Deep Plug-and-Play and Deep Unfolding Methods for Image Restoration

Kai Zhang

Computer Vision Lab, ETH Zurich, Switzerland

https://github.com/cszn

Outline

Deep Plug-and-Play Image Restoration

- IRCNN (CVPR2017), DPIR (TPAMI 2021)

Deep Unfolding Image Restoration

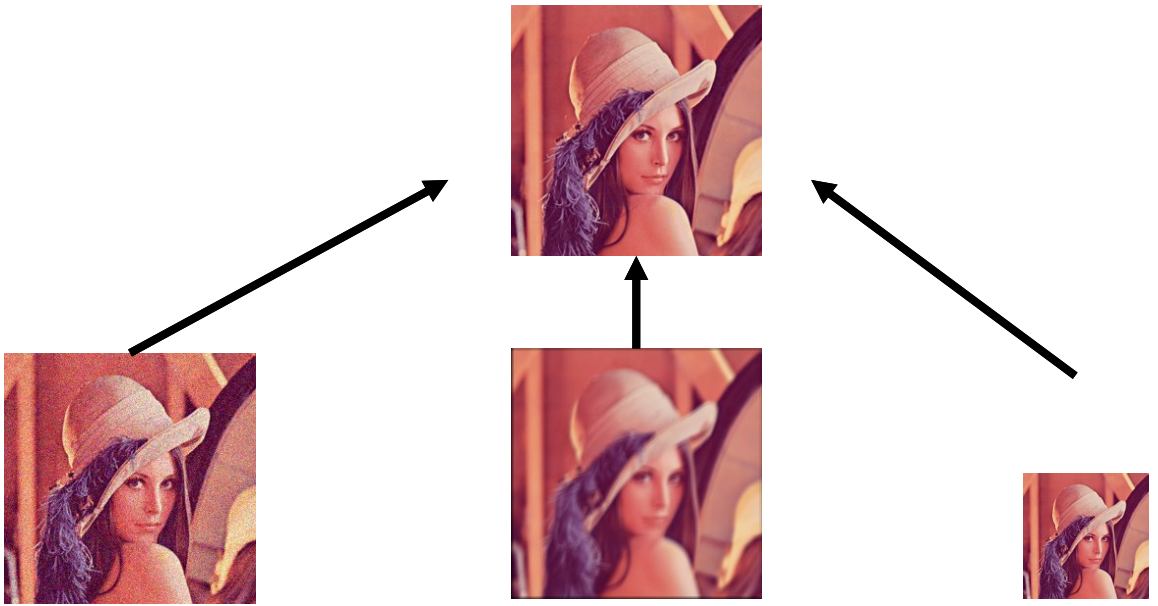
- USRNet (CVPR 2020)

Deep Blind Image Restoration

- BSRGAN (ICCV 2021)
- SCUNet (Arxiv 2022)

Image restoration: the problem

- Reconstruct the latent image from its degraded measurement
  - Denoising, deblurring, super-resolution, …



General image observation/degradation/formation model

$$\boldsymbol{y} = \boldsymbol{H}\boldsymbol{x} + \boldsymbol{n}$$

$\boldsymbol{H}$: The observation (degradation) matrix
$\boldsymbol{n}$: The additive white Gaussian noise with standard deviation σ

$$\boldsymbol{y} = (\boldsymbol{x} \otimes \boldsymbol{k}) \downarrow_{\boldsymbol{s}} + \boldsymbol{n}$$

$\otimes$: two-dimensional convolution of $\boldsymbol{x}$ with blur kernel $\boldsymbol{k}$
$\downarrow_{\boldsymbol{s}}$: Downsampling with scale factor $\boldsymbol{s}$

- Goal of image restoration

  Given observation $\boldsymbol{y}$, recover the latent image $\boldsymbol{x}$

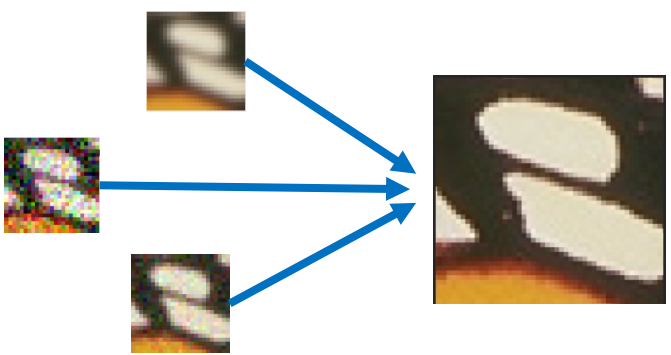
- Image restoration is a typical ill-posed inverse problem.

Many-to-many problem

Many-to-one

$$y = (x \otimes k) \downarrow_s + n$$

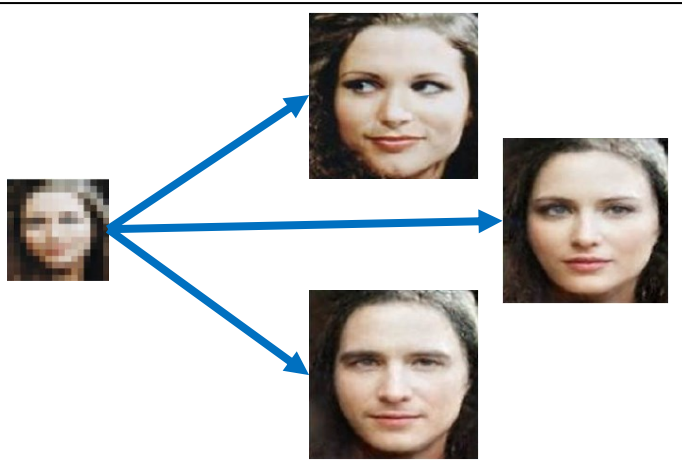
Many One



One-to-many

$$y \quad \begin{matrix} \nearrow (x \otimes k) \downarrow_s + n \\ \searrow (x \otimes k) \downarrow_s + n \end{matrix}$$

One Many



Model based methods

- Based on the image degradation process and the available image priors, build a model (objective function) and optimize it.
- General model:

Data fidelity term Regularization (prior) term

$$min_x \frac{1}{2\sigma^2} \|\boldsymbol{y} - (\boldsymbol{x} \otimes \boldsymbol{k}) \downarrow_s \|^2 + \lambda \cdot R(\boldsymbol{x})$$

- Key issues:
  - Modeling of the degradation process
  - Good priors about the latent image
  - Good objective function for minimization

Learning based methods

- Learn a compact inference or a mapping function from a training set of degraded-latent image pairs.
- General formulation:

Loss function — Set of parameters to be learned

$$min_{\Theta} loss(\hat{\boldsymbol{x}}, \boldsymbol{x}) \quad s.t. \quad \hat{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{y}, \boldsymbol{H}; \Theta)$$

- Key issues
  - The availability of paired training data
  - The design of learning architecture
  - The definition of loss function

Representative methods

- **Model based methods**

  Total variation (TV)
  Sparse representation
  Dictionary learning
  Low-rank approximation

- **Learning based methods**

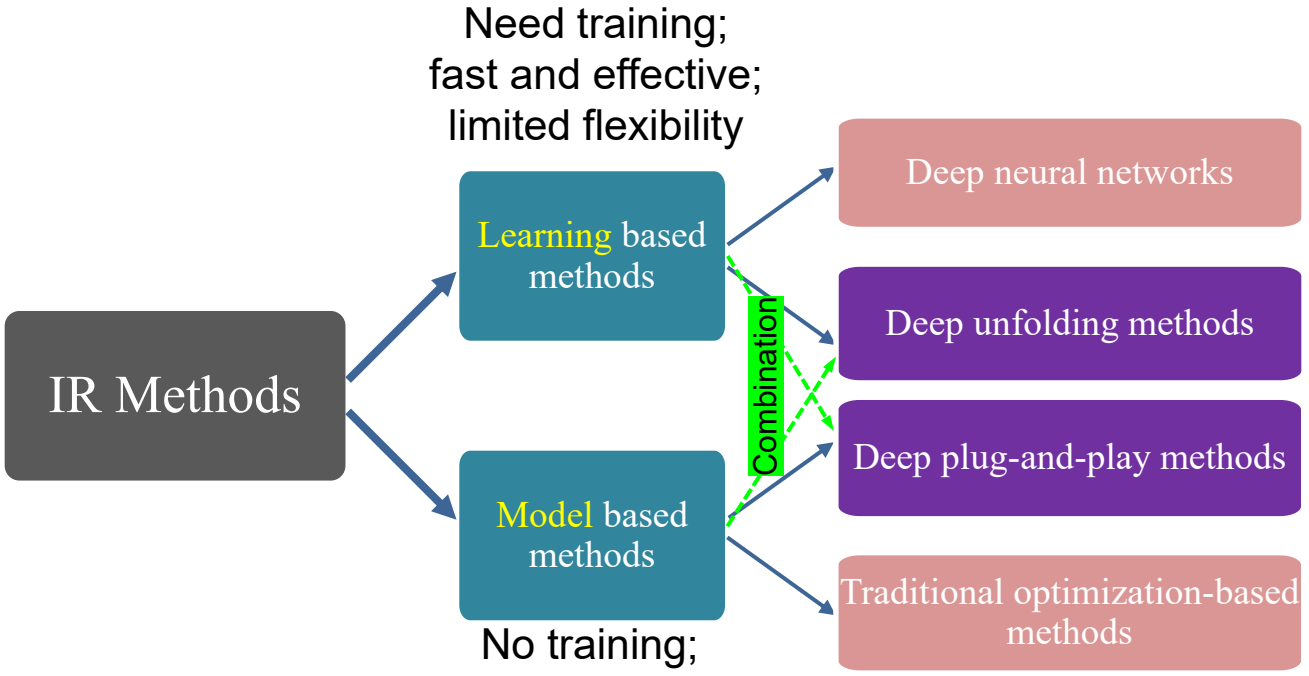
  Mixture of experts
  Shrinkage fields
  Trainable nonlinear reaction diffusion
  MLP and deep learning (Convolutional Neural Networks)

Motivation

- **Model based methods**
  - ✓High flexibility; clear interpretability; do not need training
  - × The prior may not strong enough; time-consuming
- **Learning based methods**
  - ✓End-to-end training with training paired data; fast and effective
  - × Need training; limited flexibility and interpretability

Can we integrate the model based methods and learning based methods for general image restoration?

Motivation



- **Deep unfolding methods**: design optimization algorithm-inspired network architecture
- **Deep plug-and-play methods**: plug learned deep model into optimization algorithm

Half quadratic splitting (HQS) algorithm

- The general model for image restoration

$$min_x \frac{1}{2\sigma^2} \|\boldsymbol{y} - (\boldsymbol{x} \otimes \boldsymbol{k}) \downarrow_{\boldsymbol{s}} \|^2 + \lambda \cdot R(\boldsymbol{x})$$

- Introducing an auxiliary variable $\boldsymbol{z}$ ($\boldsymbol{z} \approx \boldsymbol{x}$ when μ is large enough)

$$min_{x,z} \frac{1}{2\sigma^2} \|\boldsymbol{y} - (\boldsymbol{x} \otimes \boldsymbol{k}) \downarrow_{\boldsymbol{s}} \|^2 + \lambda \cdot R(\boldsymbol{z}) + \frac{\mu}{2} \|\boldsymbol{x} - \boldsymbol{z}\|^2$$

- Solving $\boldsymbol{x}$ and $\boldsymbol{z}$ alternatively and iteratively

(a) $\boldsymbol{x}_k = min_{\boldsymbol{x}} \|\boldsymbol{y} - (\boldsymbol{x} \otimes \boldsymbol{k}) \downarrow_{\boldsymbol{s}} \|_2^2 + \mu\sigma^2 \|\boldsymbol{x} - \boldsymbol{z}_{k-1}\|^2$ % Solving data subproblem

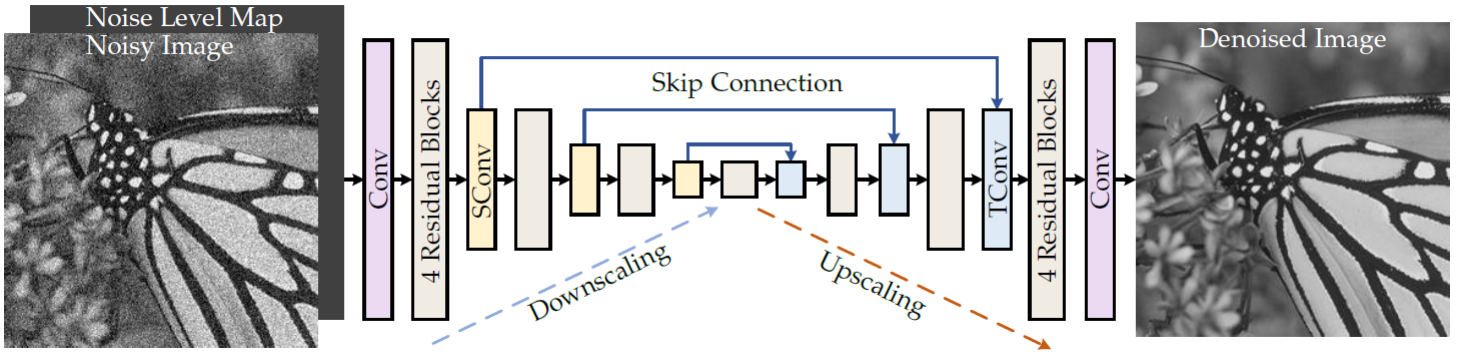
(b) $\boldsymbol{z}_k = min_{\boldsymbol{z}} \frac{1}{2\left(\sqrt{\lambda/\mu}\right)^2} \|\boldsymbol{x}_k - \boldsymbol{z}\|^2 + R(\boldsymbol{z})$ % Solving denoising sub-problem

Replace it with CNN denoiser

Plug-and-play image restoration with CNN denoiser prior

Plugging the strong CNN denoiser prior into model based methods

- Step (a): Solving data subproblem
- Step (b): Solving denoising sub-problem with CNN denoiser



The network architecture of the CNN denoiser

K. Zhang, W. Zuo, S. Gu, L. Zhang. "Learning Deep CNN Denoiser Prior for Image Restoration." *CVPR* 2017.
Code: https://github.com/cszn/ircnn

Denoising results

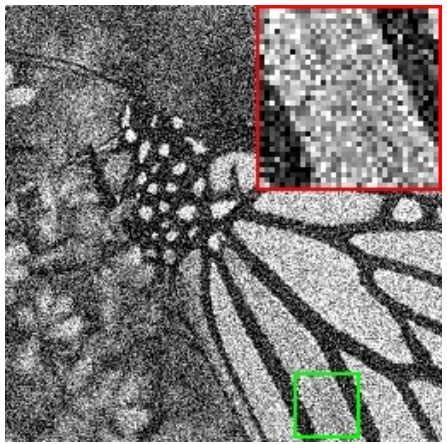
The average PSNR(dB) results of different methods on grayscale BSD68 dataset.

