Fast Domain Decomposition for Global Image Smoothing

Youngjung Kim, *Student Member, IEEE*, Dongbo Min, *Senior Member, IEEE*, Bumsub Ham, *Member, IEEE*, and Kwanghoon Sohn, *Senior Member, IEEE*

***Abstract*— Edge-preserving smoothing (EPS) can be formulated as minimizing an objective function that consists of data and regularization terms. At the price of high-computational cost, this global EPS approach is more robust and versatile than a local one that typically has a form of weighted averaging. In this paper, we introduce an efficient decomposition-based method for global EPS that minimizes the objective function of L_2 data and (possibly non-smooth and non-convex) regularization terms in linear time. Different from previous decomposition-based methods, which require solving a large linear system, our approach solves an equivalent constrained optimization problem, resulting in a sequence of 1-D sub-problems. This enables applying fast linear time solver for weighted-least squares and -L_1 smoothing problems. An alternating direction method of multipliers algorithm is adopted to guarantee fast convergence. Our method is fully parallelizable, and its runtime is even comparable to the state-of-the-art local EPS approaches. We also propose a family of fast majorization–minimization algorithms that minimize an objective with non-convex regularization terms. Experimental results demonstrate the effectiveness and flexibility of our approach in a range of image processing and computational photography applications.**

***Index Terms*— Edge-preserving image smoothing, joint image filtering, weighted-least squares, alternating minimization, majorization-minimization algorithm.**

I. Introduction

EDGE-preserving smoothing has attracted a strong interest in the fields of image processing and computational photography. Predominantly, it appears in a manipulation task that decomposes an image into a piecewise smooth layer and a detail layer. These layered signals are then recombined to meet various application goals, e.g., detail enhancement, HDR tone mapping, and contrast manipulation [1]. Recent works on joint smoothing provide a new paradigm, enabling various applications such as dense correspondence [2], joint upsampling [3], and texture removal [4]. The problems of segmentation [5], visual saliency [6], alpha matting [7], and haze removal [8] may also be interpreted as joint smoothing tasks. The basic idea of joint smoothing is to provide structural guidance indicating how the smoothing should be performed, assuming structural correlation between different kinds of feature maps, e.g., depth/color and flash/no-flash images.

A significant amount of work has been developed for EPS. We roughly classify existing methods into two groups: local filtering-based methods[1] and global optimization-based methods. The first group is highly related to the local statistics of an input image [10]. The weighted average filter computes an output using a mean value of the local distribution that is typically estimated by the Gaussian kernel. The early work in this class includes the bilateral filter [12], independently proposed in [13] and its extension to joint smoothing [14]. Similarly, the weighted median filter outputs a filtering result which reaches half of the local cumulative distribution [2], and the weighted mode filter [11] aims to find a global mode of the local distribution. Several techniques have been proposed either to accelerate filtering-based methods [15], [16], or to introduce fast alternatives [17], [18] of performing local EPS. These methods are efficient and often show real-time applications. However, they are less appropriate in preserving image details at arbitrary scale, and are not directly applicable to advanced image editing tasks [1]. In this context, global optimization-based methods are advocated in some applications. They find an optimal solution of an objective function that consists of a data fidelity term and a regularization term. Thanks to such global formulation, the optimization-based methods show the state-of-the-art performance compared with local EPS approaches. Such an outperformance is, however, achievable only with the high computational cost, mainly arising from solving the global objective function. The optimization-based methods are still an order of magnitude slower than local ones, even with recent acceleration techniques [19]–[21]. Recent progress in hardware is expected to accelerate the global EPS, but it becomes non-trivial and does not scale well when an image resolution increases.

In this paper, we formulate a global EPS (e.g., based on weighted-least squares or weighted-L_1 smoothing) as an equivalent constrained optimization problem, which is solved by an alternating minimization method [23].

Manuscript received December 1, 2016; revised April 17, 2017; accepted May 23, 2017. Date of publication June 1, 2017; date of current version June 23, 2017. This work was supported in part by the Institute for Information and communications Technology Promotion Grant through the Korea Government (MSIP) under Grant 2016-0-00197 and in part by the National Research Foundation of Korea through the MSIP under Grant 2017R1C1B2005584. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Xudong Jiang. *(Corresponding author: Kwanghoon Sohn.)*

Y. Kim, B. Ham, and K. Sohn are with the School of Electrical and Electronic Engineering, Yonsei University, Seoul 120–749, South Korea (e-mail: read12300@yonsei.ac.kr; mimo@yonsei.ac.kr; khsohn@yonsei.ac.kr).

D. Min is with the School of Computer Science and Engineering, Chungnam National University, Daejeon 305–764, South Korea (e-mail: dbmin@cnu.ac.kr).

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

Digital Object Identifier 10.1109/TIP.2017.2710621

[1] Actually, the filtering-based methods can be formulated as a local optimization problem [9]. We, however, will keep the term "filtering" by referring to literatures.

The computational efficiency is achieved by a kind of decomposition techniques. However, unlike the previous methods [20], [21], [39], our formulation decomposes the original optimization problem into a sequence of 1D sub-problems. This enables using a highly efficient algorithm in both weighted-least squares (WLS) and -L_1 (WL1) smoothing, leading to a time complexity linear to an image size. Another appealing aspect is that our method converges within only few iterations. We found that 5 iterations are typically enough to get satisfactory results both for the WLS and the WL1 smoothing. We will verify this statement through an extensive experimental analysis. Fast majorization-minimization (MM) algorithms are also introduced to optimize an objective function using a non-convex regularization term. The main contributions of this work are summarized as follows:

- We propose a new domain decomposition technique that formulates a global EPS as an equivalent constrained optimization problem. The proposed method is much faster than previous decomposition-based methods.
- We introduce fast MM algorithms using a non-convex regularization term.
- A comprehensive experimental analysis is presented to demonstrate the performance of the proposed method.

The remainder of this paper is organized as follows. Section II describes related works for EPS. Section III provides some background and the problem statement. We present the proposed method in Section IV. An extensive experimental comparison is then provided in Section V. In Section VI, we apply our method to a few image processing and computational photography applications. Finally, Section VII concludes this paper.

II. Related Work

In this section, we review EPS techniques. Weighted-average filters, such as the bilateral [12] and SUSAN [13] filters, are the most intuitive ones among local EPS methods. The input image is implicitly regularized by estimating the mean distribution within a local window [10]. These have been widely used in image abstraction [28], detail enhancement [29], and noise reduction [30]. The local distribution (or histogram) can be modified to jointly reflect the statistics of both the input and guidance images, which contributes to new applications [14]. However, the brute-force implementation of weighted-averaging process generally needs a high computational load. Numerous methods have been proposed for accelerating the bilateral filter with quantization [15] or coarsening [31]. The guided filter [17], domain transform filter [18], and adaptive manifolds filter [32] are popular and efficient alternatives. These methods can imitate a similar smoothing effect to or produce even better smoothing quality than the bilateral filter [12], while keeping a linear-time complexity with the image size. Recently, many researchers have attempted to accelerate weighted-median and -mode filters. Ma *et al.* built 3D histogram by repeatedly applying weighted-average filters depending on the number of bins, and then found a median value for each pixel on the histogram [2]. An efficient histogram representation, called median tracking, with a special data structure was proposed in [33] for fast weighted median filtering. The local EPS methods are simple and easy to implement, but have some drawbacks due to their local nature [1]. They do not maintain the global consistency of the smoothing result.

On the other hand, the optimization-based methods aim at smoothing insignificant details while preserving edges by optimizing a global objective function. Farbman *et al.* proposed an edge-preserving operator based on the WLS regularization [1] for computational photography applications. The smoothing output is obtained by solving a large linear system, where a Laplacian matrix is defined by the given guidance image. Similar methods have been proposed for image colorization [34], matting [7], segmentation [3], and edit propagation [35]. Other popular global EPS approaches employ L_1 norm. Wang *et al.* used an L_1-regularized objective function, and derive an efficient algorithm based on half-quadratic minimization [20]. This method is further extended in [36] with superpixels to enforce group sparsity. Some recent methods used non-convex regularization terms, which requires a different solver. Xu *et al.* formulated a texture extraction problem using a relative L_1 (total variation) regularization, and adopted a fixed-point iteration to solve the corresponding optimization problem [4]. In [37] and [38], Welsch's norm was employed to handle structural differences between guidance and input images in the joint smoothing tasks. Xu *et al.* [39] presented a method for L_0 gradient minimization, favoring piecewise constant solutions. They achieved a good approximation of the L_0 norm by iteratively applying a hard thresholding operator.

There were many attempts to accelerating local EPS method for weighted-average, -median, and -mode filters, but less effort has been made for global EPS approaches. Preconditioning techniques have been used to accelerate the WLS smoothing [19]. Although they greatly reduce the iteration number required to solve a linear system, the cost of constructing preconditioners is considerable. Other global EPS approaches, involving non-quadratic and non-smooth regularization terms, rely on decomposition techniques on objective function [20], [39] or solve a series of linear systems iteratively [4], [37], [38]. The former yields tractable sub-problems but requires a lot of iterations to converge, while the latter shares similar complexity issues with the WLS smoothing. We note that recently Min *et al.* [22] introduced a fast approach for global EPS by iteratively applying 1D WLS regularization. However, it is unclear what objective function is actually minimized by their algorithm. The method [22] thus cannot be generalized to fast MM algorithms.

