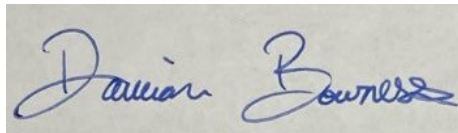


Part I & II

Review and implementation of
Buades et al. “A non-local algorithm for image denoising” (2005)

Damian Bowness
(21875841)
COMP 478 sec. DD
Prof. Yiming Xiao
December 11, 2021

*I certify that this submission is my original work
and meets the Faculty's Expectations of Originality*

A handwritten signature in blue ink, reading "Damian Bowness", is centered on a light gray rectangular background.

Damian Bowness

Part I

The paper, "A non-local algorithm for image denoising" by Buades, Coll, and Morel proposes a new denoising evaluation tool as well as a new denoising algorithm. Their proposed denoising evaluation tool, "Method Noise", isolates and displays the noise removed from an image by a denoising algorithm. This new evaluation tool allows for a visual inspection of the information removed from the original noisy image. A good denoising algorithm should produce Method Noise similar to white noise. The Method Noise produced from many algorithms includes structural information such as edges. Therefore, the authors of the paper propose a new denoising algorithm, the "Non-Local Means", to minimise the loss of structure when denoising an image.

Method Noise is a tool to evaluate the information a denoising algorithm removes from an image. Let Method Noise be $n(u, D_h)$, where u , is the original noisy image and D_h is a denoising function with parameter h . Applying D_h , to the image u produces the denoised image $D_h(u)$. Therefore, Method Noise is the difference between the original image, u , and the denoised image, $D_h(u)$

$$n(D_h, u) = u - D_h(u)$$

If a denoising algorithm works well there should be no structural information in the Method Noise which should look like white noise.

The Non-Local Means (NLM) denoising algorithm proposed by the authors extends neighbourhood filtering to take advantage of the Gaussian distribution of noise to average out the noise for similar pixel values throughout the image. This means that the estimated value for a pixel, $v_e(i)$, is computed as the weighted average of all the other pixels in the image, v

$$v_e(i) = \sum_{j \neq i} w(i, j) v(j)$$

The weights satisfy the conditions, $0 \leq w(i, j) \leq 1$ and $\sum_j w(i, j) = 1$. The weight, $w_{i,j}$, is determined by the similarity between pixel i 's neighbourhood and pixel j 's. The Euclidean distance between corresponding pixels determines similarity. A Gaussian kernel is applied to the Euclidean distances. Therefore as distance increases the weights tend to zero exponentially.

Theoretically, NLM computes a denoised pixel value from every other pixel. However, the authors only implement an approximation of the theory using a 21×21 "search window". While the results produced from this approximation are impressive and demonstrate the algorithms effectiveness, it would be interesting to see the results of a full image search. The authors note the computational complexity of the approximation but make no statement about the full implementation nor do they compare it to the computational complexity of the other denoising algorithms. With that said, exploiting the Gaussian distribution and averaging out the white noise of an image is an elegant solution.

Part II

This report presents my implementation (My NLM) of Buades et al.’s Non-Local Means (NLM) algorithm. The source code for this implementation can be found in the file `nl_means.m`. The NLM algorithm involves two neighbourhood windows. One stationary ”host” neighbourhood and a convolving ”comparison” neighbourhood. The convolving comparison neighbourhood searches for patches of pixels with similar values to the stationary host neighbourhood. After the comparison window is done searching, the host neighbourhood shifts to the next pixel and the process is repeated. (see Appendix)

To test my implementation of the algorithm I added Gaussian noise with a mean of 0 and a variance of .002 to the black and white Lena image using Matlab’s `imnoise` function. The image is cropped to 100×100 pixels to reduce processing time. For testing purposes, I use the image below in Figure 1



Figure 1: Noisy cropped Lena image.

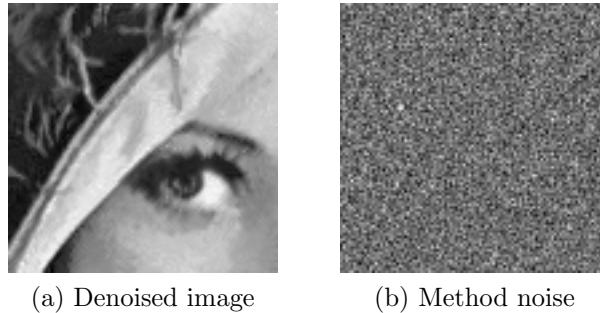


Figure 2: Results from my NLM implementation.

