Learning-Based Hierarchical Graph for Unsupervised Matting and Foreground Estimation

Chen-Yu Tseng and Sheng-Jyh Wang, *Member, IEEE*

***Abstract*— Automatically extracting foreground objects from a natural image remains a challenging task. This paper presents a learning-based hierarchical graph for unsupervised matting. The proposed hierarchical framework progressively condenses image data from pixels into cells, from cells into components, and finally from components into matting layers. First, in the proposed framework, a graph-based contraction process is proposed to condense image pixels into cells in order to reduce the computational loads in the subsequent processes. Cells are further mapped into matting components using spectral clustering over a learning based graph. The graph affinity is efficiently learnt from image patches of different resolutions and the inclusion of multiscale information can effectively improve the performance of spectral clustering. In the final stage of the hierarchical scheme, we propose a multilayer foreground estimation process to assemble matting components into a set of matting layers. Unlike conventional approaches, which typically address binary foreground/background partitioning, the proposed method provides a set of multilayer interpretations for unsupervised matting. Experimental results show that the proposed approach can generate more consistent and accurate results as compared with state-of-the-art techniques.**

***Index Terms*— Image matting, spectral graph, segmentation.**

I. Introduction

IMAGE matting is a process to extract foreground objects from an image, along with an alpha matte. This process leads to many useful applications, such as image/video editing, image layer decomposition, and scene analysis. Typically, in image matting, the image value I_i at the pixel i can be roughly expressed as a linear combination of a foreground color value F_i and a background color value B_i. That is,

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

where α_i corresponds to the opacity of the foreground color and is usually named the alpha matte value.

Up to now, several approaches have been proposed to estimate the alpha matte from a natural image. These approaches can be generally classified into two categories: supervised matting [1], [2], [6], [9], [10], [12], [16]–[18] and unsupervised matting [3], [22]–[24]. In supervised matting, certain kinds of user's guidance, like tri-maps or scribbles, are provided to define a small number of pixels to be either "foreground" or "background". Based on these labeled pixels, a supervised matting method estimates the alpha matte values for the remaining unlabeled pixels. In contrast, unsupervised matting aims to automatically estimate the alpha matte from the input image without any user's guidance. Without users' involvement, how to accurately estimate the alpha matte becomes a very challenging task. In this paper, we focus mainly on the development of a hierarchical framework for the unsupervised matting problem. The proposed framework, however, can be slightly modified to deal with supervised matting.

In contrast to the abundant publications for supervised matting, much fewer approaches have been proposed to address the unsupervised matting problem [3], [22]–[24]. Among these unsupervised matting methods, the spectral matting method proposed in [3] is a pioneering work, which extends the principles of spectral segmentation methods [19]–[21] to obtain an unsupervised decomposition of the image by analyzing the eigenvectors corresponding the smallest eigenvalues of the image's matting Laplacian matrix. This matting Laplacian matrix computes the affinity between pixel pairs and is originally proposed in [1] to deal with supervised matting. In spectral matting, Levin et al. use the matting Laplacian matrix as the graph Laplacian in spectral analysis to automatically infer matting components. In their approach, the final alpha matte is obtained by properly assembling matting components based on certain grouping criteria.

Even though the spectral matting method has demonstrated its potential to automatically generate high-quality alpha matte, we find three major issues that need to be further investigated. The first issue is the intensive computations required by spectral analysis. To deal with this issue, a coarse-to-fine scheme has been presented in [3] to scale down the input image first, compute the coarse alpha matte thereafter, and perform up-sampling to recover a fine-level matte finally. Even though this coarse-to-fine scheme can greatly reduce the required computations, some image details may get lost and the matting quality may get degraded.

The second issue is that the grouped matting components might not always be consistent with each other. One reason for the inconsistency is that the grouping criterion based on the matting Laplacian tends to overly group components together. Recently, various kinds of grouping criteria have been presented to deal with this issue. For example, Wang et al. in [22] propose a modification of the matting Laplacian by adjusting the affinity between components based on the color similarity

Manuscript received September 15, 2013; revised January 20, 2014 and April 28, 2014; accepted May 5, 2014. Date of publication May 30, 2014; date of current version October 13, 2014. This work was supported by Winbond Electronics Corporation. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Bulent Sankur.

The authors are with the Department of Electronics Engineering, Institute of Electronics, National Chiao Tung University, Hsinchu 300, Taiwan (e-mail: rocket367@gmail.com; shengjyh@faculty.nctu.edu.tw).

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

Digital Object Identifier 10.1109/TIP.2014.2323132

between matting components. If the measured color similarity exceeds an upper/lower threshold, they increase/decrease the corresponding affinity value of the matting Laplacian. However, this approach is somewhat sensitive to threshold setting. Besides, their matting Laplacian matrix is constructed based on local analysis only. The lack of global information makes it difficult to deal with cluttered scenes. On the other hand, Hu et al. [23] present the use of a foreground/background color constraint. By assuming that pixels on the image borders are more likely to be parts of the background, they establish a background color model based on the statistics of the color values on image borders and define a foreground/background constraint over the matting components. However, this approach relies only on a simple color model and their matting Laplacian matrix is also constructed in a local way. This makes their approach less capable in dealing with cluttered scenes.

The third issue is that existing unsupervised matting approaches [3], [22], [23] focus mainly on binary partitioning of the image content even though the spectral matting method allows an image to be decomposed into several matting layers. Since a natural image may contain more than one foreground object, it would be more practical to generate multiple compositions of alpha mattes and provide multiple matting layers.

In this paper, we present a hierarchical framework, as illustrated in Fig. 1, to deal with unsupervised matting. Based on a bottom-up mechanism, the proposed framework progressively condenses image data from pixels into cells, from cells into components, and finally from components into matting layers. In our design, we apply simpler operations over the huge amount of image data in the pixel level while adopting more complicated operations over the greatly condensed data in the higher levels. This strategy enables efficient and accurate estimation of the alpha mattes.

In the bottom level of the hierarchy, image pixels are first condensed into cells though a pixel-to-cell mapping. This mapping is based on the assumption that neighboring data in the feature space tend to share similar matting values. This condensation can greatly reduce the required computations for spectral analysis without generating noticeable quality degradation. More importantly, the cell-based structure is suitable for the learning of multi-scale affinity in the construction of the cell-level graph model. The inclusion of multi-scale affinity can effectively improve the performance of spectral analysis when dealing with images of cluttered scenes. After the construction of the cell-level graph, matting components are automatically extracted by solving a graph partitioning problem. Finally, in the top level of the hierarchy, a component-level graph is introduced for the estimation of multiple matting layers. Moreover, we also propose a foreground probability distribution model to stochastically generate a list of possible foreground mattes and estimate the foreground possibility for the matting layers.

The outline of this paper is organized as follows. In Section II, the overview of the proposed method is presented. The details of the proposed framework are to be presented in Section III. Finally, in Sections IV and V, experimental results and conclusions are given.

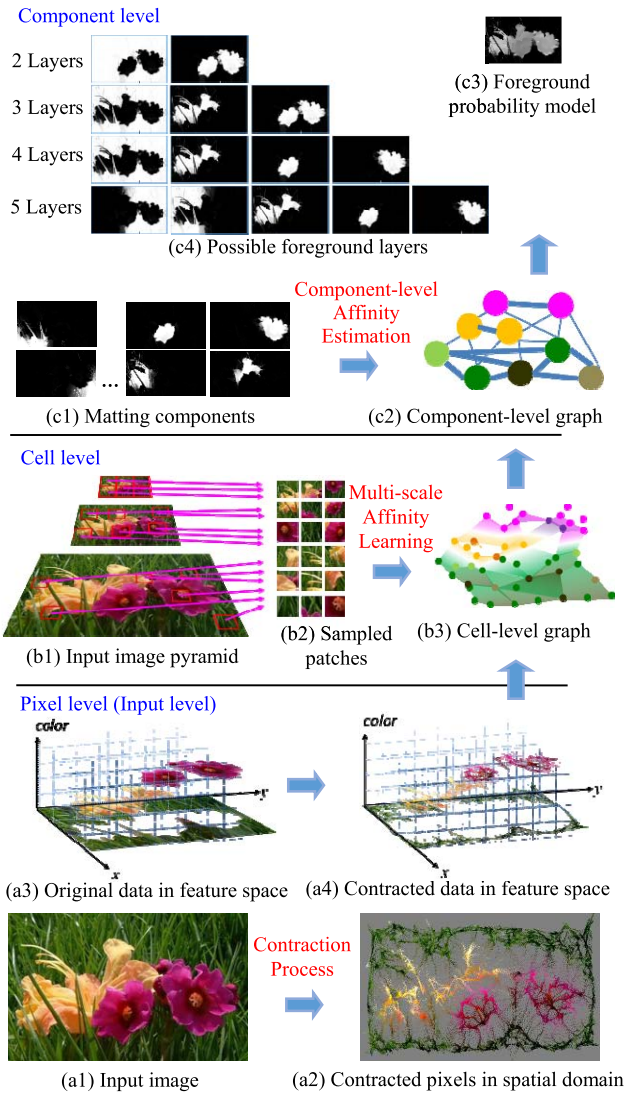


Fig. 1. Overview of the proposed framework.

II. Hierarchical Graph Model

A. Pixel-Level Graph

The first stage in our framework is to condense image pixels into compact cells in order to significantly reduce the required computations in the subsequent stages. Here, to "condense" image pixels means to spatially gather similar image pixels together. To condense pixels, we propose a local contraction process based on the minimization of a graph-based energy function. The formulation of the graph-based energy function is inspired by one of our previous works in [27], where a graph-based prior model was proposed to achieve spatially coherent inference. We adopt the concept of spatial coherence prior and use it to guide the contraction process while maintaining the image structure. The details of the contraction process are to be explained as follows.

