Seminar Report

"Plug-and-Play Image Restoration with Deep Denoiser Prior"

By Kai Zhang, Yawei Li, Wangmeng Zuo, Lei Zhang, Luc Van Gool and Radu Timofte

Yoel Bokobza

22 July 2022

Contents

1 Motivation and Background

1.1 Image Restoration

Images are often degraded during the data acquisition process. The degradation may comes in many forms such as noise, blurring, down sampling, mosaicing, etc. The purpose of image restoration (IR) is to estimate the original image from the degraded image. There are many applications of IR such as medical imaging, astronomical imaging, old photo restoration, etc. The authors use the standard model of degradation including the noise as $y = \tau(x) + n$, where x is the clean latent image and $n \sim \mathcal{N}(0, \sigma^2)$ is an additive white Gaussian noise (AWGN) of standard deviation σ.

1.2 Related Works

Plug-and-play networks utilize deep denoisers [1] as learned proximal mappings. A denoiser is used to solve an optimization problem that contains a regularization term without having to directly compute it, i.e., without expressing the desired signal domain in a tractable form. Plug-and-Play for IR is two-folded: First, the problem is decoupled into two sub-problems: The data sub-problem, and the prior sub-problem. In Section 2, we formulate the problem and define these two terms. The second step is to apply any off-the-shelf denoiser such as K-SVD [2], non-local means [3], and BM3D [4], which are classical methods for the prior sub-problem. The major advantage of such methods is that they are flexible to handle various IR tasks using the same denoiser, and specifying different $\tau(\cdot)$ functions. This work is an extension of the paper [5], which written by the same authors. Compared with their previous work, the authors suggest more flexible and powerful deep convolutional neural network (CNN) denoiser, which is not only outperforms the state-of-the-art (SOTA) deep Gaussian denoising models but also is suitable to solve plug-and-play IR for different noise levels.

2 Problem Formulation

Recall that the degraded image model is, $y = \tau(x) + n$, then from a Bayesian perspective, the solution $\hat{x}$ can be obtained by solving a maximum a posteriori (MAP) estimation problem

$$\hat{x} = \arg\max_x \{\log p(x|y)\} = \arg\max_x \{\log p(y|x) + \log p(x)\}, \tag{1}$$

where $\log p(y|x)$ represents the log-likelihood of observation y, and $\log p(x)$ represents the prior of clean image x and is independent of degraded image y.
The MAP estimator intuitively finds the most probable latent image, given an observed degraded image, by maximizing the posterior probability. Following the formulation of the degradation model, the observed image y given the latent image x is Gaussian distributed with $\tau(x)$ mean and standard deviation σ. Therefore, the MAP estimator corresponding to the following optimization problem can be written as,

$$\hat{x} = \arg\min_x \left\{ \frac{1}{2\sigma^2} \|y - \tau(x)\|^2 + \lambda \Re(x) \right\}, \tag{2}$$

where the solution minimizes an energy function composed of a data term $\frac{1}{2\sigma^2}\|y - \tau(x)\|^2$ and a regularization term $\lambda\Re(x)$ with regularization parameter λ, which corresponding

to the prior. Note that directly solving this problem can be very hard, because usually, we don't have a complete knowledge about the prior distribution of the data.

3 Method

3.1 Half Quadratic Splitting

The half quadratic splitting (HQS) [6] and alternating direction method of multipliers (ADMM) [7] are algorithms which solve optimization problems by breaking them into smaller sub-problems, each of which are then easier to handle. In the course and also as described in [1], the method suggested for variable splitting is the ADMM method. In this work the authors used the HQS method for variable splitting because its simplicity and fast convergence. In particular, the HQS splits the problem into two sub-problems, compared with the ADMM, which splits it into three sub-problems.

HQS first introduces an auxiliary variable z, resulting in a constrained optimization problem given by

$$\begin{cases} \hat{x} = \arg\min_x \left\{ \frac{1}{2\sigma^2} \|y - \tau(x)\|^2 + \lambda \Re(z) \right\} \\ s.t. \quad z = x \end{cases}, \tag{3}$$

where we replace x by z in the regularization term. This problem relaxed by minimizing the following cost function, with respect to both x, and z:

$$L_\mu(x, z) = \frac{1}{2\sigma^2} \|y - \tau(x)\|^2 + \lambda \Re(z) + \frac{\mu}{2} \|z - x\|^2, \tag{4}$$

where μ is a penalty parameter, which tries to enforce the eliminated constraint. Such problem can be addressed by iteratively solving the following sub-problems for x and z while keeping the rest of the variables fixed,

$$x_k = \arg\min_x \left\{ \|y - \tau(x)\|^2 + \mu\sigma^2 \|x - z_{k-1}\|^2 \right\} \tag{5a}$$

$$z_k = \arg\min_z \left\{ \frac{1}{2\sqrt{(\frac{\lambda}{\mu})^2}} \|z - x_k\|^2 + \Re(z) \right\} \tag{5b}$$

Let's refer to the first sub-problems as the data sub-problem and to the second sub-problems as the prior sub-problem. Note that the data term and the regularization term are decoupled. In particular, the prior sub-problem (5b) is independent of the data term (5a) and thereby we can apply a single solver (denoiser in our case) for different types of degradation operations.

This strategy has the benefit that the data problem (5a) is quadratic when the degradation operation is linear, i.e., it can be expressed by matrix multiplication. Thus, in such a case, there is a closed-form solution for the data problem.
The second sub-problem (5b), from a Bayesian perspective, corresponds to Gaussian denoising on x_k with noise level $\sqrt{\lambda/\mu}$, (see proof in 6.1). This implies that the prior sub-problem can be expressed equivalently as,

$$z_k = Denoiser\left(x_k, \sqrt{\frac{\lambda}{\mu}}\right) \tag{6}$$

3.2 The Suggested Denoiser - DRUNet

DRUNet is a deep denoiser network that combines residual blocks [8] with U-Net [9] for effective denoiser prior modeling. Because in this algorithm the denoiser network is applied iteratively, and the effective noise level changes during iterations, the denoiser network should be flexible in the sense that it can handle a wide range of noise levels. To address different noise levels the input to the network consist of the input image (RGB or Grayscale) concatenated with the noise level map along the channel axis. The noise level map is a matrix of the same size as the image, where each of its entries has the same value which is the noise level $\sqrt{\lambda/\mu}$. We think the authors used the level map instead of a single value because it is simpler to insert the information corresponding to the noise level as an additional channel. The DRUNet architecture is depicted in Fig. 1

