Image Restoration Using Joint Patch-Group-Based Sparse Representation

Zhiyuan Zha, *Member, IEEE*, Xin Yuan, *Senior Member, IEEE*, Bihan Wen, *Member, IEEE*, Jiachao Zhang, *Member, IEEE*, Jiantao Zhou, *Senior Member, IEEE*, and Ce Zhu, *Fellow, IEEE*

***Abstract*— Sparse representation has achieved great success in various image processing and computer vision tasks. For image processing, typical patch-based sparse representation (PSR) models usually tend to generate undesirable visual artifacts, while group-based sparse representation (GSR) models lean to produce over-smooth effects. In this paper, we propose a new sparse representation model, termed joint patch-group based sparse representation (JPG-SR). Compared with existing sparse representation models, the proposed JPG-SR provides an effective mechanism to integrate the local sparsity and nonlocal self-similarity of images. We then apply the proposed JPG-SR to image restoration tasks, including image inpainting and image deblocking. An iterative algorithm based on the alternating direction method of multipliers (ADMM) framework is developed to solve the proposed JPG-SR based image restoration problems. Experimental results demonstrate that the proposed JPG-SR is effective and outperforms many state-of-the-art methods in both objective and perceptual quality.**

***Index Terms*— Sparse representation, JPG-SR, nonlocal self-similarity, image restoration, ADMM.**

I. Introduction

As A popular technique in image processing, sparse representation has attracted significant interests of researchers [1]–[14]. It is usually classified into two categories: analysis sparse representation model [3], [7], [8] and synthesis sparse representation model [1], [2], [5], [6]. Analysis sparse representation model represents a signal by multiplying it with an analysis over-complete dictionary, leading to a sparse effect [15]. In this work, we focus on synthesis sparse representation model. Generally speaking, methods of synthesis sparse representation in image processing can be further classified into two categories: patch-based sparse representation (PSR) [1], [2] and group-based sparse representation (GSR) [5], [6], [11], [16]. PSR assumes that each patch of an image can be perfectly modeled by a sparse linear combination of learnable basis elements. These elements, called atoms, compose a dictionary [2], [17]. The dictionary is usually learned from an image or an image dataset. Compared with traditionally analytic dictionaries, such as discrete cosine transform (DCT), wavelet and curvelet, dictionaries learned directly from images are superior to adapt to image local structures, and thus could improve the sparsity which results in better performance. For instance, the seminal work of KSVD dictionary learning method [2] has not only achieved promising denoising result, but also been extended to various image processing and computer vision tasks [18], [19]. However, it has been shown that the PSR with an over-complete dictionary usually tends to generate undesirable visual artifacts in image restoration [20], [21]. Moreover, the PSR model usually ignores the correlation among similar patches [6], [10], [11] and thus leads to degraded results in general.

Inspired by the success of the nonlocal self-similarity (NSS) prior in images [22], [23], instead of using a single patch as the basic unit in sparse representation, recent advances in GSR consider similar *patch group* as the basic unit, while similar to PSR, it can be sparsely represented by a set of sparse codes in the group domain, *i.e.*, each patch group can also be precisely represented by a sparse linear combination of basis elements of the dictionary [5], [6]. The GSR models have demonstrated great potential in various image restoration tasks [5], [6], [10], [23], [24]. For example, Dabov *et al*. [23] proposed the BM3D method to combine NSS prior with transform domain filtering, which is still one of the state-of-the-art denoising methods. Mairal *et al*. [5] proposed the learned simultaneous sparse coding (LSSC) to improve the restoration performance of KSVD [2] via GSR. LPG-PCA [24] utilized the nonlocal similar patches as data samples to estimate statistical parameters based on PCA training. Zhang *et al*. [6] proposed a group-based sparse representation model for image restoration. Dong *et al*. [10] developed the structured sparse coding with gaussian scale mixture prior for image restoration. Though GSR models have shown great

Manuscript received April 20, 2019; revised February 24, 2020 and April 8, 2020; accepted April 18, 2020. Date of publication July 3, 2020; date of current version July 16, 2020. This work was supported in part by the Joint Funds of National Natural Science Foundation of China under Grant U19A2052, in part by the Key Project of Sichuan Provincial Department of Science and Technology under Grant 2018JY0035, in part by the Ministry of Education, Republic of Singapore, under the Start-Up Grant, in part by the Macau Science and Technology Development Fund, Macau, under Grant SKL-IOTSC-2018-2020, Grant 077/2018/A2, and Grant 022/2017/A1, and in part by the China CPSF Fund under Grant 2019M651698. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Chun-Shien Lu. *(Zhiyuan Zha and Xin Yuan contributed equally to this work.) (Corresponding author: Ce Zhu.)*

Zhiyuan Zha and Ce Zhu are with the School of Information and Communication Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China (e-mail: zhazhiyuan.mmd@gmail.com; eczhu@uestc.edu.cn).

Xin Yuan is with Nokia Bell Labs, Murray Hill, NJ 07974 USA (e-mail: xyuan@bell-labs.com).

Bihan Wen is with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798 (e-mail: bihan.wen@ntu.edu.sg).

Jiachao Zhang is with the Artificial Intelligence Institute of Industrial Technology, Nanjing Institute of Technology, Nanjing 211167, China (e-mail: zhangjc07@foxmail.com).

Jiantao Zhou is with the State Key Laboratory of Internet of Things for Smart City, University of Macau, Taipa, Macau, and also with the Department of Computer and Information Science, University of Macau, Taipa, Macau (e-mail: jtzhou@umac.mo).

Digital Object Identifier 10.1109/TIP.2020.3005515

success in various image restoration tasks, the processed images apt to suffer the over-smooth effects [25].

Bearing the above concerns in mind, we aim to address the following questions in this paper.

1) Is it possible to mitigate the drawbacks of the PSR and GSR models?
2) Is it possible to build a joint model to integrate the PSR and GSR models? If yes, can we solve the model effectively and efficiently?

We answer these questions by developing a new sparse representation model, dubbed joint patch-group based sparse representation (JPG-SR). Compared with previous sparse representation models, the proposed JPG-SR is capable of integrating the local sparsity with NSS of the image. To make the optimization tractable, we develop an iterative algorithm based on the alternating direction method of multipliers (ADMM) framework to solve the proposed optimization problem. Through applying the proposed JPG-SR model to image restoration tasks, including image inpainting and image deblocking, we demonstrate that the JPG-SR not only retains the advantages of the PSR and GSR models, but also alleviates their drawbacks, respectively. Extensive experimental results demonstrate that the proposed JPG-SR outperforms several state-of-the-art image restoration methods both quantitatively and qualitatively.

The remainder of this paper is organized as follows. Section II introduces the related works about sparse representation. Section III introduces our new sparse representation model, *i.e.*, the JPG-SR model. Section IV develops an optimization method to solve the proposed JPG-SR model for image restoration tasks. Section V presents the experimental results. Finally, several concluding remarks are given in Section VI. The preliminary work has appeared in [4].[1]

II. Background

A. Patch-Based Sparse Representation

Following the notations in [1], the basic unit of sparse representation for images is patch. Mathematically, given an (vectorized) image $\boldsymbol{x} \in \mathbb{R}^N$, let $\boldsymbol{x}_i = \boldsymbol{R}_i\boldsymbol{x}, \forall i = 1, 2, \ldots n$, denote an (vectorized) image patch of size $\sqrt{b} \times \sqrt{b}$ extracted from location i. Given a dictionary $\mathbf{D} \in \mathbb{R}^{b\times M}$, $b \leq M$, the sparse representation of each patch $\boldsymbol{x}_i$ is to find a sparse vector where most coefficients are zero. Specifically, each patch $\boldsymbol{x}_i$ can be sparsely represented as $\boldsymbol{x}_i \approx \mathbf{D}\boldsymbol{\alpha}_i^*$ by solving the following ℓ_0-norm minimization problem,

$$\hat{\boldsymbol{\alpha}}_i^* = \arg\min_{\boldsymbol{\alpha}_i^*} \left(\frac{1}{2} \left\| \boldsymbol{x}_i - \mathbf{D}\boldsymbol{\alpha}_i^* \right\|_2^2 + \lambda \left\| \boldsymbol{\alpha}_i^* \right\|_0 \right) \quad \forall i \ , \tag{1}$$

where $\| \ \|_2$ denotes ℓ_2-norm and λ is a regularization parameter. $\| \ \|_0$ signifies the ℓ_0-norm (quasi-norm), *i.e.*, counting the nonzero entries in $\boldsymbol{\alpha}_i^*$. In this manner, the entire image $\boldsymbol{x}$ can be sparsely represented by a set of sparse codes $\{\boldsymbol{\alpha}_i^*\}_{i=1}^n$. Concatenating n patches, let $\mathbf{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n] \in \mathbb{R}^{b\times n}$ denote all the patches extracted from the image. Since $\mathbf{D}$ is shared across these patches, we have

$$\hat{\boldsymbol{\alpha}}^* = \arg\min_{\boldsymbol{\alpha}^*} \left(\frac{1}{2} \left\| \mathbf{X} - \mathbf{D}\boldsymbol{\alpha}^* \right\|_F^2 + \lambda \left\| \boldsymbol{\alpha}^* \right\|_0 \right), \tag{2}$$

where $\| \ \|_F$ denotes the Frobenius norm, $\boldsymbol{\alpha}^* = [\boldsymbol{\alpha}_1^*, \ldots, \boldsymbol{\alpha}_n^*] \in \mathbb{R}^{M\times n}$ is the sparse coefficient matrix, and the ℓ_0-norm is imposed on each column of $\boldsymbol{\alpha}^*$ (corresponding to each patch).

B. Group-Based Sparse Representation

Instead of using a single patch as the basic unit in sparse representation, recent studies have shown that GSR using patch groups can produce more promising results for various image processing tasks than typical PSR models [5], [6], [10], [11]. Hereby, we briefly introduce the GSR model.

To be concrete, image $\boldsymbol{x}$ is firstly divided into n overlapped patches $\boldsymbol{x}_i$ of size $\sqrt{b} \times \sqrt{b}, i = 1, 2, \ldots, n$. Then, different from PSR, for each exemplar patch $\boldsymbol{x}_i$, the most similar m patches (*e.g.*, by the K-Nearest Neighbour (KNN) method [26]) are selected from a $W \times W$ sized searching window to form a set $\mathcal{S}_{G_i}$. Following this, all patches in $\mathcal{S}_{G_i}$ are stacked into a matrix $\mathbf{X}_{G_i} \in \mathbb{R}^{b\times m}$, which contains every patch in $\mathcal{S}_{G_i}$ as its column, *i.e.*, $\mathbf{X}_{G_i} = \{\boldsymbol{x}_{i,1}, \boldsymbol{x}_{i,2}, \ldots, \boldsymbol{x}_{i,m}\}$. This matrix $\mathbf{X}_{G_i}$ consisting of all patches with similar structures is thus called *a patch group*, where $\boldsymbol{x}_{i,j}$ denote the j-th patch (column) in the i-th patch group. Finally, similar to PSR, given a dictionary $\mathbf{D}_{G_i} \in \mathbb{R}^{b\times K}$, each patch group $\mathbf{X}_{G_i}$ can be sparsely represented by solving

$$\hat{\boldsymbol{\beta}}_{G_i}^* = \arg\min_{\boldsymbol{\beta}_{G_i}^*} \left(\frac{1}{2} \| \mathbf{X}_{G_i} - \mathbf{D}_{G_i}\boldsymbol{\beta}_{G_i}^* \|_F^2 + \lambda \| \boldsymbol{\beta}_{G_i}^* \|_0 \right) \ \forall i \ , \tag{3}$$

where $\boldsymbol{\beta}_{G_i}^*$ represents the group sparse coefficient of each patch group $\mathbf{X}_{G_i}$, and $\| \ \|_0$ signifies the ℓ_0-norm, *i.e.*, counting the nonzero entries of each column in $\boldsymbol{\beta}_{G_i}^*$. In order to put all patch groups in one shot, let $\mathbf{Q}_i \in \mathbb{R}^{n\times m}$ denote the searching and extracting operations of the similar patches for the i-th patch, *i.e.*, $\mathbf{X}_{G_i} = \mathbf{X}\mathbf{Q}_i$. Concatenating n patch groups, we have

$$\mathbf{X}_G = \mathbf{X}[\mathbf{Q}_1, \ldots, \mathbf{Q}_n] = \mathbf{X}\mathbf{Q} \in \mathbb{R}^{b\times(mn)}. \tag{4}$$

Due to the fact that each patch group has its own dictionary and the dictionaries are not necessarily shared, let

$$\mathbf{D}_G = [\mathbf{D}_{G_1}, \ldots, \mathbf{D}_{G_n}] \in \mathbb{R}^{b\times(nK)}, \tag{5}$$

$$\bar{\boldsymbol{\beta}}_G^* = [\bar{\boldsymbol{\beta}}_{G_1}^*, \ldots, \bar{\boldsymbol{\beta}}_{G_n}^*] \in \mathbb{R}^{(nK)\times(mn)}, \tag{6}$$

where $\{\bar{\boldsymbol{\beta}}_{G_i}^*\}_{i=1}^n \in \mathbb{R}^{nK\times m}$ is an expanded (longer with more rows) version of $\boldsymbol{\beta}_{G_i}^* \in \mathbb{R}^{K\times m}$, with $\boldsymbol{\beta}_{G_i}^*$ in the corresponding locations (from $((i-1)K+1)$-th row to (iK)-th row) but zeros elsewhere, *i.e.*, corresponding to $\mathbf{D}_{G_i}$ in $\mathbf{D}_G$. The problem to be solved now becomes

$$\hat{\bar{\boldsymbol{\beta}}}_G^* = \arg\min_{\bar{\boldsymbol{\beta}}_G^*} \left(\frac{1}{2} \| \mathbf{X}_G - \mathbf{D}_G\bar{\boldsymbol{\beta}}_G^* \|_F^2 + \lambda \| \bar{\boldsymbol{\beta}}_G^* \|_0 \right), \tag{7}$$

where the ℓ_0-norm is again imposed on each column and this holds true for the following derivations in this paper. It is worth

[1] Significant changes have been made compared to our previous work in [4]. Specifically, different from our previous work, the model in [4] has only solved by an average of PSR and GSR models, the proposed JPG-SR model in this paper is solved by an integral way. We have added the quantization noise model, quantization constraint prior and the adaptive parameter setting for image deblocking in Sec. IV-C. Moreover, extensive experiments have been added to demonstrate the superiority, robustness and convergence of the proposed JPG-SR model in Sec. V.

noting that both $\mathbf{X}$ in PSR and $\mathbf{X}_G$ in GSR are constructed from the same original image $\boldsymbol{x}$.

III. Joint Patch-Group Based Sparse Representation Model

As mentioned before, the PSR model usually generates some undesirable visual artifacts, while GSR may lead to over-smooth effect in various image processing tasks. To cope with these problems, instead of using Eq. (2) or Eq. (7) individually, we propose a joint patch-group based sparse representation (JPG-SR) model in this section.

A. A Joint Model by Minimizing the Lower Bound

We first introduce some preliminary transformations to link the PSR model in Eq. (2) with the GSR model in Eq. (7). Recall that each patch (column) in the patch group $\mathbf{X}_G$ is from $\mathbf{X}$ and it can be sparsely represented by Eq. (2). Therefore, in addition to the sparse representation in Eq. (7), we can also have

$$\mathbf{X}_G = \mathbf{D}\boldsymbol{\alpha}_G^*, \tag{8}$$

where $\boldsymbol{\alpha}_G^* \in \mathbb{R}^{M\times(mn)}$ is composed of the corresponding columns in $\boldsymbol{\alpha}^*$; in other words, $\boldsymbol{\alpha}_G^*$ is an expanded version of $\boldsymbol{\alpha}^*$ in Eq. (2), where each column is reproduced by m times according to the patch searching in $\mathbf{X}_G$. In this case, similar to Eq. (1), $\boldsymbol{\alpha}_G^*$ is solved by

$$\hat{\boldsymbol{\alpha}}_G^* = \arg\min_{\boldsymbol{\alpha}_G^*} \left(\frac{1}{2} \left\| \mathbf{X}_G - \mathbf{D}\boldsymbol{\alpha}_G^* \right\|_F^2 + \lambda \left\| \boldsymbol{\alpha}_G^* \right\|_0 \right). \tag{9}$$

Comparing Eq. (7) with Eq. (9), we can see that both $\mathbf{D}\boldsymbol{\alpha}_G^*$ and $\mathbf{D}_G\bar{\boldsymbol{\beta}}_G^*$ are approximations of $\mathbf{X}_G$. We can write the model jointly as

$$\{\hat{\boldsymbol{\alpha}}_G^*, \hat{\bar{\boldsymbol{\beta}}}_G^*\} = \arg\min_{\boldsymbol{\alpha}_G^*, \bar{\boldsymbol{\beta}}_G^*} \left(\frac{1}{2} \left\| \mathbf{X}_G - \mathbf{D}\boldsymbol{\alpha}_G^* \right\|_F^2 + \lambda \left\| \boldsymbol{\alpha}_G^* \right\|_0 + \frac{1}{2}\|\mathbf{X}_G - \mathbf{D}_G\bar{\boldsymbol{\beta}}_G^*\|_F^2 + \rho\|\bar{\boldsymbol{\beta}}_G^*\|_0 \right), \tag{10}$$

where ρ is now playing the same role of λ in Eq. (7).

Though the model in Eq. (10) can be solved by a simple average of PSR and GSR models [4], $\boldsymbol{\alpha}_G^*$ and $\bar{\boldsymbol{\beta}}_G^*$ are decoupled. In other words, this model may not faithfully integrate the local sparsity and nonlocal self-similarity of images. Therefore, we hereby employ the triangular inequality $(a+b)^2 \geq a^2 + b^2$ given $a \geq 0, b \geq 0$ and thus $\min_{a,b}(a+b)^2 = \min_{a,b}(a^2+b^2)$. Instead of minimizing the right-hand side of Eq. (10), we minimize

$$\{\hat{\boldsymbol{\alpha}}_G^*, \hat{\bar{\boldsymbol{\beta}}}_G^*\} = \arg\min_{\boldsymbol{\alpha}_G^*, \bar{\boldsymbol{\beta}}_G^*} \left(\frac{1}{2} \left\| \mathbf{X}_G - \mathbf{D}\boldsymbol{\alpha}_G^* + \mathbf{X}_G - \mathbf{D}_G\bar{\boldsymbol{\beta}}_G^* \right\|_F^2 + \lambda \left\| \boldsymbol{\alpha}_G^* \right\|_0 + \rho\|\bar{\boldsymbol{\beta}}_G^*\|_0 \right). \tag{11}$$

$$= \arg\min_{\boldsymbol{\alpha}_G^*, \bar{\boldsymbol{\beta}}_G^*} \left(2 \left\| \mathbf{X}_G - \mathbf{D}\frac{\boldsymbol{\alpha}_G^*}{2} - \mathbf{D}_G\frac{\bar{\boldsymbol{\beta}}_G^*}{2} \right\|_F^2 + \lambda \left\| \boldsymbol{\alpha}_G^* \right\|_0 + \rho\|\bar{\boldsymbol{\beta}}_G^*\|_0 \right). \tag{12}$$

The minimization in Eq. (12) can be seen as minimizing the upper bound of Eq. (10). Since a scalar does not change the minimization, we propose the joint model below via defining $\boldsymbol{\alpha}_G = \frac{\boldsymbol{\alpha}_G^*}{2}$ and $\bar{\boldsymbol{\beta}}_G = \frac{\bar{\boldsymbol{\beta}}_G^*}{2}$.

