Pattern matching based denoising for images with repeated sub structures.

Alice Author[1,*], Bob Author[2], Christine Author[1,2,+], and Derek Author[2,+]

[1]Affiliation, department, city, postcode, country
[2]Affiliation, department, city, postcode, country
[*]corresponding.author@email.example
[+]these authors contributed equally to this work

ABSTRACT

In electron microscopy, obtaining a noise-free image is often difficult especially when examining biological samples or delicate materials. Therefore, the suppression of noise is essential for the analysis of such noisy images. State of the art image denoising methods are dominated by supervised Convolution Neural Network (CNN) based methods. However, if a noise-free ground truth is unavailable, it is not possible to use supervised CNNs. To address this problem, a denoising algorithm is proposed that uses re-occurring patterns in images. The proposed method does not require noise-free images for the denoising task. Instead, it is based on the idea that averaging images with the same signal having independent noise, suppresses the overall noise. In order to evaluate the performance of our method, our results are compared with other state of the art denoising methods that do not require a noise free image. Additionally, a confidence map is developed for evaluating the denoising quality of the proposed method. Furthermore, to ensure scalability, time complexity of the algorithm is analyzed and optimizations are made to improve the run-time efficiency.

Introduction

Transmission Electron Microscopy (TEM) imaging is essential in solving numerous scientific questions in life and material sciences[1,2] . However, noise in the acquired images corrupt the signal beyond a useful level. This noise can appear due to various reasons like data transmission errors, properties of the imaging systems, environmental conditions like humidity, temperature, etc. Thus, image denoising plays a vital role especially in suppressing microscopy image noise.

The contrast in TEM imaging is based on the interaction of a multi keV electron beam with the specimen. While the optical resolution for modern TEM system can be below one angstrom, the high energy electrons often lead to a fast degradation of the sample. For such samples only few electrons can be used for imaging. This leads to a degradation of the image's effective resolution, thus making the image hard to interpret. Therefore, after image acquisition, numerical processing is required to enhance the image quality.

In order to get the best image quality with minimum dose, various image denoising algorithms were proposed in the past. Conventional denoising methods like Non-Local Means[3], BM3D[4], etc. mostly use a statistical approach for the denoising task, whereas most modern methods involving a deep neural network[5,6], require ground truth data for training. Since in most cases clean electron microscopy images are not available, modern denoising methods that require clean images as ground truths cannot be used. However, there are also some deep neural network based methods that use noisy supervision[7] and self supervision[8] . Although these existing methods can denoise images, they improved the denoising quality only by a relatively small margin for images with repeated patterns. To address this issue and fill this research gap, a new denoising algorithm which effectively denoises TEM images with repeated patterns is proposed. In our method, we also show that the fine structures in TEM images can be restored most effectively while maintaining image sharpness. In the following sections, the proposed method is explained in detail, results are analyzed and compared with the state-of-the-art methods showing significant gain in image quality.

Method

The proposed denoising algorithm identifies similar patches within the entire image and averages them to suppress the noise. Since the noise is assumed to be randomly distributed with zero mean, it partially cancels out when multiple patches are averaged. However, the base signal remains the same throughout and averaging does not change the signal. Hence combining different patches result in a denoised image, closer to the actual signal value.

Let x_i be the i^{th} noisy patch, k be the number of patches that are averaged, s_i, and n_i be the signal and noise in the i^{th} patch