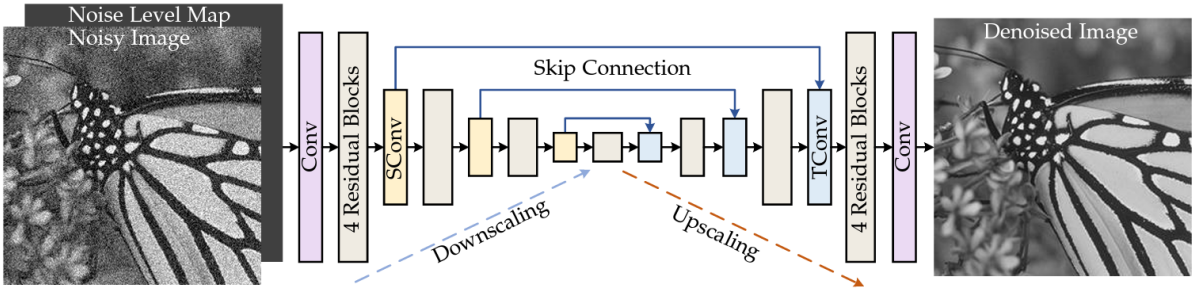


Figure 1: DRUNet architecture - U-Net of four scales combined with four successive residual blocks downscaling and upscaling of each scale

The downscaling is done via stride convolution without padding and the upscaling is implemented via transposed convolution. The network trained over 8794 different images with the $L1$ loss and Adam optimizer [10]. To handle a wide range of noise levels, during training, the noise level σ is randomly chosen from $[0, 50]$.

3.3 Parameters Setting Methodology

From the alternating iterations between (5a) and (5b), it is easy to see that there involved three adjustable parameters, including the penalty parameter μ, the regularization parameter λ, and the total number of iterations K.

In Eq. (3), when we add an auxiliary variable z in order to decouple the data term and regularization term, we impose the constraint $z = x$. Since we solve this via an iterative method, we expect that x_k will converge to z_k as the iteration number k increases. As according to Eq. (4), the coefficient of the expression $\|z - x\|^2$ is μ, then to guarantee x_k and z_k will converge to a fixed point, a large μ is needed. Therefore, the bigger the gap between x_k and z_k, the bigger the value of the term $\|z_k - x_k\|^2$, which is non-negative, and this implies that the cost function will punish more for the distance between x_k and z_k being large, for larger μ.

The common way for setting μ is to gradually increase it, resulting in a sequence of $\mu_1 < \mu_2 < \mu_3 < \cdots < \mu_K$. According to (6), μ_k controls the noise level in the k-th iteration of the denoiser prior according to the equation $\sigma_k \triangleq \sqrt{\lambda/\mu_k}$. Eventually, instead of increasing μ, the strategy is to decrease σ_k in log space, which implies that μ_k increases as k increases. Note that this makes sense because we expect for noise reduction as the iteration number increases.

3.4 Periodical Geometric Self-Ensemble

Geometric self-ensemble is a commonly-used strategy to boost IR performance [11]. This methods applies invertible transforms to the input image (such as flipping and rotation) to generate different images, then gets the corresponding restored images after feeding the model with these images, and finally produces the averaged result after applying the inverse transformation for any transformed image. In this paper, the authors used a modified version of this method to reduce computational complexity. They suggests to simply apply a set of 8 transformations for 8 successive iterations. A single transformation for each iteration before feeding it into denoiser and then applying the inverse transformation. The authors state that the geometric self-ensemble can generally improve the peak signal to noise ratio (PSNR) by $0.02dB \sim 0.2dB$.

3.5 DPIR Algorithm Pseudocode

Algorithm 1 Plug-and-play image restoration with DPIR

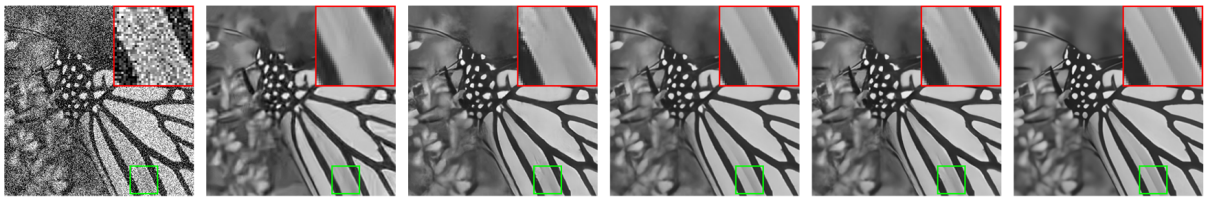
Input: Deep denoiser prior model, degraded image y, degradation operation τ, image noise level σ, σ_k of denoiser prior model at k -th iteration for a total of K iterations, trade-off parameter λ

Output: Restored image z_K

Initialize z_0 from y, pre-calculate $\alpha_k = \sigma^2 \frac{\lambda}{\sigma_k^2}$, for $k = 1, 2, ..., K$
for $k = 1, 2, ..., K$ **do**
    $x_k = \arg\min_x \{\|y - \tau(x)\|^2 + \alpha_k \|x - z_{k-1}\|^2\}$
    $z_k = Denoiser(x_k, \sigma_k)$
end for

4 Results

4.1 DRUNet Results



(a) Noisy (14.78dB) (b) BM3D (25.82dB) (c) DnCNN (26.83dB) (d) RNAN (27.18dB) (e) FFDNet (26.92dB) (f) DRUNet (27.31dB)

Figure 2: Grayscale image denoising results of different methods with noise level 50.

(a) Noisy (14.99dB) (b) BM3D (28.36dB) (c) DnCNN (28.68dB) (d) FFDNet (28.75dB) (e) IRCNN (28.69dB) (f) DRUNet (29.28dB)

Figure 3: Color image denoising results of different methods with noise level 50.

Note that according to Figs. 2, 3, the results of the DRUNet are superior compared with the other well-known denoiser in terms of PSNR. We think the results are really impressing when we visually compare the noise image to the clean image from the DRUNet.

4.2 Different IR tasks Results

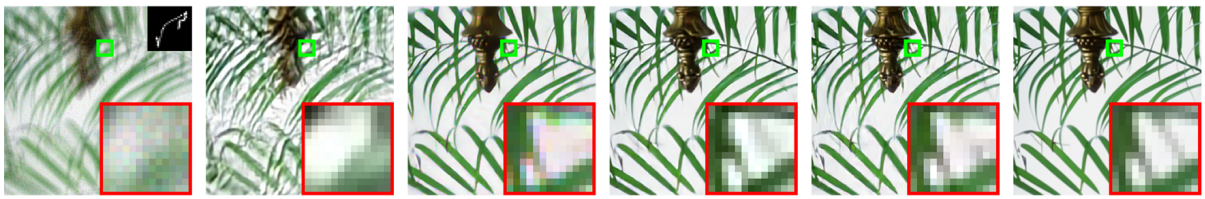
The authors examined the results over three different degradation models which corresponding to three IR tasks for different degradation operations: Blurring, single image super-resolution (SISR), and mosaicing.