In the local contraction process, the input image is first represented as a graph, where the vertices represent the pixel-wise intensity data and the edge between a pair of vertices represents the affinity between the corresponding pixel pair. Here, we adopt the affinity definition used in [1], where the

affinity between two pixels i and j within a local window ω_q is defined as

$$\mathbf{A}_q(i,j) = \frac{1}{|\omega_q|}\left(1+(\mathbf{I}_i-\boldsymbol{\mu}_q)^T\left(\boldsymbol{\Sigma}_q+\frac{\varepsilon}{|\omega_q|}\mathbf{U}\right)^{-1}(\mathbf{I}_j-\boldsymbol{\mu}_q)\right). \tag{2}$$

In (2), $\mathbf{I}_i$ and $\mathbf{I}_j$ are the color values of the input image $\mathbf{I}$ at i and j, μ_q is the 3×1 mean color vector in the window ω_q, Σ_q is a 3×3 covariance matrix, $|\omega_q|$ is the number of pixels in the window, $\mathbf{U}$ is the 3×3 identity matrix, and ε is a regularization term to avoid over-fitting in smooth regions [1]. In a smooth region, the entries in Σ_q would be quite small so that a small deviation caused by noise may induce a large variation of the affinity value. By properly adding a small value of ε, fluctuations of the affinity value in smooth regions can be effectively suppressed.

Based on (2), if two pixels have similar color appearance, their affinity value is expected to be large. In our approach, we place the local window at every image pixel to estimate the edge strength in the pixel-level graph. For a local window containing $r \times r$ pixels, we can estimate the affinity value for $C_2^r = \frac{r(r-1)}{2}$ edges in the graph. Due to the overlapping of the local windows, the affinity value between a pixel pair is estimated several time. By averaging these affinity estimates for each pixel pair, we build a pixel-level graph model. The computational complexity for calculating the graph edges based on the $r \times r$ local window is $O(r^2N)$, where N is the total number of pixels. In practice, we use a *3 × 3* window. The computational complexity is relatively light if compared with the complexity of the following analyses.

After the construction of pixel-level graph model, we imagine the affinity value of each edge as some kind of cohesion force which tries to pull the corresponding vertex pair closer to one another. A larger affinity value indicates a stronger cohesion force, and vice versa. These cohesion forces cause a spatial contraction of the graph model and can be modeled as follows. For the image $\mathbf{I}$, we first assume the x-coordinate and y-coordinate of its image pixels have been normalized to the range of [0,1]. Here, we denote (x_i, y_i) as the original spatial coordinates of the i^{th} pixel and denote $(\tilde{x}_i, \tilde{y}_i)$ as the spatial coordinates after the contraction process. Moreover, we represent these 2-D image coordinates in terms of 1-D vectors: $\mathbf{x} = [x_1 x_2 \ldots x_N]^T$, $\mathbf{y} = [y_1 y_2 \ldots y_N]^T$, $\tilde{\mathbf{x}} = \left[\tilde{x}_1\ \tilde{x}_2 \ldots \tilde{x}_N\right]^T$, and $\tilde{\mathbf{y}} = \left[\tilde{y}_1 \tilde{y}_2 \ldots \tilde{y}_N\right]^T$, where N denotes the total number of pixels in the image. With the above notations, the contraction process is formulated as the derivation of the optimal vectors $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{y}}$ which minimize the following energy functions:

$$E_x(\tilde{\mathbf{x}}) = \sum_{\omega_q}\sum_{i,j\in\omega_q}\mathbf{A}_q(i,j)\left(\tilde{x}_i-\tilde{x}_j\right)^2 + \lambda_x\sum_{k=1}^{N}(\tilde{x}_k-x_k)^2, \tag{3}$$

and

$$E_y(\tilde{\mathbf{y}}) = \sum_{\omega_q}\sum_{i,j\in\omega_q}\mathbf{A}_q(i,j)\left(\tilde{y}_i-\tilde{y}_j\right)^2 + \lambda_y\sum_{k=1}^{N}(\tilde{y}_k-y_k)^2. \tag{4}$$

In (3) and (4), the first term corresponds to the pair-wise cohesion forces that tend to pull pixels spatially closer, while the second term corresponds to the deviation cost that tries to preserve the original image structure. The two parameters λ_x and λ_y are used to control the strength of contraction.

To find the optimal vectors $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{y}}$, we rewrite (3) and (4) as

$$E_x(\tilde{\mathbf{x}}) = 2\tilde{\mathbf{x}}^T\mathbf{L}\tilde{\mathbf{x}} + \lambda_x(\tilde{\mathbf{x}}-\mathbf{x})^T(\tilde{\mathbf{x}}-\mathbf{x}), \text{ and} \tag{5}$$

$$E_y(\tilde{\mathbf{y}}) = 2\tilde{\mathbf{y}}^T\mathbf{L}\tilde{\mathbf{y}} + \lambda_x(\tilde{\mathbf{y}}-\mathbf{y})^T(\tilde{\mathbf{y}}-\mathbf{y}). \tag{6}$$

Here, $\mathbf{L}$ denotes the graph Laplacian, whose off-diagonal entries are defined as $\mathbf{L}(i,j) = -\sum_{\omega_q|(i,j)\in\omega_q}\mathbf{A}_q(i,j)$ and the diagonal entries are defined as $\mathbf{L}(i,i) = \sum_j\sum_{\omega_q|(i,j)\in\omega_q}\mathbf{A}_q(i,j)$. By taking the differentiation of (5) and (6) with respect to $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{y}}$, we can construct the following linear system:

$$(2\mathbf{L}+\lambda_x\mathbf{I})\tilde{\mathbf{x}} = \lambda_x\mathbf{x}, \text{ and} \tag{7}$$

$$(2\mathbf{L}+\lambda_y\mathbf{I})\tilde{\mathbf{y}} = \lambda_y\mathbf{y}. \tag{8}$$

The optimal solutions are then solved based on (7) and (8) by using the conjugate gradient method [28].

In Fig. 1(a1) and (a3), we show an image and its data distribution with respect to the original spatial coordinates (x, y). In Fig. 1(a2) and (a4), we show the image after the contraction process and the data distribution with respect to $(\tilde{x}, \tilde{y})$. Here, we use the gray color to represent vacant regions. It can be seen that pixels with similar appearance converge toward each other after the contraction process, while pixels with dissimilar appearance tend to depart from each other.

After the contraction process, we merge these deformed pixels into cells. Here, we represent each image pixel as a feature vector in the five-dimensional space W defined by the deformed spatial coordinates $(\tilde{x}, \tilde{y})$ and the RGB color values (I^R, I^G, I^B). In that feature space, spatially neighboring pixels with different colors will be pulled away from each other and will be less likely to get blended together.

The merging of image pixels into cells can be represented as a pixel-to-cell mapping process. Given the pixel i with the deformed coordinates $(\tilde{x}_i, \tilde{y}_i)$ and the RGB values (I_i^R, I_i^G, I_i^B), we map this pixel into the cell with the index $([\tilde{x}_i \times b_s], [\tilde{y}_i \times b_s], [I_i^R \times b_c], [I_i^G \times b_c], [I_i^B \times b_c])$. Here, [ ] denotes the rounding operation, b_s represents the number of spatial quantization intervals, and b_c represents the number of color quantization intervals. The pixel-to-cell mapping of the whole image is recorded in terms of an $N \times P$ binary matrix $\mathbf{M}$, where N and P denote the total number of image pixels and cells, respectively. Here, if the i^{th} pixel is classified as an element of the j^{th} cell, we define $\mathbf{M}(i,j) = 1$ while $\mathbf{M}(i,k) = 0$ for all $k \neq j$. In our experiments, we empirically fix the values of b_s and b_c to be 15 so that the number of cells P is roughly around 10k to 20k. In general cases, we found an appropriate range for these two parameters is $10 \sim 20$.

B. Cell-Level Graph

After merging pixels into cells, we aim to construct a cell-level graph model. Here, we denote $\boldsymbol{\Omega}$ as the set of cells,

which contains P cells in total. In our design, the cell-level graph model is to be constructed by assembling multiple sub-graph models, with each sub-graph model being learned from an image patch in the image. In the following paragraphs, we will first present a learning method to estimate the affinity values among these cells mapped by the image pixels in a given image patch. After that, we will introduce a sampling method to obtain a set of multi-resolution image patches. By integrating the sub-graph models generated from the set of multi-resolution image patches, we will explain how to integrate these sub-graph models into a cell-level graph model.

In image matting, it is usually assumed that the alpha matte values of the pixels within an image patch can be roughly expressed as an affine transformation of the corresponding image features inside the patch [3]. In [2], Singaraju et al. have specifically presented a few geometric interpretation models for the affine transformation of the RGB color-alpha compositing model within a patch. These models have covered four major kinds of color distributions: 1) a color plane and a color point; 2) two color points and a single color line; 3) two color lines; and 4) four color points. The coefficients of the affine transformation are assumed to be constant for one image patch, but can be varying across different patches. Since in the pixel-to-cell mapping these pixels mapped to the same cell typically share similar color appearance and spatial location, we would expect that these pixels also share similar alpha matte values. Hence, within an image patch, we assume the alpha matte values of the referred cells can also be roughly expressed as an affine transformation of the corresponding image features.

For an $r \times r$ patch in the image, centered at the pixel q, we can inspect its image pixels and use the pixel-to-cell mapping to get the set of mapped cells Ω_q, which is a subset of Ω. Here, we denote N_q as the number of cells in Ω_q. Since some pixels in the image patch may map to the same cell, N_q would be a value between 1 and r^2. For a cell i in Ω_q, we use the notation $\boldsymbol{\varphi}_i = [r_i^k,\ g_i^k,\ b_i^k]^T$ to represent its color feature, which is computed by averaging the RGB color values of all the related pixels of that cell. As aforementioned, within the patch, we assume the alpha value of the cell i can be estimated by an affine transformation of the feature vector $\boldsymbol{\varphi}_i$. That is, we have

$$\alpha_i = \begin{bmatrix} \boldsymbol{\varphi}_i^T & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{\beta} \\ \beta_0 \end{bmatrix}, \tag{9}$$

where $\boldsymbol{\beta} = [\beta_r, \beta_g, \beta_b]^T$ and β_0 is a scalar. Since we have assumed that the affine transformation coefficients $\{\boldsymbol{\beta}, \beta_0\}$ are locally constant, we can further derive an affine model for the alpha matte values of all the cells in Ω_q. Let $\boldsymbol{\alpha}_q$ be an $N_q \times 1$ vector of alpha matte values of the N_q cells and $\boldsymbol{\Phi}_q = [\tilde{\boldsymbol{\varphi}}_1^T, \ldots, \tilde{\boldsymbol{\varphi}}_{Nq}^T]^T$ be a matrix stacked by $\tilde{\boldsymbol{\varphi}}_i = \begin{bmatrix} \boldsymbol{\varphi}_i^T & 1 \end{bmatrix}$. Based on the above notations, the alpha matte prediction for all the cells corresponding to the image patch can be expressed as

$$\boldsymbol{\alpha}_q = \boldsymbol{\Phi}_q \begin{bmatrix} \boldsymbol{\beta} \\ \beta_0 \end{bmatrix}. \tag{10}$$

