ISSN 2347-4289

Fast Decision Based Weighted Fuzzy Mean Filtering For Extremely Corrupted Images

Easwara M, Satish Babu J

SJCIT, Chickballapur, Visveswaraya Technological University, Karnataka, INDIA
E-mail: easwaram138@gmail.com, j_satishbabu@yahoo.com

Abstract: This paper presents an uncertainty based novel filter that integrates randomness and fuzziness, for extremely corrupted images by salt and pepper impulse noise. In real time work, high computational efficiency is the most important factor while suppressing noise with better edge and detail preservation. The proposed algorithm is bounded with all these aspects. In this scheme, the corrupted pixel is identified by the strong decision and replaced it by a weighted fuzzy mean estimation. Since, the Certainty Degrees (CD) of each pixels are soft values and are used as the weights, the noisy pixels are reconstructed with an appropriate pixel values. The simulation results show that the performance of the proposed filter is much better than decision based algorithms and cloud model based detection filter across a wide range of impulse noise, as high as 90%, with edge and detail preservation.

Index terms: uncertainty, weighted fuzzy mean, cloud metrics, soft values.

I. INTRODUCTION

Impulse noise is unwanted information that alters the pixel value, producing small dark and bright spots on the image; they take minimum and maximum values in the dynamic range. Digital images are often corrupted by impulse noise during data acquisition, transmission and processing. Usually, impulse noise interfered due to noisy sensors, switching effects in circuits, timing errors in ADC, etc. Impulse noise may seriously affect the performance of image processing techniques; hence, an effective denoising technique is a very important issue. Among non-linear spatial filtering, median filtering is the simplest and has high computational efficiency, but capable only at low noise densities [1]. Reason for this failure is, it concentrates only about the basic statistics, which is bounded by unconditional decision. Later, various modifications have made for median filter to overcome its inefficiency [2]. The center weighted median filter [3], which is a weighted median filter giving more weight only to the central value of each window. This approach identifies the center pixel whether it is corrupted or not and replaced the noisy pixel with median estimation. At higher noise densities, the number of replacements of corrupted pixels increases. Also, the corrupted pixel values and replaced pixel values are less correlated. Therefore, edges are smeared. The progressive switching median filter (PSMF) [4] is a derivative of the basic switching median filter. In this filtering approach, detection and removal of impulse noise are iteratively done in two separate stages. If the reconstructed median value is also a corrupted one, then window size expands and recomputed the median value. The filter provides more improved filtering performance than many other median based filters, but it has a very high computational complexity due to its iterative procedure. In real time work, high computational efficiency is the most important factor, while suppressing noise with better edge and detail preservation. Viewing with this fact, Decision Based Algorithm (DBA) [6] takes less processing time, because of a fixed 3x3 window length used, but, it creates horizontal stripes on the restored image. Because, at high density noise level median value may also be a noisy pixel, in which case it is replaced by last processed (left) neighborhood pixel. To overcome the drawback of the DBA and to recover the possible presence of edge, it is modified by Madhu, *et.al* . In this NDBA [7], at high noise density, corrupted pixels are replaced by mean of already processed uncorrupted neighborhood pixel. Though there is a smooth transition among the pixels encountered, at high noise level edge and detail preservation is not satisfactory. Because mean is a linear filtering process from which noisy pixels lied at the edges cannot be replaced by an appropriate pixel value. This reveals that the above filters only think about randomness in their noise estimation process. Whereas impulse noise, as the knowledge of day flies, is an uncertainty fact that involves the main features both randomness and fuzziness. In addition, DBA and NDBA are failed to recognize the status of the uncorrupted pixels occurred especially, at the origin, when the window is imposed on the corrupted image. The paper introduces an uncertainty based post-filtering that integrates both randomness and fuzziness. Detection of noise pixels is based on the decision and the corrupted pixel is replaced by a weighted fuzzy mean estimation. Also, it overcomes the basic problem encountered in [6] and [7], when window is masked at the origin of the corrupted image. Since, the Certainty Degrees (CD) of each pixels are soft values between [0,1] and are used as the weights, thus, the noisy pixels are reconstructed with an appropriate pixel values. It pays weighted modification to work of Zhe Zhou's article [8] to improve the performance of DBA in salt and pepper noise detection and filtering to achieve high computational efficiency, in turn, that yields the significant results for edge and detail preservation, especially, at high noise density. Outline of the proposed paper is as follows: Section II provides background. Section III describes PA, where noise detection and estimation scheme is presented with illustration. Section IV presents a summary of implementation procedures and test results. The conclusion is drawn in Section V.

