Dynamically Unfolding Recurrent Restorer: A Moving Endpoint Control Method for Image Restoration

Xiaoshuai Zhang*
Institute of Computer Science and Technology,
Peking University
jet@pku.edu.cn

Yiping Lu*
School Of Mathmatical Science,
Peking university
luyiping9712@pku.edu.cn

Jiaying Liu
Institute of Computer Science and Technology,
Peking University
liujiaying@pku.edu.cn

Bin Dong
Beijing International Center for Mathematical Research, Peking University
Center for Data Science, Peking University
Beijing Institute of Big Data Research,
Beijing, China
dongbin@math.pku.edu.cn

Abstract

In this paper, we propose a new control framework called the moving endpoint control to restore images corrupted by different degradation levels in one model. The proposed control problem contains a restoration dynamics which is modeled by an RNN. The moving endpoint, which is essentially the terminal time of the associated dynamics, is determined by a policy network. We call the proposed model the dynamically unfolding recurrent restorer (DURR). Numerical experiments show that DURR is able to achieve state-of-the-art performances on blind image denoising and JPEG image deblocking. Furthermore, DURR can well generalize to images with higher degradation levels that are not included in the training stage.

1 Introduction

Image restoration, including image denoising, deblurring, inpainting, *etc.*, is one of the most important areas in imaging science. Its major purpose is to obtain high quality reconstructions of images corrupted in various ways during imaging, acquisiting, and storing, and enable us to see crucial but subtle objects that reside in the images. The image restoration problem can be generally formulated as an inverse problem:

$$f = Au + \eta,$$

where A is some operator (linear or nonlinear) and η denotes the additive noise whose statistics depends on the imaging modality. Typically, the noise is assumed to be Gaussian white noise. Different image restoration problem corresponds to a different type of A, e.g., the identity operator for image denoising, a restriction operator for image inpainting, a convolution operator for image

*Equal contribution.

Preprint. Work in progress.

deblurring, a quantization of a spatially varied averaging operator for JPEG image deblocking, *etc.* In this paper, we mainly focus on image deonising and JPEG image deblocking, though our approach can be naturally extended to other image restoration tasks.

Image restoration has been an active research area. Numerous models and algorithms have been developed for the past few decades. Before the uprise of deep learning methods, there were two classes of image restoration approaches that were widely adopted in the field: transformation based approach and PDE approach. Transformation based approach includes wavelet and wavelet frame based methods [15, 36, 12, 4, 45], dictionary learning based methods [1, 7, 38], similarity based methods [3, 11], low-rank models [22, 19], etc. PDE approach includes variational models [30, 34, 2], nonlinear diffusions [32, 8, 40], nonlinear hyperbolic equations [31], etc. More recently, deep connections between wavelet frame based methods and PDE approach were established [5, 6, 13].

One of the greatest challenge for image restoration is to properly handle image degradations of different levels. In the existing transformation based or PDE based methods, there is always at least one tuning parameter (e.g. the regularization parameter for variational models and terminal time for nonlinear diffusions) that needs to be manually selected. The choice of the parameter heavily relies on the degradation level.

Recent years, deep learning models for image restoration tasks have significantly advanced the state-of-the-art of the field. [21] proposed a convolutional neural network (CNN) for image denoising which has better expressive power than the MRF models [23]. Inspired by nonlinear diffusions, [10] designed a deep neural network for image denoising and [42] improves the capacity by introducing a 20-layer neural network with residual connections. [39] introduced a deep network with long term memory which was inspired by neural science. However, these models cannot gracefully handle images with varied degradation levels. Although one may train different models for images with different levels, this may limit the application of these models in practice due to lack of flexibility.

Taking image denoising for example, to the best of the authors' knowledge, the only deep learning models attempted to tackle the blind denoising problem were proposed by [42, 26]. [42] designed a 20-layer neural network, called DnCNN-B, which had a huge number of parameters. To reduce number of parameters, [26] proposed two networks, called the UNet_5 and UNLNet_5, by unrolling a projection gradient algorithm for a constrained optimization model. However, [26] also observed a drop in PSNR comparing to DnCNN. Therefore, the design of a light-weighted and yet effective and robust model for blind image denoising remains a challenge. Moreover, deep learning based models trained on simulated gaussian noise images usually fail to handle real world noise, as will be illustrated in later sections.

JPEG is one of the most commonly used standards among the lossy image compression methods. However, as the compression rate increases, this method tend to introduce undesirable artifacts. JPEG deblocking aims to eliminate the artifact and improve the image quality. Recently, deep learning based methods were proposed for JPEG deblocking [14, 42, 43]. However, most of their models are trained and evaluated on a given quality factor except for [42].

Our proposed model for image restoration is inspired by the recent development on the relation between deep learning and optimal control. The relation between supervised deep learning methods and optimal control has been discovered and exploited by [41, 28, 9, 17]. The key idea is to consider the residual block $x_{n+1} = x_n + f(x_n)$ as an approximation to the continuous dynamics $\dot{X} = f(X)$. In particular, [28, 17] demonstrated that the training process of a class of deep models (e.g. ResNet[20], PolyNet[44], etc.) can be understood as solving the following control problem:

$$\begin{aligned} &\min_{w} \left(L(X(T), y) + \int_0^\tau R(w(t), t) dt \right) \\ &s.t.\ \dot{X} = f(X(t), w(t)),\ X(0) = x_0. \end{aligned}$$

Here x_0 is the input, y is the regression target or label, $\dot{X} = f(X, w)$ is the deep neural network with parameter $w(t)$, R is the regularization term and L can be any loss function to measure the difference between the reconstructed images and the ground truths.

