Non-local Sparse Models for Image Restoration

Julien Mairal[1,5] Francis Bach[1,5] Jean Ponce[2,5] Guillermo Sapiro[3] Andrew Zisserman[2,4,5]

[1]INRIA [2]Ecole Normale Supérieure [3]University of Minnesota [4]Oxford University

Abstract

We propose in this paper to unify two different approaches to image restoration: On the one hand, learning a basis set (dictionary) adapted to sparse signal descriptions has proven to be very effective in image reconstruction and classification tasks. On the other hand, explicitly exploiting the self-similarities of natural images has led to the successful non-local means approach to image restoration. We propose simultaneous sparse coding as a framework for combining these two approaches in a natural manner. This is achieved by jointly decomposing groups of similar signals on subsets of the learned dictionary. Experimental results in image denoising and demosaicking tasks with synthetic and real noise show that the proposed method outperforms the state of the art, making it possible to effectively restore raw images from digital cameras at a reasonable speed and memory cost.

1. Introduction

This paper addresses the problem of reconstructing and enhancing a color image given the noisy observations gathered by a digital camera sensor. Today, with advances in sensor design, the signal is relatively clean for digital SLRs at low sensitivities, but it remains noisy for consumer-grade and mobile-phone cameras at high sensitivities (low-light and/or high-speed conditions). The restoration problem is thus still of acute and in fact growing importance (*e.g.*, [3, 7, 11, 15]), and we present a novel learned image model that outperforms the state of the art in denoising and demosaicking tasks on images with real and synthetic noise. This model should also prove of interest in deblurring and inpainting tasks that have become the topic of much recent research (*e.g.*, [2, 6, 23]) with the emergence of computational photography. Working with noisy images recorded by digital cameras is difficult since different devices produce different kinds of noise, and introduce different types of artefacts and spatial correlations in the noise as a result of internal post-processing (demosaicking, white balance, etc.). In this paper, we operate directly on the raw sensor output, that suffers from non-homogeneous noise, but is less spatially correlated and not corrupted by post-processing artefacts. In turn, this requires demosaicking the raw signal—that is, reconstructing a full color image from the sensor's RGB (Bayer) pattern—a difficult problem in itself. Whereas demosaicking is usually tackled using interpolation-based methods [13, 20, 32], much of the denoising effort has been aimed at finding a good model for natural images. Early work relied on various smoothness assumptions—such as anisotropic filtering [21], total variation [25], or image decompositions on fixed bases such as wavelets [17] for example. More recent approaches include non-local means filtering [3], which exploits image self-similarities, learned sparse models [11, 15], Gaussian scale mixtures [22], fields of experts [24], and block matching with 3D filtering (BM3D) [7].

In this paper, we view both denoising and demosaicking as image reconstruction problems, and propose a novel image model that combines two now classical techniques into a single framework: The *non-local means* approach to image restoration explicitly exploits self-similarities in natural images [3, 10] to average out the noise among similar patches, whereas *sparse coding* encodes natural image statistics by decomposing each image patch into a linear combination of a few elements from a basis set called a *dictionary*.[1] Although fixed dictionaries based on various types of wavelets [17] have been used in this setting, sparse decompositions based on learned, possibly overcomplete, dictionaries adapted to specific images have been shown to provide better results in practice [11, 15]. We propose to extend and combine these two approaches by using *simultaneous sparse coding* [28, 29, 31] to impose that similar patches share the same dictionary elements in their sparse decomposition. To the best of our knowledge, this is the first time that the corresponding models of image self-similarities are explicitly used in a common setting with learned dictionaries (the BM3D procedure [7] exploits both self-similarities and sparsity for the denoising task, but it is based on classical,

[5]WILLOW project, Laboratoire d'Informatique de l'Ecole Normale Supérieure, ENS/INRIA/CNRS UMR 8548.

[1]The usage of the word "basis" is slightly abusive here since the elements of the dictionaries are not (a priori) necessarily independent.

2009 IEEE 12th International Conference on Computer Vision (ICCV)
978-1-4244-4419-9/09/$25.00 ©2009 IEEE

fixed orthogonal dictionaries). Experiments with images corrupted by synthetic or real noise show that the proposed method outperforms the state of the art in both image denoising and image demosaicking tasks, making it possible to effectively restore raw images from digital cameras at a reasonable speed and memory cost. Furthermore, although it is demonstrated on image denoising and demosaicking tasks in this paper, our model is generic, admits straightforward extensions to various image and video restoration tasks such as inpainting, and can adapt to a large class of data, *e.g.*, multispectral images or MRI data.

2. Related Work

We start with a brief description of well-established approaches to image restoration that are relevant and related to the approach proposed in the next section. Since it is difficult to design a standard model for digital camera noise, these methods assume white Gaussian noise. Even though this generic setting slightly differs from that of real image denoising, it has allowed the development of effective algorithms that are now widely used in digital cameras and commercial software packages. We will use the same assumption in the rest of this paper, but will demonstrate empirically that our approach is effective at restoring real images corrupted by non-Gaussian, non-uniform noise.

