Deep Image Demosaicking Using a Cascade of Convolutional Residual Denoising Networks

Filippos Kokkinos(✉) and Stamatios Lefkimmiatis

Skolkovo Institute of Science and Technology (Skoltech), Moscow, Russia
{filippos.kokkinos,s.lefkimmiatis}@skoltech.ru

Abstract. Demosaicking and denoising are among the most crucial steps of modern digital camera pipelines and their joint treatment is a highly ill-posed inverse problem where at-least two-thirds of the information are missing and the rest are corrupted by noise. This poses a great challenge in obtaining meaningful reconstructions and a special care for the efficient treatment of the problem is required. While there are several machine learning approaches that have been recently introduced to deal with joint image demosaicking-denoising, in this work we propose a novel deep learning architecture which is inspired by powerful classical image regularization methods and large-scale convex optimization techniques. Consequently, our derived network is more transparent and has a clear interpretation compared to alternative competitive deep learning approaches. Our extensive experiments demonstrate that our network outperforms any previous approaches on both noisy and noise-free data. This improvement in reconstruction quality is attributed to the principled way we design our network architecture, which also requires fewer trainable parameters than the current state-of-the-art deep network solution. Finally, we show that our network has the ability to generalize well even when it is trained on small datasets, while keeping the overall number of trainable parameters low.

Keywords: Deep learning · Denoising · Demosaicking
Proximal method · Residual denoising

1 Introduction

Modern digital cameras perform a certain number of processing steps in order to create high quality images from raw sensor data. The sequence of the required processing steps is known as the imaging pipeline and the first two and most crucial steps involve image denoising and demosaicking. Both of these problems belong to the category of ill-posed problems while their joint treatment is very challenging since two-thirds of the underlying data are missing and the rest are perturbed by noise. It is clear that reconstruction errors during this early stage of the camera pipeline will eventually lead to unsatisfying final results.

V. Ferrari et al. (Eds.): ECCV 2018, LNCS 11218, pp. 317–333, 2018.
https://doi.org/10.1007/978-3-030-01264-9_19

Furthermore, due to the modular nature of the camera processing pipelines, demosaicking and denoising were traditionally dealt in the past in a sequential manner. In detail, demosaicking algorithms reconstruct the image from unreliable spatially-shifted sensor data which introduce non-linear pixel noise, casting denoising an even harder problem. Since, demosaicking is an essential step of the camera pipeline, it has been extensively studied. For a complete survey of recent approaches, we refer to [1]. One of the main drawbacks of several of the currently introduced methods that deal with the demosaicking problem, is that they assume a specific Bayer pattern [1–6]. This is a rather strong assumption and limits their applicability since there are many cameras available in the market that employ different Color filter Array (CFA) patterns. Therefore, demosaicking methods that are able to generalize to different CFA patterns are preferred.

One simple method that works for any CFA pattern is bilinear interpolation on the neighboring values for a given pixel for each channel. The problem with this approach is the produced zippering artifacts which occur along high frequency signal changes, e.g., edges. Therefore, many approaches involve edge-adaptive interpolation schemes which follow the direction of the gradient of strong edges [1]. However, the real challenges of demosaicking extend in the exploitation of both intra and inter-channel dependencies. The most common assumption is that color differences between color channels are constant, so that the end result leads to smooth images. Other approaches make use of the self-similarity and redundancy properties of natural images [2–4,6]. Moreover, in some cases a post-processing step is applied to remove certain type of artifacts [7]. Another successful class consists of methods that act upon the frequency domain. Any Bayer CFA can be represented as the combination of a luminance component at baseband and two modulated components [8]. Upon this interpretation, Dubois [9–11] created a successful set of filter-banks using a least-squares method that was able to generalize to arbitrary sensor patterns.

From the perspective of learning based approaches, the bibliography is short. A common problem with the design of learning based demosaicking algorithms is the lack of ground-truth images. In many approaches such as those in [12,13] the authors used already processed images as references that are simulated mosaicked again, i.e. they apply a mosaick mask on the already demosaicked images, therefore obtaining non-realistic pairs for tuning trainable methods. In a recent work Khasabi et. al. [14] provided a way to produce a dataset with realistic reference images allowing for the design of machine learning demosaicking algorithms. We use the produced Microsoft Demosaicking dataset (MSR) [14] in order to train, evaluate and compare our system. The contained images have to be demosaicked in the linear RGB (linRGB) color space before being transformed via color transformation and gamma correction into standard RGB (sRGB) space. Furthermore, two common CFA patterns are contained into the dataset, namely Bayer and Fuji X Trans which enables the development and evaluation of methods that are able to deal with different CFA patterns.

Apart from the demosaicking problem, another problem that requires special attention is the elimination of noise arising from the sensor and which distorts the acquired raw data. Firstly, the sensor readings are corrupted with *shot* noise [15] which is the result of random variation of the detected photons. Second, electronic inefficiencies during reading and converting electrical charge into a digital count exhibit another type of noise, namely *read* noise. Under certain circumstances both noises can be approximated by noise following a heteroscedastic Gaussian pdf [15]. Prior work from Kalevo and Rantanen [16], analyzed whether denoising should occur before or after the demosaicking step. It was experimentally confirmed that denoising is preferably done before demosaicking. However, the case of joint denoising and demosaicking was not analyzed. In later work, many researchers [17–19] showed that joint denoising and demosaicking yields better results. Motivated by these works, we also pursue a joint approach for denoising and demosaicking of raw sensor data.