Equation (10) relates the alpha matte values of all the cells in Ω_q with the corresponding image features. If we assume both $\boldsymbol{\alpha}_q$ and Φ_q are given, then the optimal $\boldsymbol{\beta}$ and β_0 can be derived by minimizing the following quadratic function:

$$E(\boldsymbol{\beta}, \beta_0) = \left\| \boldsymbol{\alpha}_q - \boldsymbol{\Phi}_q \begin{bmatrix} \boldsymbol{\beta} \\ \beta_0 \end{bmatrix} \right\|^2 + c_\beta \boldsymbol{\beta}^T \boldsymbol{\beta}, \tag{11}$$

where c_β is a parameter for regularization. For the cost function in (11), the optimal solution of $\boldsymbol{\beta}$ and β_0 can be derived to be

$$\begin{bmatrix} \boldsymbol{\beta} \\ \beta_0 \end{bmatrix} = \left(\boldsymbol{\Phi}_q^T \boldsymbol{\Phi}_q + c_\beta \mathbf{D}_\beta \right)^{-1} \boldsymbol{\Phi}_q^T \boldsymbol{\alpha}_q. \tag{12}$$

In (12), we denote

$$\mathbf{D}_\beta = \begin{bmatrix} \mathbf{I}_3 & 0 \\ 0 & 0 \end{bmatrix}$$

as a 4×4 matrix, where $\mathbf{I}_3$ is the 3×3 identity matrix. By substituting (12) back to (10), a local constraint over $\boldsymbol{\alpha}_q$ can be formulated as

$$\boldsymbol{\alpha}_q = \mathbf{W}_q^T \boldsymbol{\alpha}_q, \tag{13}$$

where

$$\mathbf{W}_q = \boldsymbol{\Phi}_q \left(\boldsymbol{\Phi}_q^T \boldsymbol{\Phi}_q + c_\beta \mathbf{D}_\beta \right)^{-1} \boldsymbol{\Phi}_q^T.$$

In (13), $\mathbf{W}_q$ is an $N_q \times N_q$ transformation matrix. In this equation, each entry in the left-hand-side $\boldsymbol{\alpha}_q$ is expressed as a linear combination of the entries in the right-hand-side $\boldsymbol{\alpha}_q$. This means that, the alpha matte value of each cell in Ω_q can actually be expressed as a linear combination of the alpha values of the cells in Ω_q. This local constraint over $\boldsymbol{\alpha}_q$ can be further formulated as a squared error function with respect to $\boldsymbol{\alpha}_q$:

$$\begin{aligned} J_q(\boldsymbol{\alpha}_q) &= \left\| \boldsymbol{\alpha}_q - \mathbf{W}_q^T \boldsymbol{\alpha}_q \right\|^2 \\ &= \boldsymbol{\alpha}_q^T \left(\mathbf{I}_q - \mathbf{W}_q \right) \left(\mathbf{I}_q - \mathbf{W}_q \right)^T \boldsymbol{\alpha}_q \\ &= \boldsymbol{\alpha}_q^T \mathbf{L}_q \boldsymbol{\alpha}_q. \end{aligned} \tag{14}$$

In (14), $\mathbf{I}_q$ is the $N_q \times N_q$ identity matrix. The local Laplacian matrix for the cells in Ω_q is an $N_q \times N_q$ matrix defined as

$$\mathbf{L}_q = \left(\mathbf{I}_q - \mathbf{W}_q \right) \left(\mathbf{I}_q - \mathbf{W}_q \right)^T. \tag{15}$$

To interpret the local graph Laplacian matrix, we may refer to the spectral graph theory in [25] and [26]. Assume we define a graph $\boldsymbol{\Gamma}_q$, in which the vertices represent the cells in Ω_q and the edge between a pair of vertices represents the affinity between the corresponding cell pair. For $\boldsymbol{\Gamma}_q$, its corresponding graph Laplacian matrix is defined as

$$\mathbf{L}_q = \mathbf{D}_q - \mathbf{A}_q, \tag{16}$$

where $\mathbf{D}_q$ is the degree matrix and $\mathbf{A}_q$ is the affinity matrix. The entry $\mathbf{A}_q(ij)$ represents the affinity value between the cells i and j, while the degree matrix $\mathbf{D}_q$ is a diagonal matrix with its diagonal term being defined as

$$\mathbf{D}_q(i, i) = \sum_{j=1}^{N_q} \mathbf{A}_q(i, j). \tag{17}$$

In our approach, we do not explicitly define the affinity matrix for the cell-level graph. Instead, the affinity information is derived based on the local learning scheme expressed

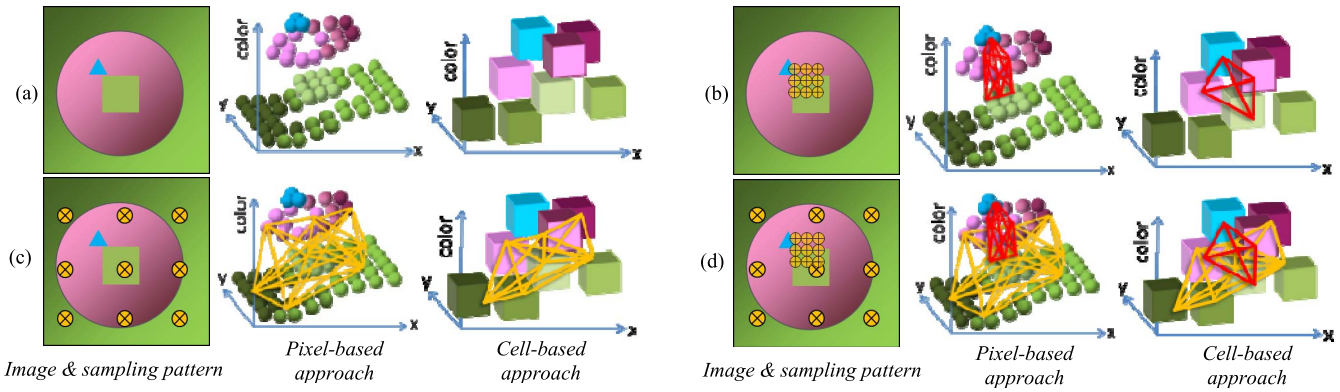


Fig. 2. Illustration of affinity computation. (a) Image. (b) Near-range affinity computation. (c) Far-range affinity computation. (d) Two-scale affinity computation.

in (14). Furthermore, the local cost function in (14) can also be interpreted as

$$J_q(\boldsymbol{\alpha}_q) = \boldsymbol{\alpha}_q^T \mathbf{L}_q \boldsymbol{\alpha}_q = \sum_{i=1}^{N_q}\sum_{j=1}^{N_q} \frac{1}{2}\mathbf{A}_q(i,j)\left\|\alpha_i - \alpha_j\right\|^2, \tag{18}$$

where α_i denotes the i^{th} element of the vector $\boldsymbol{\alpha}_q$.

After having introduced how we can learn a local graph Laplacian matrix $\mathbf{L}_q$ based on a single image patch, now we present the proposed multi-scale approach for the construction of the cell-level graph model. In the following paragraphs, we will first explain the benefit of the proposed multi-scale approach. After that, we will explain how to learn the cell-level graph based on a set of multi-resolution image patches sampled from an image pyramid, as illustrated in Fig. 1(b1) and (b2).

In Fig. 2, we use an example to compare the estimation of the affinity value at different spatial ranges. In the left image of Fig. 2(a), we show an image that contains a pink circular object with a square hole, together with a small blue triangular object. In the middle and right of Fig. 2(a), we illustrate the corresponding feature distribution in the pixel level and the cell level, respectively. Here, we use circles and cubes to represent pixels and cells. In Fig. 2(b) and 2(c), we illustrate the use of a local window at an image pixel, with 3×3 sampling pixels to explore the affinity information around that pixel. Here, each $\oplus$ symbol in the left image represents a pixel in a sampling pattern. Based on Equation (2), we calculate the affinity value for each pairing of the 9 sampling pixels. In the middle of Fig. 2(b), we show the corresponding coverage of pixel pairs in the feature space in terms of red edges. In this case, only the affinity values of adjacent sampling pixels are explored and there is no way to find the affinity between the light green pixels within the hole and the dark green pixels outside the pink object. In contrast, in the case of Fig. 2(c), where we explore the affinity among distant sampling pixels, some details may get lost, like the relation between the small blue triangle and the pink circle. To explore the affinity for both near and far ranges, we propose the multi-scale approach as illustrated in Fig. 2(d), where a small-scale window is additionally placed around the blue triangle to recover the missed affinity information in Fig. 2(c).

In the right column of Fig. 2, we show the estimation of affinity values in the cell level. It can be seen that both the near-range and far-range affinity information among cells can be explored based on the multi-scale sampling scheme illustrated in Fig. 2(d). Nevertheless, in the cell level, a lot of redundant affinity computations can be saved as compared to the case of pixel-level affinity estimation.

The concept of multi-scale affinity estimation is implemented by computing affinity over a set of multi-resolution image patches. These multi-resolution image patches are sampled from a Gaussian image pyramid as shown in Fig. 1(b1). Here, we construct a J-level image pyramid from the input image by recursively performing a down-sampling process with the sampling rate d_s along both x and y directions. Starting from the coarsest image, we use a sliding window to extract image patches. For these image paths, we perform a local affinity learning process, which is to be explained later, to estimate the affinity values among cells. At this stage, the estimation of the affinity information is like the far-range case illustrated in Fig. 2(c) and some detailed affinity information may be missing. However, as we progressively scan the image from low-resolution to high-resolution, more and more details get revealed and we can achieve multi-scale affinity estimation as illustrated in Fig. 2(d). In general, most affinity information can be extracted from the low-resolution image; only a small percentage of detailed affinity information needs to be extracted from the higher-resolution images. Hence, in practice, we use a local window to completely scan through the lowest-resolution image while only sample some of the image patches in the higher-resolution images to learn the cell-level affinity.