4.2.1 Image Deblurring

The degradation model for a blurry image with uniform blur is generally expressed as $y = x \otimes k + n$ where n is a AWGN of standard deviation σ and $x \otimes k$ denotes two-dimensional convolution between kernel k and latent clean image x. If the convolution is carried out with circular boundary conditions, then the solution for the data sub-problem (see proof in 6.2.1) is

$$x_k = \mathcal{F}^{-1}\left(\frac{\overline{\mathcal{F}(k)}\mathcal{F}(y) + \alpha_k \mathcal{F}(z_{k-1})}{\overline{\mathcal{F}(k)}\mathcal{F}(k) + \alpha_k}\right)$$

where the $\mathcal{F}(\cdot)$ and $\mathcal{F}^{-1}(\cdot)$ denote fast Fourier transform (FFT) and inverse FFT.



(a) Blurry image (b) DMPHN (9.74dB) (c) FDN (23.53dB) (d) IRCNN (28.45dB) (e) IRCNN+ (28.14dB) (f) DPIR (30.27dB)

Figure 4: Visual results comparison of different deblurring methods. The blur kernel is visualized in the upper right corner of the blurry image. The noise level is 7.65.

(a) $\mathbf{x}_1$ (16.34dB) (b) $\mathbf{z}_1$ (23.75dB) (c) $\mathbf{x}_4$ (22.33dB) (d) $\mathbf{z}_4$ (29.37dB) (e) $\mathbf{x}_8$ (29.34dB) (f) Convergence curves

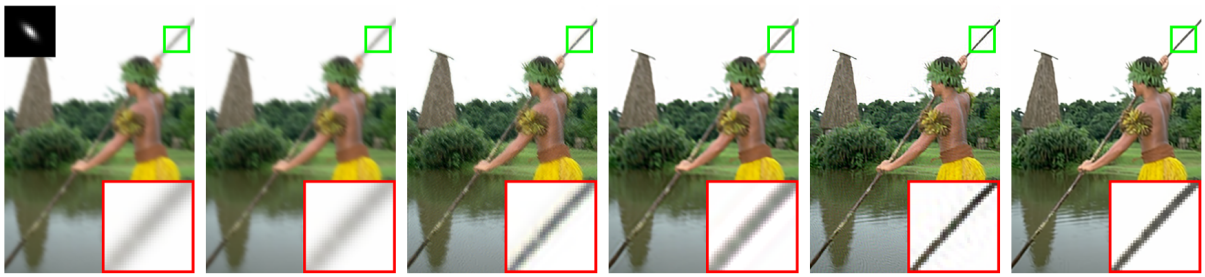
Figure 5: (a)-(e) Visual results and PSNR results of x_k and z_k at different iterations; (f) Convergence curves of PSNR (y-axis) for x_k and z_k with respect to number of iterations (x-axis).

4.2.2 Single Image Super-Resolution

The SISR degradation model is expressed by $y = (x \otimes k)\downarrow_s + n$, $\downarrow_s(\cdot)$ is standard s-fold downsampler, selecting the upper-left pixel for each distinct $s \times s$ patch. According to [12], if the convolution is carried out with circular boundary conditions then a closed form solution for the data sub-problem is

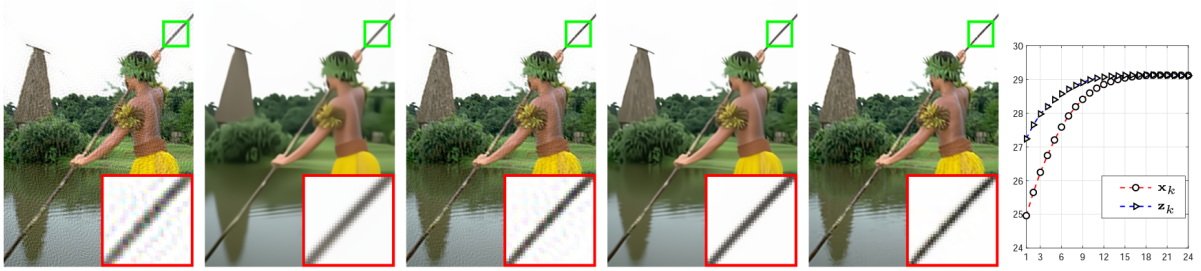
$$x_k = \mathcal{F}^{-1}\left(\frac{1}{\alpha_k}\left(d - \overline{\mathcal{F}(k)} \odot_s \frac{(\mathcal{F}(k)d)\Downarrow_s}{(\overline{\mathcal{F}(k)}\mathcal{F}(k))\Downarrow_s + \alpha_k}\right)\right)$$

where $d = \mathcal{F}(k)\mathcal{F}(y\uparrow_s) + \alpha_k\mathcal{F}(z_{k-1})$, $\odot_s$ denotes distinct block processing operator with element-wise multiplication, i.e., applying element-wise multiplication to the $s \times s$ distinct blocks of $\mathcal{F}(k)$, and $\Downarrow_s$ denotes distinct block downsampler, i.e., averaging the $s \times s$ distinct blocks.



(a) Bicubic (24.82dB) (b) RCAN (24.61dB) (c) MZSR (27.34dB) (d) IRCNN (26.89dB) (e) IRCNN+ (28.65dB) (f) DPIR (29.12dB)

Figure 6: Visual results comparison of different SISR methods on an image corrupted by classical degradation model. The kernel is shown on the upper-left corner. The scale factor is $s = 2$.



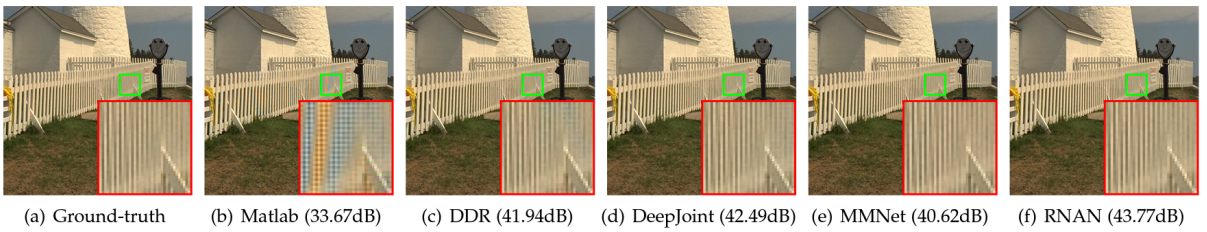
(a) $\mathbf{x}_1$ (24.95dB) (b) $\mathbf{z}_1$ (27.24dB) (c) $\mathbf{x}_6$ (27.59dB) (d) $\mathbf{z}_6$ (28.57dB) (e) $\mathbf{x}_{24}$ (29.12dB) (f) Convergence curves

Figure 7: (a)-(e) Visual results and PSNR results of x_k and z_k at different iteration; (f) Convergence curves of PSNR results (y-axis) for x_k and z_k with respect to number of iterations (x-axis).

