Joint Demosaicing and Denoising via Learned Non-parametric Random Fields

Daniel Khashabi, *Student Member, IEEE,* Sebastian Nowozin, Jeremy Jancsary, *Member, IEEE,* and Andrew W. Fitzgibbon, *Senior Member, IEEE*

***Abstract*—We introduce a machine learning approach to demosaicing, the reconstruction of color images from incomplete color filter array samples. There are two challenges to overcome by a demosaicing method: first, it needs to model and respect the statistics of natural images in order to reconstruct natural looking images; second, it needs to be able to perform well in the presence of noise. To facilitate an objective assessment of current methods we introduce a public ground truth data set of natural images suitable for research in image demosaicing and denoising. We then use this large data set to develop a machine learning approach to demosaicing. Our proposed method addresses both demosaicing challenges by learning a statistical model of images and noise from hundreds of natural images. The resulting model performs simultaneous demosaicing and denoising. We show that the machine learning approach has a number of benefits: 1. the model is trained to directly optimize a user-specified performance measure such as peak signal-to-noise ratio (PSNR) or structural similarity (SSIM), 2. we can handle novel color filter array layouts by retraining the model on such layouts, 3. it outperforms the previous state-of-the-art, in some setups by 0.7dB PSNR, faithfully reconstructing edges, textures, and smooth areas. Our results demonstrate that in demosaicing and related imaging applications, discriminatively trained machine learning models have the potential for peak performance at comparatively low engineering effort.**

***Index Terms*—Demosaicing, denoising, regression tree fields.**

I. Introduction

DIGITAL cameras capture images of the natural world by projecting light through an optical system onto a focal plane. At this focal plane, an array of photosensitive elements converts photons into electrical charge that is then read and digitized. To capture color images, most cameras place a *color filter array* (CFA) pattern in front of the sensor elements. The color filters absorb light with varying sensitivity, depending on the wavelength of the light. As a result, for common CFA choices, the light reaching the photosensitive sensor element is predominantly either in the red, green, or blue spectrum. This monochrome array of recorded responses, once digitized, is known as the *RAW image* and can be saved in many high quality cameras. The color filter array setup records different colors, but it does so at different spatial locations. In order to reconstruct a color image, we apply *image demosaicing*: for each sensor element, we reconstruct the missing color filter array channels. For example, if we recorded only the red wavelengths at an element, we would use the adjacent green and blue recordings to reconstruct the two missing measurements at that element.

The image demosaicing problem is important because almost all digital color cameras in use today use an image demosaicing algorithm to produce color images. There are natural questions that can be asked, such as: 1. How should the color filter array be designed? 2. What are the properties of a good demosaicing algorithm? 3. What are properties of natural images we could use for image demosaicing? For the color filter array, the *Bayer pattern* remains the most popular pattern [1], and most of the demosaicing methods are designed for this pattern, which is shown in Fig. 2a. However, there has been an increasing interest in using other CFA patterns, which motivates universal demosaicing models or pattern-independent demosaicing methods. A good demosaicing algorithm uses the properties of natural images to produce natural looking images. For example, a large area of any natural image is typically smooth and of the same color, perhaps with a slight gradient in brightness. Demosaicing in these parts of the images is straightforward, and even simple interpolation methods give satisfactory results. However, demosaicing is very challenging at edges and corners, and characteristic artifacts such as "zippering" and color tinting become a problem when interpolating nearby values. A good demosaicing algorithm makes use of the distribution of edges and corners in natural images to avoid these artifacts, for example by not interpolating across an edge.

We propose to use machine learning for demosaicing. By training a machine learning method, we can learn to recognize parts of the input signal, such as edges and corners, and produce the demosaiced output accordingly to avoid artifacts. There are three questions at this point: 1. How would we obtain realistic training data without having to use a demosaicing method? 2. What type of machine learning model is suitable for demosaicing? 3. How is the quality of the method assessed during training and testing?

To obtain the training data necessary, we propose a simultaneous downsampling and demosaicing method. Our method produces a low-resolution image as if taken with a sensor recording all color responses. We can then produce the corresponding mosaiced input image by simply removing some of the measurements. For machine learning, we draw on the recently proposed *Regression Tree Field* model [2, 3], which is

Daniel Khashabi is with the Cognitive Computation Group (CCG), University of Illinois, Urbana-Champaign, USA.

Jeremy Jancsary is with Nuance Communications, Vienna, Austria. This work was done while at Microsoft Research, Cambridge, United Kingdom.

Sebastian Nowozin and Andrew Fitzgibbon are with Microsoft Research, Cambridge, United Kingdom.

Manuscript received XXX, 2014; revised XXX, 2014.

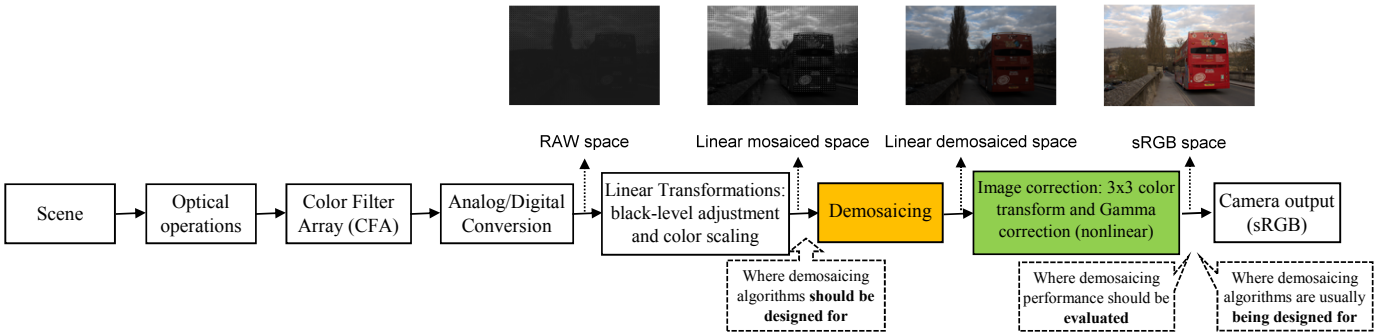


Fig. 1: A simplified camera pipeline. Many academic demosaicing algorithms work on fully developed sRGB images, masked by a CFA pattern. Instead, a practical method must use raw linear-space images as its input (orange block), since the 3x3 color transform to follow (green block) requires all missing measurements to have been filled in. Nonetheless, one should aim at *optimizing* the quality of the output in sRGB space, where images are fully developed and ready to be viewed by a human.

a non-parametric random field model with efficient and exact inference. The model is specific to each CFA pattern, i.e., a single model is trained per given CFA pattern; and we can support all known CFA patterns. It is therefore future-proof if new types of CFA were developed. To assess image quality, we adopt standard image quality measures for reference-image quality assessment, such as PSNR [4] and structural similarity (SSIM) [5]. Our contributions are the following:

1. An efficient non-parametric random field model for CFA color demosaicing;
2. a novel method for generating ground truth data for demosaicing research;
3. a new data set of natural images for demosaicing research recorded in linear light space with realistic noise. [1]

We first give an overview of the digital camera pipeline and how it relates to the demosaicing problem. In Section III, we discuss the most popular demosaicing methods. We give details on our novel procedure to produce ground truth images in Section IV. We describe our machine learning model in Section V and wrap up with experiments in Section VI.

II. System Overview

A typical digital camera pipeline [6] is depicted in Fig. 1. To understand the place of demosaicing in the pipeline, we discuss the processing that happens before demosaicing (the orange box). The *black-level adjustment* and *color scaling* transformation shifts and multiplies each color channel of the RAW recording separately by channel-specific constants. The color scaling transformation compensates for different efficiencies of the different color filter array channels, and the black-level adjustment removes a constant signal arising from the read-out mechanism used. Importantly, these transformations are carried out independently for each recorded response. We call the space of images after this adjustment the *linear color-scaled black-level adjusted light* space or in short, the *linear space*. The space is called *linear* because each sensel value is a linear function of exposure time and photon rate. A mosaiced image in linear space is the input to a demosaicing algorithm.

[1] Available at http://research.microsoft.com/en-us/um/cambridge/projects/msrdemosaic/

A demosaicing algorithm now takes a mosaiced image in linear space and produces a demosaiced image in linear space. The resulting image is then transformed non-linearly to produce an image in a standard color space such as sRGB [7] that can be displayed or compressed.

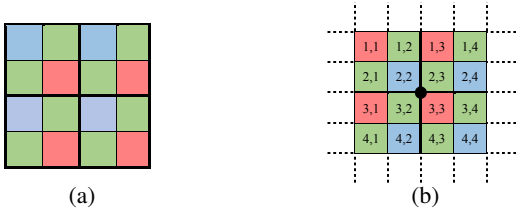
Unfortunately, ignoring the reality of digital camera systems, many researchers in the academic community design their demosaicing algorithms with a mosaiced sRGB image in mind. It is an artificial problem to demosaic a mosaiced sRGB image: this problem does not exist in any real camera system because in order to obtain sRGB colors one needs to transform the demosaiced linear-space measurement. As a result, each color channel in sRGB depends on all color channels of the linear space. We discuss these academic methods in Section III.

We believe the misplaced focus of the academic community on this artificial setting is due to two reasons. First, it is difficult to obtain realistic data sets for demosaiced images in linear space. We propose a procedure to provide such a data set for demosaicing research. Second, although demosaicing should not be done in sRGB, it might be a good idea to assess image quality in one of the final stages of the camera pipeline, e.g. sRGB because it is a color space exhibiting greater perceptual uniformity. This makes designing a demosaicing method for the linear space more challenging because it requires the color transformation (green box); our method works in linear space but optimizes quality in sRGB space. Specifically, we optimize with respect to the PSNR measure in sRGB space. However, using the machinery that we introduce, this optimization can be done using arbitrary performance measures, and at any stage of the camera pipline, so long as the transformation is continuous and twice differentiable.