II. BACKGROUND

A. Impulse Noise Model

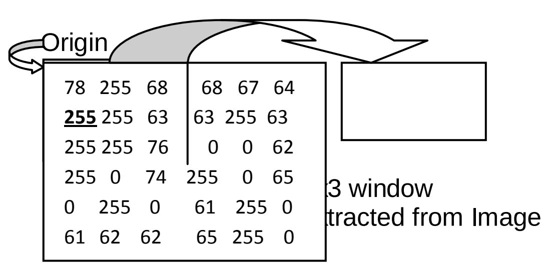
Due to faulty switching devices, pixels are randomly corrupted by the two extreme values. These noise pixels are usually set to the maximum and minimum values in the dynamic range. Of various impulse noise models, salt and pepper impulse noise is implemented to examine the performance of the proposed technique. In salt and pepper impulse noise model noise pixels are assumed to take the minimal and maximal intensities.

***Salt and pepper impulse noise*:**
For an 8-bit gray scale image pixels are randomly corrupted by two extreme values 0 and 255 with unequal probability. Let *x*(i,j) be the gray level of noisy pixel of image X at pixel location (i,j) in the dynamic range. Denote y by a reconstructed image. The probability density function is given by,

$$Y(i,j) = \begin{cases} P_p & \text{for } x = 0 \text{ (pepper)} \\ P_s & \text{for } x = 255 \text{ (salt)} \\ 1 - (P_p + P_s) & \text{for } x = 0 < (i,j) < 255 \end{cases} \quad (1)$$

Where $P_p + P_s$ is the noise level in the image X

Usually, the denoising process starts at the origin of the corrupted image [6]. When the window length 3x3 is imposed on the corrupted image the available number of noise free pixels is crucial for the accurate estimation of the noisy pixel. That means, good amount of the noise free pixels within the detection window decide the strength of the estimated value that causes to preserve the image details. This point is accountable at high noise density. Because, the beginning estimation of the central pixel being tested (noisy or not), will have a hold on the future estimation of the remaining noisy pixels of the corrupted image. If the beginning estimation is itself deviated from the closer value, then all the future estimations will lead to larger and larger deviated values. Obviously, the restored image will be a poor qualified one. This is not at all a problem for the window is sliding over and down to where the previously processed pixels included within the detection window. For example, the 3x3 window is imposed on the highly corrupted image at its origin, is shown below.



The magnified window of size 3x3 involves the four noise free pixels (63, 68, 76, 78) at different locations. According to DBA and NDBA the center pixel being processed (the noise pixel 255) cannot be estimated at all. In DBA, if median pixel is also a noise pixel, it is replaced by left neighborhood pixel. But it is 255 also a noisy pixel (underlined pixel). In NDBA, the noise estimation is carried out by the mean of the previously processed the four uncorrupted pixels. This cannot be fulfilled here. The probability of the noise free pixels can be estimated using the following relation [3].

$$ND = \frac{m.n - Sij}{m.n} \qquad 0 \leq ND \leq 1 \qquad (2)$$

Where ND is noise density, *mn* is window size and S_{ij} is the number of noise free pixels within that window. At high noise level (>75%), the number of noise free pixels available within the 3x3 window are only two (since, the impulse noise is corrupted randomly across the entire image, the number of noise free pixels available within the window is also random). That means, good amount of noise free pixels cannot be obtained to estimate the noise pixel, hence, the accurate estimation of the noise pixel is not appropriate within the 3x3 window. At this stage the only choice is, extending the window to 5x5 size. Therefore, the proposed algorithm is sufficient to handle the impulse noise estimation incorporating a satisfied number of good pixels using a threshold α (at least 4 to 5 pixels) within the 5x5 window can have even at 85% - 90% noise level.