In this paper, we propose a novel moving endpoint control model for image restoration with state-of-the-art performance. We first cast the model in the continuum setting. Let x_0 be an observed degraded

image and y be its corresponding damage-free counterpart. We want to learn a time-independent dynamic system $\dot{X} = f(X(t), w)$ with parameters w so that $X(0) = x_0$ and $X(\tau) \approx y$ for some $\tau > 0$. See Fig. 1 for an illustration of our idea. The reason that we do not require $X(\tau) = y$ is to avoid over-fitting. For varied degradation levels and different images, the optimal terminal time τ of the dynamics may vary. Therefore, we need to include the variable τ in the learning process as well. The learning of the dynamic system and the terminal time can be gracefully casted as the following moving endpoint control problem:

$$\begin{aligned} &\min_{w,\tau} L(X(\tau), y) + \int_0^\tau R(w(t), t)dt \\ &s.t.\ \dot{X} = f(X(t), w(t)), t \in (0, \tau) \\ &\quad X(0) = x_0. \end{aligned} \tag{1}$$

Different from the previous control problem, in our model the terminal time τ is also a parameter to be optimized. The dynamic system $\dot{X} = f(X(t), w)$ is modeled by a recurrent neural network (RNN) with a residual connection, which can be understood as a residual network with shared weights [27]. We shall refer to this RNN as the *restoration unit*. In order to learn the terminal time of the dynamics, we adopt a *policy network* to adaptively determine an optimal stopping time. Our learning framework is demonstrated in Fig. 2. We note that the above moving endpoint control problem can be regarded as the penalized version of the well-known fixed endpoint control problem in optimal control [16], where instead of penalizing the difference between $X(\tau)$ and y, the constraint $X(\tau) = y$ is strictly enforced.

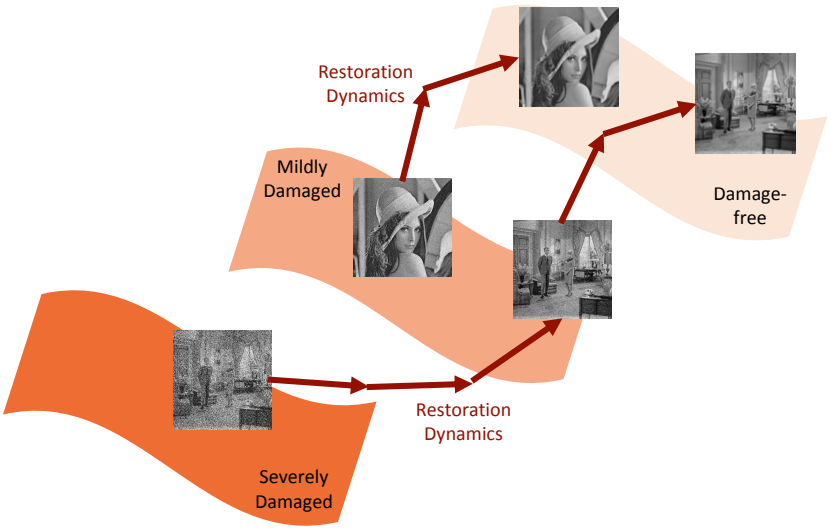


Figure 1: The proposed moving endpoint control model: evolving a learned reconstruction dynamics and ending at high-quality images.

2 Method

We start with a discretization of the moving endpoint problem (1) on the dataset $\{(x_i, y_i)|i = 1, 2, \cdots, d\}$, where $\{x_i\}$ are degraded observations of the damage-free images $\{y_i\}$. The discrete moving endpoint control problem is given as follows:

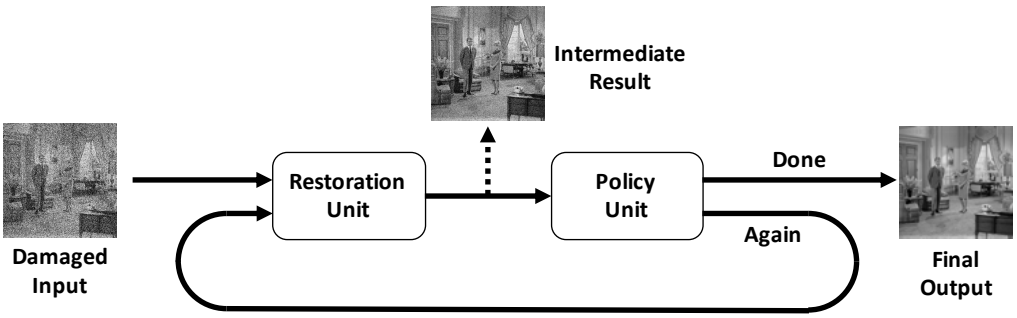


Figure 2: Pipeline of the dynamically unfolding recurrent restorer (DURR).

$$\begin{aligned} \min_{w,\{N_i\}_{i=1}^d} \; & R(w) + \lambda \sum_{i=1}^{d} L(X_{N_i}^i, y_i) \\ s.t. \; & X_n^i = X_{n-1}^i + \Delta t f(X_{n-1}^i, w), n = 1, 2, \cdots, N_i, (i = 1, 2, \cdots, d) \\ & X_0^i = x_i, i = 1, 2, \cdots, d. \end{aligned} \tag{2}$$

Here, $X_n^i = X_{n-1}^i + \Delta t f(X_{n-1}^i, w)$ is the forward Euler approximation of the dynamics $\dot{X} = f(X(t), w)$ in (1). The terminal time $\{N_i\}$ is determined by a policy network $P(x, \theta)$, where x is the output of the restoration unit at each iteration and θ the weight. In other words, the role of the policy network is to stop the iteration of the restoration unit when an ideal image restoration result is achieved. The reward function of the policy unit is naturally defined by

$$r(\{X_n^i\}) = \begin{cases} \lambda \left(L(x_{n-1}, y_i) - L(x_n, y_i) \right) & \text{If choose to continue} \\ 0 & \text{Otherwise} \end{cases}$$

In order to solve the problem (2), we need to optimize two networks simultaneously, i.e. the restoration unit and the policy unit. The first is an restoration unit which approximates the controlled dynamics and the other is the policy unit to give the optimized terminating conditions. The objective function we use to optimize the policy network can be written as

$$J = \mathbb{E}_{X \sim \pi_\theta} \sum_{n}^{N_i} [r(\{X_n^i, w\})],$$

where π_θ denotes the distribution of the trajectories $X = \{X_n^i, n = 1, \ldots, N_i, i = 1, \ldots, d\}$ under the policy network $P(\cdot, \theta)$.

