Texture Synthesis Based Adaptive Median Filter

Abdul Rasak Zubair

Abstract— Spatial filters suppress noise in an image by making each pixel's intensity roughly consistent with those of its nearest neighbours. Median Filter (MF) is an example of spatial filters which replaces each pixel with the median of its nearest neighbours. MF suppresses noise in any pixel which is affected by noise during image acquisition. However, MF adds noise to noise free pixels. An adaptive median filtering scheme which implements median filtering process only on pixels which are detected to be noisy is therefore desirable. An existing Noise Adaptive Fuzzy Switching Median Filter (NAFSMF) with two noise detection stages, two filtering stages and six fuzzy tuning parameters is somehow complex. A Texture Synthesis Based Adaptive Median Filter (TSBAMF) with a single tuning parameter is proposed. TSBAMF compares the actual intensity of each pixel with its texture synthesis' predicted intensity which is based on its nearest neighbours; the pixel is detected to be noisy if the absolute difference between the two values is greater than a tuning parameter. TSBAMF applies median filtering to only pixels detected to be noisy. For Salt and Pepper Noise (SPN) and Random Valued Impulse Noise (RVIN), TSBAMF is found to offer better image filtering/restoration and better visual quality compared with both MF and NAFSMF. TSBAMF satisfactorily restore corrupted images with improved Peak Signal to Noise Ratio (PSNR) and high Gain for noise densities up to 90%.

Index Terms— Adaptive Median Filter, Gain, Median Filter, Noise, Noise density, Peak Signal to Noise Ratio, Pixel's Neighbours, Texture Synthesis.

1 Introduction

SPATIAL filters are used to suppress various types of noise in digital images [1], [2], [3], [4], [5], [6]. Like other types of signals, an acquired image g(m,n) usually contains departures from the ideal or true image f(m,n). Such departures are referred to as noise [3], [7], [8]. Noise η(m,n) is added to the true image during image acquisition as illustrated in Fig. 1 and (1) [1], [7], [8]. Examples of sources of noise are variations in the detector sensitivity, environmental variations, discrete nature of radiation, transmission or quantization errors. There are many types of noise. Impulse noise is considered in this work. Impulse noise can be classified mainly in two categories; Salt and Pepper Noise (SPN) [9] and Random Valued Impulse Noise (RVIN) [10].

$$g(m,n) = f(m,n) + \eta(m,n) \quad (1)$$

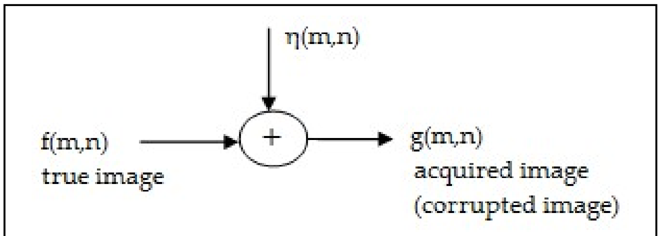


Fig. 1. Model of Image Acquisition.

Spatial filtering deduce an estimate $f_e(m,n)$ of the true image from the acquired image g(m,n) with the aid of convolution. A function of values of g in a predefined neighborhood of (m,n) is used to determine the value of f_e at (m,n) as described in (2) [1], [2], [7], [8].

$$f_e(m,n) = g(m,n) \otimes h(m,n) \quad (2)$$

The filter function h(m,n) is known as the kernel. The kernel is usually a square matrix with size 3 by 3 or 5 by 5 or 7 by 7 or 9 by 9. 3 by 3 is preferred as larger kernel sizes result in blurring [11]. Among the different types of kernel, Median Filter (MF) kernel is adjudged to be the best as it has high noise suppression capability and high computation efficiency [7], [8], [11].

Spatial filtering is referred to as local processing because it makes each pixel's intensity roughly consistent with those of its nearest neighbours. This process is applied to every pixel in the image. Spatial filtering suppresses noise in any pixel which is affected by noise during image acquisition. However, spatial filtering adds noise to noise free pixels. An adaptive spatial filtering scheme which implement spatial filtering process only on pixels which are detected to be noisy is therefore desirable.

In [12], Govindan and Saravanakumar proposed Noise Adaptive Fuzzy Switching Median Filter (NAFSMF) which involves two noise detection stages and two filtering stages. The first noise detection stage involves comparing each pixel with its 120 nearest neighbours within an 11 by 11 window neighbourhood with the aid of three fuzzy tuning parameters. Second noise detection stage involves comparing each pixel with its 24 nearest neighbours within a 5 by 5 window neighbourhood with the aid of another set of three fuzzy tuning parameters. First filtering stage involves window neighbourhood ranging from 3 by 3 to 9 by 9 while second filtering stage involves a 5 by 5 window neighbourhood. NAFSMF is rather complex.

In this work, Texture Synthesis Based Adaptive Median Filter (TSBAMF) is proposed. A texture synthesis method starts from a sample image and attempts to produce a texture with a visual appearance similar to that sample [13], [14], [15]. Potential applications of texture synthesis include occlusion fill-in, inpainting, and compression.

• *Abdul Rasak Zubair (PhD) is currently a Senior Lecturer in the Electrical & Electronic Engineering Department, University of Ibadan, Nigeria, PH- +2348023278605. E-mail: ar.zubair@ui.edu.ng; ar.zubair@yahoo.co.uk*

International Journal of Scientific & Engineering Research, Volume 9, Issue 5, May-2018
ISSN 2229-5518

In this proposed method, noise detection stage involves an 11 by 11 window neighbourhood and the noise suppression stage is limited to a 3 by 3 window neighbourhood. In the detection stage, each pixel intensity I_p is noted. Texture Synthesis predicts I_{ts} as the intensity of the pixel based on information from 40 of its nearest neighbours within an 11 by 11 window neighbourhood. If the absolute difference between I_p and I_{ts} is greater than a tuning parameter cc, the pixel is considered to be noisy. In the filtering stage, only pixels which are found to be noisy are replaced by the median of its nearest neighbours within a 3 by 3 window neighbourhood. For high density noisy images, the process is repeated once or twice.

Test images [16] are corrupted with varying densities (d) of Salt and Pepper Noise (SPN) and Random Valued Impulse Noise (RVIN) [11]. These test images are filtered with the proposed TSBAMF. TSBAMF is compared with MF [11] and NAFSMF [12].

2 Texture Synthesis Based Adaptive Median Filter (TSBAMF)

2.1 First Stage: Texture Synthesis based Noise Detection

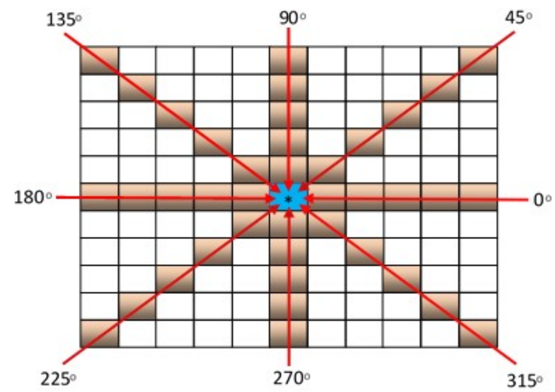
Each of the colour components of every pixel in the input image g is checked for noise presence. Input noisy image g is an M by N by 3 matrix (3D) which is like three M by N matrices (2D). These three 2D matrices are scanned for noise presence one after the other. A texture synthesis method developed in [17] is adapted for noise detection.

