Unit – III

Image Restoration-

Image restoration refers to the process of improving or recovering the quality of a degraded image. The degradation could be caused by factors such as noise, blur, compression artifacts, or low resolution. The goal of image restoration is to reverse or minimize these effects to produce a clearer and more visually accurate image.

Image restoration is the process of enhancing or upgrading an image's quality with the aid of picture editing software.

Restoration is all about how to make up for or undo imperfections that diminish an image. Shutter speed, noise, and camera focus issues are just a few examples of how things might get worse. In situations when motion blur occurs, it is feasible to predict the blurring function quite accurately and "undo" the blurring to regain the original image. The best we can expect when noise distorts an image is to make up for the damage it caused. Here are a few techniques for picture restoration utilized in the realm of image processing.

- **Reverse Filter.** This approach involves viewing an **image degrader** while assuming a well-known blurring function. You'll observe that restoration works well when there is no noise and less well when there is.
- **Weiner Filters**. In this section, you put wiener filtering to use to restore images, giving you the best possible balance between inverse and de-noising filtering. You'll observe that the outcome is generally superior to that of inverse filtering done directly.
- **Wavelet Restoration.** To recover the image, you use three wavelet-based methods.
- **Blind Deconvolution**. You make no assumptions about the image with this technique. You are completely ignorant in regards to the blurring feature or the added noiset. You'll discover how challenging it is to restore an image when you don't know anything about it.

Key Methods of Image Restoration:

1. **Noise Reduction**
    - **Median Filtering**: A non-linear filter that replaces each pixel value with the median value of its neighbors, effective for removing salt-and-pepper noise.
    - **Gaussian Filtering**: Smooths the image by averaging nearby pixels, which is good for reducing Gaussian noise.
    - **Wiener Filter**: A filter that adapts to local variations in the image, optimizing the balance between noise removal and preserving details.
2. **Deblurring**
    - **Inverse Filtering**: Tries to reverse the effects of blur by modeling the blurring process and applying the inverse function. It works well when the blur kernel is known but is sensitive to noise.
    - **Blind Deconvolution**: An advanced method that estimates both the original image and the blur kernel. It's useful when the exact cause of the blur is unknown.

    - **Richardson-Lucy Deconvolution**: An iterative process for removing motion blur, particularly when the blur is not uniform across the image.
3. **Super-Resolution**
    - **Single Image Super-Resolution**: Techniques like deep learning models (e.g., convolutional neural networks) predict and enhance missing high-frequency details in low-resolution images.
    - **Multi-Image Super-Resolution**: Combines multiple low-resolution images taken from different angles or slightly varying positions to reconstruct a higher-resolution image.
4. **Inpainting (Image Completion)**
    - **Traditional Methods**: Techniques such as **texture synthesis** fill in missing regions based on surrounding pixels.
    - **Deep Learning-Based Inpainting**: Modern deep learning techniques (e.g., Generative Adversarial Networks, GANs) can fill missing portions of an image with contextually relevant information.
5. **Artifact Removal**
    - **DCT (Discrete Cosine Transform)**: Used for artifact reduction in compressed images by filtering out high-frequency noise introduced by compression.
    - **Deep Learning for Artifact Removal**: Convolutional neural networks (CNNs) can be trained to learn how to remove specific artifacts like blockiness or ringing from compressed images.
6. **Geometric Distortion Correction**
    - **Lens Correction**: Corrects barrel or pincushion distortions caused by camera lenses.
    - **Perspective Correction**: Straightens images that have been distorted due to the angle of capture.

Image degradation –

Image degradation is the loss of image quality for a variety of reasons. When there is image deterioration, the quality of the image is greatly diminished and becomes hazy.

Image degradation is the process by which an image's quality is diminished or compromised. This can happen for a number of causes, including noise, blur, or compression, and can negatively affect the image's appearance and readability.

Noise can be introduced into an image in a number of ways, including electronic noise in the camera sensor, interference from outside sources, or quantization mistakes during image processing. Noise is one of the most frequent causes of image degradation. Noise can affect an image's clarity and detail by causing random variations in the intensity or color of the pixels.

Blur can also cause image degradation, as it can make the image appear out of focus or reduce the sharpness and detail of the image. Blur can be caused by a variety of factors, such as the

movement of the camera or the subject, the use of a low-quality lens, or the presence of atmospheric effects such as haze or smoke.