In a very recent work Gharbi et. al. [20] exploit the advantages in the field of deep learning to create a Convolutional Neural Network (CNN) that is able to jointly denoise and demosaick images. Apart from the design of the aforementioned network, a lot of effort was put by the authors to create a new large demosaicking dataset, namely the MIT Demosaicking Dataset which consists of 2.6 million patches of images. These patches were mined from a large collection of data following specific visual distortion metrics.

Our main contribution is a novel deep neural network for solving the joint image demosaicking-denoising problem[1]. The network architecture is inspired by classical image regularization approaches and a powerful optimization strategy that has been successfully used in the past for dealing with general inverse imaging problems. We demonstrate through extensive experimentation that our approach leads to higher-quality reconstruction than other competing methods in both linear RGB (linRGB) and standard RGB (sRGB) color spaces. Moreover, we further show that our derived network not only outperforms the current CNN-based state-of-the art network [20], but it achieves this by using less trainable parameters and by being trained only on a small fraction of the training data.

2 Problem Formulation

To solve the joint demosaicking-denoising problem, one of the most frequently used approaches in the literature relies on the following linear observation model

$$\mathbf{y} = \mathbf{M}\mathbf{x} + \mathbf{n}, \tag{1}$$

which relates the observed sensor raw data, $\mathbf{y} \in \mathbb{R}^N$, and the underlying image $\mathbf{x} \in \mathbb{R}^N$ that we aim to restore. Both $\mathbf{x}$ and $\mathbf{y}$ correspond to the vectorized forms of the images assuming that they have been raster scanned using a lexicographical order. Under this notation, $\mathbf{M} \in \mathbb{R}^{N \times N}$ is the degradation matrix that

[1] The code for both training and inference will be made available from the authors' website.

models the spatial response of the imaging device, and in particular the CFA pattern. According to this, $\mathbf{M}$ corresponds to a square diagonal binary matrix where the zero elements in its diagonal indicate the spatial and channel locations in the image where color information is missing. Apart from the missing color values, the image measurements are also perturbed by noise which hereafter, we will assume that is an i.i.d Gaussian noise $\mathbf{n} \sim \mathcal{N}(0, \sigma^2)$. Note, that this is a rather simplified assumption about the noise statistics distorting the measurements. However, this model only serves as our starting point based on which we will design our network architecture. In the sequel, our derived network will be trained and evaluated on images that are distorted by noise which follows statistics that better approximate real noisy conditions.

Recovering $\mathbf{x}$ from the measurements $\mathbf{y}$ belongs to the broad class of linear inverse problems. For the problem under study, the operator $\mathbf{M}$ is clearly singular. This fact combined with the presence of noise perturbing the measurements leads to an ill-posed problem where a unique solution does not exist. One popular way to deal with this, is to adopt a Bayesian approach and seek for the Maximum A Posteriori (MAP) estimator

$$\mathbf{x}^\star = \arg\max_{\mathbf{x}} \log(p(\mathbf{x}|\mathbf{y})) = \arg\max_{\mathbf{x}} \log(p(\mathbf{y}|\mathbf{x})) + \log(p(\mathbf{x})), \tag{2}$$

where $\log(p(\mathbf{y}|\mathbf{x}))$ represents the log-likelihood of the observation $\mathbf{y}$ and $\log(p(\mathbf{x}))$ represents the log-prior of $\mathbf{x}$. Problem (2) can be equivalently re-casted as the minimization problem

$$\mathbf{x}^\star = \arg\min_{\mathbf{x}} \frac{1}{2\sigma^2}\|\mathbf{y} - \mathbf{M}\mathbf{x}\|_2^2 + \phi(\mathbf{x}) \tag{3}$$

where the first term corresponds to the negative log-likelihood (assuming i.i.d Gaussian noise of variance σ^2) and the second term corresponds to the negative log-prior. According to the above, the restoration of the underlying image $\mathbf{x}$, boils down to computing the minimizer of the objective function in Eq. (3), which consists of two terms. This problem formulation has also direct links to variational methods where the first term can be interpreted as the data-fidelity that quantifies the proximity of the solution to the observation and the second term can be seen as the regularizer, whose role is to promotes solutions that satisfy certain favorable image properties.

In general, the minimization of the objective function

$$Q(\mathbf{x}) = \frac{1}{2\sigma^2}\|\mathbf{y} - \mathbf{M}\mathbf{x}\|_2^2 + \phi(\mathbf{x}) \tag{4}$$

is far from a trivial task, especially when the function $\phi(\mathbf{x})$ is not of a quadratic form, which implies that the solution cannot simply be obtained by solving a set of linear equations. From the above, it is clear that there are two important challenges that need to be dealt with before we are in position of deriving a satisfactory solution for our problem. The first one is to come up with an algorithm that can efficiently minimize $Q(\mathbf{x})$, while the second one is to select an

appropriate form for $\phi(\mathbf{x})$, which will constrain the set of admissible solutions by promoting only those that exhibit the desired properties.

In Sect. 3, we will focus on the first challenge, while in Sect. 4 we will discuss how it is possible to avoid making any explicit decisions for the regularizer (or equivalently the negative log-prior) by following a machine learning approach. Such an approach will allow us to infer the form of $\phi(\mathbf{x})$, in an indirect way, from training data.

