Proceedings of the Eighth International Conference on
Parallel, Distributed, GPU and Cloud Computing for Engineering
Edited by: P. Iványi, J. Kruis and B.H.V. Topping
Civil-Comp Conferences, Volume 12, Paper 2.6
Civil-Comp Press, Edinburgh, United Kingdom, 2025
ISSN: 2753-3239, doi: 10.4203/ccc.12.2.6

A Parallel Hybrid Method to Reduce Gaussian Noise in Computed Tomography Medical Images

J. Arnal and D. Ibarra

**Department of Computer Science and Artificial Intelligence,
University of Alicante, Spain**

Abstract

A hybrid filter to remove Gaussian noise from computed tomography medical images is presented. The method combines in an efficient manner the advantages of two denoising filters. In addition, a parallel filter based on this method is presented. Implementation of the parallel algorithm on multi-core platform using Open Multi-Processing is presented. Efficiency is measured in terms of execution time and in terms of Peak Signal-to-Noise Ratio over medical computed tomography images. Experimental results show that Gaussian noise is reduced efficiently and image details are preserved. The parallel implementation is compared to the sequential one, obtaining significative values of speed-up.

Keywords: parallel computing, image processing, medical imaging, computed tomography, noise reduction, Gaussian noise.

1 Introduction

Computed tomography (CT) is a fundamental medical imaging modality widely employed for diagnostic purposes and in image-guided interventions. However, CT images are often degraded by noise, primarily as a consequence of the image acquisition process and efforts to minimize patient exposure to radiation. As radiation dose is reduced, the level of noise inherently increases, thereby compromising image quality. Consequently, effective denoising techniques are essential to ensure reliable analysis of CT data. Among the various types of noise, additive Gaussian noise is the most prevalent in CT imagery [1, 2]. This noise can also be superimposed with other noise sources, which further complicates the denoising task. To address this issue, numerous algorithms have been developed to suppress Gaussian noise (see, for example, [3–7]).

The Block-Matching and 3D Filtering (BM3D) method [7] presents outstanding results in terms of noise suppression. However, BM3D's high computational cost significantly limits its applicability in time-critical or resource-constrained environments. In this work, we propose a hybrid denoising method that combines the strengths of the Guided Filter (GF) [8] and BM3D, aiming to improve both performance and efficiency. The guided filter, known for its edge-preserving smoothing capabilities and low computational complexity, serves as a fast and effective pre-processing step to enhance the input image before applying BM3D. This combination allows for better structure preservation and noise reduction, while also optimizing the overall denoising process.

In this context, this study builds upon these foundations by proposing a hybrid GF-BM3D method with a parallel implementation. We leverage the guided filter's efficiency to pre-process the noisy input and reduce the load on BM3D, while parallelizing the workflow to enhance scalability and applicability to real-world scenarios.

Section 2 describes the filter design. In section 3 the experimental results are shown and section 4 presents the conclusions.

2 Methods

In this section, we present the proposed hybrid image denoising method, which integrates the GF and the BM3D algorithms. The denoising process is carried out in two stages. In the first stage, the guided filter is applied to the noisy image to perform an initial smoothing while preserving important image structures. In the second stage, the output of the guided filter is fed into the BM3D algorithm to perform a more refined denoising. This two-step process aims to reduce the computational burden of BM3D by attenuating the noise level beforehand and improving patch matching accuracy.

2.1 Guided Filter

The GF filter [8] is an edge-preserving smoothing operator that leverages a guidance image I to filter an input image p. The filter assumes a local linear model between the guidance image I and the output image q, such that for a window ω_k centered at pixel k, the output is modeled as:

$$q_i = a_k I_i + b_k, \quad \forall i \in \omega_k, \tag{1}$$

where a_k and b_k are linear coefficients assumed constant in ω_k. These coefficients are computed by minimizing the following cost function:

$$E(a_k, b_k) = \sum_{i \in \omega_k} \left((a_k I_i + b_k - p_i)^2 + \epsilon a_k^2 \right), \tag{2}$$

where ϵ is a regularization parameter controlling the degree of smoothing.

