Single Image Interpolation via Adaptive Non-Local Sparsity-Based Modeling

Yaniv Romano, Matan Protter, and Michael Elad, *Fellow, IEEE*

***Abstract*— Single image interpolation is a central and extensively studied problem in image processing. A common approach towards the treatment of this problem in recent years is to divide the given image into overlapping patches and process each of them based on a model for natural image patches. Adaptive sparse representation modeling is one such promising image prior, which has been shown to be powerful in filling-in missing pixels in an image. Another force that such algorithms may use is the self-similarity that exists within natural images. Processing groups of related patches together exploits their correspondence, leading often times to improved results. In this paper we propose a novel image interpolation method which combines these two forces – non-local self-similarities and sparse representation modeling. The proposed method is contrasted with competitive and related algorithms, and demonstrated to achieve state-of-the-art results.**

***Index Terms*—Image restoration, super resolution, interpolation, nonlocal similarity, sparse representation, K-SVD.**

I. Introduction

SINGLE[1] image super resolution is the process of reconstructing a High-Resolution (HR) image from an observed Low-Resolution (LR) one. Typical applications include zoom-in of still images in digital cameras, scaling-up an image before printing, interpolating images to adapt their size to high resolution screens and conversion from low-definition to high-definition video. The image interpolation problem, which is the focus of this paper, is a special case of single image super resolution, where the LR image is assumed to be a decimated version (without blurring) of the HR image. This is an inverse problem associated with the linear degradation model

$$\mathbf{y} = \mathbf{U}_{\mathrm{L}}\mathbf{x}, \tag{1}$$

where $\mathbf{y} \in \mathbf{R}^{\mathrm{r}\times\mathrm{c}}$ is the observed LR input image, $\mathbf{x} \in \mathbf{R}^{\mathrm{rL}\times\mathrm{cL}}$ is the unknown HR image, and $\mathbf{U}_{\mathrm{L}}$ is a linear down-sampling operator of size $\mathrm{rc} \times \mathrm{L}^2\mathrm{rc}$, decimating the image by a factor of L along the horizontal and vertical dimensions. Note that $\mathbf{x}$ and $\mathbf{y}$ are held in the above equation as column vectors after lexicographic ordering. A solution to this interpolation problem is an approximation $\hat{\mathbf{x}}$ of the unknown HR image $\mathbf{x}$, that coincides with the values of $\mathbf{y}$ on the coarse grid.

Y. Romano is with the Department of Electrical Engineering, Technion - Israel Institute of Technology, Technion City, Haifa 32000, Israel. E-mail address: yromano@tx.technion.ac.il. M. Protter and M. Elad are with the Department of Computer Science, Technion Israel Institute of Technology, Technion City, Haifa 32000, Israel. E-mail addresses: matanpr@cs.technion.ac.il (M. Protter), elad@cs.technion.ac.il (M. Elad).

[1]This paper concentrates on the single image interpolation problem, which is substantially different from the classical super-resolution task, where a group of images are fused to form a higher resolution result [1]–[3].

The extensive work of single image super resolution divides by the assumptions made on the data creation model — which blur, if at all, is assumed as part of data creation, and whether these measurements are contaminated by noise, and if so, which noise is it. Often times, an algorithm developed for super-resolving an image with one set of assumptions is found less effective or even non-relevant when turning to a different data model assumption. This is especially so when dealing with the image interpolation problem – the case of no blur and no noise. Algorithms tailored for this problem are typically very different from general-purpose super-resolution methods. As said above, our focus is the interpolation problem, and thus this paper will concentrate on the existing and relevant literature in this domain.

Why should one work on the interpolation problem? After all, it seems to be an extremely easy version of the single image super-resolution problem. There are several possible reasons to study this problem, which may explain its popularity in the literature:

1) It comes up in reality: There are situations where the given image is deeply aliased (i.e., no blur) as a measure of preserving its sharpness. If this is the case and the image quality is high (i.e., no noise), dealing with super-resolving such an image calls for an interpolation scheme.
2) Relation to Single-Image-Super-Resolution: Even if the image to be scaled-up is blurred, one can always cast the super-resolution task as a two stage process: interpolate the missing pixels, and then deblur. In such a case, interpolation algorithms address the first stage.
3) Performance bounds: From a theoretical standpoint, the interpolation problem poses a guiding bound on the achievable recovery of missing pixels in an image.
4) Relation to classical interpolators: The foundations (theoretical and practical) of interpolation are well known in the numerical analysis literature. Extending bi-linear, bi-cubic, bi-splines, and other methods to be content adaptive is a fascinating subject, as it sheds light on non-linear extensions of known methods.

The simplest techniques to reconstruct $\mathbf{x}$ are linear interpolators, e.g. bi-linear, bi-cubic, and cubic-spline interpolators [4], [5]. These methods utilize a polynomial approximation to compute each missing pixel from a small local neighborhood of known pixels around it, often generating blurry results and stair-case shaped edges due to their inability to adapt to varying pixel structures in the image. These techniques can be attributed to l_2-based regularization schemes that force some sort of spatial smoothness in an attempt to regularize the inherently ill-posed problem defined in Equation (1).

More complex algorithms use more advanced and more effective priors on the image in order to gain a stabilization of the solution. Two such powerful priors that have widespread use in recent years are (i) non-local proximity and (ii) Sparse-Land modeling. The first assumes that a given patch may find similar patches within its spatial surroundings, and these could be used to help the recovery process. Several works leverage this assumption to solve various image restoration problems, e.g. denoising [6]–[8], deblurring [9], [10] and super-resolution [1], [3], [11], [12]. On the other hand, the Sparse-Land model, as introduced in [13], [14] assumes that each patch from the image can be well represented using a linear combination of a few elements (called "atoms") from a basis set (called "dictionary"). Put differently, each patch $\mathbf{x}_i$ in the original image is considered to be generated by $\mathbf{x}_i = \mathbf{D}\alpha_i$, where $\mathbf{D} \in \mathbf{R}^{n \times m}$ is a dictionary composed of (possibly $m \geq n$) atoms, where $\alpha_i \in \mathbf{R}^m$ is a sparse (mostly zero) vector of coefficients. Therefore, sparsity-based restoration algorithms seek a dictionary and sparse representations that bring the corrupted\degraded patch to be as close as possible to the original one. In other words, the first force seeks to exploit relations between different patches while the second concentrates on the redundancy that exists within the treated patch.

Recent papers, e.g. NARM [15] and LSSC [16], combine a sparsity-based model with the non-local similarity of natural image patches and achieve impressive restoration results. NARM is an image interpolation method that embeds a non-local autoregressive model (i.e. connecting a missing pixel with its nonlocal neighbors) into the data fidelity term, combined with a conventional sparse representation minimization term. Their method also exploits the nonlocal redundancy in order to further regularize the overall problem. NARM divides the image to clusters of patches and uses a local PCA dictionary learning per each cluster, in order to adaptively and sparsely represent the patches. LSSC achieves state-of-the-art results in image denoising and demosaicing by jointly decomposing groups of similar patches with a learned dictionary. Their method uses a simultaneous sparse coding [17] to impose the use of the same dictionary atoms in the sparse decomposition of similar patches.

Inspired by the promising results of the above-mentioned contributions ([15] and [16]), we propose a two-stage image interpolation scheme based on an adaptive non-local sparsity prior. The proposed method starts with an initial cubic-spline interpolation of the HR image. In the first stage we aim at recovering regions that fit with the non-local self-similarity assumption. We iteratively produce a rough approximation of the HR image, accompanied by a learned dictionary. In the second stage we obtain the final interpolated result by refining the previous HR approximation using the first-stage's adapted dictionary and a non-local sparsity model. In order to ensure that the influence of known pixels will be more pronounced compared to the approximated (actually unknown) ones, in both stages of the algorithm we use an element-wise weighted variant of the Simultaneous Orthogonal Matching Pursuit [17] and the K-SVD [18] algorithms. Note that the algorithm we introduce in this paper can be easily extended to cope with the case of interpolation under noise, i.e. when Equation (1) is replaced by $\mathbf{y} = \mathbf{U}_{\mathrm{L}}\mathbf{x} + \mathbf{v}$, where $\mathbf{v}$ is white additive Gaussian noise. However, in order to remain consistent with the work in [15], [19]–[21], we will discuss the noiseless case only.

The current state-of-the-art is the recently published NARM method [15], showing impressive results. However, based on 18 test images, our proposed method outperforms NARM by 0.11dB and 0.23dB on average for interpolation by factors of 2 and 3, respectively. Similarly to NARM, we rely on the sparsity model and the self-similarity assumption, but unlike NARM:

1) We do not assume that each patch can be represented as a linear combination of its similar patches. NARM requires that the representation of each patch shall be close (in terms of l_2 norm) to a linear combination of its similar patches representations. When this assumption holds – the recovery is indeed impressive, but there are patches that do not align with this assumption and forcing them to fit this requirement becomes harmful.
2) We suggest training a redundant dictionary, exploiting the benefit of the large number of examples, while NARM divides the image patches into 60 clusters and trains a PCA dictionary per each of these clusters. Besides the fact that the clustering is computationally "expensive" and very difficult to parallelize, NARM assumes that each cluster would contain enough examples (which are crucial for obtaining a good sub-dictionary), with the risks that (i) patches may accidentally be assigned to an inappropriate cluster (e.g. due to aliasing), and (ii) there is no suitable cluster per each patch.

To conclude, the proposed method achieves competitive and even better results than NARM without limiting the algorithm to strictly fit the non-local proximity assumption. This is attributed to the proposed stable sparse-coding and the effective K-SVD dictionary learning.

This paper is organized as follows: In Section II we provide brief background material on sparse representation and dictionary learning, for the sake of completeness of the discussion. In Section III we introduce our novel image interpolation algorithm and discuss its relation to previous works. Experiments are brought in Section IV, showing that the proposed method outperforms the state-of-the-art algorithms for the single-image interpolation task. Conclusions and future research directions are drawn in Section V.

II. Background: Sparse Representations and Dictionary Learning

The idea behind sparse and redundant representation modeling is the introduction of a new transform of a signal $\mathbf{x}_i \in \mathbf{R}^n$ to a representation $\alpha_i \in \mathbf{R}^m$ where $m > n$ (thus leading to redundancy), such that the obtained representation is the sparsest possible. This semi-linear transform assumes that a signal $\mathbf{x}_i$ can be described as $\mathbf{x}_i = \mathbf{D}\alpha_i$, thus implying that the inverse transform from the representation to the signal is indeed linear. On the other hand, the forward transform, i.e., the recovery of α_i from $\mathbf{x}_i$ (called "sparse-coding"), is obtained by

$$\hat{\alpha}_i = \min_{\alpha_i} \ \|\alpha_i\|_0 \ \text{ s.t. } \ \|\mathbf{D}\alpha_i - \mathbf{x}_i\|_2^2 \leq \epsilon^2, \tag{2}$$

where the notation $\|\alpha\|_0$ stands for the count of the nonzero entries in α, and ϵ is an a-priori error threshold. For pure transformation, ϵ should be zero, implying that $\mathbf{D}\alpha_i = \mathbf{x}_i$. When $\mathbf{x}_i$ is believed to be noisy, the very same expression above serves as a denoising process, since $\mathbf{D}\hat{\alpha}_i$ could be interpreted as the cleaned version of $\mathbf{x}_i$, and in such a case, ϵ is set as the noise level. Since Equation (2) is an NP-hard problem, the representation is approximated by a pursuit algorithm, e.g. MP [22], OMP [23], SOMP [17], basis pursuit [24], and others [14].