Datasets	Noise Level	BM3D	WNNM	DnCNN	IRCNN	FFDNet	DRUNet
Set12	15	32.37	32.70	32.86	32.77	32.75	**33.25**
	25	29.97	30.28	30.44	30.38	30.43	**30.94**
	50	26.72	27.05	27.18	27.14	27.32	**27.90**
BSD68	15	31.08	31.37	31.73	31.63	31.63	**31.91**
	25	28.57	28.83	29.23	29.15	29.19	**29.48**
	50	25.60	25.87	26.23	26.19	26.29	**26.59**

Average PSNR(dB) results of different methods for noise levels 15, 25 and 50 on different datasets.

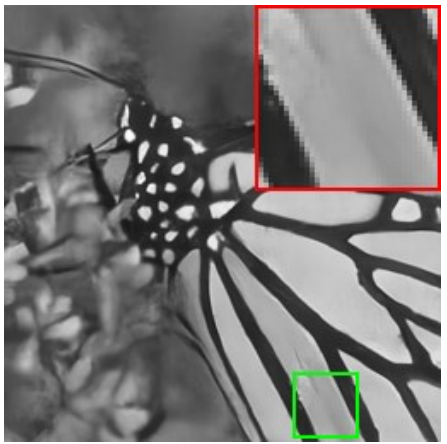
Datasets	Noise Level	CBM3D	DnCNN	IRCNN	FFDNet	DRUNet
CBSD68	15	33.52	33.90	33.86	33.87	**34.30**
	25	30.71	31.24	31.16	31.21	**31.69**
	50	27.38	27.95	27.86	27.96	**28.51**
Kodak24	15	34.28	34.60	34.69	34.63	**35.31**
	25	32.15	32.14	32.18	32.13	**32.89**
	50	28.46	28.95	28.93	28.98	**29.86**
McMaster	15	34.06	33.45	34.58	34.66	**35.40**
	25	31.66	31.52	32.18	32.35	**33.14**
	50	28.51	28.62	28.91	29.18	**30.08**

Denoising visual results

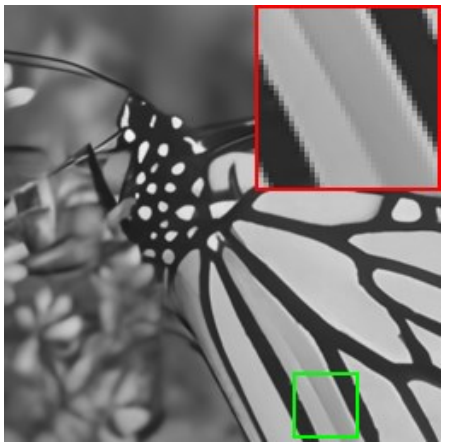


Noisy (14.78dB)

BM3D (25.82dB)



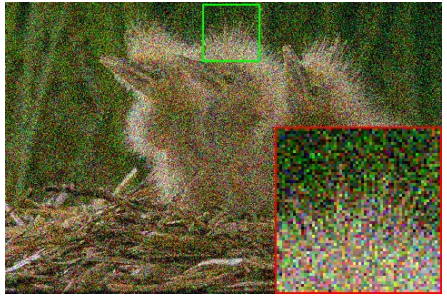
DnCNN (26.83dB)



Proposed (27.31dB)

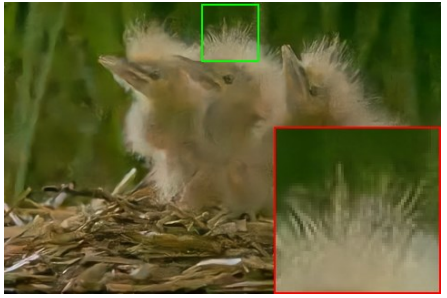
Grayscale image denoising results of different methods on image “Monarch” from Set12 dataset with noise level 50.

Denoising visual results

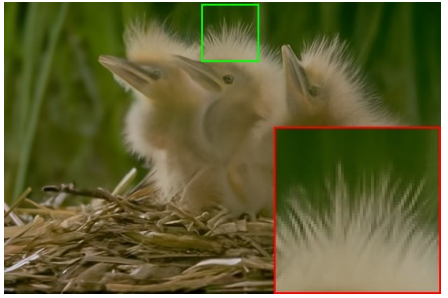


Noisy(14.99dB)

CBM3D (28.36dB)



DnCNN (28.68dB)



Proposed (29.28dB)

Color image denoising results of different methods on image “163085” from CBSD68 dataset with noise level 50.

Plug-and-play image restoration with deep denoiser prior

Algorithm 1: Plug-and-play image restoration with deep denoiser prior (DPIR)

Input : Deep denoiser prior model, degraded image $\mathbf{y}$, degradation model $\mathbf{y} = (\mathbf{x} \otimes \mathbf{k})\downarrow_{\mathbf{s}} + \mathbf{n}$, image noise level σ, β_k of denoiser prior model at k-th iteration for a total of K iterations, trade-off parameter λ.

Output: Restored image $\mathbf{z}_K$.

Initialize $\mathbf{z}_0$ from $\mathbf{y}$, pre-calculate $\alpha_k = \lambda\sigma^2/\beta_k^2$.

for $k = 1, 2, \cdots, K$ **do**

  $\mathbf{x}_k = \arg\min_{\mathbf{x}} \|\mathbf{y} - (\mathbf{x} \otimes \mathbf{k})\downarrow_{\mathbf{s}}\|^2 + \alpha_k \|\mathbf{x} - \mathbf{z}_{k-1}\|^2$; *// Solving data subproblem*

  $\mathbf{z}_k = Denoiser(\mathbf{x}_k, \beta_k)$; *// Denoising with deep DRUNet denoiser*

end

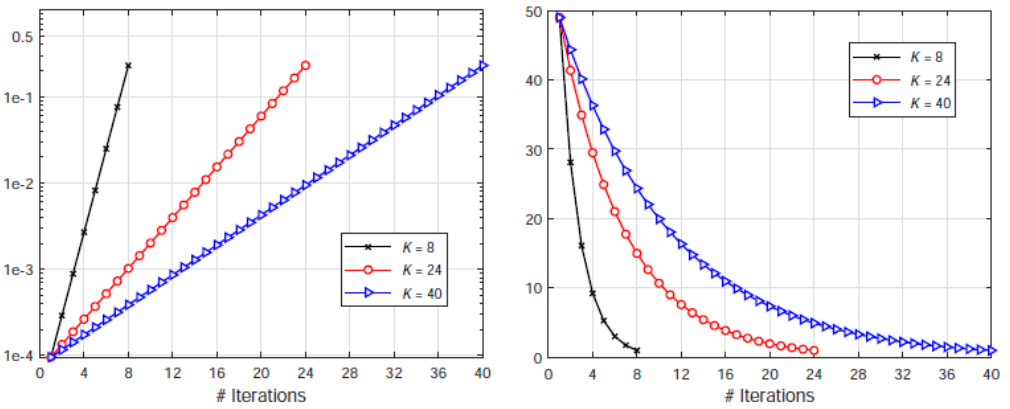
Parameter setting

SINGLE IMAGE SUPER-RESOLUTION VIA BM3D SPARSE CODING

Karen Egiazarian and Vladimir Katkovnik

Department of Signal Processing,Tampere University of Technology
Korkeakoulunkatu 10, 33720, Tampere, Finland, email: 'firstname.lastname'@tut.fi

- the threshold parameter for Color BM3D (CBM3D) filter is varying in iterations as a quadratic function decreasing from $12s$ to s.



(a) α_k (b) σ_k

Fig. 5. The values of α_k and σ_k at k-th iteration with respect to different number of iterations K = 8, 24, and 40.

Inspired by such domain knowledge, we can instead set σ_k and λ to implicitly determine μ_k. Based on the fact that μ_k should be monotonically increasing, we uniformly sample σ_k from a large noise level σ_1 to a small one σ_K in log space. This means that μ_k can be easily determined via $\mu_k = \lambda/\sigma_k^2$. Following [17], σ_1 is fixed to 49 while σ_K is determined by the image noise level σ. Since K is user-specified and σ_K has clear physical meanings, they are practically easy to set. As the fact that λ comes from the prior term and thus should be fixed, we can choose the optimal λ by a grid search on a validation dataset. Empirically, λ can yield favorable performance from the range of $[0.19, 0.55]$. In this paper, we fix it to 0.23 unless otherwise specified. It should be noted that since λ can be absorbed into σ and plays the role of controlling the trade-off between data term and prior term, one can implicitly tune λ by multiplying σ by a scalar.

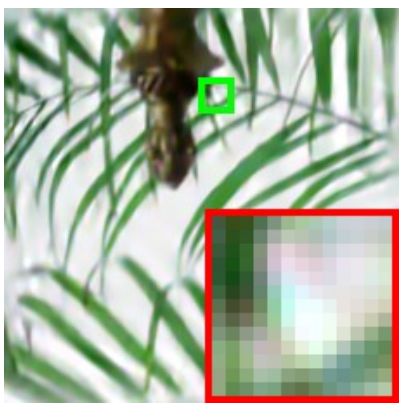
Application: deblurring

Solution of data subproblem:

$$\mathbf{x}_k = \mathcal{F}^{-1}\left(\frac{\overline{\mathcal{F}(\mathbf{k})}\mathcal{F}(\mathbf{y}) + \alpha_k \mathcal{F}(\mathbf{z}_{k-1})}{\overline{\mathcal{F}(\mathbf{k})}\mathcal{F}(\mathbf{k}) + \alpha_k}\right)$$

Blurry and noisy

DMPHN (9.74dB)



EPLL (20.06dB)

Proposed DPIR (30.27dB)

Intermediate results and convergence

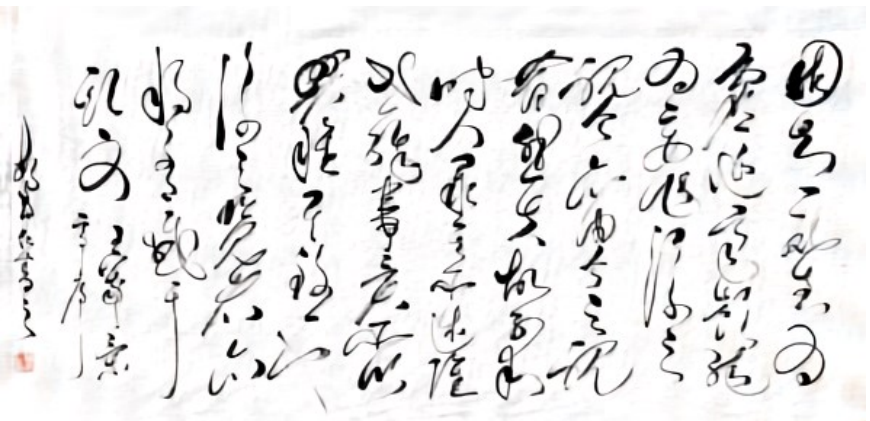
(a) $\mathbf{x}_1$ (16.34dB) (b) $\mathbf{z}_1$ (23.75dB) (c) $\mathbf{x}_4$ (22.33dB) (d) $\mathbf{z}_4$ (29.35dB) (e) $\mathbf{x}_8$ (29.34dB) (f) Convergence curves

$$\begin{cases} \mathbf{z}_k = \arg\min_{\mathbf{z}} \|\mathbf{y} - (\mathbf{z} \otimes \mathbf{k})\downarrow_{\mathbf{s}}\|^2 + \mu\sigma^2\|\mathbf{z} - \mathbf{x}_{k-1}\|^2 \\ \mathbf{x}_k = \arg\min_{\mathbf{x}} \frac{\mu}{2}\|\mathbf{z}_k - \mathbf{x}\|^2 + \lambda\Phi(\mathbf{x}). \end{cases}$$

- While the data subproblem can handle the distortion of blur, it also aggravates the strength of noise.
- The deep denoiser prior plays the role of removing noise, leading to a noise-free z_k.
- Compared with x_1 and x_4, x_8 contains more fine details, which means the data subproblem can iteratively recover the details.
- x_k and z_k enjoy a fast convergence to the fixed point.

Deblurring example

Input



Output

Application: single image super-resolution

IEEE TRANSACTIONS ON IMAGE PROCESSING, VOL. 25, NO. 8, AUGUST 2016 3683

Fast Single Image Super-Resolution Using a New Analytical Solution for ℓ_2–ℓ_2 Problems

Ningning Zhao, *Student Member, IEEE*, Qi Wei, *Student Member, IEEE*, Adrian Basarab, *Member, IEEE*, Nicolas Dobigeon, *Senior Member, IEEE*, Denis Kouamé, *Member, IEEE*, and Jean-Yves Tourneret, *Senior Member, IEEE*

$$\hat{\mathbf{x}} = \underset{\mathbf{x}}{\operatorname{argmin}} \frac{1}{2}\|\mathbf{y} - \mathbf{SHx}\|_2^2 + \tau\|\mathbf{x} - \mathbf{x}_0\|_2^2. \quad (25)$$

This optimization problem was solved using a gradient descent approach in [7]. However, it can benefit from the analytical solution provided by Theorem 1 that can be implemented using Algo. 1.