3 Majorization-Minimization Framework

One of the main difficulties in the minimization of the objective function in Eq. (4) is the coupling that exists between the singular degradation operator, $\mathbf{M}$, and the latent image $\mathbf{x}$. To circumvent this difficulty there are several optimization strategies available that we could rely on, with potential candidates being splitting variables techniques such as the Alternating Direction Method of Multipliers [21] and the Split Bregman approach [22]. However, one difficulty that arises by using such methods is that they involve additional parameters that need to be tuned so that a satisfactory convergence speed to the solution is achieved. Unfortunately, there is not a simple and straightforward way to choose these parameters. For this reason, in this work we will instead pursue a majorization-minimization (MM) approach [23,24], which does not pose such a requirement. Under this framework, as we will describe in detail, instead of obtaining the solution by minimizing (4), we compute it iteratively via the successive minimization of surrogate functions. The surrogate functions provide an upper bound of the initial objective function [23] and they are simpler to deal with than the original objective function.

Specifically, in the majorization-minimization (MM) framework, an iterative algorithm for solving the minimization problem

$$\mathbf{x}^* = \arg\min_f Q(\mathbf{x}) \tag{5}$$

takes the form

$$\mathbf{x}^{(t+1)} = \arg\min_x \tilde{Q}(\mathbf{x}; \mathbf{x}^{(t)}), \tag{6}$$

where $\tilde{Q}(\mathbf{x}; \mathbf{x}^{(t)})$ is the majorizer of the function $Q(\mathbf{x})$ at a fixed point $\mathbf{x}^{(t)}$, satisfying the two conditions

$$\tilde{Q}(\mathbf{x}; \mathbf{x}^{(t)}) > Q(\mathbf{x}), \forall \mathbf{x} \neq \mathbf{x}^{(t)} \quad \text{and} \quad \tilde{Q}(\mathbf{x}^{(t)}; \mathbf{x}^{(t)}) = Q(\mathbf{x}^{(t)}). \tag{7}$$

Here, the underlying idea is that instead of minimizing the actual objective function $Q(\mathbf{x})$, we fist upper-bound it by a suitable majorizer $\tilde{Q}(\mathbf{x}; \mathbf{x}^{(t)})$, and then minimize this majorizing function to produce the next iterate $\mathbf{x}^{(t+1)}$. Given the properties of the majorizer, iteratively minimizing $\tilde{Q}(\cdot; \mathbf{x}^{(t)})$ also decreases the objective function $Q(\cdot)$. In fact, it is not even required that the surrogate function in each iteration is minimized, but it is sufficient to only find a $\mathbf{x}^{(t+1)}$ that decreases it.

To derive a majorizer for $Q(\mathbf{x})$ we opt for a majorizer of the data-fidelity term (negative log-likelihood). In particular, we consider the following majorizer

$$\tilde{d}(\mathbf{x}, \mathbf{x}_0) = \frac{1}{2\sigma^2}\|\mathbf{y} - \mathbf{M}\mathbf{x}\|_2^2 + d(\mathbf{x}, \mathbf{x}_0), \tag{8}$$

where $d(\mathbf{x}, \mathbf{x}_0) = \frac{1}{2\sigma^2}(\mathbf{x} - \mathbf{x}_0)^T[\alpha\mathbf{I} - \mathbf{M}^T\mathbf{M}](\mathbf{x} - \mathbf{x}_0)$ is a function that measures the distance between $\mathbf{x}$ and $\mathbf{x}_0$. Since $\mathbf{M}$ is a binary diagonal matrix, it is an idempotent matrix, that is $\mathbf{M}^T\mathbf{M} = \mathbf{M}$, and thus $d(\mathbf{x}, \mathbf{x}_0) = \frac{1}{2\sigma^2}(\mathbf{x} - \mathbf{x}_0)^T[\alpha\mathbf{I} - \mathbf{M}](\mathbf{x} - \mathbf{x}_0)$. According to the conditions in (7), in order $\tilde{d}(\mathbf{x}, \mathbf{x}_0)$ to be a valid majorizer, we need to ensure that $d(\mathbf{x}, \mathbf{x}_0) \geq 0, \forall \mathbf{x}$ with equality iff $\mathbf{x} = \mathbf{x}_0$. This suggests that $a\mathbf{I} - \mathbf{M}$ must be a positive definite matrix, which only holds when $\alpha > \|\mathbf{M}\|_2 = 1$, i.e. α is bigger than the maximum eigenvalue of $\mathbf{M}$. Based on the above, the upper-bounded version of (4) is finally written as

$$\tilde{Q}(\mathbf{x}, \mathbf{x}_0) = \frac{1}{2(\sigma/\sqrt{a})^2}\|\mathbf{x} - \mathbf{z}\|_2^2 + \phi(\mathbf{x}) + c, \tag{9}$$

where c is a constant and $\mathbf{z} = \mathbf{y} + (\mathbf{I} - \mathbf{M})\mathbf{x}_0$.

Notice that following this approach, we have managed to completely decouple the degradation operator $\mathbf{M}$ from $\mathbf{x}$ and we now need to deal with a simpler problem. In fact, the resulting surrogate function in Eq. (9) can be interpreted as the objective function of a denoising problem, with $\mathbf{z}$ being the noisy measurements that are corrupted by noise whose variance is equal to σ^2/a. This is a key observation that we will heavily rely on in order to design our deep network architecture. In particular, it is now possible instead of selecting the form of $\phi(\mathbf{x})$ and minimizing the surrogate function, to employ a denoising neural network that will compute the solution of the current iteration. Our idea is similar in nature to other recent image restoration approaches that have employed denoising networks as part of alternative iterative optimization strategies, such as RED [25] and P^3 [26]. This direction for solving the joint denoising-demosaicking problem is very appealing since by using training data we can implicitly learn the function $\phi(\mathbf{x})$ and also minimize the corresponding surrogate function using a feed-forward network. This way we can completely avoid making any explicit decision for the regularizer or relying on an iterative optimization strategy to minimize the function in Eq. (9).