Naturally, adapting the dictionary $\mathbf{D}$ to a set of signals $\{\mathbf{x}_i\}_{i=1}^N$ results in a sparser representation than the one which would be based on a pre-chosen dictionary. For example, given the representations of these signals, $\{\alpha_i\}_{i=1}^N$, obtained using a dictionary $\mathbf{D}_0$, the MOD and K-SVD dictionary learning algorithms [18], [25], [26] adapt $\mathbf{D}$ to the signals by approximating the solution of the minimization problem

$$\min_{\mathbf{D},\{\alpha_i\}_{i=1}^N} \sum_i \|\mathbf{D}\alpha_i - \mathbf{x}_i\|_2^2 \quad \text{s.t.} \quad \forall i \ \ Supp\{\alpha_i\} = Supp\{\hat{\alpha}_i\}, \tag{3}$$

where $Supp\{\hat{\alpha}_i\}$ are the supports (the indices of the non-zero rows in $\hat{\alpha}_i$). This process is typically iterated several times, performing pursuit to update the representations, followed by updating the dictionary (and the non-zero values in the representations). Once $\mathbf{D}$ and $\hat{\alpha}_i$ are computed, each signal is approximated by $\hat{\mathbf{x}}_i = \mathbf{D}\hat{\alpha}_i$.

Since dictionary learning is limited in handling low-dimensional signals, a treatment of an image is typically done by breaking it into small overlapping patches, forcing each to comply with the sparse representation model. In this paper we rely on the ideas of the K-SVD denoising algorithm [27], which divides the noisy image into $\sqrt{n} \times \sqrt{n}$ overlapping patches. Then the algorithm performs iterations of: (i) sparse-coding using OMP on all these patches, followed by (ii) a dictionary-update applied as a sequence of a rank-1 approximation problems that update each dictionary atom and the representation coefficients using it. Finally, the denoised image is obtained by averaging the cleaned patches and the noisy image. All this process is essentially a numerical scheme for approximating the solution of the problem

$$\hat{\mathbf{x}}, \hat{\mathbf{D}}, \{\hat{\alpha}_i\}_{i=1}^N = \min_{\mathbf{x},\mathbf{D},\{\alpha_i\}_{i=1}^N} \frac{1}{2}\|\mathbf{x} - \mathbf{y}\|_2^2 + \lambda \sum_{i=1}^N \|\alpha_i\|_0 \quad \text{s.t.} \quad \|\mathbf{D}\alpha_i - \mathbf{R}_i\mathbf{x}\|_2^2 \leq \epsilon^2, \tag{4}$$

where N is the number of image patches, $\hat{\mathbf{x}}$ is the denoised image, $\hat{\mathbf{D}}$ is the trained overcomplete dictionary and $\mathbf{R}_i$ is a matrix that extracts the i-th patch from the image. The first term of Equation (4) demands a proximity between the noisy image $\mathbf{y}$ and its denoised version $\mathbf{x}$. The second and third terms demand that every patch $\mathbf{R}_i\mathbf{x} = \mathbf{x}_i$ is represented sparsely up to a bounded error, with respect to a dictionary $\mathbf{D}$.

III. The Proposed Algorithm

A. The General Objective Function

The single-image interpolation problem can be viewed as the need to fill-in a regular pattern of missing pixels in a desired HR image. In terms of a sparsity-based approach, the HR patches are characterized by having sparse representations with respect to a learned dictionary, and this should be assessed while relying (mainly) on the known pixels. In the spirit of the K-SVD denoising formulation for images, as described in Equation (4), the interpolation solution can be cast as the outcome of the minimization problem

$$\hat{\mathbf{x}}, \hat{\mathbf{D}}, \{\alpha_i^{sp}\}_{i=1}^N = \min_{\mathbf{x},\mathbf{D},\{\alpha_i\}_{i=1}^N} \sum_{i=1}^N \|\alpha_i\|_0 \quad \text{s.t.} \ \ \mathbf{y} = \mathbf{U}_{\mathrm{L}}\mathbf{x} \text{ and } \forall i \ \ \|\mathbf{D}\alpha_i - \mathbf{R}_i\mathbf{x}\|_{\tilde{\mathrm{W}}_i}^2 \leq \epsilon_i^2, \tag{5}$$

where $\hat{\mathbf{x}}$ is an approximation of the HR image and the constraint $\mathbf{y} = \mathbf{U}_{\mathrm{L}}\mathbf{x}$ is simply the relation to the measurements as given in Equation (1). $\tilde{\mathrm{W}}_i \in \mathbf{R}^{n \times n}$ is a diagonal weighting matrix, which sets high weights for known pixels in $\mathbf{R}_i\mathbf{x} = \mathbf{x}_i$, and low ones for the missing\approximated pixels. $\epsilon_i^2 = c_\epsilon \|\tilde{\mathrm{W}}_i\|_2^2$ are error thresholds, where c_ϵ is some constant. The essence of $\tilde{\mathrm{W}}_i$ is to ensure that the influence of known pixels will be more pronounced compared to the approximated ones[2]. In practice, the representations $\{\alpha_i^{sp}\}_{i=1}^N$ are approximated by a weighted variant of the Orthogonal Matching Pursuit (OMP), the dictionary $\mathbf{D}$ is updated using a weighted variant of the K-SVD [18], and once $\mathbf{D}$ and $\{\alpha_i^{sp}\}_{i=1}^N$ are fixed, the output HR image $\hat{\mathbf{x}}$ is computed by solving the constrained minimization problem

$$\min_{\mathbf{x}} \ \sum_{i=1}^N \|\mathbf{D}\alpha_i^{sp} - \mathbf{R}_i\mathbf{x}\|_{\tilde{\mathrm{W}}_i}^2 \ \text{ s.t. } \ \mathbf{y} = \mathbf{U}_{\mathrm{L}}\mathbf{x}. \tag{6}$$

This optimization problem leads to a simple solution, in which the approximated HR patches are averaged followed by a simple projection of the known pixels in $\mathbf{x}$ on the outcome. Appendix A describes the closed-form solution for this problem via the Lagrange multipliers method.

Intuitively, if we could improve the sparse-coding step in Equation (5), this may lead to better results. The OMP processes each of the patches separately and disregards inter-relations that may exist between them. As consequence, the recovery of similar patches may be different despite their similar content. Grouping similar patches together and applying joint sparse-coding is a powerful technique which leverages the assumption that there are several appearances of the same image content and those could help each other somehow. In the case of denoising (as suggested by LSSC [16]), similar patches contain similar image content but with different noise realization. In the case of interpolation, the aliasing can be considered as structured noise [28], and it varies between similar patches extracted from different locations, thus enriching our set.

[2]In Section IV-C we provide an experiment along with a brief discussion regarding the need for these weights.

The proposed algorithm is based on the observation that the more known pixels within a patch, the better its reconstruction. Performing group decomposition serves this goal; each patch in the group contributes its known pixels in finding the atoms decomposition for the whole group. Following the LSSC algorithm [16], we apply the simultaneous sparse-coding (SOMP) method, which forces similar patches to have similar decompositions in terms of the chosen atoms in the pursuit. SOMP is a variant of the OMP; it is an iterative greedy algorithm that finds one atom at a time for the representation of the group signals. Given a group of similar patches and a dictionary, in the first iteration SOMP finds the atom that best fits the group of weighted patches. In the next iterations, given the previously found atoms, it finds the next one that minimizes the sum of weighted l_2 norm of the residuals. Note that per iteration, the coefficients that correspond to the chosen atoms are evaluated by solving weighted least-squares.

To summarize, inspired by LSSC we suggest a strengthened version of Equation (5):

$$\hat{\mathbf{x}},\hat{\mathbf{D}},\{\mathrm{A}_i^{sp}\}_{i=1}^N = \min_{\mathbf{x},\mathbf{D},\{\mathrm{A}_i\}_{i=1}^N} \sum_{i=1}^N \|\mathrm{A}_i\|_{0,\infty} \quad (7)$$
$$\text{s.t. } \mathbf{y}=\mathbf{U}_{\mathrm{L}}\mathbf{x} \text{ and } \forall i \sum_{j\in S_i} \|\mathbf{D}(\mathrm{A}_i)_j - \mathbf{R}_j\mathbf{x}\|^2_{\mathrm{W}_{i,j}} \le \mathrm{T}_i,$$

where $S_i = \{j|1 \le j \le N, \text{ and } \|\mathbf{x}_i - \mathbf{x}_j\|_1 \le c_d\}$ is a set of the similar patches to $\mathbf{x}_i$, where c_d is a fixed threshold. $\mathrm{W}_{i,j} = \tilde{\mathrm{W}}_j \exp(\|\mathbf{x}_i - \mathbf{x}_j\|_1/c_w)$ ensures that the closer $\mathbf{x}_j$ patch to $\mathbf{x}_i$, the higher the influence of it. $\mathrm{A}_i \in \mathbf{R}^{m\times|S_i|}$ is a matrix, where the j-th column $(\mathrm{A}_i)_j$ is the representation of the j-th patch in S_i. The notation $\|\mathrm{A}_i\|_{0,\infty}$ counts the number of non-zero rows in A_i, and $\mathrm{T}_i = \sum_{j\in S_i} c_\epsilon \|\mathrm{W}_{i,j}\|_2^2$ are the error thresholds. Similar to Equation (6), the reconstruction of the HR image is obtained by solving

$$\min_{\mathbf{x}} \sum_{i=1}^N \sum_{j\in S_i} \|\mathbf{D}(\mathrm{A}_i^{sp})_j - \mathbf{R}_j\mathbf{x}\|^2_{\mathrm{W}_{i,j}} \quad \text{s.t.} \quad \mathbf{y}=\mathbf{U}_{\mathrm{L}}\mathbf{x}. \quad (8)$$

Although the non-local approach strengthens the sparsity model, there are still some potential instabilities in the sparse coding stage. These are the result of the ratio between the overall number of pixels in the HR image and the known ones. This ratio equals $\frac{1}{\mathrm{L}^2}$, where L is the up-scaling factor along the horizontal and vertical dimensions. For example, when $\mathrm{L}=3$, the number of known pixels within a 7×7 patch varies from 4 to 9 out of 49 pixels. Fig. 1 demonstrates this for $\mathrm{L}=2$ and 3×3 patch size. Assigning a low weight to unknown and approximated pixels (e.g. 0.01 in our experiments) raises the instability of the joint weighted sparse coding. Thus, in order to further stabilize the sparse-coding step we suggest modifying the formulation in Equation (7) to

$$\min_{\mathbf{x},\{\alpha_i^r\}_{i=1}^N\{\mathrm{A}_i^{sp}\}_{i=1}^N} \sum_{i=1}^N \|[\alpha_i^r\ \mathrm{A}_i^{sp}]\|_{0,\infty} \quad (9)$$
$$\text{s.t. } \mathbf{y}=\mathbf{U}_{\mathrm{L}}\mathbf{x}$$
$$\forall i\ \|\mathbf{D}\alpha_i^r - \mathbf{R}_i\hat{\mathbf{x}}^{est}\|_2^2 + \sum_{j\in S_i} \|\mathbf{D}(\mathrm{A}_i^{sp})_j - \mathbf{R}_j\mathbf{x}\|^2_{\mathrm{W}_{i,j}} \le \mathrm{T}_i,$$

