# BIT MESRA

# DEPARTMENT OF EEE

**SUBJECT**

EE305 Digital Signal Processing (DSP)

**PROJECT**

Analysis and working of following filters for image processing:

1. Decision-Based Coupled Window Median Filter
2. Switching-Based Median Filter
3. Noise Density Range Sensitive Mean Median Filter
4. Adaptive Median Filter
5. Dual Domain Image Denoising

**SUBMISSION TO**

DR. GAURI SHANKER GUPTA


**SUBMITTED BY**

DAKSH ABROL

# INTRODUCTION

1. Decision-Based Coupled Window Median Filter

The Decision-Based Coupled Window Median Filter is a time domain filter, based on the median filter algorithm. It works by calculating the median value of the neighboring pixels and compares it with the center pixel. If the absolute difference between them is less than a predefined threshold, then the pixel is considered not to be noisy, and the median of the window is used to replace the pixel value. However, if the difference is greater than the threshold, the window size is increased by a specified amount until the maximum window size is reached. The filter is efficient in removing impulse noise and preserving the image details, such as edges and textures, which are not well preserved in other median-based filters. One of the advantages of this filter is its ability to work well on images with different noise levels. It can be used in various image processing applications, such as medical images, remote sensing, and computer vision. However, one of the disadvantages of this filter is its long processing time, especially when dealing with large images.

2. Switching-Based Median Filter

The Switching-Based Median Filter is a time domain filter. It works by calculating the median of the pixel values in a neighborhood window around each pixel. If the absolute difference between the median value and the center pixel is less than a predefined threshold value, then the median value is used as the output pixel value. Otherwise, the original pixel value is used. The filter is efficient in removing impulse noise and preserving edges and details in the image. The main advantage of this filter is its simplicity and speed compared to other median-based filters. The filter can be applied in various image processing applications, such as digital photography, medical imaging, and video processing. However, one of its disadvantages is that it may not perform well when dealing with high-density noise, and it may result in the loss of fine details in the image.

3. Noise Density Range Sensitive Mean Median Filter

The Noise Density Range Sensitive Mean Median Filter is a time domain filter. It works by calculating the mean and median values of the pixel values in a neighborhood window around each pixel. Then, it calculates the range of pixel values in the window and the noise density of the window. The filter applies a threshold value to the center pixel of the window based on the noise density value. If the noise density is less than a predefined threshold, the mean value is used as the threshold value. If the noise density is greater than another predefined threshold, the median value is used as the threshold value. Otherwise, the threshold value is the average of the mean and median values. The filter is efficient in removing impulse noise, preserving image details, and reducing blur artifacts. The filter can be applied in various image processing applications, such as medical imaging, document processing, and surveillance. However, one of its disadvantages is that it may not perform well when dealing with certain types of noise, such as Gaussian noise.

4. Adaptive Median Filter

The Adaptive Median Filter is a time-domain filter. This filter removes noise by replacing each pixel with the median of the neighboring pixel values within a sliding window. The size of the window changes based on the range of pixel values within the window. The Adaptive Median Filter is efficient in reducing different types of noise such as salt and pepper noise and Gaussian noise. It is suitable for removing noise in images with a low peak-signal-to-noise ratio (PSNR). The main advantage of the Adaptive Median Filter is its ability to preserve edges and details in the image while removing noise. This filter is commonly used in image processing applications such as medical imaging, satellite imaging, and industrial inspection. However, the Adaptive Median Filter has some disadvantages. It is computationally expensive and can produce artifacts in images with complex structures.

5. Dual Domain Image Denoising Filter

The Dual Domain Image Denoising Filter is a both time domain as well as frequency-domain filter. It uses a Gaussian filter in the frequency domain and a bilateral filter in the spatial domain. This filter removes noise by applying a Gaussian filter to the Fourier transform of the image. The size of the Gaussian filter is determined by an internal parameter called sigma. The parameter sigma controls the amount of smoothing applied to the image. The Dual Domain Image Denoising Filter is efficient in removing Gaussian noise from images with a high PSNR. This filter is suitable for images with a high-frequency range, such as natural images, and is commonly used in image processing applications such as image and video compression. The main advantage of the Dual Domain Image Denoising Filter is its ability to preserve image details while removing noise. However, this filter may not be suitable for images with a low PSNR or images with non-Gaussian noise.

# LITERATURE REVIEW

1. Decision-based Coupled Window Median Filter

   The decision-based coupled window median filter is an effective method for reducing noise from pictures while keeping their key characteristics as intact. This filter is designed to reduce noise by merging the outputs of numerous median filters with varied window widths. Furthermore, the filter employs decision-based methods to select the best median filter output for each pixel, hence increasing the filter's overall performance as described in [1].

   The findings of numerous trials done in the research articles [1] [2] shows that the decision-based coupled window median filter outperforms other common noise reduction filters such as the median filter and the mean filter. The filter has been demonstrated to be effective in decreasing additive and impulsive noise in grayscale and color pictures.

   Overall, the decision-based coupled window median filter is a promising image denoising technology with applications in medical imaging, computer vision, and remote sensing. It achieves an excellent mix of noise reduction and feature retention and is simple to use in real-time applications.

2. Switching-based Median Filter

   The switching-based median filter removes high-density salt and pepper noise from digital photos effectively. The filter works by switching between two median filters with various window widths dependent on the amount of noise in the picture. The filter works effectively in both low and high noise environments, providing good noise reduction while keeping crucial picture elements as described in [3] [4].

   The research investigations [5] show that the switching-based median filter outperforms other common filters like the median filter and the mean filter. The filter has been demonstrated to be successful in decreasing various degrees of noise in grayscale and color pictures, including impulse noise and Gaussian noise.

   Furthermore, because the filter is adaptable to the noise level in the picture, it is suited for real-world applications where the noise level varies dramatically. The filter is simple to develop and may be used in real-time applications.

   In conclusion, the switching-based median filter provides an intriguing method for eliminating high-density salt and pepper noise from digital photos. The filter reduces noise while keeping critical visual elements and is ideal for real-world applications where noise levels fluctuate dramatically.

3. Noise Density Range Sensitive Mean Median Filter

The Noise Density Range Sensitive Mean-Median (NDRSMM) filter is a good way to remove impulsive noise from digital images. The noise density range is used by the NDRSMM filter to establish the proper weighting factors for the mean and median filter operations. The filter is meant to adapt to the image's fluctuating noise densities and provide better noise reduction than typical median filters as described in [6].

The research paper quoted [6] shows that the NDRSMM filter outperforms other popular filters such as the median filter, mean filter, and adaptive median filter shown in [2]. When compared to other filters, the filter is demonstrated to be successful in decreasing impulse noise in grayscale pictures, with superior noise reduction and edge retention.

Furthermore, the filter has the benefit of being adaptable to variable noise densities in the picture, making it suited for real-world applications where noise density varies dramatically. The filter is simple to develop and may be used in real-time applications.

In conclusion, the NDRSMM filter offers an intriguing method for reducing impulse noise from digital photographs. When compared to standard filters, the filter delivers better noise reduction and edge preservation, making it ideal for real-world applications where noise density varies dramatically.

4. Adaptive Median Filter

The adaptive median filter is an effective technique for eliminating impulsive noise from digital images. It is intended to maintain features and edges while efficiently decreasing noise as shown in [7]. The filter analyzes the surrounding pixels in a window and calculates the median value based on the intensity values. Based on the local picture properties, the window size and threshold value are changed mentioned in [8].

Several investigations have shown that adaptive median filters are good at minimizing impulsive noise. For example, the research work described in [7] suggests a new method for adaptive median filters that outperforms classic median filter described in [2] in terms of noise reduction. According to the findings, the suggested algorithm is successful in removing salt-and-pepper noise from a variety of photos, including medical images, satellite images, and natural images.

Other research [9] and [10] has proposed improvements to the adaptive median filter that increase noise reduction while keeping edges and features. The adaptive median filter's performance has been increased further, making it a popular choice for image denoising.

5. Dual Domain Image Denoising Filter

The dual-domain image denoising filter is a good way to reduce noise in digital images. This approach efficiently preserves edges and fine features while reducing noise by employing both the spatial and frequency domains. The filter operates by first converting the picture to the frequency domain and then using a noise reduction method. The output is then converted back into the spatial domain, and a median filter is performed to minimize residual noise even more as discussed in [11]. When compared to approaches that simply act in the spatial domain [2], the dual-domain approach provides more robust noise reduction.

Overall, the dual-domain image denoising filter is a promising method to noise reduction in digital pictures, especially where fine features and edges must be kept. Further investigation may explore the effectiveness of this filter on different types of noise and in different image contexts.