Following the above definitions, we propose the JPG-SR model to solve

$$\hat{\mathbf{C}} = \arg\min_{\mathbf{C}} \frac{1}{2}\|\mathbf{X}_G - \mathbf{U}\mathbf{C}\|_F^2 + \tau\|\boldsymbol{\alpha}_G\|_0 + \varphi\|\bar{\boldsymbol{\beta}}_G\|_0,$$
$$\mathbf{U} = [\mathbf{D} \quad \mathbf{D}_G], \quad \mathbf{C} = \begin{bmatrix} \boldsymbol{\alpha}_G \\ \bar{\boldsymbol{\beta}}_G \end{bmatrix}, \tag{13}$$

where $\tau = \frac{\lambda}{2}$ and $\varphi = \frac{\rho}{2}$ are the regularization parameters, balancing the two sparsity inducing penalties ($\|\boldsymbol{\alpha}_G\|_0$ and $\|\bar{\boldsymbol{\beta}}_G\|_0$) and the fidelity term, *i.e.*, $\frac{1}{2}\|\mathbf{X}_G - \mathbf{U}\mathbf{C}\|_F^2$. $\|\boldsymbol{\alpha}_G\|_0$ corresponds to the patch sparsity prior to retain the image local consistency, reducing the over-smooth effect, while $\|\bar{\boldsymbol{\beta}}_G\|_0$ is associated with group sparsity prior to keep image nonlocal consistency, suppressing undesirable visual artifacts. In this way, the proposed JPG-SR provides an effective way to integrate the local sparsity and non-local similarities. Here we add the scalar $\frac{1}{2}$ in the fidelity term to make the optimization convenient. In our experiments, we notice that this joint estimation plays a pivot role in the performance improvement and shows good convergence behaviors (details in Section V).

In Eq. (13), after $\mathbf{C}$ is estimated, we can obtain $\mathbf{X}_G$. Following this, the original image $\boldsymbol{x}$ can be recovered by aggregating the patches. It is worth noting that the desired signal $\mathbf{X}_G$ is only updated when both the patch sparsity prior ($\boldsymbol{\alpha}_G$) and the group sparsity prior ($\bar{\boldsymbol{\beta}}_G$) are available and therefore our model is a joint mechanism.

IV. Joint Patch-Group Based Sparse Representation for Image Restoration

We now apply the proposed JPG-SR model to different image restoration tasks, including image inpainting and image deblocking.

A. Image Restoration

The goal of image restoration is to reconstruct a high quality image $\boldsymbol{x}$ from its degraded observation $\boldsymbol{y}$, which is a typical ill-posed inverse problem and can be mathematically expressed as

$$\boldsymbol{y} = \boldsymbol{H}\boldsymbol{x} + \boldsymbol{n}, \tag{14}$$

where $\boldsymbol{H}$ is degradation operator and $\boldsymbol{n}$ is usually assumed to be a zero-mean white Gaussian noise. With different settings of $\boldsymbol{H}$, various image restoration can be derived from Eq. (14), such as image denoising [23], [27] when $\boldsymbol{H}$ is an identity matrix, image inpainting [28]–[30] when $\boldsymbol{H}$ is a diagonal matrix whose diagonal entries are either 1 or 0, keeping or killing corresponding pixels. In this paper, we mainly focus on the image inpainting and image deblocking problems. Note that, we focus on image deblocking for JPEG compression artifacts reduction [31]–[33], and then image deblocking is regarded as image denoising problem, where $\boldsymbol{n}$ is the quantization noise [34], [35], which is depicted by a Gaussian model [33] in this paper given its simplicity and effectiveness.

Algorithm 1 The ADMM Algorithm

1: Set $t=0$, $\mu>0$, $\mathbf{C}_0=0$, $\mathbf{Z}_0=0$ and $\mathbf{J}_0=0$.
2: **for** $t=0$ **to** Max-Iter **do**
3: $\mathbf{Z}^{t+1}=\arg\min_{\mathbf{Z}} f(\mathbf{Z})+\frac{\mu}{2}\|\mathbf{Z}-\mathbf{U}\mathbf{C}^t-\mathbf{J}^t\|_2^2$.
4: $\mathbf{C}^{t+1}=\arg\min_{\mathbf{C}} g(\mathbf{C})+\frac{\mu}{2}\|\mathbf{Z}^{t+1}-\mathbf{U}\mathbf{C}-\mathbf{J}^t\|_2^2$.
5: $\mathbf{J}^{t+1}=\mathbf{J}^t-(\mathbf{Z}^{t+1}-\mathbf{U}\mathbf{C}^{t+1})$.
6: $t\leftarrow t+1$.
7: **end for**

Given the degraded image $\boldsymbol{y}$ in Eq. (14) and leveraging the proposed JPG-SR in Eq. (13), we aim to recover the original image $\boldsymbol{x}$ by solving the following minimization problem,

$$\hat{\mathbf{C}}=\arg\min_{\mathbf{C}}\frac{1}{2}\|\mathbf{Y}_G-\mathbf{H}_G\mathbf{U}\mathbf{C}\|_F^2+\tau\|\boldsymbol{\alpha}_G\|_0+\varphi\|\bar{\boldsymbol{\beta}}_G\|_0, \quad (15)$$

where $\mathbf{Y}_G$ is obtained from $\boldsymbol{y}$ in the same procedure of $\mathbf{X}_G$, and similarly to $\mathbf{H}_G$, which is obtained from $\boldsymbol{H}$.

B. ADMM Based Algorithm to Solve the Proposed JPG-SR Model

Since Eq. (15) is a large-scale non-convex optimization problem, in order to make the optimization tractable, we employ the alternating direction method of multipliers (ADMM) [36], [37] framework, whose underlying principle is to split the unconstrained minimization problem into different constrained sub-problems. We give a brief introduction to the ADMM method below by considering a constrained optimization problem,

$$\min_{\mathbf{Z}\in\mathbb{R}^N,\mathbf{C}\in\mathbb{R}^M} f(\mathbf{Z})+g(\mathbf{C}), \quad \text{s.t.} \quad \mathbf{Z}=\mathbf{U}\mathbf{C}, \quad (16)$$

where $\mathbf{U}\in\mathbb{R}^{M\times N}$ and $f:\mathbb{R}^N\rightarrow\mathbb{R}$, $g:\mathbb{R}^M\rightarrow\mathbb{R}$. The basic ADMM is shown in Algorithm 1, where t denotes the iteration number.

Now, let us come back to Eq. (15) and invoke ADMM to solve it. We first translate Eq. (15) into an equivalent constrained form by introducing an auxiliary variable $\mathbf{Z}$,

$$\begin{aligned}\hat{\mathbf{C}}= \ &\arg\min_{\mathbf{C},\mathbf{Z}}\frac{1}{2}\|\mathbf{Y}_G-\mathbf{H}_G\mathbf{Z}\|_F^2+\tau\|\boldsymbol{\alpha}_G\|_0+\varphi\|\bar{\boldsymbol{\beta}}_G\|_0,\\ &\text{s.t.} \quad \mathbf{Z}=\mathbf{U}\mathbf{C}.\end{aligned} \quad (17)$$

Through defining $f(\mathbf{Z})=\frac{1}{2}\|\mathbf{Y}_G-\mathbf{H}_G\mathbf{Z}\|_F^2$, $g(\mathbf{C})=\tau\|\boldsymbol{\alpha}_G\|_0+\varphi\|\bar{\boldsymbol{\beta}}_G\|_0$, and employing Line 3 in **Algorithm** 1, we have,

$$\begin{aligned}\hat{\mathbf{Z}}^{t+1}&=\arg\min_{\mathbf{Z}} f(\mathbf{Z})+\frac{\mu}{2}\|\mathbf{Z}-\mathbf{U}\mathbf{C}^t-\mathbf{J}^t\|_F^2\\ &=\arg\min_{\mathbf{Z}}\frac{1}{2}\|\mathbf{Y}_G-\mathbf{H}_G\mathbf{Z}\|_F^2\\ &\quad+\frac{\mu}{2}\left\|\left|\mathbf{Z}-[\mathbf{D}\ \mathbf{D}_G]\begin{bmatrix}\boldsymbol{\alpha}_G^t\\ \bar{\boldsymbol{\beta}}_G^t\end{bmatrix}-\mathbf{J}^t\right\|\right|_F^2\\ &=\arg\min_{\mathbf{Z}}\frac{1}{2}\|\mathbf{Y}_G-\mathbf{H}_G\mathbf{Z}\|_F^2\\ &\quad+\frac{\mu}{2}\|\mathbf{Z}-\mathbf{D}\boldsymbol{\alpha}_G^t-\mathbf{D}_G\bar{\boldsymbol{\beta}}_G^t-\mathbf{J}^t\|_F^2.\end{aligned} \quad (18)$$

where μ is a balance factor.

Next, invoking Line 4 in **Algorithm** 1, we have

$$\begin{aligned}\hat{\mathbf{C}}^{t+1}&=\arg\min_{\mathbf{C}} g(\mathbf{C})+\frac{\mu}{2}\|\mathbf{Z}^{t+1}-\mathbf{U}\mathbf{C}-\mathbf{J}^t\|_F^2\\ &=\arg\min_{\boldsymbol{\alpha}_G,\bar{\boldsymbol{\beta}}_G}\tau\|\boldsymbol{\alpha}_G\|_0+\varphi\|\bar{\boldsymbol{\beta}}_G\|_0\\ &\quad+\frac{\mu}{2}\left\|\left|\mathbf{Z}^{t+1}-[\mathbf{D}\ \mathbf{D}_G]\begin{bmatrix}\boldsymbol{\alpha}_G\\ \bar{\boldsymbol{\beta}}_G\end{bmatrix}-\mathbf{J}^t\right\|\right|_F^2\\ &=\arg\min_{\boldsymbol{\alpha}_G,\bar{\boldsymbol{\beta}}_G}\tau\|\boldsymbol{\alpha}_G\|_0+\varphi\|\bar{\boldsymbol{\beta}}_G\|_0\\ &\quad+\frac{\mu}{2}\|\mathbf{Z}^{t+1}-\mathbf{D}\boldsymbol{\alpha}_G-\mathbf{D}_G\bar{\boldsymbol{\beta}}_G-\mathbf{J}^t\|_F^2.\end{aligned} \quad (19)$$

Then, we decouple the minimization problem of $\mathbf{C}$ in Eq. (19) with respect to $\boldsymbol{\alpha}_G$ and $\bar{\boldsymbol{\beta}}_G$, and solve them separately, *i.e.*,

$$\boldsymbol{\alpha}_G^{t+1}=\arg\min_{\boldsymbol{\alpha}_G}\tau\|\boldsymbol{\alpha}_G\|_0+\frac{\mu}{2}\|\mathbf{Z}^{t+1}-\mathbf{D}\boldsymbol{\alpha}_G-\mathbf{D}_G\bar{\boldsymbol{\beta}}_G-\mathbf{J}^t\|_F^2, \quad (20)$$

$$\bar{\boldsymbol{\beta}}_G^{t+1}=\arg\min_{\bar{\boldsymbol{\beta}}_G}\varphi\|\bar{\boldsymbol{\beta}}_G\|_0+\frac{\mu}{2}\|\mathbf{Z}^{t+1}-\mathbf{D}\boldsymbol{\alpha}_G-\mathbf{D}_G\bar{\boldsymbol{\beta}}_G-\mathbf{J}^t\|_F^2. \quad (21)$$

Following this, we update $\mathbf{J}^t$ by invoking Line 5 in **Algorithm** 1,

$$\mathbf{J}^{t+1}=\mathbf{J}^t-(\mathbf{Z}^{t+1}-\mathbf{D}\boldsymbol{\alpha}_G^{t+1}-\mathbf{D}_G\bar{\boldsymbol{\beta}}_G^{t+1}). \quad (22)$$

In summary, it can be seen that the minimization of Eq. (15) involves three minimization sub-problems, *i.e.*, $\mathbf{Z}$, $\boldsymbol{\alpha}_G$ and $\bar{\boldsymbol{\beta}}_G$. Fortunately, there is an efficient solution to each sub-problem, which will be discussed below. Furthermore, in the image restoration problem considered in this work, each problem can be solved patch by patch. Take the i-th patch $\boldsymbol{x}_i$ as an example, $\boldsymbol{y}_i=\mathbf{H}_i\boldsymbol{x}_i$, where $\mathbf{H}_i$ denotes the degraded matrix in the i-th patch. In the PSR model, recall that $\boldsymbol{\alpha}_G$ is an expanded version of $\boldsymbol{\alpha}$ and we have $\boldsymbol{x}_i=\mathbf{D}\boldsymbol{\alpha}_i$. After $\boldsymbol{\alpha}_i$ is solved, we can straightforwardly obtain $\boldsymbol{\alpha}_G$. In the GSR model, let $\tilde{\boldsymbol{\beta}}_i$ concatenate all the group coefficients including the i-th patch; we thus have $\boldsymbol{x}_i=\mathbf{D}_G\tilde{\boldsymbol{\beta}}_i$. In the following, we consider to solve the problem for each patch or each patch group and the superscript t is omitted for conciseness. More specifically, we translate the $\boldsymbol{\alpha}_G$ sub-problem to $\{\boldsymbol{\alpha}_i\}_{i=1}^n$ subproblem, translate the $\bar{\boldsymbol{\beta}}_G$ sub-problem to $\{\boldsymbol{\beta}_{G_i}\}_{i=1}^n$ sub-problem, and translate the $\mathbf{Z}$ sub-problem to $\{z_i\}_{i=1}^n$ sub-problem, respectively.

1) $\mathbf{Z}$ Sub-Problem : Given $\boldsymbol{\alpha}_G$ and $\bar{\boldsymbol{\beta}}_G$, $\mathbf{Z}$ sub-problem in Eq. (18) for each patch z_i, becomes

$$\begin{aligned}\min_{z_i}\boldsymbol{L}_1(z_i)=\min_{z_i}&\frac{1}{2}\|\boldsymbol{y}_i-\mathbf{H}_iz_i\|_2^2\\ &+\frac{\mu}{2}\|z_i-\mathbf{D}\boldsymbol{\alpha}_i-\mathbf{D}_G\tilde{\boldsymbol{\beta}}_i-\boldsymbol{j}_i\|_2^2 \quad \forall i\ ,\end{aligned} \quad (23)$$

This is a quadratic form and it has a closed-form solution,

$$\hat{z}_i=\left(\mathbf{H}_i^T\mathbf{H}_i+\mu\mathbf{I}\right)^{-1}\left(\mathbf{H}_i^T\boldsymbol{y}_i+\mu(\mathbf{D}\boldsymbol{\alpha}_i+\mathbf{D}_G\tilde{\boldsymbol{\beta}}_i+\boldsymbol{j}_i)\right) \quad \forall i, \quad (24)$$

where $\mathbf{I}$ is an identity matrix with the desired dimensions and $\boldsymbol{j}_i$ is the corresponding elements from $\mathbf{J}$. Note that each z_i is *jointly* estimated in Eq. (24) using both PSR ($\boldsymbol{\alpha}_G$) in Eq. (20)

and GSR ($\bar{\boldsymbol{\beta}}_G$) in Eq. (21) in one shot. Therefore, this is significantly different from the simple average of two results using PSR and GSR independently in our previous work [4].

2) $\boldsymbol{\alpha}_G$ Sub-Problem: Recall that $\boldsymbol{\alpha}_G$ is an expanded version of $\boldsymbol{\alpha}$, and thus the $\boldsymbol{\alpha}_G$ can be solved by the $\boldsymbol{\alpha}$ sub-problem.

According to Eq. (20), for i-th patch, $\boldsymbol{\alpha}_i$ sub-problem can be rewritten as

$$\min_{\boldsymbol{\alpha}_i} \boldsymbol{L}_2(\boldsymbol{\alpha}_i) = \min_{\boldsymbol{\alpha}_i} \left(\frac{1}{2} \|\mathbf{D}\boldsymbol{\alpha}_i - \boldsymbol{r}_i\|_2^2 + \frac{\tau}{\mu} \|\boldsymbol{\alpha}_i\|_0 \right) \quad \forall i \ , \tag{25}$$

where $\boldsymbol{r}_i = \boldsymbol{z}_i - \mathbf{D}_G \tilde{\boldsymbol{\beta}}_i - \boldsymbol{j}_i$. Obviously, this is a sparse representation problem, and we hereby directly solve the constrained form,

$$\min_{\boldsymbol{\alpha}_i} \|\boldsymbol{\alpha}_i\|_0 \quad \text{s.t.} \quad \|\boldsymbol{r}_i - \mathbf{D}\boldsymbol{\alpha}_i\|_2^2 \le \delta \quad \forall i \ , \tag{26}$$

where δ is a small constant, and apparently Eq. (26) can be efficiently solved by the orthogonal matching pursuit (OMP) algorithm [38]. For a given problem, OMP constructs a sparse solution via iteratively building up an approximation, rather than minimizing an objective function. The vector $\boldsymbol{r}_i$ in Eq. (26) is approximated as a linear combination of a few columns in $\mathbf{D}$, where the active set of columns to be used is built column by column, in a greedy fashion. At each iteration a new column is added to the active set of the column that best correlates with the current residual. Although OMP is a heuristic method, in most cases it works marvelously [39].

Moreover, for the dictionary $\mathbf{D}$, at each iteration, we define $\mathbf{R} = \mathbf{Z} - \mathbf{D}_G \bar{\boldsymbol{\beta}}_G - \mathbf{J}$ in Eq. (20) as a good approximation of $\mathbf{D}\boldsymbol{\alpha}_G$. Due to its effectiveness and efficiency, the KSVD algorithm [2] is employed to learn the dictionary $\mathbf{D}$ from $\mathbf{R}$ in each iteration.

3) $\bar{\boldsymbol{\beta}}_G$ Sub-Problem : Given $\mathbf{Z}$ and $\boldsymbol{\alpha}_G$, according to Eq. (21), $\bar{\boldsymbol{\beta}}_G$ sub-problem can be rewritten as

$$\min_{\bar{\boldsymbol{\beta}}_G} \boldsymbol{L}_3(\bar{\boldsymbol{\beta}}_G) = \min_{\bar{\boldsymbol{\beta}}_G} \left(\frac{1}{2} \|\mathbf{D}_G \bar{\boldsymbol{\beta}}_G - \mathbf{R}_G\|_F^2 + \frac{\varphi}{\mu} \|\bar{\boldsymbol{\beta}}_G\|_0 \right), \tag{27}$$

where $\mathbf{R}_G = \mathbf{Z} - \mathbf{D}\boldsymbol{\alpha}_G - \mathbf{J}$.

Recalling the relationship of $\bar{\boldsymbol{\beta}}_G$, $\tilde{\boldsymbol{\beta}}$ and $\boldsymbol{\beta}$, for each patch, we can get the other two after solving any one of them. Now, instead of considering each patch as in the $\boldsymbol{\alpha}$ sub-problem, we consider each *patch group* here. For i-th patch group, we aim to solve

$$\hat{\boldsymbol{\beta}}_{G_i} = \arg\min_{\boldsymbol{\beta}_{G_i}} \left(\frac{1}{2} \|\mathbf{R}_{G_i} - \mathbf{D}_{G_i} \boldsymbol{\beta}_{G_i}\|_F^2 + \frac{\varphi}{\mu} \|\boldsymbol{\beta}_{G_i}\|_0 \right) \quad \forall i. \tag{28}$$

One important issue of solving sub-problem $\boldsymbol{\beta}_G$ is the selection of the dictionary. To adapt to the local image structures, instead of learning an over-complete dictionary for each patch group as in [5], we learn the principle component analysis (PCA) based sub-dictionary $\mathbf{D}_{G_i}$ [21] from each patch group $\mathbf{R}_{G_i}$. Due to the orthogonality of the dictionary $\mathbf{D}_{G_i}$ and based on the orthogonal invariance, Eq. (28) can be rewritten as

$$\begin{aligned} \hat{\boldsymbol{\beta}}_{G_i} &= \min_{\boldsymbol{\beta}_{G_i}} \left(\frac{1}{2} \|\boldsymbol{\gamma}_{G_i} - \boldsymbol{\beta}_{G_i}\|_F^2 + \frac{\varphi}{\mu} \|\boldsymbol{\beta}_{G_i}\|_0 \right) \\ &= \min_{\boldsymbol{\beta}_i} \left(\frac{1}{2} \|\boldsymbol{\gamma}_i - \boldsymbol{\beta}_i\|_2^2 + \frac{\varphi}{\mu} \|\boldsymbol{\beta}_i\|_0 \right) \quad \forall i \ , \end{aligned} \tag{29}$$

Algorithm 2 Image Restoration Using JPG-SR Model

Require: The observed image $\boldsymbol{y}$ and measurement matrix $\mathbf{H}$.

