Adaptive Wiener Filtering of Noisy Images

Student: Gyarmathy Tímea
Group: 30433

Project specification

Corrupt a grayscale image with gaussian noise. Implement and apply the filter from the paper [1]. Calculate the mean squared error. Compare with Gaussian blur.

Project implementation

Signal-independent zero-mean white Gaussian noise

In order to generate this, we have to calculate the *n(i, j)* from the formula

$$y(i,j) = x(i,j) + n(i,j)$$

Where *y(i, j)* is the noisy measurement, *x(i, j)* is the noise-free image and *n(i, j)* is additive Gaussian noise. Using the default random engine from the standard library, we can generate a noise with a specified mean and standard variance by instancing a normal distribution on them. Then we call the random engine with the normal distribution for each pixel of the original image and add the generated number to it.

In code, the instancing would look like:

```
std::default_random_engine generator;
std::normal_distribution<double> dist(NOISE_MEAN, NOISE_VAR);
```

Then, calling it with `dist(generator)`, it produces a random number for our noise.

In the above syntax, I set NOISE_MEAN macro to 0, and the NOISE_VAR to 10. The results of noising can be seen on Fig. 1.

Mean squared error

To analyze the effectiveness of the filter, I implemented a function to calculate the mean squared error, by simply following the formula from the paper [1]. The mean squared error will be referred to as MSE later in this documentation.

The MSE of the original and the noisy image on the Lena sample image is 99.7588.

Gaussian blur

As the specification says, we must compare the resulted filtering with Gaussian blur. Following the instructions on how to implement this, I have a function which computes the Gaussian blurred image on the noisy one, using half the window size to compute the kernel as the Wiener filter I implemented. The reason behind this is that with the actual window size, the image became way too blurred.

The result of the MSE between the original and the Gauss blurred noisy image is 52.7905 so we can already see the error decreased to half of its value. Fig. 2. shows the images to this process.

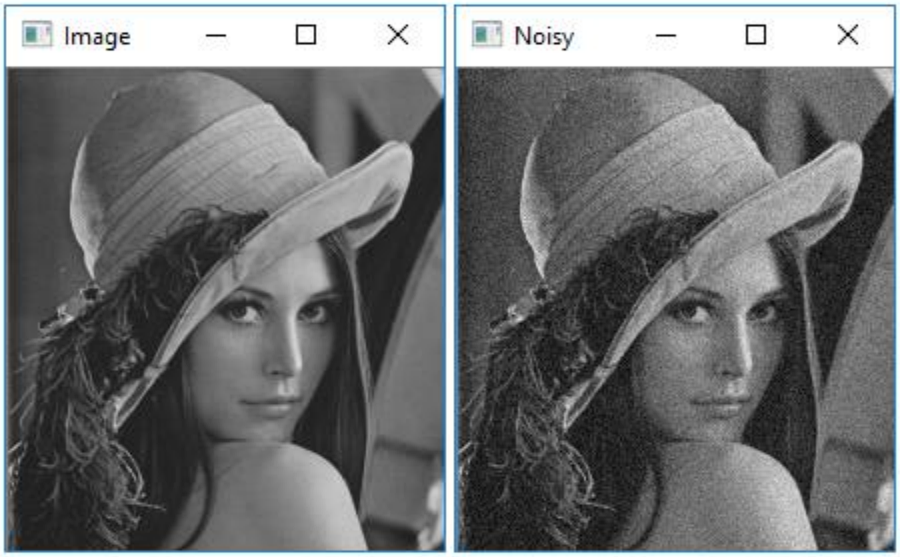

Figure 1. The original Lena sample image and the resulted noisy one

Wiener filters

The paper presents the Wiener filter in its simplest form:

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

This is the main function used to calculate the denoised image ($\hat{x}(i, j)$) while we change the estimation of the mean ($\mu_x(i, j)$) and the standard deviation ($\sigma_x(i, j)$).

I implemented the first description of the Wiener filter, by estimating the mean and the standard deviation over a uniform moving average window of size *(2*r + 1) * (2*r +1)*. In these formulas, the standard deviation of the noise is assumed to be known, so I used the one I set when implementing the normal distribution of the noise generator. The resulting MSE shows a huge improvement as it is 41.5521.

The functions I implemented for calculating the standard deviation return its squared value, as the main filter function uses it squared. The results of this simple Wiener filter can be seen on Fig. 3. The MSE of this method is 47.4166.

When transitioning to the weighted form, there is an intermediary filtering method, which uses the same estimation of the mean as above, but a weighted form of the standard deviation. The weights are calculated based on the moving window and some constant values specified beforehand:

- $\epsilon = 2.5\ \sigma_n$, where σ_n is the standard deviation of the noise
- $a > 0$ which we choose such as $a\epsilon^2 >> 1$

We cannot see much of the effect of changing the value of *a* between 0.1 and 5. It only makes the image a bit sharper or allows more noise to remain. After some testing, I set the value of a to 0.5. The results of this filter can be seen on Fig. 4.

