**Digital Image Processing**

**in**

**Image Restoration**

****

**Our presentation Consist of 5 Parts**

* **Definition**
* **How does it work**
* **Results**
* **Advantages & Disadvantages**
* **Applications**

**Let's start**

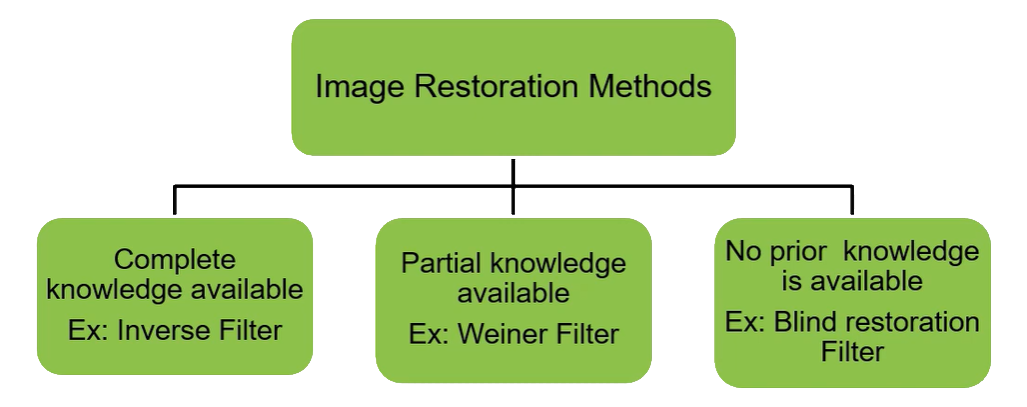
**Definition**

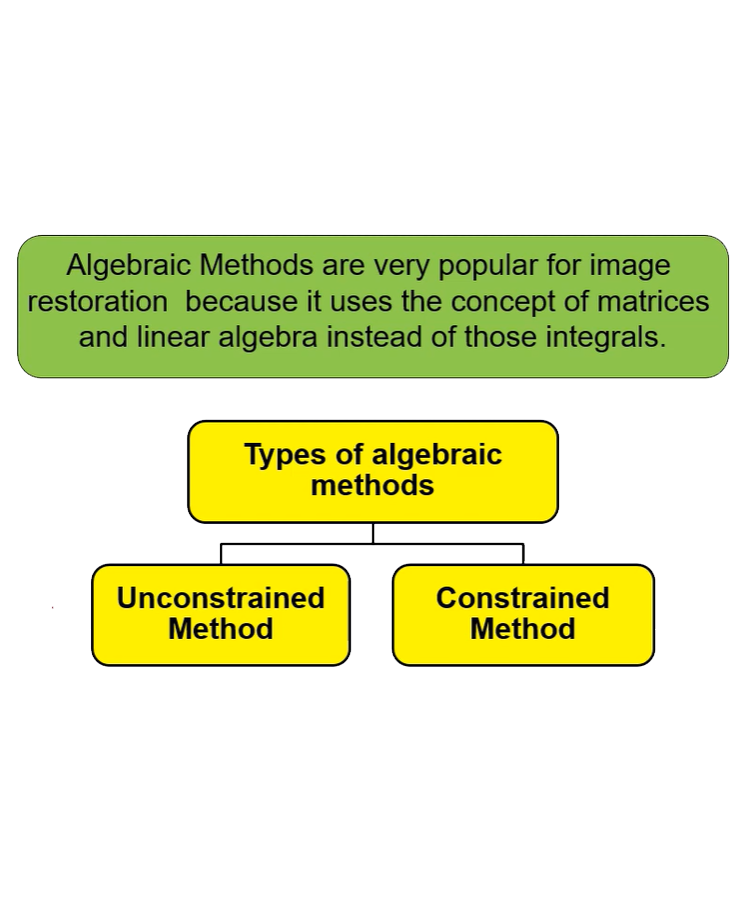
**Image restoration is the operation of taking a corrupt/noisy image and estimating the clean, original image. Corruption may come in many forms such as motion blur, noise and camera mis-focus.**

**Image restoration is performed by reversing the process that blurred the image and such is performed by imaging a point source and use the point source image, which is called the Point Spread Function (PSF) to restore the image information lost to the blurring process.**

**How does it work? & Results**

**Analysis & Techniques**





**Part 1 - Noise and Denoising**

**restoration usually considers the image formation process:**

where  is the observed or acquired image, is the ideal or original image, is the filter that explain the distortion caused in the image that is defined as a spreading of data from a given pixel to a neighbourhood centred in , and is a stochastic process that explains the noise in the image.