Set parameters t, $\mathbf{Z}$, $\boldsymbol{\alpha}$, $\boldsymbol{\beta}_G$, $\mathbf{J}$, b, c, m, W, μ, τ, ω, $\boldsymbol{\sigma}_n$, ϵ, ε.
for $t = 0$ **to** Max-Iter **do**
  Update $\mathbf{Z}^{t+1}$ by Eq. (24);
  $\mathbf{R}^{t+1} = \mathbf{Z}^{t+1} - \mathbf{D}_G \bar{\boldsymbol{\beta}}_G - \mathbf{J}^t$;
  Construct dictionary $\mathbf{D}$ by $\mathbf{R}^{t+1}$ with KSVD.
  for Each patch $\boldsymbol{r}_i$ **do**
    Update $\boldsymbol{\alpha}_i^{t+1}$ by computing Eq. (26);
  end for
  $\mathbf{R}_G^{t+1} = \mathbf{Z}^{t+1} - \mathbf{D}\boldsymbol{\alpha}_G - \mathbf{J}^t$;
  for Each patch group $\mathbf{R}_{G_i}$ **do**
    Construct dictionary $\mathbf{D}_{G_i}$ from $\mathbf{R}_{G_i}^{t+1}$ using PCA;
    Update $\boldsymbol{\beta}_{G_i}^{t+1}$ by computing Eq. (30);
  end for
  Update $\boldsymbol{\alpha}^{t+1}$ by concatenating all $\boldsymbol{\alpha}_i$;
  Update $\mathbf{D}_G^{t+1}$ by concatenating all $\mathbf{D}_{G_i}$;
  Update $\boldsymbol{\beta}_G^{t+1}$ by concatenating all $\boldsymbol{\beta}_{G_i}$;
  Update $\mathbf{J}^{t+1}$ by Eq. (22);
end for
Output: The final restored image $\hat{\boldsymbol{x}}$ by aggregating patches in $\mathbf{Z}$.

where $\mathbf{R}_{G_i} = \mathbf{D}_{G_i} \boldsymbol{\gamma}_{G_i}$, and $\{\boldsymbol{\gamma}_i, \boldsymbol{\beta}_i\}$ denote the vectorization form of the matrix $\{\boldsymbol{\gamma}_{G_i}, \boldsymbol{\beta}_{G_i}\}$, respectively.

We can achieve a closed-form solution of each $\boldsymbol{\beta}_i$ in Eq. (29) according to the so called hard thresholding [50]:

$$\hat{\boldsymbol{\beta}}_i = \text{hard}(\boldsymbol{\gamma}_i, \sqrt{2\varphi/\mu}) = \boldsymbol{\gamma}_i \odot \mathbf{1}(\text{abs}(\boldsymbol{\gamma}_i) - \sqrt{2\varphi/\mu}) \quad \forall i \ . \tag{30}$$

This process is performed across all n patch groups to achieve $\boldsymbol{\beta}_G$, which is the final solution for $\bar{\boldsymbol{\beta}}_G$ sub-problem in Eq. (21).

After solving the above three sub-problems, we summarize the overall algorithm to solve Eq. (15) in Algorithm 2.

Till now, we have applied the ADMM based algorithm to solve the proposed JPG-SR model for image restoration in detail. We exploit the case of image inpainting as an example and the flowchart of the proposed JPG-SR model is illustrated in Fig. 1. Note that the PSR (the top row in Fig. 1) and the GSR (the bottom row in Fig. 1) are fed into an ADMM framework to recover the image (right part in Fig. 1) in one shot. Experimental results demonstrate that the proposed model is effective and outperforms several state-of-the-art approaches in Section V.

C. Quantization Noise Model

Different from the image inpainting task investigated above, in image deblocking, the observed JPEG-coded image is generally modeled as an image corrupted by the quantization noise,

$$\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{e}, \tag{31}$$

where $\boldsymbol{y}$ is the JPEG-coded image with blocking artifacts, $\boldsymbol{x}$, $\boldsymbol{e}$ are the original image and quantization noise, respectively.

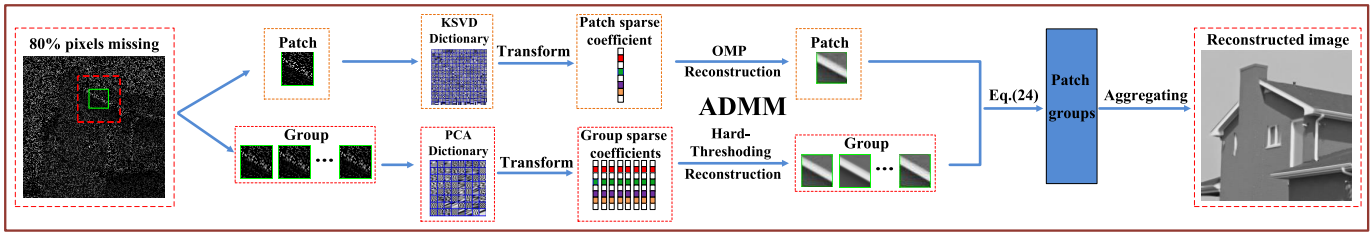


Fig. 1. Flowchart of the proposed JPG-SR model for image inpainting. The corrupted image (left) are fed into our JPG-SR model on two paths, *i.e.*, PSR on top and GSR in the bottom. These two paths are jointly optimized by the proposed ADMM framework producing the finally recovered image (right).

One important issue of image deblocking is how to set the quantization noise model to describe $\boldsymbol{e}$. There are many distinct models about the quantization noise [34], [51], [52], and the Gaussian model has been extensively used for estimating quantization noise because of its simplicity and effectiveness, which has also achieved excellent deblocking results [32], [34], [35]. Thus, in this paper, we adopt a Gaussian model to characterize the quantization noise $\boldsymbol{e}$ in Eq. (31), and specifically the approach proposed in [34] is utilized to estimate the noise variance $\boldsymbol{\sigma}_e^2$,

$$\boldsymbol{\sigma}_e^2 = 0.69\ (\tilde{\boldsymbol{s}})^{1.3}, \qquad \tilde{\boldsymbol{s}} = \frac{1}{9}\sum\nolimits_{i,j=1}^{3} \boldsymbol{M}^q_{[i,j]}, \tag{32}$$

where $\boldsymbol{M}^q$ is the 8×8 quantization matrix with quality factor (QF) of q, $\tilde{\boldsymbol{s}}$ is the mean value of the nine upper-left entries in $\boldsymbol{M}^q$, corresponding to lowest-frequency DCT harmonics, and we utilize $\boldsymbol{M}^q_{[i,j]}$ to represent the $(i, j)^{th}$ element in $\boldsymbol{M}^q$. It is noticed that the noise variance $\boldsymbol{\sigma}_e^2$ obtained by Eq. (32) is only the variance of the hypothetical Gaussian noise, which determines the level of adaptive smoothing that is able to reduce compression artifacts generated by the quantization step with $\boldsymbol{M}^q$ [53]. Moreover, after obtaining the solution of $\mathbf{Z}$ sub-problem, the quantization constraint prior [32], [35] is imposed to further improve the deblocking performance of the proposed JPG-SR algorithm (For the details of the quantization constraint prior, please refer to [35]). To make the proposed algorithm more accurate and practical, according to [54], in the t-th iteration, we set the ADMM balance factor μ to

$$\mu^t = \frac{1}{\omega\ (\boldsymbol{\sigma}_s^2)^t}, \tag{33}$$

where ω is a scaling factor. One can observe that the estimation of μ^t is dependent on the estimation of $(\boldsymbol{\sigma}_s^2)^t$. Inspired by [55], the iterative regularization strategy is exploited to update the estimation of the noise variance $\boldsymbol{\sigma}_s$. Specifically, the standard deviation of noise $\boldsymbol{\sigma}_s$ in the t-th iteration is computed as

$$(\boldsymbol{\sigma}_s)^t = \eta\ \sqrt{\boldsymbol{\sigma}_e^2 - \|\hat{\boldsymbol{x}}^{(t)} - \boldsymbol{y}\|_2^2}, \tag{34}$$

where η is a constant and this scheme has been widely used in the Gaussian noise variance estimation.

Both τ and φ are regularization parameters. In our algorithm, τ is set by using the same setting as in the KSVD dictionary learning method [1]. For φ, inspired by [56], for each group sparse coefficient $\boldsymbol{\gamma}_i$, it is set to

$$\varphi = \frac{c\ 2\sqrt{2}\boldsymbol{\sigma}_n^2}{\delta_i + \epsilon}, \tag{35}$$

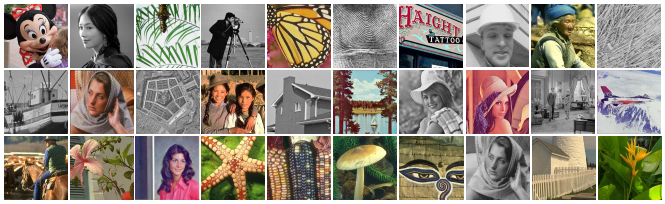


Fig. 2. Test images in the experiments. **T**op row: Mickey, Lin, Leaves, C. Man, Butterfly, F. Print, Haight, Foreman, Nanna, Straw. **M**iddle row: boat, Barbara, Pentagon, Girls, House, Lake, Elaine, Lena, Couple, Airplane. **B**ottom row: Cowboy, Flower, Miss, Starfish, Corn, Penester, Mural, Barbara, Fence, Plants.

where $\boldsymbol{\sigma}_n$ represents the noise variance. δ_i denotes the estimated variance of $\boldsymbol{\gamma}_i$ [57], and ϵ is a small positive constant.

D. Summary of the Proposed Algorithm

Till to now, we have solved the above three sub-problems $\mathbf{Z}$, $\boldsymbol{\alpha}_G$ and $\boldsymbol{\beta}_G$. We can achieve an efficient solution by solving each sub-problem separately, which can ensure the whole algorithm to be efficient and effective. Meanwhile, the quantization noise model, the quantization constraint prior and the adaptive parameter setting of ADMM balance factor μ have been described in the task of image deblocking.

V. Experimental Results

Extensive experiments are conducted in this section to verify the performance of the proposed JPG-SR based image restoration algorithm by two image restoration tasks, namely, image inpainting and image deblocking. The experimental test images are shown in Fig. 2. Both PSNR and structural similarity (SSIM) [58] metrics are used to evaluate the quality of the reconstructed images. The source codes of all competing methods are obtained from their original authors. We used the default parameters in their software packages.[2] Due to the limited space, please enlarge the tables and figures on the screen for better comparison. Throughout the numerical experiments, we choose the following stopping criterion of iteration for the proposed JPG-SR based image restoration algorithm, *i.e.*,

$$\frac{\|\hat{\boldsymbol{x}}^t - \hat{\boldsymbol{x}}^{t-1}\|_2^2}{\|\hat{\boldsymbol{x}}^{t-1}\|_2^2} < \varepsilon, \tag{36}$$

[2]The authors would like to appreciate the authors of [2], [23], [28], [33]–[35], [40]–[49], [53], [59]–[63] for providing their source codes or experimental results.

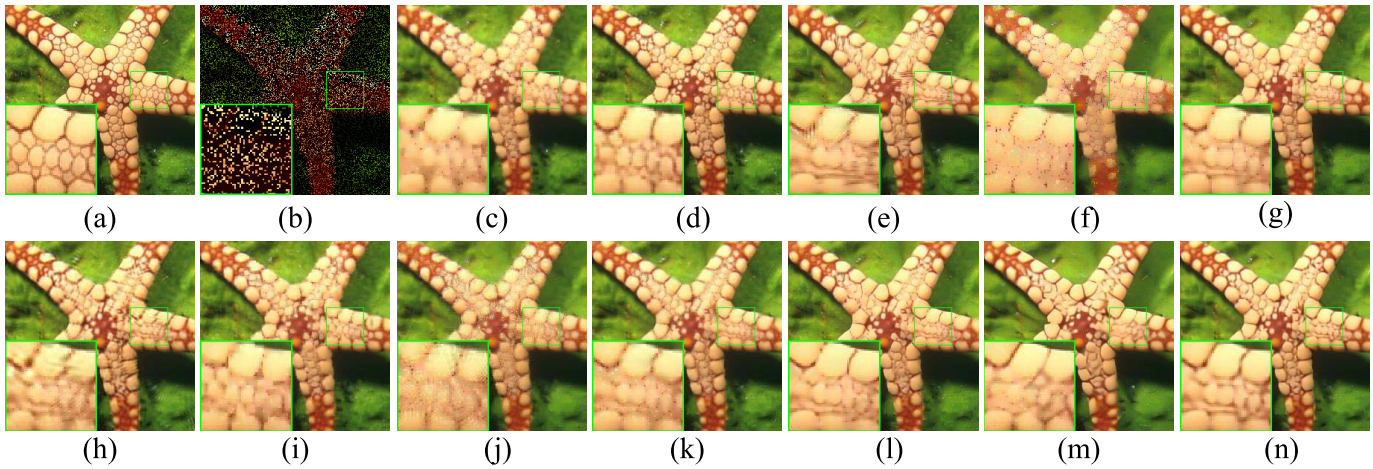


Fig. 3. Visual comparison of *Starfish* by image inpainting with 80% missing pixels. (a) Original image; (b) Degraded image with 80% pixels missing; (c) SALSA [40] (PSNR = 25.70dB, SSIM = 0.8086); (d) BPFA [28] (PSNR = 26.79dB, SSIM = 0.8379); (e) IPPO [41] (PSNR = 26.30dB, SSIM = 0.8243); (f) ISD-SB [42] (PSNR = 23.00dB, SSIM = 0.7035); (g) JSM [43] (PSNR = 27.07dB, SSIM = 0.8383); (h) Aloha [44] (PSNR = 26.33dB, SSIM = 0.8217); (i) NGS [45] (PSNR = 26.17dB, SSIM = 0.8272); (j) BKSVD [46] (PSNR = 25.36dB, SSIM = 0.7741); (k) TSLRA [47] (PSNR = 26.84dB, SSIM = 0.8311); (l) IRCNN [48] (PSNR = 27.02dB, SSIM = 0.8448); (m) IDBP [49] (PSNR = 26.88dB, SSIM = 0.8316); (n) JPG-SR (PSNR = **27.86dB**, SSIM = **0.8625**).

where ε is a small constant. The source code of the proposed JPG-SR is available at: https://drive.google.com/open?id=1KMIERcJtZYKdGt2HvUySFtviAC5RprHu.

A. Image Inpainting

We first report the performance of the proposed JPG-SR for image inpainting and compare it with several leading methods, including SALSA [40], BPFA [28], IPPO [41], ISD-SB [42], JSM [43], Aloha [44], NGS [45], BKSVD [46], TSLRA [47], IRCNN [48] and IDBP [49] methods. Note that BKSVD is a classical PSR method, and both JSM and NGS are based on the GSR methods. TSLRA is a low-rank method that delivers the state-of-the-art image inpainting result. IRCNN and IDBP are the deep learning based methods, which employ the strong deep convolutional neural networks (CNN) [64]. In this subsection, we conduct two interesting examples with different masks for image inpainting, *i.e.*, partial random samples and text inlayed sample. The parameter setting of our proposed JPG-SR for image inpainting is as follows. The size of each patch $\sqrt{b} \times \sqrt{b}$ is set to be 8×8. The size of searching window $W \times W$ is set to 25×25 and the matched patch number in each patch group $m = 60$. $\mu = 0.2$, $c = 0.2$, $\boldsymbol{\sigma}_n = \sqrt{2}$ and $\epsilon = e^{-14}$. The parameter ε is set to 0.0007, 0.00048, 0.00066, 0.0005 and 0.0003 when 80%, 70%, 60%, 50% pixels missing and text inlayed, respectively.

Table I presents the PSNR comparison for a collection of 16 test images used in all competing methods. It can be found that the proposed JPG-SR can outperform other competing methods in most cases. The average gains of the proposed JPG-SR over SALSA, BPFA, IPPO, ISD-SB, JSM, Aloha, NGS, BKSVD, TSLRA, IRCNN and IDBP methods are as much as 4.22dB, 2.17dB, 0.96dB, 5.51dB, 1.16dB, 1.50dB, 2.95dB, 2.98dB, 1.48dB, 0.82dB and 1.55dB, respectively. The SSIM comparison results are shown in Table II, where we can also observe that the proposed JPG-SR achieves better results than all competing methods in most cases.

The visual comparisons of images *Starfish* and *Flower* with 80% pixels missing are provided in Fig. 3 and Fig. 4, respectively. Meanwhile, we show the visual comparisons of the image *Haight* with text inlayed in Fig. 5. Obviously, SALSA and ISD-SB could not reconstruct sharp edges and fine details. The BPFA, IPPO, Aloha, BKSVD and IDBP methods produce images with a much better visual quality than SALSA and ISD-SB, but still suffer from some undesirable artifacts, such as the ringing effects. Note that BKSVD produces the obvious undesirable visual artifacts since it is a PSR method. Although JSM, NGS, TSLRA and IRCNN methods can obtain more visual results than BKSVD, they often generate over-smooth effect. The proposed JPG-SR not only preserves sharp edges and fine details, but also eliminates the ringing effects. Furthermore, the proposed JPG-SR has a promising performance for repairing multiple irregular scratch images. We show two inpainted images from our proposed JPG-SR and TSLRA methods for the application of multiple irregular scratch removal in Fig. 6 and Fig. 7. Note that TSLRA is a state-of-the-art inpainting method, which can be competent to various image inpainting tasks [47]. It can be seen that the proposed JPG-SR can effectively remove multiple irregular scratch of images with comparison to TSLRA method. Therefore, these experimental findings clearly demonstrate the effectiveness of our proposed JPG-SR model.

