Implementation of the Perona–Malik Model for Image Despeckling Using a Physics-informed ConvNet

Haridarshan Kumar[1] **Sanjeev Kumar**[1,2]*

[1]**Department of Mathematics, Indian Institute of Technology Roorkee**
[2]**Mehta Family School of Data Science and Artificial Intelligence, Indian Institute of Technology Roorkee, Roorkee 247667, India**

Abstract

In this study, we propose a new data-driven algorithm for Perona-Malik image despeckling problem. The advantage of the proposed algorithm over neural network-based methods is that it does not need any noisy and clean image data pair for training. The proposed algorithm is implemented using a three-dimensional convolution neural network (ConvNet) architecture. We compare its output with results obtained from the finite difference method (FDM). To evaluate the performance of the proposed algorithm, simulations are carried out using grayscale images that have been artificially corrupted with different levels of speckle noise. Using the peak signal to noise ratio (PSNR) and structural similarity index measureas (SSIM) as the evaluation metric, we observed that the proposed algorithm outperforms the FDM, demonstrating superior image quality with the same numerical scheme and the same discretization.

1 Introduction

In various areas such as remote sensing, astronomy, and radiology, the quality of images is crucial for effective decision-making. However, these images are often degraded by various types of noise. The challenge of distinguishing between noise and fine details like texture, while preserving sharp edges, has made image denoising a crucial research problem. In this work, we focus on the removal of speckle noise, which is commonly observed in satellite imagery and ultrasound images, in the form of granular patterns. Speckle noise is multiplicative in nature i.e. the observed image $I : \Omega \subset \mathbb{R}^2 \to \mathbb{R}$, which is obtained after corruption of the original image J by speckle noise θ, is expressed as $I = J\theta$. Image despeckling is an inverse problem and hence appropriate regularization is needed to solve it.

PDE-based methods have gained significant attention due to their capability to capture both linear and nonlinear features of images. PDE models are broadly classified into linear and nonlinear categories. Linear models are effective for smoothing purposes, while nonlinear PDE models are utilized for edge preservation in addition to smoothing. Initially, Perona and Malik [1990] proposed a notable nonlinear PDE model for image denoising. Their model leverages the strengths of PDEs to effectively address the challenges of distinguishing noise and significant image details and preserving the edges while reducing noise. The model can be described by the following equations:

$$\frac{\partial u}{\partial t} = \nabla \cdot (d(|\nabla u|)\nabla u), \quad \text{in } (0,T] \times \Omega, \tag{1}$$

$$\frac{\partial u}{\partial n} = 0, \quad \text{on } (0,T] \times \partial\Omega, \tag{2}$$

$$u(0,z) = u_0(z), \quad \text{in } \Omega, \tag{3}$$

where $d(s) : (0,\infty) \to (0,\infty)$ represents the diffusion coefficient. They proposed two choices for $d(s)$ which are $d_1(s) = \frac{1}{1+\frac{s^2}{k^2}}$, and $d_2(s) = \exp\left(-\frac{s^2}{2k^2}\right)$ with thresholding constant k, u_0 denotes

*Corresponding author: sanjeev.kumar@ma.iitr.ac.in

2nd Conference on Applied AI and Scientific Machine Learning (CASML 2025).

the observed noisy image, and n indicates the outward normal direction at the boundary $\partial\Omega$. We will evaluate the efficiency of our algorithm on this model with $g(\cdot) := d_1(\cdot)$ as diffusion coefficient.