My implementation performs as expected as shown above in Figure 2. Buades et al. [1] states that method noise should not contain any image structures. As seen above in Figure 2, the method noise appears as white noise with no discernable structures from the original image. The above results use a 11×11 search window and a 5×5 neighbourhood window. Using different values for these parameters, it is observed in Figure 3 that the search window size has the biggest impact on method noise quality. As the search window increases in size more of the image structure appears in the method noise (see Figure 3 j, k, l). This is a result of the weighted summation used to determine the denoised pixel value. As the search window’s size increases the weighted summation includes more terms. Also, the weight factor in each term depends on the similarity of the two neighbourhoods. As

a result, dissimilar pixels have less impact on the denoised value. However, if there are a large number of dissimilar terms relative to the number of similar terms then the dissimilar terms have more influence on the denoised value. The result is a blurring effect (see Figure 4).

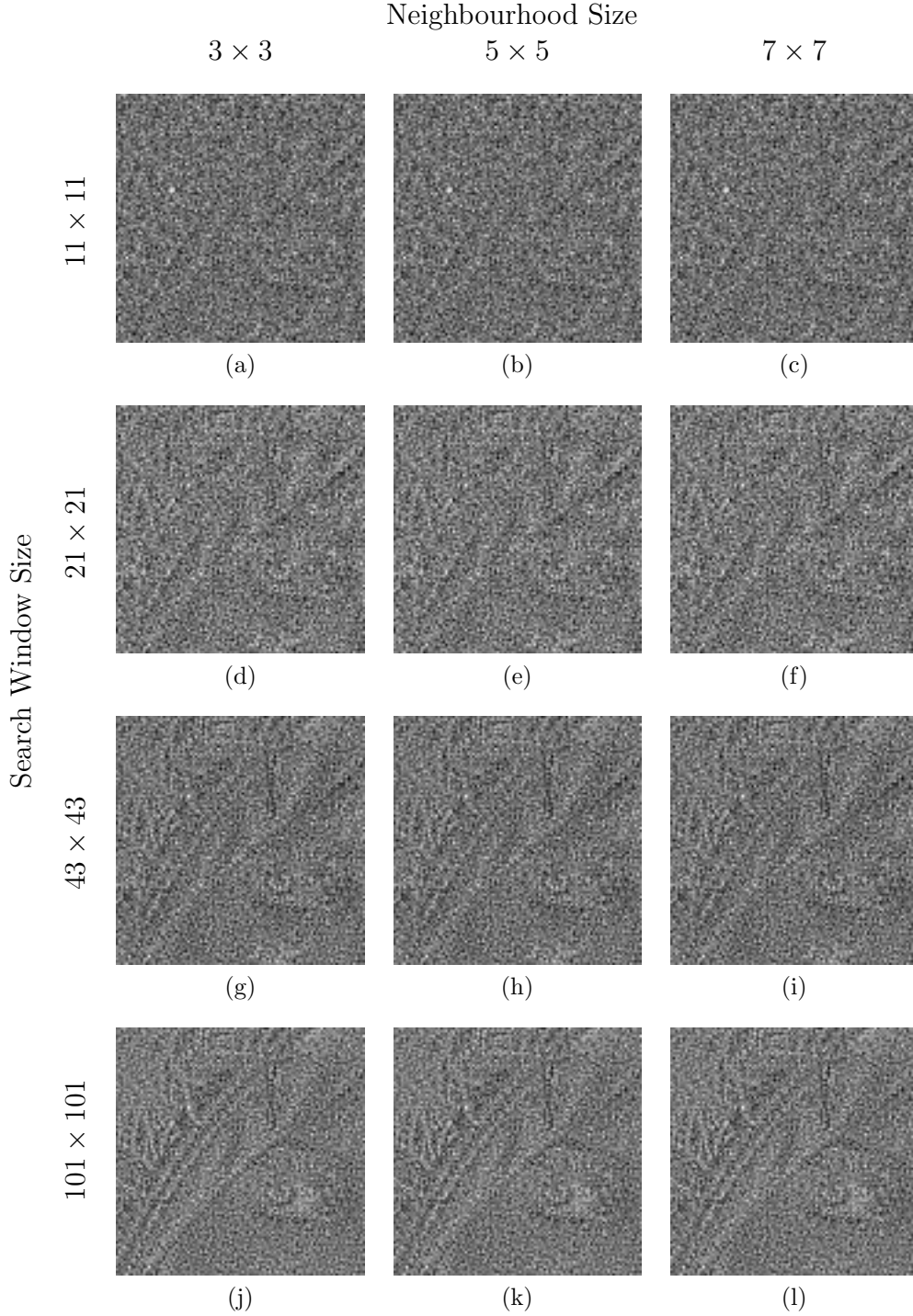


Figure 3: Method noise with different parameters

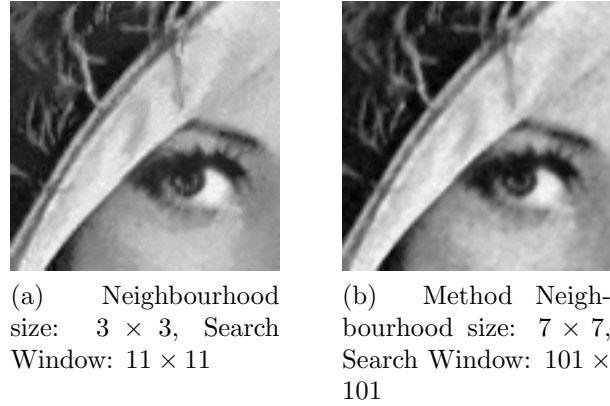


Figure 4: Denoised images

The neighbourhood size has less impact than the search window size on method noise quality due to fast decay of the Gaussian kernel. After a kernel radius of two the values are near zero. See Table 1.

