

## An Optimised Fuzzy Approach to Remove Mixed Noise from Images

Sweety Deswal<sup>1</sup>, Surbhi Singhanian<sup>2</sup>, Shailender Gupta<sup>3</sup> and Pranjal Garg<sup>4</sup>

YMCA University of science and technology,  
Faridabad, India

Sweetydeswal92@gmail.com<sup>1</sup> Surbhidec24@gmail.com<sup>2</sup>  
Shailender81@gmail.com<sup>3</sup>, prnjlg0@gmail.com<sup>4</sup>

### Abstract

*Mixed noises can be defined as a combination of different types of noises acting on a single carrier. There has been a mention of various mechanisms used to restore images corrupted with mixed noise in the past. This paper proposes a simple method based on fuzzy set theory and Bilateral Filter to remove mixed noises and compares it with previously mentioned techniques such as: Vector Median Filter (VMF), Vector Direction Filter (VDF), Fuzzy Peer Group Averaging (FPGA), Fuzzy Vector Median Filter (FVMF), Bilateral Filter (BF), Adaptive Bilateral Filter (ABF), Switching Bilateral Filter (SBF), Joint Bilateral Filter (JBF), and Trilateral Filter (TF) on the basis of performance metrics such as Peak Signal to Noise Ratio (PSNR), Mean Absolute Error (MAE), Mean Square Error (MSE) and Normalised Colour Difference (NCD). For the purpose of a detailed analysis, the performance of each method is evaluated by varying the image size and the noise density by implementing them in MATLAB-09. The mixed noise used in this paper is a combination of three noise i.e. poisson, impulse and Gaussian noise. The simulation and result shows that the proposed method provides better PSNR and hence better image quality than almost all the methods mentioned above.*

**Keywords:** Noise, Filter, Gaussian, Poisson, Impulse, PSNR, MSE, MAE.

### 1. Introduction.

Noise [4] is a commonly used term which describes visual distortion in images. It may be caused due to film grain in case of digital cameras acquisition or electronic transmission discrepancy as observed in television broadcasting. Several researchers [6-8] have proposed mechanism to remove single noise from images but only few proposals are available in literature about how to remove mixed noise [14-18] i.e. mixture of two or more noises from images. Few popular noises are impulse [11-13], poisson [27] and Gaussian noise [27]. The impulse noise is created due to transmission faults. For instance, in case of satellite transmission over long distances impulse noise is prominent. Also known as the “salt and pepper noise”, it replaces the pixels with a zero or maximum pixel value leaving black and white spots on the image.

Poisson noise or photo shot noise is caused by random variation of photons, which cause more photons to enter one sensor than the other. In real world photography, if enough images are taken, it will be seen that the deviation in intensity found for each image follows the well-known poisson distribution. In effect, we can't be sure that the intensity measured in a particular image represents the "true" intensity as it is obvious that this value will deviate from the average. It is this deviation which is considered to be the noise associated with the image. As the deviation is known to follow a Poisson distribution, we know that the likely deviation will be plus or minus the square root of the signal intensity measured.

Gaussian noise is caused during acquisition process *e.g.* electronic circuit noise. In case of telecommunications and computer networking, communication channels can be affected by wideband Gaussian noise coming from many natural sources, such as thermal vibrations of atoms in conductors.

In this paper we consider all these noises as mixed ones and de-noising is done using Fuzzy logic and Bilateral Filter with the previous ones in literature. The techniques used for comparison are as follows:

- **Average Median Filter (AMF)** is the simplest type of filter to remove impulse noise from image where each pixel is replaced by arithmetic mean of neighbouring pixels. It is generally assumed that pixel values vary slowly over space, so neighbouring pixels are likely to have similar values, and it is therefore appropriate to average them together.
- **Vector Median Filter (VMF)** is an extension of the median filter [2] to multivariate data. For an observation window  $\Omega = \{x_1, x_2, \dots, x_N\}$ , the output of the vector median filter is defined as

$$x_{VM} = \arg \min \sum_{i=1}^N \|x - x_i\|_2$$

Where  $\|\cdot\|$  denotes the  $L_p$  norm. The impulse response of this type of filter is zero; therefore, it is good for removal of impulse noise.

- **Vector Directional Filter (VDF)** is a vector directional filter (VDF) for directional [3][25-26] processing, which is a generalized Basic Vector Directional Filter (BVDF). For an observation window, the output of the BVDF is defined as:

$$x_{BVD} = \arg \min \sum_{i=1}^N A(x - x_i)$$

Where  $A(x, x_i)$  denotes the angle between  $x$  and  $x_i$ .

- **Fuzzy Vector Median Filter (FVMF)** utilizes the techniques of fuzzy set theory, Y. Shen and K.E. Barner proposed fuzzy vector median (FVM) based surface smoothing [1][19-20] which utilize the information regarding the spread of samples in image pixels. In this technique one membership function  $\mu$  is used to calculate degree of pixel as below:

$$\mu(a, b) = e^{-\frac{(a-b)^2}{\alpha^2}}$$

Where  $\alpha$  control the spread of membership function. Then output is calculated as:

$$F_{FVMF} = \frac{\sum_{i=1}^N F_i \mu(F_0 F_i)}{\sum_{i=1}^N \mu(F_0 F_i)}$$

where  $F_0$  is the central pixel and  $F_i$  is the current pixel in neighbourhood.

- **Fuzzy Peer Group Averaging (FPGA)** is a filtering method [12] where the fuzzy peer group of each image pixel is determined by means of a novel fuzzy logic-based procedure. Then output is calculated by weighting averaging operation as below where weighting coefficients for each pixel vector is its membership degree.  $FP_m^{F_0}$  to peer group's  $m^f$ .

$$F_{out} = \frac{\sum_{i=1}^{mf} F_i \times FP_m^{F_0}(F_i)}{\sum_{i=1}^{mf} [F_i FP_m^{F_0}(F_i)]}$$

- **Bilateral filter:** Bilateral Filter [28, 29, 30] was first proposed by C. Tomasi, R. Manduchi in the year 1998. It is basically a non-linear, edge-preserving and Gaussian noise reducing filter used for gray and color images. Thus mathematically at a pixel location  $(x, y)$  the output  $I(x, y)$  of the bilateral filter is calculated as follows

$$I'(x, y) = \sum_{y \in N(x)} y_{eN(x)} e^{-\frac{\|y-x\|^2}{2\sigma_d^2}} e^{-\frac{\|I(y)-I(x)\|^2}{2\sigma_r^2}} I(x, y)$$

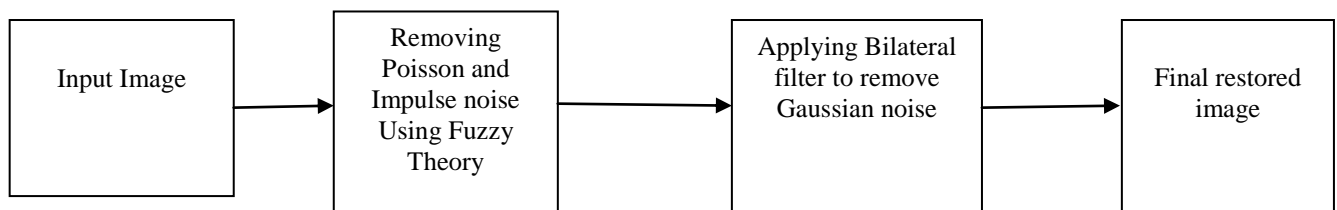
Where  $\sigma_d$  and  $\sigma_r$  are parameters controlling the fall-off of weights in spatial and intensity domains, respectively.  $N(x)$  is a spatial neighborhood of pixel  $I(x)$ .  $\sigma_d$  is the geometric spread parameter that is to be chosen based on the amount of the low pass filtering required.

- **Adaptive Bilateral Filter:** An Adaptive Bilateral filter (ABF)[31] is proposed by Zhang and Allebach in the year 2008 which not only smoothes the image but also sharpens the image by increasing the slope of the edges. ABF adds an offset ( $\phi$ ) to the existing bilateral filter in order to sharpen the edges. ABF adds an offset ( $\phi$ ) to the existing bilateral filter in order to sharpen the edges.
- **Switching Bilateral Filter:** SBF works on mixed noise (Gaussian +impulse) techniques. It is based upon the “detect and replace” methodology and for detection purpose, we used a noise detector [33, 35, 36] in the switching filtering technique. The absolute difference between current pixel and reference median is calculated. Depending upon the value of absolute difference, we can determine whether the pixel is noisy or not. To determine the value of reference median, we define an approach called Sorted Quadrant Median Vector (SQMV).
- **Joint Bilateral Filter:** A new technique [37, 38] was proposed by O.U.NirmalJith and R. Venkatesh Babu in the year 2014 to achieve high quality image from two images. It performs a Non Local Means (NLM) approach for de-noising of images. It works on images corrupted with Gaussian noise.
- **Trilateral Filter[13]:** The technique is used to remove mixed noise (Gaussian and impulse noise) from images. Instead of making use of ‘detect and replace methodology’, it adds a ROAD statistics to existing bilateral filter that helps to remove two types of noises. This provides better results as compared to Switching Bilateral Filter.

All these techniques are implemented in MATLAB-09. The rest of the paper is organised as follows: Section 2 gives the proposal. Section 3 provides the optimisation algorithm used to optimization the parameters used in our fuzzy based method. Section 4 provides the simulation set up parameters, performance metrics used for comparison purpose. Section 5 compares the results of our proposed technique with the previous ones followed by conclusion and references.

## 2. The Proposal

The block diagram for proposed technique is shown in Figure. 1. We use fuzzy logic theory to remove all the noises mentioned above and at last remove traces of Gaussian noise left by using Bilateral filter.

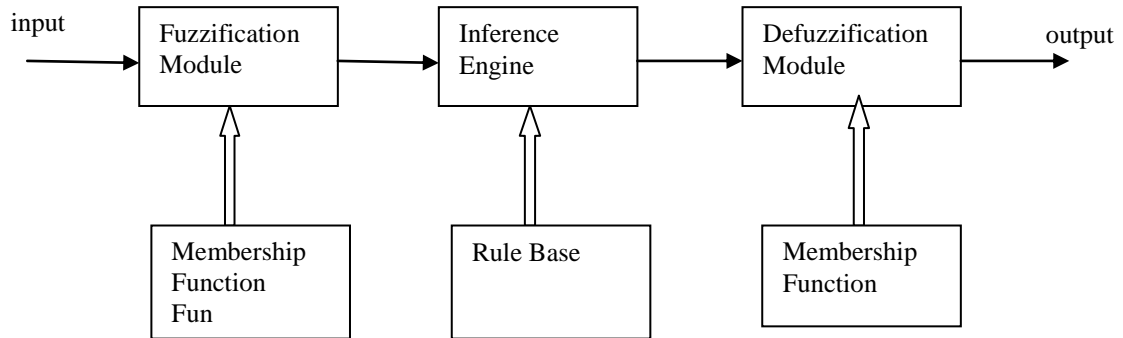


**Figure 1. Proposed Block Diagram**

Now we would like to give a brief description about Fuzzy theory for better understanding of our proposal.