Fig. 2 illustrates the texture synthesis method. The current pixel being scanned for noise is labelled as p and is painted blue. It's at location (m,n) and its actual intensity value in g is I_p. The goal of texture synthesis is to guess or predict a value I_{ts} of the pixel p based on the intensities of its neighbours. The intensity of a pixel is usually close to the intensities of its neighbours except at edges.

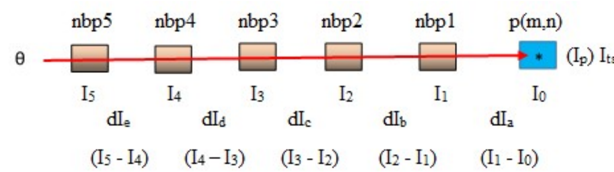
An 11 by 11 window neighbourhood is selected with p at the center as shown in Fig. 2(a). Eight directions are selected: θ = 0°, 45°, 90°, 135°, 180°, 225°, 270°, and 315°. These are vertical, horizontal and diagonal directions. Five neighbouring pixels nbp1, nbp2, nbp3, nbp4 and nbp5 are selected along each direction θ; nbp1 being the closest neighbour to p as illustrated in Fig. 2(b). I_1, I_2, I_3, I_4, and I_5 are the intensities of the five neighbouring pixels respectively.

Colours are treated as fluid that flow or diffuse from the neighbouring pixels to the pixel of interest along a direction θ: starting from nbp5 through nbp4, through nbp3, through nbp2 and through nbp1 to give I_0 at p. I_{ts} is the weighted average of the I_o's in the eight directions. Only 40 neighbouring pixels along the selected directions out of 120 neighbouring pixels participate in the texture synthesis process. The addresses of the 40 participating neighbouring pixels relative to the location (m,n) of p are listed in Table 1.

The pattern of flow along each direction is studied and limiting factor lf and weight wg are applied to obtain the contribution of the directional flow to the pixel being synthesised. lf and wg are determined based on Fig. 2(b) which shows the neighbouring pixels along a direction. Four different cases are identified.



(a) 11 by 11 window neighbourhood



(b) Five participating neighbouring pixels along a direction θ and the pixel p(m,n) under consideration

Fig. 2. Texture Synthesis.

Key
- p(m,n) is the pixel under consideration
- Neighbouring pixel which participates in texture synthesis
- Neighbouring Pixel which does not participate in Texture Synthesis

I_p is the actual intensity of p in g.
I_0 is the intensity at p contributed by neighbouring pixels along a direction θ.
I_{ts} is the texture synthesis predicted intensity of p. I_{ts} is the weighted average of I_0's in the eight directions.

2.1.1 Case 1: I1 ≥ I2 ≥ I3 ≥ I4 ≥ I5 or I1 < I2 < I3 < I4 < I5 (Continuous Trend I5 to I1)

The average of dI_e, dI_d and dI_c should give dI_b if the flow is effective. However, $dI_{bav} = \frac{dI_e + dI_d + dI_c}{3}$. The ratio of dI_b to the average of dI_e, dI_d and dI_c is termed limiting factor which is given as $lf = \frac{dI_b}{dI_{bav}}$ and is used to determine dI_a as

$$dI_a = \frac{lf(dI_e + dI_d + dI_c + dI_b)}{4} \tag{3}$$

dI_a is subject to some limit.

If $dI_a \le$ ck, $wg = 4$, $I_0 = I_1 - dI_a$. (4)

If $dI_a >$ ck, $wg = 1$, $I_0 = I_1$. (5)

where ck is a limiting constant. Optimum value of ck has been obtained as 4 in [17].

International Journal of Scientific & Engineering Research, Volume 9, Issue 5, May-2018
ISSN 2229-5518

TABLE 1
NEIGHBOURING CONTRIBUTING PIXELS ALONG EIGHT DIFFERENT DIRECTIONS OF FLOW

Pixel Name	Directions (θ)			
	0^O	45^O	90^O	135^O
p	(m,n)	(m,n)	(m,n)	(m,n)
nbp1	(m, n+1)	(m-1, n+1)	(m-1, n)	(m-1, n-1)
nbp2	(m, n+2)	(m-2, n+2)	(m-2, n)	(m-2, n-2)
nbp3	(m, n+3)	(m-3, n+3)	(m-3, n)	(m-3, n-3)
nbp4	(m, n+4)	(m-4, n+4)	(m-4, n)	(m-4, n-4)
nbp5	(m, n+5)	(m-5, n+5)	(m-5, n)	(m-5, n-5)
Pixel Name	Directions (θ)			
	180^O	225^O	270^O	315^O
p	(m,n)	(m,n)	(m,n)	(m,n)
nbp1	(m, n-1)	(m+1, n-1)	(m+1, n)	(m+1, n+1)
nbp2	(m, n-2)	(m+2, n-2)	(m+2, n)	(m+2, n+2)
nbp3	(m, n-3)	(m+3, n-3)	(m+3, n)	(m+3, n+3)
nbp4	(m, n-4)	(m+4, n-4)	(m+4, n)	(m+4, n+4)
nbp5	(m, n-5)	(m+5, n-5)	(m+5, n)	(m+5, n+5)

2.1.2 Case 2: I1 ≥ I2 ≥ I3 ≥ I4 < I5 or I1 < I2 < I3 < I4 > I5 (Continuous Trend I4 to I1 only)

I_5 is neglected. The average of dI_d and dI_c should give dI_b if the flow is effective. However, $dI_{bav} = \frac{dI_d + dI_c}{2}$. The ratio of dI_b to the average of dI_d and dI_c is termed limiting factor which is given as $lf = \frac{dI_b}{dI_{bav}}$ and is used to determine dI_a as

$$dI_a = \frac{lf(dI_d + dI_c + dI_b)}{3} \tag{6}$$

dI_a is subject to some limit.

If $dI_a \le ck$, $wg = 3$, $I_0 = I_1 - dI_a$. (7)

If $dI_a > ck$, $wg = 1$, $I_0 = I_1$. (8)

2.1.3 Case 3: I1 ≥ I2 ≥ I3 < I4 or I1 < I2 < I3 > I4 (Continuous Trend I3 to I1 only)

I_5 and I_4 are neglected. dI_c should should be equal to dI_b if the flow is effective. However, this may not be the case. The ratio of dI_b to dI_c is termed limiting factor which is given as $lf = \frac{dI_b}{dI_c}$ and is used to determine dI_a as

$$dI_a = \frac{lf(dI_c + dI_b)}{2} \tag{9}$$

dI_a is subject to some limit.

