

DSP PROJECT

IMAGE DENOISING

GROUP-11

MEMBERS:

K Niveda (S20190020224)

S Anuhya (S20190020251)

R Varshini (S20190020245)

INTRODUCTION: Image Denoising is the task of removing noise from an image so as to restore the true image. With the presence of noise, possible subsequent image processing tasks, such as video processing, image analysis, and tracking, are adversely affected. Therefore, image denoising plays an important role in modern image processing systems.

TYPES OF NOISE:

Here, we will discuss the major four types of noises:

Salt and Pepper Noise: In this type of noise, white and black pixels are sparsely found on the image. Mostly Salt & Pepper noise occurs due to the disturbances happened in the image pixels, malfunctioning of camera's sensor cell, faulty memory space in storage and errors in digitization process.

Gaussian Noise: It is statistical noise having a probability density function (PDF) equal to that of the Normal Distribution. It arises due to poor lighting or high temperature or transmission.

Poisson Noise: It also called Shot Noise or Quantum (Photon) Noise. This type of noise has a probability density function of a Poisson distribution. The statistical nature of electromagnetic waves such as x-rays and gamma rays produce this type of noise.

Speckle Noise: Speckle occurs due to the random fluctuations of signals which results in the increased grey level of the image. It is a multiplicative noise that is found in coherent imaging systems.

TYPES OF FILTERS:

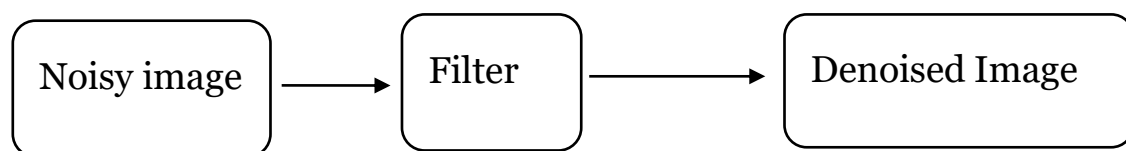
Here, we will discuss the below 7 types of filters:

- Mean Filter
- Median Filter
- Gaussian Filter
- Bilateral Filter

- Wiener Filter
- Adaptive Local Noise Reduction Filter
- Adaptive Median Filter

METHODOLOGY:

- First, we take images with no noise and add different types of noise to it.
- We apply different types of filters to these noisy images to remove noise.
- We compare the signal to noise ratios of all these images to check which filter is more effective to which noise.



IMPLEMENTATION:

- We create a function to add different types of noise to an image.
- We pass the noisy image to the filter function.
- **Mean Filter:** For every pixel in an image, we consider a square window (3x3) surrounding that particular pixel. We replace the centre value of the sliding window with the average of all the pixels in it.
- **Median Filter:** It is similar to the Mean Filter except we replace the centre value of the sliding window with the median of all pixels in the window.
- **Gaussian Filter:** We consider a Gaussian kernel and convolve it with the sliding window for each pixel. We replace the centre value of the sliding window with the result of the convolution. In order not to change the overall brightness of the image, the Gaussian kernel's normalized such that their integrals evaluate to 1.
- **Bilateral Filter:** We consider two gaussian functions in this, one for coordinate space and another for colour space. For coordinate

space, we take a 7*7 kernel and fill it with the distance of centre pixel from each pixel in the kernel. We then apply the gaussian function to this. For colour space, we apply gaussian function to the difference of intensity values from centre pixel to each pixel in the window. We multiply the values of the above two kernels with the intensity values of the sliding window. The elements in the result are added and normalized. This process is repeated for each pixel in the noisy image.

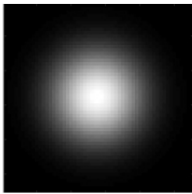
$$BF[I]_p = \frac{1}{W_p} \sum_{q \in S} G_{\sigma_s}(\|p - q\|) G_{\sigma_r}(|I_p - I_q|) I_q$$

↓

Normalization
Factor

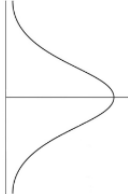
↓

Space Weight



↓

Range Weight



Where W_p is given by

$$W_p = \sum_{q \in S} G_{\sigma_s}(\|p - q\|) G_{\sigma_r}(|I_p - I_q|).$$

- **Wiener Filter:** We consider a 3x3 Gaussian kernel and apply Fourier transform to it. This is $H(u, v)$. The wiener function is calculated from the below formula where $P_n(u, v)/P_s(u, v)$ = noise to signal ratio = nsr . We multiply the wiener function to the Fourier transform of the noisy image and apply inverse Fourier transform to the resultant.

$$\hat{S}(u, v) = G(u, v) X(u, v)$$

$$G(u, v) = \frac{H^*(u, v) P_s(u, v)}{|H(u, v)|^2 P_s(u, v) + P_n(u, v)}$$

Here,

$H(u, v)$: Fourier transform of the point-spread function (PSF)

$P_s(u, v)$: Power spectrum of the signal process, obtained by taking the Fourier transform of the signal autocorrelation.

$P_n(u, v)$: Power spectrum of the noise process, obtained by taking the Fourier transform of the signal autocorrelation.

- **Adaptive Local Noise Reduction Filter:** We consider a 5x5 sliding window. We compute the local variance and local mean of the window. The overall variance of the image is calculated by taking the average of all the local variances. Now, we get the filtered image using the below equation.

$$f(x, y) = g(x, y) - \frac{\sigma_\eta^2}{\sigma_L^2} [g(x, y) - m_L]$$

S_{xy} : local region

$g(x, y)$, the value of the noisy image at (x, y) ;

σ_η^2 , the variance of the noise corrupting $f(x, y)$

to form $g(x, y)$;

m_L , the local mean of the pixels in S_{xy} ;

σ_L^2 , the local variance of the pixels in S_{xy} .

- **Adaptive Median Filter:** The adaptive median filter works in two levels denoted by Level A and Level B as follows:

Stage A:

$$A1 = Z_{med} - Z_{min}; \quad A2 = Z_{med} - Z_{max}$$

If $A1 > 0$ and $A2 < 0$, go to stage B

Else increase the window size

If window size $\leq S_{max}$, repeat stage A; Else output Z_{med}

Stage B:

$$B1 = Z_{xy} - Z_{min}; \quad B2 = Z_{xy} - Z_{max}$$

If $B1 > 0$ and $B2 < 0$, output Z_{xy} ; Else output Z_{med}

Where,

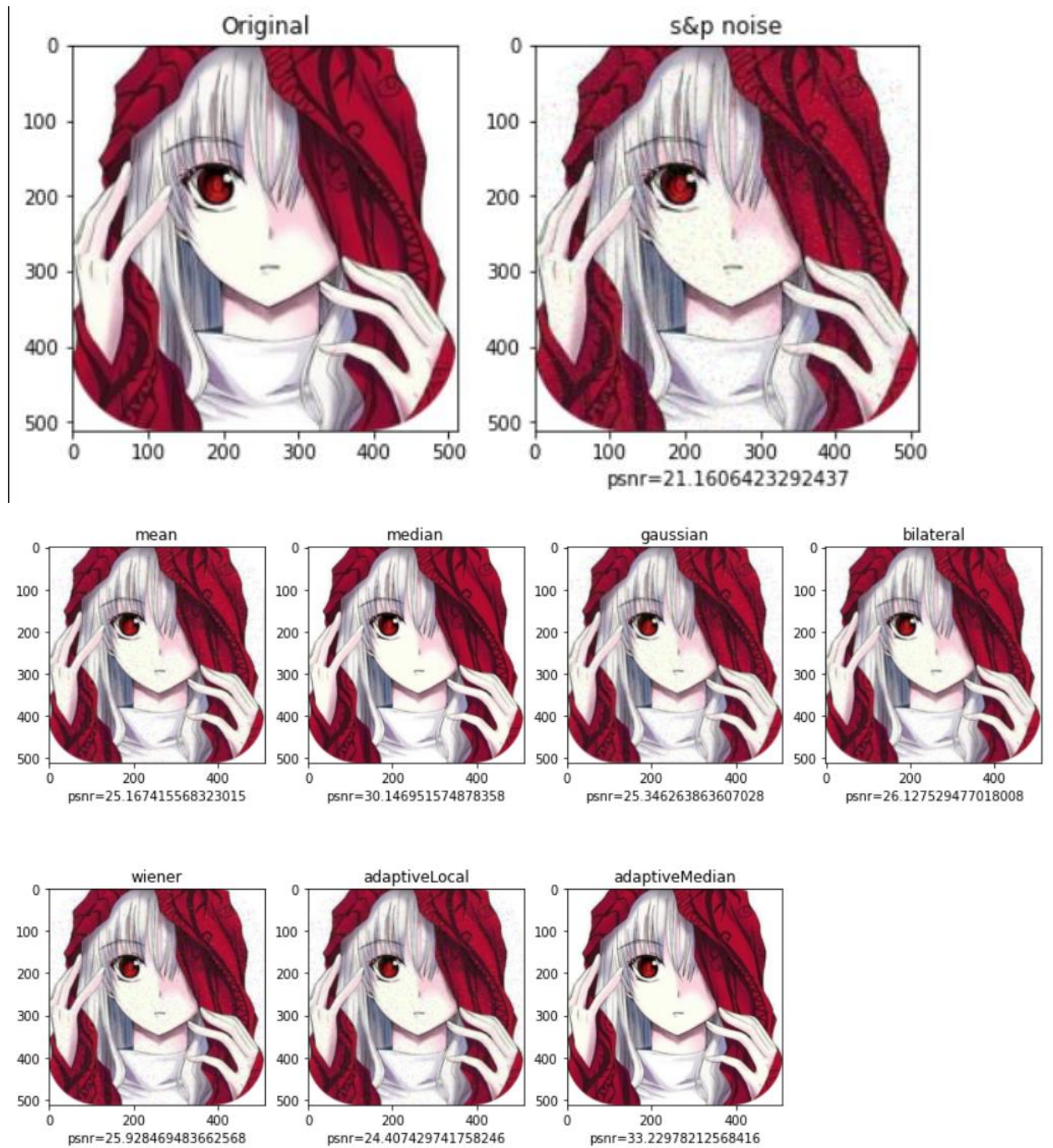
Z_{min} = minimum intensity value in S_{xy}

