Demosaicing Based on Directional Difference Regression and Efficient Regression Priors

Jiqing Wu, Radu Timofte, *Member, IEEE*, and Luc Van Gool, *Fellow, IEEE*

***Abstract*— Color demosaicing is a key image processing step aiming to reconstruct the missing pixels from a recorded raw image. On the one hand, numerous interpolation methods focusing on spatial-spectral correlations have been proved very efficient, whereas they yield a poor image quality and strong visible artifacts. On the other hand, optimization strategies, such as learned simultaneous sparse coding and sparsity and adaptive principal component analysis-based algorithms, were shown to greatly improve image quality compared with that delivered by interpolation methods, but unfortunately are computationally heavy. In this paper, we propose efficient regression priors as a novel, fast post-processing algorithm that learns the regression priors offline from training data. We also propose an independent efficient demosaicing algorithm based on directional difference regression, and introduce its enhanced version based on fused regression. We achieve an image quality comparable to that of the state-of-the-art methods for three benchmarks, while being order(s) of magnitude faster.**

***Index Terms*— Demosaicing, color filter array, super-resolution, image enhancement, linear regression.**

I. Introduction

FOR reasons of cost, most digital cameras are based on a single image sensor equipped with a color filter array (CFA). The Bayer pattern filter [1], as shown in Fig. 1, is the most frequently used CFA. Other patterns are discussed in [2]. The study of demosaicing algorithms for the Bayer pattern, aiming at recovering the missing color bands at each pixel, has a long history (see [3], [4]). We can broadly fit them into two categories: interpolation- and optimization-based methods.

Initially, interpolation-based methods were developed. Among those, nearest neighbor, bilinear or bicubic methods are the simplest as they interpolate within the R, G, and B channels independently. Later on, researchers started to exploit the spatial-spectral correlations that exist between the RGB channels.

Manuscript received September 21, 2015; revised February 3, 2016 and May 17, 2016; accepted May 25, 2016. Date of publication June 1, 2016; date of current version June 28, 2016. This work was supported by the European Research Council through the Advanced Grant VarCity under Grant 273940. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Oleg V. Michailovich.

J. Wu and R. Timofte are with the Department of Information Technology and Electrical Engineering, Zürich 8092, Switzerland (e-mail: jwu@vision.ee.ethz.ch; radu.timofte@vision.ee.ethz.ch).

L. Van Gool is with the Department of Information Technology and Electrical Engineering, Zürich 8092, Switzerland, and also with the Department of Electrical Engineering, Katholieke Universiteit Leuven, Leuven 3000, Belgium (e-mail: vangool@vision.ee.ethz.ch).

Codes available at: http://www.vision.ee.ethz.ch/~imofter/

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIP.2016.2574984

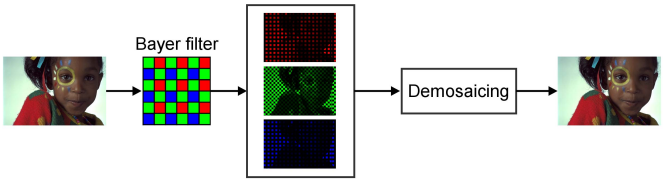


Fig. 1. Scheme of demosaicing.

HQL: Malvar *et al.* [5] introduce high-quality linear interpolation (HQL). HQL is a gradient-corrected bilinear interpolation method, with a gain parameter to weight the gradient correction term. In other words, Malvar *et al.* first apply bilinear interpolation to compute lost G values at R/B locations, then correct them by using the spatial gradients of R/B. A similar strategy is applied for the interpolation of the missing R/B values.

DLMMSE: Zhang and Wu [6] develop the directional linear minimum mean-square error estimation (DLMMSE) technique. DLMMSE builds on the assumption that differencing G and R/B channels amounts to low-pass filtering, given their strong correlation. The results are typically referred to as 'primary difference signals' or 'PDS'. In particular, DLMMSE adaptively estimates the missing G values in both horizontal and vertical directions, and then optimally fuses them. Finally, the R/B channels are computed, guided by the reconstructed G channel and the PDS.

LPAICI: Paliy *et al.* [7] propose spatially adaptive color filter array interpolation. They employ local polynomial approximation (LPA) (Katkovnik *et al.* [8]) and the paradigm of intersection of confidence intervals (ICI) (Katkovnik *et al.* [9]). ICI serves to determine the scales of LPA. LPAICI aims to filter the directional differences obtained by the Hamilton and Adam algorithm [10].

PCSD: Wu and Zhang [11] present a primary-consistent soft-decision method (PCSD). PCSD computes several estimations of the RGB channels via primary-consistent interpolation under different assumptions on edge and texture directions. Here, the primary-consistent interpolation indicates that all three primary components of a color are interpolated in the same direction. The final step is to test the assumptions and select the best, through an optimal statistical decision or inference process.

GBTF & MSG: Pekkucuksen and Altunbasak propose the gradient-based threshold-free (GBTF) method [12] and an improved version, the multiscale gradients-based (MSG) [13] color filter array interpolation. GBTF addresses certain limitations of DLMMSE by introducing gradients of color differences to compute weights for the west, east, north and south

1057-7149 © 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

directions. MSG further applies multiscale color gradients to adaptively combine color estimates from different directions.

MLRI: Incorporating the idea from GBTF, Kiku *et al.* [14] propose minimized-Laplacian residue interpolation (MLRI). They estimate the tentative pixel values by minimizing the Laplacian energies of the residuals.

AVSC: Zhang *et al.* [15] propose a robust color demosaicing method with adaptation to varying spectral correlations (AVSC). AVSC is a hybrid approach which combines an existing color demosaicing algorithm such as DLMMSE [6] with an adaptive intraband interpolation.

LDINAT: Zhang *et al.* [16] derive a color demosaicing method by local directional interpolation and nonlocal adaptive thresholding (LDINAT) and exploit the non-local image redundancy to enhance the local color results.

Besides interpolation based methods, the demosaicing problem is also tackled with optimization-based methods.

AP: For optimization, Gunturk *et al.* [17] iteratively exploit inter-channel correlation in an alternating-projections scheme (AP). After initial estimation, intermediate results are projected onto two constraint sets, which are determined by the observed data and prior information on spectral correlation.

AHD: Hirakawa and Parks [18] propose an adaptive homogeneity-directed demosaicing algorithm (AHD). AHD employs metric neighborhood modeling and filter bank interpolation in order to determine the interpolation direction and cancel aliasing, followed by artifact reduction iterations.

LSSC: Mairal *et al.* [19] derive a learned simultaneous sparse coding method (LSSC) for both denoising and demosaicing. Essentially, they unify two steps – dictionary learning adapted to sparse signal description and exploiting the self-similarities of images into LSSC.

SAPCA: Last but not least, Gao *et al.* [20] propose the sparsity and adaptive principal component analysis (PCA) based algorithm (SAPCA) by solving a minimization problem, i.e. by minimizing an l_1 function that contains sparsity and PCA terms.

We observe that most methods do not perform consistently on the IMAX and Kodak datasets (see Fig. 7), which are the two most commonly used datasets for testing demosaicing algorithms. When they perform well on Kodak, they tend to be less convincing on IMAX. Of course, part of the reason is that the study of Kodak has a longer history than that of IMAX, and the images in IMAX seem to be more challenging to reconstruct. LSSC and SAPCA report the best performances on the Kodak dataset and SAPCA substantially outperforms all other methods on the IMAX dataset. Yet, both methods come with a high computational cost.

In this paper, we propose an efficient post-processing step that can be combined with all aforementioned demosaicing methods, and boost their performance. Of particular interest is its combination with the fastest ones, as this leads to state-of-the-art performance at high speed. On top of that, we also propose modifications that go beyond sheer post-processing and that further improve the results.

Our post-processing step is coined 'efficient regression priors method' (ERP). For a given demosaicing method, ERP learns offline linear regressors for the residuals between demosaiced training images and the ground truth, and then applies them to the output of the demosaicing method at runtime. ERP is inspired by the adjusted anchored neighborhood regression (A+) [21], [22], a state-of-the-art method in image super-resolution. Farsiu *et al.* [23] were among the first to observe the connection between super-resolution and demosaicing. ERP as sheer post-processing step has already been introduced in our previous paper [24]. Here we add two further refined versions for fast demosaicing, one based on directional difference regression (DDR) and the other on fused regression (FR). DDR and FR integrate MLRI and ERP beyond simply post-processing the demosaiced images. Motivated by MLRI, we fully explore the correlation between channels by training directional differences. As a result, our methods reduce the color artifacts and achieve state-of-the-art performance comparable to those of LSSC/SAPCA, but at running times that are order(s) of magnitude lower (see Fig. 2).

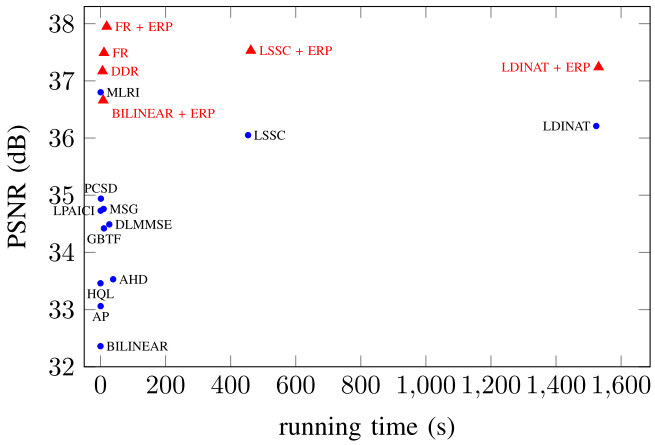


Fig. 2. Our proposed methods (DDR, FR and ERP) provide the best average demosaicing quality with low time complexity, on the IMAX dataset. Details are given in Section V.

Our paper is organized as follows. Section II briefly reviews MLRI and A+, as both underly our methods. Section III introduces our proposed post-processing method - ERP. Section IV further introduces our novel demosaicing methods DDR and FR. In section V, we discuss the choices of parameters and the experimental results. Finally, we conclude the paper in section VI.

II. Review of MLRI and A+

This section briefly reviews the two major sources of inspiration for our proposed methods: the MLRI demosaicing method [14] and the A+ super-resolution method [21].

A. Minimized-Laplacian Residue Interpolation (MLRI)

The MLRI method of Kiku *et al.* [14] is mainly motivated by the GBTF method of Pekkucuksen and Altunbasak [12]. MLRI includes two stages (see Fig. 3). Let $G_{x,y}$ and $R_{x,y}$ denote the raw values at position (x, y) for the green and red channels, resp.:

1) First Stage[1]*:* MLRI estimates the missing G values at locations with R information as well as the R values at

[1]Here we only discuss the estimation of the G values at R position in the horizontal direction, G values at B are handled similarly.