III. Problem Statement and Background

EPS can be formulated as seeking a minimum of a global objective function, defined from norms over image gradients. We can find the solution by using popular iterative methods [40] or decomposition-based approaches [20], [21], depending on the regularization terms used in the objective function. We start with a basic formulation for EPS to provide some intuition. In the following, the subscript p denotes the location of a pixel (in a vector form).

Given an input image f and a guidance image g, a desired output u is obtained by minimizing the following objective

function:

$$E(u) = \sum_p \Big((u-f)_p^2 + \lambda \sum_{j\in\{1,2\}} w_{j,p}\phi(D_j u)_p \Big), \quad (1)$$

where D_1 and D_2 are discrete implementations of the derivative in horizontal and vertical directions, respectively. The parameter λ controls the amount of the spatial smoothing. g can be the input image f or a different guidance signal correlated with f. The weight $w_{j,p} = \exp(-(D_j g)_p^2/\kappa)$ is defined using g and bandwidth κ. Note that various guidance signals and distance metrics can be used to define w. The regularization function ϕ and the weight function w allow various image priors that behave differently in preserving or smoothing image features. We always assume that f and g have the same width (W) and the height (H). When $\phi = \tau^2$, the objective function of (1) becomes quadratic, and it corresponds to the WLS smoothing [1]. Minimizing (1) with respect to u satisfies the following linear system:

$$\Big(\mathbf{I} + \lambda \sum_{j\in\{1,2\}} \mathbf{D}_j^T \mathbf{W}_j \mathbf{D}_j \Big) u = f, \quad (2)$$

where $\mathbf{D}_j$ is a discrete difference matrix, $\mathbf{W}_j$ is a diagonal matrix containing the weights w_j, and $\mathbf{I}$ is an identity matrix. Iterative solvers like the Jacobi, Successive Over-Relaxation (SOR), and Conjugate Gradient (CG) methods are applicable to solve the linear system of (2), but these are too slow to converge [19]. Despite recent progress in preconditioning techniques, the basic iterative solvers are still an order of magnitude slower than the local weighted-average filters [19]. It is mainly because the cost of constructing the preconditioner may outweigh the speed gain from the improved conditioning. To our best knowledge, no previous work has used the decomposition technique for the WLS smoothing.

EPS can also be performed by non-quadratic optimization. The WL1 regularizer, $\phi = |\tau|$, is a well-known edge-preserving operator, and often provides a better smoothing capability along object boundaries. In general, the corresponding optimization problem is solved based on the decomposition on objective function (DOF) [20], alternating between a quadratic model and soft thresholding. The basic idea is to split the data and regularization terms, as follows:

$$\min_{\substack{u,v \\ \text{s.t.} D_j u = v_j}} \sum_p \Big((u-f)_p^2 + \lambda \sum_{j\in\{1,2\}} w_{j,p}\phi(v_j)_p \Big), \quad (3)$$

An auxiliary variable v is introduced to decouple the calculation of the two terms. The constrained problem of (3) is then addressed by iteratively updating variables u, v [20]. Since v has an analytical solution, the main computation lies in the fast fourier transform (FFT) to compute u [20], [21], [39]. The computational cost required for solving (3) is thus $O(n \log n)$ per iteration ($n = HW$). Moreover, the DOF is slow to converge (see Section V).

Recently, Ham *et al.* [37] proposed the SD filter with the non-convex Welsch's norm, which is solved by the MM algorithm [44]. They iteratively compute a quadratic upper bound and solve the corresponding WLS objective as in (2). This procedure guarantees a local minimum of (1) with the non-convex ϕ [37]. However, since an intermediate weighting matrix of the MM algorithm varies during iterations, a series of preconditioners should be constructed, leading to a huge computational overhead.

IV. Proposed Method

In this section, we propose a fast and linear time algorithm for global EPS with either convex or non-convex ϕ. We first introduce an efficient alternative of minimizing (1) when $\phi = \tau^2$ or $|\tau|$. This approach is then extended to solve the non-convex objective functions with the MM algorithm. The key idea is to decompose (1) on each spatial domain using an auxiliary variable, and to use a constrained optimization technique.

Let us consider the following optimization problem with linear equality constraint:

$$\min_{\substack{u,v \\ \text{s.t.u=v}}} \sum_p \Big\{ \frac{1}{2}(u-f)_p^2 + \frac{1}{2}(v-f)_p^2 + \lambda \left(w_{1,p}\phi(D_1 u)_p + w_{2,p}\phi(D_2 v)_p \right) \Big\}. \quad (4)$$

The problem in (4) is equivalent to the original problem of (1) inside the feasible set, $\{u = v\}$. The rationale behind our formulation is that when $u = v$, the problem of (4) is decomposable with respect to each spatial domain. This property will lead to a sequence of sub-problems that is easier to solve than the conventional DOF technique [20], [21], introduced in Section III. We adopt the augmented Lagrangian method to transform the constrained optimization of (4) into an unconstrained one with the Lagrangian and an augmented penalty term. Formally, we minimize the objective function of the form:

$$\begin{aligned} &\min_{u,v,\gamma} E_{\mathrm{AL}}(u, v, \gamma) \\ &= \sum_p \Big\{ \frac{1}{2}(u-f)_p^2 + \frac{1}{2}(v-f)_p^2 + \frac{\beta}{2}(u-v-\gamma)_p^2 \\ &\quad + \lambda \Big(w_{1,p}\phi(D_1 u)_p + w_{2,p}\phi(D_2 v)_p \Big) \Big\}, \end{aligned} \quad (5)$$

where γ is the augmented Lagrangian multiplier. β is the parameter to penalize the difference between u and v. In general, it is difficult to find the minimizer of u, v exactly since they are coupled to each other. We use the accelerated alternating direction method of multipliers (ADMM) [23] to solve (5).[2]

$$\begin{aligned} u^k &= \arg\min_u E_{\mathrm{AL}}(u, \hat{v}^k, \hat{\gamma}^k) \\ v^k &= \arg\min_v E_{\mathrm{AL}}(u^k, v, \hat{\gamma}^k) \\ \gamma^k &= \hat{\gamma}^k - (u^k - v^k), \end{aligned} \quad (6)$$

where $\hat{v}^{k+1} = v^k + \frac{\alpha^k - 1}{\alpha^{k+1}}(v^k - v^{k-1})$, $\hat{\gamma}^{k+1} = \gamma^k + \frac{\alpha^k - 1}{\alpha^{k+1}}(\gamma^k - \gamma^{k-1})$, extrapolated version of v and γ, and $\alpha^{k+1} = (1 + \sqrt{1 + 4(\alpha^k)^2})/2$. The iteration of (6) is different from the standard ADMM method [21], in the sense that a Nesterov-type extrapolation [27] is applied to accelerate the algorithm.

[2]When ϕ is convex the iterative procedure of (6) guarantees quadratic convergence for the optimal value of γ^* [23].

At each iteration, we increase β by factor of $\varepsilon > 1$. These procedures are all point-wise operations, except updating u^k and v^k. Now we investigate how the variables u^k and v^k can be computed efficiently.

A. The $u - v$ Sub-Problems

Assuming that $\hat{v}^k$ (or u^k) and $\hat{\gamma}^k$ are fixed, we solve (6) with respect to u (or v), which yields

$$\min_u \sum_p \Big((u - \tilde{f})_p^2 + \frac{2\lambda}{1+\beta} w_{1,p} \phi(D_1 u)_p \Big), \tag{7}$$

$$\min_v \sum_p \Big((v - \bar{f})_p^2 + \frac{2\lambda}{1+\beta} w_{2,p} \phi(D_2 v)_p \Big), \tag{8}$$

Here, $\tilde{f} = (1+\beta)^{-1}(f + \beta(\hat{v}^k + \hat{\gamma}^k))$ and $\bar{f} = (1+\beta)^{-1}(f+\beta(u^k - \hat{\gamma}^k))$.[3] As D_1 represents a difference operator with respect to the horizontal axis, (7) can be decomposed into a set of 1D sub-problems defined with horizontal signals only. By introducing a 1D slack variable z, we have:

$$u^{k,h} = \arg\min_z \sum_x \Big((z - \tilde{f}^h)_x^2 + \frac{2\lambda}{1+\beta} w_{1,x}^h \phi(D_1 z)_x \Big), \tag{9}$$

The super-script h denotes a horizontal signal along the x dimension ($x = 1, \ldots, W$). This 1D optimization process is repeated for all horizontal signals (H in number). A similar result can be obtained for the v. In this case, (8) is decomposed into 1D sub-problems with vertical signals along the y dimension ($y = 1, \ldots, H$).

Note that the decomposition technique has been applied to decouple data and regularization terms [20], [39], [41], i.e., $[D_1 u, D_2 u] = [v_1, v_2]$, resulting in $O(n \log n)$ complexity algorithm ($n = HW$). This aims at transferring $D_1 u$ and $D_2 u$ out of the regularization function ϕ by introducing the auxiliary variables v_1 and v_2, respectively. In contrast, we introduce v to decompose the original problem (1) into a series of 1D sub-problems. The proposed method not only leads to easily solvable sub-problems, but also significantly improves the convergence rate of the algorithm. Next, we present efficient algorithms for solving the problem (9) defined with a 1D horizontal signal. Its vertical counterpart can be optimized in the same manner.

