KNN Matting

Qifeng Chen, *Student Member*, *IEEE*, Dingzeyu Li, *Student Member*, *IEEE*, and Chi-Keung Tang, *Senior Member*, *IEEE*

Abstract—This paper proposes to apply the nonlocal principle to general alpha matting for the simultaneous extraction of multiple image layers; each layer may have disjoint as well as coherent segments typical of foreground mattes in natural image matting. The estimated alphas also satisfy the summation constraint. As in nonlocal matting, our approach does not assume the local color-line model and does not require sophisticated sampling or learning strategies. On the other hand, our matting method generalizes well to any color or feature space in any dimension, any number of alphas and layers at a pixel beyond two, and comes with an arguably simpler implementation, which we have made publicly available. Our matting technique, aptly called *KNN matting*, capitalizes on the nonlocal principle by using K nearest neighbors (KNN) in matching nonlocal neighborhoods, and contributes a simple and fast algorithm that produces competitive results with sparse user markups. KNN matting has a closed-form solution that can leverage the preconditioned conjugate gradient method to produce an efficient implementation. Experimental evaluation on benchmark datasets indicates that our matting results are comparable to or of higher quality than state-of-the-art methods requiring more involved implementation. In this paper, we take the nonlocal principle beyond alpha estimation and extract overlapping image layers using the same Laplacian framework. Given the alpha value, our closed form solution can be elegantly generalized to solve the multilayer extraction problem. We perform qualitative and quantitative comparisons to demonstrate the accuracy of the extracted image layers.

Index Terms—Natural image matting, layer extraction

1 Introduction

Alpha matting refers to the problem of decomposing an image into two layers, called foreground and background, which is a convex combination under the image compositing equation:

$$I = \alpha F + (1 - \alpha)B, \tag{1}$$

where I is the given pixel color, F is the unknown foreground layer, B is the unknown background layer, and α is the unknown alpha matte. This compositing equation takes a general form when there are $n \geq 2$ layers:

$$I = \sum_{i=1}^{n} \alpha_i F_i, \quad \sum_{i=1}^{n} \alpha_i = 1. \tag{2}$$

We are interested in solving the general alpha matting problem for extracting multiple image layers simultaneously with sparse user markups, where such markups may fail approaches requiring reliable color samples to work. Refer to Figs. 1 and 2. While the output can be foreground/background layers exhibiting various degrees of spatial coherence, as in natural image matting on single RGB images, the extracted layers with fractional alpha boundaries can also be disjoint, as those obtained in material matting from multichannel images that capture spatially varying bidirectional distribution function (SVBRDF).

Inspired by nonlocal matting [12] and sharing the mathematical properties of nonlocal denoising [2], our approach capitalizes on K nearest neighbors (KNN) searching in the feature space for matching, and uses an improved matching metric to achieve good results with a simpler algorithm than [12]. We do not assume the *local* 4D color-line model [14], [15] widely adopted by subsequent matting approaches; thus our approach generalizes well in any color space (e.g., HSV) in any dimensions (e.g., 6D SVBRDF). It does not require a large kernel to collect good samples [10], [12] in defining the Laplacian, nor does it require good foreground and background sample pairs [27], [9], [6], [21] (which need user markups of more than a few clicks, much less that the foreground and background are unknown themselves), nor any learning [30], [29] (where training examples are issues), and yet our approach is not at odds with these approaches when regarded as postprocessing for alpha refinement akin to [9]. Moreover, the summation property, where the alphas are summed to one at a pixel, is naturally guaranteed in two-layer or multiple-layer extraction. Our matting technique, called *KNN matting*, still enjoys a closed-form solution that can harness the preconditioned conjugate gradient method (PCG) [1], and runs in on the order of a few seconds for high-resolution images in natural image matting after accepting very sparse user markups: Our unoptimized Matlab solver runs in 15-18 seconds on a computer with an Intel Xeon E5520 CPU running at 2.27 GHz for images of size 800×600 available at the alpha matting evaluation website [20]. Experimental evaluation on this benchmark dataset indicates that our matting approach is competitive in quality of results with acceptable speed.

- *Q. Chen is with the Department of Computer Science, Stanford University, Stanford, CA 94305. E-mail: cqf@stanford.edu.*
- *D. Li is with the Department of Computer Science, Columbia University, New York, NY 10027. E-mail: dli@cs.columbia.edu*
- *C.-K. Tang is with the Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong. E-mail: cktang@cse.ust.hk.*

Manuscript received 1 Sept. 2012; revised 10 Dec. 2012; accepted 16 Dec. 2012; published online 9 Jan. 2013.
Recommended for acceptance by M. Brown.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-2012-09-0686.
Digital Object Identifier no. 10.1109/TPAMI.2013.18.

0162-8828/13/$31.00 © 2013 IEEE Published by the IEEE Computer Society

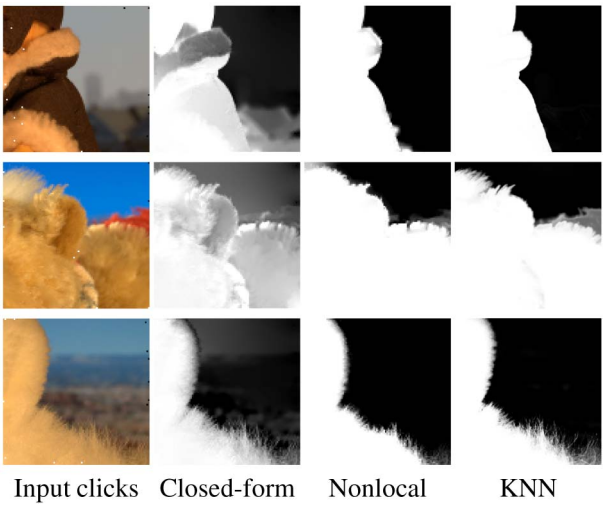


Fig. 1. Using the sparse click inputs the same as nonlocal matting [12], KNN matting produces better results. Top: Clearer and cleaner boundary; middle: more details are preserved for hairs as well as the red fuzzy object; bottom: the furs are more clearly separated from the background.

The preliminary version of this paper appeared in [3]. Besides updating the current state of the arts and presenting more examples on α-matting, in this coverage we extend the nonlocal principle to extract multiple and overlapping image layers (i.e., F) using the same Laplacian formulation, thus keeping the simple strategy and implementation. We show quantitatively and qualitatively the accuracy of the extracted layers when compared with the results obtained using closed form matting (CF matting) [14] and related techniques where the local color-line model was adopted.

2 Related Work

2.1 Natural Image Matting

For a thorough survey on matting see [28]; here, we cite the works that are closely related to ours. The matting problem is severely underconstrained, with more unknowns than equations to solve, so user interaction is needed to resolve ambiguities and constrain the solution. Spatial proximity taking the form of user-supplied trimaps or strokes was employed in [4] and [24], which causes significant errors when the labels are distant, and becomes impractical for matting materials with SVBRDF [13].

For images with piecewise smooth regions, spatial connectivity in small image windows was used in defining the *matting Laplacian* [14] for foreground extraction and later, in [15], for multiple layer extraction. Good results are guaranteed if the linear 4D color-line model within a local 3×3 window holds [15]. The solution is guaranteed to lie in the nullspace of the matting Laplacian if one of the three conditions described in their Claim 1 is satisfied. These conditions are, on the other hand, somewhat specific as to how a single layer, two, and three overlapping layers should behave in the color space. Violations are not uncommon though, and in that case, they are often manifested into tedious markups where the user needs to carefully mark up relevant colors in textured regions at times nonlocal to one another. The closed form solution for multiple layer extraction was analyzed in [22], where the summation and positivity constraints were investigated. The Laplacian construction and line model assumption from [14], [15] were still adopted.

On the other hand, the nonlocal principle has received a lot of attention for its excellent results in image and movie denoising [2]. Two recent CVPR contributions on natural image matting [12], [9] have tapped into sampling nonlocal neighborhoods.

In [12], reduced user input is achieved by accurate *clustering* of foreground and background, where ideally the user only needs to constrain a *single* pixel in each cluster for computing an optimal matte. Thus, we prefer good clustering to good sampling of reliable foreground-background pairs for the following reasons: Sampling techniques will fail in very sparse inputs that can otherwise generate good results in KNN matting; they do not generalize well to $n > 2$ layers due to the potentially prohibitive joint search space when denser input is used; adopting various modeling or sampling strategies usually leads to more complicated implementation (e.g., use of randomized patchmatch in [9], ray shooting in [6], PSF estimation in [19]), resulting in more parameter setting or requiring more careful markups/trimaps. As we will demonstrate, KNN matting requires only one noncritical parameter K.

The other recent CVPR contribution consists of correspondence search based on a cost function derived from the compositing equation [9]. Noting that relevant color sampling improves performance [27], [6], this approach samples and matches in a randomized manner relevant nonlocal neighbors in a joint foreground-background space which, as mentioned, can become prohibitively large if it is

Fig. 2. KNN matting on material matting using the *sg* dataset. Original images at the top; the bottom shows sparse user input (five clicks, one per layer) and the five layers automatically extracted. Our result distinguishes the two different gold foil layers despite their subtle difference in materials (where they were combined in [11]).

generalized to handle multiple layers. Earlier, a fast matting method (up to $20\times$ compared with [14]) was proposed in [10] that uses large kernels for achieving high-quality results. Since the same local color-line model and the same Laplacian construction in [14], [15] were adopted, unsatisfactory results are unavoidable where large windows were used and the model assumption fails. So, a separate KD-tree segmentation step was used to make the kernel size adaptive to the trimap.

Contemporary work [21] uses texture information as well as RGB color priors to define a novel objective function. This method still belongs to the category of sampling the best foreground/background pairs with sophisticated texture manipulation and postprocessing. Another recent work [29] adopted a learning approach, and uses support vector machine to address the alpha-matting problem.

2.2 Layer Extraction in Image Matting

Most existing works on image matting focus on alpha estimation but not layer extraction (in the two-layer case, foreground and background extraction) [12], [6], [18], [19], [10], [9], [30], [27], [26], [8]. One usually simply applies αI to matte out the foreground, which, as we will show, gives suboptimal results than αF.