A. *Cloud model and its Metrics*
The concept of randomness and fuzziness can be integrated by a universal Cloud Model (CM). Distribution of pixel values *x*(i,j) on normal domain is called cloud and each pixel in the domain is called cloud drop. The cloud employs its three parameters to represent the qualitative concept. Ex is the *expectation* of the cloud drops' distribution in the domain, it points out which drops can best represent the concept and reflects the distinguished feature of the concept. En, *Entropy,* is the uncertainty measurement of the qualitative concept, which is determined by both the randomness and the fuzziness of the concept. It represents the value region in which the drop is acceptable by the concept, while reflecting the correlation of the randomness and the fuzziness of the concept. *Hyper entropy* He, can be described as entropy En of entropy, reflecting the degree of dispersion of the cloud droplets [8]. CM can generate a normal cloud using two basic tools: Backward Cloud Generator (BCG) and Forward Cloud Generator (FCG) and it is characterized by three parameters. The normal random number generation method in FCG cloud generator overcomes the insufficiency of common method to generate random numbers. It can produce random numbers which can be predictable and replicated, and this random numbers present to be a random sequence as a whole. According to the normal cloud generator, the certainty degree of each drop is a probability distribution rather than a fixed value. It means that the certainty degree of each drop is a random value in a dynamic range. If He of the cloud is 0, then the certainty degree of each drop will change to be a fixed value. The fixed value is the expectation value of the certainty degree. In fact, the value is also the unbiased estimation for the average value of the certainty degrees in the range. The cloud drops and their expectations of certainty degrees can compose a curve, and the curve is the Cloud Expectation Curve (CEC), is given by, $\mu = \exp(-(x_i - Ex)^2/2En^2)$ (3) The pixels closer to expectation Ex, gains higher certainty degree and those pixels farther to Ex, gains lower certainty degree. With this stabilization tendency of the CM, it promises to generate an appropriate reconstruction value for the respective noise pixels. Thus the image local features are possibly preserved.

III. PROPOSED ALGORITHM
Since, even at 90% noise level, decision based detection approach [6] has robustness in the detection of the noise pixel; PA selected the 3x3 window with the same detection method. The pixel being processed *x*(i,j) is checked between the extreme values within the 3x3 detection window, in the dynamic range (0, 255). If the current pixel *x*(i,j) lies between 0 and 255, it is treated as an uncorrupted pixel; kept unchanged. Otherwise, it is a noisy pixel need to be estimated by the median value. If the median is also a noisy candidate,

ISSN 2347-4289

the number of good pixels collected from the same window is checked up to the threshold level α. If it is satisfied, the estimation is carried out by the weighted fuzzy mean. Otherwise, the window is expanded by one pixel outward in all four sides, to collect the good pixels in the neighborhood. Since, the certainty degree of each good pixel is a soft values between [0,1] and are the weights, the noisy pixels are replaced by the closest pixel value [8].

A. Noise detection and Estimation:

Let X denote the noise corrupted image and let the centered pixel $x(i,j)$ is to be checked whether it is noise pixel or not, within the window $w(i,j)^{(2N+1)x(2N+1)}$ at location (i,j). i.e., $w(i,j) = x(i+p, j+q)\ \forall\ p,q \in (-N, N)$ where N is positive odd integer. Let X_{min}, X_{med} and X_{max} be the gray values in the sliding window. Let y denote the restored image, after the filtering process. The proposed filtering frame work consists of the *noise detection and noise estimation* stages.

Noise detection: the noise detection starts at the origin and the detection window is sliding in the forward direction.

- **Step 1:** Impose a window $\mathbf{w}(i,j)^{(2N+1)\times(2N+1)}$ on the corrupted image X, denote n as the number of good pixels. Initialize N = 1 and α.
- **Step 2:** Find the X_{min}, X_{med} and the X_{max} value in the $\mathbf{w}(i,j)^{(2N+1)\times(2N+1)}$.
- **Step 3:** Using flag = 1, for noisy pixels and flag = 0, for noise free pixels, process the centered pixel $x(i,j)$ of $\mathbf{w}(i,j)^{(2N+1)\times(2N+1)}$ to check if noisy pixel or not. If $X_{min} < x(i,j) < X_{max}$. Then, $x(i,j)$ is noise free pixel and is left unchanged. i,e., $y(i,j) = x(i,j)$. Otherwise, go to step 4.
- **Step 4:** If $X_{min} < X_{med} < X_{max}$. Then, $y(i,j) = X_{med}$. Otherwise, $x(i,j)$ is a noisy pixel, which need to be replaced.

Noise estimation **:**The noisy pixel $x(i,j)$ is replaced by the weighted fuzzy mean estimation of the collected uncorrupted pixels within the $\mathbf{w}(i,j)^{(2N+1)\times(2N+1)}$. i,e., $X_n = [x(i+p,j+q)]$.