Z_{max} = maximum intensity value in S_{xy}

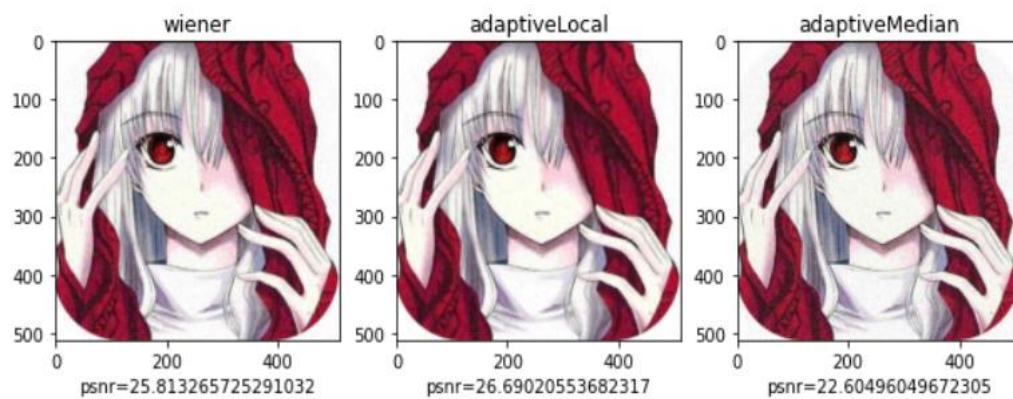
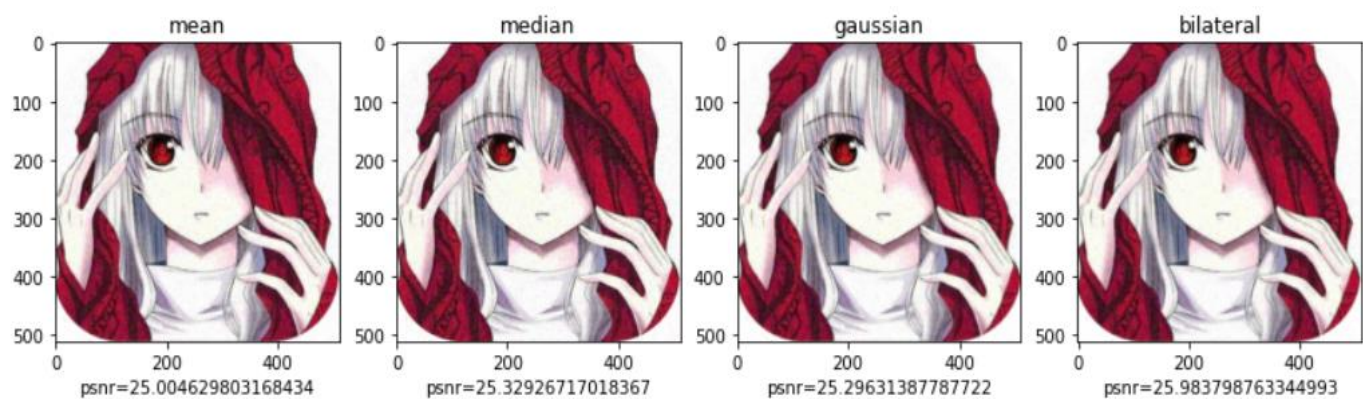
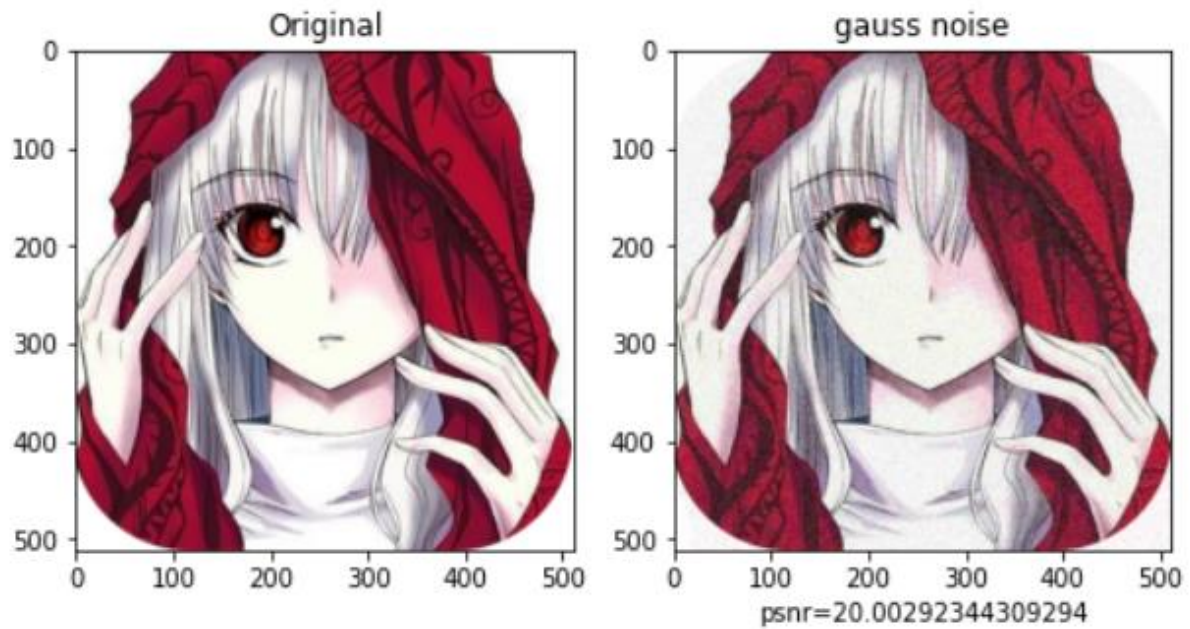
Z_{med} = median intensity value in S_{xy}
 Z_{xy} = intensity value at coordinates (x, y)
 S_{max} = maximum allowed size of S_{xy}

OUTPUT IMAGES:

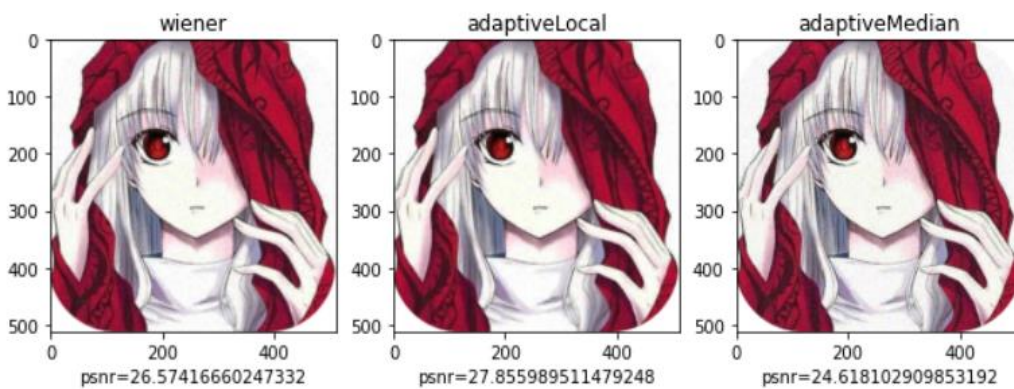
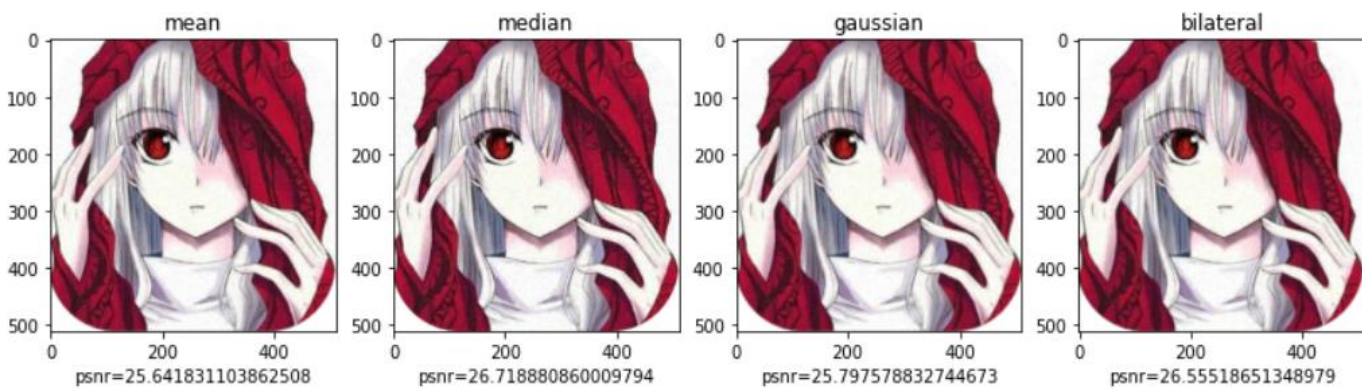
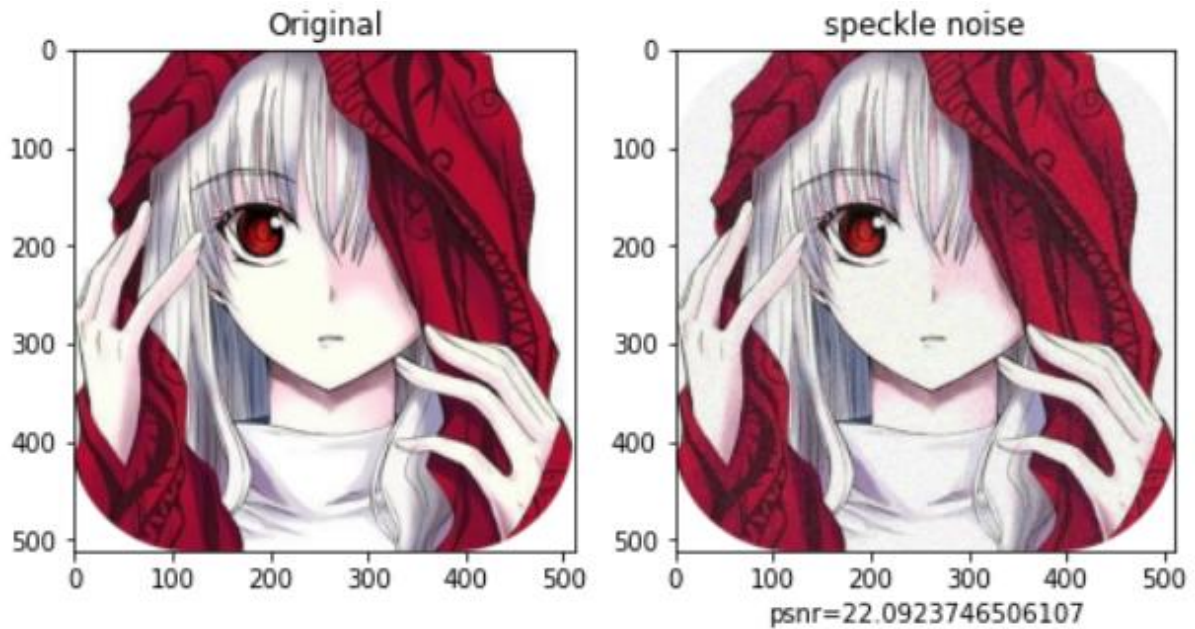
Salt and Pepper Noise Reduction:



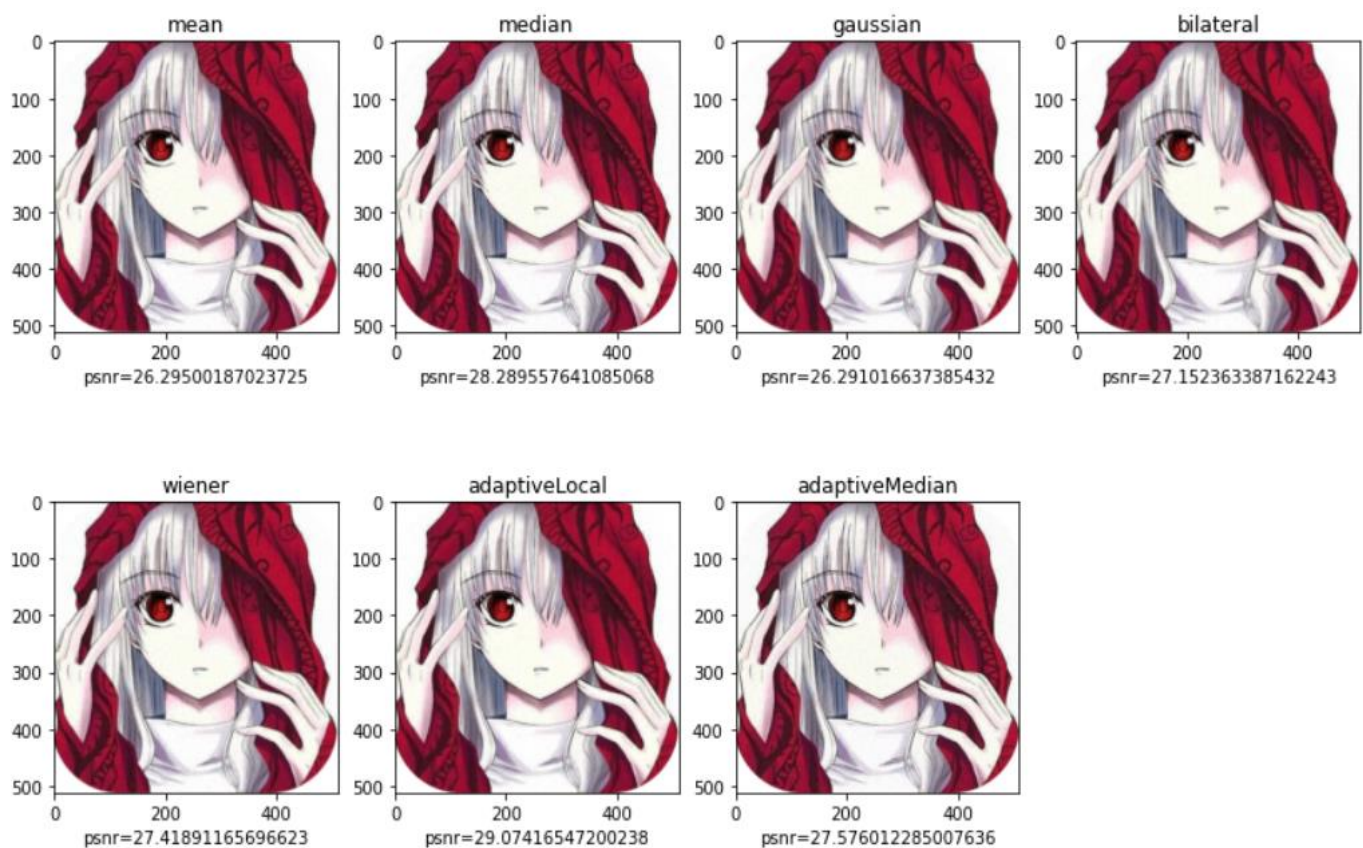
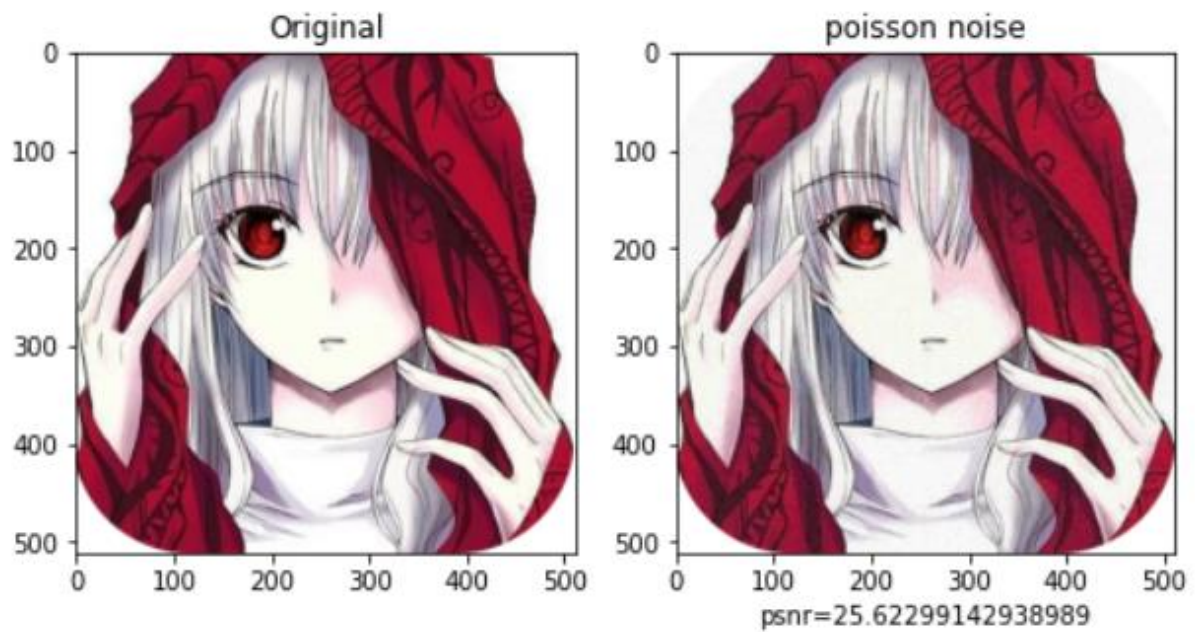
Gaussian Noise Reduction:



Speckle Noise Reduction:



Poisson Noise Reduction:



OBSERVATIONS:

From the denoised images, we observe the following:

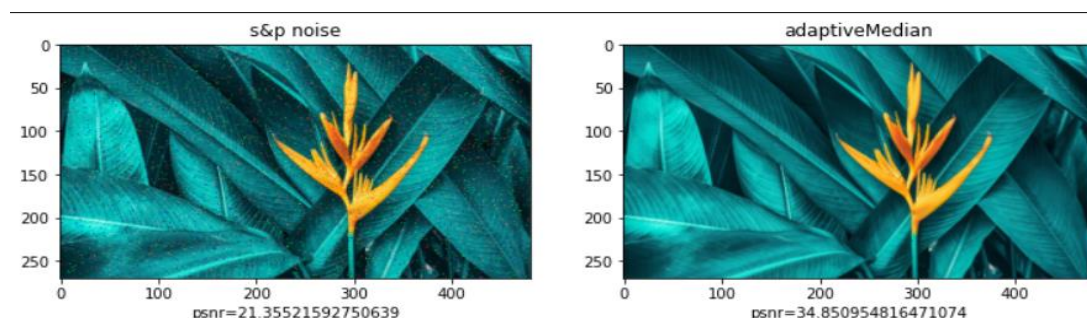
- **Salt and Pepper Noise:** The Median and Adaptive Median filters are best suited for this noise and give high signal to noise ratios. Here, we observe that the Adaptive Median filter not only reduces noise but also preserves edges.
- **Gaussian Noise:** The Gaussian and Bilateral filters are best suited for this type of noise. Here, the wiener filter also denoises especially when there is both additive noise and motion blur.
- **Speckle Noise:** Wiener and gaussian filter and adaptive local gives the best results for this type of noise.
- **Poisson Noise:** The best filter for Poisson noise is Wiener and Adaptive Local Mean filter.

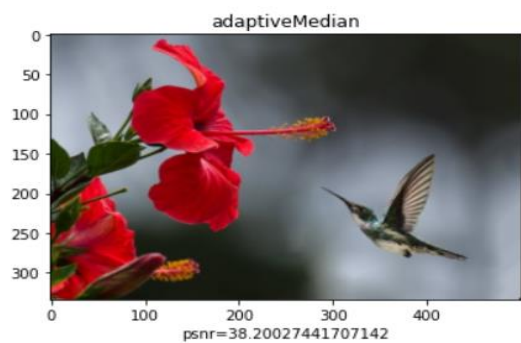
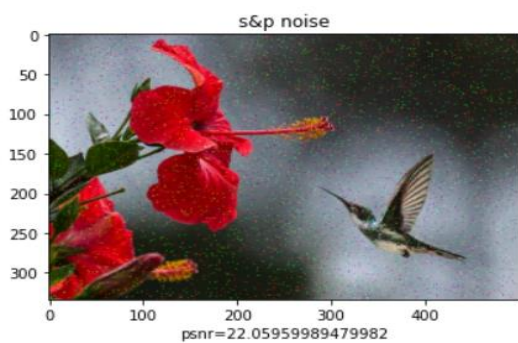
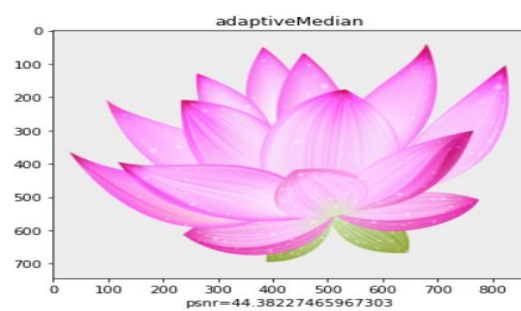
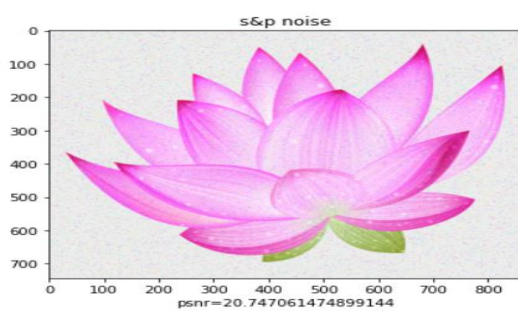
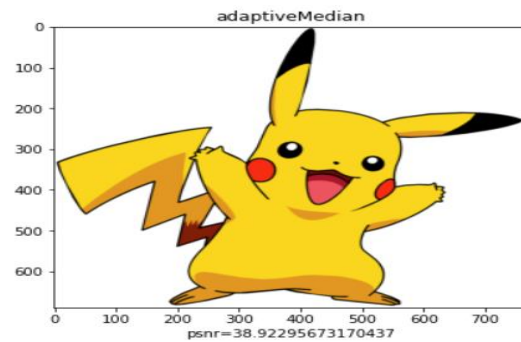
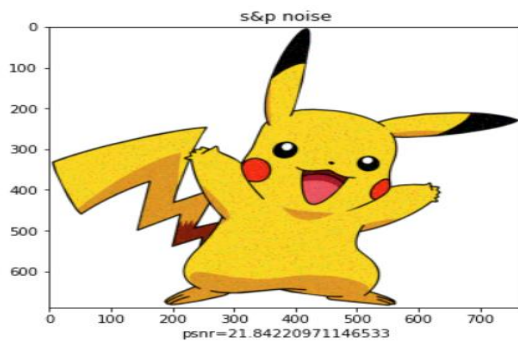
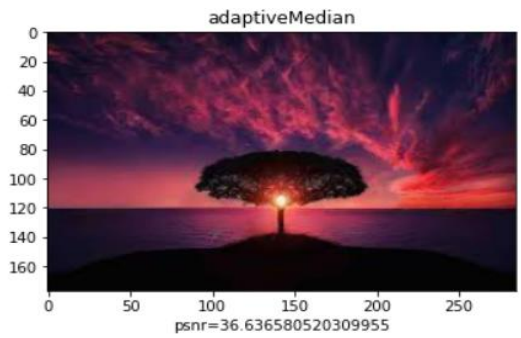
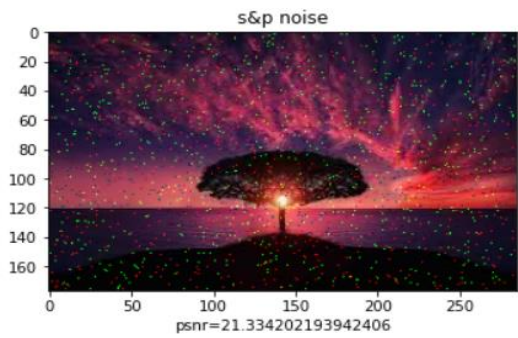
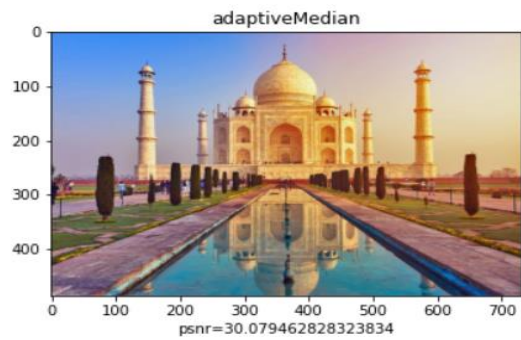
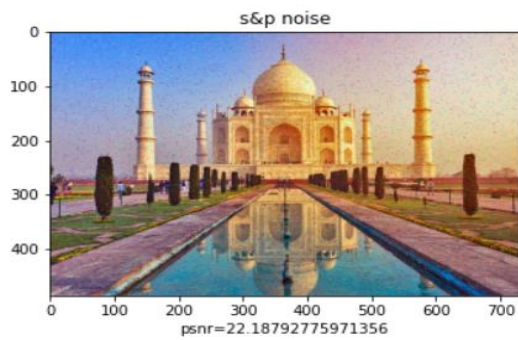
The PSNR values are given in the below table:

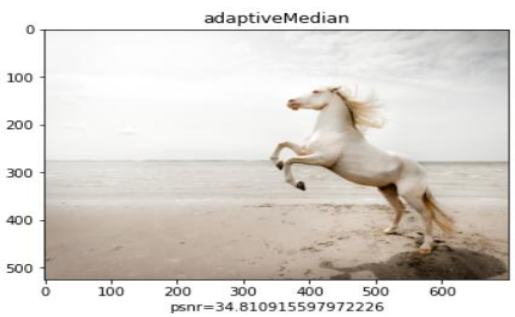
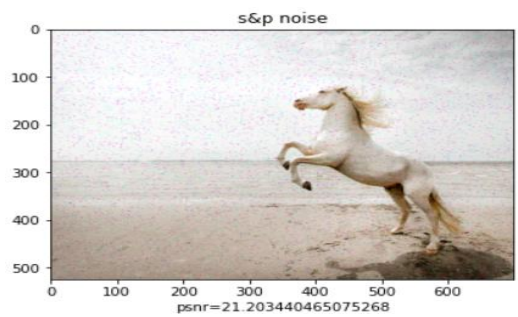
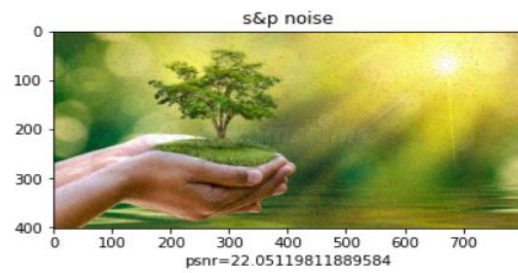
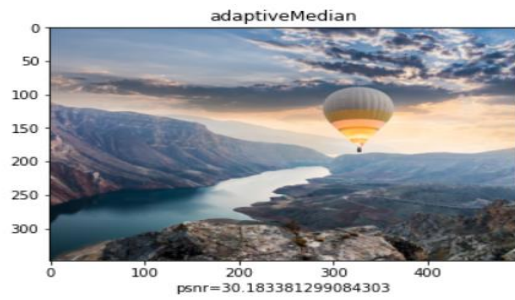
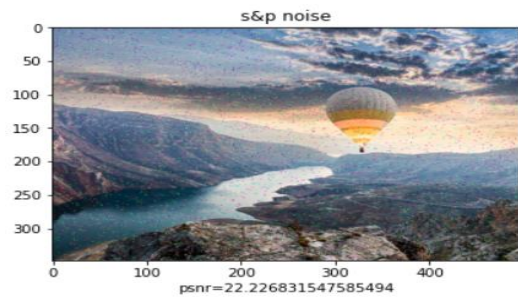
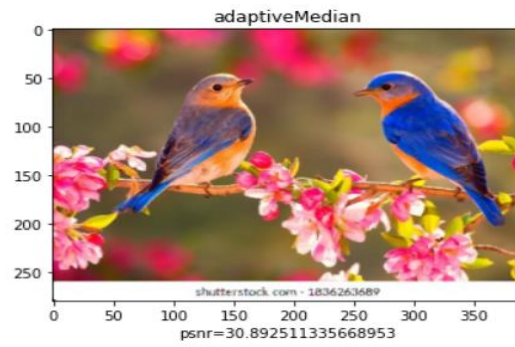
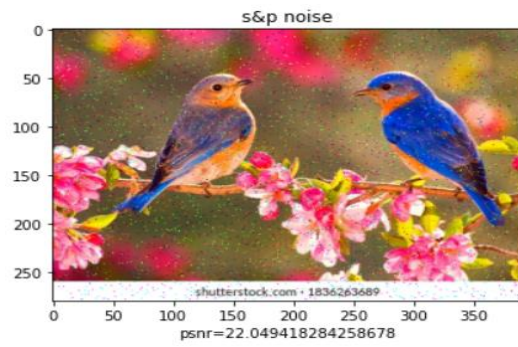
Types of Noise	Noisy	Mean filter	Median filter	Gaussian filter	Bilateral filter	Wiener filter	Adaptive local mean filter	Adaptive median filter
Salt and pepper	21.160	25.167	30.146	25.346	26.127	25.928	24.407	33.229
Gaussian	20.002	25.004	25.329	25.296	25.983	25.813	26.690	22.604
Speckle	22.092	25.641	26.718	25.797	26.555	26.574	27.855	24.618
poisson	25.622	26.295	28.289	26.291	27.152	27.418	29.074	27.576

DENOISING SET OF 10 IMAGES:

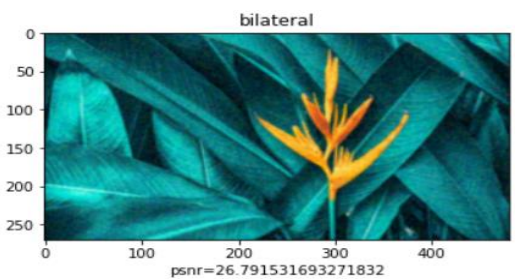
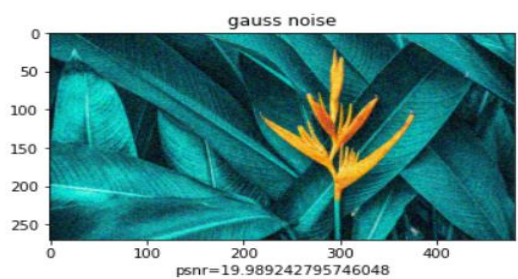
For Salt and Pepper Noise:

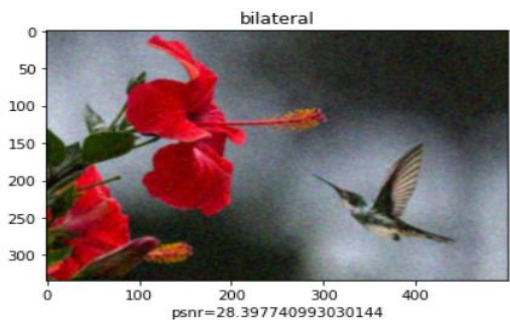
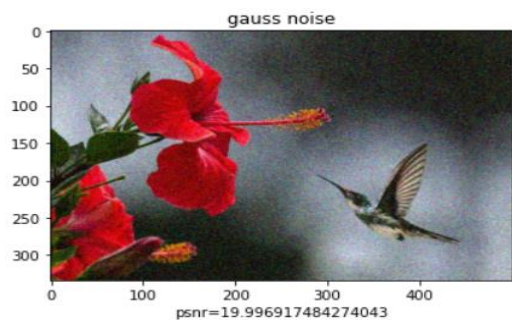
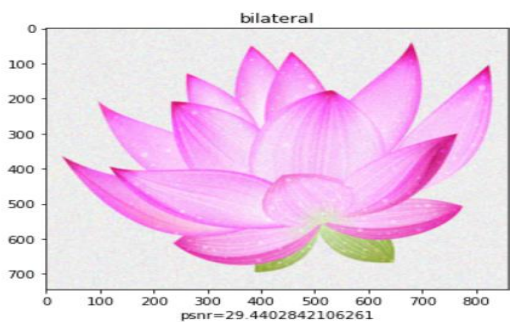
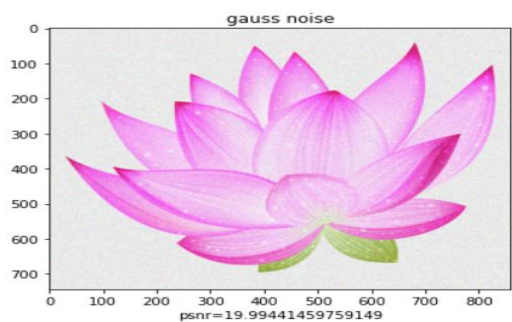
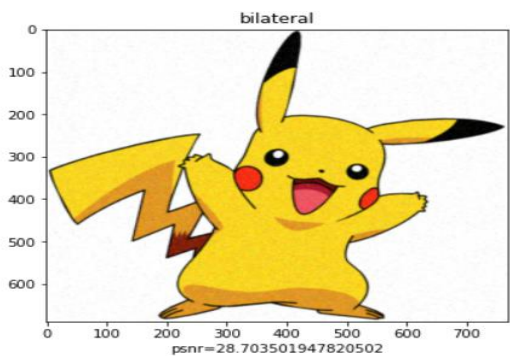
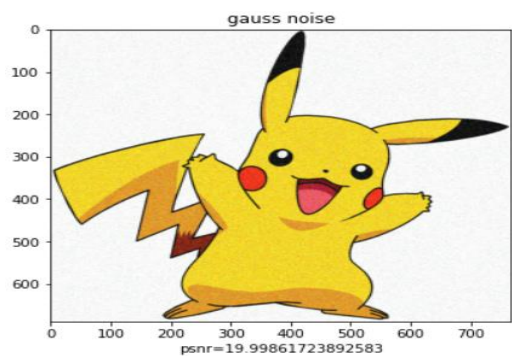
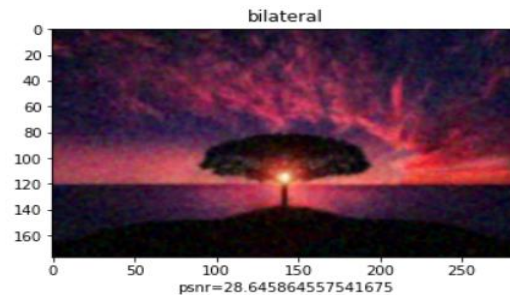
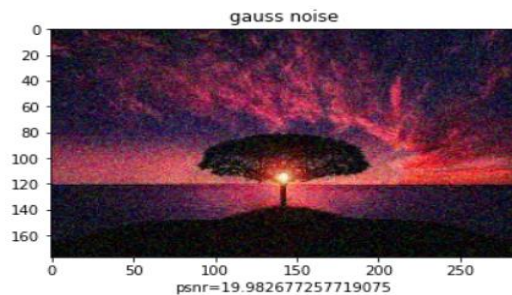
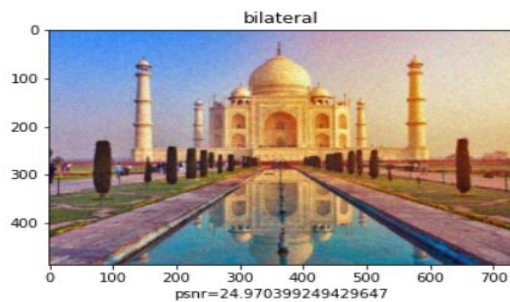
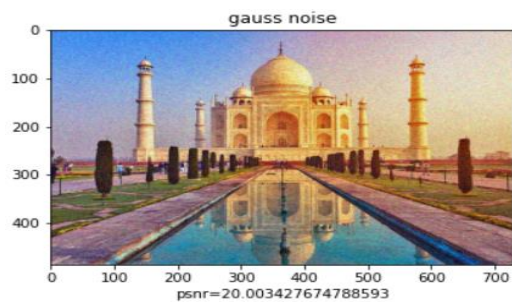


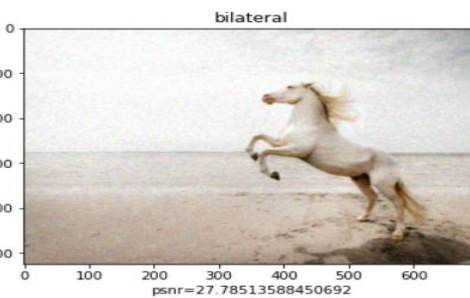
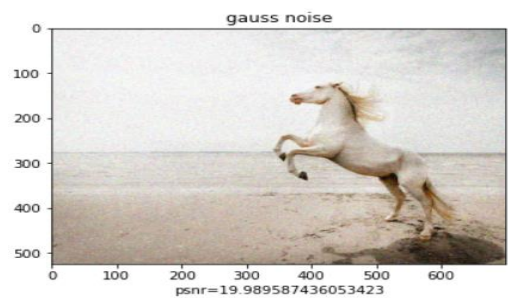
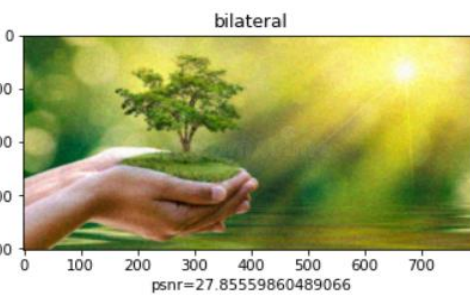
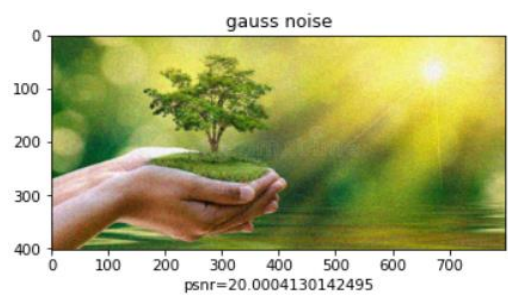
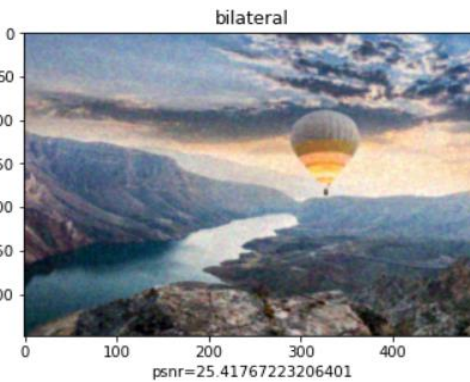
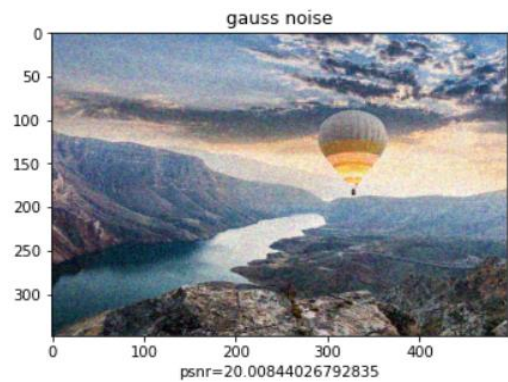
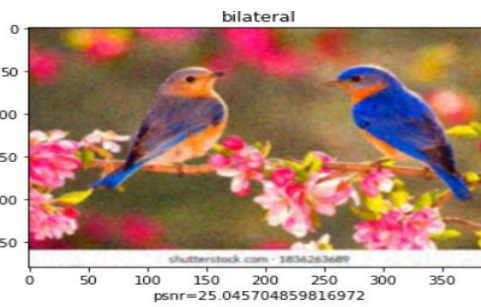
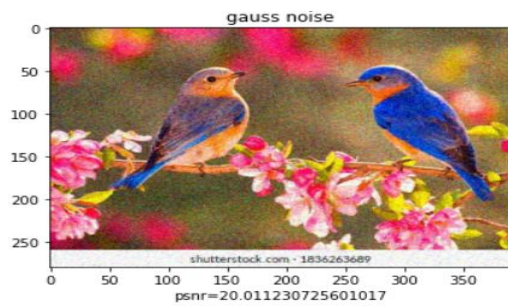