III. Related work

A. Related Demosaicing Algorithms

Prior work often assumes the demosaicing problem is to be solved for the Bayer RGGB pattern [1], even though there are many cameras which follow other CFA patterns [8, 9]. Therefore, demosaicing methods that are not specific to any particular pattern or are easy to generalize to different patterns are preferable. It is worth mentioning that many of these



(a) (b)

Fig. 2: (a) The Bayer pattern. (b) A window of size 4×4.

alternative CFAs are designed based on the performance of demosaicing of sRGB images, for example [9]; By offering the potential to learn demosaicing methods for any type of CFA, our work may enable future unconventional CFA patterns. In the remainder of the paper, we always refer to the Bayer RGGB pattern and use the term CFA for all possible color filter arrays, not necessarily in the RGGB layout.

In almost all CFA patterns, the easiest way to fill in the missing values is via bilinear interpolation on the neighboring values for a given pixel, for each channel separately. There are two problems with this approach. First, it ignores the correlation of values between different channels and therefore does not use all information in the signal. Second, along high frequency signal changes such as edges and corners, the interpolated averages exhibit color artifacts or zippering due to the spatial offset of the individual color channels.

In order to create smooth images, this inter-channel correlation must be taken into account. One approach to make better use of the correlation structure of the signal is based on the observation that typical CFAs have one of the colors sampled more often than the other colors. For example, the sampling rate for the green channel in the Bayer pattern (Fig. 2a) is twice that of the red and blue channels. This motivates doing interpolation on the green channel first, and using the results for finding better interpolations for the other channels [10]. There are many other ways to model the correlation between channels; for example *smooth-hue assumption* in Cok's work [11], which interpolates the ratio of samples from different colors, i.e. G/R and G/B. Hamilton and Adam [12], as well as Laroche and Prescott [13], are using the assumption that the differences between channels, that is $G - R$ and $G - B$, are smooth. In the work by Lukac et al. [14] and Kimmel [15] it is assumed that the intensity-ratio image between different channels is smooth. Too much smoothing will result in edge artifacts, and more recent methods attempt to create a balance between smoothness from intra-channel correlation and inter-channel averaging, while being sensitive to edge orientations.

To address the artifacts at edges and corners, a common idea is to estimate the direction of an edge and interpolate only along the edge orientation, rather than across it. There are many variations of edge-based methods, with the differences being in the edge direction estimation. For example, Hibbard [16] uses the first order derivative as estimated in a 3×3 window of the green channel along horizontal and vertical directions. The direction with the smallest derivative is the estimated edge direction, along which interpolation is performed. Both Hamilton and Adams [12], and Laroche and Prescott [13], use the second derivative to estimate the interpolation direction. Hirakawa and Parks [17] choose the direction based on a local measure of similarity between pixels along two directions in the CIELAB color space. More sophisticated directional averaging schemes are suggested by Menon et al. [18] and, Zhang and Wu [19].

A number of demosaicing methods employ a sequential estimation or iterative refinement of the demosaiced image. To construct an initial solution some methods start from the most frequently sampled channel [12, 13, 20]. This initial solution is then iteratively refined. For example Gunturk et al. [10] solve the demosaicing problem as an alternating correction of channels, given the current estimate of the other channels, in the form a projection onto a suitable chosen set. In more recent work, Lu et al. [21] extend the alternating projection method [10] by solving a constrained quadratic optimization at increased computational cost.

Another successful class of methods is based on frequency domain filtering of different image components. Alleysson et al. [22] have shown that any Bayer CFA can be represented as the combination of a luminance component at baseband and two modulated components. Using this interpretation of Bayer samples, as well as the repetition of information in the chroma component of such a decomposition, Dubois [23] provided a better demosaicing algorithm. Later Dubois [24] introduced a least-squares method for filter-bank design of such models. In Dubois's later work [25] the model is generalized to arbitrary sensor patterns.

Recent demosaicing methods have explored the use of compressive sensing and sparse representation for solving under-determined systems of linear equations, such as the ones arising from CFA sample reconstruction under certain conditions [26, 27, 28, 29].

Other recent methods are based on using self-similarity of the signal by means of intelligent signal-adaptive smoothing, identifying dominant edge directions [30, 31, 32, 33].

To the best of our knowledge, there are few learning-based approaches to demosaicing. A Markov random field of separable filter banks has been introduced by Sun and Tappen [34]. One recent learning-based approach is by He et al. [35], which uses Support Vector Regression (SVR), where the input mosaiced images are used to create training data. To create the ground-truth images, an initial demosaicing is performed, followed by downscaling. This procedure is repeated several times to obtain images at different scales, which are then used in SVR training. One problem with this approach is that artifacts present in the initial demosaicing remain in the training data. Our approach is also based on using downscaling to produce ground truth data, but in our procedure such artifacts are avoided by taking into account the spatial position of different sensor elements.

B. Noise in Demosaicing

Digital complementary-metal-oxide semiconductor (CMOS) and charge-coupled-device (CCD) sensor recordings are corrupted with noise arising from a number of sources. The two most important sources are the *shot noise* and *read noise* [38]. *Shot noise* arises from the random variation of photon counts. To high accuracy, the true signal λ plus shot

Method and reference	Abbreviation
Bilinear Interpolation [36]	BI
Matlab demosaic(.) function	MBI
Directional MMSE [19]	D-MMSE
Weighted Edge and Colors Difference [30]	WECD
One Step AP [21]	OS-AP
Alternating Projections [10]	AP
Posterior Directional Filtering [18]	PDF
Non-Local Adaptive Thresholding [31]	NAT
Non-Local Means [31]	NLM
Adaptive Homogeneity-Directed Demosaicing [17]	AHD
Adaptive color plan interpolation (Hamilton & Adams) [12]	HA
Local Polynomial Approximation [37]	LPA
Contour Stencils [33]	CS
Regression Tree Field (this work)	RTF

TABLE I: Abbreviation of the methods used in comparisons

noise together follow a Poisson distribution, Poisson(λ). For larger values of λ this is approximately equal to Normal(λ, λ), and this is often used as practical approximation. *Read noise* arises from inefficiencies during reading and converting the amount of electrical charge into a digital count. This noise is approximately normally distributed, Normal$(0, \sigma_r^2)$. The measured signal is then approximately distributed as $X \sim$ Normal$(\lambda, \lambda + \sigma_r^2)$, see [38].

Noise affects demosaicing algorithms in a number of ways. For the class of interpolation based methods, Takamatsu et al. [39] show that interpolation decreases the noise power, i.e. the noise variance diminishes in the blocks which are averaged, while it remains the same in the blocks which are captured directly. This procedure changes the noise distribution in a complicated way and general purpose image denoising methods based on assuming iid Gaussian noise may no longer work, resulting in artifacts which are hard to remove after demosaicing [40].

It is therefore surprising that many published demosaicing methods discussed in the previous section are designed for demosaicing noise-free images in sRGB space. Our proposed method performs simultaneous denoising and demosaicing.

A number of demosaicing methods have explicitly addressed noisy input images. Hirakawa and Parks [41] modeled the noisy signal as $y = x + (a_0 + a_1 n)$, where $n \sim \mathcal{N}(0, 1)$ and experiments are conducted on sRGB images. Leung et al. [42] argue that the noise in *color-balanced gamma-corrected* (sRGB) space could be modeled as white Gaussian noise, with different variances for different channels. In much of the previous work on denoising with demosaicing, e.g. [40, 43, 44, 45], the task is performed in the synthetic setting of sRGB images contaminated with stationary white noise of known noise level. This prior work does not address the realistic problem of demosaicing a noisy image: demosaicing does not happen in sRGB and the noise model in linear space is not stationary Gaussian.

A more satisfactory treatment of demosaicing noisy images is given in work by Kalevo and Rantanen [46], where the effect of denoising before and after demosaicing is analyzed and it is experimentally confirmed that denoising is preferably done *before* demosaicing. There is a line of work that shows how to systematically perform denoising before demosaicing [47, 44, 48]. In the work by Menon and Calvagno [45], as well as Zhang et al. [49], the authors argue and show that joint demosaicing and denoising will yield better results. The total least square (TLS) denoising technique introduced by Hirakawa and Parks [50] is extended by the same authors [41] by applying CFA pattern constraints.

Some of the proposed methods for denoising and demosaicing are designed for the Bayer pattern. For example Zhang et al. [40] first estimate the color difference images with a minimum squared error method. During this procedure, both spectral and spatial content are used to suppress noise. Using the difference channels the method interpolates the full green channel, followed by wavelet-based denoising to eliminate channel-dependent noise. Finally, the blue and red channels are recovered. Similarly, Paliy et al. [37] suggest a spatially adaptive nonlinear filter by using local polynomial approximations, to eliminate the demosaicing noise generated in the demosaicing process. The model by Menon and Calvagno [45] utilizes space-varying filters, the parameters of which are optimized with a quadratic regularization term. Denoising is done by thresholding the coefficients of a wavelet transformed image. Taking inspiration from similar denoising algorithms, Zhang et al. [49] use principal component analysis to remove or reduce noise. More specialized algorithms, such as the one introduced by Danielyan et al. [44], extend denoising methods to the demosaicing setting; in this case the block matching 3D denoising method, by directly modelling cross-channel correlations. In the model by Chatterjee [48] denoising precedes demosaicing. Their model first identifies similar patches on the noisy observations, and uses the same patches to construct the denoised images from the noisy inputs, followed by demosaicing. In the method by Condat [51], the image is first separated into different frequency channels and denoising is performed on luminance. This approach is extended in the author's more recent work [43] by minimizing a total variation objective with additional consistency constraints to preserve smoothness of chrominance and sharpness of edges. In the work by Jeon and Duboi [52], the luma-chroma demosaicing method [42] is adapted to the demosaicing of noisy white-balanced gamma-corrected CFA images. The filters in their model are trained on a set of images with added artificial Gaussian noise. In many of these models, the underlying ideas are not limited to the Bayer pattern, but generalizing to other CFA patterns is not trivial.

IV. Generating Ground Truth from RAW Images

Demosaicing algorithms must work on the linear response space before the 3x3 transform, as shown in Figure 1. In order to assess the quality of demosaicing algorithms, or to use machine learning to learn a demosaicing algorithm, we need to provide pairs of mosaiced and "perfectly demosaiced" images in linear response space. While it is clear that we can obtain the mosaiced image by processing suitable RAW images from digital cameras, it is not obvious how we could obtain a fully observed output image, that is, an image that has all colors measured at all sensels.