Algorithm 1 FSR With Image-Domain ℓ_2-Regularization: Implementation of the Analytical Solution (15)

Input: $\mathbf{y}$, $\mathbf{H}$, $\mathbf{S}$, $\tilde{\mathbf{x}}$, τ, d

// Factorization of $\mathbf{H}$ (FFT of the blurring kernel)

$\mathbf{H} = \mathbf{F}^H\mathbf{\Lambda}\mathbf{F}$;

// Compute $\underline{\mathbf{\Lambda}}$

$\underline{\mathbf{\Lambda}} = [\mathbf{\Lambda}_1, \mathbf{\Lambda}_2, \cdots, \mathbf{\Lambda}_d]$;

// Calculate FFT of $\mathbf{r}$ denoted as $\mathbf{Fr}$

$\mathbf{Fr} = \mathbf{F}(\mathbf{H}^H\mathbf{S}^H\mathbf{y} + 2\tau\tilde{\mathbf{x}})$;

// Hadamard (or entrywise) product in frequency domain

$\mathbf{x}_f = \left(\underline{\mathbf{\Lambda}}^H \left(2\tau d\mathbf{I}_{N_i} + \mathbf{\Lambda}\underline{\mathbf{\Lambda}}^H\right)^{-1}\underline{\mathbf{\Lambda}}\right)\mathbf{Fr}$;

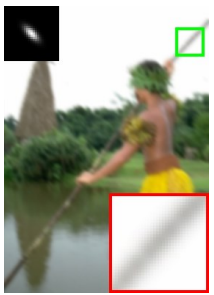
// Compute the analytical solution

$\tilde{\mathbf{x}} = \frac{1}{2\tau}\left(\mathbf{r} - \mathbf{F}^H\mathbf{x}_f\right)$;

Output: $\tilde{\mathbf{x}}$

Solution of data subproblem:

$$\mathbf{x}_k = \mathcal{F}^{-1}\left(\frac{1}{\alpha_k}\left(\mathbf{d} - \overline{\mathcal{F}(\mathbf{k})} \odot_s \frac{(\mathcal{F}(\mathbf{k})\mathbf{d}) \Downarrow_s}{(\overline{\mathcal{F}(\mathbf{k})}\mathcal{F}(\mathbf{k})) \Downarrow_s + \alpha_k}\right)\right)$$



Low-resolution

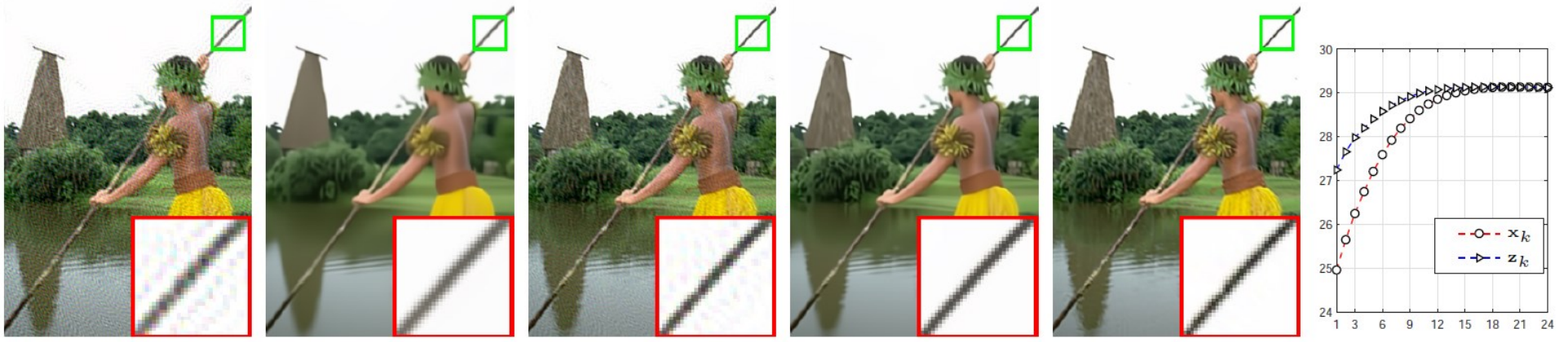
RCAN (24.61dB)

MZSR (27.34dB)

Proposed (29.14dB)

Ningning Zhao, Qi Wei, Adrian Basarab, Nicolas Dobigeon, Denis Kouame, and Jean-Yves Tourneret. Fast single image super-resolution using a new analytical solution for l2-l2 problems. IEEE TIP, 25(8):3683–3697, 2016.

Intermediate results and convergence

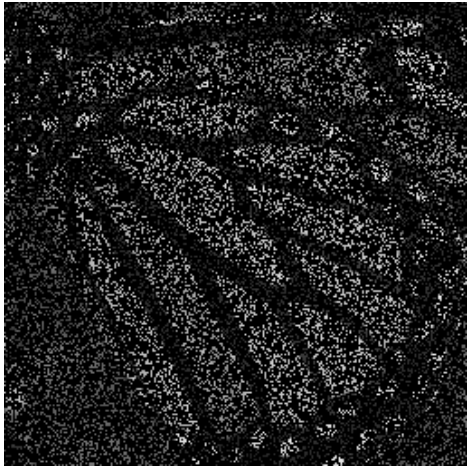


(a) $\mathbf{x}_1$ (24.95dB) (b) $\mathbf{z}_1$ (27.24dB) (c) $\mathbf{x}_6$ (27.59dB) (d) $\mathbf{z}_6$ (28.57dB) (e) $\mathbf{x}_{24}$ (29.12dB) (f) Convergence curves

Application: image inpainting

Solution of data subproblem:

$$\mathbf{x}_{k+1} = \frac{\mathbf{M} \odot \mathbf{y} + \alpha_k \mathbf{z}_k}{\mathbf{M} + \alpha_k}$$



Input

Output

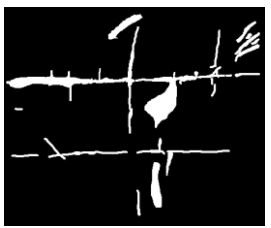
Application: image inpainting

Solution of data subproblem:

$$\mathbf{x}_{k+1} = \frac{\mathbf{M} \odot \mathbf{y} + \alpha_k \mathbf{z}_k}{\mathbf{M} + \alpha_k}$$

Input

Output



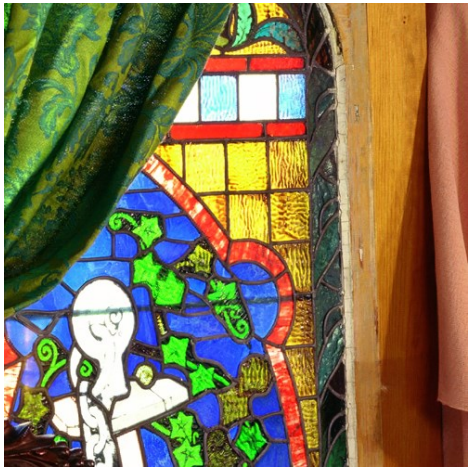
Mask M

Application: color image demosaicing

Solution of data subproblem:

$$\mathbf{x}_{k+1} = \frac{\mathbf{M} \odot \mathbf{y} + \alpha_k \mathbf{z}_k}{\mathbf{M} + \alpha_k}$$

Input

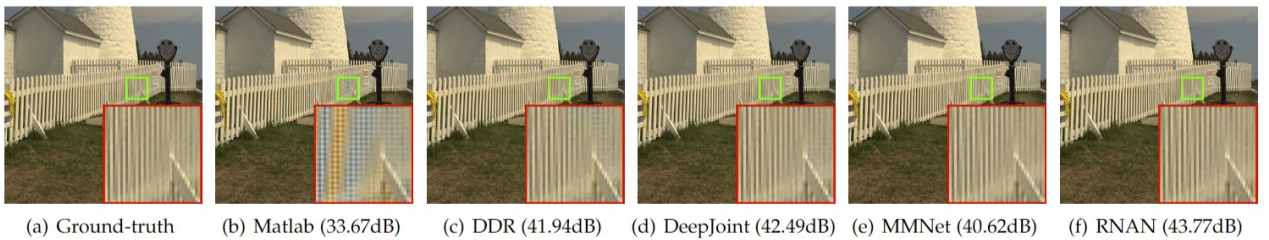


Output

Application: color image demosaicing

Demosaicing results of different methods on Kodak and McMaster datasets.

Datasets	Matlab	DDR	DeepJoint	MMNet	RLDD	RNAN	LSSC	IRI	FlexISP	IRCNN	IRCNN+	DPIR
Kodak	35.78	41.11	42.00	40.19	42.49	43.16	41.43	39.23	38.52	40.29	40.80	42.68
McMaster	34.43	37.12	39.14	37.09	39.25	39.70	36.15	36.90	36.87	37.45	37.79	39.39



(a) Ground-truth (b) Matlab (33.67dB) (c) DDR (41.94dB) (d) DeepJoint (42.49dB) (e) MMNet (40.62dB) (f) RNAN (43.77dB)

(g) LSSC (42.31dB) (h) IRI (39.49dB) (i) FlexISP (36.95dB) (j) IRCNN (40.18dB) (k) IRCNN+ (40.85dB) (l) DPIR (43.23dB)

Visual results comparison of different demosaicing methods on image kodim19 from Kodak dataset.

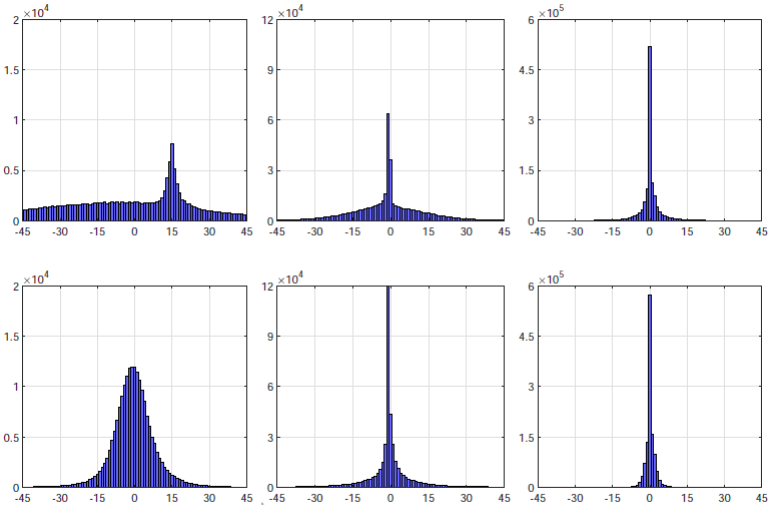
Discussion

What if the denoiser is trained for blind Gaussian denoising?

- Much lower PSNR

Noise distribution for the denoiser input.

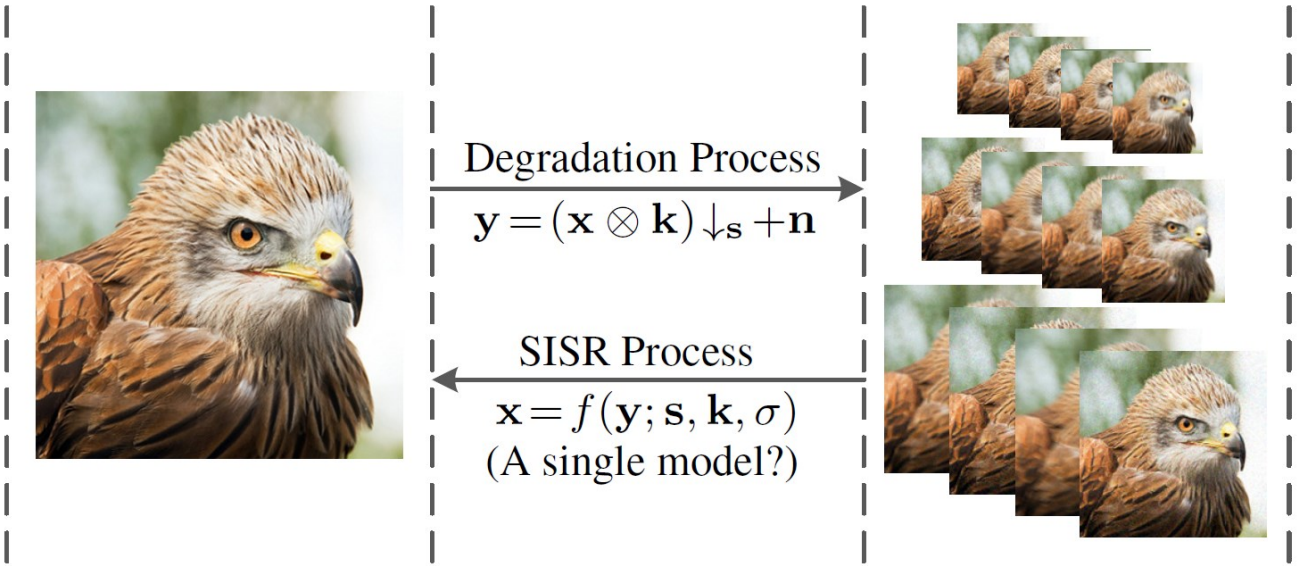
- Task-dependent



Histogram of the noise (difference) between the ground-truth and input of the denoiser in the first iteration (first row) and last iteration (second row) for (a) deblurring, (b) super-resolution, and (c) demosaicing. The histograms are based on x_1 and x_8.

The denoiser prior mostly removes the noise along with some fine details, while the subsequent data subproblem plays the role of alleviating the noise-irrelevant degradation and adding the lost details back.