B. Image Deblocking

Next, we validate the performance of the proposed JPG-SR for image deblocking, *i.e.*, restoring JPEG-compressed images. We compare it with popular or recently proposed state-of-the-art deblocking methods: KSVD [2], field of experts (FoE) [35], block matching and 3D filtering (BM3D) [23], shape adaptive discrete cosine transform (SA-DCT) [34], patch clustering and low-rank minimization (PC-LRM) [59], adaptive nonlocal coefficients estimation (ANCE) [60], DicTV [53], weighted

TABLE I

PSNR (dB) COMPARISON OF SALSA [40], BPFA [28], IPPO [41], ISD-SB [42], JSM [43], ALOHA [44], NGS [45], BKSVD [46], TSLRA [47], IRCNN [48], IDBP [49] AND JPG-SR FOR IMAGE INPAINTING

Miss pixels	Methods	Cowboy	Mickey	Barbara	Butterfly	Fence	Haight	Lake	Leaves	Lena	Flower	Starfish	Nanna	Corn	Penester	Girls	Mural	Average
80%	SALSA [40]	23.72	24.46	22.62	22.85	21.80	18.57	24.94	22.03	28.20	26.57	25.70	24.12	24.28	27.42	23.79	23.15	24.01
	BPFA [28]	24.93	24.53	25.11	24.04	26.24	19.42	25.82	23.78	29.50	27.30	26.79	24.71	25.54	27.93	24.80	24.13	25.29
	IPPO [41]	25.38	26.33	28.32	25.13	27.98	20.90	25.48	25.56	30.64	28.33	26.30	25.60	25.14	29.10	25.31	25.66	26.32
	ISD-SB [42]	21.96	22.25	22.35	18.57	21.39	17.00	22.57	18.70	26.01	24.53	23.00	22.31	20.31	26.51	21.99	21.42	21.93
	JSM [43]	25.40	26.09	26.95	25.57	28.59	21.37	25.82	26.18	30.46	27.99	27.07	25.33	25.58	28.25	25.18	25.40	26.33
	Aloha [44]	25.06	25.33	29.59	24.88	28.88	20.62	25.32	25.90	30.89	27.70	26.33	25.54	25.60	28.63	25.16	25.23	26.29
	NGS [45]	24.21	24.50	23.88	23.85	25.26	18.76	25.10	23.87	28.87	27.08	26.17	24.58	24.74	27.86	24.27	23.78	24.80
	BKSVD [46]	24.12	23.72	25.21	22.00	24.20	18.83	24.17	22.05	28.16	26.49	25.36	23.97	23.69	27.53	23.82	23.02	24.15
	TSLRA [47]	25.36	25.71	28.22	25.32	28.83	20.85	25.47	25.47	30.58	28.17	26.84	25.55	25.65	28.85	25.15	25.21	26.33
	IRCNN [48]	25.47	**26.45**	26.21	25.34	27.76	20.43	25.24	25.48	30.78	28.41	27.02	**25.76**	24.76	28.85	25.36	25.75	26.19
	IDBP [49]	24.43	25.40	22.73	25.24	25.03	19.51	25.39	25.84	29.69	28.12	26.88	25.51	26.05	27.90	25.03	25.26	25.50
	JPG-SR	**25.57**	26.34	**30.61**	**26.28**	**29.65**	**21.42**	**25.92**	**27.36**	**31.48**	**28.97**	**27.86**	25.65	**26.80**	**29.16**	**25.61**	**26.17**	**27.18**
70%	SALSA [40]	25.70	25.98	23.38	25.06	23.57	19.95	26.76	24.36	28.82	28.35	27.55	25.44	26.11	28.69	25.47	25.00	25.64
	BPFA [28]	26.76	26.16	28.32	26.68	28.87	21.46	27.93	26.98	31.62	29.30	28.93	26.62	27.82	29.65	26.86	26.46	27.53
	IPPO [41]	27.40	28.59	30.89	27.68	30.08	23.02	27.56	28.58	32.97	30.28	28.91	27.44	27.77	31.14	27.43	27.92	28.60
	ISD-SB [42]	24.28	24.40	23.56	22.65	23.16	18.89	24.41	21.85	28.16	26.46	25.09	24.36	22.46	27.99	24.13	23.48	24.08
	JSM [43]	27.11	28.25	30.48	27.97	30.46	23.01	27.88	29.28	32.69	29.83	29.36	27.34	27.66	30.31	27.20	27.59	28.53
	Aloha [44]	27.24	27.11	32.40	27.29	30.57	22.12	27.58	29.04	32.80	29.58	28.22	27.43	27.95	30.78	27.08	27.33	28.41
	NGS [45]	26.19	26.68	26.11	26.36	27.32	21.03	27.01	26.44	30.77	28.83	28.35	26.35	26.77	29.49	26.18	26.06	26.87
	BKSVD [46]	25.99	26.17	27.58	25.00	28.35	21.12	26.35	25.29	30.96	28.65	27.79	26.18	25.83	29.44	25.82	25.57	26.63
	TSLRA [47]	27.12	27.64	30.79	27.76	30.75	22.61	27.32	28.03	32.64	29.92	28.78	27.32	27.66	30.82	27.09	27.41	28.35
	IRCNN [48]	**27.54**	**29.66**	29.31	28.34	30.54	22.52	27.59	29.06	33.39	30.77	29.79	**28.27**	27.81	31.16	27.78	**28.73**	28.89
	IDBP [49]	27.15	28.69	26.23	28.29	29.24	21.93	27.87	28.99	32.31	30.49	29.28	27.24	28.09	30.33	27.45	27.68	28.20
	JPG-SR	27.51	28.91	**33.84**	**29.18**	**31.77**	**23.58**	**27.99**	**30.92**	**33.56**	**31.08**	**30.32**	27.94	**29.39**	**31.24**	**27.89**	28.29	**29.59**
60%	SALSA [40]	26.99	27.41	24.57	26.79	25.45	21.52	28.14	26.29	31.49	29.65	29.09	26.94	27.75	30.08	27.02	26.66	27.24
	BPFA [28]	28.42	27.83	31.06	28.88	30.79	23.33	29.75	29.83	33.54	31.35	30.98	28.63	30.07	31.60	28.75	28.30	29.57
	IPPO [41]	29.58	30.76	33.55	29.85	32.14	25.34	29.30	30.88	34.89	32.17	31.09	29.41	29.75	32.89	29.32	29.57	30.66
	ISD-SB [42]	26.00	26.59	24.86	25.07	25.30	21.02	26.39	24.55	30.52	28.23	27.36	26.05	24.60	29.16	25.80	25.35	26.05
	JSM [43]	28.89	29.85	33.21	29.83	32.23	24.70	29.49	31.47	34.56	31.59	31.40	29.09	29.45	32.10	29.01	29.24	30.38
	Aloha [44]	28.92	28.59	35.13	29.16	32.33	23.58	29.24	31.41	34.72	31.47	30.19	29.51	29.83	32.20	28.91	28.92	30.26
	NGS [45]	27.78	28.09	28.24	28.37	30.11	22.81	28.68	28.87	32.81	30.53	30.26	28.06	28.55	31.12	27.83	27.99	28.76
	BKSVD [46]	28.14	28.53	29.86	27.70	30.72	23.39	28.45	28.61	33.48	31.00	29.99	28.35	28.35	31.57	28.12	27.90	29.01
	TSLRA [47]	28.83	29.28	33.37	29.42	32.32	24.21	29.01	30.19	34.26	31.55	30.69	29.17	29.39	32.34	28.79	29.07	30.12
	IRCNN [48]	**29.95**	**31.82**	31.87	30.41	32.46	24.90	29.75	32.10	35.51	32.97	32.29	**30.49**	30.32	**33.21**	**30.10**	**30.58**	31.17
	IDBP [49]	28.97	31.18	28.73	30.03	31.25	24.11	29.53	31.74	33.93	32.16	31.79	29.51	29.94	32.04	29.32	29.79	30.25
	JPG-SR	29.57	30.93	**36.11**	**31.19**	**33.48**	**25.79**	**30.05**	**33.26**	**35.72**	**33.09**	**32.78**	30.22	**31.33**	32.93	29.86	29.96	**31.64**
50%	SALSA [40]	28.59	28.98	25.66	28.52	27.25	23.06	29.69	28.11	33.08	31.13	30.90	28.53	29.39	31.51	28.60	28.20	28.83
	BPFA [28]	30.21	29.43	34.01	30.98	32.82	25.40	31.78	32.79	35.61	33.41	33.13	30.68	32.10	33.32	30.58	30.46	31.67
	IPPO [41]	31.30	32.74	35.91	31.69	33.95	27.53	30.98	33.32	36.50	34.04	33.10	31.17	31.76	34.55	31.05	31.11	32.54
	ISD-SB [42]	27.80	27.96	26.57	27.76	27.60	22.92	28.31	26.97	32.04	30.05	29.16	27.50	26.65	30.42	27.56	26.95	27.89
	JSM [43]	30.75	31.96	35.87	31.47	33.75	26.67	31.18	33.78	36.39	33.46	33.24	30.75	31.33	33.94	30.68	30.89	32.26
	Aloha [44]	30.46	30.33	37.46	30.78	33.79	25.16	31.17	34.01	36.41	33.33	31.85	31.24	31.89	33.85	30.59	30.28	32.04
	NGS [45]	29.32	29.75	30.93	30.28	32.00	24.50	30.22	31.23	34.56	32.31	32.10	29.71	30.31	32.89	29.60	29.88	30.60
	BKSVD [46]	29.75	29.95	33.58	29.64	32.44	25.23	30.45	31.25	35.44	32.93	31.99	30.15	30.28	33.13	29.98	29.59	30.99
	TSLRA [47]	30.45	31.00	35.74	31.01	33.89	26.02	30.53	32.56	35.52	33.20	32.44	30.87	31.25	33.68	30.48	30.62	31.83
	IRCNN [48]	**31.96**	**34.25**	34.47	32.48	34.32	27.36	31.80	35.44	**37.53**	**35.28**	34.50	**32.42**	32.71	**35.16**	**32.10**	**32.20**	33.37
	IDBP [49]	31.40	33.14	31.57	32.44	33.24	26.71	31.47	34.34	36.25	34.15	33.81	31.35	31.61	33.68	31.09	31.35	32.35
	JPG-SR	31.64	33.71	**38.31**	**32.93**	**35.04**	**28.46**	**31.91**	**35.84**	37.40	35.20	**34.80**	32.27	**33.59**	34.82	31.90	31.75	**33.72**
Text Inlayed	SALSA [40]	30.17	30.67	29.18	29.81	26.77	24.67	32.22	29.03	33.84	33.59	32.60	30.31	30.96	33.56	30.27	29.11	30.42
	BPFA [28]	31.13	31.70	34.27	31.71	32.23	26.64	33.15	31.78	35.27	35.18	33.88	31.76	32.16	34.73	31.28	30.88	32.36
	IPPO [41]	32.61	34.04	37.65	33.98	35.10	29.10	33.25	35.26	37.29	36.67	35.35	33.01	32.48	36.93	**32.65**	33.52	34.31
	ISD-SB [42]	29.83	29.96	30.43	28.09	27.62	24.61	30.93	27.64	33.13	32.72	31.38	29.58	28.32	33.52	29.77	28.35	29.74
	JSM [43]	32.42	32.99	37.79	33.19	35.41	28.69	33.24	35.40	36.98	35.67	35.17	32.40	32.26	36.56	32.20	32.75	33.94
	Aloha [44]	30.94	30.49	39.16	31.58	34.94	26.21	31.98	34.74	36.03	34.47	32.06	31.91	32.04	35.17	30.84	31.59	32.76
	NGS [45]	30.80	31.10	33.57	31.78	28.73	26.16	32.32	30.05	34.71	34.00	33.02	31.08	31.65	33.55	31.04	30.37	31.50
	BKSVD [46]	31.20	31.43	35.16	29.09	31.69	26.59	31.86	29.74	34.66	34.01	32.83	31.39	29.70	34.46	30.81	30.51	31.57
	TSLRA [47]	32.05	32.43	37.78	32.64	35.23	28.21	32.38	33.66	32.76	35.42	34.50	31.99	31.96	36.30	31.94	32.43	33.23
	IRCNN [48]	32.34	32.96	34.94	33.01	32.94	26.88	32.77	35.52	37.09	36.33	34.46	33.10	32.14	36.59	32.27	32.82	33.51
	IDBP [49]	32.08	33.00	34.48	33.15	31.20	26.46	33.18	34.66	36.97	36.08	34.50	32.83	32.48	36.15	32.23	31.84	33.20
	JPG-SR	**33.05**	**35.10**	**40.38**	**34.49**	**37.11**	**30.08**	**33.78**	**36.74**	**37.88**	**36.69**	**36.54**	**33.54**	**33.36**	**36.96**	32.59	**33.69**	**35.12**

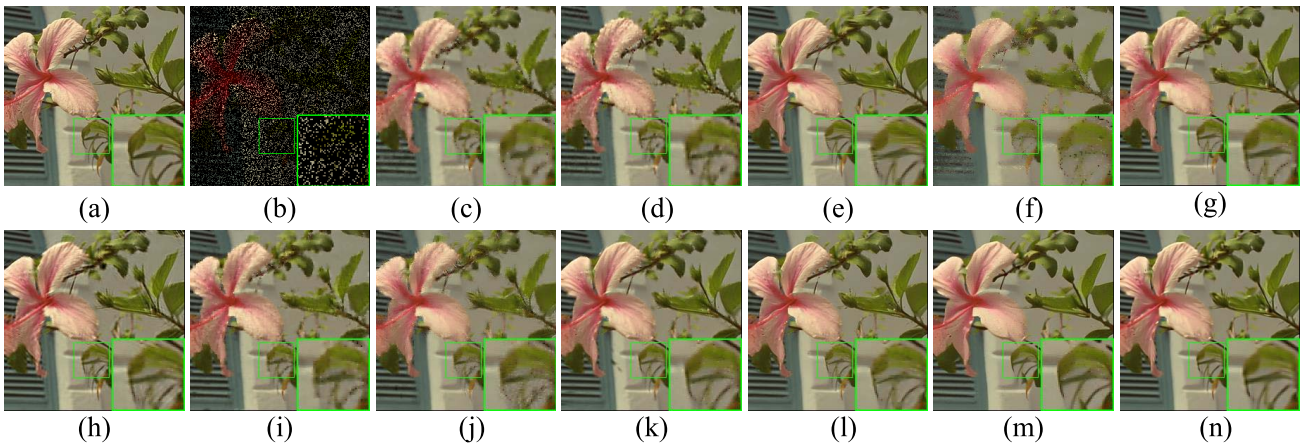


Fig. 4. Visual comparison of *Flower* by image inpainting with 80% missing pixels. (a) Original image; (b) Degraded image with 80% pixels missing; (c) SALSA [40] (PSNR = 26.57dB, SSIM = 0.7964); (d) BPFA [28] (PSNR = 27.30dB, SSIM = 0.8234); (e) IPPO [41] (PSNR = 28.33dB, SSIM = 0.8587); (f) ISD-SB [42] (PSNR = 24.53dB, SSIM = 0.7059); (g) JSM [43] (PSNR = 27.99dB, SSIM = 0.8418); (h) Aloha [44] (PSNR = 27.70dB, SSIM = 0.8402); (i) NGS [45] (PSNR = 27.08dB, SSIM = 0.8174); (j) BKSVD [46] (PSNR = 26.49dB, SSIM = 0.7804); (k) TSLRA [47] (PSNR = 28.17dB, SSIM = 0.8460); (l) IRCNN [48] (PSNR = 28.41dB, SSIM = 0.8586); (m) IDBP [49] (PSNR = 28.12dB, SSIM = 0.8413); (n) JPG-SR (PSNR = **28.97dB**, SSIM = **0.8768**).

nuclear norm minimization (WNNM) [61], structured sparse representation and quantization constraint (SSR-QC) [33], total generalized variation and shearlet transform (TGV-SH) [62] and Coefficient Graph Laplacians (COGL) [63]. Note that KSVD and DicTV exploit the PSR model for image deblocking, while SSR-QC is based on the GSR model.

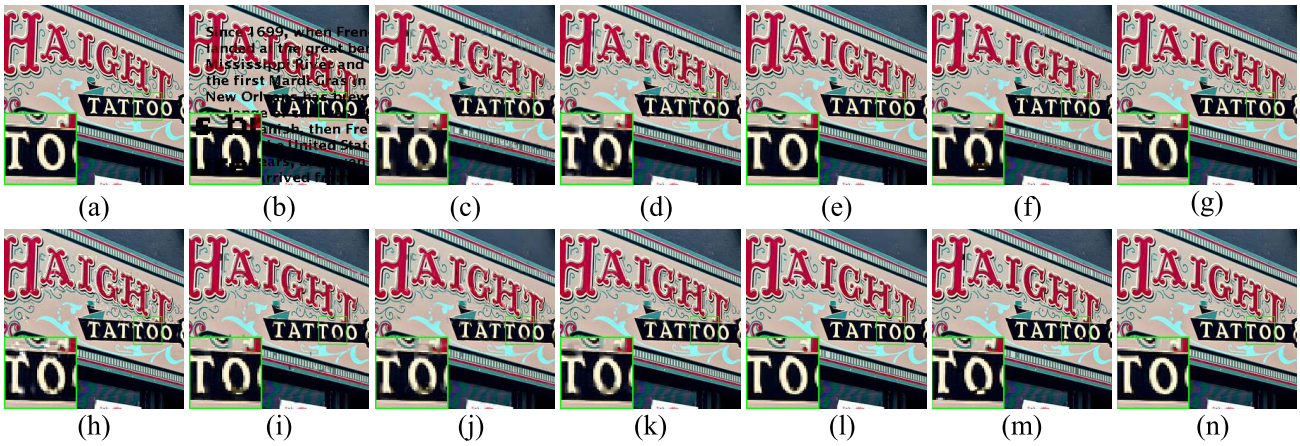


Fig. 5. Visual comparison of *Haight* by image inpainting with text inlayed. (a) Original image; (b) Degraded image with text inlayed; (c) SALSA [40] (PSNR = 24.67dB, SSIM = 0.9322); (d) BPFA [28] (PSNR = 26.64dB, SSIM = 0.9482); (e) IPPO [41] (PSNR = 29.10dB, SSIM = 0.9751); (f) ISD-SB [42] (PSNR = 24.61dB, SSIM = 0.9358); (g) JSM [43] (PSNR = 28.69dB, SSIM = 0.9729); (h) Aloha [44] (PSNR = 26.21dB, SSIM = 0.9492); (i) NGS [45] (PSNR = 26.16dB, SSIM = 0.9523); (j) BKSVD [46] (PSNR = 26.59dB, SSIM = 0.9423); (k) TSLRA [47] (PSNR = 28.21dB, SSIM = 0.9691); (l) IRCNN [48] (PSNR = 26.88dB, SSIM = 0.9582); (m) IDBP [49] (PSNR = 26.46dB, SSIM = 0.9637); (n) JPG-SR (PSNR = **30.08dB**, SSIM = **0.9770**).

TABLE II

SSIM COMPARISON OF SALSA [40], BPFA [28], IPPO [41], ISD-SB [42], JSM [43], ALOHA [44], NGS [45], BKSVD [46], TSLRA [47], IRCNN [48], IDBP [49] AND JPG-SR FOR IMAGE INPAINTING