For the sampling of image patches in the higher-resolution images, we adopt a residual-based scheme to compensate for the missing details caused by the down-sampling process. More precisely, we map individually low-resolution pixels and high-resolution pixels into grid cells to form two sets of cells. The difference between these two sets of cells indicates the missing parts after the down sampling process. Based on the residual cells, we identify the corresponding pixels and place sampling patches around these pixels.

Assume we denote S_{patch} as the set of multi-resolution image patches. For each image patch in S_{patch}, we construct a local Laplacian matrix $\mathbf{L}_q$ based on (14) and (15). With the set of $\mathbf{L}'_q$s, now we introduce a global cost function for the construction of the cell-level graph model. Here, we re-express Equation (14) as

$$J_q(\boldsymbol{\alpha}) = \boldsymbol{\alpha}^T \mathbf{L}'_q \boldsymbol{\alpha}. \tag{19}$$

In (19), $\boldsymbol{\alpha} \equiv [\alpha_1, \alpha_2, \ldots, \alpha_P]^T$ and $\mathbf{L}'_q$ denotes a $P \times P$ local Laplacian matrix, whose entries for the pairs of cells in Ω_q are equal to the corresponding ones in $\mathbf{L}_q$, while the remaining entries are set to zero. Based on (19), the global cost function is defined as a weighted sum of $J_q(\boldsymbol{\alpha})$ with the weighting function $w(l_q)$. That is

$$J(\boldsymbol{\alpha}) = \sum_{q \in \Omega} w(l_q) \left(\boldsymbol{\alpha}^T \mathbf{L}'_q \boldsymbol{\alpha} \right). \tag{20}$$

The weighting function $w(l_q)$ in (20) reflects the importance of each image patch according to the corresponding image level in the pyramid. Here, l_q denotes the level index. In the image pyramid, the number of pixels in the jth level is $(d_s^2)^{j-1}$ times smaller than that of the original image after being scaled down $(j-1)$ times in both x and y directions with the downsampling rate d_s. By assuming that each pixel in the jth-level image is $(d_s^2)^{j-1}$ times more important than that of the original image pixel, the weighting $w(l_q)$ is defined as

$$w(l_q) = (d_s^2)^{l_q - 1}. \tag{21}$$

Moreover, we can reformulate Equation (20) in a more compact form as

$$J(\boldsymbol{\alpha}) = \boldsymbol{\alpha}^T \mathbf{L} \boldsymbol{\alpha}, \tag{22}$$

$$\text{where} \quad \mathbf{L} = \sum_{q \in \Omega} w(l_q) \mathbf{L}'_q. \tag{23}$$

The $\mathbf{L}$ in (22) is named the cell-level matting Laplacian (CML) matrix for the cell-level graph model.

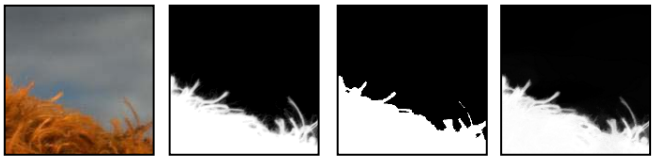
Since the CML generated by (23) is basically an unnormalized Laplacian matrix, we need to normalize it before the task of spectral clustering in order to avoid unbalanced clustering. In our approach, we apply a symmetric normalization based on [21] which modifies the affinity values between pairs of cells based on the degree matrix of the cells. The normalized CML $\overline{\mathbf{L}}$ is computed as

$$\overline{\mathbf{L}} = \mathbf{D}^{-1/2} \mathbf{L} \mathbf{D}^{-1/2}. \tag{24}$$

In (24), we use the diagonal matrix $\mathbf{D}$ to denote the $P \times P$ degree matrix of the CML.

C. Component-Level Graph

After having obtained the cell-level graph, we decompose it into a set of matting components and then form the component-level graph for the estimation of foreground mattes. During the construction of component-level graph, some prior information about the foreground model will be included. In the following subsections, we will first introduce the generation of matting components and then present the construction of the component-level graph.



Input image Ground truth Matting segment Matting component

Fig. 3. An example of hard segment and alpha matte.

1) Cell-to-Component Mapping: To merge cells into components, we perform spectral clustering, followed by an optimization process to transform hard-decision matting segments into soft-decision matting components. The concept of spectral clustering is to transfer data into a high-dimensional space, where data points with high affinity tend to share similar coordinates, and then to perform clustering in that space. To achieve that, we first generate a matrix $\mathbf{E}$ based on the S normalized eigenvectors, $\mathbf{e}^1, \ldots, \mathbf{e}^S$, corresponding to the S smallest eigenvalues of the $P \times P$ cell-level Laplacian matrix $\overline{\mathbf{L}}$. In the $P \times S$ matrix $\mathbf{E} = [\mathbf{e}^1, \ldots, \mathbf{e}^S]$, the i^{th} row vector represents the coordinates of the i^{th} cell in Ω in the space spanned by these S eigenvectors. By performing k-means clustering over the row vectors of $\mathbf{E}$, we cluster the P cells into K different clusters. These K groups of cells are treated as the initial matting segments. Here, we use $KP \times 1$ binary vectors $\mathbf{c}^k$, where $1 \leq k \leq K$, to represent the hard-decision clustering result. If the k^{th} cluster contains the i^{th} cell of Ω, we set the i^{th} element of $\mathbf{c}^k$ to 1; otherwise, the i^{th} element of $\mathbf{c}^k$ is set to 0.

To transform the hard-decision segments into soft-decision matting components, we formulate a global optimization process based on the cost function in (22). Here, we treat each matting component as an assembly of cells and represent the alpha matte information of the component k in terms of a $P \times 1$ alpha matte $\boldsymbol{\alpha}^k \equiv [\alpha_1^k, \alpha_2^k, \ldots, \alpha_P^k]^T$. The i^{th} element of $\boldsymbol{\alpha}^k$ indicates the soft membership that the i^{th} cell belongs to the k^{th} component. By expecting that the soft-decision matting components shall not deviate too much from the hard-decision matting segments, the alpha matte vector $\boldsymbol{\alpha}^k$ corresponding to the k^{th} matting component is obtained by minimizing the following equation:

$$J(\boldsymbol{\alpha}^k) = \left(\boldsymbol{\alpha}^k \right)^T \mathbf{L} \left(\boldsymbol{\alpha}^k \right) + \lambda_c \left(\boldsymbol{\alpha}^k - \mathbf{c}^k \right)^T \left(\boldsymbol{\alpha}^k - \mathbf{c}^k \right), \tag{25}$$

where λ_c is a constant to control the trade-off between the matting Laplacin cost and the deviation from the matting segments. Based on (25), the optimal $\boldsymbol{\alpha}^k$ can be found by solving the following sparse system of linear equations:

$$(\mathbf{L} + \lambda_c \mathbf{I}_c) \boldsymbol{\alpha}^k = \lambda_c \mathbf{c}^k, \tag{26}$$

where $\mathbf{I}_c$ denotes the $P \times P$ identity matrix. Fig. 3 illustrates an example of comparison between the hard-decision matting segment $\underline{\mathbf{c}}^k$ and the soft-decision matting component $\boldsymbol{\alpha}^k$. It can be seen that more detailed matting values can be recovered in the matting component.

2) Component-Level Graph Construction: In our design, the component-level graph is built based on two perspectives: one is a condensed graph from the cell-level graph, while

the other is a divergence-based graph. In the cell level, we have already introduced a cell-level matting Laplacian $\overline{\mathbf{L}}$. This matting Laplacian can be further condensed into a component-level matting Laplacian. Here, we define a $P \times K$ matrix $\mathbf{T} \equiv [\boldsymbol{\alpha}^1, \ldots, \boldsymbol{\alpha}^K]$, formed by the alpha matte vectors of the K matting components, as the cell-to-component condensing function to calculate the component-level matting Laplacian $\mathbf{L}_{condensed}$:

$$\mathbf{L}_{condensed} = \mathbf{T}^T \overline{\mathbf{L}} \mathbf{T}. \tag{27}$$

Since the cell-level matting Laplacian $\overline{\mathbf{L}}$ is constructed based on local affinity learning, the lack of knowledge between spatially isolated components will be a barrier to handle more complicated scenes. To address this problem, we further introduce a divergence-based graph, which is constructed by explicitly estimating the affinity value between every component pair. For each pairing of components, we measure the Kullback–Leibler (KL) divergence between the color distributions of the two components. For the matting component k, we use a $Z \times 1$ vector $\mathbf{h}^k = [h_1^k, \ldots, h_Z^k]^T$ to denote its color distribution, where Z is the number of color bins. Here, we use h_i^k to denote the probability value in the i^{th} bin. Since we only consider color features in the divergence measure, we group cells with similar color values into a color bin. For the i^{th} color bin of the k^{th} matting component, we denote ρ_i as the set of cells belonging to this bin. Based on the above notations, we define h_i^k to be

$$h_i^k = \frac{1}{N^k} \sum_{j \in \rho_i} \alpha_j^k N_j, \tag{28}$$

where $N^k = \sum_{j=1}^{P} \alpha_j^k N_j$ is the normalization term. In (28), N_j is the number of pixels in cell j, α_j^k is the alpha value of the cell j for the k^{th} matting component. Based on the above definitions, the KL divergence between the two matting components m and n is defined as

$$\mathbf{v}_{KL}(m,n) = \frac{D_{KL}(\mathbf{h}^m \| \mathbf{h}^n) + D_{KL}(\mathbf{h}^n \| \mathbf{h}^m)}{2}, \tag{29}$$

where $D_{KL}(\mathbf{p} \| \mathbf{q}) = \sum_i \mathbf{p}(i) \log(\mathbf{p}(i)/\mathbf{q}(i))$.

