A novel multilayer decision based iterative filter for removal of salt and pepper noise

Nikhil Sharma[1] · Prateek Jeet Singh Sohi[1] · Bharat Garg[1] (ID) · K V Arya[2]

Abstract

In this paper a novel decision based iterative filter for the detection and elimination of salt and pepper noise is proposed. Effective decisions based upon the noise density of the image are used to filter out noise while maintaining finer details in an image. A fixed size window is used at each step to maintain maximum correlation throughout the filtering process. Additional pre-edge and post-smoothing processing are also presented to further enhance the quality of image. Rigorous analysis over Kodak benchmark dataset containing 24 natural images indicates an exceptional performance boost for medium to extremely high noise density when compared with state of the art filtering techniques. The proposed filter is tested quantitatively and qualitatively using benchmark parameters including peak signal to noise ratio, image enhancement factor and visual representation. Even at noise density as high as 90% and 95%, the proposed filter outperforms the exiting filters providing better edge detail, less blurring and low streaking effects.

Keywords Salt and Pepper noise · Mean filter · Median filter · Combinational filtering · Image denoising · Decision based filtering

1 Introduction

Fixed value impulse noise also called as salt and pepper noise [7] is one of the leading cause of image corruption during transmission and accusation. Due to the increase in demand for telemedicine communication the need to transmitting medical images such as X-Ray, CT Scan, MRI and Ultrasound is an ever-increasing endeavour. This means that more and more images are being transmitted through a noisy medium and need de-noising. Salt and Pepper Noise is the most common type of image corruption that occurs during such transmission processes. It is observed that the pixels assume the highest or lowest possible digitized values *i.e.* 0 or 255 in case of grayscale images. Many filters are proposed in the recent past to restore the original image from the corrupted one. The most common are the

✉ Bharat Garg
  bharat.garg@thapar.edu

[1] Thapar Institute of Engineering and Technology, Patiala, India

[2] ABV Indian Institute of Information Technology and Management Gwalior, Gwalior, India

mean and median filters [3, 8]. The basic concept is to construct a 3×3 window around the corrupted pixel and replace the value by corresponding mean and median of the window. These approaches provide accurate results at lower noise density but fails catastrophically at higher noise densities due to lack of information in 3×3 window. Other techniques include using interpolation [13] instead of mean or median. Though it has better results even at higher noise density but it comes at the cost of large execution time and high computational complexity. This has led to the development of hybrid/fuzzy logic filters [11, 14] that are basically the combination of mean and median filters. In addition to being superior restoration techniques they also have a lower computational complexity.

More recent filters such a as a four stage median filtering algorithm (FSMA) is presented in [6]. The FSMA algorithm performs median filtering with smaller window size, large window size, running average and replacement by previously processed pixels at first, second, third and fourth stages, respectively. The algorithm moves to next stages only when the current stage fails to denoise the candidate noisy pixel. The FSMA algorithm can effectively denoise the image by considering the first two stages at low noise density while it may requires all stages at higher noise density. This filter eliminates SAP noise along with better edge preservation as it considers noise-free pixels while estimating the value of noisy pixels. An adaptive trimmed median (ATM) filter is presented in [4] can effectively denoise image corrupted with high noise density. This filter restores noisy pixel by computing median of noise-free pixels of adaptive size window and by computing via interpolation based procedure at low and high noise densities, respectively. Further, this algorithm denoise candidate pixel using nearest processed pixel procedure for the rare scenarios especially at the boundary where denoising using interpolation is not good enough. An adaptive min-max value based weighted median (AMMWM) filter is presented in [5]. This filter first computes two highly correlated groups of noise-free pixels using minimum and maximum value of the current window and then determines weighted medians of these groups to restore the noisy pixel. The filter increases the window size if the current window fails to provide any noise-free pixels. However the filter can restores noisy pixel at high noise density, it exhibits high computational complexity. A noise density range sensitivity mean median filter for impulse noise removal (NRSMF) is presented in [12]. The major contribution of this filtering technique is that the noise removal steps are based on the noise density of the input image. Special care is taken for boundary and corner pixels. These are few concepts that are critical to the development of the current work.

Various filters with superior mathematical techniques [1] are presented to filter out noise while keeping details of an image. Three value weighted approach (TVWA) [9] and Adaptive switching weighted mean filter (ASWMF) [10] are example of such case where weighted averages are used for reconstruction. Variable window sizes are used if no information pixel is available in the current window. Because of this, at higher noise density the window size becomes very large and this results in blurring effect. A rather new approach including Difference applied median filter (DAMF) [2] uses a iterative method to make use of the pre-processed values produced after each iteration. The results of these are significantly good but the use of adaptive size window and same number of iterations is causing blurring and streaking effect throughout the image at higher noise densities. But no matter what method is used the blurring effects produced by these techniques are quite evident. This means that the main task at hand is to develop technique that is better equipped to preserve edge details with reduced blurring. This paper addresses the aforementioned issues by presenting a novel multilayer decision based iterative filter.

The remaining paper is organized as follows. Section 2 explain proposed algorithm with example whereas simulation results are analysed in Section 3. Finally, Section 4 concludes the paper.

2 Proposed algorithm

In this section, a novel iterative filtering method is presented, which can be broadly segmented into three subsections *i.e.* pre-edge filtering, iterative filtering (main algorithm) and post-smoothing along with time complexity (Big O notation) of each step.

2.1 Noise density

It is necessary to calculate the noise density of the image before starting the filtering process as it is an integral part of our decision making process. Noise density (η) can be calculated using (1) and (2).

$$\gamma(x, y) = \begin{cases} 1, & P_{x,y} \in \{0, 255\} \\ 0, & Otherwise \end{cases} \tag{1}$$

$$\eta = \frac{\sum_{x=1}^{x=M} \sum_{y=1}^{y=N} \gamma(x, y)}{M \times N} \times 100\% \tag{2}$$

Where, $P_{x,y}$ are the pixels in noisy image. M and N are the dimensions of the image. The worst case time complexity of these equations will be $O(M \times N)$.