## 2.1 Basics of Fuzzy Theory.

It is a new science that enables its user to arrive to a definite conclusion despite the fact that the inputs provided are vague, imprecise and ambiguous. Unlike Classical Set Theory which deals with crisp sets, Fuzzy Set Theory provides means to convert non-numeric Linguistic Variables into an exact outcome. This science has been used for the control purpose of robotic equipments, image recognition, architectural space analysis and cluster analysis. (This Science has been used everywhere from control theory to artificial intelligence.) Basic setup of a Fuzzy Knowledge Base Controller is as depicted follows (see Figure. 2).



**Figure 2. Fuzzy Knowledge Base Controller**

The basic functioning of each block is as follows:

- **Fuzzification Module-** is responsible for the fuzzification or converting the crisp sets into fuzzy sets. It uses a pre-defined membership function for the process of conversion.
- **Inference Engine-** uses if-then rules defined in the rule base to analyse the fuzzy sets and provide corresponding outputs.
- **Defuzzification Module-** is responsible for converting the fuzzy sets back to their crisp form. In our proposal we used Centre of Gravity method to convert the fuzzy value to crisp value.

The final aim of this method is to use fuzzy logic to assign weights to pixels in the window (W) around the pixel under analysis (*i.e.*  $F_i$ ). Using the weighted average or Centre of Gravity method this pixel is then replaced with a new Pixel value ( $F'_i$ ) which is given by the formula:

$$F'_i = \frac{\sum_{i=0}^m w_i F_i}{\sum_{i=0}^m w_i} \quad (1)$$

Using this method, for each pixel an appropriate replacement is found keeping in mind the noisiness and similarity of its neighbouring pixels within a window surrounding the pixel. The value of 'm' in equation (1) is set as 3 on experimental verification. This noisiness is a property of the impulse noise and similarity aids us to remove the Poisson noise. Now we discuss how each block is used in our proposed method.

### a. **Fuzzification Module:**

This module takes two inputs named as Noisiness and Similarity discussed below:

#### • **Noisiness:**

Noisiness of a pixel ( $F_i$ ) is calculated with respect to pixels in the  $n \times n$  window around it. Then we assign a degree of certainty  $\alpha(F_i)$  for the vague statement “( $F_i$ ) is noisy”. Let

the pixel be  $(F_i)$  and the window pixels be  $F_j$  with  $j$  varying from 0 to  $n^2-1$ . The metric  $L_\infty$  is calculated for every  $F_j$  in the window using the function:

$$L_\infty(F_i, F_j) = \max\{|F_i^R - F_j^R|, |F_i^G - F_j^G|, |F_i^B - F_j^B|\}$$

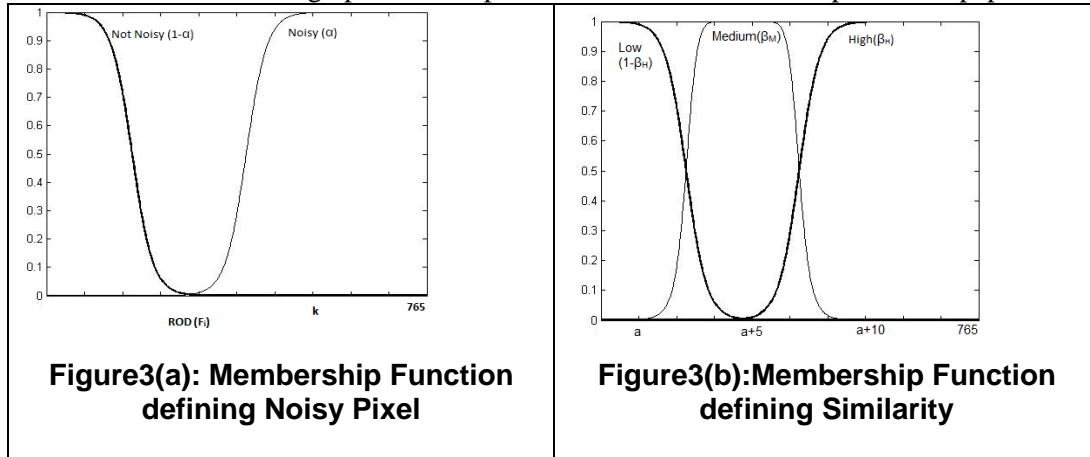
These values of all pixels are sorted and stored as a matrix  $l_1(k)$  where  $k$  varies from 0 to  $n^2-1$  and  $l_1(0)$  holds the minimum value of  $L_\infty$  and  $l_1(0) < l_1(1) < l_1(2) < \dots < l_1(n^2-1)$ . The least  $s+1$  values of  $L_\infty$  are used in calculating the statistics  $ROD_s$  given by:

$$ROD_s(F_i) = \sum_{k=0}^s l(k)$$

$ROD_s$  takes values in the interval  $[0, 255s]$ . The value of 's' is set such that only close pixels with least noise are involved in the calculation of the  $ROD_s$  factor. It has been taken to be 3 in this proposal after experimental analysis. Low value of  $ROD_s$  indicates that the pixel value of the neighbours of  $F_i$  is close to its own pixel value thus implying that  $F_i$  is expected to be noise-free. But, higher value of  $ROD_s$  indicates a noisier pixel. The value of s is taken as 3. The membership function used to define noisy pixel (see Figure. 2(a)) is given as:

$$\alpha(F_i) = \frac{1}{1 + e^{-0.1077(x-k)}}$$

Where k is selected using optimisation process discussed in the later part of the paper.



#### • Similarity

This function defines the degree of closeness  $\beta(F_i, F_j)$  between the value of the pixel in question ( $F_i$ ) and the value of pixels around it ( $F_j$ ;  $j \in (0, n^2 - 1)$ ). The similarity of window pixels with  $F_i$  is labelled as “low”, “medium” or “high” represented by  $\beta_L, \beta_M, \beta_H$  and defined by the metric  $L_1$  given by:

$$L_1(F_i, F_j) = |F_i - F_j|$$

It should be noted here that the  $L_1$  parameter is calculated individually for each colour plane. The window pixels ( $F_j$ ) are sorted into a matrix  $l_2(k)$  where  $l_2(0)$  contains the least value of  $L_1$  and  $l_2(0) \leq l_2(1) \leq l_2(2) \leq l_2(3) \dots \leq l_2(n^2 - 1)$ . First  $m+1$  pixels of this matrix are used for further inferring process. To assign a degree of closeness membership function shown in Figure. 2(b) is used and mathematically stated as:

$$\beta_H(F_i, F_j) = \frac{1}{1 + e^{0.3662(x-a)}}$$

$$\beta_M(F_i, F_j) = \frac{1}{1 + \left| \frac{x-(a+5)}{15} \right|^{2.5}}$$

The parameter 'a' is suitably chosen using the optimisation process discussed in the later part of this paper. The next subsection discusses the inference engine.

#### a. Inference Engine

This module depends on if-then conditions that make the rule base of the fuzzy system. Depending on these conditions this module analyses the degree of certainty of input conditions that is noise and similarity. In addition to it also assigns weight to be small, medium or large as shown in Table 1. The weight assigned is a number between 0 and 1. The rules are as summarised below (see Table 1):

**Table 1. Rule Base**

Is $F_i$ noisy?	Is $F_j$ noisy?	Similarity b/w $F_i$ & $F_j$	Weight
Yes	No	Medium	Medium
Yes	No	Low	Large
No	No	High	Large
Yes	Yes	Low	Small
Yes	Yes	Medium	Small
Yes	Yes	High	Small
No	Yes	Low	Small
No	Yes	Medium	Small
No	Yes	High	Small
Yes	No	High	Small
No	No	Medium	Small
No	No	Low	Small

These rules can be summarised as:

- Noisy pixels are assigned small weights.
- Noise free pixels are assigned high values of weight if similarity with central pixel is moderate or high.
- Noise free pixels are assigned high values of weight if the central pixel is noise.

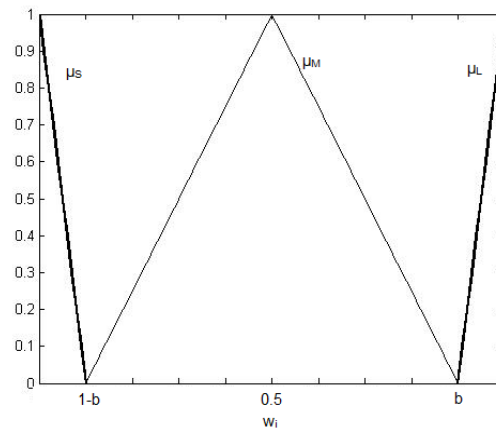
#### b. De-fuzzification Module

This module converts fuzzy sets back into crisp sets. There are various methods available in literature for de-fuzzification such as maximum principle, centroid value method, weighted average method, mean maximum membership method *etc.* In this paper, we have used the weighted average or Centre of Gravity method to calculate the de-fuzzified value as it gives the best results. The membership function shown in Figure. 3 is the degree of certainty of each pixel's weight being "small", "medium" or "large" as assigned by the inference engine lead us to calculate the weight associated with each window pixel. The membership equations are as follows:

$$\mu_M(w_i) = \begin{cases} \frac{(2w_i - 1)}{(2b - 1)} + 1, & 1 - b < w_i \leq 0.5 \\ \frac{(1 - 2w_i)}{(2b - 1)} + 1 & 0.5 < w_i \leq b \\ 0 & \text{otherwise} \end{cases}$$

$$\mu_S(w_i) = \begin{cases} \frac{w_i}{(b-1)} + 1, & 0 < w_i \leq 1 - b \\ 0, & \text{otherwise} \end{cases}$$

where the value of 'b' is found using optimisation process.



**Figure 4. Membership Function Defining Output**

The final de-fuzzified value can be found out by using the de-fuzzification process. On replacement of all  $F_i$  with  $F'_i$  using equation (1), the procedure yields a de-noised image. An important point to be kept in mind is that the image has to be padded with zeros before hand in order to perform all the steps on the edge pixel smoothly.

## 2.2 Bilateral Filter.

After applying above parameters to input image, impulse and poisson noise has been removed. Then we pass the image through a bilateral filter to remove any Gaussian noise left in the image. Mathematically at a pixel location (x,y) the output  $I(x,y)$  of the bilateral filter is calculated as follows:

$$I'(x,y) = \sum_{y \in N(x)} y_{\in N(x)} e^{\frac{-||y-x||^2}{2\sigma_d^2}} e^{\frac{-||I(y)-I(x)||^2}{2\sigma_r^2}} I(x,y)$$

Where  $\sigma_d$  and  $\sigma_r$  are parameters controlling the fall-off of weights in spatial and intensity domains, respectively.  $N(x)$  is a spatial neighborhood of pixel  $I(x)$ .  $\sigma_d$  is the geometric spread parameter that is to be chosen based on the amount of the low pass filtering required. A large value of  $\sigma_d$  means it combines values from more distance in an image. Similarly  $\sigma_r$  is the parametric spread that is set to achieve the desired amount of combination of pixel values. Finally we get our restored image as resultant image.