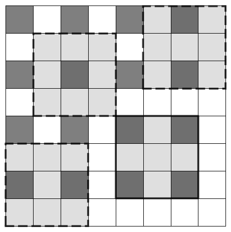


Fig. 1. Demonstration of a "strong" patch (solid line) and "weak" patches (dash line) for L = 2 and 3 × 3 patch size. The number of known pixels (dark points) within a "strong" patch is 4, and within a "weak" patch is 1 or 2.

where $\hat{\mathbf{x}}^{est}$ is an initial estimation of the HR image (e.g. a cubic-spline approximation), $\alpha_i^r \in \mathbf{R}^m$ are the representations of patches $\hat{\mathbf{x}}_i$ that are computed without discrimination between known and unknown pixels, $\|[\alpha_i^r\ \mathrm{A}_i^{sp}]\|_{0,\infty}$ counts the number of non-zero rows in the matrix $[\alpha_i^r\ \mathrm{A}_i^{sp}]$, and $\mathrm{T}_i = c_\epsilon n + \sum_{j\in S_i} c_\epsilon\|\mathrm{W}_{i,j}\|_2^2$ are the error thresholds. In general, sparse coding based on the l_2 norm (i.e. a non-weighted patch) is more stable than the weighted l_2 norm since it finds the atoms according to higher number of pixels. Following this observation, the essence of α_i^r is to stabilize the sparse-coding step (due to the non-weighted l_2 norm), and promote their preferred set of atoms to the group they serve[3]. These representations are ignored in the reconstruction step, i.e., reconstruction of $\mathbf{x}$ remains as described in Equation (8). Note that α_i^r is not included in A_i^{sp}; given the joint support, α_i^r is the outcome of a least-squares while each column in A_i^{sp} is the outcome of a weighted least-squares. Once $\{[\alpha_i^r\ \mathrm{A}_i^{sp}]\}_{i=1}^N$ are computed, we update the dictionary by a weighted variant of the K-SVD that approximates the solution to the problem

$$\min_{\hat{\mathbf{D}},\{\alpha_i\}_{i=1}^N,\{\mathrm{A}_i\}_{i=1}^N} \sum_{i=1}^N \|\hat{\mathbf{D}}\alpha_i - \mathbf{R}_i\hat{\mathbf{x}}\|_2^2 \quad (10)$$
$$+\sum_{i=1}^N\sum_{j\in S_i} \|\hat{\mathbf{D}}(\mathrm{A}_i)_j - \mathbf{R}_j\hat{\mathbf{x}}\|^2_{\mathrm{W}_{i,j}}$$
$$\text{s.t.} \quad \forall i\ Supp\{[\alpha_i\ \mathrm{A}_i]\} = Supp\{[\alpha_i^r\ \mathrm{A}_i^{sp}]\},$$

where $Supp\{[\alpha_i^r\ \mathrm{A}_i^{sp}]\}$ are the supports of the indices of the non-zero rows in $[\alpha_i^r\ \mathrm{A}_i^{sp}]$ (the joint support) that were computed in Equation (9).

The above framework is the basis of the proposed two-stage image interpolation algorithm; our proposed scheme starts with an initial cubic-spline approximation of the HR image. In the first stage we iteratively produce a rough approximation of the HR image accompanied by a learned dictionary. In the second stage, based on the first-stage's HR result, we apply a joint weighted sparse-coding to represent the HR image using the first-stage's dictionary. A detailed description of these steps now follows.

[3]In terms of PSNR and based on the test images, excluding α_i^r from the penalty function degrades the restoration performance. The average differences between the original proposed two-stage algorithm and excluding α_i^r from it are 0.39 dB and 0.38 dB for interpolation by factor of 2 and 3, respectively.

```
Algorithm 1 : Description of the first stage of the image interpolation algorithm.
Initialization Step:
1: Set $\hat{\mathbf{x}}^{est}$ = cubic-spline interpolation of $\mathbf{y}$ that satisfies $\mathbf{y} = \mathbf{U}_{\mathrm{L}}\mathbf{x}$.
2: Per each patch $\hat{\mathbf{x}}_i^{est} \in \mathbf{R}^n$, create a diagonal weighting matrix $\tilde{\mathrm{W}}_i \in \mathbf{R}^{n\times n}$, which sets high weights to known pixels and
   low ones to the missing\approximated pixels in $\hat{\mathbf{x}}_i^{est}$.
3: Set $\mathbf{D} \in \mathbf{R}^{n\times m}$ = overcomplete DCT dictionary.
Repeat several times:
1: Grouping Step: Per each patch $\hat{\mathbf{x}}_i^{est}$ (i) find a set $S_i^{strong}$ that contains the indices of up to $K$ most similar "strong"
   patches to it within a window of size $h \times h$ pixels, sorted according to their $l_1$ distance (from low to high), (ii) set
   $\forall j \in S_i^{strong} \quad \mathrm{W}_{i,j} = \tilde{\mathrm{W}}_j \exp\left(\|\mathbf{x}_i - \mathbf{x}_j\|_1 / c_w\right)$.
2: Joint Weighted Sparse-Coding Step: Solve
   $[\{\alpha_i^r\}_{i=1}^N, \{\mathrm{A}_i^{sp}\}_{i=1}^N] := \text{argmin} \sum_{i=1}^{N} \|[\alpha_i^r \ \mathrm{A}_i^{sp}]\|_{0,\infty} \ \text{s.t.} \ \forall i \ \|\mathbf{D}\alpha_i^r - \mathbf{R}_i\hat{\mathbf{x}}^{est}\|_2^2 + \sum_{j\in S_i^{strong}} \|\mathbf{D}(\mathrm{A}_i^{sp})_j - \mathbf{R}_j\hat{\mathbf{x}}^{est}\|_{\mathrm{W}_{i,j}}^2 \leq \mathrm{T}_i$,
   using a joint element-wise weighted SOMP algorithm (which we replace with a batch-SOMP without weights, followed
   by a least-squares step that take the weights into account).
3: Dictionary Update Step: Update the dictionary using an element-wise weighted variant of the K-SVD, which approximates
   $[\hat{\mathbf{D}}, \{\alpha_i\}_{i=1}^N, \{\mathrm{A}_i\}_{i=1}^N] := \text{argmin} \sum_{i=1}^{N} \|\mathbf{D}\alpha_i - \mathbf{R}_i\hat{\mathbf{x}}\|_2^2 + \sum_{i=1}^{N} \sum_{j\in S_i^{strong}} \|\mathbf{D}(\mathrm{A}_i)_j - \mathbf{R}_j\hat{\mathbf{x}}\|_{\mathrm{W}_{i,j}}^2$
   $\text{s.t.} \quad \forall i \ Supp\{[\alpha_i \ \mathrm{A}_i]\} = Supp\{[\alpha_i^r \ \mathrm{A}_i^{sp}]\}$.
4: Aggregation Step: Compute the approximation of the HR image
   $\hat{\mathbf{x}} := \text{argmin} \sum_{i=1}^{N} \|\mathbf{D}(\mathrm{A}_i^{sp})_1 - \mathbf{R}_i\mathbf{x}\|_{\mathrm{W}_{i,1}}^2 \ \text{s.t.} \ \mathbf{y} = \mathbf{U}_{\mathrm{L}}\mathbf{x}$,
   and set $\hat{\mathbf{x}}^{est} \leftarrow \hat{\mathbf{x}}$, $\mathbf{D} \leftarrow \hat{\mathbf{D}}$.
Output:
1: $\hat{\mathbf{x}}^{est}$ – rough approximation of the HR image.
2: $\hat{\mathbf{D}}$ – an overcomplete learned dictionary.
```

B. The First Stage of the Proposed Algorithm

The essence of the first-stage of our algorithm is to efficiently recover regions that fit the non-local self-similarity assumption, such as flat or smooth regions (e.g. sky, sand, walls, etc.), and repetitive or continues edges or textures (e.g. stockades, columns, bricks, etc.). This is achieved by utilizing the self-similarities between the patches within the image. Recall that grouping similar patches together in order to find their joint decomposition has a major advantage in filling-in missing pixels. The main contribution of each patch in finding the joint support comes from its small number of known pixels. Joining similar patches together increases the number of known pixels which are crucial for the success of the sparse-coding step.

Within an HR patch, the number of known pixels and their locations varies according to its location in the image; there are "strong" patches, i.e. patches that contain the maximal number of known pixels, and "weak" ones, i.e. patches that contain a lower number of known pixels, see Fig.1 for a visual demonstration of these types of patches. It is natural to expect that the restoration of "strong" patches would be more accurate than the "weak" ones, since they have more known pixels. Motivated by this assumption, we define per each patch of $\hat{\mathbf{x}}^{est}$ ("strong" or "weak") a set S_i^{strong} that contains the indices of up to K most similar "strong" patches to $\hat{\mathbf{x}}_i^{est}$, sorted according to their l_1 distance (from low to high), requiring this distance to be at most c_d.

Once $\{S_i^{strong}\}_{i=1}^N$ are computed, in the first stage we solve the minimization problems defined in (9) and (10) with one major difference – replace $\{S_i\}_{i=1}^N$ sets with $\{S_i^{strong}\}_{i=1}^N$. Armed with $\mathbf{D}$ and $\{\mathrm{A}_i^{sp}\}_{i=1}^N$, the reconstruction of the HR image is obtained by[4]

$$\min_{\mathbf{x}} \sum_{i=1}^{N} \|\mathbf{D}(\mathrm{A}_i^{sp})_1 - \mathbf{R}_i\mathbf{x}\|_{\mathrm{W}_{i,1}}^2 \ \text{s.t.} \ \mathbf{y} = \mathbf{U}_{\mathrm{L}}\mathbf{x}, \qquad (11)$$

where $(\mathrm{A}_i^{sp})_1$ is the representation that corresponds to the first index in S_i^{strong} set, i.e. the representation of the most similar "strong" patch to $\hat{\mathbf{x}}_i^{est}$. We draw the reader's attention to the resemblance and the differences between this and the equation given in (8) – here we use only one representation – the one corresponding to the closest strong patch, while in (8) we recover the image based on all the representations together.

Notice that when $\hat{\mathbf{x}}_i^{est}$ is a "weak" patch, S_i^{strong} does not contain its index, i.e., A_i^{sp} does not include the representation

[4]Referring to Equation (11), we found that using the conventional l_2 norm instead of the weighted one performs slightly better.

