Chapter 4
Improved Selective Median Filter (ISMF)

Image pre-processing is a technique that improves image quality by removing noise and unwanted objects from the background. This chapter highlights the significance of such work in the context of TGFD.

4.1 PREPROCESSING

Noise means variation in the brightness of an image or a blurred image. There are different noises, such as salt and pepper noise, Gaussian noise and Rician Noise. The real images of Gujarati Food have been collected by visiting several restaurants and from the internet with different angles, positions and lighting conditions. The images contain noise, unwanted objects, background, etc. It is necessary to remove noise from the images so the interesting content can be highlighted. The natural images are affected by all types of noise, but food images are affected by impulse noise due to poor illumination quality of the camera, poor lighting, or blur image.

In food images, it is very necessary to retain edges after pre-processing an image, as it is the factor that detects the shape of an image. Filtering in image processing is used to detect the edges of any food image. Filtering can be done in the spatial or frequency domains. Since, this study is for food images, it focuses on spatial domain approaches.

Image denoising is a technique that removes noise from images without losing crucial information. It is an important task that needs to be carried out in any image processing technique [55,139]. Image denoising techniques can be divided into two types: linear and non-linear. The advantage of using linear techniques is the execution speed factor, but the problem is that it does not protect the edges of the image in an effective manner. e.g., the image details and edges are blurred out. Nonlinear techniques are more effective than linear ones in handling image edges [8,139]. Comparison of several features of linear and nonlinear filter is given in Table 4.1. Some of the spatial nonlinear techniques are median filter, weighted median filter, midpoint filter, min filter, etc.

Table 4.1 Comparison of Linear and Nonlinear Filters

Factor	Linear Filters	Nonlinear Filters
Linearity	Linear response to the input signal	Nonlinear response to the input signal
Convolution	Typically implemented using convolution operations	Non-convolution-based operations
Superposition	Obey the principle of superposition (response to sum of inputs is the sum of individual responses)	Do not obey the principle of superposition
Frequency Response	Can be characterized by frequency response	No specific frequency response characteristics
Examples	Low pass filters, High pass filters	Median filters, mean filters, minimum filters, maximum filters
Application	Frequency-based filtering, signal analysis	Noise removal, edge detection, image enhancement

From the above comparison it is clear that Nonlinear filter techniques give better results than linear ones.

4.1.1 Non-Linear Filter

Nonlinear filters are computational algorithms that process data using non-linear transformations, enabling them to capture complex patterns and relationships.

- **Minimum Filter:** The minimal filter, also known as the min filter or erosion filter, replaces each pixel in an image with the least significant pixel in its neighborhood. It is effective at removing high-frequency noise and small details from an image because it chooses the lowest intensity value in the neighborhood, resulting in a smoothed image. The minimal filter is employed in a variety of applications, including noise reduction, edge preservation and morphological procedures such as erosion [139].

- **Maximum Filter:** The maximum filter, also known as the max filter or dilation filter, replaces each pixel in an image with the maximum value in its vicinity. It can be used to extend or thicken bright patches or objects in an image. It chooses the highest intensity value in the area, emphasizing salient features or edges. Maximum filter uses include picture enhancement, morphological procedures such as dilation and feature extraction [45].

- **Mean Filter:** A mean filter works by converging an image with a kernel or window of a specific size. The average of the pixel values within the kernel window is used to calculate the value of each pixel in the output image. Mean filters are typically used to reduce noise and blur. High-frequency noise or minor details in the image are minimised by replacing each pixel with the average of its neighbors, resulting in a smoother image. The limitation of this filter is that it reduce noise effectively and blur or smooth off critical visual details and edges. In some applications, they may not be suited for keeping fine textures or sharp edges [139].

- **Median Filter**: A median filter is the famous filtering technique used for removing impulse noise. In this technique, the value of each reconstructed pixel is equal to the median of its nearest neighbors in the input image [45].

The median filter is widely recognized for its effectiveness in eliminating impulse noise from images while preserving edge details, making it a popular choice. However, in high noise density images, the median filter may yield inaccurate outcomes, which is considered as one of its drawbacks.