2.2 Pre-Edge filtering

Initially, the underlying pixel $P_{x,y}$ of the noisy-image is classified as corrupted or original. Original pixels are left unprocessed and for others, a 3×3 window ($W_{3\times3}$) is constructed across it. Then we check similar pixels in the horizontal and diagonal directions in $W_{3\times3}$. If such a pixel pair is found, replace the $P_{x,y}$ with the pixel value of the matching pair as shown in (3).

$$P_{x,y} = \begin{cases} P_{x+1,y+1}, & \text{if } P_{x+1,y+1} = P_{x-1,y-1} \\ P_{x+1,y}, & \text{if } P_{x+1,y-1} = P_{x-1,y+1} \\ P_{x,y+1}, & \text{if } P_{x,y+1} = P_{x,y-1} \\ P_{x,y}, & Otherwise \end{cases} \tag{3}$$

It is observed that most of the pixels restored in this process are replaced by original values. It is because, the pixels following above criteria might be a part of edge in an image. This step only requires the full traversal of the image, so the time complexity is $O(M \times N)$.

2.3 Combinational filtering

Two copies of the same image namely α and β are created using the output obtained from the pre-edge filtering process. In addition to this, a discrete variable (κ) is initialized based on noise density (η) of noisy image and is given by (4).

$$\kappa = \begin{cases} 1, & \eta < \eta_1 \\ 2, & \eta_1 < \eta < \eta_2 \\ 3, & \eta_2 < \eta < \eta_3 \\ 4, & \eta > \eta_3 \end{cases} \tag{4}$$

These two images are passed through a multilayered combinational filtering algorithm along with κ as shown in Algorithm 1. Here, κ represents the number of layers or loops used in the filtering process. Both images are parallely processed with different combinations of

mean and median. On each iteration, images are updated with output of previous iteration. The threshold values (η_1, η_2 and η_3) are experimentally determined based of the theoretical concept that in order to save time, we must prevent overprocessing of the pixels. The values of the η_1, η_2 and η_3 are 50%, 75% and 95%, respectively. We have extensively experimented with the threshold values and we can say with utmost certainty that changing the threshold will cause deterioration in the resultant image.

For image α, during the first iteration, the corrupted pixels are replaced by the median of noiseless 3×3 window represented by $W_{3\times3}^{nf}$. In second and third iteration the unprocessed corrupted pixels present in the updated image are replaced by the mean of $W_{3\times3}^{nf}$. Finally in the fourth iteration corrupted pixels are replaced by the median of $W_{3\times3}^{nf}$. These iterations will be limited to the value of κ. The window size is maintained at 3×3 to retain the maximum co-relation between the information pixels and corrupted pixel.

Algorithm 1 Multilayer combinational filtering.

1: **Input** α, β, κ ▷ *Input noisy images and Layer*
2: **Output** α, β ▷ *Processed Images*
3: **for** δ in range(κ) **do**
4:      **for** each noisy $P_{x,y}$ in α_δ **do**
5:          $W_c \leftarrow W_{3\times3}^{nf}$ ▷ 3×3 *noise free window*
6:          **if** $length(W_c) > 0$ **then**
7:             **if** $\delta = 1$ *or* $\delta = 4$ **then**
8:                $\alpha_{\delta+1}(x, y) \leftarrow median(W_c)$
9:                $\beta_{\delta+1}(x, y) \leftarrow mean(W_c)$
10:             **else if** $\delta = 2$ *or* $\delta = 3$ **then**
11:                $\alpha_{\delta+1}(x, y) \leftarrow mean(W_c)$
12:                $\beta_{\delta+1}(x, y) \leftarrow median(W_c)$
13:             **end if**
14:          **end if**
15:      **end for**
16: **end for**
17: $\alpha \leftarrow \alpha_{\delta+1}$, $\beta \leftarrow \beta_{\delta+1}$
18: **Return** α , β

Operations on image β are similar to operations performed on α, except the combination of mean-median used. In this case, Mean - Median - Median - Mean will be used.

Now, we have processed images α and β, we can obtain the output filtered image by taking the average of two and mapping it to a 8 bit digital image as given by (5).

$$f(x, y) = uint8(\frac{\alpha + \beta}{2})$$ (5)