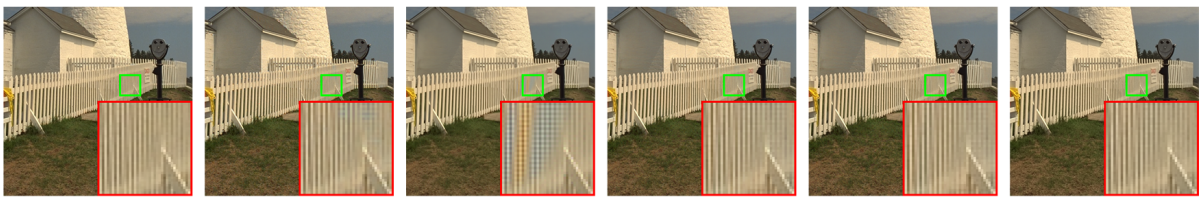
4.2.3 Image Demosaicing

The mosaicked image degradation model is expressed by $y = M \odot x$, where the operation $\odot$ denotes element wise multiplication and M is a 3D matrix with binary elements indicating the missing pixels of y. In this scenario, x is an RGB image. A closed form solution for the data sub-problem (see proof in 6.2.2) is

$$x_k = \frac{M \odot y + \alpha_k z_{k-1}}{M + \alpha_k}.$$



(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)

Figure 8: Visual results comparison of different demosaicing methods.

(a) $\mathbf{x}_1$ (33.67dB) (b) $\mathbf{z}_1$ (29.21dB) (c) $\mathbf{x}_{16}$ (32.69dB) (d) $\mathbf{z}_{16}$ (31.45dB) (e) $\mathbf{x}_{40}$ (43.18dB) (f) Convergence curves

Figure 9: (a)-(e) Visual results and PSNR results of x_k and z_k at different iterations; (f) Convergence curves of PSNR results (y-axis) for x_k and z_k with respect to number of iterations (x-axis).

Note that for the three tasks we discussed, the degradation operation is linear w.r.t x. Therefore, the operation performed over the image can be described as a product between the clean image and a matrix. In this case, we will get that the data sub-problem in the iterative procedure is the least-squares problem, which has a well-known closed-form solution. The least-square solution involves matrix inversion, which its computational complexity is $O(N^3)$ for an $N \times N$ matrix. This can results in a very slow algorithm convergence because we have to solve the data sub-problem at any iteration. Thus, the solutions that have been shown in the described paper are considered as a fast closed-form sloutions, as they involve only multiplications, Fourier transforms, and inverse Fourier transforms whose complexity is $O(N^2 \cdot \log N)$. Additionally, some of the calculations, (for example, $\mathcal{F}(k)\mathcal{F}(k) + \alpha_k$) corresponding to the solution of the data sub-problem can be done offline before we start the iterative procedure to reduce to number of calculations during iterations.

5 Discussion

In Figs. 4, 6, the authors compared the results for the deblurring and the SISR tasks with the SOTA algorithms in those fields. They also compared themselves with their previous work algorithms IRCNN and IRCNN+ [5] which are the same algorithm but with much simpler CNN denoisers. The DPIR demonstrates the best performance in terms of PSNR, compared with the reference algorithms.

According to Fig. 8, the DPIR results are good in terms of PSNR compared with the benchmark algorithms, but the best result, in this case, is the result of the RNAN [13] algorithm, which is an algorithm that involves the attention mechanism.

Figs. 5, 7, 9 illustrate the images through iterations from both perspectives of x_k and z_k. These figures are interesting because we can observe that the algorithm kind of doing what we expected it to do. Note that at the beginning of the iterative process, the x_k is less blurry or mosaicked (depends on the problem), but it is noisier compared with the corresponding z_k because z_k is produced by applying denoiser over x_k. In addition, as the number of iterations increases, the noise observed in x_k becomes smaller which emphasizes the importance of reducing the noise level σ_k as the iteration number increases, as described in Section 3.3. We also noted that after applying the denoiser, the z_k images are highly smoothed such that some fine details in the images disappeared, but then, when applying again the closed-form solution, which corresponds to the data sub-problem, the details reappear. Note also that in Figs. 5(f), 7(f), 9(f), x_k and z_k converge in terms of PSNR as one can expect from the term $\frac{\mu}{2}\|z - x\|^2$ in the cost function (4).

An interesting insight of the authors is that once they had been trained the DRUNet then, the noise it is trying to remove from the latent image has a Gaussian distribution, but this does not mean that the noise of the denoiser input image at any iteration will have a Gaussian distribution. They observed that the distribution of the noise of the input image is task-depending and also depends on the iteration number, as demonstrated in Fig. 10.

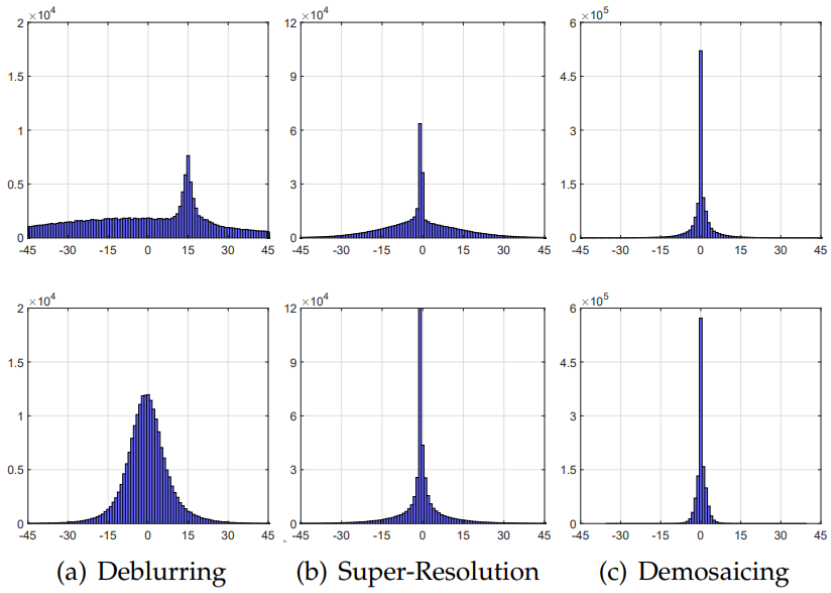


Figure 10: Histogram of the noise (difference) between the ground-truth and input of the denoiser in the first iteration (first row) and last iteration (second row) for (a) deblurring,(b) super-resolution, and (c) demosaicing. The histograms are based on x_1 and x_8 in Fig. 5, x_1 and x_{24} in Fig. 7 and x_1 and x_{40} in Fig. 9.