The main idea of the project, implementing the adaptively weighted averaging (AWA), uses the weighted form also for the mean value of the pixels within the moving window. Results can be seen on Fig. 5. The MSE of the original and the filtered is 51.112 which shows a slight improvement compared to the Gaussian blur, but we can see on the pictures, that in fact the AWA filtered image looks way smoother on continuous surfaces than the blurred one.

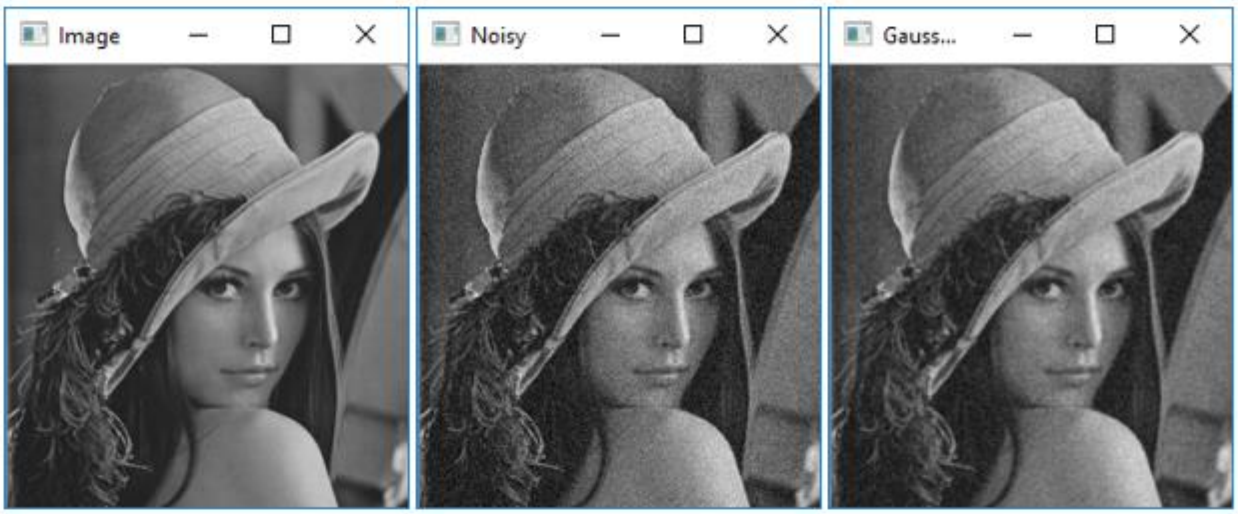

Figure 2. Effects of the Gaussian blur with window size 3x3

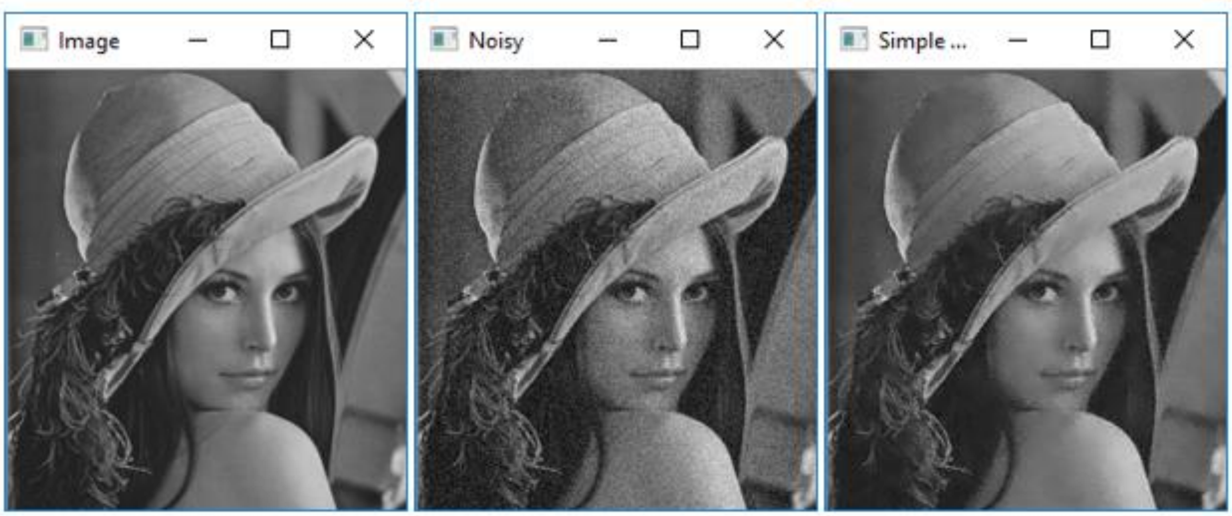

Figure 3. Simple Wiener filter

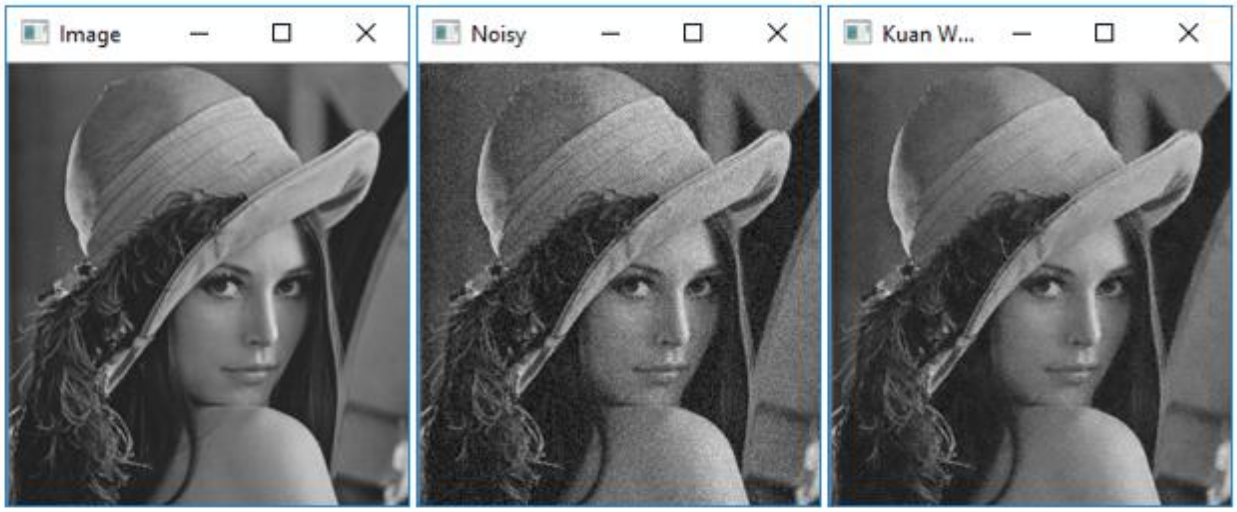

