noise density (%)	Proposed Algorithm 1 (PA1)		Proposed Algorithm 2 (PA2)		DBA [2]		Method [10]		Method [6]		Method [13]	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
10	31.66	0.989	25.04	0.950	31.30	0.987	31.97	0.989	30.44	0.987	29.35	0.982
20	28.45	0.975	23.45	0.918	27.62	0.968	28.87	0.976	27.41	0.972	26.45	0.963
30	26.60	0.958	22.95	0.901	25.48	0.943	27.06	0.961	25.57	0.955	24.89	0.942
40	25.12	0.936	22.60	0.885	23.73	0.912	25.61	0.941	24.21	0.934	23.50	0.915
50	23.89	0.909	22.26	0.869	22.36	0.874	24.43	0.915	23.14	0.908	22.28	0.859
60	22.71	0.873	21.79	0.844	21.11	0.824	23.32	0.878	22.06	0.872	21.40	0.807
70	21.65	0.828	21.17	0.810	20.03	0.760	22.32	0.830	21.10	0.828	20.60	0.751
75	21.08	0.797	20.77	0.783	19.45	0.713	21.78	0.797	20.57	0.796	20.18	0.714
80	20.49	0.756	20.32	0.748	18.85	0.653	21.22	0.754	20.05	0.756	19.73	0.677
85	19.78	0.699	19.74	0.695	18.14	0.574	20.55	0.690	19.41	0.699	19.19	0.628
90	18.94	0.619	19.01	0.620	17.26	0.459	19.75	0.600	18.59	0.621	18.31	0.563
95	17.97	0.492	18.03	0.498	16.14	0.309	18.72	0.455	16.81	0.485	17.13	0.466
96	17.75	0.458	17.86	0.467	15.83	0.272	18.44	0.416	15.96	0.448	16.81	0.435
97	17.36	0.420	17.45	0.431	15.31	0.226	17.98	0.370	12.73	0.332	16.38	0.403
98	17.20	0.366	17.09	0.369	14.89	0.173	17.58	0.312	7.77	0.057	16.01	0.358

Table 2: Results of computation time for different filters for Baboon.

Noise density (%)	Proposed Algorithm 1 (PA1) Time (s)	Proposed Algorithm 2 (PA2) Time (s)	DBA [2] Time (s)	Method [10] Time (s)	Method [6] Time (s)	Method [13] Time (s)
10	1.01	11.90	7.11	0.42	1.59	3.61
20	2.06	12.52	8.13	0.79	2.00	4.94
30	2.99	12.56	7.25	1.26	2.38	6.08
40	3.86	12.52	8.05	1.86	2.70	6.90
50	4.80	12.75	7.24	2.32	3.04	10.15
60	5.75	12.49	7.28	3.04	3.51	10.28
70	6.95	12.61	13.18	4.16	4.10	10.73
75	7.85	13.59	12.46	4.82	4.67	11.45
80	9.31	14.00	12.89	6.05	5.28	11.76
85	10.94	14.04	12.94	8.16	5.98	11.84
90	14.26	14.17	7.53	12.62	6.70	12.04
95	21.67	13.66	12.93	29.06	6.63	12.06
96	25.62	13.74	13.13	38.13	6.93	11.95
97	28.40	14.62	13.42	51.65	5.82	12.17
98	35.98	14.70	13.67	100.14	5.44	13.83

255 are noisy, because in practice, there is a modest probability that a clean pixel reaches these values. By this mean, we can estimate the noisy pixel from its neighbors, but we do not go so far distance from the noisy pixel. Eq. (2) shows how we choose a neighborhood window.

$$w_N\,(i,j) = \{(k,l)\ |-N \le k-i,\ l-j\ \le N\} \qquad (2)$$

where ω is a window of size $N \times N$. First, the initial size of window is 3×3 and we go as far as 9×9. By this approach, not only will we not lose fine details and edges, but we also gain a good estimate for noisy data. If a good estimate is found, we will replace it; otherwise we will proceed to the next pixel. By the end of this step, we may have some noisy pixels unchanged, so we repeat this procedure for the second time. After that, we will use adaptive median filtering to remove the remaining noisy pixels. The entire algorithm is depicted in Figure 1.

The underlying principle of this approach is that the clean pixels near the noisy one have more meaningful information because images have local smoothness property. Consider Figure 2. If we center a window of size 3×3 in location depicted by circle and use a standard median filter, the result is also noisy so we have to proceed to the next pixel that is depicted by a triangle. In this stage, we can replace the noisy pixel. We proceed until we reach to end of the image.

