Interactive Image Matting for Multiple Layers

Dheeraj Singaraju   René Vidal

Center for Imaging Science, Johns Hopkins University, Baltimore, MD 21218, USA

Abstract

Image matting deals with finding the probability that each pixel in an image belongs to a user specified 'object' or to the remaining 'background'. Most existing methods estimate the mattes for two groups only. Moreover, most of these methods estimate the mattes with a particular bias towards the object and hence the resulting mattes do not sum up to 1 across the different groups. In this work, we propose a general framework to estimate the alpha mattes for multiple image layers. The mattes are estimated as the solution to the Dirichlet problem on a combinatorial graph with boundary conditions. We consider the constrained optimization problem that enforces the alpha mattes to take values in $[0, 1]$ and sum up to 1 at each pixel. We also analyze the properties of the solution obtained by relaxing either of the two constraints. Experiments demonstrate that our proposed method can be used to extract accurate mattes of multiple objects with little user interaction.

1. Introduction

Image matting refers to the problem of assigning to each pixel in an image, a probabilistic measure of whether it belongs to a desired object or not. This problem finds numerous applications in image editing, where the user is interested only in the pixels corresponding to a particular object, rather than in the whole image. In such cases, one prefers assigning soft values to the pixels rather than a hard classification. This is because there can be ambiguous areas where one cannot make clear cut decisions about the pixels' membership. Matting therefore deals with assigning a partial opacity value $\alpha \in [0, 1]$ to each pixel, such that pixels that definitely belong to the object or background are assigned a value $\alpha = 1$ or $\alpha = 0$ respectively. More specifically, the matting problem tries to estimate the value α_i at each pixel i, such that its intensity I_i can be expressed in terms of the true foreground and background intensities F_i and B_i as

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

Note that in (1), the number of unknowns is more than the number of independent equations, thereby making the matting problem ill posed. Therefore, matting algorithms typically require some user interaction that specifies the object of interest and the background, thereby embedding
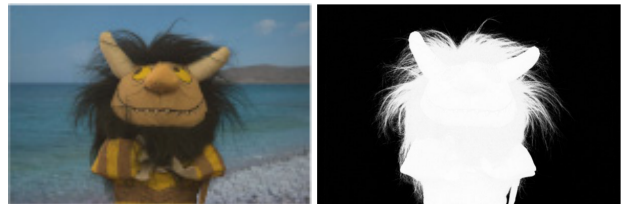

Figure 1. Typical example of the image matting problem. **Left**: Given image with a toy object. **Right**: Desired matte of the toy.

some constraints in the image. User interaction is provided in the form of a *trimap* that specifies the regions that definitely belong to the object and the background, and an intermediate region where the mattes need to be estimated. Early methods use the trimap to learn intensity models for the object as well as the background [2, 11, 3]. These learned intensity models are subsequently used to predict the alpha mattes in the intermediate region. Also, Sun *et al*. [12] use the image gradients to estimate the mattes as the solution to the Dirichlet problem with boundary conditions.

A common criticism of these methods is that they typically require an accurate trimap with a fairly narrow intermediate region to produce good results. Moreover, they give erroneous results if the background and object have similar intensity distributions or if the image intensities do not vary smoothly in the intermediate region. These issues have motivated the development of methods that extract the mattes for images with complex intensity variations. One line of work employs popular segmentation techniques for the problem of image matting [10, 5, 1]. However, these methods enforce the regions in the image to be connected. This is a major limitation, as these methods cannot deal with images such as Figure 1, where the actual matte has holes.

More recent work addresses the aforementioned issues and lets the user extract accurate mattes for challenging images with low levels of interaction [14, 8, 6, 16, 15, 13]. However, a fundamental limitation of these methods is that they extract mattes for two groups only. The only existing work dealing with multiple groups is the Spectral Matting algorithm proposed by Levin *et al*. [9]. They propose to use the eigenvectors of the so-called *Matting Laplacian* matrix to obtain the mattes for several layers. In reality, the algorithm uses multiple eigenvectors to estimate the matte for a single object. Moreover, the experiments are restricted to extracting the mattes for two layers only.

Two additional areas of concern with the aforementioned methods are the following. First, some methods do not enforce the estimated mattes to take values in $[0, 1]$. If the matte at a pixel does not lie in $[0, 1]$, its original interpretation as a probabilistic measure breaks down. Second, most existing methods exhibit a bias towards the object. In particular, consider repeating the estimation of mattes after switching the labels of the object and background in the trimap. Then, the mattes estimated at a pixel do not necessarily sum up to 1 across both trials. Consequently, one needs to postprocess the estimated mattes in order to enforce these constraints. One would like to devise a method such that these constraints are automatically satisfied, because postprocessing can degrade the quality of the mattes.

Paper contributions: In this work, we present what is to the best of our knowledge, the first method that can be used to extract accurate mattes of multiple layers in an image. We demonstrate that the algorithms of [8] and [15] that are designed for 2 layers, admit an equivalent electrical network construction that provides a unifying framework for analyzing them. We show how this framework can be used to generalize the algorithms to estimate the mattes for multiple image layers. We pose the matting problem as a constrained optimization that enforces that the alpha mattes at each pixel (a) take values in $[0, 1]$, and (b) sum up to 1 across the different image layers. Subsequently, we discuss two cases where either of these two constraints can be relaxed and we analyze the properties of the resulting solutions. Finally, we present quantitative and qualitative evaluations of our algorithms' performance on a database of nearly 25 images.