Limited flexibility of learning based methods



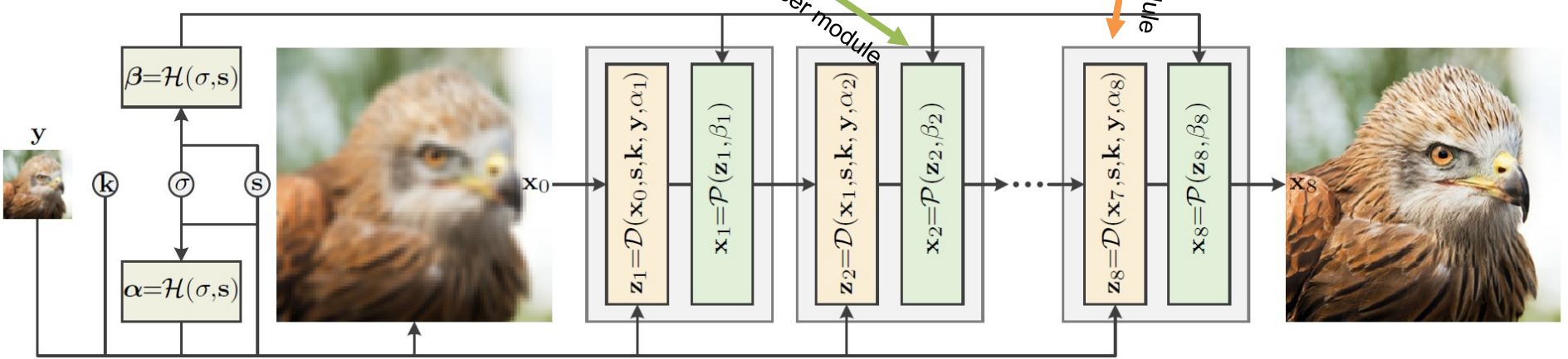
While the classical degradation model can result in various LR images for an HR image with different blur kernels, scale factors and noise, the study of learning a single deep model to invert all such LR images to HR image is still lacking.

Deep unfolding image restoration

(1) Problem $\mathbf{y} = (\mathbf{x} \otimes \mathbf{k}) \downarrow_s + \mathbf{n}$

(2) Objective function $E(\mathbf{x}) = \frac{1}{2\sigma^2}\|\mathbf{y} - (\mathbf{x} \otimes \mathbf{k})\downarrow_{\mathbf{s}}\|^2 + \lambda\Phi(\mathbf{x})$

(3) Optimization algorithm
$$\begin{cases} \mathbf{z}_k = \arg\min_{\mathbf{z}} \|\mathbf{y} - (\mathbf{z} \otimes \mathbf{k})\downarrow_{\mathbf{s}}\|^2 + \mu\sigma^2\|\mathbf{z} - \mathbf{x}_{k-1}\|^2 \\ \mathbf{x}_k = \arg\min_{\mathbf{x}} \frac{\mu}{2}\|\mathbf{z}_k - \mathbf{x}\|^2 + \lambda\Phi(\mathbf{x}). \end{cases}$$



Zhang, Kai, Luc Van Gool, and Radu Timofte. "Deep unfolding network for image super-resolution." CVPR, 2020.

1) Data module: FFT block

$$\mathbf{z}_k = \mathcal{D}(\mathbf{x}_{k-1}, \mathbf{s}, \mathbf{k}, \mathbf{y}, \alpha_{\mathbf{k}}).$$

Clearer

$$\mathbf{z}_k = \mathcal{F}^{-1}\left(\frac{1}{\alpha_k}\left(\mathbf{d} - \overline{\mathcal{F}(\mathbf{k})} \odot_{\mathbf{s}} \frac{(\mathcal{F}(\mathbf{k})\mathbf{d}) \Downarrow_{\mathbf{s}}}{(\overline{\mathcal{F}(\mathbf{k})}\mathcal{F}(\mathbf{k})) \Downarrow_{\mathbf{s}} + \alpha_k}\right)\right) \tag{6}$$

where $\mathbf{d}$ is defined as

$$\mathbf{d} = \overline{\mathcal{F}(\mathbf{k})}\mathcal{F}(\mathbf{y} \uparrow_{\mathbf{s}}) + \alpha_k \mathcal{F}(\mathbf{x}_{k-1})$$

with $\alpha_k \triangleq \mu_k \sigma^2$ and where the $\mathcal{F}(\cdot)$ and $\mathcal{F}^{-1}(\cdot)$ denote FFT and inverse FFT, $\overline{\mathcal{F}(\cdot)}$ denotes complex conjugate of $\mathcal{F}(\cdot)$, $\odot_{\mathbf{s}}$ denotes the distinct block processing operator with element-wise multiplication, *i.e.*, applying element-wise multiplication to the $\mathbf{s}\times\mathbf{s}$ distinct blocks of $\overline{\mathcal{F}(\mathbf{k})}$, $\Downarrow_{\mathbf{s}}$ denotes the distinct block downsampler, *i.e.*, averaging the $\mathbf{s}\times\mathbf{s}$ distinct blocks, $\uparrow_{\mathbf{s}}$ denotes the standard s-fold upsampler, *i.e.*, upsampling the spatial size by filling the new entries with zeros. It is especially noteworthy that Eq. (6) also works for the special case of deblurring when $\mathbf{s} = 1$. For

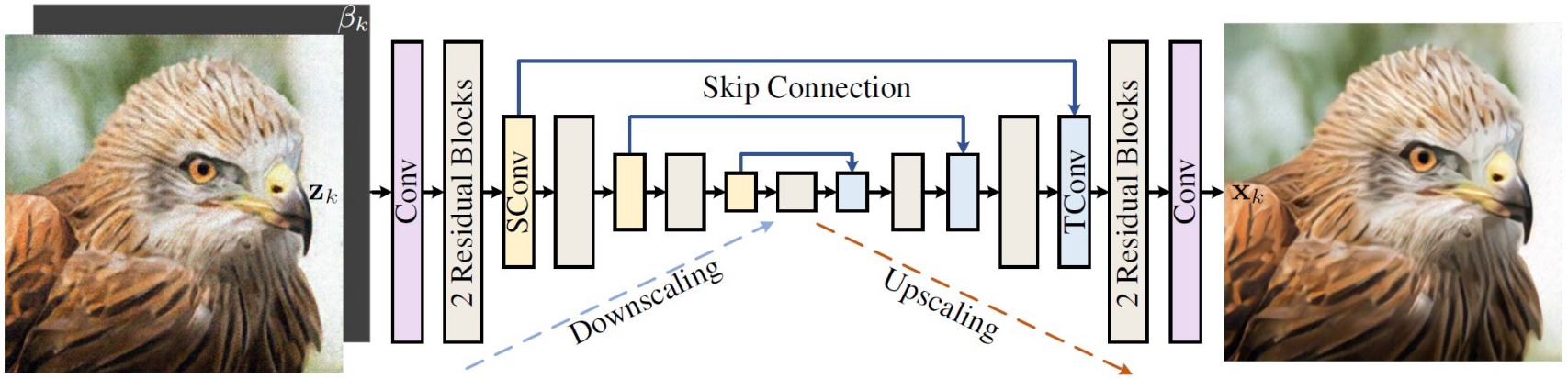
- Closed-form solution for the data subproblem
- Take the scale factor **s** and blur kernel **k** as input
- Impose degradation constraint on the solution
- Contain no trainable parameters, which in turn results in better generalizability

Ningning Zhao, Qi Wei, Adrian Basarab, Nicolas Dobigeon, Denis Kouame, and Jean-Yves Tourneret. Fast single image super-resolution using a new analytical solution for l2-l2 problems. IEEE TIP, 25(8):3683–3697, 2016.

2) Prior (denoiser) module: ResUNet

$$\mathbf{x}_k = \mathcal{P}(\mathbf{z}_k, \beta_k).$$

Cleaner



- Takes the noisy image and hyper-parameter noise level as input and output the denoised image.
- Has advantages of both U-Net and ResNet for effectiveness and efficiency.
- Shares parameters across iterations.

3) Hyper-parameter module

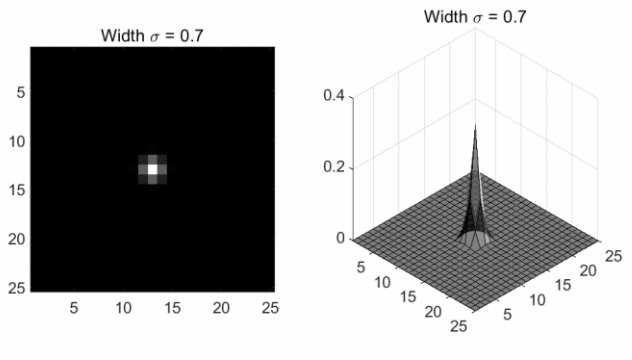
$$[\boldsymbol{\alpha}, \boldsymbol{\beta}] = \mathcal{H}(\sigma, \mathbf{s}).$$

(Input)-->Linear-->ReLU-->Linear-->ReLU-->Linear-->Softplus+1e-6-->(Output)

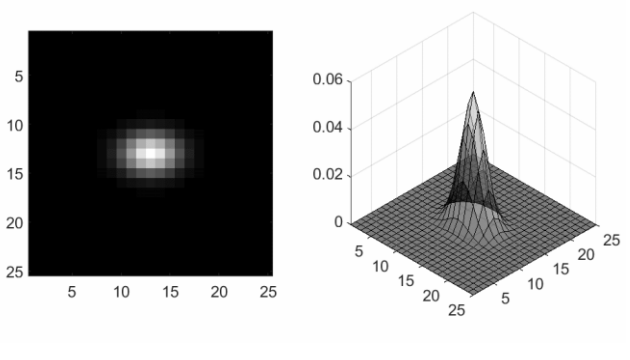
The hyper-parameter module acts as a ‘slide bar’ to control the outputs of the data module and prior module.

It consists of three fully connected layers with ReLU as the first two activation functions and Softplus as the last.

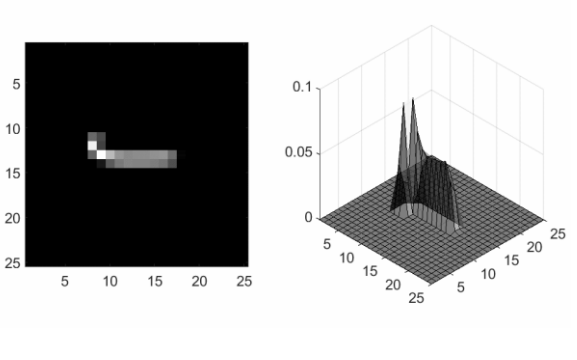
End-to-end training



Isotropic Gaussian



Anisotropic Gaussian



Motion

Scale factors: {1, 2, 3, 4}, applicable to the case of deblurring with s = 1
Noise levels: [0, 25]
Loss function: L1 loss for PSNR; L1 loss + VGG perceptual loss + PatchGAN with spectral norm loss for perceptual quality

Experiments: PSNR results

Table 1. Average PSNR(dB) results of different methods for different combinations of scale factors, blur kernels and noise levels. The best two results are highlighted in red and blue colors, respectively.

Methods	Scale Factor	Noise Level	Blur Kernel											
														
RCAN [68]	×2	0	29.48	26.76	25.31	24.37	24.38	24.10	24.25	23.63	20.31	20.45	20.57	22.04
	×3	0	24.93	27.30	25.79	24.61	24.57	24.38	24.55	23.74	20.15	20.25	20.39	21.68
	×4	0	22.68	25.31	25.59	24.63	24.37	24.23	24.43	23.74	20.06	20.05	20.33	21.47
ZSSR [50]	×2	0	29.44	29.48	28.57	27.42	27.15	26.81	27.09	26.25	14.22	14.22	16.02	19.39
	×3	0	25.13	25.80	25.94	25.77	25.61	25.23	25.68	25.41	16.37	15.95	17.35	20.45
	×4	0	23.50	24.33	24.56	24.65	24.52	24.20	24.56	24.55	16.94	16.43	18.01	20.68
IKC [21]	×4	0	22.69	25.26	25.63	25.21	24.71	24.20	24.39	24.77	20.05	20.03	20.35	21.58
IRCNN [63]	×2	0	29.60	30.16	29.50	28.37	28.07	27.95	28.21	27.19	28.58	26.79	29.02	28.96
	×3	0	25.97	26.89	27.07	27.01	26.83	26.76	26.88	26.67	26.22	25.59	26.14	26.05
	×3	2.55	25.70	26.13	25.72	25.33	25.28	25.18	25.34	24.97	25.00	24.64	24.90	24.73
	×3	7.65	24.58	24.68	24.59	24.39	24.24	24.20	24.27	24.02	23.94	23.77	23.75	23.69
	×4	0	23.99	25.01	25.32	25.45	25.36	25.26	25.34	25.47	24.69	24.39	24.44	24.57
USRNet	×2	0	30.47	30.88	30.48	29.40	29.03	29.03	29.19	28.19	30.83	30.57	30.53	30.65
	×3	0	27.07	27.68	27.81	27.78	27.62	27.58	27.65	27.48	27.57	27.36	27.38	27.31
	×3	2.55	26.92	27.33	27.17	26.72	26.50	26.53	26.66	26.08	26.80	26.70	26.61	26.41
	×3	7.65	26.39	26.46	26.06	25.53	25.42	25.34	25.45	24.96	25.32	25.40	25.14	24.93
	×4	0	25.19	25.86	26.08	26.18	26.09	26.03	26.06	26.20	25.78	25.41	25.62	25.56

USRNet with a single model significantly outperforms the other methods.

