EURASIP Journal on Advances in Signal Processing
a SpringerOpen Journal

RESEARCH Open Access

Robust flash denoising/deblurring by iterative guided filtering

Hae-Jong Seo[1*] and Peyman Milanfar[2]

Abstract

A practical problem addressed recently in computational photography is that of producing a good picture of a poorly lit scene. The consensus approach for solving this problem involves capturing two images and merging them. In particular, using a flash produces one (typically high signal-to-noise ratio [SNR]) image and turning off the flash produces a second (typically low SNR) image. In this article, we present a novel approach for merging two such images. Our method is a generalization of the guided filter approach of He et al., significantly improving its performance. In particular, we analyze the spectral behavior of the guided filter kernel using a matrix formulation, and introduce a novel iterative application of the guided filter. These iterations consist of two parts: a nonlinear anisotropic diffusion of the noisier image, and a nonlinear reaction-diffusion (residual) iteration of the less noisy one. The results of these two processes are combined in an unsupervised manner. We demonstrate that the proposed approach outperforms state-of-the-art methods for both flash/no-flash denoising, and deblurring.

1 Introduction

Recently, several techniques [1-5] to enhance the quality of flash/no-flash image pairs have been proposed. While the flash image is better exposed, the lighting is not soft, and generally results in specularities and unnatural appearance. Meanwhile, the no-flash image tends to have a relatively low signal-to-noise ratio (SNR) while containing the natural ambient lighting of the scene. The key idea of flash/no-flash photography is to create a new image that is closest to the look of the real scene by having details of the flash image while maintaining the ambient illumination of the no-flash image. Eisemann and Durand [3] used bilateral filtering [6] to give the flash image the ambient tones from the no-flash image. On the other hand, Petschnigg et al. [2] focused on reducing noise in the no-flash image and transferring details from the flash image to the no-flash image by applying joint (or cross) bilateral filtering [3]. Agrawal et al. [4] removed flash artifacts, but did not test their method on no-flash images containing severe noise. As opposed to a visible flash used by [2-4], recently Krishnan and Fergus [7] used both near-infrared and near-ultraviolet illumination for low light image enhancement. Their so-called "dark flash" provides high-frequency detail in a less intrusive way than a visible flash does even though it results in incomplete color information. All these methods ignored any motion blur by either depending on a tripod setting or choosing sufficiently fast shutter speed. However, in practice, the captured images under low-light conditions using a hand-held camera often suffer from motion blur caused by camera shake.

More recently, Zhuo et al. [5] proposed a flash deblurring method that recovers a sharp image by combining a blurry image and a corresponding flash image. They integrated a so-called flash gradient into a maximum-a-posteriori framework and solved the optimization problem by alternating between blur kernel estimation and sharp image reconstruction. This method outperformed many states-of-the-art single image deblurring [8-10] and color transfer methods [11]. However, the final output of this method looks somewhat blurry because the model only deals with a spatially invariant motion blur.

Others have used multiple pictures of a scene taken at different exposures to generate high dynamic range images. This is called multi-exposure image fusion [12] which shares some similarity with our problem in that it seeks a new image that is of better quality than any of the input images. However, the flash/no-flash photography is generally more difficult due to the fact that there are only a pair of images. Enhancing a low SNR no-flash image

* Correspondence: seoha@sharplabs.com
[1]Sharp Labs of America, Camas, WA 98683, USA
Full list of author information is available at the end of the article

with a spatially variant motion blur only with the help of a single flash image is still a challenging open problem.

2 Overview of the proposed approach

We address the problem of generating a high quality image from two captured images: a flash image (Z) and a no-flash image (Y; Figure 1). We treat these two images, Z and Y, as random variables. The task at hand is to generate a new image (X) that contains the ambient lighting of the no-flash image (Y) and preserves the details of the flash-image (Z). As in [2], the new image X can be decomposed into two layers: a base layer and a detail layer;

$$\widehat{X} = \underbrace{\widehat{Y}}_{\text{base}} + \tau \underbrace{(Z - \widehat{Z})}_{\text{detail}}. \tag{1}$$

Here, Y might be noisy or blurry (possibly both), and $\widehat{Y}$ is an estimated version of Y, enhanced with the help of Z. Meanwhile, $\widehat{Z}$ represents a nonlinear, (low-pass) filtered version of Z so that $Z - \widehat{Z}$ can provide details. Note that τ is a constant that strikes a balance between the two parts. In order to estimate $\widehat{Y}$ and $\widehat{Z}$, we employ local linear minimum mean square error (LMMSE) predictors[a] which explain, justify, and generalize the idea of *guided filtering*[b] as proposed in [1]. More specifically, we assumed that $\widehat{Y}$ and $\widehat{Z}$ are a liner (affine) function of Z in a window ω_k centered at the pixel k:

$$\begin{aligned} \widehat{y}_i &= G(y_i, z_i) = az_i + b, \\ \widehat{z}_i &= G(z_i, z_i) = cz_i + d, \forall i \in \omega_k, \end{aligned} \tag{2}$$

where $G(\cdot)$ is the guided filtering (LMMSE) operator, $\widehat{z}_i$, $\widehat{z}_i$, z_i are samples of $\widehat{Y}$, $\widehat{Z}$,Z respectively, at pixel i, and (a, b, c, d) are coefficients assumed to be constant in ωk (a square window of size $p \times p$) and space-variant. Once we estimate a, b, c, d, Equation 1 can be rewritten as

$$\begin{aligned} \widehat{x}_i &= \widehat{y}_i + \tau(z_i - \widehat{z}_i), \\ &= az_i + b + \tau z_i - \tau c z_i - \tau d, \\ &= (a - \tau c + \tau) z_i + b - \tau d, \\ &= \alpha z_i + \beta. \end{aligned} \tag{3}$$

In fact, $\hat{x}_i$ is a linear function of z_i. While it is not possible to estimate α and β directly from (equation (3);

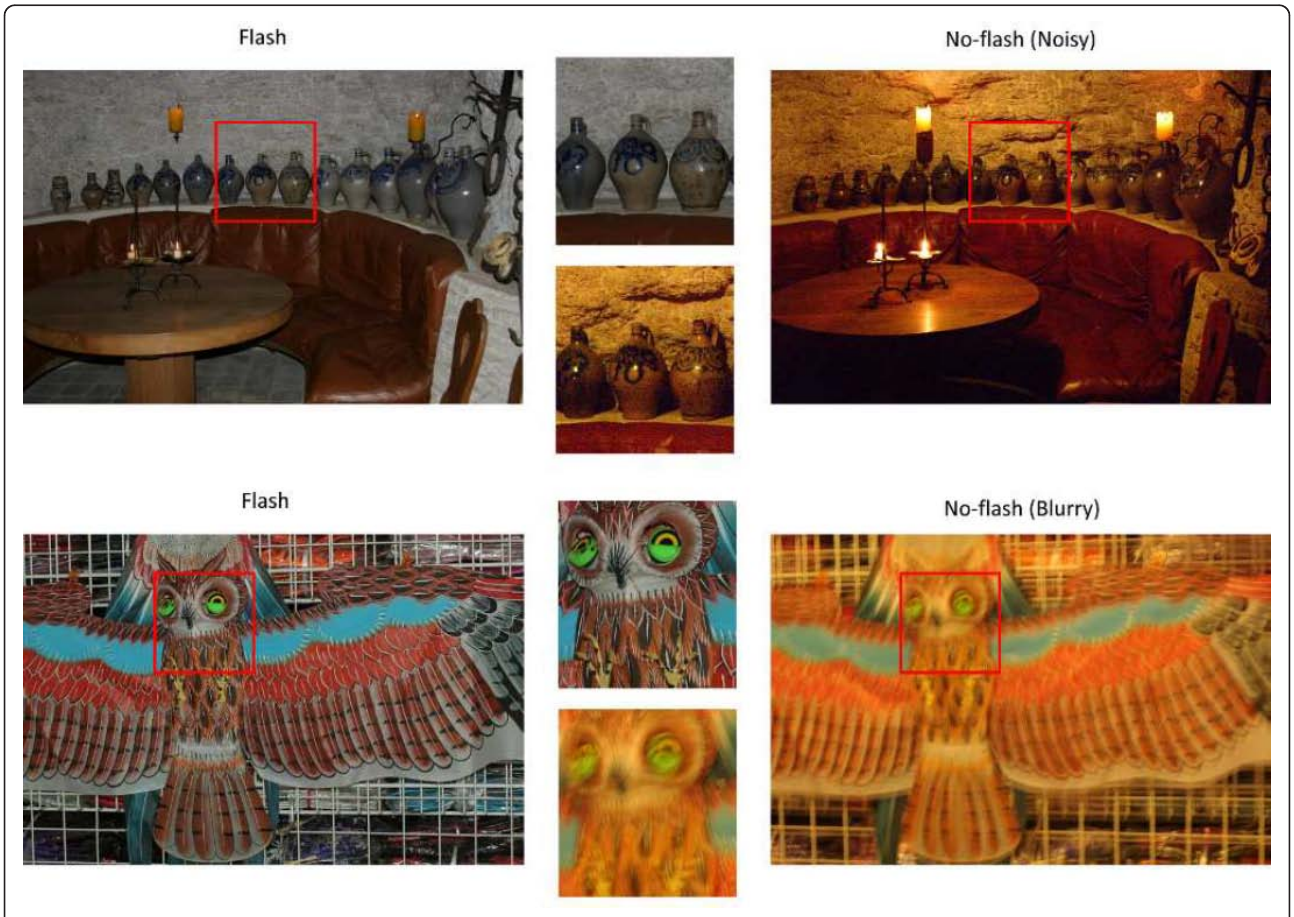

Figure 1 Flash/no-flash pairs. No-flash image can be noisy or blurry.

since they in turn depend on xi), the coefficients α, β can be expressed in terms of a, b, c, d which are optimally estimated from two different local linear models shown in Equation 2. Naturally, the simple linear model has its limitations in capturing complex behavior. Hence, we propose an iterative approach to boost its performance as follows:

$$\hat{x}_{i,n} = G(\hat{x}_{i,n-1}, z_i) + \tau_n(z_i - \hat{z}_i) = \alpha_n z_i + \beta_n, \tag{4}$$

where $\hat{x}_{i,0} = \gamma_i$ and α_n, β_n, and τ_n are functions of the iteration number n. A block-diagram of our approach is shown in Figure 2. The proposed method effectively removes noise and deals well with spatially variant motion blur without the need to estimate any blur kernel or to accurately register flash/no-flash image pairs when there is a modest displacement between them.

A preliminary version [13] of this article is appeared in the IEEE International Conference on Computer Vision (ICCV '11) workshop. This article is different from [13] in the following respects:

(1) We have provided a significantly expanded statistical derivation and description of the guided filter and its properties in Section 3 and Appendix.
(2) Figures 3 and 4 are provided to support the key idea of iterative guided filtering.
(3) We provide many more experimental results for both flash/no-flash denoising and de- blurring in Section 5.
(4) We describe the key ideas of diffusion and residual iteration and their novel relevance to iterative guided filtering in the Appendix.
(5) We prove the convergence of the proposed iterative estimator in the Appendix.
(6) As supplemental material, we share our project website[c] where flash/no-flash relighting examples are also presented.

In Section 3, we outline the guided filter and study its statistical properties. We describe how we actually estimate the linear model coefficients a, b, c, d and α, β, and we provide an interpretation of the proposed iterative framework in matrix form in Section 4. In Section 5, we demonstrate the performance of the system with some experimental results, and finally we conclude the article in Section 6.

3 The guided filter and its properties

In general, space-variant, nonparametric filters such as the bilateral filter [6], nonlocal means filter [14], and locally adaptive regression kernels filter [15] are estimated from the given corrupted input image to perform denoising. The guided filter can be distinguished from these in the sense that the filter kernel weights are computed from a (second) guide image which is presumably cleaner. In other words, the idea is to apply filter kernels W_{ij} computed from the guide (e.g., flash) image Z to the more noisy (e.g., no-flash) image Y. Specifically, the filter output sample $\hat{\gamma}$ a pixel i is computed as a weighted average[d]:

$$\hat{\gamma}_i = \sum_j W_{ij}(Z)\gamma_j. \tag{5}$$

Note that the filter kernel W_{ij} is a function of the guide image Z, but is independent of Y. The guided filter kernel[e] can be explicitly written as

$$W_{ij}(Z) = \frac{1}{|\omega|^2} \sum_{k:(i,j)\in\omega_k} \left(1 + \frac{(z_i - E[Z]_k)(z_j - E[Z]_k)}{\mathbf{var}(Z)_k + \varepsilon}\right), \quad i,j \in \omega_k, \tag{6}$$

where $|\omega|$ is the total number of pixels ($= p^2$) in ω_k, ε is a global smoothing parameter, $E[Z]_k \approx \frac{1}{|\omega|}\sum_{l\in\omega_k} z_l$, and $\mathbf{var}(Z)_k \approx \frac{1}{|\omega|}\sum_{l\in\omega_k} z_l^2 - E[Z]_k^2$. Note that W_{ij} are normalized weights, that is, $\sum_j W_{ij}(Z) = 1$ Figure 5 shows examples of guided filter weights in four different

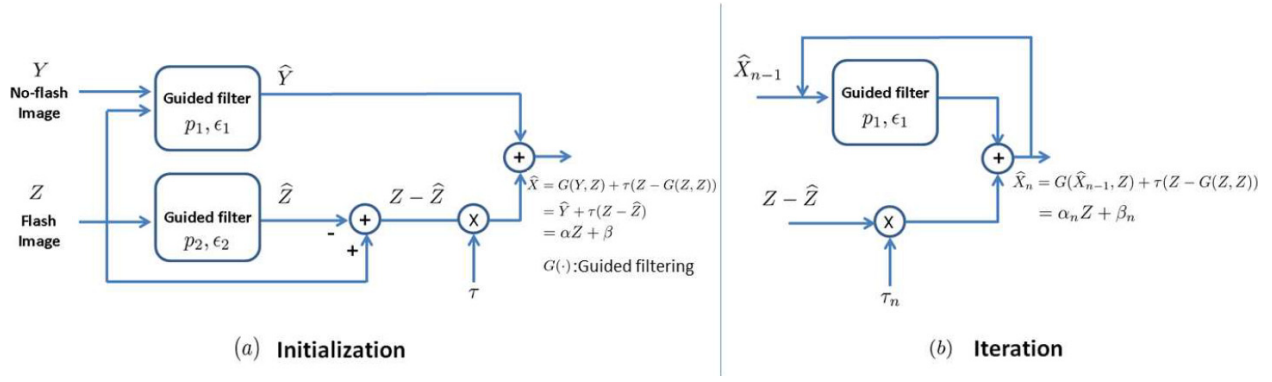

Figure 2 System overview. Overview of our algorithm for flash/no-flash enhancement.