Miss pixels	Methods	Cowboy	Mickey	Barbara	Butterfly	Fence	Haight	Lake	Leaves	Lena	Flower	Starfish	Nanna	Corn	Penester	Girls	Mural	Average
80%	SALSA [40]	0.7875	0.8145	0.6975	0.8252	0.6546	0.6961	0.8036	0.7948	0.8576	0.7964	0.8086	0.7819	0.7953	0.7721	0.7535	0.7155	0.7722
	BPFA [28]	0.8459	0.8117	0.8042	0.8517	0.7960	0.7307	0.8360	0.8557	0.8899	0.8234	0.8379	0.8011	0.8334	0.7950	0.7846	0.7402	0.8148
	IPPO [41]	0.8642	0.8678	0.8834	0.8995	0.8614	0.8251	0.8297	0.9119	0.9085	0.8587	0.8243	0.8311	0.8281	0.8378	0.8140	0.7885	0.8521
	ISD-SB [42]	0.7456	0.7506	0.6442	0.7363	0.5994	0.6403	0.7166	0.6941	0.8071	0.7059	0.7035	0.6928	0.5688	0.7126	0.6737	0.6272	0.6887
	JSM [43]	0.8615	0.8598	0.8354	0.9026	0.8530	0.8320	0.8357	0.9213	0.8988	0.8418	0.8383	0.8214	0.8380	0.7999	0.8031	0.7833	0.8454
	Aloha [44]	0.8580	0.8300	0.9118	0.8805	0.8699	0.7955	0.8270	0.9085	0.9095	0.8402	0.8217	0.8314	0.8380	0.8375	0.8055	0.7842	0.8468
	NGS [45]	0.8392	0.8230	0.7594	0.8635	0.7898	0.7351	0.8180	0.8687	0.8767	0.8174	0.8272	0.8015	0.8121	0.7967	0.7757	0.7535	0.8098
	BKSVD [46]	0.8095	0.7713	0.7912	0.7817	0.7833	0.6951	0.7599	0.7782	0.8500	0.7804	0.7741	0.7603	0.7422	0.7622	0.7327	0.6903	0.7664
	TSLRA [47]	0.8580	0.8536	0.8786	0.8928	0.8679	0.8119	0.8184	0.9029	0.9001	0.8460	0.8311	0.8228	0.8371	0.8272	0.8030	0.7826	0.8459
	IRCNN [48]	0.8622	0.8671	0.8270	0.9032	0.8454	0.8104	0.8278	0.9102	0.9036	0.8586	0.8448	0.8358	0.8154	0.8201	0.8135	0.7923	0.8461
	IDBP [49]	0.8400	0.8436	0.7629	0.8949	0.7978	0.7865	0.8203	0.9151	0.8794	0.8413	0.8316	0.8162	0.8481	0.7919	0.7999	0.7694	0.8274
	JPG-SR	**0.8760**	**0.8737**	**0.9249**	**0.9212**	**0.8780**	**0.8535**	**0.8506**	**0.9407**	**0.9195**	**0.8768**	**0.8625**	**0.8499**	**0.8745**	**0.8423**	**0.8366**	**0.8109**	**0.8745**
70%	SALSA [40]	0.8742	0.8621	0.7580	0.8838	0.7512	0.7773	0.8649	0.8746	0.8526	0.8556	0.8675	0.8369	0.8624	0.8280	0.8250	0.7917	0.8354
	BPFA [28]	0.8950	0.8661	0.8919	0.9124	0.8726	0.8269	0.8929	0.9276	0.9269	0.8856	0.8942	0.8681	0.8979	0.8591	0.8588	0.8211	0.8811
	IPPO [41]	0.9118	0.9151	0.9334	0.9356	0.9042	0.8878	0.8857	0.9538	0.9422	0.9088	0.8923	0.8876	0.9008	0.8955	0.8823	0.8581	0.9059
	ISD-SB [42]	0.8270	0.8251	0.7259	0.8541	0.7131	0.7430	0.7947	0.8222	0.8647	0.7955	0.7957	0.7894	0.7143	0.7791	0.7636	0.7297	0.7836
	JSM [43]	0.9075	0.9064	0.9228	0.9377	0.8996	0.8831	0.8883	0.9581	0.9354	0.8935	0.8954	0.8827	0.8962	0.8728	0.8739	0.8519	0.9003
	Aloha [44]	0.9056	0.8797	0.9505	0.9205	0.9105	0.8557	0.8872	0.9549	0.9420	0.8934	0.8793	0.8907	0.9009	0.8946	0.8728	0.8512	0.8993
	NGS [45]	0.8903	0.8791	0.8556	0.9145	0.8607	0.8303	0.8756	0.9233	0.9145	0.8728	0.8856	0.8624	0.8794	0.8549	0.8483	0.8291	0.8735
	BKSVD [46]	0.8682	0.8509	0.8775	0.8753	0.8615	0.8013	0.8387	0.8896	0.9094	0.8579	0.8552	0.8425	0.8389	0.8400	0.8212	0.7918	0.8512
	TSLRA [47]	0.9034	0.8993	0.9298	0.9347	0.9071	0.8717	0.8774	0.9452	0.9367	0.8949	0.8862	0.8810	0.8943	0.8840	0.8679	0.8492	0.8977
	IRCNN [48]	0.9163	**0.9220**	0.9107	0.9419	0.9035	0.8837	0.8914	0.9574	0.9434	0.9140	0.9085	0.9018	0.9014	0.8930	0.8910	**0.8721**	0.9095
	IDBP [49]	0.8955	0.9023	0.8731	0.9353	0.8836	0.8652	0.8805	0.9515	0.9222	0.8943	0.8866	0.8727	0.8986	0.8592	0.8698	0.8388	0.8893
	JPG-SR	**0.9187**	0.9209	**0.9573**	**0.9494**	**0.9189**	**0.9025**	**0.9010**	**0.9708**	**0.9467**	**0.9196**	**0.9115**	**0.9037**	**0.9272**	**0.8962**	**0.8982**	0.8685	**0.9194**
60%	SALSA [40]	0.9064	0.8977	0.8191	0.9191	0.8222	0.8392	0.9030	0.9173	0.9283	0.8934	0.9036	0.8823	0.9022	0.8755	0.8754	0.8483	0.8833
	BPFA [28]	0.9281	0.9033	0.9394	0.9436	0.9125	0.8844	0.9263	0.9615	0.9498	0.9260	0.9280	0.9132	0.9358	0.9067	0.9053	0.8703	0.9209
	IPPO [41]	0.9438	0.9425	0.9598	0.9566	0.9346	0.9287	0.9219	0.9726	0.9601	0.9394	0.9290	0.9245	0.9366	0.9289	0.9226	0.8960	0.9374
	ISD-SB [42]	0.8781	0.8749	0.7969	0.9017	0.7994	0.8260	0.8533	0.8953	0.9049	0.8552	0.8587	0.8463	0.8140	0.8276	0.8297	0.8029	0.8478
	JSM [43]	0.9368	0.9327	0.9554	0.9570	0.9296	0.9195	0.9229	0.9751	0.9557	0.9286	0.9293	0.9184	0.9304	0.9148	0.9156	0.8915	0.9321
	Aloha [44]	0.9362	0.9127	0.9697	0.9428	0.9385	0.8968	0.9210	0.9736	0.9594	0.9288	0.9171	0.9273	0.9334	0.9251	0.9150	0.8910	0.9305
	NGS [45]	0.9246	0.9119	0.9099	0.9451	0.9086	0.8842	0.9130	0.9556	0.9443	0.9115	0.9222	0.9046	0.9187	0.8996	0.8943	0.8821	0.9144
	BKSVD [46]	0.9159	0.9050	0.9324	0.9266	0.9075	0.8780	0.8943	0.9480	0.9409	0.9121	0.9059	0.8988	0.9079	0.8940	0.8886	0.8602	0.9073
	TSLRA [47]	0.9333	0.9263	0.9577	0.9531	0.9343	0.9104	0.9156	0.9666	0.9555	0.9282	0.9231	0.9173	0.9281	0.9201	0.9097	0.8900	0.9293
	IRCNN [48]	**0.9479**	**0.9474**	0.9463	0.9597	0.9343	0.9283	0.9302	0.9777	0.9621	0.9460	**0.9413**	**0.9352**	0.9419	**0.9310**	0.9327	**0.9091**	0.9419
	IDBP [49]	0.9273	0.9319	0.9194	0.9519	0.9147	0.9153	0.9155	0.9708	0.9441	0.9264	0.9239	0.9162	0.9309	0.9018	0.9102	0.8824	0.9239
	JPG-SR	0.9457	0.9431	**0.9715**	**0.9641**	**0.9421**	**0.9348**	**0.9326**	**0.9829**	**0.9625**	**0.9464**	0.9406	0.9350	**0.9519**	0.9283	**0.9327**	0.9037	**0.9449**
50%	SALSA [40]	0.9344	0.9243	0.8651	0.9432	0.8705	0.8880	0.9304	0.9444	0.9474	0.9236	0.9335	0.9173	0.9310	0.9080	0.9108	0.8876	0.9162
	BPFA [28]	0.9505	0.9312	0.9633	0.9617	0.9390	0.9226	0.9483	0.9795	0.9658	0.9516	0.9510	0.9432	0.9572	0.9344	0.9346	0.9041	0.9461
	IPPO [41]	0.9611	0.9606	0.9749	0.9697	0.9550	0.9540	0.9463	0.9832	0.9723	0.9599	0.9531	0.9485	0.9586	0.9511	0.9477	0.9262	0.9576
	ISD-SB [42]	0.9138	0.9077	0.8562	0.9364	0.8658	0.8792	0.8981	0.9355	0.9298	0.8989	0.9002	0.8884	0.8796	0.8700	0.8798	0.8536	0.8933
	JSM [43]	0.9577	0.9537	0.9741	0.9695	0.9502	0.9459	0.9460	0.9846	0.9707	0.9528	0.9518	0.9440	0.9542	0.9437	0.9433	0.9229	0.9541
	Aloha [44]	0.9547	0.9371	**0.9815**	0.9580	0.9555	0.9244	0.9465	0.9850	0.9719	0.9525	0.9418	0.9497	0.9554	0.9477	0.9420	0.9200	0.9515
	NGS [45]	0.9464	0.9386	0.9469	0.9630	0.9376	0.9190	0.9393	0.9734	0.9625	0.9408	0.9464	0.9338	0.9450	0.9317	0.9281	0.9166	0.9418
	BKSVD [46]	0.9384	0.9302	0.9561	0.9487	0.9317	0.9146	0.9271	0.9713	0.9576	0.9409	0.9341	0.9294	0.9397	0.9223	0.9232	0.8972	0.9352
	TSLRA [47]	0.9542	0.9480	0.9743	0.9672	0.9536	0.9395	0.9409	0.9803	0.9702	0.9518	0.9490	0.9433	0.9532	0.9448	0.9397	0.9220	0.9520
	IRCNN [48]	**0.9660**	**0.9637**	0.9678	0.9715	0.9541	0.9538	**0.9535**	0.9882	**0.9741**	**0.9661**	**0.9615**	**0.9567**	0.9647	**0.9546**	**0.9564**	**0.9361**	0.9618
	IDBP [49]	0.9522	0.9517	0.9481	0.9669	0.9409	0.9491	0.9421	0.9828	0.9621	0.9508	0.9484	0.9404	0.9531	0.9320	0.9371	0.9148	0.9483
	JPG-SR	0.9638	0.9627	0.9812	**0.9737**	**0.9585**	**0.9595**	0.9531	**0.9896**	0.9733	0.9646	0.9594	0.9558	**0.9698**	0.9515	0.9553	0.9311	**0.9627**
Text Inlayed	SALSA [40]	0.9567	0.9546	0.9312	0.9630	0.9136	0.9322	0.9597	0.9562	0.9612	0.9547	0.9588	0.9486	0.9552	0.9449	0.9435	0.9338	0.9480
	BPFA [28]	0.9633	0.9605	0.9658	0.9695	0.9555	0.9482	0.9616	0.9721	0.9688	0.9660	0.9630	0.9583	0.9635	0.9552	0.9508	0.9363	0.9599
	IPPO [41]	0.9749	**0.9778**	0.9841	**0.9850**	0.9764	0.9751	0.9698	0.9892	**0.9811**	**0.9771**	0.9755	0.9699	0.9707	**0.9712**	0.9676	**0.9661**	0.9757
	ISD-SB [42]	0.9505	0.9502	0.9336	0.9586	0.9195	0.9358	0.9485	0.9501	0.9549	0.9456	0.9475	0.9384	0.9233	0.9373	0.9347	0.9233	0.9407
	JSM [43]	0.9735	0.9727	0.9828	0.9836	0.9747	0.9729	0.9689	0.9892	0.9792	0.9715	0.9732	0.9660	0.9677	0.9672	0.9637	0.9637	0.9732
	Aloha [44]	0.9620	0.9530	0.9858	0.9653	0.9723	0.9492	0.9559	0.9853	0.9755	0.9630	0.9528	0.9638	0.9619	0.9612	0.9516	0.9483	0.9629
	NGS [45]	0.9635	0.9527	0.9633	0.9759	0.9412	0.9523	0.9558	0.9558	0.9654	0.9551	0.9633	0.9511	0.9627	0.9525	0.9536	0.9508	0.9572
	BKSVD [46]	0.9538	0.9474	0.9639	0.9463	0.9479	0.9423	0.9462	0.9551	0.9584	0.9524	0.9479	0.9477	0.9371	0.9416	0.9410	0.9324	0.9476
	TSLRA [47]	0.9696	0.9690	0.9829	0.9801	0.9725	0.9691	0.9632	0.9843	0.9695	0.9696	0.9697	0.9616	0.9653	0.9654	0.9596	0.9612	0.9695
	IRCNN [48]	0.9700	0.9657	0.9700	0.9764	0.9620	0.9582	0.9625	0.9880	0.9747	0.9709	0.9685	0.9659	0.9647	0.9626	0.9597	0.9585	0.9674
	IDBP [49]	0.9716	0.9732	0.9714	0.9828	0.9638	0.9637	0.9682	0.9873	0.9778	0.9733	0.9703	0.9672	0.9686	0.9649	0.9640	0.9614	0.9706
	JPG-SR	**0.9757**	0.9771	**0.9870**	0.9846	**0.9773**	**0.9770**	**0.9703**	**0.9911**	0.9798	0.9760	**0.9765**	**0.9714**	**0.9738**	0.9692	**0.9679**	0.9654	**0.9763**

Moreover, SSR-QC, TGV-SH and COGL are the recently proposed deblocking methods that deliver state-of-the-art results. In particular, TGV-SH can effectively preserve the image texture. For comparison, sixteen standard test images are firstly encoded by a JPEG coder with different QFs, and then decoded using the standard JPEG decoder. Following this, these JPEG-compressed images are used to test different image deblocking methods. The parameters of our proposed JPG-SR based image

TABLE III

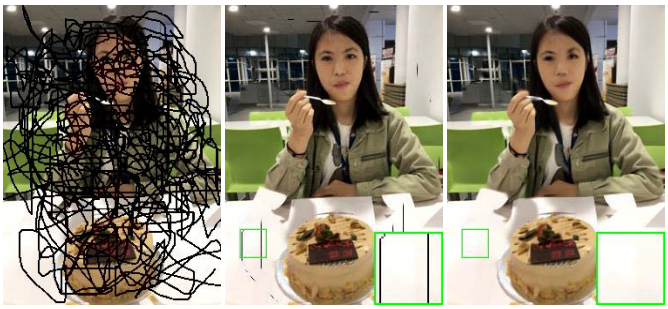
PSNR (dB) COMPARISON OF JPEG, KSVD [2], FOE [35], BM3D [23], SA-DCT [34], PC-LRM [59], ANCE [60], DICTV [53], WNNM [61], SSR-QC [33], TGV-SH [62], COGL [63] AND JPG-SR FOR IMAGE DEBLOCKING

	Quality Factor (QF) = 1												
Images	**JPEG**	**KSVD**	**FoE**	**BM3D**	**SA-DCT**	**PC-LRM**	**ANCE**	**DicTV**	**WNNM**	**SSR-QC**	**TGV-SH**	**COGL**	**JPG-SR**
Airplane	22.01	23.02	22.52	22.97	23.27	23.33	23.25	17.04	23.31	23.22	22.67	23.20	**23.39**
Barbara	21.92	22.95	22.49	22.97	23.30	23.59	23.31	16.38	23.45	**23.79**	23.21	22.88	23.76
boats	22.84	23.65	23.43	24.02	24.27	24.49	24.19	17.20	24.45	24.47	23.49	23.82	**24.53**
C. Man	22.27	23.37	22.67	23.17	23.37	23.44	23.29	16.76	23.51	23.49	22.88	23.22	**23.55**
Couple	22.04	22.76	22.51	22.83	23.11	23.11	23.11	17.72	23.10	23.11	22.66	22.94	**23.16**
Elaine	23.43	24.92	24.22	24.81	25.46	25.46	25.29	17.51	25.45	**25.53**	24.98	24.74	25.47
Fence	21.63	22.23	22.11	22.56	22.80	23.17	22.91	16.27	23.17	23.19	22.36	22.57	**23.27**
F. Print	21.63	22.23	22.11	22.56	22.80	23.17	22.91	16.27	23.17	23.19	22.36	22.57	**23.27**
Foreman	24.61	25.96	25.36	25.77	26.38	26.69	26.35	18.76	26.65	**26.85**	25.50	26.06	26.76
House	25.00	26.13	25.63	26.23	26.61	26.94	26.55	17.29	26.97	27.00	26.18	26.16	**27.10**
Lena	22.68	24.01	23.34	23.85	24.32	24.44	24.33	17.15	24.40	24.61	23.91	24.11	**24.68**
Lin	24.23	25.63	24.83	25.23	25.73	25.85	25.74	19.62	25.82	25.55	25.50	25.82	**25.99**
Miss	24.57	25.09	25.22	25.58	26.27	26.38	26.15	18.24	26.26	26.30	25.01	25.67	**26.61**
Pentagon	20.77	21.41	21.22	21.68	21.80	21.94	21.81	18.04	21.92	**22.00**	21.56	21.69	21.99
Plants	23.72	25.09	24.42	24.81	25.56	25.58	25.42	17.09	25.50	25.66	25.36	25.42	**25.83**
Straw	18.72	19.26	19.08	19.56	19.54	19.71	19.65	14.75	19.63	**19.86**	19.67	19.52	19.71
Average	22.42	23.46	22.99	23.48	23.83	24.02	23.84	17.14	23.99	24.07	23.43	23.59	**24.14**

	Quality Factor (QF) = 20												
Images	**JPEG**	**KSVD**	**FoE**	**BM3D**	**SA-DCT**	**PC-LRM**	**ANCE**	**DicTV**	**WNNM**	**SSR-QC**	**TGV-SH**	**COGL**	**JPG-SR**
Airplane	30.11	31.01	30.91	30.96	31.10	31.17	31.09	31.03	31.20	31.20	30.53	31.28	**31.31**
Barbara	29.36	30.60	30.10	30.70	30.36	31.13	31.27	30.67	31.09	31.86	29.96	30.76	**32.58**
boats	30.82	31.77	31.68	32.08	31.94	32.23	32.13	31.66	32.26	32.44	31.50	32.21	**32.48**
C. Man	28.59	29.11	29.11	29.26	29.40	29.40	29.50	29.07	29.38	29.58	29.53	**29.97**	29.80
Couple	29.38	29.95	30.03	30.20	30.17	30.26	30.18	29.73	30.27	30.23	29.66	30.28	**30.30**
Elaine	32.66	33.50	33.73	34.19	34.23	34.30	34.26	32.89	34.27	34.40	33.80	**34.53**	34.51
Fence	28.51	29.38	29.18	29.47	29.38	29.67	29.71	29.16	29.68	29.95	28.85	29.68	**30.33**
F. Print	28.51	29.38	29.18	29.47	29.38	29.67	29.71	29.16	29.68	29.95	28.85	29.68	**30.33**
Foreman	33.56	34.58	34.51	35.06	35.19	35.07	35.06	34.09	35.02	35.50	34.97	35.43	**35.63**
House	33.02	34.10	33.83	34.29	34.12	34.41	34.30	33.42	34.41	**34.76**	33.83	34.61	34.66
Lena	30.68	31.59	31.46	31.83	31.86	32.00	32.01	31.46	32.00	**32.28**	31.60	32.27	32.25
Lin	32.73	33.39	33.47	33.75	33.72	33.84	33.83	32.68	33.86	33.93	33.45	33.94	**34.02**
Miss	33.46	34.39	34.42	34.50	34.75	34.44	34.60	33.59	34.49	34.59	34.29	**35.20**	35.07
Pentagon	27.65	28.27	28.13	28.46	28.32	28.60	28.48	28.10	28.61	28.69	28.16	28.66	**28.78**
Plants	32.21	33.08	33.05	33.67	33.34	33.80	33.65	32.94	33.82	34.29	33.38	34.01	**34.39**
Straw	25.74	26.70	26.16	26.46	26.37	26.81	26.84	26.72	26.82	26.99	26.51	26.85	**27.28**
Average	30.30	31.17	31.04	31.37	31.32	31.53	31.53	30.91	31.54	31.76	31.01	31.67	**31.93**

	Quality Factor (QF) = 30												
Images	**JPEG**	**KSVD**	**FoE**	**BM3D**	**SA-DCT**	**PC-LRM**	**ANCE**	**DicTV**	**WNNM**	**SSR-QC**	**TGV-SH**	**COGL**	**JPG-SR**
Airplane	31.52	32.28	32.26	32.31	32.42	32.54	32.48	32.20	32.55	32.68	31.93	32.70	**32.70**
Barbara	31.24	32.38	31.89	32.46	32.10	32.87	33.10	32.36	32.83	33.49	31.46	32.53	**34.30**
boats	32.32	33.22	33.12	33.48	33.31	33.64	33.56	32.73	33.68	33.88	32.98	33.55	**33.96**
C. Man	29.94	30.44	30.42	30.58	30.70	30.71	30.86	30.34	30.68	30.85	30.88	**31.25**	31.16
Couple	30.78	31.31	31.39	31.53	31.49	31.62	31.63	30.86	31.64	31.69	31.10	31.73	**31.74**
Elaine	34.16	35.02	35.11	35.55	35.59	35.67	35.73	32.76	35.62	35.86	35.24	35.92	**35.98**
Fence	30.09	30.87	30.70	30.94	30.87	31.15	31.16	30.40	31.16	31.40	30.31	31.26	**31.69**
F. Print	27.80	28.61	28.25	28.37	28.17	28.70	28.81	28.51	28.73	28.84	27.48	28.36	**28.93**
Foreman	34.87	35.75	35.63	36.22	36.26	36.21	36.31	35.07	36.16	36.81	36.31	36.69	**36.96**
House	34.20	35.02	34.85	35.23	35.09	35.29	35.39	34.19	35.26	35.73	35.07	35.72	**35.80**
Lena	32.03	32.81	32.72	33.03	33.04	33.19	33.31	32.45	33.19	33.45	32.95	33.59	**33.61**
Lin	34.09	34.66	34.71	35.02	35.00	35.11	35.06	33.63	35.12	35.22	34.68	35.20	**35.26**
Miss	34.96	35.79	35.80	35.91	36.07	35.82	36.05	32.56	35.84	36.05	35.86	**36.62**	36.48
Pentagon	28.84	29.40	29.23	29.53	29.42	29.71	29.61	28.97	29.74	29.82	29.32	29.80	**29.97**
Plants	33.67	34.45	34.39	35.03	34.72	35.18	35.04	33.90	35.18	35.62	34.94	35.43	**35.84**
Straw	27.03	27.86	27.38	27.62	27.53	27.99	28.16	27.95	28.01	28.22	27.86	28.18	**28.63**
Average	31.72	32.49	32.37	32.68	32.61	32.84	32.89	31.81	32.84	33.10	32.40	33.03	**33.31**