It is worth mentioning here that the worst case time complexity of this iterative filtering is $O(\kappa \times M \times N)$. κ is a constant with maximum value of 4. So, the time complexity will be $O(M \times N)$.
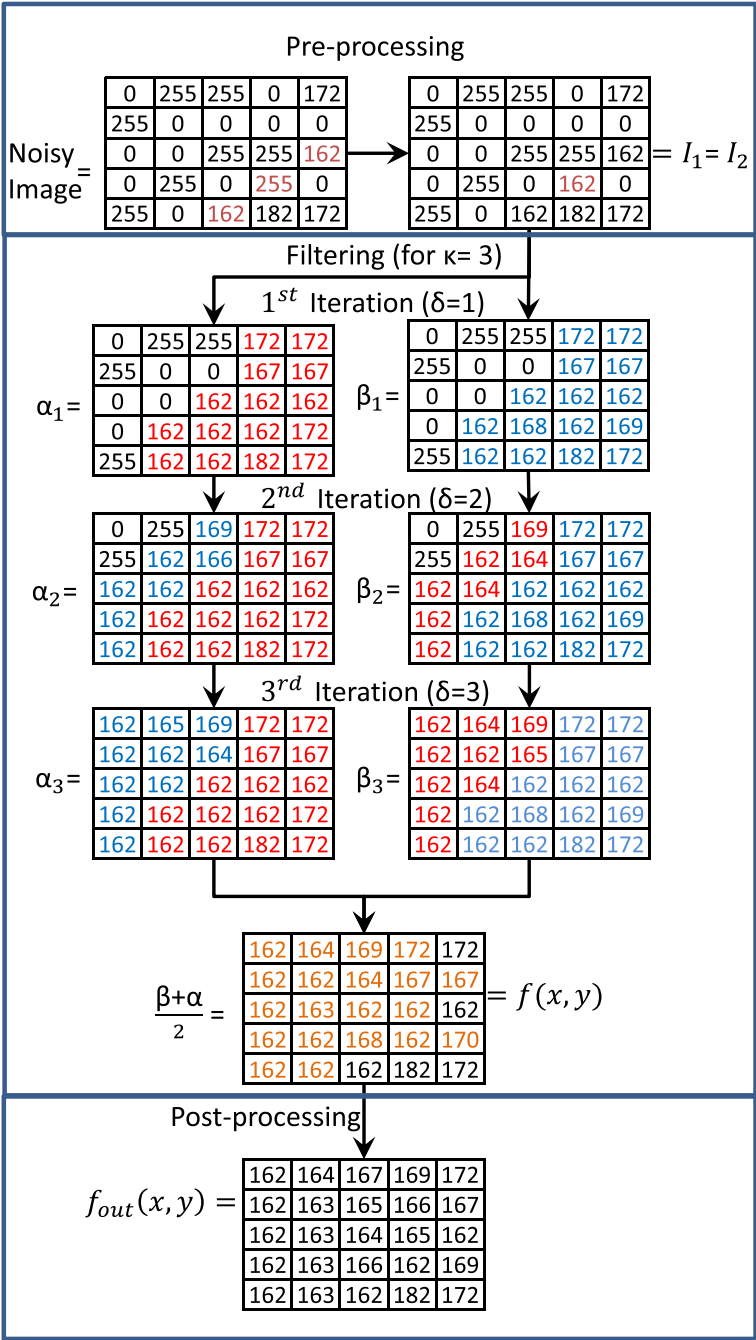
Fig. 1 Example of the proposed filter with $\eta = 80\%$

2.4 Post-smoothing filtering

To further enhance the quality of image, a smoothing step is added to the algorithm. The basic concept is to take the average of $W_{3\times3}$ constructed across the pixel $P_{x,y}$ in $f(x,y)$

Table 1 PSNR and IEF comparison of proposed filter over existing filters

	Image	ASWM	FSBMM	DAMF	TVWA	UWMF	PA
IEF	kodim01	29.09	27.84	24.66	24.32	39.51	**40.10**
	kodim02	131.55	131.29	127.60	124.18	173.3	**182.75**
	kodim03	133.36	127.92	123.33	125.07	171.87	**189.8**
	kodim04	118.07	110.48	106.38	101.18	147.43	**165.12**
	kodim05	32.24	27.74	26.03	25.09	40.89	**44.87**
	kodim06	45.26	41.28	39.44	38.62	61.96	**64.2**
	kodim07	89.63	73.16	64.6	62.76	114.96	**131.51**
	kodim08	15.77	15.51	13.98	13.72	21.63	**22.43**
	kodim09	83.43	75.99	63.7	63.31	108.6	**118.14**
	kodim10	78.2	76.82	66.62	66.7	80.51	**110.04**
	kodim11	58.92	55.33	51.06	49.49	77.59	**82.83**
	kodim12	101.64	98.40	88.03	85.83	132.69	**147.47**
	kodim13	20.84	19.85	18.64	18.07	25.5	**28.27**
	kodim14	56.33	48.98	45.8	44.5	73.31	**78.97**
	kodim15	62.81	50.6	61.63	59.48	29.43	**79.43**
	kodim16	91.17	86.39	84.15	85.63	127.21	**129.93**
	kodim17	118.28	103.71	93.74	90.64	161.05	**182.69**
	kodim18	47.96	48.37	42.94	41.55	62.53	**69.47**
	kodim19	45.91	43.14	37.08	36.45	68.56	**67.85**
	kodim20	14.92	20.78	12.91	39.76	2.45	**10.84**
	kodim21	49.05	41.76	39.28	37.91	63.86	**68.43**
	kodim22	75.29	74.12	65.3	63.54	97.68	**104.65**
	kodim23	119.59	126.37	108.92	114.65	85.29	**168.52**
	kodim24	19.4	29.22	18.21	10.66	5.93	**19.11**
	Average	68.28	64.79	59.33	59.3	82.24	**96.14**
PSNR	kodim01	21.40	21.36	21.86	22.33	22.72	**22.98**
	kodim02	27.41	27.92	27.86	28.8	28.96	**29.35**
	kodim03	27.55	27.77	28.06	28.99	28.94	**29.46**
	kodim04	26.99	27.12	27.52	28.41	28.23	**28.78**
	kodim05	21.09	20.67	21.78	22.18	22.21	**22.68**
	kodim06	22.73	22.59	23.29	23.87	24.18	**24.52**
	kodim07	26.05	25.30	26.51	27.22	27.17	**27.74**
	kodim08	18.30	18.28	18.81	19.21	19.61	**19.87**
	kodim09	25.87	25.75	26.3	27.03	27.07	**27.49**
	kodim10	25.62	25.74	25.98	26.86	25.89	**27.15**
	kodim11	24.06	24.04	24.62	25.21	25.41	**25.76**
	kodim12	26.22	26.27	26.84	27.53	27.5	**28.02**
	kodim13	19.70	19.56	20.18	20.50	20.57	**21.01**

Table 1 (continued)