Experiments: visual results

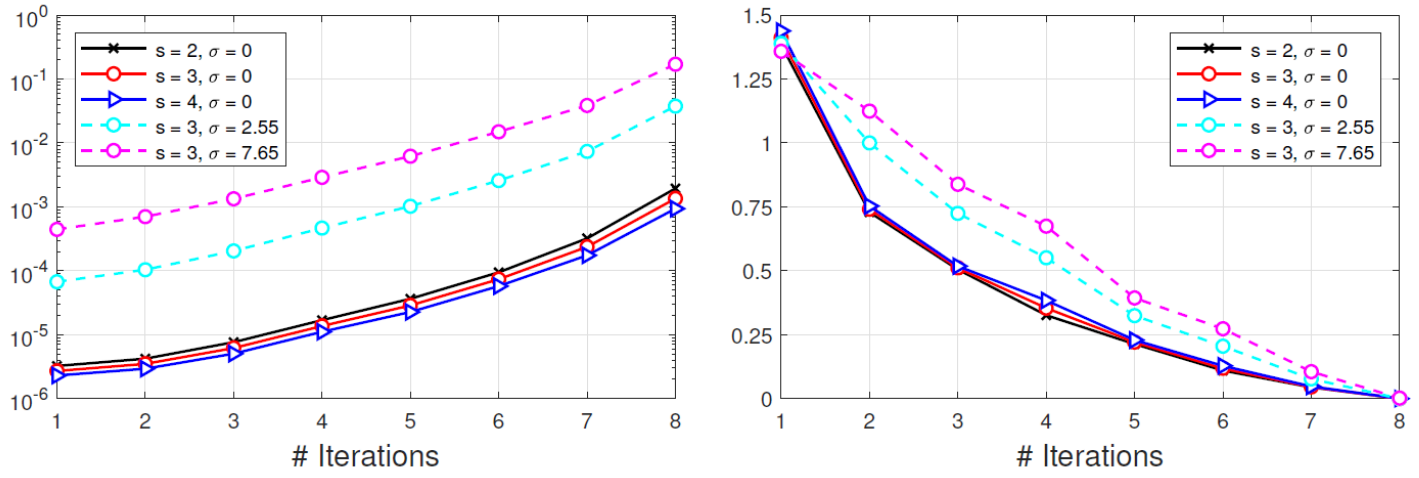
PSNR(dB) Zoomed LR (×4)	24.92/22.53/21.88 RCAN [68]	25.24/23.75/21.92 IKC [21]	25.42/23.55/26.45 IRCNN [63]	25.76/24.58/27.22 **USRNet** (ours)	24.80/22.44/21.51 RankSRGAN [67]	23.76/23.06/24.35 **USRGAN** (ours)

Experiments: analysis on data module and prior module

Figure 5. HR estimations in different iterations of USRNet (top row) and USRGAN (bottom row). The initial HR estimation $\mathbf{x}_0$ is the nearest neighbor interpolated version of LR image. The scale factor is 4, the noise level of LR image is 2.55 (1%), the blur kernel is shown on the upper-right corner of $\mathbf{x}_0$.

As expected, D can recover clearer result, while P can produce cleaner result.

Experiments: analysis on hyper-parameter module



(a) α (b) β

Figure 6. Outputs of the hyper-parameter module $\mathcal{H}$, *i.e.*, (a) α and (b) β, with respect to different combinations of $\mathbf{s}$ and σ.

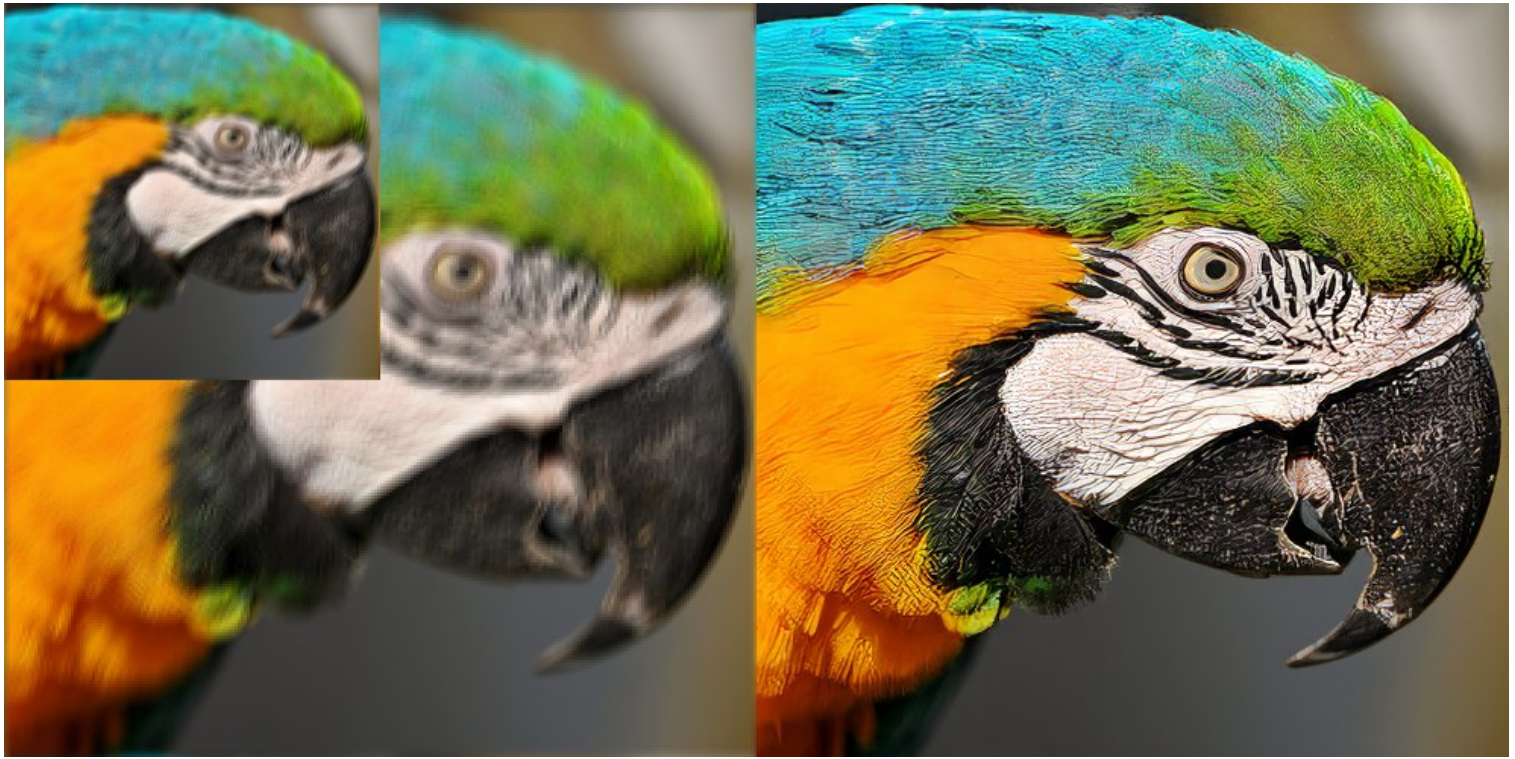
Experiments: generalizability

(a) Result of USRNet(x3) for kernel size 67x67

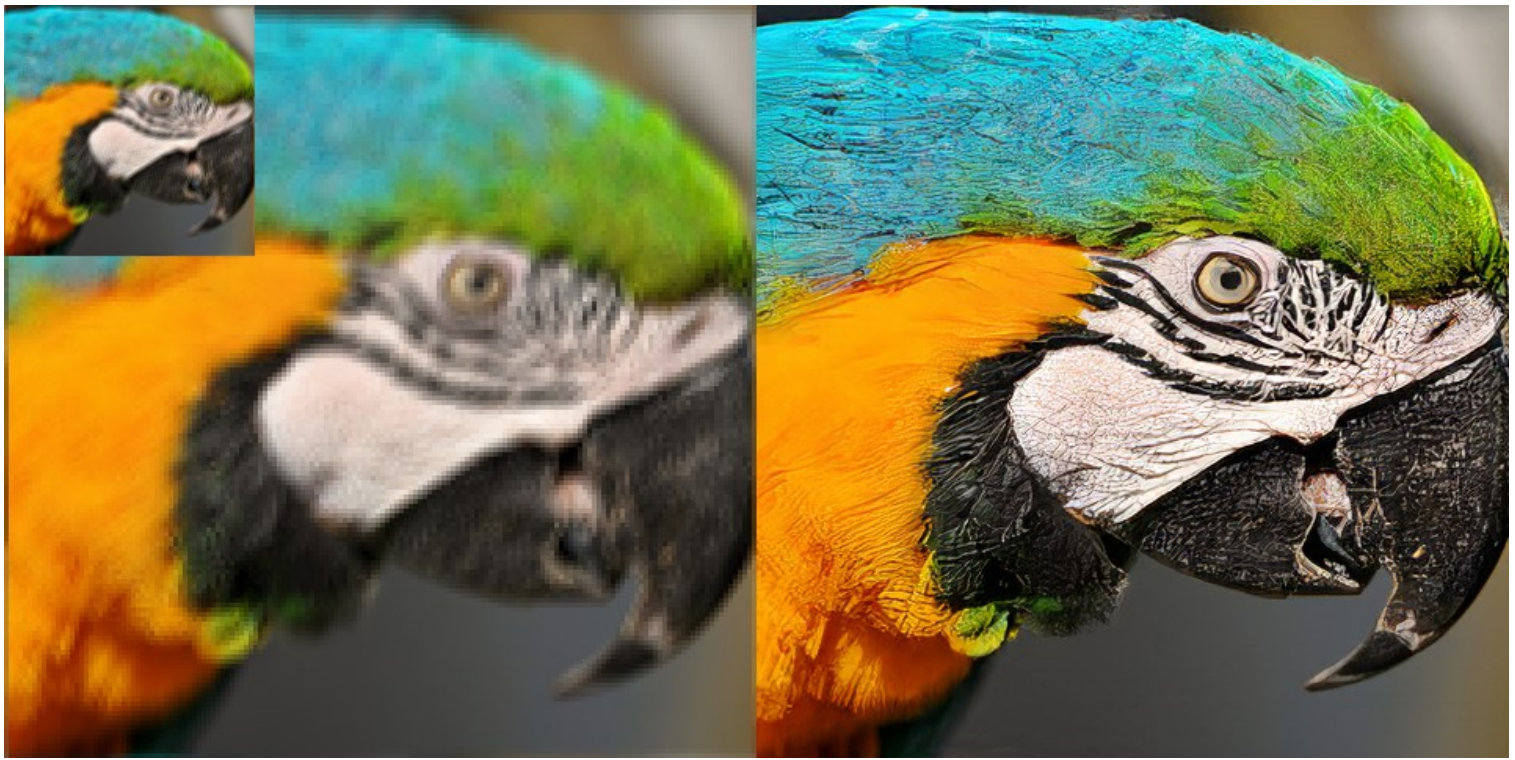
(b) Result of USRGAN(x3) for kernel size 70x70

Even trained with kernel size 25x25, USRNet and USRGAN generalize well to much larger kernel size.

Blurry low-resolution image SR (x2)



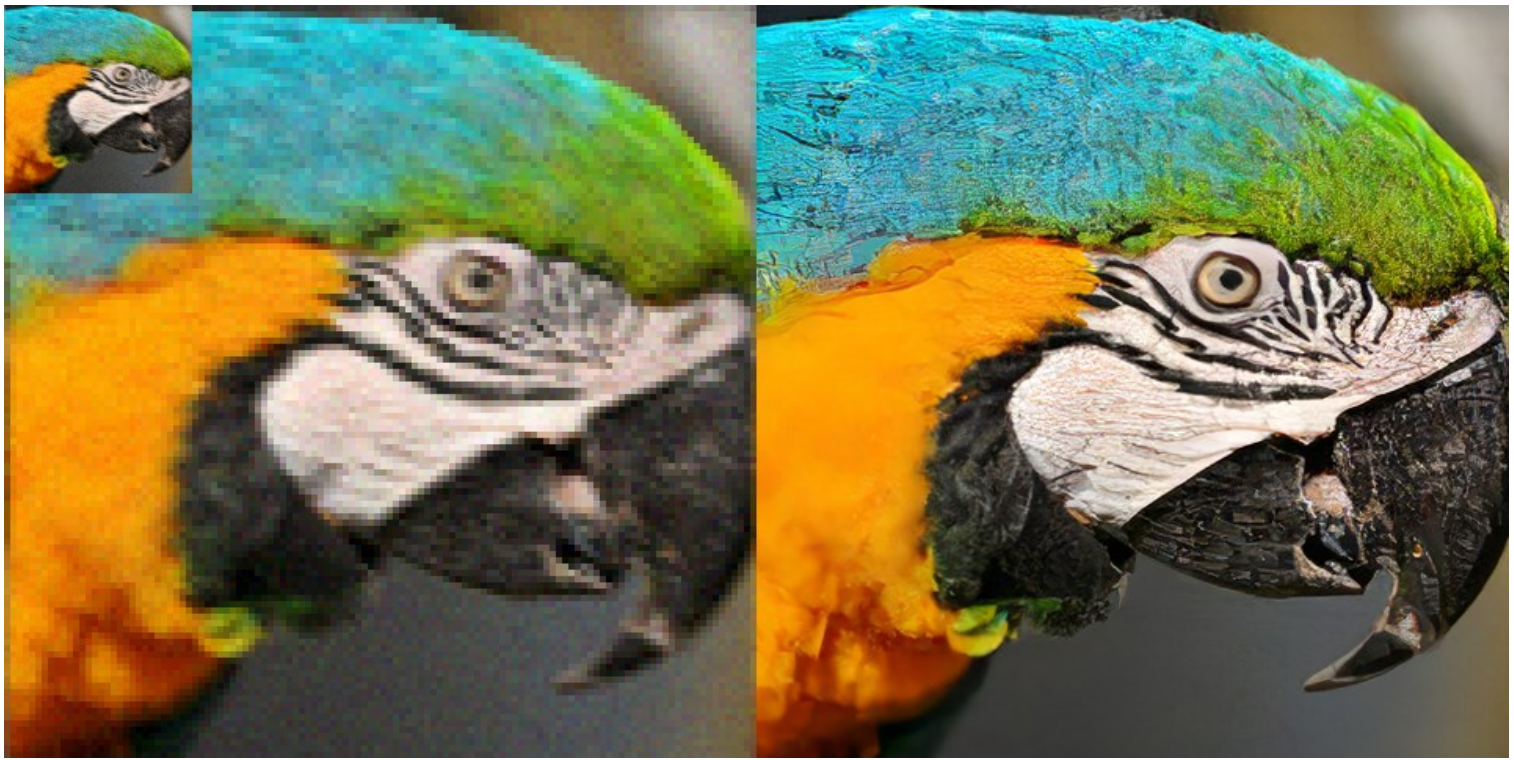
Blurry low-resolution image SR (x3)



Input

Output

Blurry noisy low-resolution image SR (x4)



Experiments: real image SR

(a) Zoomed LR (×4)

(b) USRNet

Experiments: real image SR

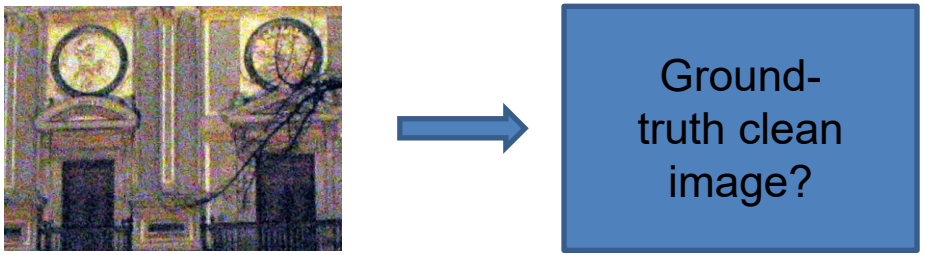
Experiments: image deblurring

Experiments: image deblurring

Deep blind image restoration

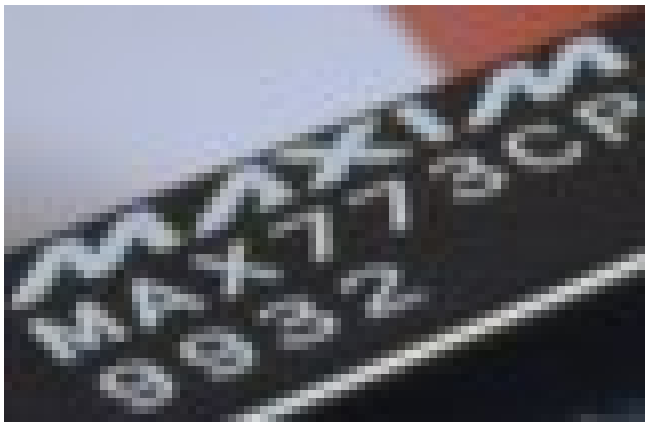
- The degradations in real images are too complex to be described by simple models
  - Non-Gaussian noise, signal dependent, non-uniform blur, compression artifacts, system distortions, …

- Deep learning for blind real image restoration?



