# Analysis of Image denoising Spatial Filtering techniques

## Abstract

With the outbreak in the number of graphical data conveyed in the form of digital images found in myriad image processing and computer vision issues, the need for more precise and visually satisfying images is advancing and is becoming a most important technique of transmission. Therefore, work is needed to reduce inevitable noise without losing image attributes (edges, corners, and other sharp structures). To overcome this issue, various noise removal methods have been developed for eliminating noise and preserving edge details in the image. Each method has its own benefits and drawbacks. In this paper, we analyze three different spatial filtering approaches which are Gaussian smoothing filter, Isotropic linear diffusion smoothing, and Isotropic non-linear diffusion smoothing in the task of image denoising, and compare their performance. Our analysis and review of these three image denoising methods demonstrates that inhomogeneous linear diffusion filtering can better preserve the edges in which blurring of the edges is reduced, however, it also indicates some artifacts by the passage of the diffusion time, which mirror the differential format of the initial image.

Keywords: image denoising, Gaussian smoothing, partial differential equations (PDE), Isotropic linear diffusion filtering, Isotropic non-linear diffusion filtering

## Introduction

In an increasingly digital world, digital Images play an important role in the day to day applications. Owing to the influence of environment, transmission channel, and other factors, images are inevitably contaminated by noise during acquisition, compression, and transmission, leading to distortion and loss of image information. This unpleasant signal degrades the critical information in potential following image processing tasks, such as video processing, image analysis, and tracking, which are adversely influenced. This gives rise to the image denoising models to retrieve significant information from noisy images in the process of noise reduction to receive high-quality images which is an important problem nowadays.

Image denoising is to remove noise from a noisy image, to restore the genuine image. However, since noise, edge, and texture are high-frequency parts, it is problematic to differentiate them in the process of denoising and the denoised images could inevitably lose some details. The main reason for this is that from a mathematical perspective, image denoising is an inverse problem and its solution is not unique. In recent decades, great accomplishments have been made in the area of image denoising, and they are reviewed in the following sections. The remainder of this paper is organized as follows. In Section "Image denoising problem statement", we give the formulation of the image denoising problem. Sections "Literature survey" recapitulate the denoising techniques presented up to now. Section "Materials and Methods" explains our investigations. Finally, conclusions are presented in Section "Results and Conclusions".

## Image denoising problem statement

Mathematically, the problem of image denoising can be modeled as follows:

$$y = x + n \qquad (1)$$

where y is the observed noisy image, x is the unknown clean image, and n represents additive white Gaussian noise (AWGN) with standard deviation $\sigma_n$, which can be estimated in practical applications by various methods, such as median absolute deviation, block-based estimation, and principle component analysis (PCA)- based methods. The objective of noise reduction is to reduce the noise in natural images while minimizing the loss of original features and improving the signal-to-noise ratio (SNR). The major challenges for image denoising are as follows:

- To stop the noise effectively in uniform regions
- To remain the flat area flat
- To cover the image borders (no blurring)
- To maintain total contrast
- To avoid further artifacts
- To supply a visually natural appearance

Owing to solving the clean image x from Eq.1 being an ill-posed problem, we cannot get the unique solution from the image model with noise. To obtain a good estimation of image x^, image denoising has been well-studied in the field of image processing over the past several years.

## Literature survey

Generally, image denoising methods can be roughly classified as spatial domain methods, transform domain methods, and wavelet-based thresholding, which are introduced in more detail in the following.

## Spatial Filtering

Spatial filtering techniques seek to remove noise by computing the gray value of each pixel based on the correlation between pixels/image patches in the actual image.

### *Linear Filters*

It is the technique of choice in cases when only additive noise is present. It blurs sharp edges, eliminates lines and other fine attributes of an image. It contains a Mean filter and Wiener filter**.** A mean filter supplies smoothness in an image by decreasing the intensity variations between the adjoining pixels. This filter is essentially an averaging filter. It applies a cover over each pixel in the signal. The main drawback is that edge-preserving criteria are poor in the Mean filter, and they also act poorly in the presence of signal-dependent noise. The Wiener filtering method is a filter that takes a statistical approach to filter out noise that has eroded a signal. For performing filtering operation it is crucial to have knowledge of the spectral properties of the initial signal and the noise.

### *Non Linear Filters*

It is the method of choice in situations when multiplicative and function-based noise is present. By using non-linear filters, such as median filtering, and weighted median filtering, noise can be suppressed without any identification. The intensity value of each pixel is replaced with a weighted average of intensity values from nearby pixels. One problem with the bilateral filter is its efficiency. Spatial filters make use of a low pass filtering on sets of pixels with the statement that the noise settles the higher region of the frequency spectrum. Normally, spatial filters eliminate noise to a reasonable extent but at the cost of blurring images which in turn makes the edges in pictures invisible.

## Transform Domain Filtering

The transform domain filtering can be divided according to the choice of basic functions. They are mainly classified into the non- data-adaptive transform and data-adaptive transform.

### *Non- Data Adaptive Transform*

#### Spatial Frequency Filtering

It refers to the use of low pass filters using fast Fourier Transform. The noise is removed by deciding a cut-off frequency and adjusting a frequency-domain filter when the components of noise are decorrelated from the useful signal. The main disadvantage of Fast Fourier Transform (FFT) is the fact that the edge information is spread across frequencies because of the FFT basis function and it is not being localized in time or space which means that time information is lost and hence low pass filtering results in a smearing of the edges. But the localized nature of Wavelet Transforms both in time and space supplies a particularly useful method for image denoising when the conservation of edges in the scene is of importance.

#### Wavelet Domain Filtering

Working in Wavelet domain is preferred because the Discrete Wavelet Transform (DWT) make the signal energy concentrate in a small number of coefficients, hence, the DWT of the noisy Working in the Wavelet domain is preferred because the Discrete Wavelet Transform (DWT) makes the signal energy concentrate in a small number of coefficients, hence, the DWT of the noisy image consists of a small number of coefficients having a high Signal to Noise Ratio (SNR) while a relatively large number of coefficients is having low SNR. After removing the coefficients with low SNR (i.e., noisy coefficients) the image is reconstructed by using inverse DWT. As a result, noise is subtracted or filtered from the observations. A major advantage of Wavelet methods is that it provides time and frequency localization simultaneously. Moreover, wavelet methods describe such signals much more efficiently than either the original domain or transforms with global basis elements such as the Fourier transform.

### *Data-Adaptive Transforms*

Lately, a new method called Independent Component Analysis (ICA) has achieved widespread attention. The ICA method was successfully implemented in denoising Non-Gaussian data. One outstanding distinction of using ICA is its presumption of signal to be Non-Gaussian which allows to denoise images with Non-Gaussian as well as Gaussian distribution. Drawbacks of ICA-based methods as compared to wavelet-based methods are the computational cost because it uses a sliding window and it requires a sample of noise-free data or at least two image frames of the same scene. In some applications, it might be difficult to acquire noise-free training data.

## Wavelet Based Thresholding

Wavelet thresholding is a signal estimation technique that uses the abilities of Wavelet transform for signal denoising. It removes noise by extinguishing coefficients that are nonessential relative to some threshold that turns out to be uncomplicated and effective, relies heavily on the selection of a thresholding parameter and the choice of this threshold defines, to a great degree the efficiency of denoising. There are several studies on thresholding the Wavelet.

### *Thresholding Method*

There are various Thresholding techniques that are used for purpose of image denoising such as hard and soft Thresholding. Hard Thresholding which is based on the keep and kills rule is more instinctively demanding and also presents artifacts in the retrieved images whereas soft thresholding is based on the shrink and kill the rule, as it shrinks the coefficients above the threshold in absolute value. In practice, soft Thresholding has been used over hard Thresholding because it gives a more visually pleasing image as compared to hard Thresholding and decreases the abrupt sharp shifts that occur in hard Thresholding. In MATLAB, by default, hard Thresholding is used for compression and soft Thresholding for denoising.

## Threshold Selection Rules

In image denoising applications, the selection of threshold value should be such that Peak Signal to Noise Ratio (PSNR) is maximized. Discovering an optimal value for Thresholding is not a straightforward task. A small threshold will give all the noisy coefficients and thus the resulting images may still be noisy whereas a large threshold makes more number of coefficients to zero, which leads to soft image and image processing may cause blur and artifacts, and consequently the resultant images may lose some signal values. Threshold selection is based on a non-adaptive threshold and adaptive threshold.

### Non Adaptive Threshold

Visu Shrink is the non-adaptive universal threshold, which relies only on a number of data points. It is found to produce an extremely smoothed estimate. It offers the best performance in terms of mean square error (MSE) when the number of pixels reaches infinity. Its threshold value is quite large due to its reliance on the number of pixels in an image. The disadvantage is that it cannot remove the Speckle noise. It can only deal with additive noise.

### Adaptive Threshold

There are two kinds of adaptive thresholds i.e. Sure Shrink and Bayes Shrink**.** Sure Shrink emanated from minimizing Stein‟s Unbiased Risk Estimator, an estimate of MSE risk. It is a mixture of a universal threshold and a SURE threshold. It is used for the suppression of noise by Thresholding the empirical wavelet coefficient. The purpose of Sure Shrink is to minimize the mean square error**.** Sure shrink stops the noise by Thresholding the empirical wavelet coefficient. The Bayes Shrink method has been drawing attention lately as an algorithm for setting different thresholds for every sub-band. Here sub-bands are frequency bands that vary from each other in level and direction. The goal of this method is to estimate a threshold value that minimizes the Bayesian risk assuming Generalized Gaussian Distribution (GGD) prior.