Algorithm 2 : Description of the second stage of the image interpolation algorithm.

```
Initialization Step:
  1: Set $\hat{\mathbf{x}}^{est}$ = the first-stage approximation of the HR image.
  2: Set $\hat{\mathbf{D}} \in \mathbf{R}^{n \times m}$ = the first-stage's learned dictionary.
  3: $\tilde{\mathrm{W}}_i \in \mathbf{R}^{n \times n}$ = a diagonal weighting matrix, that corresponds to the $i$-th patch, with higher weights for unknown pixels compared to the first stage.
Grouping Step:
  Per each patch $\hat{\mathbf{x}}_i^{est}$ (i) compute a set $S_i$ that contains the indices of up to $K$ most similar patches to $\hat{\mathbf{x}}_i^{est}$ within a window of size $h \times h$ pixels, (ii) set $\forall j \in S_i \quad \mathrm{W}_{i,j} = \tilde{\mathrm{W}}_j \exp(\|\mathbf{x}_i - \mathbf{x}_j\|_1 / c_w)$.
Joint Weighted Sparse-Coding Step:
  Solve
  $$[\{\alpha_i^r\}_{i=1}^N, \{\mathrm{A}_i^{sp}\}_{i=1}^N] := \operatorname{argmin} \sum_{i=1}^N \|[\alpha_i^r \ \mathrm{A}_i^{sp}]\|_{0,\infty} \text{ s.t. } \forall i \ \|\hat{\mathbf{D}}\alpha_i^r - \mathbf{R}_i \hat{\mathbf{x}}^{est}\|_2^2 + \sum_{j \in S_i} \|\hat{\mathbf{D}}(\mathrm{A}_i^{sp})_j - \mathbf{R}_j \hat{\mathbf{x}}^{est}\|_{\mathrm{W}_{i,j}}^2 \le \mathrm{T}_i,$$
  using a joint element-wise weighted SOMP algorithm (which we replace with a batch-SOMP without weights, followed by a least-squares step that take the weights into account).
Aggregation Step:
  Compute the approximation of the HR image:
  $$\hat{\mathbf{x}}^{final} := \operatorname{argmin} \sum_{i=1}^N \sum_{j \in S_i} \|\hat{\mathbf{D}}(\mathrm{A}_i^{sp})_j - \mathbf{R}_j \mathbf{x}\|_{\mathrm{W}_{i,j}}^2 \text{ s.t. } \mathbf{y} = \mathbf{U}_{\mathrm{L}}\mathbf{x}.$$
Output:
  $\hat{\mathbf{x}}^{final}$ – the final approximation of the HR image.
```

of the weighted "weak" i-th patch at all (as opposed to the case where $\hat{\mathbf{x}}_i^{est}$ is a "strong" one). It is important to emphasize that the approach taken here is very different from a "simple" replacement of a "weak" patch with its most similar "strong" one. The non-weighted term $\|\mathbf{D}\alpha_i^r - \mathbf{R}_i\hat{\mathbf{x}}^{est}\|_2^2$ in Equation (9) has a significant influence on deciding which atoms will be chosen to represent the $\{S_i^{strong}\}_{i=1}^N$ patches. Thereby the properties of the "weak" patch leak into the "strong" one owing to the joint decomposition and the fact that there are infinitely many ways to fill-in the missing pixels within a patch.

Notice that we suggest reconstructing the HR image based on $(\mathrm{A}_i^{sp})_1$ although α_i^r is available due to the following[5]:

1) The restoration of a "weak" reference patch based on α_i^r cancels the explicit exploitation of the self-similarity assumption.
2) When the reference patch is a "strong" one, $(\mathrm{A}_i^{sp})_1$ leads to a patch reconstruction that is close to the known pixels, while the unknown ones are naturally interpolated. The interpolation is done by a linear combination of the chosen HR dictionary atoms, which in turn are updated iteratively to obtain better and better estimation of these unknown pixels. On the other hand, α_i^r is the representation of the non-weighted patch; therefore it leads to a patch reconstruction that is close to the whole previously interpolated patch.

[5]In terms of PSNR and based on the test images, reconstructing the image based on α_i^r instead of $(\mathrm{A}_i^{sp})_1$ degrades the restoration performance. The average differences between the original proposed two-stage algorithm and reconstructing the image based on α_i^r are 0.28 dB and 0.31 dB for interpolation by factor 2 and 3, respectively.

While the joint element-wise weighted sparse-coding leads to an effective reconstruction, we found it to be computationally demanding mostly due to the inability to use a batch implementation [29]. This batch method reduces the sparse-coding complexity by relying on the fact that large number of (non-weighted) signals are coded over the same dictionary. In this work we propose a technique that leverages the batch SOMP results and approximates the joint weighted sparse-coding representations. This technique is composed of two stages: (i) compute the joint supports $\{Supp\{\mathrm{A}_i^{batch}\}\}_{i=1}^N$ using the non-weighted batch SOMP implementation, and (ii) given the supports, approximate $\{\mathrm{A}_i^{sp}\}_{i=1}^N$ representations by solving the weighted least-squares problem

$$\forall i, j \in S_i^{strong} \ \ (\hat{\mathrm{A}}_i^{sp})_j = \min_z \|\mathbf{D}_{s_i} z - \mathbf{R}_j \hat{\mathbf{x}}^{est}\|_{\mathrm{W}_{i,j}}^2, \quad (12)$$

where $(\hat{\mathrm{A}}_i^{sp})_j$ are the approximated representations, and $\mathbf{D}_{s_i}$ is a matrix where its columns are atoms from $\mathbf{D}$ that correspond to the nonzero entries of A_i^{batch}.

To conclude, the first-stage is composed of three steps which are repeated iteratively: joint weighted sparse–coding, weighted dictionary learning and aggregation of the approximated HR patches by exploiting the "strong" patches. A pseudo-code description of the proposed iterative first-stage is given in Algorithm 1.

C. The Second Stage of the Proposed Algorithm

The goal of the second stage is to generate a robust and reliable HR image by refining $\hat{\mathbf{x}}^{est}$ (the first-stage approximation) using again the non-local sparsity model, but with some important differences:

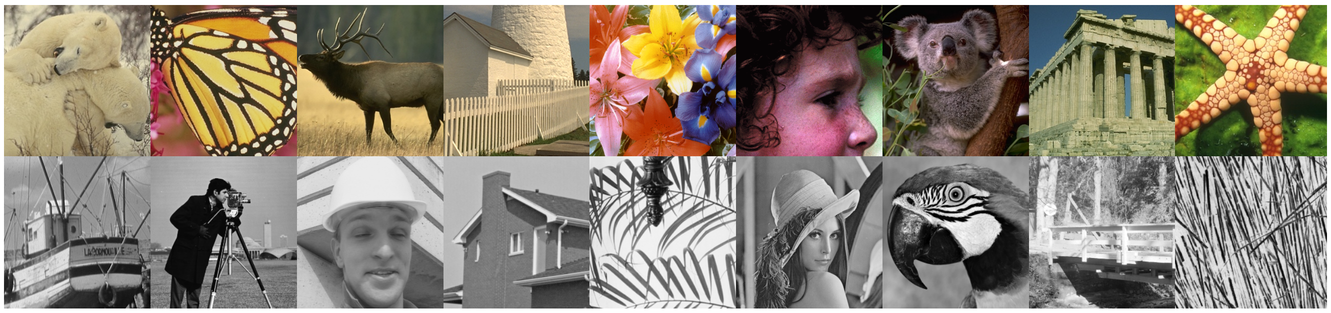


Fig. 2. Test images from left to right and top to bottom: Bears, Butterfly, Elk, Fence, Flowers, Girl, Koala, Parthenon, Starfish, Boat, Cameraman, Foreman, House, Leaves, Lena, Parrot, Stream and Texture.

TABLE I
PARAMETERS USED PER EACH SCALING FACTOR AND STAGE

Scaling factor	Stage	$\sqrt{n}$	m	K	h	c_ϵ	c_d	c_w	Weight for known pixels	Weight for unknown pixels	Number of iterations
2	1	7	256	8	37	0.01	$60n$	$0.7 \cdot 60n$	1	0.01	12
	2	7	256	9	37	0.01	$30n$	$0.6 \cdot 30n$	1	0.05	1
3	1	7	256	7	37	0.01	$30n$	$30n$	1	0.01	12
	2	7	256	7	37	0.01	$10n$	$10n$	1	0.1	1

1) As apposed to the previous stage, we no longer restrict the sets to use strong patches, and use all kinds equally.
2) In the second stage, we can assign higher weights to the approximated pixels, as they are already of better quality after the first stage.
3) Forcing the self-similarity property on patches that are not similar to others can be harmful. In order to obtain a reliable estimation, the use of self-similarities in the second stage is made more conservative than in the first-stage by reducing c_d.
4) In the second stage there is no patch replacement in the reconstruction step. In practice, given $\hat{\mathbf{x}}^{est}$ and $\hat{\mathbf{D}}$, we represent $\hat{\mathbf{x}}^{est}$ patches as described in (9) and finally we reconstruct the HR image as described in (8)[6].

To summarize, in this stage we exploit the first-stage effective recovery of regions that fit the non-local self-similarity prior, and the ability of the sparsity model to fill-in missing pixels within the patches. A pseudo-code description of the proposed second-stage is given in Algorithm 2.

IV. Experimental Results

In this section, detailed results of the proposed algorithm are presented for the images Bears, Boat, Butterfly, Cameraman, Elk, Fence, Flowers, Foreman, Girl, House, Koala, Leaves, Lena, Parthenon, Parrot, Starfish, Stream and Texture (see Fig. 2). All these are commonly used in other related publications, which enables a fair comparison of results. All the tests in this section are generated by decimating the input HR image by factor of L in each axis We tested the proposed algorithm for two scaling factors, $\text{L} = 2$ and $\text{L} = 3$. The interpolation flow for a color image is: (i) first, convert the image to YCbCr color space, (ii) interpolate the luminance channel (i.e. Y) using the proposed algorithm and interpolate the chromatic channels (i.e. Cb and Cr) using the bi-cubic method, and (iii) finalize by converting the interpolated channels back to the RGB color space. We evaluate the interpolation performance using the Peak Signal to Noise Ratio (PSNR), defined as $20\log_{10}(\frac{255}{\sqrt{\text{MSE}}})$, where MSE is the mean-squared-error between the luminance channel of the original HR image and the recovered one.

We ran many tests to tune the various parameters of the proposed algorithm. These tests have resulted in a selection

TABLE II
INTERPOLATION RESULTS [PSNR] FOR UP-SCALING BY FACTOR OF $\text{L} = 2$ AND $\text{L} = 3$. WE COMPARE BETWEEN THE GENERAL OBJECTIVE FUNCTION (SECTION III-A), THE FIRST-STAGE (SECTION III-B) AND THE SECOND-STAGE (SECTION III-C) OF THE PROPOSED ALGORITHM. NOTICE THAT IN ALL CASES THE PERFORMANCE OF SECOND-STAGE IS EQUAL OR BETTER THAN THE FIRST-STAGE, AND OVERCOMES THE GENERAL OBJECTIVE FUNCTION. THE BEST RESULTS IN EACH LINE ARE HIGHLIGHTED.

