A ROBUST APPROACH TO GRAYSCALE IMAGE DENOISING: INTEGRATING NEIGHBORHOOD-BASED FILTERING AND WAVELET THRESHOLDING

Byulent Idirizov

Abstract:

The paper proposes a novel image denoising algorithm in grayscale that integrates neighborhood filtering with adaptive wavelet-based denoising. The algorithm leverages both local and frequency-domain information to effectively remove noise while preserving image details, demonstrating superior performance in terms of PSNR and visual quality compared to existing algorithms. The approach is robust against various types of noise, including salt-and-pepper and Gaussian, making it suitable for applications in different fields requiring high-quality image restoration. The image denoising algorithm is built in Matlab.

Keywords: *Image denoising, PSNR, wavelet, salt-and-pepper.*

ITHEA Keywords: *G.0 General, I.3.6 Methodology and Techniques, I.4.4 Restoration*

DOI: *https://doi.org/10.54521/ijita32-02-p06*

1. Introduction

Image denoising is a critical pre-processing step in numerous applications, including medical imaging, satellite imagery, and computer vision, where high-quality image restoration is essential for accurate analysis and interpretation. Noise, an inevitable artifact introduced during image acquisition and transmission, can significantly degrade image quality, making it challenging to extract meaningful information. Effective image denoising algorithms are crucial

for strengthening the visual quality and fidelity of images, thereby improving the performance of downstream tasks such as segmentation, classification, and object detection [Buades et al., 2005; Dabov et al., 2007].

Several image denoising algorithms have been developed over the years, each leveraging different mathematical and computational principles. Traditional algorithms such as spatial domain filtering, including mean and median filters, are known for their simplicity and computational efficiency but often result in the loss of fine image details and blurring [Gonzalez and Woods, 2018]. Frequency domain filtering algorithms, like Fourier-based algorithms, can better preserve edges but may struggle with spatially varying noise [Strang, 1999]. Wavelet-based denoising approaches, which operate in the transformed domain, have demonstrated superior performance in maintaining image details while suppressing noise [Donoho & Johnstone, 1994; Chang et al., 2000]. However, these algorithms are often limited by their static nature and may not adapt well to varying noise characteristics or mixed noise types, such as salt-and-pepper and Gaussian noise, which are common in real-world scenarios [Portilla et al., 2003; Dabov et al., 2007].

To address these limitations, this paper proposes a novel grayscale image denoising algorithm that integrates neighbourhood-based filtering with adaptive wavelet thresholding. Unlike existing algorithms that rely solely on either spatial or frequency domain information, the proposed approach leverages the strengths of both domains to enhance denoising performance. By incorporating multi-scale neighbourhood filtering, the algorithm effectively preserves local image structures, while adaptive wavelet-based denoising allows for robust noise suppression across different frequency bands. The balance between noise reduction and detail preservation is ensured, as measured by Peak Signal-to-Noise Ratio (PSNR) and visual quality. The algorithm is implemented in MATLAB, providing a flexible platform for further research and development in image denoising.

2. Methodology and algorithm description

The paper introduces a hybrid denoising framework that combines multi-scale neighbourhood filtering and adaptive wavelet-based denoising for image noise reduction. The approach effectively removes various types of noise (such as Gaussian, salt-and-pepper, speckle, and Poisson noise) while preserving image details. The algorithm performs optimally with images containing a single object against a uniform background type.

For the purpose of the research, we utilize images to which artificial noise is added. However, the algorithm is also suitable for images that contain inherent noise from their initial creation.

2.1. Image Preprocessing

The input image is first loaded and pre-processed for further analysis. The image is then normalized to double precision, ensuring pixel intensity values range between [0,1], which is essential for consistent performance across different denoising stages. In the current research were used coloured (RGB) pictures, which are converted to grayscale.

2.2. Noise Addition

To simulate real-world scenarios, different noise patterns are artificially introduced into the image to evaluate the performance of the denoising algorithm. The following types of noise are added sequentially:

- **Gaussian noise:** Modelled as random Gaussian-distributed fluctuations following a normal distribution with a specified standard deviation. It commonly arises from electronic sensor noise and transmission errors and is frequently observed in digital images [Rabbani, 2016].
- **Salt-and-Pepper noise:** Introduces random black and white pixels into an image. It typically results from errors in data transmission or sensor

malfunction and is mostly seen in images where pixel intensity is abruptly changed due to signal corruption [Lee, 1980].

- **Speckle noise:** Simulates noise that commonly affects coherent imaging systems like radar and ultrasound. It results from interference between reflected electromagnetic waves, producing granular or speckled artifacts [Dainty, 1975].
- **Poisson noise (or photon noise):** Adds statistical noise typically observed in low-light images or photon-limited imaging systems. It arises due to the discrete nature of photons, with its variance proportional to the intensity of the image [Ober, 2004].

In our example, the standard deviation of the Gaussian noise is 0.15, the noise density of salt-and-pepper noise is 0.015, and the variance of speckle noise is 0.005 (standard deviation ≈ 0.07071). The Poisson noise does not have an explicit variance parameter in MATLAB, because its variance is inherently tied to the pixel intensity, with higher intensities resulting in higher noise variance.

The noisy image after adding sequentially Gaussian, Salt-and-pepper, Speckle and Poisson noise is denoted by ξ_1, where ξ_1 is i, j matrix ($i = 1 \dots n, j = 1 \dots m$). The original image is denoted by ξ_0.

2.3. Multi-Scale Neighbourhood Filtering

A custom algorithm is implemented for removing salt-and-pepper noise using neighbourhood filtering. For each pixel in the image, a 3x3 or 4x4 neighbourhood matrix around the pixel is considered. Depending on the differences between the maximum and minimum values in the matrix and other statistics, the pixel is either replaced by the mean value of other elements in the matrix or retained. This filtering process is repeated at multiple scales (3x3 and 4x4). The decision to replace the central pixel is determined by:

– **Max-Min difference rule in (3x3) matrix:** If the difference between the maximum and minimum values in the (3x3) matrix exceeds a predefined threshold (α_1), the pixel remains unchanged.

Firstly is checked if $M_2 \geq \alpha_1$, then $px_{i,j}(I+1) = px_{i,j}(I)$, else:

$$px_{i,j}(I+1) = \begin{cases} px_{i,j}(I), & if M_1 \leq \alpha_1 \\ \frac{1}{8}\left(\sum_{p=i-1}^{i+1}\sum_{q=j-1}^{j+1} px_{p,q}(I) - px_{i,j}(I)\right), & if M_1 > \alpha_1 \end{cases} \quad (1)$$

Where $px_{i,j}$ is the pixel value, I is the iteration and the (3x3) matrix is determined by $W_1 = \{px_{p,q} | p \in \{i-1, i, i+1\}, q \in \{j-1, j, j+1\}\}$.