1.6411e-06	1.6411e-06	1.6411e-06	1.6411e-06	1.6411e-06	1.6411e-06	1.6411e-06
1.6411e-06	0.0002	0.0002	0.0002	0.0002	0.0002	1.6411e-06
1.6411e-06	0.0002	0.0051	0.0051	0.0051	0.0002	1.6411e-06
1.6411e-06	0.0002	0.0051	0.0051	0.0051	0.0002	1.6411e-06
1.6411e-06	0.0002	0.0051	0.0051	0.0051	0.0002	1.6411e-06
1.6411e-06	0.0002	0.0002	0.0002	0.0002	0.0002	1.6411e-06
1.6411e-06	1.6411e-06	1.6411e-06	1.6411e-06	1.6411e-06	1.6411e-06	1.6411e-06

Table 1: Example of Gaussian Kernel

Comparison to Other Denoising Algorithms

As in, Buades et al. [1], I compare my results to Matlab's implementation of the Gaussian filter, Anisotropic filter, and Bilateral filter. Matlab does not have a Total Variation filter so I use Magiera and Löndahl's implementation [2].

The Gaussian filter blurs the image because it removes more information from edges than from the flat parts of the image. As a result, some of the image's structural detail is seen in the method noise (Figure 5.a). As seen in Figure 6 my NLM's denoised figure (e) is less blurry than the Gaussian (a). The Anisotropic and Total Variation filters preserves edges better than Gaussian but is still more blurry than my NLM denoised image. The method noise of the Bilateral filter is similar to my NLM in that it does not include structural information (unlike the other filters). However, it leaves significantly more noise in the denoised image than my NLM as seen in Figure 6. This demonstrates the success of my NLM as it aligns very closely to the results obtained by Buades et al.

