Bayesian Inference for Neighborhood Filters with Application in Denoising

Chao-Tsung Huang, *Member, IEEE*

***Abstract*—Range-weighted neighborhood filters are useful and popular for their edge-preserving property and simplicity, but they are originally proposed as intuitive tools. Previous works needed to connect them to other tools or models for indirect property reasoning or parameter estimation. In this work, we introduce a unified empirical Bayesian framework to do both directly. A neighborhood noise model is proposed to reason and infer the Yaroslavsky, bilateral, and modified non-local means filters by joint maximum a posteriori and maximum likelihood estimation. Then the essential parameter, range variance, can be estimated via model fitting to the empirical distribution of an observable chi scale mixture variable. An algorithm based on expectation-maximization and Quasi-Newton optimization is devised to perform the model fitting efficiently. Finally, we apply this framework to the problem of color-image denoising. A recursive fitting and filtering scheme is proposed to improve the image quality. Extensive experiments are performed for a variety of configurations, including different kernel functions, filter types and support sizes, color channel numbers, and noise types. The results show that the proposed framework can fit noisy images well and the range variance can be estimated successfully and efficiently.**

***Index Terms*—Bilateral filter, non-local means, denoising, neighborhood filter, empirical Bayesian method, noise model, image model, parameter estimation.**

I. Introduction

Range-weighted formulation has been widely used to provide edge-preserved denoising since the introduction of local neighborhood filtering, especially the Yaroslavsky [1] and bilateral [2] filters. Many variations were also proposed for different applications, such as the trilateral filter for high contrast images [3], the cross bilateral filter for fusing image pairs [4], and the dual bilateral filter for aggregating stereo matching costs [5]. The reader is referred to [6] for more applications. The non-local means (NLM) [7] was further proposed for a non-local neighborhood and got fruitful patch-based extensions on the denoising problem.

The intuitive idea behind the neighborhood filters is to assign weighting coefficients based on similarity and then perform weighted averaging, e.g. the bilateral filter is given by

$$\hat{\mathbf{z}}_l = \frac{\sum_{i\in\Lambda_l} w_{l,i} d_{l,i} \cdot \mathbf{y}_i}{\sum_{i\in\Lambda_l} w_{l,i} d_{l,i}} \tag{1}$$

where $\mathbf{y}$ and $\hat{\mathbf{z}}$ are the observed and filtered signals respectively, and Λ_l represents the neighborhood of the pixel at position l. The adaptive weight consists of one range-weighted kernel $w_{l,i} = K_r(\frac{\|\mathbf{y}_l - \mathbf{y}_i\|_2^2}{2\sigma_r^2})$ and one distance-weighted kernel $d_{l,i} = K_d(\frac{\|l-i\|_2^2}{2\sigma_d^2})$. The former adapts to pixel similarity for edge preservation, which is controlled by range variance σ_r^2. And the latter provides a spatial smoothing window. A conventional choice for the kernel functions $K_r(\cdot)$ and $K_d(\cdot)$ is the Gaussian kernel which is in form of $K(x) = \exp(-x)$. If the spatial weights are all equal, it will degenerate to the Yaroslavsky filter. On the other hand, it will evolve to the NLM if $w_{l,i}$ is defined by patch similarity:

$$w_{l,i} = K_r\left(\frac{\sum_{b\in\mathcal{B}} \| \mathbf{y}_{l+b} - \mathbf{y}_{i+b} \|_2^2}{2B\sigma_r^2}\right),\ B = |\mathcal{B}|, \tag{2}$$

where $\{\mathbf{y}_{i+b} | b \in \mathcal{B}\}$ forms the patch at position i.

Filtering based on range-weighted similarity is effective but lacks of theoretical basis. For understanding its mathematical properties, several previous works studied their connections to other classical methods, such as mean shift [8], anisotropic diffusion [9], robust estimation [10], [11] and Bayesian approach [12], and thus found improvements on the neighborhood filters. However, these connections were unable to provide further information to infer the range variance σ_r^2 directly from the observed data.

In contrast, without reasoning the properties statistical techniques have been adopted to estimate the parameters indirectly using the basic observation model

$$\mathbf{y} = \mathbf{z} + \mathbf{n}, \tag{3}$$

where $\mathbf{n}$ is additive Gaussian noise. The χ^2 test was used to choose the parameters for the NLM filter in [13]. The Stein's unbiased risk estimate (SURE) [14] can provide unbiased estimation of the mean squared error (MSE) from noisy and filtered images. Thus many parameter combinations can be tested, and the one giving the smallest estimated MSE can be selected. Though accurate, the complexity is quite high because each combination needs to filter the image separately.

The contribution of this paper is that we build a unified empirical Bayesian framework to infer the neighborhood filters with novel property reasoning and also estimate their range variance σ_r^2 through statistical inference. Experimental results on color-image denoising show that the proposed model fits noisy images well, estimates σ_r^2 as accurate as SURE does, and even works well when SURE fails. The advantage over SURE is not only computation-wise but also quality-wise when considering a recursive filtering scheme. Besides adding more details, this paper extends our conference paper [15] with formulations for generalized kernel functions and also experiments on hyperspectral denoising, natural image gradients,

This work was supported by the Ministry of Science and Technology, Taiwan, R.O.C. under Grant No. MOST 103-2218-E-007-008-MY3.

C.-T. Huang is with the National Tsing Hua University, Hsinchu 30013, Taiwan (e-mail: chaotsung@ee.nthu.edu.tw).

different kernel functions, and different noise types. We also believe that this framework can be extended to other range-weighted algorithms for deeper theoretical understanding and more efficient parameter estimation.

The rest of this paper is organized as follows. After summarizing related works in Section II, we introduce a novel neighborhood noise model and infer the neighborhood filters in Section III. For efficiently estimating σ_r^2 from noisy images, we present an expectation-maximization-plus (EM+) algorithm in Section IV to perform the model fitting. In Section V, we apply our framework to color-image denoising and introduce the recursive fitting and filtering scheme to improve image quality. The extensive experimental results are shown in Section VI, and the limitations and possible future extensions of this work are discussed in Section VII. Finally, conclusion marks are given in Section VIII.

II. Related Work

A. Joint filter and parameter inference

In [16], a wavelet-domain denoising algorithm was developed with the assumption that the latent image $\mathbf{z}$ is in form of Gaussian scale mixtures (GSM) [17]. The adaptive parameters can then be statistically inferred from the neighborhood. In [18], noise estimation and removal can be performed automatically if detailed camera information is available. Not surprisingly, these filters are different from the range-weighted ones. The PLOW filter [19] is equivalent to the NLM filter plus a residual filter. Although the residual filter is based on the covariance matrices inferred from geometric clusters, the range variance was still given in a heuristic way. In this paper, we target the joint inference for neighborhood filters.

B. Neighborhood filter inference

In [10], the bilateral filter was reasoned as the first iteration of optimizing robust estimation with a given weight function, and multiple iterations were proposed to improve the denoising performance. In [11], the weight function was linked to the probability density function of Smooth Exponential Family, which showed that the parameters can be predicted by fitting the additive noisy model, but not directly from image data. In [12], the concept in [10] was extended to generalize the neighborhood filters. These Bayesian approaches were only able to infer the filter structure from the observed neighborhood, but no specific method was proposed to infer the range variance directly from the image data.

C. SURE-based parameter estimation

The SURE-based method gives the state-of-the-art accuracy for parameter estimation, and it is basically applicable to any parameter. It needs to estimate the noise variance first, and one popular method is by median absolute deviation (MAD). SURE has been applied to the bilateral filter for grey [20] and multispectral images [21] and also to the NLM filter [22], [23], [24]. A fast implementation for the bilateral filter was also proposed in [25], but several passes of filtering were still required. For the SURE-based method to work well, two conditions should be met: the independent Gaussian noise assumption and a weakly differentiable filter kernel on the noisy image $\mathbf{y}$. In this paper, the model fitting itself is only able to estimate the range variance. However, we will show its applicability to two cases in which the SURE-based method would fail. One is recursive filtering for which the noise is no longer independent or Gaussian. The other one is for the NLM filters which use motion estimation to select candidate patches such that the kernel is dynamic and thus indifferentiable on $\mathbf{y}$.

D. Image denoising

Several types of approaches have been studied, including local-based [1], [2], transform-based [16], [26], nonlocal-based [7], [19], and sparsity-based (e.g. BM3D [27], NLSM[28] and WNNM[29]). The last type showed superior image quality recently, which was based on grouping with patch similarity and optimization for sparse representation. However, the parameters were mostly given heuristically. Besides, they were often proposed and optimized for grey images, and additional modification was required to support color images. In contrast, our method can work on multi-channel signals directly.

III. Noise Model and Bayesian Inference

We will first propose a novel noise model and infer the Yaroslavsky filter directly from it. It reasons the range-weighted kernel $K_r(\cdot)$ using maximum a posteriori (MAP) estimation on novel localized soft-edge random variables and infers the filters using maximum likelihood (ML) estimation. By modifying the likelihood function to improve the robustness of estimation, we will then infer the bilateral filter and a modified NLM filter.

A. Neighborhood noise model (NNM)

Consider the weighted averaging formulation in (1). If the combined weights $w_{l,i}d_{l,i}$ are constant, a simple Gaussian model with scaled variance can do the filter inference. However, the difficulty of the inference for neighborhood filters lies in the dependency of the range weight $w_{l,i}$ on the observed signals $\mathbf{y}$, i.e. $\mathbf{y}_i$ cannot simultaneously decide the model parameter and serve as the model realization. In the following, we solve this problem by modelling $w_{l,i}$ as localized random variables which can represent local intensity edges using soft decision.

For a latent k-channel signal $\mathbf{z}_l$ at position l, we formulate its neighbors $\mathbf{y}_{i\in\Lambda_l}$ by GSM:

$$\mathbf{y}_i = \mathbf{z}_l + \frac{\mathbf{n}_{l,i}}{\sqrt{w_{l,i}}}, \tag{4}$$

where $\mathbf{n}_{l,i}$ are additive white Gaussian noises (AWGN) of covariance matrix $\sigma^2\mathbf{I}_\mathbf{k}$, and $w_{l,l} = 1$ as the basic observation model (3). For the neighboring pixels, a smaller realization of $w_{l,i}$ means a wider distribution for $\mathbf{y}_i$, so there is more likely an edge between positions l and i. Otherwise, smooth texture will be inferred if the realization is close to one. Note that the Gaussian assumption of $\mathbf{n}_{l,i}$ is a key for the following tractable formulations and also the formation of the observable variable in Section IV.

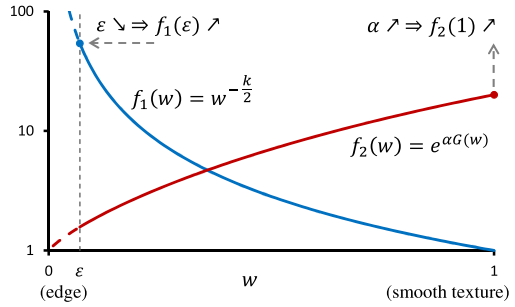


Fig. 1. Soft-edge prior distribution $f_w(w)$ comprised of two functions, i.e. $f_w(w) \propto f_1(w)f_2(w)$. The function $f_1(w) = w^{-\frac{k}{2}}$ highlights the distribution weight of edges (as w close to ϵ) which is controlled by ϵ. On the other hand, the function $f_2(w) = e^{\alpha G(w)}$ determines the distribution weight of smooth texture (as w close to 1) through parameterizing α.

To infer the range-weighted kernel, $w_{l,i\neq l}$ are defined as white hidden variables of a prior with two parameters ϵ and α:

$$f_w(w;\epsilon,\alpha) = \frac{1}{N(\epsilon,\alpha)} w^{-\frac{k}{2}} e^{\alpha G(w)},\ w \in [\epsilon, 1], \tag{5}$$

where $N(\epsilon,\alpha) = \int_\epsilon^1 w^{-\frac{k}{2}} e^{\alpha G(w)}\, dw$ for normalization. The function $G(w)$ will be shown to be directly linked to the range kernel function $K_r(\cdot)$ later. Regarding the two parameters, the α will link the noise variance σ^2 to the range variance σ_r^2, and the non-zero ϵ can guarantee the convergence of the integration of this prior for $k \geq 3$.

To explore the properties of this prior distribution, we can decompose it into two basic functions, $w^{-\frac{k}{2}}$ and $e^{\alpha G(w)}$, as shown in Fig. 1. Then, for an image with many intensity edges, we can model it by decreasing ϵ to have higher distribution weight on edges ($\epsilon^{-\frac{k}{2}} \uparrow$). In contrast, for an image with many flat regions, we can increase α to weight more on smooth textures ($e^{\alpha G(1)} \uparrow$). Therefore, this prior distribution can describe natural images reasonably by varying ϵ and α.