If $dI_a \le ck$, $wg = 2$, $I_0 = I_1 - dI_a$. (10)

If $dI_a > ck$, $wg = 1$, $I_0 = I_1$. (11)

2.1.4 Case 3: I1 ≥ I2 < I3 or I1 < I2 > I3 (Continuous Trend I2 to I1 only)

I_5, I_4 and I_3 are neglected.

$$wg = 1,\ I_0 = I_1. \tag{12}$$

2.1.5 Averaging and Noise Detection Verdict

I_0 and wg for the eight directions are determined. The texture synthesis predicted intensity I_{ts} of the pixel p under consideration is obtained as the weighted average of the I_0's in the eight directions as in (13). I_{ts} is then compared with I_p and an indicator s is set to 0 or 1 as in (14). If the absolute difference between I_{ts} and I_p is greater than a tuning parameter cc, p is considered to be noisy and s is set to 1. Otherwise, p is considered to be noise free and s is set to 0. After some trials, tuning parameter cc=20 is found to be suitable.

$$I_{ts} = \frac{\sum_{\theta=0^o}^{\theta=315^o} wg(\theta) I_0(\theta)}{\sum_{\theta=0^o}^{\theta=315^o} wg(\theta)} \tag{13}$$

$$s = \begin{cases} 0 & \text{if } |I_{ts} - I_p| < cc \\ 1 & \text{if } |I_{ts} - I_p| \ge cc \end{cases} \tag{14}$$

Percentage noisy pixels detected is given as dd and is obtained as in (15). dd can then be compared to the known actual percentage noise density d of the noisy image g. d may not be known. dd may not be exactly equal to d as its possible that some noisy pixels are not detected and some noise free pixels are erroneously detected as noisy.

$$dd = \frac{Number\ of\ Detected\ Noisy\ Pixels \times 100}{3MN}\% \tag{15}$$

2.2 Second Stage: Noise Suppression

Each colour component of the filtered image $f_e(m,n)$ is obtained from the corresponding colour component of g(m,n) and s(m,n) as in (16). For detected noisy pixel (s=1), median of the values of g in a 3 by 3 window neighbourhood of (m,n) is used as the value of f_e at (m,n). Any noiseless pixel (s=0) in g is simply copied into f_e. This is an adaptive median filter as the filtering is applied to only detected noisy pixels. The complete TSBAMF scheme is first stage and second stage.

$$f_e(m,n) = \begin{cases} \text{median of intensities in 3 by 3 window} \\ \text{neighbouhood of g(m,n) if s(m,n) = 1} \\ \\ \text{g(m,n) if s(m,n) = 0} \end{cases} \tag{16}$$

2.3 Two or Three Iterations for High Noise Density Input Images

For high noise densities, percentage noisy pixels detected (dd) may be far less than the actual noise density (d) in g. The reality is that the possible presence of noise in some of the neighbouring pixels along the eight directions may affect noise detection capability. It's, therefore, recommended to repeat the complete TSBAMF scheme once or twice with cc=15 for input

International Journal of Scientific & Engineering Research, Volume 9, Issue 5, May-2018
ISSN 2229-5518

images with high noise densities.

f_e after the first TSBAMF scheme (1st iteration with cc = 20) is re-sent as input image g and the complete TSBAMF scheme is repeated with cc = 15 (2nd iteration). f_e after the second TSBAMF scheme is re-sent as input image g and the complete TSBAMF scheme is repeated with cc=15 (3rd iteration). The percentage noisy pixels detected (dd) after the 2nd iteration is the cumulative dd for 1st and 2nd iterations. Similarly, the percentage noisy pixels detected (dd) after the 3rd iteration is the cumulative dd for 1st, 2nd and 3rd iterations.

2.4 Performance Metrics

The Peak Signal to Noise Ratio (PSNR) is a measure of the degree of corruption or degradation of an image with noise or/and blurring [8], [11], [18]. Equation (17) evaluates PSNRc which compares the corrupted input image g with the true image f. Equation (18) evaluates PSNRr which compares the filtered or recovered image f_e with the true image f. Subtracting PSNRc from PSNRr gives the Gain of the filter as in (19). Higher Gain indicates a higher degree of noise suppression by the filter. A negative Gain indicates that the filter adds more noise to the corrupted image instead of suppressing the noise in the image.

$$PSNRc = 10\log_{10}\left[\frac{255^2}{\frac{1}{3NM}\left[\sum_{m=1}^{M}\sum_{n=1}^{N}\sum_{t=1}^{3}\left(g(m,n,t)-f(m,n,t)\right)^2\right]}\right] \quad (17)$$

$$PSNRr = 10\log_{10}\left[\frac{255^2}{\frac{1}{3NM}\left[\sum_{m=1}^{M}\sum_{n=1}^{N}\sum_{t=1}^{3}\left(f_e(m,n,t)-f(m,n,t)\right)^2\right]}\right] \quad (18)$$

$$Gain = PSNRr - PSNRc \quad (19)$$

3 Results and Discussions

3.1 Five Test Images

Five test images [16] are selected to study the performance of the Texture Synthesis Based Adaptive Median Filter (TSBAMF). The test images are corrupted with Salt and Pepper Noise (SPN) or Random Valued Impulse Noise (RVIN) with noise densities 5%, 10%, 15%, 20%, 40%, 50%, 60%, 70% and 90% [11].

3.2 TSBAMF Results

The corrupted images were supplied to the TSBAMF as inputs one after the other. Three iterations were done for each test image. The results are summarised in Table 2 for SPN corrupted images and Table 3 for RVIN corrupted images. The actual noise density (d), the detected noise density (dd), PSNRc of corrupted image, PSNRr of the filtered image and filtering Gain are recorded in the Tables 2 and 3.

For lower actual noise densities d, detected noise density dd is usually greater than actual noise density d. For higher actual noise densities d, detected noise density dd is usually less than actual noise density d. As expected, cumulative detected noise density dd after 3rd iteration is greater than dd after 2nd iteration which in turn is greater than dd after 1st iteration.

For illustration, consider test image Lena corrupted with SPN with d=90% in Table 2. dd = 55.95% of all the pixels were detected to be noisy and filtered during the 1st iteration with Gain = 14.95 dB. For 2nd iteration, dd=62.27% which means that additional 6.32% (62.27%-55.95%) of all the pixels were detected to be noisy and filtered during the 2nd iteration and the Gain improved to 20.00 dB. For 3rd iteration, dd=65.67% which means that additional 3.40% (65.67%-62.27%) of all the pixels were detected to be noisy and filtered during the 3rd iteration and the Gain improved to 20.41 dB. This same trend is observed for other test images corrupted with SPN and RVIN and with various values of d in Tables 2 and 3 respectively. Tables 4 and 5 show some test images, corrupted test images and filtered images after 1st, 2nd and 3rd iterations for SPN and RVIN respectively.

For lower actual noise densities d, 1st iteration alone is sufficient and it gives the final TSBAMF results; highlighted with blue colour in Tables 2 and 3. 2nd and 3rd iterations are not useful as both PSNRr and Gain decreased; highlighted with yellow colour.