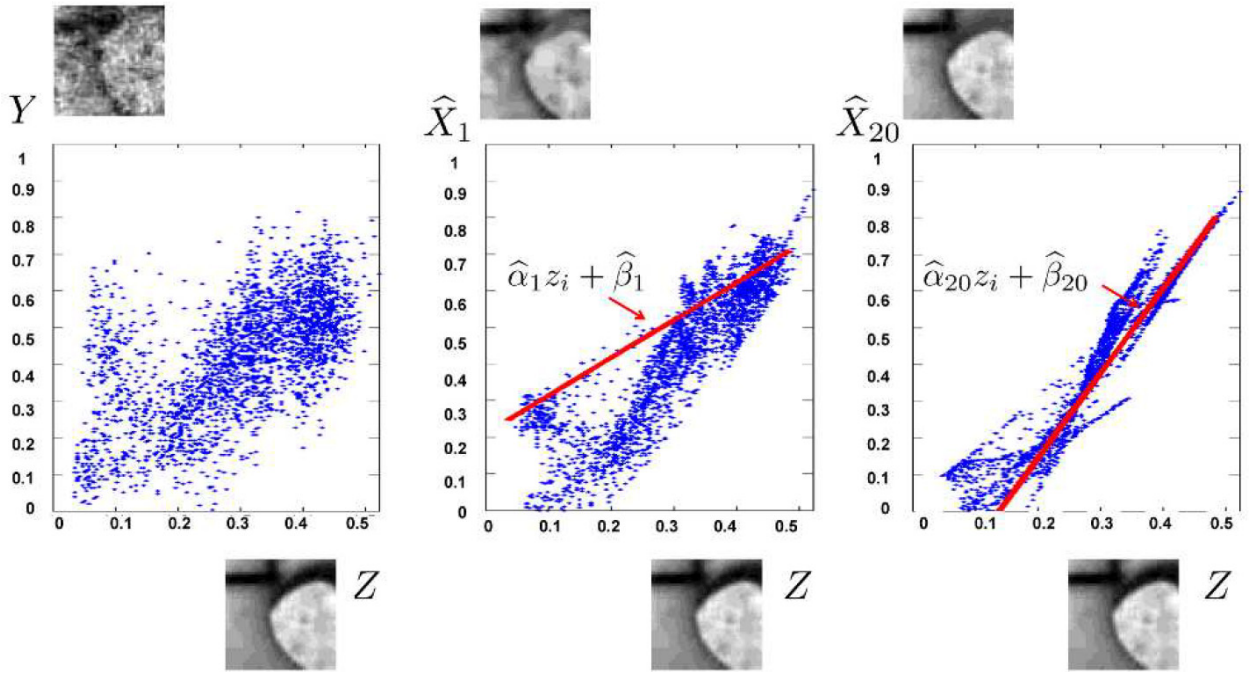

Figure 3 Iteration improves accuracy. Accuracy of $(\hat{\alpha}_{20}, \hat{\beta}_{20})$ is improved upon $(\hat{\alpha}_1, \hat{\beta}_1)$.

patches. We can see that the guided filter kernel weights neatly capture underlying geometric structures as do other data-adaptive kernel weights [6,14,15]. It is worth noting that the use of the specific form of the guided filter here may not be critical in the sense that any other data-adaptive kernel weights such as *non-local means kernels* [16] and *locally adaptive regression kernels* [15] could be used.

Next, we study some fundamental properties of the guided filter kernel in matrix form.

We adopt a convenient vector form of Equation 5 as follows:

$$\hat{y}_i = \mathbf{w}_i^T \mathbf{y}, \tag{7}$$

where $\mathbf{y}$ is a column vector of pixels in Y and $\mathbf{w}_i^T = [W(i,1), W(i,2), \ldots, W(i,N)]$ is a vector of weights for each i. Note that N is the dimension[f] of $\mathbf{y}$. Writing the above at once for all i we have,

$$\widehat{\mathbf{y}} = \begin{bmatrix} \mathbf{w}_1^T \\ \mathbf{w}_2^T \\ \vdots \\ \mathbf{w}_N^T \end{bmatrix} = \begin{bmatrix} W(1,1) & W(1,2) & \ldots & W(1,N) \\ W(2,1) & W(2,2) & \ldots & W(2,N) \\ \vdots & \vdots & \ddots & \vdots \\ W(N,1) & W(N,2) & \ldots & W(N,N) \end{bmatrix} = \mathbf{W}(\mathbf{z})\mathbf{y}, \tag{8}$$

where $\mathbf{z}$ is a vector of pixels in Z and $\mathbf{W}$ is only a function of $\mathbf{z}$. The filter output can be analyzed as the product of a matrix of weights $\mathbf{W}$ with the vector of the given the input image $\mathbf{y}$.

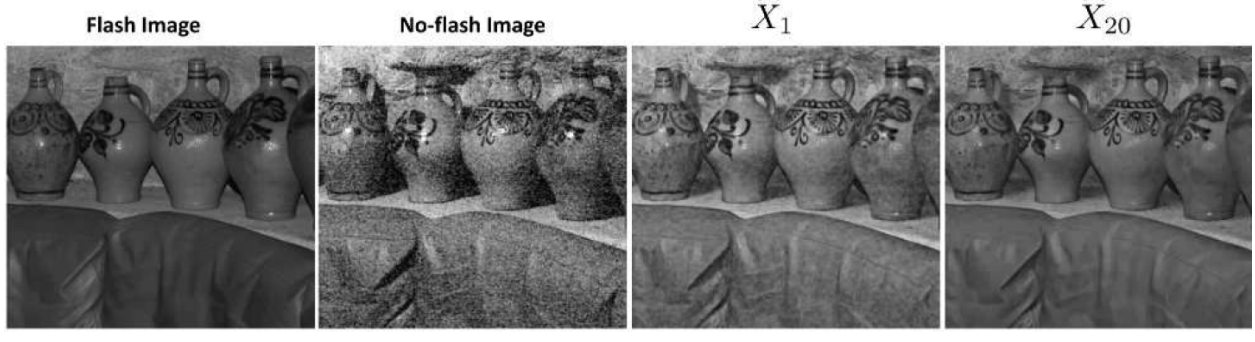

Figure 4 Effect of iteration. As opposed to $\widehat{X}_1$, noise in Y was more effectively removed in $\widehat{X}_{20}$.

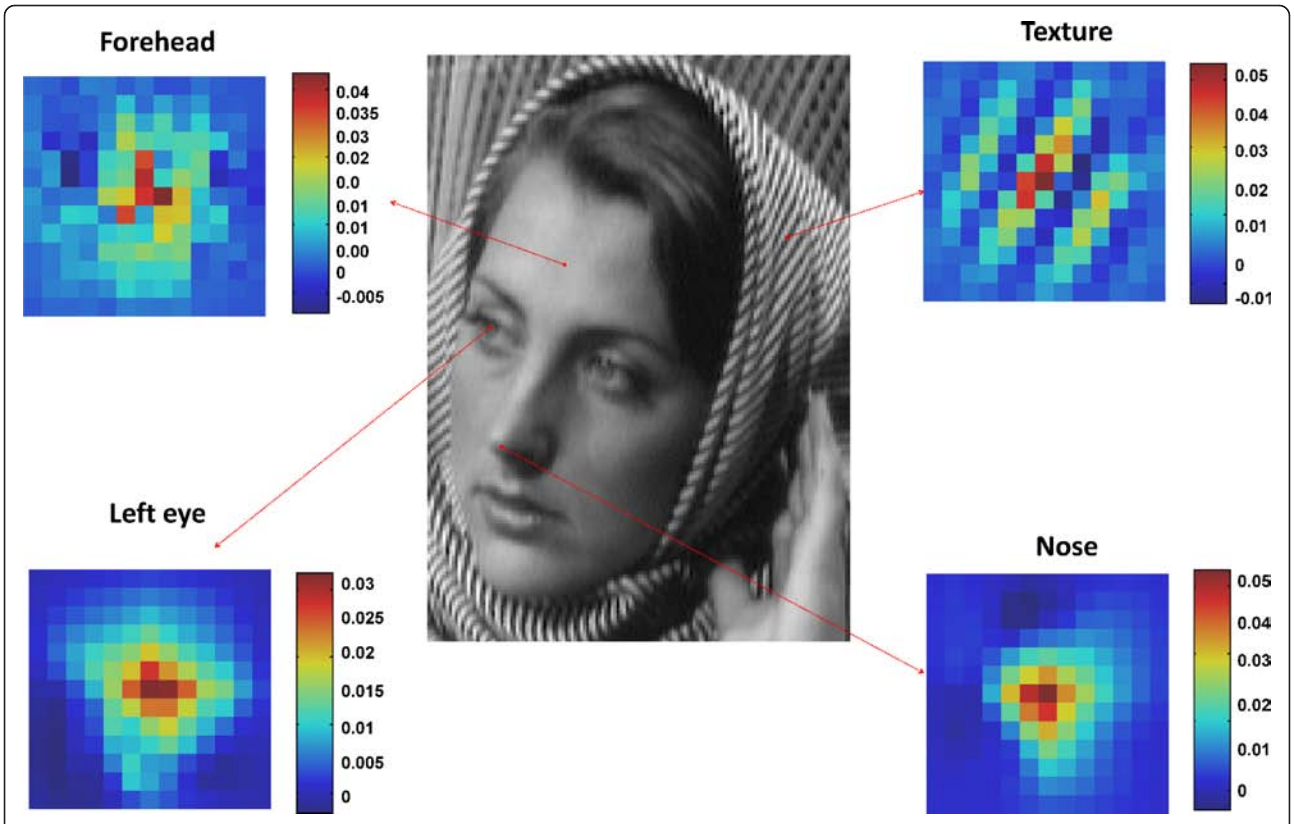

Figure 5 Examples of guided filter kernel weights in four different patches. The kernel weights well represent underlying structures.

The matrix $\mathbf{W}$ is symmetric as shown in Equation 8 and the sum of each row of $\mathbf{W}$ is equal to one ($\mathbf{W}\mathbf{1}_N = \mathbf{1}_N$) by definition. However, as seen in Equation 6, the definition of the weights does not necessarily imply that the elements of the matrix $\mathbf{W}$ are positive in general. While this is not necessarily a problem in practice, we find it useful for our purposes to approximate this kernel with a proper admissible kernel [17]. That is, for the purposes of analysis, we approximate $\mathbf{W}$ as a positive valued, symmetric positive definite matrix with rows summing to one, as similarly done in [18]. For the details, we refer the reader to the Appendix A.

With this technical approximation in place, all eigenvalues λ_i (i = 1, ..., N) are real, and the largest eigenvalue of $\mathbf{W}$ is exactly one (λ_1 = 1), with corresponding eigenvector $\mathbf{v}_1 = (1/\sqrt{N})[1, 1, \ldots, 1]^T = (1/\sqrt{N})\mathbf{1}_N$ as shown in Figure 6. Intuitively, this means that filtering by $\mathbf{W}$ will leave a constant signal (i.e., a "flat" image) unchanged. In fact, with the rest of its spectrum inside the unit disk, powers of $\mathbf{W}$ converge to a matrix of rank one, with identical rows, which (still) sum to one:

$$\lim_{n\to\infty} \mathbf{W}^n = \mathbf{1}_N \mathbf{u}_1^T. \tag{9}$$

So $\mathbf{u}1$ summarizes the asymptotic effect of applying the filter $\mathbf{W}$ many times. Figure 7 shows what a typical $\mathbf{u}_1$ looks like.

Figure 8 shows examples of the (center) row vector ($\mathbf{w}^T$) from $\mathbf{W}$'s powers in one patch of size 25 × 25. The vector was reshaped into an image for illustration purposes. We can see that powers of $\mathbf{W}$ provide even better structure by generating larger (and more sophisticated) kernels. This insight reveals that applying $\mathbf{W}$ multiple times can improve the guided filtering performance, which leads us to the iterative use of the guided filter. This approach will produce the evolving coefficients α_n, β_n introduced in (4). In the following section, we describe how we actually compute these coefficients based on Bayesian mean square error (MSE) predictions.

