CHAPTER 3

IMPROVED ADAPTIVE FILTER

3.1 INTRODUCTION

Image denoising is the significant method used in pre-processing which is vital in image processing techniques. Usually, noise is the major basis for image degradation which occurs in acquisition, transmission or due to the faulty memory location present in hardware. The image acquisition and transmission processes will corrupt the image by introducing noises. The challenge here is to restore the contents of the image from the corrupted image by applying suitable technique. When the noise density is higher, the complexity of the restoring process will be high. Ultrasound images are affected by speckle noise. Impulse noise and Gaussian noise are present in satellite images. Also, the impulse and Gaussian noise are affecting the medical images like MRI and CT images. The noise density varies depending on various factors like reflective surfaces, atmospheric variations and so on. Noise reduction is a significant step in several applications such as pattern recognition, edge detection, data compression, image segmentation and feature extraction. The method of destroying the noise from corrupted colour image is a problematic job in image processing field as the restoration filter damage valuable information while removing noise.
The performance of the non-linear filter is efficient than the linear filter as it utilizes every color space to remove impulse noise. Removal of noise from images is a highly significant step in image processing because all the subsequent steps depend on the output of this stage. In 3D space, Compared to the gray scale image each pixel in color image is represented by three values. The problem here lies in the fact that, the removal of impulse noise from color images result in the loss of useful image components, rendering the restoration process inefficient (Lien et al. 2013).

3.2 STANDARD MEDIAN FILTER (SMF)

Standard Median Filtering is a simple rank selection filter otherwise named as median smoother which is used to eliminate the noise by varying the luminance assessment of the centre pixel within the window. It removes thin lines and blur image details at low noise densities. The filtered image from SMF is defined by the following equation,

\[ S(i, j) = \text{Median} \{(k, l) \in W_{mn} \{D(i + k, j + 1)\}\} \quad (3.1) \]

where, \( S(i, j) \)- Filtered Image

\( W_{mn} \) - Sliding window with the size of \( m \times n \) pixels which is centred at coordinates \((i, j)\).

\((k, l)\) – Filtered coordinates

\( D \) – Directionality

\( m \times n \) – Row and column value
The uncorrupted pixel intensity values are altered while reducing the corruption by SMF which fails to differentiate between uncorrupted pixels from corrupted pixels. Moreover, SMF requires a large filter size if the corruption level is high (Varade et al. 2013). A larger filter size of SMF introduced a significant image distortion. Equation (3.1) uses sorting algorithm like quick sort or bubble sort to arrange the samples in increasing or decreasing order. Sorting procedure requires long computational time when \( W_{mn} \) is a large filter due to the samples \( n_s = m \times n \).

**Issues in the existing systems:** The main drawback of the SMF is that the noisy pixels are substituted by individual median value in the vicinity without considering the local features (edges). Hence, if the noise density is high, then the performance of filters remains low. The method of removing impulse noise and Gaussian noise along with its implementation is straightforward and not satisfactory.

### 3.3 IMPULSE NOISE MODEL

Impulse noise is defined as an on-off noise which affects an image and makes changes in the intensity of pixels. When an image is affected by impulse noise, the homogeneity in pixels remains inaccurate and quality of an image is ruined. Impulse noise is produced while capturing images through noisy sensors at the time of image transmission through corrupted channel. So, removal of impulse noise from the degraded image is required to improve the quality by reducing the blurring effect to preserve the edge like fine details. Here, two different types of impulse noise in images are listed as stated below:

- Fixed valued impulse noise (FVIN) also named as Salt and Pepper Noise (SPN)
- Random Valued Impulse Noise (RVIN)
3.3.1 Fixed Valued Impulse Noise

The noisy pixels in FVIN is modelled by,

\[
(Y_{ij}) = \begin{cases} 
X_{i,j} \text{ with probability } p \\
(0, 255) \text{ with probability } 1 - p 
\end{cases}
\]  

(3.2)

where \(x(i, j)\)- Intensity value of original

\(Y(ij)\)- Intensity value of corrupted image at coordinate \((i, j)\) respectively and

\(p\)-Probability

This model implies that the pixels randomly get corrupted by two fixed extreme values 0 and 255(for 8-bit model with gay scale image is considered with same probability. The model is given below by,

\[
m = \begin{cases} 
X_{ij} \text{ with probability } P1 = p \\
(0, 255) \text{ with probability } P2 = 1 - p \\
(255 - m, 255) \text{ with the probability } P. 
\end{cases}
\]  

(3.3)

where \(P = P1 + P2\). Hence, this model is described as the Random valued Impulse noise.