The following are a few exceptions where layer extraction was addressed. In Bayesian matting [4], the log likelihood is maximized by iteratively computing the alpha and foreground/background. Poisson matting [24] estimates foreground and background in their global version. In CF matting [14], [15], the foreground and background are solved by using the estimated alpha and the compositing equation with a spatial coherence term. In [22], after the mattes have been estimated, the authors used [14] to reconstruct the image layers. Earlier, the iterative optimization [26] also directly made use of the compositing equation with known alpha in their foreground and background layer estimation. Recently, material matting [13] adopted spatial and texture coherence constraints for extracting multiple layers. In this paper, we show that our closed form solution can be elegantly generalized to extract overlapping image layers. We perform qualitative and quantitative analysis, focusing on comparing the local color-line model and the nonlocal principle in transparent and overlapping layer extraction from single images.

3 Nonlocal Principle for Alpha Matting

As in nonlocal matting [12], our KNN matting capitalizes on the nonlocal principle [2] in constructing affinities to produce good graph clusters. Consequently, sparse input is sufficient for extracting the respective image layers. It was also noted in [12] that the matting Laplacian proposed in [14] in many cases is not conducive to good clusters, especially when the local color-line model assumption fails, which is manifested into small and localized clusters. These clusters are combined into larger ones through a nonlinear optimization scheme in [15] biased toward binary-valued alphas.

The working assumption of the nonlocal principle [2] is that a denoised pixel i is a weighted sum of the pixels with similar appearance to the weights given by a kernel function $\mathcal{K}(i,j)$. Recall in [12] the following:

$$E[X(i)] \approx \sum_j X(j)\, \mathcal{K}(i,j)\, \frac{1}{\mathcal{D}_i}, \tag{3}$$

$$\mathcal{K}(i,j) = \exp\left(-\frac{1}{h_1^2}\|X(i) - X(j)\|_g^2 - \frac{1}{h_2^2} d_{ij}^2\right), \tag{4}$$

$$\mathcal{D}_i = \sum_j \mathcal{K}(i,j), \tag{5}$$

where $X(i)$ is a feature vector computed using the information at/around pixel i, and d_{ij} is the pixel distance between pixels i and j, $\|\cdot\|_g$ is a norm weighted by a center-weighted Gaussian, h_1 and h_2 are some constants found empirically. By analogy of (3), the expected value of the alpha matte

$$E[\alpha_i] \approx \sum_j \alpha_j \mathcal{K}(i,j)\, \frac{1}{\mathcal{D}_i} \text{ or } \mathcal{D}_i \alpha_i \approx \mathcal{K}(i,\cdot)^T \boldsymbol{\alpha}, \tag{6}$$

where $\boldsymbol{\alpha}$ is the vector of all α values over the input image. As described in [12]:

- the nonlocal principle applies to $\boldsymbol{\alpha}$ as in (6);
- the conditional distribution $\boldsymbol{\alpha}$ given X is $E[\alpha_i | X(i) = X(j)] = \alpha_j$, that is, pixels having the same appearance are expected to share the same alpha value.

The *nonlocal* principle of alpha matting basically replaces the *local* color-line assumption of [14], [15] applied in a local window, which, although widely adopted, can be easily violated in practice when large kernels are used (such as [10]).

Following the derivation $\mathcal{D}\boldsymbol{\alpha} \approx \mathcal{A}\boldsymbol{\alpha}$, where $\mathcal{A} = [\mathcal{K}(i,j)]$ is an $N \times N$ *affinity matrix* and $\mathcal{D} = \mathrm{diag}(\mathcal{D}_i)$ is an $N \times N$ diagonal matrix, where N is the total number of pixels. Thus, $(\mathcal{D} - \mathcal{A})\boldsymbol{\alpha} \approx \mathbf{0}$ or $\boldsymbol{\alpha}^T L_c \boldsymbol{\alpha} \approx 0$, where $L_c = (\mathcal{D} - \mathcal{A})^T (\mathcal{D} - \mathcal{A})$ is called the *clustering Laplacian*. This basically solves the quadratic minimization problem, $\min_{\boldsymbol{\alpha}} \sum \mathcal{A}_{ij} (\alpha_i - \alpha_j)^2$.

In nonlocal matting, the extraction Laplacian (whose derivation is more involved) rather than the above simpler clustering Laplacian was used in tandem with user-supplied input for alpha matting. While it was shown for clustering Laplacian in [12] that sparse input suffices for good results, the estimated alphas along edges are not accurate due to the use of spatial patches in computing affinities $\mathcal{A}$. Moreover, the implementation in [12] requires a sufficiently large kernel for collecting and matching nonlocal neighborhoods, so specialized implementation considerations are needed to make it practical (c.f., a nice proof in fast matting [10]). The choice of parameters h_1 and h_2 also affect results quality.

4 KNN Matting

In the following, we describe and analyze our technical contributions of KNN matting, which does not rely on the local color-line model, does not apply regularization, does not apply machine learning, and does not have the issue of kernel size. They look straightforward at first glance (with the corresponding implementation definitely straightforward); our analysis and experimental results, on the other hand, show that our approach provides a simple, fast, and better solution than nonlocal matting [12], with an elegant

Fig. 3. KNN and nonlocal affinities comparison given the same pixel (marked white). Nonlocal matting uses a spatial window centered at the given pixel for sampling nonlocal neighborhoods (radius $= 9$ in [12]). KNN matting collects more matching neighborhoods globally rather than within an image window, while spending significantly less computation time ($K = 81$ here).

generalization to multiple layers extraction. Our unoptimized Matlab implementation runs in a few seconds on 800×600 examples available at the alpha matting evaluation website [20] and our results were ranked high in [20] among the state of the art in natural image matting, which may require a complicated implementation. In most cases, only one click is needed for extracting each material layer from SVBRDF data [11] in material matting.

4.1 Computing $\mathcal{A}$ Using KNN

Computing $\mathcal{A}$ in KNN matting involves collecting nonlocal neighborhoods j of a pixel i before their feature vectors $X(\cdot)$s are matched using $\mathcal{K}(i, j)$.

Rather than using a large kernel as in fast matting and nonlocal matting, both operating in the spatial image domain, given a pixel i, we implement the nonlocal principle by computing KNN in the feature space. Our implementation was made easy by using FLANN [25], which is demonstrated to be very efficient in practice, running on the order of a few seconds for an 800×600 image in natural image matting. We notice in nonlocal matting [12] that special implementation considerations and restrictions were needed to cope with the computation load involving large kernels. Since kernel size is not an issue in this paper due to efficient KNN search, the running time for computing one row of $\mathcal{A}$ is $O(Kq)$, where $O(q)$ is the per-query time in FLANN. $\mathcal{A}$ has up to $2NK$ entries and recall that since $\mathcal{K}(i, j) = \mathcal{K}(j, i)$, $\mathcal{A}$ is a symmetric matrix. Fig. 3 compares the nonlocal neighborhoods computed using KNN matting and nonlocal matting [12], showing the efficacy of KNN searching in feature space in implementing the nonlocal principle. Fig. 4 visualizes a typical $\mathcal{A}$ computed in KNN matting.

Typical values of K range from only 3 (for material matting with more descriptive feature vector) to 15 (for natural image matting). Despite the fact that K is not a critical parameter and is kept constant in our experiments, processing speed and memory consumption are issues. Without compromising the result quality, that is, to build sufficient relations among pixels, smaller K means a shorter KNN search time as well as a shorter time for solving a sparser/faster linear system. On the other hand, a very large K will produce undesired artifacts in the alpha result, where a larger number of irrelevant matches will start to take its toll, not to mention the 12-GB memory requirement when $K > 300$. Fig. 5 shows a qualitative comparison under different values of K. See the supplemental materials, which can be found in the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPAMI.2013.18, for more comparisons.

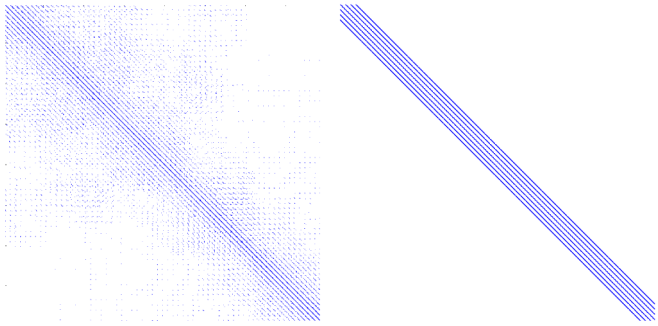


Fig. 4. Typical nonlocal affinities matrix $\mathcal{A}$ in KNN matting (left, with $K = 10$) which is not as strongly diagonal as its counterpart from nonlocal matting (right, with radius $= 3$). The KNN matrix is still sparse.

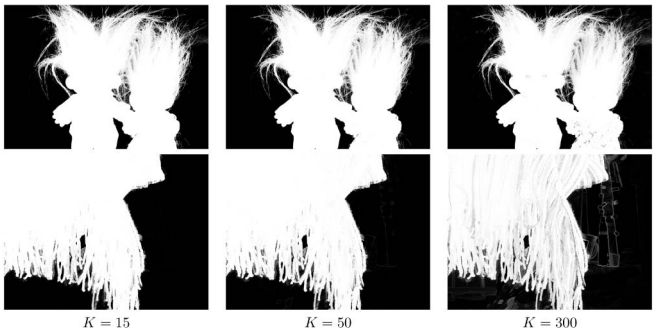


Fig. 5. Parameter K is not critical. Although the results are similar, smaller K means faster solving time and fewer artifacts caused by irrelevant matches when $K = 300$.

4.2 Feature Vector X with Spatial Coordinates

For natural matting, a feature vector $X(i)$ at a given pixel i that includes spatial coordinates to reinforce spatial coherence can be defined as

$$X(i) = (\cos(h), \sin(h), s, v, x, y)_i, \tag{7}$$

where h, s, v are the respective HSV coordinates and (x, y) are the spatial coordinates of pixel i. As shown in Fig. 6, KNN matting is better on HSV than RGB color space on the *troll* example. Few previous matting approaches use the HSV color space. Feature vector can be analogously defined for material matting by concatenating pixel observations under various lighting directions, which forms a high-dimensional vector. For material without exhibiting spatial coherence (e.g., spray paint) the spatial coordinates can be turned off.

Note the differences with nonlocal matting in encoding spatial coherence: Spatial coordinates are incorporated as part of our feature vector rather than considered separately using d_{ij} in nonlocal matting (see (4)) with empirical setting of h_2 to control its influence. Further, an image patch centered at a pixel [12] is not used in our feature vector definition. As will be demonstrated in our extensive experimental results, without the added information of a larger patch, KNN matting ranks high among the state of the art [20].

