Estimation of Alpha Mattes for Multiple Image Layers

Dheeraj Singaraju, *Member*, *IEEE*, and René Vidal, *Member*, *IEEE*

Abstract—Image matting deals with the estimation of the *alpha matte* at each pixel, i.e., the contribution of the foreground and background objects to the composition of the image at that pixel. Existing methods for image matting are typically limited to estimating the alpha mattes for two image layers only. However, in several applications one is interested in editing images with multiple objects. In this work, we consider the problem of estimating the alpha mattes of multiple ($n \geq 2$) image layers. We show that this problem can be decomposed into n *simpler* subproblems of alpha matte estimation for two image layers. Moreover, we show that, by construction, the estimated alpha mattes at each pixel are constrained to sum up to 1 across the multiple image layers. A key feature of our framework is that the alpha mattes can be estimated in closed form. We further show that, due to the nature of spatial regularization used in the estimation, the final estimated alpha mattes are not constrained to take values in $[0, 1]$. Hence, we study the optimization problem of estimating the alpha mattes for multiple image layers subject to the fact that the alpha mattes are nonnegative and sum up to 1 at each pixel. We present experiments to show that our proposed method can be used to extract mattes of multiple image layers.

Index Terms—Image matting, alpha matte, multiple layers, matting Laplacian, superposition principle.

1 Introduction

Existing literature in image matting assumes that a given composite image can be treated as the composition of two image layers: foreground and background. In particular, the intensity I_i of the ith pixel in a composite image is written as the convex combination of a foreground intensity F_i and a background intensity B_i:

$$I_i = \alpha_i F_i + (1 - \alpha_i) B_i, \tag{1}$$

where $\alpha_i \in [0, 1]$ is referred to as the pixel's partial opacity value or *alpha matte*. For a color image, we have $I_i \in \mathbb{R}^3_+$. Thus, (1) gives us three equations in seven unknowns: $\alpha_i \in [0, 1]$, $F_i \in \mathbb{R}^3_+$ and $B_i \in \mathbb{R}^3_+$ at each pixel. Therefore, the matting problem is ill posed since the number of unknowns is more than the number of independent equations. To this effect, matting algorithms require some user interaction that specifies the object of interest and the background, thereby enforcing some constraints in the image and making the problem well posed. As illustrated in Fig. 1, such interaction is typically provided in the form of a *trimap* by marking different regions in the image as 1) foreground; $\alpha = 1$ (shown in white), 2) background; $\alpha = 0$ (shown in black), and 3) unknown; $\alpha \in [0, 1]$ (shown in gray). The goal of matting algorithms is to estimate the alpha mattes of the pixels in the unknown region. Given the alpha mattes, one can estimate the foreground and background layers of the image and subsequently use them for various image editing tasks such as background modification. Fig. 1 gives an example of image matting where the user is interested in extracting the mattes of a dandelion in an image.

Incipient methods in [1], [2] use the trimap to learn color models for the object and the background and subsequently estimate the mattes in the unknown region as per (1). In this case, the pixels are processed independently and the estimated mattes are not spatially regularized. Methods in [3], [4] solve such issues by using local propagation techniques to estimate the mattes. A common criticism of these methods is that they fail on images with complex intensity variations due to the naive appearance models that are used.

Motivated by these drawbacks, subsequent research in image matting witnessed a number of algorithms in [5], [6], [7], [8], [9], [10] which employed tools traditionally used for image segmentation. In these algorithms, the user typically provides a *sparse trimap* by marking only a few pixels in the image. This trimap is then used to find a binary segmentation of the image. Since one expects most fractional alpha mattes to occur at the objects' boundaries, the segmentation boundary is slightly dilated to generate a *tighter* trimap that contains fewer unmarked pixels in comparison to the original trimap. The mattes are then estimated using this tight trimap and can subsequently be refined by alternating between reestimation of the trimap and the alpha mattes. The performance of such strategies is governed by the techniques used for alpha matte estimation in each iteration. Most algorithms use techniques such as [11], [12] to estimate the mattes with the tight trimap.

Recent work in image matting has seen a surge of research toward developing algorithms that exploit various features specific to the matting problem [11], [12], [13], [14], [15], [16]. It was shown in [11] that if one assumes that the intensities of the foreground and background layers vary linearly in small

• *The authors are with the Center for Imaging Science, Department of Biomedical Engineering, The Johns Hopkins University, Clark Hall, 3400 N. Charles St., Baltimore, MD 21218.*
E-mail: {dheeraj, rvidal}@cis.jhu.edu.

Manuscript received 29 Sept. 2009; revised 19 Apr. 2010; accepted 22 Oct. 2010; published online 24 Nov. 2010.
Recommended for acceptance by A. Fitzgibbon.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-2009-09-0652.
Digital Object Identifier no. 10.1109/TPAMI.2010.206.

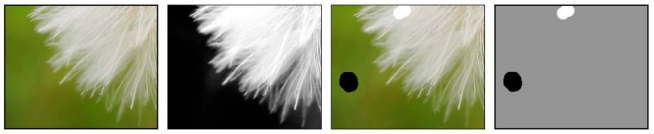


Fig. 1. In this typical example of image matting, the goal is to recover the alpha mattes (b) of the dandelion in the image (a). The user labels some representative pixels for the object (in white) and the background (black). This interaction can be represented by a *trimap* (shown in (d)) where the three regions indicate white ($\alpha = 1$), black ($\alpha = 0$), and gray ($\alpha = \text{unknown}$). (a) Image of a dandelion. (b) Desired alpha mattes. (c) Pixels labeled by the user. (d) Generated trimap.

image patches, then the alpha mattes could be estimated in closed form. It was later demonstrated that the performance of local propagation-based methods in [11] could be improved by additionally learning global color models [12], [16] or by building more compact local color models for the image [17]. Recent work has also focused on enforcing sparsity of the alpha mattes [14], [15]. For a more detailed review, we refer the reader to Wang and Cohen [18].

All of the methods discussed so far assume that the user wants to edit either the background or a single object in the image. Hence, they estimate alpha mattes for two layers only. However, a user may be interested in editing multiple regions in an image as shown in Fig. 2. In this example, the user might be interested in separately editing three different layers, namely, the two dolls and the background. In order to allow for such editing, we need to be able to reconstruct each of the layers of interest. An important step in this reconstruction is the estimation of partial opacity values $\{\alpha_i^k\}_{k=1}^n$ at each pixel i such that its intensity I_i can be expressed as a convex linear combination of the intensities $\{F_i^k\}_{k=1}^n$ of n image layers:

$$I_i = \sum_{k=1}^{n} \alpha_i^k F_i^k \text{ where } \forall i, k, \alpha_i^k \geq 0 \text{ and } \forall i, \sum_{k=1}^{n} \alpha_i^k = 1. \quad (2)$$

Note that in order to solve the n-layer matting problem ($n \geq 2$), one may use existing algorithms in [11] to estimate the mattes for each layer by treating it as foreground and treating the remaining layers as background. However, this strategy raises two concerns. The estimated mattes might not satisfy the constraints discussed in (2), i.e., 1) the mattes take values between 0 and 1, and 2) the mattes sum up to 1 at each pixel. Moreover, most methods use the Matting Laplacian proposed in [11] for spatial regularization of the alpha mattes. This Laplacian matrix was derived under the assumption that the image has two layers only. However, as shown in the image crop presented in Fig. 2b, there might be regions that contain $n \geq 2$ layers. We notice in this example that there are regions which contain three different layers, i.e., the background, the pink hair, as well as the green hair. It is unclear if the Matting Laplacian may be applicable to such regions. Hence, in this work, we build upon our previous work in [19] and present a framework for estimating the mattes for multiple image layers such that the constraints in (2) are satisfied.

We note that the most relevant work to our proposed framework is the Spectral Matting algorithm [14]. This algorithm uses eigenvectors of the Matting Laplacian to compute multiple matting components that are then combined as per the user's interaction to give the alpha mattes. We show that this algorithm considers a restricted subset of all possible linear combinations of the eigenvectors of the Matting Laplacian, while our framework admits a larger set of solutions.

Fig. 2. An example where the user is interested in editing several different regions of an image (shown in (a)) of two trolls against the background. In this case, the user may be separately editing three different layers in the image, i.e., the two dolls and the background. In (b), we present an image crop (which has been highlighted by the yellow rectangle in (a)), with regions where all three layers overlap. Notice that there are regions in the center of this crop where we can see the pink hair and the green hair, as well as the background. (a) Given image. (b) Image crop.

Although we have motivated the need for a framework for alpha matte estimation for multiple layers, our presented analysis can also be applied to problems other than alpha matting. In [20], Bleyer et al. consider the alpha mattes of $n \geq 2$ layers while solving the stereo problem. In [21], Hsu et al. solve the problem of light mixture estimation for the case of multiple light sources by using models similar to those derived for alpha matting. One may then generalize [21], which deals with two light sources only, to the case of multiple light sources. Such an analysis may also be used for studying motion blur since in [22], Shan et al. model the blurred image as a composite of shifted versions of the unblurred image.

Paper contributions: We analyze the conditions on local image patches, under which the alpha mattes of multiple image layers can be estimated in closed form. Levin et al. [11] discussed the case of two layers only, while Levin et al. [14] later explored a very small family of appearance models for the case of multiple layers. In this work, we describe more general conditions on the appearance models for multiple layers and show that under certain conditions, there exists an affine function for each image patch that relates a pixel's RGB intensities and its alpha mattes. Hence, we show that the Matting Laplacian can be used to regularize the mattes for multiple layers.

We then consider the problem of estimating the mattes for multiple image layers, subject to the constraint that the estimated mattes at each pixel sum up to 1 across all of the layers. Since this constraint couples the mattes for the different layers, the proposed optimization problem might be computationally expensive. However, we show that the problem of estimating the mattes of $n \geq 2$ image layers can be decoupled into n simpler subproblems. Each subproblem estimates the mattes for a layer k by solving a two-layer matting problem where the layer k is treated as foreground and the remaining layers are treated as background. Specifically, we first present an equivalent electrical network construction that unifies the optimization methods adopted by [11] and [12] for estimating the mattes of $n = 2$ layers. We

generalize this formulation to estimate the mattes for multiple layers and use the superposition principle from electrical network theory to show that the constraint that the mattes sum up to 1 across the different image layers is naturally satisfied and need not be explicitly enforced.

We further show that the alpha mattes estimated by the above framework are not naturally constrained to take values in $[0, 1]$. If the matte at a pixel does not lie in $[0, 1]$, its interpretation as a partial opacity breaks down. Hence, one may postprocess the estimated mattes by clipping their values to lie in $[0, 1]$ to resolve this issue. Alternatively, in this work we consider the optimization problem of estimating the mattes subject to the constraints that the mattes at each pixel: 1) take values in $[0, 1]$, and 2) sum up to 1 across the different image layers. In this case, we show that the problem of estimating the mattes of $n \geq 2$ image layers cannot be decoupled into $n \geq 2$ subproblems of estimating the mattes for two image layers since we need to explicitly enforce the constraint that the mattes sum up to 1 across the different image layers. This constraint is satisfied naturally *only* when $n = 2$ and not otherwise.

Finally, we present a qualitative as well as quantitative evaluation of our proposed framework. We show that the mattes estimated by our framework for $n = 2$ layers are similar to those given by [11]. We present a qualitative analysis of estimating the alpha mattes for $n > 2$ image layers using our proposed framework as well as the Spectral Matting algorithm [14]. We show that we are able to extract the alpha mattes for multiple layers with low levels of user interaction.

2 Notation

In this paper, we pose the problem of estimating the mattes as an optimization problem defined on a weighted graph that represents a given composite image. For this purpose, we now introduce notation that will be used in the rest of this paper.

A weighted graph $\mathcal{G}$ consists of a pair $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with nodes $i \in \mathcal{V}$ and undirected edges $\boldsymbol{e}_{ij} \in \mathcal{E}$. The nodes on the graph typically correspond to pixels in the image. An edge that spans two vertices i and j is denoted by $\boldsymbol{e}_{ij}$. The neighborhood of a node i is given by all the nodes j that share an edge with i and is denoted by $\mathcal{N}_i$. Each edge is assigned a value $\boldsymbol{w}_{ij}$ that is referred to as its weight. Since the edges are undirected, we have $\boldsymbol{w}_{ij} = \boldsymbol{w}_{ji}$. These edge weights are used to define the degree d_i of a node i as $d_i = \sum_{j \in \mathcal{N}_i} \boldsymbol{w}_{ij}$. One can use these terms to construct a *Laplacian matrix* L for the graph as $L = D - W$, where $D = \mathrm{diag}(d_1, d_2, \ldots, d_{|\mathcal{V}|})$ and W is a $|\mathcal{V}| \times |\mathcal{V}|$ matrix whose (i, j) entry is given by the edge weight $\boldsymbol{w}_{ij}$. By construction, the Laplacian matrix has the constant vector of 1s in its null space, i.e., $L\mathbf{1} = \mathbf{0}$.