- **Step 5:** Vector all the uncorrupted pixels obtained within the $\mathbf{w}(i,j)^{(2N+1)\times(2N+1)}$, with $n < α$. If α is not satisfies, then, N = N+1, go to step 2. Otherwise, go to step 6.
- **Step 6:** Compute the expectation Ex_n, of X_n. i,e., $Ex_n = \frac{1}{n}\sum x(i+p,j+q),\ x(i+p,j+q) \in X_n$ (4)
- **Step 7:** Compute the entropy En of X_n, i,e., $En_n = \sqrt{\frac{\pi}{2}} \times \frac{1}{n}\sum |x(i+p,j+q) - Ex|,\ x(i+p,j+q) \in X_n$ (5)
- **Step 8:** Calculate the weights of X_n. i,e., $\mu_{xn(i+1,j+1)} = \exp[-((x(i+p,j+q) - Ex_n)^2 / 2\, En_n^{\ 2}]$ (6)
- **Step 9:** Then, calculate the weighted mean, by multiplying each uncorrupted pixels with their weights and taking mean of all those values. $y(i,j) = \frac{1}{n}\sum x(i+p,j+q) \times \mu_{xn(i+p,j+q)}$. (7)

The steps 1 through 9 are repeated for the whole image, by subsequently moving the window to form a new set of values, with the next pixel to be processed at the centre of the window.

B. Illustration

For understanding of the proposed algorithm steps, a 3x3 windowed sub-image, shown, is illustrated below: Assume that the central pixel $x(i,j)$, **255** lies at an edge of the image.

$$X = \begin{bmatrix} 139 & 119 & 64 & 58 & 57 \\ 135 & \mathbf{255} & 255 & 255 & 65 \\ 255 & 255 & 255 & 63 & 59 \\ 0 & 255 & 58 & 75 & 79 \\ 112 & 111 & 117 & 255 & 0 \end{bmatrix}$$

Edge of image X

For $w(i,j)^{(2N+1)x(2N+1)}$ N =1, detection window = $w(i,j)^{3x3}$

$X_{min} = 64$, $X_{med} = 255$; $X_{max} = 255$; the centered pixel being tested whether it is a noisy pixel or not is $x(i,j) = 255$.

$X_{min} < x(i,j) < X_{max}$ → 64< 255 < 255; and

$X_{min} < X_{med} < X_{max}$ → 64< 255 < 255;

both conditions false, hence, $x(i,j)$ is a noisy pixel.

To replace the noisy pixel $x(i,j)$, collect available neighborhood pixel values, X_n = [139, 119, 64, 135].

Expectation Ex = 114.25; Entropy En = 31.49; the weights:

$\mu_{x(i-1,j+1)}$ = [0.734249,0.988686,0.279891,0.804827].

$y(i,j) = 86$; i,e., noisy pixel $x(i,j)$ is replaced by the reconstructed value **86**.

For the same example, DBA and NDBA noise estimation is shown in TABLE-I.

TABLE – I COMPARITION OF NOISE ESTIMATION

Noisy pixel *x*(i,j)	*y*(i,j) of different filters		
	DBA	NDBA	PA
255	135	114	86

Since $x(i,j) = 255$, DBA replaced it by the processed left neighborhood pixel, 135. This value is certainly high to recover the edge of the image. NDBA estimates 114 from the mean of already processed previous neighborhood pixel values. Though it estimates lesser intensity value compared to DBA, the edge preservation is not satisfactory. Our PA contributes with an appropriate value to retain the local features of the image.