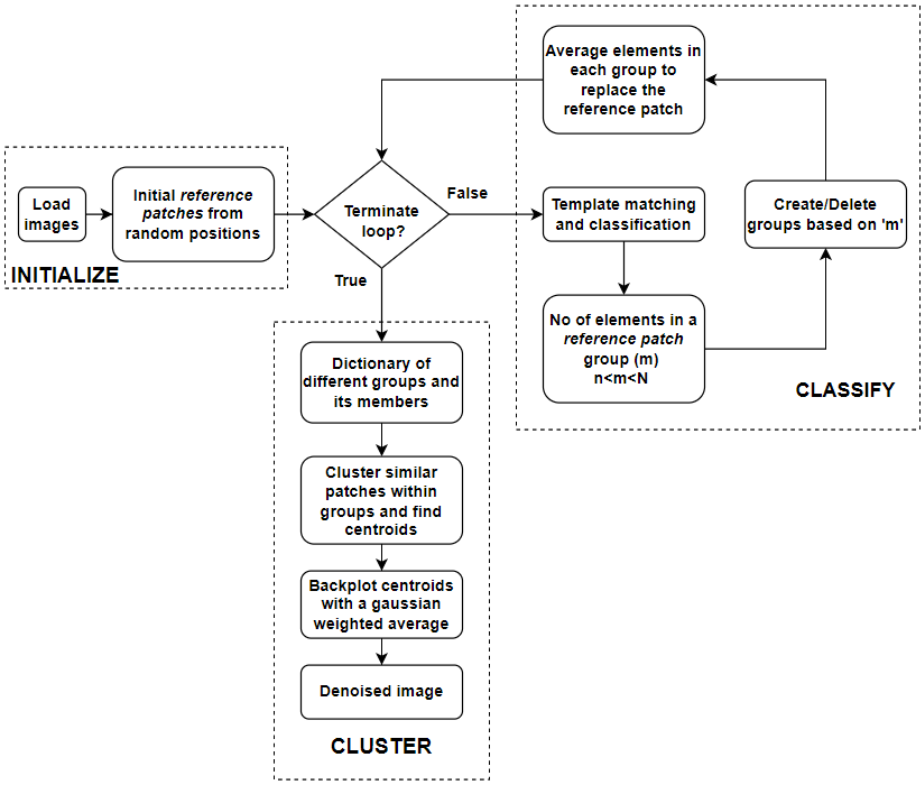

Figure 1. Flowchart of the algorithm

respectively. Then,

$$x_i = s_i + n_i \tag{1}$$

Averaging over k patches results in an expected value,

$$E[\frac{1}{k}\sum_{i}^{k} x_i] = E[\frac{1}{k}\sum_{i}^{k}(s_i + n_i)] \tag{2}$$

Since the noise is expected to have zero mean and the base signal is expected to be the same, this ideally means that,

$$E[\frac{1}{k}\sum_{i}^{k} x_i] = s \tag{3}$$

In addition, variance is

$$Var[\frac{1}{k}\sum_{i}^{k} x_i] = \frac{1}{k^2 - 1}\sum_{i}^{k} Var(n_i) \tag{4}$$

since $Var(s_i) = 0$ for all i. Hence averaging patches with the same signal suppresses noise.

Outline of the proposed algorithm

The proposed algorithm groups similar patches at two levels. Cosine similarity is used to broadly group similar patches within an image and later, clustering is used to more finely group closely matching patches within the groups obtained during the first step. A flowchart of the algorithm is shown in figure 1 and the two levels of the algorithm are represented by 'classify' and 'cluster' sections of the flowchart.

Cosine similarity measures similarities between two vectors[9] by finding the cosine angle between them. If the vectors are in the same direction (i.e., similar), cosine similarity is maximum. It is mathematically represented as,

$$similarity = cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}} \tag{5}$$

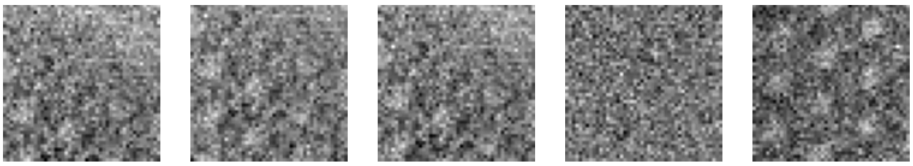

Figure 2. Example set of initial *reference patches* (of size $48 * 48$ pixels), taken from different regions of the input image. These are the first set of patches that are used as templates for pattern matching in the proposed algorithm. These raw image patches exhibit significant signal corruption, which is readily apparent.