To resolve this a lot of work has been done by many researchers. A study on the median filter with various variants to discard the impulse noise from grayscale images has been presented by Anwar et al. [8]. A comparison has been done on filters based on computational complexity and performance. The conventional filter is good for low noise but fails to preserve edges. The database algorithm is good for images with low noise density.

A hybrid median filter for removing impulse noise from an image has been proposed by M. Narsimha et al. [56]. The proposed filter is a nonlinear filter, an improved version of the median filter, which helps to remove noise and preserve main features. The experiment has been implemented in MATLAB. According to the results, the hybrid median filter is simple to understand and performs better than the median filter. According to the author, the disadvantage of the proposed filter is that it has a high

computational cost, so to avoid that, new filters should be developed. Youlian Zhu et al. have proposed an improvement in the existing median filter which adds a mask over the image [63]. It has Very acceptable time complexity reduction but Misleading in noise detection.

From the above study the following research gap has been found.

Research Gap:

- Different variations of median fitter techniques have been used for efficient noise removal, but they have difficulty removing impulse noise while preserving edges and contours.
- The median filter is effective in the case of low noise density only. As the noise density is increased by more than 50%, the size of the filtering window should be enlarged to suppress serious noise which leads to edges and details being blurred.
- Due to the automatic modulation approach median filter miss classified pixels and visually unpleasant filtered images with high noise density.

All the existing algorithm either fails to remove noise for high density images or makes the image blur. To avoid this deficiency of the median filter, it requires to identify noisy pixel first and then should be replaced by the filter operator keeping other pixels unchanged [55]. Based on this fact, to improve noise suppression and detail preservation simultaneously, a new filter has been proposed.

4.2 IMPROVED SELECTIVE MEDIAN FILTER

To retain edges after pre-processing an image and to overcome the limitation of the median filter, a novel median-based method “Improved Selective median filter (ISMF)" has been proposed which is able to differentiate between noisy and healthy pixels.

This approach only treats the noisy pixels, which reduces the time complexity and search space and improves the performance at the same time. The proposed method is simple, easy to implement and requires very little computation. In this work, the emphasis is given to removing impulse noise from food images.

The proposed algorithm sorts all the elements of the extracted window size of 3X3. The pixels that lie between 0 and 255 are considered to be healthy otherwise the pixel is termed noisy. All the operations are performed in a fixed 2D Kernel of size 3X3 window. Any operation performed is based on the pixel values in the current processing window.

4.2.1 ISMF Algorithm

The images from TGFD have been used as input for this algorithm. The current processing pixel is termed as P(i,j). The algorithm starts by checking the pixel value whether it is noisy by checking condition $0 < P(i,j) < 255$. If the pixel value falls within the range specified, it is healthy pixel and the value remains unchanged. If the pixel is noisy it will check for all neighbor pixel values. If all the four neighbor pixels are also noisy, the corrupted pixel value will be replaced by the mean of all the elements. If some of the neighbors are noisy then the number of noisy pixel values is store in variable noisy pixel (NP). if NP = 2 then the corrupted pixel will be replaced by the median value of the remaining elements by eliminating the corrupted pixels. If we take the mean of all the elements, then the resulting elements may also be noisy. To avoid it median value should be taken. If 3<=NP<=5, the corrupted pixel will be replaced by the mean of the remaining elements by eliminating the corrupted pixel. If the median value is taken then there are maximum chances that the median value will also be corrupted and hence the mean should be taken. If NP>=6, then it is necessary to check whether the median value is between the minimum and maximum value that is, it should be between 0 and 255. If max>med>min then only the corrupted pixel should be replaced with the median value else keep the original value and move to the next pixel by increasing the window size. This algorithm will be repeated for all the pixels of an input image. An algorithm for the implementation of ISMF is given in Fig. 4.1 and the visual presentation of the algorithm has been shown in Fig. 4.2.

Step 1: To begin, select a 3x3 2D window.