4.3 Kernel Function $\mathcal{K}(i, j)$ for Soft Segmentation

We analyze common choices of kernel function $\mathcal{K}(x)$ to justify ours, which is $1 - x$:

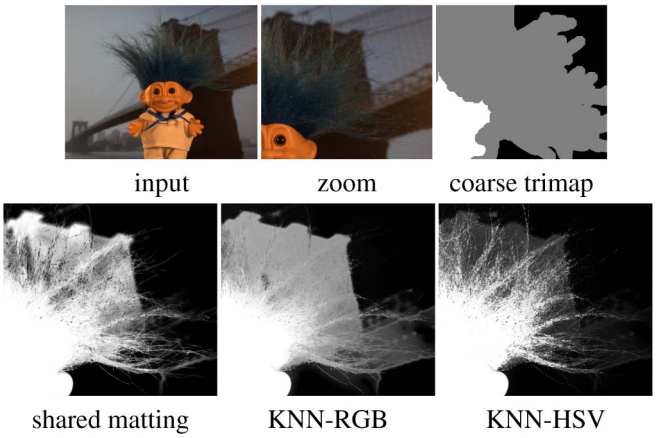


Fig. 6. KNN matting can operate in any color space simply by changing the definition of the feature vector in (7). Here we show significant improvement in the result of *troll* using the HSV space on a coarse trimap. The hairs and the bridge are dark, with close color values in the RGB space: a hair pixel has RGB (20, 31, 33) and a bridge pixel (40, 30, 33) in 255 scale, whereas the hue of the hair is 126 degrees and that of bridge is 15 degrees.

$$\mathcal{K}(i,j) = 1 - \frac{\|X(i) - X(j)\|}{C}, \tag{8}$$

where C is the least upper bound of $\|X(i) - X(j)\|$ to make $\mathcal{K}(i,j) \in [0,1]$. Because (8) puts equal emphasis over the range $[0,1]$, not biasing to either foreground or background, the three overlapping layers can be faithfully extracted as shown in Fig. 7. There is no parameter to set (c.f., h_1 in (4)) and KNN allows returning the smallest $\|X(i) - X(j)\|$.

A typical choice of kernels in machine learning, $\exp(-x)$, was used in [12]. We argue it is not a good choice for modeling optical blur and soft segmentation and, in fact, it favors hard segmentation: Fig. 7 shows a synthetic example where three layers are blended by fractional alphas; the same KNN matting is run on this image except that the kernel function is replaced by $\exp(-x)$. As shown in the figure, hard segments are obtained. The hard segmentation results can be attributed to the nonmaximal suppression property of the Gaussian kernel, where nonforeground (or nonbackground) is heavily penalized by the long tail of the Gaussian.

In nonlocal matting [12], Lee and Wu noted that the clustering Laplacian causes inaccuracy around edges, while we believe the major cause may be due to their use of the exponential term in the kernel function. Barring factors such as image outliers and color shifts due to Bayer patterns, suppose $F = (1,0,0)$ and $B = (0,0,0)$. For a pixel's value $E = (0.3,0,0)$, using (4) without the spatial term, $\mathcal{K}(F,E) = \exp(-\|F-E\|^2/h_1^2) = \exp(-0.7^2/0.01) = \exp(-49)$ and $\mathcal{K}(B,E) = \exp(-0.3^2/0.01) = \exp(-9)$. $\mathcal{K}(F,E) \ll \mathcal{K}(B,E)$, making $\mathcal{K}(F,E)$ negligible and biasing the solution toward B, and thus hard segmentation results. Numerically, this also causes instability in computing their clustering Laplacian, which is susceptible to singularity because many terms are negligibly small.

4.4 Closed-Form Solution with Fast Implementation

While the clustering Laplacian $L_c = (\mathcal{D} - \mathcal{A})^T(\mathcal{D} - \mathcal{A})$ is conducive to good graph clusters, the Laplacian $L = \mathcal{D} - \mathcal{A}$ is sparser while running much faster (up to 100 times faster than L_c) without compromising the results except for a few

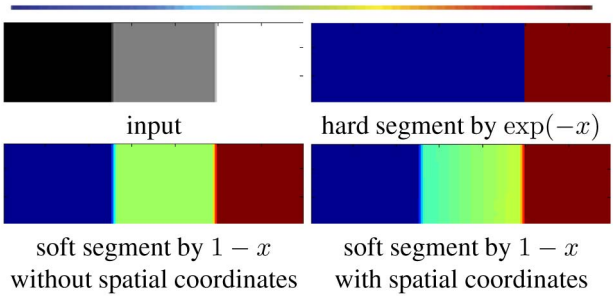


Fig. 7. The $\exp(-x)$ term tends to generate hard segments, although the input consists of overlapping image layers. On the contrary, the $1-x$ term without spatial coordinates produces soft segments closer to the ground truth. Moreover, using the $1-x$ term with spatial coordinates, we can generate an alpha matte with smoother transition between neighboring pixels.

more user inputs being required to achieve similar visual results. This can be regarded as a tradeoff between running time, amount of user input, and result qualities. Without loss of generality, L is used in this section.

When user input in the form of trimaps or scribbles comes along, it can be shown that the closed-form solution for extracting $n \geq 2$ layers is:

$$(L + \lambda D)\sum_{i}^{n} \boldsymbol{\alpha}_i = \lambda \mathbf{m}, \tag{9}$$

where $D = \mathrm{diag}(\mathbf{m})$ and $\mathbf{m}$ is a binary vector of indices of all the marked-up pixels, and λ is a constant controlling user's confidence on the markups. Our optimization function $g(x)$ has a closed-form solution:

$$g(x) = x^T L x + \lambda \sum_{i \in \mathbf{m}-\mathbf{v}} x_i^2 + \lambda \sum_{i \in \mathbf{v}} (x_i - 1)^2, \tag{10}$$

where $\mathbf{v}$ is a binary vector of pixel indices corresponding to user markups for a given layer. Then, $g(x)$ is

$$\begin{aligned} & x^T L x + \lambda \sum_{i \in \mathbf{m}-\mathbf{v}} x_i^2 + \lambda \sum_{i \in \mathbf{v}} x_i^2 - 2\lambda \mathbf{v}^T x + \lambda |\mathbf{v}| \\ &= x^T L x + \lambda \sum_{i \in \mathbf{m}} x_i^2 - 2\lambda \mathbf{v}^T x + \lambda |\mathbf{v}| \\ &= \frac{1}{2} x^T 2(L + \lambda D) x - 2\lambda \mathbf{v}^T x + \lambda |\mathbf{v}| \\ &= \frac{1}{2} x^T H x - c^T x + \lambda |\mathbf{v}|, \end{aligned}$$

where $\lambda|\mathbf{v}|$ is a constant. Note that $H = 2(L + \lambda D)$ is positive semidefinite because L is positive semidefinite and D is diagonal matrix produced by the binary vector $\mathbf{m}$. Differentiating $g(x)$ w.r.t. x and equating the result to zero:

$$\frac{\partial g}{\partial x} = Hx - c = 0. \tag{11}$$

Thus. the optimal solution is

$$H^{-1}c = (L + \lambda D)^{-1}(\lambda \mathbf{v}). \tag{12}$$

This echoes Lemma 1 in [12] that contributes a smaller and more accurate solver than the one in [30], which gives the optimal solution in closed form.

Rather than using the coarse-to-fine technique in the solver in [14], since H is a large and sparse matrix which is symmetric and semipositive definite we can leverage the PCG [1] running about five times faster than the conventional conjugate method (we use `ichol` provided in Matlab 2011b as the preconditioner), on the order of a few seconds for solving input images available at the alpha matting evaluation website. We also note that in [10] the traditional LU decomposition method and conjugate gradient method were compared. The iterative conjugate gradient method was used because, for their large kernels, information propagation can be faster.

4.5 Summation Property

KNN matting in its general form for extracting $n \geq 2$ layers satisfies the summation property, that is, the estimated alphas at any given pixel sum up to 1. From (11):

$$\begin{aligned}(L+\lambda D)\boldsymbol{\alpha}_1 &= \lambda \mathbf{v}_1\\ &\vdots\\ (L+\lambda D)\boldsymbol{\alpha}_n &= \lambda \mathbf{v}_n\end{aligned}$$

gives

$$(L+\lambda D)\sum_{i=1}^{n}\boldsymbol{\alpha}_i = \lambda\sum_{i=1}^{n}\mathbf{v}_i = \lambda\mathbf{m}. \tag{13}$$

Since

$$(L+\lambda D)\mathbf{1} = \lambda D\mathbf{1} = \lambda\mathbf{m}, \tag{14}$$

as the nullspace of Laplacian L is $\mathbf{1}$ a constant vector with all 1s. Since $L+\lambda D$ is invertible, $\sum_{i=1}^{n}\boldsymbol{\alpha}_i = \mathbf{1}$.

In [22, Theorem 2], the summation property was also shown for multiple layer extraction for alpha matting RGB images, where the same Laplacian from [14], [15] was still used. In practice, KNN matting's output alphas are almost within $[0,1]$. However, the summation property does not hold for sampling-based algorithms such as [9] when it comes to multiple layer extraction: To obtain the alpha matte of a layer, this layer is regarded as foreground while others are background. Consider three layers, $L_1 = (1,0,0)$, $L_2 = (0,1,0)$, $L_3 = (0,0,1)$, and the pixel $I = (\frac{1}{3},\frac{1}{3},\frac{1}{3})$. To obtain the alpha matte of L_1, let L_1 be foreground F and the union of L_2 and L_3 be background B. According to (2) in [9], $\alpha = \frac{(I-B)(F-B)}{\|F-B\|^2}$; the alpha value for L_1 is 0.5. Similarly, the alpha value for L_2 or L_3 is also 0.5. Consequently, they sum up to 1.5. Normalization may help, but the normalization factor will vary from pixel to pixel. Also, the approach in [9] cannot be easily generalized to handle multiple layers due to the potentially prohibitive joint layer space when more than two layers are involved.

5 Results on Alpha Estimation

We first show in this section the results on material matting ($n \geq 2$ layers) on SVBRDF data from [11]. Then, we will show results on natural image matting ($n = 2$) using real images as well as the examples in [20], calling attention to state of the art such as CF matting [14], nonlocal matting [12], fast and global matting [10], [9], learning-based (LB) matting [30], SVR matting [29], and weighted color matting [21]. All of our results, including the natural image matting results and their comparisons with state-of-the-art techniques, are included in the online supplemental materials. Due to space limits, here we highlight a few results.

5.1 Material Matting

We first present results on material matting for extracting more than two alphas at a given pixel.