ISSN 2347-4289

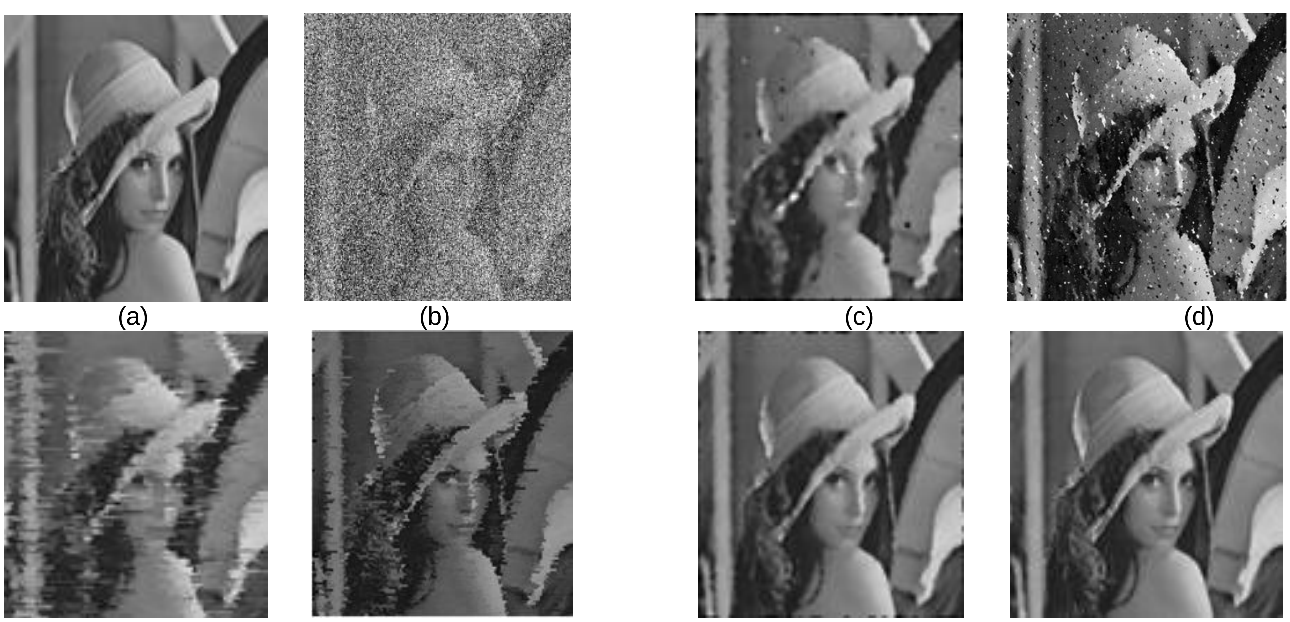
IV. SIMULATION RESULTS

The commonly tested 8-bit gray scale images of 512x512, homogeneous region Lena and high activity Bridge, have been used for simulation. These images are corrupted by salt and pepper noise with unequal probabilities at various noise densities ranging from 10% to 90%. The restoration performances are quantitatively measured by PSNR:

$$\text{PSNR} = 20 \log_{10} 255/\text{MAE} \quad (8)$$

$$\text{MAE} = \frac{1}{MN} \sum_{ij}^{MN} \left(y(i,j) - x(i,j)\right)^2 \quad (9)$$

Where M and N are the total number of pixels in rows and column of the image; x(i,j) and y(i,j) denote original and restored image pixels, respectively. The PA has superior performance in comparison, especially, preserving the image details, with high computational efficient filters. Fig.1 and fig.2 show the original image, corrupted image and restored images obtained by the simulation results for Lena and Bridge images for various filters such as, (i) ADMF, which uses 5×5, 7×7, and 15×15 window size for low, medium and high noise densities respectively, (ii) PSMF which uses 7×7 and 13×13 window size for low and high noise densities respectively, (iii) DBA filter which uses a small fixed size 3×3 window and (iv) NDBA with fixed size 3×3 window and (v) CMF 3x3, 5x5 and 7x7 with δ = 4. our PA uses a fixed size of 3×3 window for detection and 3x3 and 5x5 window size for noise estimation, with α = 4.



(e) (f) (g) (h)

Fig.1. Restoration results of different filters on Lena gray-scale image. (a) Original Image, (b)70% Noisy Image, (c) ADMF output, (d) PSMF output, (e) DBA output (f) NDBA output (g) CMF output (h) Proposed filter out