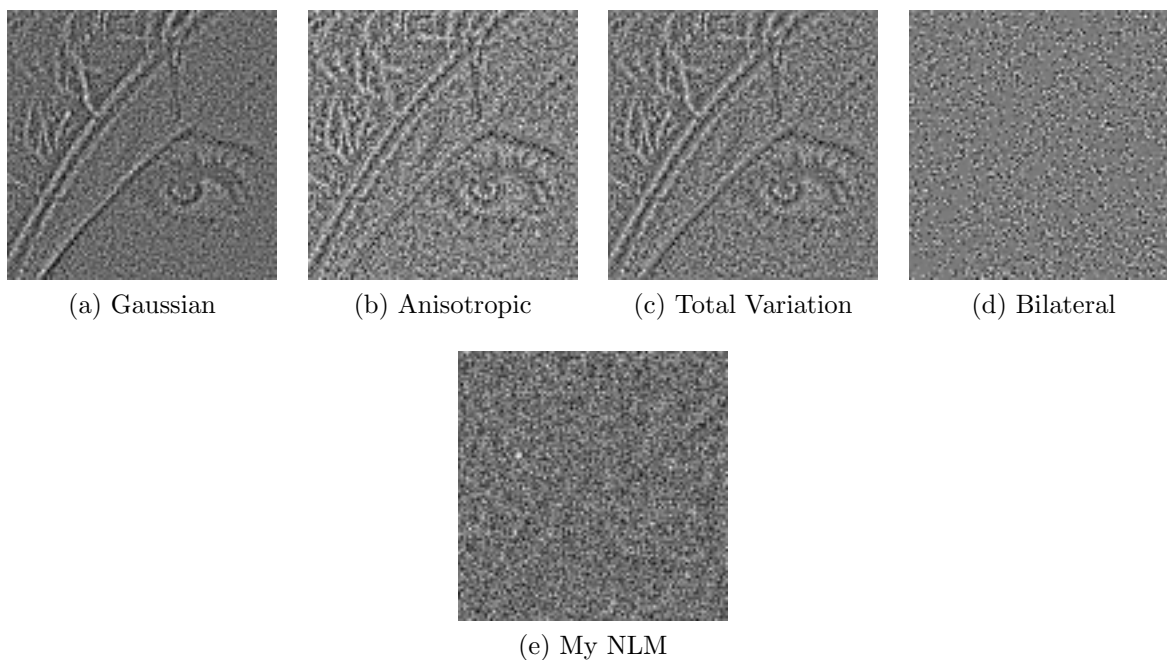


Figure 5: Denoised images from My NLM and other algorithms.

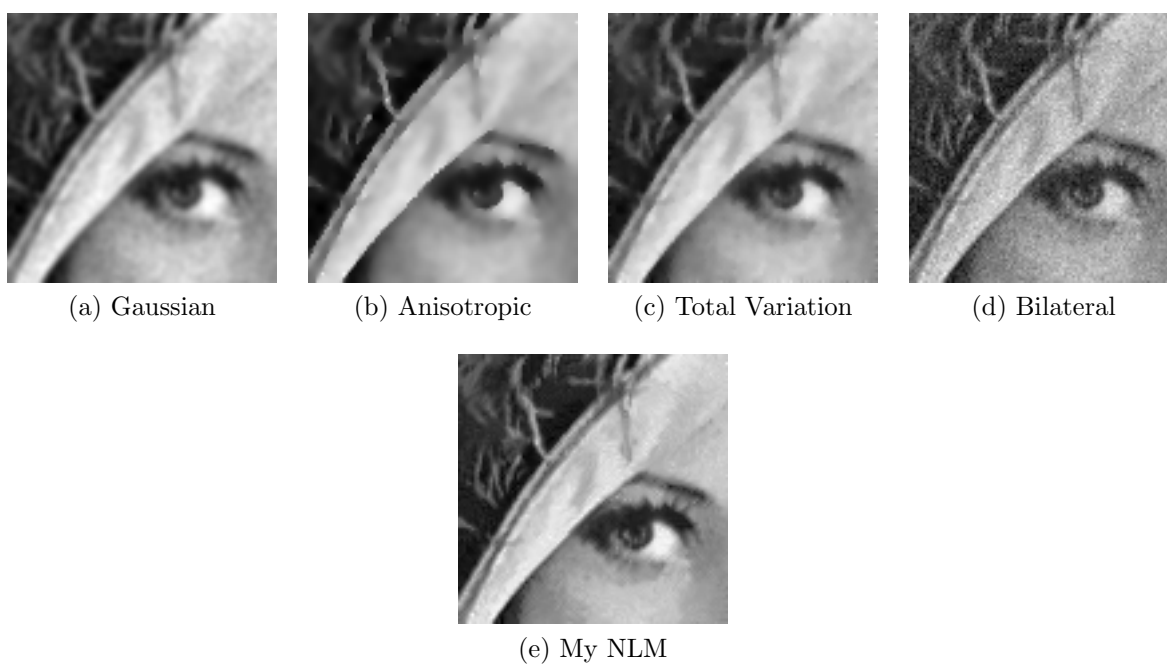


Figure 6: Method Noise from My NLM and other algorithms.

Comparison to Matlab's `imnlmfilt`

Matlab provides an implementation of the Non-Local Means filter. However, it does not use a Gaussian kernel ([3]) as recommended by Buades et al. Instead, Matlab only uses the Euclidean distance between the two neighbourhood windows. Although subtle there is more structure in the method noise which results in a slightly more blurred denoised image (see Figure 7). This is a result of the Gaussian average weighted more towards the value of central pixels in comparison to the Box filter used by Matlab.