Finite difference method (FDM) is dominantly used to implement diffusion equation by the image processing community. However, sometimes FDM results in blurriness in the images. Raissi et al. [2019] introduced a groundbreaking approach for solving PDEs utilizing neural networks, termed physics-informed neural networks (PINNs). In PINNs, the PDEs are incorporated into the loss function of the neural network, and the network learns PDE by minimizing this loss. In recent years, numerous advancements and refinements of PINNs have been proposed, some of these include the works of Jagtap and Karniadakis [2020], Jagtap et al. [2020], Kharazmi et al. [2021], Wang et al. [2024]. Namaki et al. [2023] employed the PINNs approach for image denoising tasks and demonstrated superior results compared to finite difference methods. Their work specifically addresses additive Gaussian noise and highlights the potential of PINNs in this domain. While neural network-based methods such as PINNs offer promising results, one of their primary limitations remains the high computational cost associated with their training and implementation. To address these computational challenges, Shi et al. [2024] proposed a hybrid approach called physics-informed ConvNet (PICN), which combines the strength of finite difference method with PINNs. This approach aims to enhance computational efficiency while maintaining accuracy. In this study, we build upon the methodologies developed in the work of Shi et al. [2024] and explore the application of modified PICN to image despeckling problems. By integrating the principles of PINNs with traditional numerical methods, our goal is to leverage the advantages of both approaches to solve image despeckling tasks more effectively and efficiently. Our main contributions are summarized below:

- Shi et. al. originally developed the PICN method for either two-dimensional PDEs or one-dimensional time-dependent PDEs. In our work, we extend the PICN methodology to address two-dimensional time-dependent PDEs.
- We developed a computational algorithm using our modified PICN and explored its applications for image despeckling problems whereas most of the PDE-based approaches are for additive noise removal models.

2 Physics-informed ConvNet

Traditionally, PINNs employ feed-forward neural networks (FNNs) and hence it often encounter challenges during the training process. To address these issues, Shi et al. [2024] proposed the adoption of a convolutional neural network in place of the FNN within the PINNs framework. The PICN architecture consists of three primary components: the de-convolutional layer, pre-trained convolutional layers for derivative calculation, and an interpolation network for processing irregular domains. In the first component, a constant scalar is input into the convolutional neural network, where the deconvolutional layer results $y_h \in \mathbb{R}^{N\times P}$. Subsequently, the convolutional layer uses y_h to predict the field $\hat{u} \in \mathbb{R}^{N\times P}$.

The second component utilizes the finite difference method instead of automatic differentiation (AD) for derivative computation. Specifically, finite difference-based filters are used to approximate the derivatives of the generated field. These pre-trained filters are fixed and are not trained during the training process and by applying these pre-trained filters, the PDE residuals are calculated which is a must for the solution. The third component addresses the cases of irregular non-rectangular domains. Since the boundary points of the irregular domain may not align precisely with the uniform grid within the rectangular domain, the interpolation network is designed to approximate the physical field values along the boundary of the irregular domain.

In the PICN framework, solving the PDE is equivalent to training the convolutional neural network to minimize the total loss. This total loss is comprised of two components: the PDE loss, which ensures the agreement with the governing equation and is calculated using finite difference based filters, and the data loss, which enforces the initial and boundary conditions of the equation.

3 Proposed Despeckling Algorithm

We propose a data-driven computational algorithm for Perona-Malik model. However, our algorithm can be easily implemented with other PDE-based denoising models by simply replacing the PDE.

First of all, we propose the modified PICN approach for two-dimensional time dependent PDE. Then we develop our despeckling algorithm based on it.

3.1 Modified Physics-informed ConvNet

3.1.1 Architecture

Two major modifications are introduced in PICN architecture. First, we utilize 3D convolution layers instead of 2D convolution layers since Perona-Malik model is three-dimensional. The second modifications involve the change in procedure for physical field generation in the first component of PICN. In our modified PICN, one deconvolution operation, three convolution operations and one dropout operation are performed to generate the field. Deconvolution operation expands the spatial and temporal resolution of the input, preparing it for further processing. Convolution operations are considered to incorporate learnable parameters in the network, increasing its ability to effectively capture hidden patterns in the image. Dropout operation randomly disables a fraction of neurons during training. It serves as a regularization technique designed to prevent overfitting and hence improves the generalization ability of the network. While drop-out offers various advantages but excessive use of it increases the training time. These operations are mathematically summarized as follows:

$$y_{hk} = \sigma_k \left(W_{hk} \cdot y_{hk-1} + b_{hk} \right), \quad k = 1, 2, 4, 5,$$

with y_{h3} as the result of dropout layer. The identity maps and hyperbolic tangent functions ($\tanh$) are used as activation functions for σ_1, σ_5 and σ_2, σ_4 respectively.
$W_{hk} = [W^k_{i,j}] \in \mathbb{R}^{M \times N \times P}$ is the weight matrix, and $b_{hk} \in \mathbb{R}$ is the bias term.

3.1.2 Solving Perona-Malik PDE

In a data-driven approach to the solution of PDEs, the objective is to construct a neural network that serves as a surrogate for the solution. Our proposed network is based on a ConvNet architecture, denoted as $f_\theta(a)$. It takes the scalar input a, and applies deconvolution and convolution filters to generate an output tensor $\hat{u}$ of shape $M \times N \times P$. Here, N and P denote the number of spatial discretization points in the x and y directions, respectively, while M corresponds to the temporal discretization. The parameters θ consist of weights of these learnable filters. The loss function $\mathcal{L}$ of the network is given by:

$$\mathcal{L} = w_1 \mathcal{L}_1 + w_2 \mathcal{L}_2 + w_3 \mathcal{L}_3,$$

with

$$\mathcal{L}_1 = \frac{1}{n_1} \left\| \frac{\partial \hat{u}}{\partial t} - g \left(\frac{\partial^2 \hat{u}}{\partial x^2} + \frac{\partial^2 \hat{u}}{\partial y^2} \right) - \left(\frac{\partial g}{\partial x} \frac{\partial \hat{u}}{\partial x} + \frac{\partial g}{\partial y} \frac{\partial \hat{u}}{\partial y} \right) \right\|^2_{L^2([0,T] \times \Omega)},$$

$$\mathcal{L}_2 = \frac{1}{n_2} \left\| \frac{\partial \hat{u}}{\partial n} \right\|^2_{L^2([0,T] \times \partial\Omega)},$$

$$\mathcal{L}_3 = \frac{1}{n_3} \left\| \hat{u}(0, \cdot) - u_0 \right\|^2_{L^2(\Omega)}.$$

We consider $w_1 = w_2 = w_3 = 1$ for simplicity. However, other values for these weights can also be assigned. $g \equiv d_1$ is the diffusion coefficient of Perona-Malik model and u_0 represents the initial noisy image. $\hat{u}$ is calculated by taking the input scalar $a = 1$. Values $n_1, n_2, n_3 \leq MNP$ denote the number of points utilized in the calculations for PDE loss, boundary loss, and initial data loss. The network learns to approximate the solution $u(t, z)$ by searching optimal parameters θ in order to minimize the loss $\mathcal{L}$. This optimization process is carried out using gradient-based optimizers which calculate the gradients of the network with respect to θ using backpropagation. $\mathcal{L}_1$ is calculated using the pre-trained finite difference based filters. We use the forward finite difference scheme and central finite difference scheme for first-order derivative filters and second-order derivative filters respectively. Specifically, the derivative formulations used for filter design are described below:

$$\nabla_x u^{(n)}_{i,j} \approx \frac{u^{(n)}_{i+1,j} - u^{(n)}_{i,j}}{\Delta x}, \quad \nabla_y u^{(n)}_{i,j} \approx \frac{u^{(n)}_{i,j+1} - u^{(n)}_{i,j}}{\Delta y}, \quad \frac{\partial u^{(n)}_{i,j}}{\partial t} \approx \frac{u^{(n+1)}_{i,j} - u^{(n)}_{i,j}}{\Delta t},$$

$$\Delta_x u^{(n)}_{i,j} \approx \frac{u^{(n)}_{i+1,j} - 2u^{(n)}_{i,j} + u^{(n)}_{i-1,j}}{\Delta x^2}, \quad \Delta_y u^{(n)}_{i,j} \approx \frac{u^{(n)}_{i,j+1} - 2u^{(n)}_{i,j} + I^{(n)}_{i,j-1}}{\Delta y^2}.$$