It can be observed that the distribution of noise for the first iteration varies depending on the task we want to solve, and it does make sense because the input image of the denoiser is the solution to Eq. (5a), which indeed depends on the task we want to solve. Another thing that can be noticed is that the distribution of noise for the case of image deblurring tends to a Gaussian distribution after 8 iterations, unlike the other tasks. This can be explained by that for the image deblurring task, after a small number of iterations, the blurriness caused by blur kernel is gradually alleviated which didn't happen for SISR and Demosaicing.

6 Appendices

6.1 The Prior Sub-problem as Denoiser

Notice that when we develop the MAP estimator for the denoising problem $y = x + n$, then if $n \sim \mathcal{N}(0,\, \sigma^2 I)$ where I is identity matrix of size $N \times N$ then,

$$\begin{aligned}
\hat{x} &= \arg\max_x \{\log p(y|x) + \log p(x)\} \\
&= \arg\max_x \left\{ \log \frac{1}{\sqrt{(2\pi)^N \sigma^N}} e^{-\frac{\|y-x\|^2}{2\sigma^2}} + \log p(x) \right\} \\
&= \arg\max_x \left\{ -\frac{\|y-x\|^2}{2\sigma^2} + \log p(x) \right\} \\
&= \arg\min_x \left\{ \frac{1}{2\sqrt{(\sigma^2)^2}} \|y-x\|^2 + \Re(x) \right\}
\end{aligned} \tag{7}$$

For $\sigma = \sqrt{\lambda/\mu}$, and $x = x_k$ we get the term for calculating z_k in (5b). ■

6.2 Mathematical Derivation of the Fast Closed-Form Solution for the Data Sub-problem

Let's assume that $x \in \mathbb{R}^N$, and $y \in \mathbb{R}^N$ are flattened representation of the latent image and the observed image, respectively and that $\tau : \mathbb{R}^N \to \mathbb{R}^N$. Let's also define $\tilde{x}(i) = x(-i)$, where we assume that $\tilde{x}(i) = \tilde{x}(i \pmod N)$ and that the convolution is carried out with circular boundary conditions.

6.2.1 Image Deblurring Problem

The first iterative equation we want to solve is:

$$x_k = \arg\min_x \|y - \tau(x)\|^2 + \mu\sigma^2 \|x - z_{k-1}\|^2$$

$\tau(\cdot)$ denote the degradation operation. For image deblurring, $\tau_x^d = x \otimes k$ which corresponding to two-dimensional convolution operation. Let's define the objective function which we want to minimize, $f^d(x)$,

$$f^d(x) = \|y - \tau_x^d\|^2 + \mu\sigma^2 \|x - z_{k-1}\|^2$$

We would like to find the gradient of the function $f^d(x)$ w.r.t x and compare it to zero

$$\nabla_x f^d(x) = \nabla_x ((\tau_x^d)^T (\tau_x^d) - 2y^T \tau_x^d) + \mu\sigma^2 (x^T x - 2z_{k-1}^T x) \tag{8}$$

In order to calculate the gradient of $f^d(x)$ w.r.t x we need to calculate the following three gradients and sum them up:
1. $\nabla_x \left\{ \tau_x^d(x)^T \tau_x^d(x) \right\}$
2. $\nabla_x \left\{ -2y^T \tau_x^d \right\}$
3. $\nabla_x \left\{ \mu\sigma^2 (x^T x - 2z_{k-1}^T x) \right\}$

Let's start with $\nabla_x \left\{ \tau_x^d(x)^T \tau_x^d(x) \right\}$ by first writing $\tau_x^d(x)^T \tau_x^d(x)$ explicitly:

$$\begin{aligned}\tau_x^d(x)^T\tau_x^d(x) &= \sum_{i=0}^{N-1}\Big[\Big(\sum_{l=1}^{N-1}x(l)k(i-l)\Big)^2\Big]\\ &= \sum_{i=0}^{N-1}\Big[\sum_{l=0}^{N-1}\sum_{l'=0}^{N-1}x(l)x(l')k(i-l)k(i-l')\Big]\\ &= \sum_{i=0}^{N-1}\sum_{l=0}^{N-1}\sum_{l'=0}^{N-1}x(l)x(l')k(i-l)k(i-l')\end{aligned} \tag{9}$$

Note that

$$\frac{\partial x(l)x(l')}{\partial x(m)} = \delta[l-m]\delta[l'-m]2x(m) + \delta[l-m](1-\delta[l'-m])x(l') + \delta[l'-m](1-\delta[l-m])x(l) \tag{10}$$

Thereby, by taking the derivative of the term in (9) and combining it with 10 we have,

$$\begin{aligned}\frac{\partial\big((\tau_x^d)^T(\tau_x^d)\big)}{\partial x(m)} &= \sum_{i=0}^{N-1}\sum_{l=0}^{N-1}\sum_{l'=0}^{N-1}k(i-l)k(i-l')\frac{\partial x(l)x(l')}{\partial x(m)} = 2\sum_{i=0}^{N-1}k(i-m)^2x(m)\\ &+\sum_{i=0}^{N-1}\sum_{l'=0}^{N-1}k(i-m)k(i-l')x(l') - \sum_{i=0}^{N-1}k(i-m)^2x(m)\\ &+\sum_{i=0}^{N-1}\sum_{l=0}^{N-1}k(i-l)k(i-m)x(l) + \sum_{i=0}^{N-1}k(i-m)^2x(m)\\ &= 2\sum_{i=0}^{N-1}\sum_{l=0}^{N-1}k(i-l)k(i-m)x(l) = 2\sum_{i=0}^{N-1}k(i-m)\Big[\sum_{l=0}^{N-1}x(l)k(i-l)\Big]\\ &= 2\sum_{i=0}^{N-1}k(i-m)\Big[(x\otimes k)[i]\Big] = 2(x\otimes k\otimes\tilde{k})[m]\end{aligned}$$

$$\implies \boxed{\nabla_x\big((\tau_x^d)^T(\tau_x^d)\big) = 2(x\otimes k\otimes\tilde{k})} \tag{11}$$

Let's now calculate $\nabla_x(-2y^T\tau_x^d)$. Note that:

$$\begin{aligned}-2y^T\tau_x^d &= -2\sum_{i=0}^{N-1}y(i)\tau_x^d(i) = -2\sum_{i=0}^{N-1}y(i)\Big[\sum_{l=0}^{N-1}x(l)k(i-l)\Big]\\ &= -2\sum_{i=0}^{N-1}y(i)\tau_x^d(i) = -2\sum_{i=0}^{N-1}\sum_{l=0}^{N-1}k(i-l)y(i)x(l)\end{aligned} \tag{12}$$