Image	L = 2			L = 3		
	General	**1-Stage**	**2-Stage**	**General**	**1-Stage**	**2-Stage**
Bears	28.42	28.25	**28.57**	25.30	**25.71**	**25.71**
Boat	29.44	29.62	**30.18**	26.15	26.72	**26.84**
Butterfly	28.88	28.74	**29.68**	23.88	25.07	**25.27**
Cameraman	25.88	26.31	**26.59**	22.63	**23.27**	**23.27**
Elk	32.33	33.19	**33.83**	28.30	29.28	**29.33**
Fence	24.68	24.88	**25.03**	20.74	**20.90**	**20.90**
Flowers	28.57	28.71	**28.81**	25.85	**26.38**	**26.38**
Foreman	36.23	37.66	**38.39**	32.23	34.51	**34.66**
Girl	34.29	34.28	**34.30**	31.67	**32.10**	**32.10**
House	32.36	33.68	**34.43**	28.92	30.18	**30.24**
Koala	33.39	33.19	**33.91**	29.77	30.23	**30.35**
Leaves	27.61	27.48	**28.81**	22.16	22.99	**23.17**
Lena	34.29	34.06	**34.87**	30.60	31.04	**31.21**
Pantheon	27.13	27.28	**27.51**	24.44	**24.94**	**24.94**
Parrot	26.99	27.06	**27.42**	23.43	23.97	**24.06**
Starfish	30.43	30.20	**31.24**	26.50	26.71	**26.87**
Stream	25.86	25.66	**25.96**	23.07	23.33	**23.34**
Texture	21.50	21.43	**22.01**	16.59	17.20	**17.29**
Average	29.35	29.54	**30.09**	25.68	26.36	**26.44**

[6]Notice that the only difference between the general objective function and the second stage is the existence of the dictionary update.

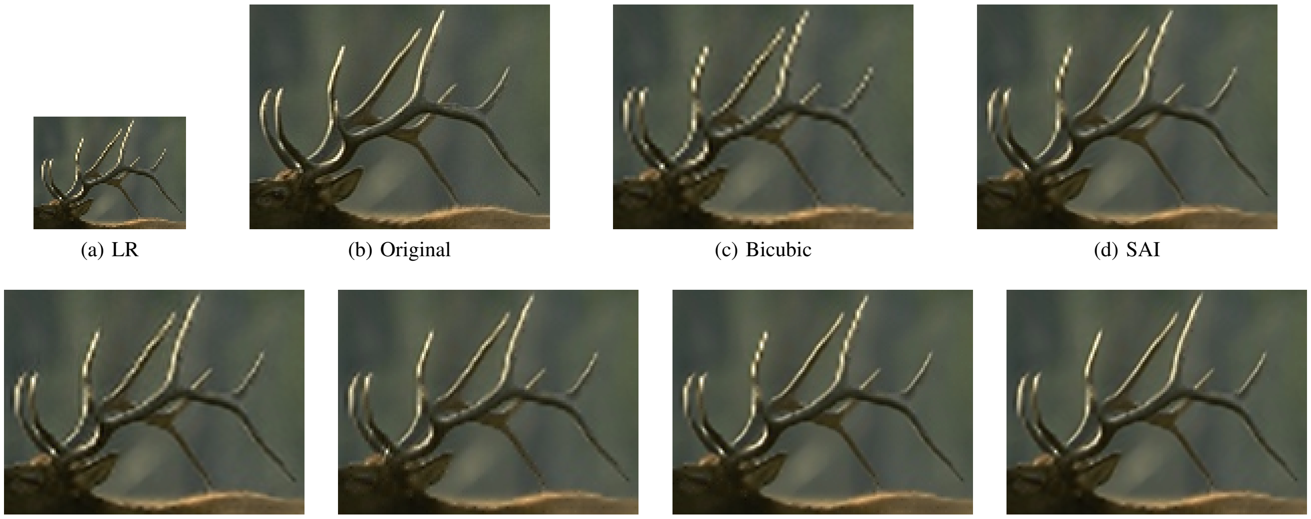

(a) LR (b) Original (c) Bicubic (d) SAI
(e) SME (f) PLE (g) NARM (h) Proposed

Fig. 3. Visual comparison of crop from Elk, magnified by factor of 2.

TABLE III
SUMMARY OF THE INTERPOLATION RESULTS [PSNR] FOR UP-SCALING BY FACTOR OF $L = 2$ AND $L = 3$. THE BEST RESULTS IN EACH LINE ARE HIGHLIGHTED.

Image	$L = 2$						$L = 3$					
	Bicubic	**SAI** [19]	**SME** [20]	**PLE** [21]	**NARM** [15]	**Ours**	**Bicubic**	**SAI** [19]	**SME** [20]	**PLE** [21]	**NARM** [15]	**Ours**
Bears	28.42	28.46	28.39	28.44	28.27	**28.57**	25.34	25.34	25.46	25.63	25.34	**25.71**
Boat	29.30	29.72	29.74	29.83	29.80	**30.18**	26.06	26.34	26.43	26.48	26.53	**26.84**
Butterfly	27.69	28.65	29.17	28.79	**30.30**	29.68	23.51	24.10	24.59	24.56	**25.57**	25.27
Cameraman	25.50	26.14	25.88	26.40	25.94	**26.59**	22.54	22.88	22.92	23.20	22.72	**23.27**
Elk	31.77	32.82	33.16	32.92	33.28	**33.83**	28.13	28.74	28.81	28.91	28.98	**29.33**
Fence	24.77	24.53	23.78	24.96	24.79	**25.03**	20.90	20.66	20.44	**21.12**	20.53	20.90
Flowers	28.05	28.41	28.65	28.58	28.75	**28.81**	25.73	25.92	26.08	26.17	**26.44**	26.38
Foreman	35.39	37.17	37.68	37.15	**38.64**	38.39	31.86	32.79	33.17	33.06	**34.80**	34.66
Girl	33.88	34.03	34.13	34.25	**34.46**	34.30	31.53	31.70	31.85	31.90	31.90	**32.10**
House	32.33	33.15	32.84	33.28	33.52	**34.43**	28.78	29.21	29.16	29.45	29.67	**30.24**
Koala	33.27	33.68	33.74	33.42	**33.84**	**33.91**	29.59	29.83	29.95	30.12	30.09	**30.35**
Leaves	26.77	28.21	28.72	27.82	**29.76**	28.81	21.76	22.45	22.65	22.55	**23.33**	23.17
Lena	34.01	34.53	34.68	34.46	**35.01**	34.87	30.24	30.78	30.98	30.91	31.16	**31.21**
Pantheon	27.24	27.13	27.10	**27.57**	27.36	27.51	24.42	24.52	24.65	24.73	24.72	**24.94**
Parrot	26.50	26.87	27.34	27.00	26.96	**27.42**	23.14	23.24	23.58	23.85	23.37	**24.06**
Starfish	30.44	30.35	30.76	30.61	**31.72**	31.24	26.36	26.28	26.52	26.62	26.86	**26.87**
Stream	25.77	25.79	25.84	25.91	25.82	**25.96**	23.06	23.02	23.09	23.27	23.19	**23.34**
Texture	20.56	21.55	21.49	21.74	21.51	**22.01**	16.40	17.07	16.81	17.00	16.54	**17.29**
Average	28.98	29.51	29.62	29.62	29.98	**30.09**	25.52	25.83	25.95	26.08	26.21	**26.44**

of a single set of parameters per each scaling factor and stage, which are given in Table I. Note that we limit the number of atoms which represent the HR patches for at most 7 atoms.

Before comparing the proposed algorithm with the state-of-the-art methods, we demonstrate the differences between the general objective function (Section III-A) and the final algorithm (Sections III-B, III-C) in terms of PSNR. We implemented the general objective function by repeating iteratively (i) joint weighted sparse-coding as described in Equation (9), (ii) dictionary learning as described in Equation (10), and (iii) image reconstruction as described in Equation (8). Table II lists the PSNR of the general objective function, the first-stage, and the second-stage (i.e. the final result) of the proposed algorithm. It is clear that the restoration of the proposed algorithm is much better than the general one. In terms of PSNR, the average differences between them for interpolation by factors of $L = 2$ and $L = 3$ are 0.74dB and 0.76dB, respectively. The first stage of the proposed algorithm offers a PSNR improvement of 0.19dB (for $L = 2$) and 0.68dB (for $L = 3$) over the general algorithm, while the second stage offers a further improvement of 0.55dB (for $L = 2$) and 0.08dB (for $L = 3$) over the first stage. To summarize, the performance of the proposed two-stage algorithm is much better than the general non-local sparsity objective function.

A. Up-scaling by a factor of 2

In Table III we compare the proposed algorithm with the current stage-of-the-art methods. The competitive methods for interpolation by factor of $L = 2$ are (i) bicubic, (ii) Decision and Adaptive Interpolator (SAI) [19], (iii) Sparse

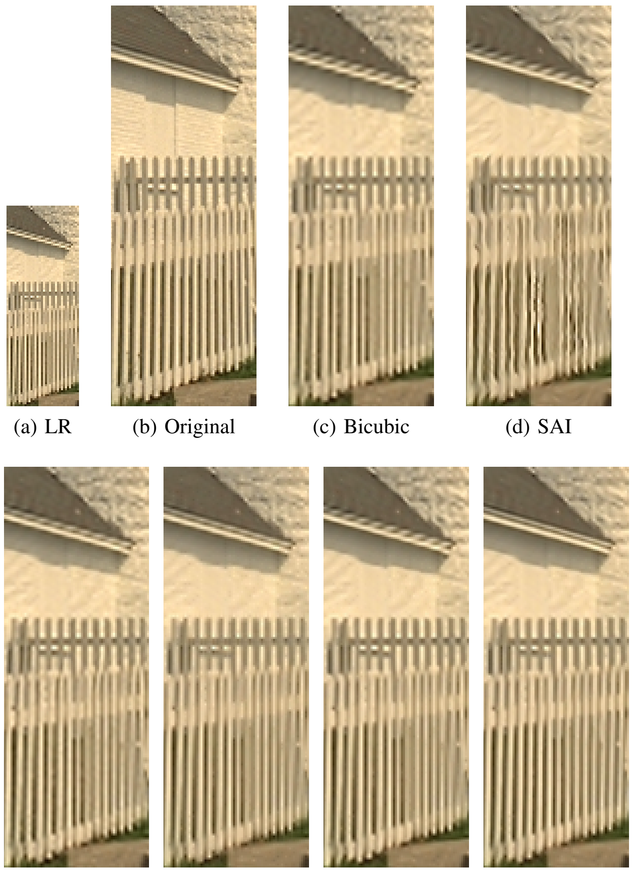

(a) LR (b) Original (c) Bicubic (d) SAI (e) SME (f) PLE (g) NARM (h) Proposed

Fig. 4. Visual comparison of crop from Fence, magnified by factor of 2.