3.3.2 Random Valued Impulse Noise

A window mask is moved across the observed image with the size of \((2N + 1)^2\), where \(N\) denotes the positive integer. Typically, centre element is deliberated as the pixel of interest. When the mask is moved starting from the left-top corner of the image to the right bottom corner, it performs certain arithmetical operations without any discrimination in a pixel. Detection is followed by identifying the noisy pixels and filtering those pixels. A mask is
relocated across the image, and the noisy pixels are detected. Then, the filtering operation is performed on those pixels which are pretended to be noisy. Removal of random-valued impulse noise is carried out in two stages such as detection of noisy pixel and replacement of that pixel. Median Filter is used as the backbone for removal of impulse noise. Many Filters with an impulse detector are used to remove impulse noise.

### 3.4 GAUSSIAN NOISE MODEL

Gaussian noise is the statistical noise which consists of probability density function (pdf) of the normal distribution (Gaussian distribution). Noise is revealed as the Additive White Gaussian Noise (AWGN), where all image pixels diverge from their original values with the Gaussian curve. For each image pixel with the intensity value \( f_{i,j} \) \((1 \leq i \leq m, 1 \leq j \leq n)\), the equivalent pixel of the noisy image \( g_{i,j} \) is given by,

\[
g_{i,j} = f_{i,j} + n_i
\]

\( n_i \)-noise drawn from a zero-mean Gaussian distribution.

### 3.5 ADAPTIVE MEDIAN FILTER (AMF)

Traditional median filter is not taken into consideration for how image characteristics vary from one location to another. It replaces every point in the image by the median of the corresponding neighborhood. In practice, adaptive filter that is capable of adapting their behavior depending on the characteristics of the image in the area being filtered, can produce more effective output image for some input noisy images. An adaptive median filter whose behaviour is changed based on statistical characteristics of the image inside the filter region can be defined by the \( m \times n \) rectangular windows. Like median filter, adaptive median filter also works in a rectangular window area.
$S_{xy}$. The adaptive median filter deviates the size of $S_{xy}$ during filter operation, based on numerous conditions. The output of the filter is a single value used to substitute the pixel value at $(x, y)$.

Consider the following notations:

$Z_{min} = \text{minimum intensity value in } S_{xy}$

$Z_{max} = \text{maximum intensity value in } S_{xy}$

$Z_{med} = \text{median of the intensity values in } S_{xy}$

$Z_{xy} = \text{intensity value at coordinates } (x, y)$

$S_{max} = \text{maximum allowed size of } S_{xy}$

The adaptive median filtering algorithm consists of two parts, denoted level A and level B:

**Level A**: If $Z_{min} < Z_{med} < Z_{max}$, go to level B. Else increase the window size. If window size $< S_{max}$, repeat level A. Else output $Z_{med}$

**Level B**: If $Z_{min} < Z_{xy} < Z_{max}$, Output $Z_{xy}$. Else output $Z_{med}$. Observing the algorithm, the purpose of level A is to determine if the median filter output, $Z_{med}$, is an impulse (black or white) or not. If the condition $Z_{min} < Z_{med} < Z_{max}$ is true, then $Z_{med}$ cannot be an impulse according to the noise theory.

In this case, go to level B and test if the point in the centre of the window, $Z_{xy}$, is itself an impulse. If the condition $Z_{min} < Z_{xy} < Z_{max}$ is true, then $Z_{xy}$ cannot be an impulse. In this case, the algorithm gives the unchanged pixel value as output, $Z_{xy}$ without changing "intermediate-level"
points present in the image. If the condition $Z_{min} < Z_{xy} < Z_{max}$ is false, then either $Z_{xy} = Z_{min}$ or $Z_{xy} = Z_{max}$. In both case, the pixel value is an extreme value and the algorithm outputs the median value $Z_{med}$, from level $A$ is not noise impulse.