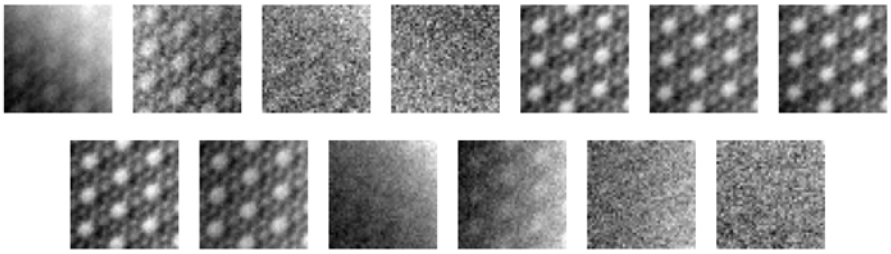

Figure 3. Example subset of the final *reference patches* (of size $48 * 48$). These are obtained from iteratively matching *reference patches* with patches from all positions of the image, grouping similar patches, splitting the group based on the size and averaging the members within a group. These are also the patches obtained after the 'classify' section of the flowchart shown in figure 1.

where A and B are two vectors, and θ is the angle between them. Cosine similarity between a template and different patches of an image results in an array whose values lie between -1 and 1. Values close to the maximum represent the patches similar to the template. Hence, cosine similarity can be used to match two image patches by considering them as vectors.

The algorithm begins with the initialization of random patches of size $m * m$, which are used for matching other patches of the same size in the image. The patches that are used as templates for matching are referred to as *reference patches*. One example for the initial choice of *reference patches* can be seen in figure 2.

In the 'classify' section shown in figure 1, the patches at every position in the image are classified into different groups based on their similarity to the *reference patch* using cosine similarity. When cosine similarity is applied, each patch of size $m * m$ in the image is compared with all the *reference patches*. Local maxima is found from each cosine similarity result, which corresponds to the best fitting positions for that particular *reference patch*. These resulting arrays from different *reference patches* are stacked together and the maxima along the new dimension corresponds to the best fitting *reference patch* for different locations of the image. Now, that the patches in the image which are most similar to the *reference patches* are identified, they can be grouped together.

Some of these groups formed can sometimes have large number of patches in the same group, while others may contain very few unique patches. Groups with few members contribute hardly to any denoising, while overly large groups might lead to a loss of detail. Hence, the groups formed are deleted or split into finer groups based on the group size. Finally for each new group, the old *reference patch* is replaced by the average of all the members in that group. In the next iteration, cosine similarity and the classification steps repeat with these new *reference patches*. When the classification becomes stable, there are no new groups formed. This is when the iteration loop is broken. Figure 3 shows the *reference patches* generated after 15 iterations. On comparing figure 2 and figure 3, one can observe that the noise in the final *reference patches* has been significantly suppressed.

If the final *reference patches* are directly used for back plotting (i.e. to replace the patches in their corresponding groups), there would be still some artifacts present due of the following reasons.

- Sometimes image patches that are not similar to any of the *reference patches* end up falling into the best available group, even though the group might not be their best representation. This is required since the entire region of the image has to be covered. When outliers are included during averaging, the mean deviates from the median signal value. Additionally, the unique features present as outliers will be lost in the averaging process, both of which are undesirable.
- Cosine similarity is only sensitive to the structure for any two patches and ignores the offset (i.e. brightness). Therefore, back plotting might not recover local brightness variations.

These problems can be solved by averaging over a small group with very closely matched patches. To achieve this, clustering

