University of Burgundy and University of Palermo

Image Restoration

Kimberly Grace Sevillano Colina, Chiara Lupo
& Alessandro Giambanco

1 Introduction

In this assignment, we compare image restoration techniques for reducing noise in corrupted images. Given a grayscale-level image I corrupted with noise N, we will:

1. Characterize the type of noise affecting the image (e.g., Gaussian, Salt-and-Pepper, Uniform)
2. Devise a sequence of image processing steps to reduce the image noise
3. Use a Markov Random Field (MRF) to denoise the image

2 Characterizing Noise Type

In our work, we use various techniques to analyze and infer the noise type in an image. Our approach includes the evaluation of histograms from a Region of Interest (ROI), the relationship between pixel intensity and noise magnitude across the full image, the probability of white and black pixel occurrence in the ROI, the correlation between pixel intensity and noise magnitude, and the standard deviation of noise within the ROI.

2.1 Method

We classify **Poisson noise** based on the statistical properties of a Poisson process. A Poisson process is characterized by its variance being equal to its mean, mathematically expressed as:

$$\mathrm{Var}(X) = E[X] = \lambda \tag{1}$$

where λ is the mean (or rate) of the Poisson process. This relationship leads to the property where the noise magnitude is proportional to the square root of the pixel intensity, mathematically described as:

$$\sigma = \sqrt{\lambda} \tag{2}$$

where σ represents the standard deviation, equivalent to noise magnitude in our case. Therefore, plotting pixel intensity against noise magnitude for an image corrupted with Poisson noise should produce a positive trend. Moreover, the histogram of the ROI exhibits a Gaussian-like bell shape, but the difference between the variance and the square root of the mean (denoted as dif_roi) within the ROI should be small, serving as a distinctive feature for Poisson noise.

For **Salt and Pepper noise**, we consider the high probabilities of both white and black pixels in the ROI, both of which should be greater than 0.009. The histogram of Salt and Pepper noise is characterized by a non-uniform distribution with noticeable peaks at the extremes, representing the 'salt' and 'pepper' noise points. This irregularity is reflected in the pixel intensity-noise magnitude plot, which appears scattered and lacks a clear trend.

Gaussian noise is identified by its standard deviation (std_noise) and mean value both being close to zero. The mean is an important parameter when considering Gaussian noise. However, in our assignment, we didn't consider the mean as a distinguishing feature because the mean values of all images were observed to be close to zero, which made it less discriminative. Nonetheless, in a histogram, Gaussian noise exhibits a bell-shaped distribution, characteristic of a Gaussian distribution. The pixel intensity-noise magnitude plot may not present a clear trend, but the properties of the histogram and the standard deviation of the noise help in classifying Gaussian noise. Thus, the focus was given more to the standard deviation of the noise (std_noise), which proved to be more effective in distinguishing Gaussian noise from others.