Compression can also lead to image degradation, as it involves reducing the size of the image by removing or approximating certain image data. This can result in lossy compression, where some of the original data is permanently lost, or lossless compression, where the original data can be recovered exactly. While compression can be useful for reducing the storage and transmission requirements of images, it can also result in a loss of quality, depending on the amount of compression applied.

Image restoration methods like denoising, deblurring, or decompression may be required to address image degradation in an effort to improve the image's quality. These methods can be used to enhance the image's visibility and readability, but they also run the risk of adding more artifacts or distortions, which could degrade the quality of the restored image.

Causes of Image Degradation:

1. **Noise**
    - Random variations in pixel values that distort the image, often caused by sensor limitations, poor lighting, or transmission errors.
    - Types of noise:
        - **Gaussian Noise**: Caused by random variations, typically in the form of a normal distribution.
        - **Salt-and-Pepper Noise**: Pixels randomly change to either black or white.
        - **Poisson Noise**: Common in low-light conditions, where pixel intensity follows a Poisson distribution.
2. **Blur**
    - Caused by factors like camera motion, defocus, or environmental movement.
    - **Types of blur**:
        - **Motion Blur**: Caused by the relative movement between the camera and the subject.
        - **Gaussian Blur**: A smooth blurring effect, commonly used in image processing for smoothing or noise reduction.
        - **Out-of-focus Blur**: When the image is not properly focused, creating a soft appearance across the whole image or parts of it.
3. **Compression Artifacts**
    - Compression algorithms (like JPEG) reduce the image size by discarding some information. However, this can introduce artifacts, including:
        - **Blocking Artifacts**: Visible blocky patterns caused by compression algorithms dividing the image into smaller blocks.
        - **Ringing Artifacts**: Wavy or halo-like patterns around sharp edges due to loss of high-frequency information.
4. **Low Resolution**
    - Reduced image quality due to insufficient pixel density, which can make the image appear pixelated or blurry.
5. **Geometric Distortion**
    - Changes to the geometry of the image, such as perspective distortion or lens distortion, which make the image appear warped.

Restoration process

The restoration process in digital image processing is crucial for improving degraded images, and the method chosen depends on the nature of the degradation. With advancements in machine learning and deep learning, restoration techniques have become more robust, providing high-quality results for various applications like medical imaging, satellite imagery, and photography.

There are various approaches for restoring an image, depending on the type of degradation.

a. Inverse Filtering

Inverse filtering assumes that the degradation process is known. The idea is to reverse the degradation process using a filter. However, this method is sensitive to noise, making it less practical for noisy images.

b. Wiener Filter

The Wiener filter is one of the most widely used methods for restoring images. It minimizes the mean square error between the original and restored image. It works well when the degradation process is known, and the noise is random (such as Gaussian noise).

c. Constrained Least Squares (CLS) Filtering

This method is used to restore images by solving an optimization problem. The goal is to find a restored image that minimizes the difference between the observed and degraded images while adhering to certain constraints, such as smoothness or sparsity.

d. Regularized Restoration

In cases where there is a high level of noise or uncertainty about the degradation process, **regularization techniques** can help. These involve adding a penalty term (e.g., smoothness, total variation) to the restoration process to prevent overfitting to noisy data.

e. Blind Image Restoration

In some cases, you don't know the degradation function (like the blur kernel). **Blind restoration** techniques attempt to simultaneously estimate the original image and the degradation function. Methods like **blind deconvolution** are used for this.

f. Deep Learning-Based Methods

Recently, **deep learning** techniques, especially **convolutional neural networks (CNNs)**, have gained popularity for image restoration. These models can learn to restore images by training on large datasets of degraded and original image pairs. Examples include:

- **Denoising** using autoencoders or deep CNNs.
- **Deblurring** using models like U-Net or generative adversarial networks (GANs).
- **Super-resolution** to recover high-resolution images from low-resolution inputs.

4. Post-Restoration Processing

- **Noise reduction**: Additional filtering to remove any residual noise.
- **Contrast adjustment**: Enhancing visibility of details.
- **Edge sharpening**: Restoring sharp edges that may have been softened during restoration.