2.1. Non-Local Means Filtering

Efros and Leung showed in [10] that the self-similarities inherent to natural images could effectively be used in texture synthesis tasks. Following their insight, Buades, Coll and Morel introduced in [3] the *non-local means* approach to image denoising, where the prominence of self-similarities is used as a prior on natural images.[2] Concretely, let us consider a noisy image written as a column vector $\boldsymbol{y}$ in $\mathbb{R}^n$, and denote by $\boldsymbol{y}[i]$ the i-th pixel and by $\boldsymbol{y}_i$ the patch of size m centered on this pixel for some appropriate size m. This approach exploits the simple but very effective idea that two pixels associated with similar patches $\boldsymbol{y}_i$ and $\boldsymbol{y}_j$ should have similar values $\boldsymbol{y}[i]$ and $\boldsymbol{y}[j]$. Using $\boldsymbol{y}_i$ as an explanatory variable for $\boldsymbol{y}[i]$ leads to the non-local means formulation, where the denoised pixel $\boldsymbol{x}[i]$ is obtained by a weighted average (the corresponding Nadaraya-Watson estimator [3]):

$$\boldsymbol{x}[i] = \sum_{j=1}^{n} \frac{K_h(\boldsymbol{y}_i - \boldsymbol{y}_j)}{\sum_{l=1}^{n} K_h(\boldsymbol{y}_i - \boldsymbol{y}_l)} \boldsymbol{y}[j], \tag{1}$$

and K_h is a Gaussian kernel of bandwidth h.

[2]This idea has in fact appeared in the literature in various guises and under different equivalent interpretations, *e.g.*, kernel density estimation [10], Nadaraya-Watson estimators [3], mean-shift iterations [1], diffusion processes on graphs [26], and long-range random fields [14].

2.2. Learned Sparse Coding

An alternative is to assume that the clean signal can be approximated by a *sparse* linear combination of elements from a basis set called dictionary. Under this assumption, denoising a patch $\boldsymbol{y}_i$ in $\mathbb{R}^m$ with a dictionary $\mathbf{D}$ in $\mathbb{R}^{m \times k}$ composed of k elements, amounts to solving the sparse decomposition problem

$$\min_{\boldsymbol{\alpha}_i \in \mathbb{R}^k} ||\boldsymbol{\alpha}_i||_p \ \text{ s.t. } \ ||\boldsymbol{y}_i - \mathbf{D}\boldsymbol{\alpha}||_2^2 \leq \varepsilon, \tag{2}$$

where $\mathbf{D}\boldsymbol{\alpha}$ is an estimate of the clean signal, and $||\boldsymbol{\alpha}||_p$ is a sparsity-inducing regularization term. This regularizer is associated with the ℓ_1 norm when $p = 1$, leading to the well-known Lasso [27] and basis pursuit [5] problems, and with the ℓ_0 *pseudo norm* when $p = 0$.[3] Note that the dictionary may be *overcomplete*—that is, the number of columns of $\mathbf{D}$ may be greater than the number of its rows. Following [11, 15], ε can be chosen according to the (supposed known) standard deviation σ of the noise. One indeed expects the residual $\boldsymbol{y}_i - \mathbf{D}\boldsymbol{\alpha}_i$ to behave as a Gaussian vector, and thus $||\boldsymbol{y}_i - \mathbf{D}\boldsymbol{\alpha}_i||_2^2/\sigma^2$ to follow a chi-squared distribution χ_m^2 concentrated around m. The strategy proposed in [15] is to threshold the cumulative distribution function F_m of the χ_m^2 distribution and choose ε as $\varepsilon = \sigma^2 F_m^{-1}(\tau)$, where F_m^{-1} is the inverse of F_m. Selecting the value $\tau = 0.9$ leads in practice to acceptable values of ε [15].

Various types of wavelets [17] have been used as dictionaries for natural images. Building on ideas proposed in [19] to model neuronal responses in the V1 area of the brain, Elad and Aharon [11] have proposed instead to *learn* a dictionary $\mathbf{D}$ adapted to the image at hand, and demonstrated that learned dictionaries lead to better empirical performance than off-the-shelf ones. Since images may be very large, efficiency concerns naturally lead to sparsely decomposing image patches rather than the full image. For an image of size n, a dictionary in $\mathbb{R}^{m \times k}$ adapted to the n overlapping patches of size m (typically $m = 8 \times 8 \ll n$) associated with the image pixels, is learned by addressing the following optimization problem

$$\min_{\mathbf{D} \in \mathcal{C}, \mathbf{A}} \sum_{i=1}^{n} ||\boldsymbol{\alpha}_i||_p \ \text{ s.t. } \ ||\boldsymbol{y}_i - \mathbf{D}\boldsymbol{\alpha}_i||_2^2 \leq \varepsilon, \tag{3}$$

where $\mathcal{C}$ is the set of matrices in $\mathbb{R}^{m \times k}$ with unit ℓ_2-norm columns, $\mathbf{A} = [\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_n]$ is a matrix in $\mathbb{R}^{k \times n}$, $\boldsymbol{y}_i$ is the i-th patch of the *noisy* image $\boldsymbol{y}$, $\boldsymbol{\alpha}_i$ is the corresponding code, and $\mathbf{D}\boldsymbol{\alpha}_i$ is the estimate of the denoised patch. Note that this procedure implicitly assumes that the patches are *independent* from each other, which is questionable since

[3]The ℓ_p norm of a vector $\boldsymbol{x}$ in $\mathbb{R}^m$ is defined, for $p \geq 1$, by $||\boldsymbol{x}||_p \triangleq (\sum_{i=1}^{m} |\boldsymbol{x}[i]|^p)^{1/p}$. Following tradition, we denote by $||\boldsymbol{x}||_0$ the number of nonzero elements of the vector $\boldsymbol{x}$. This "ℓ_0" sparsity measure is not a true norm.

they overlap. However, this approximation makes the corresponding optimization tractable. Indeed, although dictionary learning is traditionally considered as extremely costly, online procedures such as [16] make it possible to efficiently process millions of patches, allowing the use of large photographs and/or large image databases.