The adaptive median filter is mainly used to exclude the problems faced in previous filters. In the Adaptive median filter, the size of the window surrounding each pixel undergo changes among filters and the discrepancy rests on the pixel median in the window. When the median value is an impulse, the size of a window is prolonged. The additional handling is accomplished on the image part of the current window specifications. The centre pixel of the window is calculated to validate whether it is an impulse or not. If it is an impulse, then the pixel value in the filtered image exists in the window. When the centre pixel is not an impulse or gaussian, then the centre pixel value is reserved in the filtered image. Consequently, if the pixel is an impulse or Gaussian, the gray scale value of pixel present in the filtered image replicates the input image. Therefore, the adaptive median Filter resolves the elimination of the impulse noise from the image and drop in the distortion existing in the image. This filter also filters out other categories of noise, providing a good output image than the standard median filter (Varade, et al. 2013).

3.6 IMPROVED ADAPTIVE MEDIAN FILTERING

With the intention to reduce the noise degradation, damage in quality and data loss, the overall flow of proposed system is explained in Figure 3.1 and steps are specified below.

Step 1 : Read the RGB input image and divide into Distinct RGB channels.
Step 2 : Add the impulse and Gaussian noise to the image (m by n).

Step 3 : After adding the noise to the temporary matrix, copy the input matrix into temp matrix. Now create the window matrix with size 3 by 3 with the elements. For example,

Consider a matrix \( A = \begin{bmatrix} 5 & 6 & 9 \\ 2 & 5 & 3 \\ 8 & 1 & 2 \end{bmatrix} \)

Step 4 : Now pad the matrix with zeroes on all sides

\[
\begin{bmatrix}
0 & 0 & 0 & 0 & 0 \\
0 & 5 & 6 & 9 & 0 \\
0 & 2 & 5 & 3 & 0 \\
0 & 8 & 1 & 2 & 0 \\
0 & 0 & 0 & 0 & 0
\end{bmatrix}
\]

A=

Step 5 : Consider a window of size 3 by 3. The window is of any size. Initially, starting from the matrix \( A \) \((1,1)\) place the window,

\[
\text{Window} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 5 & 6 \\ 0 & 2 & 5 \end{bmatrix}
\]

Step 6 : The value of the mid element is changed to the \([\text{Value of } A(2,2)]\)

Step 7 : Sort the window matrix: \( A_{\text{sort}} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 6 \\ 0 & 2 & 5 \end{bmatrix} \)

Step 8 : After sorting, the specific output matrix is placed with a value of 0 at (2, 2) pixel position.

Step 9 : The output pixel value is used by the neighbourhood pixel median.

Step 10 : This procedure is now repeated for all the values in the input matrix by sliding the window to next position \( A \) \((1,2)\) and so on.
Step 11: The obtained final output matrix is \( A_{f\text{mat}} = \begin{bmatrix} 0 & 3 & 0 \\ 2 & 5 & 2 \\ 0 & 2 & 0 \end{bmatrix} \)

Step 12: Return the original restored image

Figure 3.1 Flow Diagram of proposed Improved Adaptive Median Filter
3.6.1 Primary Processing

RGB Model

An RGB color space is the combination of three colorants for red, green and blue. The color remains as the continuous signal of electromagnetic wavelength radiations. The visible range lies within the 380 nm to 790 nm. Most of the display system uses three channels to render the color image like Red, Green, and Blue in RGB color spaces.