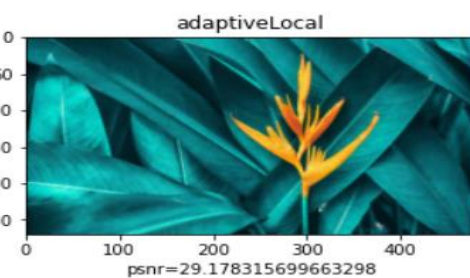
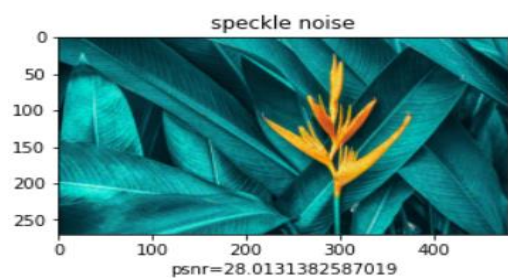
For Gaussian Noise:

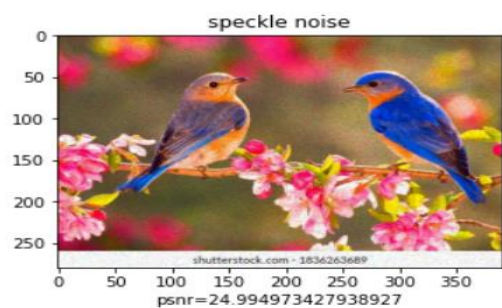
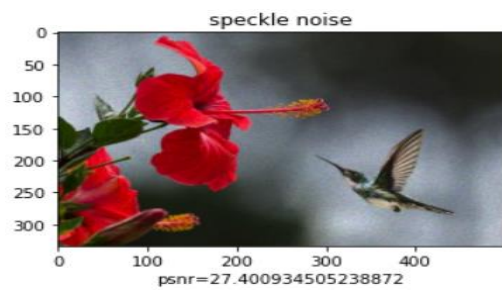
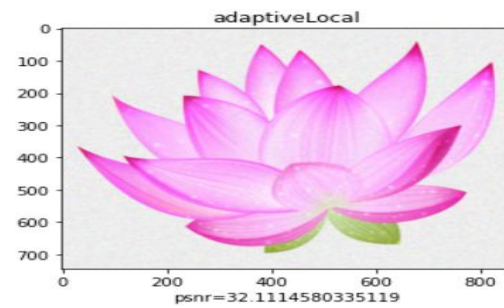
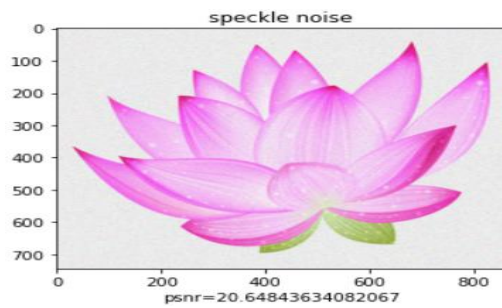
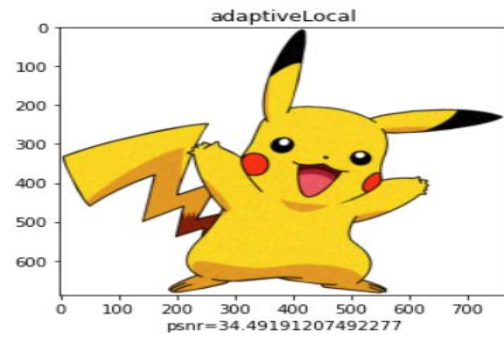
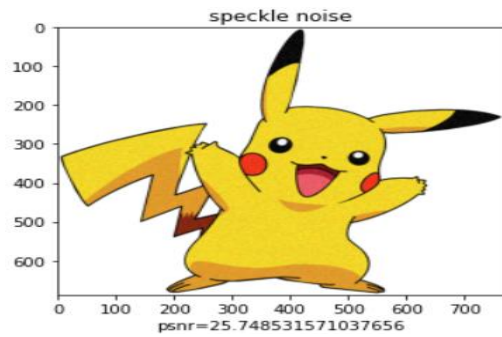
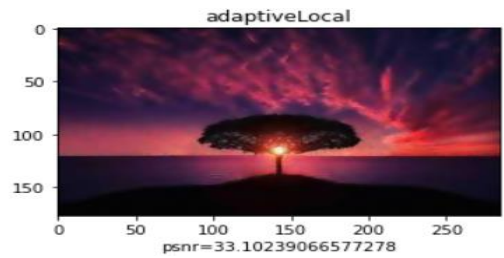
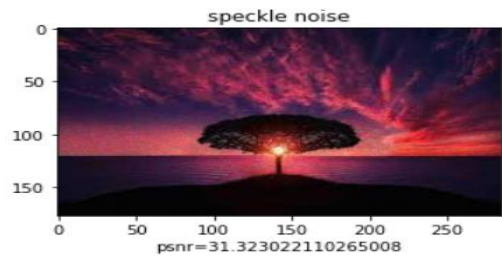
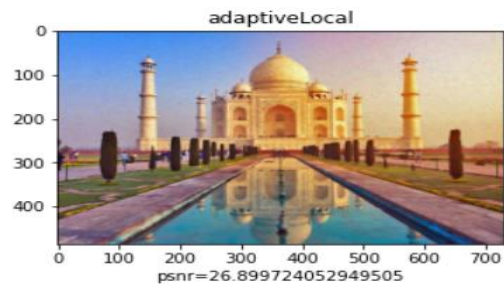
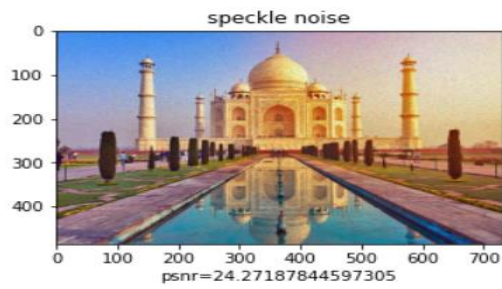


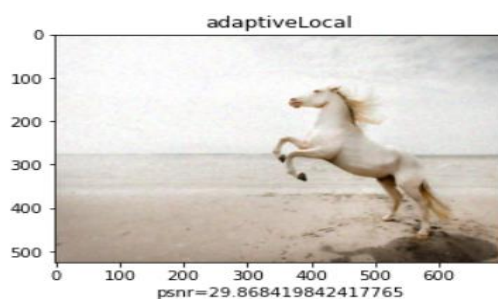
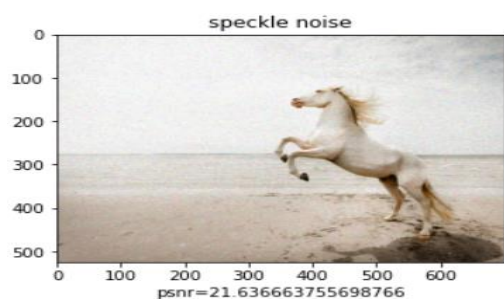
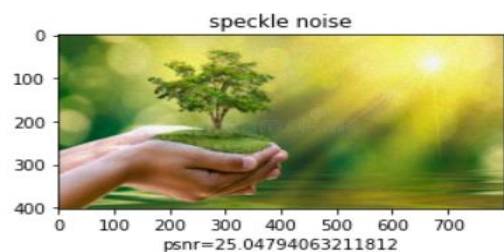
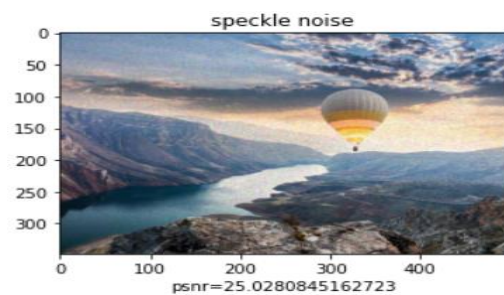