For any pair of components, a high divergence value would correspond to a low affinity value. Hence, we use the sigmoid function $\sigma(x) = 1/(1 + \exp(-x))$ to define the affinity $\mathbf{A}_{KL}(m, n)$ between Components m and n as

$$\mathbf{A}_{KL}(m,n) = \sigma\left(\overline{\mathrm{v}}_{KL} - \mathbf{v}_{KL}(m,n)\right). \tag{30}$$

In (30), we use $\overline{\mathrm{v}}_{KL}$ to denote the mean of the KL divergence values over all component pairs. After having obtained the $K \times K$ divergence-based affinity matrix $\mathbf{A}_{KL}$, the diagonal degree matrix $\mathbf{D}_{KL}$ is computed as

$$\mathbf{D}_{KL}(i,i) = \sum_{j=1}^{K} \mathbf{A}_{KL}(i,j). \tag{31}$$

Finally, the divergence-based Laplacian matrix $\mathbf{L}_{KL}$ is defined as

$$\mathbf{L}_{KL} = \mathbf{D}_{KL} - \mathbf{A}_{KL}. \tag{32}$$

By combining the divergence-based Laplacian $\mathbf{L}_{KL}$ with the condensed matting Laplacian $\mathbf{L}_{condensed}$, we define the component-level graph Laplacian $\mathbf{L}_{\text{comp}}$ to be

$$\mathbf{L}_{comp} = \mathbf{L}_{condensed} + \lambda_{KL} \mathbf{L}_{KL}. \tag{33}$$

In (33), λ_{KL} is a parameter to balance the contribution between $\mathbf{L}_{condensed}$ and $\mathbf{L}_{KL}$. Here, we define this parameter based on the ratio between the sum of the condensed matting affinity degrees and the sum of the KL affinity degrees. That is

$$\lambda_{KL} = \sum_{i=1}^{K} \mathbf{L}_{condensed}(i,i) \Big/ \sum_{i=1}^{K} \mathbf{L}_{KL}(i,i). \tag{34}$$

3) Component-to-Layer Mapping: Once we have derived the component-level matting Laplacian $\mathbf{L}_{comp}$, we further introduce a component-to-layer mapping procedure based on component-level spectral clustering. Similar to the cell-to-component mapping introduced in Section *C-1*, the component-to-layer mapping is also performed based on spectral clustering. Here, a component-level matrix $\mathbf{E}_{comp}$ is generated based on the normalized eigenvectors, $\mathbf{e}_{comp}^1, \ldots, \mathbf{e}_{comp}^K$ of the $K \times K$ component-level Laplacian matrix L_{comp}. By performing k-means clustering over the row vectors of $\mathbf{E}_{comp}$, we cluster the K matting components into Q clusters, where Q is an integer ranging from 2 to K. Besides, we use $QK \times 1$ binary vectors $\mathbf{d}^q$, where $1 \le q \le Q$, to represent the clustering result. If the q^{th} cluster contains the i^{th} matting component, we set the i^{th} element of $\mathbf{d}^q$ to 1; otherwise, the i^{th} element of $\mathbf{d}^q$ is set to 0. With $\mathbf{d}^q$, we can represent the alpha matte information of the q^{th} matting layer in terms of the $P \times 1$ vector $\boldsymbol{\alpha}_{layer}^q$, which is defined as

$$\boldsymbol{\alpha}_{layer}^q = [\boldsymbol{\alpha}^1 \quad \ldots \quad \boldsymbol{\alpha}^K]\mathbf{d}^q. \tag{35}$$

In our framework, instead of directly clustering cells into matting layers, we perform spectral clustering twice. We first cluster cells into components, as mentioned in Subsection *C-1*, and then cluster components into layers, as mentioned above. In the cell-level graph, the adjacency of the graph nodes is locally computed. Even though the multi-resolution learning scheme may expand the range of local analysis, the affinity values among distant cells are still not fully explored. On the other hand, in the component-level graph, since the number of matting components is relatively small, we can estimate the affinity between every pair of components and build a fully connected graph. This fully connected graph provides the component-level analysis a more global view as compared to the cell-level analysis. In Fig. 4, we show the clustering results when we directly cluster the P cells into two, three, four, and five clusters. As a comparison, we show the two-stage clustering results in Fig. 5 and Fig. 6. By decomposing an image into a larger number of components first, followed by the component-to-layer mapping, we can obtain much more reasonable results.

Actually, in the plotting of the alpha matte layers in Figs. 4 and 5, we have applied a cell-to-pixel mapping to convert the cell-level information $\boldsymbol{\alpha}^k$ back to the pixel domain. For any pixel i, we use j to denote the corresponding cell of i and denote $\mu(j)$ as the set of cells within a neighborhood of j.

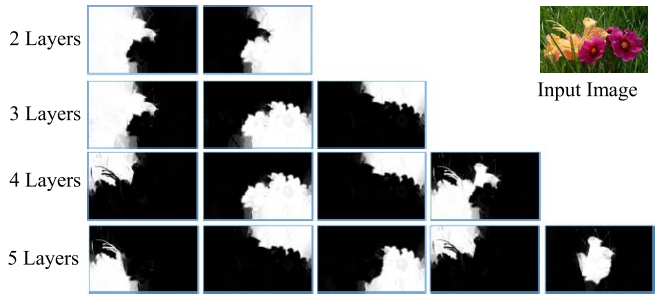


Fig. 4. Results of directly clustering cells into layers.

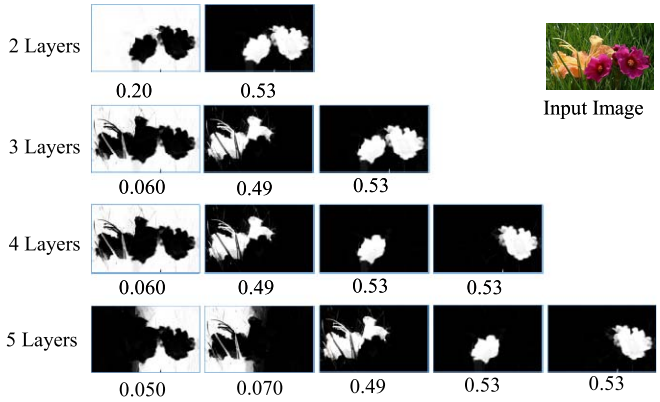


Fig. 5. Example of two-stage spectral clustering.

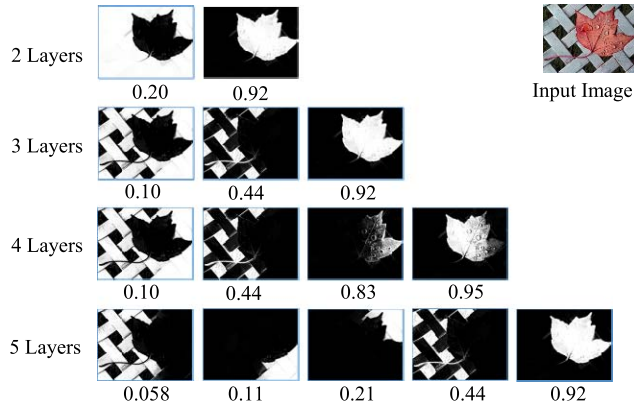


Fig. 6. Example of two-stage spectral clustering.

The pixel-level data $\mathbf{o}_i^{pixel}$ of the pixel i can be interpolated by using the cell-level data values $\mathbf{o}_k^{cell}$ of the cells in $\mu(j)$ based on the following formula:

$$\mathbf{o}_i^{pixel} = \sum_{k \in \mu(i)} \mathbf{o}_k^{cell} \cdot p_{k|i} \tag{36}$$

where

$$p_{k|i} = \{ \sum_{k \in \mu(j)} \exp\left(-\frac{\| f_i - f_k \|^2}{\sigma_f} \right) \}^{-1} \exp\left(-\frac{\| f_i - f_k \|^2}{\sigma_f} \right). \tag{37}$$

In (37), we use f_i to denote the image feature of the pixel i in the five-dimensional space W defined by the deformed spatial coordinates $(\tilde{x}, \tilde{y})$ and the RGB color values (I^R, I^G, I^B), as described in Section II-A. On the other hand, we denote f_k as the average of the feature vectors related to the cell k. The conditional probability in (37) models how likely the pixel i belongs to the cell k, based on the distance between f_i and f_k in the feature space. A shorter distance indicates a higher probability.

D. Foreground Estimation

After having presented the formation of matting layers, we further address a probability-based description of the matting layers. Here, we present each matting layer in terms of matting components and propose a scheme to estimate for each matting component the probability of being a portion of the foreground objects. In this probability-based approach, we derive the probability distribution model $p(\mathbf{b}) \equiv p(b_1, \ldots, b_K)$, where $b_k \in \{0, 1\}$ for $1 \leq k \leq K$. For the matting component k, we have $b_k = 1$ when this component is assigned as a foreground component; otherwise, $b_k = 0$. Based on the above definition, each $\mathbf{b}$ represents a foreground matte hypothesis and corresponds to an assembly of matting components. Once we have derived the probability distribution $p(\mathbf{b})$ for all possible $\mathbf{b}$'s, we can pick up a few $\mathbf{b}$'s that are more likely to represent a foreground matte.

In the proposed framework for foreground estimation, the distribution model is based on the consistency assumption that any pair of components with higher graph affinity tends to share the same foreground index; that is, these two components tend to be either both foreground or both background. Taking the image in Fig. 1 as an example, here we assume the two purple flowers have been divided into two matting components. Once one of them is classified as a part of the foreground set, the other would have a higher probability to be a foreground component too. With the consistency assumption, we evaluate a given vector $\mathbf{b}$ based on the component-level graph $\mathbf{L}_{\text{comp}}$ and a measure of "inconsistency" is defined as

$$d_{fb}(\mathbf{b}) = \mathbf{b}^T \mathbf{L}_{comp} \mathbf{b}. \tag{38}$$

Based on the definition in (38), a vector $\mathbf{b}$ with a low value of $d_{\text{fb}}(\mathbf{b})$ indicates the corresponding assembly of matting components has a high probability to be a part of the foreground matte. However, this does not imply that a proper foreground vector can thus be found simply based on this measure. One example is that the $\mathbf{b}$ vector whose entries are all ones (or all zeros) corresponds to zero inconsistency. In order to avoid this problem, we further introduce a balancing weight based on the assumption that the ratio between the foreground area and the background area should not be overly unbalanced. The balancing weight $\eta(\mathbf{b})$ is defined as

$$\eta(\mathbf{b}) = \frac{1}{N_\alpha} + \frac{1}{N_{\bar{\alpha}}}, \tag{39}$$

where $N_\alpha = \sum_{k=1}^{K} N^k b_k$ and $N_{\bar{\alpha}} = \sum_{k=1}^{K} N^k (1 - b_k)$ denote the sum of matting values in the foreground area and the background area, respectively. The term N^k has been defined before in Equation (28). Under an unbalanced circumstance, one of N_α and $N_{\bar{\alpha}}$ is small and the ratio weight $\eta(\mathbf{b})$ becomes large. By including this balancing weight $\eta(\mathbf{b})$ into the inconsistency