Recall that the alpha matting problem is underconstrained and we require the user to mark representative nodes for the different image layers. These marked nodes enforce constraints on the mattes and are subsequently used to predict the mattes of the remaining unmarked nodes. The set $\mathcal{M} \subset \mathcal{V}$ contains the locations of the nodes marked by the user and the set $\mathcal{U} \subset \mathcal{V}$ contains the locations of the unmarked nodes. By construction, $\mathcal{M} \cap \mathcal{U} = \emptyset$ and $\mathcal{M} \cup \mathcal{U} = \mathcal{V}$. We further split the set $\mathcal{M}$ that contains the locations of all of the marked nodes into the sets $\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_n$, where $\mathcal{M}_k$ contains the marked nodes representative of the kth layer. By construction, we have $\cup_{k=1}^{n} \mathcal{M}_k = \mathcal{M}$ and $\forall 1 \leq k_1 < k_2 \leq n$, $\mathcal{M}_{k_1} \cap \mathcal{M}_{k_2} = \emptyset$.

We denote the intensities of the kth image layer at pixel $i \in \mathcal{V}$ as $\mathcal{F}_i^k$. The matte at the pixel i with respect to the kth layer is denoted as α_i^k. The mattes of all pixels in the image for a particular layer can be stacked into a vector as $\boldsymbol{\alpha}^k = [\boldsymbol{\alpha}_U^{k\top} \quad \boldsymbol{\alpha}_M^{k\top}]^\top \in [0, 1]^{|\mathcal{V}|}$, where $\boldsymbol{\alpha}_U^k$ and $\boldsymbol{\alpha}_M^k$ correspond to the mattes of the unmarked and marked pixels, respectively.

3 Local Models for Alpha Mattes for Multiple Layers

Omer and Werman [23] empirically showed that the distribution of colors in natural images is locally linear in RGB space. Inspired by this work, Levin et al. [11] assume that for a small patch $\mathcal{W}_i$ around a pixel i in the query composite image, the intensities of the corresponding foreground and background layers can be treated as lying on lines in RGB space. Under this assumption, they showed that there exists an affine function $v_i = (a_i^R, a_i^G, a_i^B, b_i)$ characteristic to the patch $\mathcal{W}_i$ such that the alpha matte α_j of each pixel $j \in \mathcal{W}_i$ can be written as

$$\alpha_j = a_i^R I_j^R + a_i^G I_j^G + a_i^B I_j^B + b_i, \tag{3}$$

where I_j^R, I_j^G, and I_j^B refer to the RGB values of pixel j. This relationship is essential for constructing the Matting Laplacian that helps provide a closed form solution to alpha matting.

We now show that under certain conditions, the alpha mattes of each pixel in the patch can be written as affine functions of the RGB values, even in the case of multiple image layers. Specifically, Theorem 1 states the conditions under which the alpha mattes of the pixels j in a window $\mathcal{W}_i$ around a pixel i can be expressed in terms of the affine functions.

Theorem 1. *Denote the number of image layers present in a window $\mathcal{W}_i$ around a pixel i as n_i. Define $\mathcal{F}_i^k \in \mathbb{R}^{4 \times |\mathcal{W}_i|}$ as the matrix whose columns contain the intensities of the kth image layer as*

$$\mathcal{F}_i^k = \begin{bmatrix} \cdots & F_j^k & \cdots \\ \cdots & 1 & \cdots \end{bmatrix},$$

where $j \in \mathcal{W}_i$. Similarly, define $\mathcal{I}_i \in \mathbb{R}^{4 \times |\mathcal{W}_i|}$ as the matrix whose columns contain the composite image intensities as

$$\mathcal{I}_i = \begin{bmatrix} \cdots & I_j & \cdots \\ \cdots & 1 & \cdots \end{bmatrix},$$

where $j \in \mathcal{W}_i$. Let the dimensions of the kth image layer and the composite image patch in $\mathcal{W}_i$ be defined as $d_i^k = \mathrm{rank}(\mathcal{F}_i^k)$ and $d_i^c = \mathrm{rank}(\mathcal{I}_i)$, respectively. If $\sum_{k=1}^{n_i} d_i^k = d_i^c \leq 4$, then the alpha mattes of each pixel in the window can be written as affine functions of its RGB intensities, as $\forall k = 1, \ldots, n_i$, $\forall j \in \mathcal{W}_i : \alpha_j^k = a_{i,k}^R I_j^R + a_{i,k}^G I_j^G + a_{i,k}^B I_j^B + b_{i,k}$, for some $(a_{i,k}^R, a_{i,k}^G, a_{i,k}^B, b_{i,k}) \in \mathbb{R}^4$.

Proof. We will prove the statement only for $d_i^c = 4$. The proof for the remaining cases can be sketched in a similar manner. Note first that the color line model used by [11] for the case of two image layers (i.e., $n_i = 2$ and $(d_i^1, d_i^2) = (2, 2)$) is a particular case of the hypothesis, for which we already know from [11] that the mattes can be written as affine functions of the RGB intensities. We now enumerate the additional cases in which such relationships can be obtained.

1. *A color plane and a color point*: In this case, the image patch $\mathcal{W}_i$ is composed of two layers. The intensities of one image layer are constant and hence constitute a *point* in RGB space. The intensities of the other image layer lie on a *color plane* in RGB space. Without loss of generality, assume that the intensities of the second layer lie on a plane. In this case, we have $n_i = 2$ and $(d_i^1, d_i^2) = (1, 3)$. The intensities of these layers can be parameterized as $\forall j \in \mathcal{W}_i$, $F_j^1 = C_1$, and $F_j^2 = \beta_{j1}C_2 + \beta_{j2}C_3 + (1 - \beta_{j1} - \beta_{j2})C_4$ for some $\beta_{j1}, \beta_{j2} \in \mathbb{R}$. Hence, the composite image intensities can be expressed as

$$\begin{aligned}&\forall j \in \mathcal{W}_i : I_j \\ &= \alpha_j C_1 + (1 - \alpha_j)(\beta_{j1}C_2 + \beta_{j2}C_3 \\ &+ (1 - \beta_{j1} - \beta_{j2})C_4) \\ &= \underbrace{[C_1 - C_4 \quad C_2 - C_4 \quad C_3 - C_4]}_{H_2} \begin{bmatrix} \alpha_j \\ (1 - \alpha_j)\beta_{j1} \\ (1 - \alpha_j)\beta_{j2} \end{bmatrix} \\ &+ C_4. \end{aligned} \tag{4}$$

Note that H_2 is invertible because, otherwise, the color point corresponding to the first layer lies on the color plane corresponding to the second layer. This is a degenerate case because one cannot estimate the alpha mattes uniquely in such a situation. We ignore such degenerate cases. Now, since H_2 is invertible, we can rewrite (4) as (5) and conclude from the first row of the matrix equation, that the alpha matte of each pixel can be expressed as an affine function of the RGB intensities

$$\forall j \in \mathcal{W}_i : \begin{bmatrix} \alpha_j \\ (1 - \alpha_j)\beta_{j1} \\ (1 - \alpha_j)\beta_{j2} \end{bmatrix} = H_2^{-1} I_j - H_2^{-1} C_4. \tag{5}$$

2. *Two color points and a single color line*: In this case, the image patch is composed of three layers, where the intensities of two image layers are constant and the intensities of the third image layer lie in a color line in RGB space. Without loss of generality, assume that the intensities of the third layer lie on a line, i.e., $n_i = 3$ and $(d_i^1, d_i^2, d_i^3) = (1, 1, 2)$. The layers' intensities can be parameterized as $\forall j \in \mathcal{W}_i$, $F_j^1 = C_1$, $F_j^2 = C_2$, and $F_j^3 = \beta_j C_3 + (1 - \beta_j)C_4$, where $\beta_j \in \mathbb{R}$. The composite intensities can then be expressed as

$$\begin{aligned}&\forall j \in \mathcal{W}_i : I_j \\ &= \alpha_j^1 C_1 + \alpha_j^2 C_2 + (1 - \alpha_j^1 - \alpha_j^2)(\beta_j C_3 + (1 - \beta_j)C_4) \\ &= \underbrace{[C_1 - C_4 \quad C_2 - C_4 \quad C_3 - C_4]}_{H_3} \begin{bmatrix} \alpha_j^1 \\ \alpha_j^2 \\ \beta_j(1 - \alpha_j^1 - \alpha_j^2) \end{bmatrix} \\ &+ C_4. \end{aligned} \tag{6}$$

We note that H_3 is invertible for nondegenerate configurations of the image layers. Hence, we can rewrite (6) as (7) and conclude from the top two rows of the matrix equation that the alpha mattes of each pixel can be expressed as affine functions of the RGB intensities:

$$\forall j \in \mathcal{W}_i : \begin{bmatrix} \alpha_j^1 \\ \alpha_j^2 \\ \beta_j(1 - \alpha_j^1 - \alpha_j^2) \end{bmatrix} = H_3^{-1} I_j - H_3^{-1} C_4. \tag{7}$$

3. *Four color points*: In this final case, the image patch is composed of four layers, each of which has constant intensities. Notice that $n_i = 4$ and $(d_i^1, d_i^2, d_i^3, d_i^4) = (1, 1, 1, 1)$. The intensities of the kth layer can hence be parameterized as $\forall j \in \mathcal{W}_i$, $F_j^k = C_k$, $k = 1, 2, 3$, or 4. The composite image intensities are then given as

$$\begin{aligned}&\forall j \in \mathcal{W}_i : I_j \\ &= \alpha_j^1 C_1 + \alpha_j^2 C_2 + \alpha_j^3 C_3 + \left(1 - \alpha_j^1 - \alpha_j^2 - \alpha_j^3\right) C_4 \\ &= \underbrace{[C_1 - C_4 \quad C_2 - C_4 \quad C_3 - C_4]}_{H_4} \begin{bmatrix} \alpha_j^1 \\ \alpha_j^2 \\ \alpha_j^3 \end{bmatrix} + C_4. \end{aligned} \tag{8}$$

As before, we can rewrite (8) as (9) and conclude that the alpha mattes of each pixel can be expressed as affine functions of the RGB intensities:

$$\forall j \in \mathcal{W}_i : \begin{bmatrix} \alpha_j^1 \\ \alpha_j^2 \\ \alpha_j^3 \end{bmatrix} = H_4^{-1} I_j - H_4^{-1} C_4. \tag{9}$$

□

Remark 1. Since the intensity of each composite pixel is the convex combination of the intensities of the pixels in the constituent image layers, in general, we have $\sum_{k=1}^{n_i} d_i^k \geq d_i^c$. Notice that if $\sum_{k=1}^{n_i} d_i^k > d_i^c$, we cannot uniquely recover the mattes because the systems of equations analogous to (4), (6), and (8), would have fewer equations than the number of unknowns. Hence, Theorem 1 assumes the equality $\sum_{k=1}^{n_i} d_i^k = d_i^c$ to recover the mattes uniquely, without the knowledge of any additional constraints on the alpha mattes. Since the dimensions d_i^k are nonnegative, the constraint $\sum_{k=1}^{n_i} d_i^k = d_i^c$ implies that $n_i \leq d_i^c$, which can be at most 4. These are not strong assumptions since the scenario $4 \geq \sum_{k=1}^{n_i} d_i^k > d_i^c$ typically corresponds to *degenerate* mattes that form a zero measure set. For example, if

the composite patch consists of two layers ($n_i = 2$), where each layer has a constant color ($d_i^1 = d_i^2 = 1$) and each pixel in the patch has the same matte, then the composite patch also has constant color ($d_i^c = 1$).

Theorem 1 gives conditions under which the alpha mattes for multiple layers can be recovered in closed form. These conditions are more general than those in [14]. Specifically, the only cases considered in [14] are $n_i = 1$ and $(n_i, d_i^1, d_i^2, d_i^3) = (3, 1, 1, 1)$, plus the color line model in [11]. Theorem 1 shows that one can estimate the alpha mattes in closed form for a much larger family of appearance models which includes the models in [11], [14], [17] as particular cases.