## Materials and Methods

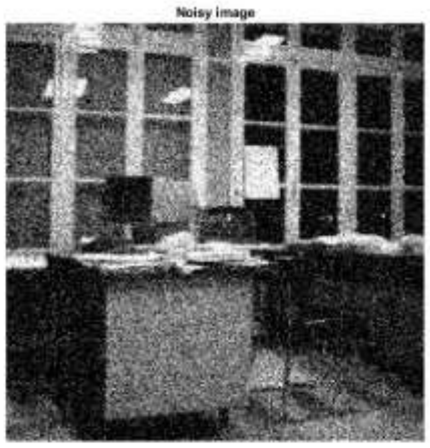
In this project we explore the problem of image denoising by employing and comparing the performance of three different image denoising methods namely: Guassian smoothing filter, Isotropic linear diffusion smoothing, and Isotropic non-linear diffusion smoothing. In each method, we also explore the effect of some parameters in the performance of denoising.

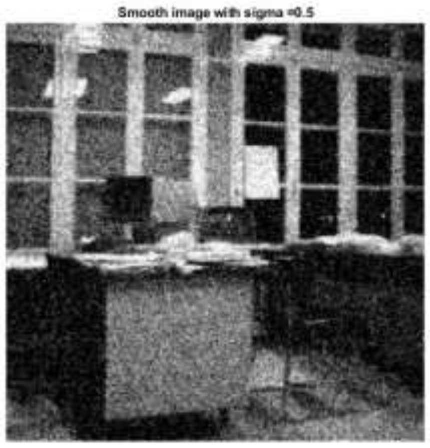
## Guassian smoothing filter

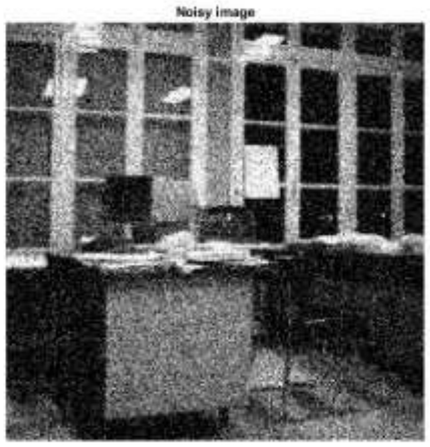
A standard denoising technique is the convolution of the image with a 2D Gaussian distribution.

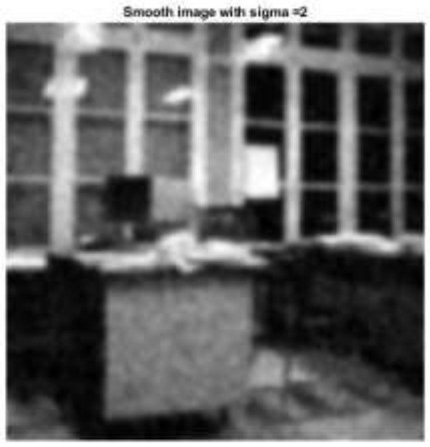
$$G(x,y) = \frac{e^{\frac{-(x^2+y^2)}{2\sigma^2}}}{\sqrt{2\pi\sigma^2}} \qquad (2)$$

First, we analyze the effect of σ in the performance of noise reduction and its influence on the edges (discontinuities) and other details in the image. We realized that when we increase the σ we are looking to the broad scene without paying attention to the details exits. In fact, details are not clear, and when we decrease the value, we will get more details. Put simply, the σ of the Gaussian determines the amount of smoothing. Increasing the σ terms will cast a broader net over the neighboring pixels and decrease the impact of the pixels nearest the pixel of interest, it makes a blurrier image. So to Increase the degree of smoothness we can increase σ as shown in Figure1. Paying attention to the edges, we noticed that in terms of noise reduction, Gaussian kernels have a very interesting property which is that, because the total kernel weight is concentrated in the kernel center and drastically - but smoothly - decreases at the kernel borders, they tend to best preserve edges, or contact zones between objects. This is a fundamental property when we are looking for patterns in a noise-reduced image. It is worth mentioning that, increasing $\sigma$ continues to the reduce/blur the intensity of the noise but also attenuates high-frequency detail (*e.g.* edges) significantly. This shows that smoothing affects edge detection. With more smoothing, fewer edges are detected.









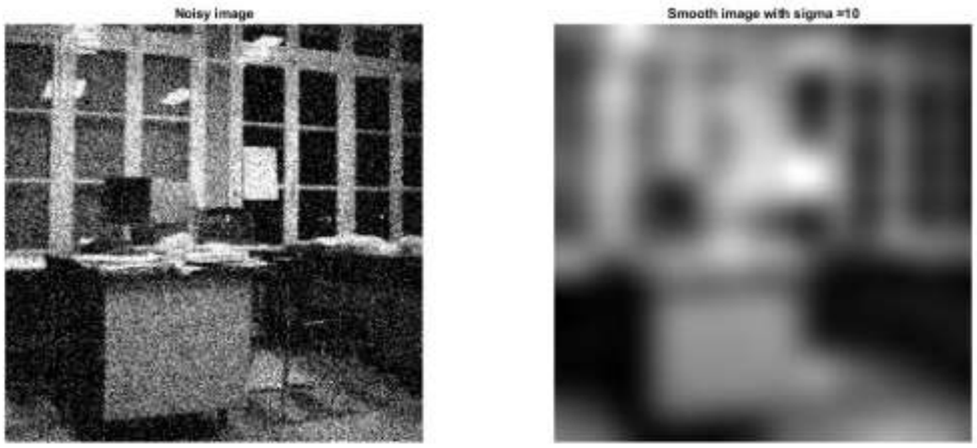
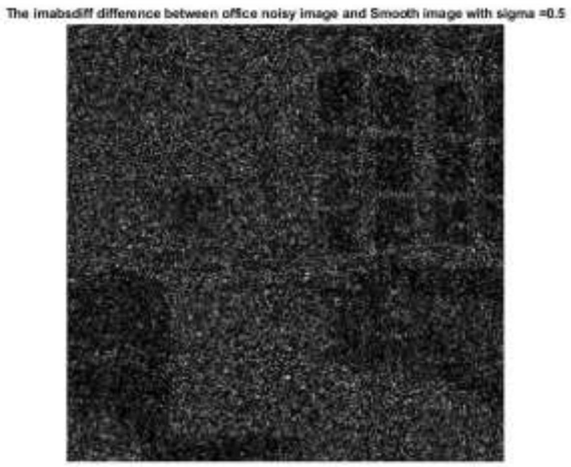
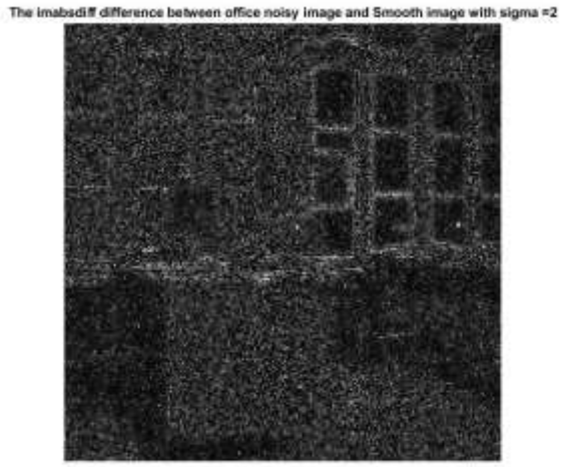


*Figure 1. Comparison between office noisy image and smoothed image by Gaussian kernel in different σ values*

In addition, as shown in Figure 2., by depicting the absolute difference of two images, ‘’Office noisy image” and “smoothed image by Gaussian smoothing function”, in different σ values it can be seen that the difference is more obvious in the edges and discontinuities as we increase the value of σ.


The imabsdiff difference between office noisy image and Smooth image with sigma =0.5


The imabsdiff difference between office noisy image and Smooth image with sigma =2

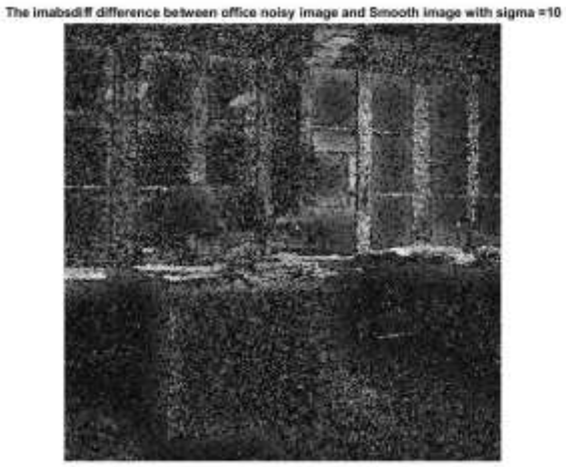

The imabsdiff difference between office noisy image and Smooth image with sigma =10

*Figure 2. Absolute difference of office noisy image, and smoothed image by Gaussian kernel in σ=0.5 and σ=50*

Now, it is obvious why denoising by a Gaussian kernel is referred to as Gaussian smoothing. Gaussian Filter is a low pass filter. Low pass filters as their names imply pass low frequencies - keeping low frequencies. So when we look at the image in the frequency domain, the highest frequencies happen in the edges (places, where there is a high change in intensity and each intensity value, corresponds to a specific visible frequency). The role of σ in the Gaussian filter is to control the variation around its mean value. So as the σ becomes larger the more variance allowed around means, and as the σ becomes smaller the less variance allowed around mean. When we apply a Gaussian filter to an image, we are doing a low pass filtering. But as we know this happens in the discrete domain (image pixels). So we have to quantize our Gaussian filter to make a Gaussian kernel. In the quantization step, as the Gaussian filter(GF) has a small σ it has the steepest pick. So the more weights will be focused in the center and the less around it.