Image	ASWM	FSBMM	DAMF	TVWA	UWMF	PA
kodim14	23.70	23.34	24.29	24.86	24.97	**25.41**
kodim15	23.28	22.89	23.08	24.69	20.49	**24.35**
kodim16	26.04	26.10	26.50	27.32	27.68	**27.99**
kodim17	26.19	26.17	27.15	27.95	27.96	**28.5**
kodim18	22.79	23.11	23.56	24.09	24.14	**24.55**
kodim19	23.10	23.06	23.66	24.30	24.78	**24.94**
kodim20	16.84	18.48	13.89	21.61	9.18	**17.35**
kodim21	23.52	22.95	23.97	24.50	24.69	**25.10**
kodim22	25.37	25.53	25.81	26.47	26.53	**26.94**
kodim23	27.02	27.67	27.35	28.93	25.99	**29.01**
kodim24	19.41	21.09	18.03	17.17	14.32	**19.77**
Average	23.76	23.86	24.04	25.00	24.13	**25.36**

The simulation results of our proposed work are highlighted in bold

Table 2 SSIM metric comparison of proposed filter over existing filters

	Image	ASWM	FSBMM	DAMF	TVWA	UWMF	PA
SSIM	kodim01	0.7405	0.7074	0.6686	0.6378	0.7833	**0.7714**
	kodim02	0.7992	0.8466	0.8344	0.8137	0.8706	**0.871**
	kodim03	0.8547	0.8911	0.8817	0.8808	0.9124	**0.918**
	kodim04	0.8368	0.854	0.853	0.8252	0.886	**0.8912**
	kodim05	0.7883	0.7539	0.7486	0.7124	0.8256	**0.8244**
	kodim06	0.7525	0.7463	0.7314	0.6953	0.8128	**0.8068**
	kodim07	0.8814	0.8533	0.8597	0.8472	0.9131	**0.9194**
	kodim08	0.7457	0.7183	0.6926	0.6587	0.7955	**0.7864**
	kodim09	0.8625	0.8758	0.8607	0.8578	0.9056	**0.914**
	kodim10	0.8526	0.854	0.8448	0.8308	0.8936	**0.9014**
	kodim11	0.7964	0.7935	0.7879	0.7542	0.8482	**0.8464**
	kodim12	0.8216	0.8574	0.8562	0.8343	0.8858	**0.892**
	kodim13	0.7087	0.6739	0.6506	0.6014	0.7478	**0.746**
	kodim14	0.7981	0.7726	0.7783	0.742	0.8424	**0.8425**
	kodim15	0.8055	0.8667	0.8622	0.8459	0.887	**0.8958**
	kodim16	0.8006	0.8031	0.7907	0.7727	0.855	**0.8545**
	kodim17	0.8385	0.8525	0.8574	0.8422	0.905	**0.9055**
	kodim18	0.7787	0.7839	0.764	0.7287	0.8363	**0.8343**
	kodim19	0.8074	0.804	0.7921	0.764	0.8525	**0.8515**
	kodim20	0.7118	0.8207	0.7584	0.8292	0.7641	**0.8427**
	kodim21	0.8389	0.8267	0.8152	0.7987	0.875	**0.8814**
	kodim22	0.8049	0.8084	0.7891	0.7577	0.8507	**0.8514**
	kodim23	0.8757	0.9176	0.9089	0.9099	0.9331	**0.9416**
	kodim24	0.7672	0.7748	0.74	0.7117	0.8038	**0.8175**
	Average	0.8028	0.8107	0.7969	0.7772	0.8535	**0.8586**

The simulation results of our proposed work are highlighted in bold
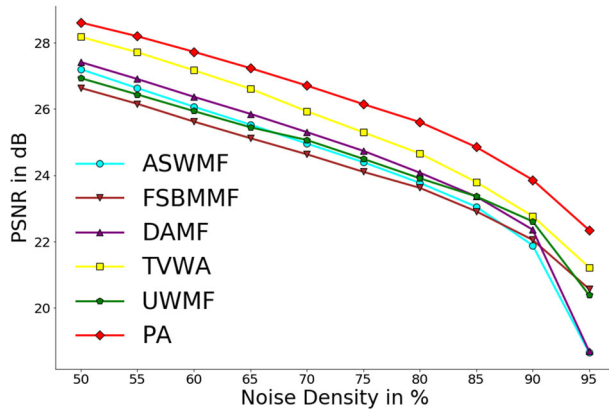
Fig. 2 Plot of average PSNR values for Kodak Benchmark Dataset with varying noise density from 50% to 95%

where noise was first discovered in the noisy image. The window is represented by $W_{P_{x,y}}^{3\times3}$ and mathematically the output filtered image is given by (6).

$$f_{out}(x, y) = \begin{cases} Mean(W_{P_{x,y}}^{3\times3}), & if \ \gamma(x, y) == 1 \\ P_{x,y}, & Otherwise \end{cases} \tag{6}$$

Again, this step also requires full traversal of the image so time complexity will be $O(M \times N)$. So, we can observe that the only computational expensive task is full image traversal. The overall worst case time complexity of complete algorithm is $O(M \times N)$. So, the restored image can be computed in polynomial time and can be used in real life scenarios.

2.5 Example of the proposed filter

To demonstrate the working of proposed filter, an example on 5×5 noisy image ($nImg$) with 80% corruption is given in Fig. 1. Initially $nImg$ is passed through a pre-edge filtering step as described by (3). The pixels highlighted in the $nImg$ block satisfies this condition and the corrupted pixel is replaced by the given value (*i.e.* 162). In the filtering step, two copies of this image are stored into α and β. Since $\eta = 80\%$, as per (4) κ will be 3, so both images are parallelly processed by three iterations. For image α Median-Mean-Mean sequence is performed in iteration 1, 2 and 3 respectively. Similarly, β is processed by sequence Mean-Median-Median. The pixels processed in each iteration are colour coded. Then image $f(x, y)$ is obtained by averaging α and β. For post-smoothing filtering, all the pixels of $f(x, y)$ that are highlighted in orange are the replaced by average of $W_{P_{x,y}}^{3\times3}$ as described in (6) to obtain the output filtered image $f_{out}(x, y)$.