# EQUATIONS AND VARIABLES USED

| FILTER | VARIABLES: NAME | DEFAULT VALUE | DESCRIPTION | EQUATIONS |
|---|---|---|---|---|
| Decision-Based Coupled Window Median Filter | image | - | input image | Median value calculation: median(windowSum / windowCount) |
| | windowSize | 3 | size of the window used for filtering | |
| | threshold | 10 | threshold value used for filtering | |
| | windowSizeIncrement | 2 | increment value for window size | |
| | maxWindowSize | 9 | maximum window size | |
| | rows, cols | - | size of the image | |
| | output | - | output image | |
| | pixelValue | - | current pixel value being processed | |
| | currentWindowSize | - | current window size | |
| | windowSum | 0 | sum of pixel values within window | |
| | windowCount | 0 | number of pixels within window that meet threshold condition | |
| | currentPixelValue | - | pixel value being examined within window | |
| | rowIdx, colIdx | - | row and column indices of current pixel within window | |
| Switching-Based Median Filter | image | - | input image | Absolute difference calculation: abs(image(i, j) – median_val) |
| | window_size | 3 | size of the window used for filtering | |
| | threshold | 10 | threshold value used for filtering | |
| | rows, cols | - | size of the image | |

| | | | | |
|---|---|---|---|---|
| | output | - | output image | |
| | window | - | subset of image containing pixels within the window around the current pixel | |
| | median_val | - | median value of the pixels within the window | |
| | abs_diff | - | absolute difference between the current pixel and median value | |
| Noise Density Range Sensitive Mean Median Filter | image | - | input image | <u>Threshold value calculation</u>: thresholdVal = (meanVal + medianVal) / 2 |
| | m, n | - | size of the image | |
| | output | - | output image | |
| | w | - | 3x3 window used for filtering | |
| | s | - | sorted array of pixel values within the window | |
| | meanVal | - | mean value of the pixel values within the window | |
| | medianVal | - | median value of the pixel values within the window | |
| | rangeVal | - | range of pixel values within the window | |
| | noiseDensity | - | ratio of rangeVal to maximum pixel value within the window | |
| | thresholdVal | - | threshold value used for filtering the center pixel within the window | |
| | image | - | input image | |

<table>
<tr><td rowspan="9">Adaptive Median Filter</td><td>m, n</td><td>-</td><td>size of input image</td><td rowspan="2">Median value calculation:<br>medianVal = s(5) for 3x3 window<br><br>medianVal = s(13) for 5x5 window<br><br>medianVal = s(25) for 7x7 window</td></tr>
<tr><td>output</td><td>-</td><td>output image after applying the filter</td></tr>
<tr><td>i, j</td><td>-</td><td>index of the current pixel</td><td rowspan="7">Range value calculation:<br>rangeVal = max(s) - min(s) for each window.</td></tr>
<tr><td>w</td><td>-</td><td>window of size 3x3, 5x5, or 7x7 centered at the current pixel</td></tr>
<tr><td>s</td><td>-</td><td>sorted elements of the window w</td></tr>
<tr><td>medianVal</td><td>-</td><td>median value of the window</td></tr>
<tr><td>rangeVal</td><td>-</td><td>range of pixel values in the window</td></tr>
<tr><td>thresholdVal</td><td>-</td><td>threshold value calculated based on the range of pixel values</td></tr>
<tr><td colspan="4"></td></tr>
<tr><td rowspan="7">Dual Domain Image Denoising Filter</td><td>image</td><td>-</td><td>input image</td><td rowspan="2">Fourier transform of the input image:<br>F = fft2(I)</td></tr>
<tr><td>sigma</td><td>200</td><td>internal parameter used to adjust the filter strength</td></tr>
<tr><td>I</td><td>-</td><td>input image in double precision</td><td rowspan="2">Gaussian filter in the frequency domain:<br>G = exp(-(U.^2 + V.^2)/(2*sigma^2))</td></tr>
<tr><td>M, N</td><td>-</td><td>size of input image</td></tr>
<tr><td>F</td><td>-</td><td>Fourier transform of the input image</td><td rowspan="2">Filtered Fourier transform:<br>Ff = F .* G</td></tr>
<tr><td>u, v</td><td>-</td><td>frequency coordinates</td></tr>
<tr><td>idx, idy</td><td>-</td><td>indices for frequencies</td><td></td></tr>
</table>

| | | | | |
|---|---|---|---|---|
| | | | above the Nyquist limit | Inverse Fourier transform:<br>f = real(ifft2(Ff)) |
| | V, U | - | meshgrid of frequency coordinates | |
| | G | - | Gaussian filter in the frequency domain | Frequency coordinates calculation:<br>u = 0:(M-1), v = 0:(N-1), idx = find(u > M/2)<br><br>u(idx) = u(idx) - M<br>idy = find(v > N/2)<br><br>v(idy) = v(idy) - N, [V, U] = meshgrid(v, u) |
| | Ff | - | filtered Fourier transform | |
| | f | - | output image after applying the filter | |

# ALGORITHMS

A. Decision-Based Coupled Window Median Filter

```
Algorithm 1: Decision Based Coupled Window Median Filter Algo-
rithm
  Input  : Input image
  Output: Output image
image ← im2double(image); windowSize ← 3; threshold ← 10;
  windowSizeIncrement ← 2; maxWindowSize ← 9;
[rows, cols] ← size(image); output ← zeros(rows, cols);
for i ← 1 to rows do
  for j ← 1 to cols do
    pixelValue ← image(i, j); currentWindowSize ← windowSize;
     windowSum ← 0; windowCount ← 0; for k ←
     -currentWindowSize to currentWindowSize do
      for l ← -currentWindowSize to currentWindowSize do
        rowIdx ← i + k; colIdx ← j + l; if (rowIdx ≥ 1)  (rowIdx
         ≤ rows)  (colIdx ≥ 1)  (colIdx ≤ cols) then
          currentPixelValue ← image(rowIdx, colIdx); if
           (|currentPixelValue − pixelValue| < threshold)
           then
            windowSum ← windowSum + currentPixelValue;
             windowCount ← windowCount + 1;
          end
          if (windowCount > 0)  (windowCount ≤
           maxWindowSize) then
            output(i, j) ← median(windowSum /
             windowCount);
          end
          else
            if (currentWindowSize < maxWindowSize) then
              currentWindowSize ← currentWindowSize +
               windowSizeIncrement;
            end
          end
        end
      end
    end
  end
end
output ← im2uint8(output);
```

Fig. 1 – Pseudo-code algorithm for Decision-Based Coupled Window Median Filter.

This pseudo code algorithm implements a decision-based coupled window median filter for image processing. The algorithm takes an input image and applies a median filter using a sliding window approach. The window size is initially set to 3x3 and is incremented by 2 until it reaches a maximum size of 9x9. A threshold value is used to decide which pixels to include in the window. The median of the pixel values within the window is then computed and used to replace the center pixel value. The output image is returned in uint8 format. The algorithm loops through each pixel of the image and performs the median filtering operation using the defined window size and threshold values. The output image is then returned.

B. Switching-Based Median Filter

**Algorithm 1:** Switching Based Median Filter

```
Input  : Image image
Output: Filtered image output
image ← im2double(image); window_size ← 3; threshold ← 10;
  [rows, cols] ← size(image); output ← zeros(rows, cols);
for i ← 1 to rows do
    for j ← 1 to cols do
        window ← image(max(i-window_size, 1) :
          min(i + window_size, rows),              max(j −
          window_size, 1) : min(j + window_size, cols));median_val ←
          median(window(:))
        abs_diff ← abs(image(i,j) - median_val)
        if abs_diff < threshold then
            output(i, j) ← median_val
        end
        else
            output(i, j) ← image(i,j)
        end
    end
end
output ← im2uint8(output);
```

Fig. 2 – Pseudo-code algorithm for Switching-Based Median Filter.

This pseudo code algorithm implements a switching-based median filter for image processing. The algorithm takes an input image and applies a median filter using a sliding window approach. The window size is set to 3x3 and a threshold value of 10 is used to determine whether to use the median value or the original pixel value as the output pixel value. The output image is returned in uint8 format. The algorithm loops through each pixel of the image and performs the median filtering operation using the defined window size and threshold value. The output pixel value is set to the median value if the absolute difference between the median value and the current pixel value is less than the threshold value. Otherwise, the output pixel value is set to the original pixel value. The output image is then returned.

C. Noise Density Range Sensitive Mean Median Filter

**Algorithm 1:** Noise Density Range-Sensitive Mean-Median Filter

```
Input  : input image image
Output: filtered image output
image ← convert image to double;
output ← zeros(size(image));
m, n ← size(image);
for i ← 1 to m do
    for j ← 1 to n do
        if i = 1 or j = 1 or i = m or j = n then
            output(i, j) ← image(i, j);
        end
        else
            w ← 3x3 window centered at (i, j) in image;
            s ← sort elements of w in ascending order;
            meanVal ← mean(s);
            medianVal ← median(s);
            rangeVal ← max(s) - min(s);
            noiseDensity ← rangeVal/(max(s) + ϵ);
            if noiseDensity < 0.15 then
                thresholdVal ← meanVal;
            end
            else if noiseDensity > 0.35 then
                thresholdVal ← medianVal;
            end
            else
                thresholdVal ← (meanVal + medianVal)/2;
            end
            output(i, j) ← apply thresholdVal to center pixel of w;
        end
    end
end
output ← convert output to uint8;
```

Fig. 3 – Pseudo-code algorithm for Noise Density Range-Sensitive Mean-Median Filter

This pseudo code algorithm implements a noise-density range-sensitive mean-median filter, which is used to filter out noise from an image. The algorithm works by sliding a 3x3 window over the image and calculating the mean and median values of the pixel intensities in the window. The range of pixel values in the window is also calculated to determine the noise density. Based on the noise density, a threshold value is calculated, which is used to filter out noise from the center pixel of the window. If the noise density is low, the threshold value is set to the mean value. If it is high, the threshold value is set to the median value. If it is in between, the threshold value is set to the average of the mean and median values. Finally, the output image is returned in uint8 format.