In order to train model (2), a restoration unit is trained to approximate the controlled dynamics first. Then the restoration unit will be fixed, and the policy unit is optimized by the policy gradient algorithm [37], where the gradient is calculated as below:

$$\begin{aligned} \nabla_\theta J &= \nabla_\theta \mathbb{E}_{u \sim \pi_\theta} \sum_{n=1}^{N_i} [r(\{X_n^i, w_n\})] \\ &= \nabla_\theta \mathbb{E}_{u \sim \pi_\theta} \sum_{n=1}^{N_i} [r(\{X_n^i, w_n\}) \nabla_\theta \log \pi_\theta(w, N_i | x)] \\ &= \nabla_\theta \mathbb{E}_{u \sim \pi_\theta} \left[\left(\sum_{n=1}^{N_i} r(\{X_n^i, w_n\}) \right) \left(\nabla_\theta \sum_{n=1}^{N_i} \log P(X_n^i, \theta) \right) \right] \end{aligned}$$

Algorithm 1 Dynamically Unfolding Recurrent Restorer (DURR) Training via Policy Gradient

Input: The target y_i and noisy observation x_i

1: Initialize the weights of the restoration unit and the policy unit.
2: Pretrain the restoration unit with defined policies.
3: Set epochs M and the hyper-parameters in the algorithm.
4: **for** $t \leftarrow 1$ to M **do**
5: Fix the restoration unit and simulate the forward trajectories using π_θ
6: Calculate the policy gradient and then perform the optimization:

$$\theta \leftarrow \theta - \eta \nabla_\theta \mathbb{E}_{X\sim\pi_\theta}\left[\left(\sum_{n=1}^{N_i} r(\{X_n^i, w\})\right)\left(\nabla_\theta \sum_{n=1}^{N_i} \log P(X_n^i, \theta)\right)\right]$$

. The expectation here is estimated on the sampled trajectory.

The entire algorithm is summarized in Algorithm 1.

We finally note that the use of the policy network to determine the terminal time $\{N_i\}$ is a natural choice given the form of the problem (2). However, one may use other techniques for the same task. For example, one may directly train a separate network to determine the stopping criterion for the restoration unit.

3 Experiments

3.1 Experiment Settings

In all denoising experiments, we follow the same settings as in [10, 42, 26]. All models are evaluated using the mean PSNR as the quantitative metric on the BSD68 dataset [29]. 432 images in the BSD500 [29] (*i.e.* BSD500 excludes BSD68) are used for training. Both the training and evaluation process is done on gray-scale images.

The restoration unit is a simple U-Net [33] style fully convolutional neural network. For the training process of the restoration unit, the noise levels of 25, 35, 45 and 55 are used. Images are cut into 64×64 patches, and the batch-size is set to 24. The Adam optimizer with the learning rate 1e-3 is adopted and the learning rate is scaled down by a factor of 10 on training plateaux.

The policy unit is composed of two ResUnit and an LSTM cell. For the policy gradient training, we reward the network with value 1 when the PSNR gain is larger than 0.01, with value -1 when the gain is less than -0.01. For training the policy unit, an RMSprop optimizer with learning rate 1e-5 is adopted. We've also tested other network structures, these tests and the detailed network structures of our model are demonstrated in the appendix.

In all JPEG deblocking experiments, we follow the settings as in [42, 43]. All models are evaluated using the mean PSNR as the quantitative metric on the LIVE1 dataset [35]. The training set and testing set of BSD500 are used for training. Both the training and evaluation processes are done on the Y channel (the luminance channel) of the YCbCr color space. The images with quality factors 20 and 30 are used during the training process of the restoration unit. All other parameter settings are the same as in the denoising experiments.

3.2 Training the Restoration Unit

If the terminal time τ for every input x_i is given (*i.e.* given a certain policy), the restoration unit can be optimized accordingly. We would like to show in this section that the policy used during training greatly influence the performace and the generalization ability of the restoration unit. More specifically, a restoration unit can be better trained by a good policy.

The simplest policy is to fix the loop time τ as a constant for every input. We name such policy as "naive policy". A more reasonable policy is to manually assign an unfolding time for each degradation level during training. We shall call this policy the "refined policy". Since we have not trained the

policy unit yet, to evaluate the performance of the trained restoration units, we manually pick the output image with the highest PSNR (*i.e.* the peak PSNR).

We take denoising as an example here. The peak PSNRs of the restoration unit trained with different policies are listed in Table. 1 . The Fig. 3 illustrates the average loop times when the peak PSNRs appear. The training is done on both single noise level ($\sigma = 40$) and multiple noise levels ($\sigma = 35, 45$). For the refined policy, the noise level and the associated loop times are (35, 6), (45, 9). For the naive policy, we always fix the loop times to 8.

Table 1: Average peak PSNR on BSD68 with different training strategies.

Strategy		Noise Level						
Training Noise	Policy	25	30	35	40	45	50	55
40	Naive	28.61	28.13	27.62	**27.19**	26.57	26.17	24.00
35, 45	Naive	27.74	27.17	26.66	26.24	26.75	25.61	24.75
35, 45	Refined	**29.14**	**28.33**	**27.67**	**27.19**	**27.69**	**26.61**	**25.88**

As we can see, the refined policy brings the best performance on all the noise levels including 40. The restoration unit trained for specific noise level (*i.e.* $\sigma = 40$) is only comparable to the one with refined policy on noise level 40. The restoration unit trained on multiple noise levels with naive policy has the worst performance.

These results indicate that the restoration unit has the potential to generalize on unseen degradation levels when trained with good policies. According to Fig. 3, the generalization reflects on the loop times of the restoration unit. It can be observed that the model with steeper slopes have stronger ability to generalize as well as better performances.

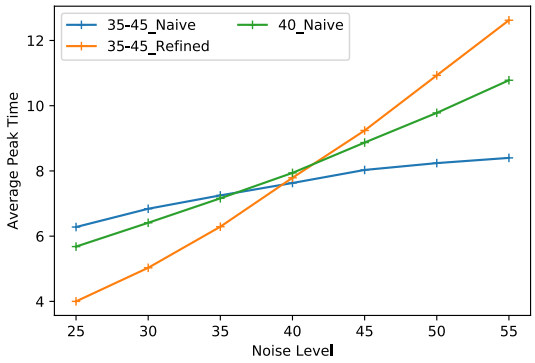


Figure 3: Average peak time on BSD68 with different training strategies.

According to these results, the restoration unit we used in DURR is trained using the refined policy. More specifically, for image denoising, the noise level and the associated loop times are set to (25, 4), (35, 6), (45, 9), and (55, 12). For JPEG image deblocking, the quality factor (QF) and the associated loop times are set to (20, 6) and (30, 4).

3.3 The Complete DURR

After training the restoration unit, the policy unit is trained using the policy gradient algorithm stated above until full convergence. Then the two units are combined to form the complete DURR model.

3.3.1 Image Denoising

We select DnCNN-B[42] and UNLNet$_5$ [26] for comparisons since these models are designed for blind image denoising. Moreover, we also compare our model with non-learning-based algorithms BM3D [11] and WNMM [19]. The noise levels are assumed known for BM3D and WNMM due to their requirements. Comparison results are shown in Table 2.