	Quality Factor (QF) = 40												
Images	**JPEG**	**KSVD**	**FoE**	**BM3D**	**SA-DCT**	**PC-LRM**	**ANCE**	**DicTV**	**WNNM**	**SSR-QC**	**TGV-SH**	**COGL**	**JPG-SR**
Airplane	32.58	33.28	33.21	33.30	33.37	33.51	33.52	33.00	33.51	33.73	32.97	33.70	**33.78**
Barbara	32.56	33.64	33.08	33.71	33.33	34.11	34.44	33.45	34.09	35.04	32.65	33.82	**35.42**
boats	33.42	34.34	34.14	34.54	34.38	34.74	34.66	33.52	34.76	34.93	34.13	34.63	**35.03**
C. Man	30.89	31.38	31.42	31.50	31.61	31.62	31.77	31.35	31.59	31.94	31.77	32.15	**32.09**
Couple	31.81	32.35	32.34	32.53	32.49	32.64	32.68	31.65	32.68	32.74	32.13	32.80	**32.84**
Elaine	35.18	35.92	35.86	36.42	36.43	36.56	36.64	34.86	36.51	36.75	36.16	36.80	**36.91**
Fence	31.17	31.88	31.65	31.95	31.89	32.14	32.12	31.24	32.16	32.48	31.33	32.20	**32.72**
F. Print	28.78	29.50	29.24	29.26	29.09	29.58	29.75	29.31	29.60	29.83	28.41	29.30	**29.98**
Foreman	35.73	36.53	36.29	36.94	36.96	36.93	37.13	35.65	36.86	37.59	37.23	37.54	**37.78**
House	35.07	35.66	35.60	35.95	35.81	35.90	36.15	34.68	35.89	36.48	35.81	36.43	**36.62**
Lena	33.03	33.80	33.61	33.97	33.98	34.14	34.44	33.36	34.14	34.54	33.93	34.68	**34.74**
Lin	35.01	35.55	35.52	35.89	35.87	36.01	35.94	34.29	36.01	36.10	35.62	36.07	**36.23**
Miss	35.95	36.70	36.61	36.81	36.95	36.74	37.00	33.55	36.73	36.92	36.72	**37.52**	37.42
Pentagon	29.65	30.19	30.08	30.28	30.18	30.47	30.42	29.67	30.51	30.66	30.19	30.61	**30.82**
Plants	34.63	35.36	35.19	35.92	35.66	36.11	36.05	34.51	36.08	36.63	35.97	36.35	**36.84**
Straw	27.96	28.70	28.10	28.47	28.38	28.83	29.12	28.81	28.84	29.20	28.77	29.09	**29.63**
Average	32.72	33.42	33.25	33.59	33.52	33.75	33.86	32.68	33.75	34.10	33.36	33.98	**34.30**



(a) (b) (c)

Fig. 6. Comparison of multiple irregular scratch removal using different methods. (a) Image *Lisha* containing a multiple irregular scratch denoted by a mask; (b) result of TSLRA [47]; (c) result of proposed JPG-SR.

(a) (b) (c)

Fig. 7. Comparison of multiple irregular scratch removal using different methods. (a) Image *Cai* containing a multiple irregular scratch denoted by a mask; (b) result of TSLRA [47]; (c) result of proposed JPG-SR.

deblocking algorithm are as follows. The size of each patch $\sqrt{b} \times \sqrt{b}$ is 8×8. The searching window for similar patches is $W = 25$. The searching matched patches m is set to 60 and $\epsilon = e^{-14}$. The parameters $(\eta, \omega, c, \varepsilon)$ are set to (0.04, 0.1, 1.4, 0.0031), (0.03, 0.2, 0.6, 0.0031), (0.02, 0.4, 1, 0.0027), (0.09, 0.7, 0.7, 0.0008), (0.01, 0.9, 0.4, 0.00079) and (0.07, 0.8, 0.4, 0.0008) for $QF \leq 1$, $1 < QF \leq 5$, $5 < QF \leq 10$, $10 < QF \leq 20$, $20 < QF \leq 30$ and $QF > 30$, respectively.

The PSNR and SSIM comparison results for all test images in the case of $QF = 1$, 20, 30 and 40 are shown in Table III and Table IV respectively, with the best results highlighted in bold. One can observe that the proposed JPG-SR performs competitively compared to other deblocking methods. In terms of PSNR, we can see that the proposed JPG-SR outperforms other competing methods in most cases. The average gains of the proposed JPG-SR over JPEG, KSVD, FoE, BM3D, SA-DCT, PC-LRM, ANCE, DicTV, WNNM, SSR-QC, TGV-SH and COGL methods are as much as 1.63dB, 0.78dB, 1.01dB, 0.64dB, 0.60dB, 0.38dB, 0.39dB, 2.79dB, 0.39dB, 0.16dB, 0.87dB and 0.35dB, respectively. Regarding SSIM, the proposed JPG-SR achieves 0.0391, 0.0241, 0.0269, 0.0152, 0.0123, 0.0096, 0.0101, 0.0784, 0.0108, 0.0056, 0.0145 and 0.0112 improvement on average over JPEG, KSVD, FoE, BM3D, SA-DCT, PC-LRM, ANCE, DicTV, WNNM, SSR-QC, TGV-SH and COGL methods, respectively. In particular, under the conditions of $QF = 30$ and 40, the proposed JPG-SR consistently outperforms other competing methods for all test images in terms of SSIM. The visual comparisons in the

TABLE IV

SSIM COMPARISON OF JPEG, KSVD [2], FOE [35], BM3D [23], SA-DCT [34], PC-LRM [59], ANCE [60], DICTV [53], WNNM [61], SSR-QC [33], TGV-SH [62], COGL [63] AND JPG-SR FOR IMAGE DEBLOCKING

	Quality Factor (QF) = 1												
Images	**JPEG**	**KSVD**	**FoE**	**BM3D**	**SA-DCT**	**PC-LRM**	**ANCE**	**DicTV**	**WNNM**	**SSR-QC**	**TGV-SH**	**COGL**	**JPG-SR**
Airplane	0.6601	0.7168	0.6938	0.7040	0.7298	0.7252	0.7269	0.5779	0.7220	0.7235	0.7177	0.7314	**0.7376**
Barbara	0.5284	0.5790	0.5686	0.5806	0.6086	0.6161	0.6050	0.3312	0.6098	**0.6335**	0.6161	0.5929	0.6323
boats	0.6134	0.6326	0.6454	0.6651	0.6749	0.6811	0.6691	0.4570	0.6794	0.6768	0.6449	0.6547	**0.6858**
C. Man	0.5993	0.6507	0.6184	0.6251	0.6364	0.6527	0.6441	0.5526	0.6531	0.6257	0.6653	0.6656	**0.6734**
Couple	0.5109	0.5177	0.5398	0.5432	0.5588	0.5515	0.5545	0.3637	0.5506	0.5589	0.5413	0.5444	**0.5650**
Elaine	0.5915	0.6995	0.6552	0.6671	0.7189	0.7126	0.7085	0.4611	0.7109	**0.7193**	0.7106	0.6948	0.7184
Fence	0.6240	0.5982	0.6403	0.6532	0.6604	0.6675	0.6576	0.4207	0.6670	0.6699	0.6415	0.6383	**0.6774**
F. Print	0.6007	0.5797	0.6097	0.6397	0.6134	0.6497	0.6264	0.0895	0.6434	**0.6717**	0.6529	0.5796	0.6698
Foreman	0.6789	0.7779	0.7318	0.7384	0.7846	0.7870	0.7824	0.6260	0.7819	0.7893	0.7755	0.7831	**0.7943**
House	0.7202	0.7487	0.7449	0.7563	0.7708	0.7762	0.7686	0.6106	0.7754	0.7756	0.7527	0.7577	**0.7816**
Lena	0.5980	0.6848	0.6443	0.6559	0.6916	0.6918	0.6906	0.4976	0.6860	0.6981	0.6852	0.6971	**0.7066**
Lin	0.6606	0.7452	0.6975	0.7004	0.7368	0.7448	0.7463	0.6182	0.7406	0.7213	0.7505	**0.7619**	0.7605
Miss	0.6807	0.7498	0.7240	0.7296	0.7685	0.7666	0.7696	0.5728	0.7598	0.7650	0.7435	0.7616	**0.7804**
Pentagon	0.4160	0.3953	0.4404	0.4548	0.4464	0.4506	0.4419	0.2388	0.4472	**0.4597**	0.4384	0.4222	0.4536
Plants	0.5819	0.6581	0.6287	0.6364	0.6812	0.6771	0.6756	0.4320	0.6747	0.6863	0.6758	0.6821	**0.6957**
Straw	0.3890	0.2913	0.3953	0.3931	0.3620	0.3704	0.3728	0.1206	0.3577	0.4185	**0.4270**	0.3567	0.4119
Average	0.5909	0.6266	0.6236	0.6339	0.6527	0.6575	0.6525	0.4356	0.6537	0.6621	0.6524	0.6453	**0.6715**

	Quality Factor (QF) = 20												
Images	**JPEG**	**KSVD**	**FoE**	**BM3D**	**SA-DCT**	**PC-LRM**	**ANCE**	**DicTV**	**WNNM**	**SSR-QC**	**TGV-SH**	**COGL**	**JPG-SR**
Airplane	0.8897	0.9104	0.9107	0.9133	0.9155	0.9152	0.9182	0.9120	0.9143	0.9160	0.9093	0.9217	**0.9223**
Barbara	0.8831	0.9021	0.8964	0.9130	0.9089	0.9158	0.9119	0.8981	0.9152	0.9229	0.8976	0.9036	**0.9331**
boats	0.8738	0.8860	0.8893	0.9038	0.8980	0.9032	0.9015	0.8861	0.9034	0.9095	0.8950	0.9035	**0.9133**
C. Man	0.8585	0.8659	0.8665	0.8781	0.8804	0.8766	0.8805	0.8647	0.8769	0.8809	0.8769	0.8837	**0.8846**
Couple	0.8545	0.8533	0.8624	0.8760	0.8722	0.8723	0.8728	0.8497	0.8715	0.8746	0.8609	0.8758	**0.8814**
Elaine	0.8887	0.9013	0.9097	0.9194	0.9178	0.9177	0.9189	0.8953	0.9165	0.9180	0.9146	0.9207	**0.9218**
Fence	0.8621	0.8657	0.8623	0.8789	0.8754	0.8774	0.8775	0.8603	0.8771	0.8831	0.8665	0.8752	**0.8904**
F. Print	0.9270	0.9348	0.9250	0.9351	0.9326	0.9377	0.9356	0.9305	0.9380	0.9362	0.9148	0.9342	**0.9386**
Foreman	0.8941	0.9133	0.9171	0.9235	0.9219	0.9221	0.9233	0.9094	0.9210	0.9253	0.9210	0.9249	**0.9275**
House	0.8613	0.8680	0.8677	0.8770	0.8722	0.8742	0.8784	0.8706	0.8740	0.8804	0.8743	0.8803	**0.8805**
Lena	0.8828	0.9003	0.8998	0.9140	0.9125	0.9135	0.9138	0.8984	0.9134	0.9178	0.9094	0.9157	**0.9196**
Lin	0.8973	0.9047	0.9122	0.9197	0.9172	0.9172	0.9205	0.9045	0.9166	0.9243	0.9216	0.9225	**0.9272**
Miss	0.8965	0.9128	0.9183	0.9201	0.9220	0.9189	0.9226	0.9093	0.9182	0.9210	0.9161	**0.9265**	0.9263
Pentagon	0.8264	0.8253	0.8232	0.8439	0.8380	0.8428	0.8390	0.8173	0.8422	0.8458	0.8314	0.8454	**0.8548**
Plants	0.8700	0.8850	0.8886	0.9074	0.8957	0.9047	0.9047	0.8841	0.9045	0.9138	0.9007	0.9077	**0.9181**
Straw	0.8597	0.8773	0.8527	0.8751	0.8715	0.8819	0.8778	0.8721	0.8817	0.8857	0.8745	0.8820	**0.8977**
Average	0.8766	0.8879	0.8876	0.8999	0.8970	0.8994	0.8998	0.8851	0.8990	0.9035	0.8928	0.9015	**0.9086**

	Quality Factor (QF) = 30												
Images	**JPEG**	**KSVD**	**FoE**	**BM3D**	**SA-DCT**	**PC-LRM**	**ANCE**	**DicTV**	**WNNM**	**SSR-QC**	**TGV-SH**	**COGL**	**JPG-SR**
Airplane	0.9166	0.9280	0.9281	0.9357	0.9361	0.9361	0.9377	0.9257	0.9352	0.9379	0.9299	0.9393	**0.9399**
Barbara	0.9173	0.9309	0.9253	0.9377	0.9345	0.9399	0.9387	0.9212	0.9393	0.9447	0.9250	0.9315	**0.9526**
boats	0.9025	0.9127	0.9132	0.9251	0.9196	0.9254	0.9226	0.9024	0.9258	0.9307	0.9193	0.9228	**0.9339**
C. Man	0.8828	0.8911	0.8844	0.9003	0.9011	0.8997	0.9015	0.8819	0.8992	0.9017	0.8981	0.9025	**0.9054**
Couple	0.8922	0.8902	0.8951	0.9071	0.9039	0.9043	0.9068	0.8762	0.9039	0.9086	0.8972	0.9086	**0.9125**
Elaine	0.9154	0.9240	0.9271	0.9362	0.9350	0.9345	0.9369	0.9030	0.9333	0.9359	0.9324	0.9372	**0.9388**
Fence	0.8938	0.8980	0.8891	0.9055	0.9031	0.9053	0.9043	0.8808	0.9052	0.9082	0.8943	0.9031	**0.9135**
F. Print	0.9471	0.9533	0.9434	0.9523	0.9506	0.9545	0.9534	0.9466	0.9548	0.9543	0.9377	0.9525	**0.9557**
Foreman	0.9166	0.9273	0.9296	0.9364	0.9333	0.9340	0.9367	0.9216	0.9334	0.9394	0.9357	0.9375	**0.9417**
House	0.8824	0.8805	0.8822	0.8904	0.8871	0.8860	0.8943	0.8807	0.8852	0.8959	0.8920	0.8965	**0.8985**
Lena	0.9093	0.9213	0.9190	0.9319	0.9304	0.9314	0.9321	0.9124	0.9313	0.9348	0.9275	0.9329	**0.9370**
Lin	0.9220	0.9245	0.9302	0.9387	0.9365	0.9362	0.9382	0.9181	0.9357	0.9403	0.9383	0.9381	**0.9434**
Miss	0.9176	0.9269	0.9303	0.9333	0.9338	0.9307	0.9357	0.9030	0.9302	0.9338	0.9318	0.9393	**0.9395**
Pentagon	0.8646	0.8683	0.8594	0.8784	0.8746	0.8797	0.8755	0.8458	0.8800	0.8819	0.8694	0.8810	**0.8889**
Plants	0.9022	0.9116	0.9126	0.9292	0.9215	0.9274	0.9274	0.9012	0.9269	0.9337	0.9255	0.9293	**0.9374**
Straw	0.8968	0.9106	0.8902	0.9076	0.9053	0.9139	0.9128	0.9047	0.9142	0.9179	0.9102	0.9153	**0.9270**
Average	0.9049	0.9125	0.9100	0.9216	0.9192	0.9212	0.9222	0.9016	0.9208	0.9250	0.9165	0.9230	**0.9291**

	Quality Factor (QF) = 40												
Images	**JPEG**	**KSVD**	**FoE**	**BM3D**	**SA-DCT**	**PC-LRM**	**ANCE**	**DicTV**	**WNNM**	**SSR-QC**	**TGV-SH**	**COGL**	**JPG-SR**
Airplane	0.9306	0.9390	0.9354	0.9445	0.9446	0.9447	0.9483	0.9339	0.9433	0.9470	0.9419	0.9495	**0.9510**
Barbara	0.9353	0.9461	0.9359	0.9507	0.9481	0.9525	0.9524	0.9334	0.9522	0.9572	0.9401	0.9463	**0.9611**
boats	0.9196	0.9299	0.9255	0.9384	0.9342	0.9394	0.9368	0.9149	0.9394	0.9432	0.9343	0.9361	**0.9454**
C. Man	0.8985	0.9057	0.9001	0.9132	0.9131	0.9123	0.9141	0.8951	0.9117	0.9144	0.9108	0.9151	**0.9173**
Couple	0.9130	0.9115	0.9078	0.9249	0.9232	0.9230	0.9255	0.8923	0.9230	0.9255	0.9162	0.9266	**0.9299**
Elaine	0.9291	0.9342	0.9323	0.9442	0.9431	0.9428	0.9452	0.9238	0.9419	0.9435	0.9412	0.9453	**0.9469**
Fence	0.9124	0.9174	0.9035	0.9220	0.9204	0.9220	0.9208	0.8947	0.9218	0.9236	0.9122	0.9195	**0.9287**
F. Print	0.9573	0.9621	0.9550	0.9609	0.9599	0.9629	0.9623	0.9546	0.9631	0.9632	0.9496	0.9616	**0.9646**
Foreman	0.9283	0.9359	0.9340	0.9435	0.9405	0.9411	0.9449	0.9265	0.9404	0.9467	0.9437	0.9455	**0.9489**
House	0.8981	0.8910	0.8954	0.9015	0.8988	0.8945	0.9056	0.8876	0.8943	0.9072	0.9033	0.9080	**0.9104**
Lena	0.9247	0.9340	0.9259	0.9421	0.9414	0.9417	0.9437	0.9217	0.9415	0.9450	0.9389	0.9441	**0.9474**
Lin	0.9350	0.9351	0.9354	0.9481	0.9462	0.9456	0.9473	0.9251	0.9451	0.9491	0.9477	0.9474	**0.9521**
Miss	0.9305	0.9365	0.9363	0.9418	0.9420	0.9395	0.9444	0.9188	0.9384	0.9421	0.9411	0.9471	**0.9475**
Pentagon	0.8857	0.8915	0.8850	0.8974	0.8950	0.8993	0.8962	0.8649	0.8999	0.9015	0.8913	0.9006	**0.9068**
Plants	0.9190	0.9265	0.9223	0.9410	0.9350	0.9399	0.9402	0.9109	0.9389	0.9452	0.9384	0.9413	**0.9487**
Straw	0.9163	0.9271	0.9030	0.9244	0.9225	0.9297	0.9308	0.9212	0.9298	0.9351	0.9274	0.9316	**0.9421**
Average	0.9208	0.9265	0.9208	0.9337	0.9317	0.9332	0.9349	0.9137	0.9328	0.9368	0.9299	0.9353	**0.9405**

Fig. 8. Visual comparison results of image *Barbara* at $QF = 20$. (a) Original image; (b) JPEG compressed image (PSNR = 29.36dB, SSIM = 0.8831); (c) KSVD [2] (PSNR = 30.60dB, SSIM = 0.9021); (d) FoE [35] (PSNR = 30.10dB, SSIM = 0.8964); (e) BM3D [23] (PSNR = 30.70dB, SSIM = 0.9130); (f) SA-DCT [34] (PSNR = 30.36dB, SSIM = 0.9089); (g) PC-LRM [59] (PSNR = 31.13dB, SSIM = 0.9158); (h) ANCE [60] (PSNR = 31.27dB, SSIM = 0.9119); (i) DicTV [53] (PSNR = 30.67dB, SSIM = 0.8981); (j) WNNM [61] (PSNR = 31.09dB, SSIM = 0.9152); (k) SSR-QC [33] (PSNR = 31.86dB, SSIM = 0.9229); (l) TGV-SH [62] (PSNR = 29.96dB, SSIM = 0.8976); (m) COGL [63] (PSNR = 30.76dB, SSIM = 0.9036); (n) JPG-SR (PSNR = **32.58dB**, SSIM = **0.9331**).

case of $QF = 20$ for images *Barbara* and *C. Man* are shown in Fig. 8 and Fig. 9, respectively. It can be seen that the blocking artifacts are obvious in the image decoded directly by the standard JPEG. FoE and DicTV methods can only suppress the blocking artifacts partially, but many blocking artifacts are still existing in the restored images. KSVD, BM3D, SA-DCT, PC-LRM, ANCE, WNNM and SSR-QC methods obtain better results than FoE and DicTV. However, they often generate zigzag artifacts and blur effects. SSR-QC and COGL[3] methods are effective at capturing image textures and edges, but they often produce over-smooth effects. Our proposed JPG-SR method not only removes blocking artifacts across the image, but also preserves sharp edges and fine details effectively.