5. Evaluation of Restoration

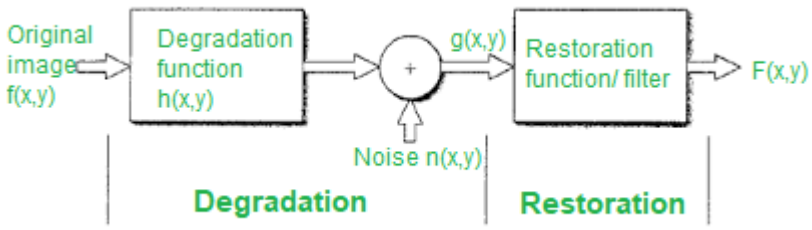
- **Peak Signal-to-Noise Ratio (PSNR)**: Measures the similarity between the restored image and the original image.
- **Structural Similarity Index (SSIM)**: Evaluates perceptual similarity, considering luminance, contrast, and structure.
- **Mean Squared Error (MSE)**: Measures the average squared difference between the original and restored images.

Noise Models –

The principal source of noise in digital images arises during image acquisition and transmission. The performance of imaging sensors is affected by a variety of environmental and mechanical factors of the instrument, resulting in the addition of undesirable noise in the image. Images are also corrupted during the transmission process due to non-ideal channel characteristics.

Generally, a mathematical model of image degradation and its restoration is used for processing. The figure below shows the presence of a degradation function *h(x,y)* and an external noise *n(x,y)* component coming into the original image signal *f(x,y)* thereby producing a final degraded image *g(x,y)*. This part composes the degradation model.

The goal of the restoration function or the restoration filter is to obtain a close replica *F(x,y)* of the original image.



The external noise is probabilistic in nature and there are several noise models used frequently in the field of digital image processing. We have several probability density functions of the noise.

Noise Models

1. Gaussian Noise

   Gaussian noise is a type of statistical noise that has a probability density function (PDF) equal to that of the normal distribution (Gaussian distribution). It is often the result of random variations in the sensor or transmission noise.

- **Characteristics**: It adds a random variation to the pixel values, and the variance determines the intensity of the noise.
- **Impact**: This type of noise is visible as graininess, especially when the noise variance is large.

2. Salt and Pepper Noise

   This type of noise is characterized by random occurrences of black and white pixels (i.e., salt and pepper). It typically occurs when there are errors during image transmission or sensor failures.

- **Characteristics**: The image appears with black (pepper) and white (salt) spots scattered randomly.
- **Impact**: The image appears noisy with many white or black spots scattered randomly.

3. Poisson Noise

   Poisson noise arises when the signal is subject to photon counting, such as in low-light conditions or in medical imaging (e.g., X-ray or MRI scans). It follows a Poisson distribution.

- **Characteristics**: The noise variance is proportional to the signal intensity, meaning that higher signal levels result in greater noise.
- **Impact**: This noise is more noticeable in areas with low pixel intensity and becomes less noticeable as the signal intensity increases.

4. Speckle Noise

Speckle noise is multiplicative noise that manifests as grainy textures in images. It is typically caused by interference in the imaging process, such as sensor malfunction or environmental factors.

- **Characteristics**: The noise manifests as random variations in intensity at each pixel and is proportional to the intensity of the original image.
- **Impact**: This noise leads to granular or salt-and-pepper-like patterns that affect the texture of an image, making it appear blurry or "speckled."

5. Uniform Noise

   Uniform noise is characterized by noise values that are uniformly distributed between a minimum and maximum value, often used to model noise introduced by digital sensors.

- **Characteristics**: The pixel values in the noisy image are affected by random variations that are equally likely across a specific range.
- **Impact**: This noise results in a uniform distribution of noise across all pixel values, which appears as a general "haze" over the image.

6. Rayleigh Noise

Rayleigh noise is a type of noise that is often used to model signals in radar systems or other applications where the signal is a combination of multiple small Gaussian noise components.

- **Characteristics**: The noise follows a Rayleigh distribution, which is a type of continuous probability distribution for non-negative values.
- **Impact**: The image appears with varying intensity levels, where the noise is often higher in areas with low signal intensity.

7. Multiplicative Noise

Multiplicative noise is a type of noise that multiplies the pixel intensity by a random factor. It is common in imaging systems where the signal is multiplied by a random variable (e.g., in remote sensing).