Despite the fact that the parameters of our model (1.8×10^5 for the restoration unit and 1.0×10^5 for the policy unit) is less than the DnCNN (approximately 7.0×10^5), one can see that DURR outperforms DnCNN on most of the noise-levels. More interestingly, DURR does not degrade too much when the the noise level goes beyond the level we used during training. The noise level $\sigma = 65, 75$ is not included in the training set of both DnCNN and DURR. DnCNN reports notable

drops of PSNR when evaluated on the images with such noise levels, while DURR only reports small drops of PSNR (see the last row of Table 2 and Fig. 5). Note that the reason we do not provide the results of UNLNet$_5$ in Table 2 is because the authors of [26] has not released their codes yet, and they only reported the noise levels from 15 to 55 in their paper. We also want to emphasize that they trained two networks, one for the low noise level ($5 \leq \sigma \leq 29$) and one for higher noise level ($30 \leq \sigma \leq 55$). The reason is that due to the use of the constraint $||y - x||_2 \leq \epsilon$ by [26], we should not expect the model generalizes well to the noise levels surpasses the noise level of the training set.

For qualitative comparisons, some restored images of different models on the BSD68 dataset are presented in Fig. 4 and Fig. 5. As can be seen, more details are preserved in DURR than other models. It is worth noting that the noise level of the input image in Fig. 5 is 65, which is unseen by both DnCNN and DURR during training. Nonetheless, DURR achieves a significant gain of nearly 1 dB than DnCNN. Moreover, the texture on the cameo is very well restored by DURR. These results clearly indicate the strong generalization ability of our model.

More interestingly, due to the generalization ability in denoising, DURR is able to handle the problem of real image denoising without additional training. For testing, we test the images obtained from [25]. We present the representative results in Fig. 6 and more results are listed in the appendix.

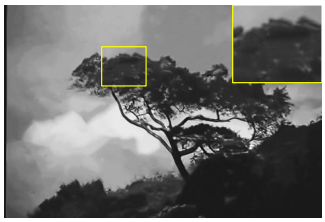
Table 2: Average PSNR (dB) results on the BSD68 dataset. Values with * means the corresponding noise level is not present in the training data of the model. The best results are indicated in red and the second best results are indicated in blue.

	BM3D [11]	WNMM [19]	DnCNN-B [42]	UNLNet$_5$ [26]	DURR
$\sigma = 15$	31.07	31.31	31.60	31.47	31.38*
$\sigma = 25$	28.55	28.73	29.15	28.96	29.15
$\sigma = 35$	27.07	27.28	27.66	27.50	27.70
$\sigma = 45$	25.99	26.26	26.62	26.48	26.71
$\sigma = 55$	25.26	25.49	25.80	25.64	25.91
$\sigma = 65$	24.69	24.51	23.40*	-	25.25*
$\sigma = 75$	22.63	22.71	18.73*	-	24.69*

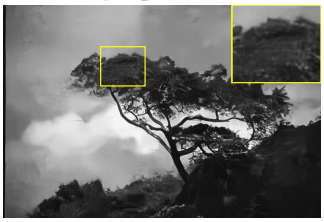
Ground Turth

Noisy Input, 17.84dB

(a) BM3D, 26.23dB



(b) WNMM, 26.35dB



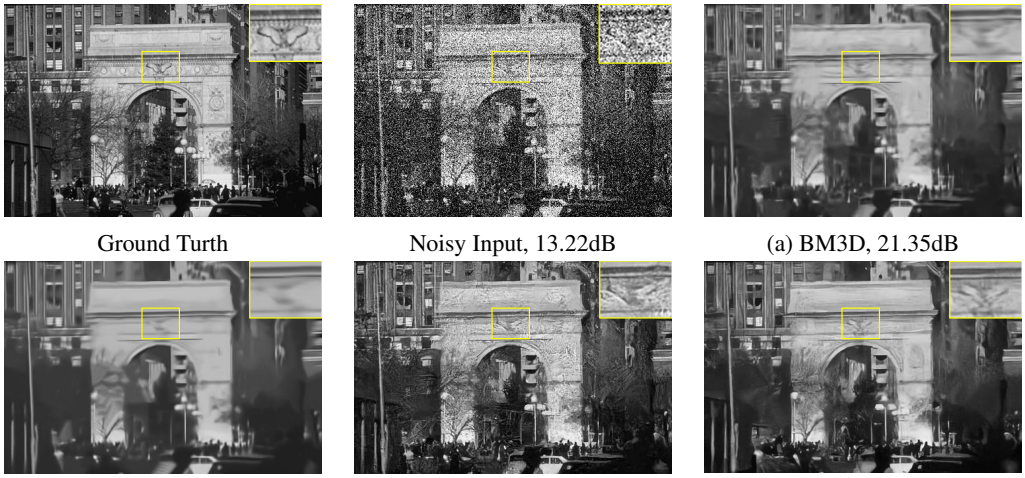
(c) DnCNN, 27.31dB

(d) DURR, 27.42dB

Figure 4: Denoising results of an image from BSD68 with noise level 35.

3.3.2 JPEG Image Deblocking

For deep learning based models, we select DnCNN-3 [42] for comparisons since it is the only known deep model for multiple QFs deblocking. As the AR-CNN [14] is a commonly used baseline, we re-train the AR-CNN on a training set with mixed QFs and denote this model as AR-CNN-B. Original AR-CNN as well as a non-learning-based method SA-DCT [18] are also tested. The quality factors are assumed known for these models.



Ground Turth | Noisy Input, 13.22dB | (a) BM3D, 21.35dB

(b) WNMM, 21.02dB | (c) DnCNN, 21.86dB | (d) DURR, 22.84dB

Figure 5: Denoising results of an image from BSD68 with noise level 65 (unseen by both DnCNN and DURR in their training sets).

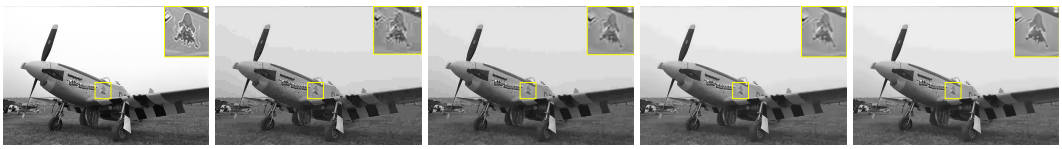
Noisy Image | BM3D [11] | DnCNN [42] | UNet_5 [24] | DURR

Figure 6: Denoising results on a real image from [25].