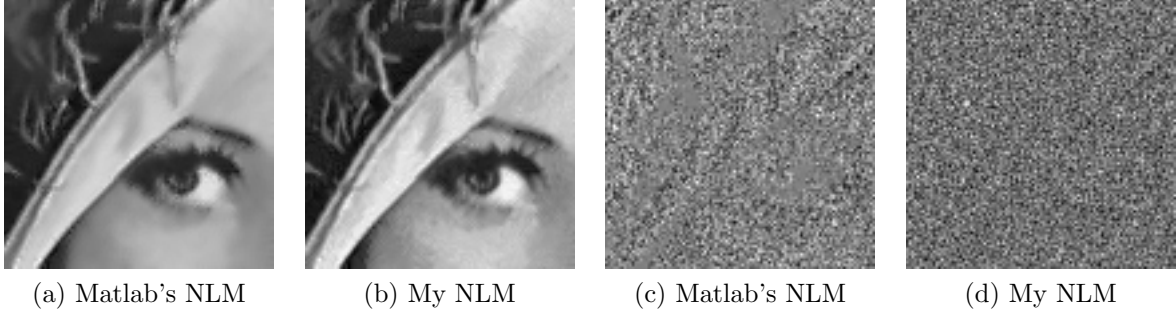


Figure 7: Matlab's NLM vs. My NLM.

Center Pixel Weight Parameter

In the NLM of Buades et al. (Classic NLM) the Center Pixel Weight (CPW) is the weight derived from the host neighbourhood (also known as Unitary CPW [4]). Initially, I implemented My NLM with a Unitary CPW. However, the results I achieved did not match the expected results. Further investigation revealed many different CPW strategies and I chose to implement Max CPW because it is relatively simple. This achieved the desired result.

The importance of CPW is illustrated when examining its role in determining the denoised pixel. Separating the contributions of the center and non-center weights in the denoised pixel, we have,

$$d = \frac{1}{W+v}(Wz + vy)$$

where W is the sum of the non-center weights, v is the center weight, z is the denoised pixel using only non-center weights, and y is the original noisy pixel. Assuming we know d , to compute the ideal v we have,

$$v = \frac{W(d-z)}{y-d}$$

This shows that the CPW, v , is a function of W , y , z . Therefore, CPW functions that do not use all three variables are missing important information. The unitary CPW has a value of 1 and does not use any of these variables. Max CPW uses W and z [4] which may not be ideal but is an improvement over Unitary CPW. This is shown in Figure 8.

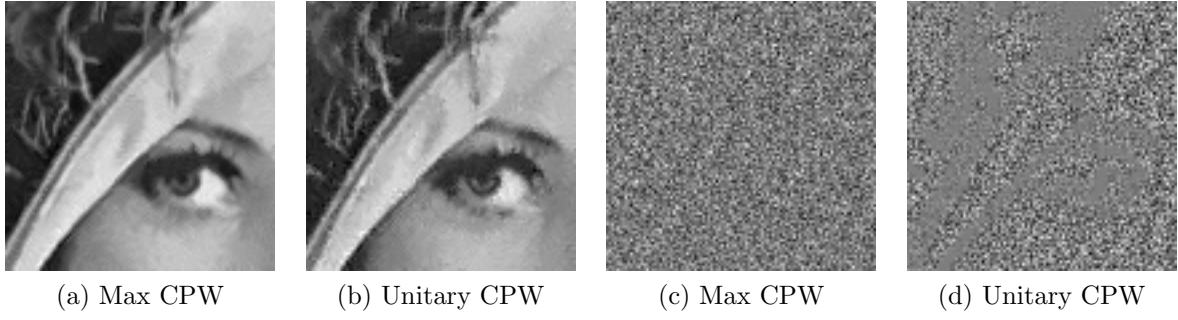


Figure 8: Max CPW vs. Unitary CPW using My NLM

Noise Tests

The NLM algorithm is fundamentally an averaging filter and works well with Gaussian noise. This is seen below in Figure 9 where noise is reduced without losing the structural detail of the image. With that said, Figure 9 shows images that were produced with the same degree of smoothing (DoS=10). Therefore, more noise persists in the denoised images. According to Buades et al. the DoS depends on noise variance. Figure 10 shows results from the same noisy images with increasing DoS. As noise is removed the results are more blurry.