- **Characteristics**: The noise is proportional to the signal, meaning it becomes more significant as the signal intensity increases.
- **Impact**: This noise causes distortion that scales with the intensity of the original image, making it appear more pronounced in brighter areas.

Noise Removal Techniques:

1. **Gaussian Filtering**: Effective for Gaussian noise.
2. **Median Filtering**: Often used to remove salt-and-pepper noise.
3. **Wiener Filter**: A statistical method for removing Gaussian noise.
4. **Non-Local Means Denoising**: Used for various types of noise by comparing pixel similarities.
5. **Wavelet Transform**: Used for removing noise while preserving important image details.

Noise is always presents in digital images during image acquisition, coding, transmission, and processing steps. It is very difficult to remove noise from the digital images without the prior knowledge of filtering techniques. In this article, a brief overview of various noise filtering techniques. These filters can be selected by analysis of the noise behaviour. In this way, a complete and quantitative analysis of noise and their best suited filters will be presented over here.

Filtering image data is a standard process used in almost every image processing system. Filters are used for this purpose. They remove noise from images by preserving the details of the same. The choice of filter depends on the filter behaviour and type of data.

Noise Filtering Techniques:

We all know that, noise is abrupt change in pixel values in an image. So when it comes to filtering of images, the first intuition that comes is to replace the value of each pixel with average of pixel around it. This process smooths the image. For this we consider two assumptions.

Assumption:

1. The true value of pixels are similar to true value of pixels nearby
2. The noise is added to each pixel independently.

Let’s first consider 1-dimensional function before going into 2-dimensional image.

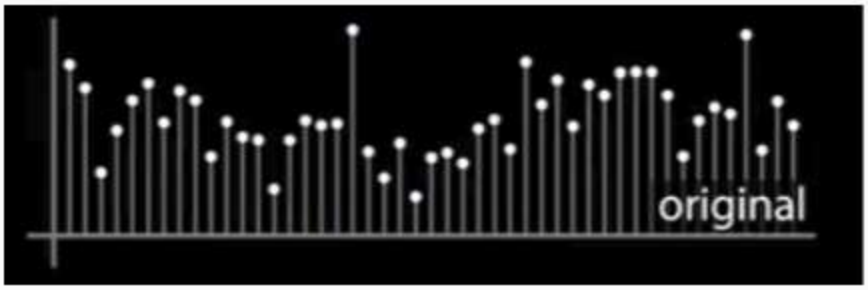


Fig.1 Original image signal in 1 Dimension with noise
Source: Udacity

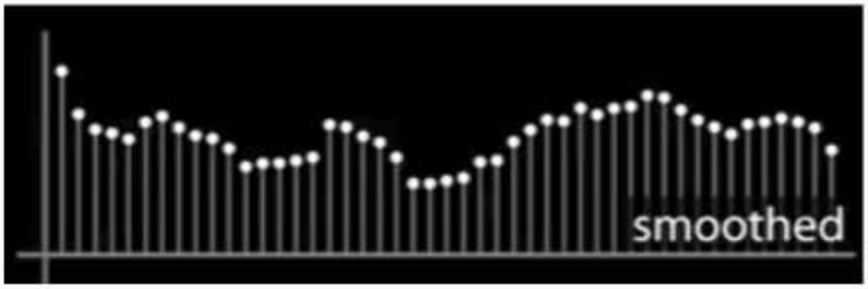


Fig.2 Image signal in 1 Dimension after averaging
Source: Udacity

In the above image of original function(fig-1), if we will consider each circle as pixel values, then the smoothed function(fig-2) is the result of averaging the side by pixel values of each pixel.

1. Filtering with weighted moving average uniform weight:

Instead of just thinking about averaging the local pixel, which is resulting in some loss of data, we consider a set of local pixel and assign them as uniform weights. Here we assume that noise is added to each pixel independently. According to this noise amount, we assign weights to different pixels.