Notice also that we proved Theorem 1 under the assumption that the space spanned by the RGB colors of the patch W_i is equal to 4. The dimension of this patch can be less than 4 under some conditions [17]. For example, the dimension of a composite image patch can be 3 if the image layers are such that $n_i = 3$ and $(d_i^1, d_i^2, d_i^3) = (1, 1, 1)$. However, this can be treated as a degenerate case of $n_i = 3$ and $(d_i^1, d_i^2, d_i^3) = (1, 1, 2)$. Hence, for the rest of this paper, we assume that $d_i^c = 4$ for each image patch, without any loss of generality.

4 Spatial Regularization of the Alpha Mattes

We now show that under the conditions given by Theorem 1, we can aggregate the information from the local patches in the image to construct a spatial regularizer for the mattes in the entire image. We show that this regularizer is the same as the Matting Laplacian proposed in [11].

Recall from Theorem 1 that under certain conditions, there exist affine functions $v_i = (a_i^R, a_i^G, a_i^B, b_i)$ characteristic to the patch $\mathcal{W}_i$ around each pixel $i \in \mathcal{V}$, such that the alpha matte α_j of each pixel $j \in \mathcal{W}_i$ can be written as

$$\alpha_j = a_i^R I_j^R + a_i^G I_j^G + a_i^B I_j^B + b_i, \tag{10}$$

where I_j^R, I_j^G, and I_j^B refer to the RGB values of pixel j. The problem of estimating the mattes $\boldsymbol{\alpha}$ in the image can consequently be posed as one of finding the minimizer of

$$J(\boldsymbol{\alpha}, v) = \sum_{i \in \mathcal{V}} \left[\sum_{j \in \mathcal{W}_i} \left(\alpha_j - a_i^R I_j^R - a_i^G I_j^G - a_i^B I_j^B - b_i \right)^2 \right], \tag{11}$$

where $v = \{v_i\}_{i \in \mathcal{V}}$. Essentially, this corresponds to minimizing the residual of the affine model v_i defined in (3) for every small patch $\mathcal{W}_i$. Along the lines of Levin et al. [11], we propose using a modification of the cost function $J(\boldsymbol{\alpha}, v)$ by introducing an additional regularization term as

$$J_\epsilon(\boldsymbol{\alpha}, v) = J(\boldsymbol{\alpha}, v) + \epsilon \sum_{i \in \mathcal{V}} \left(a_i^{R2} + a_i^{G2} + a_i^{B2} \right). \tag{12}$$

The regularization term introduces a bias toward the constant valued affine function $(a_i^R, a_i^G, a_i^B, b_i) = (0, 0, 0, c)$, $c \in [0, 1]$ for each patch $\mathcal{W}_i$, or in other words, introduces a bias toward locally constant alpha mattes. The motivation for this is twofold. First, the user provided trimap labels very few pixels and not all of the pixels have integer valued mattes, i.e., $\alpha = 0$ or 1. Hence, for many pixels in the image, an α of 0 or 1 is desired, independent of the appearance model. Second, real images often have textured patches that do not satisfy the color line model, but nonetheless may have uniform mattes across the patch. The mattes of such patches can be explained by an affine function of the form $v = (0, 0, 0, c)$, $c \in [0, 1]$. This function allows for certain complex cases beyond the color models discussed in Theorem 1.

Now, note that the constructed cost function $J_\epsilon(\boldsymbol{\alpha}, v)$ depends on two unknown quantities: the alpha mattes $\boldsymbol{\alpha}$ and the affine functions v. However, this can be reduced to a cost function that depends solely on the alpha mattes. For the sake of simplicity, let us define matrices $G_i \in \mathbb{R}^{(|\mathcal{W}_i|+3)\times 4}$ and $\bar{\alpha}_i \in \mathbb{R}^{|\mathcal{W}_i|+3}$. The first $|\mathcal{W}_i|$ rows of G_i are given by $[I_j^R\ I_j^G\ I_j^B\ 1]$, $j \in \mathcal{W}_i$, and the last three rows are given by $[\sqrt{(\epsilon)}\boldsymbol{I}_3\ \mathbf{0}]$, where we use $\boldsymbol{I}_n$ to denote an identity matrix of size $n \times n$. The first $|\mathcal{W}_i|$ entries of $\bar{\alpha}_i$ are given by $\alpha_j, j \in \mathcal{W}_i$, and the last three entries are equal to 0. Given this notation, $J_\epsilon(\boldsymbol{\alpha}, v)$ can be rewritten as

$$J_\epsilon(\boldsymbol{\alpha}, v) = \sum_{i \in \mathcal{V}} \|G_i v_i - \bar{\alpha}_i\|^2. \tag{13}$$

We can estimate the affine function v_i for each patch $\mathcal{W}_i$ as

$$v_i = \arg\ \min_v \|G_i v - \bar{\alpha}_i\|^2 = (G_i^\top G_i)^{-1} G_i^\top \bar{\alpha}_i. \tag{14}$$

Therefore, using the expression for v_i from (14), we see that the cost function $J_\epsilon(\boldsymbol{\alpha}, v)$ can be reduced to a cost function dependent on the alpha mattes only as

$$J_\epsilon(\boldsymbol{\alpha}) = \sum_{i \in \mathcal{V}} \left[\bar{\alpha}_i^\top (\boldsymbol{I}_{|\mathcal{W}_i|+3} - G_i (G_i^\top G_i)^{-1} G_i^\top) \bar{\alpha}_i \right] = \boldsymbol{\alpha}^\top L \boldsymbol{\alpha}. \tag{15}$$

Here, L is a $|\mathcal{V}| \times |\mathcal{V}|$ Laplacian matrix that is referred to as the *Matting Laplacian*, whose entries capture the local statistics of the intensity variations in a small window around each pixel [11]. It was shown in [11] that L can be constructed from the edge weights $\boldsymbol{w}_{ij}$ defined as

$$\boldsymbol{w}_{ij} = \sum_{l | i,j \in \mathcal{W}_l} \frac{1}{|\mathcal{W}_l|} \left(1 + (I_i - \mu_l)^\top \left(\Sigma_l + \frac{\epsilon}{|\mathcal{W}_l|} \boldsymbol{I}_3 \right)^{-1} (I_j - \mu_l) \right), \tag{16}$$

where $\mu_i \in \mathbb{R}^3$ and $\Sigma_i \in \mathbb{R}^{3\times 3}$ denote the mean and the covariance of the intensities of the pixels in $\mathcal{W}_i$, respectively. Note that these weights can be positive or negative. In practice, each window is chosen to be of size 3×3 and accounts for local spatial regularization of the alpha mattes of the pixels in that window. Since these windows overlap, the Matting Laplacian acts as an effective global spatial regularizer that is obtained by aggregating local spatial regularization.

5 Closed-Form Solution to Alpha Matte Estimation for Multiple Layers

In this section, we present a general graph-theoretic optimization framework for estimating the mattes in closed

form. The edge weights of the constructed graph are obtained from the Matting Laplacian. We show that this framework generalizes existing algorithms such as Levin et al. [11] and Wang and Cohen [12]. We also describe an equivalent electrical network construction whose physics exhibit the same optimization scheme as in these methods. In this process, we provide a unifying framework for studying these algorithms and understanding the fine differences between them. Subsequently, we show how this framework can be easily extended for the closed form estimation of the mattes for multiple image layers.

5.1 Closed-Form Solution for Alpha Matte Estimation for Two Layers

We first address the simpler problem of estimating the alpha mattes for two layers. Recall that the user marks some pixels that are representative of the object and the background. The matting problem thus involves solving an optimization problem subject to the constraints enforced by these marked pixels. To this effect, we define a vector $\boldsymbol{m} \in \mathbb{R}^{|\mathcal{M}|}$ such that an entry of $\boldsymbol{m}$ is set to 1 or 0 depending on whether it corresponds to the object or background, respectively. Also, for some $\lambda > 0$, we define a matrix $\Lambda = \lambda \boldsymbol{I}_{|\mathcal{M}|}$. The mattes in the image are then estimated as

$$\begin{aligned}\boldsymbol{\alpha} &= \underset{\boldsymbol{\alpha}}{\operatorname{argmin}} \left[\sum_{\boldsymbol{e}_{ij} \in \mathcal{E}} \boldsymbol{w}_{ij} (\alpha_i - \alpha_j)^2 + \sum_{i \in \mathcal{M}} \lambda (\alpha_i - \boldsymbol{m}_i)^2 \right] \\ &= \underset{\boldsymbol{\alpha}}{\operatorname{argmin}} [\boldsymbol{\alpha}^\top L \boldsymbol{\alpha} + (\boldsymbol{\alpha}_M - \boldsymbol{m})^\top \Lambda (\boldsymbol{\alpha}_M - \boldsymbol{m})] \\ &= \underset{\boldsymbol{\alpha}}{\operatorname{argmin}} \begin{bmatrix} \boldsymbol{\alpha}_U^\top & \boldsymbol{\alpha}_M^\top & \boldsymbol{m}^\top \end{bmatrix} \begin{bmatrix} L_U & B^\top & 0 \\ B & L_M + \Lambda & -\Lambda \\ 0 & -\Lambda & \Lambda \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha}_U \\ \boldsymbol{\alpha}_M \\ \boldsymbol{m} \end{bmatrix}, \\ &\text{where } L = \begin{bmatrix} L_U & B^\top \\ B & L_M \end{bmatrix}. \end{aligned} \tag{17}$$

Since the expression in (12) is nonnegative, this implies that L is positive semidefinite. As a consequence, we note that L_U is positive semidefinite in general. However, if the user marks enough pixels in the image, then L_U is positive definite and invertible. Unless specified otherwise, we assume that L_U is positive definite from now on. This makes the cost function in (17) convex, and therefore, the optimization problem has a unique solution. In this case, the unique solution to (17) can be linearly estimated in closed form as

$$\begin{aligned} &\boldsymbol{\alpha}_M = [A + \Lambda]^{-1} \Lambda \boldsymbol{m} \qquad \text{and} \\ &\boldsymbol{\alpha}_U = -L_U^{-1} B^\top \boldsymbol{\alpha}_M = -L_U^{-1} B^\top [A + \Lambda]^{-1} \Lambda \boldsymbol{m}, \end{aligned} \tag{18}$$

where $A = L_M - B L_U^{-1} B^\top$. The algorithm of Levin et al. [11] employs the above optimization by choosing a large finite valued λ, while the algorithm of Wang and Cohen [12] works in the limiting case by choosing $\lambda = \infty$. We note that Wang and Cohen [12] employ a modification of the cost function in (17) since additional unary terms are generated by sampling potential foreground and background hypothesis from the trimap. However, this does not have any effect on the following analysis.

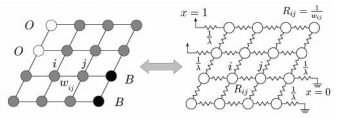


Fig. 3. Equivalent construction of combinatorial graphs and electrical networks for alpha matte estimation.

5.2 Alpha Matte Estimation via Energy Minimization in Electrical Networks

It is interesting to note that there exists an equivalent electrical network that solves the optimization problem in (17). One can construct a network as in Fig. 3, such that each node in the graph associated with the image is equivalent to a node on the network. The edge weights correspond to the conductance values of resistors connected between neighboring nodes, i.e., $\frac{1}{R_{ij}} = \boldsymbol{w}_{ij}$. Since the edge weights $\boldsymbol{w}_{ij}$ are not all positive, one can argue that this system might not be physically realizable. However, the network is constrained to dissipate positive energy due to the positive semidefiniteness of the Laplacian. Therefore, it suffices to treat the image as a real *resistive load* that dissipates energy. The marked nodes are connected to the network ground and unit voltage sources by resistive impedances of value $\frac{1}{\lambda}$. The nodes marked as background are connected to the ground and the nodes marked as object are connected to the unit voltage sources. Hence, we have $\boldsymbol{m}_i = 1V \ (i \in \mathcal{M}_1)$ at the voltage sources and $\boldsymbol{m}_i = 0V \ (i \in \mathcal{M}_2)$ at the ground, where all measurements are with respect to the network's ground.

From network theory, we know that the potentials $\boldsymbol{x}$ at the nodes in the network minimize the energy dissipated by the network. Therefore, they can be estimated as

$$\boldsymbol{x} = \underset{\boldsymbol{\alpha}}{\operatorname{argmin}} \left[\sum_{\boldsymbol{e}_{ij} \in \mathcal{E}} \frac{1}{R_{ij}} (x_i - x_j)^2 + \sum_{i \in \mathcal{M}} \lambda (x_i - m_i)^2 \right]. \tag{19}$$

This is the same expression as in (17). Therefore, if one were to construct the equivalent network and measure the potentials at the nodes, they would give us the required mattes.