Quantitative results are shown in Table 3. Though the number of parameters of DURR is significantly less than the DnCNN-3, the proposed DURR outperforms DnCNN-3 in most cases. Specifically, considerable gains can be observed for our model on seen QFs, and the performances are comparable on unseen QFs. A representative result on the LIVE1 dataset is presented in Fig. 7. Our model generates the most clean and accurate details. More experiment details are given in the appendix.

Table 3: The average PSNR(dB) on the LIVE1 dataset. Values with * means the corresponding QF is not present in the training data of the model. The best results are indicated in red and the second best results are indicated in blue.

QF	JPEG	SA-DCT [18]	AR-CNN [14]	AR-CNN-B	DnCNN-3 [42]	DURR
10	27.77	28.65	28.98	28.53	29.40	29.23*
20	30.07	30.81	31.29	30.88	31.59	31.68
30	31.41	32.08	32.69	32.31	32.98	33.05
40	32.45	32.99	33.63	33.39	33.96	34.01*



Ground Truth | JPEG | (a) AR-CNN | (b) DnCNN | (c) DURR

Figure 7: JPEG deblocking results of an image from the LIVE1 dataset, compressed using QF 10.

3.4 Other Applications

Our model can be easily extended to other applications such as deraining, dehazing and deblurring. In all these applications, there are images corrupted at different levels. Rainfall intensity, haze density and different blur kernels will all effect the image quality.

4 Conclusions

In this paper, we proposed a image restoration model based on the moving endpoint control. The problem was solved by jointly optimizing two units: restoration unit and policy unit. The restoration unit used an RNN to realize the dynamics in the control problem. A policy network was trained for the policy unit to determine the loop times of the restoration unit for optimal results. Our model achieved the state-of-the-art results in blind image denoising and JPEG deblocking. Moreover, thanks to the flexibility of the given policy, DURR has shown strong abilities of generalization in our experiments.

Acknowledgement

Bin Dong is supported in part by NSFC 11671022. Yiping Lu is supported by the elite undergraduate training program of School of Mathematical Sciences in Peking University.

References

[1] Michal Aharon, Michael Elad, and Alfred Bruckstein. k-svd: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on signal processing*, 54(11):4311–4322, 2006.

[2] K. Bredies, K. Kunisch, and T. Pock. Total Generalized Variation. *SIAM Journal on Imaging Sciences*, 3:492, 2010.

[3] Antoni Buades, Bartomeu Coll, and J-M Morel. A non-local algorithm for image denoising. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 2, pages 60–65. IEEE, 2005.

[4] J.F. Cai, S. Osher, and Z. Shen. Split Bregman methods and frame based image restoration. *Multiscale Modeling and Simulation: A SIAM Interdisciplinary Journal*, 8(2):337–369, 2009.

[5] Jian-Feng Cai, Bin Dong, Stanley Osher, and Zuowei Shen. Image restoration: total variation, wavelet frames, and beyond. *Journal of the American Mathematical Society*, 25(4):1033–1089, 2012.

[6] Jian-Feng Cai, Bin Dong, and Zuowei Shen. Image restoration: a wavelet frame based model for piecewise smooth functions and beyond. *Applied and Computational Harmonic Analysis*, 41(1):94–138, 2016.

[7] Jian-Feng Cai, Hui Ji, Zuowei Shen, and Gui-Bo Ye. Data-driven tight frame construction and image denoising. *Applied and Computational Harmonic Analysis*, 37(1):89–105, 2014.

[8] Francine Catté, Pierre-Louis Lions, Jean-Michel Morel, and Tomeu Coll. Image selective smoothing and edge detection by nonlinear diffusion. *SIAM Journal on Numerical analysis*, 29(1):182–193, 1992.

[9] Bo Chang, Lili Meng, Eldad Haber, Lars Ruthotto, David Begert, and Elliot Holtham. Reversible architectures for arbitrarily deep residual neural networks. *AAAI2018*, 2017.

[10] Y. Chen and T Pock. Trainable nonlinear reaction diffusion: A flexible framework for fast and effective image restoration. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 39(6):1256–1272, 2017.

[11] Kostadin Dabov, Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian. Image denoising by sparse 3-d transform-domain collaborative filtering. *IEEE Transactions on image processing*, 16(8):2080–2095, 2007.

[12] I. Daubechies, G. Teschke, and L. Vese. Iteratively solving linear inverse problems under general convex constraints. *Inverse Problems and Imaging*, 1(1):29, 2007.

[13] Bin Dong, Qingtang Jiang, and Zuowei Shen. Image restoration: Wavelet frame shrinkage, nonlinear evolution pdes, and beyond. *Multiscale Modeling & Simulation*, 15(1):606–660, 2017.

[14] Chao Dong, Yubin Deng, Chen Change Loy, and Xiaoou Tang. Compression artifacts reduction by a deep convolutional network. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 576–584, 2015.

[15] M. Elad, J.L. Starck, P. Querre, and D.L. Donoho. Simultaneous cartoon and texture image inpainting using morphological component analysis (MCA). *Applied and Computational Harmonic Analysis*, 19(3):340–358, 2005.

[16] Lawrence C Evans. An introduction to mathematical optimal control theory version 0.2. *Tailieu Vn*, 2005.

[17] Cong Fang, Zhenyu Zhao, Pan Zhou, and Zhouchen Lin. Feature learning via partial differential equation with applications to face recognition. *Pattern Recognition*, 69:14–25, 2017.

[18] Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian. Pointwise shape-adaptive dct for high-quality denoising and deblocking of grayscale and color images. *IEEE Transactions on Image Processing*, 16(5):1395–1411, 2007.

[19] Shuhang Gu, Lei Zhang, Wangmeng Zuo, and Xiangchu Feng. Weighted nuclear norm minimization with application to image denoising. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2862–2869, 2014.

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[21] Viren Jain and Sebastian Seung. Natural image denoising with convolutional networks. In *Advances in Neural Information Processing Systems*, pages 769–776, 2009.

[22] H. Ji, C. Liu, Z. Shen, and Y. Xu. Robust video denoising using low rank matrix completion. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[23] Xiangyang Lan, Stefan Roth, Daniel Huttenlocher, and Michael J Black. Efficient belief propagation with learned higher-order markov random fields. In *European conference on computer vision*, pages 269–282. Springer, 2006.