## Isotropic linear diffusion smoothing

we consider a linear isotropic diffusion process on an image domain for the task of denoising which can be described by:

$$\frac{\partial u}{\partial t} = div\ (d\ \nabla u) \qquad (3)$$

$$u(x, y, 0) = I(x, y) \quad (4)$$

where d is a scalar constant diffusivity, I(x; y) is the initial noisy image, u(x; y; t) is the image obtained after a diffusion time t.

By solving the diffusion PDE as described below we calculate the desired equation for updating pixel weights after each diffusion t. Our solution to the diffusion PDE is as follow:

$$\begin{aligned} \partial_t u &= \mathrm{div}(d\nabla u) \ \ (\text{if } d=1) \\ &= \Delta u \\ &= \partial_{xx}u + \partial_{yy}u \qquad (5) \end{aligned}$$

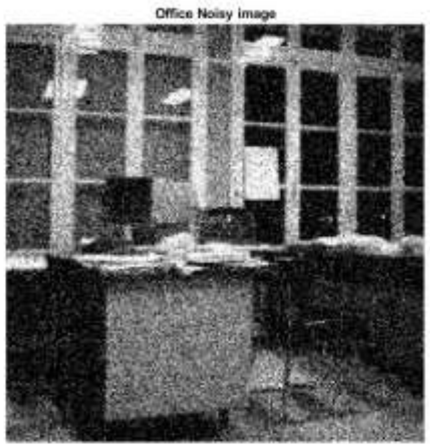
Now using forward difference to estimate the time derivative and central difference for the space derivatives we have:

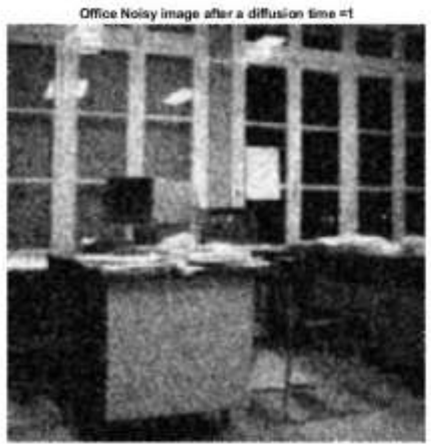
$$\begin{aligned} \partial_t u(x_i,y_j,t_k) &\approx (u^{k+1}_{i,j} - u^k_{i,j})/\ \Delta t \\ \partial_{xx} u(x_i,y_j,t_k) &\approx (u^k_{i+1,j} - 2u^k_{i,j} + u^k_{i-1,j})/\ \Delta x^2 \\ \partial yyu(x_i,y_j,t_k) &\approx (u^k_{i,j+1} - 2u^k_{i,j} + u^k_{i,j-1})/\ \Delta y^2 \qquad (6) \end{aligned}$$

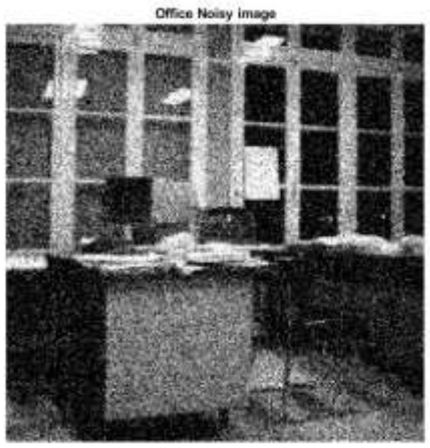
Plugging these equations into the first equation and set $\Delta x=\Delta y$ and $\mu= \Delta t/\Delta x^2$ we have the following equation:

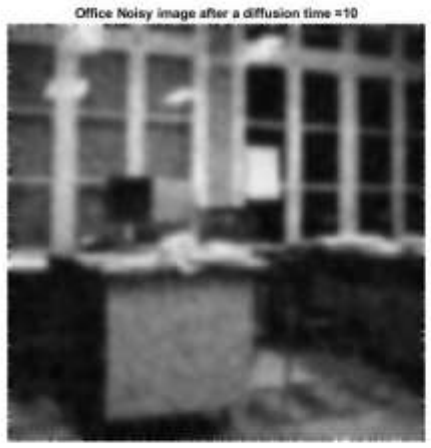
$$u^{k+1}_{i,j} = u^k_{i,j} + \mu\ (u^k_{i+1,j} + u^k_{i,j+1} - 4u^k_{i,j} + u^k_{i-1,j} + u^k_{i,j-1}) \qquad (7)$$

Performing this solution on input “office noisy” image, we realized that by the passage of the diffusion time the image becomes smoother. It supports the intuitive idea that like Gaussian smoothing, linear diffusion filtering “diffuses” (fades out) objects to each other in the image. Smoothing filters tend to blur an image because pixel intensity values that are significantly higher or lower than the surrounding neighborhood would "smear" across the area as shown in Figure3. Because of this blurring, linear filters are seldom used in practice for noise reduction


Office Noisy image


Office Noisy image after a diffusion time =1


Office Noisy image


Office Noisy image after a diffusion time =10

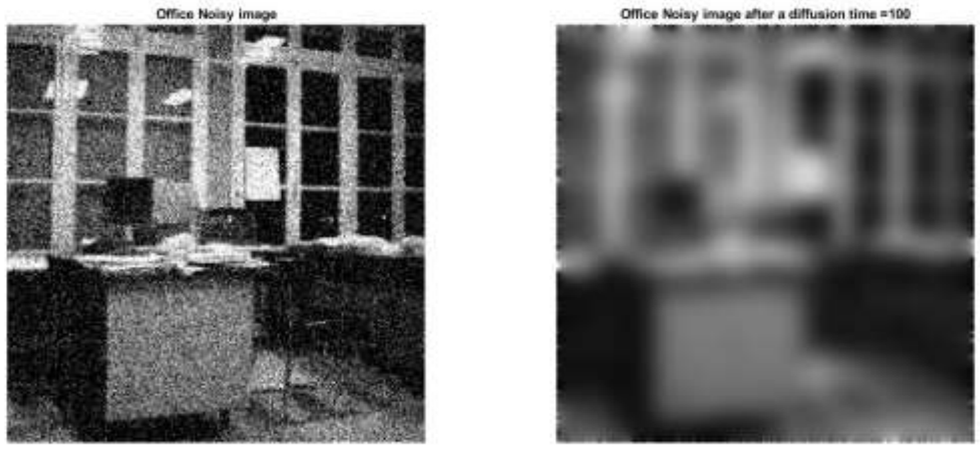


*Figure 3. Comparison between office noisy image and filtered image by linear diffusion filtering at different diffusion time*

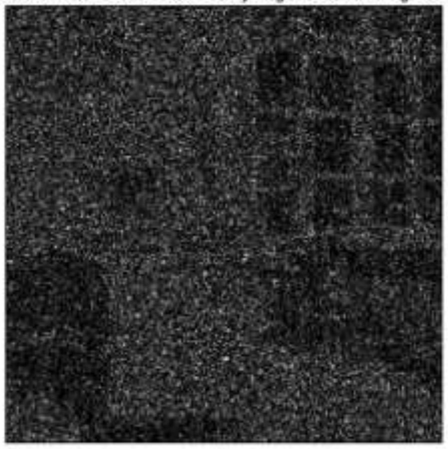
According to table1., the SNR, PSNR and MSE value of the diffused image among all given times (1,5,10,30,100). The PSNR/ SNR by the passage of time increases as the MSE decreases but not monotonically due to the final smoothness of the image as the diffused time increases.

*Table 1. MSE, SNR, and PSNR values in different diffusion time between office image and office noisy image.*

| | T=0 | T=1 | T=5 | T=10 | T=30 | T=100 |
|---|---|---|---|---|---|---|
| MSE | 0.0195 | 0.0067 | 0.0065 | 0.0086 | 0.0147 | 0.0258 |
| PSNR | 17.1065 | 21.7280 | 21.8653 | 20.6313 | 18.3151 | 15.8921 |
| SNR | 9.5035 | 14.1251 | 14.2623 | 13.0283 | 10.7121 | 8.2891 |

However, by exploring the edges (discontinuities)and other details with the increase in diffusion time, we realized that Linear diffusion filtering is equivalent to Gaussian smoothing and has the same problem as Gaussian filters about losing edges when passing to coarser levels in scale space.
In addition, as shown in Figure 4., by depicting the absolute difference of two images, ‘’Office noisy image” and “diffused image by Isotropic linear diffusion smoothing”, in different time values it can be seen that the difference is more obvious in the edges and discontinuities as we increase the value of time.