B. Inference for Yaroslavsky filter

Given observed data $\mathbf{y}_i$, we have the posterior $\Phi_l \propto p(\mathbf{y}_l;\mathbf{z}_l)\prod_{i\in\Lambda_l-\{l\}} p(\mathbf{y}_i|w_{l,i};\mathbf{z}_l) f_w(w_{l,i})$. By removing constants from $-\log\Phi_l$, we can derive the energy function L_l:

$$L_l = \frac{g_{l,l}^2}{2\sigma^2} + \sum_{i\in\Lambda_l-\{l\}} \frac{w_{l,i}g_{l,i}^2}{2\sigma^2} - \log w_{l,i}^{\frac{k}{2}} - \log f_w(w_{l,i}), \tag{6}$$

where $g_{l,i}^2 \triangleq \| \mathbf{z}_l - \mathbf{y}_i \|_2^2$. The estimation for $w_{l,i}$ and $\mathbf{z}_l$ can be derived by minimizing L_l:

$$\frac{\partial L_l}{\partial w_{l,i}} = 0 \quad \Rightarrow w_{l,i} = K_r(\frac{g_{l,i}^2}{2\sigma_r^2}),\ \sigma_r^2 = \alpha\sigma^2,\ (i \neq l) \tag{7}$$

$$\frac{\partial L_l}{\partial \mathbf{z}_l} = 0 \quad \Rightarrow \mathbf{z}_l = \frac{\sum_{i\in\Lambda_l} w_{l,i}\cdot\mathbf{y}_i}{\sum_{i\in\Lambda_l} w_{l,i}}, \tag{8}$$

where the kernel function $w = K_r(x)$ is related to $G(w)$ by

$$K_r(x) = (G')^{-1}(x) \Leftrightarrow G(w) = \int K_r^{-1}(w)dw. \tag{9}$$

The Yaroslavsky filter is then equivalent to the first-iteration estimation for solving this fixed-point problem with an initial condition $\mathbf{z}_l^{(0)} = \mathbf{y}_l$. For a given kernel function $K_r(x)$, we can now have its soft-edge prior $f_w(w)$ via the corresponding $G(w)$ by (9), and vice versa. Table I lists some examples for the kernel function, and $G(w)$ is chosen such that $G(0) = 0$ without loss of generality.

TABLE I
EXAMPLES OF RANGE-WEIGHTED KERNEL FUNCTIONS $K_r(x = \frac{g_{l,i}^2}{2\sigma_r^2})$. $G'(w) = K_r^{-1}(w)$. $r(w) = -K_r'(K_r^{-1}(w))$.

Kernel Type	$K_r(x)$	$G(w)$	$r(w)$
Gaussian	e^{-x}	$w(1-\log w)$	w
Laplacian	$e^{-x^{\frac{1}{2}}}$	$w(\log^2 w - 2\log w + 2)$	$\frac{w}{-2\log w}$
GGD4	e^{-x^2}	$w\sqrt{-\log w} + \frac{\sqrt{\pi}\,\mathrm{erfc}(\sqrt{-\log w})}{2}$	$2w\sqrt{-\log w}$
Epanechnikov	$(1-x)\mathbb{1}_{\{x\leq 1\}}$	$w(1-\frac{1}{2}w)$	$\mathbb{1}_{\{w>0\}}$
Biweight	$(1-x)^2\mathbb{1}_{\{x\leq 1\}}$	$w(1-\frac{2}{3}w^{\frac{1}{2}})$	$2w^{\frac{1}{2}}$
Triweight	$(1-x)^3\mathbb{1}_{\{x\leq 1\}}$	$w(1-\frac{3}{4}w^{\frac{1}{3}})$	$3w^{\frac{2}{3}}$

GGD: Generalized Gaussian Distribution

The NNM estimation (7) and (8) can be further interpreted into two sequential steps respectively:

1) Independent MAP estimation for each $w_{l,i}$ by maximizing $p(\mathbf{y}_i|w_{l,i};\mathbf{z}_l)f_w(w_{l,i})$ with a fixed $\mathbf{z}_l$;
2) ML estimation for $\mathbf{z}_l$ by optimizing the likelihood $\mathcal{L}_z(\mathbf{z}_l) = \frac{g_{l,l}^2}{2\sigma^2} + \sum_{i\in\Lambda_l-\{l\}} \frac{w_{l,i}g_{l,i}^2}{2\sigma^2}$ with fixed $w_{l,i}$.

By modifying the likelihood function in the second step for considering proximity and patch similarity, the bilateral and NLM filters are derived respectively in the following.

C. Extension for bilateral filter

To consider proximity in a nonparametric way using the distance-weighted kernel $d_{l,i}$, the locally weighted maximum likelihood (LWML) [30] can be applied to the likelihood $\mathcal{L}_z$ to have the pseudo likelihood function

$$\tilde{\mathcal{L}}_z(\mathbf{z}_l) = d_{l,l}\cdot\frac{g_{l,l}^2}{2\sigma^2} + \sum_{i\in\Lambda_l-\{l\}} d_{l,i}\cdot\frac{w_{l,i}g_{l,i}^2}{2\sigma^2}. \tag{10}$$

Then the LWML estimation by $\frac{\partial\tilde{\mathcal{L}}_z}{\partial\mathbf{z}_l} = 0$ gives the same formulation as the bilateral filter in (1).

D. Extension for NLM filter

For a latent pixel $\mathbf{z}_{l+b}$ in a latent patch ($b \in \mathcal{B}$), the corresponding NNM with its neighborhood Λ_{l+b} is as defined by (4). The first-iteration MAP estimation for each soft-edge variable then gives

$$w_{l+b,i+b} = K_r(\frac{\| \mathbf{y}_{l+b} - \mathbf{y}_{i+b} \|_2^2}{2\sigma_r^2}),\ i \in \Lambda_l. \tag{11}$$

Let the column vectors of the latent patch and the observed patches be $\mathbf{Z}_l$ and $\mathbf{Y}_{i\in\Lambda_l}$. They are formed by cascading $\mathbf{z}_{l+b}$ and $\mathbf{y}_{i+b}$ respectively in a predefined order Υ for $b \in \mathcal{B}$. Then the partial likelihood from $\mathbf{Y}_i$ can be formulated by

$$\mathcal{L}(\mathbf{Z}_l;\mathbf{Y}_i) = G(\mathbf{Z}_l;\mathbf{Y}_i,\mathbf{\Sigma}_{l,i}), \tag{12}$$

where $G(\cdot;\mu,\mathbf{\Sigma})$ is a multivariate Gaussian function, and the diagonal entries of $\mathbf{\Sigma}_{l,i}$ are formed by cascading

$\mathrm{diag}(\frac{\sigma^2}{w_{l+b,i+b}}\mathbf{I}_k)$ in the predefined order Υ. The entries outside of the diagonal have no effect on the following derivation.
For NLM filtering, one observed patch $\mathbf{Y}_i$ has only one summation weight $W_{l,i}$. Thus we use a patch-based likelihood function to approximate (12), which is devised as

$$\tilde{\mathcal{L}}(\mathbf{Z}_l;\mathbf{Y}_i) = G(\mathbf{Z}_l;\mathbf{Y}_i,\frac{\sigma^2}{W_{l,i}}\mathbf{I}_{kB}). \tag{13}$$

Since these two likelihood functions both behave like probability density functions (pdf), we choose $W_{l,i}$ by minimizing the Kullback-Leibler divergence (KLD) between them:

$$\mathcal{D}_{NLM} \triangleq \mathcal{D}_{KL}(\mathcal{L} \parallel \tilde{\mathcal{L}}) = -\int \mathcal{L}\log\frac{\tilde{\mathcal{L}}}{\mathcal{L}}\,d\mathbf{Z}_l, \tag{14}$$

$$\begin{aligned}\frac{\partial \mathcal{D}_{NLM}}{\partial W_{l,i}} &= -\int \mathcal{L}\left(\frac{kB}{2}\frac{1}{W_{l,i}} - \frac{\parallel \mathbf{Z}_l-\mathbf{Y}_i\parallel_2^2}{2\sigma^2}\right)d\mathbf{Z}_l \\ &= -\left(\frac{kB}{2}\frac{1}{W_{l,i}} - \frac{1}{2}\sum_{b\in\mathcal{B}}\frac{k}{w_{l+b,i+b}}\right) = 0, \end{aligned}\tag{15}$$

$$\Rightarrow W_{l,i} = \left(\sum_{b\in\mathcal{B}} w_{l+b,i+b}^{-1}/B\right)^{-1}. \tag{16}$$

Then the ML estimation for the combined patch-based likelihood functions $\sum_{i\in\Lambda_l}\tilde{\mathcal{L}}(\mathbf{Z}_l;\mathbf{Y}_i)$ becomes

$$\mathbf{Z}_l = \frac{\sum_{i\in\Lambda_l} W_{l,i}\cdot\mathbf{Y}_i}{\sum_{i\in\Lambda_l} W_{l,i}}, \tag{17}$$

which suggests a modified NLM (MNLM) filter with the coefficients $W_{l,i}$ in (16). Note that $W_{l,i}$ is the harmonic average of $w_{l+b,i+b}$, while the conventional NLM with the Gaussian kernel uses the geometric average as implied in (2).

IV. Model Fitting and Parameter Estimation

In the following, we will first introduce how to build a robust observation model in chi scale mixtures and then how we fit the model using an EM+ algorithm.

A. Observable chi scale mixtures (CSM)

Let $s_{l,i} \triangleq \parallel \mathbf{y_l}-\mathbf{y_i}\parallel_2$. Due to the AWGN assumption of $\mathbf{n}_{l,i}$ in (4), $s_{l,i}$ is independent of $\mathbf{z_l}$ and with the CSM formulation:

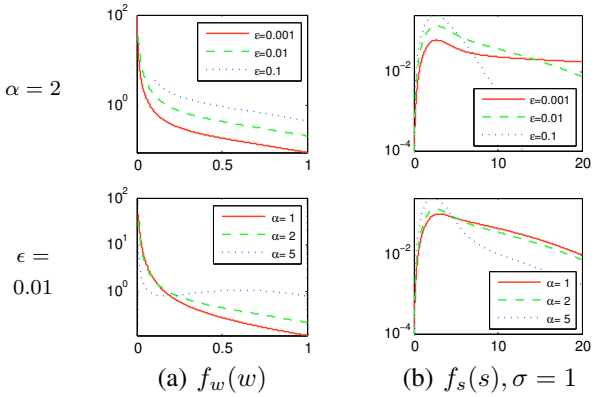
$$s_{l,i} = \sigma\sqrt{\frac{w_{l,i}+1}{w_{l,i}}}u_{l,i}, u_{l,i}\sim\chi_k, \tag{18}$$

where $u_{l,i}$ has a chi distribution with k degrees of freedom. For simplicity, we use $s = s_{l,i}$ and $w = w_{l,i}$ in the following. The marginal pdf of s can be derived by

$$f_s(s;\sigma,\epsilon,\alpha) = \int_\epsilon^1 f_{s,w}(s,w;\sigma,\epsilon,\alpha)\,dw, \tag{19}$$

$$f_{s,w} = \frac{1}{T(\epsilon,\alpha)}\frac{s^{k-1}\sigma^{-k}}{(w+1)^{\frac{k}{2}}}e^{-\frac{w}{w+1}\frac{s^2}{2\sigma^2}}e^{\alpha G(w)}, \tag{20}$$

where $f_{s,w}$ is the joint pdf of s and w, and $T(\epsilon,\alpha) = 2^{\frac{k}{2}-1}\Gamma(k/2)N(\epsilon,\alpha)$ for normalization.
Fig. 2 shows how the soft-edge prior $f_w(w)$ and the CSM pdf $f_s(s)$ behave with different α and ϵ. $f_w(w)$ represents



(a) $f_w(w)$ (b) $f_s(s), \sigma = 1$

Fig. 2. Distributions of the soft-edge w and the observable chi scale mixtures s with different values for α and ϵ (Gaussian kernel $K_r(\cdot)$ and $k = 3$).

how likely intensity edges appear, i.e. higher density for small w for more edges. For smaller α or ϵ, $f_w(w)$ tilts more to the left and $f_s(s)$ has a longer tail such that more edges are expected. For larger α or ϵ, $f_w(w)$ concentrates more on the right, and $f_s(s)$ gets closer to a chi distribution. Thus we can model noisy images by varying α and ϵ to fit different image properties and varying σ for different noise intensity.
Another advantage to define s in l^2-norm is its robustness. When the components of $\mathbf{y}_i$ are not independent in the observed color space, e.g. RGB, we may apply an orthogonal transform to diagonalize their covariance matrix $\Sigma_\mathbf{y}$ for decorrelation. However, since the l^2-norm is invariant to the orthogonal transform, we can calculate s (and w) in the observed color space without performing the decorrelation. Thus the model fitting (and the filtering) can be applied on the observed $\mathbf{y}_i$ directly without loss of optimality.

