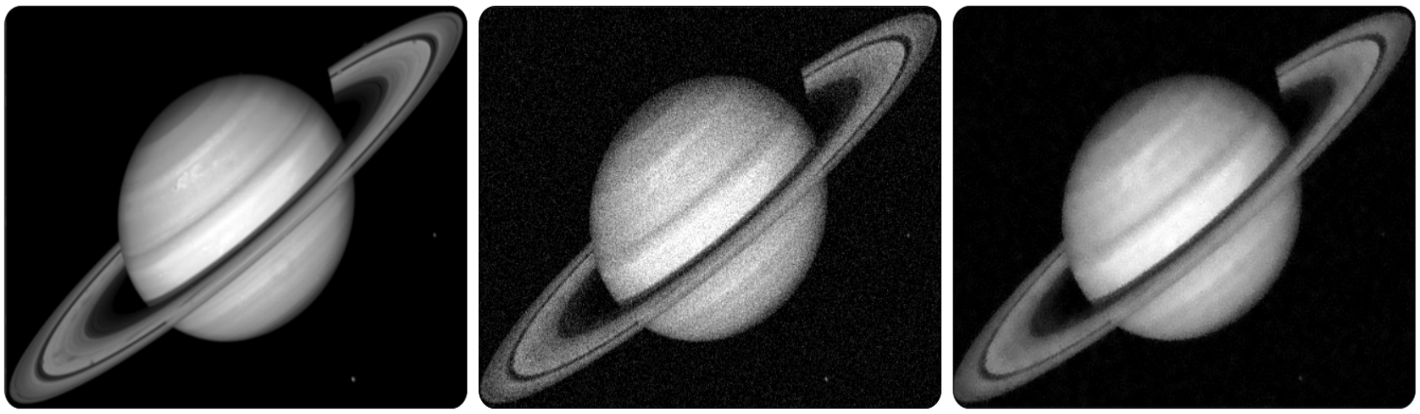
Department of Computer Science
Technical University of Cluj-Napoca

Image Processing

Laboratory activity

Adaptive image denoising with Wiener filter

Trufin Radu-Sebastian
Group: 30233
Supervisor: Asist.dr.ing Varga Robert
Date: 03-06-2021



Contents

TECHNICAL UNIVERSITY OF CLUJ-NAPOCA ROMANIA

∗ ∗ ∗

Introduction

This project aims to implement and test the Wiener Filter, which will filter out the noise from an image that is corrupted by Gaussian additive noise. Images are subject to a variety of degradations and this is the most important technique for removal of blur and noise in images due to linear motion or unfocussed optics. The Wiener filter has a variety of applications in signal processing, image processing, control systems, and digital communications, such as System identification, deconvolution, noise reduction and signal detection. The goal of this paper is to implement a C function that takes the input image without noise, corrupts the image and returns the "denoised" image as output.

State-of-the-art methods

To implement the Wiener filter in practice we have to estimate the power spectrum of the original image and the additive noise. For white additive noise the power spectrum, or the power spectral density (PSD) is a measure of the power across the frequency domain of a signal. We can acquire an estimate of the PSD at frequency, by multiplying the Fourier terms by their complex conjugate and scaling by the number of samples to produce a periodogram. To estimate the power spectrum of the original image many methods can be used.

The **Fourier** transform $\hat{F}(u,v)$ of the reconstructed signal has the following form :

$$\boxed{\hat{F}(u,v) = \frac{H^*(u,v)}{H^2(u,v) + \frac{S_y}{S_f}}\, G(u,v)}$$

Where $H^*(u,v)$ is the complex conjugate of the Fourier transform of the filter, $H^2(u,v)$ is the magnitude of the filter we have estimated, S_y and S_f represent the power spectrum of the noise and the signal and $G(u,v)$ is our observation. The power spectrum is the magnitude of the Fourier transform of the correlation function. The expression Sy/Sf is pretty difficult to estimate and some methods replace it with a constant K. However, if the filter is implemented with this constant, the reconstructed image is suboptimal in the mean square sense. There is no standard method for determining the K constant (in most cases it is found experimentally). After applying the inverse Fourier transform to the $\hat{F}(u,v)$ term we obtain the reconstructed image which minimizes the mean squared error, which is a very common metric that estimates the performance of a filter. See [2] and [5] for more mathematical explanation and theory.