2. Image Matting for Two Groups

In this section, we discuss the previous algorithms of Levin *et al.* [8] and Wang *et al.* [15] that approach the matting problem via optimization on a combinatorial graph. We describe the construction of equivalent electrical networks, the physics of which exhibit the same optimization scheme as in these methods. In this process, we provide a unifying framework for studying these algorithms and appreciating the fine differences between them. Before delving into the details, we need to introduce some notation.

A weighted graph $\mathcal{G}$ consists of a pair $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with nodes $\boldsymbol{v}_i \in \mathcal{V}$ 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 $\boldsymbol{v}_i$ and $\boldsymbol{v}_j$ is denoted by $\boldsymbol{e}_{ij}$. The neighborhood of a node $\boldsymbol{v}_i$ is given by all the nodes $\boldsymbol{v}_j$ that share an edge with $\boldsymbol{v}_i$, and is denoted by $\mathcal{N}(\boldsymbol{v}_i)$. Each edge is assigned a value w_{ij} that is referred to as its weight. Since the edges are undirected, we have $w_{ij} = w_{ji}$. These edge weights are used to define the degree d_i of a node $\boldsymbol{v}_i$ as $d_i = \sum_{\boldsymbol{v}_j \in \mathcal{N}(\boldsymbol{v}_i)} w_{ij}$. One can use these definitions 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 w_{ij}. By construction, the Laplacian matrix has the constant vector of 1s in its null space. In fact, when the graph is connected, the vector of 1s is the only null vector.

The algorithms in [8] and [15] require the user to mark representative *seed* nodes for the different image layers. These seeds embed hard constraints on the mattes and are subsequently used to predict the mattes of the remaining unmarked nodes. The set $\mathcal{S} \subset \mathcal{V}$ contains the locations of the nodes marked as seeds and the set $\mathcal{U} \subset \mathcal{V}$ contains the locations of the unmarked nodes. By construction $\mathcal{S} \cap \mathcal{U} = \emptyset$ and $\mathcal{S} \cup \mathcal{U} = \mathcal{V}$. We further split the set $\mathcal{S}$ into the sets $\mathcal{O} \subset \mathcal{S}$ and $\mathcal{B} \subset \mathcal{S}$ that contain the locations of the seeds for the foreground object and the background, respectively. By construction, we have $\mathcal{O} \cap \mathcal{B} = \emptyset$ and $\mathcal{O} \cup \mathcal{B} = \mathcal{S}$.

The alpha matte at the pixel $\boldsymbol{v}_i$ is denoted as α_i and represents the probability that $\boldsymbol{v}_i$ belongs to the object. The mattes of all pixels in the image can be stacked into a vector as $\boldsymbol{\alpha} = [\boldsymbol{\alpha}_U^\top \quad \boldsymbol{\alpha}_S^\top]^\top$, where $\boldsymbol{\alpha}_U$ and $\boldsymbol{\alpha}_S$ correspond to the mattes of the unmarked and marked pixels, respectively.

2.1. Closed Form Solution for Image Matting

Levin *et al.* [8] assume that the image intensities satisfy the *line color model*. According to this model, the intensities of the object and background vary linearly in small patches of the image. More specifically, let (I_j^R, I_j^B, I_j^G) denote the red, blue and green components of the intensity $\boldsymbol{I}_j \in \mathbb{R}^3$ at the pixel $\boldsymbol{v}_j$. The line color model assumes that there exist constants $(a_i^R, a_i^B, a_i^G, b_i)$ for every pixel $\boldsymbol{v}_i \in \mathcal{V}$, such that the mattes α_j of the pixels $\boldsymbol{v}_j$ in a small window $\mathcal{W}(\boldsymbol{v}_i)$ around $\boldsymbol{v}_i$ can be expressed as

$$\alpha_j = a_i^R I_j^R + a_i^B I_j^B + a_i^G I_j^G + b_i, \forall \boldsymbol{v}_j \in \mathcal{W}_i. \quad (2)$$

The algorithm of [8] then proceeds by minimizing the fitting error with respect to this model in every window, while enforcing the additional constraint that the model parameters vary smoothly across the image. In particular, for some $\epsilon > 0$, the algorithm aims to minimize the cost function

$$J(\boldsymbol{\alpha}, a, b) = \sum_{\boldsymbol{v}_i \in \mathcal{V}} \left[\sum_{j \in \mathcal{W}(\boldsymbol{v}_i)} \left(\alpha_j - \sum_c a_i^c I_j^c - b_i\right)^2 + \epsilon \sum_c a_i^{c^2} \right]. \quad (3)$$

The parameter $\epsilon > 0$ prevents the algorithm from estimating the trivial model $(a_i^R, a_i^B, a_i^G) = (0, 0, 0)$. Note that the quadratic cost function $J(\boldsymbol{\alpha}, a, b)$ involves two sets of unknowns, namely, the mattes and the parameters of the line color model. If one assumes that the mattes $\boldsymbol{\alpha}$ are known, the model parameters $\{(a_i^R, a_i^B, a_i^G, b_i)\}_{i=1}^N$ can be linearly estimated in terms of the image intensities and the mattes by minimizing (3). Substituting these model parameters back in to (3) makes the cost J a quadratic function of the mattes.

