Master MVA
Image Denoising

Experimental Report

Realized By: Eya Ghamgui

Summary of the algorithm:

The Ponomarenko algorithm is a noise estimation algorithm from an image without pre-filtering. It is a part of a compression algorithm. The estimation of the noise is based on the calculation of the variance of the high frequencies of overlapping blocks chosen to have the lowest energy. The choice of blocks is developed according to two methods. One is iterative and the other is statistical.

As a first step, the algorithm extracts blocks from the image with a fixed edge size. After that, it computes its Discrete Cosine Transform. This method transforms the image from the spatial domain to the frequency domain.

As a second step, the algorithm labels the blocks. Each coefficient of the transformed block is considered to be a low or medium/high frequency coefficient. Only the low frequency coefficients will be used in calculations. Medium and high frequencies are eliminated to avoid edges and textures effects. Thus, the noise estimate will not be affected by additional information from the original image.

The next step consists in calculating the empirical variance of each block by averaging the square of its coefficients that are associated with low frequencies. All these values of the variance are stored to determine the number of blocks that are used to estimate the noise. This number is known as “K” and it is determined iteratively. In a nutshell, the algorithm searches for the index of the variance that adequately retains the lowest values. Another more accurate approach to choosing “K” is to use a fixed percentile p = 0.5% instead of iterations.

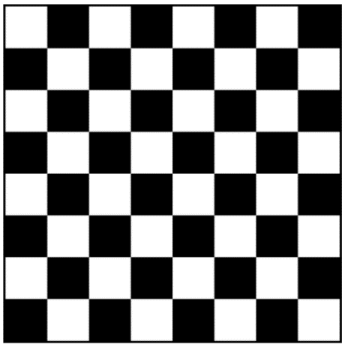
At each high frequency, new variance is calculated by averaging the square of the coefficients at this frequency for all the previously selected blocks. These frequencies are estimated to contain the noise. Finally, the estimate of the noise is given by the median of the calculated variances.

Article 1: Analysis and Extension of the Ponomarenko et al Method, Estimating a Noise Curve from a Single Image

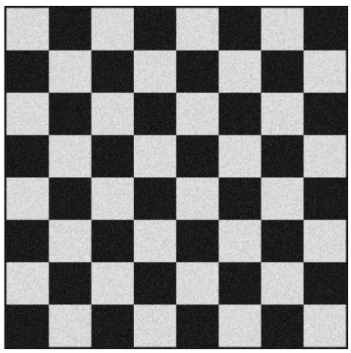
This algorithm is used to estimate the noise in an image. For a Given image, the algorithm returns curves of standard deviation for each color channel (Red, Green and Blue) as a function of the intensity of the pixels. In addition, this algorithm returns all possible scales of the image dividing by 2 the scale at each level. Hence, the number of points on the curves is reduced because the number of patches is reduced.

We will do some experiments on different images having different patterns (chessboard, highly textured image, and a flat image). In this part, we will fix the noise value to: "5350 u + 11" for all the images, where u is the input noisy image.