Limitation of existing degradation models for practical super-resolution

Low-Resolution



Simple Interpolation (x4)

Result by ESRGAN [1]

Bicubic resizing degradation

- **Traditional degradation model**: $\boldsymbol{y} = (\boldsymbol{k} * \boldsymbol{x}) \downarrow + \boldsymbol{n}$, blur kernel $\boldsymbol{k}$ and AWGN $\boldsymbol{n}$
  - Advantage: simple and mathematically convenient
  - Drawback: does not match real degradation model
- **Data-specific degradation model**
  - Unsupervised training: pixel misalignment issue
  - Supervised training: difficult to accurately estimate the blur kernel

[1] Wang, Xintao, et al. "ESRGAN: Enhanced super-resolution generative adversarial networks." ECCVW, 2018.

Limitation of existing degradation model for practical denoising

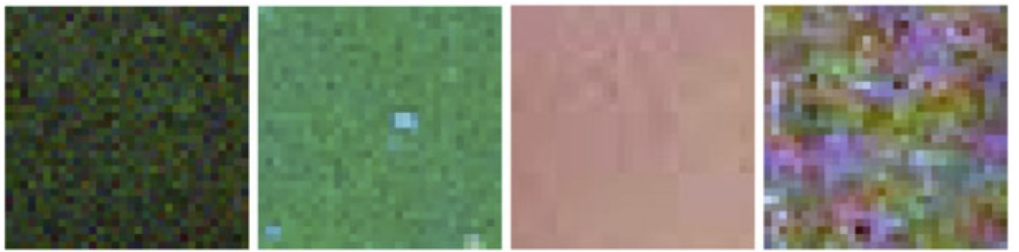
Noisy Input

Result by DnCNN [2]

Gaussian noise degradation

- **Gaussian noise model**: $\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{n}$, AWGN $\boldsymbol{n}$
  - Mathematically convenient but do not work well for real noisy images
- **Data-specific degradation model**
  - Lack generalization ability
  - Laborious collection

Real noises are much more complex!

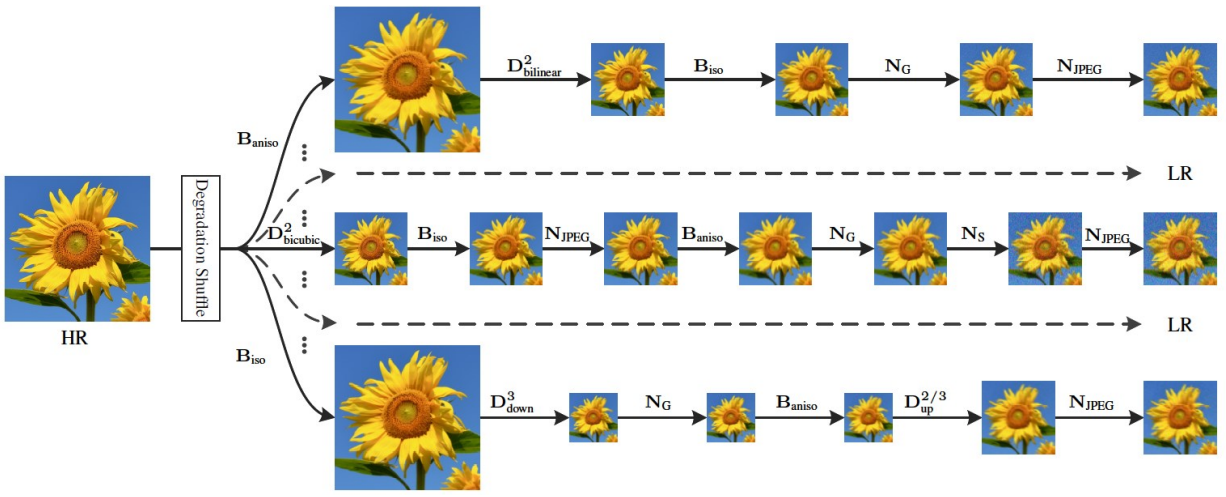


Different noise types: additive white Gaussian noise; interchannel correlated Gaussian Noise; JPEG compression noise; low-frequency noise.

[2] Zhang, Kai, et al. "Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising." IEEE TIP, 2017.

A new degradation model for practical super-resolution

Design a complex degradation model and learn a blind model with purely **synthetic** data



Schematic illustration of the proposed degradation model for scale factor 2. For an HR image, the randomly shuffled degradation sequences are first performed, then a JPEG compression degradation is applied to save the LR image into JPEG format.

A. Blur
- Isotropic Gaussian blur
- Anisotropic Gaussian blur

B. Downsampling
- Nearest
- Bilinear/Bicubic
- Down-up

C. Noise
- Gaussian noise
- JPEG compression noise
- Processed camera sensor noise

◆ **Random shuffle**

A. Blur

(a) (b) (c)

An illustration of different Gaussian blur operations with (a) isotropic Gaussian kernel with width 1.6, (b) anisotropic Gaussian kernel with axis lengths 3, 6 and rotation angle =4, and (c) a cascade of the above two kernels on the 256x256 ‘Butterfly’ image.

Model the blur from both
- HR space
- LR space

with
- isotropic Gaussian kernels
- anisotropic Gaussian kernels

B. Downsampling

(a) (b) (c)

An illustration of different downsampling methods, i.e., (a) nearest, (b) bilinear and (c) bicubic, with scale factor 2. The blur kernel (with width 1.0) for the nearest downsampling is shown on the upper-left corner of the image.

Different downsampling methods

- Nearest
- Bilinear
- Bicubic
- Down-up-sampling

C. Noise

(a) (b) (c)

An illustration of different noise types, (a) grayscale Gaussian noise with standard deviation 25, (b) JPEG compression noise with quality factor 30, and (c) processed camera sensor noise.

Noise is ubiquitous in real images as it can be caused by different sources.

Different noise types

- Gaussian noise
- JPEG compression noise
- Processed camera sensor noise

◆ Random shuffle

2015 $y = (x * k) \downarrow_s$

Revisiting Single Image Super-Resolution Under Internet Environment: Blur Kernels and Reconstruction Algorithms

Kai Zhang, Xiaoyu Zhou, Hongzhi Zhang, and Wangmeng Zuo(✉)

2018 $\boldsymbol{y} = (\boldsymbol{x} \downarrow_s) \otimes \boldsymbol{k} + \boldsymbol{n}$

Learning a Single Convolutional Super-Resolution Network for Multiple Degradations

Kai Zhang[1,2,3], Wangmeng Zuo[1,*], Lei Zhang[2]

2019 $\mathbf{y} = (\mathbf{x} \downarrow_s) \otimes \mathbf{k} + \mathbf{n}$

Deep Plug-and-Play Super-Resolution for Arbitrary Blur Kernels

Kai Zhang[1,2], Wangmeng Zuo[1,3,*], Lei Zhang[2,4]

2020 $\mathbf{y} = (\mathbf{x} \otimes \mathbf{k}) \downarrow_{\mathbf{s}} + \mathbf{n},$

Deep Unfolding Network for Image Super-Resolution

Kai Zhang Luc Van Gool Radu Timofte

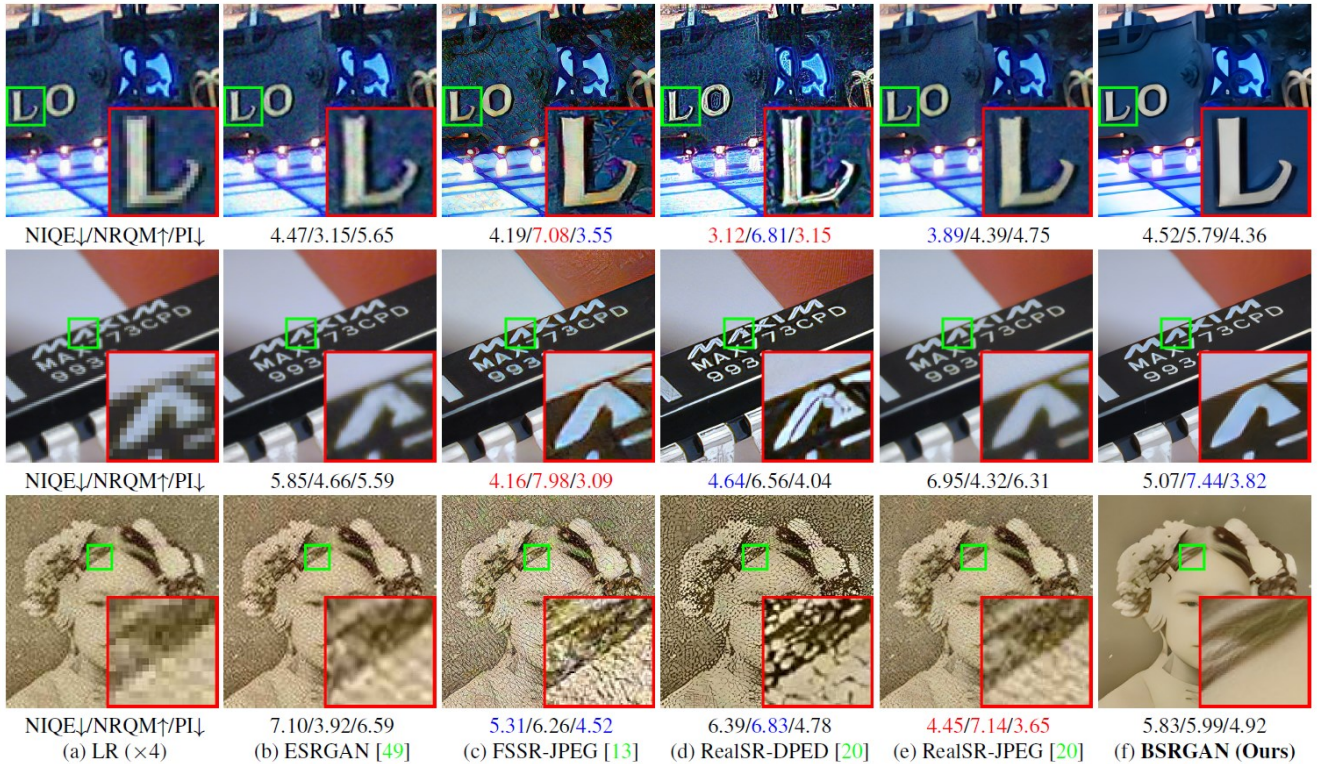
2021 

Designing a Practical Degradation Model for Deep Blind Image Super-Resolution

Kai Zhang[1] Jingyun Liang[1] Luc Van Gool[1,2] Radu Timofte[1]

- **Generalized degradation model**
  Bicubic and traditional degradation models are special cases
- **Large blur degradation space**
  Two blur operations and four downsampling methods
- **Large noise degradation space**
  Blur and downsampling could change noise characteristics

Visual results for real images



Results of different methods on super-resolving real images from RealSRSet with scale factor 4.

IQA (Image Quality Assessment) metric should also be **updated with new image degradation types**.

Visual Results

Low-Resolution (x4)

Proposed

Visual Results

- Models for challenge results
  - DF2K for corrupted images with processing noise.
  - DPED for real images taken by cell phone camera.
- Extended models
  - DF2K-JPEG for compressed jpeg image.