We now show that the fraction (η) of work done by the electrical sources that are used to drive the load is maximized as $\lambda \to \infty$. The work done to drive the load is given as

$$\begin{aligned} E_{\text{load}} &= \boldsymbol{x}^\top L \boldsymbol{x} = \boldsymbol{x}_U^\top L_U \boldsymbol{x}_U + 2 \boldsymbol{x}_U^\top B^\top \boldsymbol{x}_M + \boldsymbol{x}_M^\top L_M \boldsymbol{x}_M \\ &= \boldsymbol{x}_M^\top \left[L_M - B L_U^{-1} B^\top \right] \boldsymbol{x}_M \\ &\quad \left(\text{using } \boldsymbol{x}_U = -L_U^{-1} B^\top \boldsymbol{x}_M \text{ from} (18) \right) \\ &= \boldsymbol{m}^\top \Lambda [A + \Lambda]^{-1} A [A + \Lambda]^{-1} \Lambda \boldsymbol{m} \\ &\quad (\text{using } \boldsymbol{x}_M = [A + \Lambda]^{-1} \Lambda \boldsymbol{m} \text{ from} (18)), \end{aligned} \tag{20}$$

where A was previously defined as $L_M - B L_U^{-1} B^\top$. Similarly, we can write down the work done by the electrical sources as the sum of the work done to drive the load and the work done by the voltage sources to power the marked nodes. Specifically, this work can be expressed as

$$\begin{aligned} E_{\text{total}} &= \boldsymbol{x}^\top L \boldsymbol{x} + (\boldsymbol{x}_M - \boldsymbol{m})^\top \Lambda (\boldsymbol{x}_M - \boldsymbol{m}) \\ &= \boldsymbol{x}^\top L \boldsymbol{x} + \boldsymbol{m}^\top ([A+\Lambda]^{-1}\Lambda - I)^\top \Lambda ([A+\Lambda]^{-1}\Lambda - I)\boldsymbol{m} \\ &\quad \text{(using (18))} \\ &= \boldsymbol{m}^\top \Lambda \left[\Lambda^{-1} - [A+\Lambda]^{-1}\right]\Lambda \boldsymbol{m} \\ &\quad \big(\text{substituting the expression for } \boldsymbol{x}^\top L \;\; \boldsymbol{x} \text{ from (18)}\big). \end{aligned} \tag{21}$$

Note that since L is positive semidefinite and the constructed graph is connected, the matrix A is also positive definite. Now, consider the singular value decomposition of the matrix $A = U\Sigma_A U^\top$, where $U = [u_1 \dots u_{|\mathcal{M}|}]$ and $\Sigma_A = \text{diag}(\sigma_1, \dots, \sigma_{|\mathcal{M}|})$, $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_{|\mathcal{M}|} > 0$. Given this expression, we can write $A + \Lambda = U[\Sigma_A + \Lambda]U^\top$. We can then evaluate η as

$$\begin{aligned} \eta = \frac{E_{\text{load}}}{E_{\text{total}}} &= \frac{\boldsymbol{m}^\top \Lambda [A+\Lambda]^{-1} A [A+\Lambda]^{-1} \Lambda \boldsymbol{m}}{\boldsymbol{m}^\top \Lambda \left[\Lambda^{-1} - [A+\Lambda]^{-1}\right]\Lambda \boldsymbol{m}} \\ &= \left(\sum_{i=1}^{|\mathcal{M}|} \frac{\sigma_i (u_i^\top \boldsymbol{m})^2}{(\frac{\sigma_i}{\lambda}+1)^2}\right) \Bigg/ \left(\sum_{i=1}^{|\mathcal{M}|} \frac{\sigma_i (u_i^\top \boldsymbol{m})^2}{(\frac{\sigma_i}{\lambda}+1)}\right). \end{aligned} \tag{22}$$

Since $\lambda > 0$ and, $\forall i = 1, \dots, |\mathcal{M}|$, $\sigma_i > 0$, we have that $1 + \frac{\sigma_i}{\lambda} > 1$. Given this observation, it can be verified that $\forall \lambda \in (0, \infty)$, $\eta < 1$, and $\lim_{\lambda \to \infty} \eta = 1$. The fractional energy delivered to the load is therefore maximized by setting $\lambda = \infty$. This limiting case corresponds to setting the values of impedances between the sources and the load to zero. In terms of the image, this forces the mattes at the labeled pixels to be 1 for the foreground and 0 for the background.

The algorithm in [12] corresponds to the limiting case of $\lambda = \infty$. The algorithm in [11] solves the optimization problem in (17) by setting λ to be a large finite valued number. Since there is always a finite potential drop across the resistors connecting the voltage source and the grounds to the image, we note that the mattes (potentials) at the marked pixels are close to the desired values but *not* equal. In this paper, we shall always set $\lambda = \infty$. In practice, we can set λ to be a large finite valued number as in [11] and still recover high-quality mattes. However, we still prefer the choice $\lambda = \infty$, as this avoids redundant reestimation of mattes at the marked pixels.

5.3 Estimation of Alpha Mattes for Multiple Layers

In this section, we show how to solve the matting problem for $n \geq 2$ image layers by using generalizations of the methods discussed previously. We propose to estimate the alpha mattes for multiple layers by solving the following problem:

$$\begin{aligned} &\{\boldsymbol{\alpha}_U^k\}_{k=1}^n = \\ &\underset{\{\boldsymbol{\alpha}_U^k\}_{k=1}^n}{\text{argmin}} \sum_{k=1}^{n} \left[{\boldsymbol{\alpha}_U^k}^\top L_U \boldsymbol{\alpha}_U^k + 2{\boldsymbol{\alpha}_U^k}^\top B^\top \boldsymbol{\alpha}_M^k + {\boldsymbol{\alpha}_M^k}^\top L_M \boldsymbol{\alpha}_M^k\right], \\ &\text{s.t.} \sum_{k=1}^{n} \boldsymbol{\alpha}_U^k = \mathbf{1}, \end{aligned} \tag{23}$$

where the mattes at the marked pixels are hardcoded as $\alpha_i^k = 1$ if $i \in \mathcal{M}_k$ and $\alpha_i^k = 0$ if $i \in \mathcal{M} \setminus \mathcal{M}_k$. This corresponds to minimizing the sum of the cost functions associated with the spatial regularization of the alpha mattes for each image layer, subject to the constraint that the mattes at each pixel sum up to 1. Now, notice that the joint estimation of the $n|\mathcal{U}|$ alpha mattes for all the layers in a single step can become computationally intractable as the number of layers increases. In what follows, we show how this issue can be resolved by optimally decomposing the problem in (23) into n simpler problems of alpha matte estimation for two layers using (17).

Since L_U is positive definite, the cost function in (23) is convex. Hence, (23) involves minimizing a convex function subject to linear constraints and is guaranteed to have a unique solution. This solution must satisfy the Karusch-Kuhn-Tucker (KKT) conditions [24]. To write these conditions, we consider the Lagrangian for the problem in (23) that is given as

$$\begin{aligned} &\mathcal{L}\big(\{\boldsymbol{\alpha}_U^k\}_{k=1}^n, \Gamma\big) = \\ &\frac{1}{2}\sum_{k=1}^{n} \left[{\boldsymbol{\alpha}_U^k}^\top L_U \boldsymbol{\alpha}_U^k + 2{\boldsymbol{\alpha}_U^k}^\top B^\top \boldsymbol{\alpha}_M^k\right] + \Gamma^\top \left[\sum_{k=1}^{n} \boldsymbol{\alpha}_U^k - \mathbf{1}\right], \end{aligned} \tag{24}$$

where the vector $\Gamma \in \mathbb{R}^{|\mathcal{U}|}$ contains the Lagrange multipliers for the constraints that the mattes sum up to 1 at each pixel. We have dropped the terms ${\boldsymbol{\alpha}_M^k}^\top L_M \boldsymbol{\alpha}_M^k$ since they are constant valued and do not affect the final solution. Now, given this formulation, the KKT conditions guarantee the existence of a solution $(\{\boldsymbol{\alpha}_U^k\}_{k=1}^n, \Gamma)$ (where $\{\boldsymbol{\alpha}_U^k\}_{k=1}^n$ is the solution to the original problem (23)) that satisfies the system of equations

$$\forall k \in \{1, \dots, n\}, L_U \boldsymbol{\alpha}_U^k + B^\top \boldsymbol{\alpha}_M^k + \Gamma = \mathbf{0}. \tag{25}$$

Notice that (25) requires the solution to a huge system of equations where the alpha mattes for all the layers are estimated in one step by inverting an $n|\mathcal{U}| \times n|\mathcal{U}|$ matrix. This is due to the fact that the multiplier Γ corresponding to the constraint that the mattes sum up to 1 at each pixel, when nonzero, prevents us from decoupling the system of equations in (25) into n simpler systems corresponding to alpha matte estimation for each individual layer. However, Theorem 2 states an important result that allows such decoupling. Specifically, we prove that $\Gamma = \mathbf{0}$ since the constraint that the alpha mattes sum up to 1 at each pixel is naturally satisfied.

Theorem 2. *The alpha mattes estimated for $n \geq 2$ layers by solving (23) are naturally constrained to satisfy the property that the alpha mattes at each pixel sum up to 1.*

Proof. In order to prove the statement, we assume that the mattes sum up to 1 at each unmarked pixel and show that upon substitution of this assumption in (25), it results in $\Gamma = \mathbf{0}$. This is equivalent to proving that the constraint $\sum_{k=1}^n \boldsymbol{\alpha}^k = \mathbf{1}$ is automatically satisfied without explicitly enforcing it. Assume that $\sum_{k=1}^n \boldsymbol{\alpha}_U^k = \mathbf{1}$. Also, we have by construction, $\sum_{k=1}^n \boldsymbol{\alpha}_M^k = \mathbf{1}$. Now, summing up the KKT conditions in (25) across all the image layers gives us

$$\sum_{k=1}^{n} \left[L_U \boldsymbol{\alpha}_U^k + B^\top \boldsymbol{\alpha}_M^k + \Gamma\right] = L_U \mathbf{1} + B^\top \mathbf{1} + n\Gamma = \mathbf{0}. \tag{26}$$

Recall that the vector of 1s lies in the null space of L, and hence, $L_U \mathbf{1} + B^\top \mathbf{1} = 0$. Hence, we conclude from (26)

that $\Gamma = \mathbf{0}$. This implies that the solution automatically satisfies the constraint that the mattes sum up to 1 at each unmarked pixel. □

This result follows intuitively from the well-known Superposition Theorem in electrical network theory [25]. Now, notice that the alpha mattes for each image layer can be obtained by solving (25) after substituting $\Gamma = \mathbf{0}$ as

$$\forall k \in \{1, \ldots, n\}, L_U \boldsymbol{\alpha}_U^k + B^\top \boldsymbol{\alpha}_M^k = \mathbf{0}. \quad (27)$$

Therefore, the computationally intensive problem of alpha matte estimation for $n \geq 2$ image layers that requires the solution to (25) can now be optimally decoupled into more tractable $n \geq 2$ subproblems of alpha matte estimation for two layers that require the solution to (27). Specifically, recall that $\boldsymbol{\alpha}_M^k$ is defined as $\alpha_i^k = 1$ if pixel k is marked for layer k (i.e., $i \in \mathcal{M}_k$) and $\alpha_i^k = 0$ if pixel k is marked for one of the remaining layers (i.e., $i \in \mathcal{M} \setminus \mathcal{M}_k$). Hence, the system of equations in (27) precisely corresponds to *estimating the alpha mattes for the kth image layer by solving a problem of alpha matte estimation for two layers using (17), where the kth layer is treated as the foreground and the remaining layers are treated as the background*. This strategy for alpha matte estimation for multiple layers is summarized in Algorithm 1.

Algorithm 1 (Image matting for $n \geq 2$ image layers with summation constraints)

1: Given an image, construct the Matting Laplacian L for the image as described in Section 4.

2: For each image layer $k \in \{1, \ldots, n\}$, fix the mattes at the marked representative pixels as $\alpha_i^k = 1$ if $i \in \mathcal{M}_k$ and $\alpha_i^k = 0$ if $i \in \mathcal{M} \setminus \mathcal{M}_k$.

3: Estimate the alpha mattes for the unmarked pixels for each of the n image layers as

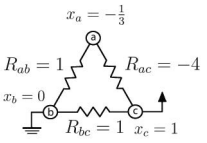
$$\forall k = 1, \ldots, n : \boldsymbol{\alpha}_U^k = -L_U^{-1} B^\top \boldsymbol{\alpha}_M^k. \quad (28)$$

Note that one may also choose finite valued λ to enforce the user's constraints rather than hardcoding the mattes at the marked pixels. Even in this case, the estimation of mattes for $n \geq 2$ layers may be optimally decoupled into n problems of alpha matte estimation for two layers.