Figure 4. Kuan Wiener filter, only the standard deviation is weighted

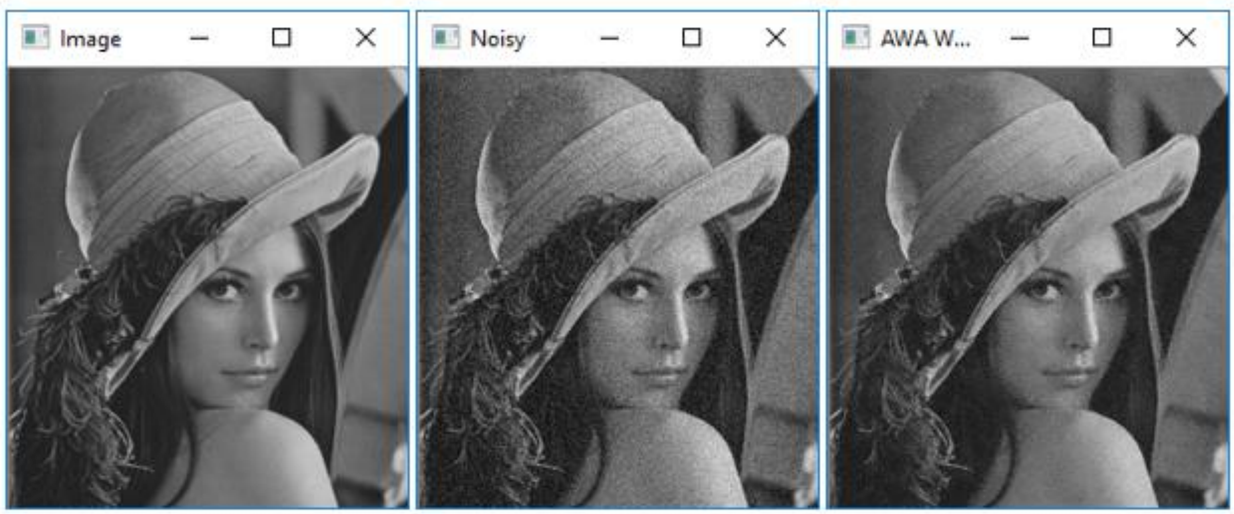

Figure 5. AWA filter

Conclusions

Judging by the numbers and by eye, we can clearly see how the simple Wiener filter is the closest to the original, maybe some more improvements on the weights could get the AWA to work better.

Compared to the Gaussian blur, all filters perform better, and judging by eye, all look better.

References

[1] F. Jin, P. Fieguth, L. Winger and E. Jernigan, *Adaptive Wiener Filtering of Noisy Images And Image Sequences,* http://www.dfmf.uned.es/~daniel/www-imagen-dhp/biblio/adaptive-wiener-noisy.pdf.