4 Iterative application of local LMMSE predictors

The coefficients[g] a_k, b_k, c_k, d_k in (3) are chosen so that "on average" the estimated value $\widehat{Y}$ is close to the observed value of Y (= y_i) in ω_k, and the estimated value $\widehat{Z}$ is close to the observed value of Z (=z_i) in ω_k. More specifically, we adopt a stabilized MSE criterion in the window ω_k as our measure of closeness[h]:

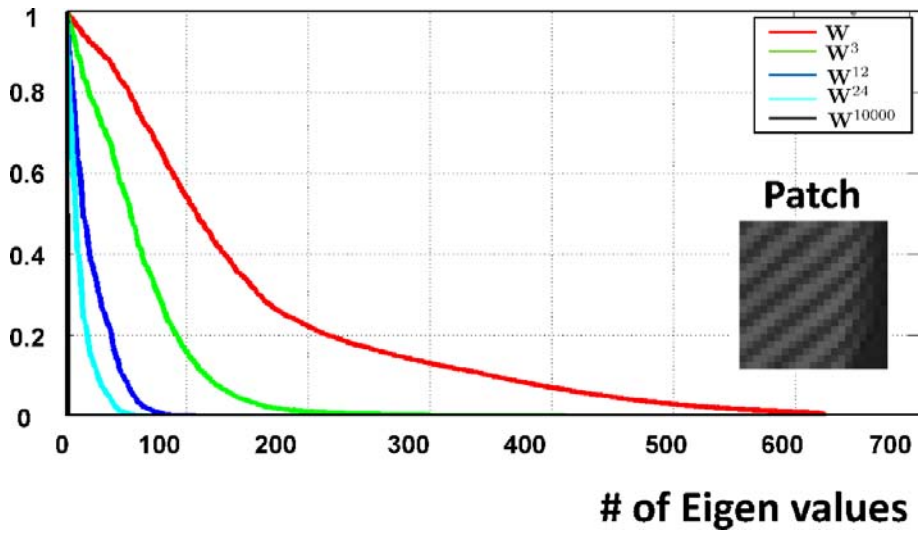

Figure 6 Examples of W in one patch of size 25 × 25. All eigenvalues of the matrix **W** are nonnegative, thus the guided filter kernel matrix is positive definite matrix. The largest eigenvalue of **W** is one and the rank of **W** asymptotically becomes one. This figure is better viewed in color.

$$\begin{aligned}\mathrm{MSE}(a_k, b_k) &= E[(Y-\widehat{Y})^2] + \varepsilon_1 a_k^2 = E[(Y - a_k Z - b_k)^2] + \varepsilon_1 a_k^2,\\ \mathrm{MSE}(c_k, d_k) &= E[(Z-\widehat{Z})^2] + \varepsilon_2 c_k^2 = E[(Z - c_k Z - d_k)^2] + \varepsilon_2 c_k^2,\end{aligned} \tag{10}$$

where ε_1 and ε_2 are small constants that prevent $\hat{a}_k$, $\hat{c}_k$ from being too large. Note that c_k and d_k become simply 1 and 0 by setting $\varepsilon_2 = 0$. By setting partial derivatives of $\mathrm{MSE}(a_k, b_k)$ with respect to a_k, b_k, and partial derivatives of $\mathrm{MSE}(c_k, d_k)$ with respect to c_k, d_k, respectively, to zero, the solutions to minimum MSE prediction in (10) are

$$\begin{aligned}\hat{a}_k &= \frac{E[ZY] - E[Z]E[Y]}{E[Z^2] - E^2[Z] + \varepsilon_1} = \left[\frac{\mathbf{cov}(Z,Y)}{\mathbf{var}(Z) + \varepsilon_1}\right]_k,\\ \hat{b}_k &= E[Y] - \hat{a}_k E[Z] = E[Y]_k - \left[\frac{\mathbf{cov}(Z,Y)}{\mathbf{var}(Z) + \varepsilon_1}\right]_k E[Z]_k,\\ \hat{c}_k &= \frac{E[Z^2] - E^2[Z]}{E[Z^2] - E^2[Z] + \varepsilon_2} = \left[\frac{\mathbf{var}(Z)}{\mathbf{var}(Z) + \varepsilon_2}\right]_k,\\ \hat{d}_k &= E[Z] - \hat{c}_k E[Z] = E[Z]_k - \left[\frac{\mathbf{var}(Z)}{\mathbf{var}(Z) + \varepsilon_2}\right]_k E[Z]_k,\end{aligned} \tag{11}$$

where we compute $E[Z] \approx \frac{1}{|\omega|}\sum_{l\in\omega_k} z_l$, $E[Y] \approx \frac{1}{|\omega|}\sum_{l\in\omega_k} y_l$, $E[ZY] \approx \frac{1}{|\omega|}\sum_{l\in\omega_k} z_l y_l$, $E[Z^2] \approx \frac{1}{|\omega|}\sum_{l\in\omega_k} z_l^2$.

Note that the use of different ω_k results in different predictions of these coefficients. Hence, one must compute an aggregate estimate of these coefficients coming from all windows that contain the pixel of interest. As an illustration, consider a case where we predict $\widehat{y}_i$ using observed values of Y in ω_k of size 3 × 3 as shown in Figure 9. There are nine possible windows that involve the pixel of interest i. Therefore, one takes into account all nine a_k, b_k's to predict $\widehat{y}_i$. The simple strategy suggested by He et al. [1] is to average them as follows:

$$\hat{a} = \frac{1}{|\omega|}\sum_{k=1}^{|\omega|}\hat{a}_k, \hat{b} = \frac{1}{|\omega|}\sum_{k=1}^{|\omega|}\hat{b}_k. \tag{12}$$

As such, the resulting prediction of $\widehat{Y}$ given the outcome $Z = z_i$ is

$$\begin{aligned}\hat{y}_i &= \hat{a} z_i + \hat{b} = \frac{1}{|\omega|}\sum_{k=1}^{|\omega|}(\hat{a}_k z_i + \hat{b}_k),\\ \hat{z}_i &= \hat{c} z_i + \hat{d} = \frac{1}{|\omega|}\sum_{k=1}^{|\omega|}(\hat{c}_k z_i + \hat{d}_k).\end{aligned} \tag{13}$$

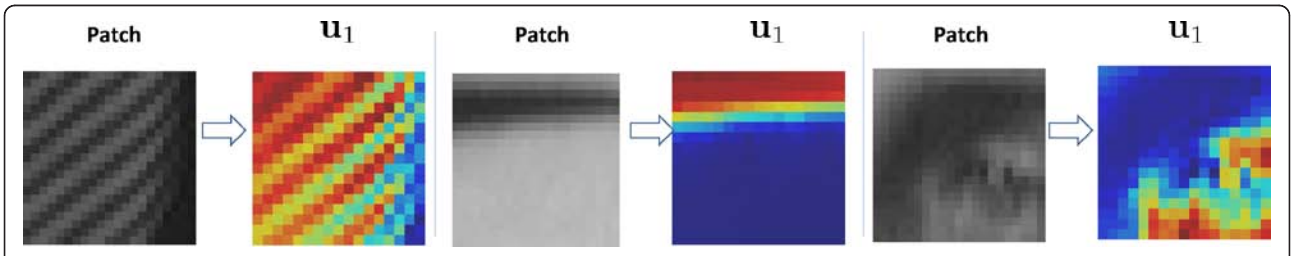

Figure 7 Examples of the first left eigenvector u in three patches. The vector was reshaped into an image for illustration purpose.

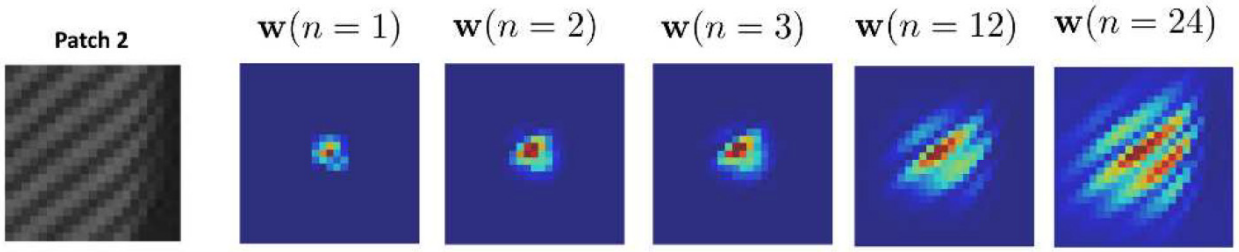

Figure 8 The guided filter kernel matrix. The guided filter kernel matrix **W** captures the underlying data structure, but powers of **W** provides even better structure by generating larger (but more sophisticated) kernel shapes. **w** is the (center) row vector of **W**. **w** was reshaped into an image for illustration purposes.

The idea of using these averaged coefficients $\hat{a}$, $\hat{b}$ is analogous to the simplest form of aggregating multiple local estimates from overlapped patches in image denoising and super-resolution literature [19]. The aggregation helps the filter output look locally smooth and contain fewer artifacts.[i] Recall that $\widehat{y_i}$ and $\hat{z}_i - z_i$ correspond to the base layer and the detail layer, respectively. The effect of the regularization parameters ε_1 and ε_2 is quite the opposite in each case in the sense that the higher ε_2 is, the more detail through $\hat{z}_i - z_i$ can be obtained; whereas the lower ε_1 ensures that the image content in $\widehat{Y}$ is not over-smoothed.

These local linear models work well when the window size p is small and the underlying data have a simple pattern. However, the linear models are too simple to deal effectively with more complicated structures, and thus there is a need to use larger window sizes. As we alluded to earlier, the estimation of these linear coefficients in an iterative fashion can deal well with more complex behavior of the image content. More

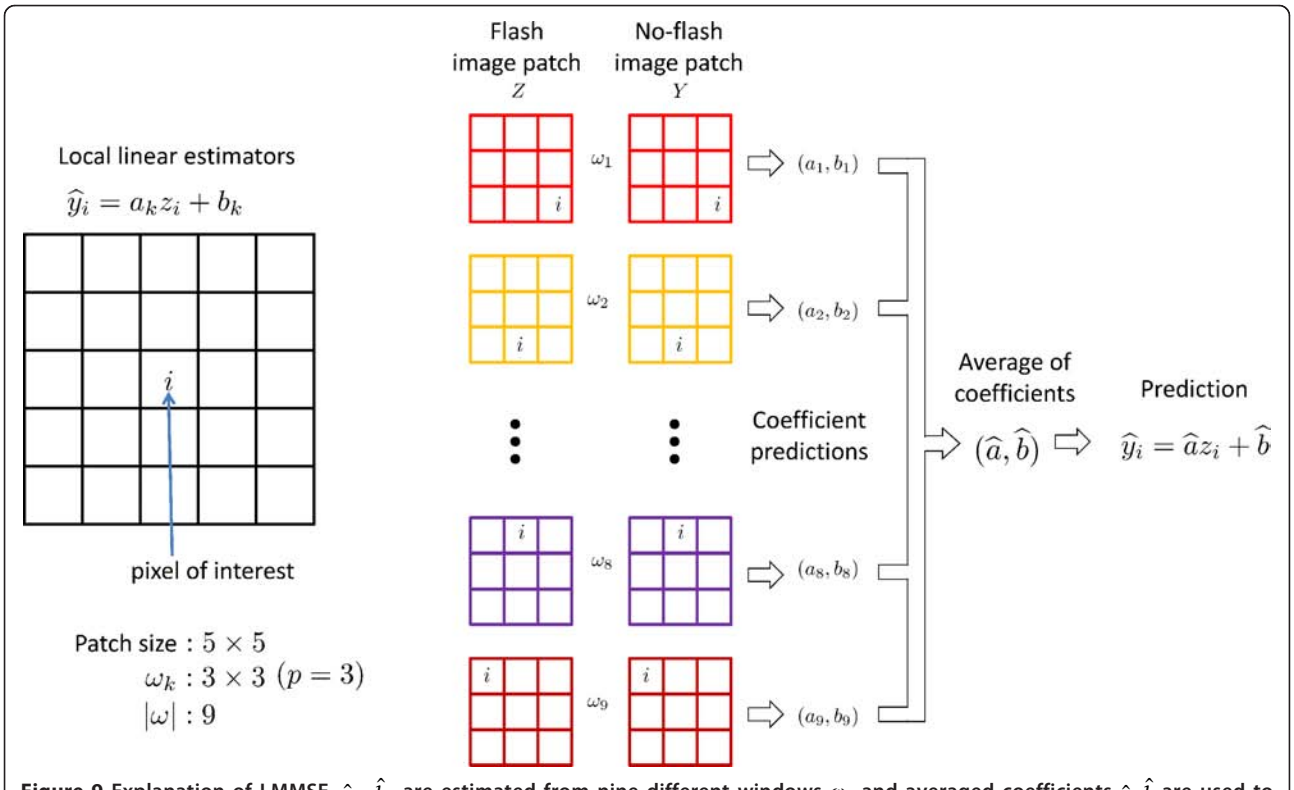

Figure 9 Explanation of LMMSE. $\hat{a}_k, \hat{b}_k$ **are estimated from nine different windows** ω_k **and averaged coefficients** $\hat{a}, \hat{b}$ **are used to predict** $\hat{y}_i$. This figure is better viewed in color.

specifically, by initializing $\hat{x}_{i,0} = y_i$, Equation 3 can be updated as follows

$$\begin{aligned}\hat{x}_{i,n} &= G(\hat{x}_{i,n-1},\ z_i) + \tau_n(z_i - \hat{z}_i),\\ &= (a_n - \tau_n c + \tau_n)z_i + b_n - \tau_n d,\\ &= \alpha_n z_i + \beta_n,\end{aligned} \tag{14}$$

where n is the iteration number and $\tau_n > 0$ is set to be a monotonically decaying function[k] of n such that $\sum_{n=1}^{\infty} \tau_n$ converges. Figure 3 shows an example to illustrate that the resulting coefficients at the 20th iteration predict the underlying data better than α_1, β_1 do. Similarly, $\widehat{X}_{20}$ improves upon $\widehat{X}_1$ as shown in Figure 4. This iteration is closely related to *diffusion* and *residual iteration* which are two important methods [18] which we describe briefly below, and with more detail in Appendix.