5.3.1 Comparison with Spectral Matting [14]

Notice that if we were to estimate the mattes by minimizing the function $\Phi(\boldsymbol{\alpha}) = \sum_{k=1}^n (\boldsymbol{\alpha}^k)^\top L \boldsymbol{\alpha}^k$ without any constraints on the mattes or any user interaction, the family of solutions is given by the null space of the matrix L. Hence, Algorithm 1 may be viewed as estimating the vector of mattes as a linear combination of the eigenvectors of the Matting Laplacian L such that the resultant vector minimizes $\Phi(\boldsymbol{\alpha})$ and the entries of the marked pixels in this vector are equal to 0 or 1, as per the user's interaction.

Levin et al. [14] also estimate the alpha mattes as linear combinations of the eigenvectors of L. Essentially, Levin et al. [14] compute a certain number of eigenvectors $\{\boldsymbol{e}^i\}_{i=1}^K$ which are linearly combined to create a new set of vectors $\{\boldsymbol{e}_s^i\}_{i=1}^K$ that are *more sparse*, i.e., the vectors $\{\boldsymbol{e}_s^i\}_{i=1}^K$ have fewer number of entries with fractional values in comparison to $\{\boldsymbol{e}^i\}_{i=1}^K$. The motivation for this is that there are only a few pixels in the image which have fractional mattes.



$$L = \begin{bmatrix} 0.75 & -1 & 0.25 \\ -1 & 2 & -1 \\ 0.25 & -1 & 0.75 \end{bmatrix}$$

Fig. 4. Example of an electrical network (with a positive semidefinite Laplacian) where all of the potentials do not lie in $[0, 1]$.

Then, given the user's interaction, the algorithm makes a binary decision $y_i = 0$ or 1, for assigning each vector $\boldsymbol{e}_s^i$ to the background or the foreground, such that the vector $\boldsymbol{\alpha} = \sum_{i=1}^K y_i \boldsymbol{e}_s^i$ minimizes $\Phi(\boldsymbol{\alpha})$. Now, notice that even if the vector of ground truth alpha mattes is indeed a null vector of the Matting Laplacian, there is no guarantee that it can be written as $\sum_{i=1}^K y_i \boldsymbol{e}_s^i$, using binary valued y_i. This implies that the estimated mattes are sensitive to the quality of the vectors $\{\boldsymbol{e}_s^i\}_{i=1}^K$, which cannot be controlled explicitly since these vectors are estimated in an unsupervised manner.

Since the set $\{\boldsymbol{e}_s^i\}_{i=1}^K$ is obtained by a linear combination of the vectors in $\{\boldsymbol{e}^i\}_{i=1}^K$ and the variables y_i are binary valued, we conclude that Levin et al. [14] estimate the mattes using a restricted subset of all possible linear combinations of the eigenvectors $\{\boldsymbol{e}^i\}_{i=1}^K$. However, if the variables y_i were allowed to be real valued rather than binary valued, then we see that the vector $\boldsymbol{\alpha} = \sum_{i=1}^K y_i \boldsymbol{e}^i$ that minimizes $\Phi(\boldsymbol{\alpha})$ subject to the constraints that its entries are consistent with the user's interaction is exactly the solution that is given by Algorithm 1. Since Algorithm 1 does not restrict the linear combinations of the eigenvectors, it is not subject to the issues discussed above for [14]. However, as shown in [14] for the case $n = 2$, the mattes produced by Algorithm 1 might require more interaction to ensure that the estimated mattes are more sparse.

Finally, we note that for Spectral Matting, one may precompute the eigenvectors of L and use them to efficiently compute different sets of mattes for different user interactions. We see that Algorithm 1 can also take advantage of such precomputations by using the method of Grady and Sinop [26].

6 Constrained Alpha Matte Estimation for Multiple Image Layers

In the previous section, we discussed the problem of estimating the alpha mattes without enforcing the constraint that they take values between 0 and 1. Wang and Cohen [12] used a standard result which states that the mattes can be interpreted as probabilities in the random walks framework [7] and are hence naturally constrained to lie between 0 and 1. This result is true when the associated Laplacian matrix has negative off-diagonal entries, i.e., the corresponding graph $\mathcal{G}$ has positive edge weights. However, this does not necessarily apply to the Matting Laplacian since it has positive as well as negative off-diagonal entries. Fig. 4 gives an example where the system has a positive semidefinite Laplacian, but the obtained potentials (mattes) do not all lie in $[0, 1]$.

Algorithms in [11] and [12] resolve this issue by solving (23) and clipping the estimated alpha mattes such that they take values in $[0, 1]$. This scheme might give visually

pleasing results, but might also alter the optimality of the postprocessed mattes. Moreover, the clipped mattes at each pixel might no longer sum up to 1 across all the layers. Hence, we propose to estimate the alpha mattes by explicitly enforcing the constraint that they take values between 0 and 1 as

$$\begin{aligned}&\{\boldsymbol{\alpha}_U^k\}_{k=1}^n = \\ &\underset{\{\boldsymbol{\alpha}_U^k\}_{k=1}^n}{\operatorname{argmin}} \sum_{k=1}^{n} \left[\boldsymbol{\alpha}_U^{k\top} L_U \boldsymbol{\alpha}_U^k + 2\boldsymbol{\alpha}_U^{k\top} B^\top \boldsymbol{\alpha}_M^k + \boldsymbol{\alpha}_M^{k\top} L_M \boldsymbol{\alpha}_M^k\right], \\ &\text{such that (a) } \mathbf{0} \leq \{\boldsymbol{\alpha}_U^k\}_{k=1}^n \leq \mathbf{1} \text{ and (b) } \sum_{k=1}^{n} \boldsymbol{\alpha}_U^k = \mathbf{1}.\end{aligned} \tag{29}$$

We have hardcoded the mattes at the marked pixels to avoid redundant computation. Theorem 3 states an important result as to why we may want to explicitly enforce a constraint in (29), i.e., the constraints on the values of the estimated mattes.

Theorem 3. *The mattes obtained by clipping the solution of (23) are not the solution to (29).*

Proof. For simplicity, we consider the problem of solving (29) for $n = 2$ layers. We show later in Theorem 4 that the mattes estimated by solving (29) for $n = 2$ are naturally constrained to sum up to 1. Hence, we do not consider constraint (b) in (29) for the following proof.

Now, notice that due to the constraints on the values of the alpha mattes, the set of feasible solutions for $\boldsymbol{\alpha}_U^1$ and $\boldsymbol{\alpha}_U^2$ is given by $[0,1]^{2|\mathcal{U}|}$, which is compact and convex. Also, the cost function in (29) is a convex and continuous function of the mattes. Hence, (29) is guaranteed to have a unique minimizer. The KKT conditions guarantee the existence of $|\mathcal{U}| \times |\mathcal{U}|$ diagonal matrices with nonpositive entries $\{\Lambda_m^k\}_{m=0,1}^{k=1,2}$ such that the solution $(\boldsymbol{\alpha}_U^1, \boldsymbol{\alpha}_U^2)$ satisfies

$$\begin{aligned}&\forall k = 1,2 : \text{(a) } L_U \boldsymbol{\alpha}_U^k + B^\top \boldsymbol{\alpha}_M^k + \Lambda_0^k - \Lambda_1^k = \mathbf{0}, \\ &\text{(b) } \Lambda_0^k \boldsymbol{\alpha}_U^k = \mathbf{0} \text{ and (c)} \Lambda_1^k(\mathbf{1} - \boldsymbol{\alpha}_U^k) = \mathbf{0}.\end{aligned} \tag{30}$$

In what follows, we denote the (i,i) diagonal entry of $\{\Lambda_m^k\}_{m=0,1}^{k=1,2}$ as $\lambda_{mi}^k \leq 0$. Notice that if the matte α_i^k at the ith unmarked pixel with respect to the kth image layer lies between 0 and 1, then the KKT conditions state that the associated Lagrange multipliers λ_{0i}^k and λ_{1i}^k are equal to zero. Consequently, the ith row of the first equation in (30) gives us the result that

$$\alpha_i^k = \frac{\sum_{j \in \mathcal{N}_i} \boldsymbol{w}_{ij} \alpha_j^k}{\sum_{j \in \mathcal{N}_i} \boldsymbol{w}_{ij}}.$$

This implies that α_i^k can be expressed as the weighted average of the mattes of the neighboring pixels. But, when $\alpha_i^k = 0$ or $\alpha_i^k = 1$, we see from (30) that α_i^k is *not* the weighted average of the mattes at the neighboring pixels and this is accounted for by the Lagrange multipliers λ_{0i}^k and λ_{1i}^k. Recall from earlier that the methods in [11] and [12] clip the estimated mattes to take values between 0 and 1. This step is equivalent to introducing Lagrange multipliers. However, after this clipping, one also needs to readjust the mattes at the unmarked pixels that lie in the neighborhoods of the pixels whose mattes are clipped. Specifically, they need to be adjusted so that they still satisfy the KKT conditions. This step is neglected in [11] and [12]. This is precisely why the clipped solution of (23) is not equal to the solution of (29). □

From Theorem 3, we see that we need to estimate the mattes as per Algorithm 2 to enforce that the estimated alpha mattes satisfy the required constraints.

Algorithm 2 (Image matting for $n \geq 2$ image layers with positivity + summation constraints)
1: Given an image, construct the Matting Laplacian L for the image as described in Section 4.
2: For each image layer $j \in \{1, \ldots, n\}$, fix the mattes at the seeds as $\boldsymbol{\alpha}_i^k = 1$ if $i \in \mathcal{M}_j$ and $\boldsymbol{\alpha}_i^k = 0$ if $i \in \mathcal{M} \setminus \mathcal{M}_j$.
3: Estimate the alpha mattes $\{\boldsymbol{\alpha}_U^k\}_{j=1}^n$ for the n image layers as the solution to (29).

Unfortunately, as argued earlier, the related optimization can be computationally cumbersome, in practice. Hence, we again propose to decompose the problem of alpha matte estimation for n layers into n problems of alpha matte estimation for two layers. Specifically, we propose to estimate the mattes at the unmarked nodes for each of the n layers by solving the following problem:

$$\begin{aligned}&\forall k = 1, \ldots, n: \boldsymbol{\alpha}_U^k = \\ &\quad \underset{\{\boldsymbol{\alpha}_U^k\}}{\operatorname{argmin}} \left[\boldsymbol{\alpha}_U^{k\top} L_U \boldsymbol{\alpha}_U^k + 2\boldsymbol{\alpha}_U^{k\top} B^\top \boldsymbol{\alpha}_M^k + \boldsymbol{\alpha}_M^{k\top} L_M \boldsymbol{\alpha}_M^k\right], \\ &\quad \text{s.t. } \mathbf{0} \leq \boldsymbol{\alpha}_U^k \leq \mathbf{1}.\end{aligned} \tag{31}$$

We have essentially relaxed the constraint that the alpha mattes sum up to 1 at each pixel. As shown in the previous section, it is precisely this relaxation that allows us to decouple the original problem of alpha matte estimation for multiple layers into simpler subproblems dealing with two image layers only. However, in the previous section, we saw that this constraint was satisfied automatically when there were no constraints on the values of the alpha mattes. In the case where we enforce the mattes to take values in $[0,1]$, Theorem 4 states that the sum of the alpha mattes at each pixel is naturally constrained to be 1 only when $n = 2$ and not otherwise.