Recently, deep learning based techniques for image deblocking have drawn considerable attention owing to its impressive performance. We compare the proposed JPG-SR with the AR-CNN [65] method, since it is the baseline of convolutional neural network (CNN) based image deblocking algorithms. As shown in Table V, it can be seen that our proposed method is comparable with AR-CNN. Under different QFs,

[3]Compared with our proposed JPG-SR method, COGL has a higher PSNR result for image *C. Man*. But COGL method markedly produces the over-smooth effect in Fig. 9 (m).

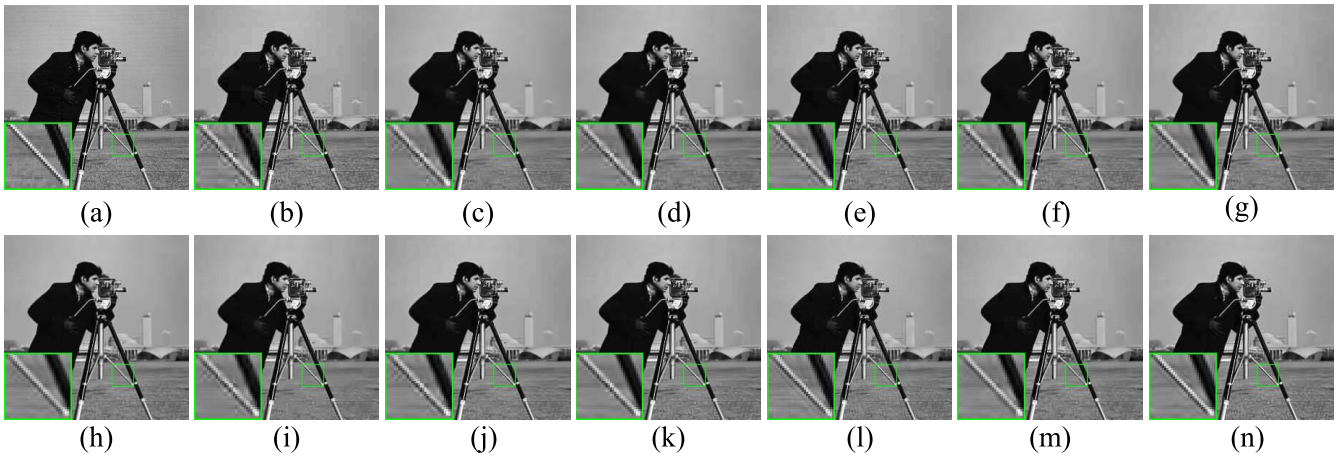


Fig. 9. Visual comparison results of image *C. Man* at $QF = 20$. (a) Original image; (b) JPEG compressed image (PSNR = 28.59dB, SSIM = 0.8585); (c) KSVD [2] (PSNR = 29.11dB, SSIM = 0.8659); (d) FoE [35] (PSNR = 29.11dB, SSIM = 0.8665); (e) BM3D [23] (PSNR = 29.26dB, SSIM = 0.8781); (f) SA-DCT [34] (PSNR = 29.40dB, SSIM = 0.8804); (g) PC-LRM [59] (PSNR = 29.40dB, SSIM = 0.8766); (h) ANCE [60] (PSNR = 29.50dB, SSIM = 0.8805); (i) DicTV [53] (PSNR = 29.07dB, SSIM = 0.8647); (j) WNNM [61] (PSNR = 29.38dB, SSIM = 0.8769); (k) SSR-QC [33] (PSNR = 29.58dB, SSIM = 0.8809); (l) TGV-SH [62] (PSNR = 29.53dB, SSIM = 0.8769); (m) COGL [63] (PSNR = **29.97dB**, SSIM = 0.8837); (n) JPG-SR (PSNR = 29.80dB, SSIM = **0.8846**).

TABLE V

PSNR IN dB (TOP ENTRY IN EACH CELL) AND SSIM (BOTTOM ENTRY) RESULTS OF AR-CNN [65] AND THE PROPOSED JPG-SR METHODS FOR IMAGE DEBLOCKING

Images	QF = 10		QF = 20		QF = 30		QF = 40	
	AR-CNN	ours	AR-CNN	ours	AR-CNN	ours	AR-CNN	ours
Airplane	**29.15**	28.67	**31.71**	31.31	**33.19**	32.70	**34.20**	33.78
	0.8841	0.8751	**0.9242**	0.9223	**0.9426**	0.9399	**0.9520**	0.9510
Barbara	27.70	**28.69**	31.00	**32.58**	32.75	**34.30**	33.95	**35.42**
	0.8306	**0.8642**	0.9105	**0.9331**	0.9373	**0.9526**	0.9499	**0.9611**
boat	**29.74**	29.58	32.37	**32.48**	33.86	**33.96**	34.84	**35.03**
	0.8545	**0.8573**	0.9047	**0.9133**	0.9283	**0.9339**	0.9398	**0.9454**
C. Man	**27.63**	27.47	29.39	**29.80**	30.76	**31.16**	31.49	**32.09**
	0.8305	**0.8375**	0.8684	**0.8846**	0.8922	**0.9054**	0.9032	**0.9173**
Couple	**27.98**	27.77	30.28	**30.30**	**31.86**	31.74	32.79	**32.84**
	0.8033	**0.8052**	0.8744	**0.8814**	0.9107	**0.9125**	0.9268	**0.9299**
Elaine	**31.92**	31.66	34.46	**34.51**	35.98	**35.98**	36.86	**36.91**
	0.8801	0.8792	0.9177	**0.9218**	0.9368	**0.9388**	0.9453	**0.9469**
Fence	27.05	**27.31**	29.65	**30.33**	31.14	**31.69**	32.09	**32.72**
	0.8161	**0.8238**	0.8790	**0.8904**	0.9060	**0.9135**	0.9223	**0.9287**
F. Print	24.80	**24.97**	27.14	**27.55**	28.57	**28.93**	29.54	**29.98**
	0.8851	**0.8905**	0.9349	**0.9386**	0.9535	**0.9557**	0.9624	**0.9646**
Foreman	**33.38**	33.14	**35.66**	35.63	**37.09**	36.96	**37.87**	37.78
	0.8995	0.8969	0.9251	**0.9275**	0.9408	**0.9417**	0.9481	**0.9489**
House	32.53	**32.61**	34.52	**34.66**	35.62	**35.80**	36.32	**36.62**
	0.8546	**0.8557**	0.8795	**0.8805**	0.8957	**0.8985**	0.9060	**0.9104**
Lena	**30.05**	29.68	**32.43**	32.25	**33.92**	33.61	**34.87**	34.74
	0.8711	0.8697	0.9151	**0.9196**	0.9352	**0.9370**	0.9450	**0.9474**
Lin	30.37	**31.50**	31.46	**34.02**	32.52	**35.26**	33.16	**36.23**
	0.8484	**0.8863**	0.8713	**0.9272**	0.8890	**0.9434**	0.8984	**0.9521**
Miss	32.65	**32.74**	**35.20**	35.07	**36.46**	36.48	37.11	**37.42**
	0.8944	0.8919	0.9241	**0.9263**	0.9387	**0.9395**	0.9470	**0.9475**
Pentagon	26.49	**26.64**	28.60	**28.78**	29.74	**29.97**	30.69	**30.82**
	0.7631	**0.7705**	0.8479	**0.8548**	0.8833	**0.8889**	0.9035	**0.9068**
Plants	**31.66**	31.43	34.32	**34.39**	**35.86**	35.84	36.73	**36.84**
	0.8633	**0.8633**	0.9110	**0.9181**	0.9337	**0.9374**	0.9444	**0.9487**
Straw	24.74	**24.81**	26.91	**27.28**	28.31	**28.63**	29.17	**29.63**
	0.8087	**0.8157**	0.8834	**0.8977**	0.9180	**0.9270**	0.9326	**0.9421**
Average	29.24	**29.29**	31.57	**31.93**	32.98	**33.31**	33.86	**34.30**
	0.8492	**0.8552**	0.8982	**0.9086**	0.9214	**0.9291**	0.9329	**0.9405**

the proposed JPG-SR obtains better results than AR-CNN in the majority of cases. The average PSNR and SSIM gain of our proposed method over AR-CNN is up to 0.30dB and 0.0079, respectively. The visual comparison results of image *Fence* and *Straw* with $QF = 20$ are presented in Fig. 10 and Fig. 11, respectively. One can observe that AR-CNN still

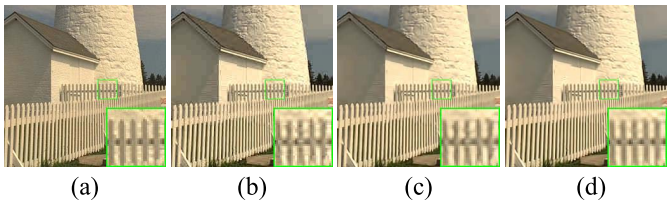


Fig. 10. Visual comparison results of image *Fence* at $QF = 20$. (a) Original image; (b) JPEG compressed image (PSNR = 28.51dB, SSIM = 0.8621); (c) AR-CNN [65] (PSNR = 29.65dB, SSIM = 0.8790); (d) JPG-SR (PSNR = **30.33dB**, SSIM = **0.8904**).

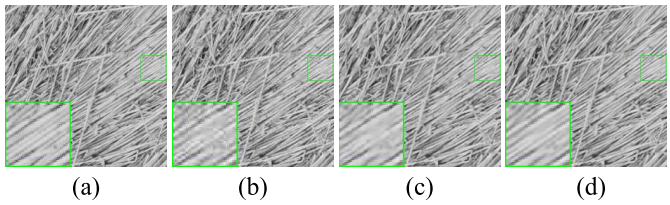


Fig. 11. Visual comparison results of image *Straw* at $QF = 20$. (a) Original image; (b) JPEG compressed image (PSNR = 25.74dB, SSIM = 0.8597); (c) AR-CNN [65] (PSNR = 26.91dB, SSIM = 0.8834); (d) JPG-SR (PSNR = **27.28dB**, SSIM = **0.8977**).

suffers from undesirable artifacts, while the proposed JPG-SR algorithm eliminates the blocking artifacts more effectively than AR-CNN. These results further verify the superiority of the proposed algorithm.

C. Suitable Setting of the ADMM Balance Factor

The ADMM balance factor μ plays an important role in our proposed JPG-SR based image restoration algorithm. In this subsection, we discuss how to set the ADMM balance factor μ for the proposed algorithm. We firstly fix other parameters. For image inpainting, the effects of μ values on average algorithm performance for five test images are shown in Fig. 12 (a)-(e). One can observe that the best performance of each case (80%,

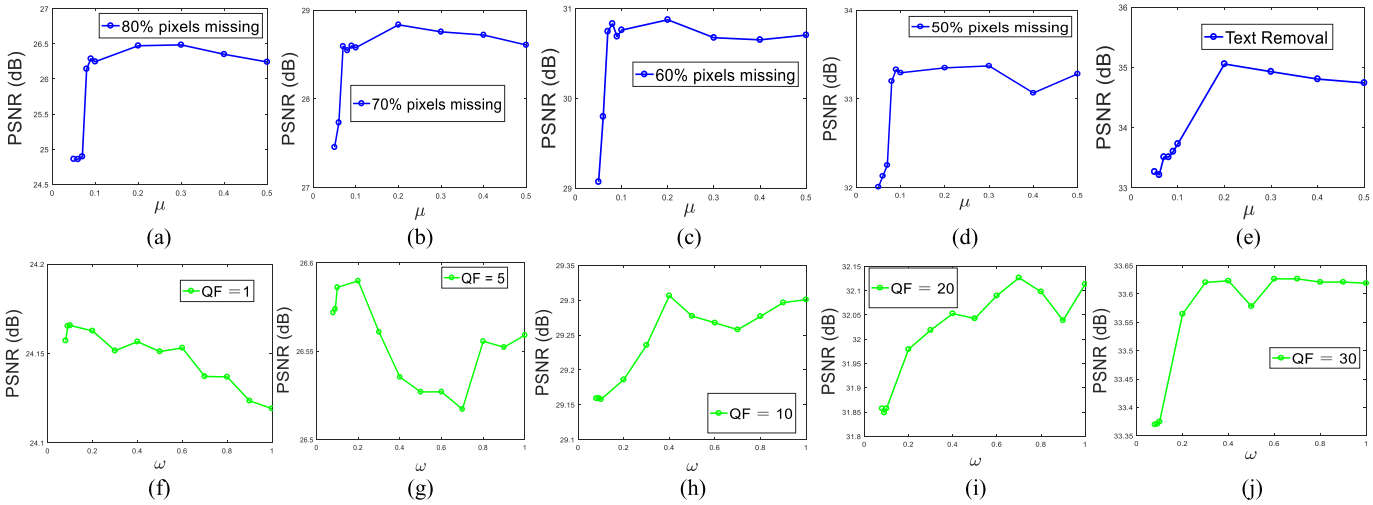


Fig. 12. Setting the different ADMM balance factor for the influence of image restoration tasks. Top row: the influence of different μ upon inpainting results under different pixels missing cases. Bottom row: the influence of different ω upon deblocked results under different QF cases.

TABLE VI

AVERAGE PSNR (dB) (TOP ENTRY) AND SSIM (BOTTOM ENTRY) COMPARISON OF PSR, GSR AND THE PROPOSED JPG-SR FOR IMAGE RESTORATION TASKS ON SET 11 DATASET [66]

Image Deblocking						
QF	**5**	**10**	**20**	**30**	**40**	**Average**
PSR	25.90	28.48	30.78	32.08	33.03	30.06
	0.7769	0.8508	0.8989	0.9188	0.9307	0.8752
GSR	26.13	28.90	31.50	32.64	33.73	30.58
	0.7883	0.8646	0.9115	0.9269	0.9387	0.8860
JPG-SR	**26.28**	**28.93**	**31.65**	**33.06**	**34.07**	**30.80**
	0.8005	**0.8713**	**0.9174**	**0.9347**	**0.9443**	**0.8936**
Image Inpainting						
pixel missing	**80%**	**70%**	**60%**	**50%**	**text removal**	**Average**
PSR	27.07	29.45	32.13	34.33	32.08	31.01
	0.8645	0.8973	0.9464	0.9655	0.9188	0.9185
GSR	**28.44**	30.72	32.51	34.15	35.80	32.32
	0.8807	0.9174	0.9383	0.9521	0.9728	0.9323
JPG-SR	28.35	**31.03**	**32.92**	**34.90**	**36.20**	**32.68**
	0.8935	**0.9316**	**0.9518**	**0.9656**	**0.9784**	**0.9442**

70%, 60%, 50% pixels missing and text removal) is usually achieved with μ in the range [0.1, 0.3]. Therefore, in this paper, μ is set 0.2 for all image inpainting tasks. μ is set by Eq. (33) in image deblocking task, which is then dependent on the scaling factor ω. We thus discuss the influence of scaling factor ω for the the proposed algorithm. The effects of ω values on average algorithm performance for five test images are shown in Fig. 12 (f)-(j). It can be seen that the best deblocking performance of the proposed JPG-SR algorithm is obtained with ω in the range [0.09, 0.2], [0.1, 0.3], [0.3, 0.5], [0.6, 0.8] and [0.3, 0.5] with $QF = 1$, 5, 10, 20 and 30, respectively. Therefore, in image deblocking, we set $\omega = 0.1$, 0.2, 0.4, 0.7 and 0.4 for $QF = 1$, 5, 10, 20 and 30, respectively.

D. Ablation Study

In this subsection, we conduct an ablation study of the proposed JPG-SR model, by prohibiting the PSR model and GSR model in the JPG-SR image restoration algorithm, respectively. The variants are reduced to PSR-based image restoration algorithm [2] and GSR-based image restoration algorithm. Table VI shows the average PSNR and SSIM results over all test images from Set 11 (11 images) dataset [66], achieved by our proposed JPG-SR based image restoration algorithm, as well as its two variants. For image deblocking, the proposed JPG-SR obtains the best performance with comparison to other competing methods. For image inpainting, on average, our proposed JPG-SR also achieves the best performance with comparison to other competing methods. Under the condition of 80% pixels missing, though the PSNR result of the proposed JPG-SR is lower than GSR, the SSIM result of our proposed JPG-SR is significantly better than GSR. This phenomenon has been explained in [58]. Therefore, this experiment shows that both the PSR and GSR models contribute significantly to the success of the proposed JPG-SR model. This experiment also further demonstrates that the proposed model is feasible. Furthermore, in our proposed model, we can use the powerful deep prior models [48], [49] to replace the PSR model, because most of deep models focused on exploiting image local properties. Then, the proposed model can provide a mutual complementary between feature space and image space.

E. Convergence

Since the proposed model is non-convex, it is difficult to give its theoretical proof for global convergence. Hereby, we provide the empirical evidence to illustrate the good convergence behavior of our proposed model in Fig. 13. It shows the curves of the PSNR values versus the iteration numbers for image deblocking with $QF = 20$ and image inpainting with 80% pixels missing, respectively. It can be seen that with the increase of iteration numbers, the PSNR curves of the reconstructed images gradually increase and eventually become flat and stable. Clearly, the proposed JPG-SR algorithm enjoys a good convergence performance.

F. Running Time

In this subsection, we report the running time of the proposed JPG-SR based image restoration algorithm with

TABLE VII
RUNNING TIME (S) OF DIFFERENT IMAGE RESTORATION METHODS

Image Deblocking												
Methods	**KSVD**	**FoE**	**BM3D**	**SA-DCT**	**PC-LRM**	**ANCE**	**DicTV**	**WNNM**	**SSR-QC**	**TGV-SH**	**COGL**	**JPG-SR**
Time	25.48	36.46	0.74	3.47	14.63	57.21	16.59	63.84	20.65	118.74	129.20	105.07
Image Inpainting												
Methods	**SALSA**	**BPFA**	**IPPO**	**ISD-SB**	**JSM**	**Aloha**	**NGS**	**BKSVD**	**TSLRA**	**IRCNN**	**IDBP**	**JPG-SR**
Time	3.46	1011.64	124.53	19.09	245.00	651.69	99.42	12156.18	691.55	12.72	23.87	1001.63

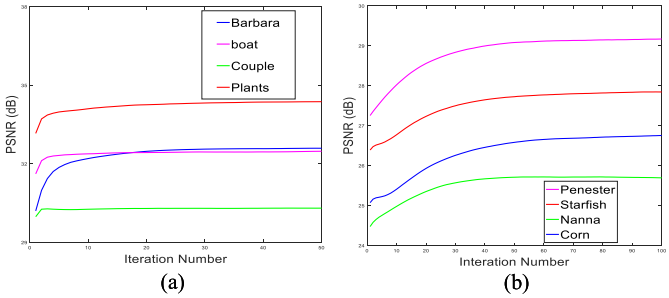


Fig. 13. Convergence behavior. (a) PSNR results versus iteration number for image deblocking with $QF = 20$. (b) PSNR results versus iteration number for image inpainting with 80% pixels missing.

comparison to all competing methods. All experiments are conducted on an Intel (R) Core (TM) i5-4570 with 3.20Hz CPU PC under the Matlab 2016b environment. We use a widely used image *House* as an example for all competing methods. For image deblocking, the JPEG-compressed image is generated by using the image *House* with QF = 5. As can be seen in the upper part of Table VII, the proposed JPG-SR requires less than 2 minutes for an image, which is faster than TGV-SH and COGL methods. For image inpainting, 80% pixels are missing for image *House*. According to lower part of Table VII, the proposed JPG-SR method requires about 15-17 minutes for an image, which is faster than BPFA and BKSVD methods. The proposed JPG-SR algorithm appears to require more time because the block matching and dictionary learning via SVD require a highly computational workload. We are working on using GPU hardware to accelerate the proposed algorithm since the block matching and SVD of each patch group can be performed in parallel.