## 3. Optimization of Parameters

For the purpose of obtaining the best possible value of PSNR in the de-noised images, we performed the optimisation of the variables  $a$ ,  $b$  and  $k$ . keeping one of the variable constant at a time and by using regression the most optimised value was obtained. The algorithm to optimise the values is as follows:

```

a=1;
b=0.9;
For k=1 to 501
    Perform denoising
    Calculate PSNR
    If PSNR (k) ≤ PSNR(k-50)
        Break;
    Else

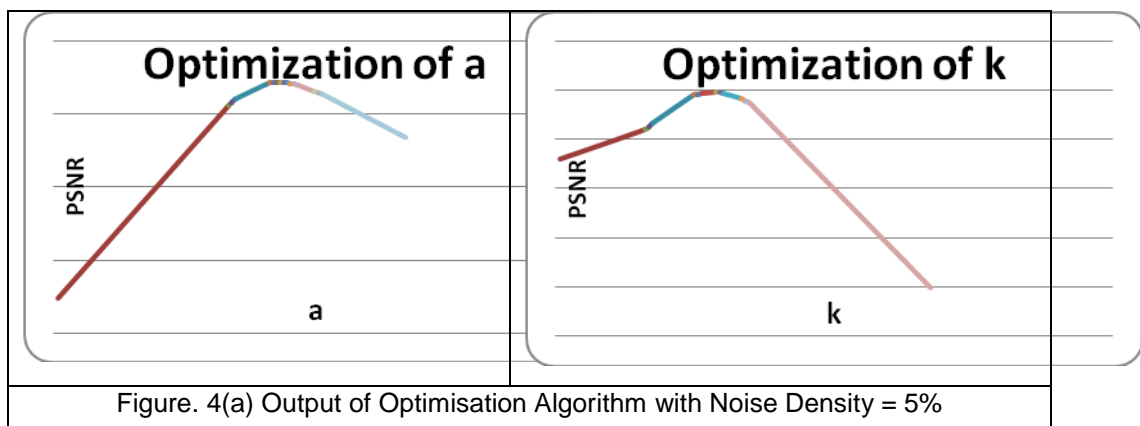
```

```

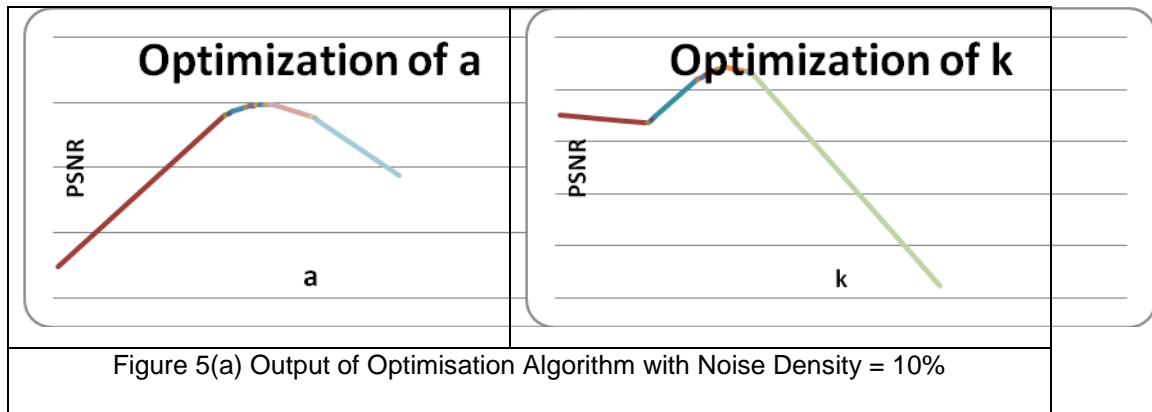
        k=k+50;
    End
End
If (k<100)
    k_left=k-50
    k_right=k
Else
    k_left=k-100
    k_right=k
End
For i=1:20
    k_max=floor ((k_left + k_right)/2)
    Calculate PSNR (k_max-1), PSNR (k_max+1)
    If [PSNR (k_max-1) > PSNR (k_max)] and [PSNR (k_max+1) < PSNR (k_max)]
        k_right=k_max-1
    End
    If [PSNR (k_max-1) < PSNR (k_max)] and [PSNR (k_max+1) > PSNR (k_max)]
        k_left=k_max+1
    End
    If [PSNR (k_max-1) < PSNR (k_max)] and [PSNR (k_max+1) < PSNR (k_max)]
        Display: k_max is the optimised value
        Break
    End
End
End

```

This algorithm yields best possible value for  $k$ . After this, the process is repeated for finding the most optimum value of  $a$ . Figure. 4 shows the output of the algorithm for image size  $128 \times 128$  and  $256 \times 256$  with noise density 5%, 10%, 15% and 20%. Similar graphs were obtained for other noise density and image sizes. Table 2 shows the optimum values of parameters obtained using the above algorithm for other noise densities. It should be noted here that  $b=0.9$ , this value has been found to yield best results when the process was run from 0.6 to 0.9 on a step of 0.1. The next section discusses the experimental setup parameters and performance metrics chosen.







**Figure 5: Output of Optimisation Algorithm**

**Table 2. Optimised Parameters**

Image Size	Noise Density	Optimised value of k	Optimised value of a
128x128	5	65	44
	10	62	44
	15	56	1
	20	54	1
256x256	5	52	41
	10	47	1
	15	45	1
	20	47	1

## 4.Experimental Setup

### 4.1Performance Metrics

Peak Signal to noise Ratio- is the measure of maximum error. It is used to express the ratio [4-5] of maximum possible power of image (signal) and the power of the noise that affects the quality of its demonstration. It is represented as:

$$PSNR=10\log_{10}\left(\frac{MAX^2}{MSE}\right)$$

$MAX_1$  is the maximum possible pixel value of the coloured image. It is equal to 255 for 8 bit represented image.

Mean square error- is the total squared error between the denoised image and the true uncorrupted image. This enables us to compare methods more precisely by analysing results under same conditions like image size noise, *etc.* It is mathematically stated as:

$$MSE=\frac{1}{M \times N \times 3} \sum_{c=1}^3 \sum_{y=1}^N \sum_{x=1}^M [F^c(x,y) - F^{c\sim}(x,y)]^2$$

Where  $m \times n$  is the image size.

Mean absolute error- is the absolute error between the original image and the denoised image obtained after denoising the image using one of the filters. It is used to measure the closeness of the denoised pixel to its original value before corruption. It is given by:

$$MAE=\frac{1}{M \times N \times 3} \sum_{c=1}^3 \sum_{y=1}^N \sum_{x=1}^M Abs [F^c(x,y) - F^{c\sim}(x,y)]^2$$

- Image Quality- The original and the denoised images were compared by placing them side by side to examine the variance in degradation of image quality in each method. This was tested on the images of varying sizes.
- Normalised Colour Difference (NCD)- is used to measure the degradation in colour quality in colour images since it approaches the human perception. It is defined as below:

$$NCD_{lab} = \frac{\sum_{i=1}^N \sum_{t=1}^M \Delta E_{lab}}{\sum_{i=1}^N \sum_{t=1}^M E_{lab}}$$

Where M, N are the image dimensions.

#### 4.2. Simulation Setup

Algorithms were developed in MATLAB-09 to simulate the methods for filtering an image consisting of dual noise. The value of variance was varied and hence, a comparative result generated. The setup parameters are shown in Table 3 as follows:

**Table 3. Setup Parameters**

Component	Parameter	Value of parameter
Image	Image Size	128x128( lena and academy) 256x256
	Type	RGB
Noise	Type	Single noise (Gaussian, Impulse, Poisson) Mixed noise (Gaussian + Impulse+ Poisson) Three noise (Gaussian + Impulse + Poisson)
	Variance of noise density	0.05 0.1 0.15 0.2
	Type	i3-64 bit
	RAM	4 GB
Processor	Speed	1.70 GHz
	Software	MATLAB-09














**Table 4. Parameters for Different Filters Used**

Bilateral Filter			Adaptive Bilateral Filter		
Symbol	Parameter	Value	Symbol	Parameter	Value
W	Window size	5	W	Window	3
$\sigma_d$	Spatial domain standard deviation	3	$\sigma_d$	Spatial domain filter	1.0
$\sigma_r$	Intensity domain standard deviation	10	$\sigma_r$	Intensity range filter	20
N	Gaussian noise intensity	0.03			
Switching Bilateral Filter			Trilateral		
Symbol	Parameter	Value	Symbol	Parameter	Value
W	Window size	5	W	Window size	5x5
$r_o$	Reference median	40	$\sigma_s$	Spatial domain standard deviation	5
N	Salt & pepper noise	0.2	$\sigma_R$	Intensity domain standard deviation	10

Tk1	Threshold 1	[25,30]	$\sigma_I$	Approx threshold	[25,55]
Tk2	Threshold 2	[5,10,15]	$\sigma_J$	Controls the shape of the function	[30,80]
$\sigma_s$		3,1			
$\sigma_r$		[30,50]			
Proposed method					
Symbol	Value				
a	1				
b	0.9				
k	1 to 501				
$w_i$	0 to 1				

## 5. Results



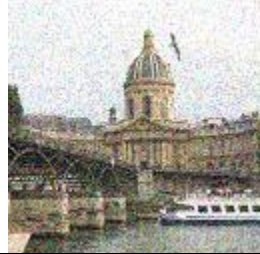
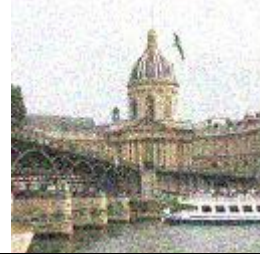



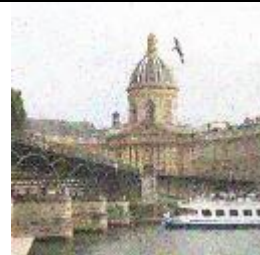
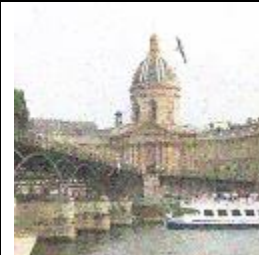
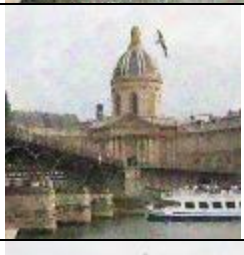
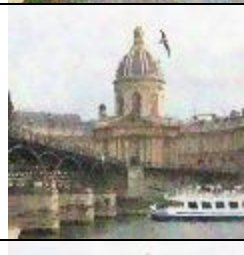
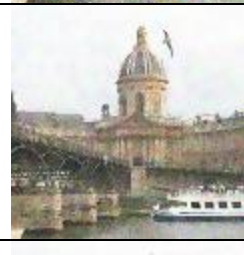
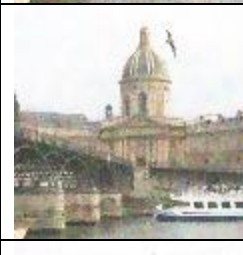





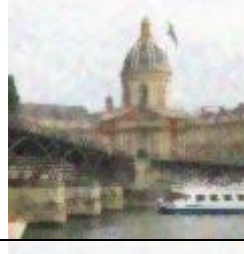
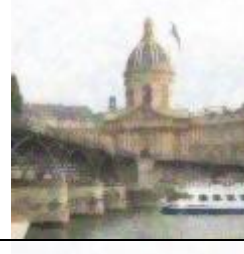





This section compares the proposed method with previously mentioned methods in the literature to remove mixed noise and proposes a new method to remove combination of three type of noise from images. The table contrasts these methods quantitatively and the image put side by side make a quantitative comparison.