Theorem 4. *The alpha mattes obtained as the solution to (29) are naturally constrained to sum up to 1 at each pixel when $n = 2$, but not when $n > 2$.*

Proof. The KKT conditions for (29) guarantee the existence of diagonal matrices $\{\Lambda_0^k\}_{k=1}^n$ and $\{\Lambda_1^k\}_{k=1}^n$ and a vector Γ such that the solution $\{\boldsymbol{\alpha}_U^k\}_{k=1}^n$ satisfies

$$\begin{aligned}&\forall 1 \leq k \leq n, L_U \boldsymbol{\alpha}_U^k + B^\top \boldsymbol{\alpha}_M^k + \Lambda_0^k - \Lambda_1^k + \Gamma = \mathbf{0}, \text{and} \\ &\forall 1 \leq k \leq n, \Lambda_0^k \boldsymbol{\alpha}_u^k = \mathbf{0}, \Lambda_i^k(\mathbf{1} - \boldsymbol{\alpha}_U^k) = \mathbf{0}.\end{aligned} \tag{32}$$

Γ acts as a Lagrange multiplier for the constraint that the mattes sum up to 1. We now assume that the mattes sum up to 1 at each pixel and inspect the value of Γ. If $\Gamma = \mathbf{0}$, it means that the estimated mattes are naturally constrained to sum up to 1 at each pixel. Assuming that

the mattes sum up to 1, we sum up the first expression in (32) across all the image layers to conclude that

$$\sum_{k=1}^{n} \Lambda_0^k - \sum_{k=1}^{n} \Lambda_1^k + n\Gamma = \mathbf{0}. \tag{33}$$

This relationship is derived using the fact that $L_U\mathbf{1} + B^\top\mathbf{1} = \mathbf{0}$. We now inspect the mattes at a pixel $i \in \mathcal{U}$ and analyze all the possible solutions. First, consider the case when the mattes $\{\alpha_i^k\}_{k=1}^{n'}$ of the first $n' < n-1$ image layers are identically equal to 0 and the remaining mattes $\{\alpha_i^k\}_{k=n'+1}^{n}$ take values in $(0,1)$. We can always consider such reordering of the image layers without any loss of generality. We see that the variables $\{\lambda_{1i}^k\}_{k=1}^{n}$ and $\{\lambda_{0i}^k\}_{k=n'+1}^{n}$ are equal to zero. Substituting these values in (33) gives $\gamma_i + \sum_{k=1}^{n'} \lambda_{0i}^k = 0$. Therefore, γ_i is nonnegative and is equal to zero if and only if the variables $\{\lambda_{0i}^k\}_{k=1}^{n'}$ are identically equal to zero. Since these variables are not constrained to be zero valued, we see that γ_i is not equal to zero. It is due to such cases that the mattes are not constrained to sum up to 1 when $n > 2$.

However, this case cannot arise when $n = 2$. In fact, for $n = 2$, we see that the alpha mattes for both layers take values in either $(0,1)$ or in $\{0,1\}$. In the first case, we see that $\lambda_{0i}^1 = \lambda_{0i}^2 = \lambda_{1i}^1 = \lambda_{1i}^1 = 0$. For the second case, assume without loss of generality that $\alpha_i^1 = 0$ and $\alpha_i^2 = 1$. We then have $\lambda_{1i}^1 = \lambda_{0i}^2 = 0$. Moreover, we notice that setting $\lambda_{0i}^1 = \lambda_{1i}^2$ satisfies the KKT conditions. Hence, we can conclude that $\gamma_i = 0$ in both cases. Consequently, the mattes are naturally constrained to sum up to 1 at each pixel for $n = 2$. □

In this case, we see that the problem of estimating the alpha mattes for multiple layers may not be optimally decomposed into simpler subproblems as in the previous section. This is due to the fact that the positivity constraints of the alpha mattes require the explicit enforcement of the constraint that the alpha mattes sum up to 1 at each pixel. Moreover, even when the alpha matte estimation can be decoupled in the case of $n = 2$, we have shown that clipping the solution given by Algorithm 1 does not correspond to the solution of Algorithm 2.

7 Experiments

In this section, we evaluate the algorithms presented in this paper. We first show that the mattes estimated by Algorithm 1 as the solution to (17) with $\lambda = \infty$ are qualitatively similar to the results obtained in [11] by solving (17) with a large finite valued λ. We then present a quantitative comparison of Algorithms 1 and 2 for the problem of estimating alpha mattes for $n = 2$ layers. Finally, we present a qualitative comparison of our proposed algorithms and the Spectral Matting algorithm in [14] for estimating the mattes for $n > 2$ image layers.

7.1 Estimation of Alpha Mattes for $n = 2$ Image Layers

In Fig. 5, we qualitatively compare the alpha mattes obtained using Algorithm 1 with the alpha mattes estimated by Levin et al. [11]. The first column indicates the user interaction overlaid on the image for which the mattes are to be estimated. The pixels labeled as the object ($\alpha = 1$) are marked in white and the pixels labeled as the background ($\alpha = 0$) are marked in black. Note that we use the exact same scribbles as in [11]. The second column displays the alpha mattes estimated in [11] by minimizing (17) with a large finite value for λ. The third column displays the mattes estimated by Algorithm 1, where we propose to minimize (17) with $\lambda = \infty$.

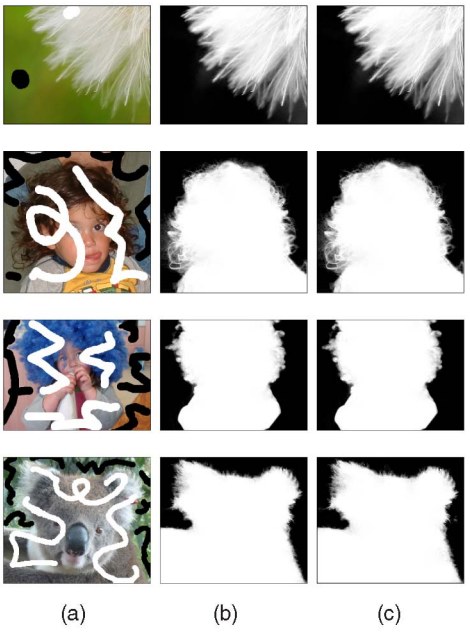


Fig. 5. Comparison of the alpha mattes estimated by [11] and Algorithm 1. (a) Image + user interaction. (b) Result of [11]. (c) Result of Algorithm 1.

We see that the mattes obtained by Algorithm 1 are very similar to the mattes estimated by Levin et al. [11]. Minor differences arise due to differences in the value of ϵ used by our method and Levin et al. [11] for constructing the Matting Laplacian as per (16). In particular, there is no significant change in the quality of the estimated mattes by hardcoding the mattes at the marked pixels.

We now compare Algorithm 1 with Algorithm 2. Note that both of these methods estimate the alpha mattes as the solution to (17) by setting $\lambda = \infty$, but Algorithm 2 estimates the solution subject to the additional constraint that the estimated mattes take values between 0 and 1. Algorithm 1 has no such restriction and simply clips the values of the estimated mattes to lie between 0 and 1. In order to compare these two algorithms, we carry out a quantitative comparison on the database used in [15], which contains 27 high-quality images. Since the database provides the ground truth alpha mattes, we can generate several trimaps for estimating the alpha mattes. We start with the *perfectly tight trimap* where all of the pixels which are pure foreground ($\alpha = 1$) or pure background ($\alpha = 0$) are labeled and the pixels with fractional α are unmarked. The algorithms are then required to estimate the alpha mattes in the unmarked regions. We inspect the performance of these algorithms for *wider dilated trimaps* which are obtained by dilation of the unmarked region in the aforementioned tight trimap. We increase the dilation in steps of 2 pixels and compare the

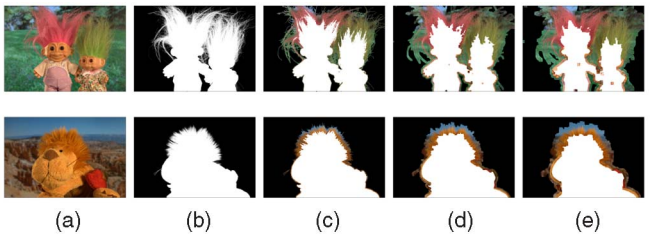


Fig. 6. Typical examples of the scenarios used for the quantitative comparison of Algorithms 1 and 2. The units of dilation mentioned in brackets denote the number of pixels by which the trimap is dilated. In trimaps (c)-(e), we have superimposed the ground truth $\boldsymbol{\alpha}_M$ on the labeled regions and displayed the true image for the unmarked regions where $\boldsymbol{\alpha}_U$ is to be estimated. (a) Composite image. (b) Ground truth matte. (c) Tight trimap. (d) Dilated trimap (10). (e) Dilated trimap (20).

performance up to a dilation of 20 pixels. Some representative examples of the trimaps used in our testing scenario are given in Fig. 6.

The error between the estimated mattes α and the ground truth α^* is quantitatively evaluated using three error metrics discussed in [27], namely,

1. **SAD**—The sum of absolute differences, which is given as $\sum_i |\alpha_i - \alpha_i^*|$.
2. **MSE**—The mean squared error, which is given as $\frac{1}{|\mathcal{U}|}\sum_i (\alpha_i - \alpha_i^*)^2$.
3. **Gradient error**—The gradient error, which is defined as $\sum_i (\nabla\alpha_i - \nabla\alpha_i^*)^2$, where $\nabla\alpha_i$ and $\nabla\alpha_i^*$ are the normalized gradients of the alpha mattes at pixel i. These gradients are computed by convolving the mattes with first-order Gaussian derivative filters with variance $\sigma = 1.4$.

The results of this test are shown in Fig. 7. A common trend is present across the different error metrics. Algorithm 2 performs marginally better than Algorithm 1 when the trimap is tight and this is true for all of the error metrics. However, as we dilate the trimap, the quality of the mattes estimated by Algorithm 2 degrades faster than Algorithm 1. When the unknown region in the perfect trimap is dilated by 6 pixels, Algorithm 2 performs worse than Algorithm 1 across all three metrics. This disparity in performance persists as the dilation is increased. Also, we noted in our experiments that for the alpha mattes computed by Algorithm 1 with the tight trimaps, the mean percentage of unmarked pixels for which

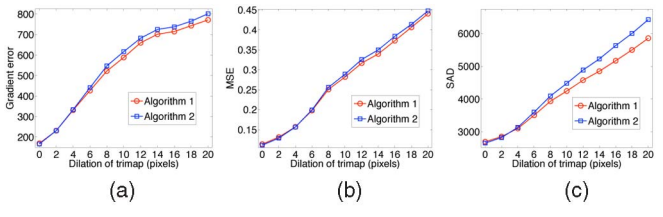


Fig. 7. Quantitative comparison of Algorithms 1 and 2 on a database of 27 high-quality images. Algorithm 1 estimates the alpha mattes without any constraints on their values and subsequently clips them to take values between 0 and 1. Algorithm 2 estimates the alpha mattes with the constraint that they take values between 0 and 1. A pdf file with the zoomed version of these plots can be found at http://www.cis.jhu. edu/~dheeraj/downloads/PAMI09-supp.pdf. (a) Gradient error. (b) Mean square error. (c) Sum of absolute differences.

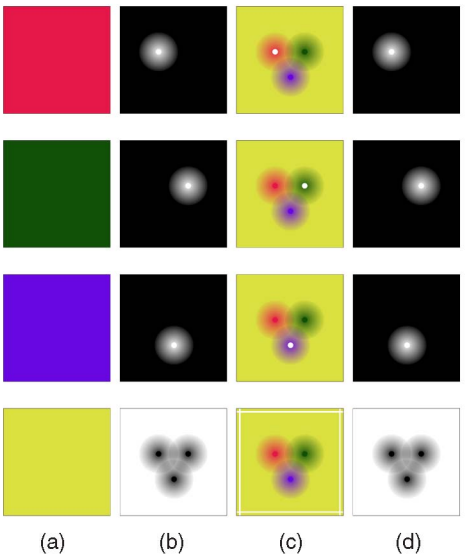


Fig. 8. Toy example for estimation of alpha mattes for multiple image layers using Algorithm 1. (a) Image layer. (b) Ground truth $\boldsymbol{\alpha}^k$. (c) Labels for layer. (d) Estimated $\boldsymbol{\alpha}^k$.

we had to clip the alpha mattes is 7.61 percent. This percentage increases to 26.1 percent when we use the trimaps dilated by 20 pixels.

In general, the user is expected to label very few pixels which would correspond to a highly dilated version of the tight trimap. We note that algorithms in [9], [10], [15] first segment the image using these few labeled pixels and use them to create a trimap for alpha matte estimation. Even in such cases, it is unclear as to how tight the trimap would be. Hence, we would prefer to use Algorithm 1 over Algorithm 2, in practice. Recall that the former also provides the attractive property that the problem of alpha matte estimation for multiple layers can be decomposed into simpler subproblems of alpha matte estimation for two layers.

7.2 Estimation of Alpha Mattes for $n \geq 2$ Image Layers

We first present a simple test on a synthetically generated image to provide a proof of concept that under the conditions specified in Theorem 1, the mattes for multiple layers may be estimated in closed form. In Fig. 8, we present a toy example where the image contains four layers. Each layer is constant in color and the layers are shown in the first column. The ground truth mattes used for the composition are shown in the second column. In the third column, we show the user interaction, where the labeled pixels are superimposed on the composite image in white. The mattes estimated using Algorithm 1 are shown in the last column. We see that the estimated mattes are qualitatively similar to the ground truth. Notice that due to the nature of the alpha mattes, different regions might have as less as one image layer and as many as all four image layers. Algorithm 1 is able to extract the correct alpha mattes even in the regions where the dimensionality of the composite image intensities is less than 4 by treating them as degenerate cases of a four-dimensional composite image patch.