The optimal coefficients are given by:

$$a_k = \frac{\frac{1}{|\omega|} \sum_{i \in \omega_k} I_i p_i - \mu_k \bar{p}_k}{\sigma_k^2 + \epsilon}, \quad b_k = \bar{p}_k - a_k \mu_k, \tag{3}$$

where μ_k and σ_k^2 are the mean and variance of I in ω_k, and $\bar{p}_k$ is the mean of p in the same window.

The final output q is obtained by averaging the overlapping windows:

$$q_i = \frac{1}{|\omega|} \sum_{k: i \in \omega_k} (a_k I_i + b_k). \tag{4}$$

In the proposed hybrid method, the guided filter serves as a preprocessing step that reduces high-frequency noise while maintaining image structures, which benefits the subsequent BM3D filtering stage.

2.2 Block-Matching and 3D Filtering (BM3D)

The BM3D filter [7] is an image denoising algorithm that exploits the self-similarity of patches in natural images. The algorithm operates in two main steps: hard-thresholding and Wiener filtering, both performed in the transform domain.

Stage 1 - Hard Thresholding: Similar patches are grouped into 3D arrays called *groups*, and a 3D linear transform (e.g., 2D DCT + 1D Haar) is applied. The coefficients are hard-thresholded to suppress noise:

$$\hat{X}_1 = T^{-1}(H_\lambda(T(X))), \tag{5}$$

where T and T^{-1} are the 3D transform and its inverse, and H_λ is the hard-thresholding operator with threshold λ.

Stage 2 - Wiener Filtering: Using the basic estimate from the first stage as a reference, new 3D groups are formed and denoised using Wiener filtering:

$$\hat{X}_2 = T^{-1}\left(W(T(Z), T(X))\right), \tag{6}$$

where Z is the group from the noisy image, X is the reference group, and W is the Wiener filtering function computed element-wise.

The outputs of both stages are aggregated using weighted averaging to form the final denoised image.

2.3 Parallelization Strategy

To reduce the computational time of the hybrid denoising algorithm, we implement a parallelization strategy based on domain decomposition. The image domain Ω is partitioned into P subdomains $\{\Omega_i\}_{i=1}^{P}$, where P corresponds to the number of parallel processing elements. This partitioning satisfies the following conditions:

$$\Omega_i \subset \Omega, \quad \bigcup_{i=1}^{P} \Omega_i = \Omega, \quad \Omega_i \cap \Omega_j = \emptyset \quad \text{for } i \neq j. \tag{7}$$

Each subdomain Ω_i contains a distinct subset of rows of the image, allowing independent denoising of image segments. Figure 1 shows an example where the image is split into four horizontal stripes.