3 Simulation results

To evaluate the efficacy of the proposed work, the proposed filter along with the well-known filters are implemented in MATLAB and then simulated with Kodak benchmark image dataset containing 24 images of size 512×768 or 768×512. The evaluation task is to compute a de-noised image with noise density from medium to high range (50% to 95%). Both qualitative and quantitative parameters are used to judge the quality of the filter.
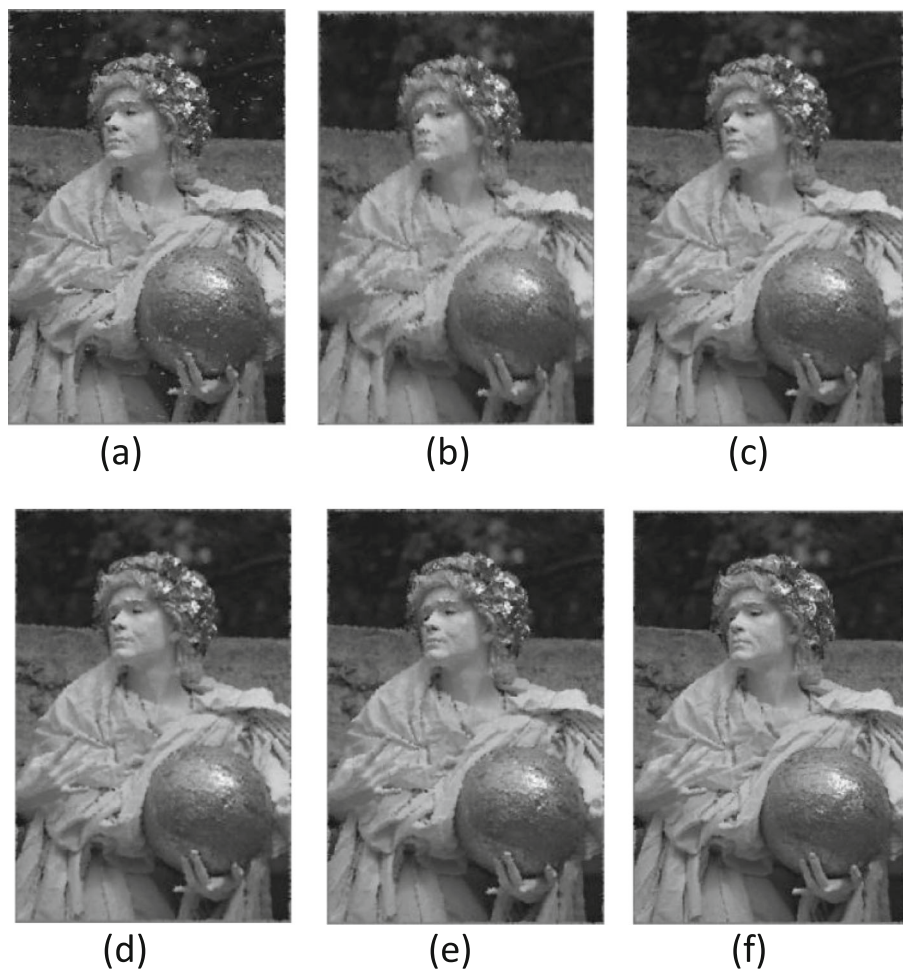
Fig. 3 Man with Globe image with 85% noise density filtered **a** ASWMF, **b** FSBMM, **c** DAMF, **d** TVWA, **e** UMMF, and **f** Proposed filter

3.1 Quality metrics

The peak signal to noise ratio (PSNR), mean square error (MSE) and image enhancement factor (IEF) is considered for analysis quantitatively. The PSNR is defined as the ratio between the maximum possible power of the signal and the power of distorting noise. Mathematically it is given by (7),

$$PSNR = 10log_{10}(MAX^2/MSE) \tag{7}$$

where, MAX is 255 for grayscale image and MSE is mean square error given by (2).

$$MSE = \frac{1}{M*N}\sum_{i=1}^{M}\sum_{j=1}^{N}(x_{i,j} - y_{i,j})^2 \tag{8}$$

Table 3 PSNR, SSIM and IEF results on Landscape Image, presented as 8th image in Kodak dataset

	ND	ASWM	FSBMM	DAMF	RSIf	TVWA	UWMF	NRSM	PA
IEF	50	19.47	18.45	12.6	18.86	11.37	28.37	26.23	**29.32**
	60	18.94	17.75	14.37	17.66	13.4	26.82	24.66	**28.02**
	70	18.05	17.31	15.51	15.89	14.83	24.75	24.1	**25.45**
	80	16.16	15.75	15.85	13.78	15.62	21.19	22.65	**23.23**
	90	12.88	13.2	14.36	10.73	14.58	16.81	16.78	**17.78**
	95	9.21	10.66	11.21	8.65	12.5	11.87	12.97	**13.04**
	Avg	15.78	15.52	13.98	14.26	13.72	21.63	21.23	**22.81**
[6pt] PSNR	50	20.96	20.72	19.07	20.8	18.62	22.59	21.18	**22.72**
	60	20.04	19.76	18.84	19.76	18.54	21.55	20.2	**21.62**
	70	19.17	18.98	18.51	18.61	18.31	20.54	19.98	**20.55**
	80	18.11	18	18.03	17.43	17.96	19.29	19.04	**19.41**
	90	16.62	16.72	17.09	15.81	17.15	17.77	17.24	**18.01**
	95	14.93	15.56	15.78	14.65	16.25	16.03	15.8	**16.44**
	Avg	18.3	18.29	17.88	17.84	17.8	19.62	18.91	**19.79**
SSIM	50	0.9587	0.9473	0.9394	0.8943	0.8786	0.9292	0.8883	**0.9396**
	60	0.9438	0.9308	0.9283	0.8554	0.874	0.9254	0.8855	**0.9304**
	70	0.924	0.9068	0.9134	0.7949	0.8661	0.9291	0.8505	**0.9245**
	80	0.8937	0.877	0.8874	0.7044	0.8535	0.9002	0.7695	**0.9047**
	90	0.7586	0.8122	0.8353	0.5208	0.8226	0.8498	0.467	**0.8617**
	95	0.3582	0.7329	0.6734	0.3619	0.7446	0.7455	0.2107	**0.749**
	Avg	0.8062	0.8678	0.8629	0.6886	0.8399	0.8799	0.6803	**0.885**