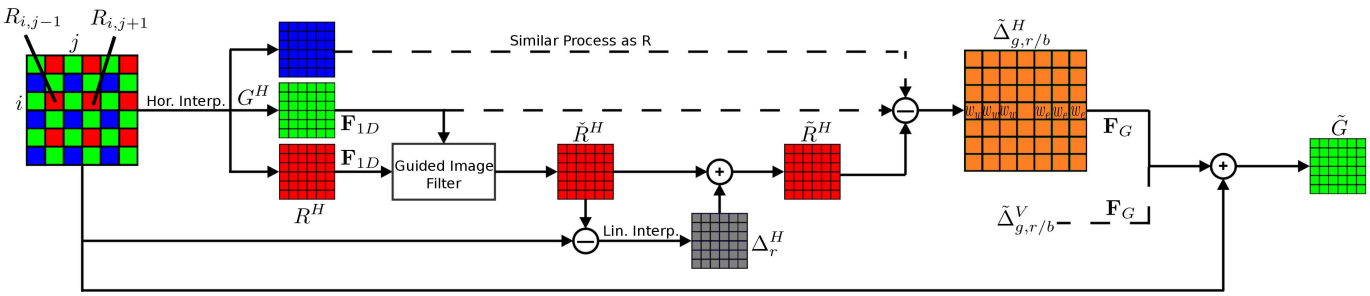


Fig. 3. G channel interpolation of MLRI.

locations with G information through linear interpolation. Assuming the raw value $G_{i,j}$ or $R_{i,j}$ is missing then we have,

$$G^H_{i,j} = (G_{i,j-1} + G_{i,j+1})/2, \quad R^H_{i,j} = (R_{i,j-1} + R_{i,j+1})/2. \tag{1}$$

Next, after computing the horizontal Laplacian of tentative R and G estimations by the 1D-filter

$$\mathbf{F}_{1D} = \begin{bmatrix} -1 & 0 & 2 & 0 & -1 \end{bmatrix}, \tag{2}$$

MLRI uses a modified version of guided image filters (GIF) [25] to obtain intermediate G values, meaning that the $\mathbf{F}_{1D}R^H$ is treated as the guided Laplacian for $\mathbf{F}_{1D}G^H$, so that the dilation coefficient $a_{i,j}$ is obtained,

$$a_{i,j} = \frac{\frac{1}{|\omega|}\sum_{(m,n)\in\omega_{i,j}}(\mathbf{F}_{1D}R^H_{m,n})(\mathbf{F}_{1D}G^H_{m,n}) - \overline{\mathbf{F}_{1D}R^H}_{i,j}\,\overline{\mathbf{F}_{1D}G^H}_{i,j}}{\sigma^2_{i,j} + \epsilon}, \tag{3}$$

where $\omega_{i,j}$ is a local image patch centered at pixel (i, j), $|\omega|$ is the number of pixels in $\omega_{i,j}$, $\sigma^2_{i,j}$ is the variance of $\mathbf{F}_{1D}R^H$ in $\omega_{i,j}$, ϵ is a regularization parameter.

The translation coefficient $b_{i,j}$ is obtained as follows,

$$b_{i,j} = \overline{G^H}_{i,j} - a_{i,j}\,\overline{R^H}_{i,j}, \tag{4}$$

where $\overline{G^H}_{i,j}$ and $\overline{R^H}_{i,j}$ are the mean values of G^H and R^H in $\omega_{i,j}$. The intermediate G value $\check{G}^H_{i,j}$ is further determined,

$$\check{G}^H_{i,j} = \frac{1}{|\omega|}\sum_{(k,l)\in\omega_{i,j}}(a_{k,l}R^H_{i,j} + b_{k,l}). \tag{5}$$

Under the assumption that the residues vary linearly in a small area, the smoothed residues Δ^H_g are estimated by linear interpolation

$$\Delta^H_g(i, j) = (G_{i,j-1} - \check{G}^H_{i,j-1})/2 + (G_{i,j+1} - \check{G}^H_{i,j+1})/2. \tag{6}$$

Correspondingly, the horizontally enhanced G values at the R locations are acquired by adding the tentative values $\check{G}^H$ and the interpolated residuals Δ^H_g. To get other enhanced R, B values at different positions MLRI applies the same modified GIF.

2) Second Stage: It starts with computing the tentative horizontal/vertical (h/v) color differences (G-R, G-B) $\tilde{\Delta}^{H,V}_{g,r/b}$

$$\tilde{\Delta}^{H,V}_{g,r/b}(i, j) = \begin{cases} \tilde{G}^{H,V}_{i,j} - R_{i,j} & \text{G is interpolated at R,} \\ \tilde{G}^{H,V}_{i,j} - B_{i,j} & \text{G is interpolated at B,} \\ G_{i,j} - \tilde{R}^{H,V}_{i,j} & \text{R is interpolated,} \\ G_{i,j} - \tilde{B}^{H,V}_{i,j} & \text{B is interpolated,} \end{cases} \tag{7}$$

where $\tilde{G}^{H,V}_{i,j}$, $\tilde{R}^{H,V}_{i,j}$ and $\tilde{B}^{H,V}_{i,j}$ are the above enhanced horizontal/vertical values. Then the color differences $\tilde{\Delta}_{g,r/b}$ are weighted and improved as

$$\begin{aligned} \tilde{\Delta}_{g,r/b}(i, j) = \{&\omega_s \mathbf{F}_G \tilde{\Delta}^V_{g,r/b}(i-4:i, j) \\ &+ \omega_n \mathbf{F}_G \tilde{\Delta}^V_{g,r/b}(i:i+4, j) \\ &+ \omega_w \tilde{\Delta}^H_{g,r/b}(i, j-4:j)\mathbf{F}^T_G \\ &+ \omega_e \tilde{\Delta}^H_{g,r/b}(i, j:j+4)\mathbf{F}^T_G\}/\omega_t, \end{aligned} \tag{8}$$

where $\mathbf{F}_G$ is the Gaussian weighted averaging filter

$$\mathbf{F}_G = \begin{bmatrix} 0.56 & 0.35 & 0.08 & 0.01 & 0 \end{bmatrix}, \tag{9}$$

$\omega_{n,s,e,w}$ are computed by color difference gradients and ω_t is the sum of $\omega_{n,s,e,w}$. Eventually, G values at R locations are obtained by

$$\tilde{G}_{i,j} = R_{i,j} + \tilde{\Delta}_{g,r/b}(i, j). \tag{10}$$

A similar derivation holds for the G values at B locations.

As to the R channel, MLRI computes the Laplacian of R and G values with the 2D-filter

$$\mathbf{F}_{2D} = \begin{bmatrix} 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 4 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 \end{bmatrix}. \tag{11}$$

Again, the modified GIF is applied. The R channel is guided by $\tilde{G}_{i,j}$ values. In the end, the output R values are enhanced by smoothing the residues as Eq. (6) indicates. The B channel goes through exactly the same process.

B. Adjusted Anchored Neighborhood Regression (A+)

A+ proposed by Timofte *et al.* [21] derives from and greatly enhances the performance of Anchored Neighborhood Regression (ANR) [26] for image super-resolution tasks. The algorithm contains two important stages:

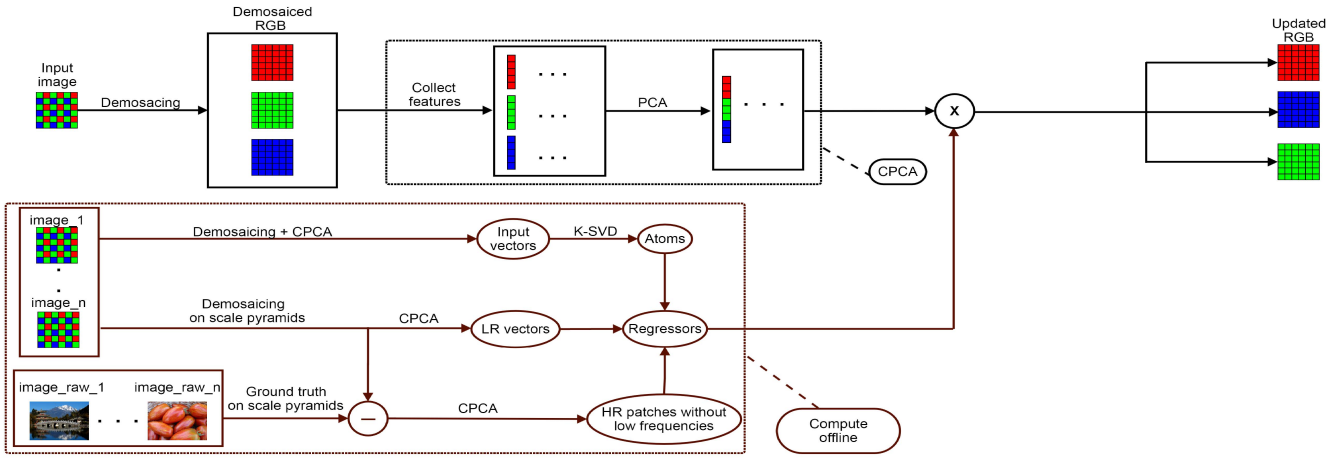


Fig. 4. Scheme of ERP.

1) Offline Stage: A+ uses Zeyde *et al.*'s algorithm [27] as a starting point, which trains a sparse dictionary from millions of low resolution (LR) patches collected from 91 training images [28]. To begin with, the LR images in the YCbCr color space are scaled up to the size of output high-resolution (HR) images by bicubic interpolation. In the next step, the upscaled LR image $\mathbf{y}_l$ is filtered by the first- and second-order gradients, and features $\{\tilde{\mathbf{p}}_l^k\}_k$ corresponding to LR patches of size 3×3 are collected accordingly. A+ projects them onto a low-dimensional subspace by PCA, discarding 0.1% of the energy. When it comes to the training, K-SVD [29], an iterative method that alternates between sparse coding of the examples and updating the dictionary atoms, is applied to solve the following optimization problem

$$\mathbf{D}_l, \{\mathbf{q}^k\} = \underset{\mathbf{D}_l, \{\mathbf{q}^k\}}{\operatorname{argmin}} \sum_k \|\mathbf{p}_l^k - \mathbf{D}_l \mathbf{q}^k\|^2 \quad \text{s.t. } \|\mathbf{q}^k\|_0 \leq L \quad \forall k. \tag{12}$$

where $\{\mathbf{p}_l^k\}_k$ are the training LR feature vectors, $\mathbf{q}^k$ are the coefficient vectors and $\mathbf{D}_l$ is the LR training dictionary. The training process of A+ goes through 20 iterations of the K-SVD algorithm, with 1024 atoms in the dictionary, and allocating $L = 3$ atoms per coefficient vector.