B. EM+ algorithm for CSM fitting

Given an observed data set $\mathcal{S}$, its empirical distribution is defined by $P(s\in\mathcal{S})$. Then we estimate the corresponding pdf $f_s(s)$ by iteratively updating the model parameters based on $P(s)$. Assume in the previous iteration the estimated parameters are σ, α, and ϵ. Our EM+ algorithm updates them to $\tilde{\sigma}$, $\tilde{\alpha}$, and $\tilde{\epsilon}$ through the following three steps: EM update, KLD update, and Quasi-Newton (QN) update.

1) EM update: $(\sigma,\alpha,\epsilon) \Rightarrow (\hat{\sigma},\hat{\alpha},\epsilon)$*:* For simplicity, we will ignore the σ, α, and ϵ in $f_{s,w}$ and f_s if the parameters in the previous iteration are used. The expected value of the log likelihood function can be derived as $Q(\hat{\sigma},\hat{\alpha}|\sigma,\alpha) = \sum_j P(s_j)q(s_j,\hat{\sigma},\hat{\alpha}|\sigma,\alpha)$ where

$$q(s,\hat{\sigma},\hat{\alpha}|\sigma,\alpha) = \int_\epsilon^1 p(w|s)\mathcal{L}_{EM}(\hat{\sigma},\hat{\alpha},\epsilon;w,s)\,dw, \tag{21}$$

$$p(w|s) = f_{s,w}/f_s,\ \mathcal{L}_{EM} = \log f_{s,w}(s,w;\hat{\sigma},\epsilon,\hat{\alpha}). \tag{22}$$

Set $\frac{\partial Q}{\partial\hat{\sigma}} = \frac{\partial Q}{\partial\hat{\alpha}} = 0$ to have the EM update for $\hat{\sigma}$ and $\hat{\alpha}$:

$$\hat{\sigma}^2 = \frac{1}{k}\sum_j P(s_j)\cdot\frac{\int_\epsilon^1 f_{s,w}(s_j,w)s_j^2\frac{w}{w+1}\,dw}{f_s(s_j)}, \tag{23}$$

$$H(\hat{\alpha},\epsilon) = \sum_j P(s_j)\cdot\frac{\int_\epsilon^1 f_{s,w}(s_j,w)G(w)\,dw}{f_s(s_j)}, \tag{24}$$

where $H(\hat{\alpha},\epsilon) \triangleq \mathbf{E}_{w;\hat{\alpha},\epsilon}[G(w)]$ and it is an increasing function with respect to $\hat{\alpha}$ because $\partial H/\partial\hat{\alpha} = \mathbf{Var}_{w;\hat{\alpha},\epsilon}[G(w)] \geqq 0$. Thus the corresponding $\hat{\alpha}$ can be found quickly using a bisection search on $H(\hat{\alpha},\epsilon)$.

2) KLD update: $(\hat{\sigma},\hat{\alpha},\epsilon) \Rightarrow (\hat{\sigma},\hat{\alpha},\hat{\epsilon})$*:* The EM algorithm is unable to update ϵ because the support of w for $p(w|s)$ and $\mathcal{L}_{EM}$ in (21) should be the same. Instead, we update it by optimizing the approximate KLD $\mathcal{D} \triangleq \sum_j -P(s_j)\cdot \log\frac{f_s(s_j;\hat{\sigma},\epsilon,\hat{\alpha})}{P(s_j)}$. With $\frac{\partial\mathcal{D}}{\partial\epsilon}=0$, a fixed-point representation of the optimal ϵ can be derived. And we use the first-iteration result with an initial condition $\hat{\epsilon}^{(0)} = \epsilon$ as our updating formulation:

$$\left(\frac{\hat{\epsilon}+1}{\hat{\epsilon}}\right)^{\frac{k}{2}} = \sum_j P(s_j)\cdot \frac{s_j^{k-1}\hat{\sigma}^{-k}e^{-\frac{\epsilon}{\epsilon+1}\frac{s_j^2}{2\hat{\sigma}^2}}}{2^{\frac{k}{2}-1}\Gamma(\frac{k}{2})f_s(s_j;\hat{\sigma},\epsilon,\hat{\alpha})}. \quad (25)$$

3) QN update: $(\hat{\sigma},\hat{\alpha},\hat{\epsilon}) \Rightarrow (\tilde{\sigma},\tilde{\alpha},\tilde{\epsilon})$*:* The update formulations in (23), (24), and (25), require numerical evaluations for lots of integrals, which prolongs the execution time for one iteration. Also, the update step sizes are usually small near the optimizing point, which increases the number of iterations. Thus the above updates usually take a long time to converge as shown by the blue dotted line in Fig. 3.

We adopt the QN method, QN1 in [31], to accelerate the fitting process. Let $\boldsymbol{\theta} = (\sigma,\alpha,\epsilon)^T$, $\hat{\boldsymbol{\theta}} = (\hat{\sigma},\hat{\alpha},\hat{\epsilon})^T$, $\tilde{\boldsymbol{\theta}} = (\tilde{\sigma},\tilde{\alpha},\tilde{\epsilon})^T$, and $\mathbf{F}(\boldsymbol{\theta}) = \hat{\boldsymbol{\theta}}(\boldsymbol{\theta}) - \boldsymbol{\theta}$. The QN1 method solves $\mathbf{F}(\boldsymbol{\theta}) = \mathbf{0}$ by maintaining a matrix $\mathbf{A}$ which approximates the inverse Jacobian matrix $\mathbf{J}_{\mathbf{F}}^{-1}$ and is updated using the Broyden's update method ($\mathbf{A} \leftarrow \mathbf{A} + \triangle\mathbf{A}$):

$$\triangle\mathbf{A} = \frac{(\triangle\boldsymbol{\theta} - \mathbf{A}\triangle\mathbf{F})\triangle\boldsymbol{\theta}^T\mathbf{A}}{\triangle\boldsymbol{\theta}^T\mathbf{A}\triangle\mathbf{F}}, \quad (26)$$

where $\triangle\boldsymbol{\theta} = \boldsymbol{\theta} - \boldsymbol{\theta}_{pre}$, $\triangle\mathbf{F} = \mathbf{F}(\boldsymbol{\theta}) - \mathbf{F}(\boldsymbol{\theta}_{pre})$, and $\boldsymbol{\theta}_{pre}$ is the estimation in the earlier iteration before $\boldsymbol{\theta}$. Then the QN update is derived by $\tilde{\boldsymbol{\theta}} = \boldsymbol{\theta} + \boldsymbol{\delta}$ where

$$\boldsymbol{\delta} = -\mathbf{A}\cdot\mathbf{F}(\boldsymbol{\theta}). \quad (27)$$

However, in practice the QN update could be unstable when $\mathbf{A}$ has a high condition number. Some heuristic, e.g. reinitializing $\mathbf{A}$, is required in this situation. But as shown in Fig. 3, the reinitialization (red dashed line) may not work well. Instead, we choose to confine $\tilde{\boldsymbol{\theta}}$ in a reasonably large range $\boldsymbol{\Psi}$ for its quick convergence (black solid line in Fig. 3). When $\tilde{\boldsymbol{\theta}}$ is out of $\boldsymbol{\Psi}$, we will directly set $\boldsymbol{\delta}$ to $\mathbf{F}(\boldsymbol{\theta})$ as the original EM/KLD update and may further scale down its value to make $\tilde{\boldsymbol{\theta}}$ inside $\boldsymbol{\Psi}$ if necessary. With this QN acceleration, the fitting can be converged in fewer than fifteen iterations in most cases.

C. Discriminative capability of KLD

The fitting quality relies on the discriminative capability of KLD between different parameters. Fig. 4 shows the KLD discrimination of $f_s(s)$ for $k=3$ when α has a small change $\triangle\alpha = 0.5$ and when ϵ increases by $\triangle\epsilon = 0.005$. It is clear that the discriminative capability declines as α or ϵ increases. For sufficiently large α or ϵ, $f_s(s)$ is very similar to a chi distribution and KLD becomes insensitive to the parameters, which usually happens when the noise intensity is high. Therefore, some upper bounds for α and ϵ may be chosen without compromising KLD.

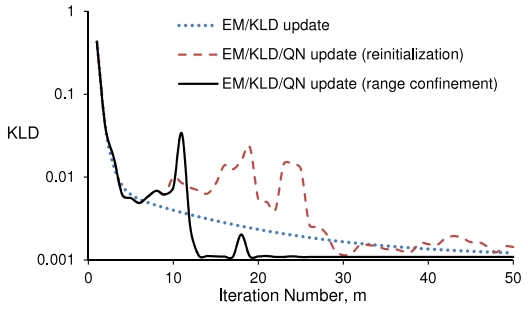
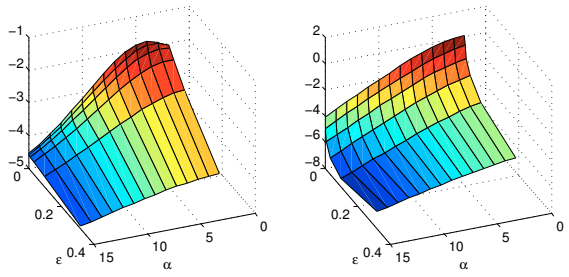


Fig. 3. KLD convergence for CSM fitting methods (*B*aboon, $\sigma_n = 20$, bilateral 9×9 filter, Gaussian kernel $K_r(\cdot)$). The blue dotted line stands for EM/KLD update without QN, the red dashed line for EM+ update with QN reinitialization when the condition number of $\mathbf{A} > 40$, and the black solid line for EM+ update with the parameter range $\boldsymbol{\Psi}$ given in Section V-A.



(a) $\log_{10}\mathcal{D}_{\triangle\alpha=0.5}$ (b) $\log_{10}\mathcal{D}_{\triangle\epsilon=0.005}$

Fig. 4. KLD discrimination (Gaussian kernel $K_r(\cdot)$ and $k=3$) on (a) α and (b) ϵ, where $\mathcal{D}_{\triangle\alpha}(\alpha,\epsilon) \triangleq \mathcal{D}_{KL}(f_s(s;\sigma,\epsilon,\alpha) \parallel f_s(s;\sigma,\epsilon,\alpha+\triangle\alpha))$ and $\mathcal{D}_{\triangle\epsilon}(\alpha,\epsilon) \triangleq \mathcal{D}_{KL}(f_s(s;\sigma,\epsilon,\alpha) \parallel f_s(s;\sigma,\epsilon+\triangle\epsilon,\alpha))$. Note that σ affects only the scaling and has no effect on $\mathcal{D}_{\triangle\alpha}$ and $\mathcal{D}_{\triangle\epsilon}$.

V. Application in Image Denoising

For the inferred filters, each observed neighborhood can be viewed as a realization of the NNM. Thus the CSM model fitting can provide parameter estimation. In the following, we will apply this approach to color-image denoising.

A. Parameter estimation

Before model fitting, we need to derive the empirical distribution $P(s)$. For the Yaroslavsky filter, we have a fixed neighborhood Λ_l for each pixel, and $P(s)$ can be simply obtained by accumulating the histogram of all $s_{l,i}$. For the bilateral filter, we construct its $P(s)$ by accumulating the histogram with the spatial weight $d_{l,i}$ to consider proximity. For example, the frequency of $s=2$ will be increased by $d_{l,i}$ for an event $s_{l,i}=2$. For the MNLM filter, we consider a dynamic neighborhood Λ_l which may depend on patch similarity sorting or hard thresholding for better performance. In this case, we need to do the computation to derive Λ_l and then accumulate the histogram of $s_{l+b,i+b}$.

Given $P(s)$, we perform the CSM fitting using the EM+ algorithm and select the final result based on the KLD calculated for $P(s)$ and the estimated distribution $\tilde{P}(s;\boldsymbol{\theta}^{(m)})$ in each iteration. Besides, we also devise an ϵ-bounded estimation to

handle the KLD insensitivity issue. It is activated when ϵ is close to a given upper bound ϵ_{bd}. Since the sensitivity of KLD becomes small when ϵ is near ϵ_{bd}, we can directly compare KLD through $\epsilon = \epsilon_{bd}$ using a bisection search on α. For each α, the σ-update (23) which converges very quickly is used to find the best corresponding σ and KLD. This parameter estimation procedure is summarized in Algorithm 1.

