DTA095A - Lab 1 Report

Noise & Spatial Filters / Deblurring: Inverse and Wiener

Name: Andrea Faccioli
Date: 17/09/2025

1 Introduction

In this work, the effects of different types of **noise** and **filtering methods** on digital images have been investigated. **Image degradation** caused by noise represents a fundamental challenge in **image processing**, as it affects both quantitative metrics such as **PSNR** and perceptual quality as experienced by human observers. By applying a variety of common noise models (**AWGN**, **Poisson**, **Speckle**, and **Salt-and-Pepper**) and evaluating the performance of **spatial** and **frequency-domain filtering** techniques, the study highlights the trade-offs between **noise reduction** and **detail preservation**.

2 Experimental Setup

The experiments have been carried out using **Python** within a **Jupyter Notebook** environment. Core libraries included **NumPy** for numerical operations, **Torch** for tensor-based computations and GPU acceleration, and **scikit-image** for image manipulation and **filtering functions**. All scripts and functions used for **noise generation**, **filtering**, and **metric evaluation** are provided in Appendix A.2 and B.2.

3 Results and Analysis

3.1 Noise types

Different types of noise have been applied to the clean image; For each type, various level of the parameters (`sigma`, `scale`, `amount`) have been tested: their effects are shown in section A.3. AWGN (Additive white gaussian noise) and Speckle Noise create a more uniform "grain" across the entire image, while Salt-and-pepper noise introduces isolated pure white ("salt") and pure black ("pepper") pixels scattered randomly across the image, it does not create a "grain" but rather destroys individual pixels. Poisson noise manifests itself as signal-dependent grainy texture. The different nature of these noise families can lead to inconsistency between rigorous metrics (e.g. PSNR, Peak signal-to-noise ratio) and human perception; in Figure 1, we can observe 4 images with distinct noise applied, but with similar PSNR value.

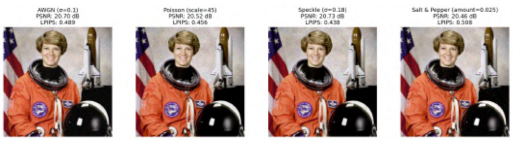


Figure 1: Similar PSNR images affected by 4 different noise types

Despite the nearly identical PSNR value, the degradation caused by salt-and-pepper noise is very noticeable, as single bright or dark pixels are highly annoying and perceptually very significant.

3.2 Denoising by filtering

Denoising performance for each type of noise and for each filtering method have been reported in Table 1. For each noise type, the top performing configuration in term of PSNR has been highlighted in bold. These metrics are coherent with visual perception: in particular, the *Median filter* is highly effective at removing impulse noise (**Salt-and-pepper**) while preserving edges, as it ignores extreme values, while the *Mean filter* provides good results on the other three noise types (**AWGN**, **Poisson** and **Speckle**).

3.2.1 Mean and Median filtering

In the setting of our experiment, a kernel size of 3x3 pixels obtained the best results for both Mean and Median filtering, achieving a good trade-off between noise reduction and edge preservation.

Table 1: Denoising performance comparison across different filters and noise types. Values show PSNR (dB). LPIPS, together with denoised images, can be found in section A.3

Filter	Parameter	AWGN $\sigma = 0.10$	Poisson $scale = 255$	Speckle $\sigma = 0.20$	Salt-Pepper $amount = 0.05$
Mean	$k = 3$	**26.39**	29.10	**26.28**	24.79
	$k = 5$	24.95	25.92	25.06	24.45
	$k = 7$	23.44	23.99	23.56	23.22
	$k = 11$	21.32	21.64	21.42	21.24
Median	$k = 3$	26.25	**30.03**	25.55	**31.44**
	$k = 5$	26.17	27.59	25.78	27.89
	$k = 7$	25.00	25.70	24.71	25.85
	$k = 11$	22.92	23.26	22.69	23.32
Bicubic Filter	$a = -0.5$	22.88	26.50	22.38	20.54
	$a = -0.75$	22.12	26.01	21.58	19.16
	$a = -1.0$	21.33	25.45	20.74	18.77
	$a = 0.0$	24.17	27.16	23.82	22.17

3.2.2 Bicubic filtering

The bicubic parameter `a` controls a direct trade-off between smoothing and ringing. A higher value (closer to 0, like bilinear) produces smoother results with less detail but no ringing, while a lower value (like -1.0) creates sharper edges with better detail preservation but introduces visible ringing artifacts near edges. The default `a = -0.5` represents a practical compromise between these two extremes. In Figure 2 we can find a comparison on a cropped part of the image denoised with two contrasting value for `a` parameter.