Once the dictionary $\mathbf{D}$ and codes $\boldsymbol{\alpha}_i$ have been learned, every pixel admits m estimates (one per patch containing it), and its value can be computed by averaging these:

$$\boldsymbol{x} = \frac{1}{m}\sum_{i=1}^{n} \mathbf{R}_i \mathbf{D}\boldsymbol{\alpha}_i, \tag{4}$$

where $\mathbf{R}_i$ in $\mathbb{R}^{n\times m}$ is the binary matrix which places patch number i at its proper position in the image. This approach learns the dictionary on the set of overlapping noisy patches, thereby adapting the dictionary to the image itself, which is a key element in obtaining better results.

How to choose between $p = 0$ or $p = 1$ is not a priori clear. Solving Eq. (2) with $p = 0$ is NP hard, leading to approximate solutions obtained with a greedy algorithm such as forward selection [30] (also known as orthogonal matching pursuit [18]). When $p = 1$, the problem is convex and can be solved efficiently with the LARS algorithm [9]. Following Elad and Aharon [11], we have observed experimentally that, given a fixed dictionary $\mathbf{D}$, the reconstructed image is in general of better quality when using the ℓ_0 pseudo norm rather than its convex ℓ_1 counterpart. However, we have also observed that dictionaries learned with the ℓ_1 norm are usually better for denoising, even when the final reconstruction is done with the ℓ_0 pseudo norm.

2.3. Block Matching 3D (BM3D)

Dabov *et al*. propose in [7] a patch-based procedure that exploits image self-similarities and gives state-of-the-art results. As in [11], they estimate the codes of overlapping patches and average the estimates. However, similar to non-local means filtering [3], they reconstruct patches by finding similar ones in the image (*block matching*), stacking them together into a 3D signal block, and denoising the block using hard or soft thresholding [8] with a 3D orthogonal dictionary (*3D filtering*). In conjunction with a few heuristics,[4] this simple idea has proven to be very efficient and gives better results than regular non-local means. A key idea of our paper is to implement a similar joint decomposition approach in the context of sparse coding with learned dictionaries, as explained in the next section.

3. Proposed Formulation

We show in this section how image self-similarities can be used to improve learned sparse models with *simultaneous sparse coding*, which encourages similar patches to admit similar sparse decompositions.

[4]Namely, using a combination of weighted averages of overlapping patches, Kaiser windows, and Wiener filtering to further improve results.

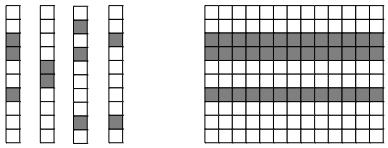

Figure 1. Sparsity vs. joint sparsity: Grey squares represents non-zeros values in vectors (left) or matrix (right).

3.1. Simultaneous Sparse Coding

A joint sparsity pattern—that is, a common set of nonzero coefficients—can be imposed to a set of vectors $\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_l$ through a *grouped-sparsity regularizer* on the matrix $\mathbf{A} = [\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_l]$ in $\mathbb{R}^{k\times l}$ (Figure 1). This amounts to restricting the number of nonzero rows of $\mathbf{A}$, or replacing the ℓ_p vector (pseudo) norm in Eq. (3) by the $\ell_{p,q}$ (pseudo) matrix norm

$$||\mathbf{A}||_{p,q} \triangleq \sum_{i=1}^{k} ||\boldsymbol{\alpha}^i||_q^p, \tag{5}$$

where $\boldsymbol{\alpha}^i$ denotes the i-th row of $\mathbf{A}$. In practice, one usually chooses for the pair (p, q) the values $(1, 2)$ or $(0, \infty)$, the former leading to a convex norm, while the latter actually counts the number of nonzero rows and is only a pseudo norm [28].

3.2. Principle of the Formulation

Non-local means filtering has proven very effective in general, but it fails in some cases. In the extreme, when a patch does not look like any other one in the image, it is impossible to exploit self-similarities to denoise the corresponding pixel value. Sparse image models can handle such situations by exploiting the redundancy between overlapping patches, but they suffer from another drawback: Similar patches sometimes admit very different estimates due to the potential instability of sparse decompositions (the ℓ_0 pseudo norm is, after all, piecewise constant, and its ℓ_1 counterpart is only piecewise differentiable), which can result in practice in noticeable reconstruction artefacts. In this paper, we address this problem by forcing similar patches to admit similar decompositions. Concretely, let us define for each patch $\boldsymbol{y}_i$ the set S_i of similar patches as

$$S_i \triangleq \{j = 1, \ldots, n \text{ s.t. } ||\boldsymbol{y}_i - \boldsymbol{y}_j||_2^2 \leq \xi\}, \tag{6}$$

where ξ is some threshold. Let us also consider for the moment a fixed dictionary $\mathbf{D}$ in $\mathbb{R}^{m\times k}$. Decomposing the patch $\boldsymbol{y}_i$ with a grouped-sparsity regularizer on the set S_i amounts to solving

$$\min_{\mathbf{A}_i} ||\mathbf{A}_i||_{p,q} \text{ s.t. } \sum_{j\in S_i} ||\boldsymbol{y}_j - \mathbf{D}\boldsymbol{\alpha}_{ij}||_2^2 \leq \varepsilon_i, \tag{7}$$

where $\mathbf{A}_i = [\boldsymbol{\alpha}_{ij}]_{j \in \mathcal{S}_i} \in \mathbb{R}^{k \times |S_i|}$. We adopt the same strategy as in Section 2.2 to choose ε_i accordingly to the size of S_i: $\varepsilon_i = \sigma^2 F^{-1}_{m|S_i|}(\tau)$. In the $\ell_{1,2}$-case, this optimization problem is convex and can be solved efficiently [12]. In the $\ell_{0,\infty}$ case, on the other hand, it is intractable, and a greedy approach such as simultaneous orthogonal matching pursuit [28] must be used to obtain an approximate solution.