Hence,

$$\begin{aligned}\frac{\partial(-2y^T\tau_x^d)}{\partial x(m)} &= -2\sum_{i=0}^{N-1}\sum_{l=0}^{N-1}k(i-l)y(i)\frac{\partial x(l)}{\partial x(m)} = -2\sum_{i=0}^{N-1}\sum_{l=0}^{N-1}k(i-l)y(i)\delta[l-m]\\ &= -2\sum_{i=0}^{N-1}k(i-m)y(i) = -2(y\otimes\tilde{k})[m]\end{aligned} \tag{13}$$

$$\implies \boxed{\nabla_x(-2y^T\tau_x^d) = -2(y \otimes \tilde{k})} \tag{14}$$

The derivation of the third term, $\nabla_x \left\{\mu\sigma^2(x^Tx - 2z_{k-1}^Tx)\right\}$, is

$$\nabla_x \left\{\mu\sigma^2(x^Tx - 2z_{k-1}^Tx)\right\} = \mu\sigma^2(2x - 2z_{k-1}) \tag{15}$$

Plugging (11), (14), and (15) into 8 and compare it to zero, we get:

$$\nabla_x f^d(x) = 2(x \otimes k \otimes \tilde{k}) - 2(y \otimes \tilde{k}) + \mu\sigma^2(2x - 2z_{k-1}) = 0 \tag{16}$$

Let's apply Fourier transform, $\mathcal{F}$, on both sizes of Eq. (16),

$$\mathcal{F}(x)\mathcal{F}(k)\overline{\mathcal{F}(k)} - \mathcal{F}(y)\overline{\mathcal{F}(k)} + \mu\sigma^2(\mathcal{F}(x) - \mathcal{F}(z_{k-1})) = 0 \tag{17}$$

By rearranging (17):

$$\implies \boxed{x_k = \mathcal{F}^{-1}\left(\frac{\overline{\mathcal{F}(k)}\mathcal{F}(y) + \mu\sigma^2\mathcal{F}(z_{k-1})}{\overline{\mathcal{F}(k)}\mathcal{F}(k) + \mu\sigma^2}\right)} \blacksquare$$

6.2.2 Color Image Demosaicing

In this section we derive the fast closed form solution for the image demosaicing data sub-problem

$$x_k = \arg\min_x \|y - \tau_{demos}(x)\|^2 + \mu\sigma^2\|x - z_{k-1}\|^2$$

First, we define our objective function that we want to minimize:

$$\begin{aligned} f^{demos}(x) =& \|y - \tau_{demos}(x)\|^2 + \mu\sigma^2\|x - z_{k-1}\|^2 \\ =& y^Ty - 2y^T\tau_{demos}(x) + (\tau_{demos}(x))^T\tau_{demos}(x) + \mu\sigma^2(x^Tx - 2x^Tz_{k-1} + z_{k-1}^Tz_{k-1}) \end{aligned} \tag{18}$$

Let's apply the gradient operation:

$$\nabla_x f^{desmos}(x) = \nabla_x\Big((\tau_{demos}(x))^T\tau_{demos}(x) - 2y^T\tau_{demos}(x)\Big) + \mu\sigma^2(2x - 2z_{k-1}) \tag{19}$$

Note that:

$$\tau_{demos}(x)^T\tau_{demos}(x) = \sum_l x^2(l)M^2(l) \tag{20}$$

By taking its derivative we get,

$$\frac{\partial\tau_{demos}(x)^T\tau_{demos}(x)}{\partial x(m)} = 2x(m)M^2(m) \tag{21}$$

Because M is a binary matrix so if we square its values they will remain the same thereby,

$$\implies \boxed{\nabla_x(\tau_{demos}(x)^T\tau_{demos}(x)) = 2(x \odot M)} \tag{22}$$

Additionally, $\frac{\partial(-2y^T\tau_{demos}(x))}{\partial x(m)} = -2y(m)M(m)$,

$$\implies \boxed{\nabla_x(-2y^T\tau_{demos}(x)) = -2(y \odot M)} \tag{23}$$

Substituting the terms (22), and (23), with the terms in (19), we get, $\nabla_x f^{demos}(x) = 2(x \odot M) - 2(y \odot M) + \mu\sigma^2(2x - 2z_{k-1})$

If we require that $\nabla_x f(x) = 0$ we get:

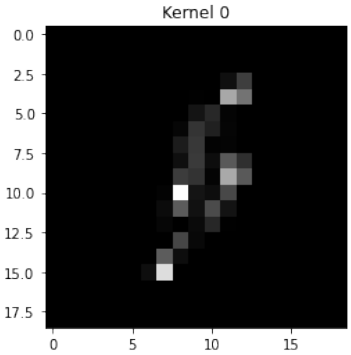
$$2(x_k \odot M) - 2(y \odot M) + \mu\sigma^2(2x_k - 2z_{k-1}) = 0 \tag{24}$$

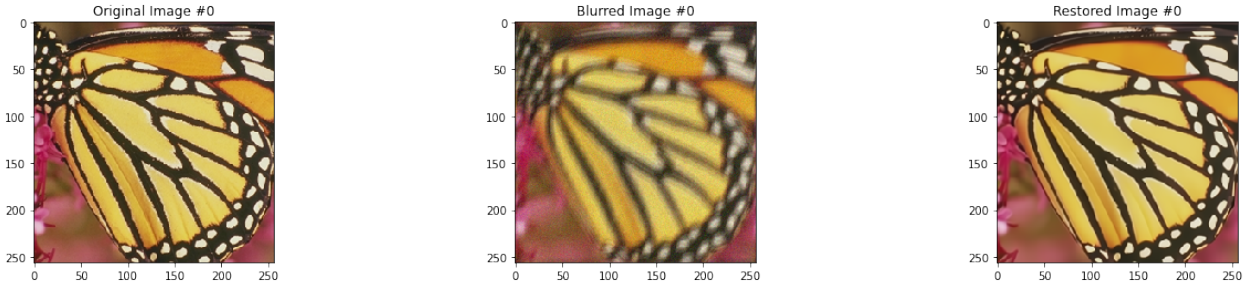
If we consider the division operation as an elementwise operation then, by rearrange Eq. 24:

$$\boxed{x_k = \frac{M \odot y + \mu\sigma^2 z_{k-1}}{M + \mu\sigma^2}}$$

6.3 Results from our Colab Code

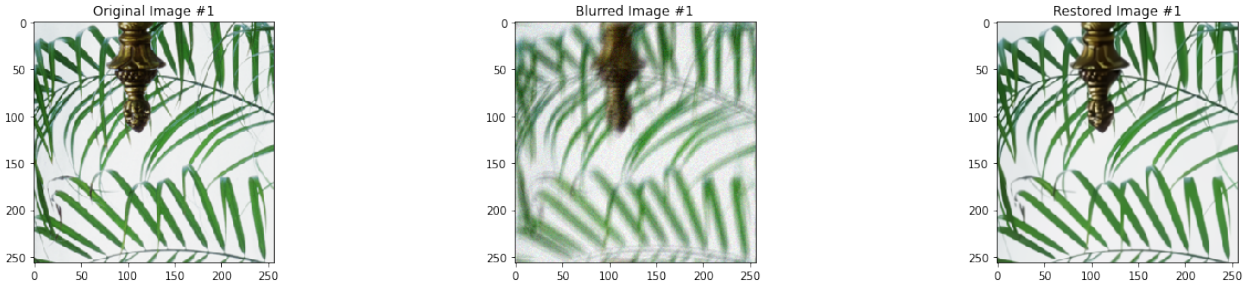
Image Deblurring





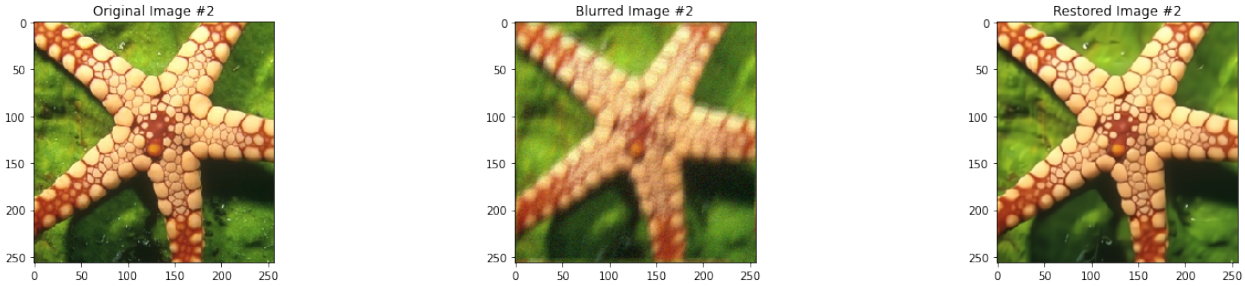
(a) Original Image (b) Blurred Image (c) Restored Image

Figure 11: Image Deblurring Example 1 with Kernel 0



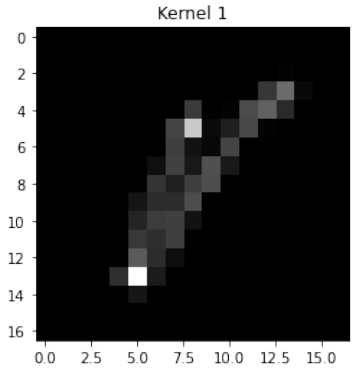
(a) Original Image (b) Blurred Image (c) Restored Image

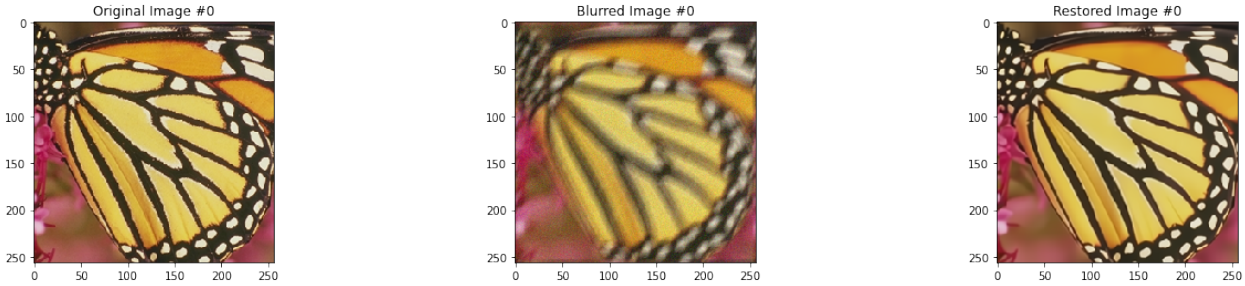
Figure 12: Image Deblurring Example 2 with Kernel 0



(a) Original Image (b) Blurred Image (c) Restored Image

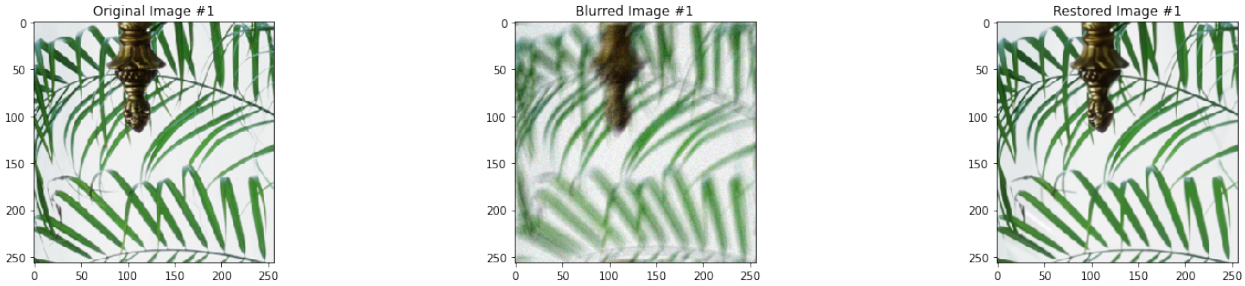
Figure 13: Image Deblurring Example 3 with Kernel 0





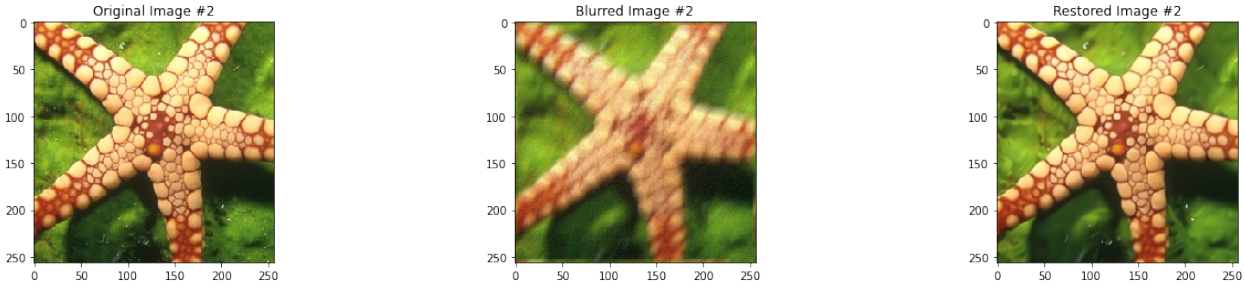
(a) Original Image (b) Blurred Image (c) Restored Image