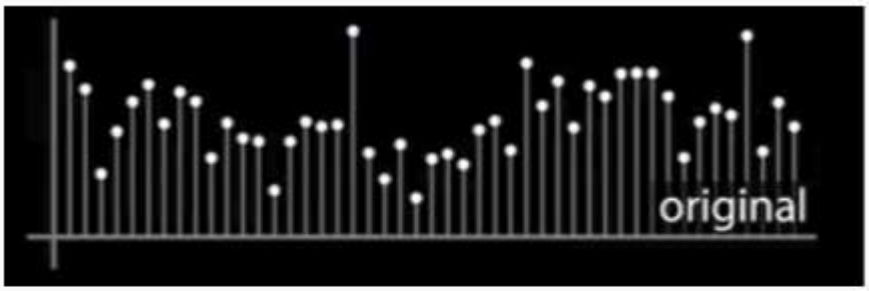


Fig.3 Original image signal in 1 Dimension with noise
Source: Udacity

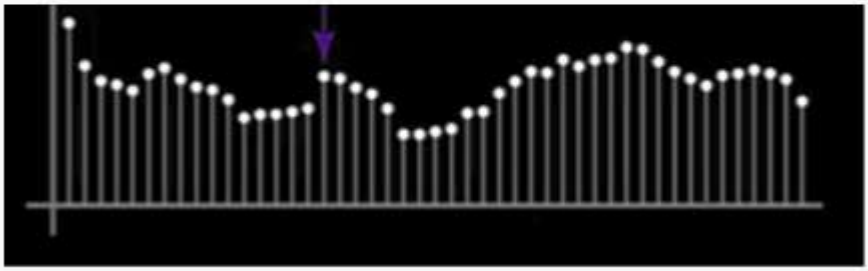


Fig.4 Image signal in 1 Dimension after uniform weighted moving averaging
Source: Udacity

2. Filtering with weighted moving average non-uniform weight

Previously we took the assumption that the true value of pixels are similar to true value of pixels nearby. But it is not always true. So for higher accuracy we assign the nearby pixels with greater weight then the pixels that are far away. This smooths the image and preserves the image information with less amount of data loss.

3. Weighted moving average in 2-dimensional image

Thinking of image as a 2-dimensional matrix, we slide a small window(the red square in fig. 5) over the whole image to replace each pixel with the average of nearby pixels. This small window is otherwise known as **mask or kernel.**

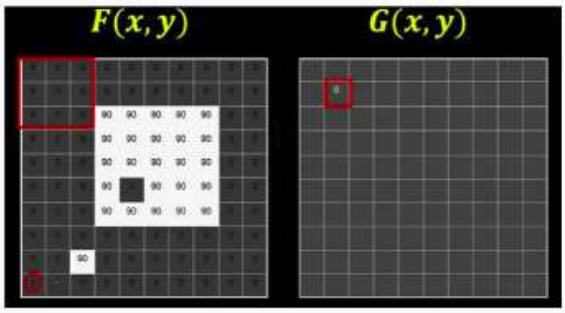


Fig.5 First step for calculating weighted moving averaging in a 2D image matrix
Source: Udacity

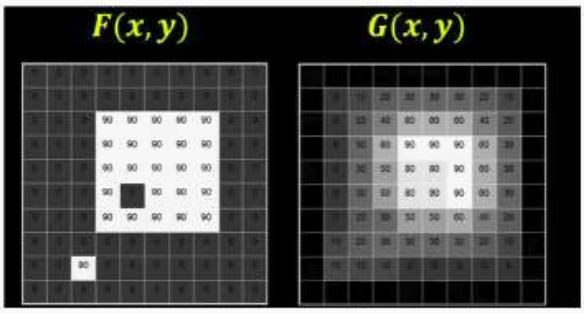


Fig.6 Last step for calculating weighted moving averaging in a 2D image matrix
Source: Udacity

Types of Image noise filters:

There are different types of image noise filters. They can typically be divided into 2 types.