B. Fast 1D Solvers

1) WLS Smoothing: When $\phi = \tau^2$, (9) is read with a 1D horizontal signal $\tilde{f}^h$ and a guide signal g^h as:

$$\min_z \sum_x \Big((z - \tilde{f}^h)_x^2 + \frac{2\lambda}{1+\beta} w_{1,x}^h (D_1 z)_x^2 \Big). \tag{10}$$

The 1D output z that satisfies the above equation is obtained by solving the following linear system of size $W \times W$.

$$\Big(\mathbf{I} + \frac{2\lambda}{1+\beta} \mathbf{D}_1^T \mathbf{W}_1^h \mathbf{D}_1 \Big) \mathbf{z} = \tilde{\mathbf{f}}^h. \tag{11}$$

[3] For scalar variables, $\arg\min_e a(e-c)^2 + b(e-d)^2 = \arg\min_e (a+b)(e - \frac{ac+bd}{a+b})^2$.

where $\mathbf{W}_1^h$ is a diagonal matrix containing w_1^h. Note that the size of $\mathbf{D}_1$ and $\mathbf{I}$ is $W \times W$, not $HW \times HW$ as in (2). Interestingly, the problem (11) becomes much easier to solve than (2) since the system matrix is tridiagonal. We can solve (11) with $O(n)$ cost ($n = W$) by the Thomas algorithm [42], the Gaussian elimination for tridiagonal systems.

Algorithm 1 Fast domain decomposition

Input: f (an input image); g (a guidance image)
Parameters:
  γ (a Lagrangian multiplier, $\gamma^0 = \hat{\gamma}^1$)
  v (an auxiliary variable, $v^0 = \hat{v}^1 = f$)
  β (a penalty parameter, $\beta^1 > 0$)
  λ (a smoothing parameter, $\lambda > 0$)
Procedure Image smoothing with $\phi = \tau^2$ or $|\tau|$
for $k = 1 : K$ **do**
  for $y = 1 : H$
  $\tilde{f}_x^{\mathrm{h}} = (1+\beta^k)^{-1}(f_{x,y} + \beta^k(\hat{v}_{x,y}^k + \hat{\gamma}_{x,y}^k))$ for all x
  Compute z minimizing (10) or (12) according to ϕ
  $u^k(x, y) = z(x)$ for all x
  end for
  for $x = 1 : W$
  $\bar{f}_y^{\mathrm{v}} = (1+\beta^k)^{-1}(f_{x,y} + \beta^k(u_{x,y}^k - \hat{\gamma}_{x,y}^k))$ for all y
  Compute z minimizing (10) or (12) according to ϕ
  $v^k(x, y) = z(y)$ for all y
  end for
  $\gamma^k = \hat{\gamma}^k - (u^k - v^k)$; $\beta^{k+1} = \varepsilon\beta^k$
  $\alpha^{k+1} = (1 + \sqrt{1 + 4(\alpha^k)^2})/2$
  $\hat{\gamma}^{k+1} = \gamma^k + \frac{\alpha^k - 1}{\alpha^{k+1}}(\gamma^k - \gamma^{k-1})$
  $\hat{v}^{k+1} = v^k + \frac{\alpha^k - 1}{\alpha^{k+1}}(v^k - v^{k-1})$
end for
Output: u^K (smoothing output)

2) WL1 Smoothing: When $\phi = |\tau|$, (9) can be rewritten as follows:

$$\min_z \sum_x \frac{1}{2}(z - \tilde{f}^h)_x^2 + \tilde{w}_{1,x}^h |D_1 z|_x, \tag{12}$$

where $\tilde{w}_1^h = \lambda w_1^h / (1+\beta)$. There exists a non-iterative, $O(n)$ method for the 1D L1 minimization [43]. While this method is designed to solve the unweighted version of (12), it is possible to extend it to minimize 1D WL1. Note that this extension will enable the fast MM algorithm using L_1 upper-bound. See Section IV.E for details.

We introduce the (Fenchel-Moreau) dual form of (12) as follows:

$$\min_s \sum_x (\tilde{f}^h - D_1^T s)_x^2, \ \text{ s.t. } \ |s|_x \le \tilde{w}_{1,x}^h, \ \ s_1 = s_W = 0, \tag{13}$$

where s is the dual variable. Once the solution s^* of the problem (13) is found, we can recover the solution z^* of its primal form by

$$z_x^* = \tilde{f}_x^h - s_x^* + s_{x-1}^*, \ \text{ for } \ 1 \le x \le W. \tag{14}$$

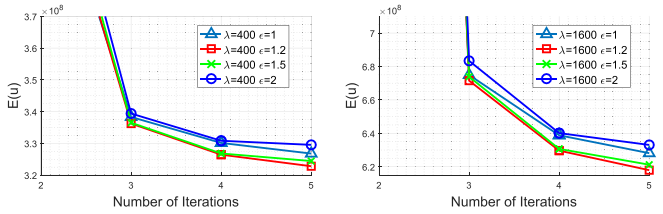


Fig. 1. Objective evolution of the WLS smoothing according to the difference choices of λ and ε: (Left) $\lambda = 400$. (Right) $\lambda = 1600$. Increasing β by a factor of ε=1.2 works well for a wider range of λ. It can be seen that the margin is more significant with larger value of λ. The input image is shown in Fig. 4 and κ is set to 7.65.

The optimality condition which characterizes the solutions z^* and s^* is then expressed as

$$\begin{cases} s_x^* = \tilde{w}_{1,x}^h & \text{if } z_{x+1}^* > z_x^* \\ s_x^* = -\tilde{w}_{1,x}^h & \text{if } z_{x+1}^* < z_x^* \\ s_x^* \in [-\tilde{w}_{1,x}^h, \tilde{w}_{1,x}^h] & \text{if } z_{x+1}^* = z_x^*. \end{cases} \tag{15}$$

The detailed derivations of (13) and (15) are available at Appendix.

C. 2D Smoothing Algorithm

The proposed method for 2D image smoothing is summarized in Algorithm 1. Given an input f, a guidance g and a smoothing parameter λ, the global 1D smoothing operations are sequentially performed along with the horizontal and vertical directions − although the original problem of (1) is decomposed into a series of sub-problems, we compute exact solutions of the objectives (7) and (8) defined on two-dimensions. At each iteration, the input for the horizontal smoothing is recalculated as the linear combination of f and $(\hat{v}^k + \hat{\gamma}^k)$ (Line 6). In the same manner, the vertical smoothing is applied to the linear combination of f and $(u^k - \hat{\gamma}^k)$ (Line 11). It can be seen that the output produced by the horizontal pass contributes to the input for the vertical one (and *vice versa*). The Lagrangian multiplier γ and penalty parameter β are also updated (Line 15). We extrapolate γ and v to guarantee the fast convergence [23] (Line 17 and 18). Finally, the algorithm is terminated after $k = K$ iterations. Note that color images are handled for each channel separately while using the same weight w for all channels.

D. Parameter Selection and More Discussion

Imposing the large regularity (with large λ) makes u and v evolve quite independently. β thus should be proportional to λ. Empirically, we set $\beta^1 = \sqrt{\lambda}/2$, and increase it during iteration by a factor of ε =1.2.[4] This strategy ensures that v should be close to u, satisfying the constraint in (4). Figure 1 shows the WLS objective evolution according to the different choices of ε and λ. It can be seen that increasing β by a factor of ε =1.2 works well for a wide range of λ. The difference is more significant with larger values of λ. The auxiliary variable v is initially set to the input image f. We use the constant initialization for the Lagrangian multiplier, e.g., $\gamma^1 = \mathbf{0}$.

[4]Increasing β during iteration is uncommon in the augmented Lagrangian method [25]. But, we found that this strategy works well in our application [26].

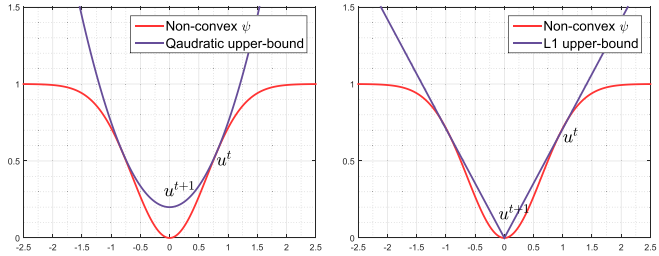


Fig. 2. Illustration of the MM principle using a convex upper bound: (Left) quadratic upper bound. (Right) L_1 upper bound. At the current iteration, a convex upper bound is computed that is locally tight at u^t. After minimizing the convex upper bound, we obtain u^{t+1} such that $E(u^{t+1}) \leq E(u^t)$.

The convergence of ADMM to solve the constrained optimization problem has been extensively studied [23], [24]. We show why our domain decomposition is more appealing than the DOF [20], [21], [39] by transforming (4) into a standard form. Consider the general problem of minimizing a convex functional subject to linear equality constraints:

$$\min_{u,v} F(u) + G(v), \quad \text{s.t} \;\; Au = v. \tag{16}$$

The convergence of ADMM is then expressed as [23], [24]:

$$H(\gamma^*) - H(\gamma^k) \leq \frac{c||\gamma^* - \gamma^1||^2}{k^\eta}. \tag{17}$$

Here H is a dual functional related to (16), and η is an order of convergence rate. An optimal Lagrange multiplier is denoted as γ^*. c is a constant proportional to $\rho(A^T A)$, where ρ denotes spectral radius of a matrix. Comparing (3) and (4), we can see that the convergence of the DOF [20], [21], [39] depends on $\rho(D^T D) < 8$, while our domain decomposition ($A = I$) does not depend on the spectral radius of $D^T D$. We found that this property considerably accelerates the convergence speed of Algorithm 1. 3-5 iterations are typically enough to get satisfactory results in both WLS and WL1 smoothing (see Section V).