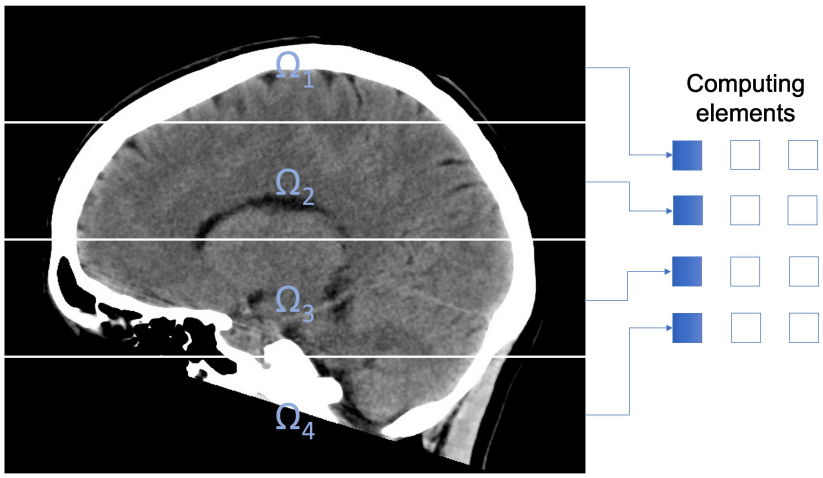


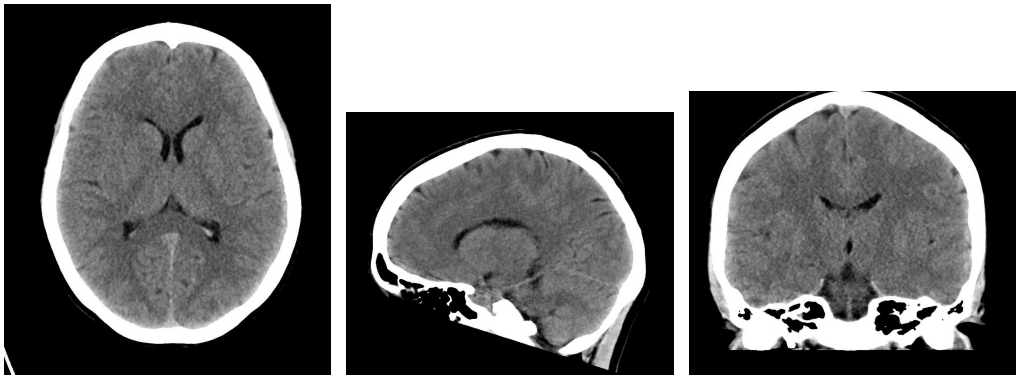
Figure 1: Image domain decomposition into 4 subdomains.

In the implementation, the guided filter is applied locally within each subdomain, and care is taken to handle boundary effects by extending subdomains with a small overlap. After filtering, the BM3D algorithm is independently applied to each region. At the end of the computation, the subdomain results are gathered and recombined into the final output image. This parallel design allows the method to scale effectively with the number of processors and significantly reduces the execution time of the BM3D stage.

3 Results

We have coded the parallel implementation of the algorithm on a multi-core platform using the Open Multi-Processing (OpenMP) [9]. Both the serial code and parallel code were implemented in C. The GNU Compiler Collection version 11.4.0 was used. We developed experiments on a multi-core AMD Ryzen Threadripper PRO 5955WX (16 cores), 4.0 GHz, with 125 GB RAM, under the operative system Ubuntu 22.04.5 LTS.

For this purpose, CT medical images (see Figure 2) from Radiopaedia database (Case Courtesy of A. Prof Frank Gaillard, Radiopaedia.org, rID 35508) have been considered in the study. These images correspond to a normal brain of a 30 years old female. CT images were corrupted with varying levels of Gaussian noise (variance $\sigma^2 \in [0.005, 0.03]$). To this end, the classical Gaussian-noise model [3] was adopted.



(a) Axial view (b) Sagittal view (c) Coronal view

Figure 2: CT images used in the experiments (a) Axial: 1024×904 pixels, (b) Sagittal: 822×1024 pixels , (c) Coronal: 890×1024 pixels.

Filter performance has been analyzed with the objective measure Peak Signal-to-Noise Ratio (PSNR) [10] that measures the noise reduction. The PSNR is defined by the following formula:

$$PSNR = 10 \cdot \log_{10} \left(\frac{MAX_I^2}{MSE} \right) \tag{8}$$

Where:

- MAX_I: Represents the maximum possible pixel value of the image. For 8-bit images, this value is typically 255.
- MSE: The Mean Squared Error (MSE) is the average of the squared differences between the original image and the processed image. It is calculated as:

$$MSE = \frac{1}{N \cdot M} \sum_{i=1}^{N} \sum_{j=1}^{M} (I(i,j) - K(i,j))^2 \tag{9}$$

where $I(i,j)$ is the pixel value at position (i,j) in the original image, $K(i,j)$ is the pixel value at the same position in the processed image, and $N \times M$ is the size of the image (number of pixels).

The PSNR value is measured in decibels (dB), and a higher PSNR indicates better quality. Table 1 presents the PSNR obtained for the CT images corrupted with different levels of Gaussian noise with variance σ^2. It can be observe that the filter obtains outstanding PSNR values.

Noise Variance	Axial View		Sagittal View		Coronal View	
	Noisy	Filtered	Noisy	Filtered	Noisy	Filtered
$\sigma^2 = 0.005$	24.44	31.36	24.56	30.75	24.33	30.75
$\sigma^2 = 0.01$	21.44	29.15	21.56	28.58	21.34	28.72
$\sigma^2 = 0.02$	18.46	26.63	18.58	26.13	18.39	26.34
$\sigma^2 = 0.03$	16.76	25.03	16.88	24.57	16.60	24.78

Table 1: PSNR values for noisy and filtered CT images at different noise variances.