The simulation results of our proposed work are highlighted in bold

where, M and N are dimensions of the image, x is the original image and y is the restored image. The second parameter used is the image enhancement factor (IEF). The mathematical expression of IEF is given by

$$IEF = \frac{\sum_{i=0}^{M-1} \sum_{j=0}^{N-1} (x(i, j) - z(i, j))^2}{\sum_{i=0}^{M-1} \sum_{j=0}^{N-1} (x(i, j) - y(i, j))^2} \tag{9}$$

where, x, y and z are the original, restored and noisy images, respectively.

Another parameter used is Structural Similarity Index (SSIM). Mathematically,

$$SSIM(x, y) = \frac{(2\mu_x \mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{10}$$

where μ_x (σ_x) and μ_y (σ_y) represent mean (standard deviation) value in x and y directions, respectively. Further, C_1 and C_2 represent constants which are considered to limit the SSIM value to 1.

3.2 Experimental results on Kodak benchmark image dataset

Tables 1 and 2 summarizes the average values of IEF, PSNR and SSIM for six filters on 24 grayscale images from the Kodak benchmark image dataset. The values clearly demonstrate that the proposed algorithm (PA) has an edge over other existing filters. There are some

Table 4 PSNR, SSIM and IEF results on Face Image, presented as 15th image in Kodak dataset

	ND	ASWM	FSBMM	DAMF	RSIf	TVWA	UWMF	NRSM	PA
IEF	50	90.23	65.49	64.41	74.15	58.93	24.46	66.52	**105.36**
	60	81.63	61.22	67.84	72.3	64.41	26.17	66.79	**114.68**
	70	77.68	54.11	73.94	59.24	70.31	32.87	69.44	**122.86**
	80	67.31	50.91	69.65	39.46	66.19	33.06	68.31	**117.92**
	90	45.17	37.73	59.9	30.69	53.62	37.49	55.36	**94.49**
	95	16.89	30.22	38.05	20.39	55.05	25.6	30.83	**53.15**
	Avg	63.15	49.95	62.3	49.37	61.42	29.94	59.54	**101.41**
PSNR	50	27.01	25.62	25.55	26.17	25.16	21.34	26.56	**27.69**
	60	25.79	24.54	24.99	25.25	24.76	20.85	25.93	**27.27**
	70	24.89	23.32	24.68	23.7	24.46	21.15	24.52	**26.88**
	80	23.67	22.46	23.82	21.39	23.6	20.59	23.44	**26.11**
	90	21.45	20.66	22.67	19.77	22.19	20.64	21.06	**24.65**
	95	16.95	19.47	20.47	17.77	21.08	18.75	19.53	**21.93**
	Avg	23.29	22.68	23.7	22.34	23.54	20.55	23.51	**25.75**
SSIM	50	0.9587	0.9473	0.9394	0.9486	0.8786	0.9292	0.9269	**0.9546**
	60	0.9438	0.9308	0.9283	0.9306	0.874	0.9254	0.9242	**0.9354**
	70	0.924	0.9068	0.9134	0.9002	0.8661	0.9291	0.9114	**0.9215**
	80	0.8937	0.877	0.8874	0.8548	0.8535	0.9002	0.8881	**0.9047**
	90	0.7586	0.8122	0.8353	0.7732	0.8226	0.8498	0.8547	**0.8617**
	95	0.3582	0.7329	0.6734	0.6884	0.7446	0.7455	0.7211	**0.741**
	Avg	0.8062	0.8678	0.8629	0.8493	0.8399	0.8799	0.8711	**0.8823**

The simulation results of our proposed work are highlighted in bold

images in the dataset where PA is unable to provide the best results. These are mostly images having very rough transitions or completely smooth areas in certain parts of the image. Nonetheless, the proposed filter has an average improvement of 13.9, 0.36 dB and .0051 in IEF, PSNR and SSIM, respectively when compared to best existing filters.

Adding to quantitative performance, a plot for PSNR values for same dataset is given in Fig. 2. It is observed that over 70% noise density the performance of the PA is exceptionally well. Even at noise density as high as 95%, where most of the filters fail to perform, the PA outshines and provides considerably good average value of PSNR. At medium noise densities, the performance of adaptive approaches come quite close the PA but still PA manages to outperform existent techniques.

Figure 3 represents the restored images from different filters when "17kodim.png" image is considered for analysis at 85% noise density. From the visual representation, we can ascertain that ASWMF and FSBMMF are unable to provide a clear restored image. The other four filters have comparative results but if we look closer at the face, especially the nose area, we can observe that only PA is able to reproduce finer edge details with less blurring.