Our formulation is not rotationally invariant as the original objective function of (1) enforces the regularization term to be aligned for each axis individually. However, visible artifacts were hardly found in all experiments for the WLS smoothing. A similar observation was also reported in [1]. When $\phi = |\tau|$, it prefers object boundaries which are minimal in the Manhattan distance (see Fig. 4).

E. Extension to Fast MM Algorithms

The proposed method can be extended for a wider class of regularization functions by using the MM algorithms [44]. This extension enables our approach to be applied to a range of applications, which require for sparse image gradients and sharp edges. Let us consider a non-convex ϕ. Examples include Welsch's norm [37], logarithm [46], and L_p norm [25]. The principle of the MM algorithm is to iteratively find a convex function which serves as the upper bound of (1), and to solve the corresponding optimization problems. It allows for explicit algorithms according to the construction of the upper bounds, as illustrated in Fig. 2. In the following, $t = (1, \ldots, T)$ denotes the *external* iteration index used to minimize the non-convex objective function.

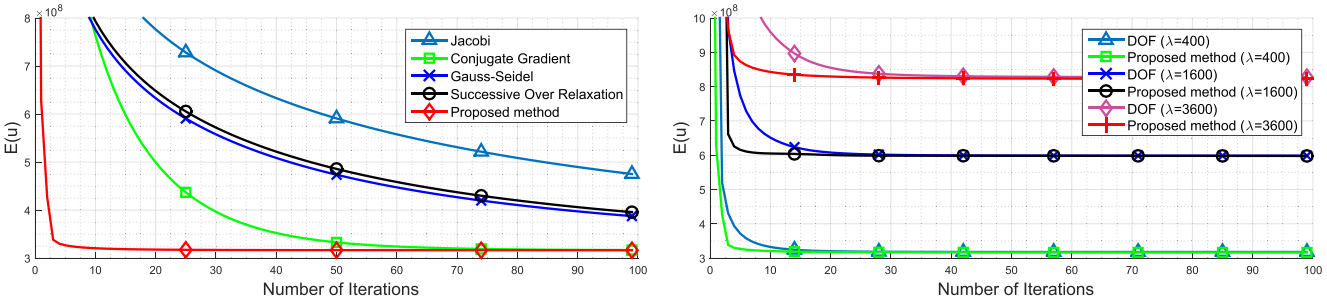


Fig. 3. Comparison of the WLS objective decrease versus the number of iterations: (Left) the proposed method vs linear system solvers. λ and κ are set to 400 and 7.65, respectively. (Right) the proposed method vs decomposition on objective function. Our approach rapidly reduces the WLS objective value of (1), and converges after a few iterations. The input image is shown in Fig. 4.

1) The Quadratic Majorization: This algorithm uses the fact that many regularization functions can be seen as minima of the quadratic upper bound [44]. Using an intermediate estimate u^t, we compute u^{t+1} by minimizing

$$E^t_{M_Q}(u) = \sum_p \Big((u-f)_p^2 + \lambda \sum_{j\in\{1,2\}} a_{j,p} (D_j u)_p^2 \Big) \quad (18)$$

where $a_{j,p} = w_{j,p} \frac{\phi'(D_j u^t)_p}{2(D_j u^t)_p}$. Minimizing (18) is equivalent to solving a linear system (2), except that a Laplacian matrix is computed with $a_{j,p}$. Thus, we can obtain u^{t+1} using Algorithm 1 with K *internal* iterations. Our approach involves only simple arithmetic operations for minimizing (18), while most existing linear system solvers [40] suffer from additionally computing the preconditioner at each *external* iteration due to intermediate linear systems that vary with $t = (1, ..., T)$.

2) The L_1 Majorization: Theoretically, the quadratic majorization can be applied only when the regularization function is well approximated with a quadratic upper bound [44]. This, however, does not cover interesting functions such as $\phi = \log(1 + |\tau|)$ [46] and L_p $(p < 1)$ norm [25], which are concave and non-differentiable at origin. Here we extend the proposed method to the MM algorithm using the L1 majorization.

$$E^t_{M_{L_1}}(u) = \sum_p \Big((u-f)_p^2 + \lambda \sum_{j\in\{1,2\}} b_{j,p} |D_j u|_p \Big), \quad (19)$$

where $b_{j,p} = w_{j,p} \partial\phi(D_j u^t)_p$, ϕ is concave, and ∂ denotes the sub-gradient. The method exploits a convex function obtained by linearizing the regularization function ϕ − (19) serves as the upper bound of (1) due to concavity of ϕ [46]. Obviously, each iteration is the WL1 problem which can be solved efficiently using Algorithm 1.

The pseudo-code is provided in Algorithm 2. Note that the exact minimum of (18) or (19) is not necessarily required to minimize an objective function using a non-convex regularizer. Instead, the MM algorithm guarantees (1) to decrease if $E^t_M(u^{t+1}) < E^t_M(u^t)$ [44]. We will show in experiments that 3-5 internal iterations of Algorithm 1 are sufficient for meeting a monotonic property of the MM algorithm.

Algorithm 2 Fast MM algorithm

1: **Input:** f (an input image); g (a guidance image)
2: **Parameters:**
   λ (a smoothing parameter, $\lambda > 0$)
3: **Procedure** Image smoothing with ψ
4: Initialize u^1
5: **for** $t = 0 : T - 1$ **do**
6: Construct the majorization function using (18) or (19)
7: Compute u^{t+1} using Algorithm 1
8: **end for**
9: **Output:** u^T (smoothing output)

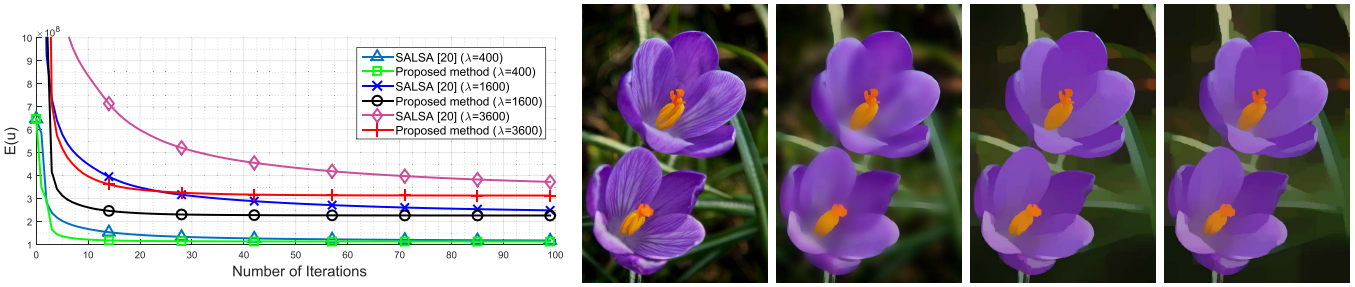
V. Experimental Validation

We evaluated the convergence and runtime performance of the proposed method. The smoothing quality in terms of structural similarity indexes (SSIM) [45] was also presented. The experiments were simulated with a single Intel i7 3.4GHz CPU. Our approach is easy to implement and the source code will be publicly available later. It was implemented on the MATLAB with MEX interface and all other results in comparison were obtained using the source codes provided by authors.

A. Objective Evolution

For convergence analysis of the WLS smoothing, we first compare our approach with the linear system solvers, including the conjugate gradient (CG), Gauss-Seidel (GS), Jacobi iteration (JI), and successive over relaxation (SOR). The SuiteSparse library [48] was used to solve a triangular system, arising from GS and SOR. Figure 3(a) shows how the WLS objective evolves at each iteration (all the methods were initialized by the input f). Although each iteration of CG, GS, JI, and SOR runs in a linear time $O(n)$ $(n = HW)$, they require a very large number of iterations to converge. In contrast, our solver converges in a few iterations only. The comparison with the DOF in (3) (using the accelerated ADMM [23] for fair comparison) is also provided in Fig. 3(b).[5] The FFTW library [49] is used to solve the u subproblem of the DOF.

[5]For the WLS (or WL1) smoothing, the v subproblem of the decomposition on objective function is component-wise division (or thresholding).



(a) WL1 objective evolution (b) Input image (c) SALSA (k=5) (d) SALSA (k=30) (e) Ours (k=5)

Fig. 4. Examples of the WL1 objective decrease versus the number of iterations: (a) Objective evolution, (b) input image, (c) SALSA [21] (at k=5), (d) SALSA [21] (at k=30), and (e) the proposed method (at k=5). Our approach rapidly reduces the WL1 objective value of (1), and converges after a few iterations. For (c)−(e), λ, κ, and k are set to 400, 25.5, and 5, respectively.