In the framework of learned sparse coding, adapting $\mathbf{D}$ to the image(s) of interest naturally leads to the following optimization problem

$$\min_{(\mathbf{A}_i)_{i=1}^n, \mathbf{D} \in \mathcal{C}} \sum_{i=1}^{n} \frac{||\mathbf{A}_i||_{p,q}}{|S_i|^p} \text{ s.t. } \forall i \sum_{j \in S_i} ||\boldsymbol{y}_j - \mathbf{D}\boldsymbol{\alpha}_{ij}||_2^2 \leq \varepsilon_i \tag{8}$$

where $\mathbf{D}$ is in $\mathbb{R}^{m \times k}$ with unit ℓ_2-norm columns. The normalization by $|S_i|^p$ is used to ensure equal weights for all groups (as before, we only consider the cases where (p, q) is $(1, 2)$ or $(0, \infty)$). As noted in the previous section, in classical learned sparse coding, we prefer the ℓ_1 norm for learning the dictionary and the ℓ_0 pseudo norm for the final reconstruction. We adopt here a similar choice: We use the convex $\ell_{1,2}$ norm for learning the dictionary, which can be done efficiently using a simple modification of [16], and we use the $\ell_{0,\infty}$ pseudo-norm for the final reconstruction. As in [11], this formulation allows all the image patches to be processed as if they were *independent* of each other. To reconstruct the final image, we average the estimates of each pixel,

$$\boldsymbol{x} = \text{diag}(\sum_{i=1}^{n} \sum_{j \in S_i} \mathbf{R}_j \mathbf{1}_m)^{-1} \sum_{i=1}^{n} \sum_{j \in S_i} \mathbf{R}_j \mathbf{D}\boldsymbol{\alpha}_{ij}, \tag{9}$$

where $\mathbf{R}_j$ is defined as in Eq. (4) and $\mathbf{1}_m$ is a vector of size m filled with ones. The term on the left is a scaling diagonal matrix, counting the number of estimates for each pixel. Note that when $\mathcal{S}_i = \{i\}$, our formulation is equivalent to regular learned sparse coding.

At first sight, the proposed technique may seem particularly costly, since decomposing a single patch requires solving a large-scale optimization problem (7). Similar concerns hold for the original formulations of non-local means [3] and BM3D [7]. As in these cases, slight changes to our approach are sufficient to make it efficient.

3.3. Practical Formulation and Implementation

The computational cost of the optimization problem (8) is dominated by the computation of the vectors $\boldsymbol{\alpha}_{ij}$. In the worst case scenario, n^2 of these vectors have to be computed. We show in the rest of this section how to modify our original formulation in order to make this number linear in n and allow efficient optimization.

Semi-local grouping. When building S_i, one can restrict the search for patches similar to $\boldsymbol{y}_i$ to a window of size $w \times w$. This semi-local approach is also used in [7], and it reduces the worst-case number of vectors $\boldsymbol{\alpha}_{ij}$ to nw^2. In practice, we never use w greater than 64 in this paper.

Clustering. It is also possible to cluster pixels into disjoint groups C_k such that all pixels i in C_k share the same set S_i. The optimization problems (7) associated with all pixels in the same cluster are identical, further reducing the overall computational cost: In fact, only n vectors $\boldsymbol{\alpha}_{ij}$ are computed in this case since each pixel belongs to exactly one cluster. This is a key ingredient to the efficiency of our implementation. Other strategies are also possible, allowing a few clusters to overlap for instance.

Initialization of D. One important asset of sparse representations is that they can benefit from dictionaries learned offline on a database of natural images, which can be used as a good initial dictionaries for the denoising procedure [11]. Using the online procedure of [16], our initial dictionaries are learned on 2×10^7 patches of natural images taken randomly from the $10\,000$ images of the PASCAL VOC'07 database. As shown in the next section, using this online procedure and such a large training sample has led to a significant performance improvement compared to methods such as [15] that use batch learning methods such as K-SVD [11] and are unusable with such large-scale data.

Improved matching. Following [7], we have noticed that better groups of similar patches can be found by using a first round of denoising on the patches (using, for example, the classical sparse coding approach of Eq. (3) presented in the previous section) before grouping them. In turn, as shown by our experiments, our simultaneous sparse coding approach greatly improves on this initial denoising step.

Patch normalization. To improve the numerical stability of sparse coding, the mean intensity (or RGB color) value of a patch is often subtracted from all its pixel values before decomposing it, then added back to the estimated values [11]. We have adopted this approach in our implementation, and our experiments have shown that it improves the visual quality of the results.

Reducing the memory cost. At first sight, Eq. (8) requires storing a large number of codes $\boldsymbol{\alpha}_{ij}$. Even though these are sparse, and their number can be reduced to the number of pixels using the clustering strategy presented above, this could potentially be a problem for large images. In fact, only a small subset of the vectors $\boldsymbol{\alpha}_{ij}$ is stored at any given time: The online procedure of [16] computes them *on the fly* and does not require storing them to learn the dictionary. In the case of Eq. (8), the maximum number of vectors $\boldsymbol{\alpha}_{ij}$ that have to be stored at any given time is the size of the largest cluster C_k of similar patches.