1. **Chessboard image:**

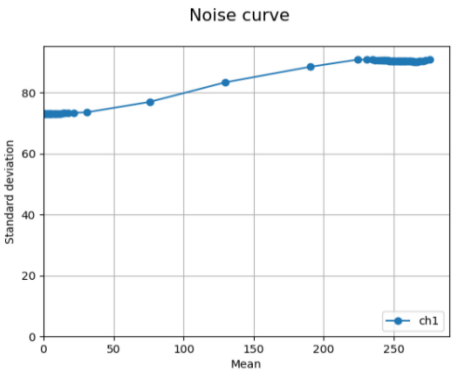


Original Image

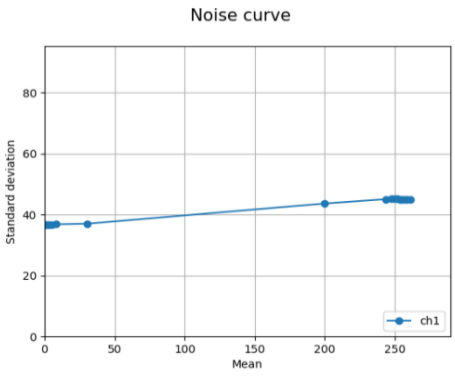


Noisy Image

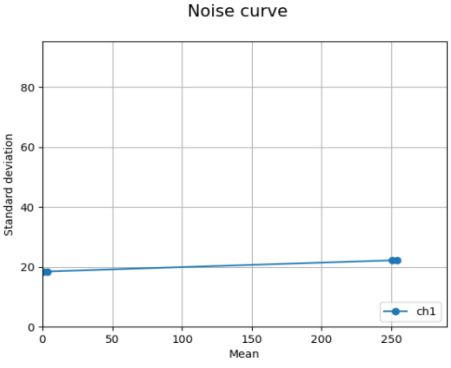
- Curves:



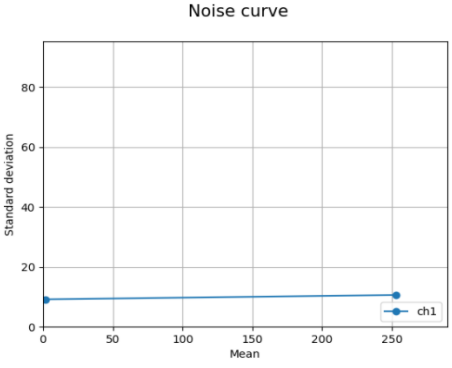
Scale 0



Scale 1



Scale 2



Scale 3

- Performances of the algorithm:

The chessboard image is an image that presents periodic discontinuities. The discontinuity is considered as a sudden change from white to black or from black to white. In frequency space, these discontinuities (edges) are represented with very high frequency coefficients. In fact, the DCT coefficients are high for sharp changes.

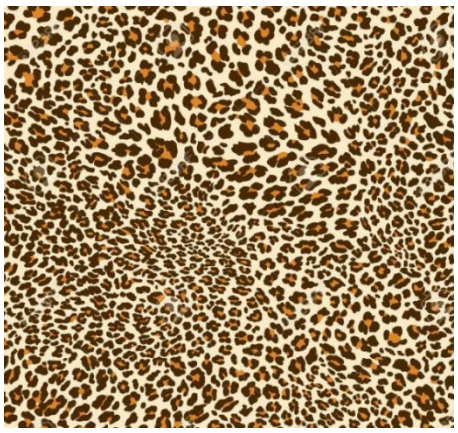
The labeling part of this algorithm consists in eliminating these frequencies. Thus, the algorithm is able to correctly estimate the standard deviation despite the presence of these remarkable forms. In fact, the standard deviation at scale 0 is close to the applied noise. We can conclude that the method achieves an acceptable estimate of the noise.

- Defects of the algorithm:

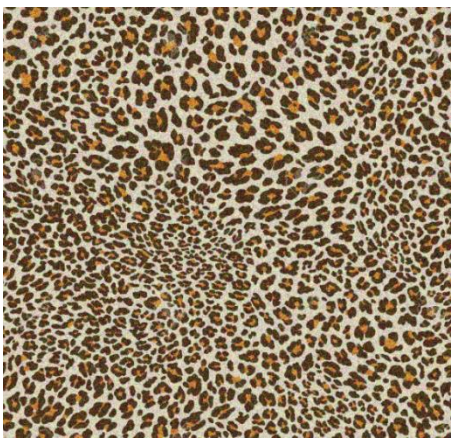
We can notice from the previous plots that there is a high number of points in high and low intensities. The points in high intensities correspond to the estimation of noise in white regions of the image. The points in low intensities correspond to the estimation of the standard deviation of the noise in black regions. In addition, we notice that the higher intensities have the higher noise. Thus, the estimation of this algorithm is affected by the intensity of the pixels in the noisy image.

In addition, the standard deviation of the noise is almost halved from one scale to another. By changing the scale of the image with the zoom by 2, the algorithm estimates half of the standard deviation. Thus, the standard deviation estimate depends on the number of pixels in the patch.