Step 2: The process begins from the current pixel P(i,j) from top left corner of the image to see if it is noisy or not.

Step 3: If the value of P(i,j) is between 0 and 255, then it is not the corrupted pixel and its value remains unchanged.

Step 4: If P(i,j) = 0 or P(i,j) = 255, then the pixel is noisy and its value will be calculated as below.

4.1 If all the four neighbour pixels are noisy, then take the mean of all the elements and replace the Pij value with the same. if we take the median of the elements, it will be noisy.

4.2 If any of the pixels are not noisy, then check the current processing window and calculate the number of noisy pixels and store the value in the variable noisy pixel (NP). The value of Pij will be decided based on the value of NP according to the following conditions:

4.2.1 If NP <=2, then convert the array into 1D and arrange the elements in ascending order. Remove the corrupted pixel from the array. Find the median of the remaining elements of the array. Replace the result value with the corrupted pixel Pij.

4.2.2 If 3<=NP<=5, then convert the array into 1D and arrange the elements in increasing order. Eliminate the noisy pixel from the array, find the mean of the remaining elements and replace the resultant value with the noisy pixel.

4.2.3 If NP >=6, then find the window's min, max and med values, where min is the window's minimum element, max is the window's maximum element and the med is the window's median. If max>med>min, then replace the median with Pij, else keep the original value and move to the next pixel by increasing the window size from 3x3 to 5x5.

Step 5: Move to the next pixel and repeat steps 1 to 4 for all the remanining pixels of the image.