All color spaces are derived from the RGB information provided by the devices such as cameras and scanners. In every instance, color space is selected for application-specific reasons. The Red, Green and Blue (RGB) color space are generally used in the progression of computer graphics. Red, green and blue are the three primary additive colors where each color usually ranges from 0 to 255, \( R=\{0,1,2,...255\} \), \( G=\{0,1,2,...255\} \) and \( B=\{0,1,2,...255\} \). Individual components are denoted by a three-dimensional, RGB color cube. In RGB color cube, each point denotes the combination of maximum and minimum emission of each primary. When the amount of three Red, Green and Blue are in minimum levels, then the black color is produced. The white color is created when the aggregates of three primaries attain maximum levels.

Using Red, Green and Blue, desired colors are created, and RGB model is efficient in dealing with real-world images. All the three components require equal bandwidth to generate arbitrary colors within the RGB color cube as shown in Figure 3.2 and Figure 3.3. Also, processing image in RGB color space is usually the inefficient methodology to modify the intensity or color of a given pixel. All the three RGB values are delivered, changed and written to the frame buffer. The process remains faster when the system had access to the image stored in intensity and color format (Nishad 2013).
Whenever RGB image is converted into gray scale, certain loss of information occurs which seems difficult to get back RGB image from the gray image. In the primary processing, there exist three images by pre-allocating the original image pixels with zeroes and then put R, G, B components separately to each image. RGB color bar is shown in Table 3.1.

![RGB Color Cube](image)

**Figure 3.2 RGB color cube**

![Red, Green, Blue Additive Colors](image)

**Figure 3.3 Red, Green, Blue -Additive Colors**
Table 3.1 100% RGB Color bar

<table>
<thead>
<tr>
<th>Color</th>
<th>R</th>
<th>G</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal Range</td>
<td>0 to 255</td>
<td>0 to 255</td>
<td>0 to 255</td>
</tr>
<tr>
<td>White</td>
<td>255</td>
<td>255</td>
<td>255</td>
</tr>
<tr>
<td>Yellow</td>
<td>255</td>
<td>255</td>
<td>0</td>
</tr>
<tr>
<td>Cyan</td>
<td>0</td>
<td>255</td>
<td>255</td>
</tr>
<tr>
<td>Green</td>
<td>0</td>
<td>255</td>
<td>0</td>
</tr>
<tr>
<td>Magenta</td>
<td>255</td>
<td>0</td>
<td>255</td>
</tr>
<tr>
<td>Red</td>
<td>255</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Blue</td>
<td>0</td>
<td>0</td>
<td>255</td>
</tr>
<tr>
<td>Black</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

3.7 ALGORITHM DESCRIPTION FOR IAF

Table 3.2 Symbol description for IAF

<table>
<thead>
<tr>
<th>Symbols</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( Z_{\text{min}} )</td>
<td>minimum gray level value in ( S_{xy} )</td>
</tr>
<tr>
<td>( Z_{\text{max}} )</td>
<td>maximum gray level value in ( S_{xy} )</td>
</tr>
<tr>
<td>( Z_{\text{med}} )</td>
<td>median of gray levels in ( S_{xy} )</td>
</tr>
<tr>
<td>( Z_{xy} )</td>
<td>gray level at coordinates ((x, y))</td>
</tr>
<tr>
<td>( S_{\text{max}} )</td>
<td>maximum allowed size of ( S_{xy} )</td>
</tr>
</tbody>
</table>

Improved Adaptive Median Filter

Step 1 : Copy the input image matrix into Temp
Step 2 : \( \text{imgProcessed} = \text{Temp(size(Image))} \);

Step 3 : Initiate filtering Process

Step 4 : \( \text{for } i = 3 \text{ to corner_mat_value} \)

Step 5 : create ones(8) intn ZMIN, ZMAX, ZMED

\[
\begin{align*}
\text{zmin} &= \text{ordfilt2(Image,1,ones(i,i),'symmetric')} ; \\
\text{zmax} &= \text{ordfilt2(Image,i * i,ones(i,i),'symmetric')} ; \\
\text{zmed} &= \text{medfilt2(Image,[i i],'symmetric')} ;
\end{align*}
\]

Step 6 : Calculate Processed B value = (zmed > zmin) \& (zmax > zmed) \& \ldots \sim \text{imgProcessed}

Step 7 : \( zB = (g > \text{zmin}) \& (\text{zmax} > g) ; \)