In particular, the cost function can be expressed as $J(\boldsymbol{\alpha}) = \boldsymbol{\alpha}^\top L \boldsymbol{\alpha}$, where L is a $|\mathcal{V}| \times |\mathcal{V}|$ Laplacian matrix. This Laplacian is referred to as the *Matting Lapla-*

cian and essentially captures the local statistics of the intensity variations in a small window around each pixel [8]. We shall now introduce some terms needed for defining the Matting Laplacian. Consider a window $\mathcal{W}(\boldsymbol{v}_k)$ around a pixel $\boldsymbol{v}_k \in \mathcal{V}$ and let the number of pixels in this window be n_k. Let the mean and the covariance of the intensities of the pixels in $\mathcal{W}(\boldsymbol{v}_k)$ be denoted as $\mu_k \in \mathbb{R}^3$ and $\Sigma_k \in \mathbb{R}^{3 \times 3}$. Denoting an $m \times m$ identity matrix as I_m, [8] showed that the Matting Laplacian can be constructed from the edge weights $\tilde{w}_{ij}$ that are defined as

$$\sum_{k|v_i, \boldsymbol{v}_j \in \mathcal{W}(\boldsymbol{v}_k)} n_k^{-1}\left(1+(\boldsymbol{I}_i-\mu_k)^\top(\Sigma_k+\frac{\epsilon}{n_k}I_3)^{-1}(\boldsymbol{I}_j-\mu_k)\right). \quad (4)$$

Note that these weights can be positive or negative. The neighborhood structure however ensures that the graph is connected. Recall that the user marks some pixels in the image as seeds that are representative of the object and the background. To this effect, we define a vector $\boldsymbol{s} \in \mathbb{R}^{|\mathcal{S}|}$, such that an entry of $\boldsymbol{s}$ is set to 1 or 0 depending on whether it corresponds to the seed of an object or background, respectively. Also, for some $\lambda > 0$, we define a matrix $\Lambda = \lambda I_{|\mathcal{S}|}$. The mattes in the image are then estimated as

$$\boldsymbol{\alpha} = \underset{\boldsymbol{\alpha}}{\mathrm{argmin}}\left[\sum_{\boldsymbol{e}_{ij} \in \mathcal{E}} w_{ij}(\alpha_i - \alpha_j)^2 + \sum_{i \in \mathcal{S}} \lambda(\alpha_i - \boldsymbol{s}_i)^2\right]$$

$$= \underset{\boldsymbol{\alpha}}{\mathrm{argmin}}\left[\boldsymbol{\alpha}^\top L \boldsymbol{\alpha} + \lambda(\boldsymbol{\alpha}_S - \boldsymbol{s})^\top(\boldsymbol{\alpha}_S - \boldsymbol{s})\right] \quad (5)$$

$$= \underset{\boldsymbol{\alpha}}{\mathrm{argmin}}\begin{bmatrix}\boldsymbol{\alpha}_U^\top & \boldsymbol{\alpha}_S^\top & \boldsymbol{s}^\top\end{bmatrix}\begin{bmatrix}L_U & B^\top & 0 \\ B & L_S + \Lambda & -\Lambda \\ 0 & -\Lambda & \Lambda\end{bmatrix}\begin{bmatrix}\boldsymbol{\alpha}_U \\ \boldsymbol{\alpha}_S \\ \boldsymbol{s}\end{bmatrix}.$$

Noticing that the expression in (3) is non-negative, [8] showed that the Matting Laplacian is positive semi-definite. This makes the cost function in (5) convex and therefore the optimization has a unique minimizer that can be linearly estimated in closed form as

$$\begin{aligned}\boldsymbol{\alpha}_S &= [A + \Lambda]^{-1}\Lambda\boldsymbol{s}, \quad \text{and} \\ \boldsymbol{\alpha}_U &= -L_U^{-1}B^\top\boldsymbol{\alpha}_S = -L_U^{-1}B^\top[A + \Lambda]^{-1}\Lambda\boldsymbol{s},\end{aligned} \quad (6)$$

where $A = L_S - BL_U^{-1}B^\top$. The algorithm of Levin *et al.* [8] employs the above optimization by choosing a large finite valued λ, while the algorithm of Wang *et al.* [15] works in the limiting case by choosing $\lambda = \infty$. We note that [15] employs a modification of the Matting Laplacian. However, it suffices for the purpose of our analysis that the modified matrix is also a positive semi-definite Laplacian matrix.

2.2. Electrical Networks for Image Matting

It is interesting to note that there exists an equivalent electrical network that solves the optimization problem in (5). In particular, one constructs a network as shown in Figure 2, such that each node in the graph associated with the image is equivalent to a node on the network.
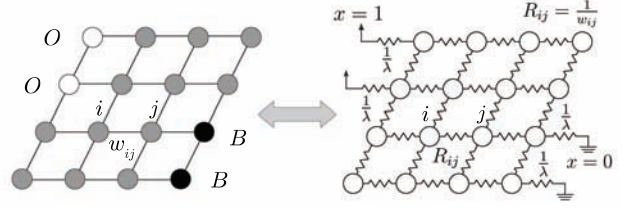

Figure 2. Equivalent construction of combinatorial graphs and electrical networks.