4 Residual Denoising Network (ResDNet)

Based on the discussion above, the most important part of our approach is the design of a denoising network that will play the role of the solver for the surrogate function in Eq. (9). The architecture of the proposed network is depicted in Fig. 1. This is a residual network similar to DnCNN [27], where the output of the network is subtracted from its input. Therefore, the network itself acts as a noise estimator and its task is to estimate the noise realization that distorts the input. Such network architectures have been shown to lead to better restoration

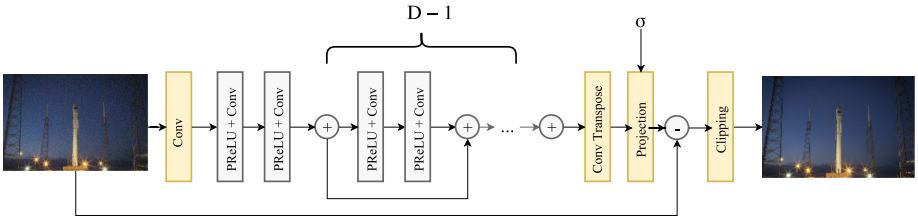

Fig. 1. The architecture of the proposed ResDNet denoising network, which serves as the back-bone of our overall system.

results than alternative approaches [27,28]. One distinctive difference between our network and DnCNN, which also makes our network suitable to be used as a part of the MM-approach, is that it accepts two inputs, namely the distorted input and the variance of the noise. This way, as we will demonstrate in the sequel, we are able to learn a single set of parameters for our network and to employ the same network to inputs that are distorted by a wide range of noise levels. While the blind version of DnCNN can also work for different noise levels, as opposed to our network it features an internal mechanism to estimate the noise variance. However, when the noise statistics deviate significantly from the training conditions such a mechanism can fail and thus DnCNN can lead to poor denoising results [28]. In fact, due to this reason in [29], where more general restoration problems than denoising are studied, the authors of DnCNN use a non-blind variant of their network as a part of their proposed restoration approach. Nevertheless, the drawback of this approach is that it requires the training of a deep network for each noise level. This can be rather impractical, especially in cases where one would like to employ such networks on devices with limited storage capacities. In our case, inspired by the recent work in [28] we circumvent this limitation by explicitly providing as input to our network the noise variance, which is then used to assist the network so as to provide an accurate estimate of the noise distorting the input. Note that there are several techniques available in the literature that can provide an estimate of the noise variance, such as those described in [30,31], and thus this requirement does not pose any significant challenges in our approach.

A ResDNet with depth D, consists of five fundamental blocks. The first block is a convolutional layer with 64 filters whose kernel size is 5×5. The second one is a non-linear block that consists of a parametrized rectified linear unit activation function (PReLU), followed by a convolution with 64 filters of 3×3 kernels. The PReLU function is defined as $\mathrm{PReLU}(\mathbf{x}) = \max(0, \mathbf{x}) + \boldsymbol{\kappa} * \min(0, \mathbf{x})$ where $\boldsymbol{\kappa}$ is a vector whose size is equal to the number of input channels. In our network we use $D * 2$ distinct non-linear blocks which we connect via a shortcut connection every second block in a similar manner to [32] as shown in Fig. 1. Next, the output of the non-linear stage is processed by a transposed convolution layer which reduces the number of channels from 64 to 3 and has a kernel size of 5×5. Then, it follows a projection layer [28] which accepts as an additional input the

noise variance and whose role is to normalize the noise realization estimate so that it will have the correct variance, before this is subtracted from the input of the network. Finally the result is clipped so that the intensities of the output lie in the range $[0, 255]$. This last layer enforces our prior knowledge about the expected range of valid pixel intensities.

Regarding implementation details, before each convolution layer the input is padded to make sure that each feature map has the same spatial size as the input image. However, unlike the common approach followed in most of the deep learning systems for computer vision applications, we use reflexive padding than zero padding. Another important difference to other networks used for image restoration tasks [27,29] is that we don't use batch normalization after convolutions. Instead, we use the parametric convolution representation that has been proposed in [28] and which is motivated by image regularization related arguments. In particular, if $\mathbf{v} \in \mathbb{R}^L$ represents the weights of a filter in a convolutional layer, these are parametrized as

$$\mathbf{v} = \frac{s\,(\mathbf{u} - \bar{\mathbf{u}})}{\|\mathbf{u} - \bar{\mathbf{u}}\|_2}, \tag{10}$$

where s is a scalar trainable parameter, $\mathbf{u} \in \mathbb{R}^L$ and $\bar{\mathbf{u}}$ denotes the mean value of $\mathbf{u}$. In other words, we are learning zero-mean valued filters whose ℓ_2-norm is equal to s.

Furthermore, the projection layer, which is used just before the subtraction operation with the network input, corresponds to the following ℓ_2 orthogonal projection

$$\mathcal{P}_{\mathcal{C}}\,(\mathbf{y}) = \varepsilon \frac{\mathbf{y}}{max(\|\mathbf{y}\|_2, \varepsilon)}, \tag{11}$$

where $\varepsilon = e^{\gamma}\theta$, $\theta = \sigma\sqrt{N-1}$, N is the total number of pixels in the image (including the color channels), σ is the standard deviation of the noise distorting the input, and γ is a scalar trainable parameter. As we mentioned earlier, the goal of this layer is to normalize the noise realization estimate so that it has the desired variance before it is subtracted from the network input.