Algorithm 1 CSM Parameter Estimation

Input: Empirical distribution $P(s)$; ϵ-bound ϵ_{bd}
Output: Estimated $\boldsymbol{\theta} = (\sigma, \alpha, \epsilon)^T$ and σ_r^2

Initialize $\boldsymbol{\theta}^{(0)} = \boldsymbol{\theta}_{ini}$, $\mathbf{F}(\boldsymbol{\theta}^{(0)}) = \mathbf{0}$, $\mathbf{A} = -\mathbf{I}$, $m = 1$
repeat ▷ EM+ algorithm
  QN update: $\boldsymbol{\theta}^{(m)} \leftarrow \boldsymbol{\theta}^{(m-1)} + \boldsymbol{\delta}$
  EM/KLD update: $\hat{\boldsymbol{\theta}}(\boldsymbol{\theta}^{(m)})$, $\mathbf{F}(\boldsymbol{\theta}^{(m)}) = \hat{\boldsymbol{\theta}} - \boldsymbol{\theta}^{(m)}$
  Broyden's update: $\mathbf{A} \leftarrow \mathbf{A} + \triangle\mathbf{A}$
  KLD: $\mathcal{D}^{(m)} = \mathcal{D}_{KL}(P(s) \parallel \tilde{P}(s; \boldsymbol{\theta}^{(m)}))$
  $m \leftarrow m + 1$
until (KLD or σ_r^2 converges) or $(m > M)$
Get $\boldsymbol{\theta} = \boldsymbol{\theta}^{(m')}$ where $m' = \arg\min_m \mathcal{D}^{(m)}$
if $|\epsilon - \epsilon_{bd}| < \triangle\epsilon$ **then** ▷ ϵ-bounded estimation
  $\epsilon \leftarrow \epsilon_{bd}$
  $(\alpha, \sigma) \leftarrow \epsilon$-bound estimated $(\alpha_{bd}, \sigma_{bd})$
end if
Get the estimated $\sigma_r^2 \leftarrow \alpha\sigma^2$

The default algorithm parameters in this paper are as follows: maximum fitting iteration $M = 15$, initial CSM parameter $\boldsymbol{\theta}_{ini} = (\frac{s_{md}}{\sqrt{2(k-1)}}, k, 10^{-3})$ where s_{md} is the mode of $P(s)$, ϵ-bound criteria $\triangle\epsilon = 10^{-3}$, and parameter range $\boldsymbol{\Psi} = \{\boldsymbol{\theta} | \sigma \in [10^{-5}, \infty), \alpha \in [k, 5k], \epsilon \in [10^{-5}, \epsilon_{bd}]\}$. The convergence condition is that the KLD is smaller than 10^{-5} or σ_r^2 changes by less than 0.1%. Note that $\boldsymbol{\Psi}$ is used to avoid unstable QN updates and is defined sufficiently large such that only seldom final results touch the boundaries. The only exception is the ϵ_{bd} which can be used to activate the ϵ-bounded estimation.

B. Recursive local filter

It is shown that the conventional Yaroslavsky/bilateral filter is equivalent to the first-iteration MAP/ML estimation. Simply applying more iterations with the same σ_r^2 indeed reduces the energy function (6) but does not help for increasing PSNR. Instead, because the image noise becomes smaller after filtering, we propose a more reasonable alternative: apply the model fitting and filtering recursively. Each iteration estimates the NNM parameters for the current noisy image $\hat{\mathbf{y}}^{(n)}$ and performs filtering on it with the estimated range variance $\hat{\sigma}_r^{2(n)}$. This process is terminated when the filter iteration exceeds N_{flt} times or the estimated noise intensity $\hat{\sigma}^{(n)}$ is smaller than a threshold σ_{cl} which represents a clean image.

C. Recursive MNLM filter

The proposed recursive MNLM filter has three differences from the recursive local one. First, the basic processing unit becomes a patch, instead of a pixel, and a flag b_{ag} decides how to aggregate these patches into one image. If $b_{ag} = 1$, each image pixel at position l will be derived by averaging all its corresponding values in neighboring estimated patches $\hat{\mathbf{X}}_{l+b}, b \in \mathcal{B}$. Otherwise, no aggregation will be performed. Second, the neighborhood for each patch is constructed dynamically based on some given constraints. Third, a DCT-Wiener filter is introduced to increase the performance, which is activated by a flag b_{dct}.

The DCT-Wiener filter serves similarly as the residual filter in PLOW [19]. We use the DCT, denoted as $\mathcal{T}(\cdot)$, to approximate the decorrelation matrix and then apply element-wise Wiener filtering to update the patch $\hat{\mathbf{X}}_l$

$$\hat{\mathbf{X}}_l \leftarrow \mathcal{T}^{-1}(\mathbf{W}_{wie} \circ \mathcal{T}(\hat{\mathbf{X}}_l)), \tag{28}$$

where the element of the shrinkage matrix $\mathbf{W}_{wie}$ is

$$(\mathbf{W}_{wie})_{i'j'k'} = \frac{\hat{\sigma}^2_{X,i'j'k'}}{\hat{\sigma}^2_{X,i'j'k'} + \hat{\sigma}^2_l}, \tag{29}$$

and the signal variance is estimated from neighbors by $\hat{\sigma}^2_{X,i'j'k'} = \mathbf{E}_{i \in \Lambda_l}[(\mathcal{T}(\hat{\mathbf{Y}}_i^{(n)}))^2_{i'j'k'}] - \hat{\sigma}^{2(n)}$ and the noise variance by $\hat{\sigma}^2_l = \frac{\hat{\sigma}^{2(n)}}{\sum_{i \in \Lambda_l} W_{l,i}}$ due to weighted averaging. The recursive MNLM filter is summarized in Algorithm 2.

Algorithm 2 Recursive MNLM Filter

Input: Noisy image $\mathbf{y}$; ϵ-bound ϵ_{bd}; DCT-Wiener flag b_{dct}; Aggregation flag b_{ag}
Output: Denoised image $\hat{\mathbf{z}}$

Initialize $\hat{\mathbf{y}}^{(1)} = \mathbf{y}$, $\hat{\mathbf{z}} = \mathbf{y}$, $n = 1$
repeat
  Get empirical $P(s)$ of $\hat{\mathbf{y}}^{(n)}$ by constructing each Λ_l
  Get $\hat{\sigma}_r^{(n)}$, $\hat{\sigma}^{(n)}$ using parameter estimation
  if $(\hat{\sigma}^{(n)} \geq \sigma_{cl})$ **then**
    for each patch $\mathbf{Y}_l$ in $\hat{\mathbf{y}}^{(n)}$ **do**
      Get the estimation $\hat{\mathbf{X}}_l = \frac{\sum_{i \in \Lambda_l} W_{l,i} \cdot \hat{\mathbf{Y}}_i^{(n)}}{\sum_{i \in \Lambda_l} W_{l,i}}$
      Perform DCT-Weiner filtering with b_{dct}
    end for
    Aggregate $\hat{\mathbf{X}}_l$ to form $\hat{\mathbf{z}}^{(n)}$ with b_{ag}
    $\hat{\mathbf{y}}^{(n+1)} \leftarrow \hat{\mathbf{z}}^{(n)}$, $\hat{\mathbf{z}} \leftarrow \hat{\mathbf{z}}^{(n)}$
  end if
  $n \leftarrow n + 1$
until $(m > N_{flt})$ or $(\hat{\sigma}^{(n)} < \sigma_{cl})$

VI. Experiments on Model Fitting and Filtering

Extensive experiments will be given for showing the robustness and applicability of the proposed framework. There are four major configuration groups: filter type/support size, kernel function $K_r(\cdot)$, color channel number, and noise type. For clarity, we use a default combination as the backbone and mostly change one group at a time. The default configuration is: bilateral 9×9 filter, Gaussian kernel (GK), true-color images ($k = 3$), and AWGN. Twelve standard color images of different properties are used in this default case with five noise intensity values of σ_n. The details of all experimental results can be found and browsed online[1].

[1] http://www.ee.nthu.edu.tw/chaotsung/nnm

A. Test configuration setting

Six configurations for filter type/support size are tested:

1) **YF-5×5**: Yaroslavsky filter, Λ: 5×5, $\epsilon_{bd} = 0.1/1.0$;
2) **YF-9×9**: Yaroslavsky filter, Λ: 9×9, $\epsilon_{bd} = 0.1/1.0$;
3) **BF-9×9**: Bilateral filter, Λ: 9×9, $\epsilon_{bd} = 0.1/1.0$;
4) **BF-13×13**: Bilateral filter, Λ: 13×13, $\epsilon_{bd} = 0.1/1.0$;
5) **MNLM-Simple**: MNLM filter, $\epsilon_{bd} = 0.1$, $b_{dct} = 0$ (DCT-Wiener off), $b_{ag} = 0$ (no aggregation);
6) **MNLM-DCT**: MNLM filter, $\epsilon_{bd} = 1.0$ (ϵ-bound off), $b_{dct} = 1$ (DCT-Wiener on), $b_{ag} = 1$ (one-pixel grid).

Two settings of ϵ_{bd} are tested for local filters. The basic $\epsilon_{bd} = 1.0$ simply follows the definition in (5). In contrast, the empirical $\epsilon_{bd} = 0.1$ will use the ϵ-bounded estimation for large ϵ. The distance-weighted kernel $K_d(\cdot)$ of bilateral filters is Gaussian, and the σ_d is set to the radius of Λ, e.g. $\sigma_d = 4$ for $\Lambda = 9 \times 9$. The patch size for the MNLM filter is 9×9. For constructing MNLM neighborhood, we apply motion estimation in a 31×31 search window around position l and choose the best ten candidates as Λ_l. For filter termination, the maximum iteration number N_{flt} is set to 3 and σ_{cl}^2 set to 10.

To obtain the best σ_r^2 and the SURE estimation for comparison, we perform a σ_r^2 scan (30 values) for each test condition. The SURE-based method uses the noise variance σ_{MAD}^2 estimated by MAD as done in [19] and is denoted as MAD+SURE. The SURE formulation for generalized range-weighted kernels is derived in Appendix A.

B. Yaroslavsky/Bilateral filter (GK, $k = 3$, AWGN)

Fig. 5 shows typical examples of the test results. The CSM model can fit the long-tailed empirical distributions well no matter when the noise is small or large. It also successfully predicts that α should become larger as the noise intensity σ_n increases, while the conventional heuristics usually apply a fixed value.

Table II lists the result of the default test configuration in detail. MAD+SURE fails in two cases. One is for small σ_n (e.g. σ_n=5, 1st iteration) due to the inaccuracy of σ_{MAD}^2. The other one is for the iterations after the first one (e.g. σ_n=50, 2nd iteration) because the noise is not Gaussian any more. In contrast, the CSM estimation performs well in terms of both PSNR and σ_r^2 accuracy in these two cases. It means that the edge information can be well captured by the CSM model even when the noise is small or becomes non-Gaussian. This useful property also enables the proposed recursive scheme. Besides, in other cases the CSM estimation shows similar performance compared to the MAD+SURE. An interesting property can also be found. The CSM estimation usually underestimates the noise variance σ^2, which means it may mistake noises for edges. In contrast, the MAD+SURE tends to overestimate because the MAD may mistake edges for noises.

The execution time is also shown in Table II, which is evaluated by running MATLAB (R2010b version) on a 3.4 GHz Intel Core i7 CPU (single-thread) with 8 MB cache. The proposed method runs much faster than the MAD+SURE since it does not require a σ_r^2 scan. Each iteration of it consists of three steps: getting $P(s)$, CSM fitting, and filtering. The first and third steps take time proportionally to the image resolution and filter support size (7.9 s and 6.5 s on average respectively). In contrast, the fitting time depends on the EM+ iteration numbers and whether the ϵ-bounded estimation is turned on. For larger noises, the fitting tends to be insensitive to KLD, and thus more time is required, e.g. 9.6 s on average for $\sigma_n = 50$ (1st iteration) while only 3.5 s for $\sigma_n = 5$. The percentage of the fitting time will be smaller for larger images. Note that MAD+SURE not only fails to predict in the recursive iterations but also requires significantly higher computation complexity due to the σ_r^2 scan.

The test results of Yaroslavsky and bilateral filters are summarized in Table III. The MATLAB code of the proposed recursive fitting and filtering for these tests is available online[1]. For $\epsilon_{bd} = 0.1$, the CSM estimation performs comparably to the SURE in the first iteration for large σ_n and outperforms for small σ_n. Moreover, the proposed recursive fitting and filtering can increase PSNR by up to 1.2 dB. As for $\epsilon_{bd} = 1.0$, its first-iteration results are not as good as $\epsilon_{bd} = 0.1$ for its less accurate σ_r^2 estimation and higher KLD (due to KLD insensitivity). However, the recursive filtering can recover its quality drop and even make it slightly better than $\epsilon_{bd} = 0.1$ with a little more iterations. Therefore, the basic $\epsilon_{bd} = 1.0$ gives slightly better quality when using recursive filtering, but the empirical $\epsilon_{bd} = 0.1$ may be preferred when only one or fewer iterations are allowed. In the following, the $\epsilon_{bd} = 0.1$ will be used, except for the performance-oriented **MNLM-DCT**. It can also be found that the choice of σ_r^2 affects the performance much more sensitively than that of Λ and $d_{l,i}$, which justifies the need of a good σ_r^2 estimator.