The edge weights correspond to the conductance values of resistors connected between neighboring nodes, *i.e.* $\frac{1}{R_{ij}} = w_{ij}$. Since the edge weights w_{ij} are not all positive, one can argue that this system might not be physically realizable. However, we note that the network is constrained to dissipate positive energy due to the positive semi-definiteness of the Laplacian. Therefore, we can think of the image as a real *resistive load* that dissipates energy. The seeded nodes are connected to the network ground and unit voltage sources by resistive impedances of value $\frac{1}{\lambda}$. In particular, the background seeds are connected to the ground and the object seeds are connected to the unit voltage sources. Hence, we have $\boldsymbol{s} = 1V$ at the voltage sources and $\boldsymbol{s} = 0V$ 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 distribute themselves such that they minimize the energy dissipated by the network. The potentials can therefore be estimated as

$$\bar{\boldsymbol{x}} = \underset{\bar{\boldsymbol{x}}}{\mathrm{argmin}}\left[\sum_{\boldsymbol{e}_{ij} \in \mathcal{E}}\frac{1}{R_{ij}}(\boldsymbol{x}_i-\boldsymbol{x}_j)^2+\sum_{i \in \mathcal{S}}\lambda(\boldsymbol{x}_i-\boldsymbol{s}_i)^2\right]. \quad (7)$$

We note that this is exactly the same expression as in (5). Therefore, if one were to construct the equivalent network and measure the potentials at the nodes they would give us the required alpha mattes.

In what follows, we shall show that as $\lambda \to \infty$, the fraction (η) of work done by the electrical sources that is used to drive the load is maximized. Note that one can use the positive definiteness of L and the connectedness of the graph to show that the matrix A is positive definite. Now, consider the singular value decomposition of the matrix A. We have $A = U\Sigma_A U^\top$, where $U = [u_1 \ldots u_{|\mathcal{S}|}]$ and $\Sigma_A = \mathrm{diag}(\sigma_1, \ldots, \sigma_{|\mathcal{S}|}), \sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_{|\mathcal{S}|} > 0$. We can then explicitly evaluate η as

$$\eta = \frac{E_{\text{load}}}{E_{\text{total}}} = \frac{\boldsymbol{x}^\top L \boldsymbol{x}}{\boldsymbol{x}^\top L \boldsymbol{x} + (\boldsymbol{x}_S - \boldsymbol{s})^\top \Lambda(\boldsymbol{x}_S - \boldsymbol{s})} \quad (8)$$

$$= \frac{(\boldsymbol{s}^\top\Lambda)\left([A+\Lambda]^{-1}A[A+\Lambda]^{-1}\right)(\Lambda\boldsymbol{s})}{(\boldsymbol{s}^\top\Lambda)\left(\Lambda^{-1}-[A+\Lambda]^{-1}\right)(\Lambda\boldsymbol{s})} = \frac{\displaystyle\sum_{i=1}^{|\mathcal{S}|}\frac{\sigma_i(u_i^\top\boldsymbol{s})^2}{(\frac{\sigma_i}{\lambda}+1)^2}}{\displaystyle\sum_{i=1}^{|\mathcal{S}|}\frac{\sigma_i(u_i^\top\boldsymbol{s})^2}{(\frac{\sigma_i}{\lambda}+1)}}.$$

Since $\lambda > 0$ and $\forall i = 1, \ldots, |\mathcal{S}|$, $\sigma_i > 0$, 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 $\lambda = \infty$. In the network, 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 pixels marked by the scribbles to be 1 for the foreground object and 0 for the background.

The algorithm of [8] solves the optimization problem in (5) 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 seeds are close to the desired values but *not* equal. The algorithm of Wang *et al.* [15] corresponds to the limiting case of $\lambda = \infty$ and forces the mattes at the seeds to be equal to the desired values. The numerical framework of the latter is referred to as the solution to the combinatorial Dirichlet problem with boundary conditions. Based on the argument given above, we favor the optimization of [15] over that of [8] because it helps in utilizing the scribbles to the maximum extent to provide knowledge about the unknown mattes.

3. Image Matting for Multiple Groups

In this section, we show how to solve the matting problem for $n \geq 2$ image layers, by using generalizations of the methods discussed in Section 2. Essentially, we assume that the intensity of each pixel in the query image can be expressed as the convex linear combination of the intensities at the same pixel location in the n image layers. We are then interested in estimating partial opacity values at each pixel as the probability of belonging to one of the image layers.

In what follows, we denote the intensity of a pixel $\boldsymbol{v}_i \in \mathcal{V}$ in the query image by I_i. Similarly, the intensity of a pixel $\boldsymbol{v}_i \in \mathcal{V}$ in the jth image layer is denoted by F_i^j, where $1 \leq j \leq n$. The alpha matte at a pixel $\boldsymbol{v}_i \in \mathcal{V}$ with respect to the jth image layer is denoted by α_i^j. Given these definitions, the matting problem requires us to estimate alpha mattes $\{\alpha_i^j\}_{j=1}^n$ and the intensities $\{F_i^j\}_{j=1}^n$ of the n image layers at each pixel $\boldsymbol{v}_i \in \mathcal{V}$, such that we have

$$I_i = \sum_{j=1}^n \alpha_i^j F_i^j, \text{ s.t. } \sum_{j=1}^n \alpha_i^j = 1, \text{ and } \{\alpha_i^j\}_{j=1}^n \in [0, 1]. \quad (9)$$

We shall pose this matting problem as an optimization problem on combinatorial graphs. While we retain most of the notations from Section 2, we need to introduce new notation for the sets that contain the seeds' locations. In particular, we split the set $\mathcal{S}$ that contains the locations of all the seeds into the sets $\mathcal{R}_1, \mathcal{R}_2, \ldots, \mathcal{R}_n$, where $\mathcal{R}_i$ contains the seed locations for the ith layer. By construction, we have $\cup_{i=1}^n \mathcal{R}_i = \mathcal{S}$ and $\forall 1 \leq i < j \leq n, \mathcal{R}_i \cap \mathcal{R}_j = \emptyset$.