Recall that Equation 14 can also be written in matrix form as done in Section 3:

$$\hat{\mathbf{x}}_n = \underbrace{\mathbf{W}\hat{\mathbf{x}}_{n-1}}_{\text{base layer}} + \tau_n \underbrace{(\mathbf{z} - \mathbf{W}_d\mathbf{z})}_{\text{detail layer}}, \tag{15}$$

where $\mathbf{W}$ and $\mathbf{W}_d$ are guided filter kernel matrices composed of the guided filter kernels W and W_d respectively.[l] Explicitly writing the iterations, we observe

$$\begin{aligned}&\hat{\mathbf{x}}_0 = \mathbf{y}\\ &\hat{\mathbf{x}}_1 = \mathbf{W}\mathbf{y} + \tau_1(\mathbf{I} - \mathbf{W}_d)\mathbf{z},\\ &\hat{\mathbf{x}}_2 = \mathbf{W}\hat{\mathbf{x}}_1 + \tau_2(\mathbf{I} - \mathbf{W}_d)\mathbf{z} = \mathbf{W}^2\mathbf{y} + (\tau_1\mathbf{W} + \tau_2\mathbf{I})(\mathbf{I} - \mathbf{W}_d)\mathbf{z},\\ &\quad\vdots\\ &\hat{\mathbf{x}}_n = \mathbf{W}\hat{\mathbf{x}}_{n-1} + \tau_n(\mathbf{I} - \mathbf{W}_d)\mathbf{z} = \mathbf{W}^n\mathbf{y} + (\tau_1\mathbf{W}^{n-1} + \tau_2\mathbf{W}^{n-2} + \cdots + \tau_n\mathbf{I})(\mathbf{I} - \mathbf{W}_d)\mathbf{z},\\ &\quad = \underbrace{\mathbf{W}^n\mathbf{y}}_{\text{diffusion}} + \underbrace{P_n(\mathbf{W})(\mathbf{I} - \mathbf{W}_d)\mathbf{z}}_{\text{residual iteration}} = \hat{\mathbf{y}}_n + \hat{\mathbf{z}}_n,\end{aligned} \tag{16}$$

where P_n is a polynomial function of $\mathbf{W}$. The block-diagram in Figure 2 can be redrawn in terms of the matrix formulation as shown in Figure 10. The first term $\hat{\mathbf{y}}_n$ in Equation 16 is called the diffusion process that enhances SNR. The net effect of each application of $\mathbf{W}$ is essentially a step of anisotropic diffusion [20]. Note that this diffusion is applied to the no-flash image $\mathbf{y}$ which has a low SNR. On the other hand, the second term $\hat{\mathbf{z}}_n$ is connected with the idea of residual iteration [21]. The key idea behind this iteration is to filter the residual signals[m] to extract detail. We refer the reader to Appendix B and [18] for more detail. By effectively combining the diffusion and residual iteration [as in (16)], we can achieve the goal of flash/no-flash pair enhancement which is to generate an image somewhere between the flash image $\mathbf{z}$ and the no-flash image $\mathbf{y}$, but of better quality than both.[n]

5 Experimental results

In this section, we apply the proposed approach to flash/no-flash image pairs for denoising and deblurring. We convert images Z and Y from RGB color space to CIE Lab, and perform iterative guided filtering separately in each resulting channel. The final result is converted back to RGB space for display. We used the implementation of the guided filter [1] from the author's website.[o] All figures in this section are best viewed in color.[p]

5.1 Flash/no-flash denoising

5.1.1 Visible flash [2]

We show experimental results on a couple of flash/no-flash image pairs where no-flash images suffer from noise[q]. We compare our results with the method based on the joint bilateral filter [2] in Figures 11, 12, and 13. Our proposed method effectively denoised the no-flash image while transferring the fine detail of the flash image and maintaining the ambient lighting of the no-flash image. We point out that the proposed iterative application of the guided filtering in terms of diffusion and residual iteration yielded much better results than one application of either the joint bilateral filtering [2] or the guided filter [1].

5.1.2 Dark flash [7]

In this section, we use the *dark flash* method proposed in [7]. Let us call the dark flash image Z. Dark flash may introduce shadows and specularities in images, which affect the results of both the denoising and detail transfer. We detect those regions using the same methods proposed by [2]. Shadows are detected by finding the regions where $|Z - Y|$ is small, and specularities are found by detecting saturated pixels in Z. After combining the shadow and specularities mask, we blur it using a Gaussian filter to feather the boundaries. By using the resulting mask, the output $\widehat{X}_n$ at each iteration is alpha-blended with a low-pass filter version of Y as similarly done in [2,7]. In order to realize ambient lighting conditions, we applied the same mapping function to the final output as in [7]. Figures 14, 15, 16, 17, 18, and 19 show that our results yield better detail with less color artifacts than the results of [7].

5.2 Flash/no-flash deblurring

Motion blur due to camera shake is an annoying yet common problem in low-light photography. Our proposed method can also be applied to flash/no-flash deblurring[r]. Here, we show experimental results on a couple of flash/no-flash image pairs where no-flash images suffer from mild noise and strong motion blur. We compare our method with Zhuo et al. [5]. As shown in Figures 20, 21, 22, 23, and 24, our method outperforms the method by [5], obtaining much finer details with better color contrast even though our method does not estimate a blur kernel at all. The results by Zhuo et al. [5] tend to be somewhat blurry and distort the ambient lighting of the real scene. We point out that we only use a single blurred image in

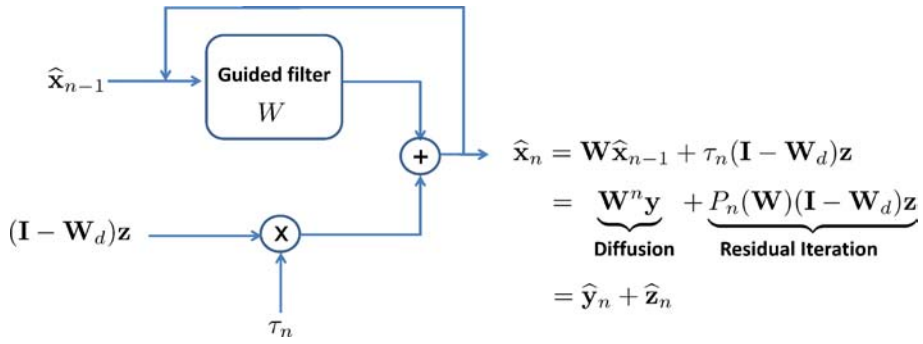

Figure 10 Diagram of the proposed iterative approach in matrix form. Note that the iteration can be divided into two parts: diffusion and residual iteration process.

Figure 24 while Zhuo et al. [5] used two blurred images and one flash image.

6 Summary and future work

The guided filter has proved to be more effective than the joint bilateral filter in several applications. Yet we have shown that it can be improved significantly more still. We analyzed the spectral behavior of the guided filter kernel using a matrix formulation and improved its performance by applying it iteratively. Iterations of the proposed method consist of a combination of diffusion and residual iteration. We demonstrated that the

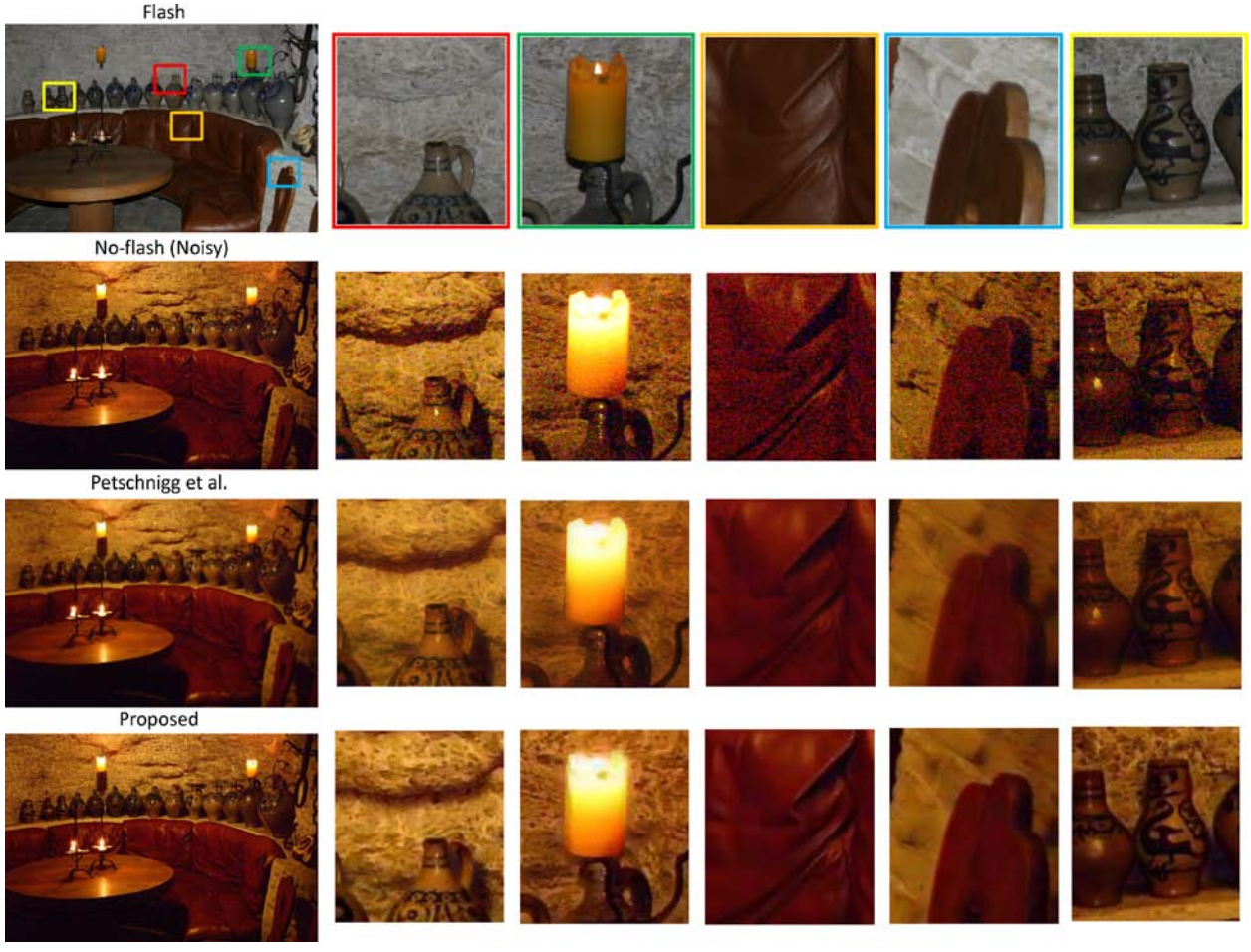

Figure 11 Flash/no-flash denoising example compared to the state of the art method [2]. The iteration *n* for this example is 10.

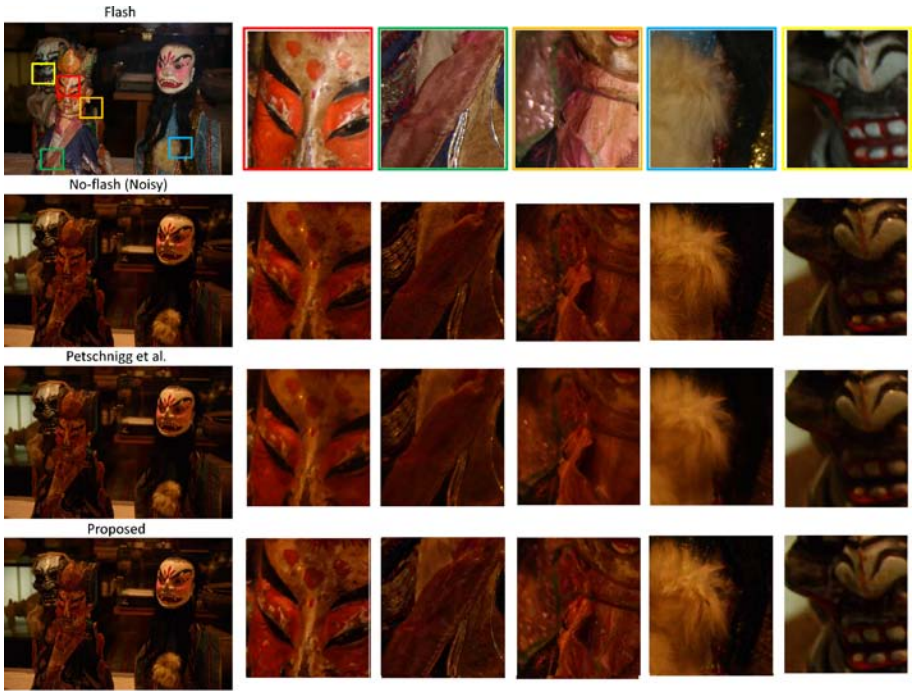

Figure 12 Flash/no-flash denoising example compared to the state of the art method [2]. The iteration *n* for this example is 10.