D. <u>Adaptive Median Filter</u>

**Algorithm 1:** Adaptive Median Filter

```
Input  : Image image
Output: Filtered image output
image ← im2double(image);
m, n ← size(image);
output ← zeros(m, n);
for i ← 1 to m do
    for j ← 1 to n do
        if i = 1 or j = 1 or i = m or j = n then
            output(i, j) ← image(i, j);
        end
        else
            w ← image(max(1,i-1):min(m,i+1),max(1,j-1):min(n,j+1));
            s ← sort(w(:));
            medianVal ← s(5);
            rangeVal ← max(s) − min(s);
            if rangeVal < 20 then
                thresholdVal ← medianVal;
            end
            else
                w ←
                  image(max(1,i-2):min(m,i+2),max(1,j-2):min(n,j+2));
                s ← sort(w(:));
                medianVal ← s(13);
                rangeVal ← max(s) − min(s);
                if rangeVal < 20 then
                    thresholdVal ← medianVal;
                end
                else
                    w ← image(max(1,i-3):min(m,i+3),max(1,j-
                      3):min(n,j+3));
                    s ← sort(w(:));
                    medianVal ← s(25);
                    thresholdVal ← medianVal;
                end
            end
            output(i, j) ← thresholdVal;
        end
    end
end
output ← im2uint8(output);
```

Fig. 4 – Pseudo-code algorithm for Adaptive Median Filter

This pseudo code algorithm for Adaptive Median Filter takes an input image, converts it to a double-precision matrix, and initializes the output matrix with zeros. It then applies a 3x3 window around each pixel in the image and calculates the median value of the window. The range of pixel values in the window is then calculated, and a threshold value is determined based on this range. If the range is less than 20, the median value is used as the threshold. If the range is greater than 20, a larger window is used to calculate the median value, and this value is used as the threshold.

The threshold value is then applied to the center pixel of the window, and the resulting pixel value is stored in the output matrix. The algorithm iterates through each pixel in the image, applies the filter, and stores the filtered image as output. Finally, the output matrix is converted to an unsigned 8-bit integer and returned.

E. Dual Domain Image Denoising Filter

**Algorithm 1:** Dual domain image denoising filter

**Input** : Grayscale image *image*

**Output:** Denoised image *output*

```
σ ← 200          // internal parameter (can be changed, based on image)
I ← im2double(image)       // convert image to double precision
[M, N] ← size(I)               // compute the size of the image
F ← fft2(I)      // compute the Fourier transform of the image
u ← 0 : (M − 1); v ← 0 : (N − 1); idx ← find(u > M/2);
  u(idx) ← u(idx) − M; idy ← find(v > N/2); v(idy) ← v(idy) − N;
  [V, U] ← meshgrid(v, u)    // compute the frequency coordinates
G ← exp(−(U^2+V^2)/(2σ^2))      // construct the Gaussian filter in the frequency domain
Ff ← F · G    // apply the filter to the Fourier transform of the image
f ← real(ifft2(Ff))   // compute the inverse Fourier transform
output ← f(1 : M, 1 : N)       // crop the result to the original size
output ← im2uint8(output)        // convert the output to uint8
return output
```

Fig. 5 – Pseudo-code algorithm for Dual Domain Image Denoising Filter.

The pseudo code algorithm is a Decision-Based Coupled Window Median Filter which takes an input image and returns the output image. The algorithm defines the window size, threshold, window size increment, and maximum window size. For each pixel in the image, the algorithm initializes the window size, window sum, and window count. Then, it loops through each pixel in the window and checks if the difference between the current pixel value and center pixel value is less than the threshold. If yes, then it adds the current pixel value to the window sum and increments the count. If the window count is greater than zero and less than or equal to the maximum window size, it calculates the median of pixels in the window and sets the output pixel value to median value. Otherwise, it increases the window size by the increment amount if less than the maximum window size. Finally, it returns the output image.

# RESULTS AND DISSCUSSION

1. Decision-Based Coupled Window Median Filter

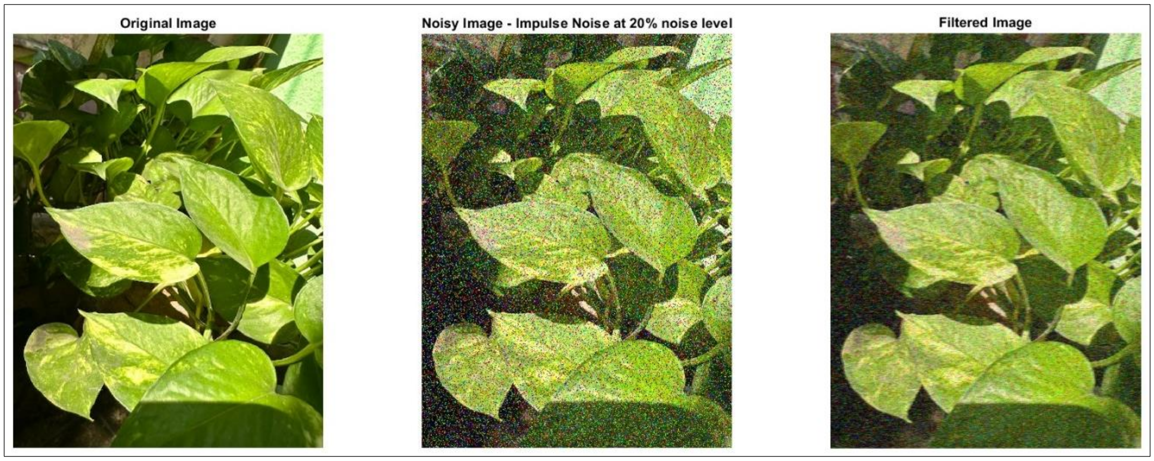


Fig. 6: Denoising leaves.jpg at 20% noise level (Impulse Noise) using Decision-Based Coupled Window Median Filter

| | MSE | RMSE | PSNR | SSIM | IEF |
|---|---|---|---|---|---|
| Fig. 6 | 1242.071544 | 35.243035 | 17.189337 | 0.636196 | 3.800423 |

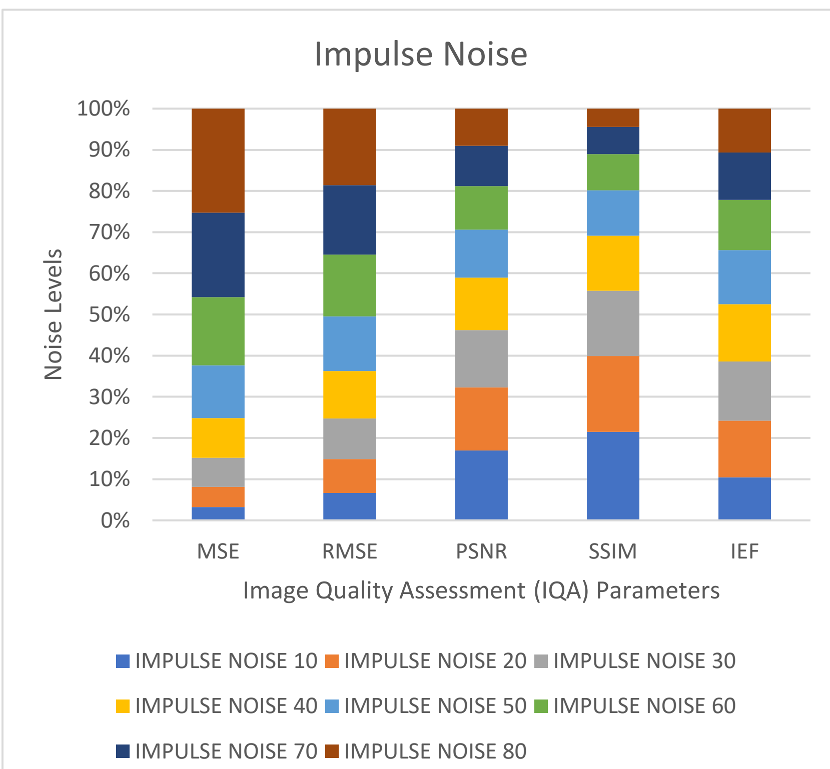


Fig. 7 – Image Quality Assessment (IQA) Parameters of Impulse Noise for leaves.jpg using Decision-Based Coupled Window Median Filter

The results are depicted by performing the filtering technique on leaves.jpg, which is already denoised using various noises.

TABLE 1 describes IQA parameters for all images when the Noise Level is set to 20% and the Noise Type being Impulse Noise. Of all the images at this configuration, it can be easily seen that Decision-Based Coupled Window Median Filter works best on coins.jpg for sure. Reason being lowest MSE i.e., 779.203896 and RMSE i.e., 27.914224 compared to satellite1.jpg which has the highest MSE and RMSE value of 1672.897475 and 40.901069 respectively. The PSNR value of coins.jpg is highest of all i.e., 19.214292.

Another conclusion can be made out of TABLE 2, where different noises are added into an image with different Noise Levels. leaves.jpg was added with various noises at various noise levels. Keeping the base at Noise Levels it can be drawn into conclusion that at specific Noise Level percentages there are specific noises which are best filtered out by the performing filter. Here, for example it is seen that at 10% Noise impunity, Speckle Noise is the best filtered out of all other noises that were added into the image. Similarly, for 20% it is Speckle and so on. This gives a clear idea about the denoising capability of the filter over various noise levels.

| | MSE | RMSE | PSNR | SSIM | IEF |
|---|---|---|---|---|---|
| train.jpg | 1604.119863 | 40.051465 | 16.078435 | 0.329607 | 2.733363 |
| coins.jpg | 779.203896 | 27.914224 | 19.214292 | 0.139471 | 4.854442 |
| leaves.jpg | 1241.032557 | 35.228292 | 17.192972 | 0.636543 | 3.805798 |
| scenery.jpg | 1074.453733 | 32.778861 | 17.818926 | 0.735278 | 3.896702 |
| satellite1.jpg | 1672.897475 | 40.901069 | 15.89611 | 0.083314 | 3.168598 |
| satellite2.jpg | 1515.433054 | 38.928563 | 16.325436 | 0.120026 | 2.951995 |

TABLE 1: IQA Parameters for all images at 20% noise level (Impulse Noise) using Decision-Based Coupled Window Median Filter