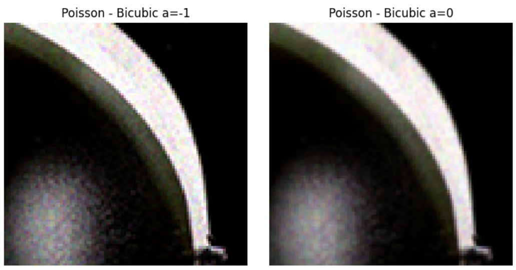


Figure 2: Comparison between a = -1 (ringing artifacts) and a = 0 (smooth)

3.2.3 Bilateral filtering

Bilateral filtering balances noise reduction and detail preservation through two parameters: σ_{space} controls the physical neighborhood size, while σ_{color} regulates intensity similarity tolerance. Large values ($\sigma_{\text{space}} = 3.0$, $\sigma_{\text{color}} = 0.2$) oversmooth and lose texture details, while smaller values ($\sigma_{\text{space}} = 1.5$, $\sigma_{\text{color}} = 0.05$) preserve edges but leave residual noise in uniform regions. An example of this behavior can be found in Figure 3.

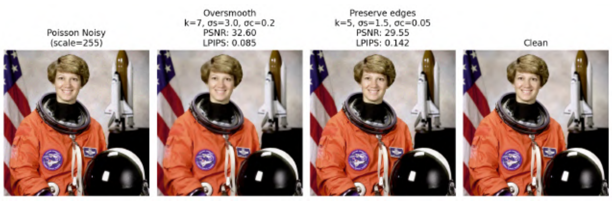


Figure 3: Comparison between two different settings for Bilateral filter

Bilateral filtering performs poorly on salt-and-pepper noise because the extreme outlier pixels strongly distort the intensity similarity weighting, causing the filter to average in corrupted values instead of removing them.

4 Inverse and Wiener filtering

In this session we examined the impact of different blur levels (mild, moderate, and strong Gaussian blur) combined with two types of noise (additive white Gaussian noise and salt-and-pepper noise) at multiple intensity levels. Through quantitative metrics (PSNR and LPIPS) and qualitative visual assessment, the investigation explored the fundamental trade-off between noise amplification and detail recovery inherent in frequency-domain deconvolution methods.

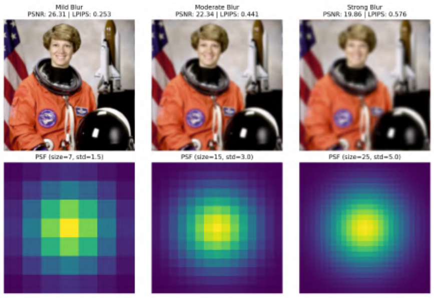


Figure 4: Different blur level degradations

Inverse and Wiener filtering results have been reported in Table 2. The inverse filter amplifies noise because it attempts to perfectly invert the blurring process in the frequency domain, which involves dividing by the frequency response of the blur. This division severely amplifies noise components at frequencies where the blur kernel's magnitude is small or zero. In contrast, the Wiener filter introduces a regularization parameter (Kreg) that acts as a noise-to-signal ratio estimator, trading detail for noise reduction by preventing division by near-zero values. For the tested images, the Wiener filter with a moderate Kreg value (around 5e-2) generally worked best, effectively suppressing noise amplification while preserving meaningful image details. The inverse filter only produced viable results for noiseless or very mildly corrupted images with careful epsilon selection.

Table 2: Deblurring performance comparison across different filters and degradation types. Values show PSNR (dB). LPIPS, together with deblurred images, can be found in section B.3 and B.4

Degradation Type	**Blur Level**	**Inverse Filter** $\epsilon = 0.01$	**Wiener Filter** $K_{reg} = 0.05$
Blur Only	Mild Blur	22.52	**22.97**
	Moderate Blur	**23.90**	22.14
	Strong Blur	**22.76**	20.51
AWGN ($\sigma = 0.01$)	Mild Blur	6.53	**22.91**
	Moderate Blur	6.93	**22.13**
	Strong Blur	6.79	**20.50**
AWGN ($\sigma = 0.05$)	Mild Blur	4.91	**21.86**
	Moderate Blur	5.15	**21.87**
	Strong Blur	5.08	**20.44**
Salt-Pepper ($amount = 0.02$)	Mild Blur	4.87	**20.25**
	Moderate Blur	6.10	**21.32**
	Strong Blur	7.17	**20.22**
Salt-Pepper ($amount = 0.1$)	Mild Blur	4.74	**15.64**
	Moderate Blur	6.14	**18.86**
	Strong Blur	7.44	**19.00**

5 Conclusion

The analysis demonstrates that optimal image restoration requires tailored approaches: spatial filters excel for specific noise types, while frequency-domain methods like Wiener filtering are essential for effective deblurring with noise regularization. The inverse filter's noise amplification confirms that robust restoration demands careful algorithm selection based on degradation characteristics.

A Appendix 1a

A.1 Images collection

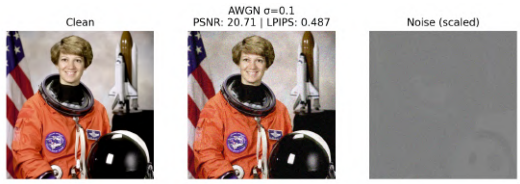
The images processed in Lab1a session, together with their PSNR and LPIPS metrics, can be found here.