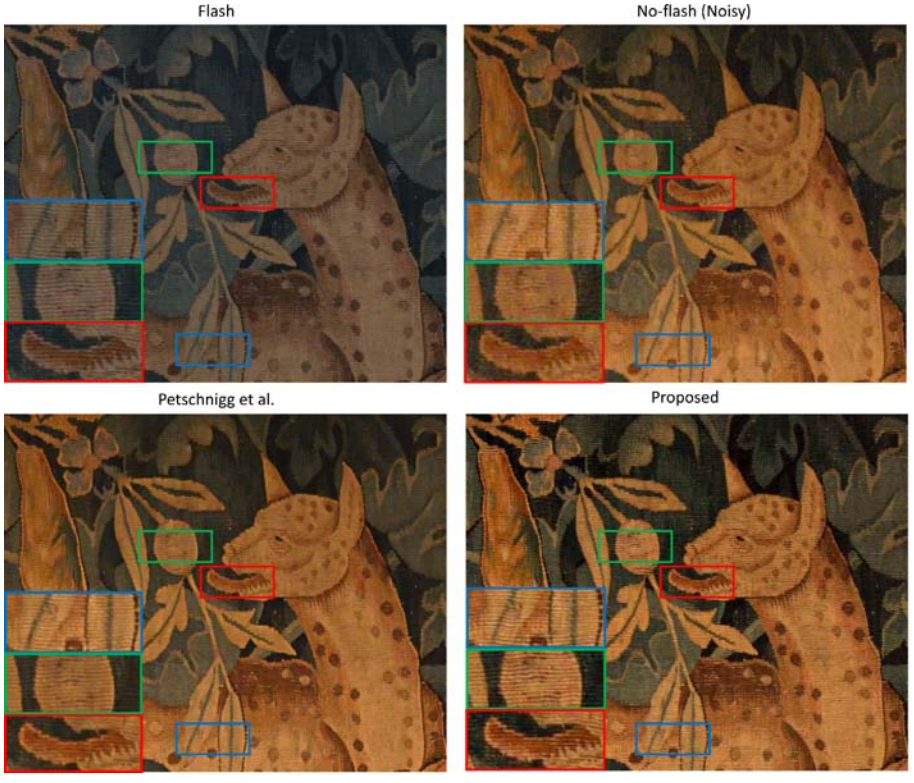

Figure 13 Flash/no-flash denoising example compared to the state of the art method [2]. The iteration *n* for this example is 2.

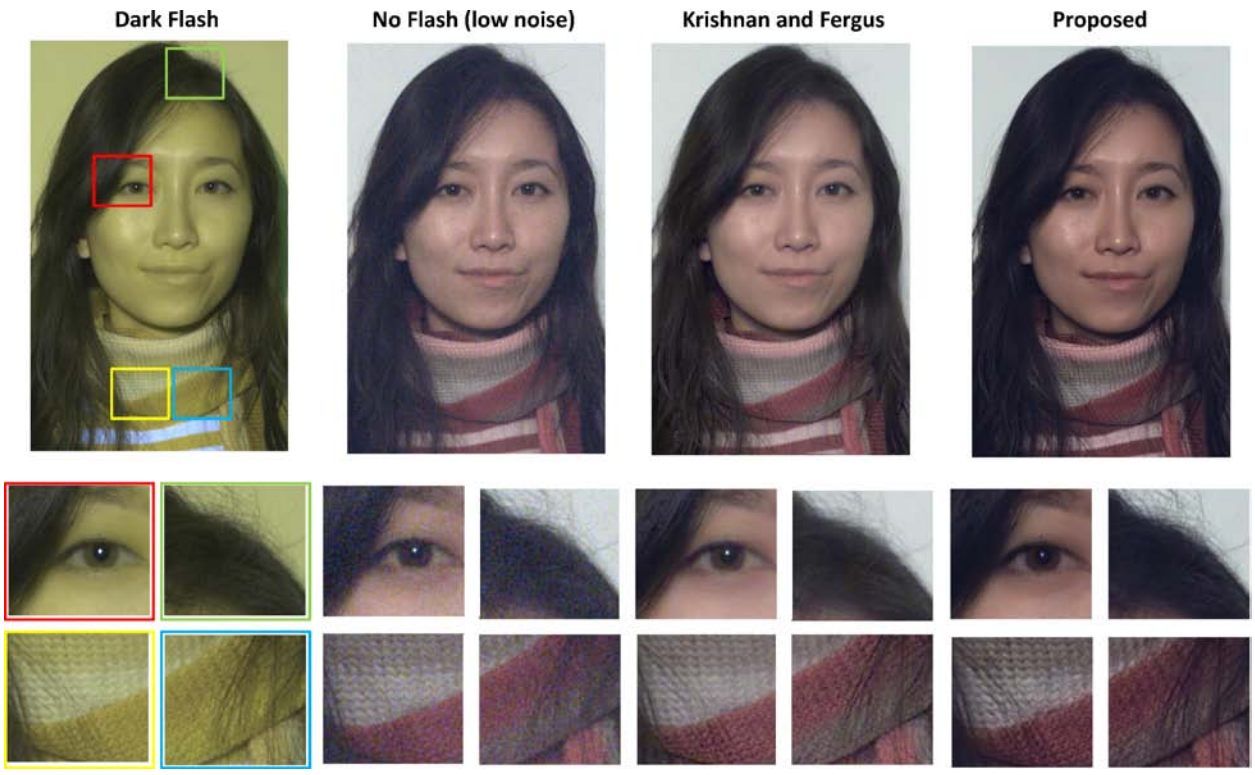

Figure 14 Dark flash/no-flash (low noise) denoising example compared to the state of the art method [7]. The iteration *n* for this example is 10.

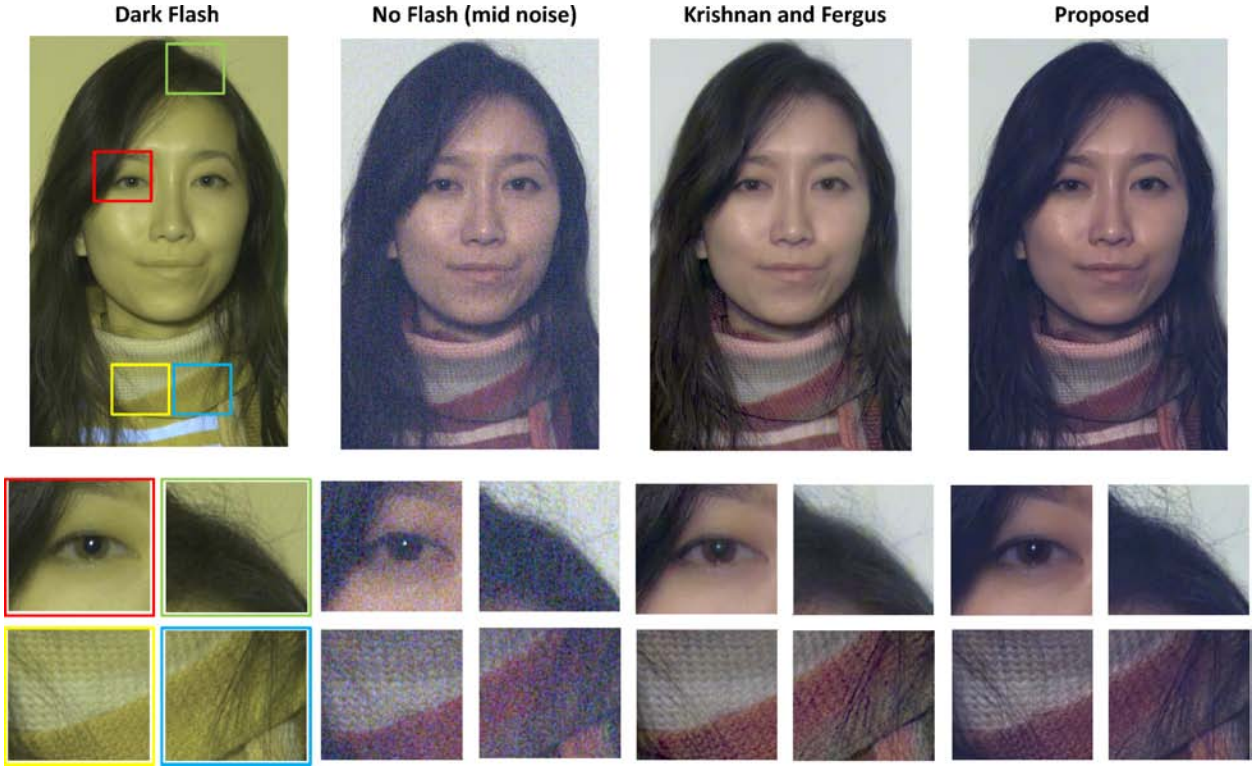

Figure 15 Dark flash/no-flash (mid noise) denoising example compared to the state of the art method [7]. The iteration *n* for this example is 10.

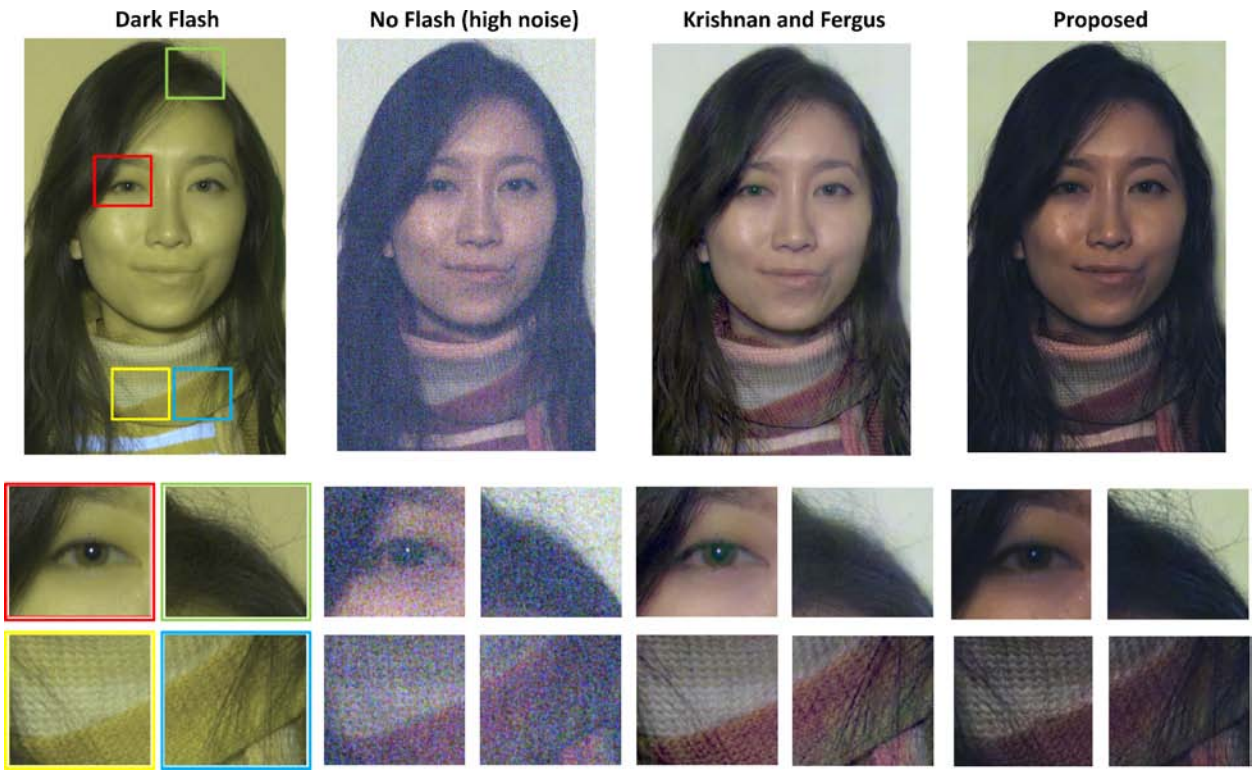

Figure 16 Dark flash/no-flash (high noise) denoising example compared to the state of the art method [7]. The iteration *n* for this example is 10.