3.3 Experimental results on multiple image categories

In order to include diversity among the result analysis, the PSNR, SSIM and IEF values are evaluated with noise density varying from 50% to 95% as tabulated in Tables 3 and 4 using images Fig. 4a, b of size 768×512, respectively. Similarly, the quality metrics using Fig. 4c
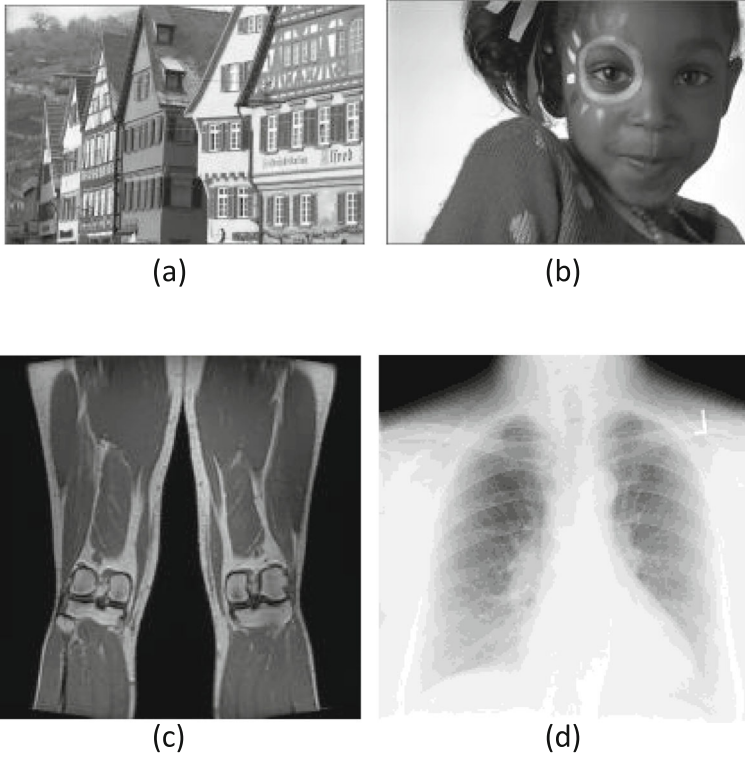
Fig. 4 Benchmark **a** Kodak Landscape **b** Kodak Face **c** Medical Thigh and **d** Medical Lungs images

Table 5 PSNR, SSIM and IEF results on thigh.jpg for very high noise density corruption

	ND	ASWM	FSBMM	DAMF	TVWA	UWMF	PA
IEF	90	18.32	61.13	27.07	61.82	75.87	**88.09**
	91	16.62	52.45	26.01	63.62	68.56	**82.15**
	92	13.86	48.25	24.4	59.83	68.65	**76.13**
	93	12.51	49.04	20.91	57.57	65.43	**74.43**
	94	10.77	44.59	18.39	52.15	58.74	**65.21**
	95	7.95	38.37	13.08	51.15	51.9	**60.68**
	96	6.78	33.58	10.8	43.85	38.62	**46.75**
	97	5.47	29.6	7.38	43.27	29.09	**43.93**
	98	4.3	22.82	4.85	35.99	14.72	**34.86**
	99	3.52	15.84	3.24	26	7.64	**26.48**
	Avg	10.01	39.57	15.61	49.53	47.92	**59.87**
PSNR	90	17.32	22.55	19.01	22.6	23.49	**24.14**
	91	16.85	21.84	18.8	22.68	23.01	**23.79**
	92	16.03	21.45	18.49	22.38	22.98	**23.43**
	93	15.56	21.5	17.79	22.19	22.75	**23.31**
	94	14.84	21.01	17.16	21.69	22.21	**22.66**

Table 5 (continued)

	ND	ASWM	FSBMM	DAMF	TVWA	UWMF	PA
	95	13.46	20.3	15.63	21.55	21.61	**22.29**
	96	12.73	19.68	14.76	20.84	20.29	**21.12**
	97	11.78	19.12	13.09	20.77	19.04	**20.89**
	98	10.65	17.9	11.18	19.88	16	**19.88**
	99	9.75	16.28	9.38	18.43	13.11	**18.8**
	Avg	13.9	20.16	15.53	21.3	20.45	**22.03**
SSIM	90	0.5553	0.6626	0.6706	0.7066	0.7566	**0.7628**
	91	0.5224	0.6349	0.6589	0.7026	0.7421	**0.7495**
	92	0.4626	0.5976	0.6206	0.6878	0.7275	**0.7311**
	93	0.4292	0.6083	0.61	0.6811	0.7217	**0.7261**
	94	0.3613	0.5565	0.5607	0.6659	0.7022	**0.7058**
	95	0.2849	0.5274	0.4841	0.6454	0.6683	**0.6776**
	96	0.2157	0.4965	0.4245	0.6306	0.6183	**0.648**
	97	0.1672	0.4508	0.3363	0.6088	0.554	**0.6042**
	98	0.0994	0.3948	0.2099	0.5598	0.4269	**0.545**
	99	0.0673	0.3459	0.0967	0.4836	0.3026	**0.4641**
	Avg	0.3165	0.5275	0.4672	0.6372	0.622	**0.6614**

The simulation results of our proposed work are highlighted in bold

of size 256×256 with noise density varying from 90% to 99% are summarized in Table 5. From the quality metrics, we can infer that proposed filter performs exceedingly well on almost all image categories, such as artificial structures, humans and medical images.
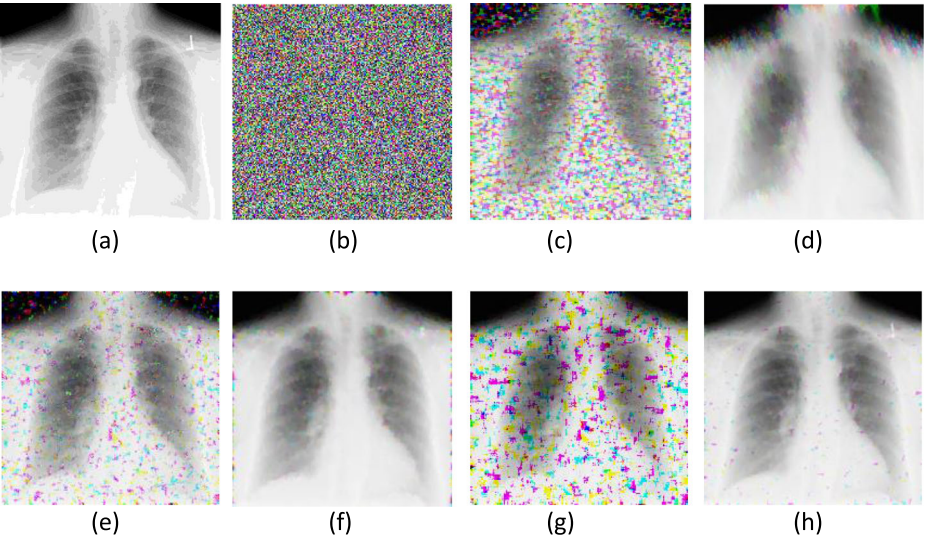