5 Demosaicking Network Architecture

The overall architecture of our approach is based upon the MM framework, presented in Sect. 3, and the proposed denoising network. As discussed, the MM is an iterative algorithm Eq. (6) where the minimization of the majorizer in Eq. (9) can be interpreted as a denoising problem. One way to design the demosaicking network would be to unroll the MM algorithm as K discrete steps and then for each step use a different denoising network to retrieve the solution of Eq. (9). However, this approach can have two distinct drawbacks which will hinder its performance. The first one, is that the usage of a different denoising neural network for each step like in [29], demands a high overall number of parameters, which is equal to K times the parameters of the employed denoiser, making

Algorithm 1. The proposed demosaicking network described as an iterative process. The ResDnet parameters remain the same in every iteration.

Input: $\mathbf{M}$: CFA, $\mathbf{y}$: input, K : iterations, $\boldsymbol{w} \in \mathbb{R}^K$: extrapolation weights, $\boldsymbol{\sigma} \in \mathbb{R}^K$: noise vector

$\mathbf{x}^0 = \mathbf{0}$, $\mathbf{x}^1 = \mathbf{y}$;

for $i \leftarrow 1$ **to** K **do**

  $\mathbf{u} = \mathbf{x}^{(i)} + \boldsymbol{w}_i(\mathbf{x}^{(i)} - \mathbf{x}^{(i-1)})$;

  $\mathbf{x}^{(i+1)} = \text{ResDNet}((\mathbf{I} - \mathbf{M})\mathbf{u} + \mathbf{y}, \boldsymbol{\sigma}_i)$;

end

the demosaicking network impractical for any real applications. To override this drawback, we opt to use our ResDNet denoiser, which can be applied to a wide range of noise levels, for all K steps of our demosaick network, using the same set of parameters. By sharing the parameters of our denoiser across all the K steps, the overall demosaicking approach maintains a low number of required parameters.

The second drawback of the MM framework as described in Sect. 3 is the slow convergence [33] that it can exhibit. Beck and Teboulle [33] introduced an accelerated version of this MM approach which combines the solutions of two consecutive steps with a certain extrapolation weight that is different for every step. In this work, we adopt a similar strategy which we describe in Algorithm 1. Furthermore, in our approach we go one step further and instead of using the values originally suggested in [33] for the weights $\boldsymbol{w} \in \mathbb{R}^K$, we treat them as trainable parameters and learn them directly from the data. These weights are initialized with $w_i = \frac{i-1}{i+2}, \forall 1 \leq i \leq K$.

The convergence of our framework can be further sped up by employing a continuation strategy [34] where the main idea is to solve the problem in Eq. (9) with a large value of σ and then gradually decrease it until the target value is reached. Our approach is able to make use of the continuation strategy due to the design of our ResDNet denoiser, which accepts as an additional argument the noise variance. In detail, we initialize the trainable vector $\boldsymbol{\sigma} \in \mathbb{R}^K$ with values spaced evenly on a log scale from σ_{max} to σ_{min} and later on the vector $\boldsymbol{\sigma}$ is further finetuned on the training dataset by back-propagation training.

In summary, our overall demosaicking network is described in Algorithm 1 where the set of trainable parameters θ consists of the parameters of the ResDNet denoiser, the extrapolation weights $\boldsymbol{w}$ and the noise level $\boldsymbol{\sigma}$. All of the aforementioned parameters are initialized as described in the current section and Sect. 4 and are trained on specific demosaick datasets. In order to speed up the learning process, the employed ResDNet denoiser is pre-trained for a denoising task where multiple noise levels are considered.

Finally, while our demosaick network shares a similar philosophy with methods such as RED [25], P^3 [26] and IRCNN [29], it exhibits some important and distinct differences. In particular, the aforementioned strategies make use of certain optimization schemes to decompose their original problem into subproblems

that are solvable by a denoiser. For example, the authors of P^3 [26] decompose the original problem Eq. (1) via ADMM [21] optimization algorithm and solve instead a linear system of equations and a denoising problem, where the authors of RED [25] go one step further and make use of the Lagrangian on par with a denoiser. Both approaches are similar to ours, however their formulation involves a tunable variable λ that weights the participation of the regularizer on the overall optimization procedure. Thus, in order to obtain an accurate reconstruction in reasonable time, the user must manually tune the variable λ which is not a trivial task. On the other hand, our method does not involve any tunable variables by the user. Furthermore, the approaches P^3, RED and IRCNN are based upon static denoisers like Non Local Means [35], BM3D [36] and DCNN [27], meanwhile we opt to use a universal denoiser, like ResDnet, that can be further trained on any available training data. Finally, our approach goes one step further and we use a trainable version of an iterative optimization strategy for the task of the joint denoising-demosaicking in the form of a feed-forward neural network (Fig. 2).

6 Network Training

6.1 Image Denoising

The denoising network ResDnet that we use as part of our overall network is pre-trained on the Berkeley segmentation dataset (BSDS) [37], which consists of 500 color images. These images were split in two sets, 400 were used to form a train set and the rest 100 formed a validation set. All the images were randomly cropped into patches of size 180×180 pixels. The patches were perturbed with noise $\sigma \in [0, 15]$ and the network was optimized to minimize the Mean Square Error. We set the network depth $D = 5$, all weights are initialized as in He et al. [38] and the optimization is carried out using ADAM [39] which is a stochastic gradient descent algorithm which adapts the learning rate per parameter. The training procedure starts with an initial learning rate equal to 10^{-2}.