This Decision-Based Coupled Window Median filter has been tested over various images. For reference leaves.jpg has been selected. The following noises have been specifically inserted into the images: Gaussian Noise, Impulse Noise, Periodic Noise, Poisson Noise, Speckle Noise, Gamma Noise, Rayleigh Noise, Quantization Noise, Brownian Noise and Rician Noise. Also, noise levels are also taken into account starting from 10% to 80% with 7 equal intervals starting from 10% for all mentioned noise types and the conclusion can be drawn out into the TABLE 2.

| Noise Level (%) | Noise Type |
|---|---|
| 10 | Speckle |
| 20 | Speckle |
| 30 | Speckle |
| 40 | Poisson |
| 50 | Poisson |
| 60 | Speckle |
| 70 | Poisson |
| 80 | Poisson |

TABLE 2: Best filtered out noise at each noise level (%) using Decision Based Coupled Window Median Filter

2. <u>Switching-Based Mean Median Filter</u>

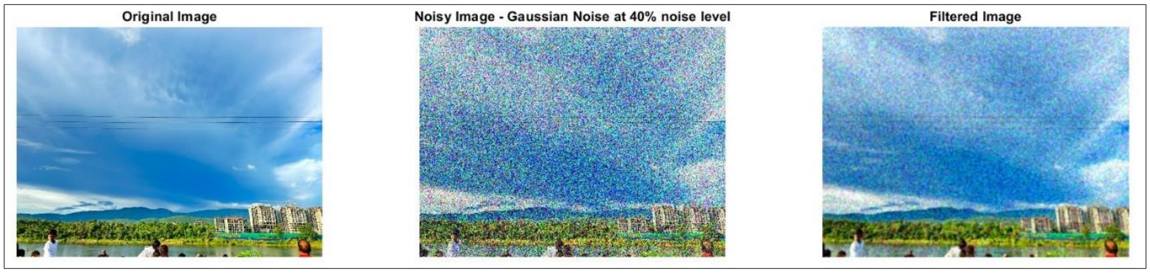


Fig. 8: Denoising scenery.jpg at 40% noise level (Gaussian Noise) using Switching-Based Mean Median Filter

| | MSE | RMSE | PSNR | SSIM | IEF |
|---|---|---|---|---|---|
| Fig. 8 | 563.331004 | 23.734595 | 20.623167 | 0.846717 | 10.829444 |

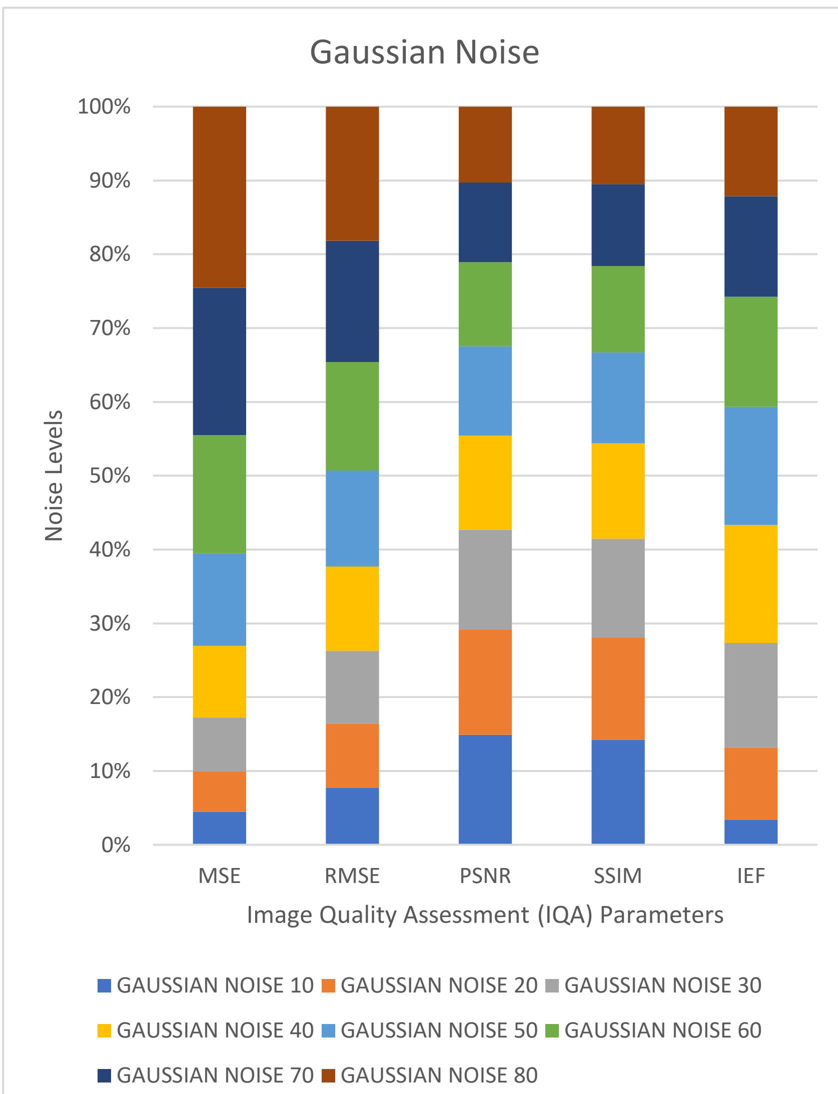


Fig. 9 - Image Quality Assessment (IQA) Parameters of Gaussian Noise for scenery.jpg using Switching-Based Mean Median Filter

The results are depicted by performing the filtering technique on scenery.jpg, which is already denoised using various noises.

TABLE 3 describes IQA parameters for all images when the Noise Level is set to 40% and the Noise Type being Gaussian Noise. Of all the images at this configuration, it can be easily seen that Switching-Based Median Filter works best on satellite2.jpg for sure. Reason being lowest MSE i.e., 391.862056 and RMSE i.e., 19.795506 compared to satellite1.jpg which has the highest MSE and RMSE value of 1579.103717 and 39.737938 respectively. The PSNR value of satellite2.jpg is highest of all i.e., 22.199471.

Another conclusion can be made out of TABLE 4, where different noises are added into an image with different Noise Levels. scenery.jpg was added with various noises at various noise levels. Keeping the base at Noise Levels it can be drawn into conclusion that at specific Noise Level percentages there are specific noises which are best filtered out by the performing filter. Here, for example it is seen that at 10% Noise impunity, Periodic Noise is the best filtered out of all other noises that were added into the image. Similarly, for 20% it is Periodic and so on. This gives a clear idea about the denoising capability of the filter over various noise levels.

| | MSE | RMSE | PSNR | SSIM | IEF |
|---|---|---|---|---|---|
| train.jpg | 744.096781 | 27.278137 | 19.414509 | 0.507334 | 7.999096 |
| coins.jpg | 911.989674 | 30.199167 | 18.530904 | 0.459156 | 8.567217 |
| leaves.jpg | 1115.751546 | 33.402867 | 17.655129 | 0.415443 | 8.454861 |
| scenery.jpg | 1333.646223 | 36.519121 | 16.880397 | 0.376546 | 8.078963 |
| satellite1.jpg | 1579.103717 | 39.737938 | 16.146697 | 0.344125 | 7.523795 |
| satellite2.jpg | 391.862056 | 19.795506 | 22.199471 | 0.795269 | 5.601788 |

TABLE 3: IQA Parameters for all images at 40% noise level (Gaussian Noise) using Switching-Based Median Filter

This Switching-Based Median filter has been tested over various images. For reference scenery.jpg has been selected. The following noises have been specifically inserted into the images: Gaussian Noise, Impulse Noise, Periodic Noise, Poisson Noise, Speckle Noise, Gamma Noise, Rayleigh Noise, Quantization Noise, Brownian Noise and Rician Noise. Also, noise levels are also taken into account starting from 10% to 80% with 7 equal intervals starting from 10% for all mentioned noise types and the conclusion can be drawn out into the TABLE 4.

| Noise Level (%) | Noise Type |
|---|---|
| 10 | Periodic |
| 20 | Periodic |
| 30 | Impulse |
| 40 | Impulse |
| 50 | Periodic |
| 60 | Periodic |
| 70 | Periodic |
| 80 | Periodic |

TABLE 4: Best filtered out noise at each noise level (%) using Switching Based Median Filter

3. <u>Noise Density Range Sensitive Mean Median Filter</u>

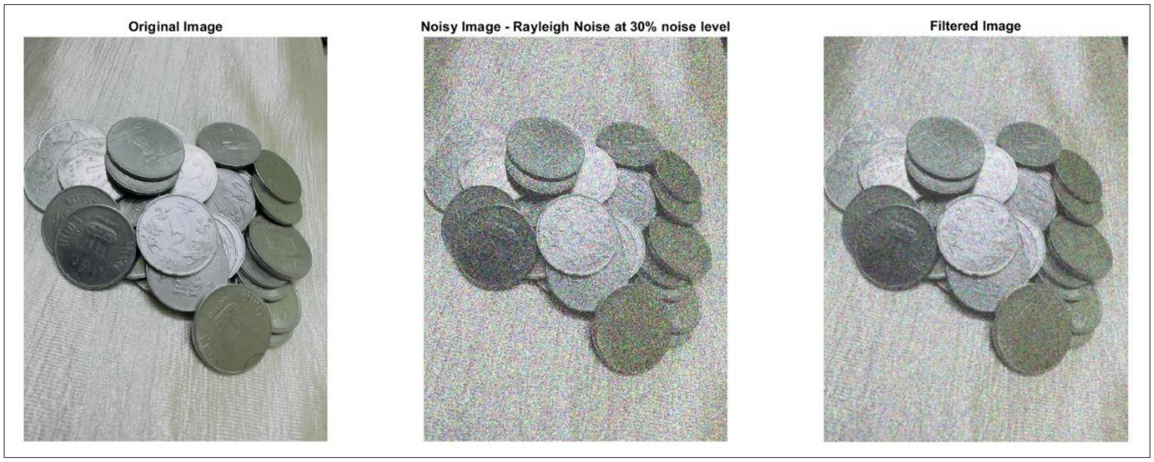


Fig. 10: Denoising coins.jpg at 30% noise level (Rayleigh Noise) using Noise Density Range Sensitive Mean Median Filter

| | MSE | RMSE | PSNR | SSIM | IEF |
|---|---|---|---|---|---|
| Fig. 10 | 687.548623 | 26.221148 | 19.757769 | 0.294949 | 3.634300 |