3.4. Real Images and Demosaicking

Single-chip digital cameras do not capture a noisy RGB signal at each pixel. Instead, combined with a red (R), green

(G), or blue (B) filter, the sensor associated with each pixel integrates the incoming light flux over the corresponding frequency range and a short period of time. The relation between the pixels and the color information they record is obtained through a specific pattern, the most famous one being the Bayer pattern, G-R-G-R on odd lines and B-G-B-G on even ones. The demosaicking problem consists of reconstructing the whole color image given the sensor measurements. Although most of the approaches found in the literature to solve this problem are based on interpolation [13, 20, 32], the image models investigated in this paper have also been used for demosaicking: Self-similarities have been exploited in [4], and learned sparse coding has been used in [15]. We adapt here [15] to our simultaneous sparse coding framework. First we learn an initial dictionary $\mathbf{D}_0$ using [16] on a database of natural color images. Our demosaicking procedure can then be decomposed into four simple steps:

(1) Cluster similar patches on the mosaicked image $\boldsymbol{y}$.

(2) Reconstruct each patch using $\mathbf{D}_0$, addressing for all i

$$\min_{\mathbf{A}_i \in \mathbb{R}^{k \times |S_i|}} ||\mathbf{A}_i||_{0,\infty} \text{ s.t. } \forall j \; M_j(\boldsymbol{y}_j - \mathbf{D}_0 \boldsymbol{\alpha}_{ij}) = 0, \quad (10)$$

where M_j is a binary masked corresponding to the Bayer pattern of measured values, and average the reconstructions to obtain an estimate $\boldsymbol{x}$ of the demosaicked image.

(3) Learn a dictionary $\mathbf{D}_1$ for $\boldsymbol{x}$ with a strong regularization—that is, replace $\boldsymbol{y}$ by $\boldsymbol{x}$ in Eq. (8), solving this equation with a large value for ε_i.

(4) Reconstruct each patch using $\mathbf{D}_2 = [\mathbf{D}_0 \; \mathbf{D}_1]$ instead of $\mathbf{D}_0$ in Eq. (10), and average the estimates using Eq. (9) to obtain the final demosaicked image.

As shown in the next section, this procedure outperforms the state of the art from quantitative and qualitative points of view. The raw mosaicked signal of digital cameras in low-light, short-exposure settings is noisy. It should therefore be denoised before demosaicking is attempted. Since our denoising procedure is generic and does not necessary assume the input data to be natural images, *the denoising procedure can be performed on the mosaicked image itself.*

4. Experimental Validation

4.1. Denoising – Synthetic Noise

Experiments on denoising with synthetic white Gaussian noise have carried out with 12 standard benchmark images. The parameters used in this experiment are $k = 512$, $m = 9 \times 9$ for $\sigma \leq 25$, $m = 12 \times 12$ for $\sigma = 50$ and $m = 16 \times 16$ for $\sigma = 100$. The value of τ is chosen a bit more conservatively than in [15] and is set to 0.8, while ξ is chosen according to an empirical rule, $\xi = (32\sigma)^2/m$ for images scaled between 0 and 255, which has shown to be appropriate in all of our denoising experiments for both real and synthetic noise. Peak signal-to-noise ratio (PNSR) is used as performance measure in our quantitative evaluation.[5] Table 1 reports the results obtained on each image for different values of the (known) standard deviation of the noise σ, and Table 2 compares the average PSNR on these images obtained by several state-of-the-art image denoising methods—namely GSM [22], FoE [24], K-SVD [11] and BM3D [7]—with our method in three settings: SC (sparse coding) uses a fixed dictionary learned on a database of natural images without grouping the patches. It is therefore similar to the *global* approach to denoising of [11]. The only differences are that we have used the online procedure of [16] to learn the dictionary from 2×10^7 natural image patches instead of the 10^5 patches used in [11], and we have used an ℓ_1 regularizer instead of an ℓ_0 one to learn the dictionary. In the second setting (LSC, for learned sparse coding), the dictionary is adapted to the test image, again using an ℓ_1 regularizer, which is similar to the *adaptive* approach of [11] except for our (better) initial dictionary and their ℓ_0 regularizer. The last setting (LSSC, for learned simultaneous sparse coding) adds a grouping step and uses the full power of our simultaneous sparse coding framework. These PSNR comparisons show that our model leads to better performance than the state-of-the-art techniques in general, and is always at least as good as BM3D, the top performer among those, especially for high values of σ. Additional qualitative examples are given in Figure 2.

Note that the parameters have not been optimized for speed but for quality in these experiments. On a recent Intel Q9450 2.66Ghz CPU, it takes for instance 0.5s to denoise the 256×256 image peppers with $\sigma = 25$ and the setting SC, 85s with LSC, and 220s with LSSC. With parameters optimized for speed ($k = 256$, fewer iterations in the dictionary learning procedure), the computation times become respectively 0.25s for SC, 10s for LSC, and 21s for LSC, and the final results' quality only drops by 0.05dB, which is visually imperceptible. Our framework is therefore flexible in terms of speed/quality compromise.

4.2. Demosaicking

We have used the standard Kodak PhotoCD benchmark to evaluate the performance of our demosaicking algorithm. This dataset consists of 24 RGB images of size 512×768 to which a Bayer mask has been applied. Ground truth is thus available, allowing quantitative comparisons. We have arbitrarily tuned the parameters of our method to optimize its performance on the 5 last images, choosing $k = 256$ (dictionary size), $m = 8 \times 8$ (patch size), and $\xi = 3 \times 10^4$ (for images scaled between 0 and 255). These parameters

[5] Denoting by MSE the mean-squared-error for images whose intensities are between 0 and 255, the PSNR is defined as PSNR $= 10 \log_{10}(255^2/\text{MSE})$ and is measured in dB. A gain of 1dB reduces the MSE by approximately 20%.