In the second pass, we can replace the noisy pixel at location depicted by circle. The original spatial location of the replaced value is in the first row and the fourth column. The Euclidian distance is $\sqrt{5}$ in this case. In our approach, we use adaptive median filter, so in the first pass, we replace the noisy pixel at the location of the circle with a value of the pixel at 4th row and 2nd column (in this case 190), which has a Euclidian distance of 2, so the gain of the information is higher than the approach used in [10]. To maintain a short distance between the noisy pixel and the pixel to be replaced for the noisy pixel, we do not increase the window size of adaptive filter more than 9×9.

Our proposed method has high PSNR results, but it has high computation time in highly dense noisy images. To obtain a good tradeoff between the computation time and accuracy, we proposed a second method in which we introduced a new approach that has fixed computation time at the cost of losing some details of image. This approach is summarized in the following steps:

1. Initialize the window size (3×3 window).
2. Divide gray levels of the pixels in the window by 4; this dividing is done to add robustness to find the noisy values, e.g. the result of dividing 121 and 122 are the same.
3. Find max value and min value within the window. Denote these values as $g_{\min}$ and $g_{\max}$.

```
I: is the noisy image
For each pixel I
Calculate the window size
Calculate α and β

If (α > β)
    K = AVG;
Else
    K = MED;
end
calculate τ
If (|K − I(i, j)| > τ)
    Restored_Image (i, j) = K;
Else
    Restored_Image (i, j) = I(i, j);
End
```

Figure 3: The algorithm for the second proposed method.

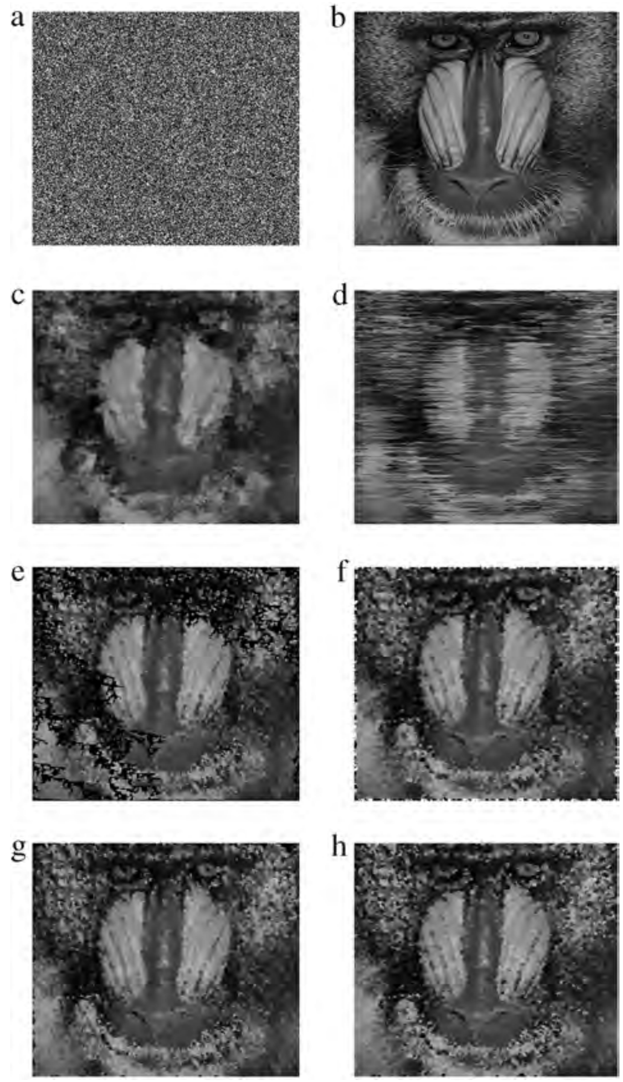

Figure 4: (a) Baboon corrupted by 97%; (b) original image; (c) using method in [10]; (d) DBA; (e) using method in [6]; (f) using method in [13]; (g) PA 2; and (h) PA 1.

4. If the gray level of the centered pixel is greater than $g_{\min}$ and less than $g_{\max}$, then the pixel is not noisy, so we proceed to the next pixel and go to step 1.