Instead of optimizing the reconstruction of high resolution (HR) patches at runtime, A+ uses offline trained anchored regressors to directly obtain them. More specifically, A+ uses the atoms of the trained dictionary $\mathbf{D}_l$ as anchors for the surrounding neighbourhood and the corresponding LR to HR patch regressor. A+ collects 5 million pairs of corresponding LR features and HR patches from a scaled pyramid of the 91 training images. For each anchor point (LR atom), A+ retrieves the nearest $n = 2048$ training samples. Due to the l_2-norm used in Eq. (12) the distance between the atom and its neighbor is also Euclidean, and all of the 5 million candidates are normalized by the l_2-norm. Then for an arbitrary input LR feature $\mathbf{y}$, A+ solves

$$\min_{\boldsymbol{\delta}} \{\|\mathbf{y} - \mathbf{S}_{l,\mathbf{y}}\boldsymbol{\delta}\|^2 + \lambda\|\boldsymbol{\delta}\|^2\}, \tag{13}$$

where $\mathbf{S}_{l,\mathbf{y}}$ is the matrix of 2048 nearest neighbors anchored to the atom $\mathbf{d}_\mathbf{y}$ and λ is set to be 0.1. 'Nearest' is measured by correlation. The algebraic solution of Eq. (13) is

$$\boldsymbol{\delta} = \mathbf{P}_{l,\mathbf{y}}\mathbf{y}, \quad \mathbf{P}_{l,\mathbf{y}} = (\mathbf{S}_{l,\mathbf{y}}^T\mathbf{S}_{l,\mathbf{y}} + \lambda\mathbf{I})^{-1}\mathbf{S}_{l,\mathbf{y}}^T, \tag{14}$$

where $\mathbf{I}$ is the unit matrix. As to the HR training images, the first thing is to remove the low frequencies by subtracting the upscaled corresponding LR image. Then, A+ collects 5 million 3×3 such HR patches corresponding each to LR patches. The HR patch values are further normalized by the corresponding l_2-norm of LR patch features. The anchored regressor $\mathbf{P}_\mathbf{y}$ corresponding to the atom $\mathbf{d}_\mathbf{y}$ is precomputed offline

$$\mathbf{P}_\mathbf{y} = \mathbf{S}_{h,\mathbf{y}}\mathbf{P}_{l,\mathbf{y}} = \mathbf{S}_{h,\mathbf{y}}(\mathbf{S}_{l,\mathbf{y}}^T\mathbf{S}_{l,\mathbf{y}} + \lambda\mathbf{I})^{-1}\mathbf{S}_{l,\mathbf{y}}^T, \tag{15}$$

where $\mathbf{S}_{h,\mathbf{y}}$ contains 2048 HR patches corresponding to LR features in $\mathbf{S}_{l,\mathbf{y}}$.

2) Online Stage: During this stage, the testing LR image is (as done in the offline stage) firstly scaled up to the target size by bicubic interpolation and the first- and second-order horizontal and vertical finite differences are calculated. After extracting the LR patch features (PCA projected), A+ searches the atom $\mathbf{d}_j$ in $\mathbf{D}_l$ with the highest correlation to each input LR feature $\mathbf{y}_j$, and the residual HR patch $\mathbf{x}_j$ without low frequencies is obtained by multiplication of the regressor $\mathbf{P}_j$ anchored to $\mathbf{d}_j$ with $\mathbf{y}_j$

$$\mathbf{x}_j = \mathbf{P}_j\mathbf{y}_j. \tag{16}$$

Subsequently the low frequencies are added. The HR patches are combined by averaging in the overlapping area to complete the output HR image.

III. Efficient Regression Priors (ERP)

Our ERP method is inspired by the A+ method introduced for image super-resolution. As a post-processing step, ERP has two major strengths. Firstly, it is capable of improving the results of many demosaicing methods. Especially MLRI+ERP combines low time complexity and good performance. Secondly, ERP trains offline a dictionary and regressors and, thus, allows for low computational times during testing. In the following we describe how ERP (see Fig. 4) is derived and

used for the post-processing of demosaiced images. ERP goes through two stages just as A+ does.

A. Offline Stage

ERP is trained using 100 high quality images collected from Internet for post-processing the results of a selected demosaicing method. The demosaicing method reconstructs the LR image. In the CPCA (Collection + PCA) step (see Fig. 4), the first and second-order finite differences of the G channel are extracted for the LR training images, in both the h/v direction,

$$\begin{aligned}\mathbf{F}_{1h} &= \begin{bmatrix}1 & -1\end{bmatrix} = \mathbf{F}_{1v}^T,\\ \mathbf{F}_{2h} &= \begin{bmatrix}1 & -2 & 1\end{bmatrix}/2 = \mathbf{F}_{2v}^T,\end{aligned} \tag{17}$$

so that we can keep information on edges and mosaic artifacts, and train a dictionary adapted to a specific demosaicing method. Small 3×3 regions at the same position of the filtered G channel are collected and concatenated to form one vector (feature), along with PCA dimensionality reduction with 99.9% preserved energy. After repeating the process for the R and B channels, the input vector $\mathbf{v}_l^k$ is eventually formed by three features at the same position of the RGB channels. This process is called CPCA step in Fig. 4.

Later, ERP applies the K-SVD [29] method as in [21], [26], and [27] to train an LR dictionary $\mathbf{W}_l$ with 4096 atoms:

$$\mathbf{W}_l = \underset{\mathbf{W}_l, \{\mathbf{c}^k\}}{\operatorname{argmin}} \sum_k \|\mathbf{v}_l^k - \mathbf{W}_l \mathbf{c}^k\|^2 \quad \text{s.t. } \|\mathbf{c}^k\|_0 \leq N \quad \forall k, \tag{18}$$

where $\{\mathbf{c}^k\}_k$ is the set of coefficient vectors. The training process goes through 20 iterations of the K-SVD algorithm, allocating $\mathrm{N} = 3$ atoms per coefficient vector. Here, the choice of 3 atoms and 20 iterations is based on A+, which shows good performances on the super-resolution task.

Assuming that the atoms are sparsely embedded in a manifold, it is natural to use input vectors $\{\mathbf{v}_l^k\}_k$ for densely sampling the manifold. Moreover, not only the input vectors $\{\mathbf{v}_l^k\}_k$ but also the LR vectors collected from the scaled pyramids of the LR training images can serve to better approximate the manifold. Here, the overall size scaling factors of the pyramids layers are of the form 0.98^p with levels $p = 0, \ldots, 11$. Thus, ERP selects 2048 nearest neighbors anchored to an atom from 5 million region/vector candidates, all of which are normalized by the l_2-norm. 'Nearest' is measured by correlation. In the following step, ERP computes $\{\mathbf{Q}_{l,i}\}_{i=1,\ldots,4096}$

$$\mathbf{Q}_{l,i} = (\mathbf{N}_{l,i}^T \mathbf{N}_{l,i} + \lambda \mathbf{I})^{-1} \mathbf{N}_{l,i}^T, \tag{19}$$

where $\mathbf{N}_{l,i}$ is the matrix of 2048 nearest neighbors anchored to the atom $\mathbf{w}_i$ and λ is set to be 0.1. As to the ground truth images, the first thing needed is to remove the low frequencies by subtracting the demosaiced LR image. Then, ERP collects 5 million high resolution (HR) patches without low frequencies corresponding to the previously collected LR candidate vectors. The HR candidates are further normalized by the corresponding l_2-norm of the LR candidates. Finally, the anchored regressor $\mathbf{Q}_i$ corresponding to the atom $\mathbf{w}_i$ is precomputed offline

$$\mathbf{Q}_i = \mathbf{N}_{h,i} \mathbf{Q}_{l,i} = \mathbf{N}_{h,i} (\mathbf{N}_{l,i}^T \mathbf{N}_{l,i} + \lambda \mathbf{I})^{-1} \mathbf{N}_{l,i}^T, \tag{20}$$

where $\mathbf{N}_{h,i}$ contains 2048 HR patches corresponding to 2048 nearest neighbors in $\mathbf{N}_{l,i}$.

B. Online Stage

The same demosaicing method is applied first at test time. Among the studied demosaicing methods, we consider MLRI to be the best match for ERP because of its low time complexity and good performance. MLRI can be used to independently interpolate RGB channels before applying ERP (see Fig. 4), or interpolate the G channel first, then guide the RB channels with the ERP updated G channel. No matter what the case may be, ERP searches the nearest neighbor atom $\mathbf{w}_j$ in $\mathbf{W}_l$ for a vector $\mathbf{v}_j$ of the input image with highest correlation; the output patch $\mathbf{y}_j$ is computed by multiplying regressor $\mathbf{Q}_j$ anchored to $\mathbf{w}_j$ and $\mathbf{v}_j$, which is indicated by the brown arrow confluencing with the black arrow in Fig. 4,

$$\mathbf{y}_j = \mathbf{Q}_j \mathbf{v}_j = \mathbf{N}_{h,j} \mathbf{Q}_{l,j} \mathbf{v}_j, \tag{21}$$

where $\mathbf{Q}_{l,j}\mathbf{v}_j$ is the algebraic solution of

$$\min_{\mathbf{x}} \{\|\mathbf{v}_j - \mathbf{N}_{l,j}\mathbf{x}\|^2 + \lambda \|\mathbf{x}\|^2\}. \tag{22}$$

After adding $\mathbf{y}_j$ to the low frequencies (the input demoisaced image) as well as averaging the overlapping area, the small patches are integrated into a complete output image.

IV. Directional Difference Regression (DDR) and Fused Regression (FR)

In this section, we make a couple of observations on the MLRI and ERP methods and introduce our proposed independent demosacing methods, DDR and FR.

A. Observations

MLRI computes the enhanced h/v differences (G-R, G-B) with the modified guided image filter and residual interpolation, which leads to a rather inaccurate estimation. To improve the h/v differences we follow the idea of regressor training. As described previously, ERP maps LR features into HR patches without low frequencies. We implement a similar idea and map inaccurate color differences into accurate color differences without low frequencies by offline trained regressors. MLRI also uses the Laplacian filtered G channel to guide the reconstruction of the R/B channels. The set of first and second-order finite differences highlight edge and blob-like profiles in the intensity patterns. Therefore we use them both in our methods.

ERP as an enhancement step can be applied to different demosaicing methods. This said, its performance depends on that choice. If we use bilinear interpolation as the starting point, the final performance is less impressive than that of ERP starting from a state-of-the-art method. This brings the question whether we can improve beyond the combination of ERP with any of the demosaicing methods that we described in the introduction. In what follows, we present two novel methods, both making use of ERP, but also improving on the initial demosaicing.

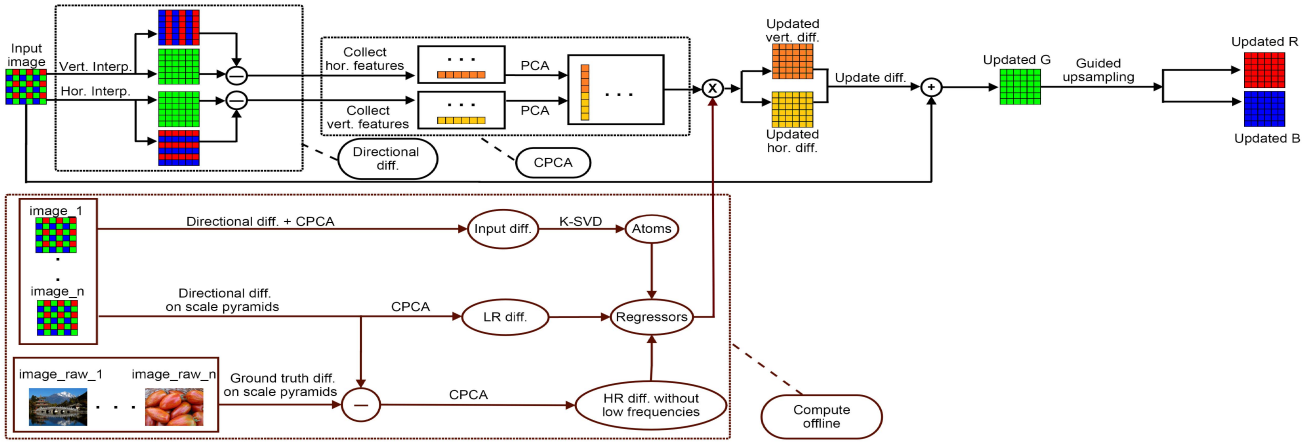


Fig. 5. Scheme of DDR.