2.2 Results

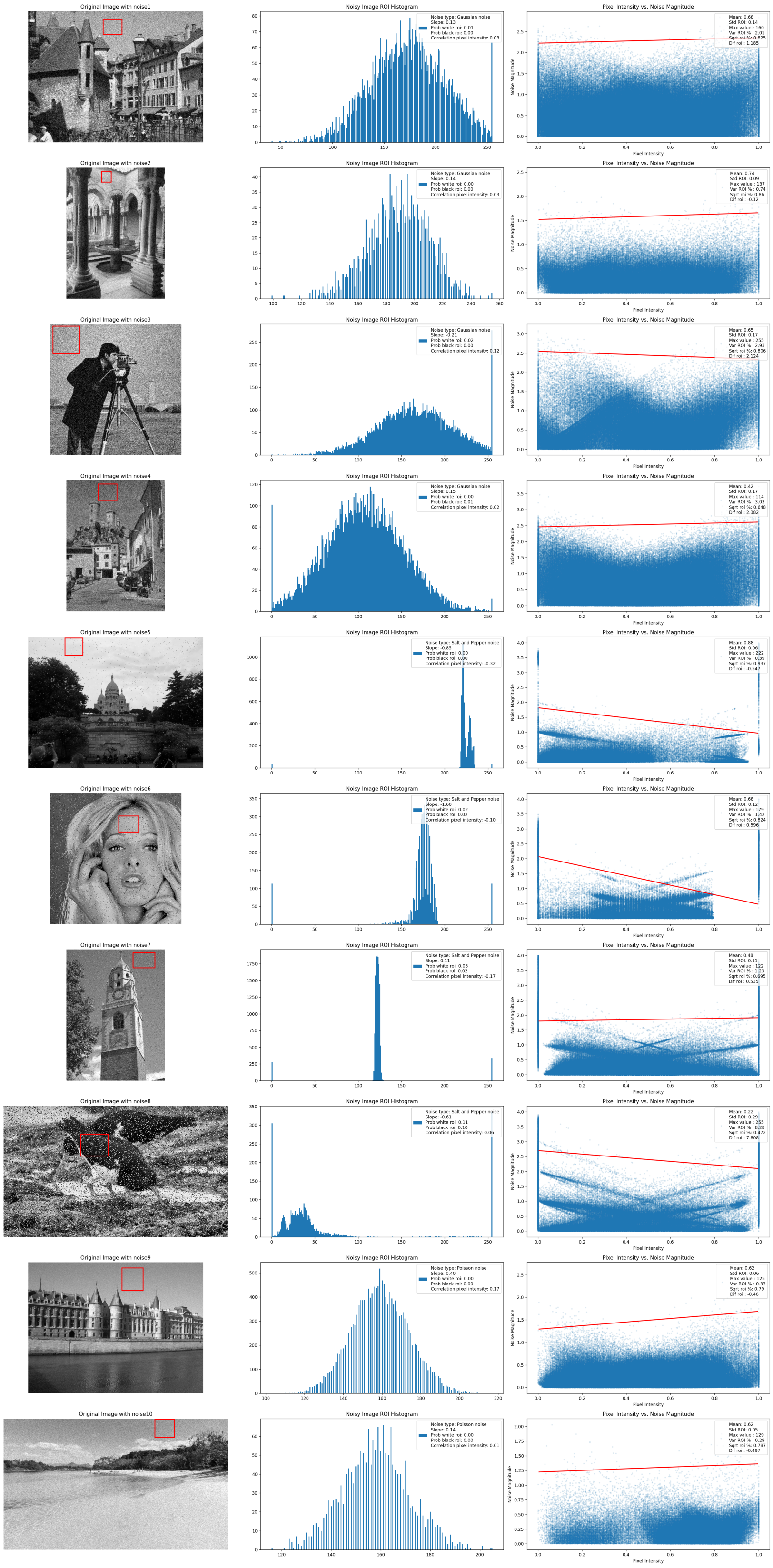


Figure 1: Through this comprehensive analysis, we can accurately identify and characterize the type of noise present in an image, thereby assisting in further processing and analysis stages.

3 Image Processing Steps

3.1 Method and Results

We use two methods, bilateral filter and FFT (Butterworth) + Non-Local Means, for denoising Gaussian noise. The bilateral filter preserves edges and details by considering both spatial distance and intensity similarity between pixels. It achieves noise reduction while maintaining image quality. The FFT (Butterworth) + Non-Local Means method combines frequency domain filtering with non-local patch-based averaging. The FFT transforms the image to the frequency domain, where noise appears as high-frequency components. The Butterworth filter attenuates noise while preserving image details. Non-Local Means denoising compares patches and averages pixel values based on similarity, effectively reducing noise and enhancing image details. By combining these techniques, we achieve better noise reduction and improved image quality compared to using each method individually.

Figure 2: Denoising images with Gauss noise

For denoising salt and pepper noise, we employ two filters: the median filter and the combination of median filter and non-local means. The median filter is a spatial domain filter that replaces each pixel with the median value of its neighboring pixels. This filter effectively reduces salt and pepper noise by smoothing out isolated noisy pixels. The second denoising method combines the median filter with non-local means. In this approach, we first apply the median filter to remove salt and pepper noise. Then, we use non-local means denoising, which compares patches in the image and averages pixel values based on their similarity. By combining these two filters, we can effectively suppress salt and pepper noise while preserving image details and edges, resulting in improved image quality.