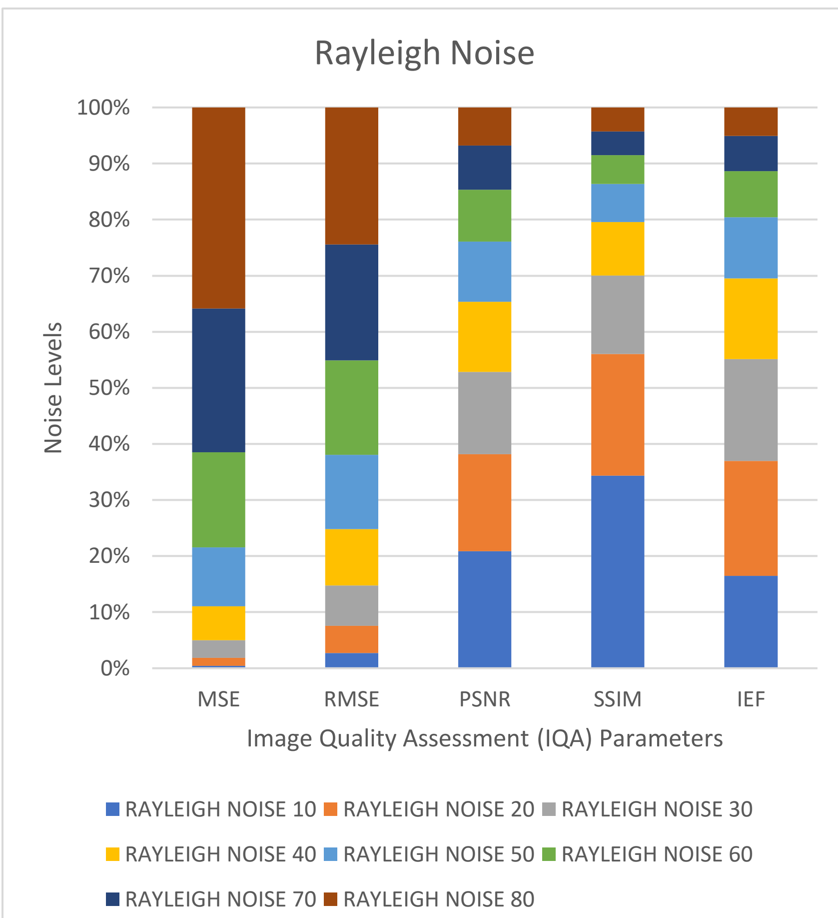


Fig. 11 - Image Quality Assessment (IQA) Parameters of Rayleigh Noise for coins.jpg using Noise Density Range Sensitive Mean Median Filter

The results are depicted by performing the filtering technique on coins.jpg, which is already denoised using various noises.

TABLE 5 describes IQA parameters for all images when the Noise Level is set to 30% and the Noise Type being Rayleigh Noise. Of all the images at this configuration, it can be easily seen that Noise Density Range Sensitive Mean Median filter works best on train.jpg for sure. Reason being lowest MSE i.e., 1003.121814 and RMSE i.e., 31.672098 compared to satellite2.jpg which has the highest MSE and RMSE value of 13927.37419 and 118.014297 respectively. The PSNR value of train.jpg is highest of all i.e., 18.117267.

Another conclusion can be made out of TABLE 6, where different noises are added into an image with different Noise Levels. coins.jpg was added with various noises at various noise levels. Keeping the base at Noise Levels it can be drawn into conclusion that at specific Noise Level percentages there are specific noises which are best filtered out by the performing filter. Here, for example it is seen that at 10% Noise impunity, Impulse Noise is the best filtered out of all other noises that were added into the image. Similarly, for 20% it is Periodic and so on. This gives a clear idea about the denoising capability of the filter over various noise levels.

| | MSE | RMSE | PSNR | SSIM | IEF |
|---|---|---|---|---|---|
| train.jpg | 1003.121814 | 31.672098 | 18.117267 | 0.505075 | 2.521603 |
| coins.jpg | 1962.822436 | 44.303752 | 15.201993 | 0.394271 | 2.22482 |
| leaves.jpg | 3541.298613 | 59.508811 | 12.639178 | 0.300399 | 1.855321 |
| scenery.jpg | 6001.759919 | 77.471026 | 10.348017 | 0.227241 | 1.489046 |
| satellite1.jpg | 9549.102997 | 97.719512 | 8.331178 | 0.16776 | 1.179102 |
| satellite2.jpg | 13927.37419 | 118.014297 | 6.692111 | 0.128666 | 0.968694 |

TABLE 5: IQA Parameters for all images at 30% noise level (Rayleigh Noise) using Noise Density Range Sensitive Mean Median Filter

This Noise Density Range Sensitive Mean Median filter has been tested over various images. For reference coins.jpg has been selected. The following noises have been specifically inserted into the images: Gaussian Noise, Impulse Noise, Periodic Noise, Poisson Noise, Speckle Noise, Gamma Noise, Rayleigh Noise, Quantization Noise, Brownian Noise and Rician Noise. Also, noise levels are also taken into account starting from 10% to 80% with 7 equal intervals starting from 10% for all mentioned noise types and the conclusion can be drawn out into the TABLE 6.

| Noise Level (%) | Noise Type |
|---|---|
| 10 | Impulse |
| 20 | Periodic |
| 30 | Periodic |
| 40 | Periodic |
| 50 | Periodic |
| 60 | Periodic |
| 70 | Periodic |
| 80 | Periodic |

TABLE 6: Best filtered out noise at each noise level (%) using Noise Density Range Sensitive Mean Median Filter

4. Adaptive Median Filter

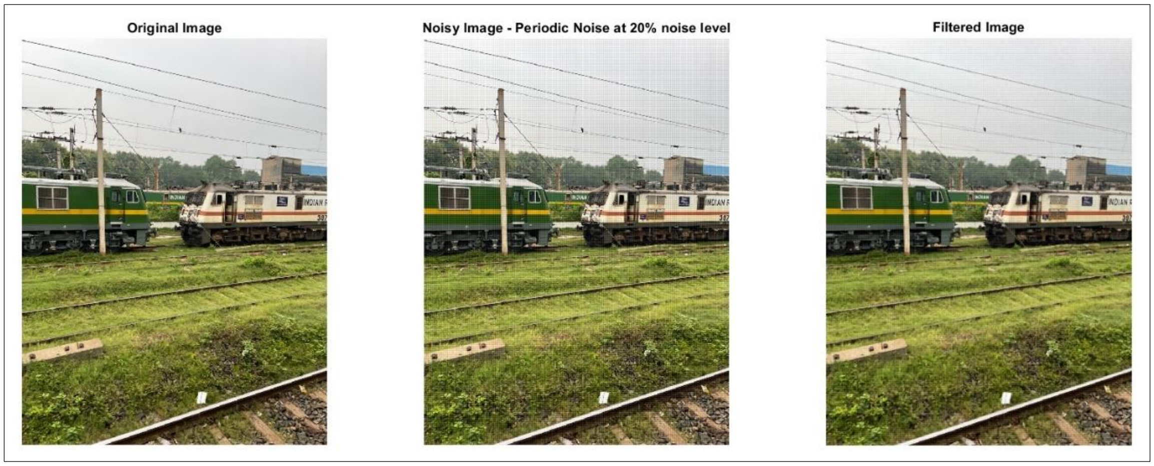


Fig. 12: Denoising train.jpg at 20% noise level (Periodic Noise) using Adaptive Median Filter

| | MSE | RMSE | PSNR | SSIM | IEF |
|---|---|---|---|---|---|
| Fig. 12 | 173.284350 | 13.163751 | 25.743210 | 0.806189 | 1.241941 |

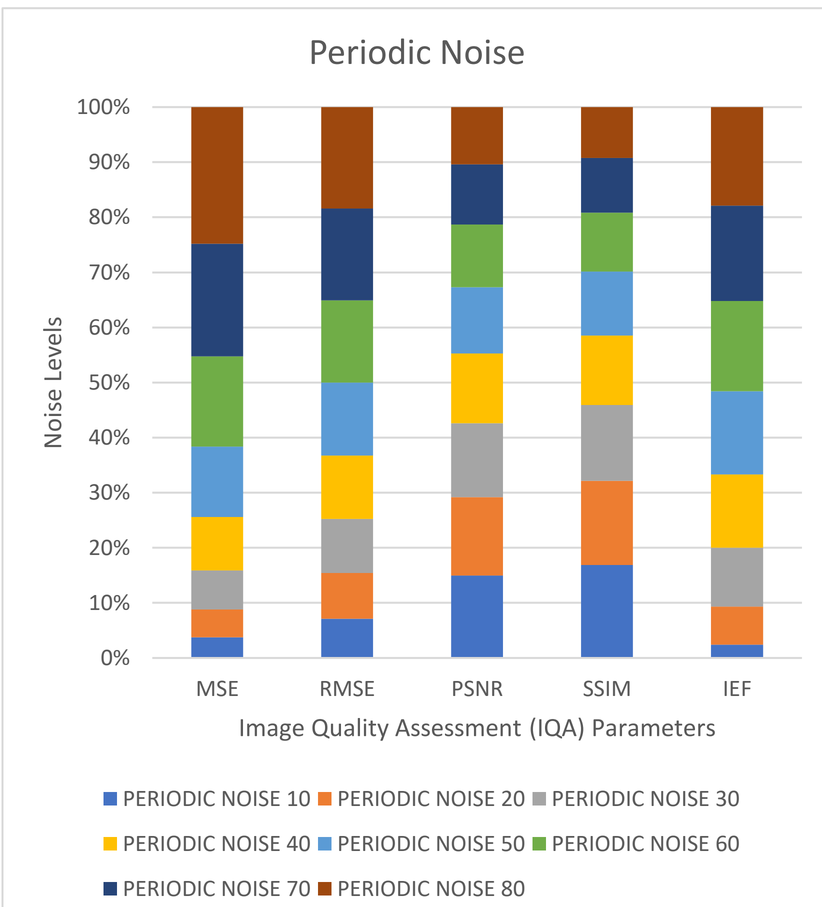


Fig. 13 - Image Quality Assessment (IQA) Parameters of Periodic Noise for train.jpg using Adaptive Median Filter

The results are depicted by performing the filtering technique on train.jpg, which is already denoised using various noises.