In Figs. 9, 10, 11, 12, 13, 14, and 15, we present examples of estimating alpha mattes for multiple image layers in natural images. We compare four sets of alpha mattes for

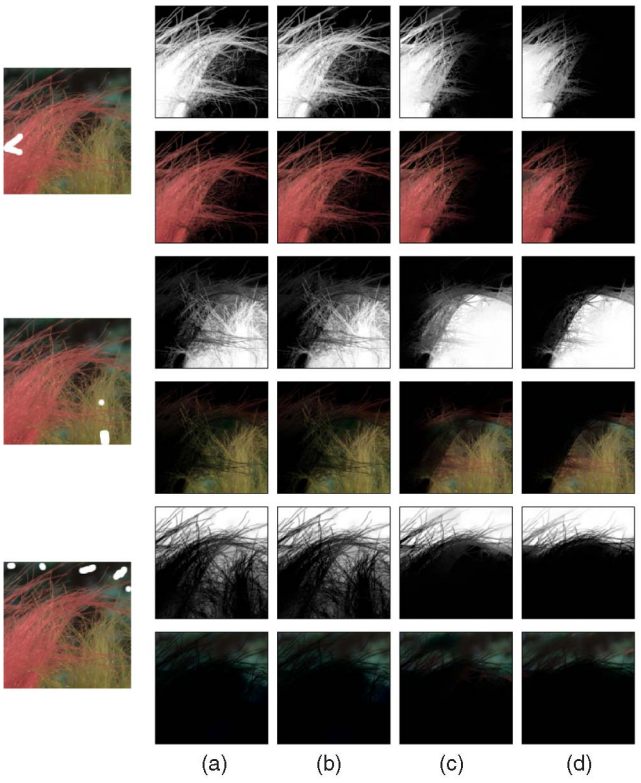


Fig. 9. An example of estimating mattes for three layers from an image of toy trolls. The first column shows the user interaction for each layer. (a) Algorithm 1. (b) Algorithm 2. (c) Spectral Matting. (d) SM-enhance.

Alpha mattes predicted using color models learnt from the marked pixels.

Alpha mattes estimated using the predicted alpha mattes as well as spatial regularization.

Layers reconstructed using the aforementioned alpha mattes.

Fig. 10. This figure shows the alpha mattes for the image of the troll dolls, which were computed using color models that are learned from the user interaction in addition to using the Matting Laplacian for spatial regularization.

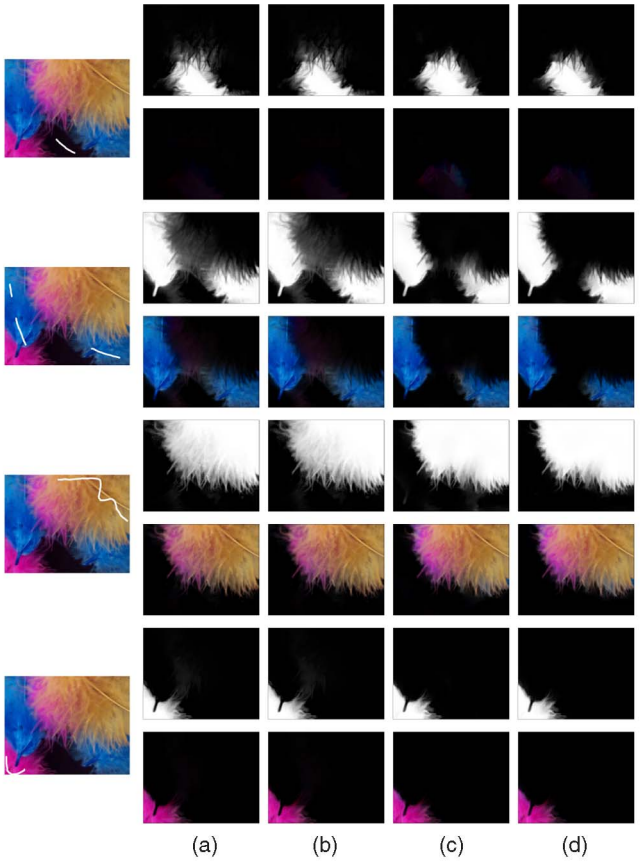
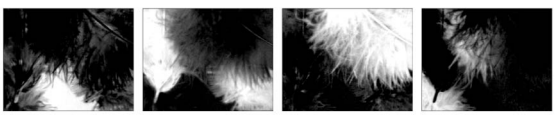


Fig. 11. An example of estimating mattes for four layers from an image of feathers, which can be found at http://www.confettico.co.uk/userimages/feathers.jpg. The first column shows the user interaction for each layer. (a) Algorithm 1. (b) Algorithm 2. (c) Spectral Matting. (d) SM-enhance.



Alpha mattes predicted using color models learnt from the marked pixels.

Alpha mattes estimated using the predicted alpha mattes as well as spatial regularization.

Layers reconstructed using the aforementioned alpha mattes.

Fig. 12. This figure shows the alpha mattes for the image of the feathers, which were computed using color models that are learned from the user interaction in addition to using the Matting Laplacian for spatial regularization.

each image. The first set corresponds to the mattes estimated using Algorithm 1 which are then clipped to take values between 0 and 1. The second set corresponds to the mattes obtained using a modification of Algorithm 2. Specifically, for computational ease, we relax the constraints that the alpha mattes must sum up to 1 at each pixel and estimate the mattes of each layer by solving (31). The third set corresponds to alpha mattes estimated by the Spectral Matting algorithm of Levin et al. [14], using the code available at http://www.vision.huji.ac.il/SpectralMatting. Since the

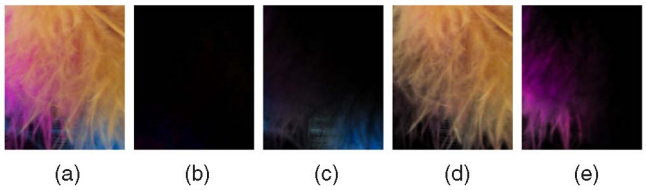


Fig. 13. The first column shows a zoom-in of a crop of the image of the feathers for which we are estimating mattes. The other columns show zoomed in versions of the reconstructed layers that are shown in the third row of Fig. 12. Notice that all of the layers overlap at the center of the image. We are able to extract the different layers accurately in most places. There is, however, some bleeding across the layers due to errors in the estimated mattes. (a) Composite image. (b) Background. (c) Blue feathers. (d) Yellow feathers. (e) Pink feathers.

algorithm estimates alpha mattes for two layers given the user's interaction, we estimate the alpha matte for each layer by treating the remaining layers as background. The fourth set, which we denote as SM-enhance, corresponds to mattes obtained by enhancing the third set of mattes by enforcing sparsity.

We present the results as follows: In the first column of each set of results, the pixels labeled for each layer are shown in either white or black, depending on the visibility of the labels. For each algorithm, we show the mattes estimated for each layer and also the contribution of that image layer (i.e., $\alpha_i^k F_i^k$) to the composite image. The layers are reconstructed as described in [11]. Due to lack of ground truth mattes for $n > 2$ layers, we qualitatively evaluate the results.

In our first example shown in Fig. 9, we consider the problem of extracting alpha mattes for three different layers from a crop of the image of troll dolls shown previously in Fig. 2. The three different layers correspond to the background, the pink hair of one doll, and the green hair of the other doll. As mentioned earlier in Section 1, there are several regions in the center of the image where

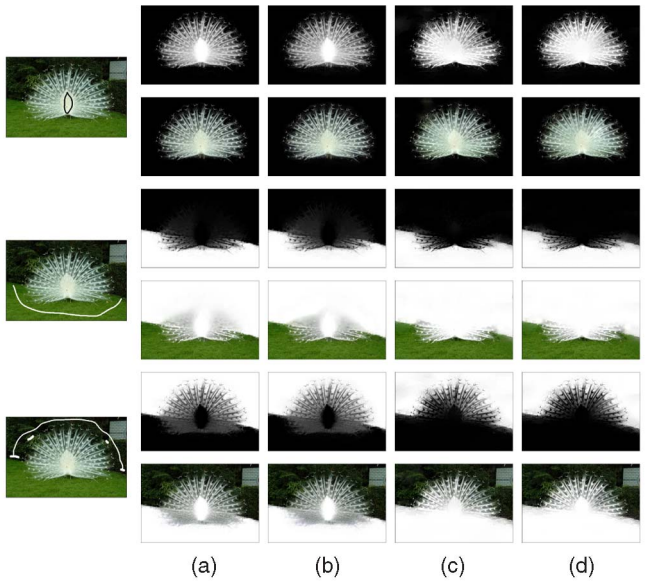


Fig. 14. An example of estimating mattes for three layers from an image of a peacock. The first column shows the user interaction for each layer, superimposed in white or black to ensure visibility. (a) Algorithm 1. (b) Algorithm 2. (c) Spectral Matting. (d) SM-enhance.

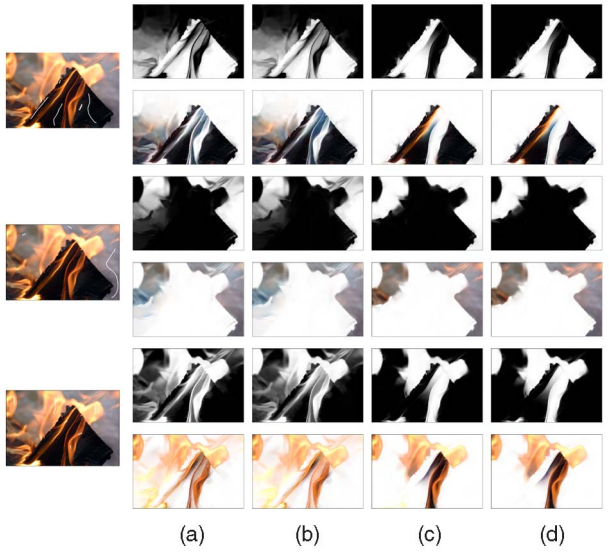


Fig. 15. An example of estimating mattes for three layers from an image of a burning book. The first column shows the user interaction for each layer. (a) Algorithm 1. (b) Algorithm 2. (c) Spectral Matting. (d) SM-enhance.

we have $n > 2$ image layers. Fig. 11 shows the alpha mattes estimated using different algorithms and also the image layers that are reconstructed using these estimated mattes. We see that, qualitatively, Algorithms 1 and 2 perform better than Spectral Matting. Spectral Matting truncates the alpha mattes for the pink hair and overestimates the alpha mattes for the green hair since the algorithm is trying to make the alpha mattes as discrete valued as possible. In contrast, Algorithms 1 and 2 are able to capture the mattes for the pink hair very well. They, however, do have some errors in the layer for the green hair. Specifically, they do let some portions of the background and the pink hair bleed into the layer for the green hair.

Notice that, in the center of the composite image, there are several regions where the background is visible through the hair. In such cases, the mattes for the green hair and the pink hair must be close to zero. However, there are several places where the mattes for the green hair are erroneously estimated as nonzero by Algorithms 1 and 2. This is primarily due to the tendency of the Matting Laplacian to produce smooth solutions [17]. As described in the proof of Theorem 3, this smoothing happens because the matte at each pixel is the weighted average of the mattes at this neighboring pixels. However, we also notice that the estimated mattes do *capture* the structure of the hair. Hence, we could have hoped to get the correct mattes if the user marked some background pixels in this region where the mattes have been oversmoothed. Unfortunately, this would require additional and very accurate interaction. This issue can be resolved by adapting [12] to the case of multiple layers.