Figure 3: Denoising images with Salt and Pepper noise

In the denoising of Poisson noise, we utilize two methods: the Anscombe transform combined with wavelet denoising and the Anscombe transform combined with wavelet denoising and non-local means (NLM) denoising. The first method involves applying the Anscombe transform to the Poisson noisy image. This transformation stabilizes the variance and linearizes the relationship between the mean and variance, making it suitable for further denoising. After the Anscombe transform, we perform wavelet denoising, which effectively removes noise while preserving image details in different frequency bands. Following the wavelet denoising step, we apply an inverted Anscombe transform to revert the image back to the original Poisson scale.

The second method follows a similar approach, starting with the Anscombe transform to stabilize the variance. Then, we perform wavelet denoising to reduce noise in the image. Finally, we apply non-local means (NLM) denoising, which considers the similarity between patches in the image and averages pixel values accordingly. This further enhances the denoising performance by effectively preserving image structures and textures.

By utilizing these methods, we can effectively mitigate Poisson noise in the images while preserving important image features, resulting in improved image quality and clarity.

Figure 4: Denoising images with Poisson noise

Table 1: Comparing Denoising Methods for Gaussian noise

Image	**Method**	**MSE**	**PSNR**
Image Original 4.JPG	Noisy image	100.461	28.111
Image R4.bmp	Bilateral Filter	97.528	28.24
Image R4.bmp	FFT + NLM	100.834	28.095
Image Original 20.jpg	Noisy image	84.388	28.868
Image R20.bmp	Bilateral Filter	87.511	28.71
Image R20.bmp	FFT + NLM	84.72	28.851
Image Original 36.bmp	Noisy image	103.87	27.966
Image R36.bmp	Bilateral Filter	99.431	28.156
Image R36.bmp	FFT + NLM	100.879	28.093
Image Original 42.bmp	Noisy image	103.277	27.991
Image R42.bmp	Bilateral Filter	99.573	28.149
Image R42.bmp	FFT + NLM	102.913	28.006

Table 2: Comparing Denoising Methods for Salt and Pepper noise

Image	Method	MSE	PSNR
Image Original 47.bmp	Noisy image	1.052	47.912
Image R47.bmp	Median Filter	22.167	34.674
Image R47.bmp	Median + NLM	28.838	33.531
Image Original 63.bmp	Noisy image	5.391	40.814
Image R63.bmp	Median Filter	33.509	32.879
Image R63.bmp	Median + NLM	39.645	32.149
Image Original 73.bmp	Noisy image	4.479	41.619
Image R73.bmp	Median Filter	35.494	32.629
Image R73.bmp	Median + NLM	41.041	31.999
Image Original 89.jpg	Noisy image	21.549	34.797
Image R89.bmp	Median Filter	66.632	29.894
Image R89.bmp	Median + NLM	72.467	29.529

Table 3: Comparing Denoising Methods for Poisson noise

Image	Method	MSE	PSNR
Image Original 91.bmp	Noisy image	60.425	30.319
Image R91.bmp	Anscombe + wavelet + Inverted Anscombe + NLM	70.828	29.629
Image R91.bmp	Anscombe + Wavelet + NLM	90.921	28.544
Image Original 105.jpg	Noisy image	71.071	29.614
Image R105.bmp	Anscombe + wavelet + Inverted Anscombe + NLM	76.057	29.319
Image R105.bmp	Anscombe + Wavelet + NLM	88.608	28.656

4 Markov Random Field Denoising

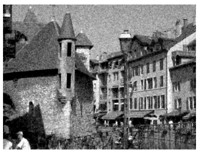
4.1 Method

We implement a Markov random field (MRF) algorithm using iterated conditional modes (ICM) for image denoising. We calculate and compare "energies" for different gray levels in noisy and de-denoised images. The "energy" comprises a data error term and a penalty term, representing, respectively, the squared difference between the noisy and de-denoised image, and a measure of disparity between neighboring pixels in the de-denoised image.