TABLE 7 describes IQA parameters for all images when the Noise Level is set to 20% and the Noise Type being Periodic Noise. Of all the images at this configuration, it can be easily seen that Adaptive Median filter works best on train.jpg for sure. Reason being lowest MSE i.e., 236.594991 and RMSE i.e., 15.381645 compared to satellite2.jpg which has the highest MSE and RMSE value of 7762.805714 and 88.106786 respectively. The PSNR value of coins.jpg is highest of all i.e., 24.390748.

Another conclusion can be made out of TABLE 8, where different noises are added into an image with different Noise Levels. train.jpg was added with various noises at various noise levels. Keeping the base at Noise Levels it can be drawn into conclusion that at specific Noise Level percentages there are specific noises which are best filtered out by the performing filter. Here, for example it is seen that at 10% Noise impunity, Impulse Noise is the best filtered out of all other noises that were added into the image. Similarly, for 20% it is Periodic and so on. This gives a clear idea about the denoising capability of the filter over various noise levels.

| | MSE | RMSE | PSNR | SSIM | IEF |
|---|---|---|---|---|---|
| train.jpg | 236.594991 | 15.381645 | 24.390748 | 0.868619 | 18.504179 |
| coins.jpg | 506.414772 | 22.503661 | 21.08574 | 0.735505 | 12.973664 |
| leaves.jpg | 1172.186569 | 34.23721 | 17.440836 | 0.537372 | 7.486834 |
| scenery.jpg | 2458.884761 | 49.587143 | 14.223422 | 0.360544 | 4.446669 |
| satellite1.jpg | 4644.081785 | 68.1475 | 11.461805 | 0.227841 | 2.823732 |
| satellite2.jpg | 7762.805714 | 88.106786 | 9.230616 | 0.134967 | 1.969759 |

TABLE 7: IQA Parameters for all images at 20% noise level (Periodic Noise) using Adaptive Median Filter

This Adaptive Median filter has been tested over various images. For reference train.jpg has been selected. The following noises have been specifically inserted into the images: Gaussian Noise, Impulse Noise, Periodic Noise, Poisson Noise, Speckle Noise, Gamma Noise, Rayleigh Noise, Quantization Noise, Brownian Noise and Rician Noise. Also, noise levels are also taken into account starting from 10% to 80% with 7 equal intervals starting from 10% for all mentioned noise types and the conclusion can be drawn out into the TABLE 8.

| Noise Level (%) | Noise Type |
|---|---|
| 10 | Impulse |
| 20 | Periodic |
| 30 | Periodic |
| 40 | Periodic |
| 50 | Periodic |
| 60 | Periodic |
| 70 | Periodic |
| 80 | Periodic |

TABLE 8: Best filtered out noise at each noise level (%) using Adaptive Median Filter

5. Dual Domain Image Denoising Filter

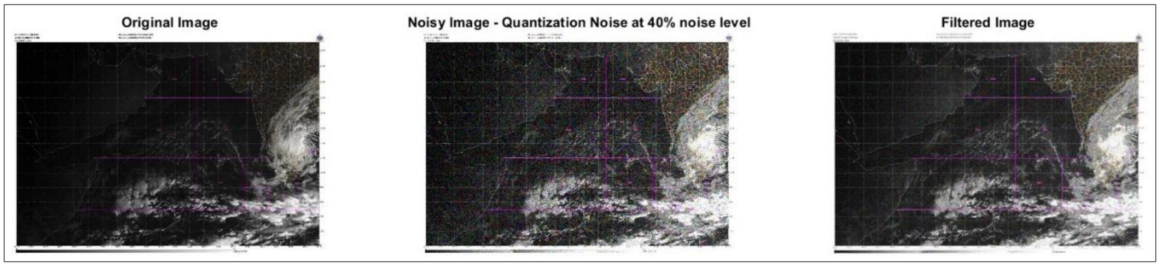


Fig. 14: Denoising satellite1.jpg at 40% noise level (Quantization Noise) using Dual Domain Image Denoising Filter

| | MSE | RMSE | PSNR | SSIM | IEF |
|---|---|---|---|---|---|
| Fig 14 | 1171.197900 | 34.222769 | 17.444501 | 0.474123 | 1.531228 |

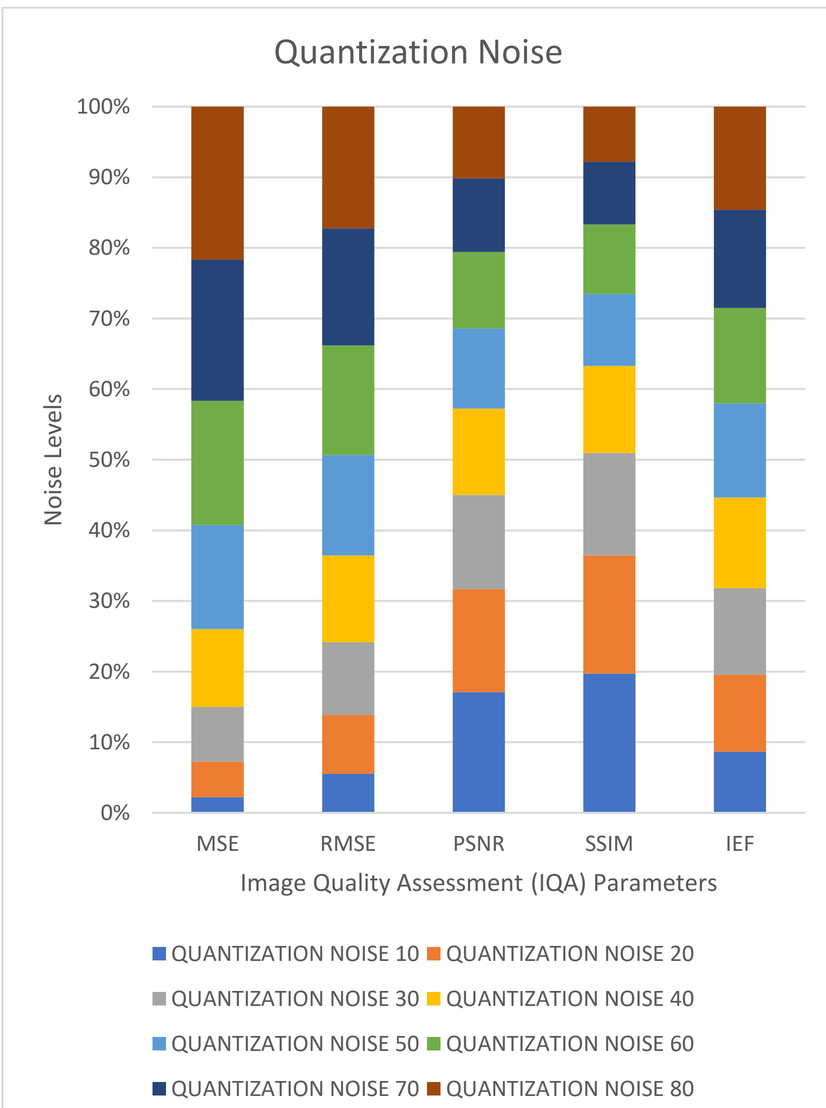


Fig. 15 - Image Quality Assessment (IQA) Parameters of Quantization Noise for satellite1.jpg using Dual Domain Image Denoising Filter

The results are depicted by performing the filtering technique on satellite1.jpg, which is already denoised using various noises.

TABLE 9 describes IQA parameters for all images when the Noise Level is set to 40% and the Noise Type being Quantization Noise. Of all the images at this configuration, it can be easily seen that Dual Domain Image Denoising filter works best on satellite1.jpg for sure. Reason being lowest MSE i.e., 1172.749294 and RMSE i.e., 34.245427 compared to coins.jpg which has the highest MSE and RMSE value of 3099.90708 and 55.676809 respectively. The PSNR value of satellite1.jpg is highest of all i.e., 17.438752.