B. Directional Difference Regression (DDR)

Our proposed DDR method has three steps (see Fig. 5). Due to space constraints, we discuss the horizontal case of three channels. Let $R_{x,y}$, $G_{x,y}$, $B_{x,y}$ to be the raw input image values at position (x, y):

(i) Without relying on sophisticated methods as the starting point, assume raw values $R_{i,j}$, $G_{i,j}$, $B_{i,j}$ are missing, based on Eq. (1) we use the simplest linear interpolation to obtain the tentative values $\bar{R}^H_{i,j}$, $\bar{G}^H_{i,j}$, $\bar{B}^H_{i,j}$ horizontally. Then the tentative horizontal color differences (G-R, G-B) $\bar{\Delta}^H_{g,r/b}$ are computed as

$$\bar{\Delta}^H_{g,r/b}(i,j) = \begin{cases} \bar{G}^H_{i,j} - R_{i,j} & \text{G is interpolated at R,} \\ \bar{G}^H_{i,j} - B_{i,j} & \text{G is interpolated at B,} \\ G_{i,j} - \bar{B}^H_{i,j} & \text{B is interpolated,} \\ G_{i,j} - \bar{R}^H_{i,j} & \text{R is interpolated.} \end{cases} \tag{23}$$

In the CPCA step (see Fig. 5), instead of applying gradient filters of h/v directions as ERP, we filter $\bar{\Delta}^H_{g,r/b}$ by horizontal $\mathbf{F}_{1h}, \mathbf{F}_{2h}$ (Eq. (17)), collect 3×3 regions at the same position of filtered $\bar{\Delta}^H_{g,r/b}$ and concatenate them to one vector (horizontal feature) respectively, along with PCA reduction.

The training images go through the same process, and we use the K-SVD method to iteratively compute the LR color (region) difference dictionary with 4096 atoms. Furthermore, we collect 5 million LR (region) differences from the scaled pyramids of LR training images, selecting 2048 LR differences most correlated to an atom as anchored neighbors. Correspondingly, we collect 5 million HR differences without low frequencies. Then, we compute the directional difference regressor $\mathbf{P_d}$ anchored to an atom $\mathbf{d}$

$$\mathbf{P_d} = \mathbf{S}_{h,\mathbf{d}}(\mathbf{S}^T_{l,\mathbf{d}}\mathbf{S}_{l,\mathbf{d}} + \lambda\mathbf{I})^{-1}\mathbf{S}^T_{l,\mathbf{d}}, \tag{24}$$

where $\mathbf{S}_{h,\mathbf{d}}$ is the matrix of 2048 HR differences corresponding to LR differences in $\mathbf{S}_{l,\mathbf{d}}$ and λ is 0.1. Finally, a tentative color difference in the input image is improved by offline computed regressors as follows

$$\mathbf{y}_{h,j} = \lambda_1 \mathbf{P}_j \mathbf{y}_{l,j}, \tag{25}$$

where $\mathbf{P}_j$ is the regressor anchored to the atom with highest correlation to the LR difference $\mathbf{y}_{l,j}$ and λ_1 is the regressor correction parameter. By adding HR differences to the channel-shared low frequencies we have the enhanced color differences $\hat{\Delta}^H_{g,r/b}$.

(ii) Now we come to the stage of color difference updating. Based on MLRI, we compute the weights $w_{n,s,e,w}$ and the sum of the weights $w_t = w_n + w_s + w_e + w_w$, where

$$\begin{aligned} w_e &= 1/(\sum_{a=i-1}^{i+1}\sum_{b=j}^{j+2} D^H_{a,b})^2, \; w_w = 1/(\sum_{a=i-1}^{i+1}\sum_{b=j-2}^{j} D^H_{a,b})^2, \\ w_s &= 1/(\sum_{a=i-2}^{i}\sum_{b=j-1}^{j+1} D^V_{a,b})^2, \; w_n = 1/(\sum_{a=i}^{i+2}\sum_{b=j-1}^{j+1} D^V_{a,b})^2, \end{aligned} \tag{26}$$

here $D^H_{i,j}$ and $D^V_{i,j}$ are given by

$$\begin{aligned} D^H_{i,j} &= \|\hat{\Delta}^H_{g,r/b}(i,j+1) - \hat{\Delta}^H_{g,r/b}(i,j-1)\|, \\ D^V_{i,j} &= \|\hat{\Delta}^V_{g,r/b}(i+1,j) - \hat{\Delta}^V_{g,r/b}(i-1,j)\|. \end{aligned} \tag{27}$$

We observe that MLRI uses the neighbor color differences weighted by a Gaussian filter (Eq. (9)) to alleviate the error of color differences in the center, and the neighbor size is fixed to be 4. Instead of a Gaussian filter, we prefer a simpler averaging filter to weigh the neighbor color differences

$$\mathbf{F}_k = \underbrace{[1/k \quad 1/k \quad \ldots \quad 1/k]}_{k}. \tag{28}$$

Finally, the color difference is updated as follows

$$\begin{aligned} \Delta_{g,r/b}(i,j) = \{ & w_s \mathbf{F}_k * \hat{\Delta}^V_{g,r/b}(i-k+1:i,j) \\ & + w_n \mathbf{F}_k * \hat{\Delta}^V_{g,r/b}(i:i+k-1,j) \\ & + w_w \hat{\Delta}^H_{g,r/b}(i,j-k+1:j) * \mathbf{F}^T_k \\ & + w_e \hat{\Delta}^H_{g,r/b}(i,j:j+k-1) * \mathbf{F}^T_k \}/w_t, \end{aligned} \tag{29}$$

where $\hat{\Delta}^V_{g,r/b}$ is the vertical enhanced color difference. By adding the ground truth R/B values and $\Delta_{g,r/b}$ we obtain the updated G values.

(iii) When it comes to the R channel, we apply the Laplacian filter (Eq. (11)) to obtain the tentative R values R^1, as well as the horizontal and vertical first-order difference filters $\mathbf{F}_h$, $\mathbf{F}_v$

$$\mathbf{F}_h = \begin{bmatrix} -1 & 0 & 0 & 0 & 1 \end{bmatrix} = \mathbf{F}_v^T, \tag{30}$$

and get the h/v R values R^2 and R^3. The above processes yield residues for the raw values of the R channel. For $k = 1, 2, 3$ consider

$$\epsilon_k(i, j) = \begin{cases} R_{i,j} - R^k_{i,j} & R_{i,j} \text{ is the raw value,} \\ 0 & \text{others.} \end{cases} \tag{31}$$

After using the bilinear filter

$$\mathbf{F}_b = \begin{bmatrix} 0.25 & 0.5 & 0.25 \\ 0.5 & 1 & 0.5 \\ 0.25 & 0.5 & 0.25 \end{bmatrix}, \tag{32}$$

we have the enhanced estimations of R values

$$\tilde{R}^k = \lambda_2 \mathbf{F}_b * \epsilon_k + R^k, \quad \text{for } k \in 1, 2, 3, \tag{33}$$

where λ_2 is the residue correction parameter. Thus, we have the Laplacian updated R value $\tilde{R}^1$. With the help of the previous weights $w_{n,s,e,w}$, we obtain the gradient updated color difference for $\tilde{R}^2$ and $\tilde{R}^3$

$$\begin{aligned} \Delta_r(i, j) = \{ & w_n \tilde{\Delta}^3_r(i-1, j) + w_s \tilde{\Delta}^3_r(i+1, j) \\ & + w_e \tilde{\Delta}^2_r(i, j-1) + w_w \tilde{\Delta}^2_r(i, j+1) \}/w_t, \end{aligned} \tag{34}$$

where

$$\tilde{\Delta}^k_r(i, j) = G_{i,j} - \tilde{r}^k_{i,j} \quad \text{for } k = 2, 3. \tag{35}$$

Then, calculating the ground truth G minus Δ_r, we have the updated R values $\tilde{R}^{2,3}$. The final R value is obtained simply by averaging $\tilde{R}^1$, $\tilde{R}^3$ and $\tilde{R}^3$. For the B values we follow the same process.

C. Fused Regression (FR)

As reported in [24], the PSNR performance significantly improves after the ERP post-processing step. Since 50% of the ground truth pixels are available in the G channel compared to only 25% for the R/B channels, the G channel is easier to enhance than the R/B channels. This means ERP works especially well on the G channel. The above observation motivates us to feed the ERP enhanced G channel into our DDR method, and so deriving our Fused Regression (FR) method. We train the regressors for the directional differences and for the MLRI demosaiced G value in the same training stage. In other words, besides the directional difference dictionaries and regressors which are trained in step **(i)** of DDR, we also train the LR dictionary and regressors for MLRI demosaiced G values, according to the **Offline stage** of ERP. After applying the ERP step to the G values of an input image demosaiced by MLRI, we obtain another updated G value at the online stage.

By simply averaging the two versions of updated G values we obtain the enhanced G values of FR. Our experiments will show us that the R/B guided image quality is highly related to the one of the G channel. The better the recovery of the G channel, the better is the R/B channel restoration. This is another crucial reason underlying the idea of fused regression. The running time of FR is merely marginally increased with respect to that of the DDR method, as also shown in the following experimental section.

V. Experiments

In this section we describe and discuss the datasets and the setup used to validate the parameters of our methods and to experimentally compare with the state-of-the-art demosaicing methods. The results are analyzed together with the limitations of our methods and future directions of improvement.

A. Datasets

1) Kodak: The Kodak dataset contains 24 images of size 512×768 pixels and photographic quality involving a variety of subjects in many locations under different lighting conditions. The images are either created by Kodak's professional photographers; or selected from the winners of the Kodak International Newspaper Snapshot Awards (KINSA). Such choice of images ensures the high-fidelity of the Kodak benchmark. Besides, it has valuable artistic merit. Another important factor about Kodak is that the images contain a large amount of constant intensity regions. Moreover, Kodak's use for testing by researchers has a long history. Therefore, the PSNR performances on Kodak are generally good and above 40dB on average.

2) IMAX: Besides the Kodak dataset, we also test our methods on another standard dataset, IMAX, which is also widely used for validation of demosaicing methods. IMAX contains 18 images of size 500x500 pixels and exhibits more color gradations than the Kodak images. IMAX is a newer dataset, and generally, considered to be more challenging. In fact, the reported PSNR performances on IMAX are a lot worse than those on Kodak, usually lower than 37dB on average. More importantly, the hue and saturation conditions of IMAX images are closer to the images acquired by current digital cameras.