Related work. Much work has been done on BRDF decomposition, aiming at reducing the dimensionality of an SVBRDF, which is 6D in its general form. Decompositions returned by principal component analysis and independent component analysis and its extensions do not in general correspond to different materials and thus are not conducive to high-level editing. Factorization approaches such as homomorphic factorization [17] and matrix factorization [5] decompose a BRDF into smaller parts, but such decompositions also do not promise that individual segments correspond to single materials. Soft segmentation is required when different materials blend together. Blending weights are available in [11], where an SVBRDF was decomposed into specular and diffuse basis components that are homogeneous, as previously done in [7]. In [13], an SVBRDF was separated into simpler components with opacity maps. The probabilistic formulation takes into consideration local and texture variations in their two-layer separation, and was applied successively rather than simultaneously to extract multiple material layers, so accumulation errors may occur.

Experimental results. The clustering Laplacian was used in our material matting experiments, where a few user-supplied clicks are all that KNN matting needed to produce satisfactory results shown in Figs. 2 and 8. On average, only one click per layer is needed. In *sg*, five overlapping material mattes are produced; despite the fact that the matte for "blue paper" has several disconnected components, one click is all it takes for matting the material. KNN matting produces good mattes for *dove*, where the moon and the sky mattes are soft segments, and also for *wp1*, where hard segments should be produced. In *wt*, the scotch tape (invisible here) was correctly matted out. In *wp2* (see in the online supplemental material), the silver foil is brushed in three general directions, which produces different BRDF responses distinguishable in the feature space for KNN matting to output the visually correct result. In a more challenging dataset *mask*, subtle materials such as the lips and the gem were matted out. This mask example is arguably more challenging than the above for the following reasons: We used budget capture equipment (c.f., precision equipment in [11]), the object geometry is highly complex and produces a lot of cast shadows (c.f., relative flat geometry in [11]), the mixing of the blue and gold paints introduces a lot of color ambiguities. As shown in the figure, more input clicks are required to produce good results. Here, spatial coordinates were not included in defining a feature vector (7) where SVBRDF does not usually exhibit strong spatial coherence. Table 1 tabulates the running times of all of the SVBRDF examples used in this paper. Thanks to FLANN computing, the Laplacian takes only a few seconds for matching nonlocal neighborhoods even when they are far away in the spatial domain. After computing Laplacians, individual layer extraction can be executed in parallel, so we record the maximum

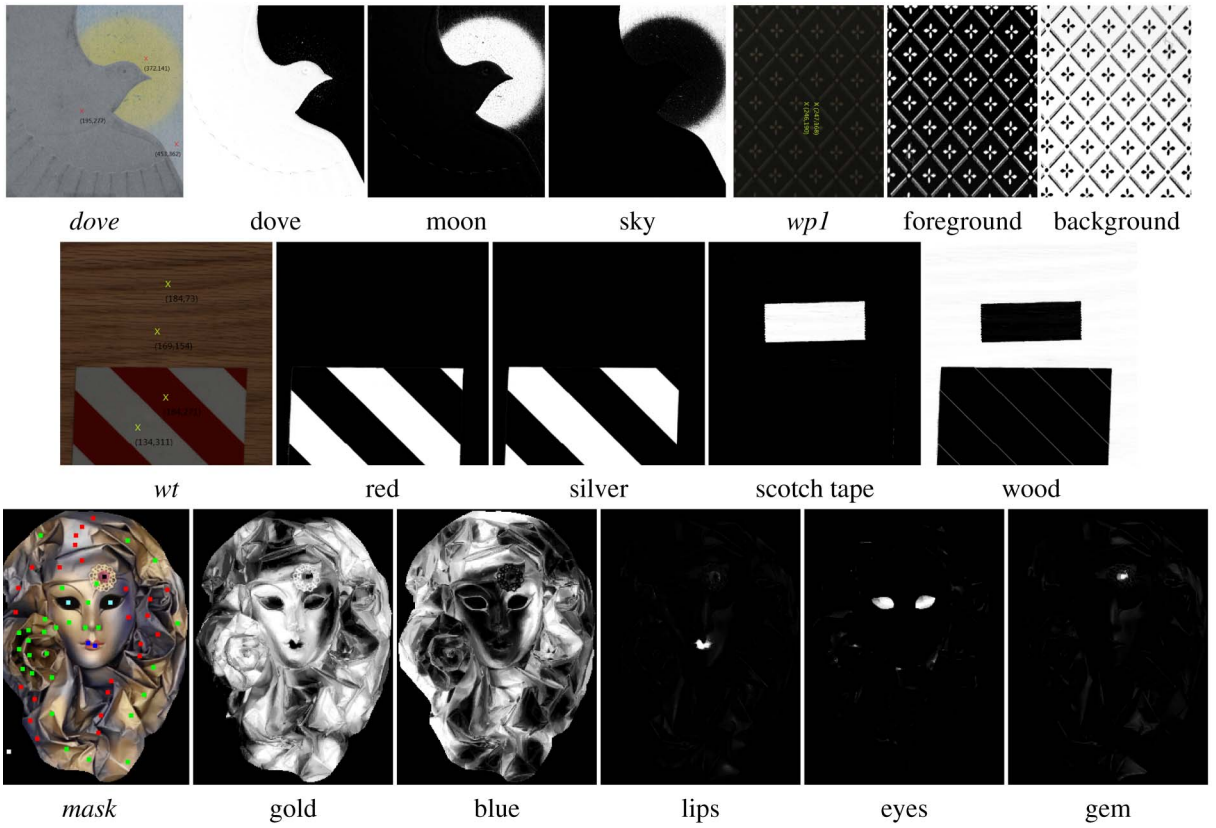


Fig. 8. KNN matting on material matting. In most cases, only one click per layer is needed. In *mask*, clicks with the same color belong to one layer. See all of the material matting result images in the online supplemental material.

extraction time among all layers for each example. More details are available in the online supplemental material.

5.2 Natural Image Matting

The Laplacian $L = \mathcal{D} - \mathcal{A}$ was used in KNN matting in this section to obtain a sparser system for efficiency in our natural image matting experiments. Recall at the beginning of Section 4.4 the difference with the clustering Laplacian.

Table 2 tabulates the partial ranking among the methods evaluated in [20], showing that KNN matting is competitive overall on the same dense trimaps. Fig. 9 shows the qualitative comparison of selected examples on fuzzy objects and objects with holes (with complete results and comparison with CF and LB matting in [20] available in the online supplemental material), noting the *pineapple* used in [10] as a failure case on local color-line assumption [14], whereas KNN matting performed better than shared matting on this example (Fig. 9 and Table 2) without sophisticated sampling and learning strategies, such as [29], [21].

KNN matting gives top performance on difficult images (*plastic bag* and *pineapple*, Fig. 9) while [20] does not rank us high on arguably easier ones (*donkey* and *elephant*, see in the online supplemental material), although we obtain good alpha mattes quantitatively the same as other top-ranked methods on such easier examples. For this reason, we define the *normalized score* of a method given a trimap as the ratio of the best MSE for that trimap to its MSE. We argue that normalized scores are fairer than average ranks: For the *donkey* user-trimap, at the time of writing, the third to 15th methods have the *same* MSE 0.3, but shared matting ranks third, while large kernel matting ranks 15th. In summary, regardless of ranking methods, given the trimaps from [20], our results are better than CF matting [14], fast and global matting [10], [9], and are visually similar to the high-quality results of shared matting [6], weighted color matting [21], and SVR matting [29]. Among all the methods available on [20] at the time of writing, KNN matting is the second best approach in terms of normalized score. The best scorer, SVR matting [29], is LB, where training data is an issue. KNN matting does not require any learning while producing comparable results.

At times a lay user may not be able provide detailed trimaps akin to those in [20]; a few clicks or thin strokes are expected. Fig. 1 shows our visually better results compared with nonlocal matting [12] based on the same input clicks

TABLE 1
Running Times in Seconds for Material Matting on a Machine with 3.4-GHz CPU

		Laplacian	n	max
wt	$400 \times 380 \times 162$	4.9	4	232.7
sg	$500 \times 523 \times 147$	4.5	5	272.1
wp1	$375 \times 480 \times 153$	5.52	2	153.4
wp2	$310 \times 390 \times 153$	3.1	4	65.7
dove	$510 \times 470 \times 141$	7.0	3	127.9
mask	$320 \times 232 \times 93$	1.34	5	52

n is the number of layers; each can be computed in parallel after the Laplacian is computed. Running times shown here are the time for computing the Laplacian and the maximum time for computing an alpha layer in each example. Refer to the online supplemental material for other details.

TABLE 2
Excerpt of the Ranking Information from the Alpha Matting Evaluation Website [20]

	avg ranking	avg user	plastic bag	pineapple	elephant	net	avg MSE	normalized score
SVR	4.7	4.1	2	1	6	6	0.733	86.99
Weighted color	5.9	6.1	6	3	11	5	0.808	74.44
Global	5.9	6.4	8	7	9	2	0.812	73.86
Shared	6.5	5.5	14	4	4	1	0.904	70.48
KNN	**7.3**	**6.0**	**1**	**2**	**12**	**9**	**0.738**	**75.92**
Segmentation	7.4	7.4	16	10	1	10	1.046	69.45
Improved color	7.5	7	5	8	7	8	0.887	67.96
Local spline	9.3	9.8	4	12	8	15	1.029	62.10
LMSPIR	9.7	9.9	11	5	10	4	0.979	62.28
Learning based	9.9	10.3	3	19	3	18	1.175	61.96
Closed-Form	10	22.8	10	17	5	20	1.2625	59.79
Global Sampling	10.2	10.6	12	9	16	3	0.958	60.08
Shared (Real Time)	10.9	10.6	15	6	14	7	1.079	57.44
Large Kernel	11.6	10.8	7	11	13	13	1.145	56.45
Cell-based	12.4	11.9	9	14	18	12	1.091	51.74
Robust	12.8	14.1	13	15	15	14	1.287	49.69
High-res	14.1	13.8	20	16	2	11	1.592	46.08
Random Walk	18	19.1	18	20	17	21	2.687	32.64
Iterative BP	18.2	18.9	21	18	19	17	1.991	30.58
Geodesic	19.2	19	19	13	22	19	2.362	22.05
Bayesian	20.7	20.1	17	22	20	16	2.083	18.10
Easy	21	21	22	21	21	22	3.971	19.03
Poisson	22.9	22.9	23	23	23	23	6.592	9.68

Normalized score *is defined in the text. Without any learning process [29] or sophisticated sampling strategy [21], [9], KNN ranks top in both average ranking and normalized score. Complete ranking information is in the online supplemental material.*

used in the paper. Fig. 10 compares the results on very sparse input, showing that KNN matting preserves the fuzzy boundaries as well as the solid portions of the foreground better than other state of the arts. Fig. 11 shows the MSE comparison of our method with closed form matting, spectral matting, LB matting on six examples with ground truth, where the input consists of only a few strokes.