**Image restoration** is a process to obtain an estimate of the original image .

**Image denoise** deals with reversing the noise process. To make it more mathematically tractable, the image formation process often considered is one that assumes additive noise:

where is a function that describes how noise is added to the image

**Noise distributions and artificial noise generation**

Let us study a few noise distributions, which are often modelled via some parametric probability distribution, such as, for example:

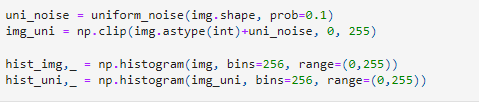
* Uniform: all range of values that may be added to the image are equally probable
* Gaussian (normal): the values to be added to the image follow a normal distribution, with some mean (usually zero) and standard deviation so that values far from the mean are less probable from both sides (positive and negative)

In addition to noise that is given by addition, there are also nonlinear noise process, for example

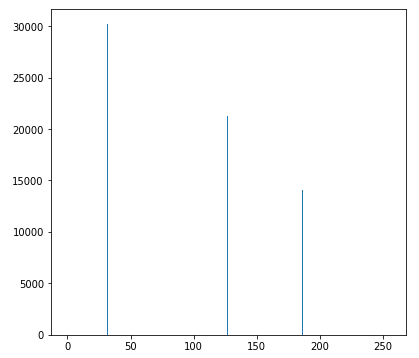
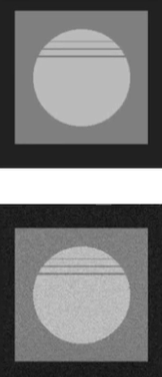
Impulsive: represents an error in a given pixel, turning it into specific values. The most commonly observed are values 0 (also known as pepper noise), 255 (also known as salt noise), or both (salt and pepper). Therefore, this only considers a limited set of possible values, thus 'impulsive'

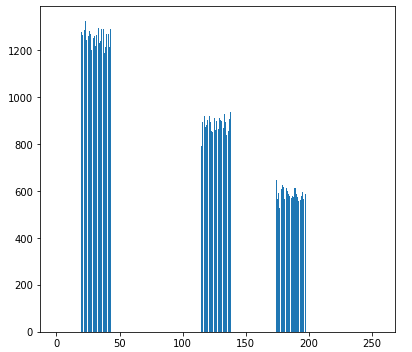
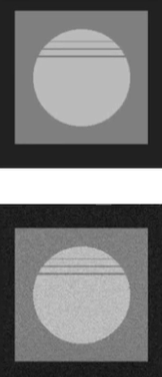
Let us start with the uniform noise

**Code:**

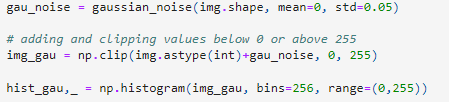
****

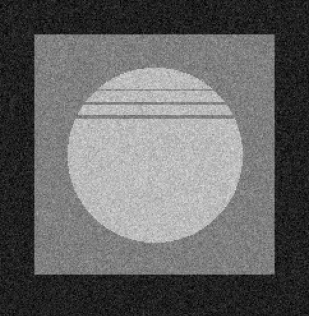
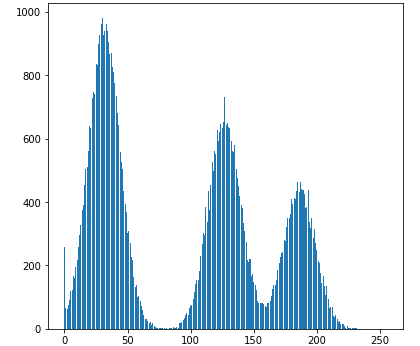
**Before & after adding noise**