Step 8 : \( \text{outputZxy} = \text{processUsingLevelB} \& zB ; \)

Step 9 : \( \text{outputZmed} = \text{processUsingLevelB} \& \sim zB ; \)

Step 10 : \( \text{Temp(outputZxy)} = g(outputZxy) ; \)

Step 11 : \( \text{Temp(outputZmed)} = \text{zmed(outputZmed)} ; \)

Step 12 : if all pixel of \( \text{imgProcessed} \)

Step 13 : return Noise removed image

The adaptive median filter also uses the noise detection and filtering algorithms to eliminate noise. The window size is increased only if the specified condition does not meet. If the condition is met, the pixel is filtered using the median of the window. Let \( Z_{ij} \) be the pixel of the corrupted image, \( Z_{min} \) be the minimum pixel value and \( Z_{max} \) be the maximum pixel value in the window, \( W \) be the current window size applied, \( W_{max} \) be the maximum window size that can be reached and \( Z_{med} \) be the median of the window assigned. Then, the algorithm of this filtering technique completes in two levels as described:
Level A:

a) If \( Z_{min} < Z_{med} < Z_{max} \), then the median value is not an impulse, so the algorithm goes to Level B to check if the current pixel is an impulse.

b) Else the size of the window is improved better and Level A is recurrent until the median value is not an impulse. So the algorithm reaches Level B; or the maximum window size is reached, in which case the median value is assigned as the filtered image pixel value.

Level B:

a) If \( Z_{min} < Z_{ij} < Z_{max} \), then the current pixel value is not an impulse, so the filtered image pixel is unchanged.

b) Else the image pixel is either equal to \( Z_{max} \) or \( Z_{min} \) (corrupted), then the filtered imaged pixel is assigned the median value from Level A.

These kinds of median filters are generally used in filtering image which is denoised with noise density superior than 20%. In every RGB component, the noise reduction stage is divided into three distinct steps namely,

- Noise Estimation
- Calculation of the degree of corruption
- Image Restoration

3.7.1 Noise Estimation

The noise of corrupted image is computed by information obtained from the neighboring pixel. The primary step is to determine the centre pixel
\( J(i, j) \) of a \( 3 \times 3 \) or \( 5 \times 5 \) filter window which lies in the trimming range. The pixel is corrupted and need to get replaced. At first, a \( 3 \times 3 \) filter is used to define the window within maximum and minimum grayscale values as \( g_{\min} \) and \( g_{\max} \) respectively. Then, the number of pixels present within the window is counted as \( K \), if

\[
[J(i + n, j + m)/4] \neq [g_{\max}/4] \quad \text{or} \quad [g_{\min}/4] \quad (3.5)
\]

where \( m, n \in [-1, +1] \). If \( k = 0 \), then repeat the above process for window size \( 5 \times 5 \) and \( m, n \in [-2, +2] \).

If \( K > 0 \), the centre pixel \( J(i, j) \) is not present in the trimming range, then the estimated value of this pixel is said to be median value \( K \) pixels. Or else, if \( K = 0 \), then the pixel is located in trimming range and estimated value is same as the average value of four neighboring pixels (Rahman & Uddin 2014).

### 3.7.2 Calculation of the Degree of Corruption

Based on the degree of corruption, the adaptive membership value is calculated. For each membership value the corrupted pixel is determined using the given equation.

\[
\mu[w(i, j)] = \max \left( 1 - \frac{D}{E(i, j)} \left| J(i, j) - E(i, j) \right| \right) \quad (3.6)
\]

where, \( \mu[w(i, j)] \) = Corrupted pixel

\[
D = \text{median} \left( \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} \left| J(i, j) - E(i, j) \right| \right) \quad (3.7)
\]

\( J(i, j) \) - Centre pixel

\( E(i, j) \) - Nearest edge pixel
The transformation in the intensity value of noisy pixel makes the estimated value as zero when the pixel remains noise free.