Fig. 4.1 Algorithm ISMF

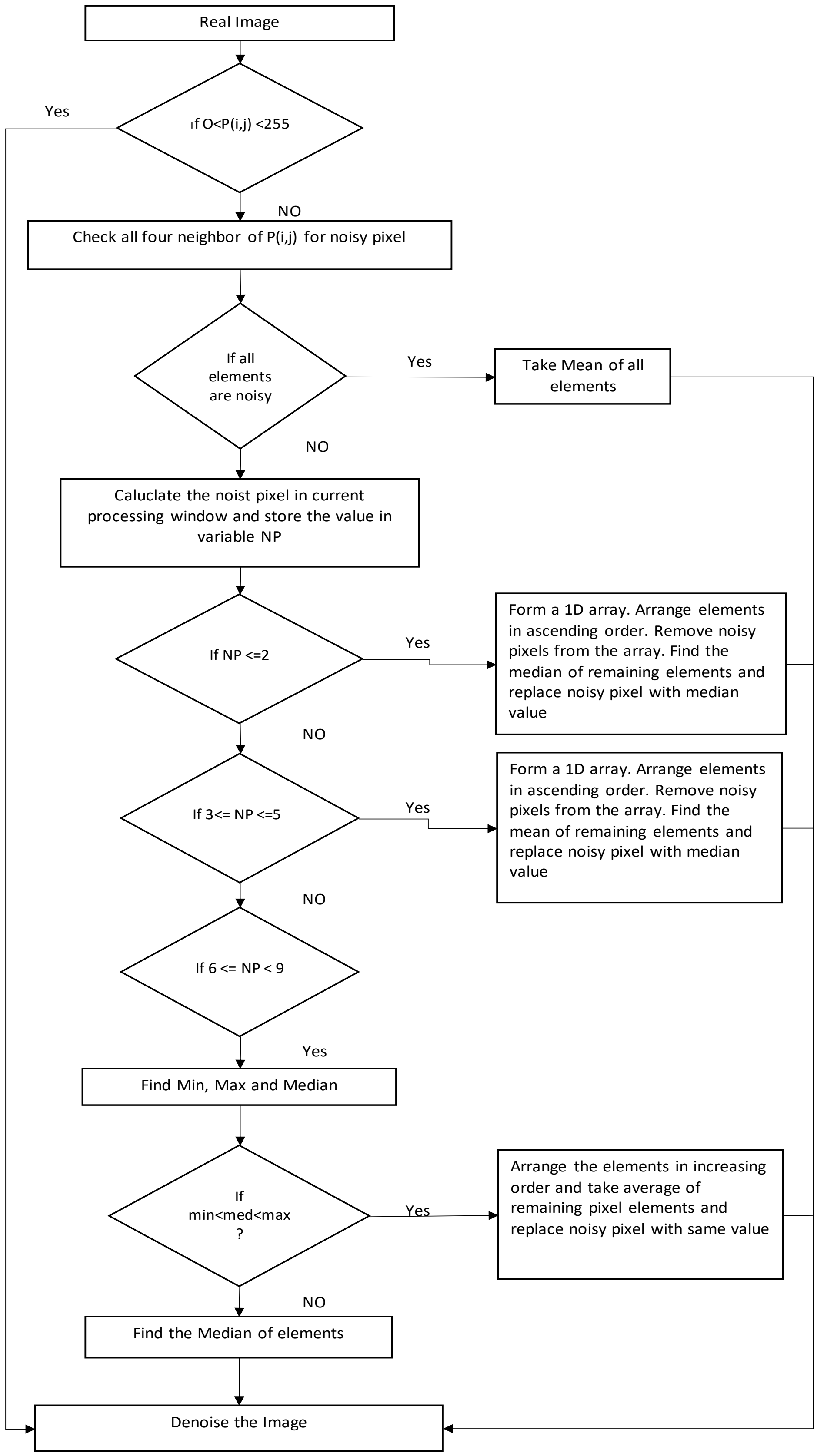


Fig 4.2 Flowchart of ISMF

4.2.2 Illustration of ISMF

As discussed, in ISMF algorithm every pixel of an image is checked for the presence of impulse noise. This algorithm is mainly divided into two phases. In the first phase, it checks whether each and every pixel is noisy or not, and if it is noisy then in next phase it checks for selecting a pixel value replaced with the noisy one. Different cases of ISMF algorithm are illustrated with examples.

Case 1: No noisy pixel (All Values in the window are Between 0 and 255)

$$\begin{bmatrix} 43 & 67 & 70 \\ 55 & < 90 > & 79 \\ 85 & 81 & 66 \end{bmatrix}$$

Fig. 4.3

In Fig. 4.3, the current pixel holds a value of 90. Its value lies between 0 and 255. It is a non noisy pixel and hence it does not require further processing.

Case 2. The processed pixel is noisy and all four neighbor pixels are also noisy.

$$\begin{bmatrix} 0 & 255 & 0 \\ 0 & < 255 > & 255 \\ 0 & 255 & 0 \end{bmatrix} \qquad \begin{bmatrix} 0 & 255 & 0 \\ 0 & < 191 > & 255 \\ 0 & 255 & 0 \end{bmatrix}$$

Fig. 4.4 (a) **Fig. 4.4 (b)**

In this example the pixel p(i,j) to be processed is noisy(i.e., 255) and all the 4 neighbor pixels are also noisy (255,0,255,255) as shown in Fig.4.4(a). As discussed in the algorithm, it will be replaced by the mean of all the elements. If we take the median of these elements, it will be again a noisy image. The mean in this example is (0+255+255+255)/4 = 191. The corrupted pixel is replaced by this value and the result has been shown in Fig. 4.4 (b).

Case 3: If P(i,j) is noisy and all four neighbor are not noisy and the number of noisy pixels in current processing window is : (NP) <=2.

$$\begin{bmatrix} 154 & 250 & 80 \\ 180 & < 0 > & 65 \\ 255 & 90 & 146 \end{bmatrix} \qquad \begin{bmatrix} 154 & 250 & 80 \\ 180 & < 145 > & 65 \\ 255 & 90 & 146 \end{bmatrix}$$

Fig. 4.5 (a) **Fig. 4.5 (b)**

As shown in Fig. 4.5(a) the current pixel is noisy (i.e, 0) and all four neighbor pixels are not noisy (250,180,90,65). According to step 4.2.1 of ISMF algorithm, First converting an array in 1D by arranging the elements in ascending order and secondly by removing the corrupted pixels the resulting array will be [0 65 80 90 146 154 180 255]. The value of P(i,j) would be then replaced by the median of the array that is 145. The result has been shown in Fig. 4.5 (b).