ISSN 2347-4289

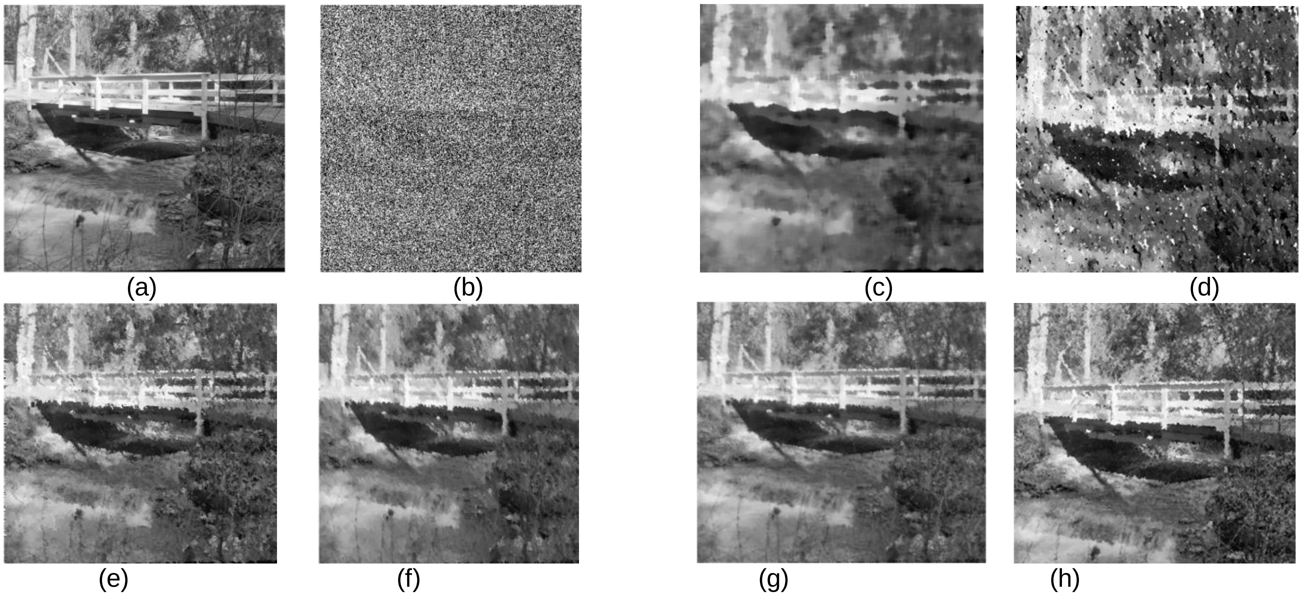


Fig.2. Restoration results of different filters on Bridge gray-scale image. (a) Original Image, (b)70% Noisy Image, (c) ADMF output, (d) PSMF output (e) DBA output (f) NDBA output (g) CMF output (h) Proposed filter output.

TABLE – II PSNR (dB) AND MAE OF DIFFERENT FILTERING METHODS FOR LENA.JPG TEST IMAGE.

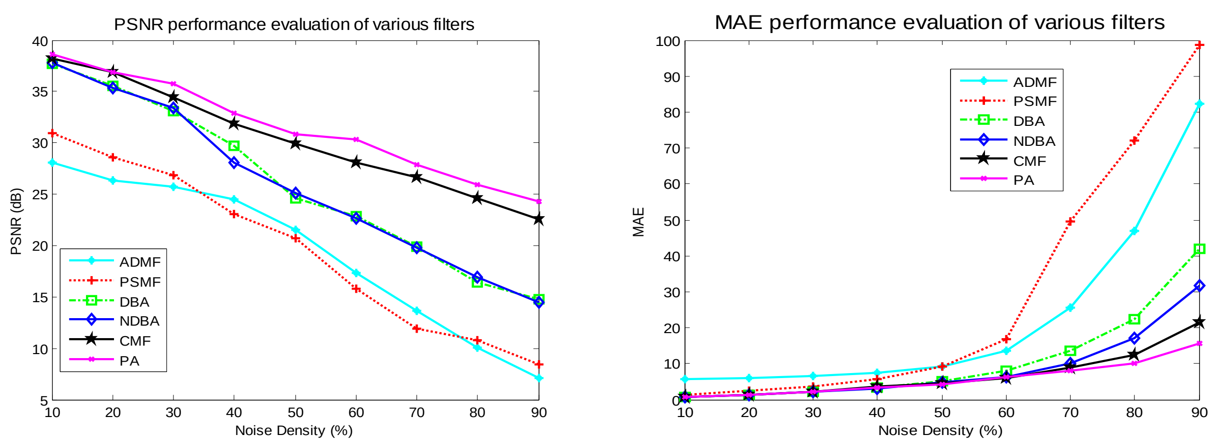
	PSNR in dB						MAE					
Noise ratio	ADMF	PSMF	DBA	NDBA	CMF	**PA**	ADMF	PSMF	DBA	NDBA	CMF	**PA**
10 %	25.1	28.9	34.6	34.7	36.6	**36.1**	6.7	2.1	1.2	1.3	1.1	**1.2**
20%	24.3	26.4	32.5	32.6	33.8	**33.2**	7.1	3.4	3.3	3.3	2.4	**2.8**
30%	22.7	24.8	30.3	30.5	31.7	**31.9**	8.4	4.6	5.8	5.3	3.6	**4.3**
40%	21.5	22.3	27.2	28.1	29.6	**30.4**	9.0	6.9	7.1	7.2	5.3	**6.0**
50%	18.0	19.1	24.0	24.1	26.9	**27.5**	10.9	10.2	9.8	9.4	6.9	**7.4**
60%	14.3	15.8	22.1	22.5	23.3	**24.1**	15.7	14.3	10.6	10.1	9.1	**9.2**
70%	11.4	11.4	18.4	18.5	21.3	**22.1**	27.3	59.5	13.1	13.0	10.9	**10.4**
80%	08.1	10.1	16.0	16.5	19.5	**20.4**	48.9	79.2	19.5	19.4	15.2	**13.6**
90%	06.4	08.5	14.1	14.3	17.9	**18.7**	89.3	98.9	39.7	38.8	27.6	**25.6**