It is important not to use sRGB images for demosaicing research. There are two reasons why we should instead work in the linear representation. For one, demosaicing algorithms work in the linear representation, which is the key reason. Even

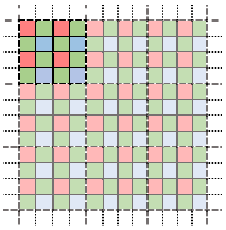


Fig. 3: The averaging blocks with $W = 2$.

obtaining a single sRGB channel per pixel already requires demosaicing. It is somewhat surprising to see that much of the academic demosaicing research has adopted mosaiced sRGB input images. A second reason why the linear representation should be used is that any sRGB image is the result of a demosaicing algorithm. Therefore, using sRGB images for research may treat demosaicing artifacts of existing algorithms as ground truth.

We generate ground truth data by downsampling the linear response image. The basic idea is to construct a larger "virtual sensel" composed of a large number of physical sensels. This virtual sensel has all color responses, each of which is a suitably chosen weighted combination of the physical sensel responses. By selecting the weights accordingly, we reduce or remove spatial bias arising from the physically different positions of different color filters on the sensel array.

Downsampling may change the structure of the signal in three ways. First, because natural images depict physical objects of the world, downsampling may change the signal statistics such as frequencies of different edges in different orientations. We argue that this effect is negligible because of the known *scale-invariance* of natural images [53]. Second, most digital cameras have optical low pass filters to avoid aliasing artifacts. In addition, most images have out-of-focus blur in some parts of the image. The downsampling method we describe below produces aliasing as would be obtained without a low pass filter. While our method can be adapted to not produce these artifacts, we argue that aliased input signals are intrinsic to the demosaicing problem and should be handled by a demosaicing method. In fact, recent cameras such as models by Pentax, Fuji, and Nikon have removed the optical low pass filter to increase fidelity.[2] Third, downsampling forms linear combinations of sensel measurements, thereby reducing the image noise. We analyze this effect in detail and use a recent noise model by Foi et al. [38] to add realistic noise to our downsampled images. We now discuss our proposed procedure.

A. Naive Block Downsampling

Consider the Bayer pattern shown in Figure 2b. One straightforward idea to simultaneously downscale and demosaic the image is to define a block of Bayer patterns as one new virtual sensel. In this sensel, all color measurements are available. For example, we can select non-overlapping blocks of even size $2W \times 2W$, and average the values of each color channel inside the block. This is shown in Figure 3 for $W = 2$, resulting in blocks of 4×4 sensels for averaging; we average the values for each channel, i.e., green, blue and red, and put together these values to create a new picture with height and width one fourth its original size. One sample result of doing this is depicted in the left-most column of Figure 4.

A careful look at this downscaling strategy makes it clear that it suffers from a systematic spatial shift between colors. In the case of the Bayer pattern, this happens for the red and the blue sensel because the spatial averages of their locations within a block differ. This shift is visible in the images shown in Figure 4. Note that this systematic shift appears only in even-sized blocks for the Bayer pattern. In the next section, we devise two ways to compensate for this shift.

B. Maximum Entropy Downsampling

Consider again a windows of size $2W \times 2W$, for example $W = 2$ as shown in Figure 2b. To compensate for the effect of the shift inside each color channel, we would like to select non-negative weights for each sensel contribution, so that we align the center of mass of each color channel at the center of the block. For the case $W = 2$, the center of mass would be $(2.5, 2.5)$, as shown in Figure 2b.

There are many possible weightings possible that satisfy this constraint. We therefore make two additional assumptions. First, the weights should correspond to a distribution and sum to one. Second, among all such possible distributions, we would like to pick the least concentrated distribution, as measured by the entropy. This is the principle of maximum entropy, and the resulting distribution is uniquely determined.

For each color channel $c \in \{r, g, b\}$, we solve one problem of the following form:

$$\max_{p_c} \quad -\sum_{x,y} p_c(x,y) \log p_c(x,y) \tag{1}$$

$$\text{sb.t.} \quad \sum_{x,y} p_c(x,y) = 1,$$

$$\sum_{x,y} x\, p_c(x,y) = W + 0.5, \quad \forall y, \tag{2}$$

$$\sum_{x,y} y\, p_c(x,y) = W + 0.5, \quad \forall x, \tag{3}$$

$$p_c(x,y) \geq 0, \qquad \forall x, y,$$

$$p_c(x,y) = 0, \qquad \forall (x,y) \notin S_c. \tag{4}$$

In general, the center point for a block of size $2W \times 2W$ is at $(W + 0.5, W + 0.5)$, and the pair of constraints (2) and (3) ensure that the spatial average is at the center of the block. For a suitably large block size, a solution to the problem is guaranteed to exist. Note that by constraint (4) the weights are zero for positions where the different color filters are used, that is when $(x, y) \notin S_c$, where S_c denotes the set of all sensel positions of a color filter type c.

We solve (1) for each channel separately. An exemplary solution for the block in Figure 2b gives the following weights, for p_r, p_g, and p_b, respectively:

$$\frac{1}{16}\left[\begin{array}{cc|cc} 1 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 \\ \hline 3 & 0 & 9 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right], \quad \frac{1}{8}\left[\begin{array}{cc|cc} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ \hline 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{array}\right], \quad \frac{1}{16}\left[\begin{array}{cc|cc} 0 & 0 & 0 & 0 \\ 0 & 9 & 0 & 3 \\ \hline 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 1 \end{array}\right].$$

[2]For example, the Nikon D7100 introduced in February 2013 and the Fuji X20 introduced in April 2013 use no optical low pass filter, and the Pentax K-3 introduced in October 2013 allows disabling the low pass filter.

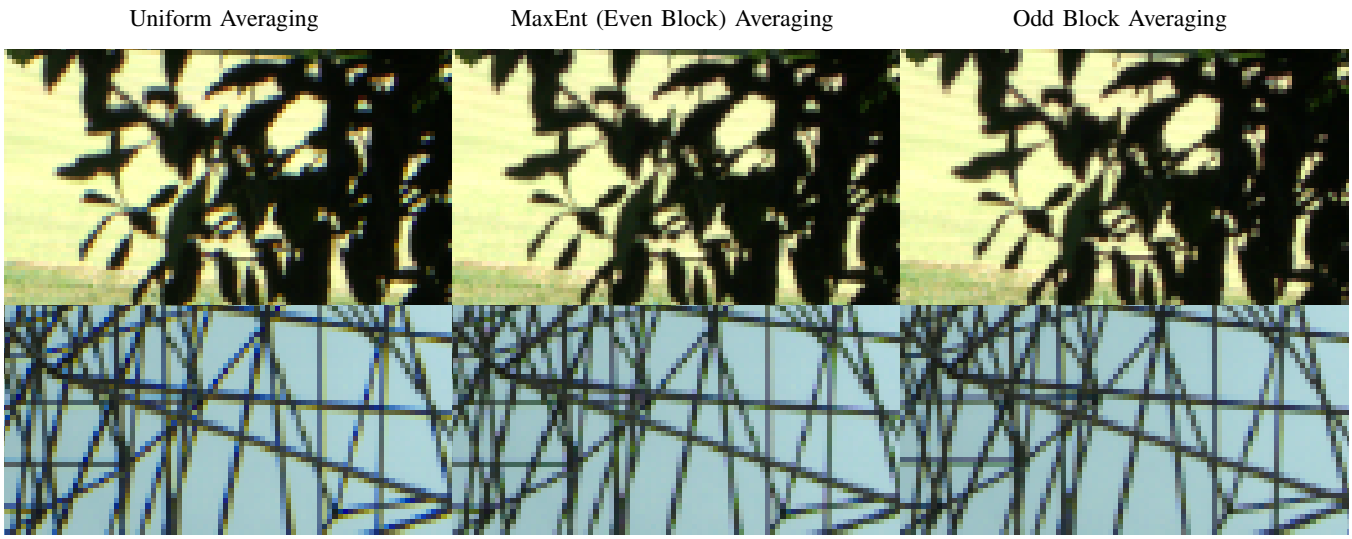


Fig. 4: Comparison between different downsampling methods: uniform averaging (left) results in a red/blue color bias. This effect is stronger in the edges, where there is a sharp color change. In the left figure, one can see the red shade below and to the right of edges, and the blue shade above and to the left of edges. This color bias is caused by the bias between the colors of the CFA; the phenomenon is substantially reduced in the center and right figures.

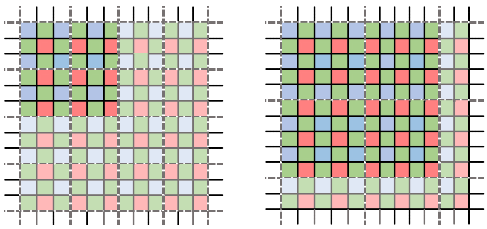


Fig. 5: Averaging with odd windows, with $W = 1$ (left) and $W = 2$ (right), where the window size is $(2W+1)\times(2W+1)$.

The result of using these weights for downscaling is shown in Figure 4. Comparing MaxEnt downscaling with the uniform downscaling scheme, we see that MaxEnt reduces the bias between colors inside the CFA. In general, the artifacts are reduced as the size of the averaging blocks is increased.

C. Odd Block Downsampling

Another way to downscale the linear-mosaiced images is to consider averaging blocks with odd sizes. Examples of such blocks are shown in Figure 5 for the cases $W = 1$ and $W = 2$, resulting in block sizes of $(2W + 1) \times (2W + 1)$. In the left rectangle in Figure 5, we consider averaging for each 3×3 block. In other words, each 3×3 block will be replaced with one single full RGB triple. Note that for odd windows, contrary to even-blocks, the center of mass for each channel lies at the center of each block; hence odd block averaging is a special case of the maximum entropy scheme, resulting in perfectly uniform weights. Averaging in such blocks does not create color shift. Also note that the pattern of colors change for each neighboring 3×3 block (and gets repeated every 6×6 block). Although the patterns change and may create some variation between neighboring blocks, this effect becomes negligible when the block size is large enough.