3.7.3 Image Recovery from Degraded Observation

Image recovery is the method used to recover and improve an original image from the degraded observations. It is a subset of Inverse problems and the values of a certain set of functions are estimated from known properties of other functions. The image degradation and relevant restoration is depicted in Figure 3.4 and noise part of entire degradation is dealt in Figure 3.5. Filter generates the following output based on the membership values

\[ Y(i, j) = E(i, j) + \mu[w(i, j)] \times [J(i, j) - E(i, j)] \]  

(3.8)

where \( Y(i, j) \)- Noise free pixels remain unaltered.

Finally, the concatenation of R, G, B components (image) is applied to all denoised color components to generate the desired output image.

![Diagram of Image degradation and restoration model](image)

**Figure 3.4 Image degradation and restoration model**
3.7.4 Image Quality

The foremost goal is to ensure the quality of output and develop the efficient methods for minimizing the visual impact of degradation. The source of distortion is ranged from motion blurring, sensor inadequacy and Gaussian noise. To improve the performance of the visual information acquisition, transmission, processing and storage systems, it is significant to enhance the qualities of the image before transmission. A degraded image is quantified by comparing it with the original and uncorrupted image. To measure the quality of obtained image, the PSNR values and MSE values are compared.

3.8 SIMULATION RESULTS

Real time images contain noises like impulse noise, Gaussian noise, etc. These noises exist because of image acquisition and transmission and many Benchmark images are used here such as
• Lena
• Barbara
• Camera man
• Building
• Girl
• Hat
• Monarch
• House
• Parrot
• Pens
• Pepper
• Sails and
• Window.

Depending on the amount of smooth and rich details available in the image, the complexity of de-noising algorithm will vary. Here, list of input images taken for experimentation is denoted in Figure 3.6. The input images are added with impulse noise of 20% as described in Figure 3.7. The obtained restored image with good quality after removing noise by filtering method is indicated in Figure 3.8. Likewise, Input image corrupted with Gaussian noise is described in Figure 3.9. Then, the original image is restored by removing the Gaussian noise of 20% with the help of improved adaptive filtering technique is shown in Figure 3.10.
Figure 3.6 Input images for IAF

Figure 3.7 Images affected by Impulse noise for IAF
Figure 3.8 Restored images from Impulse noise for IAF

Table 3.3 Parametric analysis of IAF for images affected by Impulse noise

<table>
<thead>
<tr>
<th>File Name</th>
<th>For Impulse Noise</th>
<th>PSNR in dB</th>
<th>MSE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Noisy image</td>
<td>Restored image</td>
</tr>
<tr>
<td>Barbara.bmp</td>
<td>34.0997</td>
<td>39.4231</td>
<td>75.4988</td>
</tr>
<tr>
<td>Bike.bmp</td>
<td>34.1096</td>
<td>39.7777</td>
<td>75.8299</td>
</tr>
<tr>
<td>Building.bmp</td>
<td>34.1003</td>
<td>41.8785</td>
<td>76.0351</td>
</tr>
<tr>
<td>Girl.bmp</td>
<td>34.1268</td>
<td>42.9182</td>
<td>75.1263</td>
</tr>
<tr>
<td>Hat.bmp</td>
<td>34.1097</td>
<td>43.5084</td>
<td>75.5494</td>
</tr>
<tr>
<td>Lena.bmp</td>
<td>34.0637</td>
<td>41.8335</td>
<td>76.1526</td>
</tr>
<tr>
<td>Monarch.bmp</td>
<td>34.0756</td>
<td>43.4611</td>
<td>75.973</td>
</tr>
<tr>
<td>Parrot.bmp</td>
<td>34.1232</td>
<td>44.8272</td>
<td>75.2205</td>
</tr>
<tr>
<td>Pens.bmp</td>
<td>34.1584</td>
<td>42.2759</td>
<td>74.4848</td>
</tr>
<tr>
<td>Pepper.bmp</td>
<td>34.3172</td>
<td>41.2956</td>
<td>72.0272</td>
</tr>
<tr>
<td>Window.bmp</td>
<td>34.1108</td>
<td>42.7031</td>
<td>75.7066</td>
</tr>
<tr>
<td>sails.png</td>
<td>34.0626</td>
<td>39.1775</td>
<td>76.1027</td>
</tr>
</tbody>
</table>
Table 3.3 shows the list of input images (Impulse noise) with specific PSNR value and MSE. The PSNR values of image are compared in two stages. Image obtained after applying the filtering technique for noise removal is named as restored image. Consider the image Barbara, this image exhibits the PSNR and MSE values correspondingly. Noisy image consists of 34.0997 dB PSNR value and 39.4231 dB PSNR value for restored image. Barbara restored image shows 13.5032% improvement when compared with the noisy image.