To measure the parallel performance, the speed-up S_P is computed as:

$$S_P = \frac{T_{seq}}{T_P} \tag{10}$$

where T_{seq} is the computational time of the sequential method and T_P is the computational time of the parallel algorithm.

Tables 2, 3 and 4 show the computational time and speed-up obtained for the test CT images contaminated with different levels of Gaussian noise. Results demonstrate that the parallel algorithm achieves speed-ups in the range of 7.65 to 9.90 for the CT Axial image, in the range of 7.35 to 9.03 for the CT Sagittal image, and in the range of 7.95 to 9.47 for the CT Coronal image when using all 16 cores of the multi-core system.

Cores	$\sigma^2 = 0.005$		$\sigma^2 = 0.01$		$\sigma^2 = 0.02$		$\sigma^2 = 0.03$	
	Time	**Speed-up**	**Time**	**Speed-up**	**Time**	**Speed-up**	**Time**	**Speed-up**
1	18.60		18.92		19.69		27.11	
2	10.44	1.78	10.38	1.82	10.89	1.81	15.12	1.79
4	5.32	3.50	5.31	3.56	5.55	3.55	7.74	3.50
8	3.12	5.96	3.12	6.06	3.23	6.10	4.65	5.83
16	1.96	9.49	2.01	9.41	1.99	9.90	3.54	7.65

Table 2: Computational time in seconds and Speed-up for CT Axial image contaminated with Gaussian noise of different variances.

Cores	$\sigma^2 = 0.005$		$\sigma^2 = 0.01$		$\sigma^2 = 0.02$		$\sigma^2 = 0.03$	
	Time	**Speedup**	**Time**	**Speedup**	**Time**	**Speedup**	**Time**	**Speedup**
1	16.59		16.60		16.85		23.72	
2	9.58	1.73	9.68	1.71	9.84	1.71	13.51	1.75
4	4.97	3.34	4.98	3.33	5.00	3.37	7.05	3.36
8	2.89	5.74	2.84	5.85	2.96	5.70	4.05	5.86
16	1.87	8.87	1.87	8.85	1.87	9.03	3.23	7.35

Table 3: Computational time in seconds and speed-up for CT Sagittal image contaminated with Gaussian noise of different variances.

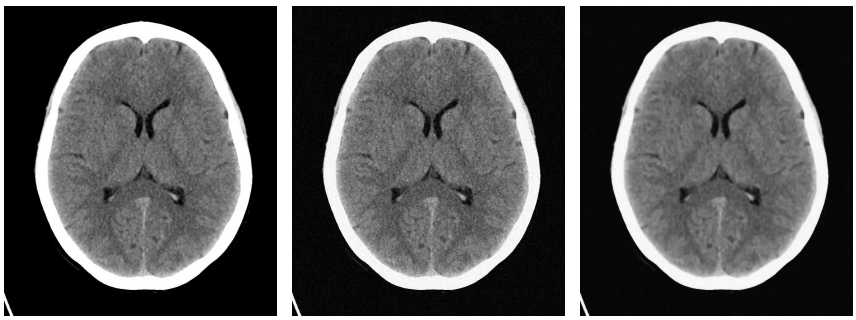
Cores	$\sigma^2 = 0.005$		$\sigma^2 = 0.01$		$\sigma^2 = 0.02$		$\sigma^2 = 0.03$	
	Time	**Speed-up**	**Time**	**Speed-up**	**Time**	**Speed-up**	**Time**	**Speed-up**
1	18.51		18.33		18.69		26.21	
2	10.48	1.77	10.48	1.75	10.54	1.77	14.77	1.77
4	5.44	3.40	5.49	3.34	5.58	3.35	7.64	3.43
8	3.17	5.84	3.16	5.80	3.17	5.89	4.36	6.02
16	1.96	9.44	1.97	9.29	1.97	9.47	3.30	7.95

Table 4: Computational time in seconds and Speed-up for CT Coronal image contaminated with Gaussian noise of different variances.