DF2K_JPEG JPEG Images

DPED Smartphone Images

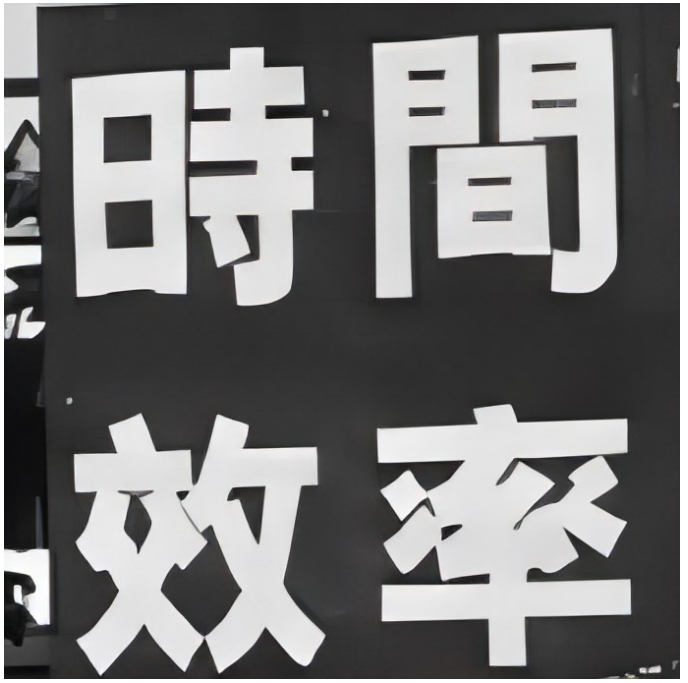
Visual Results

Low-Resolution (x4)

Proposed

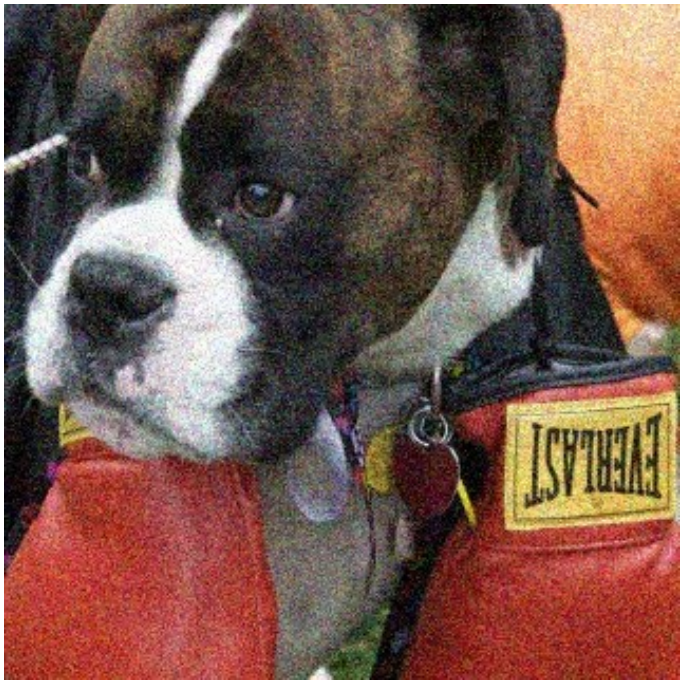
Visual Results

Low-Resolution (x4)



Proposed

Visual Results



Low-Resolution (x4)

Proposed

Cited by “Stable Diffusion”

High-Resolution Image Synthesis with Latent Diffusion Models

Robin Rombach[1] * Andreas Blattmann[1] * Dominik Lorenz[1] Patrick Esser[R] Björn Ommer[1]
[1]Ludwig Maximilian University of Munich & IWR, Heidelberg University, Germany [R]Runway ML

https://github.com/CompVis/latent-diffusion

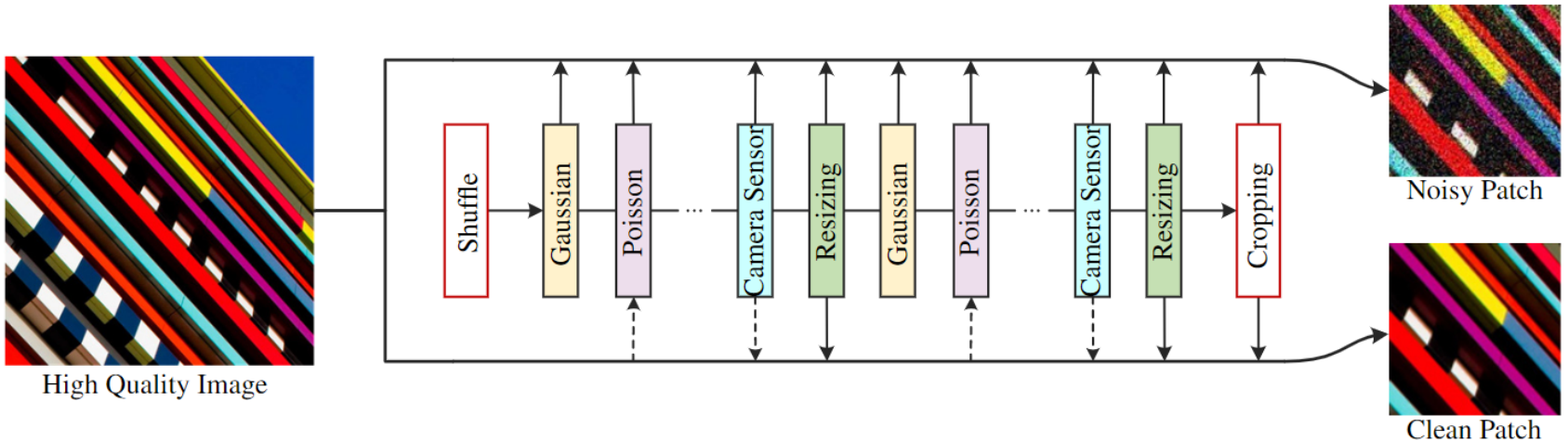
Text-to-Image with Stable Diffusion

D.6.1 LDM-BSR: General Purpose SR Model via Diverse Image Degradation

To evaluate generalization of our LDM-SR, we apply it both on synthetic LDM samples from a class-conditional ImageNet model (Sec. 4.1) and images crawled from the internet. Interestingly, we observe that LDM-SR, trained only with a bicubicly downsampled conditioning as in [72], does not generalize well to images which do not follow this pre-processing. Hence, to obtain a superresolution model for a wide range of real world images, which can contain complex superpositions of camera noise, compression artifacts, blurr and interpolations, we replace the bicubic downsampling operation in LDM-SR with the degration pipeline from [105]. The BSR-degradation process is a degradation pipline which applies JPEG compressions noise, camera sensor noise, different image interpolations for downsampling, Gaussian blur kernels and Gaussian noise in a random order to an image. We found that using the bsr-degredation process with the original parameters as in [105] leads to

[105] K. Zhang, Jingyun Liang, Luc Van Gool, and Radu Timofte. Designing a practical degradation model for deep blind image super-resolution. *ArXiv*, abs/2103.14006, 2021. 23

Degradation model for practical blind denoising



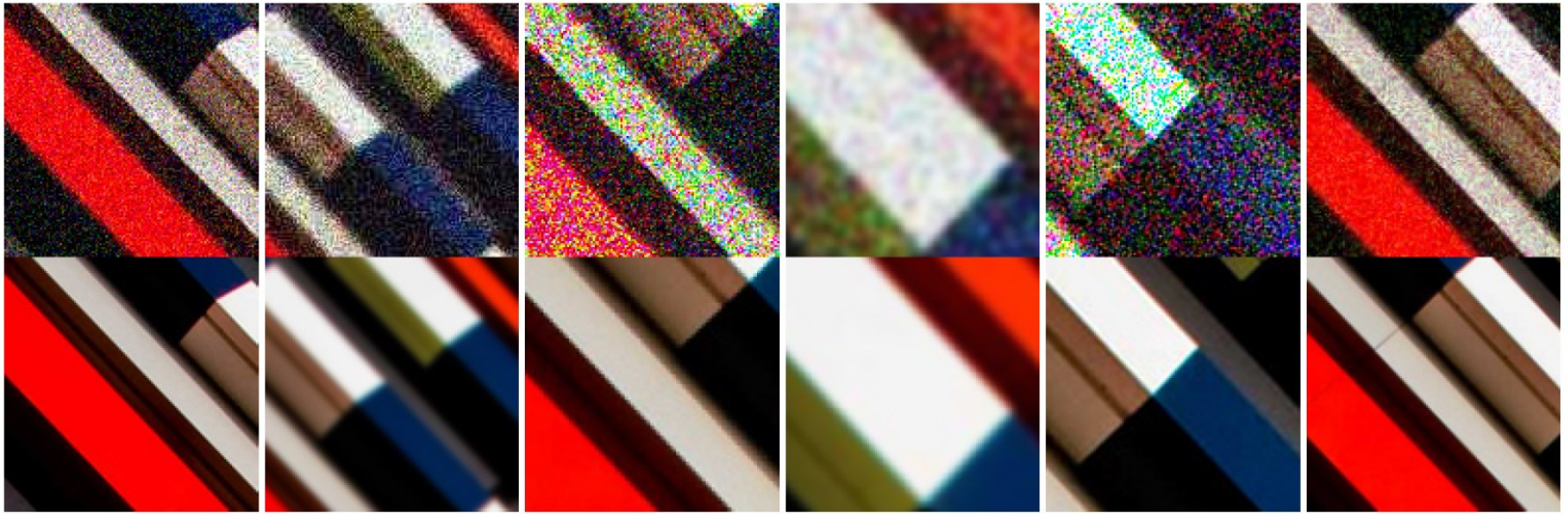
Schematic illustration of the proposed paired training patches synthesis pipeline.

- Gaussian, Poisson, speckle, JPEG compression, and processed camera sensor noises
- Resizing
- Random shuffle strategy
- Double degradation strategy

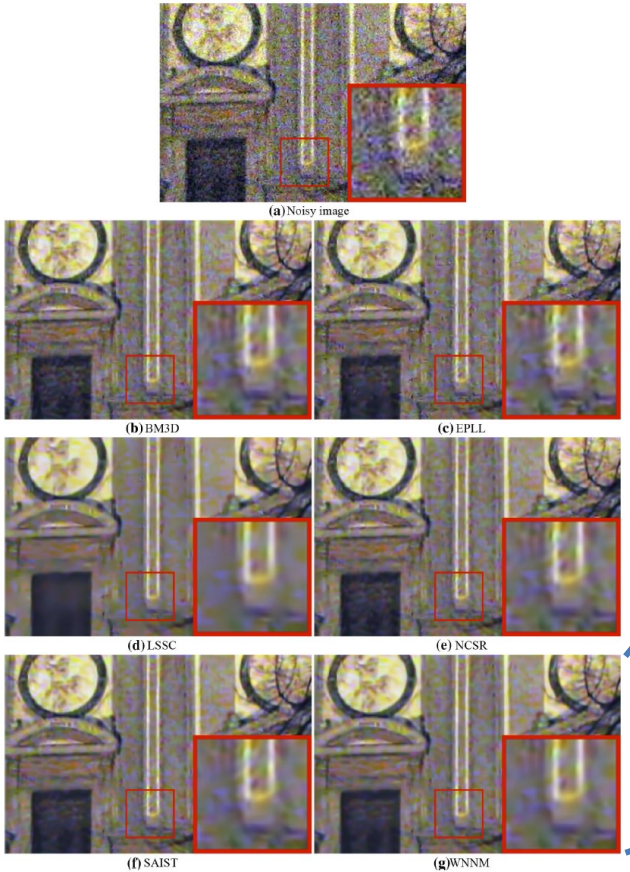
Synthesized noisy/clean patch pairs

Noisy

Clean

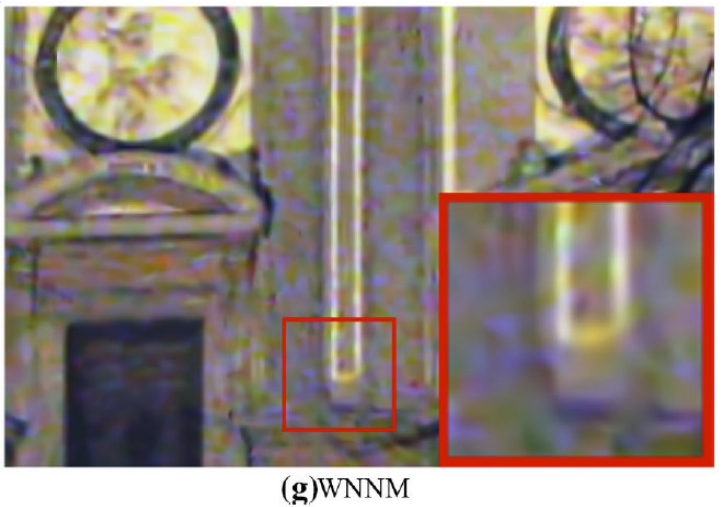


Synthesized noisy/clean patch pairs via our proposed training data synthesis pipeline

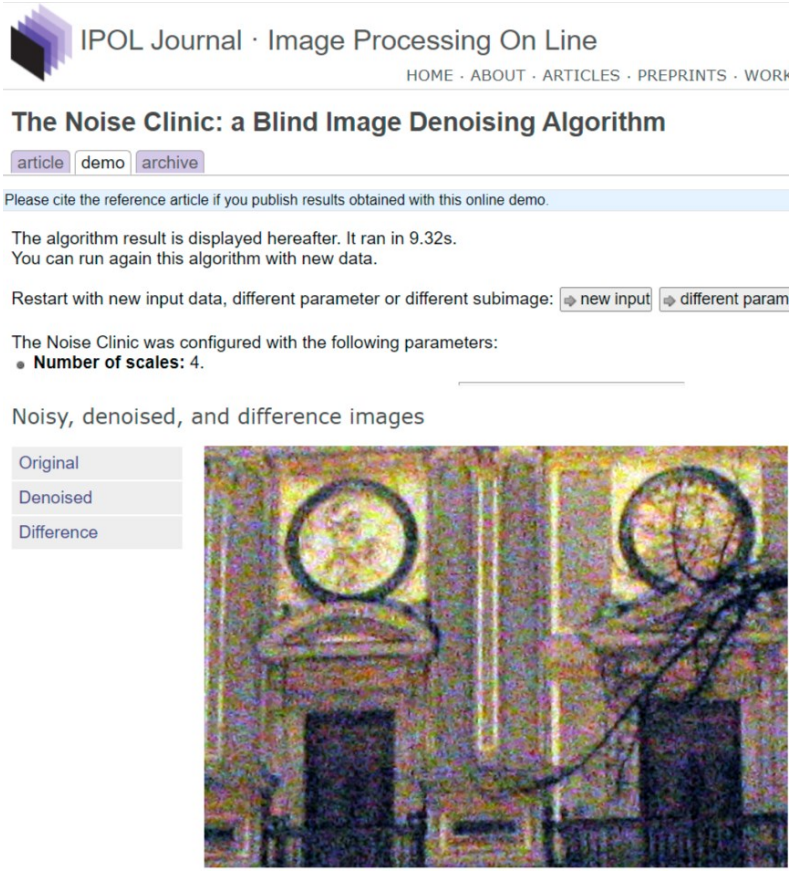


Weighted Nuclear Norm Minimization and Its Applications to Low Level Vision

Shuhang Gu[1] · Qi Xie[2] · Deyu Meng[2] · Wangmeng Zuo[3] · Xiangchu Feng[4] · Lei Zhang[1]



Figures from WNNM (IJCV 2017)



Noise Clinic (2015)

Lebrun, Marc, Miguel Colom, and Jean-Michel Morel. "The noise clinic: a blind image denoising algorithm." Image Processing On Line 5 (2015): 1-54.