Proposed method

Wiener filters are usually applied in the frequency domain, but this paper will focus on the implementation in the spatial domain. Sometimes the result is combined from both domains, each one of them having it's own weight. The wiener2 function from **Matlab** will be implemented using **OpenCV**, which is an open source **C++** library for image processing and computer vision, originally developed by **Intel**. The function tailors to the local image variance. Where the variance is large, wiener2 performs little smoothing and where the variance is small, wiener2 performs more smoothing. The filter will work best when the noise is a white additive noise, such as a **Gaussian noise**. Consider $x(i,j)$ as the input image with size $N \cdot M$ (N rows and M columns), which has no noise. The additive Gaussian noise $n(i,j)$ will be applied to this initial image and we will get the corrupted $y(i,j)$ image, which we will have to 'denoise':

$$\boxed{y(i,j) = x(i,j) + n(i,j)}$$

We want to obtain a new image $\hat{x}(i,j)$ that has a good compression quality. This performance measure is given by the **Mean Squared Error** (MSE), which represents the cumulative squared error between the compressed and the original image:

$$\boxed{MSE(\hat{x}) = \frac{1}{NM}\sum_{i=0}^{N-1}\sum_{j=0}^{M-1}\left(\hat{x}(i,j) - x(i,j)\right)^2}$$

Another used metric is the **peak signal-to-noise ratio** (PSNR), which is an expression for the ratio between the maximum possible value (power) of a signal and the power of distorting noise that affects the quality of its representation. Because many signals have a very wide dynamic range, the PSNR is usually expressed in terms of the logarithmic decibel scale.

$$\boxed{PSNR(\hat{x}) = 10 \cdot log_{10}\left(\frac{1}{MSE(\hat{x})}\right)}$$

In order to reconstruct the image, we will filter the noisy image spatiotemporally, which means for each pixel in the noisy image we will calculate a spatiotemporal variance $\sigma_x^2(i,j)$ and mean $\mu_x(i,j)$. In digital image processing, the pixels of a white noise image are typically arranged in a rectangular grid, and are assumed to be independent random variables with uniform probability distribution over some interval. If we are working with an AWGN (additive white Gaussian noise), then the Wiener filter is linear and we get the following equation in order to calculate $\hat{x}$:

$$\boxed{\hat{x}(i,j) = \frac{\sigma_x^2(i,j)}{\sigma_x^2(i,j) + \sigma_n^2}\left[y(i,j) - \mu_x(i,j)\right] + \mu_x(i,j)}$$

Where σ_n^2 is the variance of the noisy image. It can be calculated by using the definition:

$$\sigma_n^2 = \sum_{i=0}^{L} (i - \mu_n)^2 \cdot f(i)$$

$$\mu_n = \sum_{i=0}^{L} i \cdot f(i)$$

Where μ_n is the mean of the noisy image and $f(i)$ is the probability density function, which tells us the probability that a pixel from an image has the intensity i, $i \in [0, L]$. Determining the histogram of the noisy image is an efficient and simple way of calculating the probability density function.

$$f(i) = \frac{h(i)}{NM}$$

After processing the noisy image, we can start calculating the spatiotemporal mean and variance, which means for each pixel in the noisy image we will consider a $(2r+1) \times (2r+1)$ region and calculate a local mean or variance:

$$\mu_x(i,j) = \frac{1}{(2r+1)^2} \sum_{u=i-r}^{i+r} \sum_{v=j-r}^{j+r} y(u,v)$$

$$\sigma_x^2(i,j) = \frac{1}{(2r+1)^2} \sum_{u=i-r}^{i+r} \sum_{v=j-r}^{j+r} (y(u,v) - \mu_x(i,j))^2 - \sigma_n^2$$

When the spatiotemporal signal variance is much smaller than the noise variance σ_n^2, the estimate approaches to the spatiotemporal average and when the spatiotemporal signal variance is much larger than the noise variance, the estimate approaches to the noisy image value to avoid blurring. This type of filtering may either cause blurring or does not result in satisfactory noise reduction. We need to introduce a new type of filter that overcomes those limitations of the initial design.