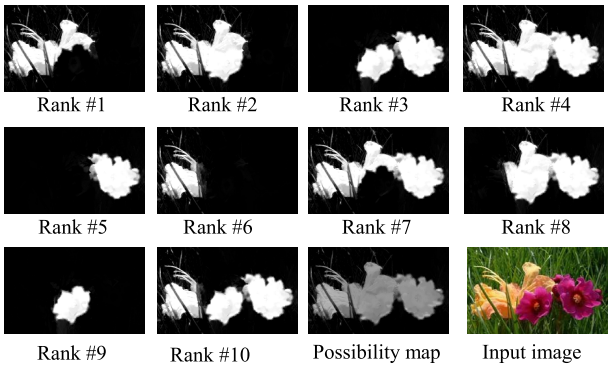


Fig. 7. Illustration of foreground analysis with the 10 leading partitions and the corresponding foreground possibility map.

measure, we redefine $d_{\text{fb}}(\mathbf{b})$ as

$$d_{fb}(\mathbf{b}) = \eta(\mathbf{b}) \left(\mathbf{b}^T \mathbf{L}_{comp} \mathbf{b}\right). \tag{40}$$

With this inconsistency measure, the probability distribution model is defined as

$$p_f(\mathbf{b}) \propto \frac{1}{1 + \exp\left(c\left(d_{fb}(\mathbf{b}) - \overline{d_{fb}}\right)\right)}, \quad \text{where} \quad \overline{d_{fb}} = \sum_{\mathbf{b} \in \Im_f} d_{fb}(\mathbf{b}). \tag{41}$$

In (41), $\Im_f$ is a set of foreground vectors, which have the smallest values of $d_{\text{fb}}(\mathbf{b})$ over all feasible $\mathbf{b}$'s. On the other hand, $\overline{d_{fb}}$ is the mean of the inconsistency measures of the vectors in $\Im_f$ and the parameter c is a constant, which is empirically chosen to be 0.02. In practice, the number of components is typically small (about 10 to 20) and we can generate all feasible vectors $\mathbf{b}$'s and check the corresponding $d_{\text{fb}}(\mathbf{b})$ for each $\mathbf{b}$. For the sake of computational efficiency, we simply ignore most $\mathbf{b}$'s which have a rather large value of $d_{\text{fb}}(\mathbf{b})$ and focus only over a few $\mathbf{b}$'s that have small values of $d_{\text{fb}}(\mathbf{b})$. With the formulation in (40) and (41), if a combination of matting components is consistent with respect to the component-level graph $\mathbf{L}_{\text{comp}}$ and is balanced with respect to the remaining components, the corresponding $p_{\text{f}}(\mathbf{b})$ will have a larger value. However, for any $\mathbf{b}$ and its complement $(\mathbf{1} - \mathbf{b})$, the values of $p_f(\mathbf{b})$ and $p_f(\mathbf{1} - \mathbf{b})$ are actually equal. That is, we cannot discriminate foreground from background simply based on the inconsistency measure in (40). Hence, we further evaluate the convexity of a matte and that of its complement by assuming that a foreground matte usually tends to be convex. Here, the convexity is measured based on the ratio of the areas between a matte and its corresponding convex hull. By comparing the convexity between any pair of $\mathbf{b}$ and $(\mathbf{1} - \mathbf{b})$, we eliminate the one with lower convexity.

Fig. 7 shows some leading mattes which correspond to these ten $\mathbf{b}$'s with the largest values of $p_f(\mathbf{b})$. It can be seen that these leading mattes typically have a large overlap with the foreground area: the flower regions. Hence, if we denote ξ as the set of leading foreground vectors with the largest values of $p_f(\mathbf{b})$, we can estimate the foreground vector $\mathbf{b}_{FG}$ as the expectation of the $\mathbf{b}$ vectors in ξ. That is,

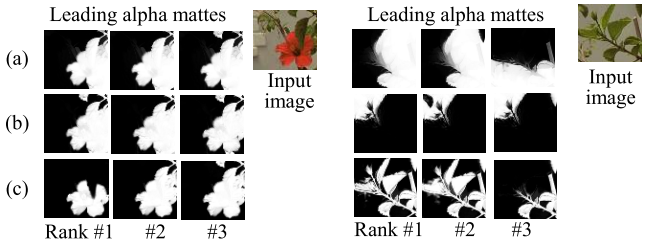


Fig. 8. Comparison of affinity measures. (a) Spectral matting [3], (b) coarse-to-fine spectral matting [3], (c) Proposed multi-scale affinity analysis.

$$\mathbf{b}_{FG} = \frac{\sum_{\mathbf{b} \in \xi} p_f(\mathbf{b}) \mathbf{b}}{\sum_{\mathbf{b} \in \xi} p_f(\mathbf{b})}. \tag{42}$$

Based on (42), we define the foreground possibility map for the foreground matte as a weighted sum of the component-level alpha matte values:

$$\boldsymbol{\alpha}_{FG} = [\, \boldsymbol{\alpha}^1 \ \ldots \ \boldsymbol{\alpha}^K \,] \cdot \mathbf{b}_{FG}. \tag{43}$$

As shown in the middle bottom of Fig. 7, the possibility map reflects how likely an image pixel belongs to the foreground region.

Finally, for the multi-layer decomposition shown in Figs. 5 and 6, we can evaluate the foreground factor for each alpha matte layer $\boldsymbol{\alpha}^q_{layer}$. Here, we define

$$F_{layer}(\boldsymbol{\alpha}^q_{layer}) = \frac{\boldsymbol{\alpha}^T_{FG} \boldsymbol{\alpha}^q_{layer}}{\mathbf{1}^T_P \boldsymbol{\alpha}^q_{layer}}, \tag{44}$$

where $\mathbf{1}_P$ denotes a $P \times 1$ all-one vector. In Fig. 5 and 6, the number listed below each matting layer indicates the foreground factor of that layer. Basically, a matting layer with a larger foreground factor is more likely to be a portion of the foreground region.

III. Experiments

A. Evaluation of Alpha Matte Estimation

Since the multi-scale affinity estimation is a foundation in our framework, we first compare in Fig. 8 how the leading alpha mattes may look like if using different kinds of affinity measures. Here, we compare the alpha measures defined in the original spectral matting [3], in the coarse-to-fine scheme [3], and in the proposed multi-scale approach. On the other hand, the leading alpha mattes are chosen based on the value of $p_f(\mathbf{b})$ defined in (41). It can be seen in Fig. 8(a) and (b) that the spectral matting approach and the coarse-to-fine scheme in [3] have difficulty in obtaining satisfactory results due to the lack of multi-scale information. In contrast, by applying the multi-scale learning scheme, the leading alpha mattes become much more discriminative for the partitioning of foreground and background.

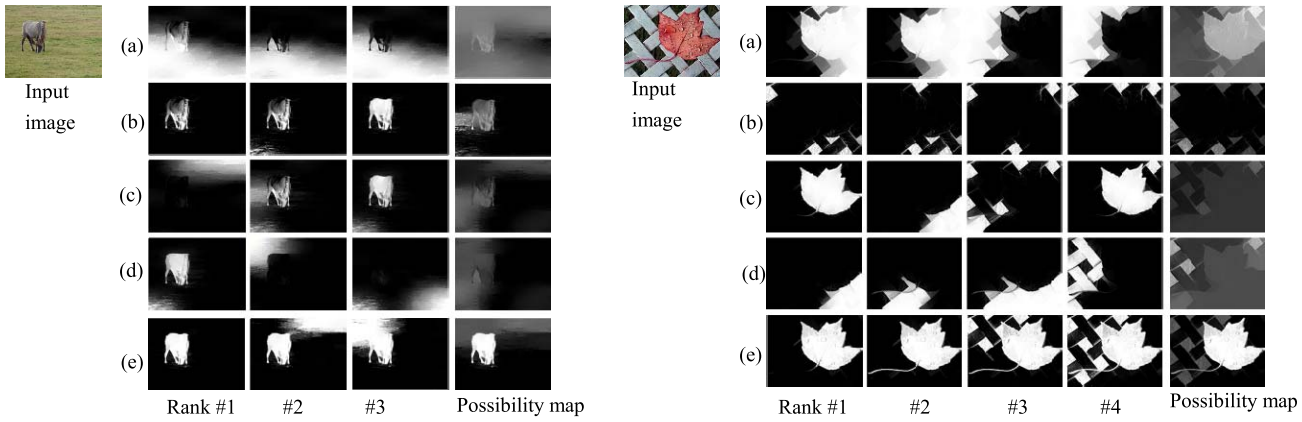


Fig. 9. Comparison of leading alpha mattes and the foreground possibility map: (a) spectral matting [3], (b) coarse-to-fine spectral matting [3], (c) color histogram based approach [22], (d) classification based approach [23], and (e) the proposed approach.

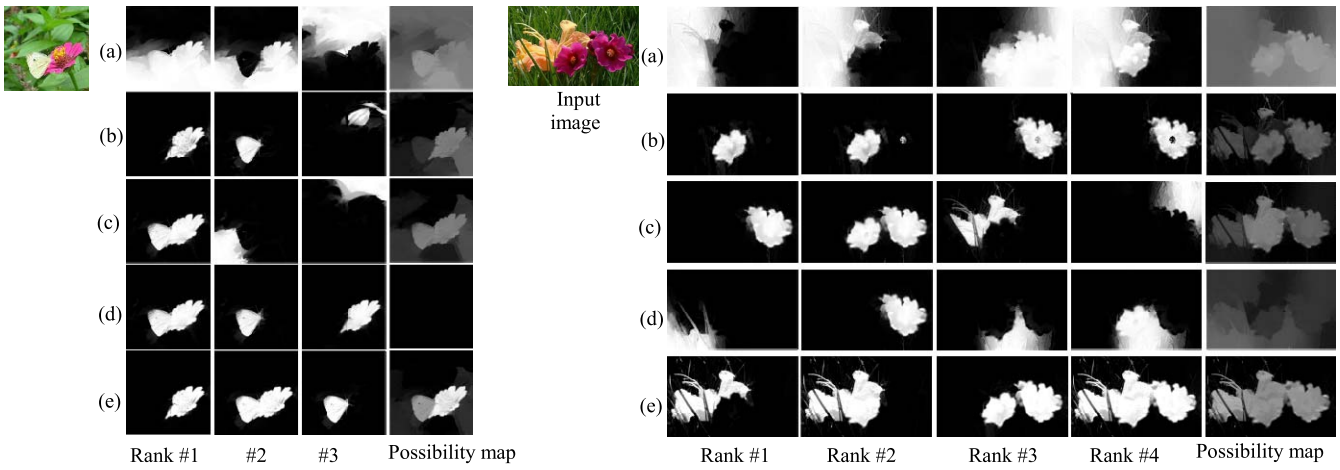


Fig. 10. Comparison of leading alpha mattes and the foreground possibility map: (a) spectral matting [3], (b) coarse-to-fine spectral matting [3], (c) color histogram based approach [22], (d) classification based approach [23], and (e) the proposed approach.