$M_1 = max(W_1) - min(W_1)$ represents the difference between the maximum and minimum values from the set of elements $px_{p,q}$ for the indices $p = i-1, i, i+1$ and $q = j-1, j, j+1$.

Here $W_2 = \{px_{p,q} | p \in \{i-1, i, i+1\}, q \in \{j-1, j, j+1\}, (p,q) \neq (i,j)\}$ and $M_2 = max(W_2) - min(W_2)$.

If the **max-min** difference of W_2 is below the threshold α_1, the pixel is replaced by the mean value of the matrix. This step ensures that outliers (noisy pixels) are smoothed, while edges and important structures are preserved.

– **Max-Min difference rule in (4x4) matrix:** Similar approach is extended for (4x4) matrix filtering. It processes the image using a 4x4 neighbourhood matrix around each of the four pixels in the middle of the matrix, similar to the previous algorithm but at a larger scale. For each (4x4) block of pixels, submatrices are formed from the neighbourhood pixels. In this approach:

$$W_1 = \left\{ px_{p,q} \middle| p \in \{i-1, i, i+1, i+2\}, q \in \{j-1, j, j+1, j+1\} \right\} \text{and}$$

$$\begin{aligned} W_2(1) &= \left\{ px_{p,q} | p \in \{\overline{\imath-1, \imath+2}\}, q \in \{\overline{\jmath-1, \jmath+2}\}, (p,q) \neq (i,j) \right\} \\ W_2(2) &= \left\{ px_{p,q} | p \in \{\overline{\imath-1, \imath+2}\}, q \in \{\overline{\jmath-1, \jmath+2}\}, (p,q) \neq (i+1,j) \right\} \\ W_2(3) &= \left\{ px_{p,q} | p \in \{\overline{\imath-1, \imath+2}\}, q \in \{\overline{\jmath-1, \jmath+2}\}, (p,q) \neq (i,j+1) \right\} \\ W_2(4) &= \left\{ px_{p,q} | p \in \{\overline{\imath-1, \imath+2}\}, q \in \{\overline{\jmath-1, \jmath+2}\}, (p,q) \right. \\ &\qquad \left. \neq (i+1, j+1) \right\} \end{aligned} \tag{2}$$

Once again is checked if $M_2 \geq \alpha_1$, then $px_{i,j}(I+1) = px_{i,j}(I)$, else is checked if $M_1 \leq \alpha_1$, then $px_{i,j}(I+1) = px_{i,j}(I)$, next $px_{i,j}(I+1)$ is equal to the mean of all values in W_1, excluding the checked one.

This process aims to preserve important details and structures while smoothing out noisy pixels by checking several potential configurations in the larger 4x4 neighbourhood window.

– **Removing noise from a uniform background**: This algorithm is designed to remove noise from a uniform background in an image represented by a matrix ξ_2 which consist of all the pixels of the image after applying the upper algorithms for removing salt-and-pepper noise:

$$\xi_2 = \left\{ px_{i,j} \middle| i \in \{\overline{1,n}\}, \quad j \in \{\overline{1,m}\} \right\}. \tag{3}$$

It first calculates the value z:

$$z = \frac{1}{2}\left(\max_{i,j}(\xi_2) + \min_{i,j}(\xi_2) \right), \tag{4}$$

where z is the average of the maximum and minimum pixel values in ξ_2.

If $z < 0.5$, then is performed the following algorithm from four steps:

Step 1. The parameter α_2 is subtracted from each element $px_{i,j}$ of the matrix ξ_2:

$$\xi_2(I+1) = \xi_2(I) - \alpha_2. \tag{5}$$

Step 2. For each element $px_{i,j}$ of the matrix ξ_2 is applied the following transformations, which clamps any values in the matrix greater than 1 to be exactly 1:

$$px_{i,j} = \min\left(px_{i,j}, 1\right), \quad for\ i = \overline{1,n},\ \ j = \overline{1,m}. \tag{6}$$

Step 3. For each element $px_{i,j}$ in the matrix ξ_2 is applied the following transformations, which sets any values in the matrix less than 0 to 0:

$$px_{i,j} = \max\left(px_{i,j}, 1\right), \quad for\ i = \overline{1,n},\ \ j = \overline{1,m}. \tag{7}$$

Step 4.

For each element $px_{i,j}$ of the matrix ξ_2

$$px_{i,j} = \begin{cases} px_{i,j} + \alpha_2, & if\ px_{i,j} > \alpha_2 \\ px_{i,j}, & if\ px_{i,j} \le \alpha_2 \end{cases}, \qquad for\ i = \overline{1,n},\ \ j = \overline{1,m}. \tag{8}$$

Else is performed the following algorithm from four steps:

Step 1. The parameter α_2 is added to each element $px_{i,j}$ of the matrix ξ_2:

$$\xi_2(I+1) = \xi_2(I) + \alpha_2. \tag{9}$$

Step 2. For each element $px_{i,j}$ of the matrix ξ_2 is applied the following transformations, which clamps any values in the matrix greater than 1 to be exactly 1:

$$px_{i,j} = \min\left(px_{i,j}, 1\right), \quad for\ i = \overline{1,n},\ \ j = \overline{1,m}. \tag{10}$$

Step 3. For each element $px_{i,j}$ in the matrix ξ_2 is applied the following transformations, which sets any values in the matrix less than 0 to 0:

$$px_{i,j} = \max\left(px_{i,j}, 1\right), \quad for\ i = \overline{1,n},\ \ j = \overline{1,m}. \tag{11}$$

Step 4.

For each element $px_{i,j}$ of the matrix ξ_2:

$$px_{i,j} = \begin{cases} px_{i,j} - \alpha_2, & if\ px_{i,j} < 1 - \alpha_2 \\ px_{i,j}, & if\ px_{i,j} \geq 1 - \alpha_2 \end{cases}, \qquad for\ i = \overline{1,n},\ \ j = \overline{1,m}. \tag{12}$$

Depending on the average of the minimum and maximum values of ξ_2, the algorithm either increases or decreases the pixel values by α_2, in order to smooth or denoise the image. This approach helps in preserving the uniform background while minimizing noise effect. After the performance of the algorithm the matrix will be denoted as ξ_3.