σ	5	10	15	20	25	50	100
house	39.93	36.96	35.35	34.16	33.15	30.04	25.83
peppers	38.18	34.80	32.82	31.37	30.21	26.62	23.00
camera.	38.32	34.21	32.01	30.57	29.51	26.42	23.08
lena	38.69	35.83	34.15	32.90	31.87	28.87	25.82
barbara	38.48	34.97	33.00	31.57	30.47	27.06	23.59
boat	37.35	34.02	32.20	30.89	29.87	26.74	23.84
hill	37.17	33.67	31.89	30.71	29.80	27.05	24.44
couple	37.45	33.98	32.06	30.69	29.61	26.30	23.28
man	37.89	34.06	32.01	30.64	29.63	26.69	24.00
fingerp.	36.70	32.57	30.31	28.78	27.62	24.25	21.26
bridge	35.78	31.22	28.92	27.46	26.42	23.68	21.46
flintst.	36.13	32.46	30.78	29.63	28.71	25.16	21.10
Av.	37.67	34.06	32.12	30.78	29.74	26.57	23.39

Table 1. Quantitative denoising experiments on 12 standard images. The PSNR values are averaged over 5 experiments with 5 different noise realizations and values of σ between 5 and 100. The variance is negligible and not reported due to space limitations.

σ	[22]	[24]	[11]	[7]	SC	LSC	LSSC
5	37.05	37.03	37.42	37.62	37.46	37.66	**37.67**
10	33.34	33.11	33.62	34.00	33.76	33.98	**34.06**
15	31.31	30.99	31.58	32.05	31.72	31.99	**32.12**
20	29.91	29.62	30.18	30.73	30.29	30.60	**30.78**
25	28.84	28.36	29.10	29.72	29.18	29.52	**29.74**
50	25.66	24.36	25.61	26.38	25.83	26.18	**26.57**
100	22.80	21.36	22.10	23.25	22.46	22.62	**23.39**

Table 2. Quantitative comparative evaluation. We compare our algorithm to GSM [22], FoE [24], K-SVD [11] and BM3D [7], that were the top performers so far on this benchmark, and whose implementations are available online. The PSNR is chosen as before as performance measure. Best results are in bold.

Figure 2. Qualitative evaluation of our denoising method with standard images. Left: noisy images. Right: restored images. Note that we reproduce the original brick texture in the house image ($\sigma = 15$) and the hair texture for the man image ($\sigma = 50$), both hardly visible in the noisy images. (The details are better seen by zooming on a computer screen.)

Figure 3. Left: Demosaicking with LSC sometimes causes artefacts such as the yellow and blue pixels in the middle of the fence. Right: The reconstruction obtained with the LSSC algorithm does not exhibit such artefacts. (This figure should be viewed in color.)

have been used for all 24 photos.

We evaluate the performance of the three variants SC, LSC, LSSC of our framework defined in the previous subsection, and compare them with the state of the art using the experimental protocol of Paliy *et al*. [20] whose LPA method is, to the best of our knowledge, the top performer so far in terms of PSNR (or equivalently mean-squared error) on the Kodak PhotoCD benchmark. Following [20], we have excluded a 15-pixel border in fairness to methods that are susceptible to boundary effects. Table 3 adds our results to those reported in [20] for each one of the 24 photos. The proposed LSSC method outperforms the state-of-the-art algorithms AP [13], DL [32] and LPA [20] by a significant margin of 0.87dB even though our formulation is generic and not tuned to the task of demosaicking, demonstrating the promise of our image model.

When including the image border so as to be able to compare our results with those of [15], it is interesting to note that, in the SC setting, we achieve a mean PSNR of 40.72dB on the 24 images, compared to the 39.56dB of [15]. Clearly, it is thus preferable in this case to learn the dictionary from a large dataset of natural images. With LSC, we achieve a mean PSNR of 40.98dB, compared to the 40.32dB of [15], reaching a mean PSNR of 41.24dB with LSSC. Although this quantitative improvement may seem small, it is qualitatively quite significant. Even though SC and LSC perform very well in terms of PSNR, they suffer from classical demosaicking artefacts, as shown by the example of Figure 3. On the other hand, our new LSSC model, which exploits self-similarities as well as learned sparse coding, is usually free of most of these artefacts.

4.3. Denoising – Real Noise

To evaluate qualitatively our denoising method on real images, we have taken three RAW photographs using a Canon Powershot G9 digital camera at 1600 ISO with a short time exposure. At such a setting, the images are quite noisy. We have extracted the mosaicked data from the RAW image using the open-source *dcraw* software. We have then scaled manually the R,G,B channels so that they

Im.	AP	DL	LPA	SC	LSC	LSSC
1	37.84	38.46	40.47	40.84	40.92	**41.36**
2	39.64	40.89	41.36	41.76	42.03	**42.24**
3	41.40	42.66	43.47	43.15	43.92	**44.24**
4	39.92	40.49	40.84	41.99	42.14	**42.45**
5	37.28	38.07	37.51	38.72	39.15	**39.45**
6	38.69	40.19	40.92	41.29	41.36	**41.71**
7	41.75	42.35	43.06	43.30	43.59	**44.06**
8	35.58	36.02	37.13	37.42	37.38	**37.57**
9	41.84	43.05	43.50	43.17	43.74	**43.83**
10	41.93	42.54	42.77	43.01	43.17	**43.33**
11	39.25	40.01	40.51	41.19	41.29	**41.51**
12	42.62	43.45	44.01	44.29	44.49	**44.90**
13	34.28	34.75	36.08	36.16	36.29	**36.35**
14	35.66	36.91	36.86	37.64	38.48	**38.77**
15	39.17	39.82	40.09	41.04	41.24	**41.74**
16	42.10	43.75	44.02	44.36	44.42	**44.91**
17	41.23	41.68	41.75	41.75	41.86	**41.98**
18	37.31	37.64	37.59	38.05	38.27	**38.38**
19	39.99	41.01	41.55	41.58	41.71	**42.31**
20	40.63	41.24	41.48	41.95	42.25	**42.27**
21	38.72	39.10	39.61	40.55	40.59	**40.65**
22	37.63	38.37	38.44	38.73	38.97	**39.24**
23	41.93	43.22	43.92	43.47	43.93	**44.34**
24	34.74	35.55	35.44	35.59	35.85	**35.89**
Av.	39.21	40.05	40.52	40.88	41.13	**41.39**