Some images obtained using this procedure are shown in the right column of Figure 4. The results are visually similar to the MaxEnt results, and the remaining artifacts decrease rapidly with increased block size. In particular, the visible blue/red patterns in high frequency areas are a result of small block sizes. Both the MaxEnt method and the odd block downsampling improve on the naive downsampling approach.

D. Image Noise

Downsampling images through weighted combinations of measurements reduces image noise. While desirable for generating low-noise ground truth images, we eventually will have to demosaic noisy images and hence need to produce an input mosaiced image with realistic noise characteristics in our data set. A simple argument for analyzing the change in the noise distribution is given by Takamatsu et al. [39], assuming the noise is Gaussian. Then, the variance of the average gets n-times smaller when uniformly averaging n samples.

The same argument can be generalized to the non-Gaussian case and to weighted averages: for a set of sensel measurements $y_i = x_i + \epsilon_i$ with underlying signal x_i and *spatially uncorrelated* zero-mean noise ϵ_i of variance σ_i^2, we can analyze the effect of taking weighted averages by a simple argument due to Bienaymé's formula. In particular, the weighted average $z = \sum_i w_i y_i$ has variance $\mathbb{V}[z] = \sum_i w_i^2 \sigma_i^2$, so that with uniform weights $w_i = 1/n$ this would yield $\mathbb{V}[z] = (\sum_i \sigma_i^2)/n^2$. For the special case of uniform noise $\sigma_i = \sigma$ this produces $\mathbb{V}[z] = \sigma^2/n$. Hence, for odd block averaging with large block sizes the resulting downsampled images are almost noise-free.

Since our goal is to create ground-truth images for training and evaluating demosaicing methods on the original linear mosaiced images, we devise a procedure for adding back realistic noise into the images. The noise models of RAW image samples are generally known to be accurately modeled as a combination of Poisson and Gaussian distribution. We use an established model for estimating the noise distribution

Fig. 6: Samples of downsampled images using MaxEnt strategy with window of size $W = 8$. The images are processed in linear space, but shown in sRGB. For large block sizes the downscaling method produces no visible color artifacts.

directly from a RAW image [38].[3] Once the noise model has been fitted to an input image, we then simulate additive noise from the estimated noise distribution and add it to the downsampled images. We show example images illustrating the process in Figure 7.

E. Camera Pipeline Simulation

To produce a data set for demosaicing research, we need to be able to process raw images in a realistic camera pipeline. To this end, we use dcraw,[4] a popular software package for raw image processing in academia and industry. Using dcraw and its camera pipeline, we design our own pipeline, enabling us to transfer the images up to any arbitrary processing stage.

Our procedure for creating the ground-truth images is summarized in Figure 8. The procedure mirrors the canonical camera pipeline, with two differences: (1) instead of demosaicing, we perform downscaling; and (2) we can add simulated noise to the demosaiced images. Although not strictly required for producing the data set, we would like to transfer the linear-space images to sRGB in order to visualize the outputs and we include a model of the gamma transformation for this purpose.

Once we have produced these ground truth images in linear space, we can create mosaiced input images by simply leaving out the corresponding measurements from the ground truth image. This enables us to produce input images for different color filter array patterns, despite having created the ground truth images from a Bayer CFA.

In the experiments we are interested in solving two tasks:

1) *Demosaicing in linear-space*: the ground-truth image and the corresponding mosaiced input are obtained by running our pipeline in Fig. 8 without noise addition.
2) *Simultaneous demosaicing and denoising in linear-space*: the ground-truth images are the noise-free linear-space image as in the previous task, but the input images are noisy in linear-space, and obtained from our pipeline with noise addition as explained in Section IV-D.

The second task corresponds to the realistic demosaicing task in a digital camera. We now introduce our model for these tasks, followed by experimental evaluation.

[3]We used the version 2.22 from http://www.cs.tut.fi/~foi/sensornoise.html

[4]Available at http://www.cybercom.net/~dcoffin/dcraw/

V. Regression Tree Fields for Demosaicing

Regression Tree Fields (RTF) are a recently proposed non-parametric random field model [3, 2] shown to be effective in low-level image processing applications. The model combines non-parametric regression tree models with Gaussian conditional random fields (CRF), for simultaneous prediction of a large number of dependent variables. Each "tree" is a binary decision tree that is trained in a greedy fashion, jointly with the parameters of the Gaussian random field, which are assigned to the leaves of the tree.

To make this more concrete, let us denote the observation variables with $\mathbf{x} \in \mathcal{X}$, and let us write the predicted output variables as $\mathbf{y} \in \mathcal{Y}$. These sets of variables are related by means of an *energy function* that defines a jointly Gaussian density over the output variables,

$$p(\mathbf{y}|\mathbf{x};\boldsymbol{\Theta}) \propto \exp\{-E(\mathbf{y}|\mathbf{x};\boldsymbol{\Theta})\}. \tag{5}$$

The energy function is specified by means of a *factor graph* [54] in which *factors* define interactions between pairs of output variables y_i, y_j. Typically the factors are between variables that are spatially close in the image plane. We use a *regular homogeneous structure* in that we define a set of *factor types* that are instantiated around each output variable [55, 56]. We denote the set of factors of type t by $\mathcal{F}_t$. For each type t there is a regression tree defined, parameterized with $\boldsymbol{\Theta}_t$. The tree of factor type t defines a set of leaves referred to by a set of indices $\mathcal{I}_t$, such that each leaf corresponds to one leaf index $i(t, \mathbf{x}_F) \in \mathcal{I}_t$. Each factor F acts on a set of output variables which we denote by $\mathbf{y}_F$, and a set of input variables of possibly different size $\mathbf{x}_F$. Any leaf with index $i(t, \mathbf{x}_F) \in \mathcal{I}_t$ contains a set of parameters $\{\mathbf{Q}_{i(t,\mathbf{x}_F)}, \mathbf{L}_{i(t,\mathbf{x}_F)}\}$. The set of parameters for the tree of type t consists of all parameters stored at the leaves of that tree, i.e., $\boldsymbol{\Theta}_t = \{\mathbf{Q}_{i(t,\mathbf{x}_F)}, \mathbf{L}_{i(t,\mathbf{x}_F)}\}_{i(t,\mathbf{x}_F)\in\mathcal{I}_t}$.

We can now write the energy function additively over energies relating to each factor individually. Each such local energy function is a quadratic function in $\mathbf{y}_F$,

$$E_t(\mathbf{y}_F|\mathbf{x}_F;\boldsymbol{\Theta}_t) = \frac{1}{2}\mathbf{y}_F^\top\mathbf{Q}_t(\mathbf{x}_F;\boldsymbol{\Theta}_t)\mathbf{y}_F - \mathbf{y}_F^\top\mathbf{L}_t(\mathbf{x}_F;\boldsymbol{\Theta}_t)\mathbf{b}_t(\mathbf{x}_F), \tag{6}$$

where we define $\mathbf{b}_t(\mathbf{x}_F) \in \mathbb{R}^{B_t}$ as a set of linear *basis vectors* specific to each factor type t. From the decomposition into

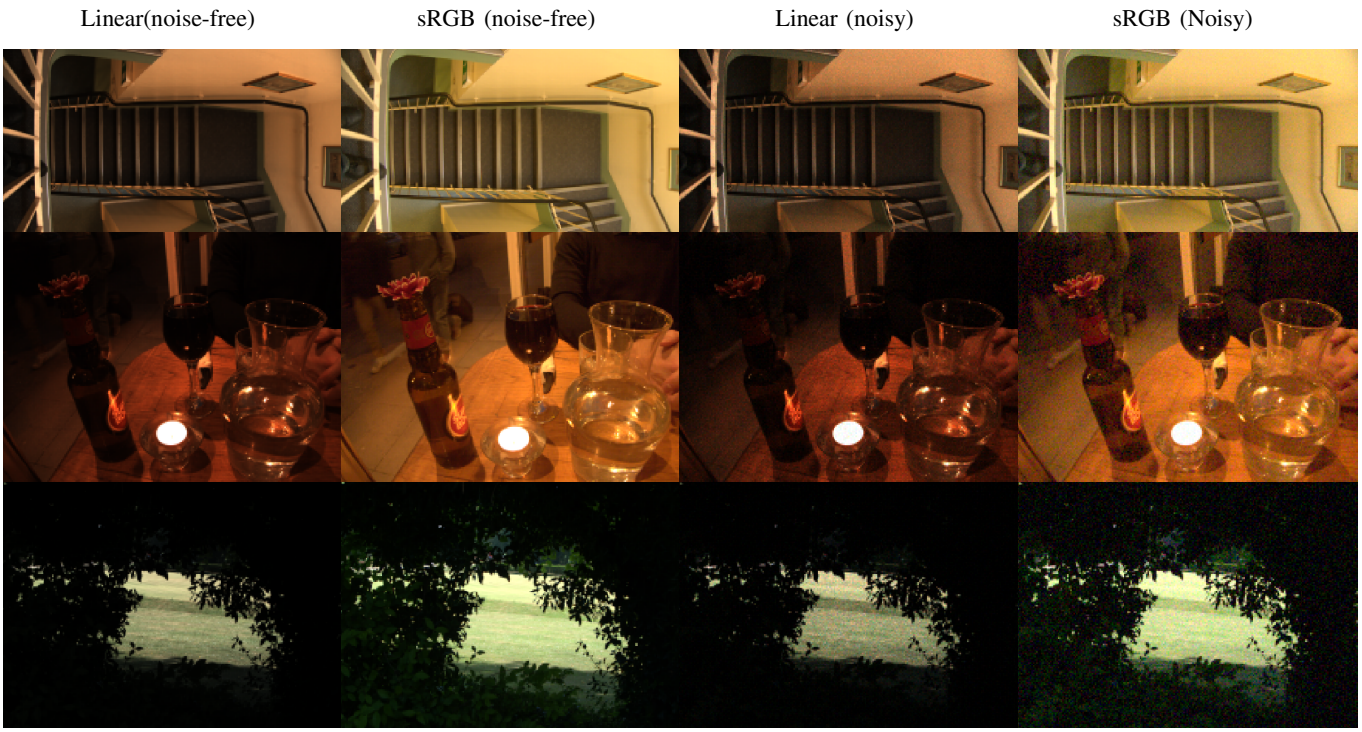


Fig. 7: Adding simulated noise to the downsampled linear images. We estimate the noise model on linear mosaiced images using the model and code of Foi et al. [38]. We then add simulated noise to the linear demosaiced images after downsampling. For each linear-space image we also show its sRGB-space visualization. Note the strong noise in the low-light areas of the right-most column (best viewed zoomed-in).