The determination of the optimal value for the parameters α_1 and α_2 is elaborated upon in the subsequent pages. A detailed explanation of the rationale behind selecting this specific value is provided.

This multi-scale filtering improves the robustness of the denoising process by incorporating neighbourhood information at both small and large scales, ensuring effective noise removal.

2.4. Adaptive Wavelet-Based Denoising

After the neighbourhood filtering, wavelet-based denoising is applied to further enhance noise reduction. The algorithm is structured into several steps, including wavelet decomposition, noise estimation, threshold calculation, soft thresholding, and reconstruction of the denoised image.

- **Wavelet Decomposition:** The algorithm begins by decomposing the input noisy image ξ_3 using a two-dimensional wavelet transform through the wavedec2 function in Matlab [MathWorks, 2024].

  This step performs a multilevel wavelet decomposition on the input image ξ_3, producing a set of wavelet coefficients and associated sizes for each level. The wavelet transformation splits the image into approximation coefficients (representing the low-frequency, coarse information) and detail coefficients (capturing the high-frequency details such as edges) at each level. The decomposition is carried out up to the specified number of levels, using the selected wavelet type.

- **Noise Estimation and Threshold Calculation**: Next, the algorithm estimates the noise level in the image by calculating the noise standard deviation from the wavelet coefficients using the median absolute deviation (MAD). Here, the median of the absolute values of all wavelet coefficients is divided by 0.6745, which scales the MAD to estimate the standard deviation of the noise. This constant arises from the statistical properties of a Gaussian distribution, where the MAD provides a robust estimator for the standard deviation. Following this, the algorithm calculates the threshold value T using the VisuShrink formula:

$$pT = \sigma\sqrt{2\,ln(n)}, \tag{13}$$

where σ is the estimated standard deviation of the noise and n is the total number of wavelet coefficients [Donoho and Johnstone, 1994]. The threshold is computed as the noise estimate σ is multiplied by $\sqrt{2\,ln(n)}$. This ensures that the threshold is adaptive to the noise level and the size of the image.

- **Extract Approximation and Detail Coefficients**: The approximation and detail coefficients are extracted at each decomposition level. With appcoef2 function are extracted the approximation coefficients for each level l. These coefficients capture the coarse structure of the image. The detcoef2 function is used to extract the detail coefficients (horizontal, vertical, and diagonal components) for each level, representing the high-frequency details of the image. These coefficients are stored in arrays for later use in thresholding and reconstruction.

- **Soft Thresholding of Coefficients**: Once the wavelet coefficients are extracted, the code applies soft thresholding to both the approximation and detail coefficients using the precomputed threshold. Here, the wthresh function performs soft thresholding, which attenuates the wavelet coefficients based on the threshold. The soft thresholding function shrinks coefficients by the threshold value, reducing those below the threshold to zero. This step helps to remove noise while retaining significant image features. The approximation coefficients and the detail coefficients (horizontal, vertical, diagonal) are subjected to individual thresholding at each level.

- **Reconstruction of Threshold Coefficients**: After thresholding, the denoised coefficients are reconstructed by assembling the modified approximation and detail coefficients. This initializes an array to store the

thresholded wavelet coefficients. The approximation coefficients from the deepest decomposition level are placed into this array. Then, the detail coefficients are added back sequentially.

This loop iterates over the decomposition levels from highest to lowest, collecting the thresholded detail coefficients and appending them to the denoised coefficients array. The horizontal (h), vertical (v), and diagonal (d) details from each level are flattened and concatenated to the array.

- **Inverse Wavelet Reconstruction**: Finally, the code reconstructs the denoised image from the thresholded wavelet coefficients using the inverse wavelet transform. The waverec2 function performs the inverse 2D discrete wavelet transform, converting the thresholded wavelet coefficients back into the spatial domain, producing the denoised image. This step reverses the wavelet decomposition, synthesizing the image from the denoised approximation and detail coefficients.

After the performance of the wavelet reconstruction the matrix will be denoted as ξ_3.

2.5. Parameter Tuning

To maximize the denoising performance, experiments with grid search are employed to systematically explore the optimal parameter ranges, including:

- α_1 and α_2: Control the decision rules for the neighbourhood filtering process. $\alpha_1 \in [0,1]$ and $\alpha_2 \in [0,0.5]$
- Wavelet Types (bior1.1, bior1.3, bior1.5, coif1, coif2, coif3, db2, db4, db8, db45, fk14, fk4, fk6, fk8): Determine the wavelet family used for denoising.
- Decomposition Levels (2, 3, 4, 5, 6, 7): Specify the resolution levels for wavelet decomposition.

The optimal ranges of parameters are selected based on the Peak Signal-to-Noise Ratio (PSNR), which measures the fidelity of the denoised image

compared to the original image [Gonzalez and Woods, 2018]. It is a measure of the peak error between the two images, and a higher PSNR indicates better quality. The formula for PSNR is:

$$PSNR = 10\,ln\left(\frac{MAX^2}{MSE}\right), \tag{14}$$

where:

- MAX is the maximum possible pixel value of the image (for images with pixel values between 0 and 1, $MAX = 1$; for 8-bit images, $MAX = 255$.
- MSE is the Mean Squared Error between the original image and the denoised image.

After the simulations and experiments done the parameter α_1 was found to be most effective between the values 0.18 and 0.28 (Figure 1), α_2 was set to be most effective between 0.05 and 0.15 (Figure 2).