Figure 14: Image Deblurring Example 1 with Kernel 1



(a) Original Image (b) Blurred Image (c) Restored Image

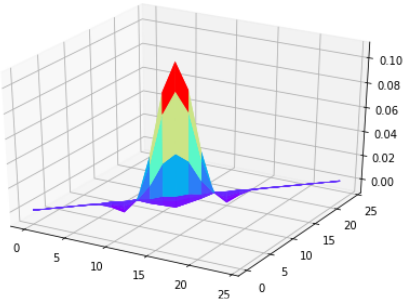
Figure 15: Image Deblurring Example 2 with Kernel 1

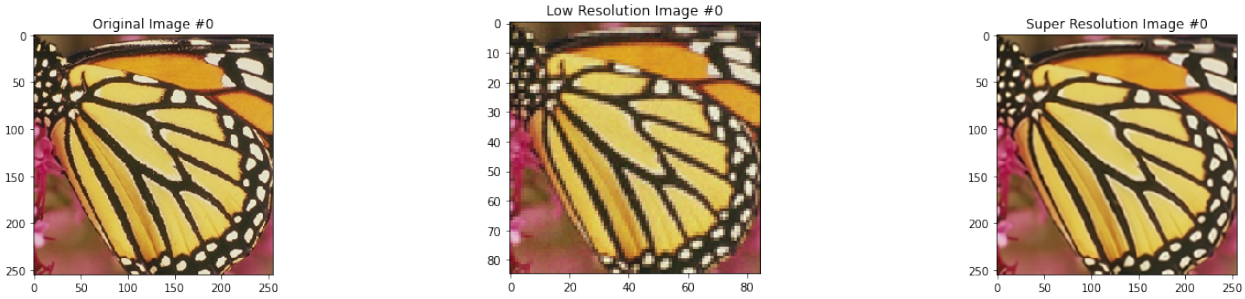


(a) Original Image (b) Blurred Image (c) Restored Image

Figure 16: Image Deblurring Example 3 with Kernel 1

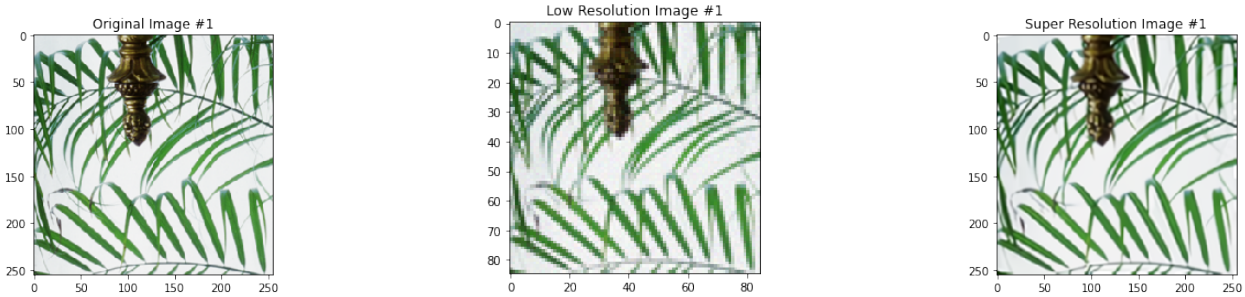
Figure 17: Blurry Kernel





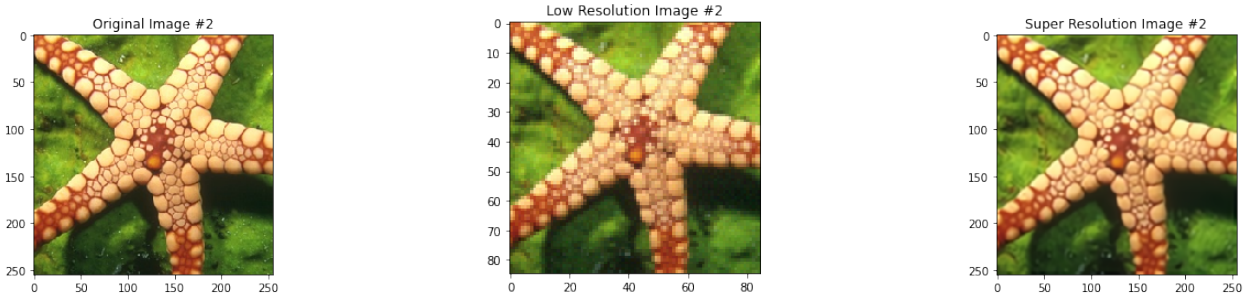
(a) Original Image (b) Blurred Image (c) Restored Image

Figure 18: SISR Example 1



(a) Original Image (b) Blurred Image (c) Restored Image

Figure 19: SISR Example 2

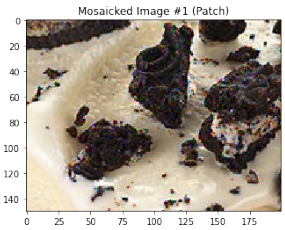


(a) Original Image (b) Blurred Image (c) Restored Image

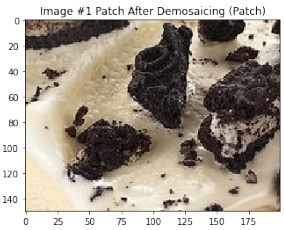
Figure 20: SISR Example 3

Color Image Demosaicing

(a) Original Image Patch



(b) Mosaicked Image



(c) Patch After Demosaicing

Figure 21: Image Demosaicing Example

7 References

[1] Nir Shlezinger, Jay Whang, Yonina C Eldar, and Alexandros G Dimakis. Model-based deep learning. *arXiv preprint arXiv:2012.08405*, 2020.

[2] Michael Elad and Michal Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image Processing*, 15(12):3736–3745, 2006.

[3] Antoni Buades, Bartomeu Coll, and J-M Morel. A non-local algorithm for image denoising. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 2, pages 60–65. Ieee, 2005.

[4] 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.

[5] Kai Zhang, Wangmeng Zuo, Shuhang Gu, and Lei Zhang. Learning deep cnn denoiser prior for image restoration. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3929–3938, 2017.

[6] Donald Geman and Chengda Yang. Nonlinear image recovery with half-quadratic regularization. *IEEE transactions on Image Processing*, 4(7):932–946, 1995.

[7] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, Jonathan Eckstein, et al. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning*, 3(1):1–122, 2011.

[8] 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.

[9] 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.

[10] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[11] Radu Timofte, Rasmus Rothe, and Luc Van Gool. Seven ways to improve example-based single image super resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1865–1873, 2016.

[12] Ningning Zhao, Qi Wei, Adrian Basarab, Nicolas Dobigeon, Denis Kouamé, and Jean-Yves Tourneret. Fast single image super-resolution using a new analytical solution for ℓ_2 - ℓ_2 problems. *IEEE Transactions on Image Processing*, 25(8):3683–3697, 2016.

[13] Yulun Zhang, Kunpeng Li, Kai Li, Bineng Zhong, and Yun Fu. Residual non-local attention networks for image restoration. *arXiv preprint arXiv:1903.10082*, 2019.