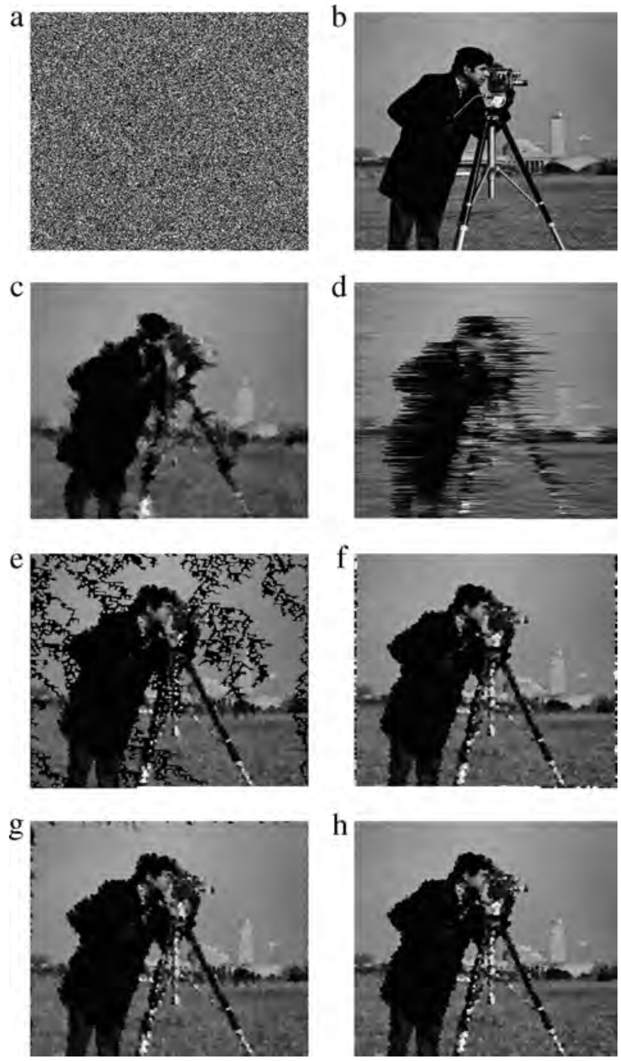

Figure 5: (a) Cameraman corrupted by 97%; (b) original image; (c) using method in [10]; (d) DBA; (e) using method in [6]; (f) using method in [13]; (g) PA 2; and (h) PA 1.

5. If we reach this step, it means that the centered pixel is noisy. We will find other noisy pixels in the window, base on step 3.
6. If the size of window is 3×3, then increase the window size to 5×5 and go to step 2.
7. If all of the pixels in the window are noisy, and the window size is 5×5 then use the mean of the three previously calculated neighbors as the gray level value of the noisy pixel and proceed to the next pixel.
8. Calculate the mean and median of the window. Find which of them is near the center of interval $[g_{\min}, g_{\max}]$. By calculation of α and β in Eqs. (4) and (5), we can find which one of the mean values or median values is near the center of the mentioned interval. The min of α and β shows the preferred value.

$$a = |g_{\max} + g_{\min} - 2 \times AVG| \,, \tag{3}$$

$$b = |g_{\max} + g_{\min} - 2 \times MED| \,. \tag{4}$$

The noisy pixel will be replaced with the value computed from step 8 if the difference between its intensity and the value from

Table 3: Results of PSNR and SSIM for different filters for Barbara.

Noise density (%)	Proposed Algorithm 1 (PA1)		Proposed Algorithm 2 (PA2)		DBA [2]		Method [10]		Method [6]		Method [13]	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
10	33.56	0.987	26.37	0.933	32.88	0.985	33.96	0.989	31.89	0.986	30.61	0.983
20	30.47	0.974	25.32	0.914	29.44	0.968	31.04	0.977	28.93	0.973	27.67	0.969
30	28.41	0.956	24.85	0.900	27.22	0.945	29.06	0.962	27.29	0.957	26.27	0.952
40	26.98	0.937	24.55	0.890	25.48	0.919	27.72	0.945	25.91	0.939	24.90	0.930
50	25.67	0.914	24.18	0.879	23.94	0.888	26.48	0.924	24.72	0.918	24.18	0.913
60	24.63	0.891	23.77	0.866	22.62	0.851	25.42	0.900	23.71	0.894	23.30	0.885
70	23.54	0.858	23.15	0.844	21.33	0.797	24.34	0.866	22.71	0.861	22.50	0.850
75	22.94	0.836	22.73	0.826	20.62	0.758	23.79	0.843	22.10	0.838	22.02	0.828
80	22.40	0.816	22.31	0.811	19.84	0.713	23.28	0.823	21.56	0.818	21.53	0.808
85	21.76	0.785	21.80	0.783	18.87	0.641	22.50	0.783	21.01	0.787	20.94	0.775
90	20.95	0.735	21.00	0.737	17.49	0.525	21.55	0.719	20.11	0.737	19.83	0.730
95	19.85	0.655	19.61	0.650	15.50	0.345	19.48	0.585	18.26	0.637	18.39	0.654
96	19.68	0.640	19.35	0.633	15.04	0.298	19.06	0.553	16.80	0.594	18.20	0.641
97	19.20	0.594	18.62	0.577	14.26	0.237	17.90	0.480	11.56	0.348	17.50	0.593
98	18.55	0.548	17.64	0.513	13.41	0.174	16.92	0.403	8.29	0.088	16.58	0.544