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

1: Given an image, construct the matting Laplacian L for the image as described in Section 2.

2: For each image layer $j \in \{1, \cdots, n\}$, fix the mattes at the seeds as $\boldsymbol{\alpha}_i^j = 1$ if $\boldsymbol{v}_i \in \mathcal{R}_j$ and $\boldsymbol{\alpha}_i^j = 0$ if $\boldsymbol{v}_i \in \mathcal{S} \setminus \mathcal{R}_j$.

3: Estimate the alpha mattes $\{\boldsymbol{\alpha}_U^j\}_{j=1}^n$ for the unmarked pixels with respect to the n image layers as

$$\{\boldsymbol{\alpha}_U^j\}_{j=1}^n = \operatorname*{argmin}_{\{\boldsymbol{\alpha}_U^j\}_{j=1}^n} \sum_{j=1}^n \left[\boldsymbol{\alpha}^{j\top} L \boldsymbol{\alpha}^j \right] \quad (10)$$

such that (a) $\forall j \in \{1, \ldots, n\}, \mathbf{0} \leq \boldsymbol{\alpha}_U^j \leq \mathbf{1}$, and

$$\text{(b) } \sum_{j=1}^n \boldsymbol{\alpha}_U^j = \mathbf{1}. \quad (11)$$

We propose to solve this problem by minimizing the sum of the cost functions associated with the estimation of mattes for each image layer. Unlike [8] and [15], we propose to impose constraints that the mattes sum up to 1 at each pixel, and take values in $[0, 1]$. In particular, we propose that the matting problem can be solved using Algorithm 1. Notice that this is a standard optimization of minimizing a quadratic function subject to linear constraints. Recall from Section 2 that the cost function in (10), is a convex function of the alpha mattes. Also, notice that the set of feasible solutions of the mattes $\{\boldsymbol{\alpha}_U^j\}_{j=1}^n$ is compact and convex. Therefore, the optimization problem posed in Algorithm 1 is guaranteed to have a unique solution. However, this optimization can be computationally cumbersome due to the large number of unknown variables. In what follows, we shall discuss relaxing either one of the constraints and analyze its effect on the solution to the optimization problem.

3.1. Image matting for multiple layers, without constraining the sum of the mattes at a pixel

We analyze the properties of the mattes obtained by solving the optimization problem of (10) without enforcing the mattes to take values between 0 and 1. In particular, Theorem 1 states an important consequence of this case.

Theorem 1 *The alpha mattes obtained as the solution to Algorithm 1 without imposing the constraint that they take values between 0 and 1, are naturally constrained to sum up to 1 at every pixel.*

Proof. Notice that the set of feasible solutions for the mattes is convex, even when we do not impose constraint (a). The optimization problem is hence guaranteed to have a unique solution. Now, we know that this solution must satisfy the Karusch-Kuhn-Tucker (KKT) conditions [7]. In particular, the KKT conditions guarantee the existence of a vector $\Gamma \in \mathbb{R}^{|\mathcal{U}|}$ such that the solution α_U satisfies

$$\forall j \in \{1, \ldots, n\}, L_U \boldsymbol{\alpha}_U^j + B^\top \boldsymbol{\alpha}_S^j + \Gamma = \mathbf{0}. \quad (12)$$

The entries of Γ are the Lagrange multipliers for the constraints that the mattes sum up to one at each pixel. We now assume that the mattes sum up to one, and show that $\Gamma = \mathbf{0}$. This is equivalent to proving that the constraint (b) in (11) is automatically satisfied without explicitly enforcing it. Assume that $\sum_{j=1}^{n} \boldsymbol{\alpha}_U^j = \mathbf{1}$. Also, recall that by construction, $\sum_{j=1}^{n} \boldsymbol{\alpha}_S^j = \mathbf{1}$. Now summing up the KKT conditions in (12) across all the image layers gives us

$$\sum_{j=1}^{n}[L_U \boldsymbol{\alpha}_U^j + B^\top \boldsymbol{\alpha}_S^j + \Gamma] = L_U \mathbf{1} + B^\top \mathbf{1} + n\Gamma = \mathbf{0} \quad (13)$$

Recall that the vector of 1s lies in the null space of L, and hence we have $L_U \mathbf{1} + B^\top \mathbf{1} = 0$ Therefore, we can conclude from (13) that $\Gamma = \mathbf{0}$. This implies that the solution automatically satisfies the constraint that the mattes sum up to 1 at each unmarked pixel. In fact, if one does not enforce constraint (a) in (11), then the solution is the same, irrespective of whether constraint (b) is enforced or not. ∎

This is an important result which follows intuitively from the well known Superposition Theorem in electrical network theory [4]. It is important to notice that the estimation of mattes for the ith image layer is posed as the solution to the Dirichlet problem with boundary conditions. In fact, this is a simple extension of [15], where the ith image layer is treated as the foreground and the rest of the layers are treated as the background. Theorem 1 guarantees that the estimation of the mattes for any $n-1$ of the n image layers automatically determines the mattes for the remaining layer.