B. Evaluation of Foreground Analysis

To evaluate the performance of foreground estimation, the proposed method is compared with the spectral matting method in [3] and some modified approaches which have addressed foreground matte extraction. The modified approaches include the color histogram based approach in [22] and the classification based approach in [23]. In Fig. 9, we test two different cases. The horse image has a single foreground object in a simple background, while the leaf image has a single image in a cluttered background. A number of leading alpha mattes are demonstrated. It shows that due to the lack of the affinity information between spatially disjoint components, the spectral matting method [3] fails to produce consistent results. In the coarse-to-fine scheme, the performance has been improved by introducing far-range affinity information. However, it still fails to deal with cluttered scenes. The color histogram based approach in [22] has difficulty to deal with the regions where the foreground color and the background color are not well separated, like the horse image. On the other hand, the classification based approach in [23] fails to deal with the leaf image since this approach only relies on a simple background color model. In comparison, the proposed approach has demonstrated its capability in solving these problems and in generating satisfactory results. Based on the leading alpha mattes generated by these methods, we also present the corresponding possibility maps. Since the proposed approach generates more consistent matting components, the generated possibility map is far more reasonable than the others.

In Fig. 10, we demonstrate a comparison when dealing with multiple foreground objects. These conventional methods which aim to find the best single alpha matte by using a binary partition would have difficulty to handle this case. For multiple foreground objects, there could be several reasonable combinations of components to interpret the possible foreground. However, we can find there are several unreasonable results in the leading alpha mattes produced by conventional approaches. In comparison, our approach can generate more consistent foreground mattes based on the hierarchical graph model. In Fig. 11, we demonstrate more foreground estimation results, including some successful ones and some less successful ones. Basically, our approach can achieve quite satisfactory performance for various kinds of image contents.

Fig. 11. Input images (top row) and foreground possibility maps (bottom row).

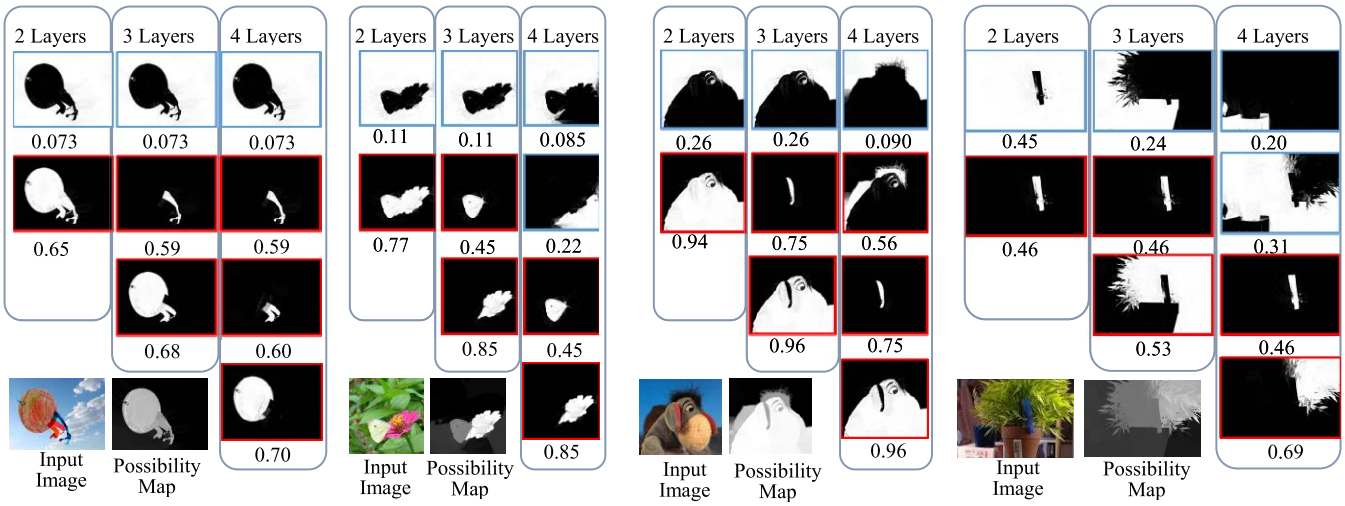


Fig. 12. Multiple component-to-layer mapping.

C. Multiple Matting Layers

Unlike conventional approaches which only address binary foreground/background layers, our approach provides a more general solution for multiple matting layers, as illustrated in Fig. 12. Moreover, any matting layer can be evaluated by a foreground factor and we can rank how likely a matting layer belongs to the foreground region. As shown in Fig. 12, there is an apparent gap in the values of the foreground factor. By setting a threshold over the foreground factor values, we can easily identify these foreground layers (framed in red).

A limitation of the proposed approach appears when dealing with objects that contain several distinct components. For example, the structures of the rightmost image (the plant) in Fig. 12 are somewhat complicated and it is difficult to obtain satisfactory results simply based on the color features. However, even though the proposed approach does not identify all parts of the foreground object, some salient parts of the foreground object have been extracted.

D. Qualitative Evaluation

The proposed hierarchical approach provides improvement in both matting quality and computational efficiency. Here, a quantitative evaluation is presented to assess the improvement. In this experiment, we focus on images which contain only one foreground layer and one background layer in order to focus mainly on the benefits of hierarchical decomposition. Table I shows the test images and the matting results. The test images and the ground truth are cropped from the image data in [7], which are originally provided to evaluate supervised matting algorithms. In this experiment, we apply unsupervised matting approaches to decompose each test image into two components and evaluate the corresponding foreground component with the ground truth. Table I shows a quantitative comparison in terms of the mean squared error (MSE) and running time. All algorithms are implemented in Matlab on an AMD FX6100 3.3 GHz CPU with 4GB of memory. It shows that the proposed method can efficiently obtain matting layers of reliable quality, even when the foreground color and background color are similar.

In Table II, we evaluate the performance of our approach without using the pixel-level contraction process. It shows that the contraction process can effectively reduce the number of cells and thus improve the computational efficiency. We also evaluate the matting performance without using multi-scale learning. The experiment shows that the use of multi-scale learning can greatly improve both accuracy and efficiency.

E. Complexity Evaluation

In our spectral matting experiments, we empirically fix the number of quantization intervals b_s and b_c to be 15 so that the number of cells P is roughly around 10k to 20k. For conventional spectral matting [3], the complexity of the Laplacian construction is $O(r^2N)$ and that of spectral analysis is $O(N^3)$, where N is the number of pixels. Additionally, both

TABLE I

TESTING IMAGE SETS, OUTPUT MATTES, MEAN SQUARED ERROR (MSE), AND RUNNING TIME OF THE ESTIMATION

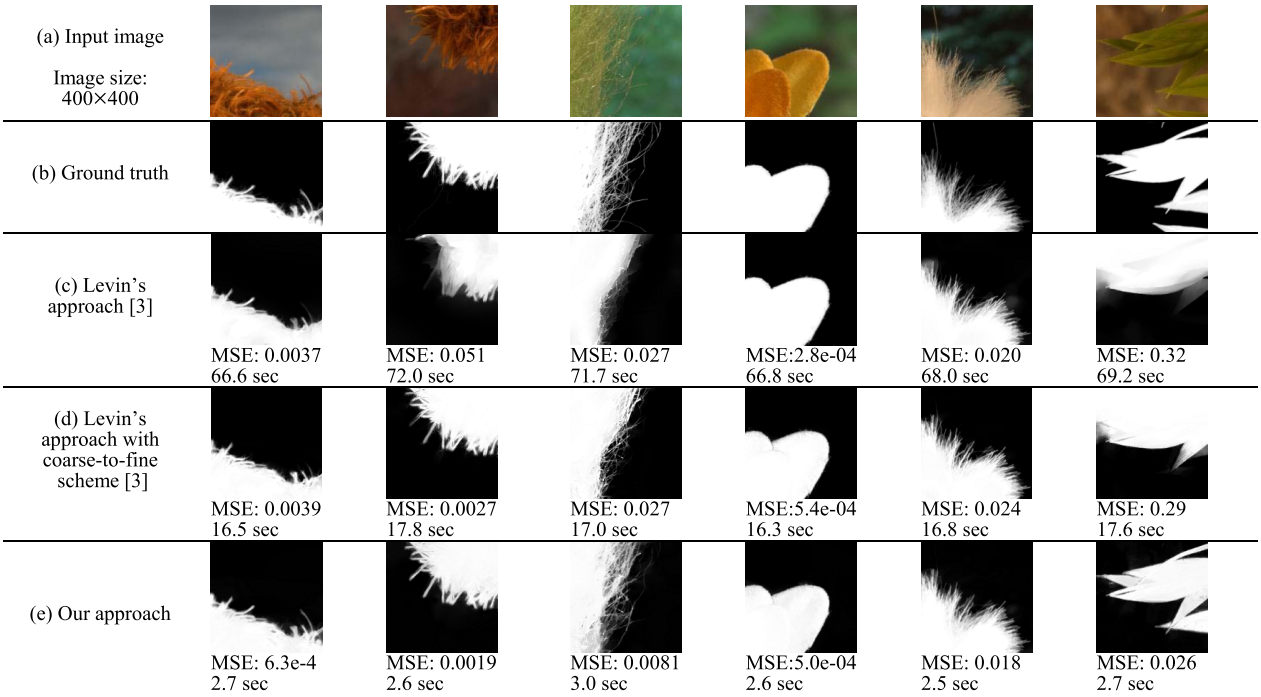


TABLE II

PERFORMANCE EVALUATION OF OUR APPROACH

Our approach	Measurements	#1	#2	#3	#4	#5	#6
With Contraction and Multi-scale Learning	P (number of cells)	5.3k	4.9k	7.4k	5.8k	4.9k	6.5k
	MSE (mean square error)	6.3e-4	0.0019	0.0081	5.0e-04	0.018	0.026
	Run time	2.7 sec	2.6 sec	3.0 sec	2.6 sec	2.5 sec	2.7 sec
Without Contraction	P (number of cells)	7.8k	7.2k	10.4k	8.1k	6.3k	8.9k
	MSE (mean square error)	6.2e-4	0.0021	0.0084	5.0e-04	0.019	0.026
	Run time	2.9 sec	2.9 sec	3.2 sec	2.9 sec	2.5 sec	3.0 sec
Without Multi-scale Learning	P (number of cells)	5.3k	4.9k	7.4k	5.8k	4.9k	6.5k
	MSE (mean square error)	0.010	0.011	0.12	3.8e-04	0.034	0.28
	Run time	27.7 sec	26.4 sec	27.5 sec	26.3 sec	26.9 sec	27.2 sec