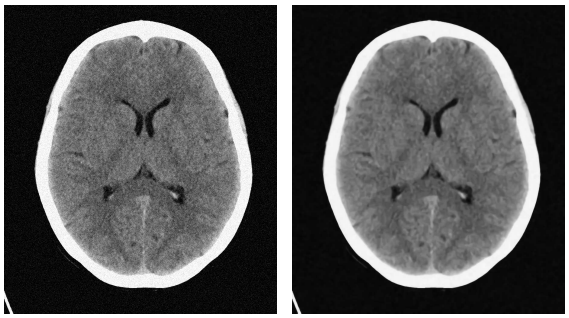
Figures 3, 4 and 5 present the filter outputs for the axial view CT image contaminated with different levels of Gaussian noise ($\sigma^2 = 0.005, 0.01, 0.02, 0.03$) for visual comparison. It can be observed the efficient denoising performance and the good conservation of the edges and contours of the image.

4 Conclusions

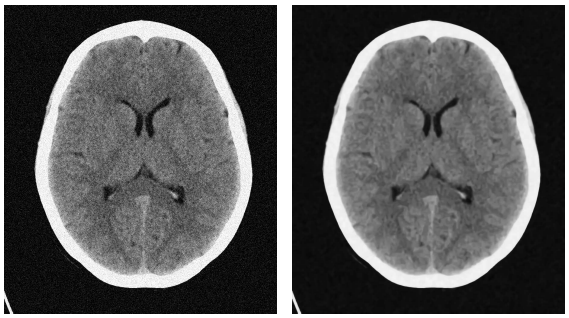
A parallel method to reduce Gaussian noise in CT images has been presented. This method combines, in two stages, the guided filter and the block-matching and 3D filtering method. The method has been implemented on multi-cores using OpenMP. The implementation has been used to reduce the Gaussian noise on real CT medical images from the from Radiopaedia database. Experiments show that the filter efficiently reduces noise and preserves image structures, obtaining competitive values in terms of the PSNR objective measure. The guided filter, used as a preprocessing stage in the proposed method, improves BM3D filtering by reducing high-frequency noise and preserving image structures. The parallel algorithm introduced demonstrated a significant speed-up, resulting in reduced computational times that make the new method appropriate for medical image processing. In future works, we will implement the algorithm on clusters of multicores using hybrid MPI-OpenMP programming and on GPUs using CUDA.



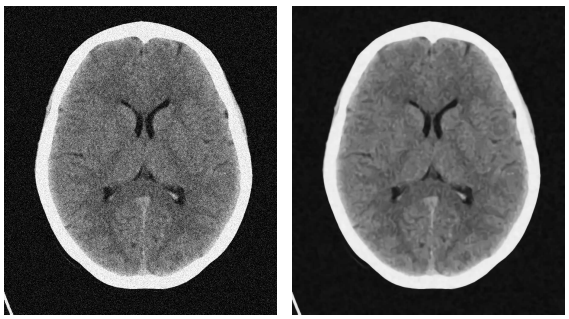
(a) Axial image (b) Corrupted $\sigma^2 = 0.005$ (c) Filtered $\sigma^2 = 0.005$



(d) Corrupted $\sigma^2 = 0.01$ (e) Filtered $\sigma^2 = 0.01$

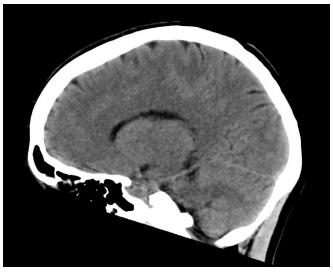


(f) Corrupted $\sigma^2 = 0.02$ (g) Filtered $\sigma^2 = 0.02$

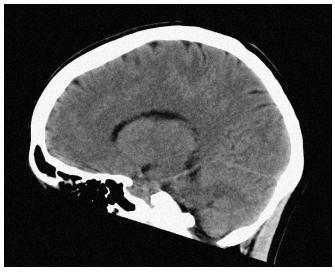


(h) Corrupted $\sigma^2 = 0.03$ (i) Filtered $\sigma^2 = 0.03$

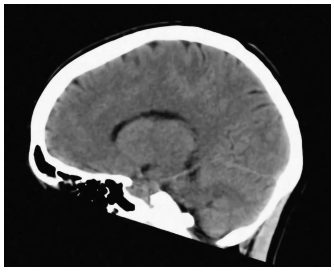
Figure 3: Filter outputs for visual comparison. Axial view contaminated with different levels of Gaussian noise.