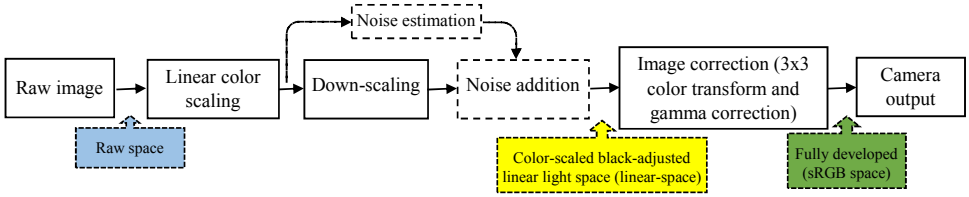


Fig. 8: Our proposed procedure for creating the ground truth demosaiced images.

local energy functions, the overall energy emerges as the sum over all factor energies,

$$E(\mathbf{y}|\mathbf{x};\boldsymbol{\Theta}) = \sum_t \sum_{F\in\mathcal{F}_t} E_t(\mathbf{y}_F|\mathbf{x}_F;\boldsymbol{\Theta}_t). \tag{7}$$

A *regression tree field* is then of the form (6) and (7), where each energy function E_t is determined by evaluating a regression tree on the observed image $\mathbf{x}$. We achieve this by storing one pair of quadratic and linear forms, $\mathbf{Q}_i$ and $\mathbf{L}_i$, at leaves of the regression tree. Evaluating the regression tree on the image observation $\mathbf{x}_F$ yields a leaf index $i(t,\mathbf{x}_F)$ that identifies the respective parameters. Thus,

$$\mathbf{Q}_t(\mathbf{x}_F;\boldsymbol{\Theta}) = \mathbf{Q}_{i(t,\mathbf{x}_F)}, \quad \text{and} \quad \mathbf{L}_t(\mathbf{x}_F;\boldsymbol{\Theta}) = \mathbf{L}_{i(t,\mathbf{x}_F)}.$$

Given this definition, the minimizer of the overall energy (7) is uniquely determined by

$$\boldsymbol{\mu} = [\mathbf{Q}(\mathbf{x};\boldsymbol{\Theta})]^{-1}\,\mathbf{L}(\mathbf{x};\boldsymbol{\Theta})\mathbf{b}(\mathbf{x}), \tag{8}$$

where $\mathbf{Q}(\mathbf{x};\boldsymbol{\Theta})$ is the quadratic form defined by summation of $\mathbf{Q}_t(\mathbf{x}_F;\boldsymbol{\Theta})$ over all types t and factors F. Notably, $\boldsymbol{\mu}$ is the mean of the Gaussian distribution, whereas $[\mathbf{Q}(\mathbf{x};\boldsymbol{\Theta})]^{-1}$ describes its covariance. The prediction of a regression tree field is thus determined as the mean of the Gaussian random field instantiated by its trees.

The role of the trees is to provide additional flexibility to the model, by instantiating different groups of parameters depending on the image context around the factors.

Regression tree fields support parameter estimation using either maximum likelihood [2], or empirical risk minimization (ERM) with arbitrary loss functions. The latter was recently shown to be preferable for image restoration [3].

The basic RTF model we use, as well as its risk minimization training algorithm, follow the presentation of Jancsary et al. [3]. However, to successfully apply the model to demosaicing, one needs to take into account the specific properties of this problem. In particular, we will use ERM with loss functions specifically designed for demosaicing, which reflect our model of the camera pipeline (cf. Figure 8). That is, we directly optimize the model parameters to provide optimal demosaicing performance. Moreover, compared to the generic image restoration tasks considered by [3], we use different types of features due to the different nature of the problem (e.g., the missing input values).

A. RTF Model for Demosaicing

To specialize the RTF model to the demosaicing problem we incorporate three demosaicing-specific features.

First, the model should have enough flexibility to represent different types of color filter arrays. In other words, each factor needs to know about the color at its spatial location. Since the regression trees determine the effective parameters of each factor, we can achieve this by using a group of color-sensitive branching features: Any (perfect) binary tree with $k-1$ nodes has k leaves. For a CFA of size $k \times k$, ignoring all symmetries in the CFA, we hence include $2 \times (k-1)$ branching features, $k-1$ for each x– and y–dimension.

Second, the factors must be sensitive to the edge directions in the mosaiced image. To achieve this, we allow further regression tree splits on responses of the *RFS filterbank*.[5] This filterbank includes 38 filters that correspond to derivatives of different directions and scales, computed on the mosaiced input image, as well as on a version of the image with missing values obtained through bilinear interpolation.

Third, we specify the basis functions $\mathbf{L}$ used in the linear term of the energy function. For the basis functions, we include the neighboring pixels in a neighborhood of size 5×5 around each pixel. This basis already allows learning of arbitrary linear filters in the neighboring window. In addition, we also include the RFS filter responses as basis functions.

As in the work by Schmidt et al. [57], we use a cascade of RTF models, where models are organized sequentially and each model receives as input the mosaiced input image as well as the demosaiced output of the previous model. Intuitively, the demosaiced output of a previous model will enable more reliable edge direction estimation in following cascade stages, thus improving demosaicing results. We treat the number of cascade layers as a hyper-parameter of the model that is chosen based on the best performance achieved on a validation set. Empirically, we observed monotonic performance improvements up to a certain number of cascade stages, after which the performance started declining again due to overfitting. For simultaneous demosaicing and denoising task we use the same model setup as for demosaicing, that is, we do not need to separate denoising from demosaicing [46, 47, 49, 48]. The RTF model learns the characteristics of imaging noise from the training data and does not require separate noise variance estimation prior to its use. However, if the noise characteristics were to change drastically, say if a much smaller sensor were to be used, this might require recreating a training set with matching noise characteristics.

B. Loss Functions for Demosaicing

The RTF model for demosaicing is trained to produce optimal results as measured with a specific loss function such as PSNR or mean absolute error. It is not apriori clear which loss functions best correspond to high visual image quality. By designing sensible loss functions it may be possible to improve the qualitative results produced by the model.

Although demosaicing happens in linear-space, images are displayed in a perceptually meaningful color space such as sRGB. We therefore propose to define the loss function in sRGB while performing demosaicing in linear-space. Because the linear-to-sRGB transformation can be specified as a non-linear differentiable function, we can still employ the standard RTF training procedure [3]. The differentiable transformation is given by $f : \mathbf{L} \to \mathbf{I}$, where $\mathbf{L}$ is the "linear response space image" which is mapped to $\mathbf{I}$, i.e. the sRGB space. Real camera response functions differ among camera vendors [58].

As an example, consider the squared error (SE) loss function for linear space, $\ell_{\text{SE}}^{(L)} : \mathbf{L} \times \mathbf{L} \to \mathbb{R}$. It is defined as

$$\ell_{\text{SE}}^{(L)}(\mathbf{L}, \tilde{\mathbf{L}}) = \sum_{i \in \{R,G,B\}} \sum_{x,y} (L_{x,y}^i - \tilde{L}_{x,y}^i)^2.$$

We now define the SE in image space (sRGB), $\ell_{\text{SE}}^{(I)} : \mathbf{I} \times \mathbf{I} \to \mathbb{R}$. We compose the response function with the SE loss,

$$\ell_{\text{SE}}^{(I)}(\mathbf{L}, \tilde{\mathbf{L}}) = (\ell_{\text{SE}} \circ f)(\mathbf{L}, \tilde{\mathbf{L}}) \quad (9)$$

$$= \sum_{i \in \{R,G,B\}} \sum_{x,y} (f(L_{x,y}^i) - f(\tilde{L}_{x,y}^i))^2. \quad (10)$$

For the response function we define a differentiable approximation using the imaging pipeline of dcraw, as

$$f(L_{x,y}) = \gamma(T\,[L_{x,y}^R, L_{x,y}^G, L_{x,y}^B]^\top),$$

where T is the average 3×3 color transformation matrix over all training images and $\gamma : [0,1] \to [0,\infty)$ is the average gamma curve used by dcraw for our training images.

VI. Experiments

To evaluate the performance of our models, we compute two performance measures commonly used in demosaicing research: peak signal-to-noise ratio (PSNR) and structural similarity (SSIM) [5]. We report PSNR as the average per-image PSNR values. For computing SSIM, we use the code provided by the original authors with default parameters.[6] Since SSIM is designed for grayscale images, we measure SSIM per channel and then take the average over all color channels.

For each experiment, we choose the RTF model achieving the best PSNR (or SSIM) on the validation data, and report its performance on the test images. The best RTF model is chosen among several models of different tree depth, cascade depth, and connectivity; each trained according to Section V.

We create our reference demosaicing data set using the procedure explained in Section IV, using MaxEnt downsampling for the creation of ground-truth images. In total, we use 500 natural images captured in both indoor and outdoor situations using a Panasonic Lumix DMC-LX3 CCD camera with Bayer CFA. We randomly partition the image set into 200 images for training, 100 for validation, and 200 for testing. The validation set is used to select hyper-parameters and the test set is used only once for reporting the results. In our experiments, we use a 16-bit range for the downsampled images. The dataset of our images will be released with the publication of this paper.