In [20], most images are shot in front of a computer screen that may not accurately represent natural images in real applications. Fig. 12 shows KNN matting results on real photos. Notice without the large hue difference induced by a computer screen, KNN matting is still capable of extracting the details of hair in real photos.

The failure mode of KNN matting is shown in Fig. 13. Our method degrades under severe color ambiguity because color information largely dominates our feature vector (7). On the other hand, a blurry image in general is modeled by image convolution rather than the image compositing equation (1) assumed in alpha matting. Recent work [16] tackled this problem by adding a motion regularization term to the Laplacian energy function. Fig. 14 shows more comparisons from [20].

6 Layer Estimation

Most existing works on natural image matting focus only on alpha extraction, with the few exceptions described in the related work section. To matte out the foreground, αI is usually applied. Using the same alpha, Fig. 15 shows one example where αF is more faithful than αI in foreground extraction, which we believe should be done when F can be reliably estimated.

Given $\boldsymbol{\alpha}$, we show in this section that respective image layers F_i can be reliably extracted simultaneously in closed form by solving a similar Laplacian energy introduced in the previous sections. Thus, our method not only generalizes to $n \geq 2$ layers but also provides a uniform and easy-to-implement scheme for both alpha and layer estimation.

As was done in the image matting works reviewed in the related work section, where layer extraction was addressed, while our objective function still makes use of the compositing equation to encode the data term, we harness the power of KNN for searching matching neighbors in the feature space in a nonlocal manner, thus also avoiding the drawbacks of the color-line model in a local window in encoding the overall energy, as in the case of our alpha estimation.

Specifically, given $\boldsymbol{\alpha}$, $I = \sum_{i=1}^{n} \alpha_i F_i$ is still an underconstrained system of linear equations with $3nN$ unknowns and $3N$ equations, where N is the total number of pixels and n is the total number of image layers to be estimated at each pixel location. Similar to the assumptions used in alpha estimation, we employ two soft constraints for each layer F_i: Given two pixel locations j and k,

1. if $I(j)$ and $I(k)$ share a similar color, it is likely that $F_i(j) \approx F_i(k)$;
2. if $I(j)$ and $I(k)$ are spatial neighbors, it is likely that $F_i(j) \approx F_i(k)$.

Similarly done in alpha estimation, each pixel can be represented as a feature vector by concatenating its color and location coordinates, with its matching neighbors found by KNN search.

Now, for each color channel, we proceed to define a quadratic energy function that consists of a KNN matching term and a data term, as follows:

$$\sum_{i,j,k} \mathcal{A}_i(j,k)(F_i(j) - F_i(k))^2 + \lambda \sum_k \left(\sum_i \alpha_i(k) F_i(k) - I(k) \right)^2. \tag{15}$$

Compared with alpha estimation, the user markup is already implied in the data term here when $\alpha = 1$ or 0.

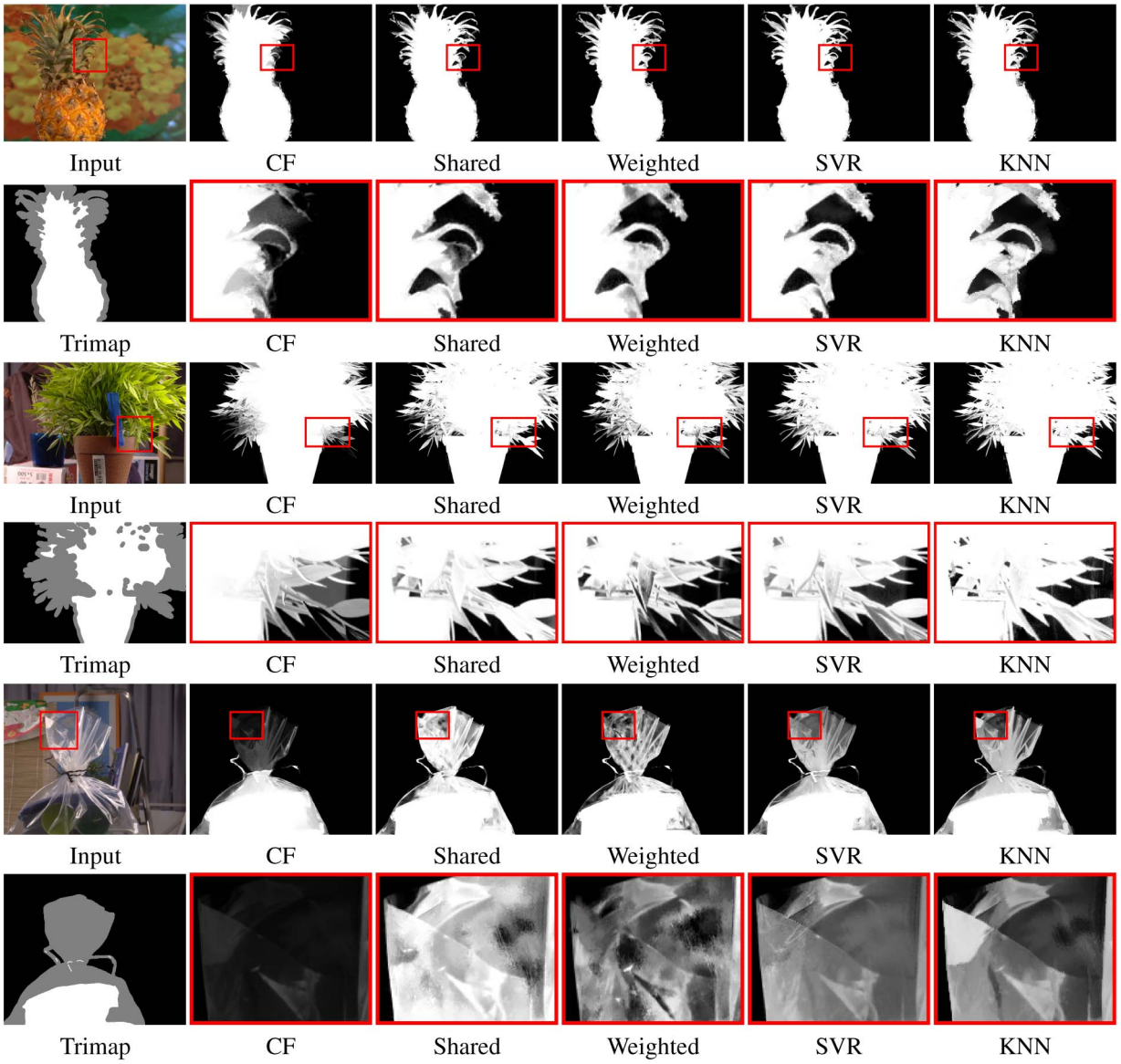


Fig. 9. KNN matting on natural images from [20]. The MSE rankings are from [20]. Top: Images with very similar background/foreground color and fuzzy boundary; KNN ranks the second after SVR. Middle: Images with holes; KNN ranks the fourth with the same MSE as the second and third ranked methods. Bottom: Images with high transparency; KNN ranks the first in this example. This figure is best viewed in the electronic version. More comparisons are available on [20].

We impose in this layer estimation problem stronger spatial coherence along the matte boundary by considering both KNN matching neighbors in encoding the affinity matrix $\mathcal{A} = [A_i(j,k)]$. Mathematically, in the matching term, $\mathcal{A}$ is defined as follows:

$$\begin{aligned} A_i(j,k) &= \min(W_i(j), W_i(k))\mathcal{K}(I_j, I_k), \\ W_i(j) &= 1 - |2\alpha_i(j) - 1|, \end{aligned} \tag{16}$$

where W is used to reweigh pixel contributions, giving more weight to those along the matte boundary which are indicated by smaller alpha values. We believe that using the weight W is more robust than the derivatives of α suggested in (19) in [14]: Consider the case $\alpha_i(j)$ is neither 1 nor 0 (else the case is trivial). If it is equal to its four connected neighbors' alpha values, then the derivative of $\alpha_i(j)$ is zero and only the data term remains effective. Thus, we cannot determine the optimal F_i. When α_i is very close to its four connected neighbors' alpha values, the system to solve tends to be numerically unstable. On the contrary, W is always nonzero when α is neither 1 nor 0.

The solution that minimizes (15) can be found by differentiating the energy function with respect to each unknown $F_i(k)$. The following details the mathematics.

First, let $\mathbf{F}$ be a column vector that concatenates all F_i, $\mathcal{D}$ be a matrix of size $nN \times nN$ defined for each two-tuple $(F_i(j), F_i(k))$ such that $\mathcal{D}[(i-1)N + j, (i-1)N + k] = \alpha_i(j)\alpha_i(k)$. Thus, $\mathcal{D}$ is a block diagonal matrix. In matrix form we have

$$\mathbf{F} = \begin{bmatrix} F_1 \\ F_2 \\ \vdots \\ F_n \end{bmatrix}_{nN \times 1}, \tag{17}$$

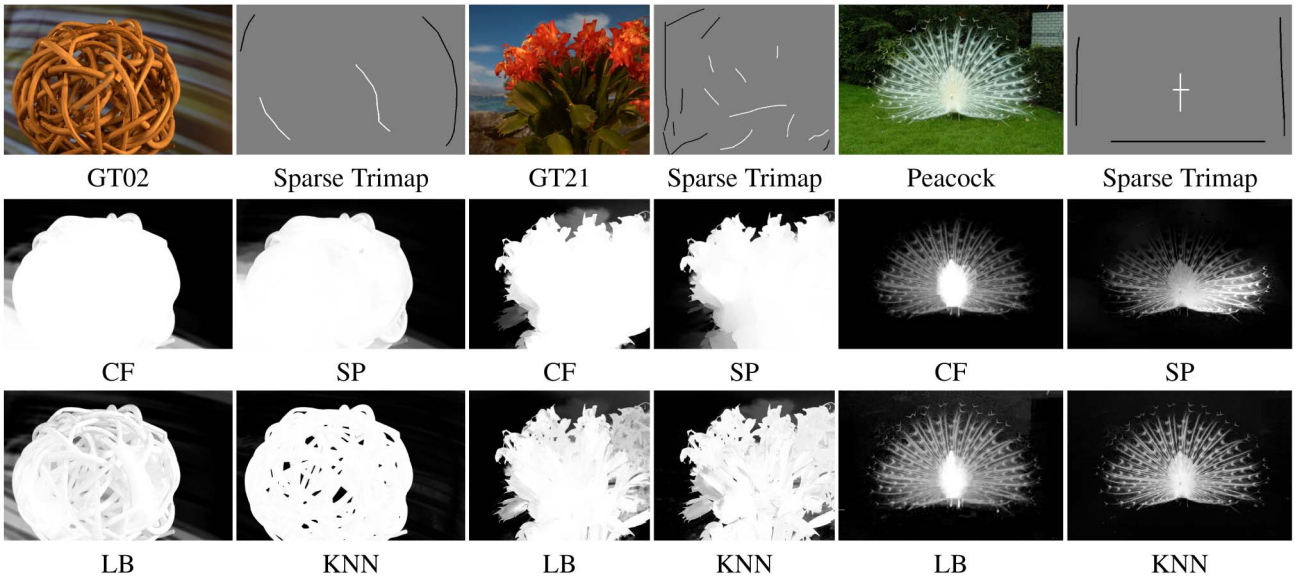


Fig. 10. Comparison on sparse user-supplied trimaps. KNN matting produces better results in around 15 seconds using PCG in each case, whereas it takes 150 seconds for SP matting. See more comparisons in the online supplemental materials.