The solution is introducing an **Adaptive Weighted Averaging** (AWA) filter which assigns a weight to each image value. The expressions for the spatiotemporal mean and variance remain the same, but this time the weight is not equal to the window size:

$$\mu_x(i,j) = \sum_{u=i-r}^{i+r} \sum_{v=j-r}^{j+r} w(i,j,u,v)\, y(u,v)$$

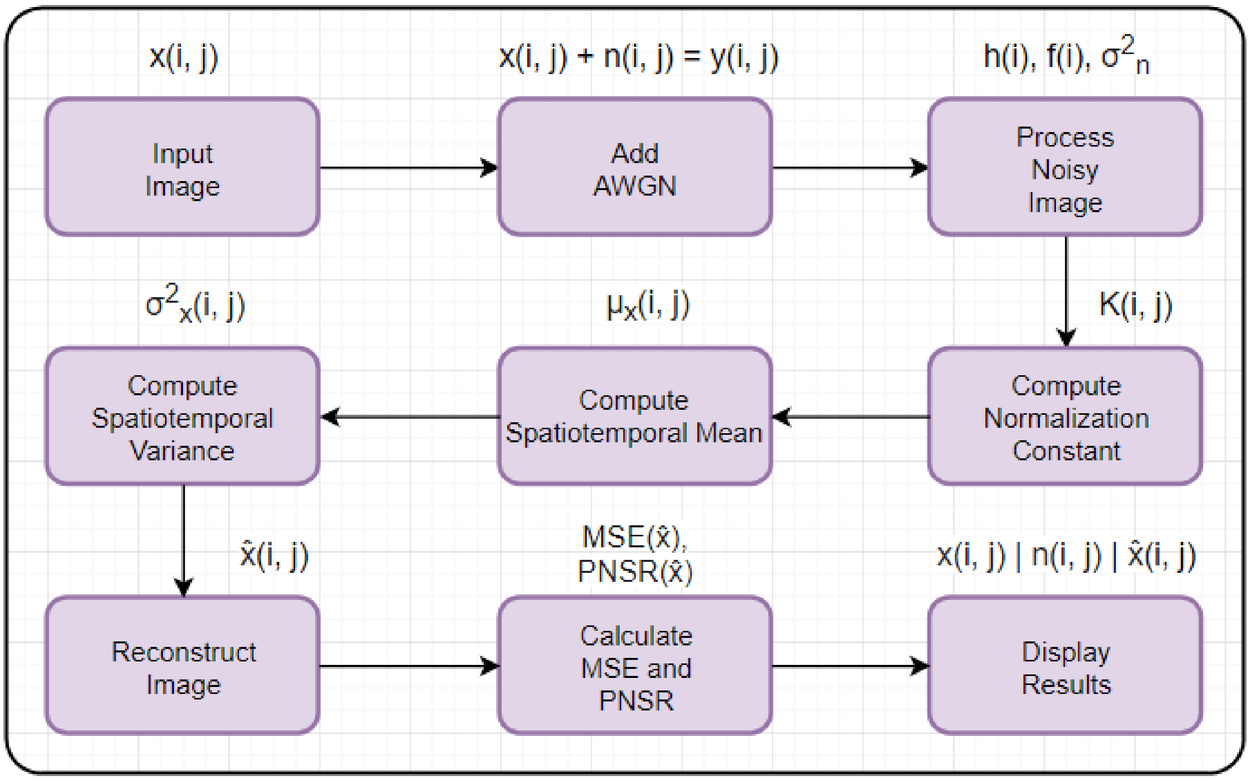
$$\sigma_x^2(i,j) = \sum_{u=i-r}^{i+r} \sum_{v=j-r}^{j+r} w(i,j,u,v)\, (y(u,v) - \mu_x(i,j))^2$$