Figure 3.9 Gaussian noisy images
Figure 3.10 Restored image from Gaussian noise for IAF

Table 3.4 Parametric analysis of IAF for images affected by Gaussian noise

<table>
<thead>
<tr>
<th>File Name</th>
<th>For Gaussian Noise</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>PSNR in dB</td>
<td>MSE</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Noisy image</td>
<td>Restored image</td>
<td></td>
</tr>
<tr>
<td>Barbara.bmp</td>
<td>28.7919</td>
<td>29.006</td>
<td>15.7468</td>
<td></td>
</tr>
<tr>
<td>Bike.bmp</td>
<td>28.7977</td>
<td>28.6464</td>
<td>8.1013</td>
<td></td>
</tr>
<tr>
<td>Building.bmp</td>
<td>28.7944</td>
<td>28.9584</td>
<td>15.8756</td>
<td></td>
</tr>
<tr>
<td>Girl.bmp</td>
<td>28.8189</td>
<td>29.0297</td>
<td>14.4782</td>
<td></td>
</tr>
<tr>
<td>Hat.bmp</td>
<td>28.8324</td>
<td>28.9958</td>
<td>13.2421</td>
<td></td>
</tr>
<tr>
<td>Lena.bmp</td>
<td>28.7729</td>
<td>28.8527</td>
<td>7.3676</td>
<td></td>
</tr>
<tr>
<td>Monarch.bmp</td>
<td>28.7561</td>
<td>28.8658</td>
<td>9.8931</td>
<td></td>
</tr>
<tr>
<td>Parrot.bmp</td>
<td>28.8119</td>
<td>28.9117</td>
<td>10.933</td>
<td></td>
</tr>
<tr>
<td>Pens.bmp</td>
<td>28.8281</td>
<td>29.2052</td>
<td>24.7442</td>
<td></td>
</tr>
<tr>
<td>Pepper.bmp</td>
<td>29.0084</td>
<td>28.2094</td>
<td>17.5015</td>
<td></td>
</tr>
<tr>
<td>Window.bmp</td>
<td>28.9484</td>
<td>28.9484</td>
<td>11.4167</td>
<td></td>
</tr>
<tr>
<td>sails.png</td>
<td>28.7689</td>
<td>28.7689</td>
<td>2.4192</td>
<td></td>
</tr>
</tbody>
</table>
Table 3.4 shows the list of input images (Gaussian noise) with specific PSNR value and MSE. The PSNR values of image are compared in two stages. The noisy images after removal of the noise are named as restored image. The noisy Benchmark (Barbara) image consists of 28.7919 dB PSNR value and restored image shows 29.006 dB PSNR value for restored image. Barbara restored image shows 0.7461 % improvement when compared with the noisy image.

3.9 SUMMARY

This chapter reviews the impact of noises present in the image and demonstrated the denoising concept with the help of filters. The noise present in the image leads to loss of sensual data, image degradation and corruptions. In order to overcome this situation, an algorithm is proposed to process the corrupted images by removing the noisy pixels. The importance of adaptive median filter is discussed when compared to standard median filter. Typical noise models are analysed followed by detailed description of modules present in the proposed system. An uncorrupted image is obtained after denoising is compared with the noise-free version of original image. The Mean Square Error (MSE) and the Peak-Signal to Noise Ratio (PSNR) are used to measure the quality restored image. To further enhance the feature of the images resulting in the restoration of complete original image, the upcoming chapter deals with the improved weighted average filtering.