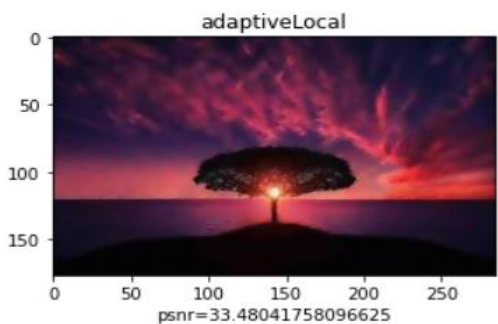
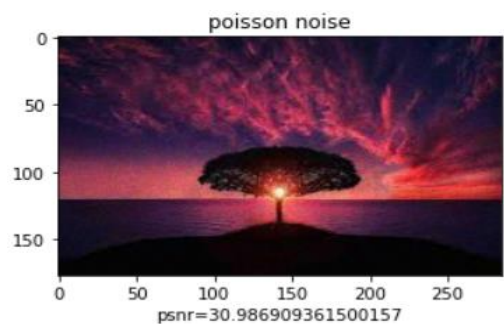
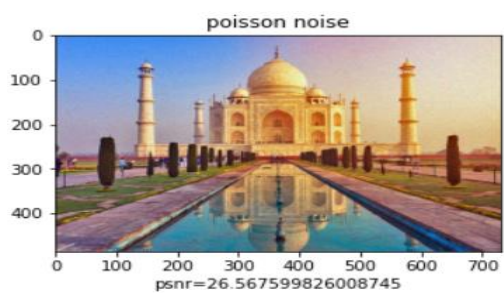
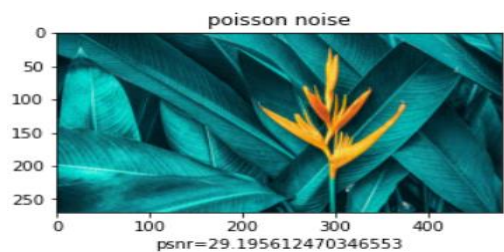
For Speckle Noise:

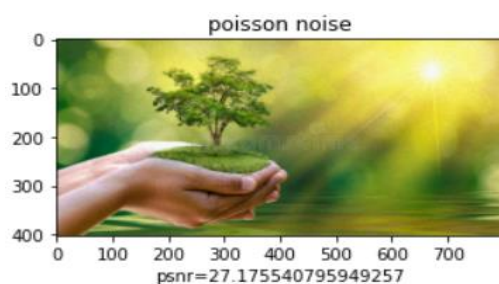
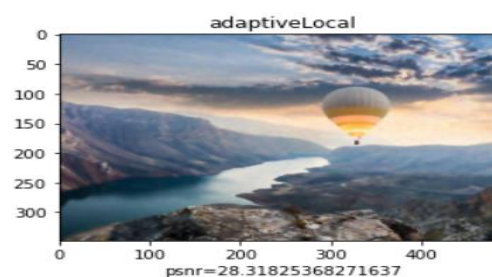
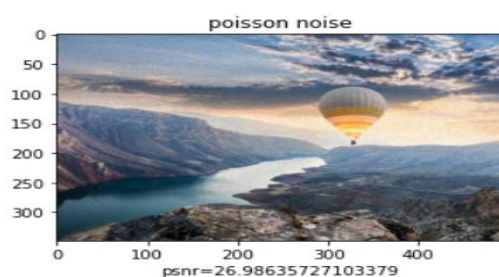
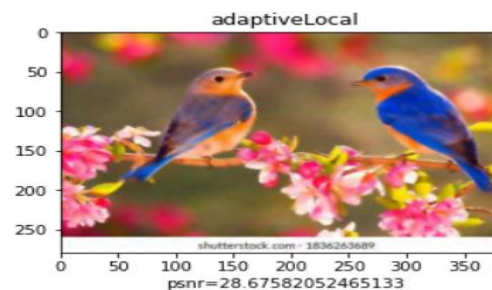
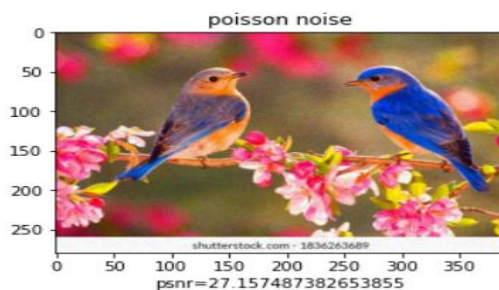
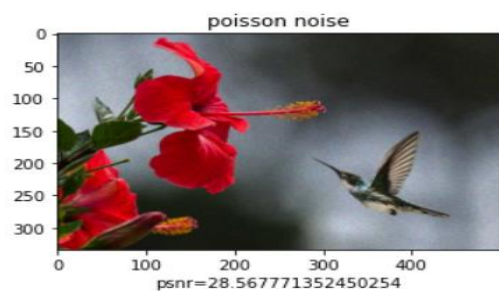
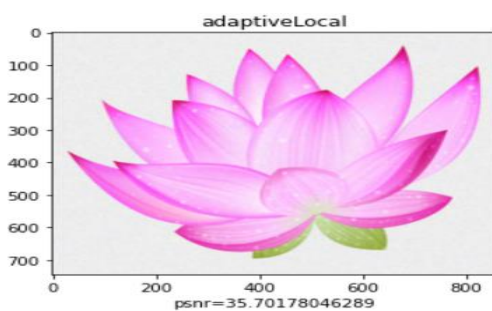
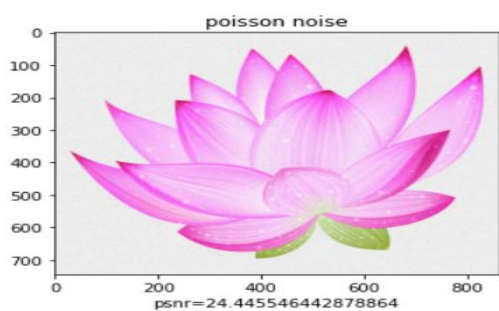
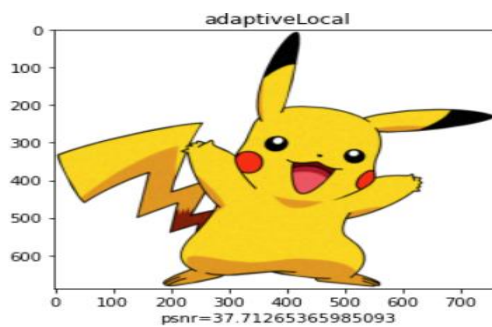
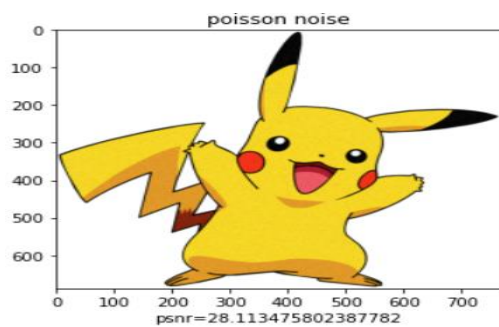


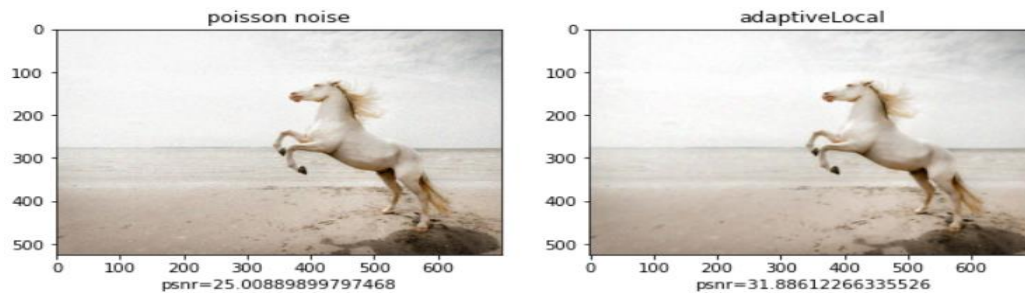




For Poisson Noise:







CONTRIBUTIONS:

Varshini R – Adding noises to the images and analysing the outputs.

Niveda K – Mean, Median, Bilateral, Adaptive Median filters

Anuhya S – Gaussian, Wiener, Adaptive Local filters and analysing the outputs.

GITHUB LINK:

<https://github.com/Anuhya27/Image-Deonising>

REFERENCES:

- <https://iq.opengenus.org/image-denoising-and-image-processing-techniques/>
- <https://www.ijstr.org/final-print/sep2019/Comparative-Analysis-Of-Various-Filtering-Techniques-In-Image-Processing.pdf>
- <https://www.irjet.net/archives/V6/i10/IRJET-V6I10148.pdf>