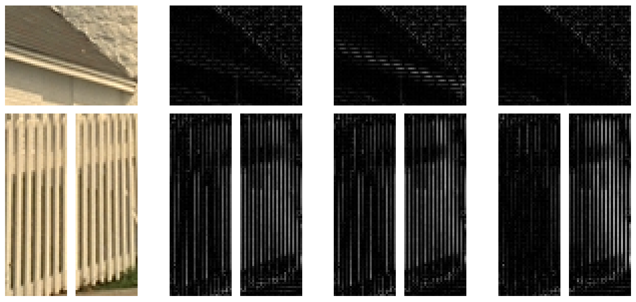

(a) Original (b) PLE Err (c) NARM Err (d) Proposed Err

Fig. 5. Visual comparison of the absolute difference between portions from the original and the interpolated Fence image, magnified by factor of 2.

Mixing Estimation (SME) [20], (iv) Piecewise Linear Estimation (PLE) [21] and (v) Nonlocal Auto-Regressive Modeling (NARM) [15][7]. SAI is an adaptive edge-directed interpolator which exploits the image directional regularity, SME is a zooming algorithm that exploits directional structured sparsity in wavelet representations, PLE is a general framework for solving inverse problems based on Gaussian mixture model, and NARM combines the non-local self-similarities and the sparsity prior as described in the Section I. The results are generated by the original authors' software. Note that PLE has a special treatment for color images while in the following simulations we interpolate and measure the PSNR on the luminance channel only. From table III we can see that our algorithm is competitive with the current state-of-the-art NARM interpolator with an average gain of 0.11dB. The proposed algorithm outperforms the bicubic, SAI, SME and PLE methods with an average gain of 1.11dB, 0.58dB, 0.47dB and 0.47dB, respectively.

Figs. 3 and 4 demonstrate a visual comparison between the above algorithms for interpolation by factor of 2. According to these figures, the proposed method results in less ringing and aliasing artifacts than the others. Visually, the results are comparable to NARM, showing pleasant outcome with hardly any artifacts. Fig. 5 shows the reconstruction error images of PLE, NARM, and the proposed method. These error images show the absolute difference between the original and the interpolated results, with a proper common magnification. As can be seen, the proposed method succeeds in recovering the house's roof and the left portion of the fence better than PLE and NARM, while performing roughly equivalent on the right portion of the fence.

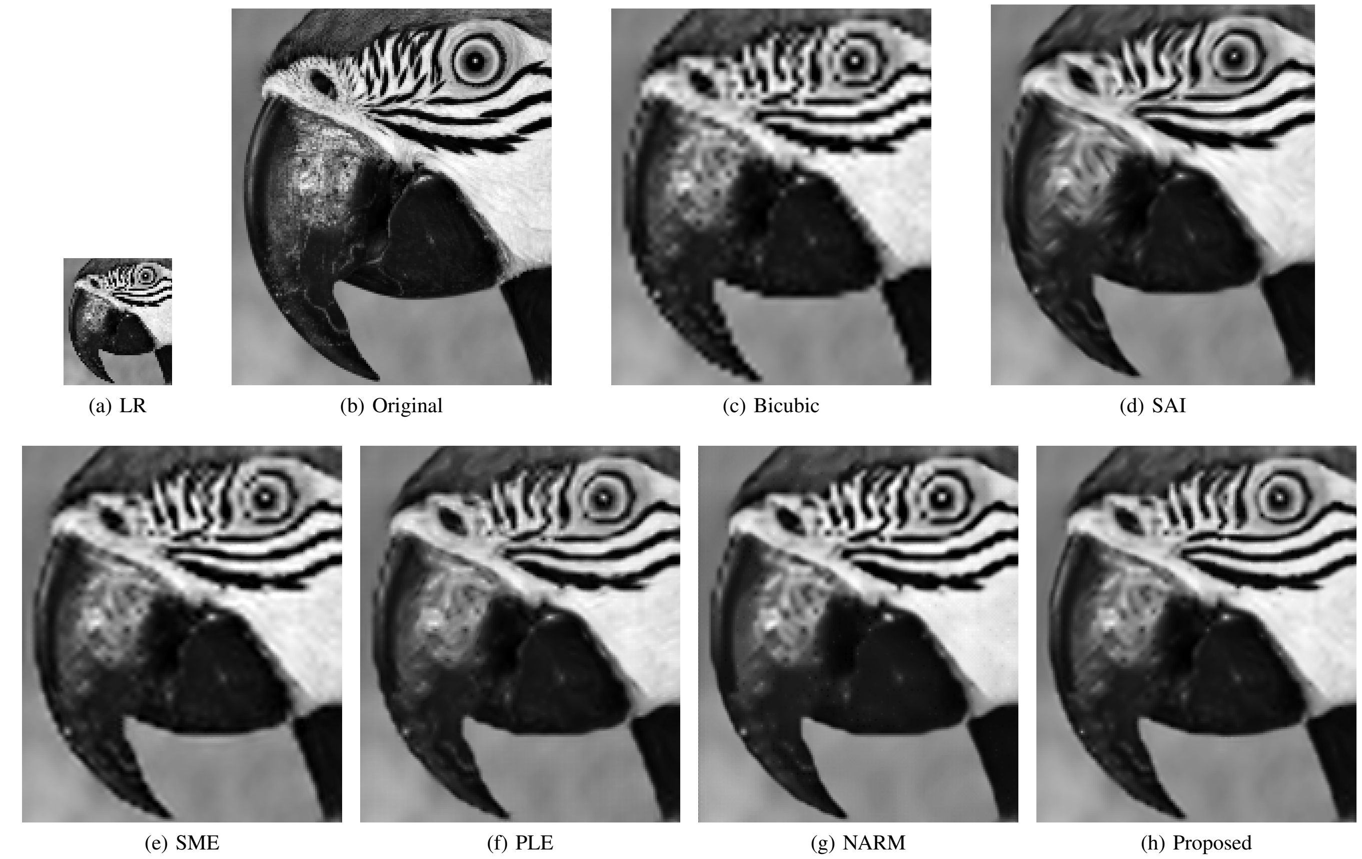
B. *Up-scaling by a factor of 3*

For interpolation by factor of $L = 3$ we compare the proposed algorithm with the bicubic, SAI, SME, PLE and NARM methods. Since the available implementations for SAI and SME do not support upscaling of factors other than 2, we have tested these algorithms by upscaling twice by a factor of 2 and then reducing the result back by a factor of $\frac{3}{4}$ using the bicubic interpolation. According to Table III, the proposed algorithm outperforms the bicubic, SAI, SME, PLE and NARM interpolators with an average gap of 0.92dB, 0.61dB, 0.49dB, 0.36dB and 0.23dB, respectively.

A visual comparison between the above algorithms and the proposed method can be found in Figs. 6 and 7. The ability of the proposed algorithm to recover continues edges and fine details (e.g. the parrot's eye) despite the large magnification is demonstrated in Fig. 6, and the ability to handle severe aliasing is demonstrated in Fig. 7. A comparison between the reconstruction error images of PLE, NARM, and the proposed method is given in Fig. 8, supporting the quantified results and showing improved recovery for the proposed algorithm over PLE and NARM.

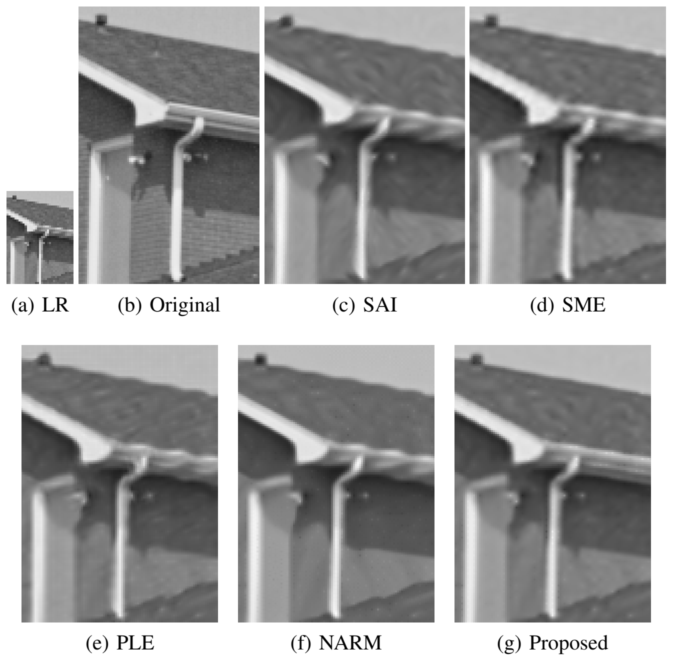
Note that for some images, PLE and NARM perform slightly better than the proposed algorithm. NARM may outperform the proposed algorithm for images with large regions that fit the self-similarity assumption (e.g. Foreman and Leaves) due to the implicit way that NARM relies on the image self-similarities. On the other hand, the initialization of PLE is very effective and leads to visually very pleasant results with a low computational complexity. Learning an HR dictionary based on the LR image is challenging in general, especially for images that suffer from very strong aliasing artifacts (e.g. Fence and Parthenon). PLE may handle these type of images in a better way than the proposed method thanks to its special initial dictionary, which represent the image very well even without any dictionary update steps.

[7]We do not compare the proposed algorithm with LSSC [16] since it was not designed\tested for image interpolation.


(a) LR (b) Original (c) Bicubic (d) SAI

(e) SME (f) PLE (g) NARM (h) Proposed

Fig. 6. Visual comparison of crop from Parrot, magnified by factor of 3.


(a) LR (b) Original (c) SAI (d) SME

(e) PLE (f) NARM (g) Proposed

Fig. 7. Visual comparison of crop from House, magnified by factor of 3.

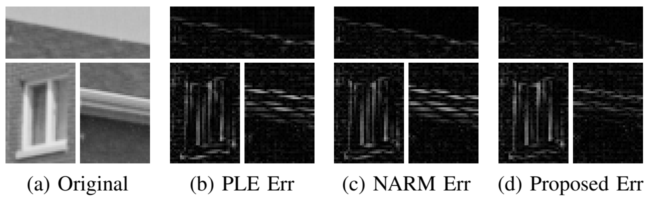

(a) Original (b) PLE Err (c) NARM Err (d) Proposed Err

Fig. 8. Visual comparison of the absolute difference between portions from the original and the interpolated House image, magnified by factor of 3.

C. Weighting the Unknown Pixels

Assigning a high weight to known pixels and a low one for the interpolated pixels is necessary and highly influential for the success in the overall interpolation task. Fig. 9 plots the average PSNR over the test images as a function of the weight for the unknown pixels, ranging from 0.005 up to 1. Note that *differently* from the proposed algorithm, here we use the same weight for both stages in order to measure its impact. As can be seen, the sensitivity of the algorithm to varying weights between 0.005 to 0.1 is small. The best restorations are achieved around the weights 0.01 and 0.05 for interpolation by factor of 2 and 3, respectively. In the context of interpolation by a factor of 3, there is a slight advantage for the weight 0.05 over 0.005 since higher weight results in more stable sparse-coding.