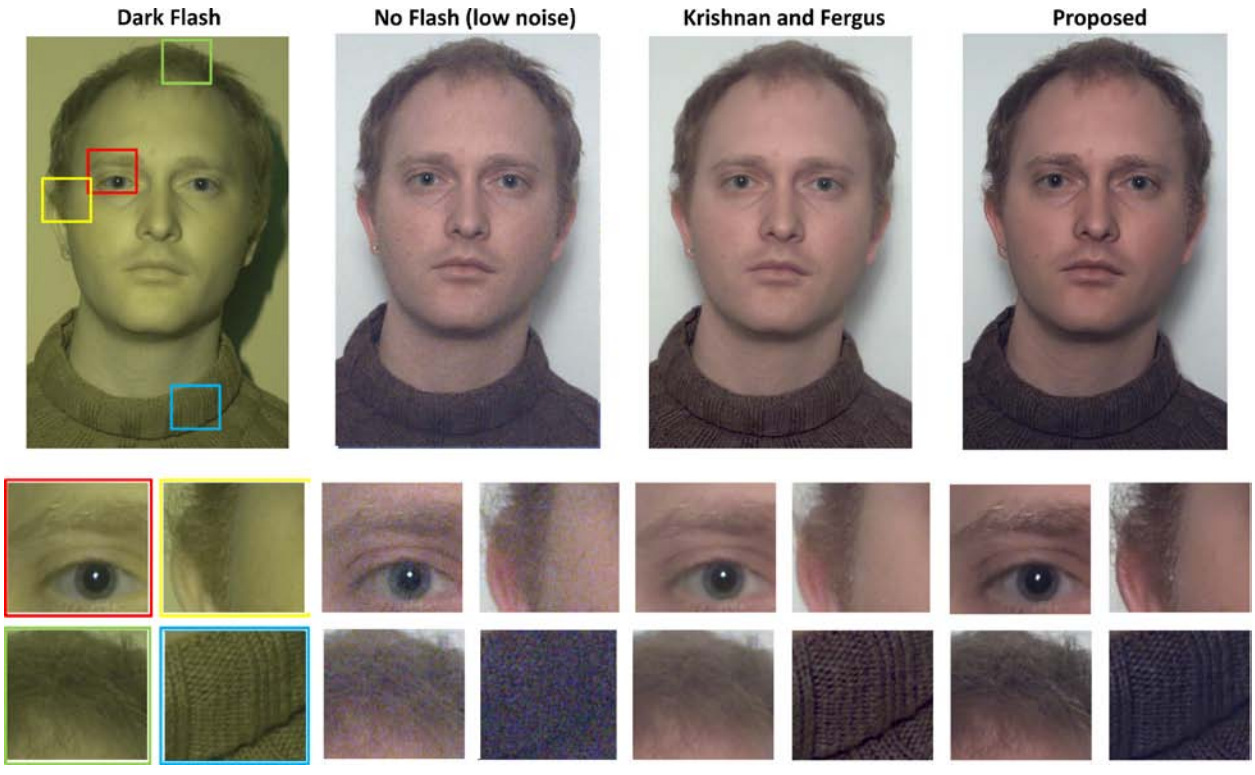

Figure 17 Dark flash/no-flash (low noise) denoising example compared to the state of the art method [7]. The iteration *n* for this example is 10.

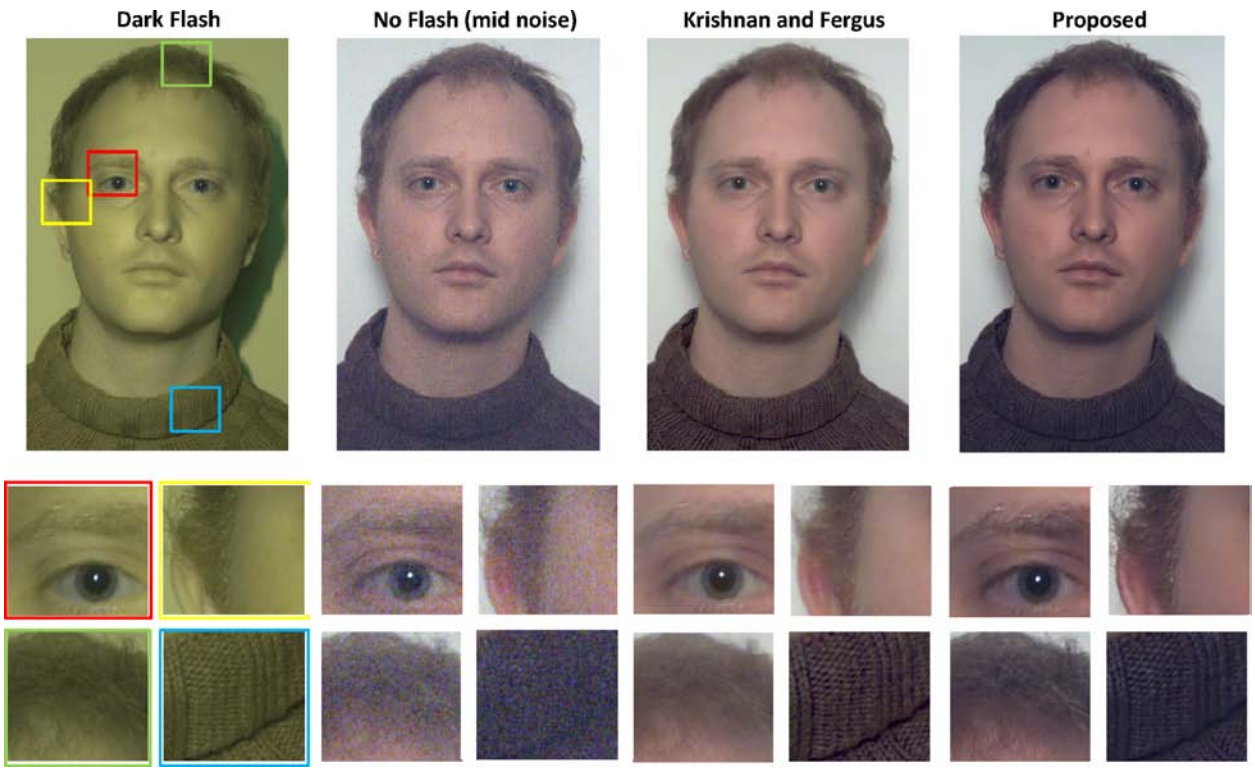

Figure 18 Dark flash/no-flash (mid noise) denoising example compared to the state of the art method [7]. The iteration *n* for this example is 10.

Figure 19 Dark flash/no-flash (high noise) denoising example compared to the state-of-the-art method [7]. The iteration *n* for this example is 10.

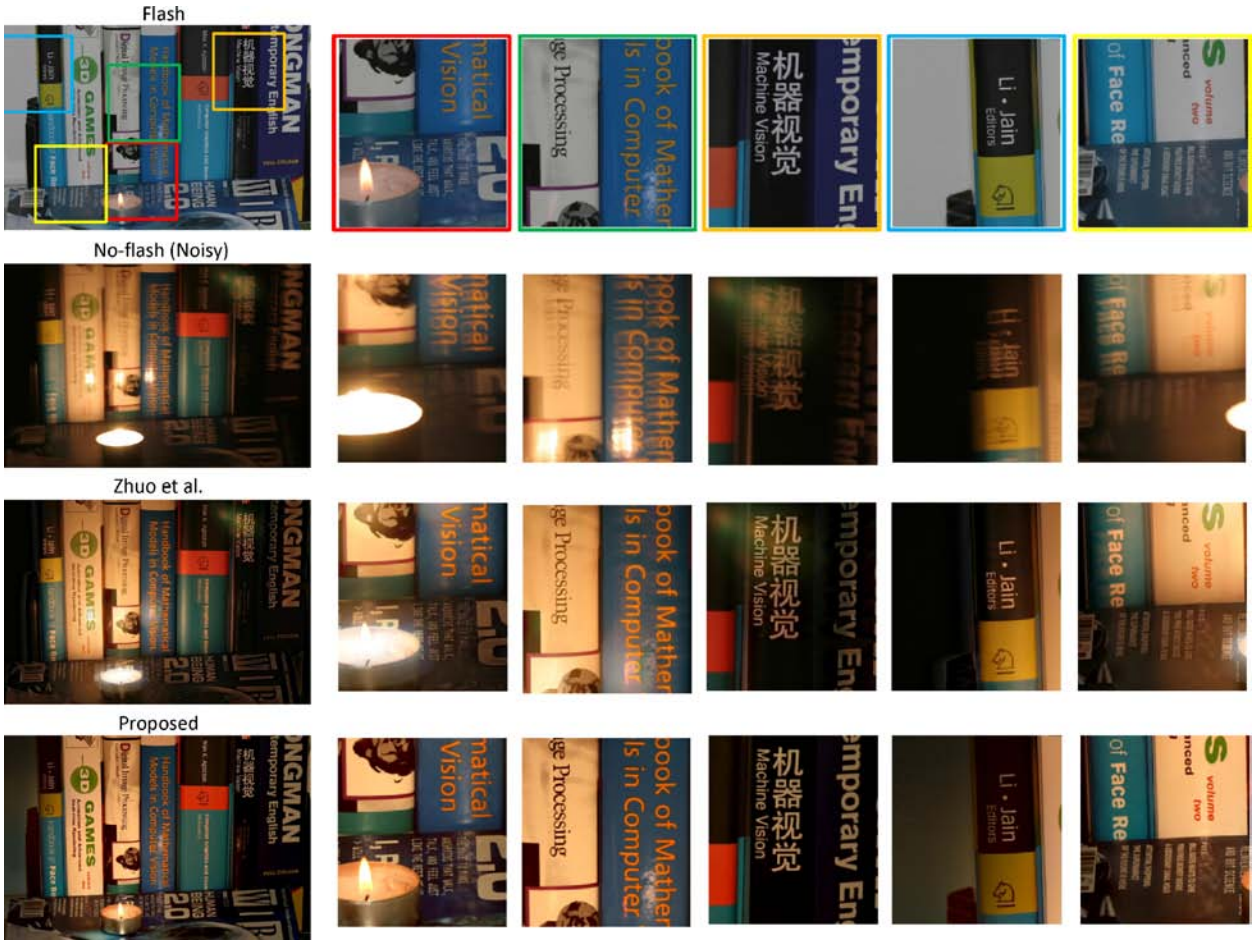

Figure 20 Flash/no-flash deblurring example compared to the state-of-the-art method [5]. The iteration n for this example is 20.

proposed approach yields outputs that not only preserve fine details of the flash image, but also the ambient lighting of the no-flash image. The proposed method outperforms state-of-the-art methods for flash/no-flash image denoising *and* deblurring. It would be interesting to see if the performance of other nonparametric filer kernels such as bilateral filters and locally adaptive regression kernels [15] can be further improved in our iterative framework. It is also worthwhile to explore several other applications such as joint upsampling [22], image matting [23], mesh smoothing [24,25], and specular highlight removal [26] where the proposed approach can be employed.

Appendix

Positive definite and symmetric row-stochastic approximation of W

In this section, we describe how we approximate $\mathbf{W}$ with a symmetric, positive definite, and row-stochastic matrix. First, as we mentioned earlier, the matrix $\mathbf{W}$ can contain negative values as shown in Figure 3. We employ the Taylor series approximation ($\exp(t) \approx 1 + t$) to ensure that $\mathbf{W}$ has both positive elements, and is positive-definite. To be more concrete, consider a simple example; namely, a local patch of size 5 × 5 as follows:

$$Z = \begin{Bmatrix} z_1 & z_6 & z_{11} & z_{16} & z_{21} \\ z_2 & z_7 & z_{12} & z_{17} & z_{22} \\ z_3 & z_8 & z_{13} & z_{18} & z_{23} \\ z_4 & z_9 & z_{14} & z_{19} & z_{24} \\ z_5 & z_{10} & z_{15} & z_{20} & z_{25} \end{Bmatrix} \tag{17}$$

In this case, we can have up to 9 ω_k of size $|\omega| = 3 \times 3$. Let $M(z_i, z_j) = \sum_{k:(i,j\in\omega_k)} \frac{(z_i - E[Z]_k)(z_j - E[Z]_k)}{\text{var}(Z)_{k+\varepsilon}}$ in Equation 6. Then, W centered at the index 13 can be written and approximated as follows:

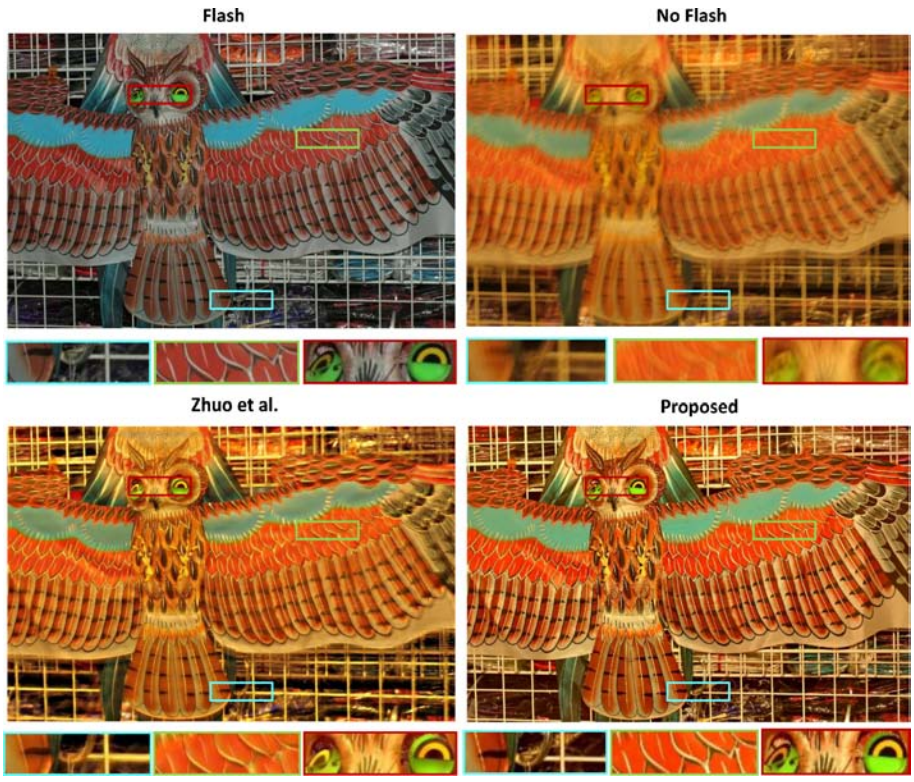

Figure 21 Flash/no-flash deblurring example compared to the state-of-the-art method [5]. The iteration *n* for this example is 20.