****

****

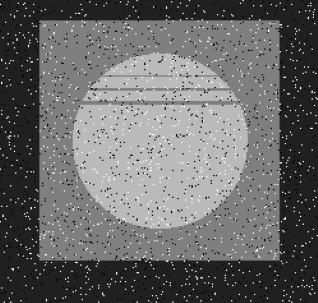
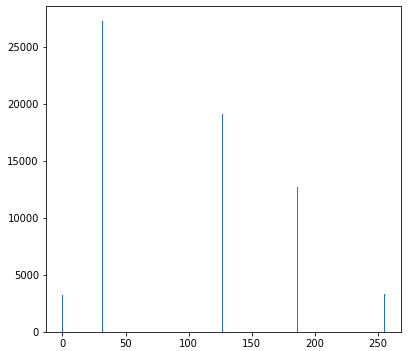
Note how the original image, which is noise-free has just three intensities values (32, 127 and 186). When we add uniform noise, values are added (or subtracted) from those intensities, producing a wider range of intensities around the original values. Note how the values are approximately uniform





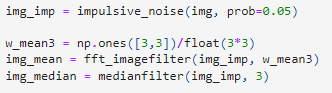
In the case of Gaussian noise, the values are more likely around the intensity values, which are the mean of the distribution. Note the histogram shape, which indicates a multimodal normal distribution.

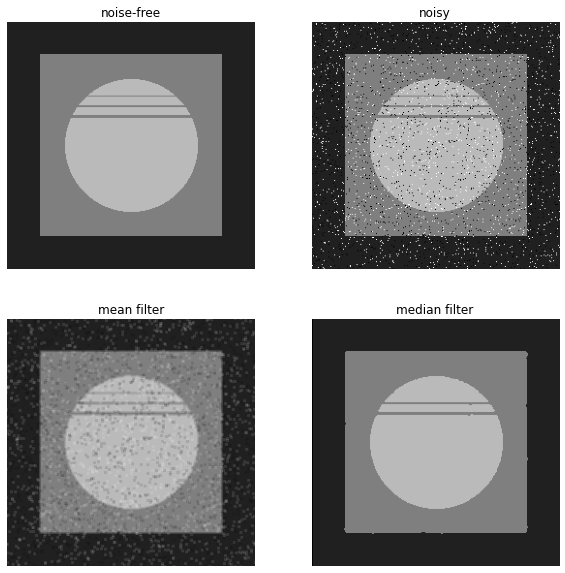




**Denoising methods**

In principle, considering an additive noise formation, you could just generate the noise and subtract it from the image. However, this may not produce the desired results. Therefore, smoothing filters such as the Gaussian filter, the Median filter may be used to reduce the variance locally in an attempt to denoise the images.

****

****

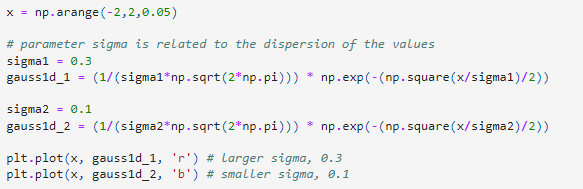
This shows how relevant is to know the noise process before desining a filter to remove it

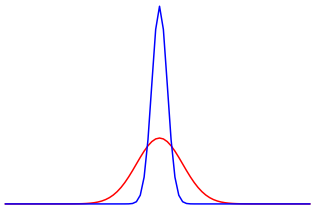
**Part 2 - Blur**

**Degradation function and Point Spread Function**

Let us build a function to create a Gaussian filter (often called kernel as well) as a way to simulate the point spread function. This function may be used to simulate the blur introduced by an acquisition system.