ISSN 2347-4289

TABLE – III PSNR (dB) AND MAE OF DIFFERENT FILTERING METHODS FOR BRIDGE.JPG TEST IMAGE.

	PSNR in dB						MAE					
Noise Ratio	ADMF	PSMF	DBA	NDBA	CMF	**PA**	ADMF	PSMF	DBA	NDBA	CMF	**PA**
10 %	28.1	30.9	37.7	37.7	38.2	**38.1**	5.6	1.4	0.6	0.6	0.7	**0.6**
20%	26.3	28.6	35.5	35.6	36.9	**36.4**	6.1	2.4	1.3	1.3	1.4	**1.4**
30%	25.7	26.8	33.1	33.5	34.4	**34.9**	6.6	3.7	2.1	2.3	2.2	**2.1**
40%	24.5	23.1	29.6	30.1	31.9	**32.4**	7.4	5.6	3.1	3.2	3.3	**3.2**
50%	21.5	20.7	24.9	25.6	29.9	**30.5**	9.1	7.9	4.8	4.4	4.5	4.4
60%	17.3	15.8	22.9	23.5	28.0	**28.7**	13.7	12.3	6.4	6.8	6.1	**6.2**
70%	13.6	11.9	19.4	20.4	26.6	**27.5**	25.6	49.5	10.1	10.0	8.8	**8.4**
80%	10.1	10.8	16.8	17.5	24.5	**25.4**	46.9	71.2	17.0	16.5	12.2	**10.4**
90%	07.1	08.4	14.8	15.3	22.6	**23.3**	82.3	98.2	31.7	28.8	20.3	**15.9**

ADMF uses the larger window size (15x15) at high noise density, hence, the images in fig.1(c) and fig.2(c) are blurred and image details cannot be preserved. But, it works well for smaller window size at low noise density (<40%). In PSMF for detection, it uses 3x3 window; however, decision is not strong and creates dots on the restored images fig.1 (d) and fig.2 (d) and noise estimation is similar to ADMF. DBA uses a fixed 3x3 window length for both detection and estimation. The median value itself can be noisy, especially in the case of high noise density. It is in this case, the pixel value is replaced by the left neighborhood processed pixels. Hence, creates stripes on the restored images fig.1 (e) and fig.2 (e). NDBA is just modification to DBA, in which at high noise density, the noise pixel value is replaced by the mean of the neighborhood processed pixels. The corresponding restored images shown in fig.1(f) and fig.2 (f), respectively. CMF involves the cloud model based detector to detect the noise pixel. For computer simulation the window size is extended to 7x7; therefore, the restored images have little blurriness. Comparatively, our PA yields the better visual quality, even at high noise density. PSNR in dB and MAE performance evaluation of different filters for Lena and Bridge images is given in TABLE – II and TABLE – III, respectively. The comparison plots of different filters of PSNR and MAE of the corrupted image "Lena" is shown in fig. 3(a) and fig.3 (b), respectively. Reconstructed images with higher PSNR are judged better. MAE is used for numeric estimation of Algorithm's efficiency. That means, it proves the detail preserving characteristic of the filter. Comparing in either way, the PA is universal filter.



(a) (b)

Fig. 3(a) PSNR performance evaluation and (b) MAE performance evaluation of various algorithms for Lena test image.

ISSN 2347-4289

In real time work, high computational efficiency is the most important factor, while suppressing noise with better edge and detail preservation. Viewing with this fact, the PA performs two times faster than the uncertainty based filtering method, CMF and two times slower than DBA, only at high density noise level. This is because, when the window size is expanded, the amount of pixels that needs to be processed takes more time and hence, the computational efficiency of the filter will be decreased. Due to its high complexity in noise detection and also used a 7x7 window span at high noise level, the CMF is slower than the tested filters. DBA and NDBA used a fixed 3x3 window, therefore, both run faster, but they perform poor at high noise density. PA balances the drawback of CMF and decision based filters. All algorithms are coded in MATLAB (2012a) 7.14 version on a personal computer equipped with the 2.65-GHz CPU Intel core2duo processor and 2 GB RAM. CPU run time of different filters for Noise Density (10% to 90%) for Lena image is tabulated in table – IV and the comparative plot is shown in the fig. 4.