The AWA filter will weigh down the effect of the values that are different from the $y(i,j)$ intensity and avoids excessive blur or inefficient filtering. We define the following estimates:

$$\boxed{\begin{aligned} w(i,j,u,v) &= \frac{K(i,j)}{1 + a\max\left(\epsilon^2, (y(i,j) - y(u,v))^2\right)} \\ K(i,j) &= \left(\sum_{u,v} \frac{1}{1 + a\max\left(\epsilon^2, (y(i,j) - y(u,v))^2\right)}\right)^{-1} \end{aligned}}$$

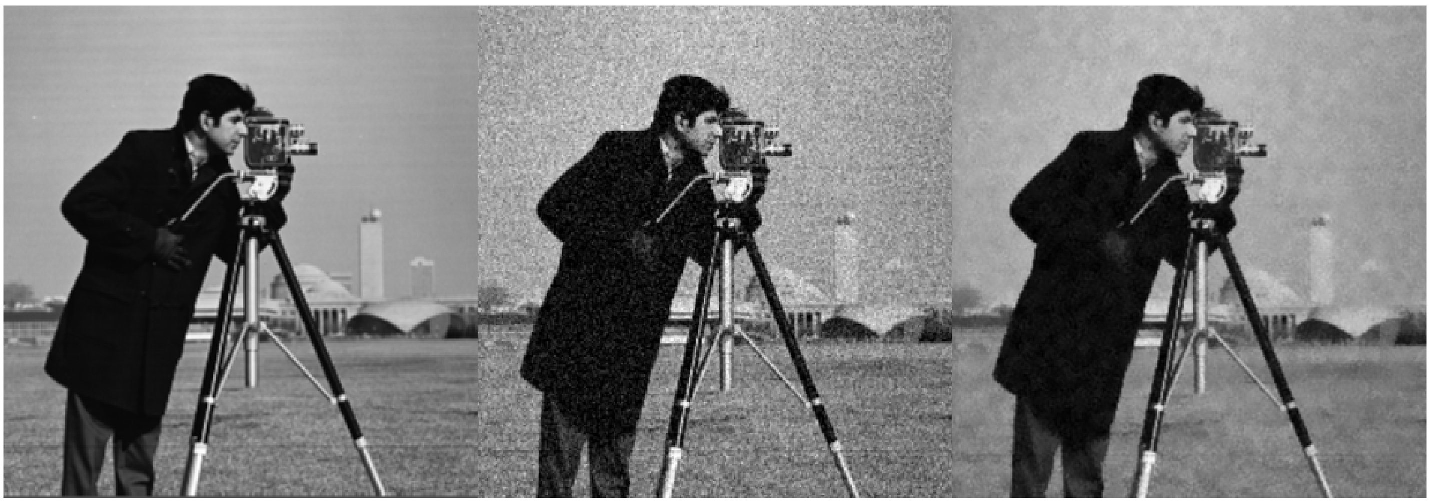
Where $K(i,j)$ is a normalization constant. The quantities $a > 0$ and ϵ are the parameters of the filter. Check [1] for the mathematical proof that $\epsilon^2 \approx b\,\sigma_n^2$ and $a\epsilon^2 \gg\ 1$

All the main steps explained in this section are found in this block diagram. In summary, we corrupt the input image $x(i,j)$ by adding some AWGN and then we compute the histogram of the noisy image $y(i,j)$ which helps us calculate the noisy variance that we need in order to reconstruct the image. The spatiotemporal mean and variance use the normalization constant, therefore we need to compute it first, and then we can find the weight of every pixel for $\sigma_x^2(i,j)$ and $\mu_x(i,j)$. After using the linear formula to reconstruct the image, we display everything we calculated, including the performance measures (MSE and PNSR).