As proposed in [12], we use the marked pixels for each layer to learn candidate colors for that layer. In most cases of our experiments, we estimated just one color model F^k_{pred} for each layer k by taking the average of the RGB values of the pixels marked for that layer. This is a valid approximation for the layers such as the green hair or the pink hair of the trolls. In the cases of layers that cannot be described using just one color model, such as the background, we apply k-means to

the RGB values of the marked pixels for each layer to learn several candidate color models for that layer. These learned color models can be used to predict the alpha mattes for each of the unmarked pixels by using the equations described in Section 3. The alpha mattes estimated for each pixel using this procedure are shown in the first row of Fig. 10. Notice that since these alpha mattes are estimated at every pixel without any spatial regularization, the estimated solution is quite noisy. We can, however, use these candidate alpha mattes along with the spatial regularization provided by the Matting Laplacian to estimate a smoother solution for the alpha mattes. In particular, we can estimate the alpha mattes by minimizing the energy

$$E(\boldsymbol{\alpha}) = \sum_{k=1}^{n} \big[{\boldsymbol{\alpha}_U^k}^\top L_U \boldsymbol{\alpha}_U^k + 2{\boldsymbol{\alpha}_U^k}^\top B^\top \boldsymbol{\alpha}_M^k + \big(\boldsymbol{\alpha}_U^k - \boldsymbol{\alpha}_{U-\text{pred}}^k\big)^\top \Lambda \big(\boldsymbol{\alpha}_U^k - \boldsymbol{\alpha}_{U-\text{pred}}^k\big)\big], \tag{34}$$

where $\boldsymbol{\alpha}_{U-\text{pred}}^k$ are the alpha mattes predicted for the kth layer and $\Lambda \in \mathbb{R}^{|\mathcal{U}|\times|\mathcal{U}|}$ is a diagonal matrix whose (i,i) term is the confidence value for the predicted mattes for the ith unmarked pixel. This value is estimated as $\Lambda(i,i) = \lambda e^{-10^2 \times res}$, where $res = \|I_i - \sum_{k=1}^{n} \alpha_{i-\text{pred}}^k F_{\text{pred}}^k\|^2$. This strategy of estimating the alpha mattes is equivalent to generalizing Algorithm 1, where we now also utilize the value of the mattes predicted for each unmarked pixel. We can derive an equivalent generalization for Algorithm 2 in this manner. However, it is unclear as to how such additional information provided by the candidate alpha mattes predicted from the color models learned from the marked pixels may be used for the Spectral Matting algorithm.

The set of mattes for the image of the trolls which are estimated by minimizing the energy defined in (34) is shown in the second row of Fig. 10. The layers that are reconstructed using these mattes are shown in the third row. Notice that in comparison to Fig. 9, the quality of the estimated mattes shown in Fig. 10 has improved in several regions. This can be appreciated from the better reconstruction of the background layer toward the center of the image.

In our second example presented in Fig. 11, we consider a crop of an image of feathers that contains four different layers, i.e., pink, orange, and blue feathers, and the black background. It can be seen that there are several image patches that contain $n > 2$ image layers. We notice from the results in Fig. 11 that Algorithms 1 and 2 perform on par with Spectral Matting. The only difference is in the alpha mattes at the junction of the blue feathers and the orange feather. Specifically, we notice at the intersection of the blue feather on the top and the orange feather, the Spectral Matting algorithm assigns discrete valued alpha mattes to a larger number of pixels than there should be. Also, at the intersection with the other blue feather, we see that Spectral Matting blurs out the fine details at the top of the blue feather. This is not so in the case of the alpha mattes estimated by Algorithms 1 and 2. All of the algorithms do, however, err in estimating the alpha mattes for the pink feather present between the blue and the orange feather. This is primarily because there is no user interaction provided for this feather, and hence, it gets assigned to the blue and the orange feathers' layers.

This can be resolved by using the candidate alpha mattes that are predicted by using the color models learned from the marked pixels. These predicted mattes are shown in the first row of Fig. 12. The mattes estimated by using these candidate mattes as well as spatial regularization, i.e., by minimizing the energy in (34), are shown in the second row of Fig. 12. The layers reconstructed using these mattes are shown in the third row and we can see that the mattes have improved, in general. There are some errors, though, in the mattes estimated for the background and the orange feather. We further zoom into the layers in Fig. 13. Notice that there are regions where all four layers are present and that we are able to extract the layers accurately to a large extent in such regions. There are some errors in the orange feather's layer, where we see bleeding of the blue and the pink feather layers.

In Fig. 14, we present an example of extracting three image layers from an image of a peacock. We see that the mattes estimated by Spectral Matting are much sharper than the mattes estimated by our proposed methods. There are portions between the feathers of the peacock where the mattes estimated by our proposed methods bleed across the different layers. In particular, we see that there is unwanted blending of the grass and the trees layers in the reconstructed layers. These errors are not present in the results of Spectral Matting. On the other hand, notice that the mattes for the peacock layer that are estimated by Spectral Matting seem to be integer valued near the feathers, where we would expect them to be fractional.

In Fig. 15, we present an example of extracting three layers from an image of a burning book. Our method estimates erroneous fractional mattes for the region to the left of the book. Specifically, we expect the alpha mattes for the book layer to be 0 in these regions. Such errors are not present in the Spectral Matting results. Also notice that there are several portions of the fire where we expect fractional alpha mattes, e.g., the flames on the book. While our proposed algorithms estimate fractional alpha mattes, Spectral Matting estimates integer valued alpha mattes in most places. These errors can be appreciated by looking at the reconstructed layers.

In conclusion, we see that Algorithms 1 and 2 can be used to extract the alpha mattes for multiple image layers. Although there is a difference in the quantitative evaluation of their performances for the two-layer case, they do not differ much in terms of the presented qualitative results. Since Algorithm 1 imposes the least constraints in the optimization problem, it makes an attractive choice for practical use. In most of our results, the mattes produced by the Spectral Matting algorithm are qualitatively worse than our proposed algorithms. In our evaluation, we used 70-100 eigenvectors, as suggested in the Spectral Matting code. A higher number of eigenvectors might help improve the performance, but would come at the expense of additional computational cost. Moreover, as discussed at the end of Section 5.3, there is no guarantee that the vectors $\{\boldsymbol{e}_s^i\}_{i=1}^K$ might produce the desired alpha mattes.

8 Conclusions

We have proposed a solution to the problem of estimating mattes for $n \geq 2$ image layers. We generalized the color line model of Levin et al. [11] for the case of $n = 2$ image layers and derived various conditions under which we can recover the mattes for $n \geq 2$ layers, in closed form. We discussed a strategy that allows us to decouple the problem of alpha

matte estimation for $n \geq 2$ layers into n simpler subproblems of alpha matte estimation for two layers such that the constraint that the mattes at each pixel sum up to 1 across the different layers is naturally satisfied. Since the mattes estimated by this method are not constrained to lie between 0 and 1, we also studied the optimization problem of estimating the mattes subject to this additional constraint.

ACKNOWLEDGMENTS

This work was supported by Johns Hopkins University startup funds and US Office of Naval Research Grant YIP N00014-09-1-0839. The authors would like to thank Dr. Carsten Rother and Dr. Christoph Rhemann for providing the data set used in [17] and also Dr. Anat Levin for making the images and code of [11] and [14] publicly available. They would also like to thank Stuart Chandler for providing the image of the feathers used in Figs. 11, 12, and 13.

REFERENCES

[1] A. Berman, A. Dadourian, and P. Vlahos, "Method for Removing from an Image the Background Surrounding a Selected Object," US Patent 6,134,346, Oct. 2000.
[2] M.A. Ruzon and C. Tomasi, "Alpha Estimation in Natural Images," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 1018-1025, 2000.
[3] Y.-Y. Chuang, B. Curless, D. Salesin, and R. Szeliski, "A Bayesian Approach to Digital Matting," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* vol. 2, pp. 264-271, 2001.
[4] J. Sun, J. Jia, C.-K. Tang, and H.-Y. Shum, "Poisson Matting," *ACM Trans. Graphics,* vol. 23, no. 3, pp. 315-321, 2004.
[5] C. Rother, V. Kolmogorov, and A. Blake, "'GrabCut': Interactive Foreground Extraction Using Iterated Graph Cuts," *ACM Trans. Graphics,* vol. 23, no. 3, pp. 309-314, 2004.
[6] J. Wang and M. Cohen, "An Iterative Optimization Approach for Unified Image Segmentation and Matting," *Proc. IEEE Int'l Conf. Computer Vision,* pp. 936-943, 2005.
[7] L. Grady, T. Schiwietz, S. Aharon, and R. Westermann, "Random Walks for Interactive Alpha-Matting," *Proc. Int'l Conf. Visualization, Imaging and Image Processing,* pp. 423-429, Sept. 2005.
[8] Y. Guan, W. Chen, X. Liang, Z. Ding, and Q. Peng, "Easy Matting: A Stroke Based Approach for Continuous Image Matting," *Proc. Eurographics '06 Conf.,* vol. 25, no. 3, pp. 567-576, 2006.
[9] X. Bai and G. Sapiro, "A Geodesic Framework for Fast Interactive Image and Video Segmentation and Matting," *Proc. IEEE Int'l Conf. Computer Vision,* 2007.
[10] Y. Zheng, C. Kambhamettu, J. Yu, T. Bauer, and K. Steiner, "Fuzzymatte: A Computationally Efficient Scheme for Interactive Matting," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2008.
[11] 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.
[12] J. Wang and M. Cohen, "Optimized Color Sampling for Robust Matting," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2007.
[13] J. Wang and M. Cohen, "Simultaneous Matting and Compositing," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2007.
[14] 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.
[15] C. Rhemann, C. Rother, A. Rav-Acha, and T. Sharp, "High Resolution Matting via Interactive Trimap Segmentation," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2008.
[16] C. Rhemann, C. Rother, and M. Gelautz, "Improving Color Modeling for Alpha Matting," *Proc. British Machine Vision Conf.,* 2008.
[17] D. Singaraju, C. Rother, and C. Rhemann, "New Appearance Models for Natural Image Matting," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 659-666, 2009.
[18] J. Wang and M. Cohen, "Image and Video Matting: A Survey," *Foundations and Trends in Computer Graphics and Vision,* vol. 3, no. 2, pp. 97-175, 2007.
[19] D. Singaraju and R. Vidal, "Interactive Image Matting for Multiple Layers," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2008.
[20] M. Bleyer, M. Gelautz, C. Rother, and C. Rhemann, "A Stereo Approach that Handles the Matting Problem via Image Warping," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 501-508, 2009.
[21] E. Hsu, T. Mertens, S. Paris, S. Avidan, and F. Durand, "Light Mixture Estimation for Spatially Varying White Balance," *ACM Trans. Graphics,* vol. 27, no. 3, 2008.
[22] Q. Shan, W. Xiong, and J. Jia, "Rotational Motion Deblurring of a Rigid Object from a Single Image," *Proc. IEEE Int'l Conf. Computer Vision,* pp. 1-8, 2007.
[23] I. Omer and M. Werman, "Color Lines: Image Specific Color Representation," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* vol. 2, pp. 946-953, 2004.
[24] H.W. Kuhn and A.W. Tucker, "Nonlinear Programming," *Proc. Second Berkeley Symp.,* pp. 481-492, 1951.
[25] P. Doyle and L. Snell, *Random Walks and Electric Networks,* no. 22. The Math. Soc. of Am., 1984.
[26] L. Grady and A. Sinop, "Fast Approximate Random Walker Segmentation Using Eigenvector Precomputation," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 1-8, 2008.
[27] 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.

Dheeraj Singaraju received the BTech degree in electrical engineering from the Indian Institute of Technology, Bombay, in 2004, and the MSE and PhD degrees in electrical and computer engineering from The Johns Hopkins University in 2007 and 2010, respectively. His research interests include computer vision and machine learning. He is a member of the IEEE.

René Vidal received the BS degree in electrical engineering (highest honors) from the Pontificia Universidad Catolica de Chile in 1997, and the MS and PhD degrees in electrical engineering and computer sciences from the University of California, Berkeley, in 2000 and 2003, respectively. He was a research fellow at the National ICT Australia in the Fall of 2003 and joined The Johns Hopkins University in January 2004, where he is currently an associate professor in the Department of Biomedical Engineering and the Center for Imaging Science. He has coauthored more than 100 articles in biomedical image analysis, computer vision, machine learning, hybrid systems, and robotics. He is the recipient of the 2009 US Office of Naval Research Young Investigator Award, the 2009 Sloan Research Fellowship, the 2005 US National Science Foundation (NFS) CAREER Award, and the 2004 Best Paper Award Honorable Mention (with Professor Yi Ma) for his work on "A Unified Algebraic Approach to 2D and 3D Motion Segmentation" presented at the European Conference on Computer Vision. He also received the 2004 Sakrison Memorial Prize for "completing an exceptionally documented piece of research," the 2003 Eli Jury Award for "outstanding achievement in the areas of Systems, Communications, Control, or Signal Processing," the 2002 Student Continuation Award from NASA Ames, the 1998 Marcos Orrego Puelma Award from the Institute of Engineers of Chile, and the 1997 Award of the School of Engineering of the Pontificia Universidad Catolica de Chile to the best graduating student of the school. He is a member of the IEEE and the ACM.