2. Highly textured image:

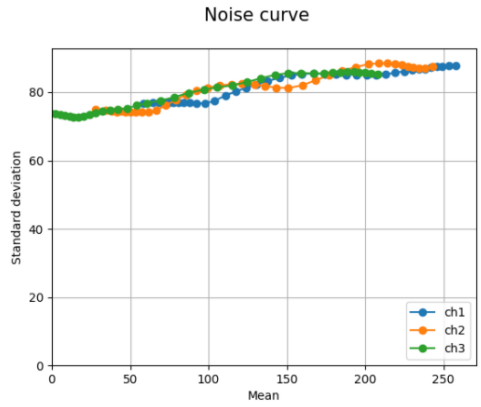


Original Image

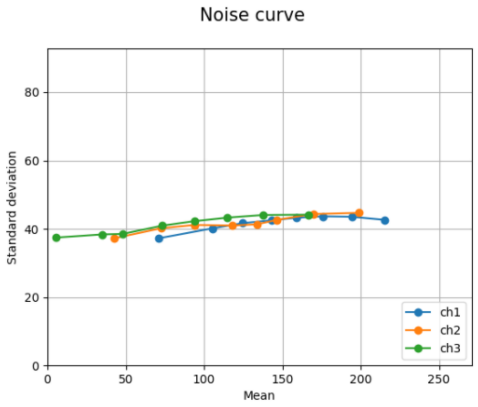


Noisy Image

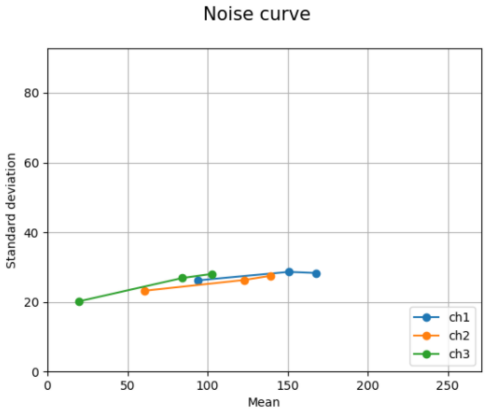
- Curves:



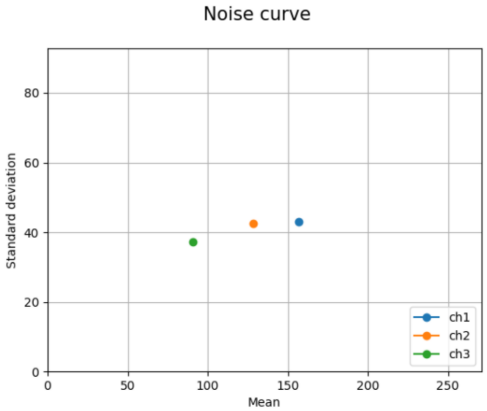
Scale 0



Scale 1



Scale 2



Scale 3

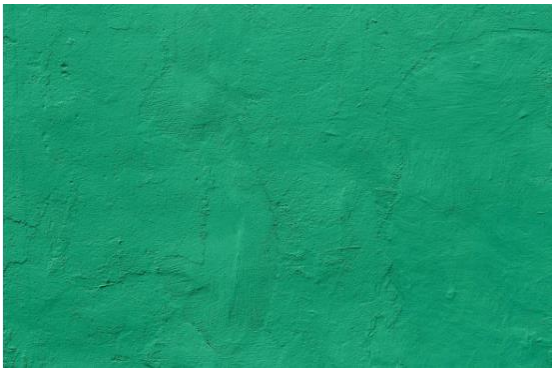
- Performances of the algorithm:

This image is an image with repeated structures. It stands out for its very important color changes and repeated shapes. Thus, it is considered as a very textured image. This texture will affect the frequency coefficients. For this type of images, the coefficients will be very high. Despite the texture of the image, the algorithm gives a good estimate of the noise for scale 0. In addition, there are a lot of dots in all intensities due to the large variations of colors in the image. This large variety lead the algorithm to equally estimate standard deviation in all intensities and give results in the same range of values as the previous image (chessboard image).