Table 4: Results of computation time for different filters for Barbara.

Noise density (%)	Proposed Algorithm 1 (PA1) Time (s)	Proposed Algorithm 2 (PA2) Time (s)	DBA [2] Time (s)	Method [10] Time (s)	Method [6] Time (s)	Method [13] Time (s)
10	1.03	12.40	7.27	0.42	1.62	3.68
20	2.05	12.62	7.33	0.82	1.96	5.00
30	3.38	12.51	7.45	1.26	2.35	5.92
40	3.86	13.13	7.30	1.76	2.76	8.04
50	4.84	13.52	7.20	2.25	3.12	10.48
60	6.05	13.84	7.75	3.02	4.27	10.59
70	7.80	17.72	7.86	4.44	5.06	14.38
75	10.15	17.18	14.46	4.86	4.78	14.89
80	11.05	16.29	9.88	7.98	6.79	12.59
85	11.41	15.52	13.43	8.30	5.85	13.32
90	17.13	14.24	16.34	15.33	7.43	12.92
95	24.96	15.66	17.35	33.41	6.67	12.42
96	25.32	16.96	13.62	47.87	8.05	16.41
97	35.67	15.89	17.04	61.61	7.00	12.88
98	43.25	18.63	13.85	111.43	7.07	17.42

step 8 is higher than a threshold τ. This threshold is calculated according to the following equations:

$$P\left(Z_k\right) = \frac{n_k}{M \times N}, \tag{5}$$

$$m = \sum_{K=0}^{L-1} z_K \times P\left(z_K\right), \tag{6}$$

$$\sigma^2 = \sum_{K=0}^{L-1} \left(z_K - m\right)^2 \times P\left(z_K\right), \tag{7}$$

$$\tau = \frac{100}{\sqrt{\sigma/30}} - 40, \tag{8}$$

where n_k is the number of pixels in image which their intensity value is k; m is the average intensity of the image; σ^2 is variance of the picture; and τ is the threshold on which we decide to substitute the noisy pixel. The algorithm in Figure 3 shows this method in detail.

4. Experimental results

Here we report the performance of the proposed algorithms, DBA [3,6,10,13] in terms of PSNR, computation time and Structural Similarity Index Measure (SSIM). These experiments are carried out on six standard images, Lena and Gold hill and pepper as representatives for low detail images, cameraman for medium detail images, baboon and Barbara for high detail images. All images are grayscale ones with intensity values in the range of $[0, 255]$. Images are of size 512×512. Eqs. (9)–(11) are used to calculate the quantitative and qualitative measures. We have developed [2,6,10,12] to compare our results with them. It is important to note that the results of PSNR are highly dependent on the noisy image which means two version of 90% noise of the same image may have different PSNRs after denoising by difference of more than 1.5 units. One of the main advantages of our proposed algorithms is that for a given noisy image, there is no need to tune a parameter by trial to achieve the best result.

$$\mathrm{MSE} = \frac{\sum_i \sum_j \left(x\left(i,j\right) - y\left(i,j\right)\right)^2}{M \times N}, \tag{9}$$

$$\mathrm{PSNR} = 10 \log_{10}\left(\frac{255^2}{\mathrm{MSE}}\right), \tag{10}$$

$$\mathrm{SSIM} = \frac{\left(2\mu_x\mu_y + c_1\right)\left(2\sigma_{xy} + c_2\right)}{\left(\mu_x^2 + \mu_y^2 + c_1\right)\left(\sigma_x^2 + \sigma_y^2 + c_2\right)}, \tag{11}$$

where:

$y\left(i,j\right)$ is the intensity of the noisy picture;
$x\left(i,j\right)$ is the intensity of the restored image;