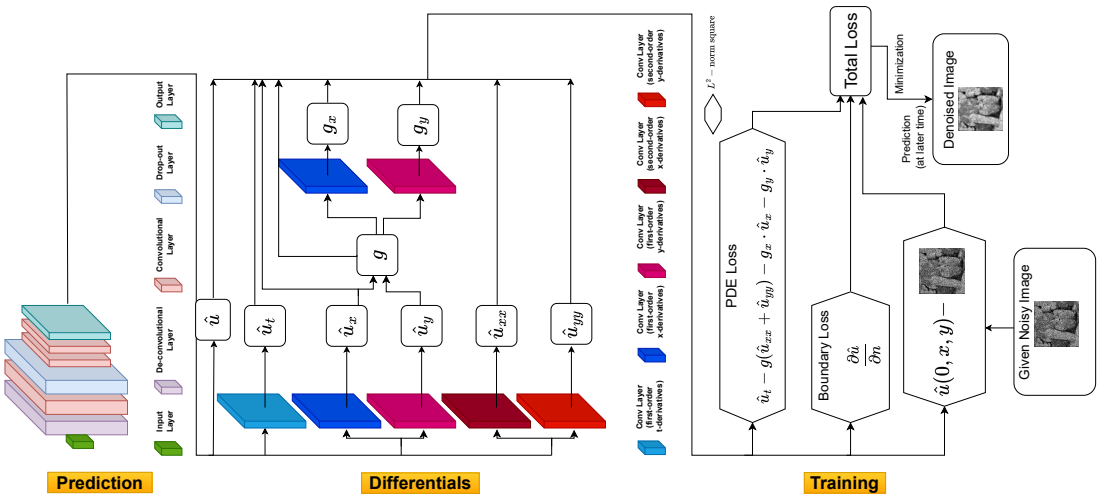


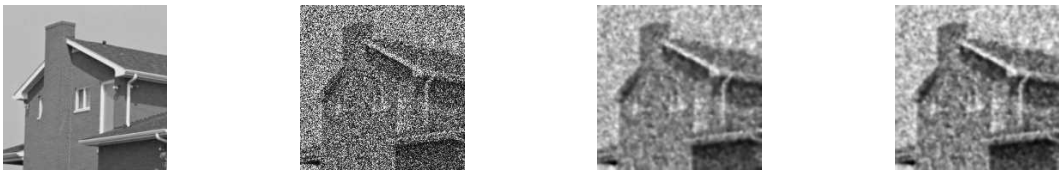
Figure 1: Image despeckling algorithm with modified PICN

In our case, the spatial domain has a rectangular shape and hence we have a regular (cuboidal) domain for the computation. Once the neural network is trained, the predicted solution at a later time results our despeckled image. The complete architecture of our despeckling algorithm is shown in Figure 1.

4 Results and Discussions

We perform our experiments on grayscale images[2][3]. We have degraded the original image artificially by speckle noise generated using Gamma distribution with varying numbers of "look" $L = \{1, 3, 5, 10\}$. The lower number of looks signifies higher noise. Then, speckle noise is multiplicatively applied to the original image to yield the noisy version of the images. To evaluate the performance of despeckling/denoising algorithms, two quantitative metrics viz. PSNR and SSIM are employed. It quantifies the quality of a denoised image by comparing it with the original noise-free image. Higher PSNR and SSIM values are desirable, as they signify improved denoising quality.

In this study, we do not focus on exploring different discretization schemes or thresholding parameters within Perona-Malik model to achieve better denoising results. Instead, our novelty lies in demonstrating the superiority of our computation algorithm over different methods with the same discretization (if applicable). Hence we have considered fixed time step $\Delta t = 0.15$ and threshold $k = 0.15$ throughout our experiment.