3) RW: Despite the high-fidelity and artistic merit of Kodak and IMAX, they are not representative enough for the images taken by normal people. Because the color and composition of those images are biased to the artistic taste. Therefore, as a dataset complementary to the standard benchmarks, we also selected 500 real world (RW) color images with RGB channels as HR images, using the Google search engine. Dozens of keywords – such as nature, landscape, people, city – yielded images from daily life. We made sure that all the categories contain a similar amount of images. Then we added a Bayer pattern mask on them to obtain LR images.

As a result, we not only consider images of high visual quality as those in Kodak and IMAX, but also the products of everyday photography. Whereas regions with slowly varying color intensities tend to show good performance with interpolation methods, reconstructing high-quality outputs from 'busy' images is more difficult. Therefore, we also focus on images which are highly textured and have a rich color gamut.

B. Experimental Setup

1) DDR and FR: The Kodak images have relatively mild intensity shifts, while IMAX images are richer in detail and

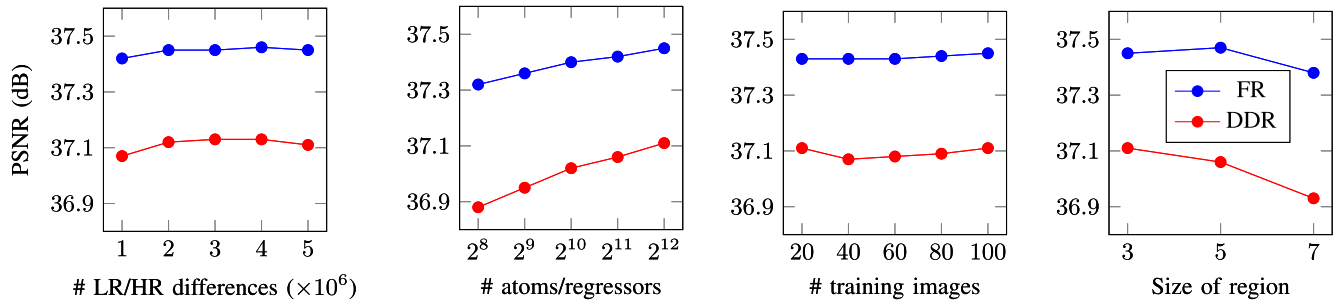


Fig. 6. PSNR versus number of training LR/HR differences, atoms/regressors, training images, and region size for the IMAX dataset.

high frequencies, which have a smaller number of neighboring pixels with similar color intensities, on average. Thus, for our methods we set the neighbor size (Eq. (28)) to 1 for IMAX and 4 for Kodak. Due to rich high frequencies of the IMAX images, it is difficult for IMAX to benefit from linear regression. Therefore, we set the regressor correction parameter λ_1 (Eq. (25)) to 1 for IMAX and 1.5 for Kodak. IMAX and RW share the same parameters, since RW images also show obvious intensity shifts. As to the residual correction parameter, we optimized it on several arbitrarily selected training images, and we fix $\lambda_2 = 1.2$ (Eq. (33)) for all datasets.

2) ERP: The Kodak dataset has been used for decades and most of the state-of-the-art methods have already achieved good performances ($\sim$ 40dB PSNR). So there is not so much space left for any ERP enhancement. In order to make the final results comparable, we multiply $\mathbf{y}_j$ of Eq. (21) by the regressor correction parameter $\gamma_1 = 0.5$ for all compared methods. Due to the same reason, we use a small residual correction parameter $\gamma_2 = 0.5$ for all methods. On IMAX the average PSNR results achieved by the compared methods are less impressive than on Kodak, and are lower than 37dB. This is the reason we set γ_1 and γ_2 to the larger value of 1.5 in case of the IMAX dataset. We refer to our previous work [24] for more ERP experiments.

3) Compared Methods: We compare our DDR and FR methods to BILINEAR, HQL [5], AHD [18], AP [17], PCSD [11], DLMMSE [6], LPAICI [7], GBTF [12], MSG [13], LDINAT [16], MLRI [14], and LSSC [19]. Unfortunately neither the code nor output images of SAPCA [20] and AVSC [15] are available to us, so we cannot reproduce their results. We refer to the introduction Section I for the brief description of the methods.

4) Default Settings: In all our experiments, if not stated otherwise, we use the following default parameters for the DDR, FR, and ERP methods: 4096 atoms/regressors, 2048 nearest neighbors for learning each anchored regressor, 3×3 region size, 100 training images, 5 million training candidates/regions. We keep the same 100 high-quality training images for all above demosaicing methods during the experiments on the three datasets. With this training set we ensure the relevance of our training dictionary and its anchored regressors.

5) Performance Measures: In order to evaluate the performance of the demosaicing methods we employ the standard Peak-Signal-to-Noise-Ratio (PSNR), the Structural Similarity Index (SSIM) [30], the Zipper Effect Ratio (ZER) [31], and the runtime at test. All the compared methods along with our proposed methods share the same testing environment – Intel(R) Core^{TM} i7-930 @2.80GHz with 8 GB RAM. PSNR measures quantitatively the fidelity of the restoration in comparison with the ground truth, while SSIM measures the structural similarity with the ground truth and ZER the ratio of pixels affected by the zipper effect or edge blurring.

C. Parameters

The main parameters that influence the PSNR performance of our proposed DDR and FR demosaicing methods are evaluated in Fig. 6 on the IMAX dataset.[2] Besides the 100 shared training images, we collect 4 other sets. The results achieved with the 5 training sets are between 37.01dB and 37.17dB for DDR and between 37.41dB and 37.49dB for FR on IMAX. On the Kodak dataset, the performances are 41.06dB - 41.10dB for DDR and 41.01dB - 41.08dB for FR. The variance is very small; a training set of 100 images offers a stable performance. We report the mean performance for these 5 training sets for all curves in Fig. 6.

1) Number of Differences: The performance changes slightly when increasing the collected (region) differences from 1 to 5 million. For example, the mean PSNR of DDR varies from 37.07dB to 37.13dB, less than 0.06dB. We believe that after collecting 1 million differences, the results are sufficiently relevant.

2) Number of Atoms/Regressors: As shown in Fig. 6 the PSNR performance of our methods improves with the number of atoms/regressors. The more atoms we train, the more chances we have to better approximate the input LR differences with anchored regressors. However, we should point out that the improvements gained by increasing the atoms tend to be trivial, that is, we can observe more obvious improvements when raising the atom number from 16 to 4096 than from 4096 to 8192. There is a speed/quality trade-off.

3) Number of Training Images: When we increase the number of training images from 20 to 100, as shown in Fig. 6, the mean PSNR only slightly grows. Here, we ignore the unstable starting point at 20 for DDR. We believe that 20 images form a rather small training pool lack statistical significance, therefore we fix by default to 100 images the training pool in all our experiments. The results confirm that the training set

[2]On the Kodak and RW datasets the performances show a similar pattern.

TABLE I

PSNR PERFORMANCE COMPARISON OF 14 DEMOISAICING METHODS ON 3 DATASETS. THE BEST TWO RESULTS ARE IN BOLD

Method	IMAX					Kodak					RW				
	R	G	B	All	Time (s)	R	G	B	All	Time (s)	R	G	B	All	Time (s)
BILINEAR	31.72	35.41	31.27	32.36	**0.02**	29.30	33.19	29.24	30.22	**0.04**	26.85	30.99	27.16	27.92	**0.02**
HQL	34.02	37.57	33.03	34.46	**0.02**	34.85	39.08	34.74	35.80	**0.02**	30.80	34.98	30.93	31.84	**0.01**
AHD	33.06	37.00	32.17	33.53	38.96	37.02	39.67	37.33	37.79	59.54	31.51	35.20	31.61	32.36	19.39
AP	32.85	34.92	32.01	33.06	0.46	38.03	41.56	38.64	39.11	0.74	31.72	34.79	31.78	32.49	0.25
PCSD	34.66	38.12	33.46	34.94	1.21	38.07	40.55	38.30	38.78	1.45	32.72	36.12	32.85	33.55	0.92
DLMMSE	34.06	38.00	33.04	34.49	27.13	39.19	42.63	39.58	40.12	42.50	32.81	36.88	32.89	33.72	12.84
LPAICI	34.40	37.87	33.30	34.73	0.61	39.68	43.04	39.95	40.56	1.00	33.01	36.80	33.02	33.86	0.33
GBTF	34.05	37.37	33.09	34.42	10.60	39.69	43.36	40.03	40.64	16.42	33.11	37.03	33.15	33.99	5.28
MSG	34.44	37.68	33.40	34.76	9.65	40.09	43.81	40.35	41.02	14.89	33.50	37.37	33.51	34.36	4.73
LDINAT	36.28	39.76	34.39	36.21	1523.63	37.02	39.47	37.15	37.72	2418.24	32.84	36.33	32.95	33.67	717.23
MLRI	36.62	40.03	35.43	36.80	0.54	39.33	42.92	39.63	40.24	0.74	33.76	37.44	33.98	34.66	0.26
LSSC	35.90	38.69	34.64	36.05	453.58	**40.60**	**44.42**	**40.74**	**41.52**	707.59	33.93	37.56	33.91	34.76	258.71
DDR (ours)	**37.12**	**40.34**	**35.63**	**37.17**	6.31	**40.18**	**43.92**	**40.39**	**41.10**	9.60	**34.12**	**37.53**	**34.26**	**34.95**	3.17
FR (ours)	**37.50**	**41.01**	**35.82**	**37.49**	10.75	40.19	43.85	40.34	41.07	16.35	**34.29**	**37.85**	**34.42**	**35.13**	5.20

TABLE II

PSNR PERFORMANCE COMPARISON AFTER ERP POST-PROCESSING ON 3 DATASETS. THE BEST TWO RESULTS ARE IN BOLD

Method	IMAX					Kodak					RW				
	R	G	B	All	Time (s)	R	G	B	All	Time (s)	R	G	B	All	Time (s)
BILINEAR	36.36	40.05	35.14	36.66		32.26	35.65	32.00	33.01		32.72	36.54	33.08	33.74	
HQL	36.62	40.13	35.19	36.78		36.70	40.50	36.52	37.54		33.37	37.55	33.47	34.33	
AHD	34.53	39.27	33.35	34.96		37.46	40.09	37.66	38.19		31.94	36.39	32.01	32.91	
AP	36.86	40.28	35.19	36.89		38.82	42.65	39.07	39.80		33.20	37.41	33.25	34.15	
PCSD	37.19	40.40	35.59	37.21		38.55	41.21	38.61	39.24		33.74	37.62	33.82	34.65	
DLMMSE	37.23	40.53	35.57	37.23		39.57	42.88	39.77	40.41		33.98	38.02	34.00	34.88	
LPAICI	37.29	40.66	35.55	37.26	+8.35	39.94	43.18	40.01	40.73	+12.74	33.97	37.96	33.94	34.85	+4.01
GBTF	37.26	40.63	35.62	37.28		39.96	43.46	40.11	40.81		34.06	38.09	34.06	34.95	
MSG	37.28	40.63	35.60	37.28		40.23	43.83	40.32	41.08		34.14	38.20	34.14	35.03	
LDINAT	37.50	40.54	35.42	37.24		37.71	40.23	37.65	38.34		33.81	37.50	33.74	34.62	
MLRI	37.81	40.93	36.08	37.72		39.60	43.17	39.80	40.47		34.24	38.07	34.37	35.16	
LSSC	37.52	40.74	35.90	37.53		**40.68**	**44.41**	**40.68**	**41.54**		**34.49**	**38.47**	**34.54**	**35.40**	
DDR (ours)	**38.00**	**40.76**	**36.20**	**37.82**		40.25	43.91	**40.40**	41.13		34.46	38.29	**34.54**	35.35	
FR (ours)	**38.15**	**41.03**	**36.26**	**37.95**		**40.27**	**43.93**	40.38	**41.14**		**34.57**	**38.42**	**34.66**	**35.47**	