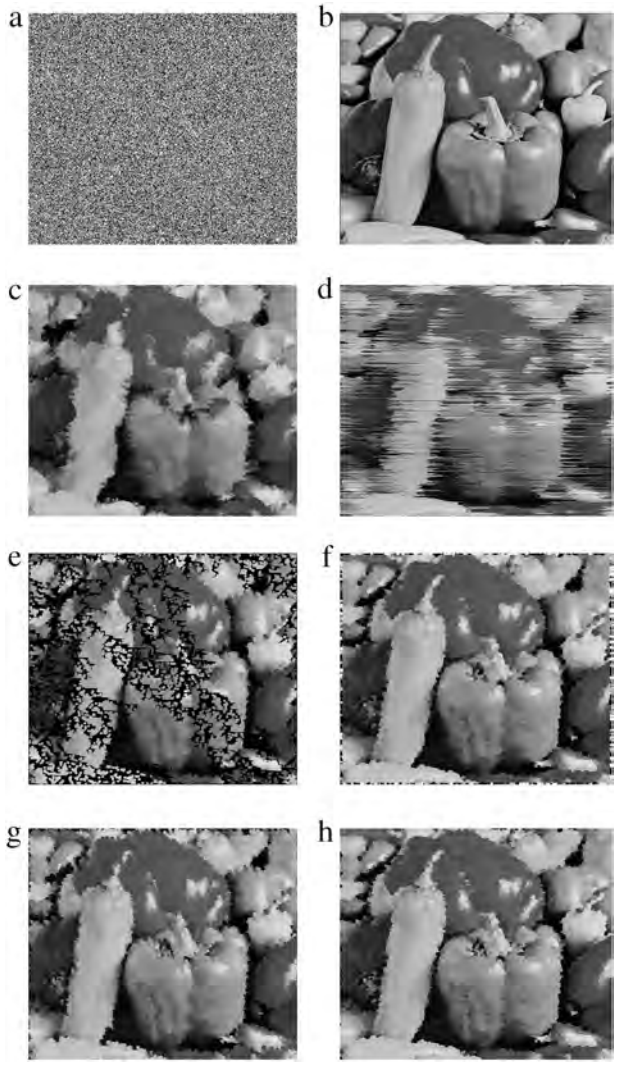

Figure 6: (a) Pepper corrupted by 97%; (b) original image; (c) using method in [10]; (d) DBA; (e) using method in [6]; (f) using method in [13]; (g) PA 2; and (h) PA 1.

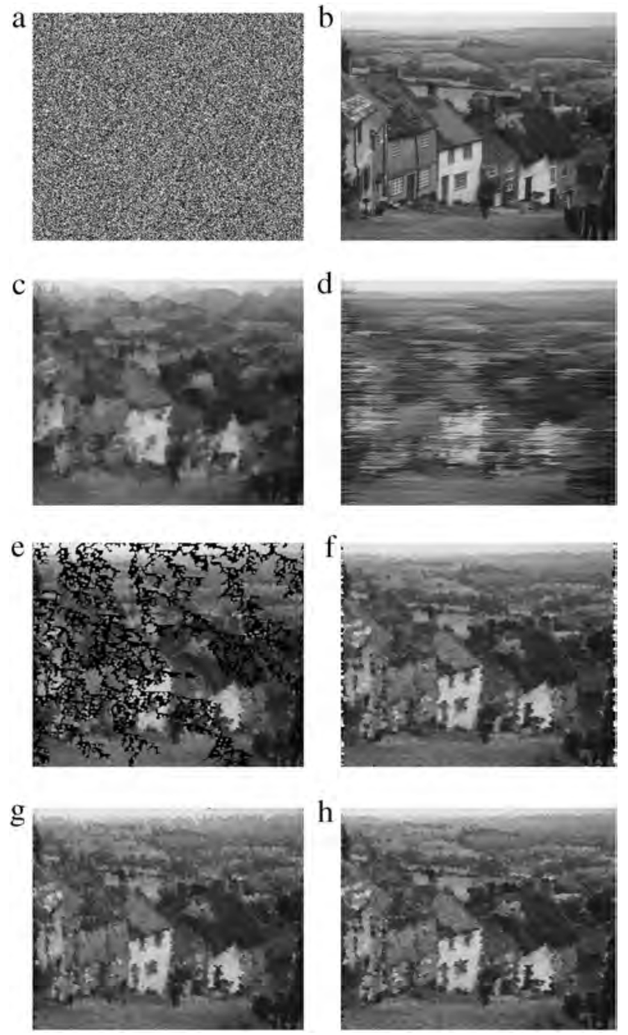

Figure 7: (a) Gold hill corrupted by 97%; (b) original image; (c) using method in [10]; (d) DBA; (e) using method in [6]; (f) using method in [13]; (g) PA 2; and (h) PA 1.