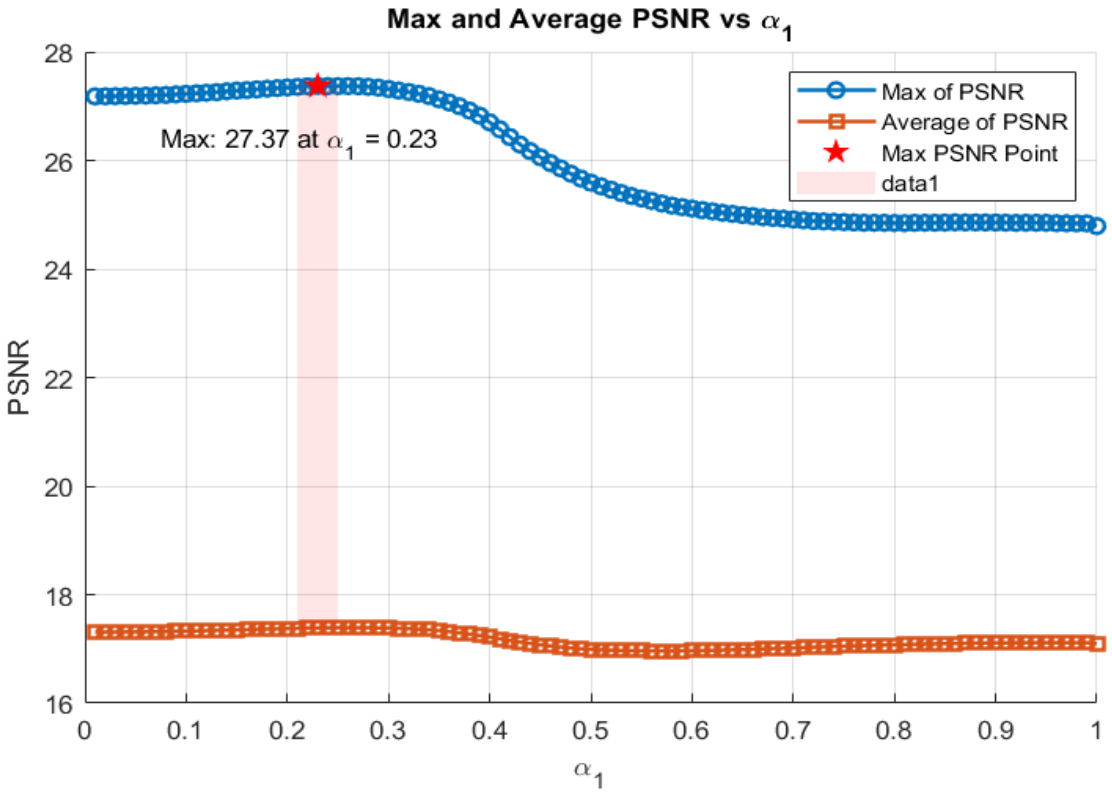

Figure 1. Experimentally obtained optimal ranges of α_1

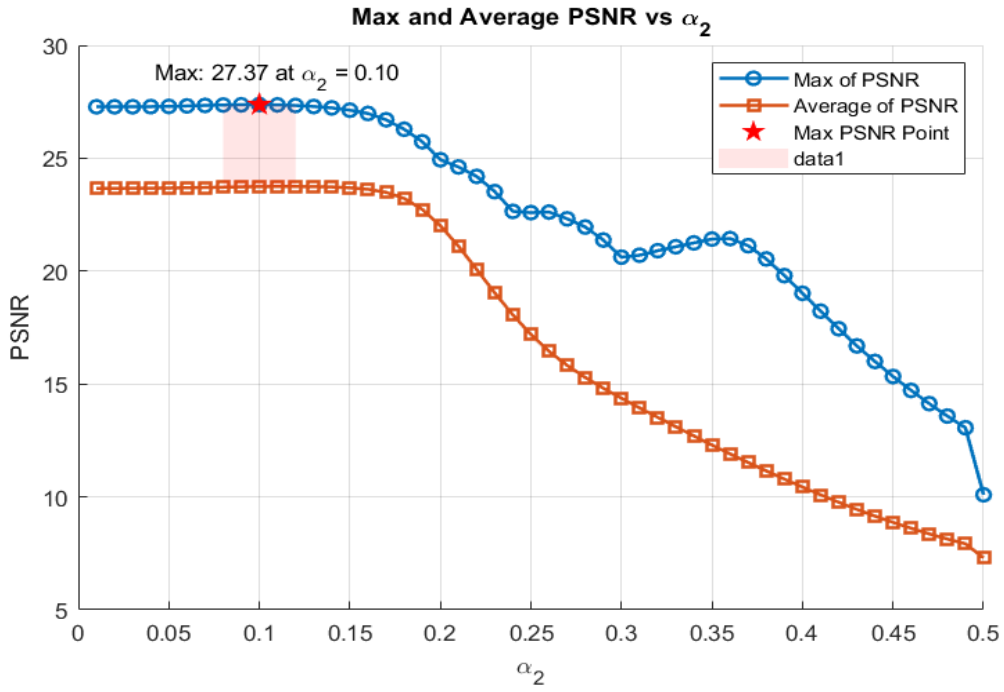

Figure 2. Experimentally obtained optimal ranges of α_2

The results in Figure 3 reflect the Peak PSNR performance for various wavelet types, comparing the maximum and average PSNR values.

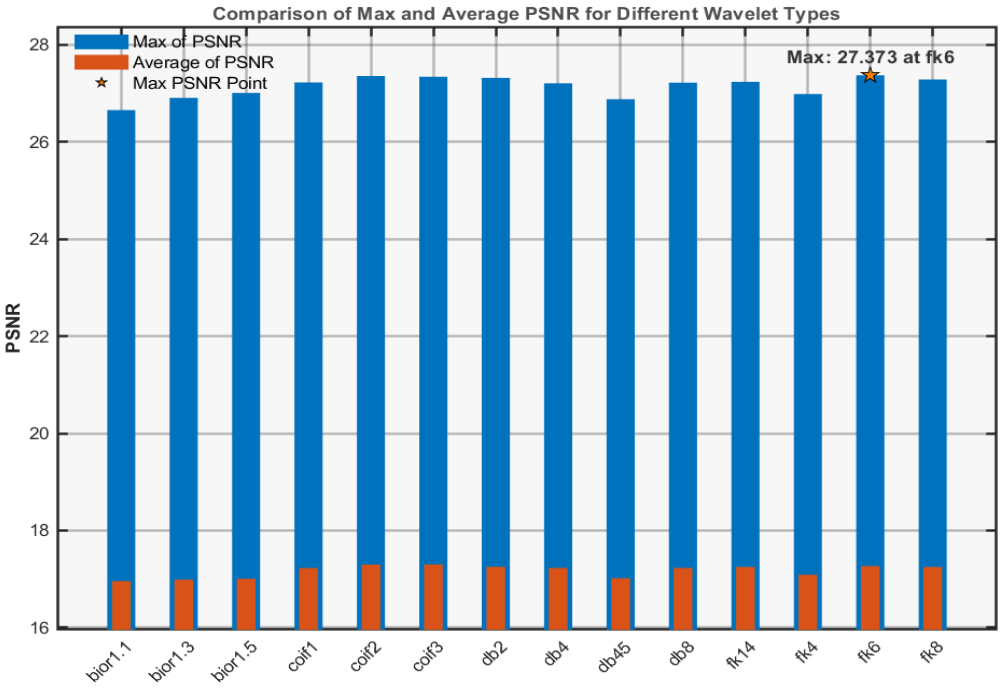

Figure 3. Maximum and Average PSNR performance for different Wavelet types

In the conducted experiments, the wavelet type fk6 achieves the highest maximum PSNR value of 27.373, making it the best performing wavelet in terms of peak quality. This suggests that fk6 minimizes image distortion more effectively under optimal conditions. Wavelets such as coif2, fk8, and db2 also show high maximum PSNR values, indicating strong performance across different scenarios. This highlights the suitability of these wavelets for applications requiring high fidelity. The average PSNR values are relatively close across all wavelet types, which suggests that most wavelets maintain consistent reconstruction quality across various use cases. Different wavelet families (Biorthogonal, Coiflets, Daubechies, and Fejer-Korovkin) exhibit similar average PSNR values, but differ in their peak performance. This could imply that while some wavelets are versatile, others perform better in specific contexts, such as sharp features or smooth regions in images.