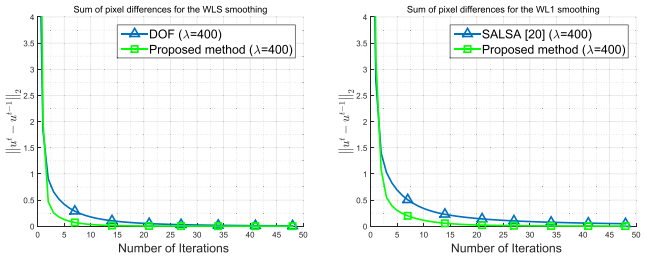


Fig. 5. Comparison of a sum of intensity difference between successive iterations: (Left) the WLS smoothing. (Right) the WL1 smoothing. Our approach rapidly reduces the sum of pixel difference between successive iterations. After k=5 iterations, the average values of the per-pixel difference are 0.13 and 0.3, respectively (the range of image values is [0 ∼ 255]). We set $u^0 = f$.

We can see that as the smoothing parameter λ gets larger, the DOF converges more slowly. Our approach, however, gives almost the same convergence property regardless of λ.

The objective evolution of the WL1 smoothing, with varying the regularization parameter λ, is shown in Fig. 4(a). We compare our approach with the split augmented Lagrangian shrinkage algorithm (SALSA) [21]. This method decomposes the WL1 objective function as in (3), and adopts the ADMM algorithm. We additionally perform the Nesterov-type extrapolation [23] for fair comparison. The SALSA [21] has $O(n \log n)$ complexity per iteration ($n = H \times W$), while our solver runs in linear time. It should be noted that the the convergence of SALSA [21] using the DOF is slow as in Fig. 4(a). It did not produce the desired smoothing result at early iterations (see Fig. 4(c)). In contrast, the proposed method achieves the satisfactory result after only 5 iterations as shown in Fig. 4(e).

In general, the WL1 smoothing requires more iterations than the WLS method for convergence (see Fig. 3(b) and Fig. 4(a)). However, our method gives almost the same smoothing results after few iterations (Fig. 5). The per-pixel color difference $\|u^k - u^{k-1}\|_2$ is reduced substantially after 5 iterations, where the average values are 0.13 and 0.3 for the WLS and the WL1 smoothing, respectively. We also evaluate the performance ratio using the criterion proposed in [47].

$$\mathcal{R} = \frac{\mathcal{T}_s}{\mathcal{T}_{our}}, \tag{20}$$

TABLE I

COMPARISON OF THE PERFORMANCE RATIO [47] BETWEEN OURS AND OTHER METHODS. USING λ [100 ∼ 3600] AND κ [0.03 ∼ 0.3], WE RUN OTHERS UNTIL THEY ACHIEVE THE OBJECTIVE VALUES OF (1) SMALLER THAN (OR EQUAL TO) THOSE OF OURS

Smoothing type	WLS			WL1
Method	Jacobi	GS	CG	SALSA [21]
Performance ratio, $\mathcal{R}$ [47]	54.5	34.2	28.5	10.2

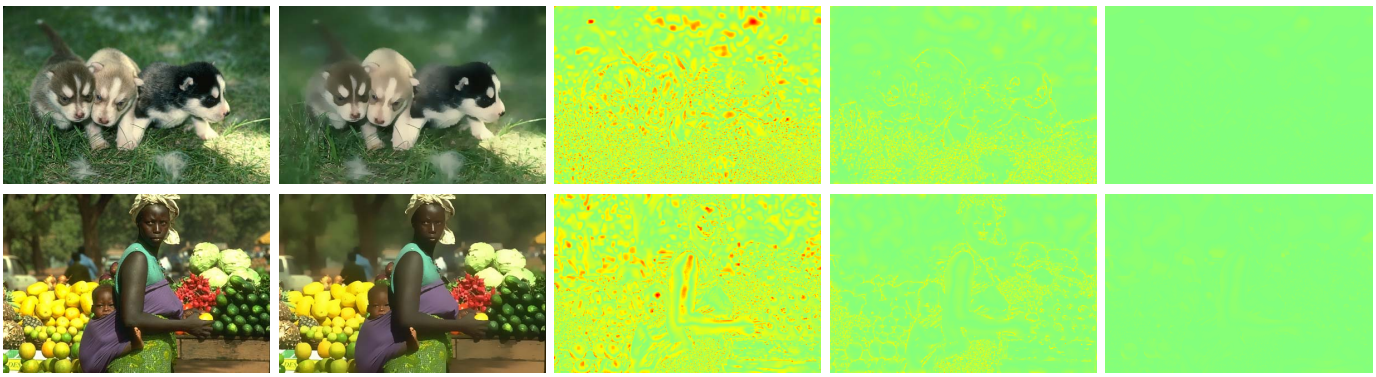
where $\mathcal{T}_s$ and $\mathcal{T}_{our}$ represent the computing times required to solve the same problem using existing methods and the proposed method, respectively. The objective values of (1), with λ [100 ∼ 3600] and κ [0.03 ∼ 0.3], are first measured after 5 iterations of Algorithm 1, and then we run other methods until they achieve the objective values of (1) equal to ours. The average performance ratios between ours and other methods are reported in Table 1.

B. Smoothing Quality

To further demonstrate the effectiveness of our approach, we additionally collect 100 natural images containing diverse contents from the BSD500 dataset [50], and compare the smoothing results. For the WLS smoothing, the result obtained by MATLAB "\" (backslash) command is used as the reference. In the case of the WL1 smoothing, the result obtained by the SALSA [21] with tight stopping criterion ($K = 200$) is assumed to be the reference image. The difference images between the reference and the results at $k = 5$ are visualized in Fig. 6. The parameters are set as $\kappa = 0.03$ and $\lambda = 100$. The smoothing results of ours are almost same as the reference image. Using a variety of parameters λ [100 ∼ 3600] and κ [0.03 ∼ 0.3], we also measure the SSIM index [45] between the references and the results obtained by our method. The average value of the SSIM index [45] is plotted as a function of k in Fig. 7. In general, we found that the proposed method yields satisfactory results after $k = 3 \sim 5$ iterations. The average SSIM values at $k = 3$ and 5 are 0.983 (0.981) and 0.996 (0.991), respectively for the WLS (WL1) smoothing.

C. Runtime

The average running time, on single CPU core, is reported in Table 2. Ten images are sampled from the BSD500 [50] and



(a) Input (b) Reference (c) CG ($k = 5$) (d) DOF ($k = 5$) (e) Ours ($k = 5$)

Fig. 6. Visualization of differences between the reference and results of each method for the WLS smoothing (at $k = 5$): (a) the input image, (b) the reference image obtained by MATLAB backslash, (c) CG, (d) DOF, and (e) the proposed method. The smoothing result of ours is almost same as the reference image. We set the parameters κ and λ as 0.03 and 100, respectively.

TABLE II

COMPARISON OF THE COMPUTATIONAL COMPLEXITY FOR DIFFERENT METHODS (IN SECONDS). ALL METHODS ARE IMPLEMENTED IN MATLAB WITH MEX INTERFACE

	Image size		PCG [19]	MATLAB "\"	DOF	ours		SALSA [21]	ours
Runtime (sec)	427×640	WLS	0.85	0.95	0.99	0.08	WL1	1.46	0.29
	660×800		1.58	1.97	1.61	0.15		2.79	0.61
	923×1128		3.04	4.14	4.27	0.31		6.64	1.27

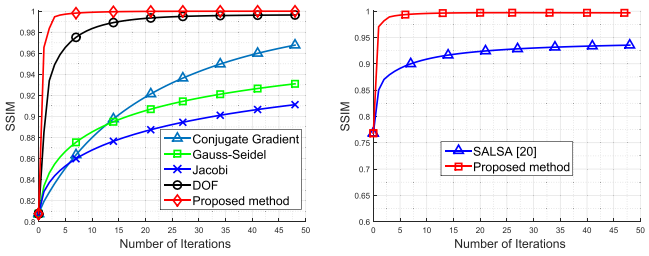
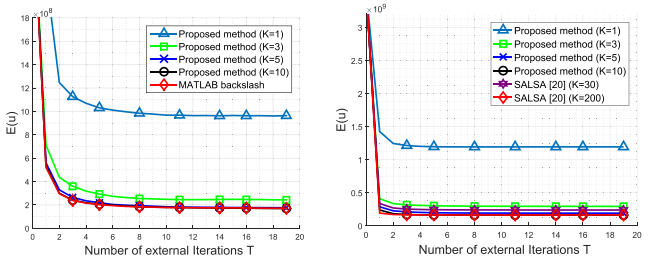


Fig. 7. The average SSIM indexes [45] for the WLS and WL1 smoothing, as a function of the number of iterations: (Left) the WLS smoothing. (Right) the WL1 smoothing. For the reference images, MATLAB backslash and the SALSA method [21] are used for the WLS and WL1 smoothing, respectively. The average SSIM values exceed 0.99 after $k = 5$ iterations for both cases.