Another related test we present here studies the impact of increasing this weight as a function of the iterations. Clearly, there are many strategies for increasing the weight, and we explored several options. For example, we linearly increased the weights of the unknown pixels from the initial first stage value up to the second stage value, i.e. from 0.01 up to 0.05 and from 0.01 up to 0.1 for interpolation by factor of 2 and 3,

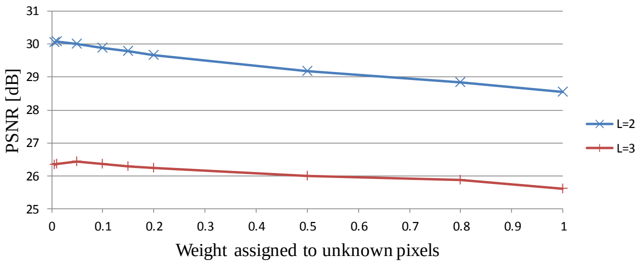


Fig. 9. The average PSNR [dB] over the test images as a function of the weight assigned to unknown pixels, ranging from 0.005 up to 1.

respectively. In terms of the resulting PSNR, a constant weight per-stage performs slightly better than increasing weights, with an average difference of 0.003 dB and 0.01 dB for interpolation by factor of 2 and 3, respectively.

Returning to Fig. 9, the worst performance is obtained for a weight that is equal to 1 (i.e. using the conventional l_2 norm instead of the weighted l_2 norm). This result is not surprising. As a reminder, each (pixel-weighted) group is represented subject to a low error threshold ($c_\epsilon = 0.01$ according to Table I). Using the weighted l_2 norm with this threshold leads to a patch reconstruction that is close to the known pixels, while the unknown ones are naturally interpolated. The interpolation is done by a linear combination of the chosen HR dictionary atoms, which in turn are updated iteratively to obtain better and better estimation of these unknown pixels.

Using the same strategy but via the conventional l_2 norm leads to a patch reconstruction that is close to the whole interpolated patch. The joint sparse coding algorithm in this case chooses atoms that better fit the interpolated pixels, rather than the known ones, since the number of known pixels within a patch is relatively small. This resembles a computation of a rough approximation of the HR image by replacing every "weak" patch with its most similar "strong" one over and over again (see Equation (11)).

Another explanation for the above degradation emerges from [15], which claims that the conventional sparsity-based methods (e.g. [30]) are less effective because the data fidelity term fails to impose structural constraint on the missing pixels. The authors of [15] suggest exploiting the self-similarity assumption in order to connect a missing pixel with its non-local neighbors. Our algorithm exploits the non-local self-similarity assumption too, but somewhat differently. We (i) use a joint weighted sparse coding, and (ii) replace the representation of each "weak" patch (whose central pixel is missing) with its most similar "strong" patch (whose central pixel is known) representation, all this in the first stage of the algorithm. However, using the conventional l_2 norm instead of the weighted l_2 norm cancels the discrimination between the known pixels and the unknown ones, and this results in inferior interpolation performance.

D. Similarity Function

Many recent papers on subspace clustering (e.g. [31]) indicate that joint sparse-coding should group patches that belong to the same subspace, and this in turn means that the grouped patches are not necessarily expected to be close-by in l_2 (or l_1). Therefore, grouping similar patches according to their l_2 or l_1 norm is not the best choice. However, the popularity, simplicity and low-computational cost of these norms, together with an impressive restoration performance, make them very attractive. In our work, we have chosen the l_1 norm over the l_2 because it is more robust to outliers. In terms of the resulting PSNR, we found that the l_1 norm is slightly better than the l_2 norm, with an average difference of 0.1 dB and 0.07 dB for interpolations by factor of 2 and 3, respectively.

E. Computational Complexity

The complexity of the proposed algorithm is composed of three main parts: (i) computing K-Nearest Neighbors (K-NN) per each patch within a window of size $h \times h$ pixels, (ii) sparse coding using an element-wise weighted variant of the SOMP, and (iii) dictionary learning using an element-wise weighted variant of the K-SVD. The complexity of computing the K-NN per-patch is $O(n \cdot h^2 + h^2 \cdot \log h^2)$. A detailed complexity analysis of the OMP, its batch implementation, and the K-SVD are given in [29]. Note that we approximate the solution of the weighted SOMP by applying its batch implementation without weights followed by a weighted least-squares. The complexity of the batch-SOMP per-patch is $O\left(K \cdot (n \cdot m + s^2 \cdot n + s^3)\right)$, where s is the average number of atoms that participate in the representation of the HR patches. Adding now the weights costs additional $O\left(K \cdot (s^2 \cdot n + s^3)\right)$. The complexity of one dictionary-update step is approximately $O\left(n \cdot (m^3 + s^2 \cdot K \cdot N)\right)$, where $K \cdot N$ is the number of examples (since the number of patches of the HR image is N and each patch has K similar patches). The overall complexity of the proposed algorithm under the assumptions that $s \ll n < m \approx h^2$ is

$$O\left(I \cdot n \cdot m^3 + I \cdot K \cdot N \cdot (n \cdot m + s^2 \cdot n)\right), \tag{13}$$

where I is the number of iterations. The chosen parameters effect the complexity, which can be reduced by choosing sub-optimal parameters with a minor impact on the overall interpolation performance.

We compared the complexity of the proposed algorithm with SME [20], PLE [21] and NARM [15]. Unfortunately SAI [19] does not provide a complexity analysis; therefore we do not include it in this comparison. It is worth mentioning, though, that the runtime of the published implementation of SAI is faster than the others.

For an image of size N pixels, the complexity of SME is $O(N \cdot \log N)$, PLE costs $O(I \cdot B \cdot n^2 \cdot N)$, where I is the number of iterations and B is the number of PCA bases (typically $B = 19$), NARM costs $O(I \cdot n \cdot N^2)$, where n is the patch-size. The proposed algorithm costs $O(I \cdot n^3 \cdot (n^{2.5} + N))$. Note that in order to obtain a comparable complexity terms we made several assumptions about the variables dimensions[8].

We test the runtime of an un-optimized Matlab implementation on an Intel Core i7 3 GHz processor. The runtime for

[8]Regarding the complexity analysis of NARM and their notations, we set $T = I$, $N_L = N/L^2$, and assume that $q \approx N/K$, $t_1 < u \approx p \approx n \approx K \ll q$ and $\kappa \approx \sqrt{n}$. The complexity of the proposed algorithm is obtained under the assumptions that $K \approx s \approx \sqrt{n}$, $n < m \approx h^2$, $m \approx n^{1.5}$.

interpolating a 128×128 LR image to a 256×256 HR image is about 1 minute per iteration; therefore it takes 10-15 minutes to obtain our best interpolation performance. The algorithm could be optimized by replacing the exhaustive K-NN search with a fast patch matching (e.g. [32]), reducing the number of examples for the dictionary update (e.g. by choosing mostly active\textured patches), applying the dictionary update every several iterations, etc. However, we did not take this route, and this is left for future work.

To conclude, we demonstrated the efficiency of the proposed technique to recover an HR image from an observed LR one. We presented the advantages of the proposed two-stage algorithm over the general objective function. Furthermore, the experimental results indicate that the proposed method outperforms the state-of-the-art algorithms for interpolation by factors of L = 2 and L = 3. We illustrated the influence of the weights on the final result, discussed the similarity function and provided a complexity analysis along with a comparison to the competitive algorithms.

V. Conclusion

The interpolation problem is a special case of the super-resolution task, where pure decimation is applied on the original high-resolution image, leading to a subset of known pixels and void values around them in a regular pattern. The problem of filling-in these missing values has drawn a considerable attention in the past several years, and various techniques that rely on image statistics have been proposed. Inspired by the work reported in [15], [16], we proposed in this paper a novel image interpolation scheme that is composed of two phases. In both, the main forces exploited are sparse representation of the high-resolution image patches with a trained dictionary, and non-local relations that exist between image patches. The obtained results are encouraging, and our hope is to leverage this approach to treat more complicated scenarios such as single image super-resolution with assumed blur and noise, or fusion of several images.

Appendix

A closed-form solution for Equation (6) via the Lagrange multipliers method: This problem could be described more concisely as

$$\hat{\mathbf{x}} = \min_{\mathbf{x}} \ \frac{1}{2}\|\mathbf{A}\mathbf{x}-\mathbf{b}\|_2^2 \ \text{s.t.} \ \mathbf{y}=\mathbf{B}\mathbf{x}, \tag{14}$$

where $\mathbf{B}$ is the down-sampling operator $\mathbf{U}_\mathrm{L}$, the matrix $\mathbf{A} = \left[\sqrt{\tilde{\mathrm{W}}_1}\mathbf{R}_1; \sqrt{\tilde{\mathrm{W}}_2}\mathbf{R}_2; \cdots; \sqrt{\tilde{\mathrm{W}}_N}\mathbf{R}_N\right]$, and the vector $\mathbf{b} = \left[\sqrt{\tilde{\mathrm{W}}_1}\mathbf{D}\alpha_1^{sp}; \sqrt{\tilde{\mathrm{W}}_2}\mathbf{D}\alpha_2^{sp}; \cdots; \sqrt{\tilde{\mathrm{W}}_N}\mathbf{D}\alpha_N^{sp}\right]$. Using Lagrange multipliers, the solution is obtained by the following steps:

1) Form the Lagrangian with Lagrange multiplier vector $\mathbf{z}$ by

$$\mathcal{L}(\mathbf{x},\mathbf{z}) = \tfrac{1}{2}\|\mathbf{A}\mathbf{x}-\mathbf{b}\|_2^2 + \mathbf{z}^T(\mathbf{B}\mathbf{x}-\mathbf{y}). \tag{15}$$

2) Null the derivative w.r.t. $\mathbf{x}$,

$$\tfrac{\partial\mathcal{L}}{\partial\mathbf{x}} = 0 \Rightarrow \hat{\mathbf{x}} = \left(\mathbf{A}^T\mathbf{A}\right)^{-1}\left(\mathbf{A}^T\mathbf{b}-\mathbf{B}^T\mathbf{z}\right). \tag{16}$$

where $\left(\mathbf{A}^T\mathbf{A}\right)$ is a diagonal matrix, which counts the number of representations per each element.

3) Find $\mathbf{z}$ by forcing the constraint $\mathbf{y}=\mathbf{B}\hat{\mathbf{x}}$:

$$\mathbf{y} = \mathbf{B}\left(\mathbf{A}^T\mathbf{A}\right)^{-1}\left(\mathbf{A}^T\mathbf{b}-\mathbf{B}^T\mathbf{z}\right), \tag{17}$$

leading to a closed-form solution for $\mathbf{z}$ by

$$\mathbf{z} = \left(\mathbf{B}\left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{B}^T\right)^{-1}\left(\mathbf{B}\left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T\mathbf{b}-\mathbf{y}\right). \tag{18}$$

Notice that the matrix to invert here is positive definite if $\mathbf{A}^T\mathbf{A}$ is positive definite, and $\mathbf{B}$ has full row-rank. In our case, these two requirements are met.