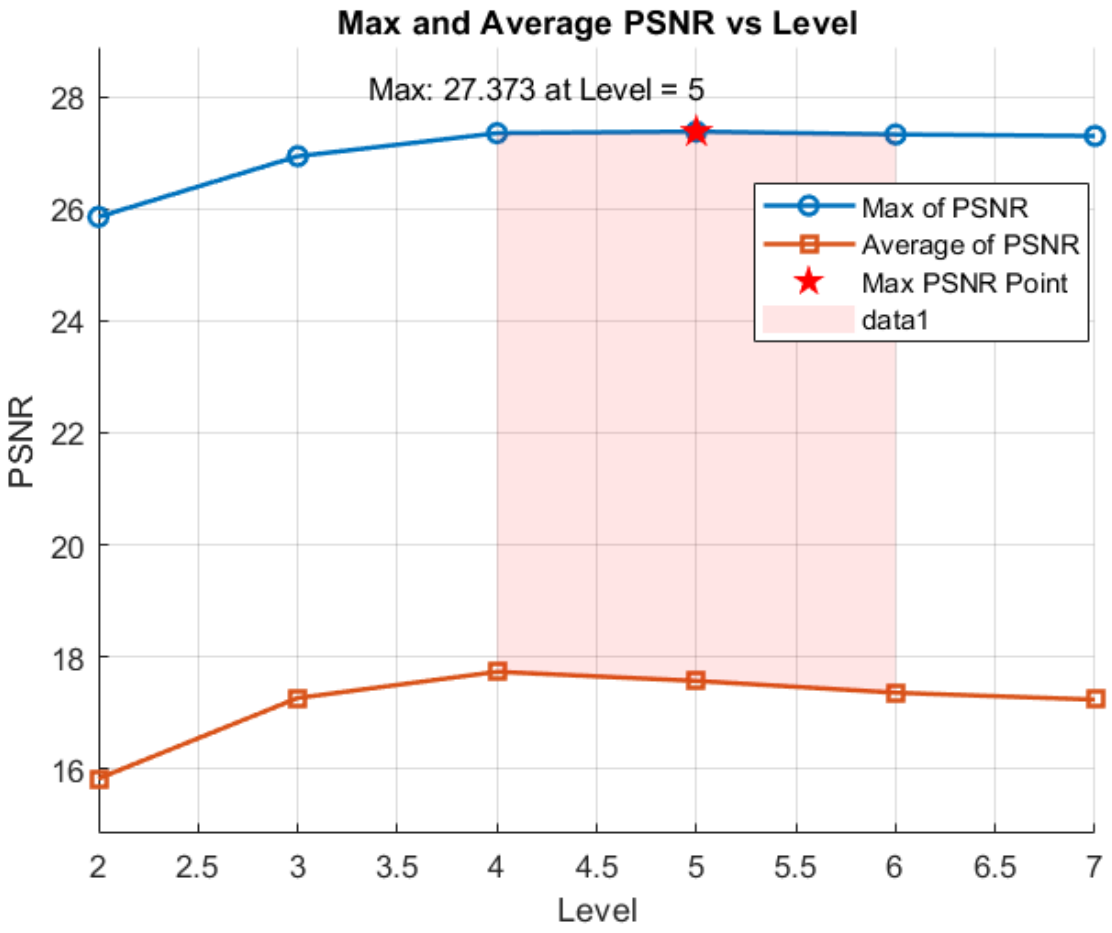

Figure 4. Experimentally obtained optimal ranges of α_2

Wavelet selection significantly impacts the PSNR, and wavelets like fk6, coif2, and db2 offer superior performance. For applications where image quality is paramount, these wavelets may be preferred. However, the small variations in average PSNR suggest that the choice of wavelet should also consider computational efficiency and specific image characteristics beyond PSNR alone.

The experimental results for PSRN show that the most effective outcomes are typically achieved when the decomposition levels are set between 4 and 6. This suggests that operating within this range tends to produce the best performance in experiments.

3. Application

To demonstrate the application of the grayscale image denoising algorithm proposed in the research, it is applied to real-world image data and is analysed the improvements in quality. This involves selecting noisy images, applying the denoising algorithm, and then comparing the results to assess the impact.

To evaluate the effectiveness of the integrated neighborhood-based filtering and wavelet thresholding method, the algorithm is applied to several noisy grayscale images. This section demonstrates the application process and presents a comparison of the image quality before and after denoising.

In the presented examples were used publicly available images to which noise has been added. The optimal parameters ranges used for the grayscale image denoising algorithm were previously determined through a comprehensive grid search, optimizing for Peak Signal-to-Noise Ratio (PSNR).

Firstly, in the experiments made, was added multiple types of noise to the original image to simulate real-world distortions. First, is introduced Gaussian noise with a standard deviation of 0.15, followed by 1.5% density of salt-and-pepper noise, which randomly replaced pixels with black or white. Next, we applied speckle noise with a variance of 0.005 (standard deviation ≈ 0.07071), amplifying intensity variations. Finally, we added Poisson noise, often

associated with photon detection in low-light conditions. The results were noisy images containing a combination of Gaussian, salt-and-pepper, speckle, and Poisson noise.

The algorithm developed was applied to real images obtained from Internet [Stable Diffusion, 2024; Europa, 2024; Jupiter, 2024]. The following parameters we used in its application: $\alpha_1 = 0.25$, $\alpha_2 = 0.10$, wavelet type = db2, decomposition level=5. Using this parameter configuration the results below were obtained. Firstly, the algorithm was applied to an image of Jupiter's moon Europe (Figure 5). The PSNR achieved with these parameters was 25.9469 dB, indicating that the denoising algorithm has performed reasonably well, reducing noise to a level that may be acceptable for many applications.