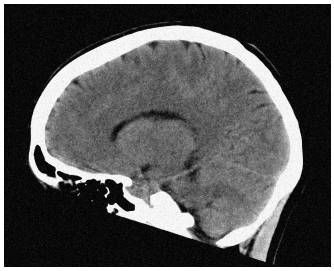
(a) Sagittal image



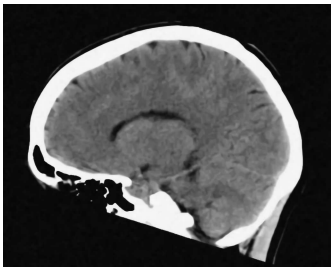
(b) Corrupted $\sigma^2 = 0.005$



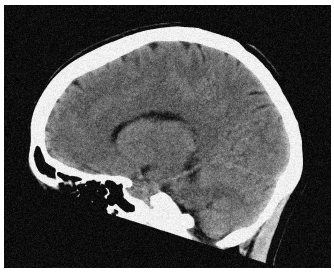
(c) Filtered $\sigma^2 = 0.005$



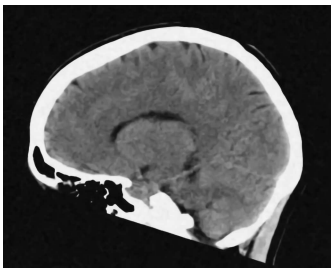
(d) Corrupted $\sigma^2 = 0.01$



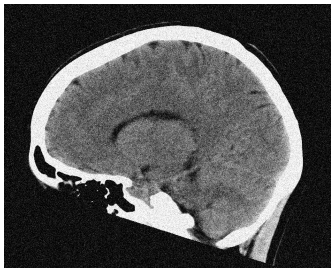
(e) Filtered $\sigma^2 = 0.01$



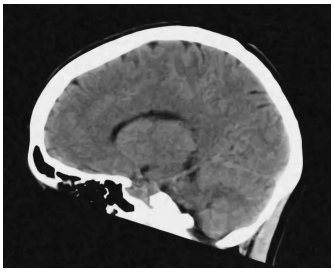
(f) Corrupted $\sigma^2 = 0.02$



(g) Filtered $\sigma^2 = 0.02$

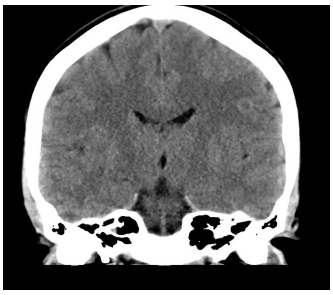


(h) Corrupted $\sigma^2 = 0.03$

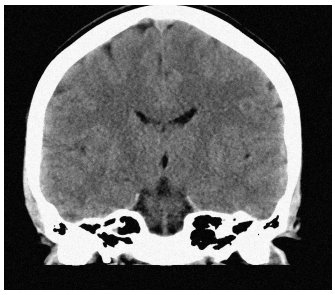


(i) Filtered $\sigma^2 = 0.03$

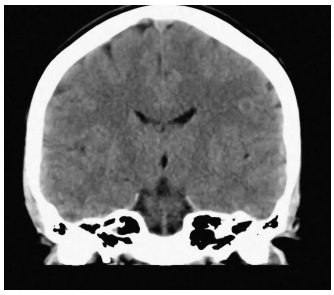
Figure 4: Filter outputs for visual comparison. Sagittal view contaminated with different levels of Gaussian noise.