4) Finally, obtain a closed-form solution for $\hat{\mathbf{x}}$ by substituting $\mathbf{z}$ into Equation (16).

$$\begin{aligned}\hat{\mathbf{x}} =& \left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T\mathbf{b} - \left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{B}^T\mathbf{z} \\ =& \left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T\mathbf{b} \\ &- \left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{B}^T\left(\mathbf{B}\left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{B}^T\right)^{-1}\mathbf{B}\left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T\mathbf{b} \\ &+ \left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{B}^T\left(\mathbf{B}\left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{B}^T\right)^{-1}\mathbf{y}.\end{aligned} \tag{19}$$

Notice that this expression is misleadingly complex and long, and in fact it has a simple interpretation and easy computation. This can be exposed by separating the pixels in $\hat{\mathbf{x}}$ to two kinds - the ones that were known in the low-resolution image, and the others. First, by multiplying $\hat{\mathbf{x}}$ by $\mathbf{B}$ we isolate the known pixels, and using Equation (19) this leads to

$$\begin{aligned}\mathbf{B}\hat{\mathbf{x}} =& \\ & \mathbf{B}\left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T\mathbf{b} \\ &-\mathbf{B}\left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{B}^T\left(\mathbf{B}\left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{B}^T\right)^{-1}\mathbf{B}\left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T\mathbf{b} \\ &+\mathbf{B}\left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{B}^T\left(\mathbf{B}\left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{B}^T\right)^{-1}\mathbf{y} \\ &= \mathbf{B}\left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T\mathbf{b} - \mathbf{B}\left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T\mathbf{b} + \mathbf{y} \\ &= \mathbf{y},\end{aligned} \tag{20}$$

where we have used the self-cancellation of the term $\mathbf{B}\left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{B}^T$ with its inverse. The outcome should not be surprising, as it is exactly the constraint posed in (14).

As to the interpolated pixels, define the operator $\tilde{\mathbf{B}}$ as a decimation that removes the known pixels, leaving only the others. As above, we multiply $\hat{\mathbf{x}}$ by $\tilde{\mathbf{B}}$ in order to see who those pixels are in the solution obtained in Equation (19). Observe that since $\mathbf{A}^T\mathbf{A}$ is diagonal, then the term $\tilde{\mathbf{B}}(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{B}^T$ is zero. This is because $\mathbf{B}^T$ interpolates an image by zero filling, the multiplication by $(\mathbf{A}^T\mathbf{A})^{-1}$ scales every pixel in the outcome, and eventually $\tilde{\mathbf{B}}$ chooses only the zero pixels. Therefore, in Equation (19), the second and third terms are nulled, leading to

$$\tilde{\mathbf{B}}\hat{\mathbf{x}} = \tilde{\mathbf{B}}(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b}, \tag{21}$$

which means that we put each reconstructed patch in its location, average them all, and normalize by the number of contributions per each pixel along with their weights. So, to summarize, the solution of the problem posed in Equation (14) is given by

$$\hat{\mathbf{x}} = \begin{cases} (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b} & \text{if this is an interpolated pixel} \\ \mathbf{y} & \text{if this is a known pixel.} \end{cases}$$

Acknowledgment

The authors thank the anonymous reviewers for their helpful comments. This research was supported by the European Research Council under EU's 7th Framework Program, ERC Grant agreement no. 320649, and by the Intel Collaborative Research Institute for Computational Intelligence.

References

[1] M. Protter, M. Elad, H. Takeda, and P. Milanfar, "Generalizing the nonlocal-means to super-resolution reconstruction," *IEEE Trans. on Image Proc.*, vol. 18, no. 1, pp. 36–51, 2009.

[2] S. Farsiu, M. D. Robinson, M. Elad, and P. Milanfar, "Fast and robust multiframe super resolution," *IEEE Trans. on Image proc.*, vol. 13, no. 10, pp. 1327–1344, 2004.

[3] A. Danielyan, A. Foi, V. Katkovnik, and K. Egiazarian, "Image and video super-resolution via spatially adaptive block-matching filtering," in *Proc. Int. Workshop Local and Non-Local Approx. Image Proc., Switzerland*, 2008.

[4] R. Keys, "Cubic convolution interpolation for digital image processing," *IEEE Trans. on Acoustics, Speech and Signal Proc.*, vol. 29, no. 6, pp. 1153–1160, 1981.

[5] H. Hou and H. Andrews, "Cubic splines for image interpolation and digital filtering," *IEEE Trans. on Acoustics, Speech and Signal Proc.*, vol. 26, no. 6, pp. 508–517, 1978.

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

[7] M. Lebrun, A. Buades, and J.-M. Morel, "Implementation of the "Non-Local Bayes" (NL-Bayes) Image Denoising Algorithm," *Image Proc. On Line*, vol. 2013, pp. 1–42, 2013.

[8] A. Buades, B. Coll, and J.-M. Morel, "A review of image denoising algorithms, with a new one," *Multiscale Modeling & Simulation*, vol. 4, no. 2, pp. 490–530, 2005.

[9] A. Danielyan, V. Katkovnik, and K. Egiazarian, "BM3D frames and variational image deblurring," *IEEE Trans. on Image Proc.*, vol. 21, no. 4, pp. 1715–1728, 2012.

[10] S. Kindermann, S. Osher, and P. W. Jones, "Deblurring and denoising of images by nonlocal functionals," *Multiscale Modeling & Simulation*, vol. 4, no. 4, pp. 1091–1115, 2005.

[11] D. Glasner, S. Bagon, and M. Irani, "Super-resolution from a single image," in *Int. Conf. on Computer Vision*. IEEE, 2009, pp. 349–356.

[12] G. Freedman and R. Fattal, "Image and video upscaling from local self-examples," *ACM Trans. on Graphics*, vol. 30, no. 2, p. 12, 2011.

[13] A. M. Bruckstein, D. L. Donoho, and M. Elad, "From sparse solutions of systems of equations to sparse modeling of signals and images," *SIAM review*, vol. 51, no. 1, pp. 34–81, 2009.

[14] M. Elad, *Sparse and redundant representations: from theory to applications in signal and image processing*. Springer, 2010.

[15] W. Dong, L. Zhang, R. Lukac, and G. Shi, "Sparse representation based image interpolation with nonlocal autoregressive modeling," *IEEE Trans. on Image Proc.*, vol. 22, pp. 1382–1394, 2013.

[16] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman, "Non-local sparse models for image restoration," in *Int. Conf. on Computer Vision*. IEEE, 2009, pp. 2272–2279.

[17] J. A. Tropp, A. C. Gilbert, and M. J. Strauss, "Algorithms for simultaneous sparse approximation. Part I: Greedy pursuit," *Signal Proc.*, vol. 86, no. 3, pp. 572–588, 2006.

[18] M. Aharon, M. Elad, and A. Bruckstein, "The K-SVD: An algorithm for designing of overcomplete dictionaries for sparse representation," *IEEE Trans. on Signal Proc.*, vol. 54, no. 11, pp. 4311–4322, 2006.

[19] X. Zhang and X. Wu, "Image interpolation by adaptive 2-D autoregressive modeling and soft-decision estimation," *IEEE Trans. on Image Proc.*, vol. 17, no. 6, pp. 887–896, 2008.

[20] S. Mallat and G. Yu, "Super-resolution with sparse mixing estimators," *IEEE Trans. on Image Proc.*, vol. 19, no. 11, pp. 2889–2900, 2010.

[21] G. Yu, G. Sapiro, and S. Mallat, "Solving inverse problems with piecewise linear estimators: from Gaussian mixture models to structured sparsity," *IEEE Trans. on Image Proc.*, vol. 21, no. 5, pp. 2481–2499, 2012.

[22] S. G. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Trans. on Signal Proc.*, vol. 41, no. 12, pp. 3397–3415, 1993.

[23] D. L. Donoho, Y. Tsaig, I. Drori, and J.-L. Starck, "Sparse solution of underdetermined systems of linear equations by stagewise orthogonal matching pursuit," *IEEE Trans. on Information Theory*, vol. 58, no. 2, pp. 1094–1121, 2012.

[24] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM journal on scientific computing*, vol. 20, no. 1, pp. 33–61, 1998.

[25] K. Engan, S. O. Aase, and J. Hakon Husoy, "Method of optimal directions for frame design," in *IEEE Int. Conf. on Acoustics, Speech, and Signal Proc.*, vol. 5. IEEE, 1999, pp. 2443–2446.

[26] L. N. Smith and M. Elad, "Improving dictionary learning: Multiple dictionary updates and coefficient reuse," *IEEE Signal Proc. Letters*, vol. 20, no. 1, pp. 79–82, 2013.

[27] M. Elad and M. Aharon, "Image denoising via sparse and redundant representations over learned dictionaries," *IEEE Trans. on Image Proc.*, vol. 15, no. 12, pp. 3736–3745, 2006.

[28] S. K. Park and R. Hazra, "Aliasing as noise: a quantitative and qualitative assessment," in *Optical Engineering and Photonics in Aerospace Sensing*. Int. Society for Optics and Photonics, 1993, pp. 2–13.

[29] R. Rubinstein, M. Zibulevsky, and M. Elad, "Efficient implementation of the K-SVD algorithm using batch orthogonal matching pursuit," *CS Technion*, 2008.

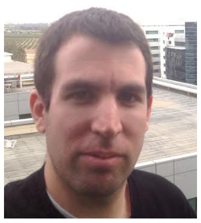
[30] W. Dong, L. Zhang, G. Shi, and X. Wu, "Image deblurring and super-resolution by adaptive sparse domain selection and adaptive regularization," *IEEE Trans. on Image Proc.*, vol. 20, no. 7, pp. 1838–1857, 2011.

[31] A. Adler, M. Elad, and Y. Hel-Or, "Probabilistic subspace clustering via sparse representations," *IEEE Signal Proc. Letters*, vol. 20, no. 1, pp. 63–66, 2013.

[32] C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman, "PatchMatch: a randomized correspondence algorithm for structural image editing," *ACM Trans. on Graphics*, vol. 28, no. 3, p. 24, 2009.

Yaniv Romano received his B.Sc. (2012) from the department of Electrical engineering at the Technion, Israel, and he is currently a Ph.D. candidate in the same department.

In parallel, Yaniv has been working in the industry since 2011 as an image processing algorithm developer. His research interests include sparse and redundant representations, inverse problems and super-resolution.



Matan Protter received his B.Sc. (2003) in Physics, Mathematics and Computer Science from the Hebrew University and his M.Sc. (2008) and Ph.D. (2011) from the Computer Science Department at the Technion, Israel.

Matan worked for 6 years as a senior researcher in the image processing group in RAFAEL. He has founded two startups and consulted to various hi-tech companies in the field of computer vision.

Michael Elad received his B.Sc. (1986), M.Sc. (1988) and D.Sc. (1997) from the department of Electrical engineering at the Technion, Israel. Since 2003 he is a faculty member at the Computer-Science department at the Technion, and since 2010 he holds a full-professorship position.

Michael Elad works in the field of signal and image processing, specializing in particular on inverse problems, sparse representations and super-resolution. Michael received the Technion's best lecturer award six times, he is the recipient of the 2007 Solomon Simon Mani award for excellence in teaching, the 2008 Henri Taub Prize for academic excellence, and the 2010 Hershel-Rich prize for innovation. Michael is an IEEE Fellow since 2012. He is serving as an associate editor for SIAM SIIMS, and ACHA. Michael is also serving as a senior editor for IEEE SPL.