Another conclusion can be made out of TABLE 10, where different noises are added into an image with different Noise Levels. satellite1.jpg was added with various noises at various noise levels. Keeping the base at Noise Levels it can be drawn into conclusion that at specific Noise Level percentages there are specific noises which are best filtered out by the performing filter. Here, for example it is seen that at 10% Noise impunity, Poisson Noise is the best filtered out of all other noises that were added into the image. Similarly, for 20% it is Periodic and so on. This gives a clear idea about the denoising capability of the filter over various noise levels.

| | MSE | RMSE | PSNR | SSIM | IEF |
|---|---|---|---|---|---|
| train.jpg | 2138.960771 | 46.2489 | 14.828775 | 0.64129 | 1.343039 |
| coins.jpg | 3099.90708 | 55.676809 | 13.217317 | 0.475678 | 1.178674 |
| leaves.jpg | 2119.264495 | 46.03547 | 14.868952 | 0.726648 | 1.298706 |
| scenery.jpg | 2163.018074 | 46.508258 | 14.780202 | 0.737012 | 1.25068 |
| satellite1.jpg | 1172.749294 | 34.245427 | 17.438752 | 0.474157 | 1.530675 |
| satellite2.jpg | 2189.687319 | 46.794095 | 14.726983 | 0.495684 | 1.328665 |

TABLE 9: IQA Parameters for all images at 40% noise level (Quantization Noise) using Dual Domain Image Denoising Filter

This Dual Domain Image Denoising filter has been tested over various images. For reference satellite1.jpg has been selected. The following noises have been specifically inserted into the images: Gaussian Noise, Impulse Noise, Periodic Noise, Poisson Noise, Speckle Noise, Gamma Noise, Rayleigh Noise, Quantization Noise, Brownian Noise and Rician Noise. Also, noise levels are also taken into account starting from 10% to 80% with 7 equal intervals starting from 10% for all mentioned noise types and the conclusion can be drawn out into the TABLE 10.

| Noise Level (%) | Noise Type |
|---|---|
| 10 | Poisson |
| 20 | Periodic |
| 30 | Speckle |
| 40 | Speckle |
| 50 | Periodic |
| 60 | Poisson |
| 70 | Periodic |
| 80 | Periodic |

TABLE 10: Best filtered out noise at each noise level (%) using Dual Domain Image Denoising Filter

# SUPPLEMENT

Various noises have been added in the images and filtered out through the respective filters. The noises were namely Gaussian Noise, Impulse Noise, Periodic Noise, Poisson Noise, Speckle Noise, Gamma Noise, Rayleigh Noise, Quantization Noise, Brownian Noise and Rician Noise. This section will analyze each image where Noise Level will be set to 10% with all noises present in an image at the same time. This can be analyzed for each filter as follows:

1. Decision-Based Coupled Window Median Filter

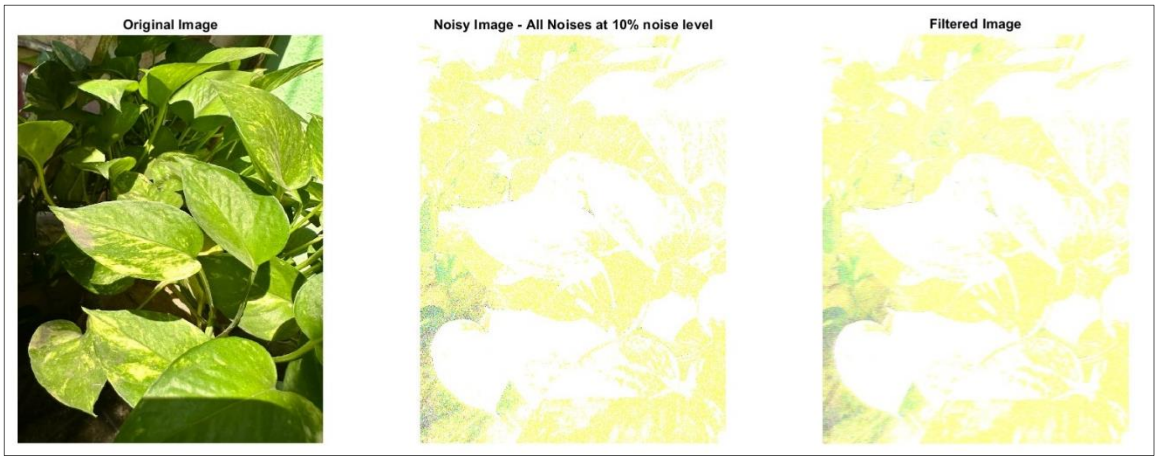


Fig. 15: Denoising leaves.jpg at 10% noise level including each noise using Decision-Based Coupled Window Median Filter

| | MSE | RMSE | PSNR | SSIM | IEF |
|---|---|---|---|---|---|
| coins.jpg | 11286.68436 | 106.238808 | 7.60514 | 0.333438 | 1.000299 |
| leaves.jpg | 26701.11439 | 163.404756 | 3.86551 | 0.140444 | 1.013262 |
| satellite1.jpg | 38587.96764 | 196.438203 | 2.266285 | 0.114051 | 1.009836 |
| satellite2.jpg | 26235.57862 | 161.974006 | 3.941897 | 0.255062 | 1.001661 |
| scenery.jpg | 13872.36872 | 117.78102 | 6.709297 | 0.096361 | 1.002137 |
| train.jpg | 19722.84115 | 140.438033 | 5.181109 | 0.272871 | 0.999958 |

TABLE 11: IQA Parameters for all images at 10% noise level including each noise using Dual Domain Image Denoising Filter

Analysis of TABLE 11 tells that for all the images when each noise at 10% noise level is added, coins.jpg is best one to be filtered out using Decision-Based Coupled Window Median Filter. The reason is lowest MSE and RMSE which is 11286.68436 and 106.238808. In comparison to this satellite1.jpg is weakly filtered due to its highest MSE and RMSE i.e., 38587.96764 and 196.438203 respectively. The PSNR value of coins.jpg is highest (7.60514) among all the six images.

2. Switching-Based Median Filter

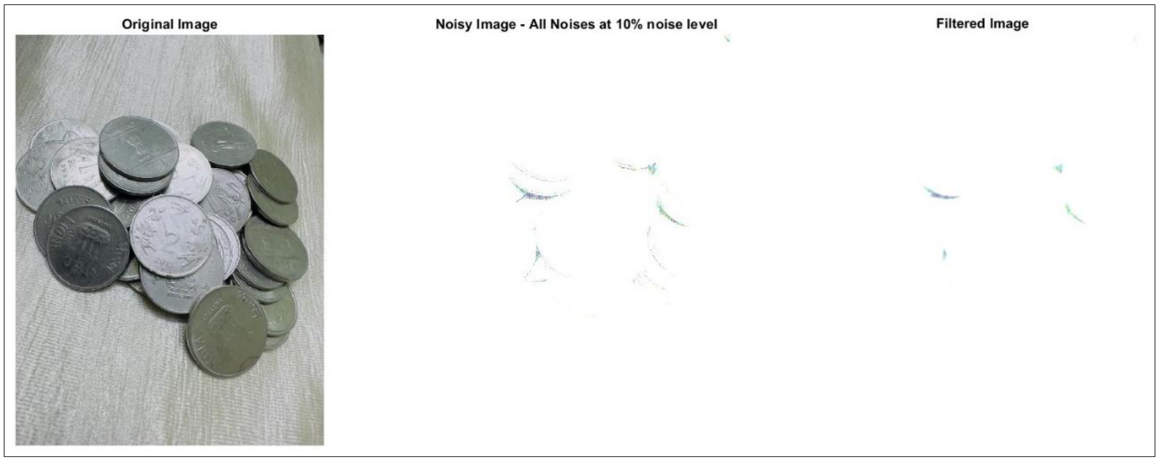


Fig. 16: Denoising coins.jpg at 10% noise level including each noise using Switching-Based Median Filter

| | MSE | RMSE | PSNR | SSIM | IEF |
|---|---|---|---|---|---|
| coins.jpg | 11362.48945 | 106.594979 | 7.576069 | 0.332556 | 0.993072 |
| leaves.jpg | 26157.23393 | 161.731982 | 3.954885 | 0.134105 | 0.992106 |
| satellite1.jpg | 40689.37611 | 201.716078 | 2.035993 | 0.125153 | 0.961956 |
| satellite2.jpg | 26777.76731 | 163.639137 | 3.85306 | 0.255984 | 0.981062 |
| scenery.jpg | 14344.33781 | 119.76785 | 6.563999 | 0.081667 | 0.967864 |
| train.jpg | 20378.95493 | 142.754877 | 5.038985 | 0.256044 | 0.970797 |

TABLE 12: IQA Parameters for all images at 10% noise level including each noise using Switching-Based Median Filter

Analysis of TABLE 12 tells that for all the images when each noise at 10% noise level is added, coins.jpg is best one to be filtered out using Switching-Based Median Filter. The reason is lowest MSE and RMSE which is 11362.48945 and 106.594979. In comparison to this satellite1.jpg is weakly filtered due to its highest MSE and RMSE i.e., 40689.37611 and 201.716078 respectively. The PSNR value of coins.jpg is highest (7.576069) among all the six images.