Original image (lena 128.png)	Gaussian noise 5%	10%	15%	20%
				
BF				
ABF				



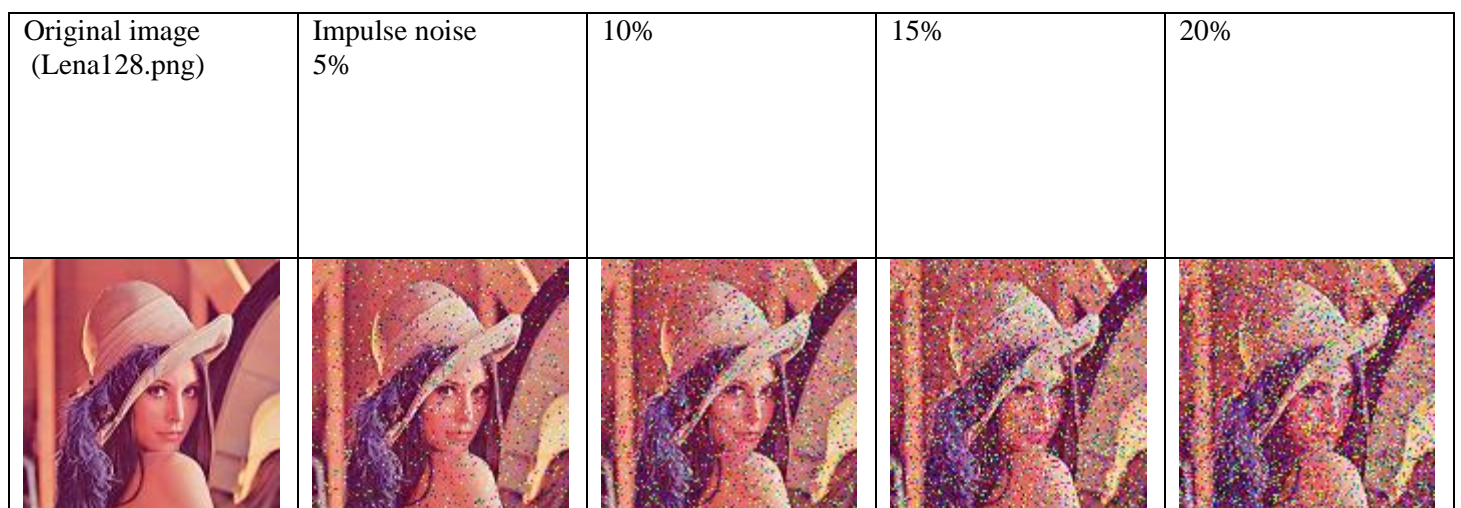
JBF				
PGA				
FVMF				
FBF				
SBF				
TF				
Proposed method				








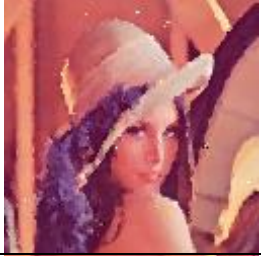
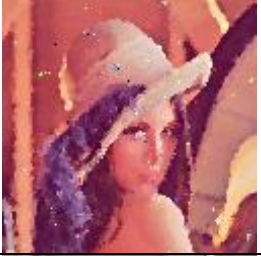
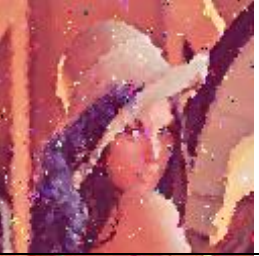
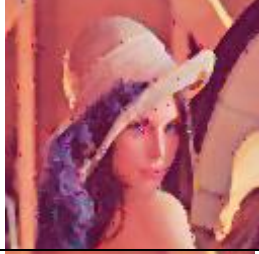
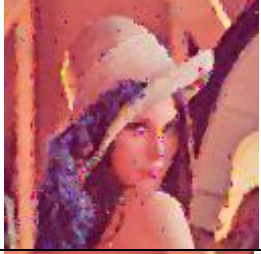
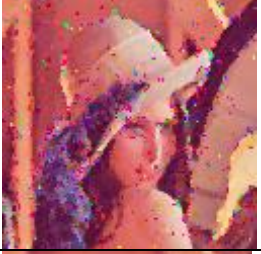
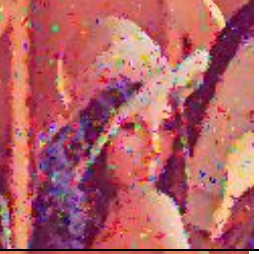
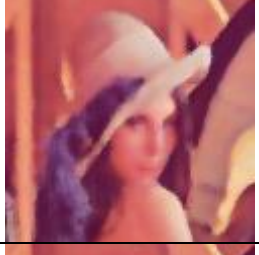
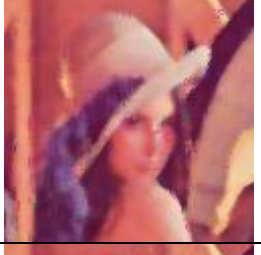
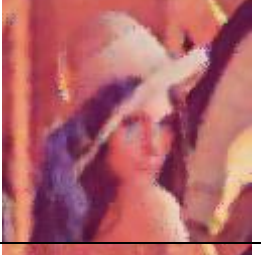
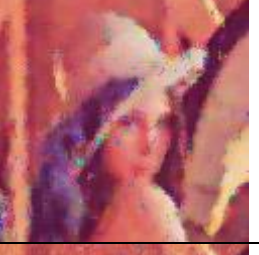

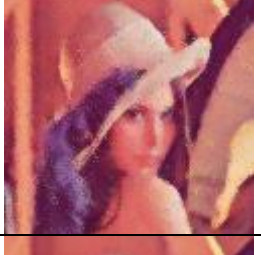
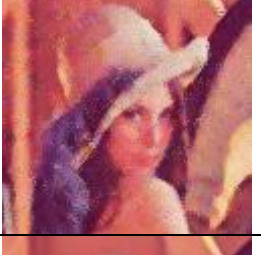

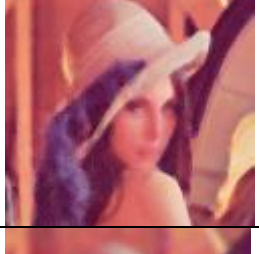
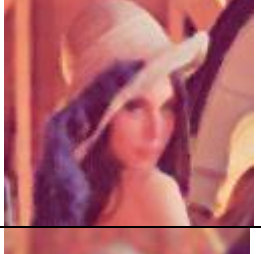
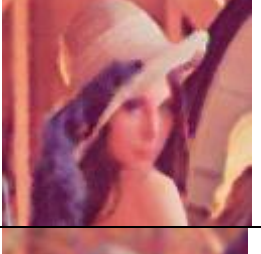
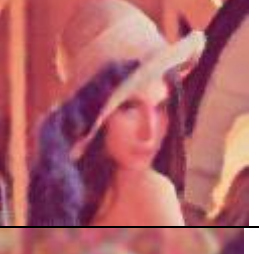
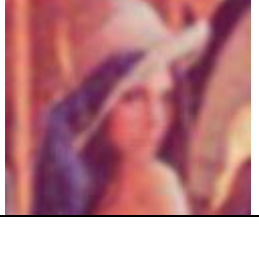
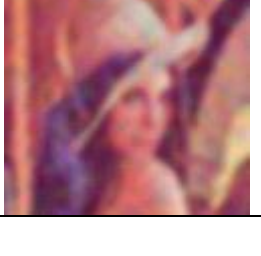
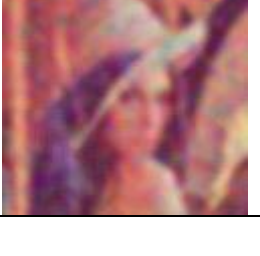
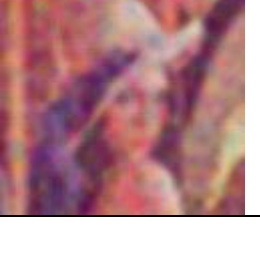
Original image Academy128.png	Gaussian noise 5%	10%	15%	20%
				
Filtered by BF				
Filtered by ABF				
Filtered by JBF				
PGA				
FVMF				






**Figure 6. Comparison for Gaussian Noise Removal Techniques:**





AMF				
VDF				
VMF				
FVMF				
PGA				
SBF				
FBF				



TF				
Proposed				
IMAGE 2 Academy128.png	Impulse noise 5%	10%	15%	20%
				
AMF				
VDF				
VMF				



















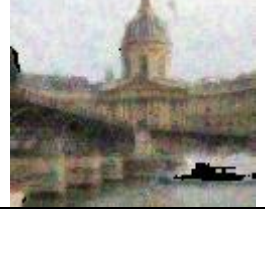
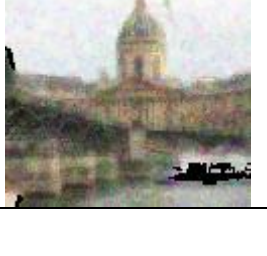

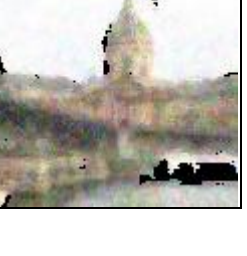




**Figure 7. Comparison of Impulse Noise Removal Techniques**

original image	Mixed noise Impulse=5% Gaussian=5	Impulse=10% Gaussian=10	Impulse=15% Gaussian=15	Impulse=20% Gaussian=20
				