The imabsdiff difference between office noisy image and Smooth image at time t =1



The imabsdiff difference between office noisy image and Smooth image at time t =10

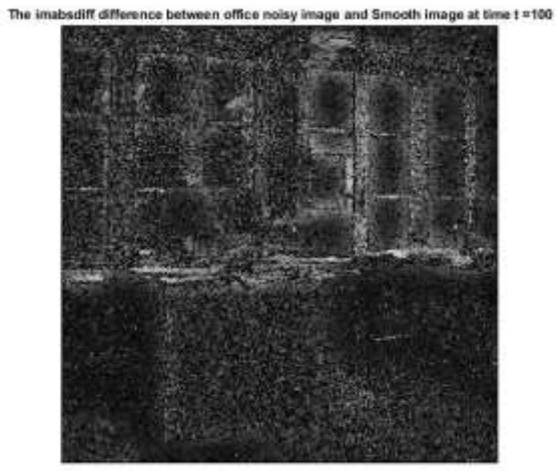
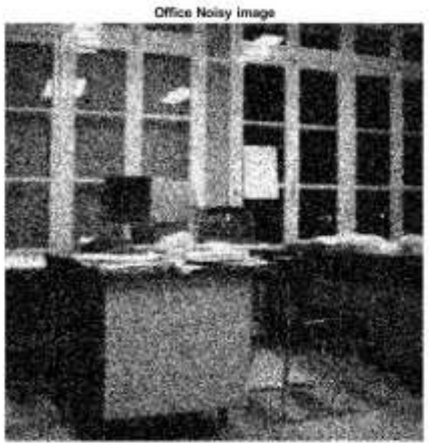

The imabsdiff difference between office noisy image and Smooth image at time t =100

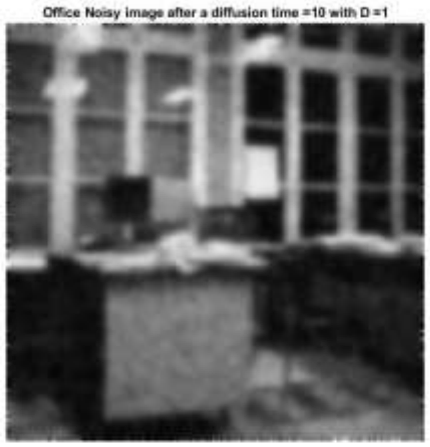
*Figure 4. Absolute difference of office noisy image, and smoothed image by linear diffusion filtering at t=1, and t=100*

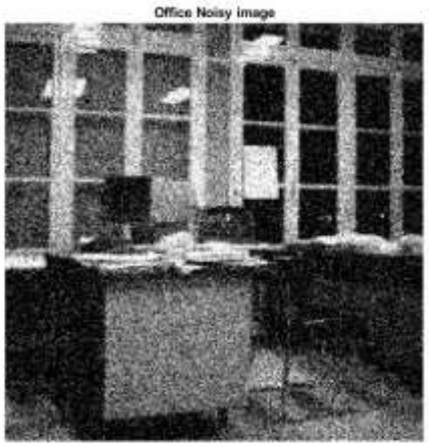
Then, in order to realize the effect of D on denoising, we perform the solution in different D values at a fixed time and realized that the diffusivity D describes the speed of the diffusion process and blurring level as shown in Figure5. As seen in table2., by increasing the value of D, the MSE decreases subsequently. Likewise SNR/PSNR increases by the increasing o the value of D.

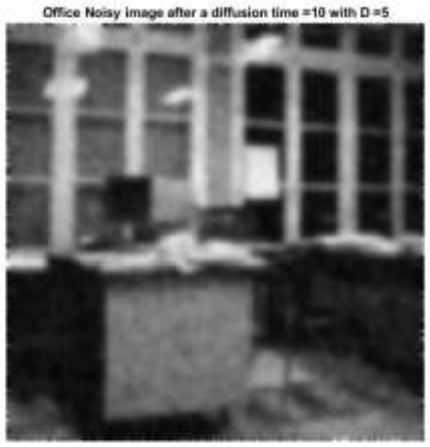
*Table 2. MSE, SNR, and PSNR value in different D values between office image and office noisy image*

| | First | D=1 | D=5 | D=10 |
|---|---|---|---|---|
| MSE | 0.0195 | 0.0086 | 0.0082 | 0.0077 |
| PSNR | 17.1065 | 20.6313 | 20.8824 | 21.1481 |
| SNR | 9.5035 | 13.0283 | 13.2795 | 13.5451 |


Office Noisy image


Office Noisy image after a diffusion time =10 with D =1


Office Noisy image


Office Noisy image after a diffusion time =10 with D =5

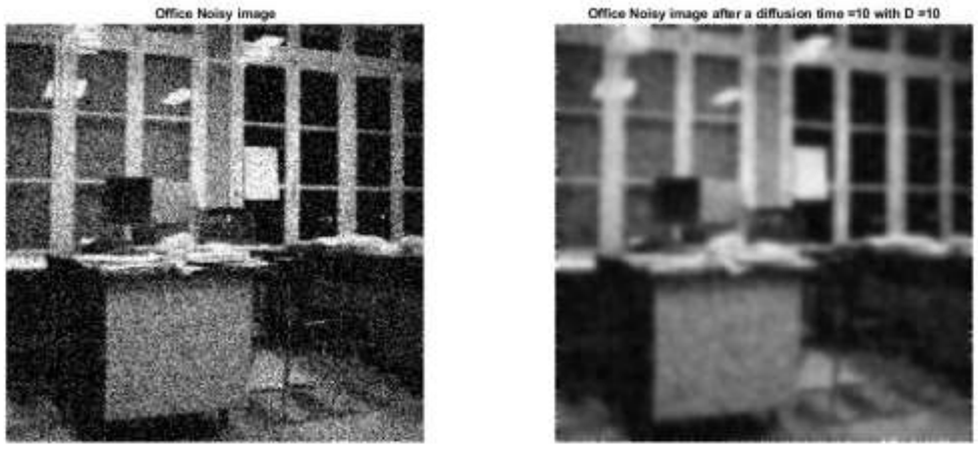
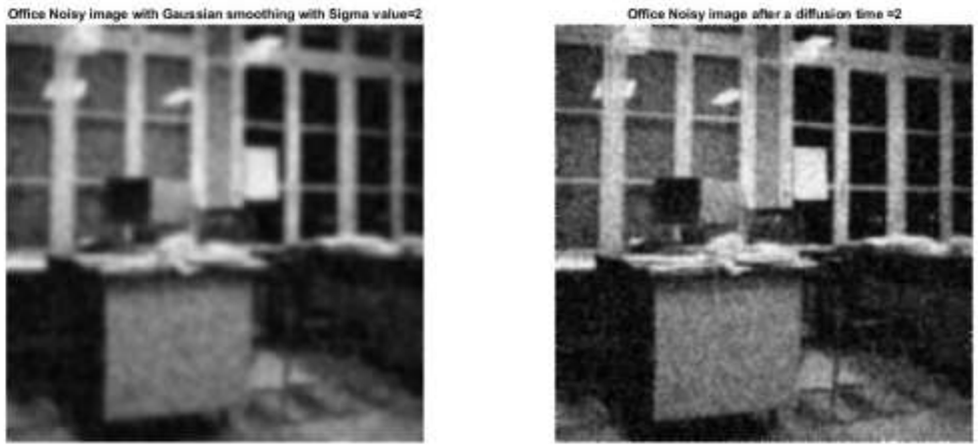
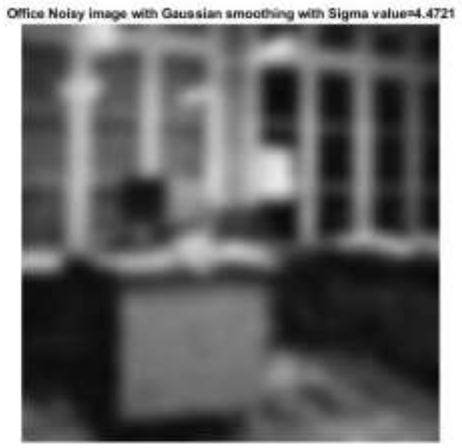


*Figure 5. visualizing the effect of D in linear diffusion filtering on the noisy image at time t=10*

All the above results, sort of, show great similarity between, the isotropic linear diffusion method, and the Gaussian smoothing method. So as proven mathematically, we prove this fact by comparing the image obtained from performing isotropic linear diffusion for a time t with d = 1, and the one obtained after being smoothed by Gaussian smoothing with $\sigma = \sqrt{2t}$.

Figure7. depicted the ssim index map of images performed by linear diffusion filtering and Gaussian smoothing at time t=10. Small values of local SSIM appear as dark pixels in the local SSIM map. Regions with small local SSIM values correspond to areas where the blurred image noticeably differs from the reference image. Large values of local SSIM value appear as bright pixels. Regions with large local SSIM correspond to uniform regions of the reference image, where blurring has less of an impact on the image.