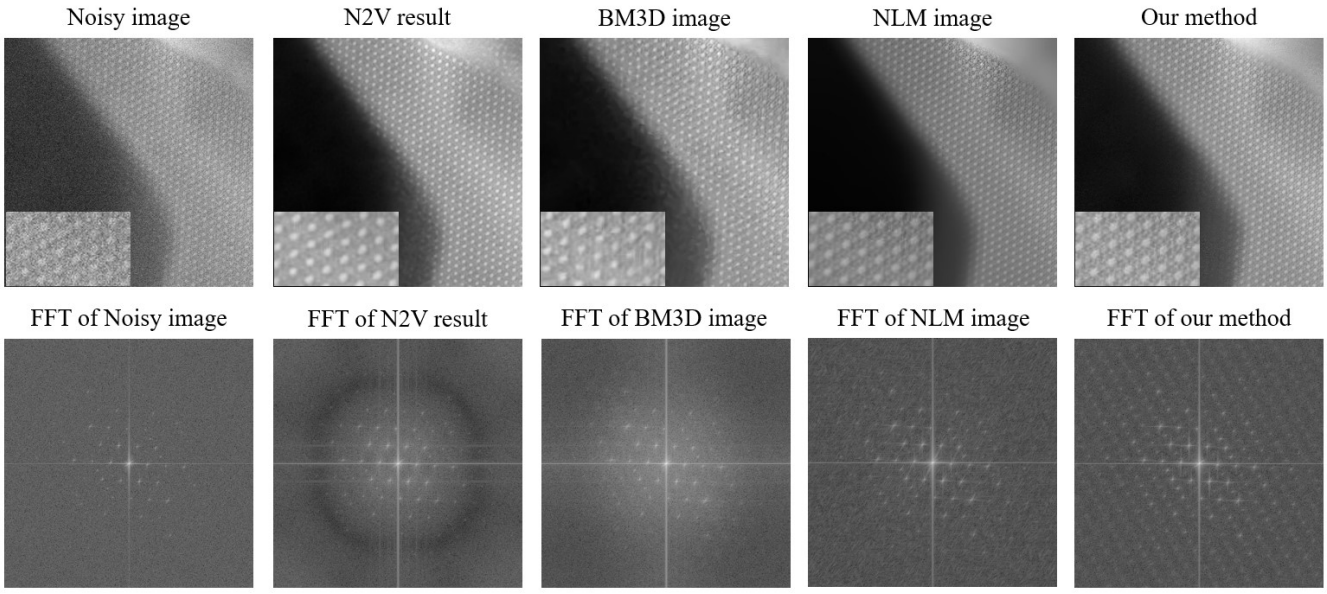

Figure 4. First row shows the noisy image and results obtained from N2V, BM3D, NLM and our proposed method. Corresponding FFTs can be seen in the second row. Comparison of results obtained from different methods. In the first row, insets show a zoomed-in region of the image. These insets are taken from the same location in all images and help in interpreting the denoised results.

is applied within every group (represented by the final *reference patch*) to create smaller subgroups. The number of clusters in a group can be adjusted by a user set parameter. In other words, the target signal-to-noise ratio can be adjusted by changing this parameter value. While previously the whole group was represented by a single *reference patch*, it is now represented by centroids of the subgroups after clustering. Centroids are back plotted with a $2D$ Gaussian weighted average. These Gaussian weights smoothen the edges of the centroids, thus preventing artifacts in the reconstructed image.

A pseudo implementation of the algorithm is shown in the additional information section 1. The implementation of the denoising algorithm can be found on github (`https://github.com/mbanil/img-denoiser`).

Parameters of the algorithm and stability

For optimal performance, the algorithm requires a few parameters which the users can tune. The parameters include size of the features defined by patch size, position of the initial patches and depending on the amount of denoising desired, the upper and lower limit of the group size for cosine similarity classification. Finally, the group size for clustering can also be adjusted, which closely defines the desired signal to noise enhancement. If the average number of elements in a subgroup is N^2, there is an improvement in the signal quality by a factor of N times[3].

Results

The proposed algorithm is mainly developed to denoise TEM images with repeated structures. In figure 4 the noisy image, the results from all the methods and their Fast Fourier transforms (FFT) can be seen. The FFT converts data from the spatial domain to the frequency domain. The signal corresponding to the low frequency components are represented at the center of the FFT and higher frequency components are present as we move away from the center. Noise corresponds to the unstructured background, as seen in the high frequency region of the FFT, and is present close to the edges.

The following can be concluded from the figure 4.

- The noisy image contains a lot of grainy structures which makes it hard to interpret the information. This is also reflected in the FFT, where clear structures are only visible close to the center.
- N2V[8] is a self supervised, deep learning based image denoising method. N2V was trained with the patches of the noisy image. The training was done with $64 * 64$ patches, 100 epochs and a neighboring radius of 5.

  The results from N2V show a decently reconstructed circular structures and the noise in the black region of the image has been removed fairly well. However, the substructures have not been reconstructed well. The FFT shows enhanced structures at the center, whereas the boundary mostly looks dark representing the suppression of high frequency noise.