V. CONCLUSION

This paper frame work presents the decision based detector and the weighted fuzzy mean estimation, as an effective combination that to suppress the high density salt and pepper impulse noise. Extensive experimental results reveal that the PA consistently out performs the existing filters, both traditional and popular methods, by attaining higher PSNR, lower MAE and superior computational efficiency across a wide range of noise densities from 10% to 90% with good detail preservation of the image, accurately.

TABLE – IV CPU RUN TIME (Sec) OF DIFFERENT FILTERS FOR LENA.JPG TEST IMAGE.

Noise ratio	ADMF	PSMF	DBA	NDBA	CMF	PA
	0.01	0.18	0.87	0.88	0.98	0.88
20%	0.01	0.21	0.88	0.88	1.21	0.88
30%	0.02	0.19	0.88	0.89	1.89	0.91
40%	0.05	0.21	0.89	0.89	2.98	0.98
50%	0.09	0.19	0.91	0.90	3.42	1.19
60%	0.12	0.21	0.91	0.91	4.26	2.21
70%	0.22	0.21	0.92	0.91	4.65	2.36
80%	0.43	0.21	0.93	0.91	5.29	2.86
90%	0.42	0.22	0.94	0.92	5.55	3.21

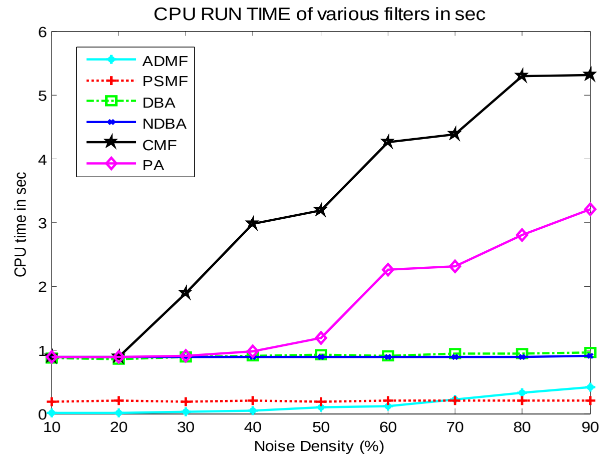


Fig. 4. Comparative plots for CPU run time of various filters ranging from (10% - 90%) noise level.

REFERENCE

[1]. Rafael Gonzalez, Richard Woods, "Digital Image Processing". Pearson Publications, 2002. 2nd Edition.

[2]. A.K.Jain,"Fundamentals of digital image processing". Prentice Hall, 1989.

[3]. S.-J. Ko and Y. H. Lee, "Center weighted median filters and their applications to image enhancement," IEEE Trans. Circuits and Systems, vol. 38, no. 9, pp. 984–993, 1991.

[4]. Z. Wang and D. Zhang, "Progressive switching median filter for the removal of impulse noise from highly corrupted images," IEEE Trans. on Circuits and Systems II: Analog and Digital Signal Processing, vol. 46, no. 1, pp. 78–80, 1999.

[5]. P.-E. Ng and K.-K. Ma, "A switching median filter with boundary discriminative noise detection for extremely corrupted images," IEEE Trans. Image Process. vol. 15, no. 6, pp. 1506–1516, Jun. 2006.

[6]. K. S. Srinivasan, D. Ebenezer, "A New Fast and Efficient Decision-Based Algorithm for Removal of High-Density Impulse Noises," IEEE Signal Processing Papers, Vol. 14, No. 3, pp. 189-192, March 2007.

[7]. Madhu S. Nair, K. Revathy, and Rao Tatavarti, "Removal of Salt and Pepper Noise in Images: A New Decision-Based Algorithm", Proceedings of the International Multi-Conference of Engineers and Computer Scientists 2008 Vol. I IMECS 2008, 19-21 March, 2008, Hong Kong.

[8]. Zhe Zhou, "Cognition and Removal of Impulse Noise With Uncertainty", IEEE Transactions on Image Processing, Vol. 21, No. 7, July 2012, pp-3157-3164