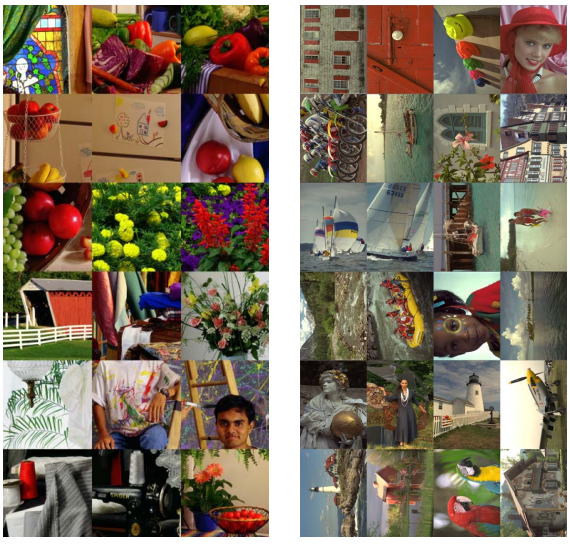
containing 100 images is representative and large enough to collect millions of LR/HR differences.

4) Size of the Region: As to the size of the region, we witness a clear quality drop when going up to 7×7 regions, not to mention the increase in runtime. For 5×5 regions the DDR and FR methods behave differently. Clearly, 3×3 appears to be the most cost-effective choice in our methods.

D. Results

In order to rule out boundary effects, we shave off 2, 4, 6, 8 boundary pixels for all the methods. The compared methods are stable on the boundary, only for DLMMSE we need to cut 6 boundary pixels to reach good stable performance.

1) PSNR: In Tables I and II we report the best PSNR results from the above discussed 4 candidates for 14 demosaicing methods. The best two results are in bold. Our DDR and FR methods are the best on both the IMAX and RW datasets. For instance, the mean PSNR of DDR is 37.17dB on IMAX, 1.12dB higher than the state-of-the-art method LSSC, while FR reaches an improvement of 1.34dB over LSSC. At the same time (see Fig. 2), DDR is almost **80** times faster than LSSC and **250** times faster than LDINAT. The ranking is preserved on the 500 images of the RW dataset. Our proposed methods are among the fastest demosaicing methods, while performing significantly better.



(a) (b)

Fig. 7. The IMAX and Kodak datasets. (a) IMAX dataset. (b) Kodak dataset.

2) SSIM and ZER: Complementary to the PSNR results, we report ZER [31] (computed on the CIELAB color space) and SSIM [30] results on each image of the IMAX and Kodak datasets (see Table III and IV). The listed results correspond

TABLE III
IMAX PER IMAGE RESULTS. THE BEST IS IN BOLD

Image	Measure	DDR (ours)	FR (ours)	LSSC	MLRI	LDINAT
1	PSNR	29.5787	**29.8766**	28.3874	29.3128	29.0508
	ZER	**0.2504**	0.2512	0.2885	0.2702	0.2518
	SSIM	0.9707	**0.9724**	0.9648	0.9696	0.9683
2	PSNR	39.3412	**39.6974**	38.2949	38.8945	38.7399
	ZER	0.0540	**0.0532**	0.0631	0.0654	0.0544
	SSIM	0.9850	0.9856	**0.9870**	0.9844	0.9858
3	PSNR	40.1342	**40.4334**	39.4959	39.9989	39.4450
	ZER	0.0549	**0.0532**	0.0660	0.0683	0.0564
	SSIM	0.9889	**0.9894**	0.9882	0.9891	0.9876
4	PSNR	39.6395	**40.1709**	38.0073	39.6490	38.8778
	ZER	0.0290	**0.0263**	0.0429	0.0348	0.0399
	SSIM	0.9984	**0.9986**	0.9977	0.9984	0.9980
5	PSNR	41.0852	**41.2874**	40.1714	40.6218	40.7822
	ZER	0.0158	**0.0152**	0.0220	0.0208	0.0182
	SSIM	0.9983	**0.9984**	0.9979	0.9981	0.9981
6	PSNR	39.1238	**39.4262**	38.3310	39.0300	38.6947
	ZER	0.0512	0.0514	0.0495	0.0564	**0.0478**
	SSIM	0.9826	**0.9829**	0.9825	0.9815	0.9827
7	PSNR	39.3334	**39.6756**	38.6516	39.1818	38.8972
	ZER	0.0478	0.0483	0.0589	0.0575	**0.0455**
	SSIM	0.9856	0.9860	**0.9866**	0.9844	0.9855
8	PSNR	34.5391	**35.3247**	32.2477	35.2574	33.4692
	ZER	0.1547	**0.1411**	0.1936	0.1749	0.1699
	SSIM	0.9886	0.9902	0.9838	**0.9906**	0.9856
9	PSNR	33.7582	**34.2723**	32.4733	33.2218	32.8558
	ZER	0.2353	0.2292	0.2704	0.2640	**0.2163**
	SSIM	0.9843	**0.9859**	0.9804	0.9830	0.9804
10	PSNR	35.7958	**36.2427**	34.7990	36.0956	34.9374
	ZER	**0.1027**	0.1029	0.1124	0.1149	0.1204
	SSIM	0.9735	0.9750	0.9719	**0.9751**	0.9716
11	PSNR	35.3552	**35.6413**	34.4012	35.2228	35.0137
	ZER	0.1246	0.1253	0.1369	0.1355	**0.1232**
	SSIM	0.9624	**0.9638**	0.9614	0.9613	0.9621
12	PSNR	34.5080	**34.5717**	33.1745	33.8563	32.6291
	ZER	**0.1417**	0.1440	0.1728	0.1681	0.1970
	SSIM	0.9622	**0.9628**	0.9567	0.9605	0.9567
13	PSNR	38.9513	**39.2534**	37.1949	37.8776	36.0160
	ZER	0.0426	**0.0424**	0.0751	0.0564	0.0848
	SSIM	**0.9628**	0.9616	0.9606	0.9606	0.9600
14	PSNR	34.8557	**35.1883**	32.4814	34.4732	34.1576
	ZER	0.1102	**0.1087**	0.1483	0.1213	0.1151
	SSIM	0.9743	**0.9753**	0.9674	0.9725	0.9720
15	PSNR	38.9375	**39.4944**	37.8871	38.6092	37.8008
	ZER	0.0770	**0.0715**	0.0941	0.0891	0.0734
	SSIM	0.9705	**0.9722**	0.9692	0.9692	0.9666
16	PSNR	37.3275	36.8685	**39.5520**	36.8480	36.0186
	ZER	0.0890	0.1081	**0.0309**	0.1107	0.1361
	SSIM	0.9446	0.9410	**0.9616**	0.9418	0.9342
17	PSNR	39.2806	**39.4434**	36.4009	37.5119	37.4848
	ZER	**0.0498**	0.0519	0.0611	0.0636	0.0692
	SSIM	0.9130	0.9151	**0.9199**	0.9141	0.9089
18	PSNR	37.5248	**38.0197**	37.0091	36.7090	36.8613
	ZER	0.0739	**0.0712**	0.0878	0.0931	0.0778
	SSIM	0.9914	**0.9921**	0.9910	0.9902	0.9906
AVG	PSNR	37.1705	**37.4938**	36.0534	36.7984	36.2073
	ZER	0.0947	**0.0942**	0.1097	0.1092	0.1054
	SSIM	0.9743	**0.9749**	0.9738	0.9736	0.9719