Office Noisy image with Gaussian smoothing with Sigma value=2
Office Noisy image after a diffusion time =2


Office Noisy image with Gaussian smoothing with Sigma value=4.4721

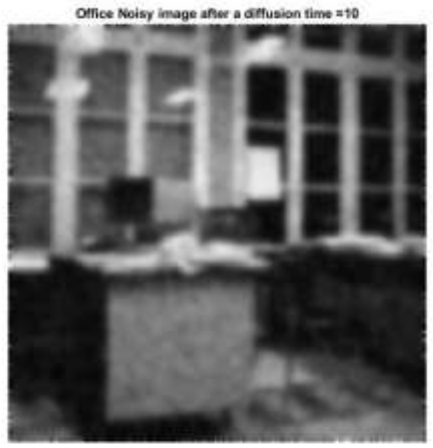
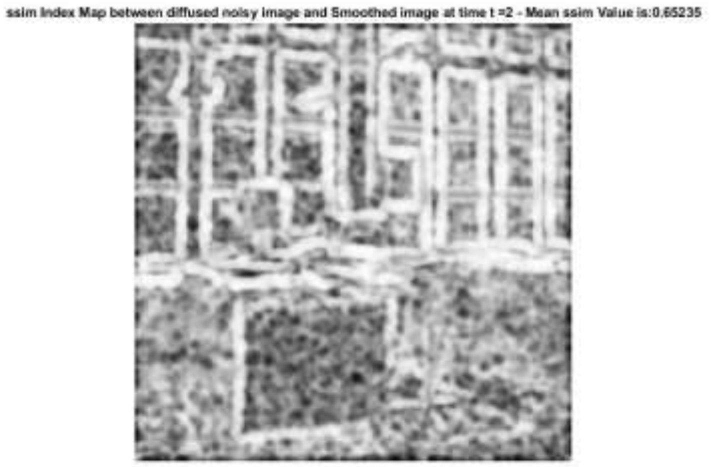


*Figure 6. comparison of the diffused linear filter image and smoothed image by Gaussian smoothing at time t=2 and t=10*



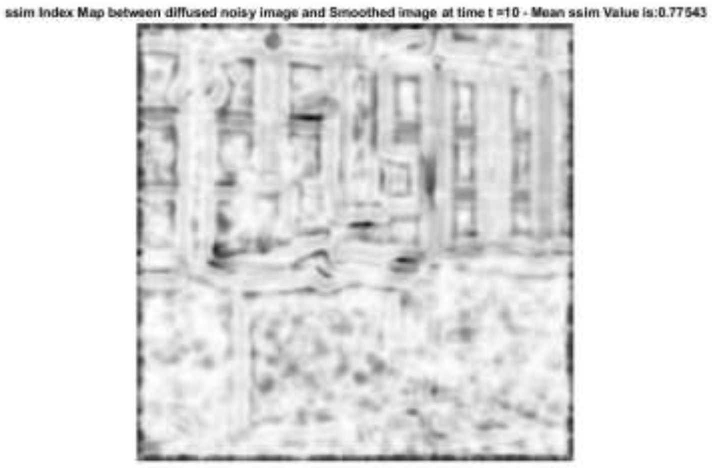


*Figure 7. ssim index map between diffused image and smoothed image by Gaussian smoothing at time t=2 and t=10*

According to Table 3. the small relative MSE of two outputted image and their Structural Similarity index can also prove this fact.

*Table 3. MSE, and Structural similarity index of two ouputted images of linear diffusion, and Gaissian smoothing filter denoising methods*

| | T=2 | T=4 | T=6 | T=10 |
|---|---|---|---|---|
| MSE | 0.0029 | 0.0026 | 0.0026 | 0.0030 |
| Structural Similarity index | 0.6524 | 0.7050 | 0.7421 | 0.7754 |

## Isotropic non-linear diffusion smoothing

We consider a non-linear isotropic diffusion process on an image domain for the task of denoising, which can be described by:

$$\frac{\partial u}{\partial t} = div\ (d\ \nabla u) \qquad (8)$$

$$u(x, y, 0) = I(x, y) \qquad (9)$$

Here, D is not a constant but varies across the image domain. A popular choice is the Perona-Malik diffusivity which is given as:

$$D(x, y) = \frac{1}{1 + \frac{|\nabla u(x,y)|^2}{\lambda^2}} \qquad (10)$$

where λ is the contrast parameter. ∇u(x; y) is the gradient of the image at pixel (x; y).

By solving the diffusion PDE as described below we calculate the desired equation for updating the pixel weights after each diffusion t. Our solution to the diffusion PDE is as follow:

$$\partial_t u = \mathrm{div}(D(|\nabla u|)\ \nabla u)$$

$$D(|\nabla u|) = 1/(1 + |\nabla u|^2/\lambda^2) \qquad (12)$$

To implement any filtering method, we need to approximate the Equation 12.. We can write $D(|\nabla u|)$ as $D(|\nabla u|^2)$ since both functions are also a function of $|\nabla u|^2$. We start by looking at $D(|\nabla u|^2)\nabla u$ and approximating it using second order central difference:

$$D(|\nabla u|^2)\nabla u = D(|\nabla u|^2)\ [\partial u_{i,j}/\partial x \quad \partial u_{i,j}/\partial y\ ]^T$$
$$\approx [D(|\nabla u|^2)\ (u_{i+1/2,j} - u_{i-1/2,j})/\Delta x \quad D(|\nabla u|^2)\ (u_{i,j+1/2} - u_{i,j-1/2})/\Delta y]^T \qquad (13)$$

Now we replace this into the first equation:

$$\partial_t u = \mathrm{div}(D(|\nabla u|)\ \nabla u)$$
$$\approx \mathrm{div}\ ([D(|\nabla u|^2)\ (u_{i+1/2,j} - u_{i-1/2,j})/\Delta x \quad D(|\nabla u|^2)\ (u_{i,j+1/2} - u_{i,j-1/2})/\Delta y]^T)$$
$$= \frac{\partial}{\partial x}\ ((u_{i+1/2,j} - u_{i-1/2,j})/\Delta x) + \frac{\partial}{\partial y}\ ((u_{i,j+1/2} - u_{i,j-1/2})/\Delta y)$$
$$\approx \frac{1}{\Delta x}(D(|\nabla u_{i+1/2,j}|^2)\ \frac{1}{\Delta x}(u_{i+1,j} - u_{i,j}) - D(|\nabla u_{i-1/2,j}|^2)\ \frac{1}{\Delta x}(u_{i,j} - u_{i-1,j}) + \frac{\partial}{\Delta y}(D(|\nabla u_{i,j+1/2}|^2)\ \frac{1}{\Delta y}(u_{i,j+1} - u_{i,j}) - D(|\nabla u_{i,j-1/2}|^2)\ \frac{1}{\Delta y}(u_{i,j} - u_{i,j-1})) \qquad (14)$$

to estimate the values for $D(|\nabla u_{i\pm1/2,j}|^2$and $D(|\nabla u_{i,j\pm1/2}|^2$ we can take the average:

$$D(|\nabla u_{i+1/2,j}|^2 \approx \frac{1}{2}\left( D(|\nabla u_{i+1,j}|^2) + D(|\nabla u_{i,j}|^2)\right)$$
$$D(|\nabla u_{i-1/2,j}|^2 \approx \frac{1}{2}\left( D(|\nabla u_{i-1,j}|^2) + D(|\nabla u_{i,j}|^2)\right)$$
$$D(|\nabla u_{i,j+1/2}|^2 \approx \frac{1}{2}\left( D(|\nabla u_{i,j+1}|^2) + D(|\nabla u_{i,j}|^2)\right)$$
$$D(|\nabla u_{i,j-1/2}|^2 \approx \frac{1}{2}\left( D(|\nabla u_{i,j-1}|^2) + D(|\nabla u_{i,j}|^2)\right) \qquad (15)$$

using the standard second order central difference formula to estimate $\nabla u_{i,j}$, which leads to:

$$\nabla u_{i,j} \approx \left[\frac{1}{2\Delta x}(u_{i+1,j} - u_{i-1,j}) \quad \frac{1}{2\Delta y}(u_{i,j+1} - u_{i,j-1})\right]^T$$

$$(|\nabla u_{i,j}|^2 \approx \left(\frac{1}{2\Delta x}(u_{i+1,j} - u_{i-1,j})\right)^2 + \left(\frac{1}{2\Delta y}(u_{i,j+1} - u_{i,j-1})\right)^2 \qquad (16)$$

using forward difference to estimate the time derivative. Setting $\Delta x=\Delta y$, we obtain the following numerical scheme:

$$u^{k+1}_{i,j} = u^k_{i,j} + \mu \Big(\frac{1}{2}\left(D(|\nabla u^k_{i+1,j}|^2) + D(|\nabla u^k_{i,j}|^2)\right)(u^k_{i+1,j} - u^k_{i,j}) - \frac{1}{2}\left(D(|\nabla u^k_{i-1,j}|^2) + D(|\nabla u^k_{i,j}|^2)\right)(u^k_{i,j} - u^k_{i-1,j}) + \frac{1}{2}\big( D(|\nabla u^k_{i,j+1}|^2) + D(|\nabla u^k_{i,j}|^2)\big)(u^k_{i,j+1} - u^k_{i,j}) - \frac{1}{2}\left( D(|\nabla u^k_{i,j-1}|^2) + D(|\nabla u^k_{i,j}|^2)\right)(u^k_{i,j} - u^k_{i,j-1})\Big) \qquad (17)$$

Another way to solve the Equation(12). Is as follows:

$$\frac{\partial I_t(x,y)}{\partial t} = \operatorname{div}\left[c_t \,.\, \nabla I_t(x,y)\right] \qquad (18)$$