VI. CONCLUSION

A new sparse representation model, dubbed joint patch-group based sparse representation has been proposed in this paper. Compared to existing sparse representation models, the proposed JPG-SR integrated both local sparsity and non-local self-similarity of the image. We have developed an iterative algorithm based on the ADMM framework to solve the proposed model for image restoration tasks, including image inpainting and image deblocking. Experimental results have demonstrated that the proposed algorithm is effective and outperforms many state-of-the-art methods both quantitatively and qualitatively.

ACKNOWLEDGMENT

The authors would like to thank the associate editor and anonymous reviewers for their constructive suggestions to improve the manuscript.

REFERENCES

[1] M. Elad and M. Aharon, "Image denoising via sparse and redundant representations over learned dictionaries," *IEEE Trans. Image Process.*, vol. 15, no. 12, pp. 3736–3745, Dec. 2006.
[2] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Trans. Signal Process.*, vol. 54, no. 11, pp. 4311–4322, Nov. 2006.
[3] B. Wen, S. Ravishankar, and Y. Bresler, "Structured overcomplete sparsifying transform learning with convergence guarantees and applications," *Int. J. Comput. Vis.*, vol. 114, nos. 2–3, pp. 137–167, Sep. 2015.
[4] Z. Zha, X. Yuan, B. Wen, J. Zhou, and C. Zhu, "Joint patch-group based sparse representation for image inpainting," in *Proc. The 10th Asian Conf. Mach. Learn.*, in Proceedings of Machine Learning Research, vol. 95, J. Zhu and I. Takeuchi, Eds. Beijing, China: PMLR, Nov. 2018, pp. 145–160.
[5] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman, "Non-local sparse models for image restoration," in *Proc. IEEE 12th Int. Conf. Comput. Vis.*, Sep. 2009, pp. 2272–2279.
[6] J. Zhang, D. Zhao, and W. Gao, "Group-based sparse representation for image restoration," *IEEE Trans. Image Process.*, vol. 23, no. 8, pp. 3336–3351, Aug. 2014.
[7] R. Rubinstein, T. Peleg, and M. Elad, "Analysis K-SVD: A dictionary-learning algorithm for the analysis sparse model," *IEEE Trans. Signal Process.*, vol. 61, no. 3, pp. 661–677, Feb. 2013.
[8] X. Li, H. Shen, L. Zhang, and H. Li, "Sparse-based reconstruction of missing information in remote sensing images from spectral/temporal complementary information," *ISPRS J. Photogramm. Remote Sens.*, vol. 106, pp. 1–15, Aug. 2015.
[9] X. Wei, H. Shen, and M. Kleinsteuber, "Trace quotient with sparsity priors for learning low dimensional image representations," *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, Jun. 5, 2019, doi: 10.1109/TPAMI.2019.2921031.
[10] W. Dong, G. Shi, Y. Ma, and X. Li, "Image restoration via simultaneous sparse coding: Where structured sparsity meets Gaussian scale mixture," *Int. J. Comput. Vis.*, vol. 114, nos. 2–3, pp. 217–232, Sep. 2015.
[11] Z. Zha, X. Yuan, B. Wen, J. Zhou, J. Zhang, and C. Zhu, "A benchmark for sparse coding: When group sparsity meets rank minimization," *IEEE Trans. Image Process.*, vol. 29, pp. 5094–5109, Mar. 2020.
[12] X. Li, H. Shen, L. Zhang, H. Zhang, Q. Yuan, and G. Yang, "Recovering quantitative remote sensing products contaminated by thick clouds and shadows using multitemporal dictionary learning," *IEEE Trans. Geosci. Remote Sens.*, vol. 52, no. 11, pp. 7086–7098, Nov. 2014.
[13] Y. Sun *et al.*, "Discriminative local sparse representation by robust adaptive dictionary pair learning," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Jan. 14, 2020, doi: 10.1109/TNNLS.2019.2954545.
[14] B. Wen, S. Ravishankar, L. Pfister, and Y. Bresler, "Transform learning for magnetic resonance image reconstruction: From model-based learning to building neural networks," *IEEE Signal Process. Mag.*, vol. 37, no. 1, pp. 41–53, Jan. 2020.
[15] N. Qi, Y. Shi, X. Sun, J. Wang, B. Yin, and J. Gao, "Multi-dimensional sparse models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 1, pp. 163–178, Jan. 2018.

[16] X. Li, H. Shen, H. Li, and L. Zhang, "Patch matching-based multitemporal group sparse representation for the missing information reconstruction of remote-sensing images," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 9, no. 8, pp. 3629–3641, Aug. 2016.

[17] J. Ling, Z. Chen, and F. Wu, "Class-oriented discriminative dictionary learning for image classification," *IEEE Trans. Circuits Syst. Video Technol.*, early access, May 24, 2019, doi: 10.1109/TCSVT.2019.2918852.

[18] Q. Zhang and B. Li, "Discriminative K-SVD for dictionary learning in face recognition," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 2691–2698.

[19] Budianto and D. P. K. Lun, "Robust fringe projection profilometry via sparse representation," *IEEE Trans. Image Process.*, vol. 25, no. 4, pp. 1726–1739, Apr. 2016.

[20] M. Elad and I. Yavneh, "A plurality of sparse representations is better than the sparsest one alone," *IEEE Trans. Inf. Theory*, vol. 55, no. 10, pp. 4701–4714, Oct. 2009.

[21] W. Dong, L. Zhang, G. Shi, and X. Wu, "Image deblurring and super-resolution by adaptive sparse domain selection and adaptive regularization," *IEEE Trans. Image Process.*, vol. 20, no. 7, pp. 1838–1857, Jul. 2011.

[22] A. Buades, B. Coll, and J.-M. Morel, "A non-local algorithm for image denoising," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 2, Jun. 2005, pp. 60–65.

[23] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3-D transform-domain collaborative filtering," *IEEE Trans. Image Process.*, vol. 16, no. 8, pp. 2080–2095, Aug. 2007.

[24] L. Zhang, W. Dong, D. Zhang, and G. Shi, "Two-stage image denoising by principal component analysis with local pixel grouping," *Pattern Recognit.*, vol. 43, no. 4, pp. 1531–1549, Apr. 2010.

[25] M. Li, J. Liu, Z. Xiong, X. Sun, and Z. Guo, "Marlow: A joint multiplanar autoregressive and low-rank approach for image completion," in *Proc. Eur. Conf. Comput. Vis.* Amsterdam, The Netherlands: Springer, Sep. 2016, pp. 819–834.

[26] J. M. Keller, M. R. Gray, and J. A. Givens, "A fuzzy K-nearest neighbor algorithm," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-15, no. 4, pp. 580–585, Jul. 1985.

[27] W. Dong, P. Wang, W. Yin, G. Shi, F. Wu, and X. Lu, "Denoising prior driven deep neural network for image restoration," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 10, pp. 2305–2318, Oct. 2019.

[28] M. Zhou *et al.*, "Nonparametric Bayesian dictionary learning for analysis of noisy and incomplete images," *IEEE Trans. Image Process.*, vol. 21, no. 1, pp. 130–144, Jan. 2012.

[29] Z. Xu and J. Sun, "Image inpainting by patch propagation using patch sparsity," *IEEE Trans. Image Process.*, vol. 19, no. 5, pp. 1153–1165, May 2010.

[30] Q. Zhang, Q. Yuan, C. Zeng, X. Li, and Y. Wei, "Missing data reconstruction in remote sensing image with a unified spatial–temporal–spectral deep convolutional neural network," *IEEE Trans. Geosci. Remote Sens.*, vol. 56, no. 8, pp. 4274–4288, Aug. 2018.

[31] X. Liu, X. Wu, J. Zhou, and D. Zhao, "Data-driven sparsity-based restoration of JPEG-compressed images in dual transform-pixel domain," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 5171–5178.

[32] J. Zhang, R. Xiong, C. Zhao, Y. Zhang, S. Ma, and W. Gao, "CONCOLOR: Constrained non-convex low-rank model for image deblocking," *IEEE Trans. Image Process.*, vol. 25, no. 3, pp. 1246–1259, Mar. 2016.

[33] C. Zhao, J. Zhang, S. Ma, X. Fan, Y. Zhang, and W. Gao, "Reducing image compression artifacts by structural sparse representation and quantization constraint prior," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 27, no. 10, pp. 2057–2071, Oct. 2017.

[34] A. Foi, V. Katkovnik, and K. Egiazarian, "Pointwise shape-adaptive DCT for high-quality denoising and deblocking of grayscale and color images," *IEEE Trans. Image Process.*, vol. 16, no. 5, pp. 1395–1411, May 2007.

[35] D. Sun and W.-K. Cham, "Postprocessing of low bit-rate block DCT coded images based on a fields of experts prior," *IEEE Trans. Image Process.*, vol. 16, no. 11, pp. 2743–2751, Nov. 2007.

[36] B. He, L.-Z. Liao, D. Han, and H. Yang, "A new inexact alternating directions method for monotone variational inequalities," *Math. Program.*, vol. 92, no. 1, pp. 103–118, Mar. 2002.

[37] S. Boyd, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, 2011.

[38] J. A. Tropp and A. C. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Trans. Inf. Theory*, vol. 53, no. 12, pp. 4655–4666, Dec. 2007.

[39] D. L. Donoho and Y. Tsaig, "Fast solution of ℓ_1-norm minimization problems when the solution may be sparse," *IEEE Trans. Inf. Theory*, vol. 54, no. 11, pp. 4789–4812, Nov. 2008.

[40] M. V. Afonso, J. M. Bioucas-Dias, and M. A. T. Figueiredo, "An augmented Lagrangian approach to the constrained optimization formulation of imaging inverse problems," *IEEE Trans. Image Process.*, vol. 20, no. 3, pp. 681–695, Mar. 2011.

[41] I. Ram, M. Elad, and I. Cohen, "Image processing using smooth ordering of its patches," *IEEE Trans. Image Process.*, vol. 22, no. 7, pp. 2764–2774, Jul. 2013.

[42] L. He and Y. Wang, "Iterative support detection-based split Bregman method for wavelet frame-based image inpainting," *IEEE Trans. Image Process.*, vol. 23, no. 12, pp. 5470–5485, Dec. 2014.

[43] J. Zhang, D. Zhao, R. Xiong, S. Ma, and W. Gao, "Image restoration using joint statistical modeling in a space-transform domain," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 6, pp. 915–928, Jun. 2014.

[44] K. Hwan Jin and J. Chul Ye, "Annihilating filter-based low-rank Hankel matrix approach for image inpainting," *IEEE Trans. Image Process.*, vol. 24, no. 11, pp. 3498–3511, Nov. 2015.

[45] H. Liu, R. Xiong, X. Zhang, Y. Zhang, S. Ma, and W. Gao, "Nonlocal gradient sparsity regularization for image restoration," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 27, no. 9, pp. 1909–1921, Sep. 2017.

[46] J. G. Serra, M. Testa, R. Molina, and A. K. Katsaggelos, "Bayesian K-SVD using fast variational inference," *IEEE Trans. Image Process.*, vol. 26, no. 7, pp. 3344–3359, Jul. 2017.

[47] Q. Guo, S. Gao, X. Zhang, Y. Yin, and C. Zhang, "Patch-based image inpainting via two-stage low rank approximation," *IEEE Trans. Vis. Comput. Graphics*, vol. 24, no. 6, pp. 2023–2036, Jun. 2018.

[48] K. Zhang, W. Zuo, S. Gu, and L. Zhang, "Learning deep CNN denoiser prior for image restoration," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2808–2817.

[49] T. Tirer and R. Giryes, "Image restoration by iterative denoising and backward projections," *IEEE Trans. Image Process.*, vol. 28, no. 3, pp. 1220–1234, Mar. 2019.

[50] M. V. Afonso, J. M. Bioucas-Dias, and M. A. T. Figueiredo, "Fast image recovery using variable splitting and constrained optimization," *IEEE Trans. Image Process.*, vol. 19, no. 9, pp. 2345–2356, Sep. 2010.

[51] B. K. Gunturk, Y. Altunbasak, and R. M. Mersereau, "Super-resolution reconstruction of compressed video using transform-domain statistics," *IEEE Trans. Image Process.*, vol. 13, no. 1, pp. 33–43, Jan. 2004.

[52] M. A. Robertson and R. L. Stevenson, "DCT quantization noise in compressed images," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 15, no. 1, pp. 27–38, Jan. 2005.

[53] H. Chang, M. K. Ng, and T. Zeng, "Reducing artifacts in JPEG decompression via a learned dictionary," *IEEE Trans. Signal Process.*, vol. 62, no. 3, pp. 718–728, Feb. 2014.

[54] Z. Zha, X. Yuan, J. Zhou, C. Zhu, and B. Wen, "A hybrid structural sparse error model for image deblocking," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2020, pp. 2493–2497.

[55] S. Osher, M. Burger, D. Goldfarb, J. Xu, and W. Yin, "An iterative regularization method for total variation-based image restoration," *Multiscale Model. Simul.*, vol. 4, no. 2, pp. 460–489, Jan. 2005.

[56] S. G. Chang, B. Yu, and M. Vetterli, "Adaptive wavelet thresholding for image denoising and compression," *IEEE Trans. Image Process.*, vol. 9, no. 9, pp. 1532–1546, Sep. 2000.

[57] W. Dong, G. Shi, and X. Li, "Nonlocal image restoration with bilateral variance estimation: A low-rank approach," *IEEE Trans. Image Process.*, vol. 22, no. 2, pp. 700–711, Feb. 2013.

[58] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.

[59] J. Ren, J. Liu, M. Li, W. Bai, and Z. Guo, "Image blocking artifacts reduction via patch clustering and low-rank minimization," in *Proc. Data Compress. Conf.*, Mar. 2013, p. 516.

[60] X. Zhang, R. Xiong, X. Fan, S. Ma, and W. Gao, "Compression artifact reduction by overlapped-block transform coefficient estimation with block similarity," *IEEE Trans. Image Process.*, vol. 22, no. 12, pp. 4613–4626, Dec. 2013.

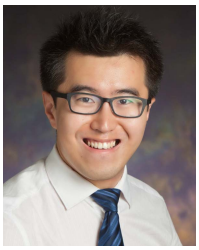
[61] S. Gu, Q. Xie, D. Meng, W. Zuo, X. Feng, and L. Zhang, "Weighted nuclear norm minimization and its applications to low level vision," *Int. J. Comput. Vis.*, vol. 121, no. 2, pp. 183–208, Jan. 2017.

[62] Y. Gao and X. Yang, "A cartoon-texture approach for JPEG/JPEG 2000 decompression based on TGV and shearlet transform," *IEEE Trans. Image Process.*, vol. 28, no. 3, pp. 1356–1365, Mar. 2019.

[63] S. I. Young, A. T. Naman, and D. Taubman, "COGL: Coefficient graph Laplacians for optimized JPEG image decoding," *IEEE Trans. Image Process.*, vol. 28, no. 1, pp. 343–355, Jan. 2019.
[64] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, Dec. 2012, pp. 1097–1105.
[65] C. Dong, Y. Deng, C. Change Loy, and X. Tang, "Compression artifacts reduction by a deep convolutional network," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 576–584.
[66] K. Kulkarni, S. Lohit, P. Turaga, R. Kerviche, and A. Ashok, "ReconNet: Non-iterative reconstruction of images from compressively sensed measurements," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 449–458.

Zhiyuan Zha (Member, IEEE) received the Ph.D. degree with the School of Electronic Science and Engineering, Nanjing University, Nanjing, China, in 2018. His current research interests include inverse problems in image/video processing, sparse signal representation, and machine learning. He was a recipient of the Best Paper Award at the IEEE International Conference on Multimedia and Expo (ICME) in 2017.

Xin Yuan (Senior Member, IEEE) received the B.Eng. and M.Eng. degrees from Xidian University in 2007 and 2009, respectively, and the Ph.D. degree from The Hong Kong Polytechnic University in 2012. He is currently a Video Analysis and a Coding Lead Researcher with Bell Labs, Murray Hill, NJ, USA. Prior to this, he was a Postdoctoral Associate with the Department of Electrical and Computer Engineering, Duke University, from 2012 to 2015, where he was working on compressive sensing and machine learning. He has been an Associate Editor of *Pattern Recognition* since 2019.



Bihan Wen (Member, IEEE) received the B.Eng. degree in electrical and electronic engineering from Nanyang Technological University, Singapore, in 2012, and the M.S. and Ph.D. degrees in electrical and computer engineering from the University of Illinois at Urbana–Champaign, USA, in 2015 and 2018, respectively. He is currently a Nanyang Assistant Professor with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore. His research interests span areas of machine learning, computational imaging, computer vision, image and video processing, and big data applications. He was a recipient of the 2016 Yee Fellowship and the 2012 Professional Engineers Board Gold Medal.

Jiachao Zhang (Member, IEEE) received the M.S. degree in electrical engineering from the University of Dayton, Dayton, OH, USA, in 2015, and the Ph.D. degree from the Nanjing University of Science and Technology, Nanjing, China, in 2011, respectively. She is currently an Assistant Professor with the Artificial Intelligence Institute of Industrial Technology, Nanjing Institute of Technology, Nanjing. Her research interests include color image processing and computer vision.

Jiantao Zhou (Senior Member, IEEE) received the B.Eng. degree from the Department of Electronic Engineering, Dalian University of Technology, in 2002, the M.Eng. degree from the Department of Radio Engineering, Southeast University, in 2005, and the Ph.D. degree from the Department of Electronic and Computer Engineering, Hong Kong University of Science and Technology, in 2009. He held various research positions with the University of Illinois at Urbana–Champaign, the Hong Kong University of Science and Technology, and McMaster University. He is currently an Associate Professor with the Department of Computer and Information Science, Faculty of Science and Technology, University of Macau. He holds four granted U.S. patents and two granted Chinese patents. His research interests include multimedia security and forensics, and multimedia signal processing. He has coauthored two articles that received the Best Paper Award at the IEEE Pacific-Rim Conference on Multimedia in 2007 and the Best Student Paper Award at the IEEE International Conference on Multimedia and Expo in 2016. He has been an Associate Editor of the IEEE TRANSACTIONS ON IMAGE PROCESSING since 2019.

Ce Zhu (Fellow, IEEE) received the B.S. degree in electronic and information engineering from Sichuan University, Chengdu, China, in 1989, and the M.Eng. and Ph.D. degrees in electronic and information engineering from Southeast University, Nanjing, China, in 1992 and 1994, respectively. He held a postdoctoral research position with the Chinese University of Hong Kong in 1995, The City University of Hong Kong, and the University of Melbourne, Australia, from 1996 to 1998. He was with Nanyang Technological University, Singapore, for 14 years from 1998 to 2012, where he was a Research Fellow, a Program Manager, an Assistant Professor, and then promoted to an Associate Professor in 2005. He has been with the University of Electronic Science and Technology of China, Chengdu, China, as a Professor since 2012. His research interests include video coding and communications, video analysis and processing, 3D video, and visual perception and applications. He has served on the editorial boards of a few journals, including as an Associate Editor of IEEE TRANSACTIONS ON IMAGE PROCESSING, the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, the IEEE TRANSACTIONS ON BROADCASTING, the IEEE SIGNAL PROCESSING LETTERS, an Editor of the IEEE COMMUNICATIONS SURVEYS AND TUTORIALS, and an Area Editor of *Signal Processing: Image Communication*. He has also served as a Guest Editor of a few special issues in international journals, including the IEEE JOURNAL OF SELECTED TOPICS IN SIGNAL PROCESSING. He is an IEEE Distinguished Lecturer of Circuits and Systems Society from 2019 to 2020.