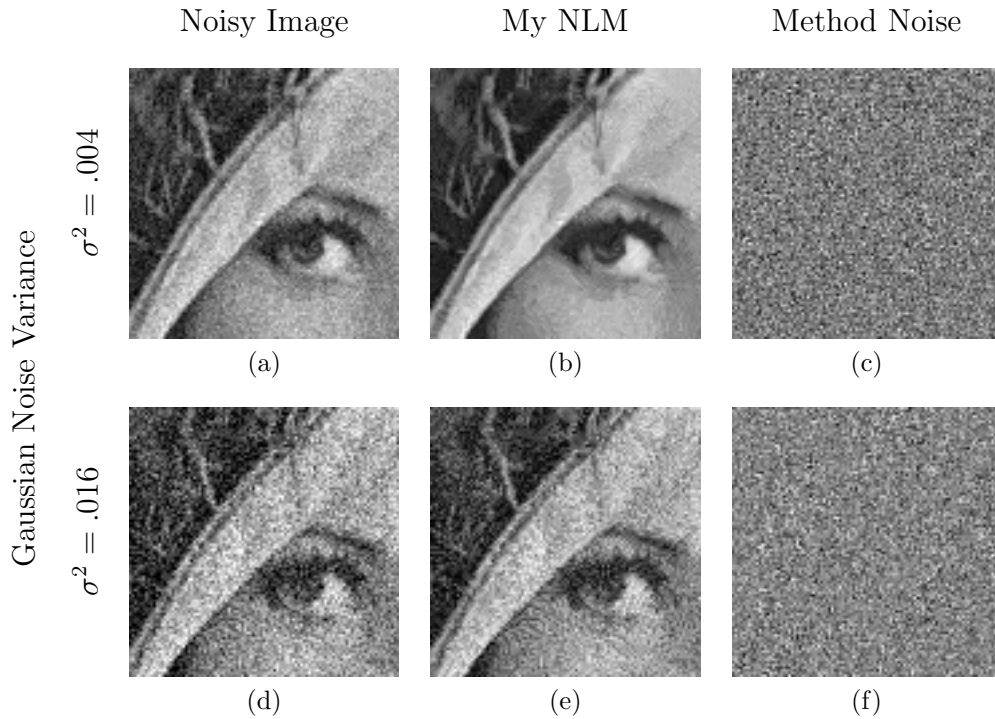


Figure 9: Results with different Gaussian noise variance with DoS = 10

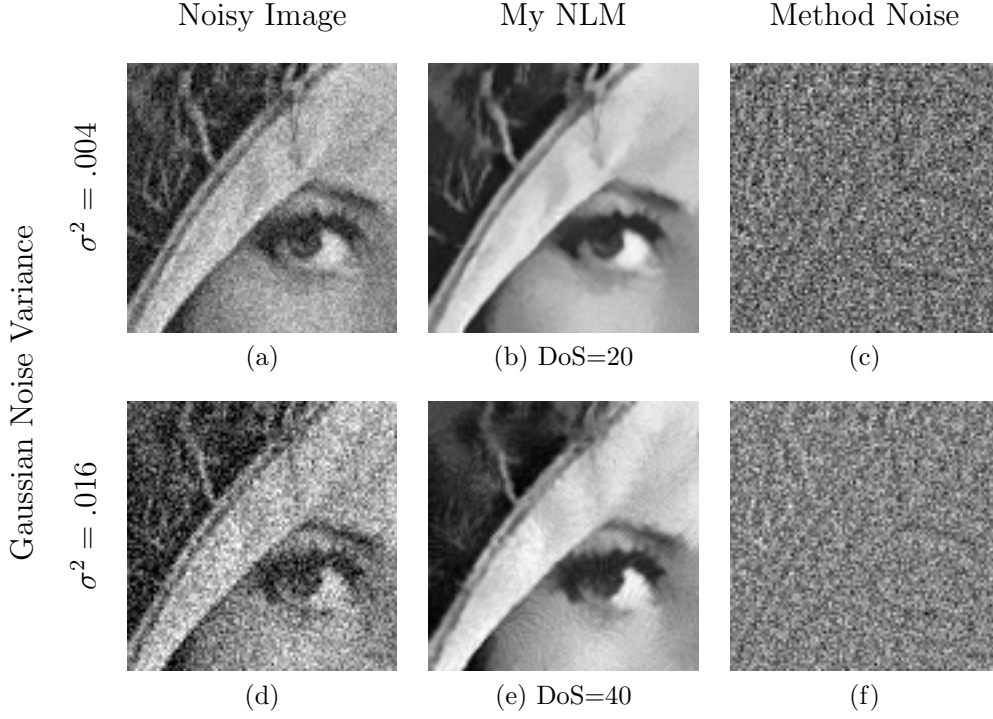


Figure 10: Results with different DoS

NLM does not work with Salt & Pepper as seen in Figure 11. From these images, it is observed that very little of the noise is removed. The method noise is almost a flat gray because the NLM algorithm struggles to find noise and removes a constant value from each pixel.