The algorithm selects the gray level that minimizes the total energy for each pixel. If the energy decreases as the pixel value changes to the minimum energy gray level, the pixel value is updated.

After each iteration, the mean square error (MSE) and peak signal-to-noise ratio (PSNR) between the noisy image and the de-denoised image are calculated and stored. After all iterations, the denoised image and the MSE and PSNR values are stored for further analysis.

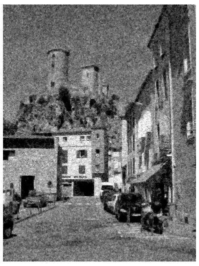
4.2 Results



Denoised 4.bmp

Denoised 20.bmp

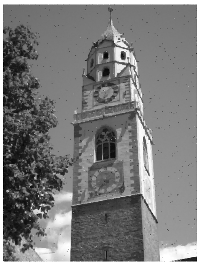
Denoised 36.bmp



Denoised 42.bmp

Denoised 47.bmp

Denoised 63.bmp



Denoised 73.bmp

Denoised 89.bmp

Denoised 91.bmp

Denoised 105.bmp

Image	**it1**	**it2**	**it3**	**it4**	**it5**	**it6**	**it7**	**it8**	**it9**	**it10**
4.bmp MSE	2939.57	1537.89	2815.68	1713.77	2800.14	1785.97	2798.88	1819.90	2801.57	1841.03
4.bmp PSNR	13.45	16.26	13.63	15.79	13.66	15.61	13.66	15.53	13.66	15.48
20.bmp MSE	2649.20	1413.16	2541.10	1604.00	2536.73	1697.57	2558.18	1769.29	2596.47	1836.63
20.bmp PSNR	13.90	16.63	14.08	16.08	14.09	15.83	14.05	15.65	13.99	15.49
36.bmp MSE	339.74	339.86	382.76	378.43	407.17	398.97	423.11	411.46	432.04	419.26
36.bmp PSNR	22.82	22.82	22.30	22.35	22.03	22.12	21.87	21.99	21.78	21.91
42.bmp MSE	366.26	227.30	372.21	260.85	378.44	274.76	381.35	280.44	382.96	283.84
42.bmp PSNR	22.49	24.56	22.42	23.97	22.35	23.74	22.32	23.65	22.29	23.60
47.bmp MSE	671.32	399.72	685.32	465.29	701.40	494.30	712.19	509.01	717.87	516.20
47.bmp PSNR	19.86	22.11	19.77	21.45	19.67	21.19	19.60	21.06	19.57	21.00
63.bmp MSE	274.18	197.15	307.74	238.95	325.85	259.16	335.92	269.82	341.81	275.94
63.bmp PSNR	23.75	25.18	23.25	24.35	23.00	23.99	22.87	23.82	22.79	23.72
73.bmp MSE	1210.14	1034.45	1245.12	1118.82	1279.76	1160.02	1301.56	1180.36	1313.73	1191.73
73.bmp PSNR	17.30	17.98	17.18	17.64	17.06	17.49	16.99	17.41	16.95	17.37
89.bmp MSE	2066.74	1130.01	2037.72	1289.66	2049.19	1355.72	2062.37	1390.61	2072.23	1412.90
89.bmp PSNR	14.98	17.60	15.04	17.03	15.01	16.81	14.99	16.70	14.97	16.63
91.bmp MSE	6147.49	3366.88	5897.03	3720.42	5841.98	3849.05	5827.12	3916.64	5823.76	3947.22
91.bmp PSNR	10.24	12.86	10.42	12.42	10.47	12.28	10.48	12.20	10.48	12.17
105.bmp MSE	1091.94	903.04	1082.06	954.83	1099.82	984.79	1117.72	1007.19	1134.44	1022.45
105.bmp PSNR	17.75	18.57	17.79	18.33	17.72	18.20	17.65	18.09	17.58	18.03

Table 4: MSE and PSNR values for denoising images for the MRF process

5 Conclusion

As can be seen, the algorithms we use to denoise each image with noise have better results than the MRF iterations, and are even visually superior. Perhaps further iterations and better fitting of the data will allow us to obtain better results in future work.