TABLE IV
KODAK PER IMAGE RESULTS. THE BEST IS IN BOLD

Image	Measure	DDR (ours)	FR (ours)	LSSC	MLRI	LDINAT
1	PSNR	40.0882	39.7031	**41.3972**	39.2617	34.4912
	ZER	0.0274	0.0354	**0.0157**	0.0479	0.2114
	SSIM	0.9946	0.9938	**0.9964**	0.9929	0.9766
2	PSNR	41.4867	41.6328	**42.2242**	40.9048	39.8064
	ZER	0.0145	**0.0143**	0.0156	0.0187	0.0399
	SSIM	0.9995	0.9995	**0.9997**	0.9994	0.9990
3	PSNR	44.0570	44.1495	**44.4702**	42.8711	41.8840
	ZER	0.0119	**0.0116**	0.0119	0.0163	0.0324
	SSIM	**0.9969**	0.9969	0.9960	0.9963	0.9954
4	PSNR	41.4366	41.5841	**42.5509**	40.4821	39.7177
	ZER	0.0158	**0.0156**	0.0161	0.0231	0.0514
	SSIM	0.9986	0.9986	**0.9988**	0.9979	0.9969
5	PSNR	39.3056	39.3367	**39.7253**	37.8291	35.8183
	ZER	0.0479	0.0483	**0.0449**	0.0682	0.1485
	SSIM	0.9953	0.9953	**0.9957**	0.9931	0.9878
6	PSNR	41.4361	41.3260	**41.8365**	40.6487	35.8244
	ZER	0.0164	0.0173	**0.0140**	0.0236	0.1281
	SSIM	0.9963	0.9963	**0.9970**	0.9955	0.9840
7	PSNR	43.6870	43.8461	**44.1721**	42.4423	41.5526
	ZER	0.0147	0.0140	**0.0136**	0.0206	0.0360
	SSIM	0.9979	**0.9980**	0.9978	0.9972	0.9967
8	PSNR	37.5114	37.3933	**37.6831**	36.7108	32.6177
	ZER	**0.0511**	0.0546	0.0517	0.0690	0.2027
	SSIM	**0.9887**	0.9881	0.9874	0.9863	0.9716
9	PSNR	43.8016	43.7787	**43.9857**	43.1668	40.5876
	ZER	0.0073	**0.0071**	0.0078	0.0096	0.0347
	SSIM	**0.9792**	0.9782	0.9764	0.9761	0.9686
10	PSNR	43.3092	**43.3516**	43.3361	42.5853	40.8827
	ZER	0.0085	**0.0083**	0.0089	0.0109	0.0292
	SSIM	**0.9882**	0.9880	0.9870	0.9867	0.9813
11	PSNR	41.3479	41.2523	**41.6387**	40.5579	36.9755
	ZER	0.0207	0.0228	**0.0200**	0.0285	0.0909
	SSIM	**0.9856**	0.9854	0.9801	0.9838	0.9750
12	PSNR	44.6544	44.7106	**45.0720**	43.8345	41.0818
	ZER	0.0074	0.0073	**0.0066**	0.0096	0.0295
	SSIM	0.9979	0.9979	**0.9981**	0.9973	0.9936
13	PSNR	36.1702	35.7880	**36.6171**	35.2809	30.7352
	ZER	0.0885	0.1048	**0.0767**	0.1262	0.3267
	SSIM	0.9904	0.9894	**0.9915**	0.9880	0.9688
14	PSNR	38.2368	38.3700	**39.0632**	37.0632	36.2735
	ZER	**0.0477**	0.0485	0.0492	0.0606	0.1190
	SSIM	**0.9919**	0.9916	0.9915	0.9898	0.9805
15	PSNR	40.1363	40.1631	**41.6871**	39.1055	38.5148
	ZER	0.0276	0.0289	**0.0245**	0.0374	0.0635
	SSIM	0.9905	0.9905	**0.9908**	0.9879	0.9869
16	PSNR	44.7720	44.7449	**45.0422**	44.2519	39.0930
	ZER	0.0050	0.0055	**0.0045**	0.0077	0.0744
	SSIM	0.9931	0.9932	**0.9934**	0.9924	0.9779
17	PSNR	**42.4419**	42.4173	42.2897	41.8726	39.0506
	ZER	**0.0097**	0.0102	0.0102	0.0131	0.0480
	SSIM	**0.9838**	0.9836	0.9815	0.9819	0.9745
18	PSNR	38.1835	38.0551	**38.3631**	37.4112	34.7181
	ZER	0.0488	0.0510	**0.0458**	0.0623	0.1342
	SSIM	**0.9805**	0.9801	0.9797	0.9775	0.9675
19	PSNR	42.0490	41.9158	**42.4187**	41.2460	37.5344
	ZER	**0.0112**	0.0130	0.0113	0.0184	0.1041
	SSIM	**0.9857**	0.9856	0.9850	0.9840	0.9778
20	PSNR	**42.3627**	42.3495	42.2987	41.4251	39.3560
	ZER	**0.0158**	0.0165	0.0165	0.0219	0.0542
	SSIM	0.9317	0.9327	0.8686	0.9194	**0.9350**
21	PSNR	40.5918	40.3939	**40.7610**	39.7414	35.9479
	ZER	0.0257	0.0294	**0.0228**	0.0385	0.1315
	SSIM	**0.9923**	0.9921	0.9923	0.9909	0.9826
22	PSNR	39.4463	**39.5640**	39.4511	38.8388	37.1633
	ZER	0.0329	**0.0329**	0.0352	0.0412	0.0856
	SSIM	0.9867	**0.9868**	0.9865	0.9847	0.9816
23	PSNR	44.0747	44.2184	**44.5398**	43.2718	42.3016
	ZER	0.0101	0.0097	**0.0097**	0.0124	0.0226
	SSIM	**0.9961**	0.9961	0.9956	0.9953	0.9943
24	PSNR	35.7649	35.6556	**35.9736**	34.9547	33.2356
	ZER	**0.0564**	0.0590	0.0581	0.0672	0.1238
	SSIM	**0.9800**	0.9790	0.9721	0.9771	0.9733
AVG	PSNR	41.0980	41.0708	**41.5249**	40.2399	37.7152
	ZER	0.0260	0.0278	**0.0246**	0.0355	0.0968
	SSIM	**0.9884**	0.9882	0.9850	0.9863	0.9803

to the images in Fig. 7 from left to right and top to bottom for each dataset. Fig. 9 shows that the ZER relative ranking of the top 5 methods, together with FR + ERP, stays the same for a large range of values for the ZER threshold. The ZER values in Tables III and IV are obtained by fixing the threshold to 2.5. The results for different measures confirm the top performance achieved by the proposed DDR/FR and ERP.

3) ERP Post-Processing: By applying the ERP post-processing step we significantly boost the performance of most methods, especially on IMAX, as shown in Table II, at the cost of some extra seconds. The improvements on the IMAX dataset vary from +0.46dB to +4.36dB (see Table II). Broadly speaking, the worse the initial demosaicing methods, the larger the improvements. As to our post-processing method on MLRI, the PSNR is almost 1dB better than the original MLRI and outperforms all the other compared methods on IMAX, even when equipped with ERP. DDR with ERP achieves 37.82dB while 37.88dB is the best result to date, reported for

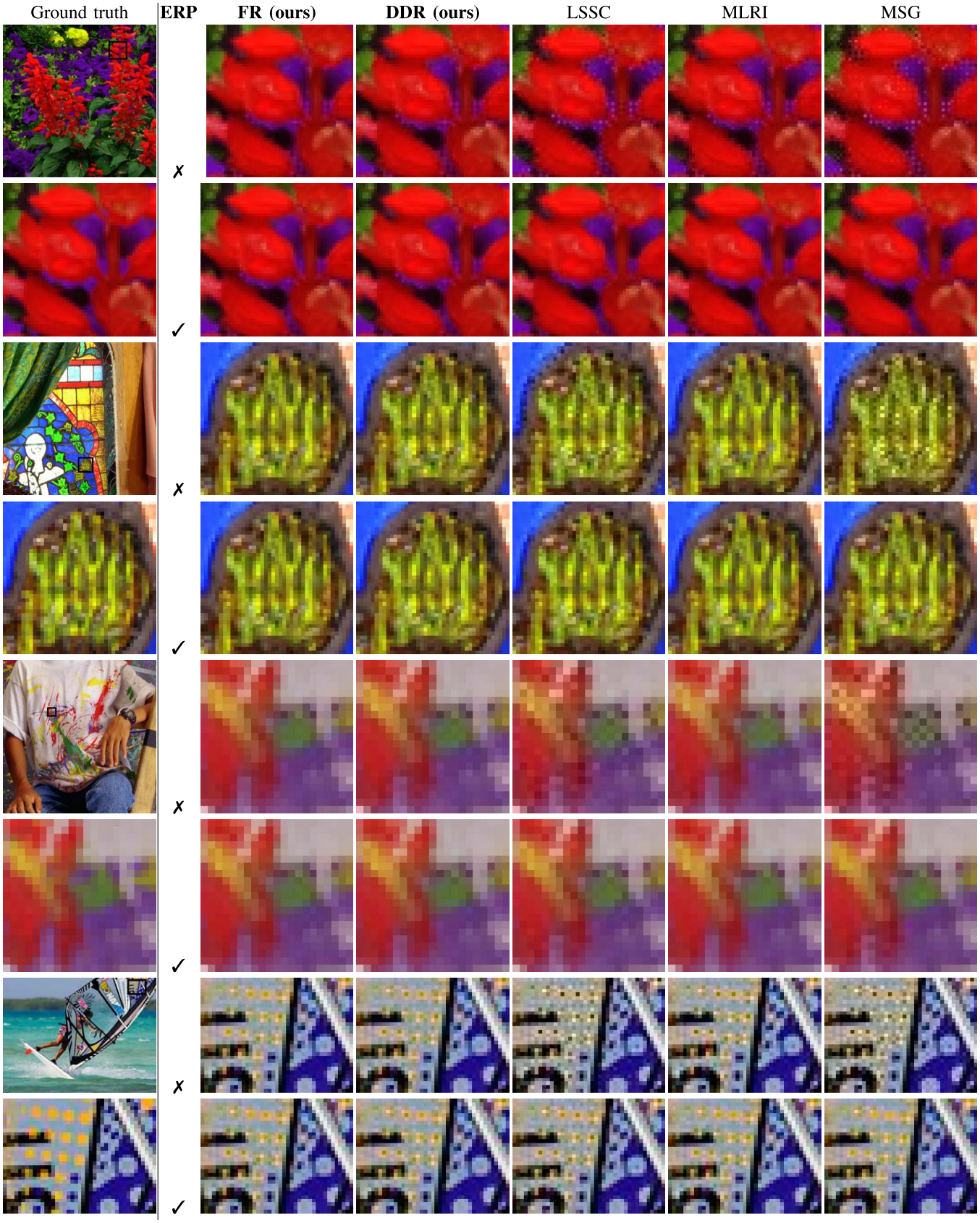


Fig. 8. Visual assessment of demosaicing results with and without ERP post-processing.

SAPCA. FR even slightly improves over SAPCA by achieving **37.95db**. FR with the ERP step takes less than **20** seconds while SAPCA costs 20 minutes. To sum up, all our proposed methods MLRI + ERP and DDR/FR + ERP achieve very good results, comparable to SAPCA on IMAX, while being significantly faster than SAPCA (about 2 orders of magnitude).

When it comes to the Kodak dataset, the improvement of our post-processing methods is less impressive. It varies from +2.79dB for BILINEAR to as low as +0.02dB for LSSC. This is mainly due to the fact that on the Kodak dataset many methods have achieved impressive results above 40dB, and hence, there may not be much space left for further improvement. Still, our ERP methods achieve 40.47dB/40.63dB on 24 Kodak images, 0.23dB/0.39dB higher than MLRI.

Also note that, if we apply the ERP regression training method multiple times, we repeatedly benefit, which is further confirmed by the results presented in [24].

4) Visual Comparison: For visual quality assessment we show a couple of image results of the top methods in Fig. 8. For example, in the image 'Flower' we clearly observe false color artifacts near red petals for the other methods. Our DDR method accomplishes good improvements over compared methods, and FR makes further progress to show natural transition near red petals. The comparison on the image 'T-shirt' also confirms the experimental results. First of all, the zippering effects on LSSC demosaiced images is quite obvious near the edges of color stains. In contrast, both our methods have output images very close to the ground truth, and one can barely observe any zippering effects. Finally, the 'Sail' image from the RW dataset demonstrates the effectiveness of our methods. The false black colors in the yellow balls generated by LSSC and other methods are only weakly visible for FR and DDR. In conclusion, DDR and FR indeed provide natural-looking images close to the ground truth, while the other methods exhibit stronger color artifacts. Moreover, the visual performance is consistent with the numerical PSNR results presented in tables I and II. Last but not least, the visual artifacts are generally alleviated by the ERP post-processing.

E. Limitations and Future Work

1) Self-Similarities: Our DDR and FR methods rely on trained priors and do not exploit the self-similarities and the particular content of the input image. LSSC does exploit the self-similarities and is capable to achieve 0.4dB better PSNR performance than our methods on the Kodak dataset, but not on the IMAX dataset and the RW dataset. We believe this to be caused by the particularities of the Kodak dataset with respect to the other datasets, such as larger flat regions and larger images, that better suit the LSSC method. The 16th IMAX image is the only in that dataset where LSSC achieves superior demosaicing results than our methods. It differs from the other IMAX images by the highly regular texture content, a perfect fit for LSSC. The use of self-similarities is a direction for further performance improvement of our methods.