Case 4: If If P(i,j) is noisy and all four neighbor are not noisy. The number of noisy pixels

In current processing window is: 3<=NP<=5…

$$\begin{bmatrix} 154 & 255 & 80 \\ 255 & <0> & 65 \\ 0 & 90 & 146 \end{bmatrix} \qquad \begin{bmatrix} 154 & 255 & 80 \\ 255 & <107> & 65 \\ 0 & 90 & 146 \end{bmatrix}$$

Fig. 4.6 (a) **Fig. 4.6 (b)**

In this case, the P(i,j) is zero and two of the four neighbors are noisy and value of NP is four as shown in Fig. 4.6 (a). By arranging the elements in 1D array in ascending order and by removing the noisy pixel it will look like [65 80 90 146 154]. Find the mean of the remaining elements and it is to be replaced with noisy pixel. The mean is (65+80+90+146+154)/5 = 107 which will be replaced with the noisy pixel. The result has been shown in Fig. 4.6 (b)

Case 5: If If P(i,j) is noisy and all four neighbor are not noisy. The number of noisy pixels in current processing window is : NP>=6…

$$\begin{bmatrix} 255 & 85 & 255 \\ 0 & <0> & 68 \\ 0 & 255 & 0 \end{bmatrix} \qquad \begin{bmatrix} 255 & 85 & 255 \\ 0 & <68> & 68 \\ 0 & 255 & 0 \end{bmatrix}$$

Fig. 4.7 (a) **Fig. 4.7 (b)**

Here the current pixel P(i,j) is noisy (0) and two of the four neighbor are not noisy and NP=7 as shown in Fig. 4.7(a). Since NP=7 an algorithm finds the min, max and med values. In this case, min=0, max=255 and med are 68. As 0<68<255, the noisy pixel will be replaced by 68. The result has been shown in Fig. 4.7(b).

4.3 FEATURES OF THE PROPOSED ISMF

The ISMF adds features to the median filter and merges both mean and median filters to calculate a more accurate pixel value from noisy images. The following are some of the features of ISMF.

- **Noise Handling:** ISMF takes specific conditions into consideration for handling noisy pixels. In order to process and replace noisy pixel values more accurately, it takes into account the state of nearby pixels. This results in better noise reduction and improve performance.

- **Adaptive Window size:** The algorithm starts by taking the window size 3X3, but it can dynamically increase window size to 5X5 when the noisy pixel size NP>=6. This approach allows better noise reduction for highly corrupted images.

- **Local Structure Preservation:** The ISMF takes into consideration the minimum, maximum and median values when NP>=6, which is helpful to preserve edges while denoising an image. This functionality helps to preserve visual details that might be blur otherwise.

- **Noise level sensitivity:** The ISMF algorithm exhibits sensitivity to different noise levels by assessing the number of noisy pixels within the current processing window. Depending on the value of NP, it replaces the noisy pixel value by calculating either the mean or median of the remaining elements, ensuring appropriate denoising for different levels of noise.

4.4 EXPERIMENTAL RESULTS

To analyze the performance of the ISMF algorithm, noise density in images has been increased from 10% to 90%. The window size of 3X3 for all filters has been considered. The performance comparison of ISMF has been made with the original median filter using the matrices PSNR (peak signal-to-noise ratio) and MSE (Mean Square Error). PSNR and MSE are calculated using eq 4.1 and eq 4.2.

$$\text{PSNR} = 10\log_{10} X\,(\frac{255^2}{MSE}) \qquad (4.1)$$

$$\text{MSE} = \frac{1}{a*b}\sum_{i=0}^{a}\sum_{j=0}^{b}(Ni,j - Ci,j)^2 \qquad (4.2)$$

Here Ni,j and Ci,j denotes the original image and restored image, a and b are the width and height of an image.

PSNR is widely used metric for image processing to measure the quality of reconstructed image. MSE is a popular metric used for calculating the average squared difference between anticipated and actual data. The higher PSNR result indicates better image denoising. The lower MSE resultant pixel indicates better image quality.