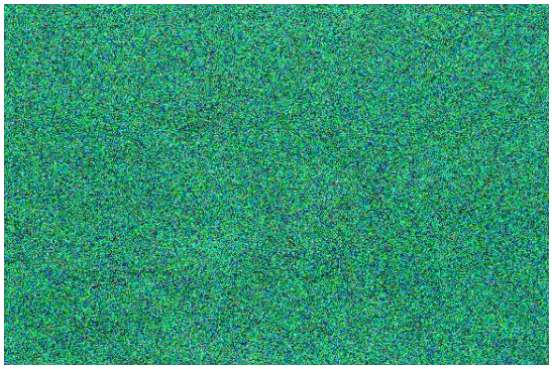
- Defects of the algorithm:

This image is very textured. Their corresponding high frequencies can affect the estimation. In fact, the algorithm, will consider these values as noise which will distort the estimation. We can notice that the last scale (scale 3) gives points around 40 which is the same for scale 1, despite the two times zoom-down of the image. Here, the texture of the image, which is very complicated, leads the algorithm to make mistakes in estimating the variance of the noise.

3. Flat image:

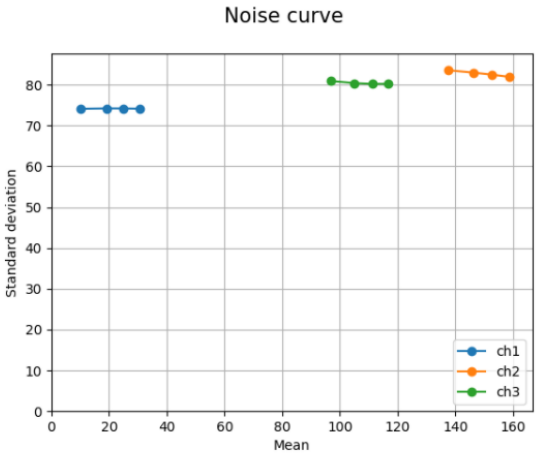


Original Image

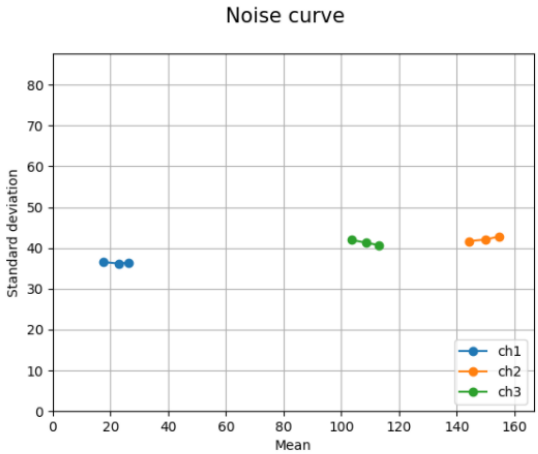


Noisy Image

- Curves:



Scale 0



Scale 1

- Performances of the algorithm:

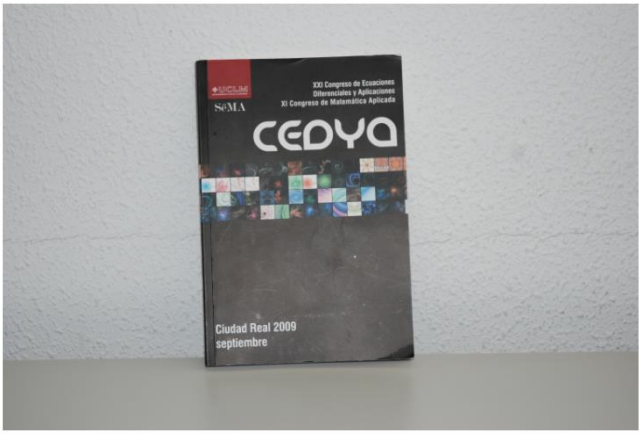
The image is flat, this is remarked from the number of points detected by the algorithm. For scale 0, the algorithm is able to correctly estimate the standard deviation of the noise. The estimated values are in the same range of the two previous images at scale 0. In addition, we can notice that each channel gives a different estimation value which means that the noise differs according to the channels. However, their values are close to each other.