$$\mathcal{D} = \begin{bmatrix} \boldsymbol{\alpha}_1\boldsymbol{\alpha}_1^T & & & \\ & \boldsymbol{\alpha}_2\boldsymbol{\alpha}_2^T & & \\ & & \ddots & \\ & & & \boldsymbol{\alpha}_n\boldsymbol{\alpha}_n^T \end{bmatrix}_{nN\times nN}, \tag{18}$$

and we let

$$\mathcal{A}' = \begin{bmatrix} \mathcal{A} & & & \\ & \mathcal{A} & & \\ & & \ddots & \\ & & & \mathcal{A} \end{bmatrix}_{nN\times nN}. \tag{19}$$

Let L be the Laplacian matrix derived from $\mathcal{A}'$ and $\mathbf{B}$ is a $nN \times 1$ vector, where $\mathbf{B}_{(i-1)N+k} = \alpha_i(k)I(k)$. By differentiating the energy function and equating the result to zero, we get

$$\begin{aligned} 2L\mathbf{F} + 2\lambda\mathcal{D}\mathbf{F} &= 2\lambda\mathbf{B}, \\ (L + \lambda\mathcal{D})\mathbf{F} &= \lambda\mathbf{B}, \\ \mathbf{F} &= (L + \lambda\mathcal{D})^{-1}\lambda\mathbf{B}. \end{aligned}$$

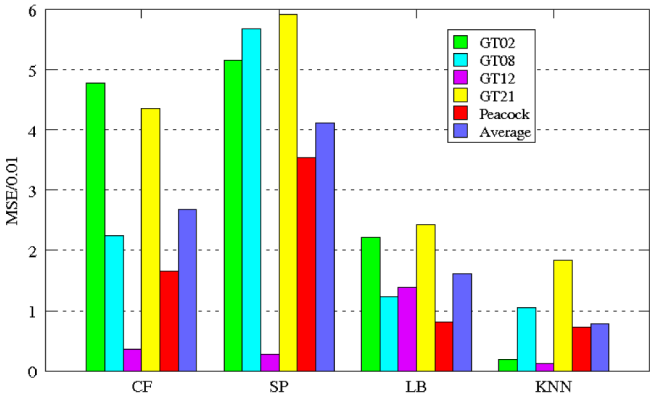


Fig. 11. For very sparse input in Fig. 10, KNN matting is better than other state-of-the-art matting methods that rely on foreground/background color sampling and/or local color line model.

Thus, the closed solution of $\mathbf{F}$ is derived where all layers can be estimated simultaneously in theory. In practice, we adopt an iterative computation scheme such as PCG, which is similarly done in the alpha estimation.

6.1 Qualitative Evaluation

In this section, we show empirically that our layer estimation based on KNN matting can recover more faithful layer color information compared to closed form

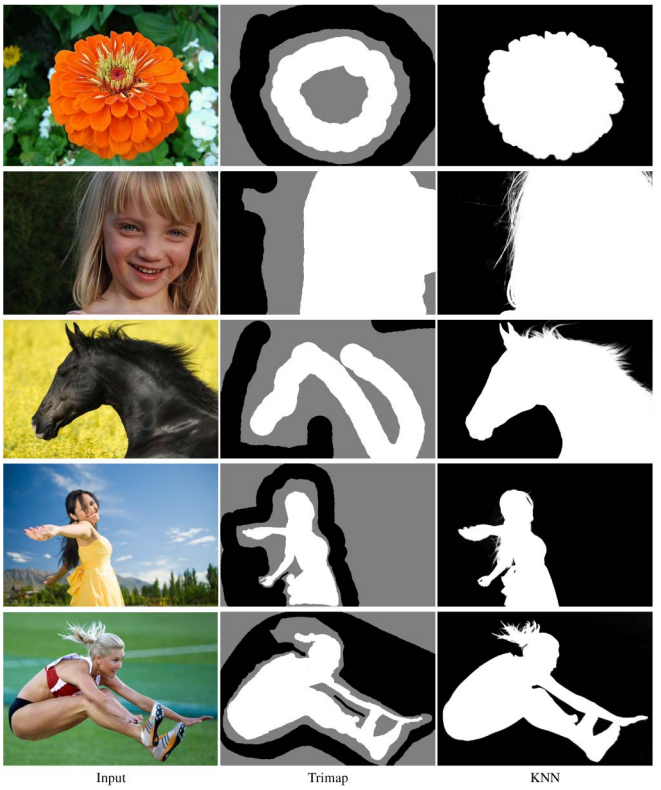


Fig. 12. KNN matting on real photos.

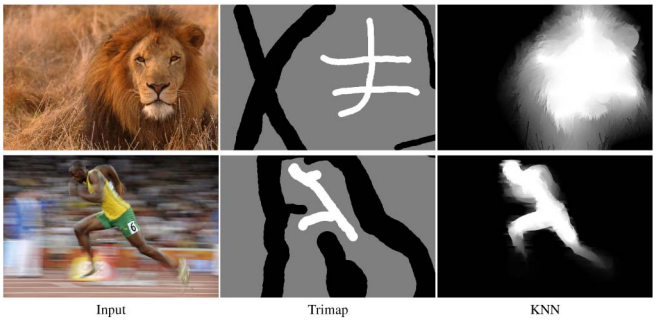


Fig. 13. KNN matting degrades gracefully under color ambiguity and motion blur due to, respectively, insufficient color information and different image model.

Fig. 15. Our layer estimation can better separate the foreground from the background. The pink hair is contaminated with green or purple colors, whereas in our case the hair remains pink.

matting [14] for two-layer matting and [22] for n-layer matting, both of which are based on the color-line model within a local support.

The performance of the tested algorithms differs mostly around fractional boundaries where $\alpha \approx 0.5$ when most ambiguous situations occur. Fig. 16 shows the qualitative comparison on benchmark images of hairy objects obtained from [20]. Note that αI is highly affected by the background color in all of the examples. The layers output by closed form matting are better but cannot outperform our layers, where more fine details are preserved.

Fig. 17 compares the multiple layer extraction results of [22] with those extracted by our method, using the same input images and strokes. As shown in the figure, our

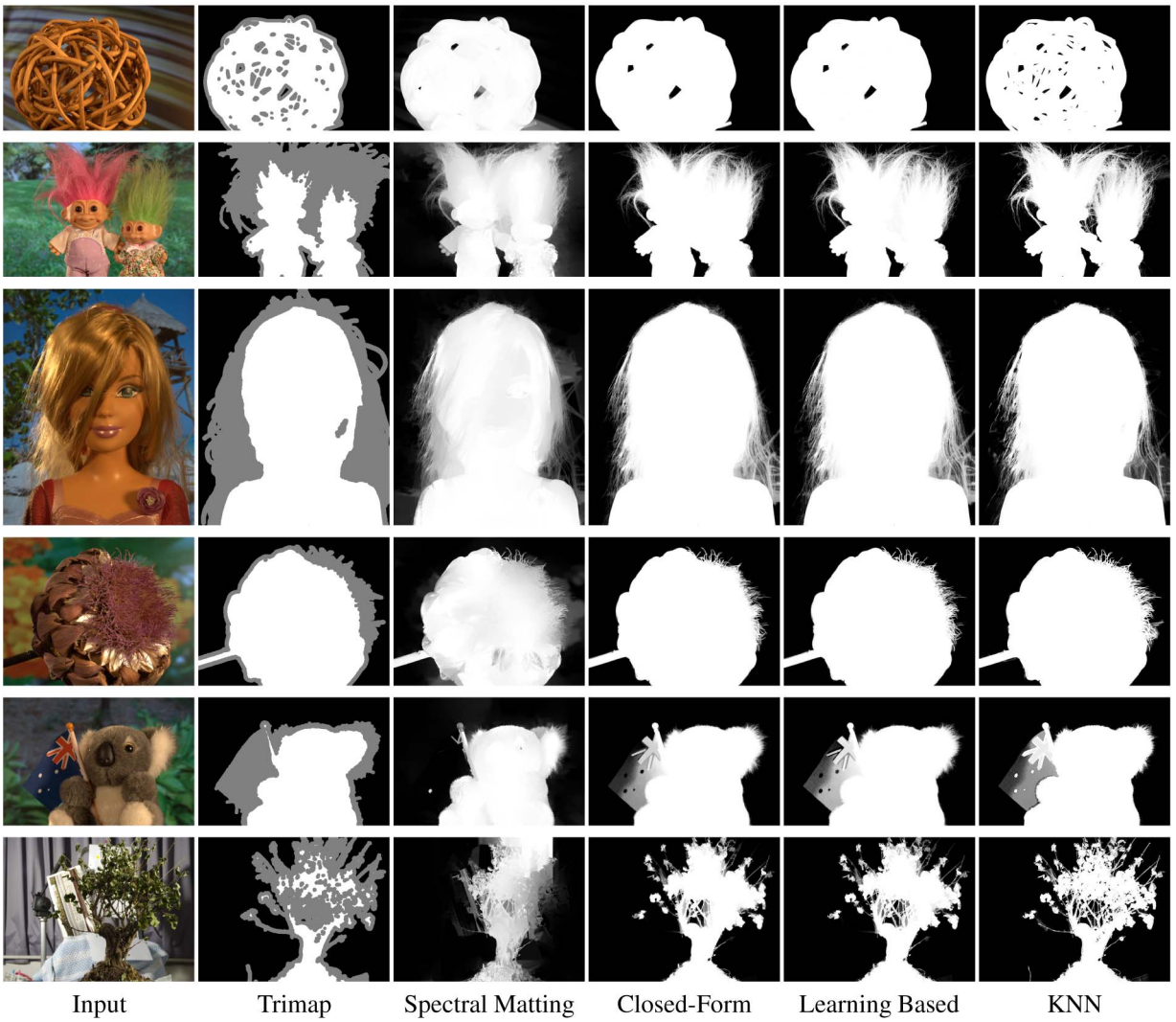


Fig. 14. Natural image matting comparison from [20]. Results of all of the 27 cases are included in the online supplemental material.