Table 3. Comparison of demosaicking performance in terms of PSNR between AP [13], DL [32], LPA [20] and the SC, LSC and LSSC variants of our method. Best results are in bold.

visually appear to contain similar amounts of noise. At this point, the noise is, to a first approximation, roughly uniform, and we apply our denoising algorithm to the scaled mosaicked image, before performing demosaicking, white balance, sRGB space conversion, gamma correction, and contrast enhancement to reconstruct the final image. This approach has proven experimentally to lead to better results than denoising each R,G,B channel independently. Of course, assuming that the noise is uniform is only a rough approximation. Non-spatially uniform noise models are available for specific cameras, and exploited by commercial software packages such at those discussed later in this section. Incorporating these models into our framework is feasible (following [15]), but beyond the scope of this paper. Instead, we demonstrate that, even with a uniform assumption, our algorithm is qualitatively competitive with top-of-the-line commercial denoising software.

The parameters we have used are a patch size of $m = 8 \times 8$ pixels, and $k = 256$ dictionary elements, which is typical for sparse coding methods [11, 15]. The noise level σ is estimated by the user and assumed to be uniform across the image, and ξ is chosen according to the empirical rule presented in Section 4.1. Demosaicking is performed using the same parameters as in Section 4.2. Figure 4 compares closeups of the images reconstructed from the RAW file by the camera itself (jpeg output), the image obtained with Adobe Camera Raw 5.0 (no denoising), two state-of-the-art denoising softwares NoiseWare 4.2 and the DxO Optics Pro 5.3 package, and our method. The commercial programs have been run with their default parameters, and these could certainly be further tuned to improve image quality a bit.[6] However, note that, unlike ours, these programs do take advantage of a detailed, non-uniform noise model specific to the camera, yet do not appear to give qualitatively better results. Although a quantitative comparison is not possible, we believe (subjectively) that our method does best on the first and third images, while DxO Optics Pro is slightly better for the second one. As in our previous experiments, LSSC suffers from fewer artefacts than LSC in general. The noise's non-uniformity does not seem to affect our results much, except perhaps for the background of the third image, where part of the noise is reconstructed.

5. Conclusion

We have proposed in this paper a new image model that combines the non-local means and sparse coding approaches to image restoration into a unified framework where similar patches are decomposed using similar sparsity patterns. Quantitative and qualitative experiments with images corrupted with synthetic or real noise have shown that the proposed algorithm outperforms the state of the art in image demosaicking and denoising tasks. Next on our agenda is to include non-uniform noise models in the reconstruction process, then adapt our approach to other challenging image manipulation problems in computational photography, including deblurring, inpainting, and texture synthesis in still images and video sequences.

Acknowledgments

This paper was supported in part by ANR under grant MGA. The work of Guillermo Sapiro is partially supported by NSF, NGA, ONR, ARO, and DARPA. We would like to thank Frédéric Guichard and Frédéric Cao from DxO for interesting discussions.

References

[1] S. Awate and R. Whitaker. Unsupervised, information-theoretic, adaptive image filtering for image restoration. *IEEE T. PAMI*, 364–376, 2006.

[2] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester. Image inpainting. In *Proc. Comp. Graph. and Interact. Tech.*, 2000.

[3] A. Buades, B. Coll, and J. Morel. A non-local algorithm for image denoising. In *Proc. IEEE CVPR*, 2005.

[4] A. Buades, B. Coll, J. Morel, and C. Sbert. Non-local demosaicing. Technical report, 2007. Preprint CMLA 2007-15.

[5] S. Chen, D. Donoho, and M. Saunders. Atomic decomposition by basis pursuit. *SIAM J. Sc. Comp.*, 20:33–61, 1999.

[6] Note that NoiseWare does not process directly the RAW files, but requires at first to use a demosaicking software. We have chosen to combine NoiseWare and Adobe Camera Raw in our experiments.