(a) (b)

(c) (d) (e)

Fig. 8. Objective evolution of the MM algorithms depending on the number of inner iterations K (please see legend): (a) the MM using quadratic upper bound, (b) the MM using L_1 upper bound, (c) the input image, (d) MATLAB backslash, and (e) ours. The quadratic majorization method is used to obtain (d) and (e) from (c) (see text). Each subproblem is minimized by MATLAB backslash and the proposed method ($K = 5$) for (d) and (e), respectively.

resized to specific resolution as in Table 2. In our method, the number of iterations K is fixed to 5 based on the convergence analysis in Section V.A and V.B. For the WLS smoothing, our approach is compared with the state-of-the-art preconditioned conjugate gradient (PCG) method [19], MATLAB backslash operator, and the DOF. The result for the PCG [19] is obtained from the source code provided by the author. Note that the MATLAB backslash uses the sparse cholesky decomposition in the SuiteSparse libray [48] to solve the linear system of (2). Although the preconditioning method proposed in [19] improves the convergence rate significantly,[6] constructing the preconditioner is very time-consuming, taking about 2.6 seconds for a one megapixel (RGB) image. The stopping criterion of the DOF and the SALSA [21] is $\left\|u^{k+1} - u^k\right\|_2 < 0.1$ and 0.3[7] for the WLS and the WL1 smoothing, respectively. On average, these methods require 15 iterations to meet the stopping criterion. The runtime analysis may vary according to the degree of code optimization, but the proposed method tends to be about an order of magnitude faster than

[6]After preconditioning [19], 5 CG iterations are enough to solve (2).

[7]These are based on the result obtained by our method. See Section V.A.

(a) Input (b) RGF [51] (c) FWMF [33] (d) RTV [4] (e) ours

Fig. 9. Examples of the texture removal task: (a) the input image, (b) RGF [51], (c) WMF [33], (d) RTV [4], and (e) the proposed method. For each method, the running times are reported in seconds (yellow). The image sizes are 1254×1067 (top) and 495×536 (bottom), respectively. In this example, we employ the model proposed in [37], and minimize the corresponding optimization problem using our fast MM algorithm (with quadratic upper bound).

competing methods. It is even comparable to state-of-the-art local filters. For instance, the guided filter [17], one of the most popular and fastest local filters, takes 0.2 seconds for filtering a 923×1128 RGB image.

D. Analysis of Fast MM Algorithms

In this section, we present the analysis of our fast MM algorithms. We first minimize the objective function of (1) with $\phi = \sigma(1 - \exp(-\frac{\tau^2}{\sigma}))$, known as Welsch's function and $\sigma = 12$. Each quadratic upper bound is minimized by Algorithm 1. Figure 8(a) shows that the results differ depending on the number of inner iterations K. For $\phi = \log(1 + |\tau|)$, we use the L_1 upper bound as the logarithm function has an absolute behavior around 0 [46]. The objective evolution is plotted in Fig. 8(b). Overall, we observe that the objective evolutions with $K = 1$ (navy blue line of Fig. 8(a) and (b)) are very different from those of the reference implementation (red line of Fig. 8(a) and (b)). It is thus crucial to solve the sub-problems of MM algorithms until reaching the certain level of accuracy. Figure 8(c)-(e) shows that 5 inner iterations of Algorithm 1 are enough when using quadratic upper bounds. We set the external iteration $T = 5$ to obtain Fig. 8(d) and (e). The corresponding quadratic sub-problems are minimized by using MATLAB backslash (Fig. 8(d)) and Algorithm 1 (Fig. 8(e)). The result obtained by MATLAB backslash is very similar to ours. The input image is taken from [36]. In general, we found $K = 5$ and $T = 5$ iterations to be a good choice for fast MM algorithms. The MM algorithms guarantee convergence to a local minimum of the non-convex E, and thus different initializations for u^1 may give different solutions. The results in Figs. 8(d) and 8(e) are obtained from $u^1 = f$.

VI. Application

Our flexible approach finds several applications. We apply our method to texture removal, scale-space filtering, content-based color quantization, sparse color denoising, and style transfer. To this end, the various image priors, i.e., regularization function and weight w, are exploited to match different application goals. Table 3 summarizes the configuration of each application. The results of other methods were obtained from the source codes provided by the authors. The parameters were carefully tuned through extensive experiments.

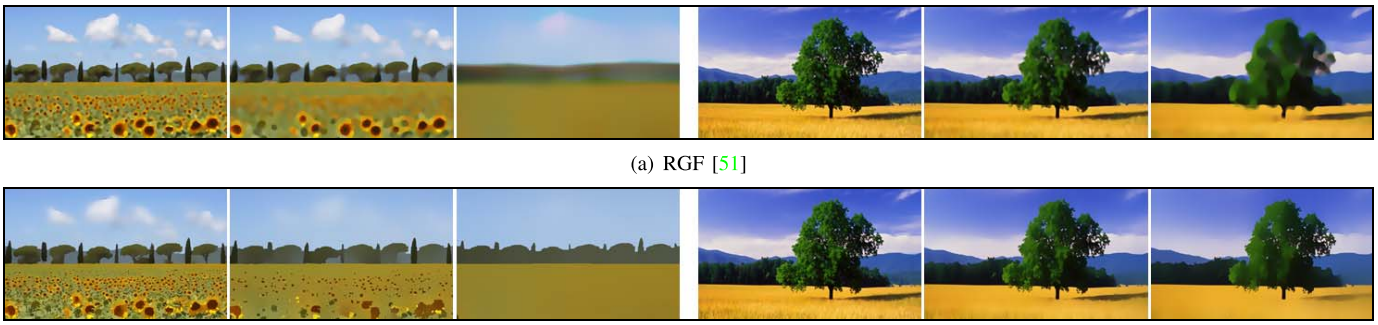
TABLE III

The Configuration of Each Application. M_Q and M_{L_1} Denote the Quadratic and L_1 Majorization, Respectively

Application	Regularization function, ϕ	Guidance, g	Solver
Texture removal	$\sigma(1 - \exp(-\tau^2/\sigma))$	$G * f$	Alg. 2 (M_Q)
Scale-space filtering	$\sigma(1 - \exp(-\tau^2/\sigma))$	f	Alg. 2 (M_Q)
Color quantization	$\log(1 + \lvert\tau\rvert)$	f	Alg. 2 (M_{L_1})
Sparse color denoising	$\log(1 + \lvert\tau\rvert)$	$\mathbf{1}$	Alg. 2 (M_{L_1})
Style transfer	τ^2 or $\lvert\tau\rvert$	Edge(f)	Alg. 1

A. Texture Removal

For texture removal, we employ the model proposed in [37]: ϕ is set to $\sigma(1 - \exp(-\frac{\tau^2}{\sigma}))$ and $g = G * f$ where G is the Gaussian kernel with standard deviation 2. This type of guidance image is very effective since texture on the object is usually of small scale structures. The smoothing parameters κ, and σ are fixed to 5 and 7.65, respectively, but λ varies according to image size. In this setting, we minimize the objective function of (1) using our fast MM solver (with the quadratic majorization). Figure 9 shows examples of texture removal task obtained by the rolling guidance filter (RGF) [51], the fast weighted median filter (FWMF) [33], the relative total variation (RTV) [4], and ours. The RGF [51] is implemented by iteratively applying the fast bilateral grid [31]. The runtime of our approach is comparable (or even faster) to the texture smoothing tools based on the local filtering (RGF [51] and FWMF [33]), while outperforming them in the subjective evaluation. The quality of RTV [4] is similar to ours, but the proposed method runs about 10 times faster. For example,



(a) RGF [51]

(b) ours

Fig. 10. Examples of scale-space filtering: (a) RGF [51] (from left to right $\sigma_s = 4, 8, 40$ for each image) and (b) the proposed method (from left to right $\lambda = 100, 1200, 6500$ for each image). The smoothing parameters were chosen, such that two methods have similar smoothing effect.

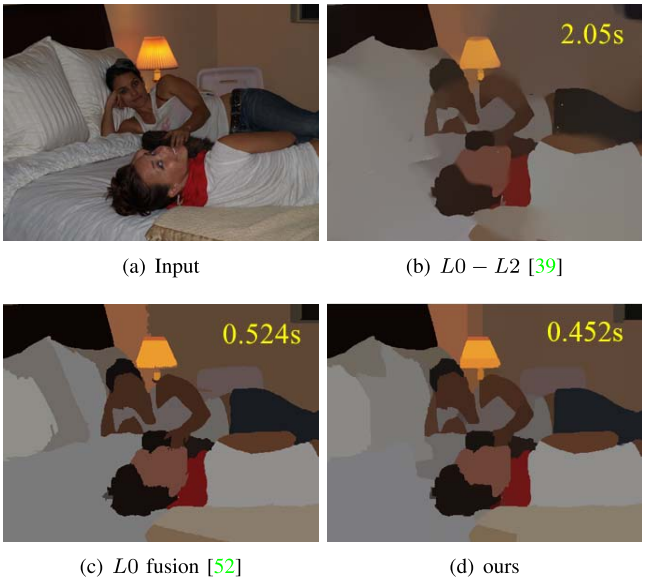
the RTV [4] and the proposed method take about 14.5 and 1.5 seconds to process a 1254×1067 image, respectively. Interestingly, minimizing the objective function in the RTV [4] needs to iteratively solve a large linear system, where a system matrix varies during iterations. Thus, it can be also accelerated by Algorithm 1.

B. Scale-Space Filtering

We obtain a scale-space representation by minimizing the same objective function as the texture removal task, except that the input image is guided by itself ($g = f$). Two images in Fig. 10 are of size 640×376 and 1500×730, respectively. We alter the scale of filtered image by adjusting the smoothing parameters, and these are chosen such that two methods have similar amount of smoothing. The proposed method better preserves edges and corners than the RGF [51] with a faster runtime. The RGF [51] shows color artifacts and poor boundary localization at coarse scales (Fig. 10(a)). For the image of 1500×730, RGF [51] and our fast MM algorithm (with the quadratic majorization) take 1.62 and 1.35 seconds, respectively.