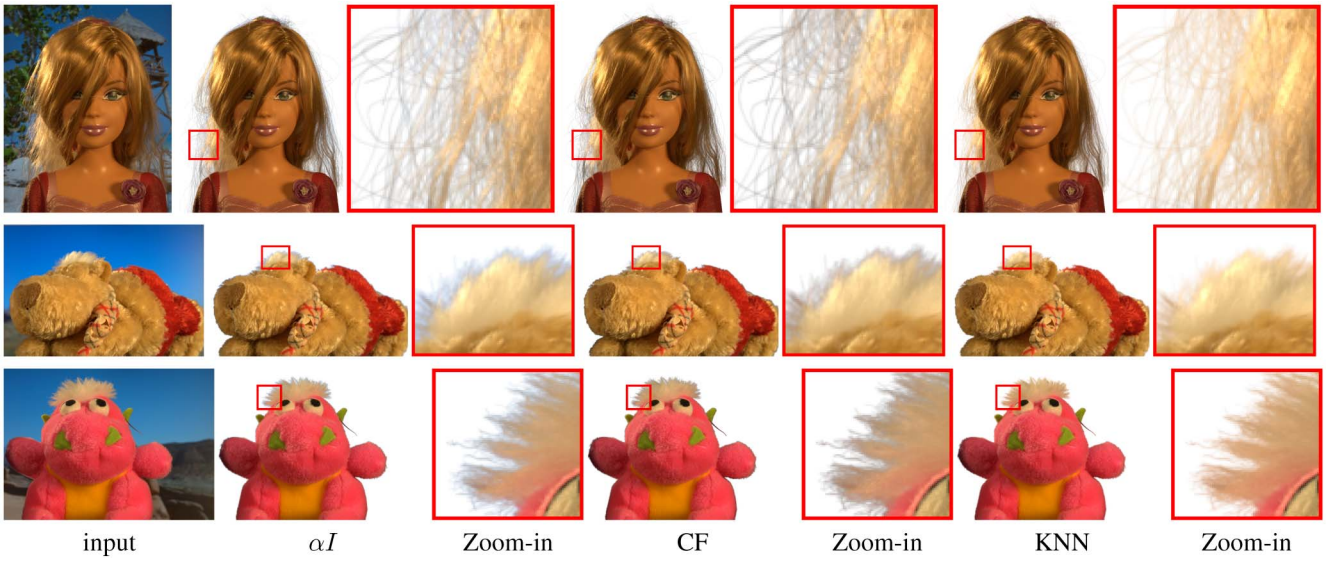


Fig. 16. Qualitative comparison on two-layer extraction with known α. Top: KNN matting preserves the highest amount of details without mixing the background colors. Middle: αI and CF fail to completely eliminate the background blue sky, while the foreground extracted using KNN matting does not visually have any remnant of the background, and preserves more and better details. Bottom: Due to the local color-line model assumption, the end of the hair appears darker than the true color. On the other hand, this artifact is less apparent in the foreground layer extracted by KNN matting.

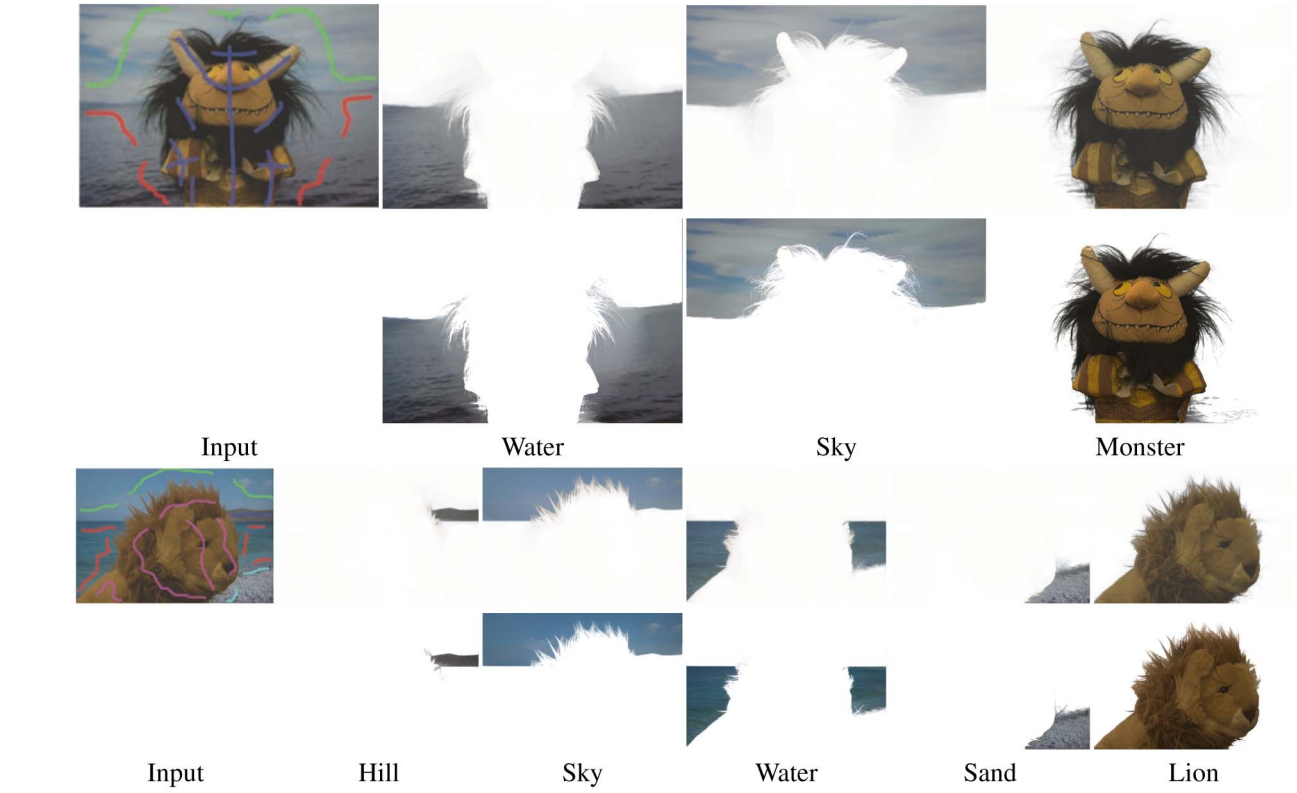


Fig. 17. Qualitative comparison with [22] on n-layer extraction with known α. The top two rows compare their results with ours on the *Monster* example. KNN Matting extracts the sky and the monster layers with less blurring and suffers fewer artifacts around the hair. The bottom two rows show the results on *Lion*. The hair/sky boundary in the sky layer is blurry in their estimation, while our method produces a clearer boundary. Similarly, our sky/lion boundary depicted in the lion layer is sharper in delineating the fine hair strands. Input and output of [22] are courtesy of D. Singaraju.

results present fewer artifacts and are less contaminated by the background in the three layers of *Monster* and five layers of *Lion*.

6.2 Quantitative Evaluation

To quantitatively evaluate our layer estimation results, we tabulate the errors against known or ground-truth foregrounds; the latter are computed using the following scheme. Our evaluation here still focuses on comparing the local color-line model and the nonlocal principle.

To obtain ground-truth foreground, we use images of furry objects shot in front of a blue screen. Theoretically, given a known background B and α, we can get $F = (I - (1 - \alpha)B)/\alpha$. In practice, however, this method is

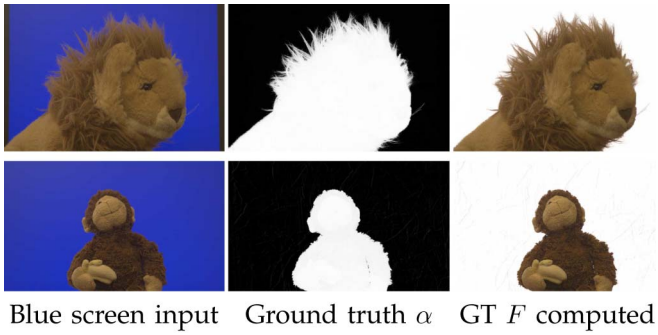


Fig. 18. Ground-truth foreground image computed using our proposed method.

not stable because α can be zero or very close to zero at some pixels. Also, $I-(1-\alpha)B$ may be negative when α or B is in fact not accurate. To tackle this problem, while one can use blue screen matting [23] we propose an alternative by solving the following energy function to obtain our ground-truth F when α and B are given:

$$\begin{aligned}&\|I-\alpha F-(1-\alpha)B\|_2^2\\&\quad+\lambda_B\sum\|B(i)-B(j)\|_2^2+\lambda_F\sum\|F(i)-F(j)\|_2^2,\end{aligned}\tag{20}$$

where pixels i and j are spatial neighbors. We impose strong spatial coherence on B, which is the blue or constant-colored background, and weaker spatial coherence on F to avoid overfitting: In our experiments, we set $\lambda_B=1$ and $\lambda_F=0.01$. We obtain very good ground truth even when the background is noisy and contains more than one color. Fig. 18 shows one set of sample images with the computed ground-truth foregrounds.

Fig. 19 shows the quantitative sum of absolute difference (SAD) comparison results on the 21 images available from the dataset in [15] where the ground-truth foregrounds are computed using our method described above. In almost all cases, our layer extraction based on KNN matting produces the lowest error among the three approaches. Fig. 20 shows the result of αI, closed form, and KNN matting, as well as the difference between the respective ground-truth foregrounds. The difference images are boosted by histogram equalization for visualization purpose.