(a) Original (b) Noisy (L=1) (c) FDM (d) PICN-I

Figure 2: Comparison of denoising effect of FDM with PICN-I.

In this subsection, we compare the despeckling performance of our algorithm with the finite difference method, and physics-informed neural networks. Comparison of the results of our method with that of FDM is presented in Table 1 and Figure 2. Due to the high computational cost associated with PINNs, we conduct the comparison with a 91×91-sized Chemical Plant image. The corresponding results

[2] https://www.kaggle.com/datasets/leweihua/set12-231008

[3] https://sipi.usc.edu/database/

are displayed in Table 2. In the PINNs setup, training is performed with 30K collocation points using the Adam optimizer, a learning rate of 0.001, and the activation function tanh. It is observed that PICN significantly outperforms PINNs both in terms of computational efficiency and despeckling performance. Our algorithm is implemented using a learning rate of 0.001, tanh activation, Adam optimizer, and 5000 training iterations. We report results for dropout rates of 0.2 and 0.3, referred to as PICN-I (sometimes abbreviated as PICN) and PICN-II, respectively.

Table 1: PSNR / SSIM results comparison.

Image	**L**	**PSNR/SSIM**			
		Noisy Image	**FDM**	**PICN-I**	**PICN-II**
Aeroplane	1	8.00 / 0.0783	13.20 / 0.4249	13.48 / **0.4272**	**13.68** / 0.4260
	3	11.08 / 0.1476	17.80 / 0.4957	**17.94 / 0.5054**	17.76 / 0.4994
	5	12.67 / 0.1882	19.64 / 0.5193	19.81 / 0.5248	**20.18 / 0.5254**
	10	14.89 / 0.2513	21.59 / 0.5644	**22.01 / 0.5755**	21.73 / 0.5646
Cameraman	1	9.93 / 0.1634	17.08 / 0.4646	**17.81 / 0.4752**	17.67 / 0.4654
	3	12.64 / 0.2589	20.86 / 0.5433	21.40 / 0.5464	**21.49 / 0.5494**
	5	14.12 / 0.3070	21.93 / 0.5806	**22.43** / 0.5822	22.38 / **0.5837**
	10	16.36 / 0.3715	23.51 / 0.5985	**23.63** / 0.6035	23.58 / **0.6051**
House	1	9.14 / 0.0533	16.93 / 0.4781	**17.40** / 0.4820	17.18 / **0.4841**
	3	12.02 / 0.1041	22.04 / 0.5764	22.25 / **0.5826**	**22.72** / 0.5801
	5	13.56 / 0.1397	23.95 / 0.6054	24.08 / **0.6128**	**24.18** / 0.6125
	10	15.96 / 0.2029 6	25.73 / 0.6852	**25.75 / 0.6864**	25.70 / 0.6791
Starfish	1	9.59 / 0.0998	16.05 / 0.4964	16.40 / **0.4975**	**16.60** / 0.4835
	3	12.66 / 0.2114	20.15 / 0.6284	20.47 / **0.6286**	**20.74** / 0.6203
	5	14.29 / 0.2817	21.64 / **0.6743**	**21.92** / 0.6708	21.67 / 0.6630
	10	16.77 / 0.4026	23.55 / 0.7390	**23.79 / 0.7394**	22.66 / 0.7219
Parrot	1	10.20 / 0.1753	16.62 / 0.5091	**16.96 / 0.5138**	16.81 / 0.5112
	3	13.06 / 0.2871	19.75 / 0.6236	**20.27** / 0.6241	20.02 / **0.6243**
	5	14.66 / 0.3454	21.58 / 0.6373	**21.67 / 0.6393**	21.62 / 0.6375
	10	17.03 / 0.4293	23.01 / **0.7016**	**23.11** / 0.6966	23.06 / 0.6817

Table 2: Comparison of PINNs and PICN methods on image despeckling performance.