For medium actual noise densities d, 1st iteration alone is not sufficient; highlighted with pink colour in Tables 2 and 3. 2nd iteration is sufficient and it gives the final TSBAMF results; highlighted with blue colour. 3rd iteration is not useful as both PSNRr and Gain decreased; highlighted with yellow colour.

For higher actual noise densities d, 1st and 2nd iterations are not sufficient; highlighted with pink colour in Tables 2 and 3. 3rd iteration is sufficient and it gives the final TSBAMF results; highlighted with blue colour.

If going to the next iteration leads to reduction in PSNRr and Gain, then the current iteration is sufficient and the next iteration is neither necessary nor useful.

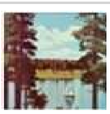
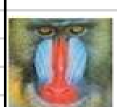
3.3 Comparison of TSBAMF Results with MF Results.

For the five test images, the final TSBAMF results (highlighted with blue colour) extracted for SPN and RVIN from Tables 2 and 3 respectively are compared with MF results for SPN and RVIN extracted from Tables 1 and 4 of [11] respectively as presented in Fig. 3. TSBAMF is found to give higher PSNRr and Gain in all cases compared with MF.

The following three limitations of MF were recorded in [11]. For both SPN and RVIN, MF Gain increases with noise density d up to 40% and then reduces with further increase in noise density. Median filtering of RVIN corrupted images is satisfactory for noise densities up to the maximum of 40%. Median filtering of SPN corrupted images is found satisfactory for noise densities up to the maximum of 60%. TSBAMF is free of these three limitations. The Gain of TSBAMF increases with noise density d up to 90% as shown in Fig. 3. Filtering by TSBAMF is found satisfactory up to 90% noise density for both SPN and RVIN corrupted images. This is illustrated in Table 6 which shows the appearance of some of the filtered images for both TSBAMF and MF. TSBAMF gives better image restoration and better visual quality compared with MF. Both TSBAMF and MF give higher Gain for SPN restoration compared with RVIN restoration.

International Journal of Scientific & Engineering Research, Volume 9, Issue 5, May-2018
ISSN 2229-5518

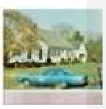
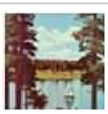
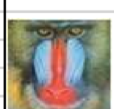
TABLE 2
TSBAMF RESULTS FOR SPN CORRUPTED IMAGES

Test Image		d %	5	10	15	20	40	50	60	70	90
		PSNRc (dB)	22.99	20.03	18.30	17.07	14.22	13.33	12.59	11.99	11.05
	1st iteration	dd %	6.52	10.21	13.88	17.48	30.82	36.81	42.31	47.38	55.95
		PSNRr (dB)	38.99	38.00	37.18	36.47	33.80	32.35	30.71	29.03	26.00
		Gain (dB)	16.00	17.97	18.88	19.40	19.58	19.01	18.12	17.04	14.95
	2nd iteration	dd %	10.94	14.55	18.16	21.78	35.14	41.26	47.01	52.50	62.27
		PSNRr (dB)	37.30	36.68	36.09	36.68	34.12	33.43	32.83	32.20	31.05
		Gain (dB)	14.31	16.65	17.79	19.60	19.90	20.10	20.24	20.21	20.00
Lena	3rd iteration	dd %	14.09	17.66	21.24	24.87	38.23	44.38	50.15	55.77	65.67
		PSNRr (dB)	36.81	36.25	35.74	36.25	33.92	33.30	32.79	32.24	31.45
		Gain (dB)	13.82	16.23	17.44	19.18	19.70	19.97	20.20	20.25	20.41
Test Image		d %	5	10	15	20	40	50	60	70	90
		PSNRc (dB)	22.80	19.77	18.03	16.84	13.97	13.07	12.36	11.74	10.79
	1st iteration	dd %	8.20	12.29	16.17	19.76	33.01	38.70	44.02	48.86	56.99
		PSNRr (dB)	30.64	30.53	30.49	30.20	29.38	28.72	27.92	27.01	24.64
		Gain (dB)	7.83	10.76	12.46	13.36	15.42	15.65	15.56	15.27	13.85
	2nd iteration	dd %	13.40	17.34	21.14	24.62	37.81	43.55	49.08	54.24	63.53
		PSNRr (dB)	30.25	30.09	29.97	29.82	29.35	29.06	28.87	28.65	28.02
		Gain (dB)	7.44	10.32	11.94	12.97	15.38	15.98	16.51	16.90	17.23
Pepper	3rd iteration	dd %	16.65	20.54	24.29	27.75	40.93	46.70	52.25	57.48	66.91
		PSNRr (dB)	30.16	30.01	29.88	29.74	29.26	28.99	28.83	28.60	28.17
		Gain (dB)	7.35	10.24	11.85	12.90	15.29	15.92	16.47	16.85	17.38
Test Image		d %	5	10	15	20	40	50	60	70	90
		PSNRc (dB)	22.90	19.95	18.22	17.03	14.15	13.25	12.53	11.93	10.97
	1st iteration	dd %	10.96	14.19	17.42	20.59	32.97	38.72	43.95	48.87	57.41
		PSNRr (dB)	28.39	28.31	28.15	28.10	27.31	26.75	26.15	25.41	23.58
		Gain (dB)	5.49	8.36	9.93	11.07	13.16	13.50	13.61	13.48	12.61
	2nd iteration	dd %	20.74	23.92	27.12	30.30	42.79	48.71	54.23	59.54	69.41
		PSNRr (dB)	27.93	27.82	27.68	27.56	27.06	26.79	26.61	26.33	25.81
		Gain (dB)	5.03	7.87	9.46	10.53	12.91	13.55	14.08	14.40	14.84
House	3rd iteration	dd %	28.74	31.81	34.98	38.14	50.52	56.44	62.02	67.42	77.48
		PSNRr (dB)	27.75	27.64	27.51	27.38	26.89	26.65	26.46	26.21	25.81
		Gain (dB)	4.84	7.69	9.28	10.36	12.74	13.40	13.92	14.28	14.84
Test Image		d %	5	10	15	20	40	50	60	70	90
		PSNRc (dB)	22.74	19.78	18.01	16.81	13.97	13.06	12.35	11.74	10.79
	1st iteration	dd %	13.96	17.78	21.51	25.04	37.62	43.08	48.04	52.61	60.19
		PSNRr (dB)	29.39	29.19	28.89	28.68	27.66	27.07	26.33	25.57	23.48
		Gain (dB)	6.66	9.41	10.88	11.88	13.69	14.01	13.98	13.82	12.69
	2nd iteration	dd %	24.73	28.36	31.92	35.34	47.65	53.15	58.26	63.13	72.01
		PSNRr (dB)	28.62	28.45	28.24	28.06	27.44	27.18	26.88	26.58	25.91
		Gain (dB)	5.88	8.67	10.22	11.26	13.47	14.11	14.53	14.84	15.12
Boat	3rd iteration	dd %	31.92	35.44	38.91	42.33	54.44	59.90	65.03	69.93	79.02
		PSNRr (dB)	28.32	28.16	27.97	27.80	27.23	26.98	26.73	26.48	26.01
		Gain (dB)	5.58	8.38	9.96	10.99	13.26	13.92	14.38	14.73	15.22
Test Image		d %	5	10	15	20	40	50	60	70	90
		PSNRc (dB)	23.04	20.09	18.36	17.15	14.28	13.36	12.67	12.07	11.11
	1st iteration	dd %	30.71	33.57	36.32	38.93	48.38	52.63	56.24	59.56	65.24
		PSNRr (dB)	23.59	23.48	23.36	23.23	22.70	22.40	22.08	21.65	20.72
		Gain (dB)	0.55	3.39	5.00	6.07	8.41	9.04	9.41	9.57	9.61
	2nd iteration	dd %	49.71	52.59	55.32	57.88	67.76	72.33	76.28	80.20	87.45
		PSNRr (dB)	23.10	22.99	22.90	22.80	22.43	22.27	22.10	21.93	21.60
		Gain (dB)	0.06	2.91	4.54	5.65	8.14	8.91	9.43	9.85	10.49
Baboon	3rd iteration	dd %	61.25	64.05	66.68	69.16	79.00	83.61	87.65	91.76	99.39
		PSNRr (dB)	22.87	22.77	22.69	22.60	22.25	22.11	21.95	21.81	21.52
		Gain (dB)	-0.17	2.68	4.33	5.45	7.97	8.75	9.28	9.73	10.42