[24] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Fractalnet: Ultra-deep neural networks without residuals. *arXiv preprint arXiv:1605.07648*, 2016.

[25] Marc Lebrun, Miguel Colom, and Jean-Michel Morel. The noise clinic: a blind image denoising algorithm. *Image Processing On Line*, 5:1–54, 2015.

[26] Stamatios Lefkimmiatis. Universal denoising networks: A novel cnn-based network architecture for image denoising. *arXiv preprint arXiv:1711.07807*, 2017.

[27] Qianli Liao and Tomaso Poggio. Bridging the gaps between residual learning, recurrent neural networks and visual cortex. *arXiv preprint*, 2016.

[28] Yiping Lu, Aoxiao Zhong, Quanzheng Li, and Bin Dong. Beyond finite layer neural networks: Bridging deep architectures and numerical differential equations. *Thirty-fifth International Conference on Machine Learning (ICML)*, 2018.

[29] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proc. 8th Int'l Conf. Computer Vision*, volume 2, pages 416–423, July 2001.

[30] D. Mumford and J. Shah. Optimal approximations by piecewise smooth functions and associated variational problems. *Communications on pure and applied mathematics*, 42(5):577–685, 1989.

[31] Stanley Osher and Leonid Rudin. Feature-oriented image enhancement using shock filters. *SIAM Journal on Numerical Analysis*, 27(4):919–940, Aug 1990.

[32] 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.

[33] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[34] Leonid I Rudin, Stanley Osher, and Emad Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: nonlinear phenomena*, 60(1-4):259–268, 1992.

[35] HR Sheikh. Live image quality assessment database release 2. *http://live.ece.utexas.edu/research/quality*, 2005.

[36] J.L. Starck, M. Elad, and D.L. Donoho. Image decomposition via the combination of sparse representations and a variational approach. *IEEE transactions on image processing*, 14(10):1570–1582, 2005.

[37] Richard S Sutton and Andrew G Barto. *Introduction to reinforcement learning*, volume 135. MIT press Cambridge, 1998.

[38] Cheng Tai and E Weinan. Multiscale adaptive representation of signals: I. the basic framework. *The Journal of Machine Learning Research*, 17(1):4875–4912, 2016.

[39] Ying Tai, Jian Yang, Xiaoming Liu, and Chunyan Xu. Memnet: A persistent memory network for image restoration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4539–4547, 2017.

[40] Joachim Weickert. *Anisotropic diffusion in image processing*, volume 1. Teubner Stuttgart, 1998.

[41] E Weinan. A proposal on machine learning via dynamical systems. *Communications in Mathematics & Statistics*, 5(1):1–11, 2017.

[42] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE Transactions on Image Processing*, 26(7):3142–3155, 2017.

[43] Xiaoshuai Zhang, Wenhan Yang, Yueyu Hu, and Jiaying Liu. Dmcnn: Dual-domain multi-scale convolutional neural network for compression artifacts removal. In *Proceedings of the 25th IEEE International Conference on Image Processing*, 2018.

[44] Xingcheng Zhang, Zhizhong Li, Chen Change Loy, and Dahua Lin. Polynet: A pursuit of structural diversity in very deep networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3900–3908. IEEE, 2017.

[45] Yong Zhang, Bin Dong, and Zhaosong Lu. l_0 minimization for wavelet frame based image restoration. *Mathematics of Computation*, 82(282):995–1015, 2013.

Appendix

4.1 Network Structure

4.1.1 The Adopted Structure

For the restoration unit, we use a minimal U-Net [33] style network to predict the residual image. The input of the restoration unit is the processed image (*i.e.* the last output) and the original degraded observation. The architecture of the network is listed in Table. 4. The architecture of the policy unit is listed in Table. 5.

Table 4: Architecture of the restoration unit. After each convolution layer, except the last one, there is a Parametric Rectified Linear Unit (PReLU) layer. The output of the second conv. layer is concatenated as a part of the input of the second-to-last conv. layer.

Type	Kernel	Dilation	Stride	Outputs
conv.	5×5	1	1×1	32
conv.	3×3	1	1×1	32
conv.	3×3	1	2×2	64
conv.	3×3	1	1×1	64
dilated conv.	3×3	2	1×1	64
dilated conv.	3×3	4	1×1	64
deconv.	4×4	1	$1/2 \times 1/2$	64
conv.	3×3	1	1×1	32
conv.	5×5	1	1×1	1

Table 5: Architecture of the policy unit. After each convolution layer, there is a Rectified Linear Unit (ReLU) layer.

Type	Kernel	Dilation	Stride	Outputs	Remark
conv.	5×5	1	1×1	16	
conv.	3×3	1	1×1	16	Link 1
conv.	3×3	1	1×1	16	
conv.	3×3	1	1×1	16	Add Link 1
conv.	3×3	1	2×2	32	Link 2
conv.	3×3	1	1×1	32	
conv.	3×3	1	1×1	32	Add Link 2
conv.	3×3	1	2×2	64	Link 3
conv.	3×3	1	1×1	64	
conv.	3×3	1	1×1	64	Add Link 3
Global Average Pooling					
LSTM with 32 hidden units					
fc.	-	-	-	1	Sigmoid Activation

4.1.2 Discussions

Fig. 8 justify the rationality of our restoration unit design. An AR-CNN-like structure is tested in our experiments, the number of parameters is comparable with the restoration unit adopted in DURR.

As the Fig. 8 illustrates, the U-Net style network (the one we adopted) generated images tend to have significantly less artifacts as well as more pleasing qualities, though the PSNR results of these images are close.

4.2 Further Results

4.2.1 Image Denoising

The performaces of DnCNN and DURR under extreme noise conditions ($sigma = 95$) is tested. It can be easily observed from Fig. 9 that the proposed DURR outperforms DnCNN on both quantitative measurements and visual qualities.

We further report the results of our algorithm in Fig. 13 and Fig. 12. We demonstrate the output of every second iteration of the restoration unit in Fig. 15. We also plot the PSNR variety when passing the restoration unit different times, the tendency is plotted in Fig. 14. The test performance increases during passing the first few steps, but the benefit seems to diminish after a peak. To demonstrate this point more intuitively, the residual image with our output and ground truth is also demonstrated in Fig15. This indicates us that adaptively choose a stopping time is reasonable and necessary.