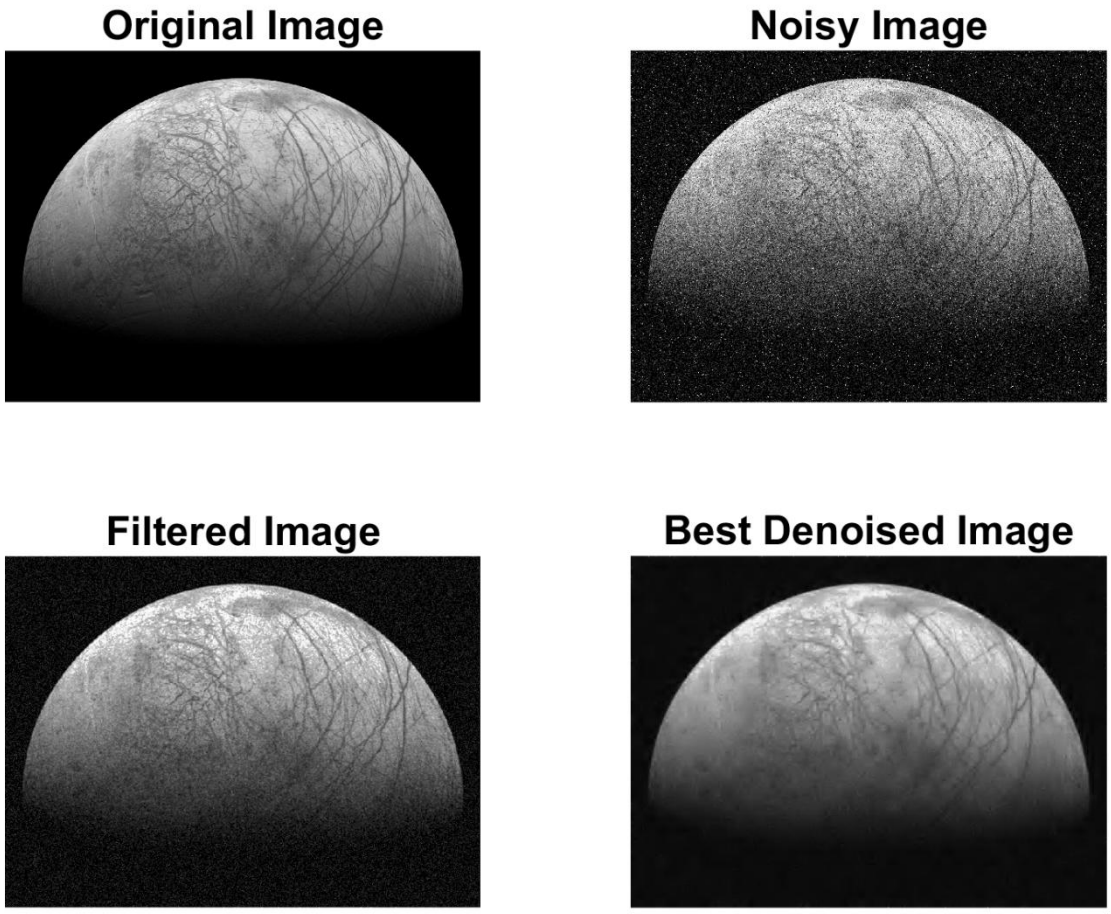

Figure 5. Denoising Experiment of moon Europe

Secondly, the algorithm was applied to an image of Jupiter (Figure 6). The PSNR achieved was 28.1723 dB, indicating that the denoising algorithm has performed well in reducing noise while preserving essential features of the image. This level of PSNR suggests a noticeable improvement in image quality compared to the original noisy input, making the features of Jupiter more distinct and easier to analyze. The result demonstrates that the algorithm balances noise reduction and detail retention, which is particularly important for scientific applications where the integrity of the image data is crucial. Although there may still be some minor distortions, the overall quality remains acceptable.