Conclusion: the algorithm is robust to different image structures. In addition, the texture of the image has a small effect on the estimation of the noise. It can affect the estimate only in the case of very complicated textures.

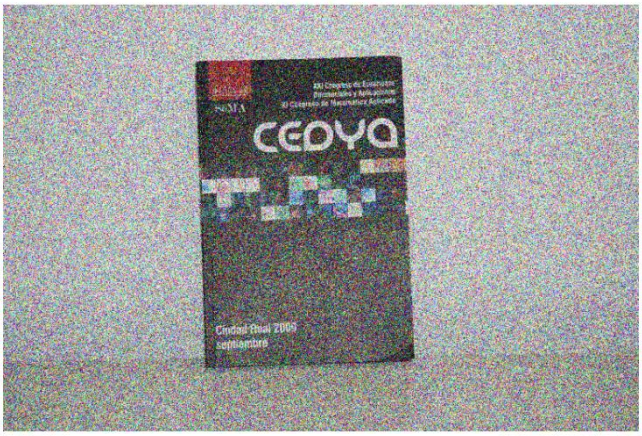
Article 2: Multi-Scale DCT Denoising

Textures or high-detailed structures contain information that can be exploited in pattern recognition and classification. If an acquired image is noisy, noise removal becomes an operation to improve image quality before further stages of processing. Among possible variants of denoising, there is discrete cosine transform (DCT) known to be able to effectively remove additive white Gaussian noise. Consequently, many denoising algorithms using this method are developed. The multi-scale DCT denoising is a patch-based method. It improves denoising results and visual image quality and reduces the artifacts. First, this method decomposes the image in a pyramid DCT images with different scales, which maintains white noise at all scales. After that, it applies the two-step DCT method to all scales. This method is considered to be a simple scale denoising. Finally, it merges the results of all the scales.

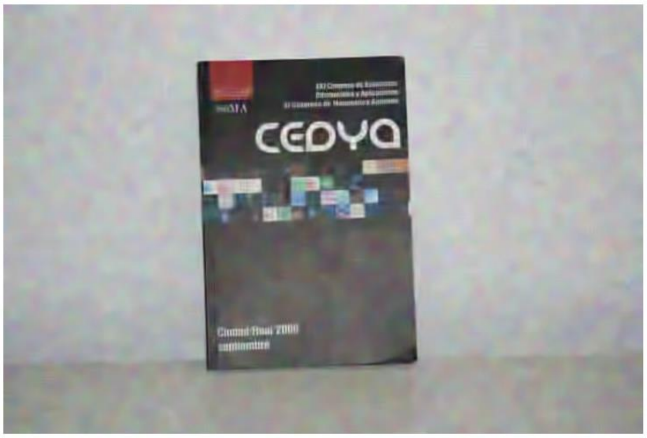
First experiment (book image):



Original Image

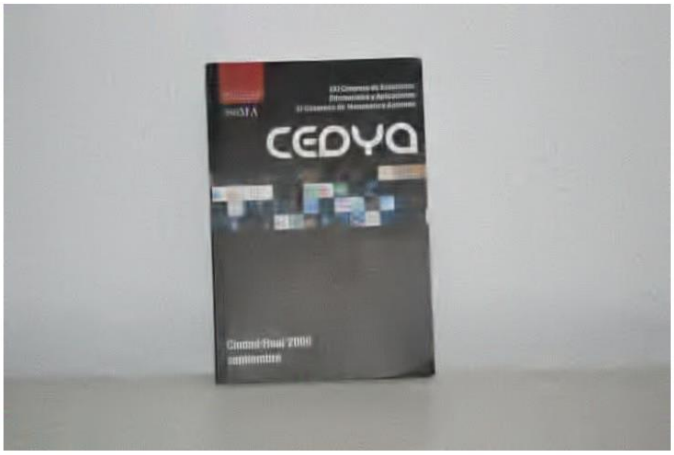


Noisy Image (σ = 72)