C. MNLM filter (GK, $k = 3$, AWGN)

The test results are summarized in Table IV. The **MNLM-Simple** is directly inferred from the patch-based NNM variation, and the CSM estimation can track the σ_r^2 well for different σ_n and in different iterations. The recursive **MNLM-Simple** filter can increase PSNR by up to 3.9 dB. On the other hand, the **MNLM-DCT** can provide better PSNR due to the use of DCT-Wiener filtering, and it also terminates earlier. Though less accurate, the CSM fitting still provides good estimation of σ_r^2 for it. Note that the MAD+SURE cannot be applied here because the neighborhood is dynamically derived. For comparing the conventional NLM and MNLM, we also tested NLM in form of (2) using a σ_r^2 scan. Its best PSNR differs by less than 1% compared to **MNLM-Simple**, which indicates NLM and MNLM have similar performance.

Two state-of-the-art denoising algorithms, CPLOW [19] and CBM3D [27] ("C" stands for the versions for color images), are also tested for comparison. The MAD-derived noise variance σ_{MAD}^2 is used for them, and all other parameters are set as default in the software provided by their authors[2,3]. The CBM3D performs best due to the processing in sparse representation and the good heuristic parameters. Our **MNLM-DCT** outperforms the CPLOW which is the state-of-the-art non-local filter, though sharing a similar flow. Besides

[2]BM3D: http://www.cs.tut.fi/~foi/GCF-BM3D/BM3D.zip
[3]PLOW: https://users.soe.ucsc.edu/~priyam/PLOW/

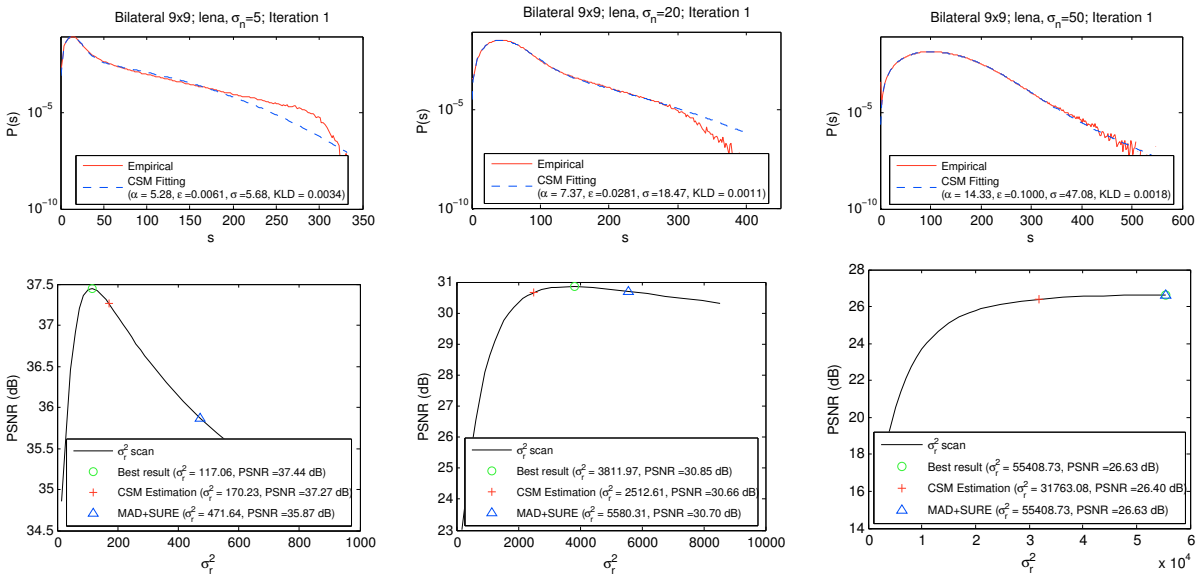


Fig. 5. **BF-9×9** ($\epsilon_{bd} = 0.1$, GK, $k = 3$, AWGN) for *Lena* with $\sigma_n = 5$, 20, and 50. The top row shows the model fitting of empirical distributions. The bottom row presents the filtered results, where the best σ_r^2 is marked by a circle, CSM estimation by a cross, and MAD+SURE by a triangle.

TABLE II
DETAILED RESULT FOR **BF-9×9** ($\epsilon_{bd} = 0.1$, GK, $k = 3$, AWGN).

Image	**σ_n=5, 1st iteration**												**σ_n=20, 1st iteration**											
	Best*		CSM Fitting						MAD+SURE				Best		CSM Fitting						MAD+SURE			
	PSNR (dB)	$\sigma_r{}^2$	σ^2	α	KLD	ΔPSNR† (dB)	$\Delta\sigma_r{}^2$ (rel.)‡	time (sec)	$\sigma^2{}_{MAD}$	ΔPSNR (dB)	$\Delta\sigma_r{}^2$ (rel.)	time (sec)	PSNR (dB)	$\sigma_r{}^2$	σ^2	α	KLD	ΔPSNR (dB)	$\Delta\sigma_r{}^2$ (rel.)	time (sec)	$\sigma^2{}_{MAD}$	ΔPSNR (dB)	$\Delta\sigma_r{}^2$ (rel.)	time (sec)
Lena	37.4	117	32.2	5.3	0.0034	**-0.2**	45%	16.1	52.8	-1.6	303%	264.0	30.9	3812	341.1	7.4	0.0011	-0.2	-34%	17.8	448.8	**-0.2**	46%	251.8
Baboon	35.0	98	70.5	3.0	0.0059	**-0.8**	115%	17.0	193.8	-5.3	949%	297.2	26.3	1725	381.2	3.3	0.0011	**-0.2**	-27%	17.9	646.2	-0.8	109%	325.9
Barbara	37.3	130	26.2	3.8	0.0011	**-0.1**	-24%	22.4	71.8	-1.9	298%	519.3	28.7	2356	326.8	4.6	0.0009	-0.4	-37%	24.8	501.5	**-0.3**	55%	548.1
Peppers	36.7	92	40.4	5.7	0.0035	**-0.7**	149%	15.0	61.9	-2.1	539%	304.4	30.7	4384	339.8	7.0	0.0020	-0.3	-45%	17.9	448.8	**-0.1**	27%	298.6
F16	38.8	134	24.7	5.5	0.0040	**0.0**	2%	13.8	44.3	-1.1	202%	329.0	31.3	3526	315.5	6.5	0.0024	-0.6	-42%	16.7	448.8	**-0.1**	46%	289.4
House	37.7	106	29.2	5.6	0.0042	**-0.2**	54%	5.9	44.3	-1.2	208%	89.7	30.8	3537	316.5	6.3	0.0043	-0.5	-43%	7.6	448.8	**-0.1**	46%	73.4
Kodim04	38.7	127	28.0	5.2	0.0113	**0.0**	15%	19.2	44.3	-1.0	147%	482.9	31.0	3280	347.2	8.1	0.0008	-0.1	-14%	25.4	423.5	**-0.1**	37%	480.9
Kodim08	37.3	127	31.1	3.9	0.0116	**0.0**	-5%	21.7	118.7	-3.1	428%	468.5	27.9	2101	332.7	4.3	0.0017	**-0.3**	-31%	25.6	586.1	-0.5	77%	481.2
Kodim13	36.8	131	41.3	3.5	0.0224	**0.0**	10%	23.4	146.5	-4.1	468%	485.8	27.4	2023	373.2	4.8	0.0009	**0.0**	-11%	25.0	615.8	-0.8	106%	478.6
Kodim19	38.5	133	26.7	5.2	0.0088	**0.0**	5%	18.3	52.8	-1.4	159%	440.0	29.9	2744	320.9	6.0	0.0018	-0.2	-30%	22.1	474.8	**-0.1**	31%	458.6
Kodim22	38.0	124	30.4	4.8	0.0158	**0.0**	17%	21.3	52.8	-1.4	174%	434.2	29.9	2860	352.9	7.5	0.0008	**0.0**	-8%	23.6	448.8	-0.1	38%	444.5
Kodim23	40.3	180	25.0	6.5	0.0065	**0.0**	-10%	20.2	36.6	-0.7	127%	437.3	32.8	4792	327.5	8.4	0.0014	-0.4	-43%	23.7	423.5	**-0.1**	24%	442.6
Average	*37.7*	*125*	*33.8*	*4.8*	*0.0082*	**-0.2**	*31%*	*17.9*	*76.7*	*-2.1*	*334%*	*379.4*	*29.8*	*3095*	*339.6*	*6.2*	*0.0016*	**-0.3**	*-30%*	*20.7*	*492.9*	*-0.3*	*53%*	*381.1*

Image	**σ_n=50, 1st iteration**												**σ_n=50, 2nd iteration**											
	Best		CSM Fitting						MAD+SURE				Best		CSM Fitting						MAD+SURE			
	PSNR (dB)	$\sigma_r{}^2$	σ^2	α	KLD	ΔPSNR (dB)	$\Delta\sigma_r{}^2$ (rel.)	time (sec)	$\sigma^2{}_{MAD}$	ΔPSNR (dB)	$\Delta\sigma_r{}^2$ (rel.)	time (sec)	PSNR (dB)	$\sigma_r{}^2$	σ^2	α	KLD	ΔPSNR (dB)	$\Delta\sigma_r{}^2$ (rel.)	time (sec)	$\sigma^2{}_{MAD}$	ΔPSNR (dB)	$\Delta\sigma_r{}^2$ (rel.)	time (sec)
Lena	26.6	55409	2216.3	14.3	0.0018	-0.2	-43%	20.1	2523.7	**0.0**	0%	295.7	27.5	439	30.0	5.3	0.0106	**-0.1**	-64%	13.1	23.9	-0.9	-97%	314.4
Baboon	21.6	17733	2459.3	10.4	0.0020	**-0.1**	44%	22.8	2836.9	-0.2	79%	330.6	21.5	69	49.8	4.4	0.0134	-0.1	213%	13.4	50.2	**0.0**	-64%	352.2
Barbara	23.7	25776	2306.5	11.8	0.0014	**0.0**	5%	27.3	2646.8	-0.1	62%	475.0	24.0	159	35.1	4.2	0.0072	**0.0**	-8%	22.3	38.1	-0.2	-89%	515.3
Peppers	26.1	46991	2181.2	11.9	0.0018	-0.3	-45%	20.5	2584.9	**0.0**	16%	306.9	27.2	698	38.6	5.3	0.0091	**-0.2**	-70%	15.4	33.0	-1.2	-97%	292.0
F16	25.8	38353	2120.4	10.4	0.0018	-0.3	-42%	20.4	2523.7	**0.0**	10%	307.5	27.0	561	43.5	5.6	0.0083	**-0.2**	-57%	15.3	41.6	-1.2	-96%	308.7
House	25.6	39298	2172.6	10.8	0.0042	-0.2	-40%	11.8	2584.9	**0.0**	19%	72.8	26.8	586	40.0	5.3	0.0072	**-0.2**	-64%	6.1	39.6	-1.2	-97%	73.7
Kodim04	26.6	54958	2198.3	14.3	0.0015	-0.2	-43%	26.5	2523.7	**0.0**	0%	457.9	27.5	410	31.7	6.0	0.0132	**-0.1**	-54%	21.4	22.4	-0.9	-96%	464.8
Kodim08	22.1	16971	2093.8	6.0	0.0018	-0.2	-26%	25.5	2772.8	**-0.1**	38%	454.7	22.9	668	98.8	4.5	0.0085	**0.0**	-33%	24.3	154.7	-0.7	-93%	477.4
Kodim13	22.2	17463	2154.5	7.2	0.0016	**0.0**	-12%	29.8	2772.8	-0.1	38%	434.1	22.7	319	78.2	4.9	0.0090	**0.0**	19%	22.7	101.6	-0.4	-88%	454.9
Kodim19	24.3	27073	2098.1	8.9	0.0015	-0.1	-31%	29.6	2584.9	**0.0**	13%	464.3	25.2	419	54.7	5.8	0.0082	**0.0**	-24%	20.9	56.8	-0.8	-93%	445.8
Kodim22	25.6	43316	2186.0	12.6	0.0015	-0.1	-37%	25.9	2584.9	**0.0**	26%	494.0	26.2	320	37.4	6.0	0.0123	**0.0**	-30%	19.8	28.8	-0.6	-94%	430.4
Kodim23	27.2	52888	2115.5	12.2	0.0017	-0.4	-51%	28.3	2523.7	**0.0**	0%	446.2	28.6	709	39.2	6.6	0.0127	**-0.2**	-64%	23.6	29.7	-1.6	-97%	459.3
Average	*24.8*	*36352*	*2191.9*	*10.9*	*0.0019*	*-0.2*	*-27%*	*24.0*	*2622.0*	**-0.1**	*25%*	*378.3*	*25.6*	*446*	*48.1*	*5.3*	*0.0100*	**-0.1**	*-20%*	*18.2*	*51.7*	*-0.8*	*-92%*	*382.4*

∗ The PSNR and σ_r^2 in the columns under "Best" are derived by the σ_r^2 scan;
† △PSNR is calculated by subtracting the best PSNR from the PSNR of CSM fitting or MAD+SURE;
‡ $\triangle\sigma_r^2$ is presented in form of relative percentage, i.e. $(\sigma_r^2 - \sigma_{r,best}^2)/\sigma_{r,best}^2$.