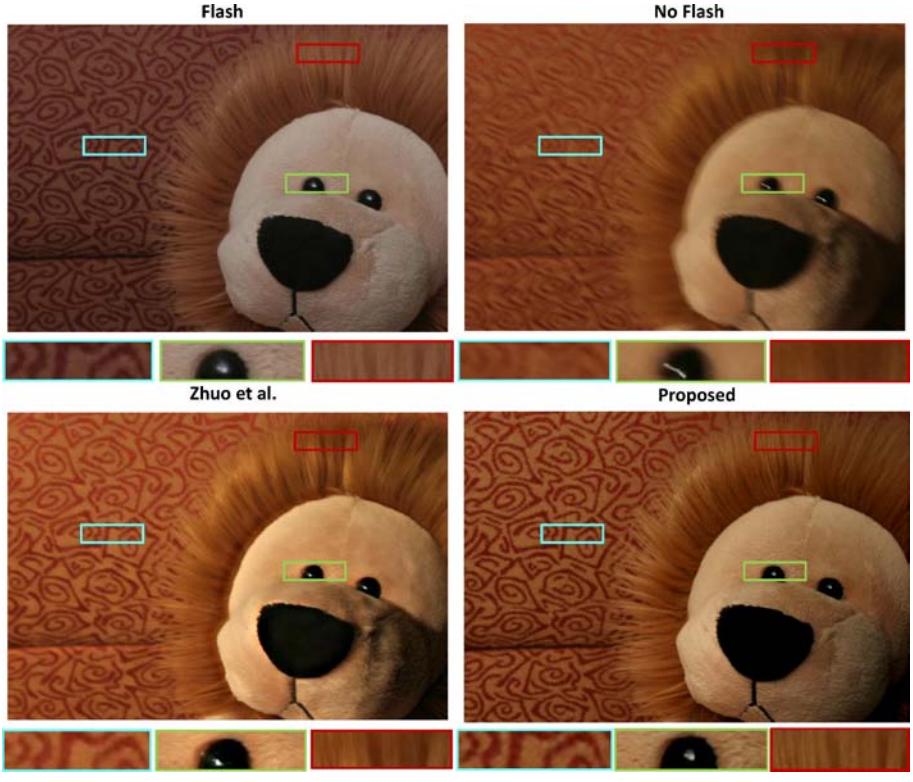

Figure 22 Flash/no-flash deblurring example compared to the state-of-the-art method [5]. The iteration *n* for this example is 20.

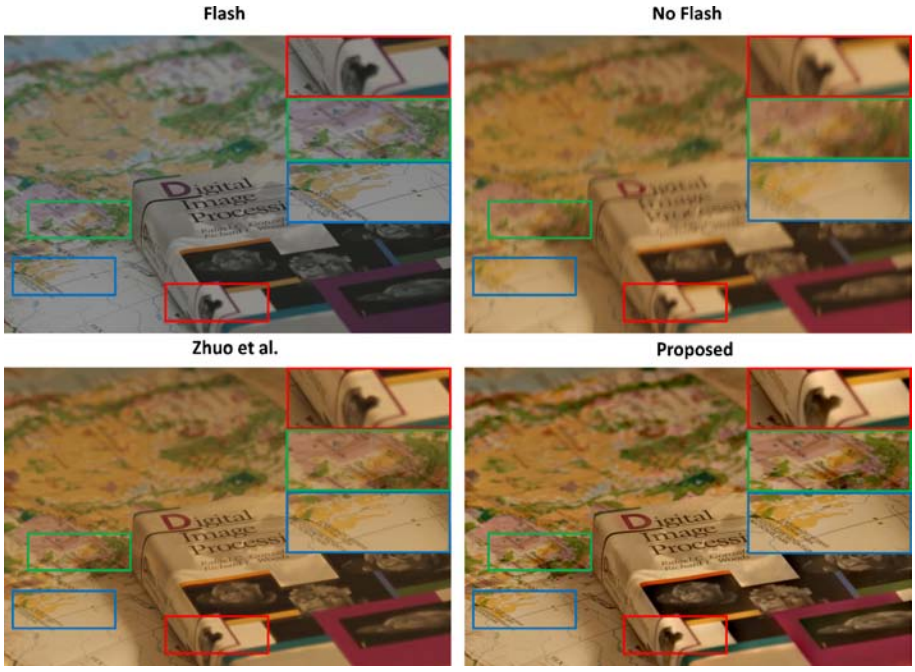

Figure 23 Flash/no-flash deblurring example compared to the state-of-the-art method [5]. The iteration *n* for this example is 5.

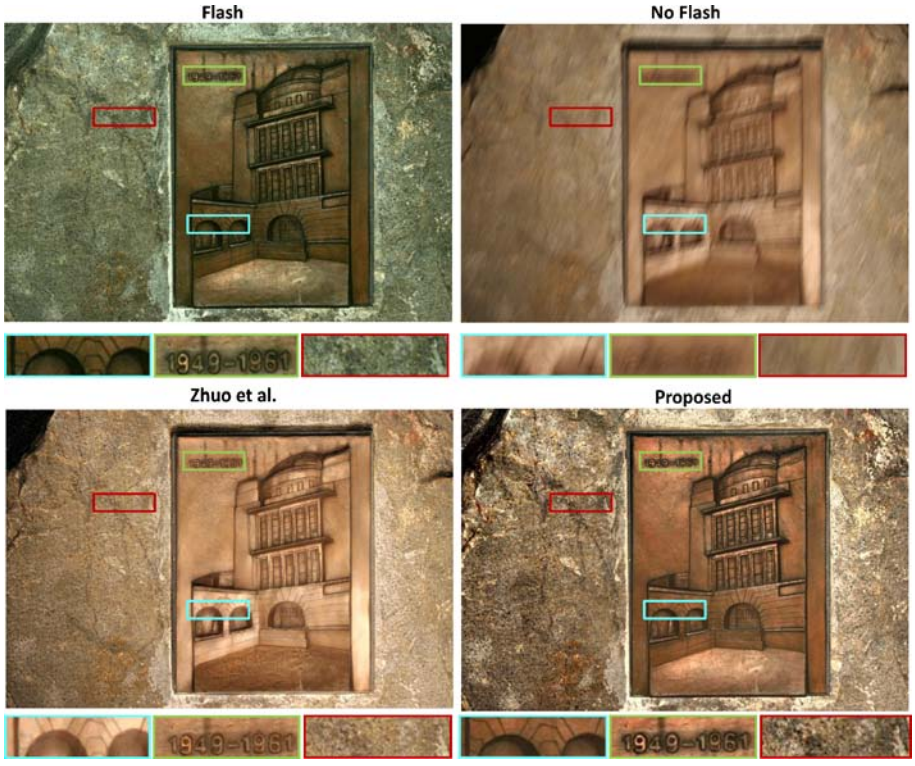

Figure 24 Flash/no-flash deblurring example compared to the state-of-the-art method [5]. The iteration *n* for this example is 20.

$$W_{13,j} = \frac{1}{81}\begin{bmatrix} 1+M(z_{13},z_1)2(1+\frac{1}{2}M(z_{13},z_{16}))3(1+\frac{1}{3}M(z_{13},z_{11}))2(1+\frac{1}{2}M(z_{13},z_{16}))1+M(z_{13},z_{21}) \\ 2(1+\frac{1}{2}M(z_{13},z_2))4(1+\frac{1}{4}M(z_{13},z_7))6(1+\frac{1}{6}M(z_{13},z_{12}))4(1+\frac{1}{4}M(z_{13},z_{17}))2(1+\frac{1}{2}M(z_{13},z_{22})) \\ 3(1+\frac{1}{3}M(z_{13},z_3))6(1+\frac{1}{6}M(z_{13},z_8))9(1+\frac{1}{9}M(z_{13},z_{13}))6(1+\frac{1}{6}M(z_{13},z_{18}))3(1+\frac{1}{3}M(z_{13},z_{23})) \\ 2(1+\frac{1}{2}M(z_{13},z_4))4(1+\frac{1}{4}M(z_{13},z_9))6(1+\frac{1}{6}M(z_{13},z_{14}))4(1+\frac{1}{4}M(z_{13},z_{19}))2(1+\frac{1}{2}M(z_{13},z_{24})) \\ 1+M(z_{13},z_5)2(1+\frac{1}{2}M(z_{13},z_{10}))3(1+\frac{1}{3}M(z_{13},z_{15}))2(1+\frac{1}{2}M(z_{13},z_{20}))1+M(z_{13},z_{25}), \end{bmatrix}$$

$$\approx \frac{1}{81}\begin{bmatrix} \exp(M(z_{13},z_1))2\exp(\frac{1}{2}M(z_{13},z_6))3\exp(\frac{1}{3}M(z_{13},z_{11}))2\exp(\frac{1}{2}M(z_{13},z_{16}))\exp(M(z_{13},z_{21})) \\ 2\exp(\frac{1}{2}M(z_{13},z_2))4\exp(\frac{1}{4}M(z_{13},z_7))6\exp(\frac{1}{6}M(z_{13},z_{12}))4\exp(\frac{1}{4}M(z_{13},z_{17}))2\exp(\frac{1}{2}M(z_{13},z_{22})) \\ 3\exp(\frac{1}{3}M(z_{13},z_3))6\exp(\frac{1}{6}M(z_{13},z_8))9\exp(\frac{1}{9}M(z_{13},z_{13}))6\exp(\frac{1}{6}M(z_{13},z_{18}))3\exp(\frac{1}{3}M(z_{13},z_{23})) \\ 2\exp(\frac{1}{2}M(z_{13},z_4))4\exp(\frac{1}{4}M(z_{13},z_9))6\exp(\frac{1}{6}M(z_{13},z_{14}))4\exp(\frac{1}{4}M(z_{13},z_{19}))2\exp(\frac{1}{2}M(z_{13},z_{24})) \\ \exp(M(z_{13},z_5))2\exp(\frac{1}{2}M(z_{13},z_{10}))3\exp(\frac{1}{3}M(z_{13},z_{15}))2\exp(\frac{1}{2}M(z_{13},z_{15}))\exp(M(z_{13},z_{25})) \end{bmatrix},$$

Next, we convert the matrix $\mathbf{W}$ (composed of strictly positive elements now) to a doubly-stochastic, symmetric, positive definite matrix as again done in [18]. The algorithm we use to effect this approximation is due to Sinkhorn [27,28], who proved that given a matrix with strictly positive elements, there exist diagonal matrices $\mathbf{R} = \mathrm{diag}(\mathbf{r})$ and $\mathbf{C} = \mathrm{diag}(\mathbf{c})$ such that

$$\widehat{\mathbf{W}} = \mathbf{R}\,\mathbf{W}\,\mathbf{C}$$

is doubly stochastic. That is,

$$\widehat{\mathbf{W}}\mathbf{1}_N = \mathbf{1}_N \text{ and } \mathbf{1}_N^T\widehat{\mathbf{W}} = \mathbf{1}_N^T \tag{18}$$

Furthermore, the vectors $\mathbf{r}$ and $\mathbf{c}$ are unique to within a scalar (i.e., $\alpha\,\mathbf{r}$, $\mathbf{c}/\alpha$.) Sinkhorn's algorithm for obtaining $\mathbf{r}$ and $\mathbf{c}$ in effect involves repeated normalization of the rows and columns (see Algorithm 1 for details) so that they sum to one, and is provably convergent and optimal in the cross-entropy sense [29].

Algorithm 1 Algorithm for scaling a matrix A to a nearby doubly-stochastic matrix $\hat{\mathbf{A}}$

Given a matrix $\mathbf{A}$, let (N, N) be size($\mathbf{A}$) and initialize $\mathbf{r}$ = ones(N, 1);
for $k = 1 : iter$;
$\mathbf{c} = 1./(\mathbf{A}^T\,\mathbf{r})$;
$\mathbf{r} = 1./(\mathbf{A}\,\mathbf{c})$;
end
$\mathbf{C} = \mathrm{diag}(\mathbf{c})$; $\mathbf{R} = \mathrm{diag}(\mathbf{r})$;
$\widehat{\mathbf{A}} = \mathbf{RAC}$

Diffusion and residual iteration

Diffusion

Here, we describe how multiple direct applications of the filter given by $\mathbf{W}$ is in effect equivalent to a nonlinear anisotropic *diffusion* process [20,30]. We define $\hat{\mathbf{y}}_0 = \mathbf{y}$, and

$$\hat{\mathbf{y}}_n = \mathbf{W}\hat{\mathbf{y}}_{n-1} = \mathbf{W}^n\mathbf{y}. \tag{19}$$

From the iteration (19) we have

$$\hat{\mathbf{y}}_n = \mathbf{W}\hat{\mathbf{y}}_{n-1}, \tag{20}$$

$$= \hat{\mathbf{y}}_{n-1} - \hat{\mathbf{y}}_{n-1} + \mathbf{W}\hat{\mathbf{y}}_{n-1}, \tag{21}$$

$$= \hat{\mathbf{y}}_{n-1} + (\mathbf{W} - \mathbf{I})\hat{\mathbf{y}}_{n-1}, \tag{22}$$

which we can rewrite as

$$\hat{\mathbf{y}}_n - \hat{\mathbf{y}}_{n-1} = (\mathbf{W} - \mathbf{I})\hat{\mathbf{y}}_{n-1} \longleftrightarrow \frac{\partial\bar{\mathbf{y}}(t)}{\partial t} = \nabla^2\bar{\mathbf{y}}(t) \quad \text{Diffusion Equation} \tag{23}$$

where $\bar{\mathbf{y}}$ is a scaled version of by, and therefore the left-hand side of the above is a discretization of the derivative operator $\frac{\partial\bar{y}(t)}{\partial t}$, and as detailed in [18], $\mathbf{W}$ - $\mathbf{I}$ is effectively the nonlinear Laplacian operator corresponding to the kernel in (6).