PGA				
FVMF				
FBF				
SBF				
TF				




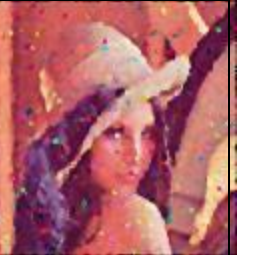





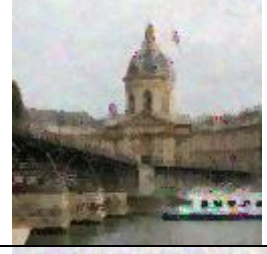








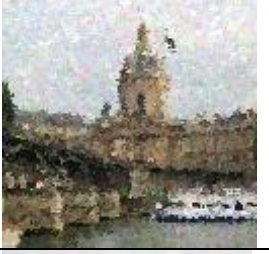








Proposed				
				
PGA				
FVMF				
FBF				
SBF				
TF				

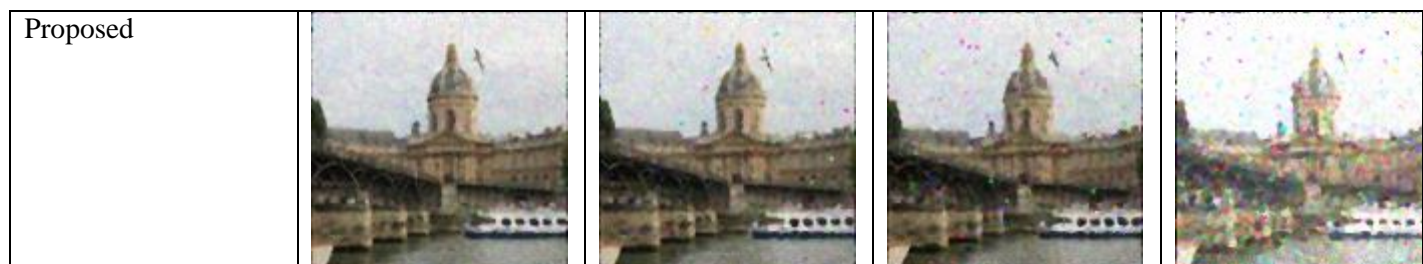






Proposed				
Original image (academy128.png)	Poison and (salt and pepper 5%)	10%	15%	20%
				
FPGA				
VMF				
VDF				
FVMF				





**Figure 9. Comparison of Mixed Noise ( Poisson and Salt & Pepper Noise)**



**Figure 10. Comparison for Removal of Mixed Noise (Impulse+Poisson+Gaussian) Proposed Technique:**

## 5.2 Impact on PSNR, MSE, MAE and NCD

Table 5-18 shows the PSNR, MSE, MAE for different noise density (5%, 10%, 15% and 20%) and different image sizes(128 × 128 and 256 × 256). The following inference can be drawn:

It can be concluded from that the proposed method provides highest value of PSNR and lowest value of NCD in all the cases.

On increasing the impulse noise factor while keeping the image size constant, PSNR decreases, MSE and MAE increases. This is for the reason that the ratio of image size to noise density decreases with increase in image size and constant noise, therefore the output image is lesser de-noised.

On increasing the image size with constant impulse noise density, PSNR increases, MSE and MAE decreases. This is because the ratio of image size to noise density increases with increasing image size and constant noise, therefore the output image is better de-noised.

### (1) For lena image (128 x 128):

**Table 5. For Different Kinds of Noises:**

Gaussian noise:						Impulse noise:					
	$\sigma$	5	10	15	20		$\sigma$	5	10	15	20
PGA	PSNR	58.0726	55.3988	53.5967	52.4672	AMF	PSNR	55.7429	54.9812	53.2522	52.7084
	MSE	0.1013	0.1876	0.2841	0.3684		MSE	0.0027	0.0065	0.0090	0.0139
	MAE	0.1013	0.1876	0.2841	0.3684		MAE	0.0027	0.0065	0.0090	0.0139
	NCD	0.1469	0.1925	0.2498	0.3150		NCD	0.0120	0.0231	0.0345	0.0499
FVMF	PSNR	56.3475	56.0565	56.1313	56.7393	VDF	PSNR	59.9415	58.6266	57.3982	56.5027
	MSE	0.1508	0.1612	0.1585	0.1378		MSE	0.0659	0.0892	0.1184	0.0687
	MAE	0.1508	0.1612	0.1585	0.1378		MAE	0.0659	0.0892	0.1184	0.0687
	NCD	0.2976	0.2807	0.2664	0.2500		NCD	0.1899	0.2730	0.3613	0.1479
BF	PSNR	61.7931	61.4045	61.6220	61.8531	VMF	PSNR	59.9443	58.4739	57.2031	56.2887
	MSE	0.0430	0.0471	0.0448	0.0424		MSE	0.0659	0.0924	0.1238	0.1528
	MAE	0.0430	0.0471	0.0448	0.0424		MAE	0.0659	0.0924	0.1238	0.1528
	NCD	0.0725	0.0697	0.0669	0.0637		NCD	0.1899	0.3010	0.0687	0.4836
FBF	PSNR	57.3988	54.9369	53.1825	56.7393	FVMF	PSNR	58.9578	57.6845	56.6944	55.8679
	MSE	0.1206	0.2086	0.3125	0.4000		MSE	0.0827	0.1108	0.1392	0.1684
	MAE	0.1206	0.2086	0.3125	0.4000		MAE	0.0827	0.1108	0.1392	0.1684
	NCD	0.1513	0.1970	0.2531	0.3158		NCD	0.2007	0.3039	0.3914	0.4719
ABF	PSNR	58.1676	55.4917	53.6556	52.5030	PGA	PSNR	61.3561	60.5396	59.6973	58.6394
	MSE	0.0992	0.1836	0.2802	0.3021		MSE	0.0476	0.0574	0.0697	0.0963
	MAE	0.0992	0.1836	0.2802	0.3021		MAE	0.0476	0.0574	0.0697	0.0963
	NCD	0.1526	0.1990	0.2523	0.3654		NCD	0.0806	0.1119	0.3914	0.2096
JBF	PSNR	58.2959	55.6972	54.2882	53.3292	FBF	PSNR	58.9566	58.5418	58.2965	58.1898
	MSE	0.1212	0.1751	0.2422	0.3021		MSE	0.0827	0.0910	0.0963	0.0987
	MAE	0.1212	0.1751	0.2422	0.3021		MAE	0.0827	0.0910	0.0963	0.0987
	NCD	0.1663	0.1995	0.2395	0.2826		NCD	0.1487	0.1765	0.2096	0.2390
SBF	PSNR	57.6339	55.2116	53.3890	52.3143	SBF	PSNR	60.2572	60.2007	60.1151	60.0137
	MSE	0.1121	0.1958	0.2980	0.3816		MSE	0.0613	0.0621	0.0633	0.0648
	MAE	0.1121	0.1958	0.2980	0.3816		MAE	0.0613	0.0621	0.0633	0.0648
	NCD	0.1591	0.2026	0.2570	0.3174		NCD	0.1005	0.1056	0.1113	0.1182
TF	PSNR	57.7566	55.3499	53.5503	52.3937	TF	PSNR	61.0537	60.4132	59.7607	59.5831
	MSE	0.1090	0.1897	0.2871	0.3747		MSE	0.0510	0.0591	0.0687	0.0716
	MAE	0.1090	0.1897	0.2871	0.3747		MAE	0.0510	0.0591	0.0687	0.0716
	NCD	0.1645	0.1958	0.2596	0.3214		NCD	0.0967	0.1261	0.1479	0.1616
proposed	PSNR	70.0219	66.8986	64.2533	62.1717	Proposed	PSNR	70.6712	70.9327	70.8982	69.1310
	MSE	0.0065	0.0133	0.0244	0.0394		MSE	0.0056	0.0052	0.0053	0.0079
	MAE	0.0626	0.1027	0.1447	0.1869		MAE	0.0389	0.0381	0.0381	0.0446
	NCD	0.1357	0.1734	0.2183	0.2675		NCD	0.1028	0.1005	0.1007	0.1281
Mixed noise(Gaussian + impulse)						Mixed noise(poisson+ impulse)					

	$\sigma$	5	10	15	20		$\sigma$	5	10	15	20
PGA	PSNR	57.7721	55.2218	53.5303	52.5230	FPGA	PSNR	60.4633	59.6852	59.0186	58.1504
	MSE	0.1086	0.1954	0.2884	0.3637		MSE	0.0584	0.0699	0.0815	0.0995
	MAE	0.1086	0.1954	0.2884	0.3637		MAE	0.0584	0.0699	0.0815	0.0995
	NCD	0.1576	0.221	0.2995	0.3784		NCD	0.1121	0.1462	0.1938	0.2454
FVMF	PSNR	55.7870	55.0693	54.6806	54.5174	VDF	PSNR	58.0190	57.0440	56.3440	55.7496
	MSE	0.1715	0.2024	0.2213	0.2298		MSE	0.1026	0.1284	0.1509	0.1730
	MAE	0.1715	0.2024	0.2213	0.2298		MAE	0.1026	0.1284	0.1509	0.1730
	NCD	0.3705	0.4119	0.4499	0.4839		NCD	0.2551	0.3457	0.4119	0.4747
FBF	PSNR	57.4342	54.9956	53.2415	52.2377	VMF	PSNR	58.0190	57.1034	56.3609	55.5604
	MSE	0.1174	0.2058	0.3083	0.3884		MSE	0.1004	0.1267	0.1503	0.1807
	MAE	0.1174	0.2058	0.3083	0.3884		MAE	0.1004	0.1267	0.1503	0.1807
	NCD	0.1769	0.2372	0.3020	0.3703		NCD	0.2830	0.3847	0.4575	0.5385
SBF	PSNR	57.5113	55.1269	53.4501	52.3423	FVMF	PSNR	57.7530	56.7839	56.0611	55.3833
	MSE	0.1153	0.1997	0.2938	0.3792		MSE	0.1091	0.1364	0.1611	0.1883
	MAE	0.1153	0.1997	0.2938	0.3792		MAE	0.1091	0.1364	0.1611	0.1883
	NCD	0.1661	0.2153	0.2705	0.3345		NCD	0.2702	0.3579	0.4356	0.5037
TF	PSNR	57.4558	55.0882	53.3985	52.3094	Proposed	PSNR	73.3915	71.9852	70.6346	69.1445
	MSE	0.1168	0.2015	0.2973	0.3821		MSE	0.0334	0.0041	0.0056	0.0079
	MAE	0.1168	0.2015	0.2973	0.3821		MAE	0.0334	0.0370	0.0413	0.0478
	NCD	0.1872	0.2397	0.3055	0.3698		NCD	0.0837	0.0941	0.1112	0.1382
Proposed	PSNR	69.4300	66.2824	63.7289	61.5795						
	MSE	0.0661	0.1070	0.0276	0.1951						
	MAE	0.0661	0.1070	0.0276	0.1951						
	NCD	0.1443	0.1905	0.2450	0.3033						
<b>Mixed noise (Gaussian + impulse+poisson)</b>											
Proposed	PSNR	69.6158	66.2339	63.6126	61.6922						
	MSE	0.0071	0.0155	0.0283	0.0440						
	MAE	0.0647	0.1078	0.1515	0.1927						
	NCD	0.1421	0.1910	0.2474	0.3026						