Noisy Input

Noise Clinic (2015)

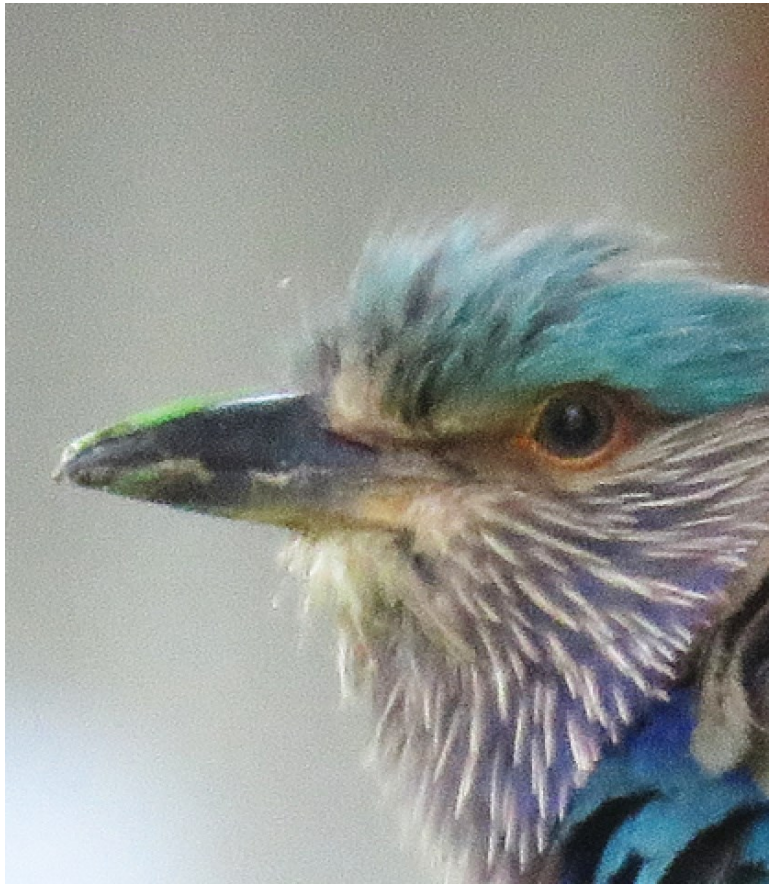
Lebrun, Marc, Miguel Colom, and Jean-Michel Morel. "The noise clinic: a blind image denoising algorithm." Image Processing On Line 5 (2015): 1-54.

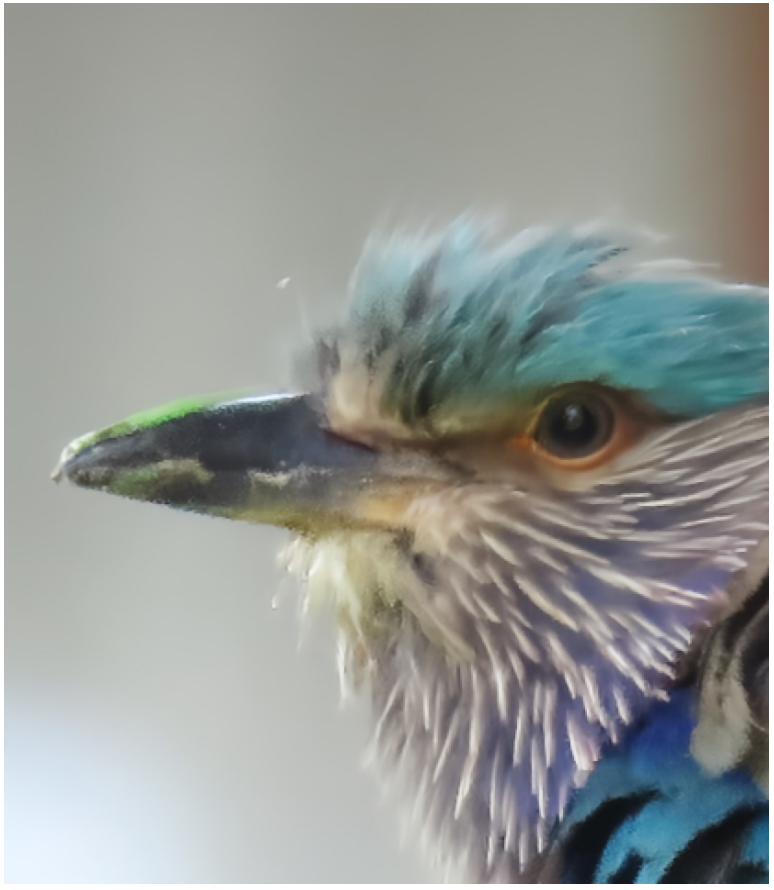
CBDNet (2019)

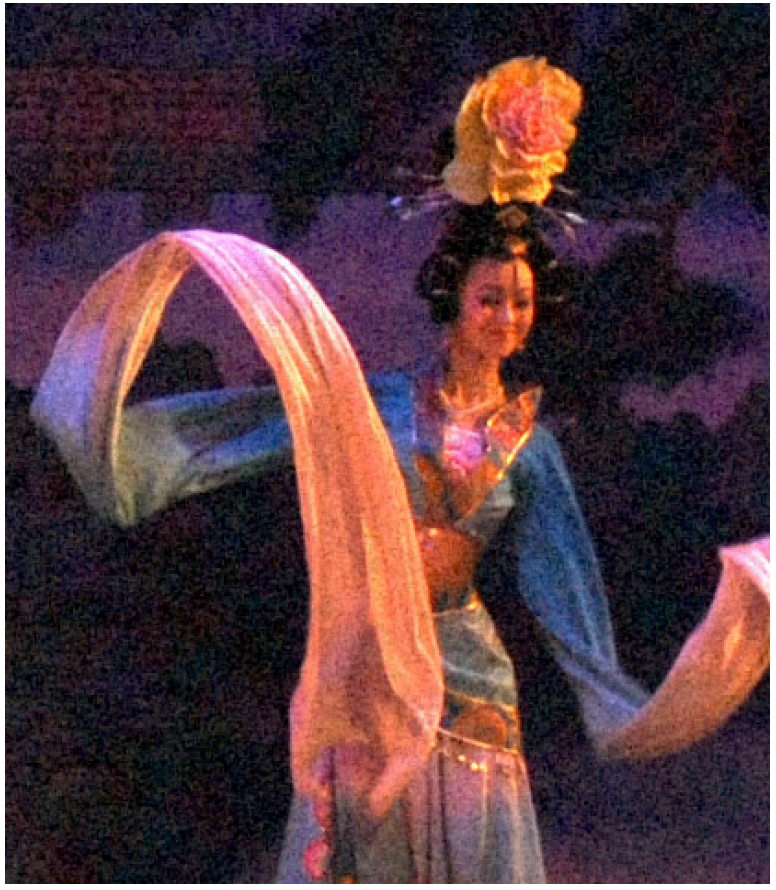
DeamNet (2021)

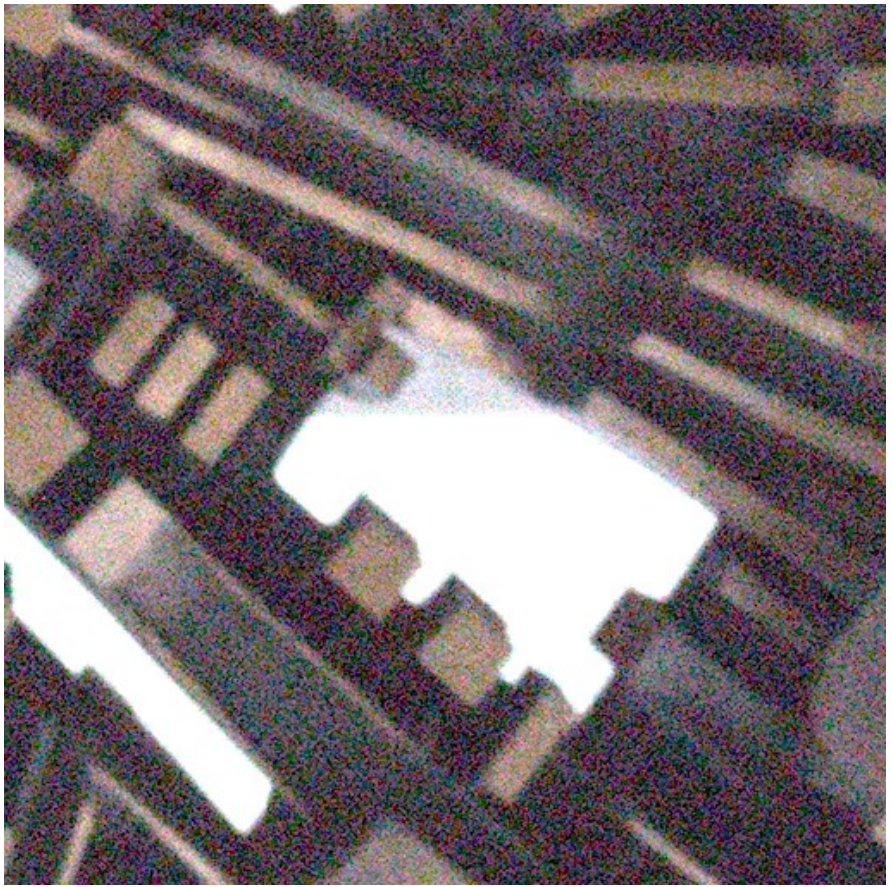
Noisy Input

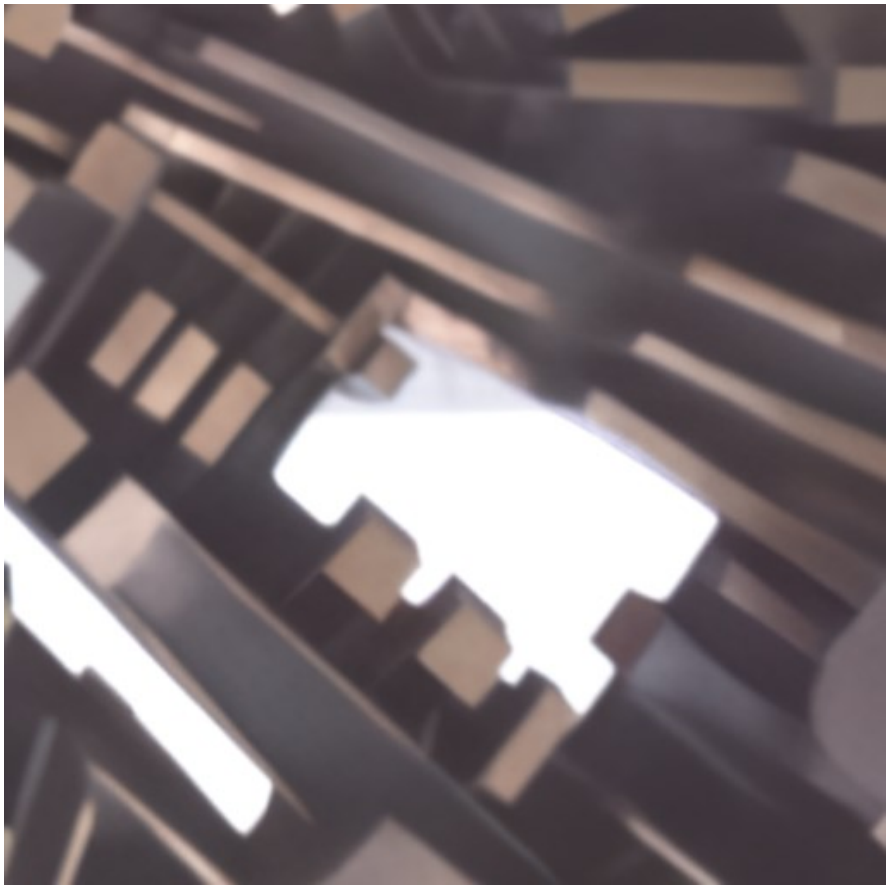
Our result



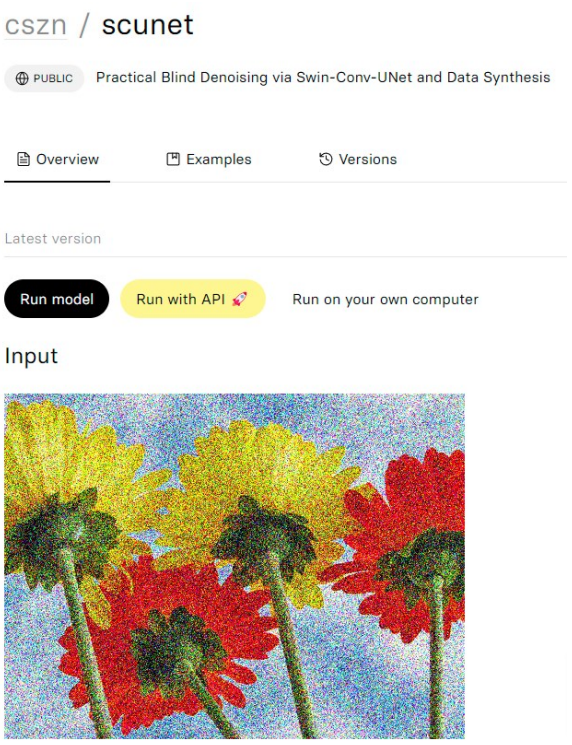








Online demo



https://replicate.com/cszn/scunet


THANKS