3. Noise Density Range Sensitive Mean Median Filter

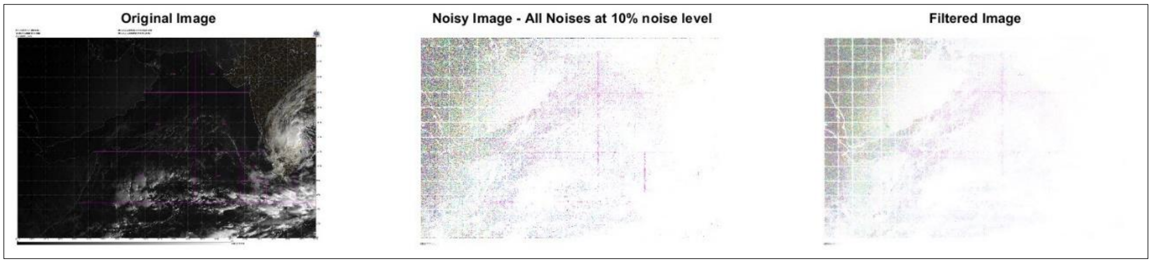


Fig. 17: Denoising satellite1.jpg at 10% noise level including each noise using Noise Density Range Sensitive Mean Median Filter

| | MSE | RMSE | PSNR | SSIM | IEF |
|---|---|---|---|---|---|
| coins.jpg | 11343.46575 | 106.505708 | 7.583346 | 0.332981 | 0.997204 |
| leaves.jpg | 25901.99965 | 160.940982 | 3.997471 | 0.138156 | 1.003528 |
| satellite1.jpg | 39651.46494 | 199.126756 | 2.148211 | 0.123562 | 0.977439 |
| satellite2.jpg | 26390.99421 | 162.453052 | 3.916246 | 0.257919 | 0.989502 |
| scenery.jpg | 14253.9496 | 119.389906 | 6.591451 | 0.085083 | 0.981976 |
| train.jpg | 19882.02145 | 141.003622 | 5.146198 | 0.272961 | 0.978926 |

TABLE 13: IQA Parameters for all images at 10% noise level including each noise using Noise Density Range Sensitive Mean Median Filter

Analysis of TABLE 13 tells that for all the images when each noise at 10% noise level is added, coins.jpg is best one to be filtered out using Noise Density Range Sensitive Mean Median Filter. The reason is lowest MSE and RMSE which is 11343.46575 and 106.505708. In comparison to this satellite1.jpg is weakly filtered due to its highest MSE and RMSE i.e., 39651.46494 and 199.126756 respectively. The PSNR value of coins.jpg is highest (7.583346) among all the six images.

4. Adaptive Median Filter

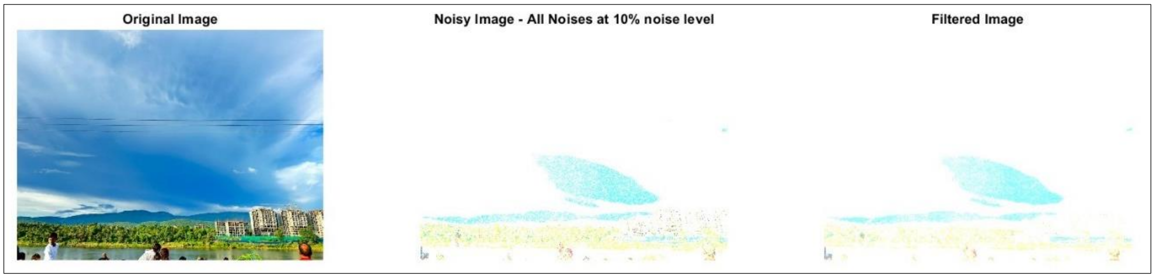


Fig. 18: Denoising scenery.jpg at 10% noise level including each noise using Adaptive Median Filter

| | MSE | RMSE | PSNR | SSIM | IEF |
|---|---|---|---|---|---|
| coins.jpg | 11355.18019 | 106.560688 | 7.578863 | 0.332775 | 0.996069 |
| leaves.jpg | 27540.72557 | 165.953986 | 3.73105 | 0.104545 | 0.989489 |
| satellite1.jpg | 40085.16663 | 200.212803 | 2.100967 | 0.119544 | 0.958888 |
| satellite2.jpg | 26545.46209 | 162.927782 | 3.890901 | 0.256153 | 0.985891 |
| scenery.jpg | 14389.72159 | 119.957166 | 6.55028 | 0.079701 | 0.978808 |
| train.jpg | 20072.02544 | 141.675776 | 5.104892 | 0.264549 | 0.977143 |

TABLE 14: IQA Parameters for all images at 10% noise level including each noise using Adaptive Median Filter

Analysis of TABLE 14 tells that for all the images when each noise at 10% noise level is added, coins.jpg is best one to be filtered out using Adaptive Median Filter. The reason is lowest MSE and RMSE which is 11355.18019 and 106.560688. In comparison to this satellite1.jpg is weakly filtered due to its highest MSE and RMSE i.e., 40085.16663 and 200.212803 respectively. The PSNR value of coins.jpg is highest (7.578863) among all the six images.

5. Dual Domain Image Denoising Filter

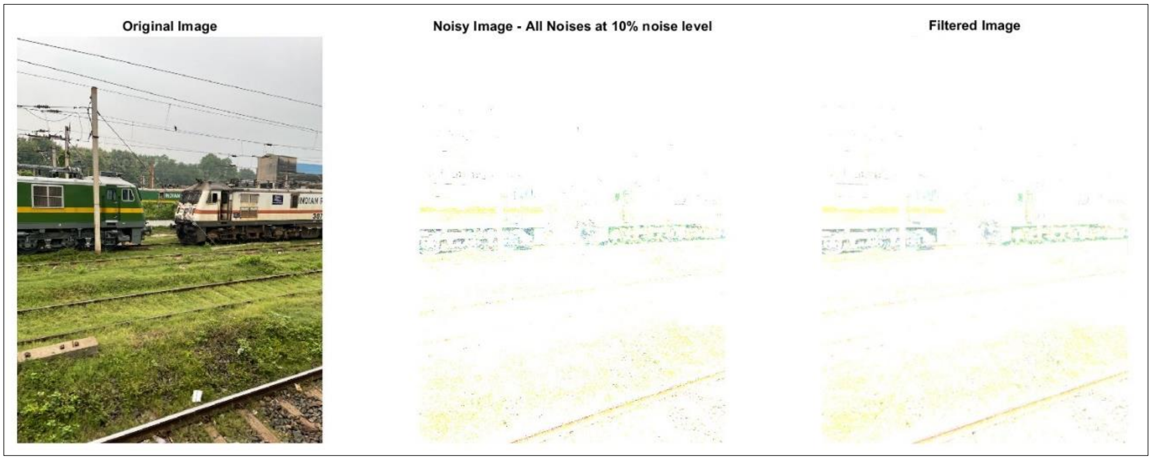


Fig. 19: Denoising train.jpg at 10% noise level including each noise using Dual Domain Image Denoising Filter

| | MSE | RMSE | PSNR | SSIM | IEF |
|---|---|---|---|---|---|
| coins.jpg | 11277.65524 | 106.196305 | 7.608615 | 0.334714 | 1.000978 |
| leaves.jpg | 26979.77876 | 164.255224 | 3.82042 | 0.127784 | 1.013767 |
| satellite1.jpg | 38434.10115 | 196.046171 | 2.283636 | 0.129757 | 1.011075 |
| satellite2.jpg | 26157.79079 | 161.733703 | 3.954793 | 0.26109 | 1.002642 |
| scenery.jpg | 13772.79593 | 117.357556 | 6.740582 | 0.105222 | 1.008006 |
| train.jpg | 19553.91675 | 139.83532 | 5.218466 | 0.290791 | 1.004013 |

TABLE 15: IQA Parameters for all images at 10% noise level including each noise using Dual Domain Image Denoising Filter

Analysis of TABLE 15 tells that for all the images when each noise at 10% noise level is added, coins.jpg is best one to be filtered out using Dual Domain Image Denoising Filter. The reason is lowest MSE and RMSE which is 11277.65524 and 106.196305. In comparison to this satellite1.jpg is weakly filtered due to its highest MSE and RMSE i.e., 38434.10115 and 196.046171 respectively. The PSNR value of coins.jpg is highest (7.608615) among all the six images.

# REFERNCES


1. Bhadouria, V. S., Ghoshal, D., & Siddiqi, A. H. (2014). A new approach for high density saturated impulse noise removal using decision-based coupled window median filter. Signal, Image and Video Processing, 8(1), 71–84. https://doi.org/10.1007/s11760-013-0487-5
2. Ko, S., & Lee, Y. (1991). Center weighted median filters and their applications to image enhancement. IEEE Transactions on Circuits and Systems, 38(9), 984–993. https://doi.org/10.1109/31.83870
3. Jayaraj, V., & Ebenezer, D. (2010). A New Switching-Based Median Filtering Scheme and Algorithm for Removal of High-Density Salt and Pepper Noise in Images. EURASIP Journal on Advances in Signal Processing, 2010(1). https://doi.org/10.1155/2010/690218
4. Eng, H., & Ma, K. (2001). Noise adaptive soft-switching median filter. IEEE Transactions on Image Processing, 10(2), 242–251. https://doi.org/10.1109/83.902289
5. Fabijańska, A., & Sankowski, D. (2011). Noise adaptive switching median-based filter for impulse noise removal from extremely corrupted images. Iet Image Processing, 5(5), 472. https://doi.org/10.1049/iet-ipr.2009.0178
6. Sohi, P. J. S., Sharma, N., Garg, B., & Arya, K. V. (2021). Noise Density Range Sensitive Mean-Median Filter for Impulse Noise Removal. In Springer eBooks (pp. 150–162). https://doi.org/10.1007/978-981-15-6067-5_18
7. Hwang, H., & Haddad, R. A. (1995). Adaptive median filters: new algorithms and results. IEEE Transactions on Image Processing, 4(4), 499–502. https://doi.org/10.1109/83.370679
8. Ibrahim, H. F., Kong, N., & Ng, T. F. (2008). Simple adaptive median filter for the removal of impulse noise from highly corrupted images. IEEE Transactions on Consumer Electronics, 54(4), 1920–1927. https://doi.org/10.1109/tce.2008.4711254
9. Chen, T., & Wu, H. (2001). Adaptive impulse detection using center-weighted median filters. IEEE Signal Processing Letters, 8(1), 1–3. https://doi.org/10.1109/97.889633
10. Fitch, J. A., Coyle, E. F., & Gallagher, N. C. (1984). Median filtering by threshold decomposition. IEEE Transactions on Acoustics, Speech, and Signal Processing, 32(6), 1183–1188. https://doi.org/10.1109/tassp.1984.1164468
11. Knaus, C., & Zwicker, M. (2013). Dual-domain image denoising. https://doi.org/10.1109/icip.2013.6738091