Figure 8: Denoising results of different restoration unit structure designs. Left images are produced by the AR-CNN-like unit. Right images are produced by our proposed minimal U-Net style network.

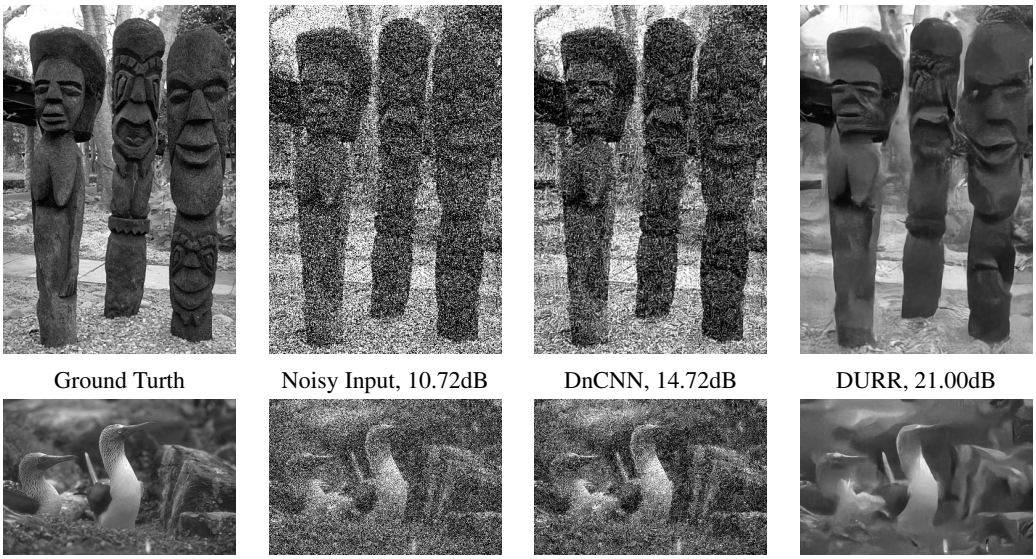
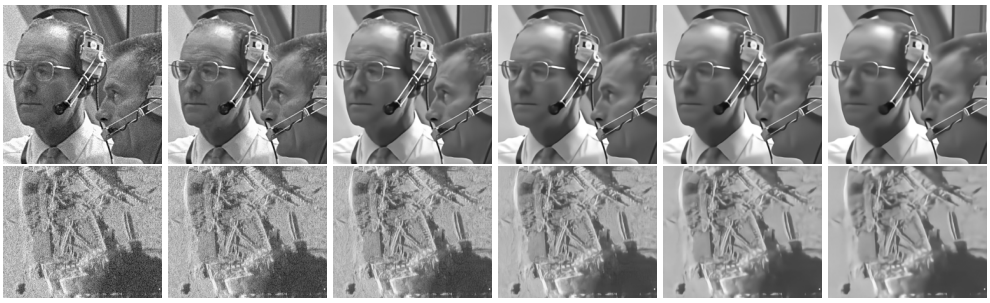


Figure 9: Denoising results of images from BSD68 with extreme noise conditions ($\sigma = 95$).

4.3 Real Image Denoising

In this section, we demonstrate more results of processing the real images. In Fig. 10, we demonstrate the output of the restoration unit with different unfolding times (*i.e.* passing the restoration unit with different times). Results demonstrate that our network has strong generalization ability and can be used to handle the problem of real image denoising. Fig. 10 show that our restoration unit behaves

much like a bilateral filter, which preserves the edges and reduces the noise. If we filter the images for too many times, the images tend to become over-smoothed.



Noisy Image 1 2 3 4 5

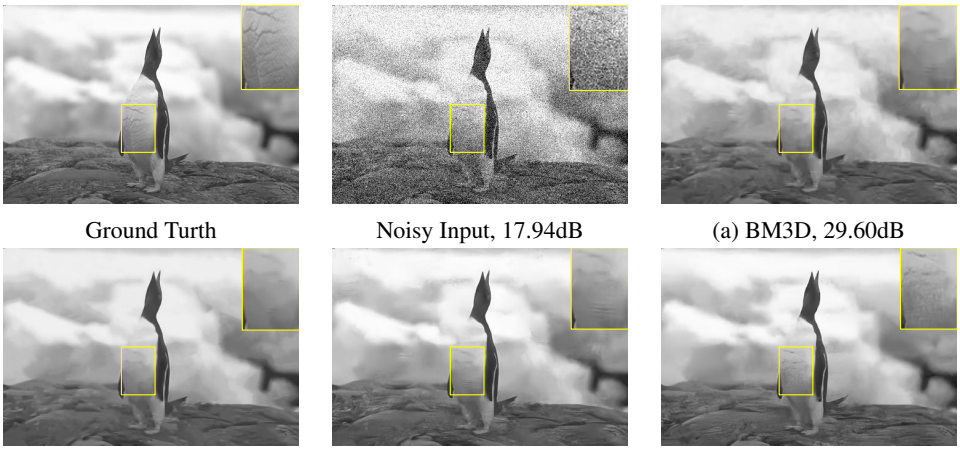
Figure 10: Denoising result of a real image. The subcaption denotes the unfolding time.

4.4 JPEG Deblocking

Here we demonstrate in Fig. 11 that our model is able to remove the noise while preserving the structures. It can be easily seen in the white zoom-in boxes that the edges of the windows is well-preserved after the processing of DURR. In the meantime DnCNN fails to keep the structure.