We compare our performance against a wide variety of methods from the literature. For some of these methods, we found considerable artifacts at the image margins, for

[5] Available at http://www.robots.ox.ac.uk/~vgg/research/texclass/filters.html

[6] Available at https://ece.uwaterloo.ca/~z70wang/research/iwssim/

	PSNR/Linear	PSNR/sRGB	SSIM/Linear	SSIM/sRGB
BI	30.86 ± 0.325	24.94 ± 0.238	0.882	0.727
MBI	35.22 ± 0.313	27.92 ± 0.225	0.960	0.881
D-MMSE	38.82 ± 0.313	31.71 ± 0.241	0.979	0.932
WECD	38.62 ± 0.321	31.50 ± 0.250	0.977	0.936
OS-AP	38.29 ± 0.324	31.07 ± 0.253	0.976	0.930
AP	38.19 ± 0.314	31.05 ± 0.242	0.976	0.929
PDF	38.09 ± 0.313	30.94 ± 0.241	0.976	0.922
NAT	37.64 ± 0.327	31.62 ± 0.250	0.973	0.929
NLM	38.42 ± 0.335	32.09 ± 0.255	0.978	0.939
HA	36.69 ± 0.327	29.97 ± 0.243	0.968	0.907
AHD	37.23 ± 0.326	31.26 ± 0.258	0.971	0.929
CS	**39.41** ± 0.318	**32.89** ± 0.238	0.980	0.939
LPA	39.24 ± 0.316	32.40 ± 0.249	**0.981**	**0.944**
PSNR-RTF	39.39 ± 0.323	32.63 ± 0.257	0.980	0.942

TABLE II: Results of demosaicing on linear-space noise-free images (strongest method shown bold). "Linear" columns: values calculated and optimized on linear-space images. "sRGB" columns: values calculated and optimized in sRGB space. We report PSNR along with the standard error. PSNR-RTF has 7 cascade stages, 3x3 connectivity and a tree depth of 13.

	PSNR/Linear	PSNR/sRGB	SSIM/Linear	SSIM/sRGB
BI	30.40 ± 0.296	24.09 ± 0.219	0.859	0.649
MBI	33.90 ± 0.258	26.62 ± 0.218	0.926	0.792
D-MMSE	36.65 ± 0.251	29.94 ± 0.243	0.949	0.864
WECD	36.38 ± 0.255	29.75 ± 0.244	0.947	0.867
OS-AP	36.09 ± 0.253	29.34 ± 0.243	0.944	0.860
AP	36.03 ± 0.250	29.23 ± 0.240	0.943	0.854
PDF	35.99 ± 0.252	29.14 ± 0.242	0.943	0.846
NAT	36.05 ± 0.270	29.78 ± 0.249	0.947	0.854
NLM	36.46 ± 0.269	30.06 ± 0.254	0.949	0.862
HA	35.02 ± 0.273	28.21 ± 0.245	0.936	0.823
AHD	35.59 ± 0.271	29.66 ± 0.248	0.943	0.866
CS	37.17 ± 0.256	30.93 ± 0.246	0.953	0.875
LPA	36.88 ± 0.250	30.37 ± 0.247	0.950	0.872
PSNR-RTF	**37.78** ± 0.280	**31.48** ± 0.246	**0.961**	**0.897**

TABLE III: Results of demosaicing on noisy images (strongest method is shown bold). "Linear" columns: values calculated and optimized on linear-space images. "sRGB" columns: values calculated and optimized in sRGB space. PSNR-RTF has 7 cascade stages, 1x1 connectivity, and a tree depth of 9.

	PSNR/Linear	PSNR/sRGB	SSIM/Linear	SSIM/sRGB
BI	30.40 ± 0.296	24.54 ± 0.212	0.860	0.679
MBI	34.16 ± 0.267	27.31 ± 0.212	0.934	0.818
D-MMSE	36.67 ± 0.252	29.96 ± 0.243	0.949	0.865
WECD	36.51 ± 0.261	30.02 ± 0.237	0.949	0.875
OS-AP	36.25 ± 0.262	29.66 ± 0.239	0.949	0.870
AP	36.13 ± 0.258	29.60 ± 0.235	0.947	0.867
PDF	36.16 ± 0.261	29.47 ± 0.238	0.947	0.857
NAT	36.11 ± 0.274	29.98 ± 0.248	0.949	0.862
NLM	36.56 ± 0.275	30.28 ± 0.254	0.951	0.870
HA	35.19 ± 0.280	28.66 ± 0.244	0.940	0.842
AHD	35.77 ± 0.279	29.95 ± 0.243	0.947	0.874
CS	37.27 ± 0.261	31.06 ± 0.235	0.955	0.879
LPA	37.00 ± 0.256	30.59 ± 0.247	0.953	0.878
PSNR-RTF	**37.78** ± 0.280	**31.48** ± 0.246	**0.961**	**0.897**

TABLE IV: Results of demosaicing on noisy images with additional denoising preprocessing: for methods other than RTF, the images are first denoised (with the noise level estimated from RAW images, cf. Section VI-B), and then the method is applied. "Linear" columns: values calculated and optimized on linear-space images. "sRGB" columns: values calculated and optimized in sRGB space. Same RTF model as in Table III.

example in the output of BI [36], D-MMSE [19], AP [10], and OS-AP [21]. Therefore, so as not put these methods at a disadvantage, we crop a margin of 7 pixels at the image edges before evaluating the performance. Our model does not exhibit such artifacts specific to image boundaries.

A. Demosaicing Noise-free Images in Linear-space

In this first experiment, we evaluate the demosaicing performance of different methods in the *noise-free* case. Although the assumption of noise-free input is unrealistic, it is an interesting "pure" form of the demosaicing problem. We include it because many of the competing methods do not explicitly handle noisy images and it may therefore be perceived unfair to evaluate on the noisy input images.

The results are depicted in Table II. Since the images are in linear response space, we report each measure both in the original linear responses as well as in transformed sRGB. Since this conversion is not the same for each image, it can change the relative ordering of the methods. Comparing the results, we see that our proposed RTF model is a very competitive model for demosaicing.

B. Joint Denoising and Demosaicing of Noisy Images

In this experiment, we demonstrate the effectiveness of our model for the task of simultaneous denoising and demosaicing. The inputs are *linear-space* images with synthetic Poissonian-Gaussian noise added. Following our explanations in Section IV-D, we estimate the noise level on the original raw images, using an established method for noise estimation on RAW images [38]. As such, we do not report any noise levels here, since the noise distribution varies from one image to another. For a detailed explanation regarding the generation of this dataset, we refer the reader to the Section IV-D.

As in the previous experiment, we report performance both in linear response space, as well as in sRGB space. To compare with previous work, we need a method that performs denoising on the mosaiced RAW images (rather than full sRGB images). Therefore, many of the academic denoising methods cannot be used because they are designed for full sRGB denoising. As such, we chose the popular wavelet-based denoising algorithm implemented in dcraw, variations of which are widely used in the industry. For existing methods, we report two performance numbers: one with, and one without input images pre-processed by denoising. For all existing methods, when using denoising as preprocessing, we use the dcraw wavelet-based denoiser with 10 different parameter values and pick the parameter with the best performance on the validation data. One parameter value corresponds to "no denoising". In practice, this parameter would need to be determined manually or using noise level estimation, a limitation that is not shared by our approach.

We report the results of this experiment in Table III (without prior denoising) and Table IV (with prior denoising). The results reveal the strong performance of our joint demosaicing and denoising model. In our current C++ implementation, the model in Tables III–IV takes about 0.15 seconds per stage to process a 220×132 image patch (as used in our dataset) on a 3.2 GHz Intel Core $i5$ machine. We expect further speed-up from an optimized implementation, possibly leveraging GPU hardware. To get a better sense of the output of different algorithms, we show a few output samples in Figure 9. In Section VI-E, we visualize real demosaiced images.

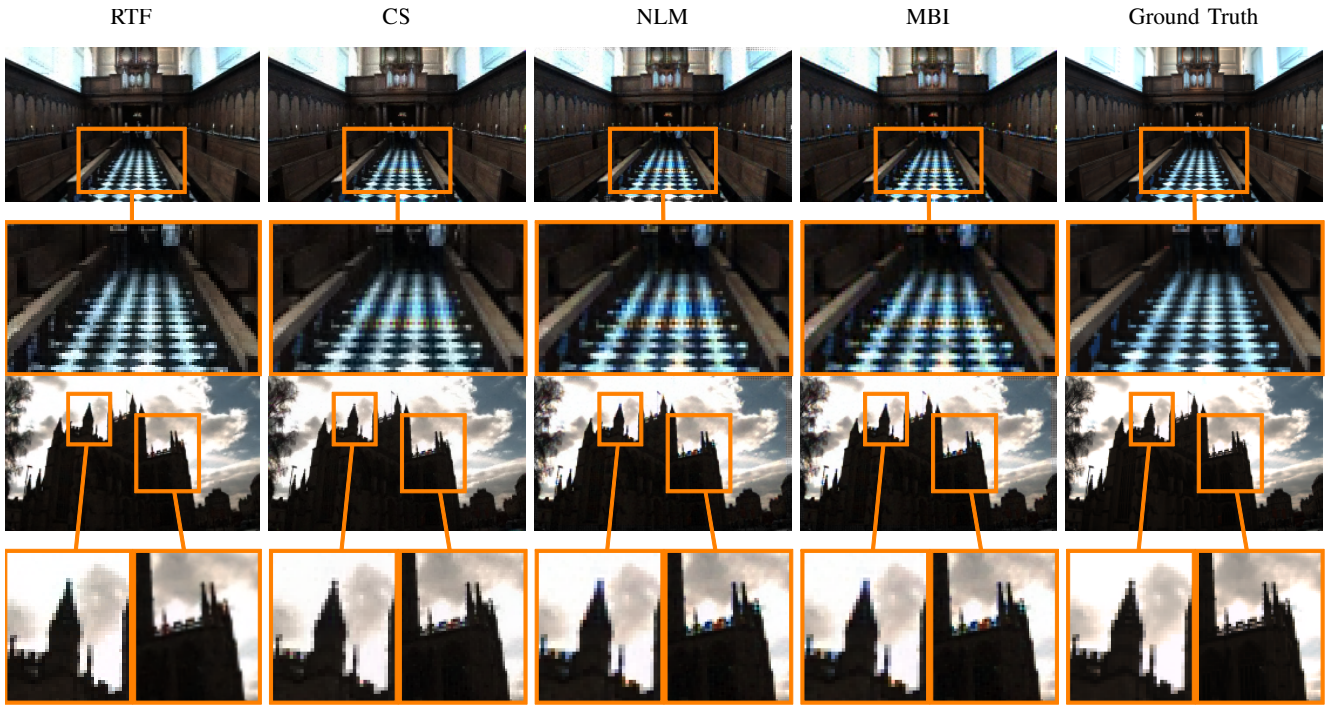


Fig. 9: Sample output images, for a few methods shown in Table IV.