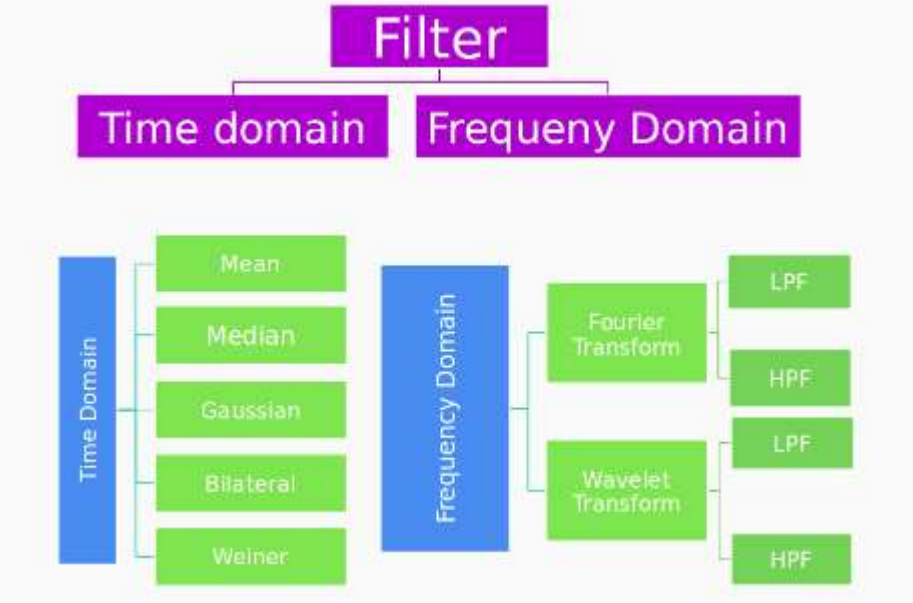


Fig.9 Classification of Filters

Though there are many types of filters, for this article we will consider 4 filters which are mostly used in image processing.

1. Gaussian Filter:

In **image processing**, a **Gaussian** blur (also known as **Gaussian** smoothing) is the result of blurring an **image** by a **Gaussian** function (named after mathematician and scientist Carl Friedrich **Gauss**). It is a widely used effect in graphics software, typically to reduce **image** noise and reduce detail.

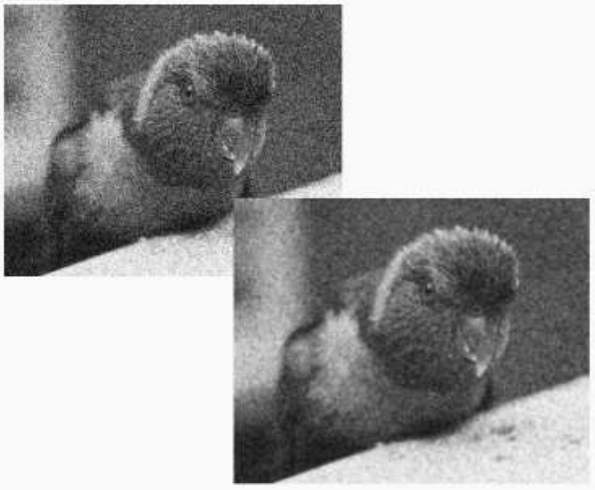
1.1 Implementation of Gaussian Filter with OpenCV and Python:

```
import cv2
import numpy as np
from matplotlib import pyplot as plt

img = cv2.imread('gaussnoise.jpg')

blur = cv2.GaussianBlur(img,(5,5),0)

plt.subplot(121),plt.imshow(img),plt.title('Original')
plt.xticks([]), plt.yticks([])
plt.subplot(122),plt.imshow(blur),plt.title('Averaging')
plt.xticks([]), plt.yticks([])
cv2.imwrite('gaussfiltered.jpg',blur)
plt.show()
```



(Filtering Gaussian Noise)

2. Mean Filter:

Mean filter is a simple sliding window that replace the center value with the average of all pixel values in the window. The window or kernel is usually a square but it can be of any shape.

Fig.10 Calculating Mean by sliding window

2.1 Implementation of Mean Filter with OpenCV ans Python:

```python
import cv2
import numpy as np
from matplotlib import pyplot as plt

img = cv2.imread('poisson.jpg')

# kernel = np.ones((5,5),np.float32)/25
# dst = cv2.filter2D(img,-1,kernel)

#or
blur = cv2.blur(img,(5,5))

plt.subplot(121),plt.imshow(img),plt.title('Original')
plt.xticks([]), plt.yticks([])
plt.subplot(122),plt.imshow(blur),plt.title('Averaging')
plt.xticks([]), plt.yticks([])
cv2.imwrite('poissonfiltered.jpg',blur)
plt.show()

```

(Filtering Poisson Noise)

3. Median Filter:

Mean filter is a simple sliding window that replace the center value with the Median of all pixel values in the window. The window or kernel is usually a square but it can be of any shape.