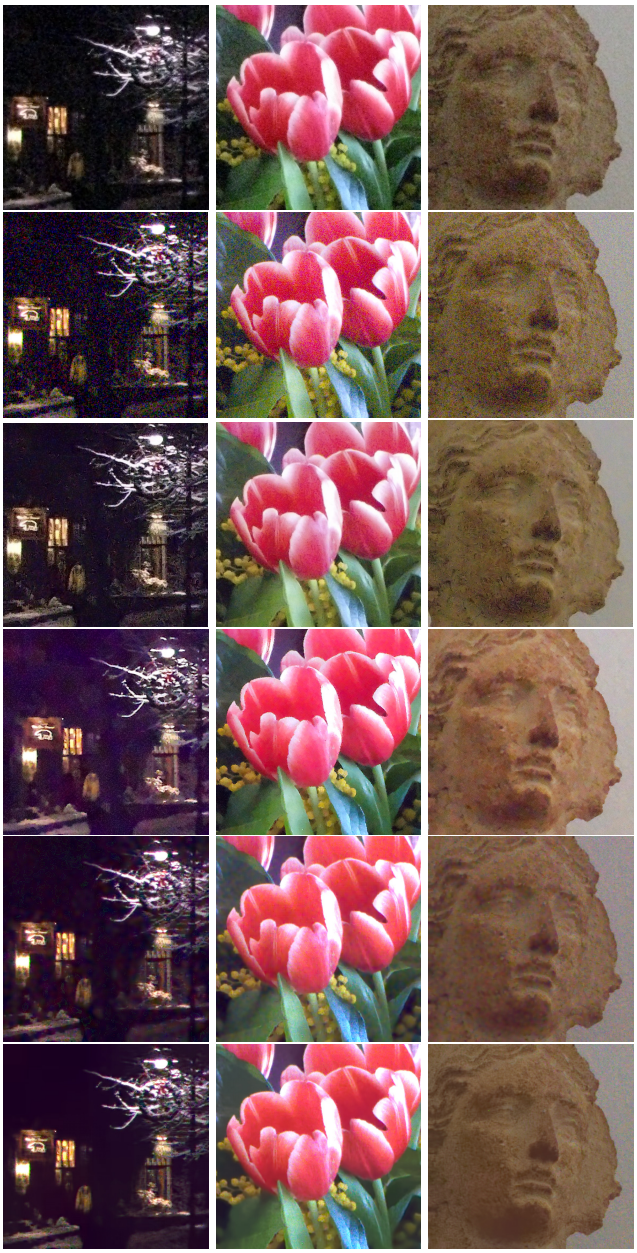

Figure 4. From top to bottom: Camera jpeg output, Adobe Camera Raw, NoiseWare, DxO Optics Pro, LSC, and proposed LSSC algorithm. (This figure should be viewed in color and by zooming on a computer screen.)

[6] A. Criminisi, P. Pérez, and K. Toyama. Region filling and object removal by exemplar-based inpainting. *IEEE T. IP*, 13(9):1200–1212, 2004.

[7] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Image Denoising by Sparse 3-D Transform-Domain Collaborative Filtering. *IEEE T. IP*, 16(8):2080–2095, 2007.

[8] D.L. Donoho, I.M. Johnstone. Adapting to Unknown Smoothness Via Wavelet Shrinkage. *J. of the American Stat. Assoc.*, 90(432):1200–1224, 1995.

[9] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *Ann. Statist.*, 32(2):407–499, 2004.

[10] A. Efros and T. Leung. Texture synthesis by non-parametric sampling. In *Proc. ICCV*, 1999.

[11] M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE T. IP*, 54(12):3736–3745, 2006.

[12] J. Friedman, T. Hastie, H. Hölfling, and R. Tibshirani. Pathwise coordinate optimization. *Ann. Stats.*, 1(2):302–332, 2007.

[13] B. Gunturk, Y. Altunbasak, and R. Mersereau. Color plane interpolation using alternating projections. *IEEE T. IP*, 11(9):997–1013, 2002.

[14] Y. Li and D. P. Huttenlocher. Sparse long-range random field and its application to image denoising. In *Proc. ECCV*, 2008.

[15] J. Mairal, M. Elad, and G. Sapiro. Sparse representation for color image restoration. *IEEE T. IP*, 17(1):53–69, 2008.

[16] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online dictionary learning for sparse coding. ICML, 2009.

[17] S. Mallat. *A Wavelet Tour of Signal Processing, 2nd Edition*. Academic Press, 1999.

[18] S. Mallat and Z. Zhang. Matching pursuit in a time-frequency dictionary. *IEEE T. SP*, 41(12):3397–3415, 1993.

[19] B. A. Olshausen and D. J. Field. Sparse coding with an overcomplete basis set: A strategy employed by V1? *Vision Research*, 37:3311–3325, 1997.

[20] D. Paliy, V. Katkovnik, R. Bilcu, S. Alenius, and K. Egiazarian. Spatially adaptive color filter array interpolation for noiseless and noisy data. *Int. J. Imag. Sys. Tech.*, 17(3), 2007.

[21] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE T. PAMI*, 12(7):629–639, 1990.

[22] J. Portilla, V. Strela, M. Wainwright, and E. Simoncelli. Image denoising using scale mixtures of Gaussians in the wavelet domain. *IEEE T. IP*, 12(11):1338–1351, 2003.

[23] R. Puetter, T. Gosnell, and A. Yahil. Digital image reconstruction: deblurring and denoising. *Annu. Rev. Astron. Astrophys.*, 43, 2005.

[24] S. Roth and M. J. Black. Fields of experts: A framework for learning image priors. In *Proc. IEEE CVPR*, 2005.

[25] L. Rudin and S. Osher. Total variation based image restoration with free local constraints. In *Proc. ICIP*, 1994.

[26] A. Szlam, M. Maggioni, and R. Coifman. Regularization on graphs with function-adapted diffusion processes. *JMLR*, 2007.

[27] R. Tibshirani. Regression shrinkage and selection via the lasso. *J. Royal. Statist. Soc. B*, 58(1):267–288, 1996.

[28] J. A. Tropp. Algorithms for simultaneous sparse approximation. *Sig. Proc.*, 86:572–602, 2006.

[29] B. Turlach, W. Venables, and S. Wright. Simultaneous variable selection. *Technometrics*, 47(3):349, 2005.

[30] S. Weisberg. *Applied Linear Regression*. Wiley, 1980.

[31] M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *J. Royal. Statist. Soc. B*, 68(1):49–67, 2006.

[32] L. Zhang and X. Wu. Color demosaicking via directional linear minimum mean square-error estimation. *IEEE T. IP*, 14(12):2167–2178, 2005.