6.2 Joint Denoising and Demosaicking

Using the pre-trained denoiser Sect. 6.1, our novel framework is further trained in an end-to-end fashion to minimize the averaged L_1 loss over a minibatch of size d,

$$L(\theta) = \frac{1}{N} \sum_{i=1}^{d} \|\mathbf{y}_i - f(\mathbf{x}_i)\|_1, \tag{12}$$

where $\mathbf{y}_i \in \mathbb{R}^N$ and $\mathbf{x}_i \in \mathbb{R}^N$ are the rasterized groundtruth and input images, while $f(\cdot)$ is the output of our network. The minimization of the loss function is carried via the Backpropagation Through Time (BPTT) [40] algorithm since the weights of the network remain the same for all iterations.

During all our experiments, we used a small batch size of $d = 4$ images, the total steps of the network were fixed to $\mathrm{K} = 10$ and we set for the initialization of

vector $\boldsymbol{\sigma}$ the values $\sigma_{max} = 15$ and $\sigma_{min} = 1$. The small batch size is mandatory during training because all intermediate results have to be stored for the BPTT, thus the memory consumption increases linearly to iteration steps and batch size. Furthermore, the optimization is carried again via Adam optimizer and the training starts from a learning rate of 10^{-2} which we decrease by a factor of 10 every 30 epochs. Finally, for all trainable parameters we apply ℓ_2 weight decay of 10^{-8}. The full training procedure takes 3 hours for MSR Demosaicking Dataset and 5 days for a small subset of the MIT Demosaicking Dataset on a modern NVIDIA GTX 1080Ti GPU.

Table 1. Comparison of our system to state-of-the-art techniques on the demosaick-only scenario in terms of PSNR performance. The Kodak dataset is resized to 512×768 following the methodology of evaluation in [1]. *Our system for the MIT dataset was trained on a small subset of 40,000 out of 2.6 million images.

	Kodak	McM	Vdp	Moire
Non-ML Methods:				
Bilinear	32.9	32.5	25.2	27.6
Adobe Camera Raw 9	33.9	32.2	27.8	29.8
Buades [4]	37.3	35.5	29.7	31.7
Zhang (NLM) [2]	37.9	36.3	30.1	31.9
Getreuer [41]	38.1	36.1	30.8	32.5
Heide [5]	40.0	38.6	27.1	34.9
Trained on MSR Dataset:				
Klatzer [19]	35.3	30.8	28.0	30.3
Ours	39.2	34.1	29.2	29.7
Trained on MIT Dataset:				
Gharbi [20]	41.2	39.5	34.3	**37.0**
Ours*	**41.5**	**39.7**	**34.5**	**37.0**

7 Experiments

Initially, we compare our system to other alternative techniques on the demosaick-only scenario. Our network is trained on the MSR Demosaick dataset [14] and it is evaluated on the McMaster [2], Kodak, Moire and VDP dataset [20], where all the results are reported in Table 1. The MSR Demosaick dataset consists of 200 train images which contain both the linearized 16-bit mosaicked input images and the corresponding linRGB groundtruths that we also augment with horizontal and vertical flipping. For all experiments, in order to quantify the quality of the reconstructions we report the Peak signal-to-noise-ratio (PSNR) metric.

Apart from the MSR dataset, we also train our system on a small subset of 40,000 images from MIT dataset due to the small batch size constraint. Clearly our system is capable of achieving equal and in many cases better performance than the current the state-of-the art network [20] which was trained on the full MIT dataset, i.e. 2.6 million images. We believe that training our network on the complete MIT dataset, it will produce even better results for the noise-free scenario. Furthermore, the aforementioned dataset contains only noise-free samples, therefore we don't report any results in Table 2 and we mark the respective results by using N/A instead. We also note that in [20], the authors in order to use the MIT dataset to train their network for the joint demosaicking denoising scenario, pertubed the data by i.i.d Gaussian noise. As a result, their system's performance under the presence of more realistic noise was significantly reduced, which can be clearly seen from Table 2. The main reason for this is that their noise assumption does not account for the *shot* noise of the camera but only for the *read* noise.

Table 2. PSNR performance by different methods in both linear and sRGB spaces. The results of methods that cannot perform denoising are not included for the noisy scenario. Our system for the MIT dataset case was trained on a small subset of 40,000 out of 2.6 million images. The color space in the parentheses indicates the particular color space of the employed training dataset.

	Noise-free		Noisy	
	linRGB	sRGB	linRGB	sRGB
Non-ML Methods:				
Bilinear	30.9	24.9	-	-
Zhang(NLM) [2]	38.4	32.1	-	-
Getreuer [41]	39.4	32.9	-	-
Heide [5]	40.0	33.8	-	-
Trained on MSR Dataset:				
Khasabi [14]	39.4	32.6	37.8	31.5
Klatzer [19]	40.9	34.6	38.8	32.6
Bigdeli [42]	-	-	38.7	-
Ours	**41.0**	**34.6**	**39.2**	**33.3**
Trained on MIT Dataset:				
Gharbi (sRGB)[20]	41.6	35.3	38.4	32.5
Gharbi (linRGB)	**42.7**	**35.9**	38.6	32.6
Ours* (linRGB)	42.6	**35.9**	N/A	N/A

Similarly with the noise free case, we train our system on 200 training images from the MSR dataset which are contaminated with simulated sensor noise [15]. The model was optimized in the linRGB space and the performance was evaluated on both linRGB and sRGB space, as proposed in [14]. It is clear that in

the noise free scenario, training on million of images corresponds to improved performance, however this doesn't seem to be the case on the noisy scenario as presented in Table 2. Our approach, even though it is based on deep learning techniques, is capable of generalizing better than the state-of-the-art system while being trained on a small dataset of 200 images (Fig. 3). In detail, the proposed system has a total 380,356 trainable parameters which is considerably smaller than the current state-of-the art [20] with 559,776 trainable parameters.