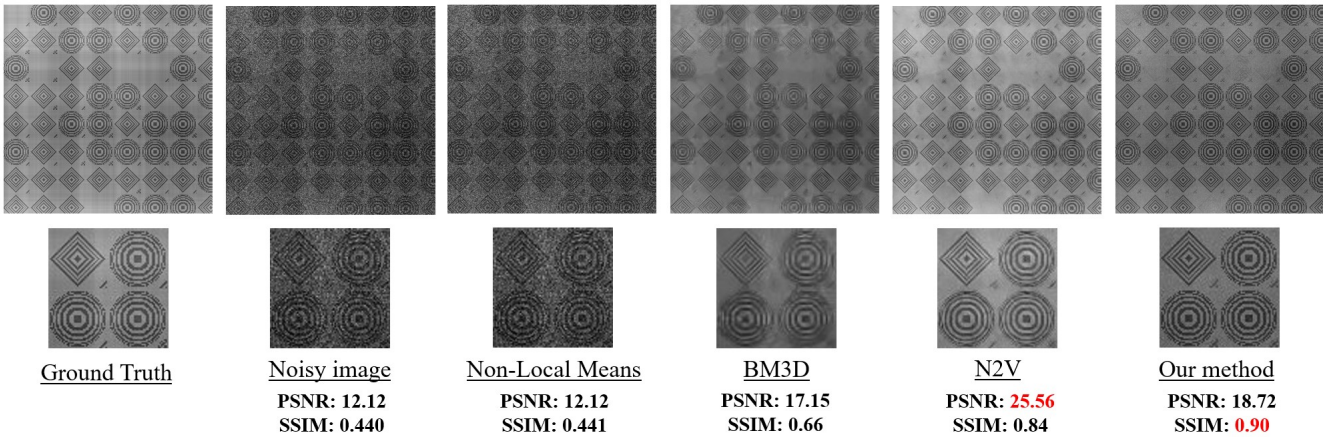

Figure 5. Comparison of results obtained from different methods

- BM3D[4] is one of the widely used classical denoising methods. BM3D uses collaborative filtering in the transform domain for denoising images and it is a non blind denoising method, which means that the standard deviation of the noise is required for denoising. The standard deviation was estimated with trial and error. The best results we obtained for a standard deviation of 0.06 for the normalized image.

  The results from BM3D are similar to that obtained from N2V. The denoising effect is visible but the images are still not very useful for further analysis. This is also supported by the FFT.

- Non Local Means (NLM)[3] is a conventional image denoising method that finds similar patches of images within a region and averages them to suppress noise. An NLM implementation with a patch size of $48*48$, a search area of $100*100$ and a cut off distance of 0.36 was used.

  The result shows a good level of denoising. The circular structures and sub structures between them are visible fairly well. At most regions, the the level of noise suppression is good. But at some regions, the existence of noise can still be seen. The FFT also shows stronger structures supporting the indicating the enhancement of image features. Overall, the results look good and more interpretable.

- Results of our proposed denoising algorithm were obtained with a patch size of $48*48$, a group size was between 5 and 100, and the clustering parameter equal to 2.7 were used.

  From the denoised result, it can be observed that the quality of denoising is marginally better than that from NLM. The image noise levels are suppressed fairly well and the information from the image can be well interpreted. The sub structures between the circular structures are also better visible. From the FFT too, it can be seen that the structures are more prominently visible. Some features can also be seen in the high frequency region which was previously not visible so well.

Apart from the qualitative improvement in the results, the proposed algorithm has a bigger advantage with regards to the computational time. The time complexity of the convolution operation is $O(m^2n^2)$, where $m*m$ is the size of the patch and $n*n$ is the size of the image, which is worse than the runtime of the template matching algorithm when m is large. Complexity of the template matching algorithm is $O(n^2log(n^2))$[10], where $n*n$ is the image size. Since the convolution operation is widely used in convolution neural networks, there are Python libraries that support GPU computation for performing the convolution operation. Running the computations on the GPU makes the algorithm significantly faster.

The runtime of the proposed method for the images in figure 4 was 19.4 seconds, where as NLM, which was the closest to our results had a runtime of 171.4 seconds. The optimization in the runtime is particularly helpful when denoising image stacks from Transmission Electron Microscopy. The images in the stacks are often similar. In such cases, our algorithm not only matches the templates in the current image, but also the patches from other images in the stack. This significantly improves the quality of the results and with GPU computations, the computation speed is quite fast. For reference, the denoising of an image stack with 10 images of size 1024*1024 pixels took 281.8 seconds. The computations were carried out on a computer with 128 GB of RAM, Intel Xenon processor (16 CPUs), and Nvidia RTX6000 GPU.