ϑ (i, j) is the intensity of the original image;
$M \times N$ is the size of the picture;
μ_x is the average of x;
μ_y is the average of y;
σ_x^2 is the variance of x;
σ_y^2 is the variance of y;
σ_{xy} is the covariance of x and y;
$c_1 = (k_1L)^2, c_2 = (k_2L)^2$ are two variables to stabilize the division with weak denominator;
L is the dynamic range of the pixel values (typically this is $2^{\#bits\ per\ pixel} - 1$);
$k_1 = 0.01$ and $k_2 = 0.03$ by default.

For measuring SSIM, we have used the SSIM code provided by Wang et al. [15].

We have tested our methods and some state-of-the-art algorithms in images corrupted by salt & pepper noise with densities 10% through 98%.

Figure 4 depicted the restored images for baboon corrupted by 97% noise. Table 1 shows our Proposed Algorithms (PAs) by measure of PSNR and SSIM. As it is obvious, we get high PSNR for very dense noisy environments. As shown in Table 2, our method has comparable running time with other methods.

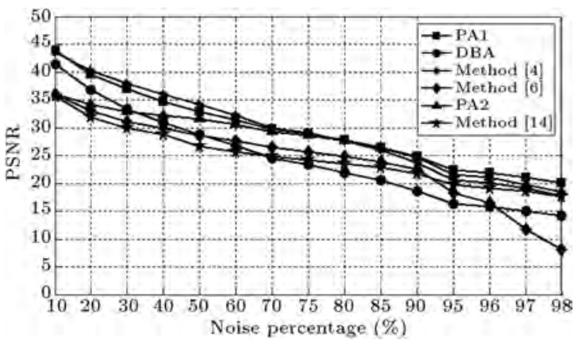

Figure 8: Results of PSNR for different filters for cameraman.

Figures 5–7 show the qualitative results of our methods and other methods for noise density of 97% for cameraman, pepper and Gold hill. In Figure 5, PSNR and SSIM of our first proposed method are 21.03 and 0.765, PSNR and SSIM for our second proposed method are 19.7 and 0.727 while the best competitive methods have PSNR equal to 19.04 and SSIM equal to 0.746.

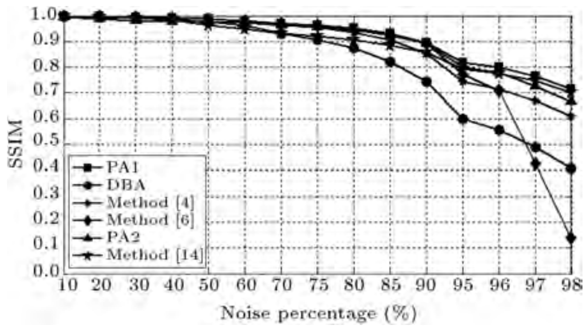

Figure 9: Results of SSIM for different filters for cameraman.

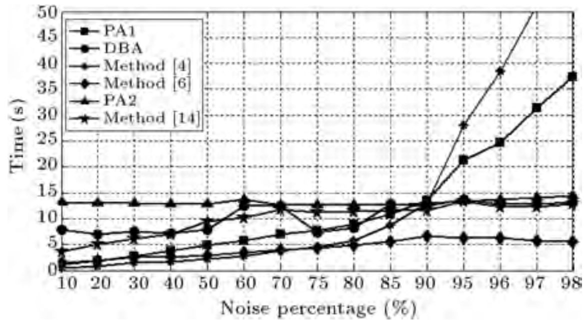

Figure 10: Results of computation time for different filters for cameraman.

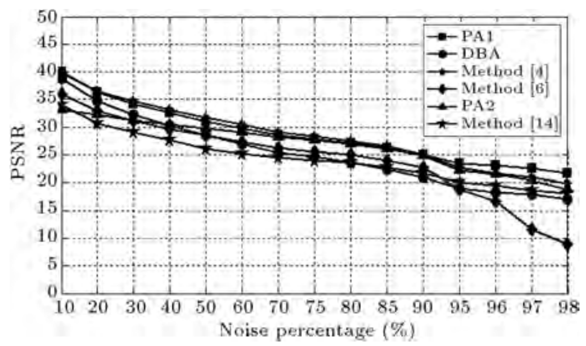

Figure 11: Results of PSNR for different filters for Gold hill.