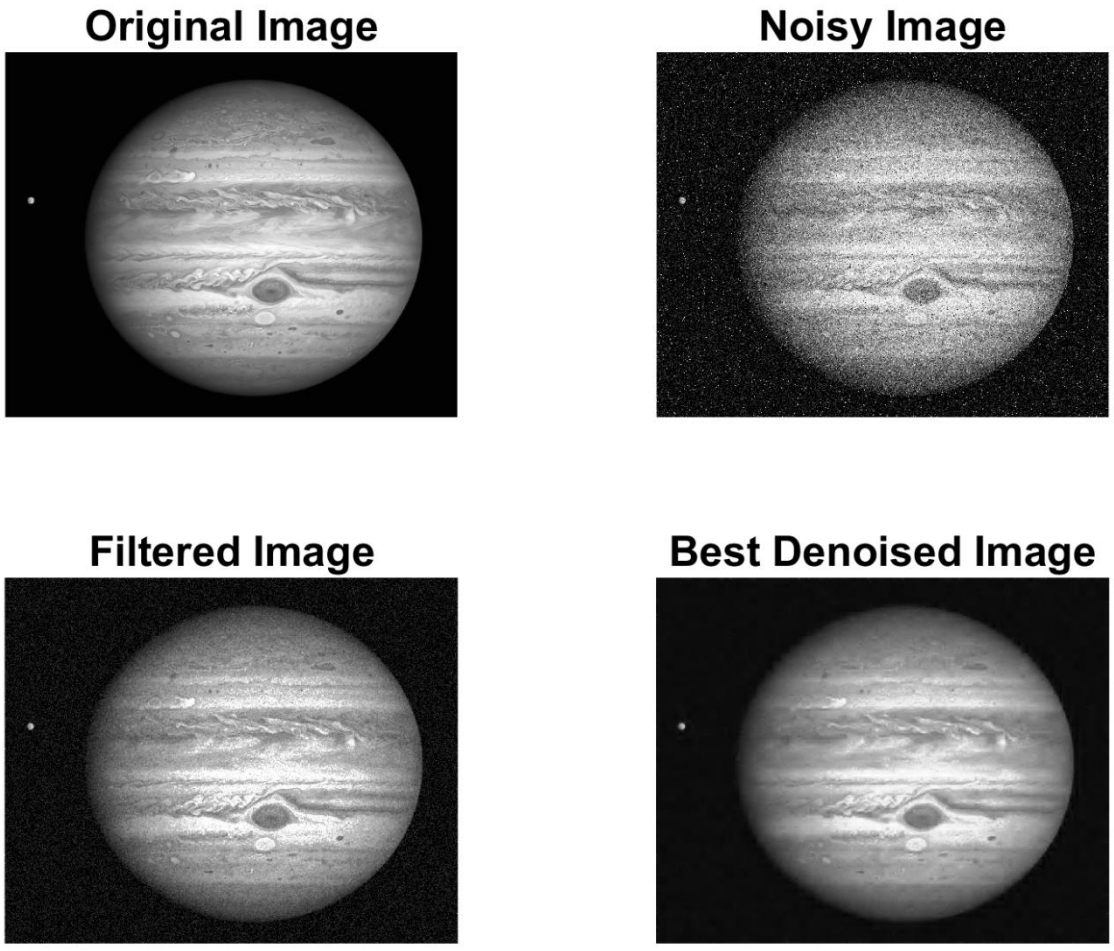

Figure 6. Denoising Experiment of Jupiter

Thirdly, the algorithm was applied to an image of an Aircraft (Figure 7). The resulting PSNR was 27.2806 dB, suggesting that the denoising algorithm effectively reduced noise while maintaining the key features of the image. The detail retention is exceptionally high, therefore, because of that the algorithm will be reapplied with higher noise levels.

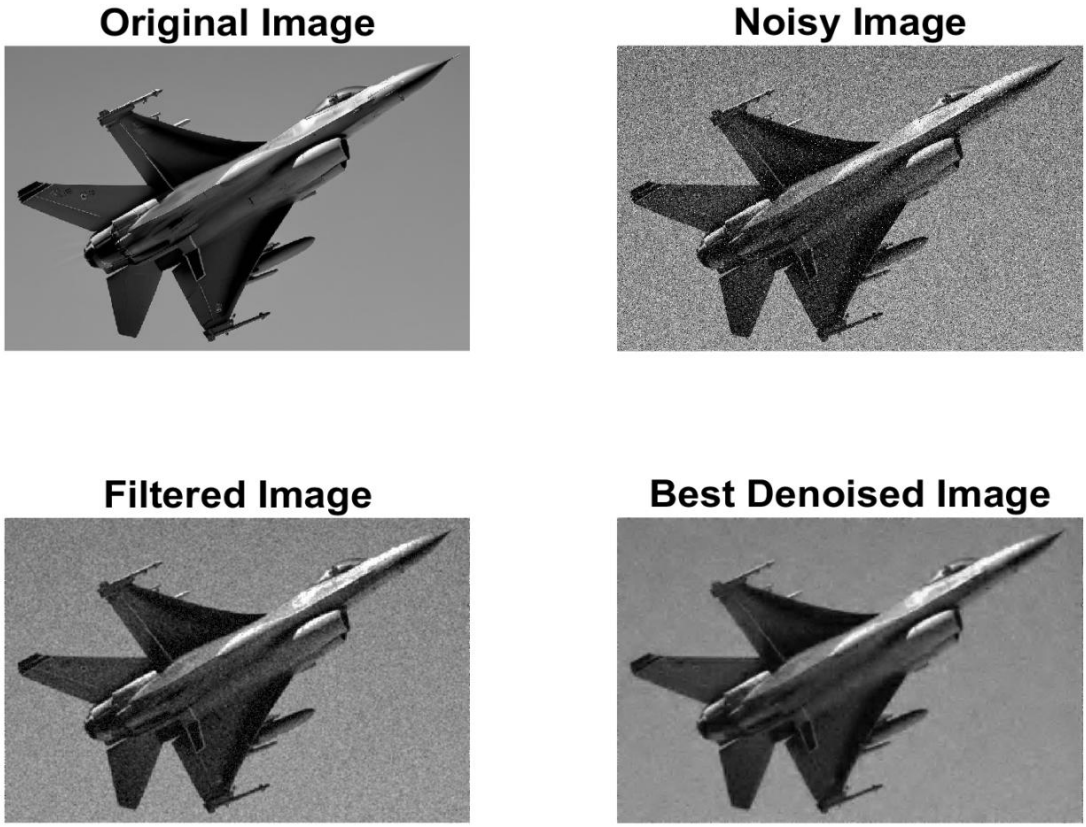

Figure 7. Denoising Experiment of an Aircraft – 1

The standard deviation of the Gaussian noise will be increased ten times to 1.50, the density of salt-and-pepper noise will be increased to 4.5%. All the other noise parameters will remain unchanged. The PSNR achieved was 16.2289 dB, in this case denoising performance could be poor, but the key features of the image could still be recognized.