7 Conclusion

Rather than adopting the color-line model assumption in a local window or relying on sophisticated sampling strategies on foreground and background pixels, or any learning strategy where training data is an issue, we propose KNN matting that employs the nonlocal principle for natural image matting and material matting, taking a significant step toward producing a fast system that outputs better or competitive results and is easier to implement (our implementation only has about 50 lines of Matlab codes, see the online supplemental material; also available at the first author's website). It generalizes well to extracting $n\geq 2$ multiple layers in non-RGB color space in any dimensions where kernel size is also not an issue. Our general alpha matting approach allows the simultaneous extraction of multiple overlapping layers based on sparse input trimaps and outputs alphas satisfying the summation property. Extensive experiments and comparisons using standard datasets show that our method is competitive among the state of the art. Meanwhile, because KNN matting constructs clustering Laplacian based on feature vector, the choice of elements in feature vector is instrumental.

In this paper, we show that the same Laplacian formulation can be used for layer extraction once the alpha values are known. The above implementation can be directly deployed. We performed qualitative and quantitative evaluation for extracting overlapping layers in natural image matting where the number of layers $n\geq 2$. Our results indicate that KNN matting, which adopts the nonlocal principle, performs in general better than closed form matting and related techniques [22] where the local color-line model was adopted.

Future work includes investigating the relationship between the nonlocal principle and the color-line model applied *nonlocally* in general alpha matting of multiple layers from images and video matting.

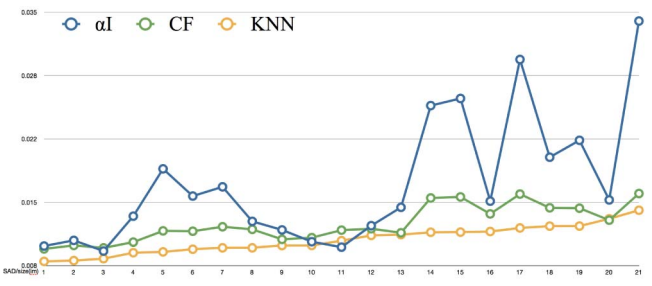


Fig. 19. SAD on the difference images. Our layer extraction based on KNN matting has the lowest errors in almost all of the 21 test cases.

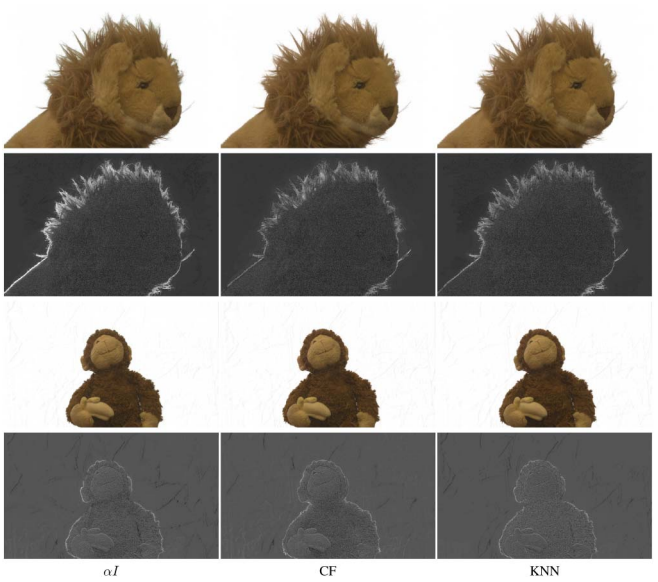


Fig. 20. Foreground images computed and corresponding difference maps.

Acknowledgments

This research was supported by the Research Grant Council of Hong Kong Special Administrative Region under grant number 619112.

REFERENCES

[1] R. Barrett, M. Berry, T.F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H.V. der Vorst, *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods,* second ed. SIAM, 1994.

[2] A. Buades, B. Coll, and J.-M. Morel, "Nonlocal Image and Movie Denoising," *Int'l J. Computer Vision,* vol. 76, no. 2, pp. 123-139, 2008.

[3] Q. Chen, D. Li, and C.-K. Tang, "KNN Matting," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 869-876, 2012.

[4] Y. Chuang, B. Curless, D.H. Salesin, and R. Szeliski, "A Bayesian Approach to Digital Matting," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* vol. II, pp. 264-271, 2001.

[5] F.H. Cole, "Automatic BRDF Factorization," bachelor honors thesis, Harvard College, 2002.

[6] E.S.L. Gastal and M.M. Oliveira, "Shared Sampling for Real-Time Alpha Matting," *Computer Graphics Forum,* vol. 29, no. 2, pp. 575-584, May 2010.

[7] D.B. Goldman, B. Curless, A. Hertzmann, and S.M. Seitz, "Shape and Spatially-Varying BRDFS from Photometric Stereo," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 32, no. 6, pp. 1060-1071, June 2010.

[8] Y. Guan, W. Chen, X. Liang, Z. Ding, and Q. Peng, "Easy Matting —A Stroke Based Approach for Continuous Image Matting," *Computer Graphics Forum,* vol. 25, no. 3, pp. 567-576, Sept. 2006.

[9] K. He, C. Rhemann, C. Rother, X. Tang, and J. Sun, "A Global Sampling Method for Alpha Matting," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 2049-2056, 2011.

[10] K. He, J. Sun, and X. Tang, "Fast Matting Using Large Kernel Matting Laplacian Matrices," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 2165-2172, 2010.

[11] J. Lawrence, A. Ben-artzi, C. Decoro, W. Matusik, H. Pfister, R. Ramamoorthi, and S. Rusinkiewicz, "Inverse Shade Trees for Non-Parametric Material Representation and Editing," *ACM Trans. Graphics,* pp. 735-745, 2006.

[12] P. Lee and Y. Wu, "Nonlocal Matting," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 2193-2200, 2011.

[13] D. Lepage and J. Lawrence, "Material Matting," *ACM Trans. Graphics,* vol. 30, no. 6, article 144, 2011.

[14] A. Levin, D. Lischinski, and Y. Weiss, "A Closed-Form Solution to Natural Image Matting," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 30, no. 2, pp. 228-242, Feb. 2008.

[15] A. Levin, A. Rav-Acha, and D. Lischinski, "Spectral Matting," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 30, no. 10, pp. 1699-1712, Oct. 2008.

[16] H. Lin, Y.-W. Tai, and M.S. Brown, "Motion Regularization for Matting Motion Blurred Objects," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 33, no. 11, pp. 2329-2336, Nov. 2011.

[17] M.D. McCool, J. Ang, and A. Ahmad, "Homomorphic Factorization of BRDFS for High-Performance Rendering," *Proc. ACM Siggraph,* pp. 171-178, 2001.

[18] C. Rhemann, C. Rother, and M. Gelautz, "Improving Color Modeling for Alpha Matting," *Proc. British Machine Vision Conf.,* pp. 1-10, 2008.

[19] C. Rhemann, C. Rother, P. Kohli, and M. Gelautz, "A Spatially Varying PSF-Based Prior for Alpha Matting," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 2149-2156, 2010.

[20] C. Rhemann, C. Rother, J. Wang, M. Gelautz, P. Kohli, and P. Rott, "A Perceptually Motivated Online Benchmark for Image Matting," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 1826-1833, 2009.

[21] E. Shahrian and D. Rajan, "Weighted Color and Texture Sample Selection for Image Matting," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 718-725, 2012.

[22] D. Singaraju and R. Vidal, "Estimation of Alpha Mattes for Multiple Image Layers," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 33, no. 7, pp. 1295-1309, July 2011.

[23] A.R. Smith and J.F. Blinn, "Blue Screen Matting," *Proc. ACM Siggraph,* pp. 259-268, 1996.

[24] J. Sun, J. Jia, C.-K. Tang, and H.-Y. Shum, "Poisson Matting," *ACM Trans. Graphics,* vol. 23, pp. 315-321, Aug. 2004.

[25] A. Vedaldi and B. Fulkerson, "VLFeat: An Open and Portable Library of Computer Vision Algorithms," http://www.vlfeat.org/, 2008.

[26] J. Wang and M.F. Cohen, "An Iterative Optimization Approach for Unified Image Segmentation and Matting," *Proc. 10th IEEE Int'l Conf. Computer Vision,* pp. 936-943, 2005.

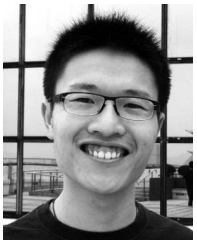
[27] J. Wang and M.F. Cohen, "Optimized Color Sampling for Robust Matting," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2007.

[28] J. Wang and M.F. Cohen, *Image and Video Matting.* Now Publishers Inc., 2008.

[29] Z. Zhang, Q. Zhu, and Y. Xie, "Learning Based Alpha Matting Using Support Vector Regression," *Proc. IEEE Int'l Conf. Image Processing,* pp. 2109-2112, 2012.

[30] Y. Zheng and C. Kambhamettu, "Learning Based Digital Matting," *Proc. IEEE Int'l Conf. Computer Vision,* pp. 889-896, 2009.

Qifeng Chen received the BSc degree in computer science and mathematics from the Hong Kong University of Science and Technology (HKUST) in 2012. He is currently working toward the PhD degree in computer science at Stanford University, California. His research areas include computer vision, computer graphics, computational photography, and image processing. In 2012, he was awarded the academic achievement medal and named the champion of the Mr. Armin and Mrs. Lillian Kitchell Undergraduate Research Competition from HKUST. In 2011, he won a gold medal (second place) at the ACM International Collegiate Programming Contest World Finals. He is a student member of the IEEE and the IEEE Computer Society.



Dingzeyu Li received the BEng degree in computer engineering from the Hong Knog University ofScience and Technology (HKUST) in 2013. Currently he is a PhD student in computer science at Columbia University, New York. He was an exchange student at ETH Zurich, Switzerland. He received the Computer Science & Engineering Department Scholarship and the Lee Hysan Foundation Exchange Scholarship. His research interests include computer graphics and computer vision. He is a student member of the IEEE and the IEEE Computer Society.

Chi-Keung Tang received the MSc and PhD degrees in computer science from the University of Southern California, Los Angeles, in 1999 and 2000, respectively. Since 2000, he has been with the Department of Computer Science, Hong Kong University of Science and Technology, where he is currently a full professor. He is an adjunct researcher in the Visual Computing Group of Microsoft Research Asia. His research areas include computer vision, computer graphics, and human-computer interaction. He is an associate editor of the *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, and is on the editorial board of the *International Journal of Computer Vision (IJCV)*. He served as an area chair for ACCV '06, ICCV '07, ICCV '09, ICCV '11, and as a technical papers committee member for the inaugural SIGGRAPH Asia 2008, SIGGRAPH 2011, SIGGRAPH Asia 2011, SIGGRAPH 2012, and SIGGRAPH Asia 2012. He is a senior member of the IEEE and a member of the IEEE Computer Society.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/publications/dlib.