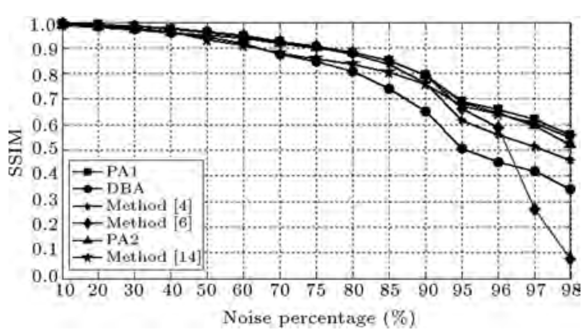

Figure 12: Results of SSIM for different filters for Gold hill.

Figures 6 and 7 show the restored images for pepper and Gold hill.

Figures 8–10 plot PSNR, SSIM and computation time obtained by these methods when variable noise density is applied to cameraman image. As it can be seen, our methods have superior PSNR values and our first proposed method has the best SSIM result. Figures 11–13 show PSNR, SSIM and computation time for image. Figures 14–16 show the same measures for Lena. Tables 3 and 4 provide the detail results for Barbara.

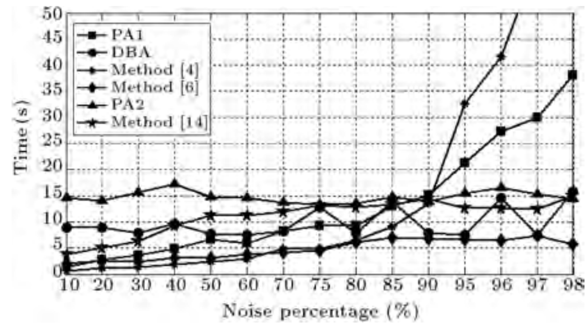

Figure 13: Results of computation time for different filters for Gold hill.

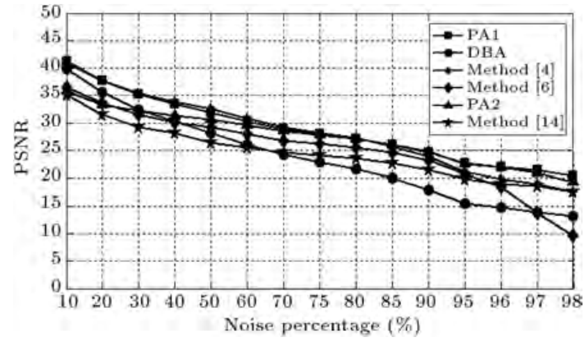

Figure 14: Results of PSNR for different filters for Lena.

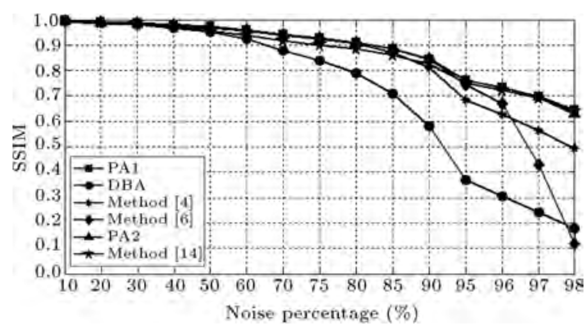

Figure 15: Results of SSIM for different filters for Lena.

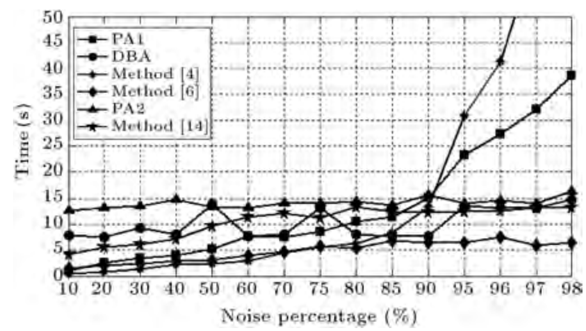

Figure 16: Results of computation time for different filters for Lena.

5. Conclusion

In this work, we have proposed a new algorithm for recovering highly corrupted images by impulse Noise. In this method, we have combined the advantages of standard and state-of-the-art methods to achieve high results in terms of PSNR and SSIM. Furthermore, this method has reasonable computation time in high noise densities. The computation time of our proposed methods and the methods discussed in [2,6,10,12] are compared in Tables 2 and 4 for different ratio of impulse noise. The first proposed method has growing computation time, but has much better time than the method in [10] which has the nearest PSNR result. The second proposed method has a constant computation time which makes it desirable for real-time applications.