## (2) Academy128.png

**Table 6. For Different Kinds of Noise**

<b>Gaussian noise:</b>						<b>Impulse noise:</b>					
	$\sigma$	5	10	15	20		$\sigma$	5	10	15	20
PGA	PSNR	60.0590	57.6769	55.8129	54.3902	AMF	PSNR	56.3730	54.8743	52.4420	50.8346
	MSE	0.0641	0.1110	0.1705	0.2366		MSE	0.0030	0.0067	0.0117	0.0170
	MAE	0.0641	0.1110	0.1705	0.2366		MAE	0.0030	0.0067	0.0117	0.0170
	NCD	0.1508	0.1913	0.2413	0.2937		NCD	0.0140	0.0284	0.0472	0.0669
FVMF	PSNR	57.5904	57.2163	57.1086	57.2224	VDF	PSNR	59.2380	58.0110	56.9659	56.2578
	MSE	0.1133	0.1234	0.1265	0.1233		MSE	0.0775	0.1028	0.1308	0.1539
	MAE	0.1133	0.1234	0.1265	0.1233		MAE	0.0775	0.1028	0.1308	0.1539
	NCD	0.3028	0.2686	0.2346	0.2036		NCD	0.1776	0.2804	0.3683	0.4346
BF	PSNR	64.1015	63.7236	63.5633	63.6129	VMF	PSNR	59.5484	58.1224	57.0826	56.2513
	MSE	0.0253	0.6276	0.0286	0.0263		MSE	0.0721	0.1002	0.1273	0.1542
	MAE	0.0253	0.6276	0.0286	0.0263		MAE	0.0721	0.1002	0.1273	0.1542
	NCD	0.0769	0.0722	0.0674	0.0578		NCD	0.1941	0.3039	0.4009	0.4884
FBF	PSNR	59.0650	58.7961	58.3333	58.2633	FVMF	PSNR	58.7304	57.6794	56.8859	55.990
	MSE	0.0806	0.0858	0.0911	0.0970		MSE	0.0871	0.1110	0.1332	0.1634
	MAE	0.0806	0.0858	0.0911	0.0970		MAE	0.0871	0.1110	0.1332	0.1634
	NCD	0.1754	0.2175	0.2519	0.2758		NCD	0.1999	0.3094	0.3854	0.4710
ABF	PSNR	60.4675	57.7806	55.9129	54.4177	PGA	PSNR	61.3305	60.6847	60.2778	59.9990
	MSE	0.0584	0.01084	0.1666	0.2351		MSE	0.0479	0.0555	0.0610	0.0703



	MAE	0.0584	0.01084	0.1666	0.2351		MAE	0.0479	0.0555	0.0610	0.0703
	NCD	0.1560	0.1971	0.2412	0.2971		NCD	0.0759	0.3094	0.3854	0.4710
JBF	PSNR	58.3419	57.0618	56.0322	55.1455	FBF	PSNR	59.2712	59.2072	59.0614	58.9554
	MSE	0.0953	0.1279	0.1621	0.1989		MSE	0.0769	0.0780	0.0807	0.0827
	MAE	0.0953	0.1279	0.1621	0.1989		MAE	0.0769	0.0780	0.0807	0.0827
	NCD	0.1337	0.1632	0.1934	0.2202		NCD	0.1401	0.1660	0.1988	0.2277
SBF	PSNR	59.0197	57.3119	55.6669	54.2721	SBF	PSNR	59.8977	59.9332	59.8414	59.7777
	MSE	0.0815	0.1208	0.1764	0.2431		MSE	0.0666	0.0660	0.0674	0.0684
	MAE	0.0815	0.1208	0.1764	0.2431		MAE	0.0666	0.0660	0.0674	0.0684
	NCD	0.1693	0.2103	0.2557	0.2990		NCD	0.0952	0.1038	0.1132	0.1223
TF	PSNR	59.3195	57.2438	55.6043	54.2601	TF	PSNR	60.8653	60.4661	60.1022	59.6200
	MSE	0.0761	0.1227	0.1789	0.2438		MSE	0.0533	0.0584	0.0635	0.0710
	MAE	0.0761	0.1227	0.1789	0.2438		MAE	0.0533	0.0584	0.0635	0.0710
	NCD	0.1798	0.2146	0.2573	0.3038		NCD	0.0844	0.1031	0.1238	0.1386
proposed	PSNR	69.3128	66.7176	64.4361	62.6226	proposed	PSNR	74.2217	71.1501	69.6657	68.4261
	MSE	0.0076	0.0138	0.0234	0.0355		MSE	0.0025	0.0050	0.0070	0.0093
	MAE	0.0660	0.1008	0.1375	0.1713		MAE	0.0290	0.0375	0.0434	0.0488
	NCD	0.1369	0.1658	0.1986	0.2290		NCD	0.0576	0.0805	0.0989	0.1269
<b>Mixed noise(Gaussian + impulse)</b>						<b>Mixed noise(poisson+ impulse)</b>					
	$\sigma$	5	10	15	20		$\sigma$	5	10	15	20
PGA	PSNR	59.6811	57.4105	55.6541	54.3731	FPGA	PSNR	60.9647	60.1645	59.2927	58.5753
	MSE	0.0700	0.1180	0.1769	0.2376		MSE	0.0521	0.0626	0.0765	0.0903
	MAE	0.0700	0.1180	0.1769	0.2376		MAE	0.0521	0.0626	0.0765	0.0903
	NCD	0.1639	0.2250	0.2934	0.3629		NCD	0.1209	0.1563	0.1985	0.2638
FVMF	PSNR	56.7468	55.9044	55.3603	54.7644	VDF	PSNR	58.7710	57.6667	56.6409	55.6791
	MSE	0.1375	0.1670	0.1893	0.2171		MSE	0.1081	0.1113	0.1409	0.1759
	MAE	0.1375	0.1670	0.1893	0.2171		MAE	0.1081	0.1113	0.1409	0.1759
	NCD	0.3691	0.3968	0.4099	0.4496		NCD	0.3192	0.4086	0.4850	0.5688
FBF	PSNR	58.6908	57.1110	55.3921	54.0773	VMF	PSNR	57.7930	57.0315	56.3889	55.7867
	MSE	0.0879	0.1265	0.1879	0.2543		MSE	0.1081	0.1259	0.1493	0.1716
	MAE	0.0879	0.1265	0.1879	0.2543		MAE	0.1081	0.1259	0.1493	0.1716
	NCD	0.1641	0.2050	0.2570	0.2961		NCD	0.2923	0.3658	0.4269	0.4844
SBF	PSNR	59.0348	57.3408	55.7755	54.3313	FVMF	PSNR	58.2722	57.4049	56.5908	55.853
	MSE	0.0812	0.1199	0.1720	0.2399		MSE	0.0968	0.1182	0.1426	0.1677
	MAE	0.0812	0.1199	0.1720	0.2399		MAE	0.0968	0.1182	0.1426	0.1677
	NCD	0.1764	0.2228	0.2693	0.3186		NCD	0.2891	0.3737	0.4451	0.5055
TF	PSNR	58.9566	56.9053	55.2275	54.0598	Proposed	PSNR	71.5152	70.5853	69.6474	68.3453
	MSE	0.0827	0.1326	0.1951	0.2553		MSE	0.0046	0.0057	0.0071	0.0095
	MAE	0.0827	0.1326	0.1951	0.2553		MAE	0.0398	0.0435	0.0474	0.0539
	NCD	0.1967	0.2397	0.2870	0.3344		NCD	0.0861	0.0982	0.1144	0.1464
Proposed	PSNR	68.6859	66.1393	63.9407	62.1186						
	MSE	0.0088	0.0158	0.0262	0.0399						
	MAE	0.0693	0.1046	0.1412	0.1757						
	NCD	0.1466	0.1812	0.2220	0.2582						
<b>Mixed noise (Gaussian + impulse+poisson)</b>											
Proposed	PSNR	68.6052	66.0514	64.0136	62.2536						
	MSE	0.0090	0.0161	0.0258	0.0387						
	MAE	0.0692	0.1049	0.1383	0.1722						
	NCD	0.1548	0.1871	0.2224	0.2576						

(3) Lena256.png

**Table 7. For Different Kinds of Noises**

Gaussian noise:						Impulse noise:					
	$\sigma$	5	10	15	20		$\sigma$	5	10	15	20
PGA	PSNR	58.4496	55.6894	53.7878	52.5479	AMF	PSNR	56.6224	55.3680	54.1635	53.7243
	MSE	0.0929	0.1754	0.2718	0.3617		MSE	0.0022	0.0047	0.0079	0.0110
	MAE	0.0929	0.1754	0.2718	0.3617		MAE	0.0022	0.0047	0.0079	0.0110
	NCD	0.1412	0.1885	0.2474	0.3122		NCD	0.0086	0.0176	0.0270	0.0382
FVMF	PSNR	56.6802	56.4057	56.4966	56.9909	VDF	PSNR	60.3751	58.8331	57.6598	56.7323
	MSE	0.1397	0.1488	0.1457	0.1300		MSE	0.0596	0.0851	0.1115	0.1380
	MAE	0.1397	0.1488	0.1457	0.1300		MAE	0.0596	0.0851	0.1115	0.1380
	NCD	0.2897	0.2737	0.2594	0.2435		NCD	0.1714	0.2689	0.3585	0.4330
BF	PSNR	57.5378	55.5510	53.9122	52.7627	VMF	PSNR	60.3774	58.7518	57.4264	56.4641
	MSE	0.1146	0.1811	0.2642	0.3442		MSE	0.0596	0.0867	0.1176	0.1468
	MAE	0.1146	0.1811	0.2642	0.3442		MAE	0.0596	0.0867	0.1176	0.1468
	NCD	0.2286	0.2590	0.3036	0.3562		NCD	0.1834	0.2899	0.3902	0.4786
FBF	PSNR	57.9201	55.4308	53.4298	52.3886	FVMF	PSNR	59.6889	58.2880	57.2279	56.3038
	MSE	0.1050	0.1862	0.2952	0.3752		MSE	0.0699	0.0964	0.1231	0.1523
	MAE	0.1050	0.1862	0.2952	0.3752		MAE	0.0699	0.0964	0.1231	0.1523
	NCD	0.1304	0.1863	0.2404	0.3058		NCD	0.1882	0.2884	0.3767	0.4569
ABF	PSNR	58.4252	55.7342	53.7883	52.6046	PGA	PSNR	62.2666	61.3554	60.5805	59.4335
	MSE	0.0934	0.1736	0.2718	0.3570		MSE	0.0386	0.0476	0.0569	0.0741
	MAE	0.0934	0.1736	0.2718	0.3570		MAE	0.0386	0.0476	0.0569	0.0741
	NCD	0.1457	0.1920	0.2497	0.3138		NCD	0.0717	0.0986	0.1379	0.1884
JBF	PSNR	57.7072	55.7248	54.1963	53.1404	FBF	PSNR	60.1678	59.7448	59.3695	58.8880
	MSE	0.1102	0.1737	0.2474	0.3155		MSE	0.0626	0.0690	0.0752	0.0840
	MAE	0.1102	0.1737	0.2469	0.3147		MAE	0.0626	0.0690	0.0752	0.0840
	NCD	0.1369	0.1784	0.2235	0.2700		NCD	0.1206	0.1563	0.1900	0.2228
SBF	PSNR	58.0088	55.4869	53.6220	52.4744	SBF	PSNR	61.1132	61.0992	60.9746	60.8696
	MSE	0.1028	0.1838	0.2824	0.3678		MSE	0.0503	0.0505	0.0520	0.0532
	MAE	0.1028	0.1838	0.2824	0.3678		MAE	0.0503	0.0505	0.0520	0.0532
	NCD	0.1464	0.1915	0.2499	0.3127		NCD	0.0873	0.0908	0.0951	0.1003
TF	PSNR	58.2006	55.6476	53.7697	52.5662	TF	PSNR	61.8670	61.0485	60.6188	60.3489
	MSE	0.0984	0.1771	0.2730	0.3601		MSE	0.0423	0.0511	0.0564	0.0600
	MAE	0.0984	0.1771	0.2730	0.3601		MAE	0.0423	0.0511	0.0564	0.0600
	NCD	0.1560	0.1989	0.2544	0.3162		NCD	0.0898	0.1195	0.1403	0.1574
proposed	PSNR	70.0368	66.8986	64.2533	62.1717	Proposed	PSNR	69.6712	68.9327	68.8982	67.1013
	MSE	0.0063	0.0133	0.0244	0.0394		MSE	0.0052	0.0056	0.0052	0.0079
	MAE	0.0610	0.1027	0.1447	0.1869		MAE	0.0389	0.0381	0.0381	0.0446
	NCD	0.1284	0.1734	0.2183	0.2675		NCD	0.1028	0.1005	0.1007	0.1281
Mixed noise(Gaussian + impulse)						Mixed noise(poisson+ impulse)					
	$\sigma$	5	10	15	20		$\sigma$	5	10	15	20
PGA	PSNR	58.1961	55.4924	53.6943	52.6537	FPGA	PSNR	61.5104	60.7635	60.0200	59.1230
	MSE	0.0985	0.1836	0.2777	0.3529		MSE	0.0459	0.0545	0.0647	0.0796
	MAE	0.0985	0.1836	0.2777	0.3529		MAE	0.0459	0.0545	0.0647	0.0796
	NCD	0.1513	0.2127	0.2923	0.3718		NCD	0.0967	0.1225	0.1559	0.2069
FVMF	PSNR	56.0833	55.3185	54.9376	54.7106	VDF	PSNR	58.4104	57.4428	56.6380	55.9690
	MSE	0.1602	0.1911	0.2086	0.2198		MSE	0.0938	0.1172	0.1410	0.1645
	MAE	0.1602	0.1911	0.2086	0.2198		MAE	0.0938	0.1172	0.1410	0.1645
	NCD	0.3603	0.4053	0.4399	0.4768		NCD	0.2492	0.3306	0.4031	0.4698
FBF	PSNR	57.8764	55.3446	53.5197	52.4438	VMF	PSNR	60.3377	58.7482	57.5141	56.4742
	MSE	0.1060	0.1899	0.2891	0.3704		MSE	0.0602	0.0867	0.1153	0.1464
	MAE	0.1060	0.1899	0.2891	0.3704		MAE	0.0602	0.0867	0.1153	0.1464
	NCD	0.1564	0.2202	0.2892	0.3563		NCD	0.1834	0.2883	0.3863	0.4766
SBF	PSNR	57.9695	55.4855	53.6743	52.4955	FVMF	PSNR	58.2878	57.2988	56.4812	55.7570
	MSE	0.1038	0.1839	0.2790	0.3660		MSE	0.0965	0.1211	0.1462	0.1727
	MAE	0.1038	0.1839	0.2790	0.3660		MAE	0.0965	0.1211	0.1462	0.1727
	NCD	0.1529	0.2015	0.2608	0.3261		NCD	0.2583	0.3453	0.4197	0.4896
TF	PSNR	58.0203	55.5081	53.7117	52.4849	Proposed	PSNR	73.3915	71.9852	70.6346	69.1445