Fig.11 Calculating Median by sliding window

3.1 Implementation of Median Filter with OpenCV and Python:

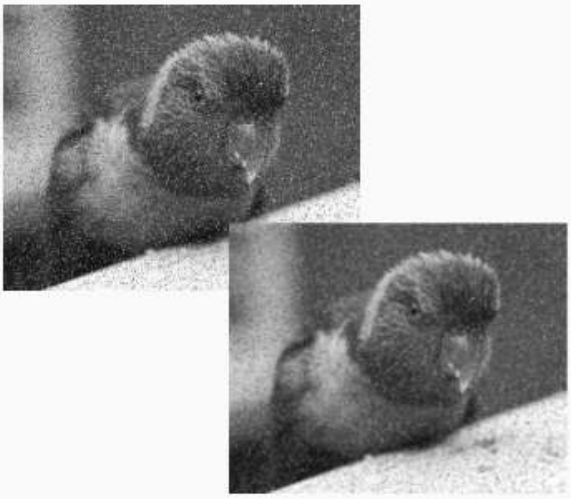
```
import cv2
import numpy as np
from matplotlib import pyplot as plt

img = cv2.imread('poisson.jpg')

# kernel = np.ones((5,5),np.float32)/25
# dst = cv2.filter2D(img,-1,kernel)

#or
blur = cv2.blur(img,(5,5))

plt.subplot(121),plt.imshow(img),plt.title('Original')
plt.xticks([]), plt.yticks([])
plt.subplot(122),plt.imshow(blur),plt.title('Averaging')
plt.xticks([]), plt.yticks([])
cv2.imwrite('poissonfiltered.jpg',blur)
plt.show()
```



(Filtering Salt and Pepper noise)

4. Bilateral Filter

Bilateral filter uses Gaussian Filter but it has one more multiplicative component which is a function of pixel intensity difference. It ensures that only pixel intensity similar to that of the central pixel is included in computing the blurred intensity value. This filter preserves edges.

```
import cv2
import numpy as np
from matplotlib import pyplot as plt

img = cv2.imread('gaussnoise.jpg')

blur = cv2.GaussianBlur(img,(5,5),0)

bilateral = cv2.bilateralFilter(img,9,75,75)

plt.subplot(121),plt.imshow(img),plt.title('Original')
plt.xticks([]), plt.yticks([])
plt.subplot(122),plt.imshow(blur),plt.title('Averaging')
plt.xticks([]), plt.yticks([])
cv2.imwrite('gaussfiltered.jpg',blur)

plt.show()
plt.subplot(122),plt.imshow(bilateral),plt.title('Averaging')
plt.xticks([]), plt.yticks([])
cv2.imwrite('bilaterlfiltered.jpg',bilateral)
plt.show()
```

Fig.12 Result of filtering gaussian noise with Bilateral filter

Fig.13 Result of filtering gaussian noise with Gaussian filter

Degradation function-

Image degradation, in the most straightforward sense, refers to the loss of image quality between what the scene/image originally was and what is captured, processed, stored, or reproduced. It's a constant problem in digital imaging systems, encompassing everything from digital cameras and scanners to display screens and printers.

This degradation can transpire due to a variety of factors, including distortion during image capture, noise interference, lossy compression, transmission errors, or color inaccuracies, among others. It's important to note that these degradations may affect not only the visual perception of the image but also the effectiveness of subsequent operations like image recognition or classification that hinge on the quality of input images.

Image degradation affects an image in various noticeable ways, significantly impacting its quality and the clarity of the information it conveys. Here's a quick roundup of some key areas influenced by image degradation:

- **Loss of detail.** Fine details of an image might become blurred or lost entirely, reducing the texture clarity and making crisp edges indistinguishable.
- **Visual "noise".** Random variation or 'speckling' in brightness or color information in an image adds unneeded noise.
- **Color shifts or distortion.** The colors in an image may appear distorted or poorly balanced, leading to a shift in color representation.
- **Decreased resolution.** There may be a perceivable loss in the sharpness and resolution of the image, making it appear pixelated or 'blocky'.
- **Compromised applicability.** Image degradation can significantly hamper the effectiveness of subsequent operations like image recognition, classification, or any other computer vision tasks.