Table 4.2 shows the PSNR and MSE for the ISMF and original median filter by increasing noise from 10% to 90% for TGFD images.

Table 4.2 Comparison of Median filter and ISMF on TGFD

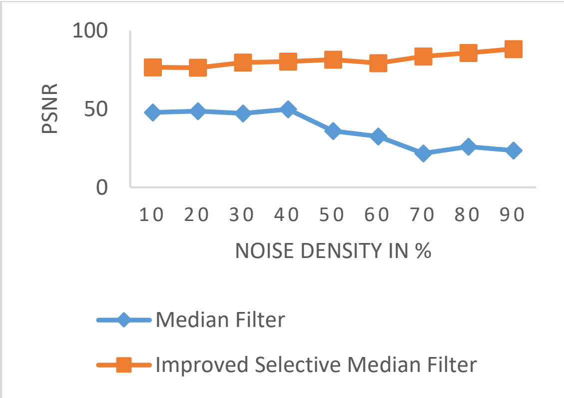
Noise Density in %	Median Filter		Improved Selective Median Filter	
	PSNR	MSE	PSNR	MSE
10	47.86	0.8755	76.48	0.0212
20	48.56	0.8466	76.25	0.0546
30	47.25	0.7855	79.54	0.0475
40	49.78	0.8965	80.25	0.0325
50	35.89	0.7845	81.48	0.0254
60	32.45	0.7895	79.25	0.0154
70	21.69	0.8566	83.45	0.0212
80	25.89	0.8854	85.78	0.0114
90	23.45	0.9014	88.23	0.011

From the results in Table 4.2, The value of PSNR starts decreasing and MSE value increase as the noise increases in the image for Median Filter. It can be observed from the Table 4.2 that the value of PSNR decreases and MSE increases drastically after 50% noise level in images. Hence, it is proved that the standard median filter does not give good results for images with more than 50% noise density.

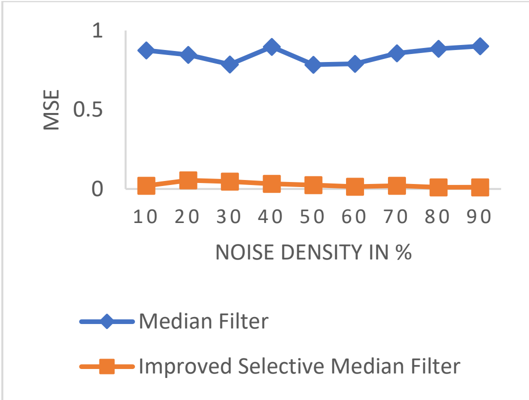
It can be observed from Table 4.2 that PSNR increases as the noise increases in images from 10% to 90 % for ISMF. It shows that the denoise image has low level of noise and better quality. MSE values decrease as the amount of noise increases for ISMF. It shows that the denoise image has less error between the original and reconstructed image.

It shows ISMF performs better than the standard median filter in each of the cases. As compared to the median filter the MSE for ISMF has decreased by 3 to 5 % which is remarkable.

Fig. 4.8 shows the graphical representation of both filters.



4.8 (a) PSNR performance for Different Noise



4.8 (b) MSE performance for Different Noise

Fig. 4.8 Comparison graph of PSNR and MSE for different noise for Food image

Following Fig.4.9 shows the image output after applying ISMF and Standard Median filter. It is clearly visible from Fig.4.8 and Fig.4.9 that ISMF performs better than the standard median filter. It also preserved the edges of food images and gives better visualization of images.

(a) Noisy Image (b) Standard Median Filter (c) ISMF

Fig. 4.9 Applying filters to noise images

Concluding Remarks: A new preprocessing technique, ISMF has been proposed and implemented. ISMF removes impulse noise from food images of TGFD, which gives a better result as compared to the existing median filter.

The next chapter discusses the implementation and results of simulation, transfer learning and fine tuning on five prebuilt CNN models for the proposed TGFD.