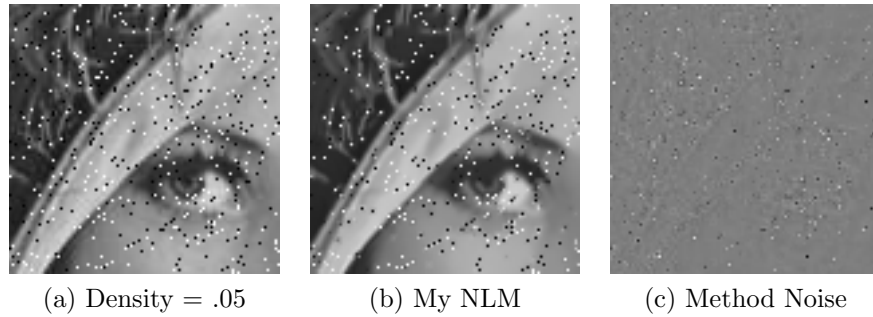


Figure 11: My NLM with Salt & Pepper noise

Reflection on Re-implementation

Buades et al. makes the statement that “as the size of the image grows we can find many similar patches for all the details of the image” [1]. This suggests that the results should improve when using larger search windows. Furthermore, they state that for computational purposes “in all the experimentation we have a fixed search window of 21×21 ” [1]. As a

consequence, they do not show results from a search window size that covers a full image. My tests indicate that as the search window grows beyond a certain size the results worsen. This can be seen above in Figure 3 on page 3 where more structural detail emerges in the method noise as the search window increases. Likewise, Figure 12 below shows the denoised image becoming more noisy as the search window increases.

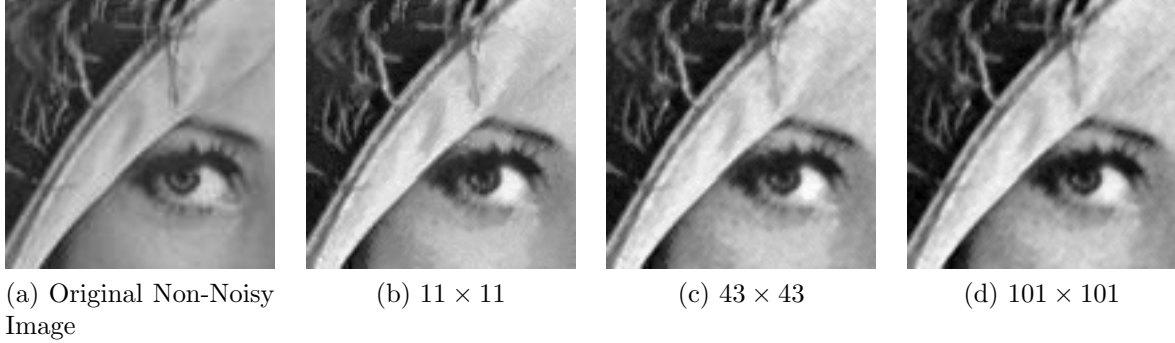


Figure 12: Result of different search window sizes

An initial implementation was not difficult however I found that the results with respect to method noise contained more image structure than expected. Further investigation, made me realize that I had misinterpreted some mathematical notation in the original paper. As a result, I was taking the "difference of the Guassian" as opposed to "the Gaussian of the difference". In other words, I was applying the Gaussian kernel to both neighbourhoods first and then finding the Euclidean distance between them instead of first finding the Euclidean distance between the neighbourhoods and then applying the Gaussian kernel. While this removed the noise of the images it also led to very smooth denoised images with structural artefacts in the method noise.

Additionally, my initial implementation used a Unitary CPW as implied by Buades et al. However, through further investigation, I discovered that CPW is an active area of research. Therefore, I modified my implementation to incorporate Max CPW which significantly improved my method noise. It is unclear to me how Buades et al. achieved method noise that appeared as white noise with no structural artefacts using a Unitary CPW.

Finally, I initially implemented My NLM to search the entire image as recommended by Buades et al. Eventually, I implemented a search window to reduce processing time and discovered that a smaller search window gave better results.