Comparison with sample images

Since obtaining a noise free image is often very difficult when it comes to microscopy, we generated a noise free image (ground truth) artificially for quantitative comparison. The generated image tries to imitate the kind of microscopy images which are best suited for our algorithm's application, i.e. images with similar patterns spread across them. A noisy image is simulated by adding Poisson noise to the generated image. Different denoising methods were applied on this noisy image and the Peak Signal to Noise Ratio (PSNR) and Structural Similarity Index Metric (SSIM) values of the denoised images have been found with respect to the ground truth. The ground truth, noisy image and the results from different methods can be seen in the figure 5. Additionally, a comparison of FFT of these images is shown in section .

The results from Non-Local Means show minimum improvement. The reason is, this method requires similar image patches that are present close to each other, which is not always the case in the generated image. BM3D processes the image in the Fourier domain and hence suppressing the high frequency components. This results in the sharp features of the image being less prominent. N2V does a better denoising job which is reflected in it's PSNR value. However, on close inspection it can be observed that the high frequency components appear slightly blurred. This is where our method out performs the others. The pattern matching used to find similar pattern across the image ensures the correctness while preserving the sharpness of the image. This is also reflected in a higher SSIM value.

It should be noted that PSNR is based on mean squared error and is a distortion based evaluation metric. In image restoration there is always a trade-off between the distortion and the perceptual quality of the restored image. Often it is not possible to achieve both simultaneously[11]. In the figure 5, for our method even though we do not achieve the best PSNR value, we obtain the best results with respect to SSIM which is a more perceptual metric[11].

Confidence map

In the absence of a ground truth, it is difficult to identify artifacts in the denoised results. However, it is essential to recognize these artifacts to prevent wrong deductions from the denoised images. Since it is challenging to detect minute artifacts from FFT, a confidence map is developed. This confidence map is based on the variance within each cluster obtained after applying clustering. The idea behind this approach is that the variance should be small if the centroid is a good representation of its members. Also, a good centroid should not have any patterns in its variance. Patterns in the variance show that the centroid does not generalize it's members well.

The confidence map of the denoised image is calculated by combining the variances[12] for different centroids as they are back plotted. The overall variance map, i.e., the confidence map of a denoised image is as shown in figure 6. This result has been produced for demonstration purpose with a very few initial *reference patches* selected very close to each other. The bright regions in the confidence map correspond to the denoised image artifacts. For instance, a bright region can be seen in the confidence map at the top-right position. An artifact can be found by inspecting the same region on the denoised image. Similarly, irregularities in the black region on the left side of the denoised image can be recognized by the brighter regions of the confidence map.

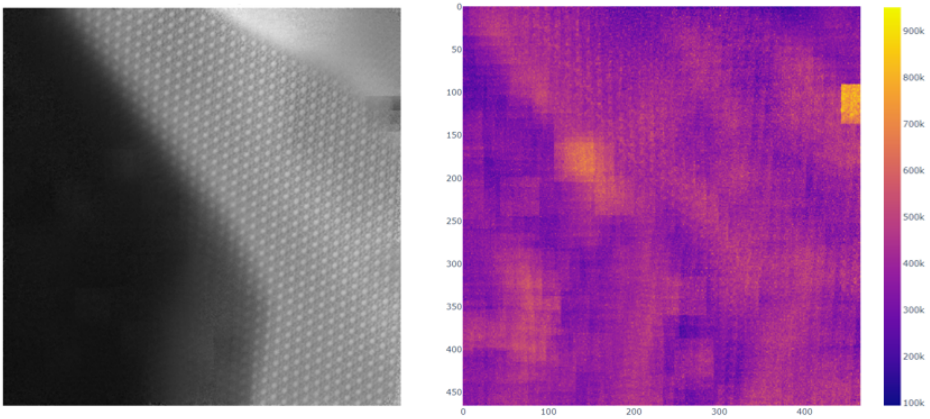

Figure 6. Comparison of results obtained from different methods

Discussion

Denoising of the experimental images obtained from electron microscopy was performed by using popular denoising methods - N2V, BM3D and NLM. Even though, these methods successfully suppressed noise, they enhanced fine image features only

by a small margin. To address this problem, a new denoising algorithm was developed which makes use of similar patterns present in images and averages them to suppress noise. This new method successfully denoises images and also enhances fine structures. The results of all the denoising methods are compared using FFT. Additionally, a confidence map is developed to evaluate the denoising results when ground truth data is absent. Quantitative comparison of the denoising results is done using an artificially generated image. The quantitative comparison demonstrates the ability of our proposed method to preserve high frequency components in images.