Fig. 5 Lungs images **a** original **b** noisy image with 97% noise density and **c-h** denoised images

Table 6 Average computational cost (computation time in second) comparison of proposed filter over existing filters on various benchmark images with noise varying from 10% to 90%

	ND	ASWM	RSIf	FSBMM	DAMF	TVWA	UWMF	PA
Time (sec)	10	2.2536	4.639	0.1414	0.5477	0.6867	0.1474	0.3215
	30	4.3379	10.4739	0.2453	0.6048	0.8083	0.2872	0.5543
	50	6.2333	16.8853	0.368	0.7721	0.8224	0.5728	0.7675
	70	8.1644	23.4423	0.4987	0.9889	1.0315	0.8723	1.2789
	90	8.6382	29.2621	0.6461	1.5471	1.4856	1.2494	2.1481

For testing the visual quality at higher noise density we have chosen Lungs.png with 97% SAP noise. From Fig. 5 we can observe that at very high noise density the proposed filter is able to reproduce the original image structure. The proposed filter is able to filter out finer/sharper details in the image with less blurring and no streaking effect like in case of FSBMMF.

3.4 Computational cost comparison

Table 6 depicts the average computational cost comparison of proposed filter over existing filters on various benchmark images with noise varying from 10% to 90%. The cost of the proposed filter is comparable to the other existing filter but has slightly higher value at higher noise density range. So, high performance is achieved on the expense of high computational cost. The trade-off between the time complexity and image restoration compatibility can be justified by the overall qualitative improvement achieved. There is an improvement of 3.42%, 3.79% and 20% in the PSNR, SSIM and IEF repectively for higher noise densities from 90% to 99% at the expense of computational time.

4 Conclusion

In this paper a novel decision based iterative filter is proposed for the detection and elimination of impulse noise. The filter efficiently uses fixed size window and the combination of mean and median to provide better details in an image. The number of iterations is controlled based on noise density of the image. Moreover, an additional pre-edge and post-smoothing step is also proposed to further enhance the quality of reconstructed image. Simulation results verify the filters efficacy in terms of PSNR, IEF and visual representation. The simulation results show that the proposed filter is able to restore the edge details with less blurring and low streaking effect on 24 natural images in Kodak benchmark image dataset when compared to existing filters. Time complexity analysis shows that the filter could be used in real life scenarios.

References

1. Enginoğlu S, Erkan U, Memiş S (2019) Pixel similarity-based adaptive riesz mean filter for salt-and-pepper noise removal. Multimed Tools Appl 78(24):35401–35418
2. Erkan U, Gökrem L, Enginoğlu S (2018) Different applied median filter in salt and pepper noise. Comput Electric Eng 70:789–798

3. Esakkirajan S, Veerakumar T, Subramanyam AN, PremChand C (2011) Removal of high density salt and pepper noise through modified decision based unsymmetric trimmed median filter. IEEE Signal Process Lett 18(5):287–290
4. Garg B (2020) Restoration of highly salt-and-pepper-noise-corrupted images using novel adaptive trimmed median filter. Signal Image Vid Process
5. Garg B (2020) An adaptive minimum-maximum value-based weighted median filter for removing high density salt and pepper noise in medical images. Int J Ad Hoc Ubiquit Comput 35(2):84–95
6. Garg B, Arya K (2020) Four stage median-average filter for healing high density salt and pepper noise corrupted images. Multimed Tools Appl 79(43):32305–32329
7. Gonzalez RC, Wood RE (2002) Digital image processing, 2nd edn. Prentice Hall, Hoboken
8. Hwang H, Haddad RA (1995) Adaptive median filters new algorithms and results. IEEE Trans Image Process 4(4):499–502
9. Lu C-T, Chen Y-Y, Wang L-L, Chang C-F (2016) Removal of salt-and-pepper noise in corrupted image using three-values-weighted approach with variable-size window. Pattern Recogn Lett 80:188–199
10. Meher SK, Singhawat B (2014) An improved recursive and adaptive median filter for high density impulse noise. AEU-Int J Electron Commun 68(12):1173–1179
11. Selvi AS, Kumar KPM, Dhanasekeran S, Maheswari PU, Ramesh S, Pandi SS (2020) De-noising of images from salt and pepper noise using hybrid filter, fuzzy logic noise detector and genetic optimization algorithm (hfgoa). Multimed Tools Appl 79(5):4115–4131
12. Sohi PJS, Sharma N, Garg B, Arya K (2020) Noise density range sensitive mean-median filter for impulse noise removal. In: Innovations in computational intelligence and computer vision. Springer, pp 150–162
13. Veerakumar T, Esakkirajan S, Vennila I (2014) Recursive cubic spline interpolation filter approach for the removal of high density salt-and-pepper noise. Signal Image Vid Process 8(1):159–168
14. Vijaykumar V, Mari GS, Ebenezer D (2014) Fast switching based median–mean filter for high density salt and pepper noise removal. AEU-Int J Electron Commun 68(12):1145–1155