Method	**PSNR**	**SSIM**	**Training Time (s)**
PINNs	23.41	0.6681	2503.48
PICN	24.34	0.7106	179.02
Noisy Image	17.23	0.4194	–

5 Conclusions

This paper presents a data-driven computational algorithm for image despeckling using a convolutional neural network architecture. Experimental results demonstrate that it offers superior despeckling compared to FDM under the same discretization and numerical scheme. Moreover, our algorithm succeeds in removing the blurriness which is commonly observed in FDM results. It also provides good results even with a small number of iterations, which is generally missing in neural network-based methods. An additional advantage of the proposed method is that its performance can be further enhanced by considering improved discretization techniques and more advanced numerical schemes.

6 Acknowledgements

First author is very grateful to the Ministry of Education, Government of India, for the financial support under the Prime Minister's Research Fellowship (PMRF) scheme [PMRF ID: 2803603].

References

Shengze Cai, Zhiping Mao, Zhicheng Wang, Minglang Yin, and George Em Karniadakis. Physics-informed neural networks (pinns) for fluid mechanics: A review. *Acta Mechanica Sinica*, 37(12):1727–1738, 2021.

MR Eslahchi, Sakine Esmaili, Neda Namaki, and Rezvan Salehi. Application of finite difference method in solving a second-and fourth-order pde blending denoising model. *Mathematical Sciences*, 17:93–106, 2021.

Robert Eymard, Thierry Gallouët, and Raphaèle Herbin. Finite volume methods. *Handbook of Numerical Analysis*, 7:713–1018, 2000.

Joseph W Goodman. Some fundamental properties of speckle. *Journal of the Optical Society of America*, 66 (11):1145–1150, 1976.

Ameya D Jagtap and George Em Karniadakis. Extended physics-informed neural networks (xpinns): A generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations. *Communications in Computational Physics*, 28(5):2002–2041, 2020.

Ameya D Jagtap, Ehsan Kharazmi, and George Em Karniadakis. Conservative physics-informed neural networks on discrete domains for conservation laws: Applications to forward and inverse problems. *Computer Methods in Applied Mechanics and Engineering*, 365:113028, 2020.

Ehsan Kharazmi, Zhongqiang Zhang, and George Em Karniadakis. hp-vpinns: Variational physics-informed neural networks with domain decomposition. *Computer Methods in Applied Mechanics and Engineering*, 374:113547, 2021.

Zuzana Krivá and Karol Mikula. An adaptive finite volume scheme for solving nonlinear diffusion equations in image processing. *Journal of Visual Communication and Image Representation*, 13(1-2):22–35, 2002.

John Martin and Hanspeter Schaub. Physics-informed neural networks for gravity field modeling of the earth and moon. *Celestial Mechanics and Dynamical Astronomy*, 134(2):13, 2022.

Neda Namaki, MR Eslahchi, and Rezvan Salehi. The use of physics-informed neural network approach to image restoration via nonlinear pde tools. *Computers & Mathematics with Applications*, 152:355–363, 2023.

Pietro Perona and Jitendra Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(7):629–639, 1990.

Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.

Pengpeng Shi, Zhi Zeng, and Tianshou Liang. Physics-informed convnet: Learning physical field from a shallow neural network. *Communications in Nonlinear Science and Numerical Simulation*, 132:107911, 2024.

Gordon D Smith. *Numerical solution of partial differential equations: Finite difference methods*. Oxford University Press, 1985.

Endre Süli. *Finite element methods for partial differential equations*. Oxford University Computing Laboratory, 2002.

Sifan Wang, Shyam Sankaran, and Paris Perdikaris. Respecting causality for training physics-informed neural networks. *Computer Methods in Applied Mechanics and Engineering*, 421:116813, 2024.

J Weickert. *Anisotropic diffusion in image processing*. Teubner, 1998.