1-Scale DCT (PSNR = 30.43 dB)

MS DCT Denoising Without Aggregation
(PSNR = 31.04 dB)



MS DCT Denoising Only Hard Threshold
(PSNR = 30.72 dB)

MS DCT Denoising
(PSNR = 31.36 dB)

- Interpretations:

From the previous denoising results, we can notice that the 1-scale DCT Denoising has the worst denoising results. In fact, flat regions present a lot of artefacts, and the noise is still noticeable. This result may be due to the fact that this method uses the hard thresholding of a patch-wise DCT of the image. Therefore, we can say that the soft thresholding is better in denoising than the hard thresholding.

From the resulted images of the multi-scale denoising, we notice that denoising results have improved considerably, especially in flat regions. The PSNR of these images has improved when compared to the simple DCT. In addition, the back of the book does not show any artefacts or noise. This result can be explained by the fact that these areas have low frequencies of noise. Therefore, it will be easier for the algorithm to remove the noise in these zones. However, we can clearly notice that these regions become blurry which means that we have lost information about the details that are present in these regions, for example, the texture of the wall.

Moreover, we can notice that the algorithm is not able to reconstruct the cover of the book. This region is considered as a highly textured region because it presents a lot of structures such as the small pictures and text zones. We can deduce that the algorithm makes a lot of errors during the construction of these regions because these areas have very high frequency coefficients and thus, they are confused with the noise. These errors, lead to the loss of information when constructing the image.

Multi-scale denoising without aggregation presents a lot of artifacts at the edges. In fact, this problem occurs due to the high frequency in the lower resolution levels which are likely damaged or removed by the denoising method. However, this is not the case for the multi-scale DCT denoising using the two-step method. Actually, the aggregation helps to eliminates ringing artifacts caused by the denoising of the coarse scale images. It does so

by dropping the high frequency components from the subsampled images of the pyramid. We can conclude that the aggregation improves the PSNR and the denoising results.

Second experiment (changing parameters):

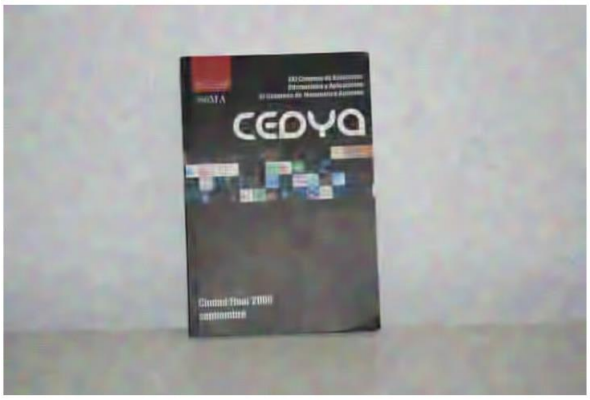
- Changing the "Multiscale recomposition factor"

MS DCT Denoising
(PSNR = 30.32 dB), frec = 1

- Interpretations:

This factor aims at reducing the Gibbs artifacts resulting from the global DCT subsampling. From the previous result, we notice a lot of artifacts in the image when choosing this factor equal to 1. These artifacts are the oscillations of Gibbs. In addition, the PSNR has decreased from 31.36dB to 30.42dB, which means that the denoising in this case is worser. To avoid these artifacts, we must choose frec < 1.

- Changing the "DCT patch size"



1-Scale DCT (PSNR = 30.56 dB)

- Interpretations:

Here, we have increased the size of the patch from 14 to 16, we notice that the 1-scale DCT denoising has improved, for example in homogeneous areas. This is explained by the fact that in homogeneous zones the larger patch will give more observations for the same pixel due to the overlap of patches which will help to reduce the noise.

Moreover, by increasing the size of the patch, we notice that the MS DCT Denoising is always better than the 1-scale DCT Denoising. However, for smaller patch size, the MS DCT Denoising gives better gain than with a larger one.