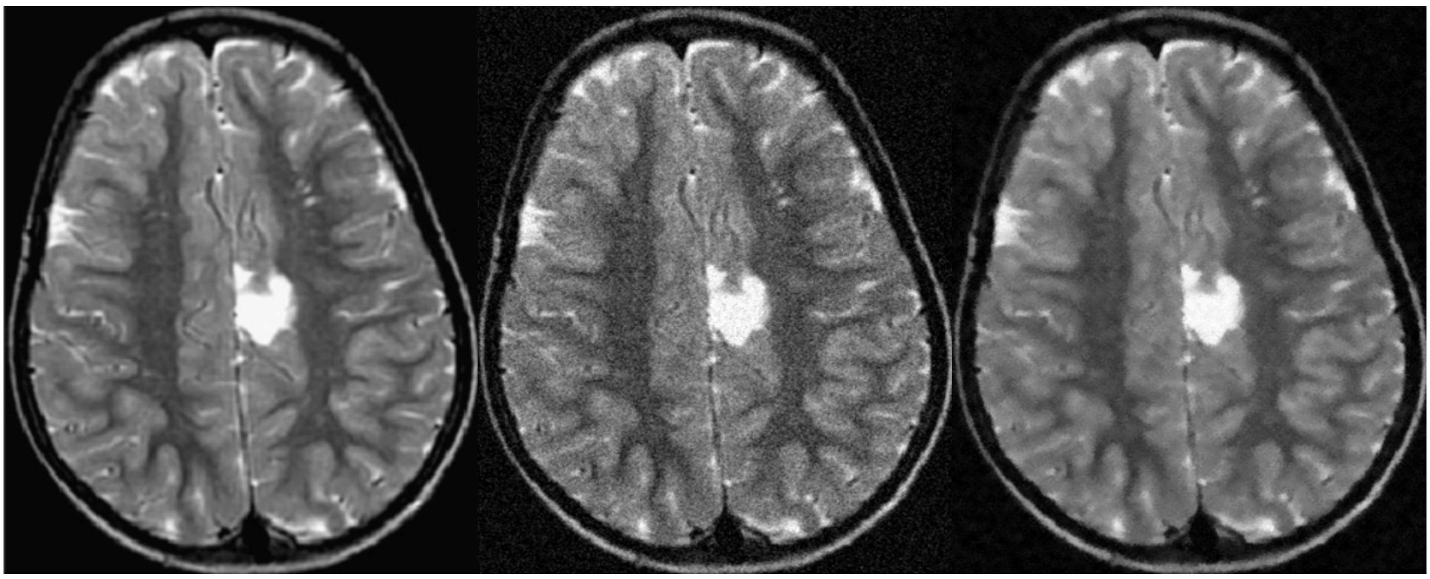
Experimental results

When you run the application, you are asked to give 3 important input values for the filter parameters. The chosen value for r is 9 because if r is too small it might not denoise the image well and if r is too big it might blur the image too much. The value for a should be much larger than the value of b and the value of b shouldn't be higher than 10 (10 times the noisy variance). ($r = 9$, $a = 1e6$, $b = 10$)



(a) Original image (b) Noisy image (c) AWA filtered image

Figure 3.1: Cameraman (PNSR = 27.6dB, MSE = 0.44)



(a) Original image (b) Noisy image (c) AWA filtered image

Figure 3.2: Human brain (PNSR = 28.2dB, MSE = 0.37)

Conclusions

The local adaptive filtering succesfully removed the noise from the image and returned a low mean-squared error and peak signal-to-noise ratio. The program works quite well even with an input image that has a large amount of noise added. However, if the image resolution is too big, the computation time might exceed 20-25 seconds which is undesirable. Some future developements in this project might be optimizing the functions that calculate the spatiomporal means and variances and also combining this filter with an adaptive Wiener filter from the wavelet domain.

Bibliography

[1] http://www.heathershrewsbury.com/dreu2010/wp-content/uploads/2010/07/AdaptiveMotionCompensatedFilteringOfNoisyImageSequences.pdf

[2] http://www.dfmf.uned.es/ daniel/www-imagen-dhp/biblio/adaptive-wiener-noisy.pdf

[3] https://en.wikipedia.org/wiki/Wiener_filter

[4] https://www.mathworks.com/help/images/ref/wiener2.html

[5] https://core.ac.uk/download/pdf/38895005.pdf

* * *