Wang *et al.* [15] claim that this optimization scheme finds analogies in the theory of random walks. Therefore, a standard result states that the mattes are constrained to lie between 0 and 1. However, this result is derived specifically for graphs with positive edge weights. This is not true for the graphs constructed for the matting problem since the graphs' edge weights are both positive and negative.

In fact, Figure 3 gives an example where the system has a positive semi-definite Laplacian matrix, but the obtained potentials (mattes) do not all lie in $[0, 1]$. Note that x_c takes the value $\frac{4}{3}$ on switching the locations of the voltage source and the ground. An easy fix employed by [8] and [15] to resolve this issue, is to clip all the alpha values such that they take values in $[0, 1]$. This scheme might give results that are visually pleasing, but they might not obey the KKT conditions. Moreover, the postprocessed mattes at each pixel might not sum up to 1 across all the layers anymore.

3.2. Image matting for multiple layers, without constraining the limits of the mattes

In this section, we discuss the properties of the solution of Algorithm 1, when we do not enforce the constraint that the mattes sum up to 1 at each pixel. Before we discuss

the general scheme for $n \geq 2$ layers, we consider the simple case of estimating the mattes for $n = 2$ layers. In this case, the mattes at the unmarked pixels are estimated as the solution of the quadratic programming problem given by

$$\boldsymbol{\alpha}_U^j = \underset{\boldsymbol{\alpha}_U^j}{\operatorname{argmin}} \left[{\boldsymbol{\alpha}_U^j}^\top L_U \boldsymbol{\alpha}_U^j + 2{\boldsymbol{\alpha}_U^j}^\top B \boldsymbol{\alpha}_S + \boldsymbol{\alpha}_S^\top L_S \boldsymbol{\alpha}_S \right],$$

subject to $\mathbf{0} \leq \boldsymbol{\alpha}_U^j \leq \mathbf{1}$, for $j = 1, 2$. $\quad (14)$

Note that the set of feasible solutions for the mattes of the unmarked pixels is given by $[0, 1]^{|\mathcal{U}|}$, which is a compact convex set. Also, the objective function being minimized in (14) is a convex function of the alpha mattes. Hence, the optimization problem is guaranteed to have a unique minimizer. In this case, the KKT conditions guarantee the existence of $|\mathcal{U}| \times |\mathcal{U}|$ diagonal matrices $\{\Lambda_0^j\}_{j=1}^2$ and $\{\Lambda_1^j\}_{j=1}^2$, with non-positive entries such that the solution $\boldsymbol{\alpha}_U^j$ for the jth image layer satisfies

$$L_U \boldsymbol{\alpha}_U + B^\top \boldsymbol{\alpha}_S + \Lambda_0 - \Lambda_1 = \mathbf{0}, \text{ and}$$
$$\Lambda_0^j \boldsymbol{\alpha}_U^j = \mathbf{0}, \Lambda_1^j (\mathbf{1} - \boldsymbol{\alpha}_U^j) = \mathbf{0}, j = 1, 2. \quad (15)$$

In what follows, we denote the (i, i) diagonal entry of $\{\Lambda_k^j\}_{k=0,1}^{j=1,2}$ as $\lambda_{ki}^j \leq 0$. Notice that if the matte α_i^j at the ith unmarked pixel with respect to the jth image layer lies between 0 and 1, then the KKT conditions state that the associated slack variables λ_{0i}^j and λ_{1i}^j are equal to zero. Consequently, the ith row of the first equation in (15) gives us the result that $\alpha_i^j = \frac{\sum_{\boldsymbol{v}_k \in \mathcal{N}(\boldsymbol{v}_k)} w_{ik} \alpha_k^j}{\sum_{\boldsymbol{v}_k \in \mathcal{N}(\boldsymbol{v}_i)} w_{ik}}$. In particular, this implies that α_i can be expressed as the weighted average of the mattes of the neighboring pixels. But, when $\alpha_i^j = 0$ or $\alpha_i^j = 1$, we see that α_i^j is *not* the weighted average of the mattes at the neighboring pixels and this is accounted for by the slack variables λ_{0i}^j and λ_{1i}^j. Recall from Section 3.1 that the methods of [8] and [15] clip the estimated mattes to take values between 0 and 1, when necessary. This step is somewhat equivalent to using such slack variables. However, one also needs to re-adjust the alpha mattes at the unmarked pixels that lie in the neighborhoods of the pixels whose mattes are clipped. In particular, they need to be adjusted so that they still satisfy the KKT conditions. However, this step is neglected in the methods of [8] and [15].

$$x_a = -\tfrac{1}{3}$$

$$R_{ab} = 1 \qquad R_{ac} = -4$$

$$x_b = 0$$

$$R_{bc} = 1 \quad x_c = 1$$

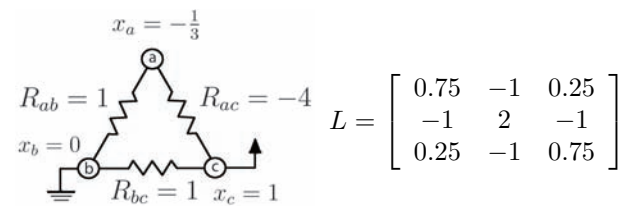$$L = \begin{bmatrix} 0.75 & -1 & 0.25 \\ -1 & 2 & -1 \\ 0.25 & -1 & 0.75 \end{bmatrix}$$

Figure 3. Example where all the potentials do not lie in $[0, 1]$.