Key: General; final TSBAMF results: This iteration is sufficient; This iteration is not sufficient.; This iteration is not necessary

International Journal of Scientific & Engineering Research, Volume 9, Issue 5, May-2018
ISSN 2229-5518

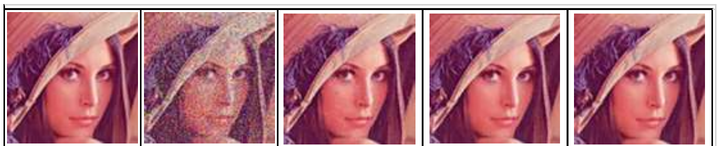
TABLE 3
TSBAMF Results for RVINN Corrupted Images

Test Image	Iteration										
Test Image		d %	5	10	15	20	40	50	60	70	90
		PSNRc (dB)	24.42	21.42	19.72	18.49	15.63	14.72	14.01	13.41	12.46
	1st iteration	dd %	6.22	9.71	13.14	16.56	29.41	35.17	40.43	45.20	53.54
		PSNRr (dB)	39.03	37.85	36.87	36.07	32.77	30.89	29.34	27.57	24.47
		Gain (dB)	14.61	16.43	17.15	17.57	17.14	16.17	15.33	14.16	12.02
	2nd iteration	dd %	10.69	14.17	17.67	21.11	34.42	40.69	46.52	52.23	63.00
		PSNRr (dB)	37.38	36.64	36.05	35.57	33.68	32.83	32.00	31.18	28.99
		Gain (dB)	12.96	15.22	16.33	17.07	18.05	18.12	17.99	17.77	16.54
Lena	3rd iteration	dd %	13.87	17.34	20.84	24.25	37.64	43.98	49.91	55.81	67.18
		PSNRr (dB)	36.87	36.22	35.69	35.27	33.54	32.79	32.09	31.47	29.86
		Gain (dB)	12.46	14.80	15.97	16.78	17.91	18.08	18.08	18.07	17.41
Test Image		d %	5	10	15	20	40	50	60	70	90
		PSNRc (dB)	23.46	20.52	18.76	17.55	14.70	13.79	13.05	12.47	11.53
	1st iteration	dd %	8.18	12.12	16.05	19.82	33.34	39.33	44.73	49.28	57.31
		PSNRr (dB)	30.66	30.46	30.24	29.99	28.63	27.73	26.38	25.13	22.59
		Gain (dB)	7.19	9.94	11.47	12.44	13.93	13.94	13.33	12.66	11.06
	2nd iteration	dd %	13.42	17.30	21.17	24.93	38.84	45.21	51.43	56.96	67.82
		PSNRr (dB)	30.26	30.11	29.93	29.75	29.12	28.76	28.28	27.76	26.43
		Gain (dB)	6.79	9.59	11.16	12.20	14.42	14.98	15.22	15.29	14.89
Pepper	3rd iteration	dd %	16.69	20.53	24.35	28.11	42.02	48.46	54.81	60.55	72.22
		PSNRr (dB)	30.16	30.02	29.84	29.67	29.06	28.75	28.39	27.96	27.15
		Gain (dB)	6.70	9.50	11.08	12.12	14.36	14.96	15.33	15.50	15.62
Test Image		d %	5	10	15	20	40	50	60	70	90
		PSNRc (dB)	25.50	22.52	20.79	19.58	16.72	15.83	15.09	14.48	13.57
	1st iteration	dd %	10.18	12.74	15.30	17.83	27.59	32.24	36.69	40.71	47.85
		PSNRr (dB)	28.38	28.24	28.12	27.94	27.03	26.48	25.90	25.13	23.63
		Gain (dB)	2.87	5.72	7.33	8.36	10.31	10.65	10.81	10.65	10.07
	2nd iteration	dd %	20.00	22.54	25.12	27.69	37.73	42.69	47.49	52.11	60.62
		PSNRr (dB)	27.95	27.82	27.69	27.54	26.89	26.59	26.24	25.85	25.07
		Gain (dB)	2.44	5.30	6.90	7.95	10.17	10.76	11.15	11.36	11.51
House	3rd iteration	dd %	28.00	30.50	33.05	35.56	45.51	50.51	55.35	60.01	68.71
		PSNRr (dB)	27.76	27.63	27.51	27.37	26.75	26.46	26.12	25.77	25.14
		Gain (dB)	2.25	5.11	6.72	7.78	10.02	10.63	11.04	11.29	11.58
Test Image		d %	5	10	15	20	40	50	60	70	90
		PSNRc (dB)	23.85	20.98	19.18	17.95	15.11	14.21	13.50	12.89	11.94
	1st iteration	dd %	13.70	17.26	20.86	24.29	36.27	41.36	45.97	50.26	57.48
		PSNRr (dB)	29.40	29.12	28.90	28.55	27.24	26.38	25.44	24.38	22.14
		Gain (dB)	5.55	8.14	9.72	10.60	12.12	12.16	11.93	11.49	10.20
	2nd iteration	dd %	24.52	27.94	31.44	34.80	46.87	52.39	57.53	62.60	72.22
		PSNRr (dB)	28.63	28.42	28.26	28.04	27.29	26.90	26.46	25.97	24.78
		Gain (dB)	4.78	7.44	9.07	10.09	12.18	12.68	12.96	13.08	12.84
Boat	3rd iteration	dd %	31.73	35.07	38.50	41.81	53.75	59.38	64.58	69.86	80.10
		PSNRr (dB)	28.33	28.14	28.00	27.79	27.13	26.76	26.41	26.02	25.14
		Gain (dB)	4.47	7.16	8.81	9.85	12.01	12.55	12.91	13.12	13.20
Test Image		d %	5	10	15	20	40	50	60	70	90
		PSNRc (dB)	24.43	21.44	19.74	18.55	15.65	14.74	14.06	13.43	12.49
	1st iteration	dd %	30.29	32.89	35.23	37.63	46.34	50.23	53.67	56.81	62.39
		PSNRr (dB)	23.59	23.47	23.33	23.20	22.55	22.16	21.77	21.20	20.09
		Gain (dB)	-0.84	2.02	3.59	4.65	6.90	7.42	7.72	7.77	7.60
	2nd iteration	dd %	49.30	51.91	54.26	56.66	65.83	70.17	74.08	78.07	85.50
		PSNRr (dB)	23.11	23.02	22.91	22.82	22.38	22.16	21.95	21.66	21.03
		Gain (dB)	-1.32	1.57	3.17	4.27	6.73	7.42	7.90	8.23	8.54
Baboon	3rd iteration	dd %	60.84	63.34	65.68	67.97	77.06	81.48	85.54	89.78	97.71
		PSNRr (dB)	22.88	22.80	22.70	22.62	22.23	22.03	21.84	21.59	21.07
		Gain (dB)	-1.55	1.36	2.96	4.07	6.58	7.29	7.78	8.16	8.58