C. Color Quantization

Content-based color quantization is used to reduce the number of colors in images. The quantization effect makes prominent structures easier to be detected and more visually distinct. This is useful for many computer vision and computational photography applications, including image retrieval and segmentation. We impose the sparsity using the regularization function, $\phi = \log(1 + |\tau|)$ to reduce the number of colors, and set $g = f$. With this setting, we minimize the objective function of (1) using our fast MM algorithm (with L_1 majorization). The results were compared in Fig. 11. Our solver shows performance comparable to state-of-the-art L_0 minimization [39], [52], while running faster. The method [39] decomposes their objective function into two sub-problems and solve them by an L_0-L_2 framework. A continuation parameter was introduced to balance the influence of the L_0 norm on the smoothing result. A smaller continuation value gives better L_0 regularized results, but at the cost of a longer running time. It should be noted that Fig. 11(c) was obtained using the region fusion (RF) method tailored to a fast L_0 gradient minimization only [52].



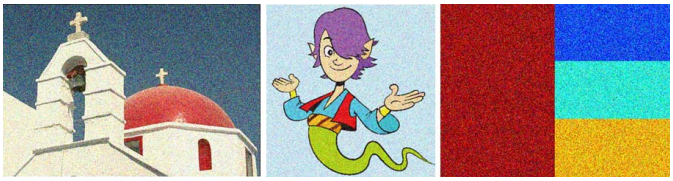
(a) Input (b) $L0 - L2$ [39]

(c) $L0$ fusion [52] (d) ours

Fig. 11. Examples of the content-based color quantization: (a) the input image, (b) $L_0 - L_2$ [39], (c) L_0 fusion [52], and (d) ours. Our approach shows performance comparable to state-of-the-art L_0 minimization. The runtime is reported in seconds (the image size is 494×371). In this example, we minimize the objective function of (1) with $\phi = \log(1+|\tau|)$ using our fast MM algorithm (L_1 majorization).

D. Sparse Color Denoising

The regularization function, $\phi = \log(1 + |\tau|)$ approximates the input signal by a series of piecewise constant functions. Thus, it is also suitable for denoising the image that contains sparse colors with sharp transitions, e.g., cartoon and clip art. The guidance image is not used in this application, i.e., $w_{j,p} = 1$ for all j and p. We compare our method against L_0 gradient minimization proposed by Xu *et al.* [39]. Figure 12 shows three examples of image denoising. In our experiments, we set the continuation parameter of [39] such that the best results are obtained in terms of peak signal-to-noise-ratio (PSNR). Refer to Section VI.C for more details. Thus, its runtime varies although the images in Fig. 12 are of similar spatial resolution (three images are of 481×321, 367×372,



(a) Input

(b) $L_0 - L_2$ [39], PSNR=24.2, 28.7, 20.7.

(c) ours, PSNR=24.6, 29.2, 23.1.

Fig. 12. Comparisons of image denosing: (a) the input image, (b) $L_0 - L_2$ [39], and (c) ours. We also report the running time (in seconds) and the SNR (in dB). We set $\phi = \log(1 + |\tau|)$ and the guidance image is not used in this application. The results of (c) are obtained by our fast MM algorithm with L_1 majorization.

(a) Input (b) Guidance

(c) FWMF [33] (d) ours

Fig. 13. Style transfer effect: (a) the input image, (b) the guidance image, (c) FWMF [33], and (d) ours. The edge map of the input is used as the guidance image. Note that the input (a) and guidance (b) images are taken from [33].

and 400 × 300, respectively). Our quantitative results are consistently better than [39], and the runtime is also faster.

E. Style Transfer

The aim of style transfer is to transfer fine structures of guidance signals to input images, enhancing or altering image structures. Any feature map from input image or others can be taken as the guidance image g. Therefore, it can be applied to image editing, RGB/NIR restoration, flash/non-flash denoising, and depth super-resolution. Figure 13 shows a specific style transfer example obtained by the FWMF [33] and ours. The edge map of the input image is used as the guidance signal. The window size of the FWMF [33] is set to 5×5, taking about 0.4 seconds. Our WLS (WL1) smoothing takes only 0.09 (0.3) seconds for an image of size 640×480.

VII. Conclusion

We have introduced an efficient global EPS method that is widely applicable to image processing and computational photography tasks. Contrary to previous decomposition methods, our formulation enables fast, linear time solvers for both WLS and WL1 smoothing. We have showed through intensive experiments that our method converges quickly after few iterations. The runtime is much faster than conventional methods, and is even comparable to the state-of-the-art local EPS approaches. A family of fast MM algorithms were also proposed using a non-convex regularization term. As confirmed by our results, the proposed method has been successfully used in several applications. In the future, we will implement the proposed method on the GPU and embedded system.

Appendix

We consider 1D WL1 problem as follows:

$$\arg\min_z \sum_x \left(\frac{1}{2}(z-f)_x^2 + w_x|Dz|_x\right). \tag{21}$$

By the MinMax theorem [53], we have

$$\begin{aligned}&\min_z \sum_x \left(\frac{1}{2}(z-f)_x^2 + w_x|Dz|_x\right)\\ &= \max_{|s|_x \le w_x} \min_z \sum_x \left(\frac{1}{2}(z-f)_x^2 + \langle s, Dz\rangle_x\right),\end{aligned} \tag{22}$$

for all $x = 1, \ldots, \mathrm{W}$. s is the dual variable. The minimizer of the last expression with respect to z is

$$z = f - D^T s. \tag{23}$$

Substituting (23) into (22), we obtain the dual form of (21).

$$\min_s \sum_x (f - D^T s)_x^2, \quad \text{s.t. } |s|_x \le w_x, \ \forall x = 2, \ldots, W-1, \tag{24}$$

where $s_W = s_1 = 0$. Let us consider the first-order optimality condition of the Euclidean projection problem (24).

$$D(f - D^T s^*) \in N(s^*), \tag{25}$$

where $N(s^*)$ is the normal cone with respect to a set $|s|_x \le w_x$, $x = 1, \ldots, W$. According to (23), we also have

$$D(f - D^T s^*) = \begin{cases} z_2^* - z_1^* \\ \vdots \\ z_W^* - z_{W-1}^* \end{cases} \tag{26}$$

From (25), (26), and the definition of the normal cone, we finally conclude

$$\begin{cases} s_x^* = w_x & \text{if } z_{x+1}^* > z_x^* \\ s_x^* = -w_x & \text{if } z_{x+1}^* < z_x^* \\ s_x^* \in [-w_x, w_x] & \text{if } z_{x+1}^* = z_x^*, \end{cases} \tag{27}$$

and

$$z_x^* = f_x - s_x^* + s_{x-1}^*. \tag{28}$$

REFERENCES

[1] Z. Farbman, R. Fattal, D. Lischinski, and R. Szeliski, "Edge-preserving decompositions for multi-scale tone and detail manipulation," *ACM Trans. Graph.*, vol. 27, no. 3, pp. 1–67, 2008.

[2] Z. Ma, K. He, Y. Wei, J. Sun, and E. Wu, "Constant time weighted median filtering for stereo matching and beyond," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 49–56.

[3] B. Ham, D. Min, and K. Sohn, "A generalized random walk with restart and its application in depth up-sampling and interactive segmentation," *IEEE Trans. Image Process.*, vol. 22, no. 7, pp. 2574–2588, Jul. 2013.

[4] L. Xu, Q. Yan, Y. Xia, and J. Jia, "Structure extraction from texture via relative total variation," *ACM Trans. Graph.*, vol. 31, no. 6, p. 139, 2012.

[5] T. Kim, K. Lee, and S. Lee, "Generative image segmentation using random walks with restart," in *Proc. Eur. Conf. Comput. Vis.*, 2008, pp. 264–275.

[6] F. Perazzi, P. Krähenbühl, Y. Pritch, and A. Hornung, "Saliency filters: Contrast based filtering for salient region detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 733–740.

[7] A. Levin, D. Lischinski, and Y. Weiss, "A closed form solution to natural image matting," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 2, pp. 228–242, Feb. 2008.

[8] K. He, J. Sun, and X. Tang, "Single image haze removal using dark channel prior," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, May 2009, pp. 1956–1963.

[9] L. Grady and J. Polimeni, *Discrete Calculus: Applied Analysis on Graphs for Computational Science*. London, U.K.: Springer-Verlag, 2010.

[10] M. Kass and J. Solomon, "Smoothed local histogram filters," *ACM Trans. Graph.*, vol. 29, no. 4, p. 100, 2010.

[11] D. Min, J. Lu, and M. N. Do, "Depth video enhancement based on weighted mode filtering," *IEEE Trans. Image Process.*, vol. 21, no. 3, pp. 1176–1190, Mar. 2012.

[12] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *Proc. IEEE Int. Conf. Comput. Vis.*, Sep. 1998, pp. 839–846.

[13] S. Smith and J. Brady, "SUSAN—A new approach to low level image processing," *Int. J. Comput. Vis.*, vol. 23, no. 1, pp. 45–78, 1995.

[14] J. Kopf, M. F. Cohen, D. Lischinski, and M. Uyttendaele, "Joint bilateral upsampling," *ACM Trans. Graph.*, vol. 26, no. 3, pp. 96–100, 2007.

[15] Q. Yang, K. Tan, and N. Ahuja, "Real-time O(1) bilateral filtering," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 557–564.