	MSE	0.1026	0.1829	0.2766	0.3669		MSE	0.0334	0.0041	0.0056	0.0079
	MAE	0.1026	0.1829	0.2766	0.3669		MAE	0.0334	0.0370	0.0413	0.0478
	NCD	0.1774	0.2317	0.2932	0.3617		NCD	0.0837	0.0941	0.1112	0.1382
Proposed	PSNR	69.4315	66.2854	63.7292	61.5834						
	MSE	0.0667	0.1075	0.0276	0.1951						
	MAE	0.0667	0.1075	0.0276	0.1951						
	NCD	0.1451	0.1910	0.2450	0.3033						
<b>Mixed noise (Gaussian + impulse+poisson)</b>											
Proposed	PSNR	69.6158	66.2339	63.6126	61.6922						
	MSE	0.0071	0.0155	0.0283	0.0440						
	MAE	0.0647	0.1078	0.1515	0.1927						
	NCD	0.1421	0.1910	0.2474	0.3026						

(4) Academy256:

Table 8. For Different Kinds of Noise

Gaussian noise:						Impulse noise:					
	$\sigma$	5	10	15	20		$\sigma$	5	10	15	20
PGA	PSNR	60.3367	57.8561	56.0292	54.6434	AMF	PSNR	55.3736	54.0322	52.8423	50.4269
	MSE	0.0602	0.1065	0.1622	0.2232		MSE	0.0024	0.0051	0.0085	0.0118
	MAE	0.0602	0.1065	0.1622	0.2232		MAE	0.0024	0.0051	0.0085	0.0118
	NCD	0.1452	0.1887	0.2395	0.2893		NCD	0.0112	0.0228	0.0356	0.0500
FVMF	PSNR	58.0199	57.8015	57.6714	57.7962	VDF	PSNR	59.7719	58.5082	57.3805	56.5237
	MSE	0.1026	0.1079	0.1112	0.1080		MSE	0.0685	0.0917	0.1189	0.1448
	MAE	0.1026	0.1079	0.1112	0.1080		MAE	0.0685	0.0917	0.1189	0.1448
	NCD	0.2975	0.2625	0.2287	0.1972		NCD	0.1721	0.2688	0.3565	0.4296
BF	PSNR	59.4866	57.6149	56.0045	54.7606	VMF	PSNR	60.3124	58.6664	57.4693	56.4088
	MSE	0.0732	0.1126	0.1632	0.2173		MSE	0.0605	0.0884	0.1165	0.1487
	MAE	0.0732	0.1126	0.1632	0.2173		MAE	0.0605	0.0884	0.1165	0.1487
	NCD	0.2435	0.2642	0.2944	0.3322		NCD	0.1778	0.2927	0.3889	0.4807
FBF	PSNR	59.5130	57.6353	55.7921	54.3345	FVMF	PSNR	59.1891	58.0724	57.1156	56.3435
	MSE	0.0727	0.1121	0.1713	0.2397		MSE	0.0784	0.1014	0.1263	0.1509
	MAE	0.0727	0.1121	0.1713	0.2397		MAE	0.0784	0.1014	0.1263	0.1509
	NCD	0.1260	0.1721	0.2252	0.2771		NCD	0.1925	0.2920	0.3799	0.4540
ABF	PSNR	60.5250	57.8884	55.9962	60.5991	PGA	PSNR	61.9974	61.3277	60.8662	60.2612
	MSE	0.0576	0.1057	0.1635	0.0566		MSE	0.0411	0.0479	0.0533	0.0612
	MAE	0.0576	0.1057	0.1635	0.0566		MAE	0.0411	0.0479	0.0533	0.0612
	NCD	0.1509	0.1923	0.2424	0.1510		NCD	0.0643	0.0985	0.1453	0.2026
JBF	PSNR	58.7404	57.2192	56.0890	55.1740	FBF	PSNR	60.1299	60.0004	59.8300	59.6282
	MSE	0.0869	0.1234	0.1600	0.1976		MSE	0.0631	0.0650	0.0676	0.0708
	MAE	0.0865	0.1228	0.1593	0.1960		MAE	0.0631	0.0650	0.0676	0.0708
	NCD	0.1232	0.1563	0.1892	0.2186		NCD	0.1224	0.1562	0.1855	0.2121
SBF	PSNR	59.4983	57.6253	55.8555	54.5360	SBF	PSNR	60.7488	60.7185	60.6557	60.5486
	MSE	0.0730	0.1123	0.1689	0.2288		MSE	0.0547	0.0551	0.0559	0.0573
	MAE	0.0730	0.1123	0.1689	0.2288		MAE	0.0547	0.0551	0.0559	0.0573
	NCD	0.1594	0.2032	0.2505	0.2951		NCD	0.0777	0.0852	0.0933	0.1017
TF	PSNR	59.9347	57.6415	55.9278	54.5774	TF	PSNR	62.0538	61.5447	61.1541	60.8374
	MSE	0.0660	0.1119	0.1661	0.2266		MSE	0.0405	0.0456	0.0499	0.0536
	MAE	0.0660	0.1119	0.1661	0.2266		MAE	0.0405	0.0456	0.0499	0.0536
	NCD	0.1664	0.2037	0.2482	0.2951		NCD	0.0707	0.0912	0.1071	0.1202
proposed	PSNR	69.4361	66.3128	64.7176	62.4361	proposed	PSNR	74.2217	71.1501	69.6657	68.4261
	MSE	0.0067	0.0118	0.0245	0.0350		MSE	0.0025	0.0050	0.0070	0.0093
	MAE	0.0660	0.1012	0.1266	0.1801		MAE	0.0290	0.0375	0.0434	0.0488
	NCD	0.1254	0.1658	0.1986	0.2290		NCD	0.0576	0.0805	0.0989	0.1269
<b>Mixed noise(Gaussian + impulse)</b>						<b>Mixed noise(poisson+ impulse)</b>					
	$\sigma$	5	10	15	20		$\sigma$	5	10	15	20
PGA	PSNR	60.0345	57.5867	55.7913	54.5536	FPGA	PSNR	61.8570	61.0293	60.3144	59.4035

	MSE	0.0645	0.1133	0.1714	0.2279		MSE	0.0424	0.0513	0.0605	0.0746
	MAE	0.0645	0.1133	0.1714	0.2279		MAE	0.0424	0.0513	0.0605	0.0746
	NCD	0.1585	0.2177	0.2879	0.3582		NCD	0.1068	0.1372	0.1749	0.2248
FVMF	PSNR	57.2297	56.3120	55.6165	55.6165	VDF	PSNR	58.2627	57.4767	56.7152	56.1098
	MSE	0.1231	0.1520	0.1784	0.1784		MSE	0.0970	0.1163	0.1385	0.1593
	MAE	0.1231	0.1520	0.1784	0.1784		MAE	0.0970	0.1163	0.1385	0.1593
	NCD	0.3632	0.3882	0.4121	0.4121		NCD	0.2822	0.3579	0.4223	0.4803
FBF	PSNR	59.3930	57.5607	55.8236	54.4246	VMF	PSNR	60.3455	58.6228	57.4273	56.4256
	MSE	0.0748	0.1140	0.1701	0.2348		MSE	0.0601	0.0893	0.1176	0.1481
	MAE	0.0748	0.1140	0.1701	0.2348		MAE	0.0601	0.0893	0.1176	0.1481
	NCD	0.1506	0.1979	0.2456	0.2890		NCD	0.1766	0.2894	0.3895	0.4791
SBF	PSNR	59.4806	57.6630	55.8749	54.5381	FVMF	PSNR	58.7626	57.7752	56.8408	56.0974
	MSE	0.0733	0.1114	0.1681	0.2287		MSE	0.0865	0.1085	0.1346	0.1597
	MAE	0.0733	0.1114	0.1681	0.2287		MAE	0.0865	0.1085	0.1346	0.1597
	NCD	0.1658	0.2134	0.2621	0.3090		NCD	0.2806	0.3587	0.4331	0.4957
TF	PSNR	59.6311	57.4673	55.7074	54.4390	Proposed	PSNR	71.5152	70.5853	69.6474	68.3453
	MSE	0.0708	0.1165	0.1747	0.2340		MSE	0.0046	0.0057	0.0071	0.0095
	MAE	0.0708	0.1165	0.1747	0.2340		MAE	0.0398	0.0435	0.0474	0.0539
	NCD	0.1818	0.2258	0.2745	0.3214		NCD	0.0861	0.0982	0.1144	0.1464
Proposed	PSNR	68.6859	66.1393	63.9407	62.1186						
	MSE	0.0088	0.0158	0.0262	0.0399						
	MAE	0.0693	0.1046	0.1412	0.1757						
	NCD	0.1466	0.1812	0.2220	0.2582						
<b>Mixed noise (Gaussian + impulse+poisson)</b>											
Proposed	PSNR	68.6052	66.0514	64.0136	62.2536						
	MSE	0.0096	0.0161	0.0284	0.0391						
	MAE	0.0695	0.1049	0.1389	0.1724						
	NCD	0.1567	0.1871	0.2224	0.2580						

## 6. Conclusion and Future Scope

In this paper, a novel mechanism is proposed to de-noise image corrupted with impulse, Gaussian and poisson noise. The following important inference can be drawn from the proposed technique as follows:

In terms of picture quality, the best results are obtained for the proposed technique for all the noises.