Key:
- General
- final TSBAMF results: This iteration is sufficient
- This iteration is not sufficient.
- This iteration is not necessary

International Journal of Scientific & Engineering Research, Volume 9, Issue 5, May-2018
ISSN 2229-5518

TABLE 4
SOME TSBAMF RESULTS FOR LENA TEST IMAGE WITH SPN



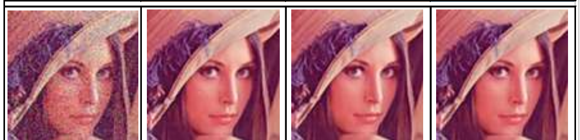
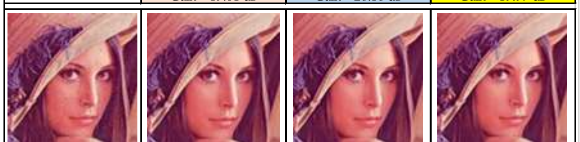
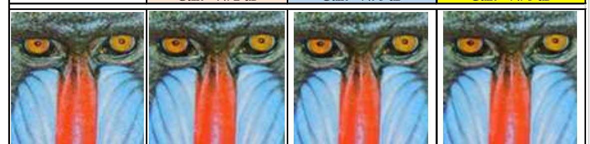
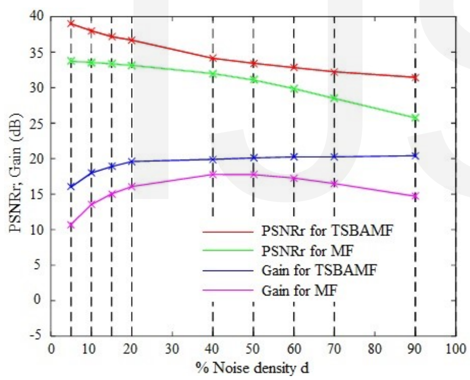
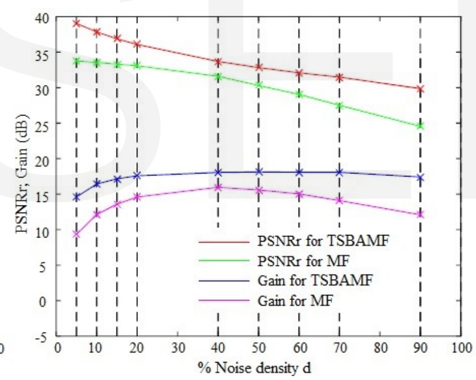
Lena (Original)	Lena with 90% SPN PSNRc = 11.05 Db	1st Iteration: dd = 55.85% 55.85% of all the pixels detected to be noisy PSNRr = 26.00 dB Gain = 14.95 dB	2nd Iteration: dd = 62.27% 6.32% of all the pixels detected to be noisy PSNRr = 31.05 dB Gain = 20.00 dB	3rd Iteration: dd = 65.67% 3.40% of all the pixels detected to be noisy PSNRr = 31.45 dB Gain = 20.41 dB
				
	Lena with 50% SPN PSNRc = 13.33 dB	1st Iteration: dd = 36.81% 36.81% of all the pixels detected to be noisy PSNRr = 32.35 dB Gain = 19.01 dB	2nd Iteration: dd = 41.26% 4.45% of all the pixels detected to be noisy PSNRr = 33.43 dB Gain = 20.10 dB	3rd Iteration: dd = 44.38% 3.11% of all the pixels detected to be noisy PSNRr = 33.30 dB Gain = 19.97 dB
				
	Lena with 5% SPN PSNRc = 22.99 dB	1st Iteration: dd = 6.52% 6.52% of all the pixels detected to be noisy PSNRr = 38.99 dB Gain = 16.00 dB	2nd Iteration: dd = 10.94% 4.41% of all the pixels detected to be noisy PSNRr = 37.30 dB Gain = 14.31 dB	3rd Iteration: dd = 14.08% 3.16% of all the pixels detected to be noisy PSNRr = 36.81 dB Gain = 13.82 dB
Key:	General	This iteration is not sufficient.	final TSBAMF results: This iteration is sufficient	This iteration is not useful

TABLE 5
SOME TSBAMF RESULTS FOR BABOON TEST IMAGE WITH RVIN

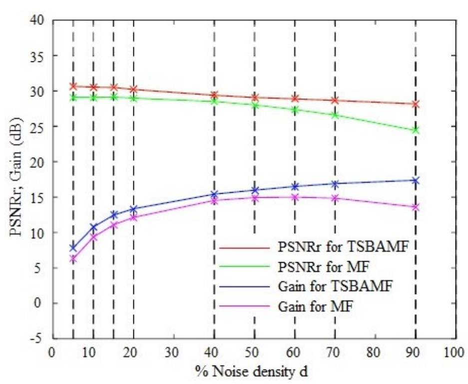
Baboon (Original)	Baboon with 90% RVIN PSNRc = 12.49 dB	1st Iteration: dd = 62.39% 62.39% of all the pixels detected to be noisy PSNRr = 20.09 dB Gain = 7.60 dB	2nd Iteration: dd = 85.50% 23.11% of all the pixels detected to be noisy PSNRr = 21.03 dB Gain = 8.54 dB	3rd Iteration: dd = 97.71% 12.21% of all the pixels detected to be noisy PSNRr = 21.07 dB Gain = 8.58 dB
				
	Baboon with 60% RVIN PSNRc = 14.06 dB	1st Iteration: dd = 53.67% 53.67% of all the pixels detected to be noisy PSNRr = 21.77 dB Gain = 7.72 dB	2nd Iteration: dd = 74.08% 20.41% of all the pixels detected to be noisy PSNRr = 21.95 dB Gain = 7.90 dB	3rd Iteration: dd = 85.54% 11.46% of all the pixels detected to be noisy PSNRr = 21.84 dB Gain = 7.78 dB
				