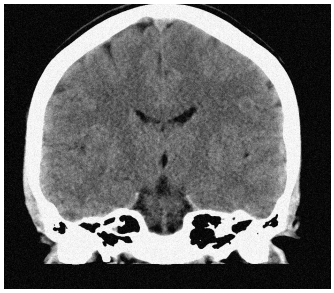
(a) Coronal image



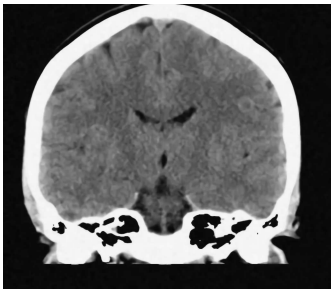
(b) Corrupted $\sigma^2 = 0.005$



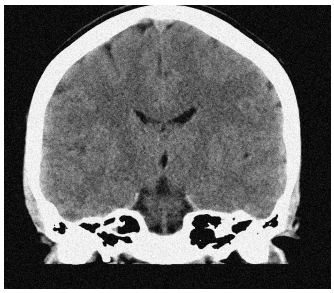
(c) Filtered $\sigma^2 = 0.005$



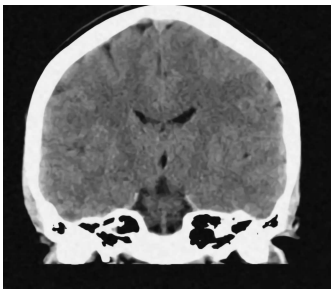
(d) Corrupted $\sigma^2 = 0.01$



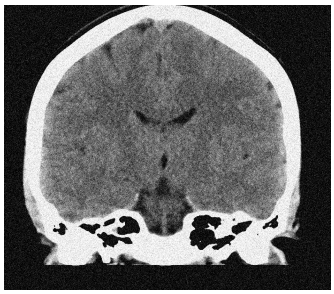
(e) Filtered $\sigma^2 = 0.01$



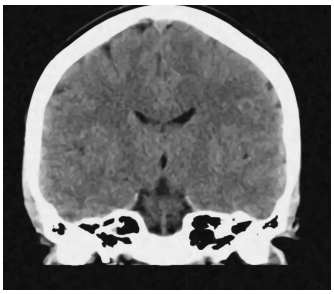
(f) Corrupted $\sigma^2 = 0.02$



(g) Filtered $\sigma^2 = 0.02$



(h) Corrupted $\sigma^2 = 0.03$



(i) Filtered $\sigma^2 = 0.03$

Figure 5: Filter outputs for visual comparison. Coronal view contaminated with different levels of Gaussian noise.

Acknowledgements

This research was supported by the Spanish Ministry of Science and Innovation (Grant PID2021-123627OB-C55) co-financed by FEDER funds and by the University of Alicante (Grants UAUSTI24-03, UADIF23-09 and VIGROB24-020).

References

[1] P. Gravel, G. Beaudoin, J.A. De Guise, "A method for modeling noise in medical images", *IEEE Trans. Med. Imaging*, 23(10): 1221–1232, 2004.

[2] T. Lei, W. Sewchand, "Statistical approach to X-ray CT imaging and its applications in image analysis. II. A new stochastic model-based image segmentation technique for X-ray CT image", *IEEE Trans. Med. Imaging*, 11(1): 62–69, 1992.

[3] K.N. Plataniotis, A.N. Venetsanopoulos, *Color image processing and applications*, Springer Science & Business Media, 2013.

[4] C. Tomasi, R. Manduchi, "Bilateral Filtering for Gray and Color Images", in *Proceedings of the Sixth International Conference on Computer Vision*, ICCV '98, pages 839–846. IEEE Computer Society, Washington, DC, USA, 1998.

[5] X. Li, "On modeling interchannel dependency for color image denoising", *Int. J. Imaging Syst. Technol.*, 17(3): 163–173, Oct. 2007.

[6] S. Schulte, B. Huysmans, A. Pižurica, E.E. Kerre, W. Philips, "A new fuzzy-based wavelet shrinkage image denoising technique", in *Proceedings of the 8th international conference on Advanced Concepts For Intelligent Vision Systems*, ACIVS'06, pages 12–23. Springer-Verlag, Berlin, Heidelberg, 2006.

[7] K. Dabov, A. Foi, V. Katkovnik, K. Egiazarian, "Image Denoising by Sparse 3-D Transform-Domain Collaborative Filtering", *IEEE Transactions on Image Processing*, 16(8): 2080–2095, 2007.

[8] K. He, J. Sun, X. Tang, "Guided image filtering", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(6): 1397–1409, 2013.

[9] L. Dagum, R. Menon, "OpenMP: An industry-standard API for shared-memory programming", *Computational Science & Engineering, IEEE*, 5(1): 46–55, 1998.

[10] R.C. Gonzalez, R.E. Woods, *Digital Image Processing*, Prentice Hall, Upper Saddle River, NJ, 2nd edition, 2002.