Preventing image degradation involves several key practices to maintain the integrity and quality of images over time. Image degradation can occur due to compression, format conversion, and improper handling. To minimize this risk, consider the following strategies:

- **Use high-quality originals.** Always start with the highest quality original image. Higher resolution and less compressed formats are a better foundation for further processing or editing.
- **Avoid repeated editing and saving.** Each time an image is edited and saved, especially in lossy formats like JPEG, it loses some quality. Minimize the number of edits and save in lossless formats (e.g., PNG, TIFF) when possible.
- **Proper storage and backup.** Store images in a stable, digital environment. Use reliable storage devices and maintain regular backups to protect against data loss.
- **Control compression levels.** When compression is necessary, control the compression level to avoid significant quality loss. Understand the trade-off between file size and image quality.

Inverse filtering -

Inverse filtering is a deterministic and direct method for image restoration.

The images involved must be lexicographically ordered. That means that an image is converted to a column vector by pasting the rows one by one after converting them to columns. An image of size 256 × 256 is converted to a column vector of size 65536 ×1.

The degradation model is written in a matrix form, where the images are vectors and the degradation process is a huge but sparse matrix.

Homomorphic filtering-

Homomorphic filtering is sometimes used for image enhancement. It simultaneously normalizes the brightness across an image and increases contrast. Here homomorphic filtering is used to remove multiplicative noise. Illumination and reflectance are not separable, but their approximate locations in the frequency domain may be located. Since illumination and reflectance combine multiplicatively, the components are made additive by taking the logarithm of the image intensity, so that these multiplicative components of the image can be separated linearly in the frequency domain. Illumination variations can be thought of as a multiplicative noise, and can be reduced by filtering in the log domain.

To make the illumination of an image more even, the high-frequency components are increased and low-frequency components are decreased, because the high-frequency components are assumed to represent mostly the reflectance in the scene (the amount of light reflected off the object in the scene), whereas the low-frequency components are assumed to represent mostly the illumination in the scene. That is, high-pass filtering is used to suppress low frequencies and amplify high frequencies, in the log-intensity domain.

Operation

Homomorphic filtering can be used for improving the appearance of a grayscale image by simultaneous intensity range compression (illumination) and contrast enhancement (reflection).

Where,

m = image,

i = illumination,

r = reflectance

We have to transform the equation into frequency domain in order to apply high pass filter. However, it's very difficult to do calculation after applying Fourier transformation to this equation because it's not a product equation anymore. Therefore, we use 'log' to help solve this problem.

Then, applying Fourier transformation

Next, applying high-pass filter to the image. To make the illumination of an image more even, the high-frequency components are increased and low-frequency components are decrease.

Where

H = any high-pass filter

N = filtered image in frequency domain

Afterward, returning frequency domain back to the spatial domain by using inverse Fourier transform.

Finally, using the exponential function to eliminate the log we used at the beginning to get the enhanced image

The following figures show the results of applying the homomorphic filter, high-pass filter, and the both homomorphic and high-pass filter. All figures were produced using Matlab.

Figure 1: Original image: trees.tif

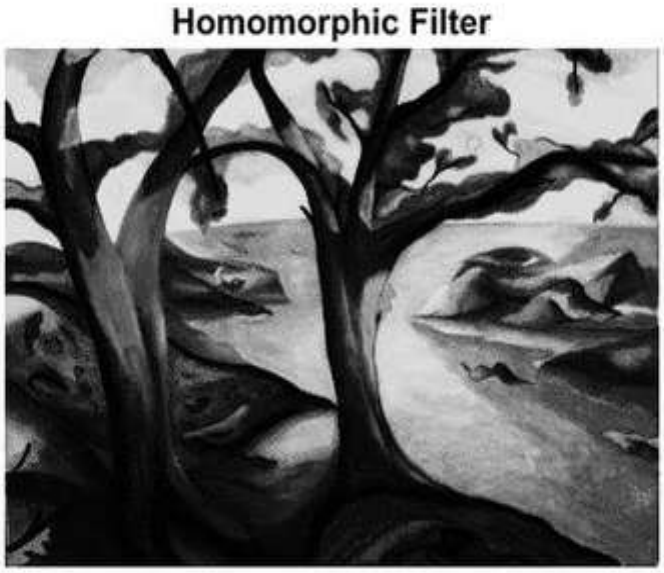


Figure 2: Applying homomorphic filter to original image