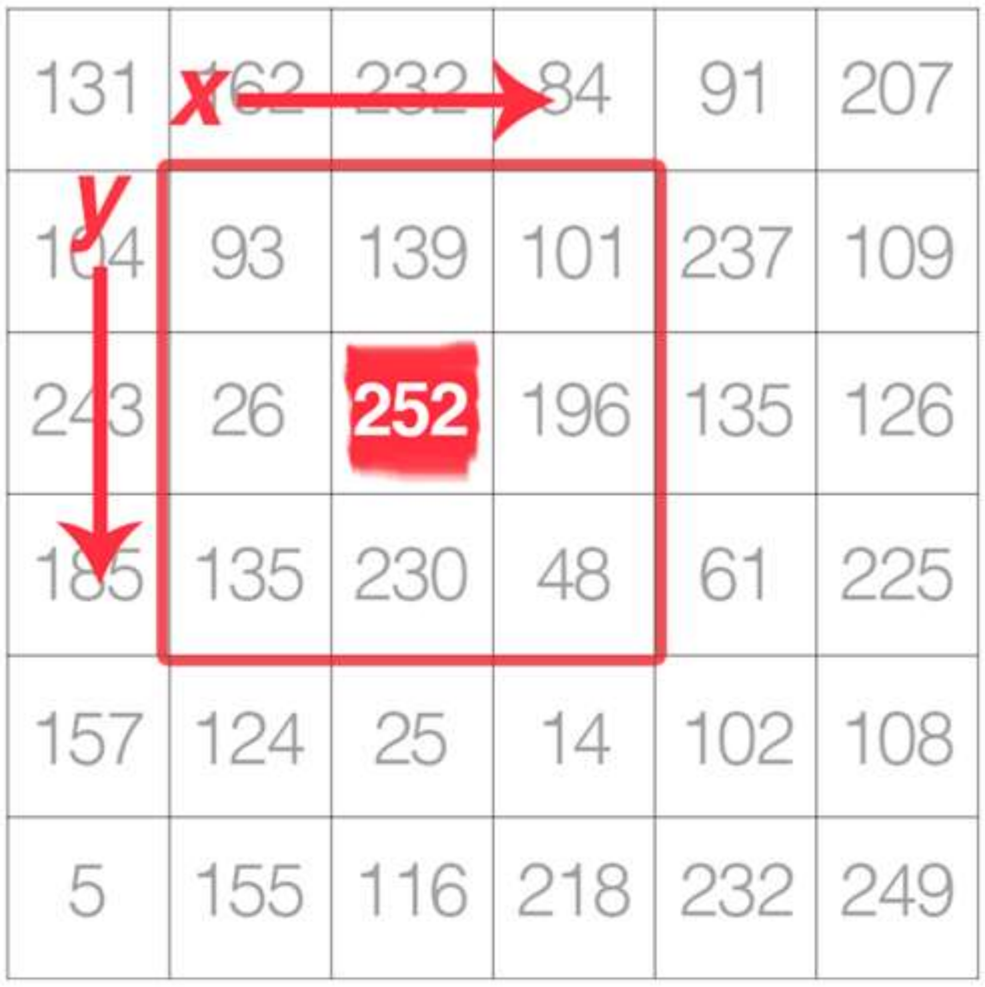
where $I_t(x,y)$ is the image at time t; div represents the divergence operator; $\nabla I_t(x,y)$ is the gradient of the image, and $c_t$ represents the diffusion coefficient.

The continuous anisotropic diffusion in Eq. (18) can be discretely implemented by using four nearest neighbors and the Laplacian operator:

$$I_{t+1}(x,y) = I_t(x,y) + \frac{\sum_{i=1}^{4} c_t^i(x,y).\nabla I_t^i(x,y)}{4} \qquad (19)$$

where $\nabla I_t^i(x,y)$, $i$ = 1, 2, 3 and 4, represent the gradients of four neighbors in the north, south, east and west directions, respectively, i.e.

Formally, an image gradient is defined as a directional change in image intensity. Or put more simply, at each pixel of the input (grayscale) image, a gradient measures the change in pixel intensity in a given direction. By estimating the direction or *orientation* along with the *magnitude* (i.e. how strong the change in direction is), we are able to detect regions of an image that look like edges. Our goal here is to find the change in direction to the central pixel marked in red in both the *x* and *y* direction as shown in Fig8.:



| | | | | | |
|---|---|---|---|---|---|
| 131 | 162 | 232 | 84 | 91 | 207 |
| 104 | 93 | 139 | 101 | 237 | 109 |
| 243 | 26 | 252 | 196 | 135 | 126 |
| 185 | 135 | 230 | 48 | 61 | 225 |
| 157 | 124 | 25 | 14 | 102 | 108 |
| 5 | 155 | 116 | 218 | 232 | 249 |

*Figure 8. change in direction to the central pixel in x and y direction*

When computing image gradients, we need to find the difference in image pixel intensities along both the *x* and *y* direction within the given kernel size. The first step is to simply find and mark the north, south, east and west pixels surrounding the center pixel:

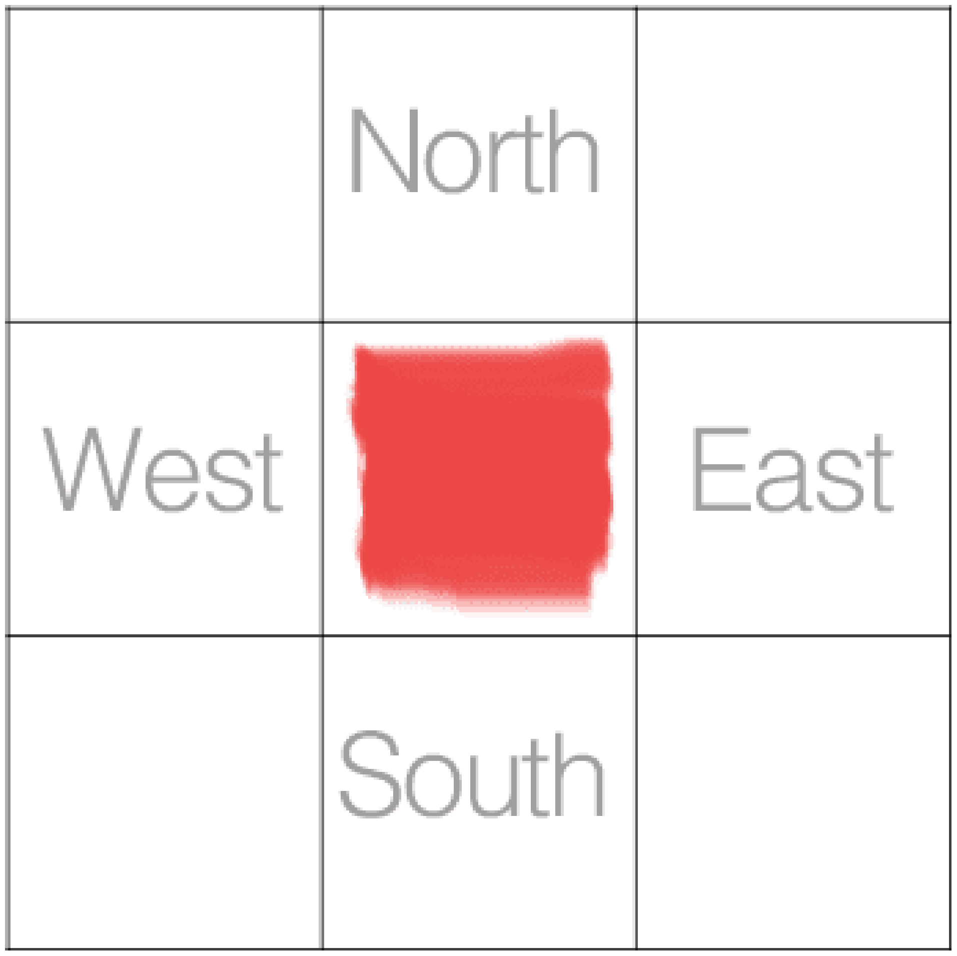


*Figure 9. Labeling the north, south, east, and west pixel surrounding the center red pixel.*

In the image above, we examine the *3×3* neighborhood surrounding the central pixel. Our *x* values run from left to right, and our *y* values from top to bottom. In order to compute any changes in direction we'll need the north, south, east, and west pixels, which are marked on Figure9..

If we denote our input image as *I*, then we define the north, south, east, and west pixels using the following notation:

- **North:** $I(x, y-1)$
- **South:** $I(x, y+1)$
- **East:** $I(x+1, y)$
- **West:** $I(x-1, y)$

Again, these four values are *critical* in computing the changes in image intensity in both the $x$ and $y$ direction.