	Baboon with 10% RVIN PSNRc = 21.44 dB	1st Iteration: dd = 32.89% 32.89% of all the pixels detected to be noisy PSNRr = 23.47 dB Gain = 2.02 dB	2nd Iteration: dd = 51.91% 19.01% of all the pixels detected to be noisy PSNRr =23.02 dB Gain = 1.57 dB	3rd Iteration: dd = 63.34% 11.43% of all the pixels detected to be noisy PSNRr = 22.80 dB Gain = 1.35 dB
Key:	General	This iteration is not sufficient.	final TSBAMF results: This iteration is sufficient	This iteration is not useful



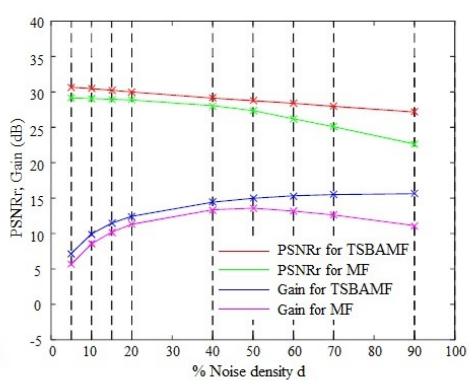
(a) Lena with SPN



(b) Lena with RVIN



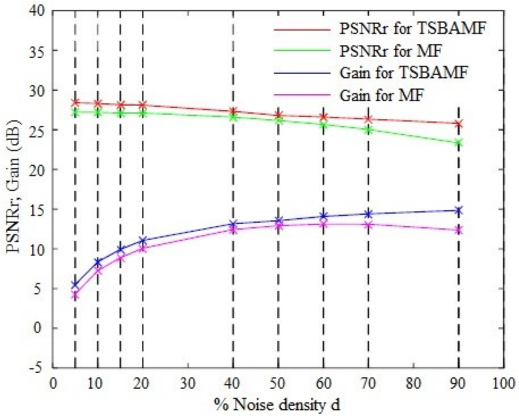
(c) Pepper with SPN



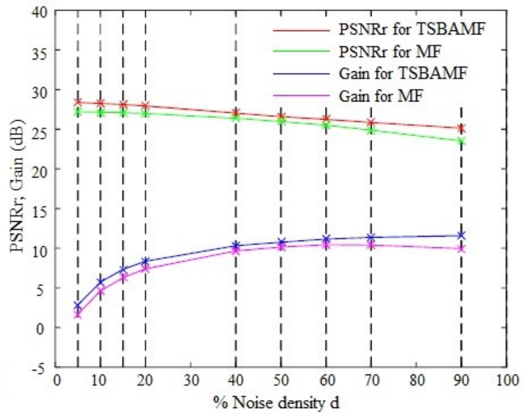
(d) Pepper with RVIN

Fig. 3 Comaprison of TSBAMF and MF [11] results for five test images with SPN and RVIN.

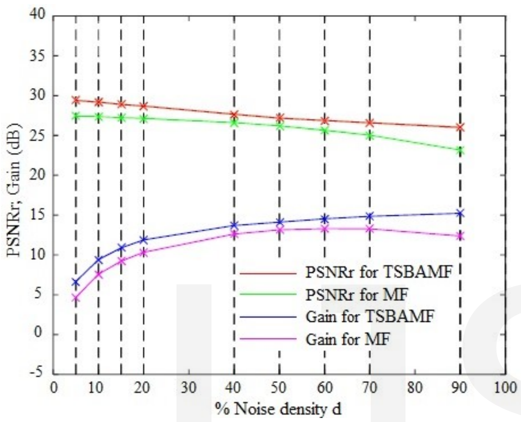
International Journal of Scientific & Engineering Research, Volume 9, Issue 5, May-2018
ISSN 2229-5518



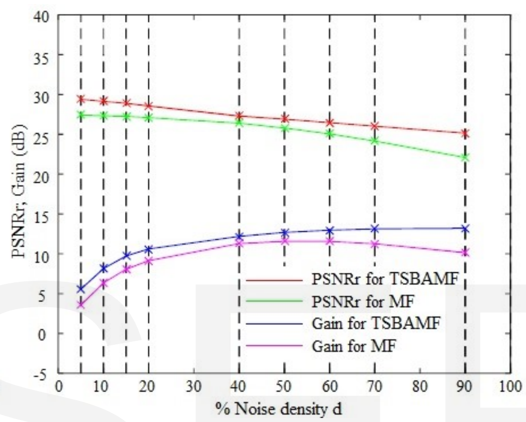
(e) House with SPN



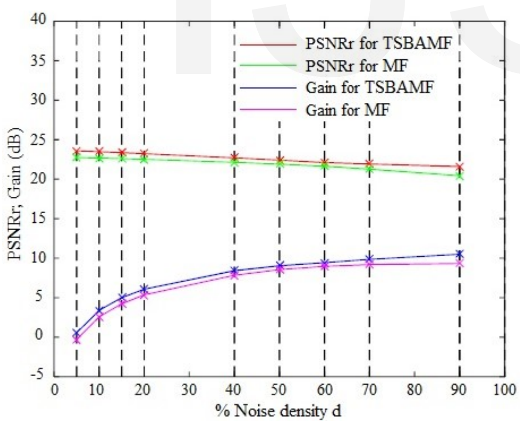
(f) House with RVIN



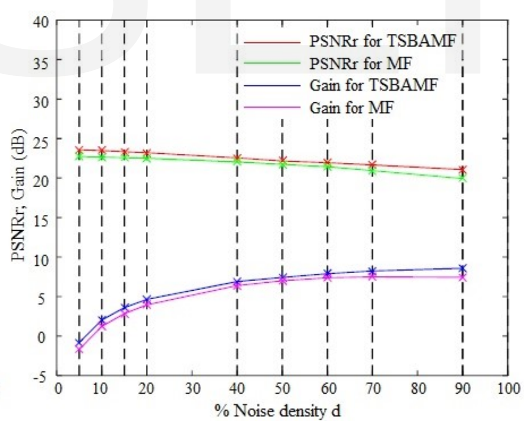
(g) Boat with SPN



(h) Boat with RVIN



(i) Baboon with SPN



(j) Baboon with RVIN

Fig. 3 (Continued) Comaprison of TSBAMF and MF [11] results for the five test images with SPN and RVIN.