Appendix

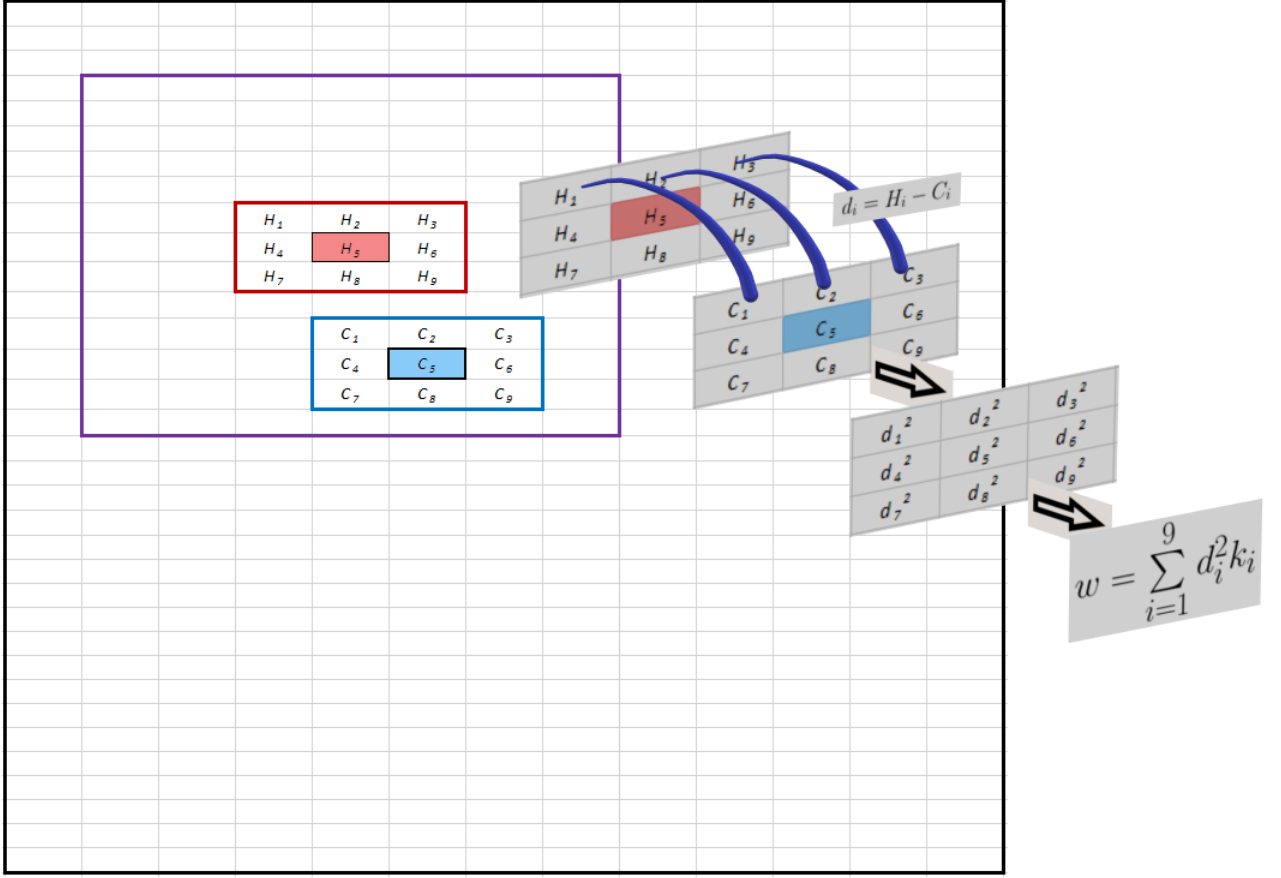


Illustration of how a weight, w , is calculated from a single neighbourhood comparison.

- Red: host neighbourhood
- Blue: comparison neighbourhood
- Purple: search window
- H : pixel values in host neighbourhood
- C : pixel values in comparison neighbourhood
- d^2 : Euclidean distance between the host and comparison pixels
- k : a value in the Gaussian kernel.

References

- [1] A. Buades, B. Coll, and J.-M. Morel, “A non-local algorithm for image processing,” in *2005 Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05)*, 2005, pp. 60–65. 2, 4, 8
- [2] P. Magiera and C. Löndahl, “Rofdenoise,” 2008. [Online]. Available: <https://www.mathworks.com/matlabcentral/fileexchange/22410-rof-denoising-algorithm> 4
- [3] Matlab, “imnlmfilt,” 2021. [Online]. Available: <https://www.mathworks.com/help/images/ref/imnlmfilt.html> 6
- [4] Y. Wu, B. Tracey, P. Natarajan, and J. P. Noonan, “James–stein type center pixel weights for non-local means image denoising,” *IEEE Signal Processing Letters*, vol. 20, no. 4, pp. 411–414, 2013. 6
- [5] M. P. Nguyen and S. Y. Chun, “Center pixel weight estimation for non-local means filtering using local james-stein estimator with bounded self-weights,” in *2016 IEEE 13th International Symposium on Biomedical Imaging (ISBI)*, 2016, pp. 82–85.
- [6] Y. Wu, B. Tracey, and J. P. Noonan, “James-stein type center pixel weights for non-local means image denoising,” *CoRR*, vol. abs/1211.1656, 2012. [Online]. Available: <http://arxiv.org/abs/1211.1656>