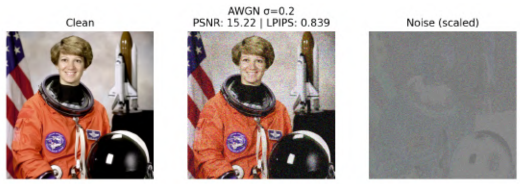
A.2 Source code

The complete code of Lab1a session can be found here.

A.3 Noise degradations

A.3.1 AWGN

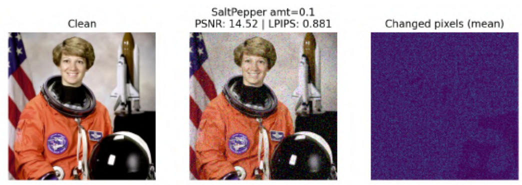




A.3.2 Poisson Noise

A.3.3 Speckle Noise

A.3.4 Salt-and-pepper Noise



B Appendix 1b

B.1 Images collection

The images processed in Lab1b session can be found here.

B.2 Source code

The complete code of Lab1b session can be found here.

B.3 Inverse filtering

B.3.1 Blur-only

Figure 5: Different blur level degradations

B.3.2 AWGN

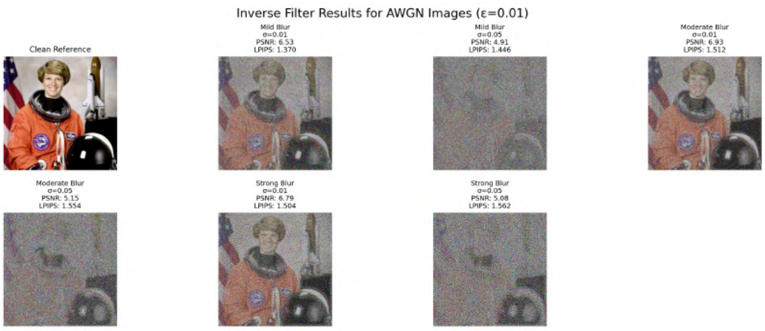


Figure 6: Different blur level degradations

B.3.3 Salt-and-pepper

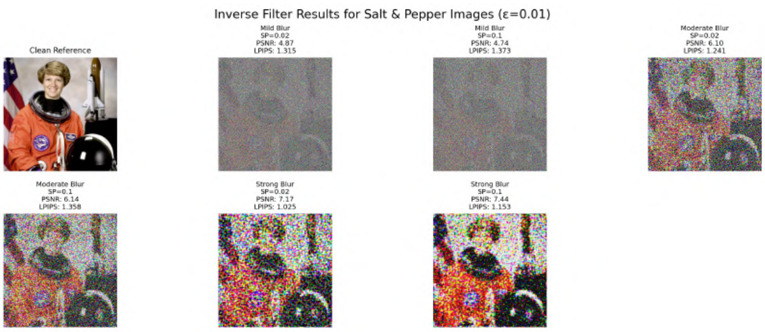


Figure 7: Different blur level degradations

B.4 Wiener filtering

B.4.1 Blur-only

Figure 8: Different blur level degradations

B.4.2 AWGN

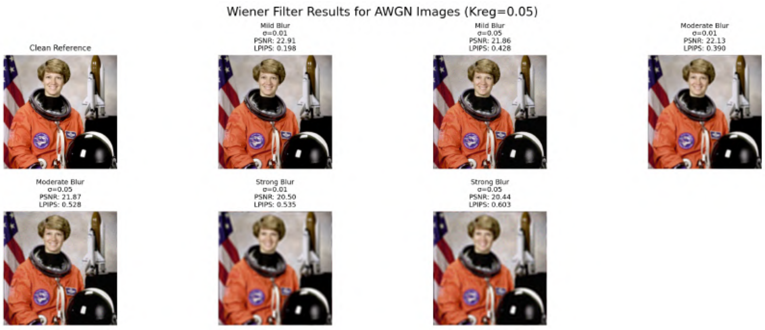


Figure 9: Different blur level degradations

B.4.3 Salt-and-pepper

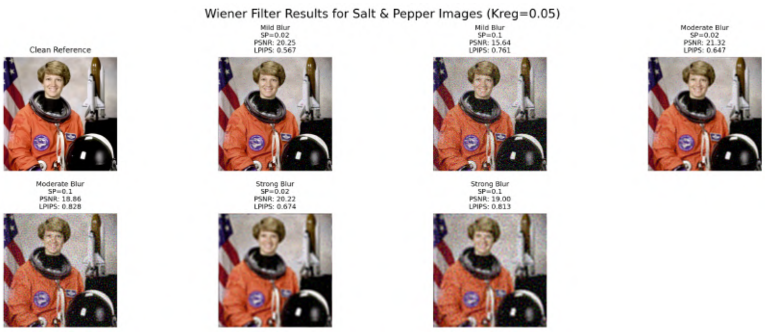


Figure 10: Different blur level degradations