To demonstrate this, let's compute the vertical change or the ***y***-change by taking the difference between the south and north pixels:

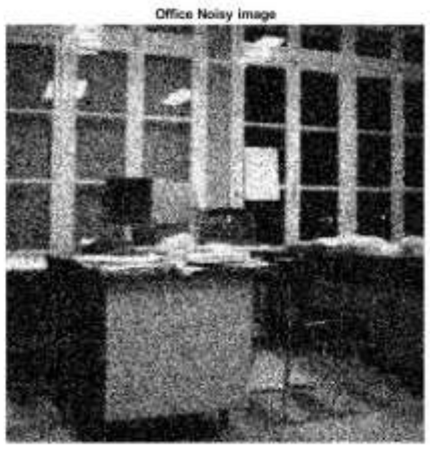
$$G_y = I(x, y + 1) - I(x, y - 1)$$

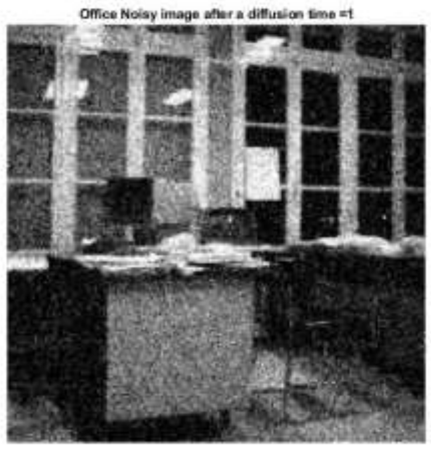
Similarly, we can compute the horizontal change or the $x$-change by taking the difference between the east and west pixels:

$$G_x = I(x + 1, y) - I(x - 1, y)$$

As shown in Figure10. performing this solution on input "office noisy" image, we realized that by the passage of the diffusion time the image becomes smoother. The diffusion model of Perona and Malik only considers the local gradient information of each pixel in the image. That is, it smoothes the inner region with a lower gradient and stops the diffusion process at inter-region edges with the higher gradient in the image. Consequently, in the edge regions where the gradient magnitude is large, the diffusion coefficient is small and as a result, the blurring effect will be below. Similarly, for the smooth regions where the gradient magnitude is small, the diffusion coefficient will be large and as a result, the blurring effect will be high.

The important local features defined in a small neighborhood of each pixel in the image may have low gradients such as blurred edges or fine details of an object, which should also be preserved during the diffusion process but this method inevitably smoothes both noise and fine details with low gradient strength in an ill-structured image. Therefore, the traditional P-M model is very sensitive to the number of diffusion iterations. A careful selection of the number of iterations is required to ensure the success of the diffusion result. According to table 4, that is why the PNR/SNR value increases by the passage of time but decreases at t=100. So as the MSE values.





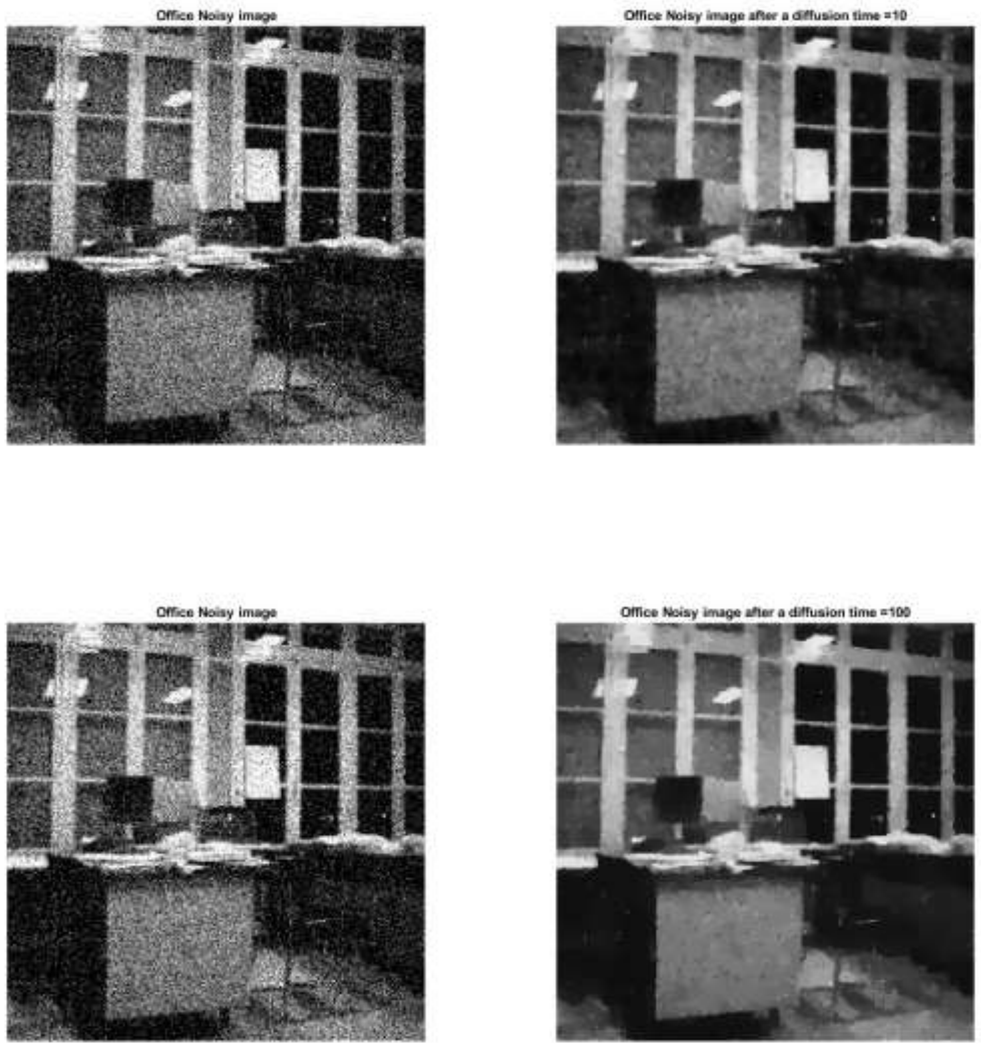
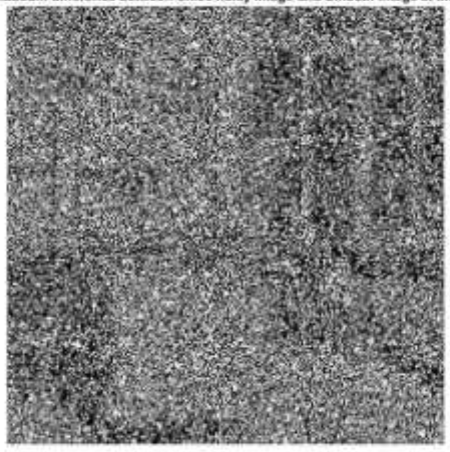

Office Noisy image
Office Noisy image after a diffusion time =10
Office Noisy image
Office Noisy image after a diffusion time =100

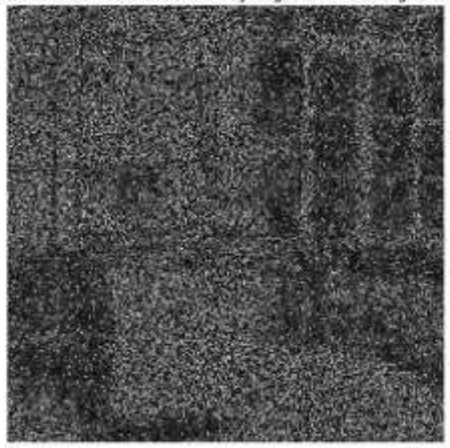
*Figure 10. Comparison between office noisy image and filtered image by non-linear diffusion filtering at different diffusion time*

*Table 4 MSE, SNR, and PSNR values in different diffusion time between office image and office noisy image*

| | T=0 | T=1 | T=5 | T=10 | T=30 | T=100 |
|---|---|---|---|---|---|---|
| MSE | 0.0195 | 0.0089 | 0.0046 | 0.0041 | 0.0039 | 0.0040 |
| PSNR | 17.1065 | 20.4823 | 23.4041 | 23.9181 | 24.0754 | 24.0258 |
| SNR | 9.5035 | 12.8794 | 15.8011 | 16.3152 | 16.4725 | 16.4229 |

By exploring the edges (discontinuities)and other details with the increase in diffusion time, we realized that since the diffusion model of Perona and Malik only considers the local gradient information of each pixel in the image, intra-regions in an image with a lower gradient become smooth and stops the diffusion process at inter-region edges with a higher gradient in the image so significant parts of the image content, typically edges, lines or other details that are important for the interpretation of the image are preserved despite noise removal. A high gradient magnitude is generally a good indication of edges, whereas a low gradient magnitude may not always point to non-edge regions or noise. The important local features defined in a small neighborhood of each pixel in the image may have low gradients such as blurred edges or fine details of an object, which should also be preserved during the diffusion process but this method inevitably smoothes both noise and fine details with low gradient strength in an ill-structured image. In addition, as shown in Figure11., by depicting the absolute difference of two images, ''Office noisy image" and "diffused image by diffusion model of Perona and Malik", at different diffusion time, it can be seen that the difference is more obvious in the edges and discontinuities as the time passes.