TABLE III
SUMMARY FOR YAROSLAVSKY/BILATERAL FILTERS ($\epsilon_{bd} = 0.1/1.0$, GK, $k = 3$, AWGN). THE NUMBERS REPRESENT THE CORRESPONDING AVERAGES FOR THE TWELVE TEST IMAGES. THE RESULTS OF $\epsilon_{bd} = 0.1/1.0$ DIFFER ONLY FOR $\sigma_n = 40/50$.

		1st iteration								Recursive filtering			
		Best	CSM Fitting					MAD+SURE		CSM Fitting			
	σ_n	PSNR (dB)	KLD ($\times 10^{-3}$)		ΔPSNR (dB)		$\lvert\Delta\sigma_r^2\rvert$ (rel.)	ΔPSNR (dB)	$\lvert\Delta\sigma_r^2\rvert$ (rel.)	$\bar{m}$		ΔPSNR (dB)	
			ϵ_{bd} 0.1	ϵ_{bd} 1.0	ϵ_{bd} 0.1	ϵ_{bd} 1.0	ϵ_{bd} 0.1	ϵ_{bd} 1.0			ϵ_{bd} 0.1	ϵ_{bd} 1.0	ϵ_{bd} 0.1	ϵ_{bd} 1.0
YF 5x5	5	37.5	7.9		-0.2		51%	-2.0	377%	1.0		-0.2	
	10	33.4	1.5		-0.1		16%	-0.8	133%	1.1		-0.1	
	20	29.5	1.4		-0.2		25%	-0.3	57%	2.0		0.5	
	40	25.4	1.5	2.0	-0.2	-0.6	30%	43%	-0.1	31%	2.3	2.8	0.9	1.0
	50	24.1	1.5	1.7	-0.1	-0.7	31%	51%	0.0	23%	2.6	3.0	1.1	1.2
YF 9x9	5	37.5	8.8		-0.5		76%	-2.0	300%	1.0		-0.5	
	10	33.4	2.3		-0.1		32%	-0.8	133%	1.2		-0.2	
	20	29.5	1.3		-0.1		19%	-0.3	52%	2.0		0.3	
	40	25.6	1.1	1.4	-0.1	-0.2	22%	28%	-0.1	22%	2.1	2.1	0.6	0.7
	50	24.5	0.9	1.1	-0.1	-0.4	20%	34%	0.0	23%	2.1	2.1	0.6	0.8
BF 9x9	5	37.7	8.2		-0.2		38%	-2.1	334%	1.0		-0.2	
	10	33.7	1.7		-0.1		18%	-0.8	127%	1.2		-0.1	
	20	29.8	1.6		-0.3		30%	-0.3	53%	2.0		0.4	
	40	25.9	1.9	2.4	-0.3	-0.7	35%	45%	-0.1	30%	2.2	2.2	0.7	0.7
	50	24.8	1.9	2.2	-0.2	-1.1	35%	56%	-0.1	25%	2.2	3.0	0.7	0.9
BF 13x13	5	37.7	8.9		-0.3		64%	-2.0	309%	1.0		-0.3	
	10	33.6	1.9		-0.1		21%	-0.9	128%	1.2		-0.1	
	20	29.7	1.6		-0.1		22%	-0.3	48%	2.0		0.3	
	40	25.9	1.4	1.7	-0.2	-0.4	25%	32%	-0.1	25%	2.1	2.1	0.7	0.7
	50	24.7	1.2	1.5	-0.1	-0.6	26%	39%	0.0	22%	2.1	2.1	0.6	0.8

$\triangle$PSNR here is calculated by subtracting PSNR from the best first-iteration PSNR; $|\triangle\sigma_r^2|$ here is presented in form of relative percentage of absolute difference; $\bar{m}$ stands for the average number of iterations.

the CSM parameter optimization, one other reason is that the CPLOW processes color channels separately while we consider all channels jointly.

The visual comparison of CPLOW, CBM3D and **MNLM-DCT** is shown in Fig. 6. The CPLOW has obvious color-misalignment artifacts, while the CBM3D gives the best quality for its processing in the sparse 3-D transform. The **MNLM-DCT** has DCT-basis artifacts which result from the usage of non-sparse 2-D transform. Except this issue, it can deliver the same level of details as the CBM3D.

*D. Range-weighted kernel (**BF-9×9**, $k = 3$, AWGN)*

The kernel functions for $K_r(\cdot)$ in Table I are tested and summarized in Table V. For the GGD4, Epanechnikov, Biweight and Triweight kernels, the upper bound of α in the fitting parameter range $\mathbf{\Psi}$ is set to $20k$ for accommodating larger range variance. The performance of these kernels is very similar to the Gaussian one, and their estimated range variances are also highly correlated as shown in Fig. 7. An alternative way to estimate the corresponding σ_r^2 ratios is by minimizing the L^2 distance to the Gaussian kernel. The optimized ratios in this manner for the GGD4, Epanechnikov, Biweight and Triweight kernels are 1.08, 1.87, 2.83 and 3.81 respectively, which differs from the linear coefficients in Fig. 7 by less than 0.3. This not only shows the adaptability of CSM fitting but also suggests that those kernels can be used interchangeably for denoising under the proposed framework.

TABLE V
SUMMARY FOR GENERALIZED KERNELS (**BF-9×9**, $k = 3$, AWGN).

		1st iteration						Recursive filtering			
		Best	CSM Fitting			MAD+SURE		CSM Fitting			
	σ_n	PSNR (dB)	KLD ($\times 10^{-3}$)	ΔPSNR (dB)	$\lvert\Delta\sigma_r^2\rvert$ (rel.)	ΔPSNR (dB)	$\lvert\Delta\sigma_r^2\rvert$ (rel.)	$\bar{m}$	ΔPSNR (dB)	PSNR (dB)	ΔPSNR$_{GK}$ (dB)
Gaussian	5	37.7	8.2	-0.2	38%	-2.1	334%	1.0	-0.2	37.5	-
	10	33.7	1.7	-0.1	18%	-0.8	127%	1.2	-0.1	33.5	-
	20	29.8	1.6	-0.3	30%	-0.3	53%	2.0	0.4	30.2	-
	40	25.9	1.9	-0.3	35%	-0.1	30%	2.2	0.7	26.7	-
	50	24.8	1.9	-0.2	35%	-0.1	25%	2.2	0.7	25.5	-
Laplacian	5	37.5	37.5	-0.9	166%	-1.9	422%	1.1	-1.0	36.5	-1.1
	10	33.4	13.3	-0.3	72%	-0.8	183%	1.2	-0.5	32.9	-0.7
	20	29.4	5.2	-0.2	46%	-0.3	75%	2.0	0.3	29.6	-0.6
	40	25.6	2.4	-0.2	49%	-0.1	47%	2.3	0.6	26.2	-0.5
	50	24.5	2.1	-0.2	43%	0.0	41%	2.3	0.6	25.2	-0.3
GGD4	5	37.7	12.0	-0.1	23%	-2.0	246%	1.0	-0.1	37.6	0.0
	10	33.7	2.2	-0.1	20%	-0.8	97%	1.1	-0.1	33.6	0.1
	20	30.0	1.2	-0.4	32%	-0.3	44%	2.0	0.1	30.1	-0.1
	40	26.2	1.8	-0.4	31%	-0.1	20%	2.2	0.4	26.6	-0.1
	50	25.0	1.9	-0.2	28%	-0.1	19%	2.1	0.5	25.5	0.0
Epanech-nikov	5	37.7	12.5	-0.1	25%	-1.8	253%	1.0	-0.1	37.6	0.0
	10	33.7	2.4	-0.1	17%	-0.8	99%	1.2	-0.2	33.6	0.0
	20	30.0	2.9	-0.9	41%	-0.2	37%	2.0	-0.1	29.9	-0.3
	40	26.1	2.2	-0.7	34%	-0.1	17%	2.3	0.3	26.4	-0.2
	50	24.9	2.0	-0.3	29%	0.0	14%	2.2	0.5	25.5	0.0
Biweight	5	37.7	10.8	-0.1	25%	-1.9	269%	1.0	-0.1	37.6	0.0
	10	33.7	2.9	-0.2	21%	-0.8	109%	1.2	-0.2	33.5	0.0
	20	29.9	1.8	-0.5	35%	-0.3	43%	2.0	0.2	30.1	-0.1
	40	26.1	2.1	-0.5	35%	-0.1	21%	2.2	0.6	26.7	0.0
	50	24.9	2.0	-0.2	29%	0.0	16%	2.3	0.6	25.5	0.0
Triweight	5	37.7	10.0	-0.1	23%	-1.8	247%	1.0	-0.1	37.6	0.1
	10	33.7	2.9	-0.2	20%	-0.8	115%	1.3	-0.1	33.6	0.0
	20	29.9	2.1	-0.4	32%	-0.3	46%	1.9	0.3	30.1	-0.1
	40	26.0	2.1	-0.4	34%	-0.1	18%	2.2	0.7	26.7	0.0
	50	24.8	1.9	-0.2	25%	0.0	14%	2.1	0.7	25.5	0.0

$\triangle$PSNR, $|\triangle\sigma_r^2|$ and $\bar{m}$ have the same definitions as those in Table III; $\triangle$PSNR$_{GK}$ is calculated by subtracting PSNR from the PSNR of the Gaussian kernel.

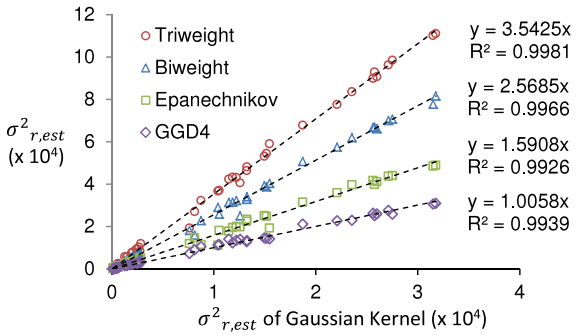


Fig. 7. Correlation of estimated σ_r^2 between Gaussian and other kernels. The Laplacian kernel is not shown for its lower R^2 value 0.9836.

In contrast, the Laplacian kernel has worse PSNR and higher KLD, which indicates it is not suitable for bilateral filtering.

*E. Hyperspectral image (**BF-9×9**, GK, AWGN)*

We use the hyperspectral HYDICE[4] image for testing more color channels. The test results for up to nine channels across the available spectrum are shown in Table VI. One notable effect as k increases is that the fitting KLD goes up either, and this is especially obvious for low-noise cases or the iterations after the first one. It reflects the fact that there is still inconsistency which cannot be modelled well by NNM

