Observation Report

Kushagra Sharma
18110091

Methodology

The general idea of the Non-Local Means (NLM) algorithm is to estimate the actual pixel value in the denoised image using a weighted mean of pixel neighborhoods having some similarity in the noisy image. Ideally pixel similarity should be searched in the complete image but for high resolution images, this procedure can take quite some time. So, pixel similarity is searched only in some neighborhood of each pixel, defined by the variable `big_window`. Also, to increase the performance while calculating the similarity of each pixel, instead of considering only one pixel, a small area of pixels is chosen around it, given by the `small_window` parameter.

Now, gaussian and salt & pepper noises are added to the input image using the standard procedures and the noisy image is passed as a parameter to the NLMeans' `solve()` function. Along with the noisy image, the `small_window`, `big_window` and `h` (filtering parameter) are also given. A result image is initialised as a numpy array of zeros having the same dimensions as the input noisy image. To calculate the similarity values by considering a `big_window` * `big_window` neighborhood around each pixel, the noisy image is padded along each of the four boundaries. The padded width is `big_window/2` since the neighborhood is created by considering each pixel in consideration as the center. The mode of the padding is `reflect` to account for better similarity values for the boundary pixels.

Since, we want a `small_window` * `small_window` patch for similarity calculation, they are preprocessed by traversing the padded image and storing a 2D array of size `small_window` * `small_window` for each pixel in the input noisy image. Now, each pixel of the input noisy image is traversed and a `big_window` * `big_window` neighborhood around that pixel is obtained by indexing the previously preprocessed array. So, in each iteration of the loops we get, a 2D `small_window` * `small_window` patch for current

pixel, a 4D `big_window` * `big_window` neighborhood around the current pixel having each entry as a 2D `small_window` * `small_window` patch.

Now, we have all the required elements to calculate the weighted mean for the current pixel. Each pixel is traversed in the calculated 4D `big_window` * `big_window` neighborhood. For each such pixel we get a 2D `small_window` * `small_window` patch and the squared norm of this 2D window and current pixel's 2D window is taken. This norm is multiplied by `-1/(size of neighborhood window)` and its exponential is taken. This value is used to calculate the numerator of the expected NLMeans value for each pixel (as defined in the paper `A non-local algorithm for image denoising`). The denominator of this fraction is calculated by the summation of such exponential terms.

The expected NLMeans value is clipped so that each value is between 0 and 255 and the denoised image is returned.

Metrics

MSE

Salt And Pepper Noise

Image Number	Noisy Image	SkImage	NLMeans	Gaussian Filtering
1	1074.82	86.89	88.15	114.02
2	1019.97	280.56	278.62	139.64
3	1043.67	202.22	201.92	282.50
4	1107.05	114.64	118.01	180.28
5	1371.12	73.41	69.96	142.75
6	1050.92	257.15	261.54	303.39
7	1291.81	195.13	192.68	192.54
8	1409.07	46.38	44.31	135.31

9	1166.67	92.71	90.75	141.92
10	947.91	365.45	364.86	193.54

Gaussian Noise

Image Number	Noisy Image	SkImage	NLMeans	Gaussian Filtering
1	608.35	50.86	51.87	64.45
2	611.45	155.35	155.08	95.18
3	580.48	123.34	122.78	246.66
4	614.96	71.76	74.16	125.94
5	416.75	76.62	73.12	89.58
6	593.36	138.91	139.50	250.72
7	477.23	129.51	126.39	126.87
8	413.01	54.17	52.79	77.99
9	529.50	63.40	61.59	87.24
10	630.72	192.30	194.00	153.18

PSNR

Salt And Pepper Noise

Image Number	Noisy Image	SkImage	NLMeans	Gaussian Filtering
1	17.82	28.74	28.68	27.56
2	18.04	23.65	23.68	26.68
3	17.95	25.07	25.08	23.62

4	17.69	27.54	27.41	25.57
5	16.76	29.47	29.68	26.59
6	17.92	24.03	23.96	23.31
7	17.02	25.23	25.28	25.29
8	16.64	31.47	31.67	26.82
9	17.46	28.46	28.55	26.61
10	18.36	22.50	22.51	25.26

Gaussian Noise

Image Number	Noisy Image	SkImage	NLMeans	Gaussian Filtering
1	20.29	31.07	30.98	30.04
2	20.27	26.22	26.23	28.35
3	20.49	27.22	27.24	24.21
4	20.24	29.57	29.43	27.13
5	21.93	29.29	29.49	28.61
6	20.40	26.70	26.69	24.14
7	21.34	27.01	27.11	27.10
8	21.97	30.79	30.91	29.21
9	20.89	30.11	30.24	28.72
10	20.13	25.29	25.25	26.28

Parameters

Sigma_h:

The `sigma_h` parameter is the filtering parameter

which determines the impact of each exponential value in the calculations, i.e. the impact of the similarities as a function of distance. It is used to control the weight of the patch difference values. Optimising `sigma_h` value can lead to significant improvements in the quality of the denoised image produced.

Small_window:

The `small_window` parameter is used to create a small patch around each pixel to help find the similarity score between the pixels. To calculate the weighted value for a pixel, we search for similar pixels, so as to find the expectation of the real value of the pixel without noise. This can be done using the pixel values at each pixel but to increase the accuracy of the expected value, a small patch is created around each pixel and its value if taken to be the weighted average inside this small patch. This is so that a better similarity score can be calculated and hence the resulting image.

Big_window:

The `big_window` parameter is used to create a neighborhood of pixels around each pixel to search for similar values. Ideally, to find the expected value of each pixel in the denoised image, the weighted average for all pixels in the image can be calculated. But this approach is not suitable for larger sized images and suffers from large computation times. Thus, to optimise the calculation a neighborhood is created within which the weighted average of similarity values is taken to calculate the expected value.

Observations

The Non Local Means algorithm works better than the Gaussian Filtering algorithm for all of the images for both the salt and pepper noise and gaussian noise. For the taken values of `salt_and_paper_h` and `gaussian_h`, the Non Local Means algorithm produces images with lower MSE and higher PSNR than the Gaussian Filtering algorithm. The only exceptions being image 2 and 10 in which the MSE and PSNR of the Gaussian

Filtering are quantitatively better but the metrics and visual quality of the image produced by Non Local Means and Gaussian Filtering are comparable.

The stated deviation for both the images may be because for the values of `h` parameter taken the Non Local Means algorithm cannot find much similarity in a neighborhood defined by `big_window`. This can be rectified by tuning the parameters for the specific image to produce better results or performing a full search of the image for similarity values instead of just a neighborhood defined by `big_window`.

References

Buades, A., Coll, B., & Morel, J. M. (2005). A non-local algorithm for image denoising. *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, *2*, 60-65. doi: 10.1109/CVPR.2005.38.

Ingraham, K. (2017, February 4). *Salt & Pepper Noise and Median Filters, Part II*. blog.kyleingraham.com. Retrieved August 18, 2021, from https://blog.kyleingraham.com/2017/02/04/salt-pepper-noise-and-median-filters-part-ii-the-code/

Mahmoudi, M., & Sapiro, G. (2005, December 12). Fast Image and Video Denoising via Nonlocal Means of Similar Neighborhoods. *IEEE Signal Processing Letters*, *12*(12), 839-842. doi:10.1109/lsp.2005.859509

Palou, G. (2015, July 7). *An approach to Non-Local-Means denoising*. http://dsvision.github.io/. http://dsvision.github.io/an-approach-to-non-local-means-denoising.html