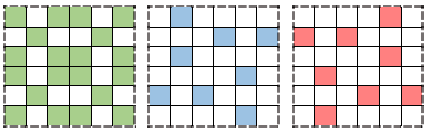


Fig. 10: Fujifilm X-Trans CFA pattern.

	PSNR/Linear	PSNR/sRGB	SSIM/Linear	SSIM/sRGB
NA	26.81 ± 0.296	21.06± 0.248	0.803	0.631
PSNR-RTF	**36.94** ± 0.466	**30.56** ± 0.409	**0.960**	**0.908**

TABLE V: Results of demosaicing for Fuji X-Trans CFA pattern and linear-space noisy images. PSNR-RTF has 7 cascade stages, 3x3 connectivity, and tree depth 4.

C. Demosaicing of Fuji X-Trans CFA Images

We now evaluate our model on another, less conventional CFA pattern. We use the *Fuji X-Trans* CFA pattern, which has recently been introduced.[7] The pattern is shown in Figure 10 and consists of a 6×6 pattern. To adapt our model to this CFA, we only need to produce a suitable training data set and specify the CFA pattern size. We generate training data by leaving out the corresponding color measurements according to the X-Trans pattern.

For this experiment, we evaluate on the noise-free linear-space images. We report the results in Table V. The results show that our approach generalizes to new patterns while retaining much of its performance. Because there are no competing methods for the X-Trans pattern, we provide a simple baseline similar to bilinear interpolation. We call this baseline "Neighbor Averaging" (NA) because for each position (x, y) it simply averages the color channels of neighboring samples within a rectangle of size 3×3 centered at (x, y).

Our system improves substantially on the simple baseline, which indicates that our learning-based approach to demosaicing is a promising way to rapidly produce competitive demosaicing methods for future CFA patterns.

[7] www.fujifilm.com/products/digital_cameras/x/fujifilm_x_pro1/features/

D. Camera-independence of the Model

Our model applies to different cameras based on the assumption that colors in RAW images behave similarly once black level correction and linear scaling have been performed. This assumption is likely valid because consumer cameras of leading vendors have very similar spectral sensitivities [59].

In this experiment, we verify the assumption by testing the performance of our method on images from a different camera vendor. The ground truth data is captured using a Panasonic Lumix DMC-LX3, but we evaluate the performance on a Canon 650D, also known as Canon Rebel T4i. If the performance remains competitive, we can reasonably assume that other Canon cameras would also work well. We use our trained model from Section VI-B and test it on a ground truth data set generated from 57 images of the Canon camera. We create the images using the same procedure as before and add synthetic noise. As in Section VI-B, we first perform dcraw wavelet denoising, and then perform demosaicing. We report the results in Table VI.

The results demonstrate that the method performs robustly despite using images from a different camera. It indicates that our model has learned to demosaic and denoise by learning about natural image statistics instead of using camera-specific properties of the input signal.

	PSNR/Linear	PSNR/sRGB	SSIM/Linear	SSIM/sRGB
BI	32.31 ± 0.606	24.88 ± 0.356	0.901	0.650
MBI	36.37 ± 0.558	28.08 ± 0.376	0.951	0.813
D-MMSE	39.48 ± 0.559	31.38 ± 0.445	0.968	0.872
OS-AP	38.99 ± 0.556	31.00 ± 0.468	0.964	0.881
AP	38.74 ± 0.552	30.70 ± 0.441	0.963	0.876
PDF	38.82 ± 0.567	30.51 ± 0.466	0.962	0.850
NAT	38.21 ± 0.595	30.61 ± 0.457	0.965	0.861
NLM	38.81 ± 0.601	31.21 ± 0.502	0.966	0.874
HA	37.43 ± 0.614	29.50 ± 0.463	0.958	0.844
AHD	38.42 ± 0.608	31.51 ± 0.486	0.964	0.890
CS	39.81 ± 0.543	32.18 ± 0.468	0.968	0.872
LPA	39.64 ± 0.563	31.85 ± 0.489	0.968	0.886
PSNR-RTF	**40.35** ± 0.552	**32.87** ± 0.439	**0.973**	**0.904**

TABLE VI: Testing our RTF model, trained on Panasonic images, on Canon images: for methods other than RTF, the images are first denoised (with the noise level estimated from RAW images, cf. Section VI-B), and then the method is applied. "Linear" columns: values calculated and optimized on linear-space images. "sRGB" columns: calculated and optimized on sRGB-space images. Same RTF model as in Table III.

E. Visual Quality on Real RAW Images

We demonstrate our method on real RAW images; because there is no ground truth available for quantitative evaluation, we show visual results in Figure 11. The RAW image was captured by a Panasonic Lumix LX3, to a size of 3983×2249 pixels. Next to the original image, we include zoomed versions of the output at important areas. The zoomed-in areas contain high-frequency contents that are hard to demosaic. Yet, our system produces convincing output, see Figure 11.

VII. Conclusion

We demonstrated a machine learning approach to the demosaicing problem that is versatile in that it can be applied to arbitrary tiled CFA patterns. The experiments validate the superior performance when the input signal is corrupted by noise. While the machine learning approach increases the reconstruction fidelity, it does so by adapting to the specific signal; in turn, this may mean that it is less robust than existing approaches in case the characteristics of the camera are changed. Synthesizing a suitable training set then requires an accurate characterization of the camera. In future work, we will investigate further demosaicing-specific loss functions.

Acknowledgment

The authors would like to thank Bruce Lindbloom for providing feedback on the project and the manuscript.

References

[1] B. E. Bayer, "Color imaging array," Jul. 20 1976, US Patent 3,971,065.

[2] J. Jancsary, S. Nowozin, and C. Rother, "Regression tree fields—An efficient, non-parametric approach to image labeling problems," in *Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2012.

[3] ——, "Loss-specific training of non-parametric image restoration models: a new state of the art," in *European Conference on Computer Vision (ECCV)*. Springer, 2012, pp. 112–125.

[4] Q. Huynh-Thu and M. Ghanbari, "Scope of validity of PSNR in image/video quality assessment," *Electronics letters*, vol. 44, no. 13, pp. 800–801, 2008.

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

[6] H.-C. Lee, *Introduction to Color Imaging Science*. Cambridge University Press, 2009.

[7] M. Stokes, M. Anderson, S. Chandrasekar, and R. Motta, "A standard default color space for the internet — sRGB," http://www.color.org/contrib/sRGB.html, 1996.

[8] Y. Li, P. Hao, Z. Lin, Y. Li, P. Hao, and Z. Lin, "Color filter arrays: Representation and analysis," *Dept. of Science, Queen Mary, Univ. London (QMUL), London, UK, Tech. Report no. RR-08-04*, 2008.

[9] J. Wang, C. Zhang, and P. Hao, "New color filter arrays of high light sensitivity and high demosaicking performance," in *International Conference on Image Processing (ICIP)*. IEEE, 2011, pp. 3153–3156.

[10] B. K. Gunturk, Y. Altunbasak, and R. M. Mersereau, "Color plane interpolation using alternating projections," *Image Processing, IEEE Transactions on*, vol. 11, no. 9, pp. 997–1013, 2002.

[11] D. R. Cok, "Signal processing method and apparatus for producing interpolated chrominance values in a sampled color image signal," Feb. 10 1987, US Patent 4,642,678.

[12] J. F. Hamilton Jr and J. E. Adams Jr, "Adaptive color plan interpolation in single sensor color electronic camera," May 13 1997, US Patent 5,629,734.

[13] C. A. Laroche and M. A. Prescott, "Apparatus and method for adaptively interpolating a full color image utilizing chrominance gradients," Dec. 13 1994, US Patent 5,373,322.

[14] R. Lukac, K. N. Plataniotis, and D. Hatzinakos, "Color image zooming on the Bayer pattern," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 15, no. 11, pp. 1475–1492, 2005.

[15] R. Kimmel, "Demosaicing: image reconstruction from color CCD samples," *Transactions on Image Processing*, vol. 8, no. 9, pp. 1221–1228, 1999.

[16] R. H. Hibbard, "Apparatus and method for adaptively interpolating a full color image utilizing luminance gradients," Jan. 17 1995, US Patent 5,382,976.

[17] K. Hirakawa and T. W. Parks, "Adaptive homogeneity-directed demosaicing algorithm," *IEEE Transactions on Image Processing*, vol. 14, no. 3, pp. 360–369, 2005.

[18] D. Menon, S. Andriani, and G. Calvagno, "Demosaicing with directional filtering and a posteriori decision," *Image Processing, IEEE Transactions on*, vol. 16, no. 1, pp. 132–141, 2007.

[19] L. Zhang and X. Wu, "Color demosaicking via directional linear minimum mean square-error estimation," *Image Processing, IEEE Transactions on*, vol. 14, no. 12, pp. 2167–2178, 2005.

[20] D. Taubman, "Generalized Wiener reconstruction of images from colour sensor data using a scale invariant prior," in *International Conference on Image Processing*

Fig. 11: Sample output of our model on a real RAW image, of size 3983×2249 pixels. Several important areas are zoomed in.

(ICIP), vol. 3. IEEE, 2000, pp. 801–804.

[21] Y. M. Lu, M. Karzand, and M. Vetterli, "Demosaicking by alternating projections: theory and fast one-step implementation," *Image Processing, IEEE Transactions on*, vol. 19, no. 8, pp. 2085–2098, 2010.

[22] D. Alleysson, S. Susstrunk, and J. Hérault, "Linear demosaicing inspired by the human visual system," *Image Processing, IEEE Transactions on*, vol. 14, no. 4, pp. 439–449, 2005.