References

[1] Pitas, I. and Venetsanopoulos, A.N. "Order statistics in digital image processing", *Proceedings of the IEEE*, 80(12), pp. 1893–1921 (1992).
[2] Srinivasan, K.S. and Ebenezer, D. "A new fast and efficient decision-based algorithm for removal of high-density impulse noises", *IEEE Signal Processing Letters*, 14(3), pp. 189–192 (2007).
[3] Hwang, H. and Haddad, R.A. "Adaptive median filters: new algorithms and results", *IEEE Transactions on Image Processing*, 4(4), pp. 499–502 (1995).
[4] Sun, T. and Neuvo, Y. "Detail-preserving median based filter in image processing", *Pattern Recognition Letters*, 15(4), pp. 341–347 (1994).
[5] Wang, S.S. and Wu, C.H. "A new impulse detection and filtering method for removal of wide range impulse noises", *Pattern Recognition*, 42(9), pp. 2194–2202 (2009).
[6] Wang, C., Chen, T. and Qu, Z. "A novel improved median filter for salt-and-pepper noise from highly corrupted images", *IEEE International Symposium on Systems and Control in Aeronautics and Astronautics*, ISSCAA, Harbin, China, pp. 718–722 (2010).
[7] Han, W. and Lin, J. "Minimum–maximum exclusive mean (MMEM) filter to remove impulse noise from highly corrupted images", *Electronics Letters*, 33(2), pp. 124–125 (1997).
[8] Chan, R.H., Ho, C. and Nikolova, M. "Salt-and-pepper noise removal by median-type noise detectors and detail-preserving regularization", *IEEE Transactions on Image Processing*, 14(10), pp. 1479–1485 (2005).
[9] Chen, P. and Lien, C. "An efficient edge-preserving algorithm for removal of salt-and-pepper noise", *IEEE Signal Processing Letters*, 15, pp. 833–836 (2008).
[10] Majid, A. and Tariq Mahmood, M. "A novel technique for removal of high density impulse noise from digital images", *IEEE International Conference on Emerging Technologies*, ICET, Islamabad, Pakistan, pp. 139–143 (2010).
[11] Duan, D., Mo, Q., Wan, Y. and Han, Z. "A detail preserving filter for impulse noise removal", *IEEE International Conference on Computer Application and System Modeling*, ICCASM, Taiyuan, China, pp. V2-265–V2-268 (2010).
[12] Ng, P. and Ma, K. "A switching median filter with boundary discriminative noise detection for extremely corrupted images", *IEEE Transactions on Image Processing*, 15(6), pp. 1506–1516 (2006).
[13] Fabijańska, A. and Sankowski, D. "Noise adaptive switching median-based filter for impulse noise removal from extremely corrupted images", *Image Processing, IET*, 5(5), pp. 472–480 (2011).
[14] Feghahati, A.H., Jourabloo, A. and Jamzad, M. "New algorithms for recovering highly corrupted images with impulse noise", *2nd International Conference on Contemporary Issues in Computer and Information Sciences*, CICIS'11, Zanjan, Iran, pp. 14–18, May 31–June 2 (2011).
[15] Wang, Z., Bovik, A.C., Sheikh, H.R. and Simoncelli, E.P. "Image quality assessment: from error visibility to structural similarity", *IEEE Transactions on Image Processing*, 13(4), pp. 600–612 (2004).

Amin Jourabloo received his B.Sc. from Ferdowsi University of Mashhad and is pursuing his M.Sc. in Sharif University of Technology. His main research interests are machine learning, sparse representation and machine vision.

Amir Hossein Feghahati received his B.S. in Computer Engineering from Iran University of Science and Technology. He is a student of M.S. in Artificial Intelligence in Sharif University of Technology. His main research interests are machine learning, computer vision and Sparse Representation.

Mansour Jamzad has obtained his M.S. degree in Computer Science from McGill University, Montreal, Canada and his Ph.D. in Electrical Engineering from Waseda University, Tokyo, Japan in 1989. For a period of two years after graduation, he worked as a Post Doctorate Researcher in the Department of Electronics and Communication Engineering, Waseda University. He was a visiting researcher at Tokyo Metropolitan Institute of Gerontology during 1990–1994.

He joined Department of Computer Engineering, Sharif University of Technology, Tehran, Iran, as faculty member in 1995. He has been teaching digital image processing and machine vision graduate courses for the last 17 years. His main research interests are digital image processing, machine vision and its applications. In recent years, he has been very active in research fields such as watermarking, content-based image retrieval, image annotation and tracking.