2) Design Choices: All our methods (ERP, DDR, FR) follow closely the settings of the A+ super-resolution method [21]. The effect of the patch features and training procedure on the overall performance are unclear. If we were to train regressors specific to the different offsets with respect to the underlying mosaic pattern, further improvements are to be expected. As shown in [24], cascading the demosaicing methods such that each cascade stage starts from the demosaicing result of the previous cascade stage is another direction for future research.

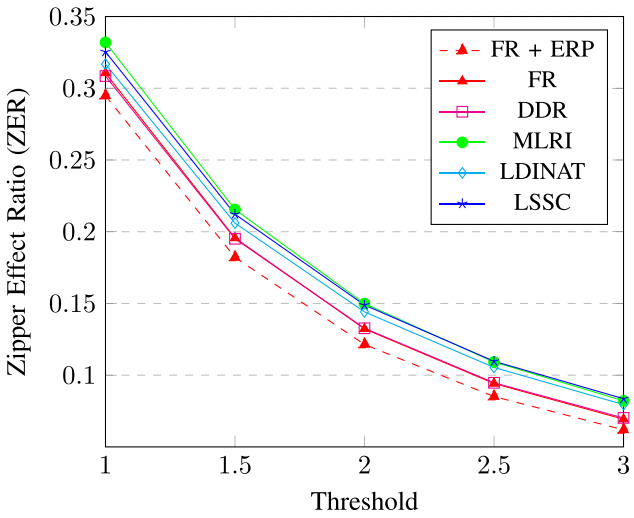


Fig. 9. Zipper Effect Ratio (ZER) vs. threshold on IMAX.

3) Time Complexity: The proposed methods are highly parallelizable, but the time complexity depends linearly on the number of regressors and anchors in the dictionary. However, the use of a better sublinear data search structure instead of the current linear search is rather straightforward and can lower the computation time [32].

VI. Conclusions

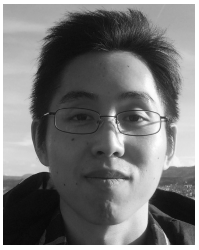
We propose[3] a novel fast demosaicing method based on directional difference regression (DDR) where the regressors are offline learned on training data and its enhanced version based on fused regression (FR), along with an efficient regression priors (ERP) post-processing step. We keep time complexity limited during the online stage and shift the learning and the bulk of computations to the offline stage. Thus, we achieve order(s) of magnitude lower running times at testing time than the state-of-the-art LSSC, LDINAT, and SAPCA methods. Moreover, the experimental results on various datasets prove competitive performances. Last but not least, the performance of the proposed DDR and FR methods, and of any other demosaicing method can be further improved by applying our ERP post-processing method.

References

[1] B. E. Bayer, "Color imaging array," U.S. Patent 3 971 065, Jul. 20 1976.
[2] C. Bai, J. Li, Z. Lin, J. Yu, and Y.-W. Chen, "Penrose demosaicking," *IEEE Trans. Image Process.*, vol. 24, no. 5, pp. 1672–1684, May 2015.
[3] X. Li, B. Gunturk, and L. Zhang, "Image demosaicing: A systematic survey," *Proc. SPIE*, vol. 6822, p. 68221J, Jan. 2008.
[4] D. Menon and G. Calvagno, "Color image demosaicking: An overview," *Signal Process., Image Commun.*, vol. 26, nos. 8–9, pp. 518–533, Oct. 2011.
[5] H. S. Malvar, L.-W. He, and R. Cutler, "High-quality linear interpolation for demosaicing of Bayer-patterned color images," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, vol. 3. May 2004, p. iii-485.
[6] D. Zhang and X. Wu, "Color demosaicking via directional linear minimum mean square-error estimation," *IEEE Trans. Image Process.*, vol. 14, no. 12, pp. 2167–2178, Dec. 2005.
[7] 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. Syst. Technol.*, vol. 17, no. 3, pp. 105–122, 2007.

[3] Codes available at http://www.vision.ee.ethz.ch/~timofter/

[8] V. Katkovnik, K. Egiazarian, and J. Astola, *Local Approximation Techniques in Signal and Image Processing*. Bellingham, WA, USA: SPIE, 2006.
[9] V. Katkovnik, K. Egiazarian, and J. Astola, "Adaptive window size image de-noising based on intersection of confidence intervals (ICI) rule," *J. Math. Imag. Vis.*, vol. 16, no. 3, pp. 223–235, 2002.
[10] J. E. Adams, Jr., and J. F. Hamilton, Jr., "Adaptive color plan interpolation in single sensor color electronic camera," U.S. Patent 5 629 734, May 13, 1997.
[11] X. Wu and N. Zhang, "Primary-consistent soft-decision color demosaicking for digital cameras (patent pending)," *IEEE Trans. Image Process.*, vol. 13, no. 9, pp. 1263–1274, Sep. 2004.
[12] I. Pekkucuksen and Y. Altunbasak, "Gradient based threshold free color filter array interpolation," in *Proc. 17th IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2010, pp. 137–140.
[13] I. Pekkucuksen and Y. Altunbasak, "Multiscale gradients-based color filter array interpolation," *IEEE Trans. Image Process.*, vol. 22, no. 1, pp. 157–165, Jan. 2013.
[14] D. Kiku, Y. Monno, M. Tanaka, and M. Okutomi, "Minimized-Laplacian residual interpolation for color image demosaicking," *Proc. SPIE*, vol. 9023, p. 90230L, Mar. 2014.
[15] F. Zhang, X. Wu, X. Yang, W. Zhang, and D. Zhang, "Robust color demosaicking with adaptation to varying spectral correlations," *IEEE Trans. Image Process.*, vol. 18, no. 12, pp. 2706–2717, Dec. 2009.
[16] L. Zhang, X. Wu, A. Buades, and X. Li, "Color demosaicking by local directional interpolation and nonlocal adaptive thresholding," *J. Electron. Imag.*, vol. 20, no. 2, p. 023016, 2011.
[17] B. K. Gunturk, Y. Altunbasak, and R. M. Mersereau, "Color plane interpolation using alternating projections," *IEEE Trans. Image Process.*, vol. 11, no. 9, pp. 997–1013, Sep. 2002.
[18] K. Hirakawa and T. W. Parks, "Adaptive homogeneity-directed demosaicing algorithm," *IEEE Trans. Image Process.*, vol. 14, no. 3, pp. 360–369, Mar. 2005.
[19] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman, "Non-local sparse models for image restoration," in *Proc. IEEE 12th Int. Conf. Comput. Vis.*, Sep./Oct. 2009, pp. 2272–2279.
[20] D. Gao, X. Wu, G. Shi, and L. Zhang, "Color demosaicking with an image formation model and adaptive PCA," *J. Vis. Commun. Image Represent.*, vol. 23, no. 7, pp. 1019–1030, 2012.
[21] R. Timofte, V. De Smet, and L. Van Gool, "A+: Adjusted anchored neighborhood regression for fast super-resolution," in *Proc. IEEE Asian Conf. Comput. Vis. (ACCV)*, Nov. 2014, pp. 111–126.
[22] R. Rothe, R. Timofte, and L. Van Gool, "Efficient regression priors for reducing image compression artifacts," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2015, pp. 1543–1547.
[23] S. Farsiu, M. Elad, and P. Milanfar, "Multiframe demosaicing and super-resolution of color images," *IEEE Trans. Image Process.*, vol. 15, no. 1, pp. 141–159, Jan. 2006.
[24] J. Wu, R. Timofte, and L. Van Gool, "Efficient regression priors for post-processing demosaiced images," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2015, pp. 3495–3499.
[25] K. He, J. Sun, and X. Tang, "Guided image filtering," in *Computer Vision*. Berlin, Germany: Springer, 2010, pp. 1–14.
[26] R. Timofte, V. De Smet, and L. Van Gool, "Anchored neighborhood regression for fast example-based super-resolution," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2013, pp. 1920–1927.
[27] R. Zeyde, M. Elad, and M. Protter, "On single image scale-up using sparse-representations," in *Curves and Surfaces*. Berlin, Germany: Springer, 2012, pp. 711–730.
[28] J. Yang, J. Wright, T. S. Huang, and Y. Ma, "Image super-resolution via sparse representation," *IEEE Trans. Image Process.*, vol. 19, no. 11, pp. 2861–2873, Nov. 2010.
[29] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Trans. Signal Process.*, vol. 54, no. 11, pp. 4311–4322, Nov. 2006.
[30] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.
[31] W. Lu and Y.-P. Tan, "Color filter array demosaicking: New method and performance measures," *IEEE Trans. Image Process.*, vol. 12, no. 10, pp. 1194–1210, Oct. 2003.
[32] R. Timofte, R. Rothe, and L. Van Gool. (2015). "Seven ways to improve example-based single image super resolution." [Online]. Available: http://arxiv.org/abs/1511.02228



Jiqing Wu received the B.S. degree in mechanical engineering from Shanghai Maritime University, China, in 2006, the B.S. degree in mathematics from TU Darmstadt, Germany, in 2012, and the M.Sc. degree in mathematics from ETH Zürich, Switzerland, in 2015, where. he is currently pursuing the Ph.D. degree under the supervision of Prof. L. Van Gool. His research interests mainly concern image demosaicing, image restoration, and early vision.

Radu Timofte received the Ph.D. degree in electrical engineering from the KU Leuven, Belgium, in 2013, the M.Sc. degree from the University of Eastern Finland in 2007, and the Dipl.-Ing. degree from the Technical University of Iasi, Romania, in 2006. He was a Researcher with the University of Joensuu (2006-2007), Czech Technical University (2008), and KU Leuven (2008-2013). Since 2013, he has held a post-doctoral position with the Laboratory of Prof. Luc Van Gool, ETH Zürich, Switzerland. He serves as a Reviewer for major journals, such as PAMI, TIP, IJCV, TNNLS, TKDE, PRL, and T-ITS, and conferences, such as CVPR, ICCV, and ECCV. His current research interests include sparse and collaborative representations, image restoration and enhancement, and multi-view object class recognition. He received best paper awards at the ICPR 2012, at the CVVT Workshop from ECCV 2012, and at the ChaLearn Workshop from ICCV 2015.

Luc Van Gool received the master's and Ph.D. degrees in electrical engineering from KU Leuven, Belgium, in 1982 and 1991, respectively. He is a Full Professor of Computer Vision with both KU Leuven and ETH Zürich. With his two research laboratories, he focuses on object recognition, tracking and gesture analysis, and 3D acquisition and modeling. He was a Program Chair of the ICCV 2005, and a General Chair (GC) of ICCV 2011, and acted as a GC of the ECCV 2014. He is the Editor-in-Chief of *Foundations and Trends in Computer Graphics and Vision*. He is also a Co-Founder of the spin-off companies—Eyetronics, GeoAutomation, Kooaba, Procedural, eSaturnus, Upicto, Fashwell, Merantix, Spectando, and Parquery. He received several best paper awards, including at the ICCV 1998, CVPR 2007, ACCV 2007, ICRA 2009, BMVC 2011, and ICPR 2012.