The imabsdiff difference between office noisy image and Smooth image at time t =1


The imabsdiff difference between office noisy image and Smooth image at time t =10

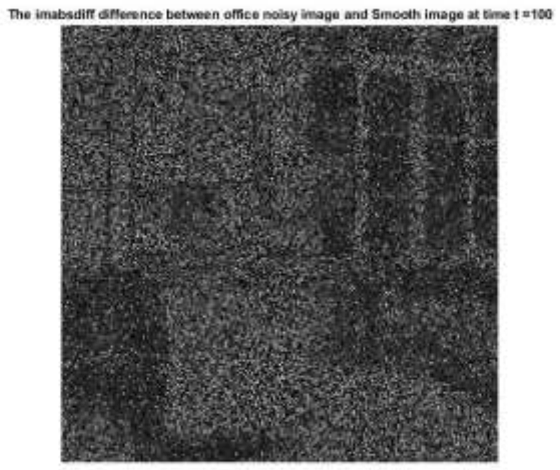

The imabsdiff difference between office noisy image and Smooth image at time t =100

*Figure 11. Absolute difference of office noisy image, and filtered image by non-linear diffusion filtering at t=1, t=10, and t=100*

Then, to realize the effect of D on denoising we compute the diffusivity D(x; y) on the office image devoid of any noise and visualize the diffusivity as a gray-scale image, and realized that the diffusion coefficient, D is one of the edges stopping function. It's known as edge stopping function as its ability to control the diffusion process with the diffusivity is upgrading in the inside of homogenous district while zero at the borderline. It gives the blurring impact lessen at any spots with high conceivable outcomes to be the borderline observed by the local gradient magnitude function, $|\nabla I|$. Hence, the image domain will not entirely be blurred out at the same rate and successfully maintain the edges features on the image. In an edge region, where the image gradient varies considerably, the diffusivity $D(\|\nabla I\|)$ can be used to control the image and achieve weaker smoothing, thereby successfully protecting the edge. In a flat region, strong smoothing must be achieved for noise removal.

All of the above facts can be proven according to the depicted gray-scale image in Figure12. it can be seen that the gradient modulus after t iterations is an edge detector, and its value is smaller in flat areas.

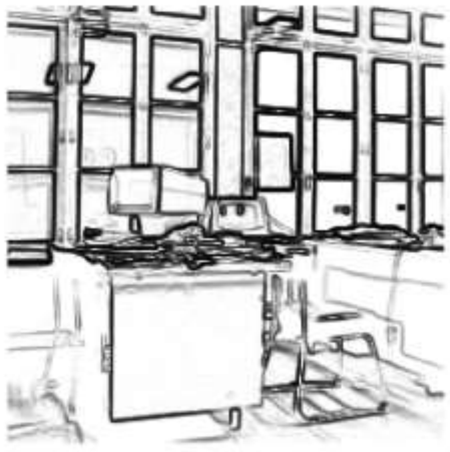
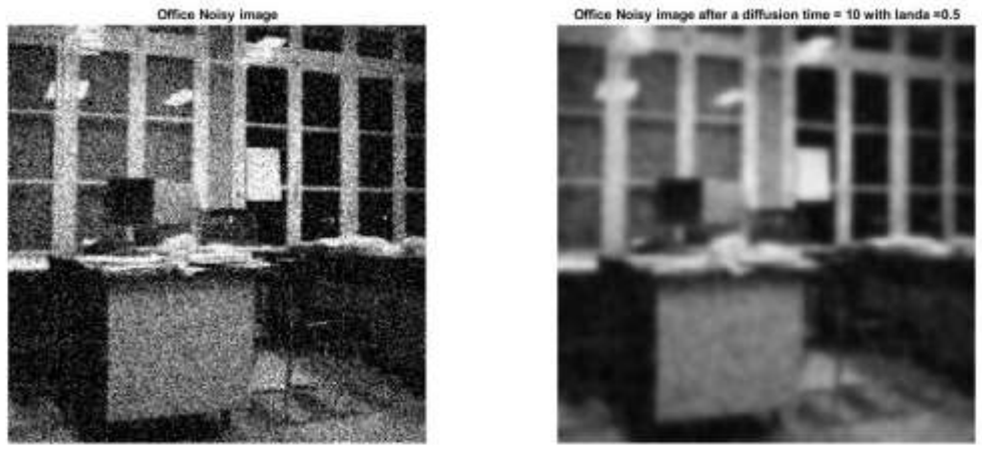


*Figure 12. Diffusivity as a gray-scale image*

We also analyze the effect of Parameter λ in the diffusion coefficient function on denoising and edge preservation by performing the diffusion on various λ values at a fixed time and we realized that λ acts as an edge strength threshold that determines the blurring and sharpening effects. If the λ value is too large, the diffusion process will be over smooth and result in a blurred image. In contrast, if the K value is too small, the diffusion process will stop the smoothing in early iterations and yield a restored image similar to the original one. So, as shown in Figure13. the constant λ controls the sensitivity to edges and is usually chosen experimentally or as a function of the noise in the image.


Office Noisy image
Office Noisy image after a diffusion time = 10 with landa =0.5

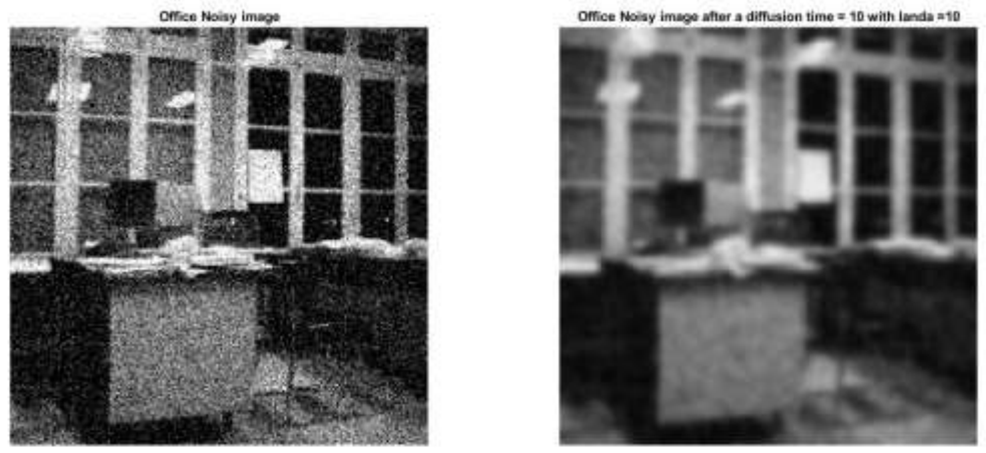

Office Noisy image
Office Noisy image after a diffusion time = 10 with landa =10

*Figure 13. comparing the effect of λ in denoising at two different values at a fixed time.*

Table5. shows the MSE, SNR, and PSNR between the noise free office image and filtered office image at different λ values.

*Table 5. MSE, PSNR, and SNR of office image and filtered office noisy image by non-linear diffusion filtering at different λ values*

| | λ = 0.5 | λ = 1 | λ = 2 | λ = 5 | λ = 10 |
|---|---|---|---|---|---|
| MSE | 0.0078 | 0.0083 | 0.0085 | 0.0086 | 0.0086 |
| PSNR | 21.0705 | 20.8066 | 20.6984 | 20.6532 | 20.6457 |
| SNR | 13.4676 | 13.2036 | 13.0955 | 13.0503 | 13.0428 |

## Results and Conclusion

In this paper, we perform three image denoising methods on a noisy image and analyze the effect of their various parameters on the final performance. In order to have a total comparison between the three image denoising methods explored in this study in preserving the edges, we came to these conclusions:

- Gaussian smoothing does not only reduce noise but also blurs important features such as edges and thus makes them harder to identify Since Gaussian scale space is designed to be completely uncommitted, it cannot take into account any prior information on structures that are worth being preserved or even enhanced.

- Linear diffusion filtering dislocates edges when moving from finner to coarser scales. So structures that are identified at a coarse-scale do not give the right location and have to be traced back to the original image. In practice relating dislocated information obtained at different scales is difficult and bifurcations may give rise to instabilities. These coarse to fine tracking difficulties are generally denoted as the correspondence problem. So linear diffusion approach is not efficient, because the diffusion operates in all directions leading to edge degradation.

- Inhomogeneous linear diffusion filtering is one of the simplest models for including a priori knowledge and for reducing the correspondence problem. Compared with homogeneous linear diffusion, edges remain better localized and their blurring is reduced. On the other hand for large t the filtered image reveals some artifacts which reflect the differential structure of the initial image.

## References

Yadav, Sudha, Swapnesh Taterh, and Mr Ankit Saxena. "A literature review of various techniques available on Image Denoising." (2021).

Egana, Alvaro. "Diffusion-based Image Filtering Technical Report Advanced Mining Technology-University of Chile." (2012).

Fan, Linwei, et al. "Brief review of image denoising techniques." *Visual Computing for Industry, Biomedicine, and Art* 2.1 (2019): 1-12.

Alisha, P. B., and K. Gnana Sheela. "Image denoising techniques-an overview." *IOSR J. Electr. Commun. Eng* (2016).

Jain, Meenal, Sumit Sharma, and Ravi Mohan Sairam. "Effect of Blur and Noise on Image Denoising based on PDE." *International Journal of Advanced Computer Research (IJACR) Volume-3 Number-1 Issue-8 March-2013* (2013).

Patil, Rajesh, and Surendra Bhosale. "Medical Image Denoising Techniques: A Review." *International Journal on Engineering, Science and Technology (IJonEST)* 4.1 (2022): 21-33.

Tomar, Richa, and Harbinder Singh. "Image Denoising Techniques: A Review."

Swamy, D., and P. K. Kulkarni. "a Basic overview on image denoising techniques." *Int. Res. J. Eng. Technol.* (2020): 850-857.

Rajni, Rajni, and Anutam Anutam. "Image denoising techniques-an overview." *International Journal of Computer Applications* 86.16 (2014): 13-17.