[23] E. Dubois, "Frequency-domain methods for demosaicking of Bayer-sampled color images," *Signal Processing Letters, IEEE*, vol. 12, no. 12, pp. 847–850, 2005.

[24] ——, "Filter design for adaptive frequency-domain Bayer demosaicking," in *International Conference on Image Processing (ICIP)*. IEEE, 2006, pp. 2705–2708.

[25] ——, "Color-filter-array sampling of color images: Frequency domain analysis and associated demosaicking algorithms," in *Single-Sensor Imaging: Methods and Applications for Digital Cameras*, R. Lukac, Ed. Boca Raton, FL: CRC Press, 2009.

[26] A. Moghadam, M. Aghagolzadeh, M. Kumar, and H. Radha, "Compressive framework for demosaicing of natural images," *Image Processing, IEEE Transactions on*, vol. 22, no. 6, pp. 2356–2371, 2013.

[27] A. A. Moghadam, M. Aghagolzadeh, M. Kumar, and H. Radha, "Compressive demosaicing," in *Multimedia Signal Processing (MMSP)*. IEEE, 2010, pp. 105–110.

[28] M. Aghagolzadeh, A. A. Moghadam, M. Kumar, and H. Radha, "Bayer and panchromatic color filter array demosaicing by sparse recovery," in *IS&T/SPIE Electronic Imaging*. International Society for Optics and Photonics, 2011, pp. 787 603–787 603.

[29] J. Mairal, M. Elad, and G. Sapiro, "Sparse representation for color image restoration," *Image Processing, IEEE Transactions on*, vol. 17, no. 1, pp. 53–69, Jan 2008.

[30] C.-Y. Su, "Highly effective iterative demosaicing using weighted-edge and color-difference interpolations," *Consumer Electronics, IEEE Transactions on*, vol. 52, no. 2, pp. 639–645, 2006.

[31] L. Zhang, X. Wu, A. Buades, and X. Li, "Color demosaicking by local directional interpolation and nonlocal adaptive thresholding," *Journal of Electronic Imaging*, vol. 20, no. 2, 2011.

[32] P. Milanfar and P. Chatterjee, "A tour of modern image filtering," *IEEE Signal Process. Mag.*, vol. 30, no. 1, pp. 106–128, January 2013.

[33] P. Getreuer, "Color demosaicing with contour stencils," in *Digital Signal Processing (DSP)*. IEEE, 2011.

[34] J. Sun and M. Tappen, "Separable markov random field model and its applications in low level vision," *Image Processing, IEEE Transactions on*, vol. 22, no. 1, pp. 402–407, Jan 2013.

[35] F.-L. He, Y.-C. F. Wang, and K.-L. Hua, "Self-learning approach to color demosaicking via support vector regression," in *ICIP*. IEEE, 2012, pp. 2765–2768.

[36] D. R. Cok, "Single-chip electronic color camera with color-dependent birefringent optical spatial frequency filter and red and blue signal interpolating circuit," Aug. 12 1986, US Patent 4,605,956.

[37] D. Paliy, V. Katkovnik, R. Bilcu, S. Alenius, and K. Egiazarian, "Spatially adaptive color filter array interpolation for noiseless and noisy data," *International Journal of Imaging Systems and Technology*, vol. 17, no. 3, pp. 105–122, 2007.

[38] A. Foi, M. Trimeche, V. Katkovnik, and K. Egiazarian, "Practical Poissonian-Gaussian noise modeling and fitting for single-image raw-data," *Image Processing, IEEE Transactions on*, vol. 17, no. 10, pp. 1737–1754, 2008.

[39] J. Takamatsu, Y. Matsushita, T. Ogasawara, and K. Ikeuchi, "Estimating demosaicing algorithms using image noise variance," in *Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2010, pp. 279–286.

[40] L. Zhang, X. Wu, and D. Zhang, "Color reproduction from noisy CFA data of single sensor digital cameras," *Image Processing, IEEE Transactions on*, vol. 16, no. 9, pp. 2184–2197, 2007.

[41] K. Hirakawa and T. W. Parks, "Joint demosaicing and denoising," *Image Processing, IEEE Transactions on*, vol. 15, no. 8, pp. 2146–2157, 2006.

[42] B. Leung, G. Jeon, and E. Dubois, "Least-squares luma–chroma demultiplexing algorithm for Bayer demosaicking," *Image Processing, IEEE Transactions on*, vol. 20, no. 7, pp. 1885–1894, 2011.

[43] L. Condat and S. Mosaddegh, "Joint demosaicking and

denoising by total variation minimization," in *International Conference on Image Processing (ICIP)*, 2012, pp. 2781–2784.

[44] A. Danielyan, M. Vehvilainen, A. Foi, V. Katkovnik, and K. Egiazarian, "Cross-color BM3D filtering of noisy raw data," in *Local and Non-Local Approximation in Image Processing*. IEEE, 2009, pp. 125–129.

[45] D. Menon and G. Calvagno, "Joint demosaicking and denoisingwith space-varying filters," in *Image Processing (ICIP)*. IEEE, 2009, pp. 477–480.

[46] O. Kalevo and H. Rantanen, "Noise reduction techniques for Bayer-matrix images," in *Electronic Imaging*. Int. Soc. for Optics and Photonics, 2002, pp. 348–359.

[47] S. H. Park, H. S. Kim, S. Lansel, M. Parmar, and B. A. Wandell, "A case for denoising before demosaicking color filter array data," in *Signals, Systems and Computers*. IEEE, 2009, pp. 860–864.

[48] P. Chatterjee, N. Joshi, S. B. Kang, and Y. Matsushita, "Noise suppression in low-light images through joint denoising and demosaicing," in *Computer Vision and Pattern Recognition (CVPR)*, 2011, pp. 321–328.

[49] L. Zhang, R. Lukac, X. Wu, and D. Zhang, "PCA-based spatially adaptive denoising of CFA images for single-sensor digital cameras," *Image Processing, IEEE Transactions on*, vol. 18, no. 4, pp. 797–812, 2009.

[50] K. Hirakawa and T. W. Parks, "Image denoising for signal-dependent noise," in *IEEE ICASSP*, vol. 2, 2005, pp. 29–32.

[51] L. Condat, "A simple, fast and efficient approach to denoisaicking: Joint demosaicking and denoising," in *International Conference on Image Processing (ICIP)*. IEEE, 2010, pp. 905–908.

[52] G. Jeon and E. Dubois, "Demosaicking of noisy Bayer-sampled color images with least-squares luma-chroma demultiplexing and noise level estimation," *IEEE Trans. on Image Proc.*, vol. 22, no. 1-2, pp. 146–156, 2013.

[53] D. Zoran and Y. Weiss, "Scale invariance and noise in natural images," in *International Conference on Computer Vision (ICCV)*. IEEE, 2009, pp. 2209–2216.

[54] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 498–519, Feb. 2001.

[55] S. Nowozin and C. H. Lampert, "Structured learning and prediction in computer vision," *Foundations and Trends in Computer Graphics and Vision*, vol. 6, no. 3-4, 2011.

[56] D. Koller and N. Friedman, *Probabilistic graphical models: principles and techniques*. The MIT Press, 2009.

[57] U. Schmidt, C. Rother, S. Nowozin, J. Jancsary, and S. Roth, "Discriminative non-blind deblurring," in *Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2013.

[58] S. J. Kim, H. T. Lin, Z. Lu, S. Susstrunk, S. Lin, and M. Brown, "A new in-camera imaging model for color computer vision and its application," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 34, no. 12, pp. 2289–2302, 2012.

[59] J. Jiang, D. Liu, J. Gu, and S. Süsstrunk, "What is the space of spectral sensitivity functions for digital color cameras?" in *Winter Conference on Applications of Computer Vision*. IEEE, 2013.

Daniel Khashabi obtained his B.Sc. from Amirkabir University of Technology (Tehran Polytechnic) in 2012 and now he is a graduate student at University of Illinois, Urbana-Champaign, working at Cognitive Computation Group. In past he was an undergraduate researcher at Media Processing Lab at Tehran Polytechnic, as well as a research intern at Microsoft Research, Cambridge, UK. His interests lie at the intersection of theoretical and practical aspects of information processing, for learning, inference and decision making. His research is broadly related to combination of statistics, machine learning and other related applications, in Computer Vision and Natural Language Processing.

Sebastian Nowozin is a researcher in the Machine Learning and Perception group at Microsoft Research Cambridge. He received his Master of Engineering degree from the Shanghai Jiaotong University (SJTU) and his diploma degree in computer science with distinction from the Technical University of Berlin in 2006. He received his PhD degree summa cum laude in 2009 for his thesis on learning with structured data in computer vision, completed at the Max Planck Institute for Biological Cybernetics, Tübingen and the Technical University of Berlin. His research interest is at the intersection of computer vision and machine learning.

Jeremy Jancsary received his MSc in computer science in 2008 and his PhD in electrical engineering in 2012, both from Vienna University of Technology. Following a post-doctoral stay in the Machine Learning and Perception group of Microsoft Research, Cambridge, UK, Jeremy is now a Senior Research Scientist with Nuance Communications in Vienna, Austria. He is a recipient of the 2007-2012 ÖGAI prize for the best Austrian master's thesis in one of the areas of Artificial Intelligence, and a co-recipient of the CVPR 2013 Best Student Paper Award for joint work on Discriminative Non-blind Deblurring. Jeremy's research interests cover a broad spectrum, ranging from fundamental topics in machine learning and signal processing to applications in computer vision, image processing, as well as natural language and speech.

Andrew Fitzgibbon is a principal researcher at Microsoft Research, Cambridge, UK. His research interests are in the intersection of computer vision and computer graphics, with excursions into neuroscience. Recent papers have been on models of nonrigid motion, human 3D perception, large-scale image search, and nonlinear optimization.

He has co-authored prize papers at ICCV (2x), CVPR (2x), ISMAR (2x), and BMVC (2x), and software he wrote won an Engineering Emmy Award in 2002 for significant contributions to the creation of complex visual effects. He studied at University College Cork, at Heriot-Watt university, and received his PhD from Edinburgh University in 1997. Until June 2005 he held a Royal Society University Research Fellowship at Oxford University's Department of Engineering Science.