[4]https://engineering.purdue.edu/~biehl/MultiSpec/hyperspectral.html

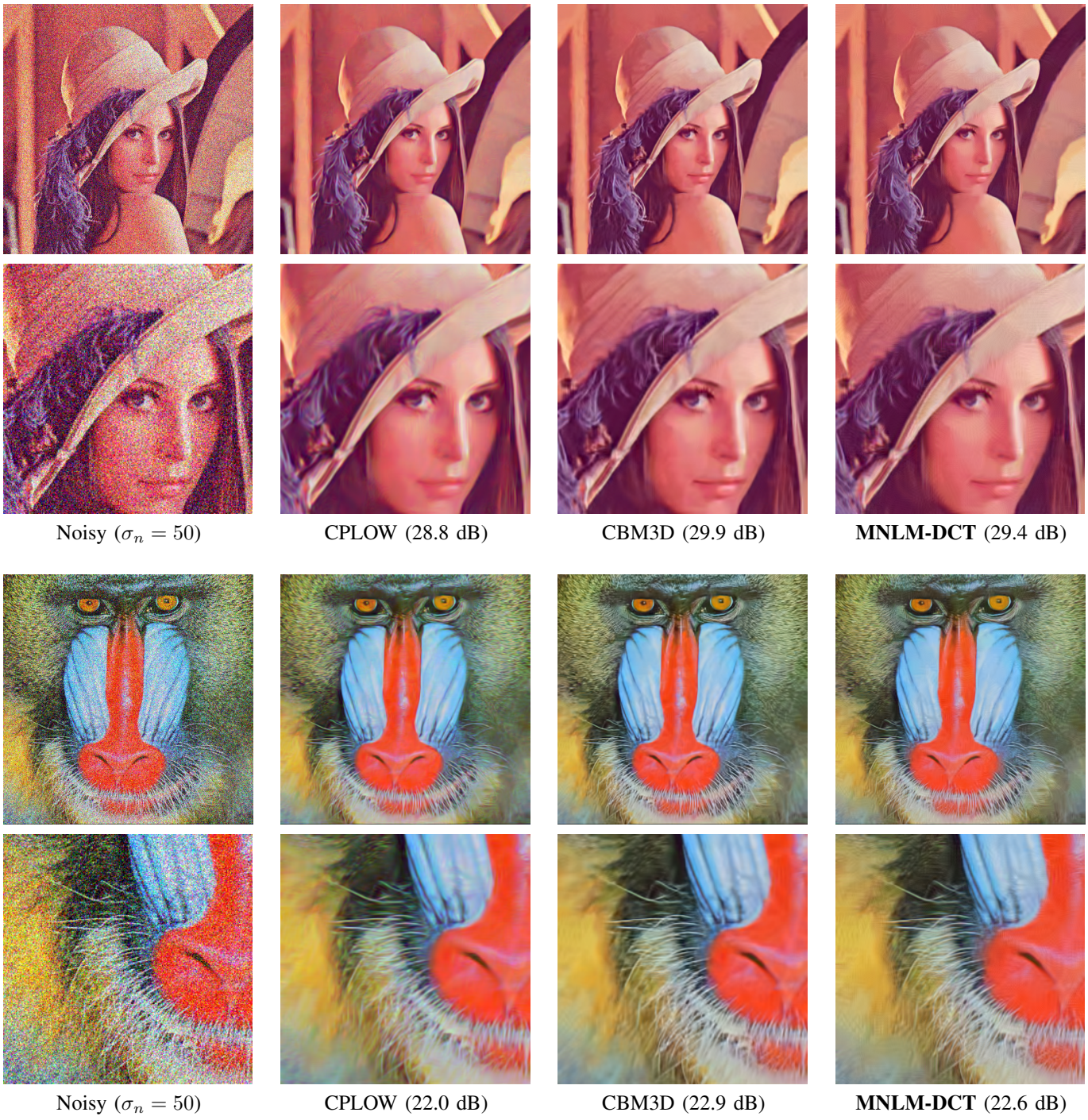


Fig. 6. Visual comparison for the denoising results of CPLOW, CBM3D and **MNLM-DCT** in the case of large AWGN intensity $\sigma_n = 50$. The whole images and cropped regions for *Lena* (top two rows) and *Baboon* (bottom two rows) are shown.

among the latent signals in different channels. To exclude the fitting results with high KLD, we exit the recursive processing when the KLD is higher than 0.3. Compared to using only RGB channels ($k = 3$), the denoising performance using more channels is better for high-noise cases ($\sigma_n \geq 10$) due to having more information of intensity texture, and the range variance is also well tracked. However, the performance for the low-noise case ($\sigma_n = 5$) becomes worse because of the channel inconsistency and thus the over-estimated range variance.

*F. Noise type (***BF-9×9***, GK, $k = 3$)*

The assumption of Gaussian noise in the NNM (4) is for the tractable formulations. However, we have also shown that the CSM fitting still works well for the iterations after the first one, i.e. when the noise is no more Gaussian. In the following, we further apply other noise models to the twelve standard color images to evaluate the effectiveness of our framework when the model is not matched. The results are summarized in Table VII.

For the uniform noise models, the CSM fitting performs sim-

TABLE IV
SUMMARY OF TEST RESULTS FOR MNLM FILTERS. THE RESULTS OF CPLOW AND CBM3D ARE GIVEN FOR REFERENCE.

MNLM -Simple	1st iteration				2nd iteration				Recursive filtering		
	Best	CSM Fitting			Best	CSM Fitting			CSM Fitting		
σ_n	PSNR (dB)	KLD	ΔPSNR$_1$ (dB)	$\lvert\Delta{\sigma_r}^2\rvert_1$ (rel.)	PSNR (dB)	KLD	ΔPSNR$_2$ (dB)	$\lvert\Delta{\sigma_r}^2\rvert_2$ (rel.)	$\bar{m}$	ΔPSNR$_1$ (dB)	PSNR (dB)
5	36.5	0.0047	-0.2	38%					1.4	-0.3	36.2
10	32.5	0.0008	-0.2	29%					1.3	-0.2	32.3
20	28.5	0.0005	-0.1	20%	29.8	0.0048	-0.03	22%	2.3	1.3	29.8
40	23.9	0.0002	0.0	39%	26.6	0.0015	-0.04	29%	3.0	3.2	27.1
50	22.2	0.0002	0.0	51%	25.5	0.0008	-0.03	34%	3.0	3.9	26.1

MNLM -DCT	1st iteration				2nd iteration				Recursive filtering			**CPLOW -MAD**	**CBM3D -MAD**
	Best	CSM Fitting			Best	CSM Fitting			CSM Fitting				
σ_n	PSNR (dB)	KLD	ΔPSNR$_1$ (dB)	$\lvert\Delta{\sigma_r}^2\rvert_1$ (rel.)	PSNR (dB)	KLD	ΔPSNR$_2$ (dB)	$\lvert\Delta{\sigma_r}^2\rvert_2$ (rel.)	$\bar{m}$	ΔPSNR$_1$ (dB)	PSNR (dB)	PSNR (dB)	PSNR (dB)
5	37.6	0.0047	-0.3	330%					1.0	-0.3	37.3	33.6	37.1
10	34.8	0.0008	-0.3	131%					1.0	-0.3	34.4	31.8	34.7
20	31.6	0.0012	-0.2	81%					1.0	-0.2	31.3	29.6	32.0
40	28.1	0.0003	-0.1	47%	28.6	0.0107	-0.01	62%	2.0	0.5	28.6	27.2	29.2
50	27.0	0.0004	-0.1	57%	27.7	0.0068	-0.02	60%	2.0	0.7	27.7	26.4	28.3

$\triangle$PSNR$_1$ and $\triangle$PSNR$_2$ are calculated by subtracting PSNR from the first-iteration and the second-iteration best PSNR respectively; $|\triangle\sigma_r^2|_1$ and $|\triangle\sigma_r^2|_2$ present relative percentages of absolute difference for the first and second iterations respectively.

TABLE VI
SUMMARY FOR HYPERSPECTRAL HYDICE (**BF-9×9**, GK, AWGN).

		1st iteration						Recursive filtering			
		Best	CSM Fitting			MAD+SURE		CSM Fitting			
k	σ_n	PSNR (dB)	KLD (x10^{-3})	ΔPSNR (dB)	$\Delta{\sigma_r}^2$ (rel.)	ΔPSNR (dB)	$\Delta{\sigma_r}^2$ (rel.)	m	ΔPSNR (dB)	PSNR (dB)	ΔPSNR$_{k=3}$ (dB)
3	5	36.3	12.8	-0.1	-34%	-2.9	376%	1.0	-0.1	36.2	-
	10	31.4	6.8	-0.3	-40%	-1.4	159%	1.0	-0.3	31.1	-
	20	26.7	2.3	-0.6	-44%	-0.5	74%	2.0	-0.1	26.5	-
	40	22.5	1.8	0.0	-2%	-0.2	46%	3.0	0.2	22.7	-
	50	21.4	2.0	0.0	-4%	-0.1	31%	3.0	0.2	21.6	-
5	5	36.3	50.8	-0.3	62%	-3.2	435%	1.0	-0.3	36.0	-0.1
	10	31.8	17.0	0.0	1%	-1.6	174%	1.0	0.0	31.8	0.7
	20	27.3	7.6	0.0	-13%	-0.6	69%	2.0	0.1	27.5	0.9
	40	22.9	6.9	-0.1	21%	-0.1	30%	2.0	0.0	22.9	0.2
	50	21.7	6.4	0.0	20%	-0.1	36%	3.0	0.1	21.8	0.2
7	5	36.5	163.7	-1.0	121%	-3.7	464%	1.0	-1.0	35.5	-0.7
	10	32.0	53.4	-0.2	44%	-1.7	197%	1.0	-0.2	31.8	0.8
	20	27.6	14.3	0.0	7%	-0.6	66%	1.0	0.0	27.6	1.0
	40	23.1	16.3	-0.3	49%	-0.2	37%	2.0	-0.4	22.8	0.1
	50	21.8	13.8	-0.2	45%	-0.2	33%	2.0	-0.2	21.6	0.0
9	5	36.6	303.3	-1.3	188%	-3.1	413%	1.0	-1.3	35.3	-0.9
	10	32.3	111.0	-0.4	48%	-1.6	140%	1.0	-0.4	31.8	0.8
	20	27.9	25.9	-0.1	25%	-0.6	62%	1.0	-0.1	27.8	1.2
	40	23.5	15.0	-0.1	31%	-0.2	37%	1.0	-0.1	23.4	0.7
	50	22.2	15.7	-0.1	26%	-0.1	22%	2.0	-0.1	22.1	0.5

For each case, the first k channels are selected from this set (in wavelength, nm): {459(blue), 504(green), 759(red), 401, 953, 1175, 1768, 2200, 2473}.

TABLE VII
SUMMARY FOR DIFFERENT NOISE TYPES (**BF-9×9**, GK, $k = 3$).

	1st iteration						Recursive filtering		
	Best	CSM Fitting			MAD+SURE		CSM Fitting		
Noise Type	PSNR (dB)	KLD (x10^{-3})	ΔPSNR (dB)	$\lvert\Delta{\sigma_r}^2\rvert$ (rel.)	ΔPSNR (dB)	$\lvert\Delta{\sigma_r}^2\rvert$ (rel.)	$\bar{m}$	ΔPSNR (dB)	PSNR (dB)
U5	37.7	7.4	-0.2	42%	-2.2	352%	1.0	-0.2	37.5
U20	29.8	9.1	-0.1	21%	-0.7	113%	2.0	0.4	30.1
U50	24.8	13.6	-0.1	36%	-0.2	79%	2.2	0.5	25.3
POS	32.9	6.9	-0.4	36%	-0.4	67%	1.2	-0.4	32.5
SNP	25.2	86.1	-0.1	775%	-0.2	3460%	1.0	-0.1	25.1

U5/U20/U50: Uniform noise with standard deviation 5/20/50;
POS: Poisson noise with the pixel value as the number of photons;
SNP: Salt and pepper noise with 1% density.

ilarly well compared to the corresponding cases of Gaussian noises in Table V, while the MAD+SURE becomes worse for both PSNR and the accuracy of σ_r^2. For the Poisson noise, the performance of the two methods is similar on average. Regarding the salt and pepper noise, which the bilateral filter is not good at, the CSM fitting happens to suggest small range variances because it treats the sparkles as intensity edges. In contrast, the MAD+SURE still suggests higher range variances and leads to more blurry images. The above experiments demonstrate the robustness of the proposed framework across different noise models.

G. Natural image gradient (GK, $k = 3$)

Given the definition of the observable s with the simplest four-connected neighborhood, the NNM is then equivalent to a model for image gradients. For noisy images, the CSM can fit as well as **YF-5×5** and **YF-9×9** do. A more interesting question is how the CSM fitting performs for natural images since image noise is inevitable during picture capturing. We summarize the fitting results for the test color images in Table VIII and show the best and worst cases in Fig. 8. Generally speaking, the main bodies, where $P(s) \geq 10^{-4}$, can be fit well, but the tails of the CSM drop faster than the heavy tails of the empirical distributions. Therefore, if the deviation of the tails is acceptable, the proposed framework can be applied to modelling natural images and also inferring novel algorithms which use gradients as important cues.

VII. DISCUSSION

A. Robustness of CSM fitting over SURE

The robustness of CSM fitting over SURE mainly comes from its independent MAP framework for each range weight $w_{l,i}$. It estimates the range variance based on a simple assumption, range kernel $K_r(x)$ is invertible, and decouples itself from the following ML estimation for the adaptive filter

TABLE VIII
SUMMARY FOR CSM FITTING TO COLOR-IMAGE GRADIENTS
(FOUR-CONNECTED NEIGHBORHOOD, GK, $k = 3$).

Image	σ^2	α	KLD
Lena	7.2	4.6	0.0293
Baboon	37.6	3.1	0.0265
Barbara	3.3	4.0	0.0420
Peppers	18.6	6.0	0.0271
F16	1.9	4.9	0.1098
House	3.0	4.3	0.1711
Kodim04	1.3	4.0	0.0878
Kodim08	2.7	4.1	0.1269
Kodim13	2.5	3.0	0.1135
Kodim19	1.9	4.7	0.0935
Kodim22	2.2	4.0	0.0852
Kodim23	1.4	5.6	0.0999
Average	*7.0*	*4.4*	*0.0844*

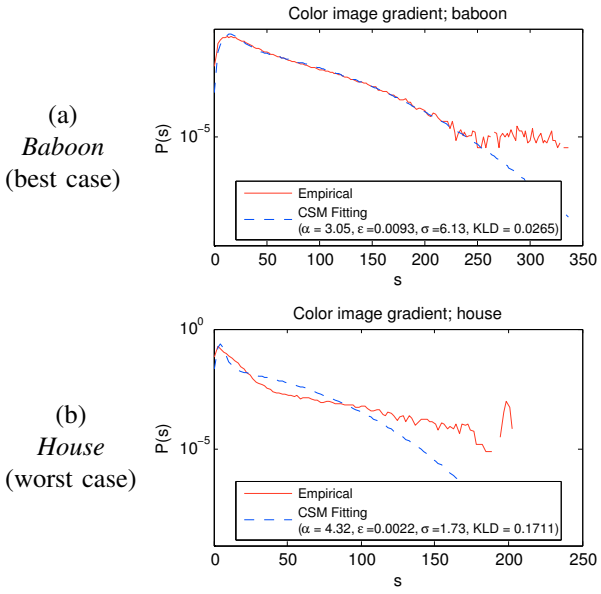


Fig. 8. CSM fitting results for natural image gradients.

kernel. Therefore, it works well even when the neighborhood is dynamically allocated for the non-local filters. On the other hand, the SURE-based method explicitly relies on the weakly differentiable filter kernel assumption to derive an accurate risk estimation, so it fails in this case. Similarly, the SURE accuracy also counts on the independent Gaussian noise assumption, and thus it does not work for the filtered images in which the noise is neither independent nor Gaussian. In contrast, the MAP estimation for the CSM fitting is accurate as long as the fitting is good with a small KLD value.