We now consider the problem of solving this constrained optimization for $n \geq 2$ image layers. In particular, we want to analyze whether the sum of the mattes at each pixel is constrained to be 1 or not. Theorem 2 states an important property of this optimization that the sum of mattes is enforced to be 1, only when $n = 2$ and not otherwise.

Theorem 2 *The alpha mattes obtained as the solution to Algorithm 1 without imposing the constraint that they sum up to 1 at each pixel, are in fact naturally constrained to sum up to 1 at each pixel when $n = 2$, but not when $n > 2$.*

Proof. Consider the following optimization problem for estimating the mattes at the unmarked nodes.

$$\{\boldsymbol{\alpha}_U^j\}_{j=1}^n = \underset{\{\boldsymbol{\alpha}_U^j\}_{j=1}^n}{\operatorname{argmin}} \sum_{j=1}^n \left[{\boldsymbol{\alpha}_U^j}^\top L_U \boldsymbol{\alpha}_U^j + 2{\boldsymbol{\alpha}_U^j}^\top B \boldsymbol{\alpha}_S^j \right],$$

$$\text{s.t. } \mathbf{0} \leq \{\boldsymbol{\alpha}_U^j\}_{j=1}^n \leq \mathbf{1}, \text{ and } \sum_{j=1}^n \boldsymbol{\alpha}_U^j = \mathbf{1}. \qquad (16)$$

As discussed earlier, the KKT conditions guarantee the existence of diagonal matrices $\{\Lambda_0^j\}_{i=1}^n$ and $\{\Lambda_1^j\}_{i=1}^n$, and a vector Γ, such that the solution $\{\boldsymbol{\alpha}_U^j\}_{j=1}^n$ satisfies

$$\forall 1 \leq j \leq n, L_U \boldsymbol{\alpha}_U^j + B^\top \boldsymbol{\alpha}_S^j + \Lambda_0^j - \Lambda_1^j + \Gamma = \mathbf{0}, \text{ and}$$

$$\forall 1 \leq j \leq n, \Lambda_0^j \boldsymbol{\alpha}_u^j = \mathbf{0}, \Lambda_i^j(\mathbf{1} - \boldsymbol{\alpha}_U^j) = \mathbf{0}. \qquad (17)$$

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. Essentially, 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 (17) across all the image layers to conclude that

$$\sum_{j=1}^n \Lambda_0^j - \sum_{j=1}^n \Lambda_1^j + n\Gamma = \mathbf{0} \qquad (18)$$

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 $\boldsymbol{v}_i \in \mathcal{U}$ and analyze all the possible solutions. Firstly, consider the case when the mattes $\{\alpha_i^j\}_{j=1}^k$ of the first $k < n-1$ image layers are identically equal to 0 and the remaining mattes $\{\alpha_i^j\}_{j=k+1}^n$ 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 $\{\lambda_{1i}^j\}_{j=1}^n$ and $\{\lambda_{0i}^j\}_{j=k+1}^n$ are equal to zero. Substituting these values in (18), gives $\gamma_i + \sum_{j=1}^k \lambda_{0i}^j = 0$. Therefore, γ_i is non-negative, and is equal to zero if and only if the variables $\{\lambda_{0i}^j\}_{j=1}^k$ 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 precisely due to this case exactly 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 both the mattes 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 $n = 2$. ∎

4. Experiments

We denote the variants of Algorithm 1 discussed in Sections 3.1 and 3.2 as Algorithm 1a and Algorithm 1b respectively. We now evaluate these algorithms' performance on the database proposed by Levin *et al.* [9]. The database consists of images of three different toys, namely a lion, a monkey and a monster. Each toy is imaged against 8 different backgrounds and so the database has 24 distinct images. The database provides the ground truth mattes for the toys. All the images are of size 560×820 and are normalized to take values between 0 and 1. In our experiments, the Matting Laplacian for each image is estimated as described in Section 2.1, using windows of size 3×3 and with $\epsilon = 10^{-6}$.

4.1. Quantitative Evaluation for $n = 2$ Layers

We present a quantitative evaluation of our algorithms on the described database. We follow the same procedure as in [9] and define the trimap by considering the pixels whose true mattes lie between 0.05 and 0.95 and dilating this region by 4 pixels. The algorithms performance is then evaluated by considering the SSD (sum of squared differences) error between the estimated mattes and the ground truth. Table 1 shows the statistics of performance.

We see that the performance of both variants is on par. Recall that the numerical framework of Algorithm 1a is very similar to that of the closed form matting solution proposed by Levin *et al.* [8]. Consequently, both these algorithms would have similar performances. The analysis in [9] shows that the algorithm of [8] performs better than the matting algorithms of [12, 5, 14] and is outperformed only by the Spectral Matting algorithm. Therefore, we conclude that both variants of Algorithm 1 potentially match up to the state of the art algorithms.