[16] S. Paris and F. Durand, "A fast approximation of the bilateral filter using a signal processing approach," *Int. J. Comput. Vis.*, vol. 81, no. 1, pp. 24–52, 2009.

[17] K. He, J. Sun, and X. Tang, "Guided image filtering," in *Proc. Eur. Conf. Comput. Vis.*, 2010, pp. 1–14.

[18] E. S. L. Gastal and M. M. Oliveira, "Domain transform for edge-aware image and video processing," *ACM Trans. Graph.*, vol. 30, no. 4, pp. 1–69, 2011.

[19] D. Krishnan, R. Fattal, and R. Szeliski, "Efficient preconditioning of Laplacian matrices for computer graphics," *ACM Trans. Graph.*, vol. 32, no. 4, pp. 142-1–142-14, 2013.

[20] Y. Wang, J. Yang, W. Yin, and Y. Zhang, "A new alternating minimization algorithm for total variation image reconstruction," *SIAM J. Imag. Sci.*, vol. 1, no. 3, pp. 248–272, Aug. 2008.

[21] M. V. Afonso, J.-M. Bioucas-Dias, and M. A. T. Figueiredo, "Fast image recovery using variable splitting and constrained optimization," *IEEE Trans. Image Process.*, vol. 19, no. 9, pp. 2345–2356, Sep. 2010.

[22] D. Min, S. Choi, J. Lu, B. Ham, K. Sohn, and M. N. Do, "Fast global image smoothing based on weighted least squares," *IEEE Trans. Image Process.*, vol. 23, no. 12, pp. 5638–5653, Dec. 2014.

[23] T. Goldstein, B. O'Donoghue, S. Setzer, and R. Baraniuk, "Fast alternating direction optimization methods," *SIAM J. Imag. Sci.*, vol. 7, no. 3, pp. 1588–1623, 2014.

[24] M. Hong and Z. Luo. (Aug. 2012). "On the linear convergence of the alternating direction method of multipliers." [Online]. Available: https://arxiv.org/abs/1208.3922

[25] L. Xu, X. Tao, and J. Jia, "Inverse kernels for fast spatial deconvolution," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 33–48.

[26] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, Jan. 2011.

[27] Y. Nesterov, "A method of solving a convex programming problem with convergence rate $O(1/k^2)$," *Soviet. Math. Dokl.*, vol. 27, no. 3, pp. 372–376, 1983.

[28] H. Winnemöller, S. Olsen, and B. Gooch, "Real-time video abstraction," *ACM Trans. Graph.*, vol. 25, no. 3, pp. 1221–1226, 2006.

[29] R. Fattal, M. Agrawala, and S. Rusinkiewicz, "Multiscale shape and detail enhancement from multi-light image collections," *ACM Trans. Graph.*, vol. 26, no. 3, p. 51, 2007.

[30] C. Liu, W. T. Freeman, R. Szeliski, and S. B. Kang, "Noise estimation from a single image," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2006, pp. 901–908.

[31] A. Adams, J. Baek, and M. Davis, "Fast high-dimensional filtering using the permutohedral lattice," *Comput. Graph. Forum*, vol. 29, no. 2, pp. 753–762, 2010.

[32] E. S. L. Gastal and M. M. Oliveira, "Adaptive manifolds for real-time high-dimensional filtering," *ACM Trans. Graph.*, vol. 31, no. 4, p. 33, 2012.

[33] Q. Zhang, L. Xu, and J. Jia, "100+ times faster weighted median filter (WMF)," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 2830–2837.

[34] A. Levin, D. Lischinski, and Y. Weiss, "Colorization using optimization," *ACM Trans. Graph.*, vol. 23, no. 4, pp. 689–694, 2004.

[35] Z. Farbman, R. Fattal, and D. Lischinski, "Diffusion maps for edge-aware image editing," *ACM Trans. Graph.*, vol. 29, no. 6, p. 145, 2010.

[36] S. Bi, X. Han, and Y. Yu, "An L_1 image transform for edge-preserving smoothing and scene-level intrinsic decomposition," *ACM Trans. Graph.*, vol. 34, no. 4, p. 778, 2015.

[37] B. Ham, M. Cho, and J. Ponce, "Robust image filtering using joint static and dynamic guidance," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Sep. 2015, pp. 4823–4831.

[38] Y. Kim, S. Choi, C. Oh, and K. Sohn, "A majorize-minimize approach for high-quality depth upsampling," in *Proc. IEEE Int. Conf. Image Process.*, Sep. 2015, pp. 392–396.

[39] L. Xu, C. Lu, Y. Xu, and J. Jia, "Image smoothing via L_0 gradient minimization," *ACM Trans. Graph.*, vol. 30, no. 6, p. 174, 2011.

[40] Y. Saad, *Iterative Methods for Sparse Linear Systems*. Philadelphia, PA, USA: SIAM, 2003.

[41] D. Geman and C. Yang, "Nonlinear image recovery with half-quadratic regularization," *IEEE Trans. Image Process.*, vol. 4, no. 7, pp. 923–946, Mar. 1995.

[42] H.R. Schwarz, N. Köckler, "Numerische mathematik," Stuttgart, Germany: Vieweg+Teubner Verlag, 1988.

[43] L. Condat, "A direct algorithm for 1-D total variation denoising," *IEEE Signal Process. Lett.*, vol. 20, no. 11, pp. 1054–1057, Nov. 2013.

[44] J. Mairal, F. Bach, and J. Ponce, "Sparse modeling for image and vision processing," *Found. Trends Comput. Graph. Vis.*, vol. 8, nos. 2–3, pp. 283–285, 2014.

[45] Z. Wang, A. C. Bovik, H. Rahim, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.

[46] P. Ochs, A. Dosovitskiy, T. Brox, and T. Pock, "On iteratively reweighted algorithms for nonsmooth nonconvex optimization in computer vision," *SIAM J. Imag. Sci.*, vol. 8, no. 1, pp. 331–372, 2015.

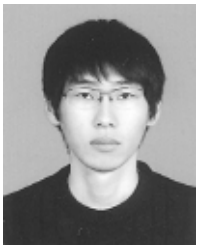
[47] E. D. Dolan and J. J. More, "Benchmarking optimization software with performance profiles," *Math. Program.*, vol. 91, no. 2, pp. 201–213, 2002.

[48] *SuiteSparse*, accessed on Sep. 15, 2016. [Online]. Available: http://faculty.cse.tamu.edu/davis/suitesparse.html/

[49] *FFTW*, accessed on Oct. 8, 2016. [Online]. Available: http://www.fftw.org/

[50] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *Proc. 8th IEEE Int. Conf. Comput. Vis. (ICCV)*, Jul. 2001, pp. 416–423.

[51] Q. Zhang, X. Shen, L. Xu, and J. Jia, "Rolling guidance filter," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 1–2.
[52] R. M. H. Nguyen and M. S. Brown, "Fast and effective L_0 gradient minimization by region fusion," in *Proc. IEEE Int. Conf. Comput. Vis.*, Apr. 2015, pp. 208–216.
[53] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.



Youngjung Kim (S'14) received the B.S. degree in electrical and electronic engineering from Yonsei University, Seoul, South Korea, in 2013, where he is currently pursuing the joint M.S. and Ph.D. degrees in electrical and electronic engineering. His current research interests include variational method and optimization, both in theory and applications in image processing and computer vision.

Dongbo Min (M'09–SM'15) received the B.S., M.S., and Ph.D. degrees from the School of Electrical and Electronic Engineering, Yonsei University, Seoul, South Korea, in 2003, 2005, and 2009, respectively. He was a Post-Doctoral Researcher with Mitsubishi Electric Research Laboratories, Cambridge, MA, USA, from 2009 to 2010. From 2010 to 2015, he was with the Advanced Digital Sciences Center, Singapore. Since 2015, he has been an Assistant Professor with the Department of Computer Science and Engineering, Chungnam National University, Daejeon, South Korea. His current research interests include computer vision, 2-D/3-D video processing, computational photography, augmented reality, and continuous/discrete optimization.

Bumsub Ham (M'13) received the B.S. and Ph.D. degrees in electrical and electronic engineering from Yonsei University in 2008 and 2013, respectively. From 2014 to 2016, he was a Post-Doctoral Research Fellow with the Willow Team of INRIA Rocquencourt, École Normale Supérieure de Paris, and the Center National de la Recherche Scientifique. He is currently an Assistant Professor of Electrical and Electronic Engineering with Yonsei University, Seoul, South Korea. His research interests include computer vision, computational photography, and machine learning, in particular, regularization and matching, both in theory and applications.

Kwanghoon Sohn (M'92–SM'12) received the B.E. degree in electronic engineering from Yonsei University, Seoul, South Korea, in 1983, the M.S.E.E. degree in electrical engineering from the University of Minnesota, Minneapolis, MN, USA, in 1985, and the Ph.D. degree in electrical and computer engineering from North Carolina State University, Raleigh, NC, USA, in 1992. He was a Senior Member of the Research engineer with the Satellite Communication Division, Electronics and Telecommunications Research Institute, Daejeon, South Korea, from 1992 to 1993, and a Post-Doctoral Fellow with the MRI Center, Medical School of Georgetown University, Washington, DC, USA, in 1994. He was a Visiting Professor with Nanyang Technological University, Singapore, from 2002 to 2003. He is currently a Professor with the School of Electrical and Electronic Engineering, Yonsei University. His research interests include 3-D image processing and computer vision.