The time complexity of the algorithm was analyzed. Optimizations were made to enhance the computation speed by introducing cosine similarity instead of template matching. This helped in enhancing the run-time efficiency by a significant factor, which ensures scalability.

References

1. Curry, A., Appleton, H. & Dowsett, B. Application of transmission electron microscopy to the clinical study of viral and bacterial infections: Present and future. *Micron* **37**, 91–106, DOI: https://doi.org/10.1016/j.micron.2005.10.001 (2006).

2. Wang, Z. L. & Lee, J. L. Chapter 9 - electron microscopy techniques for imaging and analysis of nanoparticles. In Kohli, R. & Mittal, K. (eds.) *Developments in Surface Contamination and Cleaning (Second Edition)*, 395–443, DOI: https://doi.org/10.1016/B978-0-323-29960-2.00009-5 (William Andrew Publishing, Oxford, 2008), second edition edn.

3. Buades, A., Coll, B. & Morel, J.-M. Non-Local Means Denoising. *Image Processing On Line* **1**, 208–212 (2011).

4. Mäkinen, Y., Azzari, L. & Foi, A. Collaborative filtering of correlated noise: Exact transform-domain variance for improved shrinkage and patch matching. *IEEE Trans. Image Process.* **29**, 8339–8354, DOI: 10.1109/TIP.2020.3014721 (2020).

5. Zhang, K., Zuo, W. & Zhang, L. Ffdnet: Toward a fast and flexible solution for cnn-based image denoising. *IEEE Transactions on Image Process.* **27**, 4608–4622 (2018).

6. Zhang, K., Zuo, W., Chen, Y., Meng, D. & Zhang, L. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE transactions on image processing* **26**, 3142–3155 (2017).

7. Lehtinen, J. *et al.* Noise2noise: Learning image restoration without clean data. *CoRR* **abs/1803.04189** (2018). 1803.04189.

8. Krull, A., Buchholz, T.-O. & Jug, F. Noise2void-learning denoising from single noisy images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2129–2137 (2019).

9. Alake, R. Understanding Cosine Similarity And Its Application (2021).

10. Lewis, J. Fast normalized cross-correlation. *Ind. Light. Magic* **10** (2001).

11. Blau, Y. & Michaeli, T. The perception-distortion tradeoff. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6228–6237, DOI: 10.1109/CVPR.2018.00652 (2018).

12. Chan, T. F., Golub, G. H. & LeVeque, R. J. Updating formulae and a pairwise algorithm for computing sample variances. In *COMPSTAT 1982 5th Symposium held at Toulouse 1982*, 30–41 (Springer, 1982).

Acknowledgements (not compulsory)

Acknowledgements should be brief, and should not include thanks to anonymous referees and editors, or effusive comments. Grant or contribution numbers may be acknowledged.

Author contributions statement

Must include all authors, identified by initials, for example: A.A. conceived the experiment(s), A.A. and B.A. conducted the experiment(s), C.A. and D.A. analysed the results. All authors reviewed the manuscript.