Note that what the image formation considers is a function that **spreads the energy of a point to surrounding pixels**, so it is indeed what a filter such as the Gaussian does. Let us take a look at the function:

****

****

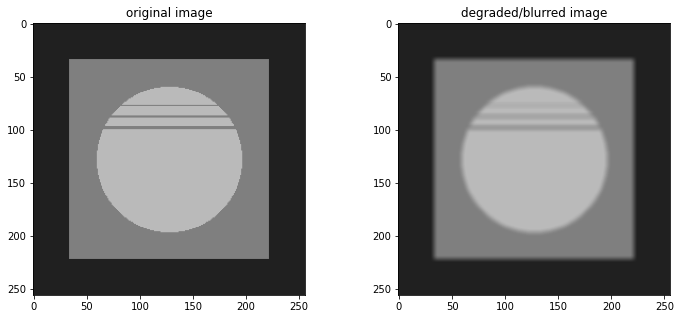
Note how the blue plot distributes the values in a region that is more concentrated around 0, while the red line defines a distribution that spreads values. By increasing , we approach the uniform distribution.

In order to use such function in images we need a 2D version of it, and discretized to be encoded in filters to be applied via convolution.

**Simulating blur**

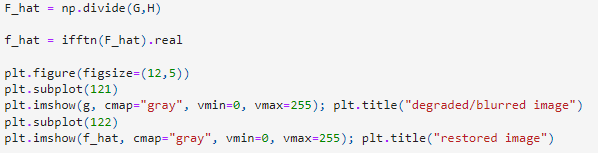
In order to simulate blur, we are going to convolve an image with one of such functions using the Fourier Transform, creating an image. In the Fourier domain:

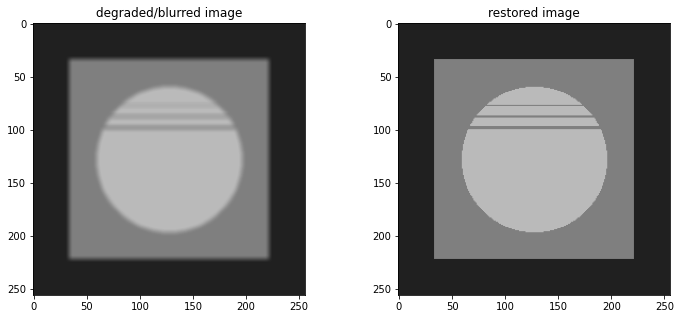
**Fourier transform (FT)** is a mathematical transform that decomposes functions depending on space or time into functions depending on spatial or temporal frequency

****

**Inverse filter**

The inverse filter is an attempt to inverse the convolution process, obtaining an estimate of the original image

****

****

As we can see, the results of the inverse filter are really good! However, we assumed a noise-free scenario and a well-behaved H function.

**Advantages & Disadvantages**

**Advantages:** This method does not require prior knowledge for the removal of noise and blur. It efficiently removes the blurred and it is mostly like to the weiner filter and it overcome the problem of inverse filter.

**Disadvantages:** of the restores method are edge artifacts in the form of horizontal and vertical stripes on the image and lack of information about the optimal number of algorithm iterations.

**Applications of Image Restoration**

* **Identification**
* **Computer Vision**
* **Image Enhancement**
* **Image Analysis in Medical**
* **Space Image Analysis**
* **Bottling and IC Industry**

**Denoising and deblurring using Wiener Filter**

For a signal d(n), possessing a noise n(n), we have effective signal as 𝑥(𝑛)=𝑑(𝑛)∗ℎ(𝑛)+𝑛(𝑛) in the sample domain. Our aim is to reconstruct the signal d(n) from a given x(n). For this we use Wiener Filters.

Wiener Filter is an Adaptive Filter. It works based on the PSF of the functions; such that based on the training data it sees, the filter will find a 𝑔(𝑛) to convert 𝑥(𝑛) to reconstructed signal 𝑠(𝑛) to match 𝑑(𝑛). In frequency domain for PSD's D and N respectively, the transfer function corresponds to,

function is based on the fact that using Wiener Filter in frequency domain for PSD's D and N

Finally, 𝐺 (𝑢, 𝑣) gives us the final filter coefficients

Input Output