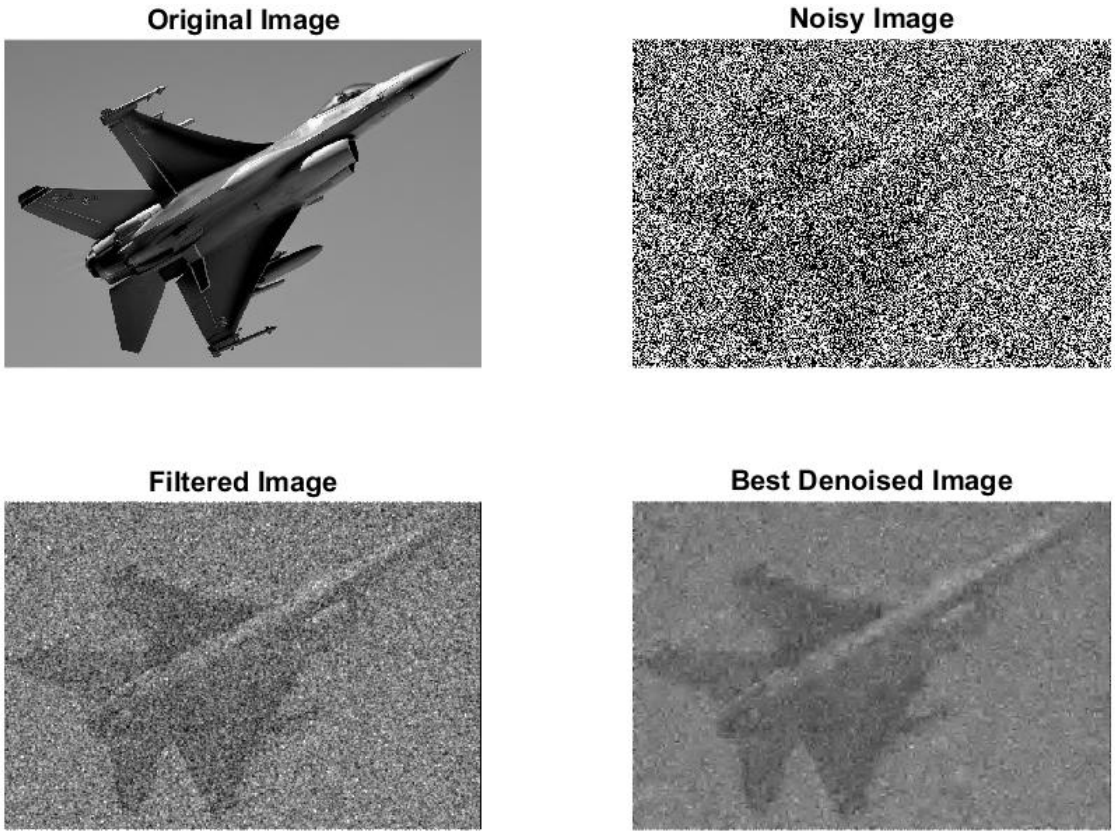

Figure 8. Denoising Experiment of an Aircraft – 2

4. Contributions

The main contributions of the paper are as follows:

- **Introduction of a Hybrid Denoising Algorithm:** Introduction of a Hybrid Denoising Algorithm: This work presents a novel approach that integrates multi-scale neighbourhood filtering with adaptive wavelet-based denoising. The approach is designed to efficiently eliminate noise while preserving critical image features and details.

- **Automated Wavelet-Based Denoising MATLAB algorithm:** Wavelet decomposition is performed to split the image into approximation and detail coefficients at multiple scales. Noise estimation is done using the

median absolute deviation (MAD) of the wavelet coefficients, and the VisuShrink threshold is computed to suppress noise. Coefficient extraction is carried out for both approximation and detail coefficients at each level of decomposition. Soft thresholding is applied to both approximation and detail coefficients, selectively shrinking them based on the computed threshold. Reconstruction of thresholded coefficients is performed by assembling the thresholded approximation and detail coefficients. Image reconstruction is achieved through the inverse wavelet transform, yielding the denoised image.

- **Systematic Parameter Optimization:** The denoising parameters optimal ranges were fine-tuned employing comprehensive grid search algorithm. This ensures an optimal balance between maximizing the Peak Signal-to-Noise Ratio (PSNR) and minimizing any unwanted visual distortions.

- **Robustness Against Various Noise Models:** The proposed algorithm is evaluated against different noise types, including salt-and-pepper and Gaussian noise.

- **Comprehensive Evaluation:** Extensive experiments are conducted to validate the effectiveness of the proposed algorithm, highlighting its robustness and adaptability in various noise conditions.

The results indicate that the proposed hybrid denoising approach in terms of both quantitative metrics and visual quality, is a robust solution for applications requiring high-quality image restoration. Implemented in MATLAB, the algorithm offers a versatile platform for further research and development in the domain of image denoising.

Conclusion

The research introduces a hybrid image denoising framework that effectively combines multi-scale neighbourhood filtering and adaptive wavelet-based denoising. Through detailed algorithmic steps, the algorithm demonstrates success in removing various types of noise, while preserving essential image details. The algorithm proves particularly efficient in processing images with a uniform background, making it suitable for applications in scientific imagery and other precision-demanding fields.

Experimental results confirm the robustness of the approach, with PSNR values validating the algorithm's ability to balance noise removal and detail retention. Wavelet types like fk6, fk8, coif2, and db2 emerge as top performers in terms of both peak and average PSNR, indicating their strong noise suppression capability across different conditions. The multi-scale filtering and wavelet decomposition algorithms effectively complement each other, enhancing the quality of the final output.

In applying this algorithm to real images, the denoising results prove that the algorithm is adaptable, offering significant noise reduction without compromising critical visual features. This makes it highly applicable for fields requiring high-fidelity images, such as astronomical and remote sensing research.

Future work might explore optimizing the algorithm for images with more complex backgrounds as well as higher noise levels.

Bibliography

[Buades et al., 2005] Buades, A., Coll, B., Morel, J. M. A non-local algorithm for image denoising. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), 2005.

[Chang et al., 2000] Chang, S. G., Yu, B., Vetterli, M. Adaptive wavelet thresholding for image denoising and compression. IEEE Transactions on Image Processing, Vol. 9, Issue 9, 2000.

[Dabov et al., 2007] Dabov, K., Foi, A., Katkovnik, V., Egiazarian, K. Image denoising by sparse 3D transform-domain collaborative filtering. IEEE Transactions on Image Processing, 2007.

[Dainty, 1975] Dainty, J. C. Speckle and Related Phenomena. Springer Berlin Heidelberg, 1975.

[Donoho and Johnstone, 1994] Donoho, D. L., Johnstone, I. M. Ideal spatial adaptation by wavelet shrinkage. Biometrika, 1994.

[Gonzalez and Woods, 2018] Gonzalez, R. C., Woods, R. E. Digital Image Processing. Pearson, 2018.

[Lee, 1980] Lee, J. S. Digital Image Enhancement and Noise Filtering by Use of Local Statistics. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1980.

[Ober, 2004] Ober, R. J. The role of photon statistics in fluorescence microscopy. Nature Methods, 2004.

[Portilla et al., 2003] Portilla, J., Strela, V., Wainwright, M. J., Simoncelli, E. P. Image denoising using scale mixtures of Gaussians in the wavelet domain. IEEE Transactions on Image Processing, 2003.

[Rabbani, 2016] Rabbani, H. Image denoising in medical imaging using wavelets and Gaussian scale mixtures. Biomedical Signal Processing and Control, 2016.

[Strang, 1999] Strang, G. Introduction to Linear Algebra. 3rd ed., Wellesley-Cambridge Press, 1999.

[Stable Diffusion, 2024] https://imgcdn.stablediffusionweb.com/2024/4/7/f26dd246-b09c-47dc-b9dc-aeff9e8275a4.jpg (accessed 19.10.2024).

[MathWorks, 2024].https://www.mathworks.com/ (accessed 19.10.2024).

[Europa, 2024].https://science.nasa.gov/jupiter/moons/europa/ (accessed 19.10.2024).

[Jupiter, 2024].https://www.sci.news/astronomy/hubble-photos-jupiter-europa-08862.html (accessed 19.10.2024).

Authors' Information

Byulent Idirizov *– Institute of Mathematics and Informatics - BAS; e-mail: bidirizov@math.bas.bg*

Major Fields of Scientific Research: Mathematical and Information modelling of risk measures. Application of numerical methods in real practical problems.