Our demosaicking network is also capable of handling non-Bayer patterns equally well, as shown in Table 3. In particular, we considered demosaicking using the Fuji X-Trans CFA pattern, which is a 6×6 grid with the green being the dominant sampled color. We trained from scratch our network on the same train-set of MSR Demosaick Dataset but now we applied the Fuji X-Trans mosaick. In

Table 3. Evaluation on noise-free linear data with the non-Bayer mosaick pattern Fuji XTrans.

	Noise-free	
	linear	sRGB
Trained on MSR Dataset:		
Khashabi [14]	36.9	30.6
Klatzer [19]	39.6	33.1
Ours	**39.9**	**33.7**
Trained on MIT Dataset:		
Gharbi [20]	39.7	33.2

Fig. 2. Progression along the steps of our demosaick network. The first image which corresponds to Step 1 represents a rough approximation of the end result while the second (Step 3) and third image (Step 10) are more refined. This plot depicts the continuation scheme of our approach.

comparison to other systems, we manage to surpass state of the art performance on both linRGB and sRGB space even when comparing with systems trained on million of images.

On a modern GPU (Nvidia GTX 1080Ti), the whole demosaicking network requires 0.05 sec for a color image of size 220×132 and it scales linearly to images of different sizes. Since our model solely consists of matrix operations, it could also be easily transfered to application specific integrated circuit (ASIC) in order to achieve a substantial execution time speedup and be integrated to cameras.

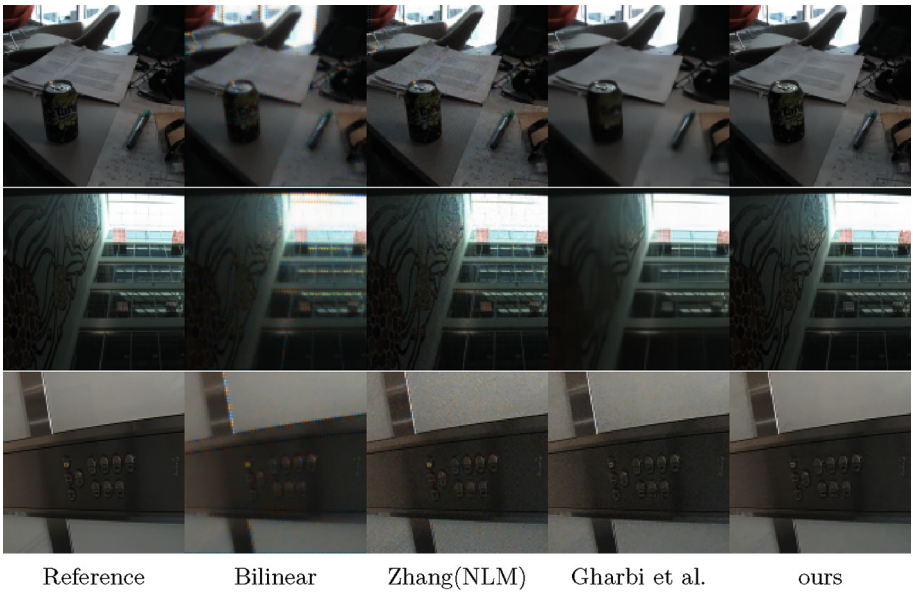

Fig. 3. Comparison of our network with other competing techniques on images from the noisy MSR Dataset. From these results is clear that our method is capable of removing the noise while keeping fine details.On the contrary, the rest of the methods either fail to denoise or they oversmooth the images.

8 Conclusion

In this work, we presented a novel deep learning system that produces high-quality images for the joint denoising and demosaicking problem. Our demosaick network yields superior results both quantitative and qualitative compared to the current state-of-the-art network. Meanwhile, our approach is able to generalize well even when trained on small datasets, while the number of parameters is kept low in comparison to other competing solutions. As an interesting future research direction, we plan to explore the applicability of our method on

other image restoration problems like image deblurring, inpainting and super-resolution where the degradation operator is unknown or varies from image to image.