Residual iteration

An alternative to repeated applications of the filter $\mathbf{W}$ is to consider the *residual* signals, defined as the difference between the estimated signal and the measured signal. This results in a variation of the diffusion estimator which uses the residuals as an additional forcing term. The net result is a type of reaction-diffusion process [31]. In statistics, the use of the residuals in improving estimates has a rather long history, dating at least back to the study of Tukey [21] who termed the idea "twicing". More recently, the idea has been suggested in the applied mathematics community under the rubric of *Bregman* iterations [32], and in the machine learning and statistics literature [33] as *L_2-boosting*.

Formally, the residuals are defined as the difference between the estimated signal and the measured signal: $\mathbf{r}_n = \mathbf{z} - \mathbf{z}_{n-1}$, where here we define the initialization[s] $\hat{\mathbf{z}}_0 = \mathbf{W}\mathbf{z}$. With this definition, we write the iterated estimates as

$$\hat{\mathbf{z}}_n = \hat{\mathbf{z}}_{n-1} + \mathbf{W}\mathbf{r}_n = \hat{\mathbf{z}}_{n-1} + \mathbf{W}(\mathbf{z} - \hat{\mathbf{z}}_{n-1}). \tag{24}$$

Explicitly writing the iterations, we observe:

$$\begin{aligned} \hat{\mathbf{z}}_1 &= \hat{\mathbf{z}}_0 + \mathbf{W}(\mathbf{z} - \hat{\mathbf{z}}_0) = \mathbf{W}\mathbf{z} + \mathbf{W}(\mathbf{z} - \mathbf{W}\mathbf{z}) = (2\mathbf{W} - \mathbf{W}^2)\mathbf{z}, \\ \hat{\mathbf{z}}_2 &= \hat{\mathbf{z}}_1 + \mathbf{W}(\mathbf{z} - \hat{\mathbf{z}}_1) = (\mathbf{W}^3 - 3\mathbf{W}^2 + 3\mathbf{W})\mathbf{z}, \\ &\vdots \\ \hat{\mathbf{z}}_n &= F_n(\mathbf{W})\mathbf{z}, \end{aligned} \tag{25}$$

where F_n is a polynomial function of $\mathbf{W}$ of order $n + 1$. The first iterate $\hat{\mathbf{z}}_1$ is precisely the "twicing" estimate of Tukey [21].

Convergence of the proposed iterative estimator

Recall the iterations:

$$\hat{\mathbf{x}}_n = \mathbf{W}^n\mathbf{y} + \underbrace{(\tau_1\mathbf{W}^{n-1} + \tau_2\mathbf{W}^{n-2} + \cdots + \tau_n\mathbf{I})}_{P_n(\mathbf{W})}(\mathbf{I} - \mathbf{W}_d)\mathbf{z}. \tag{26}$$

where P_n is a polynomial function of $\mathbf{W}$. The first (direct diffusion) term in the iteration is clearly stable and convergent as described in (9). Hence, we need to show the stability of the second part. To do this, we note that the spectrum of P_n can be written, and bounded in terms of the eigenvalues λ_i of $\mathbf{W}$ as follows:

$$\begin{aligned} \lambda_i(P_n) &= \mathrm{eig}(P_n) = \tau_1\lambda_i^{n-1} + \tau_2\lambda_i^{n-2} + \cdots + \tau_n, \\ &\le \tau_1\lambda_1^{n-1} + \tau_2\lambda_1^{n-2} + \cdots + \tau_n = \tau_1 + \tau_2 + \tau_3 + \cdots + \tau_n \le c. \end{aligned} \tag{27}$$

where the inequality follows from the knowledge that $0 \leq \lambda_N \leq \ldots \lambda_3 \leq \lambda_2 < \lambda_1 = 1$. Furthermore, in Section 4 we defined τ_n to be a monotonically decreasing sequence such that $\sum_{n=1}^{\infty} \tau_n = c < \infty$. Hence, all eigenvalues $\lambda_i(P_n)$ are upper bounded by the constant c, independent of the number of iterations n, ensuring the stability of the iterative process.

End notes

[a]More detail is provided in Section 4. [b]The guided filter [1] reduces noise while preserving edges as bilateral filter [6] does. However, the guided filter outperforms the bilateral filter by avoiding the gradient reversal artifacts that may appear in such applications as detail enhancement, high dynamic range (HDR) compression, and flash/no-flash denoising. [c]http://users.soe.ucsc.edu/~milanfar/research/rokaf/.html/IGF/. [d] $\hat{y}$ in Equation 2 can be rewritten in terms of filtering and we refer the reader to a supplemental material http://personal.ie.cuhk.edu.hk/~hkm007/eccv10/eccv10supp.pdf for derivation. [e]Cross (or joint) bilateral filter [2,3] is defined in a similar way. [f] N is different from the window size p ($N \geq p$). [g] Note that k is used to clarify that the coefficients are estimated for the window ω_k. [h] Shan et al. [8] recently proposed a similar approach for high dynamic range compression. [i] It is worthwhile to note that we can benefit from more adaptive way of combining multiple estimates of coefficients, but this subject is not treated in this article. [k] We use $\tau_n = \frac{1}{n^2}$ throughout the all experiments. [l] Recall W in Equation 6. The difference between W and W_d lies in the parameter ε as follows ($\varepsilon_2 > \varepsilon_1$):

$$W(Z) = \frac{1}{|\omega|^2} \sum_{k:(i,j)\in\omega_k} \left(1 + \frac{(z_i - E[z]_k)(z_j - E[Z]_k)}{\mathbf{var}(Z)_k + \varepsilon_1}\right),$$
$$W_d(Z) = \frac{1}{|\omega|^2} \sum_{k:(i,j)\in\omega_k} \left(1 + \frac{(z_i - E[z]_k)(z_j - E[Z]_k)}{\mathbf{var}(Z)_k + \varepsilon_2}\right). \quad (28)$$

[m] This is generally defined as the difference between the estimated signal $\widehat{Z}$ and the measured signal Z, but in our context refers to the detail signal. [n] We refer the reader to Appendix C for proof of convergence of the proposed iterative estimator. [o] http://personal.ie.cuhk.edu.hk/~hkm007/. [p] We refer the reader to the project Website http://users.soe.ucsc.edu/~milanfar/research/rokaf/.html/IGF/. [q] The window size p for $\mathbf{W}_d$ and $\mathbf{W}$ was set to 21 and 5 respectively for all the denoising examples. [r] The window size p for $\mathbf{W}_d$ and $\mathbf{W}$ was set to 41 and 81, respectively, for the deblurring examples to deal with displacement between the flash and no-flash image. [s] Note that due to the use of residuals, this is a different initialization than the one used in the diffusion iterations.

Acknowledgements

We thank Dilip Krishnan for sharing the post-processing code [7] for dark-flash examples. This study was supported by the AFOSR Grant FA 9550-07-01-0365 and NSF Grant CCF-1016018. This study was done while the first author was at the University of California.

Author details

[1]Sharp Labs of America, Camas, WA 98683, USA [2]University of California-Santa Cruz, 1156 High Street, Santa Cruz, CA 95064, USA

Authors' contributions

HS carried out the design of iterative guided filtering and drafted the manuscript. PM participated in the design of iterative guided filtering and performed the statistical analysis. All authors read and approved the final manuscript.

Competing interests

The authors declare that they have no competing interests.

Received: 23 June 2011 Accepted: 6 January 2012
Published: 6 January 2012

References

1. K He, J Sun, X Tang, Guided image filtering, in *Proceedings of European Conference Computer Vision (ECCV)* (2010)
2. G Petschnigg, M Agrawala, H Hoppe, R Szeliski, M Cohen, K Toyama, Digital photography with flash and no-flash image pairs. ACM Trans Graph. **21**(3), 664–672 (2004)
3. E Eisemann, F Durand, Flash photography enhancement via intrinsic relighting. ACM Trans Graph. **21**(3), 673–678 (2004)
4. A Agrawal, R Raskar, S Nayar, Y Li, Removing photography artifacts using gradient projection and flash-exposure sampling. ACM Trans Graph. **24**, 828–835 (2005). doi:10.1145/1073204.1073269
5. S Zhuo, D Guo, T Sim, Robust flash deblurring. IEEE Conference on Computer Vison and Pattern Recognition (2010)
6. C Tomasi, R Manduchi, Bilateral Filtering for Gray and Color Images, in *Proceedings of the 1998 IEEE International Conference of Compute Vision Bombay, India*, pp. 836–846 (1998)
7. D Krishnan, R Fergus, Dark flash photography. ACM Trans Graph. **28**(4), 594–611 (2009)
8. Q Shan, J Jia, MS Brown, Globally optimized linear windowed tone-mapping. IEEE Trans Vis Comput Graph. **16**(4), 663–675 (2010)
9. R Fergus, B Singh, A Hertsmann, ST Roweis, WT Freeman, Removing camera shake from a single image. ACM Trans Graph (SIGGRAPH). **25**, 787–794 (2006). doi:10.1145/1141911.1141956
10. L Yuan, J Sun, L Quan, HY Shum, Progressive inter-scale and intra-scale non-blind image deconvolution. ACM Trans Graph. **27**(3), 1–10 (2008)
11. YW Tai, J Jia, CK Tang, Local color transfer via probabilistic segmentation by expectation-maximization. IEEE Conference on Computer Vison and Pattern Recognition (2005)
12. W Hasinoff, Variable-aperture photography, (PhD Thesis, Department of Computer Science, University of Toronto, 2008)
13. H Seo, P Milanfar, Computational photography using a pair of flash/no-flash images by iterative guided filtering. IEEE International Conference on Computer Vision (ICCV) (2011). Submitted
14. A Buades, B Coll, JM Morel, A review of image denoising algorithms, with a new one. Multi-scale Model Simulat (SIAM). **4**(2), 490–530 (2005)
15. H Takeda, S Farsiu, P Milanfar, Kernel regression for image processing and reconstruction. IEEE Trans Image Process. **16**(2), 349–366 (2007)
16. A Buades, B Coll, JM Morel, Nonlocal image and movie denoising. Int J Comput Vis. **76**(2), 123–139 (2008). doi:10.1007/s11263-007-0052-1
17. T Hofmann, B Scholkopf, AJ Smola, Kernel methods in machine learning. Ann Stat. **36**(3), 1171–1220 (2008). doi:10.1214/009053607000000677
18. P Milanfar, A tour of modern image processing. IEEE Signal Process Mag http://users.soe.ucsc.edu/~milanfar/publications/journal/ModernTour_FinalSubmission.pdf (2011)

19. M Protter, M Elad, H Takeda, P Milanfar, Generalizing the non-local-means to super-resolution reconstruction. IEEE Trans Image Process. **18**, 36–51 (2009)
20. P Perona, J Malik, Scale-space and edge detection using anistropic diffusion. IEEE Trans Pattern Anal Mach Intell. **12**(9), 629–639 (1990)
21. JW Tukey, *Exploratory Data Analysis* (Addison Wesley, Reading, MA, 1977)
22. J Kopf, MF Cohen, D Lischinski, M Uyttendaele, Joint bilateral upsampling. ACM Trans Graph. **26**(3), 96 (2007). doi:10.1145/1276377.1276497
23. K He, J Sun, X Tang, Fast matting using large kernel matting Laplacian matrices. IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2010)
24. S Fleishman, I Drori, D Cohen-Or, Bilateral mesh denoising. ACM Trans Graph. **22**(3), 950–953 (2003). doi:10.1145/882262.882368
25. T Jones, F Durand, M Desbrun, Non-iterative feature preserving mesh smoothing. ACM Trans Graph. **22**(3), 943–949 (2003). doi:10.1145/882262.882367
26. Q Yang, S Wang, N Ahuja, Real-time specular highlight removal using bilateral filtering. ECCV (2010)
27. PA Knight, The Sinkhorn-Knopp algorithm: convergence and applications. SIAM J Matrix Anal Appl. **30**, 261–275 (2008). doi:10.1137/060659624
28. R Sinkhorn, A relationship between arbitrary positive matrices and doubly stochastic matrices. Ann Math Stat. **35**(2), 876–879 (1964). doi:10.1214/aoms/1177703591
29. J Darroch, D Ratcliff, Generalized iterative scaling for log-linear models. Ann Math Stat. **43**, 1470–1480 (1972). doi:10.1214/aoms/1177692379
30. A Singer, Y Shkolinsky, B Nadler, Diffusion interpretation of nonlocal neighborhood filters for signal denoising. SIAM J Imaging Sci. **2**, 118–139 (2009). doi:10.1137/070712146
31. G Cottet, L Germain, Image processing through reaction combined with nonlinear diffusion. Math Comput. **61**, 659–673 (1993). doi:10.1090/S0025-5718-1993-1195422-2
32. S Osher, M Burger, D Goldfarb, J Xu, W Yin, An iterative regularization method for total variation-based image restoration. Multiscale Model Simulat. **4**(2), 460–489 (2005). doi:10.1137/040605412
33. P Buhlmann, B Yu, Boosting with the L_2 loss: regression and classification. J Am Stat Assoc. **98**(462), 324–339 (2003). doi:10.1198/016214503000125

doi:10.1186/1687-6180-2012-3
Cite this article as: Seo and Milanfar: **Robust flash denoising/deblurring by iterative guided filtering.** *EURASIP Journal on Advances in Signal Processing* 2012 **2012**:3.