Algorithm 1a			*Algorithm 1b*		
Toy	Mean	Median	Toy	Mean	Median
Lion	532.06	486.24	Lion	547.00	509.18
Monkey	273.34	277.03	Monkey	273.40	278.00
Monster	822.07	958.06	Monster	796.03	864.42

Table 1. Statistics of SSD errors in estimating mattes of 2 layers.

4.2. Qualitative Evaluation for $n \geq 2$ Layers

We now present results of using Algorithm 1a to extract mattes for $n > 2$ image layers with low levels of user interaction. Since there is no ground truth data available for multiple layers, we validate the performance visually. In particular, we first estimate the matte for the multiple layers using Algorithm 1a. We then use the scheme outlined in [8] to reconstruct the intensities of each image layer. Finally, we show the contribution of each image layer to the query image. Figures 4 and 5 show the results for the extraction of mattes for 3 and 5 image layers, respectively. The scribbles are color coded to demarcate between the different image layers. We note that the results visually appear to be quite good. The results highlight a disadvantage of our method that the mattes can be erroneous if the intensities near the boundary of adjoining image layers are similar. This problem can be fixed by adding more seeds, and hence at an expense of increased levels of user interaction.

5. Conclusions

We have proposed a constrained optimization problem to extract accurate mattes for multiple image layers with low levels of user interaction. We discussed two variants of the method where we relaxed the constraints of the problem and presented a theoretical analysis of the properties of the estimated mattes. Experimental evaluation of both these variants shows that they provide visually pleasing results.
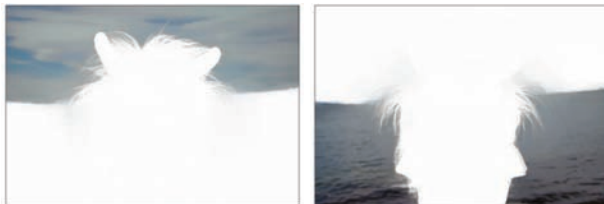
References

[1] X. Bai and G. Sapiro. A Geodesic Framework for Fast Interactive Image and Video Segmentation and Matting. In *ICCV*, pages 1–8, 2007.

[2] A. Berman, A. Dadourian, and P. Vlahos. Method for Removing from an Image the Background Surrounding a Selected Object. US Patent no. 6,134,346, October 2000.

[3] Y.-Y. Chuang, B. Curless, D. Salesin, and R. Szeliski. A Bayesian Approach to Digital Matting. In *CVPR (2)*, pages 264–271, 2001.

[4] P. Doyle and L. Snell. *Random walks and electric networks*. Number 22 in The Carus Mathematical Monographs. The Mathematical Society of America, 1984.

[5] L. Grady, T. Schiwietz, S. Aharon, and R. Westermann. Random Walks for Interactive Alpha-Matting. In *ICVIIP*, pages 423–429, September 2005.

[6] Y. Guan, W. Chen, X. Liang, Z. Ding, and Q. Peng. Easy Matting: A Stroke Based Approach for Continuous Image Matting. In *Eurographics 2006*, volume 25, pages 567–576, 2006.

[7] H. W. Kuhn and A. W. Tucker. Nonlinear Programming. In *2nd Berkeley Symposium*, pages 481–492, 1951.

[8] A. Levin, D. Lischinski, and Y. Weiss. A Closed Form Solution to Natural Image Matting. *IEEE Trans. on PAMI*, 30(2): 228–242, 2008

[9] A. Levin, A. Rav-Acha, and D. Lischinski. Spectral Matting. In *CVPR*, Pages 1–8, 2007.

[10] C. Rother, V. Kolmogorov, and A. Blake. "Grabcut": Interactive Foreground Extraction Using Iterated Graph Cuts. *ACM Trans. Graph.*, 23(3):309–314, 2004.

[11] M. A. Ruzon and C. Tomasi. Alpha Estimation in Natural Images. In *CVPR*, pages 1018–1025, 2000.

[12] J. Sun, J. Jia, C.-K. Tang, and H.-Y. Shum. Poisson Matting. *ACM Trans. Graph.*, 23(3):315–321, 2004.

[13] J. Wang, M. Agrawala, and M. F. Cohen. Soft Scissors: An Interactive Tool for Realtime High Quality Matting. *ACM Trans. Graph.*, 26(3):9, 2007.

[14] J. Wang and M. F. Cohen. An Iterative Optimization Approach for Unified Image Segmentation and Matting. In *ICCV*, pages 936–943, 2005.

[15] J. Wang and M. F. Cohen. Optimized Color Sampling for Robust Matting. In *CVPR*, Pages 1–8, 2007.

[16] J. Wang and M. F. Cohen. Simultaneous Matting and Compositing. In *CVPR*, Pages 1–8, 2007.

(a) Scribbles for marking seeds   (b) $\alpha_{\text{monster}} F_{\text{monster}}$

(a) Scribbles for marking seeds   (b) $\alpha_{\text{lion}} F_{\text{lion}}$

(c) $\alpha_{\text{sky}} F_{\text{sky}}$ (d) $\alpha_{\text{hill}} F_{\text{hill}}$