Additional information

FFT comparison of the sample images

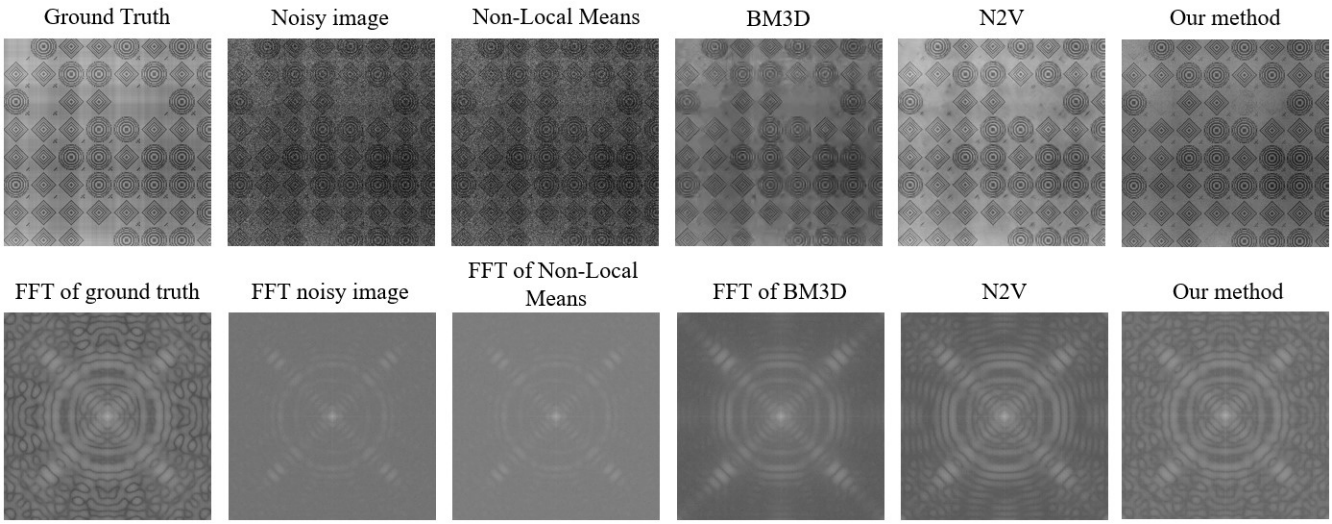

Figure 7. Comparison of FFTs of images obtained from different methods

In addition to the quantitative comparison of different denoising algorithms for the artificially generated image, an FFT comparison is shown in figure 7. The fine details are not visible on the FFTs for the images denoised using Non-Local Means and BM3D. The FFT of these images are closer to the FFT of noisy images with small improvements. The FFT of the image denoised with N2V shows finer details. However, the FFT of the image denoised with our method shows the most details compared to the others. The FFT from our method is also the one which is the closest to that of the ground truth. This clearly shows that our method, outperforms the other methods in image restoration while preserving the fine details of the image.

Algorithm

Algorithm 1 Denoising Algorithm

Input: Noisy image (*imgs*)
Result: Denoised image
set [*templateSize, minGroupSize, maxGroupSize, clustParam, loopTermination*] ;
refPatches ← generate_initial_ref_patches(*imgs*, *templateSize*) ;
for *loopTermination* ≥ 0 **or** (check if the number of *refPatches* in the last two iterations has changed) **do**
    tmpMatchResult ← [ ] ;
    for *img* in *imgs* **do**
        for *refPatch* in *refPatches* **do**
            tmpMatchResult.append(template_matching(*img*, *refPatch*)) ;
        end for
    end for
    maxValue ← max(*tmpMatchResult*, axis=2) ;
    refPatchIndex ← '*refPatches*' index at *maxValue* ;
    [*sorted_maxValue, positionX, positionY*] ← sortWithIdx(*maxValue*);
    t = *templateSize* ;
    list[idx,posX,posY] = [ ];
    for each *s,posX,posY* in *sorted_maxValue, positionX, positionY* **do**
        if *maxValue[posX,posY]* **not equal** 0 **then**
            maxValue[*posX: posX*+(*t*/4); *posY: posY*+(*t*/4)] ← 0 ;
            idx ← *refPatchIndex*[*posX*,*posY*] ;
            list.append([*idx*,*posX*,*posY*]) ;
        end if
    end for
    [*count,refPatchID*] ← count_patches_with_same_id(*list*);
    for *cnt,ID* in *count,refPatchID* **do**
        if *cnt* ≤ *minGroupSize* **then**
            list ← delete(*list*, *ID*);
        else if *cnt* ≥ *maxGroupSize* **then**
            list ← splitGroup(*list*, *ID*, *maxGroupSize*);
        end if
    end for
    refPatches ← average_patches_with_same_id(*list*) ;
    loopTermination ← *loopTermination*-1 ;
end for
subGroup[*centroid, posX, posY*] = [ ];
for *refPatchID* in *refPatches* **do**
    patches_with_sameRef ← get_patches_with_same_id(*list*,*refPatchID*);
    numClusters ← count(*patches_with_sameRef*)/*clustParam* ;
    subGroup.append(cluster(*patches_with_sameRef, numClusters*)) ;
end for
gaussianWeight ← createGaussianWeight();
denoisedImg ← backPlot(*subGroup*, *gaussianWeight*);