References

1. Li, X., Gunturk, B., Zhang, L.: Image demosaicing: a systematic survey (2008)
2. Zhang, L., Wu, X., Buades, A., Li, X.: Color demosaicking by local directional interpolation and nonlocal adaptive thresholding. J. Electron. Imaging **20**(2), 023016 (2011)
3. Duran, J., Buades, A.: Self-similarity and spectral correlation adaptive algorithm for color demosaicking. IEEE Trans. Image Process. **23**(9), 4031–4040 (2014)
4. Buades, A., Coll, B., Morel, J.M., Sbert, C.: Self-similarity driven color demosaicking. IEEE Trans. Image Process. **18**(6), 1192–1202 (2009)
5. Heide, F., et al.: Flexisp: a flexible camera image processing framework. ACM Trans. Graph. (TOG) **33**(6), 231 (2014)
6. Chang, K., Ding, P.L.K., Li, B.: Color image demosaicking using inter-channel correlation and nonlocal self-similarity. Signal Process. Image Commun. **39**, 264–279 (2015)
7. Hirakawa, K., Parks, T.W.: Adaptive homogeneity-directed demosaicing algorithm. IEEE Trans. Image Process. **14**(3), 360–369 (2005)
8. Alleysson, D., Susstrunk, S., Herault, J.: Linear demosaicing inspired by the human visual system. IEEE Trans. Image Process. **14**(4), 439–449 (2005)
9. Dubois, E.: Frequency-domain methods for demosaicking of bayer-sampled color images. IEEE Signal Process. Lett. **12**(12), 847–850 (2005)
10. Dubois, E.: Filter design for adaptive frequency-domain bayer demosaicking. In: 2006 International Conference on Image Processing, pp. 2705–2708, October 2006
11. Dubois, E.: Color filter array sampling of color images: Frequency-domain analysis and associated demosaicking algorithms, pp. 183–212, January 2009
12. Sun, J., Tappen, M.F.: Separable markov random field model and its applications in low level vision. IEEE Trans. Image Process. **22**(1), 402–407 (2013)
13. He, F.L., Wang, Y.C.F., Hua, K.L.: Self-learning approach to color demosaicking via support vector regression. In: 19th IEEE International Conference on Image Processing (ICIP), pp. 2765–2768. IEEE (2012)
14. Khashabi, D., Nowozin, S., Jancsary, J., Fitzgibbon, A.W.: Joint demosaicing and denoising via learned nonparametric random fields. IEEE Trans. Image Process. **23**(12), 4968–4981 (2014)
15. Foi, A., Trimeche, M., Katkovnik, V., Egiazarian, K.: Practical poissonian-gaussian noise modeling and fitting for single-image raw-data. IEEE Trans. Image Process. **17**(10), 1737–1754 (2008)
16. Ossi Kalevo, H.R.: Noise reduction techniques for bayer-matrix images (2002)
17. Menon, D., Calvagno, G.: Joint demosaicking and denoisingwith space-varying filters. In: 2009 16th IEEE International Conference on Image Processing (ICIP), pp. 477–480, November 2009
18. Zhang, L., Lukac, R., Wu, X., Zhang, D.: PCA-based spatially adaptive denoising of CFA images for single-sensor digital cameras. IEEE Trans. Image Process. **18**(4), 797–812 (2009)
19. Klatzer, T., Hammernik, K., Knobelreiter, P., Pock, T.: Learning joint demosaicing and denoising based on sequential energy minimization. In: 2016 IEEE International Conference on Computational Photography (ICCP), pp. 1–11, May 2016

20. Gharbi, M., Chaurasia, G., Paris, S., Durand, F.: Deep joint demosaicking and denoising. ACM Trans. Graph. **35**(6), 191:1–191:12 (2016)
21. Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J., et al.: Distributed optimization and statistical learning via the alternating direction method of multipliers. Found. Trends® Mach. Learn. **3**(1), 1–122 (2011)
22. Goldstein, T., Osher, S.: The split bregman method for l1-regularized problems. SIAM J. Imaging Sci. **2**(2), 323–343 (2009)
23. Hunter, D.R., Lange, K.: A tutorial on MM algorithms. Am. Stat. **58**(1), 30–37 (2004)
24. Figueiredo, M.A., Bioucas-Dias, J.M., Nowak, R.D.: Majorization-minimization algorithms for wavelet-based image restoration. IEEE Trans. Image Process. **16**(12), 2980–2991 (2007)
25. Romano, Y., Elad, M., Milanfar, P.: The little engine that could: Regularization by denoising (red). SIAM J. Imaging Sci. **10**(4), 1804–1844 (2017)
26. Venkatakrishnan, S.V., Bouman, C.A., Wohlberg, B.: Plug-and-play priors for model based reconstruction. In: 2013 IEEE Global Conference on Signal and Information Processing, pp. 945–948, December 2013
27. Zhang, K., Zuo, W., Chen, Y., Meng, D., Zhang, L.: Beyond a gaussian denoiser: residual learning of deep cnn for image denoising. IEEE Trans. Image Process. **26**(7), 3142–3155 (2017)
28. Lefkimmiatis, S.: Universal denoising networks: a novel CNN architecture for image denoising. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3204–3213 (2018)
29. Zhang, K., Zuo, W., Gu, S., Zhang, L.: Learning deep CNN denoiser prior for image restoration. arXiv preprint (2017)
30. Foi, A.: Clipped noisy images: Heteroskedastic modeling and practical denoising. Signal Process. **89**(12), 2609–2629 (2009)
31. Liu, X., Tanaka, M., Okutomi, M.: Single-image noise level estimation for blind denoising. IEEE Trans. Image Process. **22**(12), 5226–5237 (2013)
32. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
33. Beck, A., Teboulle, M.: A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM J. Imaging Sci. **2**(1), 183–202 (2009)
34. Lin, Q., Xiao, L.: An adaptive accelerated proximal gradient method and its homotopy continuation for sparse optimization. Comput. Optim. Appl. **60**(3), 633–674 (2015)
35. Buades, A., Coll, B., Morel, J.M.: A non-local algorithm for image denoising. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005, vol. 2, pp. 60–65. IEEE (2005)
36. Dabov, K., Foi, A., Katkovnik, V., Egiazarian, K.: Image denoising by sparse 3-d transform-domain collaborative filtering. IEEE Trans. Image Process. **16**(8), 2080–2095 (2007)
37. Martin, D., Fowlkes, C., Tal, D., Malik, J.: A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In: Proceedings Eighth IEEE International Conference on Computer Vision, ICCV 2001, vol. 2, pp. 416–423 (2001)
38. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: surpassing human-level performance on imagenet classification. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1026–1034 (2015)

39. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
40. Robinson, A.J., Fallside, F.: The utility driven dynamic error propagation network. Technical report CUED/F-INFENG/TR.1, Engineering Department, Cambridge University, Cambridge, UK (1987)
41. Getreuer, P.: Color demosaicing with contour stencils. In: 2011 17th International Conference on Digital Signal Processing (DSP), pp. 1–6, July 2011
42. Bigdeli, S.A., Zwicker, M., Favaro, P., Jin, M.: Deep mean-shift priors for image restoration. In: Advances in Neural Information Processing Systems, pp. 763–772 (2017)