B. Limitations of this framework

The effectiveness of the proposed framework depends on if the empirical distribution $P(s)$ can be explained well by the CSM model. And we use KLD as the numerical assessment for this explanatory capability. The denoising experiments in Section VI work well thanks to their small KLD values. However, this framework may fail if KLD is large, i.e. CSM cannot fit $P(s)$ well. For example, this may happen when a very large neighborhood size $|\Lambda|$ is used for local or non-local filters. Therefore, examining the KLD value is necessary before applying this framework to a specific problem.

C. Possible future extensions

Range-weighted formulation is a very useful and widely adopted tool in many computer vision and signal processing problems, but mostly applied in an intuitional way. This paper presents how to systematically solve this formulation for the denoising problem in an empirical Bayesian way. We believe that this approach can be extended to many other algorithms using similar range-weighted formulation, such as cross bilateral filter [4] and stereo matching (especially for [5] and more recently [32], [33]). Besides, we show that the NNM can capture the statistics of the image gradient in Section VI-G. Thus, this framework can also be extended to construct image prior and estimate its parameters for regularization of image restoration problems, e.g. for those using sparse representation.

VIII. CONCLUSION

In this paper, we propose and study a unified empirical Bayesian framework which can both infer the neighborhood filters and estimate the range variance. With the neighborhood noise model, we show that the Yaroslavsky, bilateral, and MNLM filters can be derived by joint MAP and ML optimization. We also present an EM+ algorithm for parameter estimation via fitting the observable CSM. The extensive experimental results on image denoising show the effectiveness and robustness for a variety of application scenarios. The noisy images can be fit well and the range variance can be tracked as accurate as the multi-pass SURE-based method. Moreover, the CSM fitting also works for filtered images, and this enables a recursive filtering scheme and improves PSNR.
Instead of heuristic tuning for the essential range variance, the proposed framework can be used to build efficient filters automatically for different constraints, e.g. different filter types/supports, patch sizes, and color channel numbers. It can also be expected that it will be applied to other range-weighted algorithms by formulating the corresponding likelihood functions or modelling the image gradients. Therefore, we believe that it will help many computer vision and signal processing problems be solved in an empirical Bayesian way, instead of an intuitive way.

APPENDIX A
SURE FOR GENERALIZED KERNEL $K_r(x)$

By extending the analytical expressions of [22] for the local filter and k-channel signals in (1), we have

$$\text{SURE} = \frac{1}{kI}\sum_{l\in\mathcal{I}} \| \mathbf{y}_l - \hat{\mathbf{z}}_l \|_2^2 - \sigma^2 + \frac{2\sigma^2}{kI}\sum_{l\in\mathcal{I},c\in\mathcal{K}} \frac{\partial \hat{z}_{l,c}}{\partial y_{l,c}}, \tag{30}$$

$$\frac{\partial \hat{z}_{l,c}}{\partial y_{l,c}} = \frac{1}{W_l}\left(d_{l,l} + \sum_{i\in\Lambda_l} d_{l,i}\frac{\partial w_{l,i}}{\partial y_{l,c}}(y_{i,c} - \hat{z}_{l,c}) \right), \tag{31}$$

where $\mathcal{I}$ represents the pixel array, $I = |\mathcal{I}|$, $\mathcal{K} = \{1, 2, .., k\}$, and $W_l = \sum_{i \in \Lambda_l} w_{l,i} d_{l,i}$. Given $w_{l,i} = K_r(x = \frac{\|\mathbf{y}_l - \mathbf{y}_i\|_2^2}{2\sigma_r^2})$ and $r(w) \triangleq -K_r'(K_r^{-1}(w))$, we can have

$$\frac{\partial w_{l,i}}{\partial y_{l,c}} = K_r'(x)\frac{y_{l,c} - y_{i,c}}{\sigma_r^2} = r(w_{l,i})\frac{y_{i,c} - y_{l,c}}{\sigma_r^2}. \tag{32}$$

Then combining the above equations gives the SURE formulation for generalized kernels:

$$\begin{aligned}\text{SURE} = &\frac{1}{kI}\sum_{l \in \mathcal{I}} \| \mathbf{y}_l - \hat{\mathbf{z}}_l \|_2^2 - d_{l,l} \cdot \sigma^2 \\ &+ \frac{2\sigma^2}{I}\sum_{l \in \mathcal{I}} \frac{1}{W_l}\left(\frac{p_l}{k\sigma_r^2} + 1\right),\end{aligned} \tag{33}$$

where $p_l \triangleq \sum_{i \in \Lambda_l, c \in \mathcal{K}} d_{l,i} r(w_{l,i})(y_{i,c} - y_{l,c})(y_{i,c} - \hat{z}_{l,c})$. The cases of $r(w)$ used in this paper can be found in Table I.

REFERENCES

[1] L. P. Yaroslavsky, *Digital Picture Processing - An Introduction*. Springer Verlag, 1985.

[2] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *Proc. IEEE ICCV*, 1998, pp. 839–846.

[3] P. Choudhury and J. Tumblin, "The trilateral filter for high contrast images and meshes," in *ACM SIGGRAPH 2005 Courses*.

[4] G. Petschnigg, R. Szeliski, M. Agrawala, M. Cohen, H. Hoppe, and K. Toyama, "Digital photography with flash and no-flash image pairs," in *ACM SIGGRAPH 2004 Papers*, pp. 664–672.

[5] K.-J. Yoon and I. S. Kweon, "Adaptive support-weight approach for correspondence search," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 4, pp. 650–656, Apr. 2006.

[6] S. Paris, P. Kornprobst, J. Tumblin, and F. Durand, "Bilateral filtering: Theory and applications," in *Foundations and Trends in Computer Graphics and Vision*, vol. 4, no. 1, 2008, pp. 1–73.

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

[8] D. Barash and D. Comaniciu, "A common framework for nonlinear diffusion, adaptive smoothing, bilateral filtering and mean shift," *Image and Vision Computing*, vol. 22, no. 1, pp. 73–81, Jan. 2004.

[9] A. Buades, B. Coll, and J.-M. Morel, "The staircasing effect in neighborhood filters and its solution," *IEEE Trans. Image Process.*, vol. 15, no. 6, pp. 1499–1505, Jun. 2006.

[10] M. Elad, "On the origin of the bilateral filter and ways to improve it," *IEEE Trans. Image Process.*, vol. 11, no. 10, pp. 1141–1151, Oct. 2002.

[11] L. Caraffa, J.-P. Tarel, and P. Charbonnier, "The guided bilateral filter: When the joint/cross bilateral filter becomes robust," *IEEE Trans. Image Process.*, vol. 24, no. 4, pp. 1199–1208, Apr. 2015.

[12] P. Milanfar, "A tour of modern image filtering: New insights and methods, both practical and theoretical," *IEEE Signal Process. Mag.*, vol. 30, no. 1, pp. 106–128, Jan. 2013.

[13] C. Kervrann and J. Boulanger, "Optimal spatial adaptation for patch-based image denoising," *IEEE Trans. Image Process.*, vol. 15, no. 10, pp. 2866–2878, Oct. 2006.

[14] C. M. Stein, "Estimation of the mean of a multivariate normal distribution," *The Annals of Statistics*, vol. 9, no. 6, pp. 1135–1151, 1981.

[15] C.-T. Huang, "Bayesian inference for neighborhood filters with application in denoising," in *Proc. IEEE CVPR*, 2015, pp. 1657–1665.

[16] J. Portilla, V. Strela, M. J. Wainwright, and E. P. Simoncelli, "Image denoising using scale mixtures of gaussians in the wavelet domain," *IEEE Trans. Image Process.*, vol. 12, no. 11, pp. 1338–1351, Nov. 2003.

[17] M. J. Wainwright and E. P. Simoncelli, "Scale mixtures of Gaussian and the statistics of natural images," *Advaned Neural Information Processing Systems*, vol. 12, pp. 855–861, May 2000.

[18] C. Liu, R. Szeliski, S. B. Kang, C. L. Zitnick, and W. T. Freeman, "Automatic estimation and removal of noise from a single image," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 2, pp. 299–314, Feb. 2008.

[19] P. Chatterjee and P. Milanfar, "Patch-based near-optimal image denoising," *IEEE Trans. Image Process.*, vol. 21, no. 4, pp. 1635–1649, Apr. 2012.

[20] H. Peng and R. Rao, "Bilateral kernel parameter optimization by risk minimization," in *Proc. IEEE Int. Conf. Image Processing*, 2010, pp. 3293–3296.

[21] H. Peng, R. Rao, and S. A. Dianat, "Multispectral image denoising with optimized vector bilateral filter," *IEEE Trans. Image Process.*, vol. 23, no. 1, pp. 264–273, Jan. 2014.

[22] D. V. D. Ville and M. Kocher, "SURE-based non-local means," *IEEE Signal Process. Lett.*, vol. 16, no. 11, pp. 973–976, Nov. 2009.

[23] ——, "Nonlocal means with dimensionality reduction and SURE-based parameter selection," *IEEE Trans. Image Process.*, vol. 20, no. 9, pp. 2683–2690, Sep. 2011.

[24] Y. Chen and K. J. R. Liu, "Image denoising games," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 23, no. 10, pp. 1704–1716, Oct. 2013.

[25] H. Kishan and C. S. Seelamantula, "SURE-fast bilateral filters," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing*, 2012, pp. 1129–1132.

[26] L. Yaroslavsky, K. Egiazarian, and J. Astola, "Transform domain image restoration methods: review, comparison, and interpretation," in *Nonlinear Image Processing and Pattern Analysis XII*, ser. Proc. SPIE, vol. 4304, 2001, pp. 155–169.

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

[28] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman, "Non-local sparse models for image restoration," in *Proc. IEEE ICCV*, 2009, pp. 2272–2279.

[29] S. Gu, L. Zhang, W. Zuo, and X. Feng, "Weighted nuclear norm minimization with application to image denoising," in *Proc. IEEE CVPR*, 2014, pp. 2862–2869.

[30] D. P. McMillen and J. F. McDonald, "Locally weighted maximum likelihood estimation: Monte carlo evidence and an application," in *Advances in Spatial Econometrics*. Springer Berlin Heidelberg, 2004, pp. 225–239.

[31] M. Jamshidian and R. I. Jennrich, "Acceleration of the EM algorithm by using quasi-newton methods," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 59, no. 3, pp. 569–587, 1997.

[32] C. Kim, H. Zimmer, Y. Pritch, A. Sorkine-Hornung, and M. Gross, "Scene reconstruction from high spatio-angular resolution light fields," *ACM Transactions on Graphics*, vol. 32, no. 4, pp. 73:1–73:12, 2013.

[33] C. Chen, H. Lin, Z. Yu, S. Bing Kang, and J. Yu, "Light field stereo matching using bilateral statistics of surface cameras," in *Proc. IEEE CVPR*, Jun. 2014.

Chao-Tsung Huang (M'11) received the B.S. degree in electrical engineering and the Ph.D. degree in electronics engineering from the National Taiwan University, Taiwan, in 2001 and 2005 respectively.

He is now with the National Tsing Hua University, Taiwan, as an Assistant Professor. From 2005 to 2011, he was with the Novatek Microelectronics Corp., Taiwan, for developing multi-standard image and video codecs. He performed postdoctoral research on an HEVC decoder chip at Massachusetts Institute of Technology, Cambridge, MA, USA, from March 2011 to August 2012. He then worked on light-field camera design as his postdoctoral research at National Taiwan University, Taiwan, until July 2013.

His research interests include low-level computer vision and light-field signal processing, especially from algorithm exploration to VLSI architecture design, chip implementation, and demo system.

Dr. Huang serves as an Associate Editor for the *Springer Circuits, Systems and Signal Processing (CSSP)*. He was a recipient of the MediaTek Fellowship from 2003 to 2005.