The value of PSNR decreases with increase of noise density.

The proposed technique provides good PSNR, NCD and MAE value for all noise densities in comparison to other methods used in this paper.

## References

- [1] Y. Shen and K. E. Barner, "Fuzzy Vector Median-Based Surface Smoothing", in the proceedings of IEEE transactions, vol. no 10, issue no.3, (2004).
- [2] J. Astola, P. Haavisto and Y. Neuvo, "Vector Median Filters", in the Proceedings of IEEE, vol.78, no. 4,(1990).
- [3] B. P. Lamichhane, "Finite element techniques for removing the mixture of Gaussian and impulsive noise", in the proceedings of IEEE Transactions on Image Processing, vol. 57, no. 7, (2009) Jul., pp. 2538-2547.
- [4] F. Luisier, T. Blu and M. Unser, "Image Denoising in Mixed Poisson–Gaussian Noise", IEEE Transactions on Image Processing, vol. no 20, Issue no. 3, (2011) March, pp. 696 – 708.
- [5] J. Liu, Z. Huan and H. Huang, "Image restoration under mixed noise using globally convex segmentation", in Journal of Visual Communication and Image Representation, vol. 22, no.3, (2011), pp. no 263-270.
- [6] S. Setzer, G. Steidl and T. Teuber, "Deblurring Poissonian images by split Bregman techniques", in the Journal of Visual Communication and Image Representation, vol. 21, no. 3, (2010) April, pp.193–199.
- [7] R. Lukac and K.N. Plataniotis, "A taxonomy of color images filtering and enhancement solutions", in Advances in Imaging and Electron Physics, P.W. Hawkes, Elsevier, no. 140, (2006), pp. 127-264.
- [8] S. Durand, J. Fadili and M. Nikolova, "Multiplicative Noise Removal Using L1 Fidelity on Frame coefficients", Journal of Mathematical Imaging and Vision, Springer, vol. 36, (2009), pp. 201-226.

- [9] G. Hewer, C. Kenney, L. Peterson and A. Van Nevel, "Applied partial differential variational techniques", in Proceedings of International Conference on Image Processing, vol. 3, (1997), pp. 372-375
- [10] J. Y. F. Ho, "Peer region determination based impulsive noise detection", Proceedings of International Conference on Acoustics, Speech and Signal Processing ICASSP'03, vol. 3, (2003), pp. 713-716.
- [11] Y. Xiao, T. Zeng, J. Yu and M. K. Ng, "Restoration of images corrupted by mixed Gaussian-impulse noise via  $l_1 - l_0$  minimization", Pattern Recognition, vol. 44, no. 8, (2011) Aug., pp. 1708-1720.
- [12] S. Morillas, V. Gregori and A. Hervás, "Fuzzy peer groups for reducing mixed Gaussian impulse noise from color images", in the proceedings IEEE Transactions on Image Processing, vol. no. 18, issue no. 7, (2008) July, pp. 1452-1466.
- [13] R. Garnett, T. Huegerich, C. Chui, W. He, "A universal noise removal algorithm with an impulse detector", in the proceedings IEEE Transactions on Image Processing, , vol. 14, no. 11, (2005) Nov., pp. 1747-1754
- [14] J. Liu, Z. Huan and H. Huan, "Image restoration under mixed noise using globally convex segmentation", Journal of Visual Communication and Image Representation, vol. no. 22, issue no. 3, (2011), Apr., pp. 263-270.
- [15] X. Zeng and L. Yang, "Mixed impulse and Gaussian noise removal using detail-preserving regularization", Optical Engineering, vol. 49, issue no. 9, (2010) Sep., pp. 097002-1-097002-9.
- [16] R. Liu, S. Fu and C. Zhang, "Adaptive mixed image denoising based on image decomposition", Optical Engineering Letters, vol. 50, no. 2, (2011) Feb., pp. 020502-1-020502-3.
- [17] O. Ghita and P.F. Whelan, "A new GVF-based image enhancement formulation for use in the presence of mixed noise", Pattern Recognition, vol. 43, no. 8, (2010) Aug., pp. 2646-2658.
- [18] G. Plonka and J. Ma, "Nonlinear regularized reaction-diffusion filters for denoising of images with textures", IEEE Transactions on Image Processing, vol. 17, no. 8, (2007) Aug., pp. 1283-1294.
- [19] Z. Ma, H.R. Wu and D. Feng, "Partition Based Vector Filtering Technique for Suppression of Noise in Digital Color Images", IEEE Transactions on Image Processing, vol. 15, no. 8, (2006) Aug. , pp. 2324-2342.
- [20] Z. Ma, H.R. Wu and D. Feng, "Fuzzy Vector Partition Filtering Technique for Color Image Restoration", Computer Vision and Image Understanding, vol.107,no. 1-2,(2007) July-Aug., pp. 26-37.
- [21] E. L'opez-Rubio, "Restoration of images corrupted by Gaussian and uniform impulsive noise", Pattern Recognition, pp. 1835-1846, vol. 43, no. 5, (2010) May.
- [22] D. Keren and A. Gotlib, "Denoising Color Images using regularization and correlation terms", Journal of Visual Communication and Image Representation, vol. 9, issue no. 4, (1998) Dec., pp. 352-365.
- [23] O. Lezoray, A. Elmoataz and S. Boughleux, "Graph regularization for color image processing", Computer Vision and Image Understanding, vol. 107, no. 1-2, (2007) July-Aug., pp. 38-55.
- [24] A. Elmoataz, O. Lezoray and S. Boughleux, "Nonlocal discrete regularization on weighted graphs: A framework for image and manifold processing", in the proceedings of IEEE Transactions on Image Processing, pp. 1047-1060, vol.17, issue no. 7, (2008) July.
- [25] P. Blomgren and T. Chan, "Color TV: total variation methods for restoration of vector-valued images", in the proceedings of IEEE Transactions on Image Processing, vol.7,no. 3, (1998) Mar., pp. 304-309.
- [26] D. Tschumperl'e and R. Deriche, "Vector-valued image regularization with PDEs: A Common framework from different applications", in the proceedings of IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 27 no. 4.
- [27] R. Sharma and J. Ali, "A Comparative study various types of noise and efficient noise removal techniques", in IJARCSSE, vol. 3, no. 10, (2013), pp. 617-622.
- [28] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images", in the proceedings of 6th IEEE International conference on computer vision, (1998), pp no.834-846.
- [29] Garimagoyal, "Impact and analysis of improved bilateral filter on TEM images", in International journals of science and research, vol. 3. issue 6, (2014).
- [30] N. Himayat and S.A.Kassam, "Approximate performance analysis of edge preserving filters", in the proceedings of IEEE Transaction, (1993), pp. 2764-77.
- [31] Buyue Zhang and J.P. Allebach, "Adaptive Bilateral Filter for Sharpness Enhancement and Noise Removal", in the proceedings of IEEE transactions, vol. 17, no. 5, (2008), pp. 664-678.
- [32] D. Barash, "A fundamental relationship between bilateral filtering, adaptive smoothing and the non linear diffusion equation:, in the proceedings of IEEE transactions, vol. 24, no. 6, (2002) June, pp. 844-847.
- [33] R.Garnett, T.Huegerich, C. Chui and W.He, "A universal noise removal algorithm with an impulse detector", in the proceedings of IEEE transactions, vol.17, no. 7, (2008) July, pp.1109-1120.
- [34] M.Zhang and B.K. Gunturk, "Multiresolution bilateral filtering for image denoising", in the proceedings of IEEE transactions, vol. 17, no. 12, (2008) Dec., pp. 2324-2333.
- [35] C.-H. Lin, J.-S. Tsai and Ching-Te Chiu, "Switching bilateral filter with a texture/noise detector for universal noise removal" in the proceedings of IEEE transactions on the image processing, vol. 19, no. 19, (2010), pp.2307-2320.

- [36] A. K. Nain, S. Gupta and B. Bhushan, "An extension to switching bilateral filter for mixed noise removal from color images" in the proceedings of Int. J. Signal and Imaging Systems Engineering, vol. X, no. Y, (2014).
- [37] O.U.NirmalJith and R. VenkateshBabu "Joint bilateral filtering based non local means image denoising", in the proceedings of IEEE transactions on image processing, (2014).
- [38] G.Petschnigg, R. Szeliski, M.Agrawala, M.cohen, H.Hoppe and K.Toyama, "Digital photography with flash and no flash image pairs", in ACM SIGGRAPH , pp. 664-672,2004.
- [39] K. Malik and B. Smolka, "Improved bilateral filtering scheme for noise removal in color images", in International Conference on Informatics and Applications, (2004), pp. 118-130.
- [40] M.I Elad, "On the Origin of bilateral filter and ways to improve it", in the processing of IEEE transactions, vol. 11, no. 10, (2002).
- [41] M. Nagao and T.Matusuyama, "Edge preserving smoothing", in the proceedings of IEEE transactions, (1979), pp. 394-407

## Authors

**Mr. Shailender Gupta** is working as an Assistant Professor in YMCA University of Science and Technology, Faridabad, Haryana, India. He has teaching experience of seven years. His research interest is in Image processing.

**Ms Sweety Deswal** is an M.tech student in branch Electronics and communication engineering in YMCA University of Science and Technology, Faridabad, Haryana, India.

**Ms Surbhi Singhania** is a Phd student in branch Electronics and computer engineering in Carnegie Mellon University, Pittsburgh,US.

**Ms Pranjal Garg** is an M.tech student in branch Electronics and communication engineering in YMCA University of Science and Technology, Faridabad, Haryana, India.