International Journal of Scientific & Engineering Research, Volume 9, Issue 5, May-2018
ISSN 2229-5518

TABLE 6
COMPARISON OF SOME TSBAMF AND MF [11] RESULTS

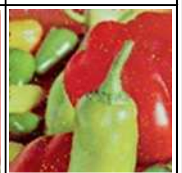
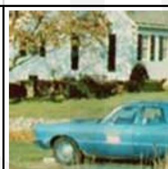
			
Pepper (Original)	Pepper with 90% SPN PSNRc = 10.79 dB	Output of TSBAMF PSNRr = 28.17 dB Gain = 17.38 dB	Output of MF PSNRr = 24.41 dB Gain = 13.62 dB
			
	Pepper with 90% RVIN PSNRc = 11.53 dB	Output of TSBAMF PSNRr = 27.15 dB Gain = 15.62 dB	Output of MF PSNRr = 22.64 dB Gain = 11.11 dB
			
House (Original)	House with 70% SPN PSNRc = 11.93 dB	Output of TSBAMF PSNRr = 26.33 dB Gain = 14.40 dB	Output of MF PSNRr = 25.03 dB Gain = 13.10 dB
			
	House with 70% RVIN PSNRc = 14.48 dB	Output of TSBAMF PSNRr = 25.85 dB Gain = 11.36 dB	Output of MF PSNRr = 24.88 dB Gain = 10.40 dB

TABLE 7
COMPARISON OF TSBAMF AND NAFSMF RESULTS

	NAFSMF Results with Impulse Noise [12]	TSBAMF Results with SPN (Table 2)	TSBAMF Results with RVIN (Table 3)
d %	PSNRr (dB)	PSNRr (dB)	PSNRr (dB)
40	29.58	34.12	33.68
50	28.80	33.43	32.83
60	27.61	32.83	32.09
70	25.93	32.24	31.47

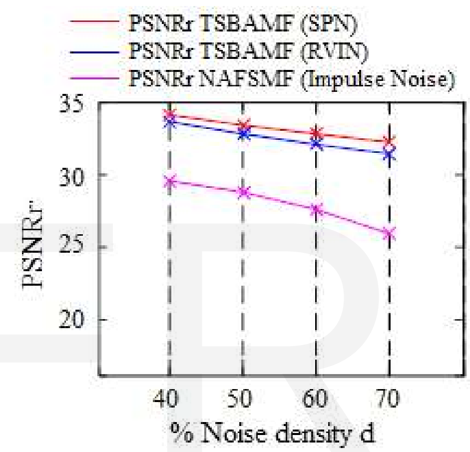


Fig. 4. Comparison of TSBAMF Results with NAFSMF [12] Results.

3.4 Comparison of TSBAMF Results with NAFSMF Results.

For Lena 512 by 512 test image, the final TSBAMF results (highlighted with blue colour) extracted for SPN and RVIN from Tables 2 and 3 respectively are compared with NAFSMF results for impulse noise extracted from Table 2 of [12] as shown in Table 7 and Fig. 4. TSBAMF is found to give higher PSNRr compared with NAFSMF. Thus, TSBAMF offer better restoration compared with NAFSMF. NAFSMF can restore corrupted image with noise density up to 60%. TSBAMF can restore corrupted image with noise density up to 90%.

4 CONCLUSION

Texture Synthesis Based Adaptive Median Filter (TSBAMF) has been developed. TSBAMF applies median filtering to only pixels detected to be noisy. Noise detection is based on Texture Synthesis approach.

Texture Synthesis Based Adaptive Median Filter (TSBAMF) is found to offer better image filtering/restoration and visual quality compared with Median Filter (MF) which applies median filtering to all pixels. Texture Synthesis Based Adaptive Median Filter (TSBAMF) is less complex and offer better image filtering/restoration compared with an existing adaptive median filter which is known as Noise Adaptive Fuzzy Switching Median Filter (NAFSMF).

Satisfactory filtering by Noise Adaptive Fuzzy Switching Median Filter (NAFSMF) and Median Filter (MF) is limited to corrupted images with noise densities up to 60%. Texture Synthesis Based Adaptive Median Filter (TSBAMF) can restore corrupted images with noise densities up to 90%.

REFERENCES

[1] B. Chanda and D.D. Majumer, *Digital Image Processing and Analysis.* India: Prentice-Hall, 2000.

[2] F.C. Tony and S. Jianhong, *Image Processing and Analysis: Variational, PDE, Wavelet, and Stochastic Methods.* Philadelphia: SIAM, 2005.

[3] C. Saxena and D. Kourav, "Noises and Image Denoising Techniques: A Brief Survey," *International Journal of Emerging Technology and Advanced Engineering,* vol. 4, no. 3, pp. 878-885, Mar. 2014.

[4] A.K. Das, "Review of Image Denoising Techniques," *International Journal of Emerging Technology and Advanced Engineering,* vol. 4, no. 8, pp. 519-522, Aug. 2014.

[5] K. Gupta and S.K. Gupta, "Image Denoising Techniques – A Review

paper," *International Journal of Innovative Technology and Exploring Engineering (IJITEE),* vol. 2, no. 4. pp. 6-9, Mar. 2013.

[6] P. Patidar, M. Gupta, S. Srivastava and A.K. Nagawat, "Image De-noising by Various Filters for Different Noise," *International Journal of Computer Applications,* vol. 9, no. 4, pp. 45-50, Nov. 2010.

[7] R.C. Gonzalez and R. E. Woods, *Digital Image Processing*. Massachusetts: Addison-Wesley, 2002.

[8] A.R. Zubair and O.A. Fakolujo, "Development of Statistics and Convolution as Tools for Image Noise Suppression: Statistical Performance Analysis of Spatial Filters," *IEEE Afr J. of Comp & ICTs,* vol. 6, no. 5, pp. 53-66, Dec. 2013.

[9] P. Chen and L. Chih-Yuan, "An efficient edge-preserving algorithm for removal of salt-and-pepper noise," *IEEE Signal Processing Letters,* vol. 15, pp. 833-836, Dec. 2008.

[10] A.S. Ali and M. Hong, "Robust Detection Technique for Removing Random-Valued Impulse Noise," *Proc. IEEE Int. Symp. Signal Processing and Information Technology,* pp. 575-577, 2007.

[11] A. R. Zubair and H. O. Busari. "Robustness of Median Filter for suppression of Salt and Pepper Noise (SPN) and Random Valued Impulse Noise (RVIN)," *International Journal of Image Processing (IJIP),* vol. 12, no. 1, pp. 12-27, Apr. 2018.

[12] S. Govindan and S. Saravanakumar, "Removal of Impulse Noise using Noise Adaptive Fuzzy Switching Median Filter," *Proc. International Conference on VLSI, Communication & Instrumentation (ICVCI),* pp 39-42, 2011.

[13] M. Ashikhmin, "Synthesizing Natural Textures," *Proc. ACM Symp. Interactive 3D Graphics, Research Triangle Park, NorthCarolina,* pp. 217-226, Mar. 2001.

[14] H. Igehy and L. Pereira, "Image replacement through texture synthesis," *Proc. International Conference on Image Processing,* vol. 3, pp. 186–189, Oct 1997.

[15] A.A. Efros and T. K. Leung, "Texture synthesis by non-parametric sampling," *Proc. IEEE International Conference Computer Vision (ICCV'99), Corfu, Greece,* pp. 1033-1038, Sept. 1999.

[16] USC-SIPI Image Database, "Standard Test Images.'' http://sipi.usc.edu/database/index.html [Oct. 5, 2006].

[17] A.R. Zubair, "A Non-Iterative Automated Mechanism for Image Inpainting," *IEEE Afr J. of Comp & ICTs,* vol. 5, no. 4, pp. 1-8, 2012.

[18] S.S.O. Choy, Y. Chan and W. Siu ''An Improved Quantitative Measure of Image Restoration Quality,'' *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'96),* vol. III, pp. 1613-1616, 1996.