Ground Truth JPEG (a) AR-CNN (b) DnCNN (c) DURR

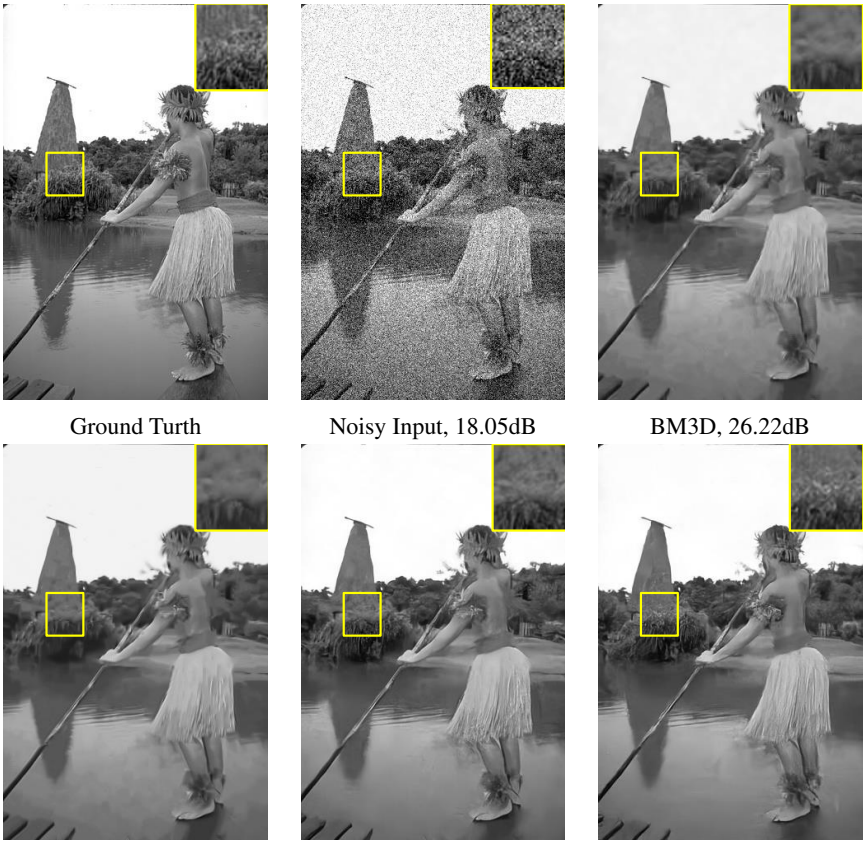
Figure 11: JPEG deblocking results of an image from the LIVE1 dataset, compressed using QF 10.



Ground Turth | Noisy Input, 17.94dB | (a) BM3D, 29.60dB

(b) WNMM, 29.73dB | (c) DnCNN, 31.67dB | (d) DURR, 31.72dB

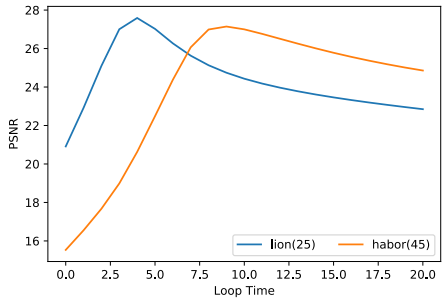
Figure 12: Denoising results of an image from BSD68 with noise level 35.



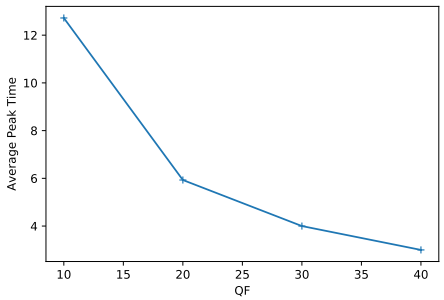
Ground Turth | Noisy Input, 18.05dB | BM3D, 26.22dB

WNMM, 26.54dB | DnCNN, 28.64dB | DURR, 28.72dB

Figure 13: Denoising results of an image from BSD68 with noise level 35.



PSNR tendency for lion and harbor images in Fig. 15.



Averge loop time relates to QF.

Figure 14: Image quality's relation to loop times.

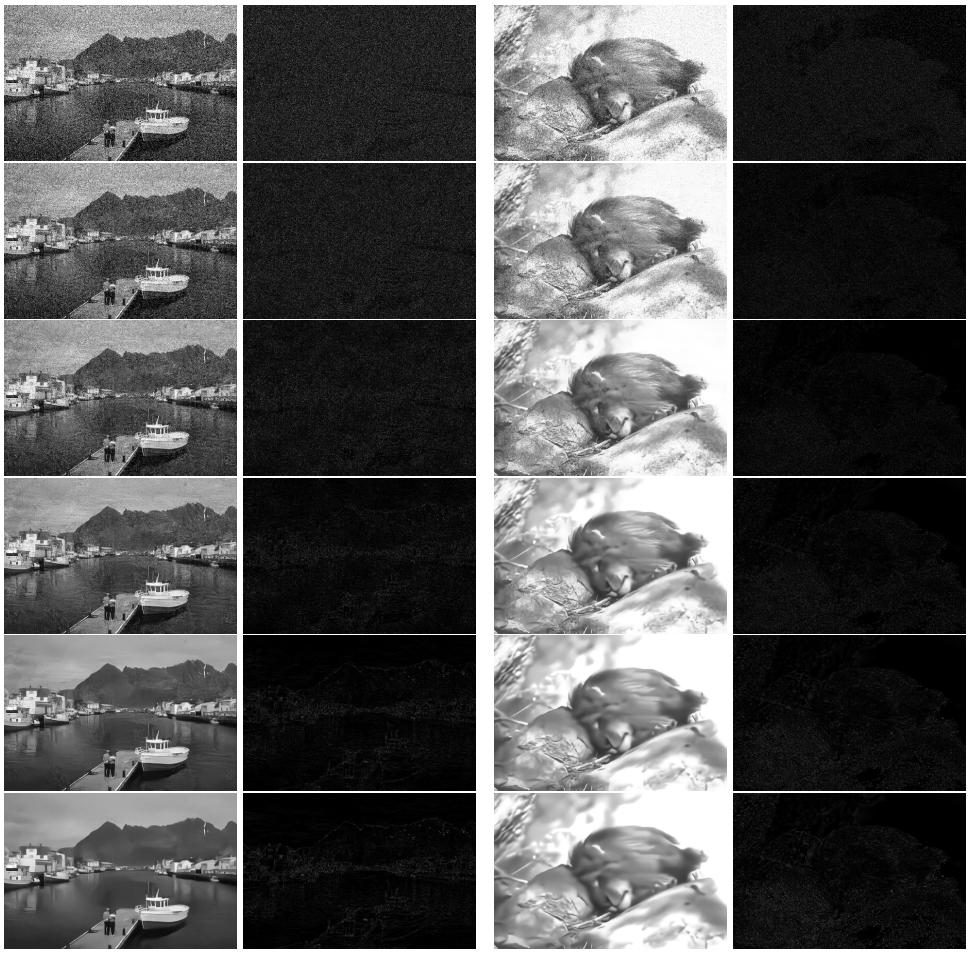


Figure 15: Denoising results on images from the BSD68 dataset. The input harbor image's noise level is set to 45 and the lion image's noise level is set to 25.