TABLE III

SUMMARY OF NOTATIONS

Symbol	Description	Symbol	Description	Symbol	Description
α_i	alpha matte value at pixel i	$\mathbf{M}$	pixel-to-cell mapping function	S	number of leading eigenvectors
ω_q	local window centered at pixel q	Ω	set of total cells	$\overline{\mathbf{L}}$	cell-level Laplacian matrix
r	width of local window	Ω_q	mapped cells from ω_q	K	number of matting components
$\mathbf{I}_i$	r,g,b color vector at pixel i	N_q	number of cells in Ω_q	$\mathbf{c}^k$	binary vector clustering result of k^{th} cluster
$\boldsymbol{\mu}_q$	mean color vector in window ω_q	$\boldsymbol{\varphi}_i$	color feature vector of cell i	$\boldsymbol{\alpha}^k$	alpha matte of k^{th} component
ε	regularization parameter	$\boldsymbol{\beta}$, β_0	parameters of affine model	$\mathbf{h}^k$	color distribution of the component k
N	total number of image pixels	$\boldsymbol{\Phi}_q$	feature matrix in Ω_q	Z	number of color bins
x_i, y_i	original spatial coordinates of pixel i	c_β	parameter for regularization	$\mathbf{L}_{KL}$	divergence-based Laplacian
$\widetilde{x}_i$, $\widetilde{y}_i$	spatial coordinates after contraction	$\mathbf{W}_q$	local transformation matrix	$\mathbf{A}_{KL}$	divergence-based affinity matrix
$\mathbf{x}$, $\mathbf{y}$	vector of spatial coordinates	$\boldsymbol{\Gamma}_q$	local graph	$\mathbf{D}_{KL}$	divergence-based degree matrix
$\widetilde{\mathbf{x}}$, $\widetilde{\mathbf{y}}$	vector of contracted coordinates	$\mathbf{L}_q$	local graph Laplacian	$\mathbf{L}_{comp}$	component-level matting Laplacian
λ_x and λ_y	regularization parameters of the contraction process	$\mathbf{A}_q$	local affinity matrix	$\boldsymbol{\alpha}^q_{layer}$	alpha matte of layer q
		$\mathbf{D}_q$	local degree matrix		
$\mathbf{L}$	graph Laplacian	l_q	level index	$\mathbf{d}^q$	binary combination vector of layer q
b_s	number of spatial discretization intervals	$\boldsymbol{\alpha}_q$	vector of alpha matte values of the Nq cells in Ω_q	$\mathbf{o}_i^{pixel}$	interpolated alpha value of pixel i
b_c	number of color discretization intervals	d_s	down-sampling rate	$\mathbf{B}$	component-level combination vector
		P	total number of cells	$\mathfrak{I}_f$	a set of foreground vectors

the approaches in [22] and [23] take spectral matting as the core process in their algorithms. Since the computational complexity of the remaining operations in these two algorithms is relatively small, these two algorithms basically have the $O(N^3)$ computational complexity. On the other hand, for the proposed approach, the complexity of the construction of the cell-level

Laplacian is $O(|S_{patch}|)$ where $|S_{patch}|$ denotes the number of multi-resolution patches. In general, $|S_{patch}|$ is around 20k, which is much smaller than N. Besides, the complexity of the cell-level spectral analysis is $O(P^3)$, and the complexity of the cell-to-pixel mapping is $O(N)$. Hence, the complexity of the cell-based computations is $O(P^3 + N)$, which is much efficient than the conventional complexity $O(N^3)$. Table III is the summary of notations.

IV. Limitations

Currently, our method uses color features only. This makes it difficult to separate objects with very similar color appearance or to merge components with very different color appearance. Compared to local decision, the inclusion of global information allows us to better discriminate objects of similar color appearance with the leverage of relative similarity measure. Taking the shadow problem as an example, if there are some other relatively dissimilar objects around, they would be a support for the merging of the shadowed region with the unshadowed region. However, the use of global information may still get confused when dealing with objects with extremely similar color appearance. In the future, we will discuss the inclusion of some extra features, like textures or learning-based features, to better distinguish objects with similar color appearance.

V. Conclusion

This paper presents an efficient and effective hierarchical framework for unsupervised matting. This approach provide a solution with multiple matting layers to interpret an image containing more than one single foreground matte. Besides, a probability based approach is present to evaluate the foreground possibility for matting layers. To enhance the consistency of foreground layers, a multi-scale graph learning scheme is presented. Experimental results show that this approach can greatly improve the performance for unsupervised matting. Quantitative evaluation also shows that that our approach is superior to state-of-the-art techniques in both accuracy and efficiency.

References

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

[2] D. Singaraju and R. Vidal, "Estimation of alpha mattes for multiple image layers," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 7, pp. 1295–1309, Jul. 2011.

[3] A. Levin, A. Rav-Acha, and D. Lischinski, "Spectral matting," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 10, pp. 1699–1712, Oct. 2008.

[4] S. Paris and F. Durand, "A fast approximation of the bilateral filter using a signal processing approach," in *Proc. 9th ECCV*, 2006, pp. 568–580.

[5] J. Chen, S. Paris, and F. Durand, "Real-time edge-aware image processing with the bilateral grid," *ACM Trans. Graph.*, vol. 26, no. 3, p. 103, 2007.

[6] K. He, J. Sun, and X. Tang, "Fast matting using large kernel matting laplacian matrices," in *Proc. IEEE CVPR*, Jun. 2010, pp. 2165–2172.

[7] (2009). *Alpha Matting Evaluation* [Online]. Available: http://www.alphamatting.com

[8] (2009). *Database Saliency and Image Summaries* [Online]. Available: http://ivrg.epfl.ch/supplementary_material/RK_SIGGRAPH_Asia09/index.html

[9] C. Rhemann, C. Rother, and M. Gelautz, "Improving color modeling for alpha matting," in *Proc. BMVC*, 2008.

[10] E. S. L. Gastal and M. M. Oliveira, "Shared sampling for real-time alpha matting," *Eurographics*, vol. 29, no. 2, pp. 575–584, 2010.

[11] U. von Luxburg, "A tutorial on spectral clustering," Max Plank Inst. Biol. Cybern., Tübingen, Germany, Tech. Rep. 149, 2006.

[12] M. Huang, F. Liu, and E. Wu, "A GPU-based matting Laplacian solver for high resolution image matting," *J. Vis. Comput.*, vol. 26, nos. 6–8, pp. 943–950, 2010.

[13] (2010). *Fast Matting Using Large Kernel Matting Laplacian Matrices* [Online]. Available: http://research.microsoft.com/enus/um/people/kahe/cvpr10/index.htm

[14] C. Rhemann, C. Rother, J. Wang, M. Gelautz, P. Kohli, and P. Rott, "A perceptually motivated online benchmark for image matting," in *Proc. IEEE CVPR*, Jun. 2009, pp. 1826–1833.

[15] (2007). *Anat Levin's* [Online]. Available: http://www.wisdom.weizmann.ac.il/~levina/

[16] D. Singaraju, C. Rother, and C. Rhemann, "New appearance models for natural image matting," in *Proc. IEEE CVPR*, Jun. 2009, pp. 659–666.

[17] Y. Zheng and C. Kambhamettu, "Learning based digital matting," in *Proc. 12th Int. Conf. Comput. Vis.*, Sep./Oct. 2009, pp. 889–896.

[18] P. Lee and Y. Wu, "Nonlocal matting," in *Proc. IEEE CVPR*, Jun. 2011, pp. 2193–2200.

[19] Y. Weiss, "Segmentation using eigenvectors: A unifying view," in *Proc. 7th Int. Conf. Comput. Vis.*, 1999, pp. 975–982.

[20] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 888–905, Aug. 2000.

[21] A. Ng, M. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Advances in Neural Information Processing Systems*. Cambridge, MA, USA: MIT Press, 2001.

[22] J.-Z. Wang and C.-H. Li, "Spectral matting based on color information of matting components," in *Advances in Wireless Networks and Information Systems*, vol. 72. Berlin, Germany: Springer-Verlag, 2010, pp. 119–130.

[23] W.-C. Hu, J.-J. Jhu, and C.-P. Lin, "Unsupervised and reliable image matting based on modified spectral matting," *J. Vis. Commun. Image Represent.*, vol. 23, no. 4, pp. 665–676, 2012.

[24] M. Eisemann, J. Wolf, and M. Magnor, "Spectral video matting," in *Proc. VMV*, 2009, pp. 121–126.

[25] B. Bollobás, *Modern Graph Theory*. New York, NY, USA: Springer-Verlag, 1998.

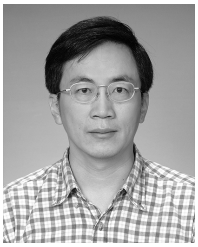
[26] A. Brouwer and W. Haemers, *Spectra of Graphs*. New York, NY, USA: Springer-Verlag, 2012.

[27] C.-Y. Tseng and S.-J. Wang, "Maximum-a-posteriori estimation for global spatial coherence recovery based on matting Laplacian," in *Proc. 19th IEEE ICIP*, Sep./Oct. 2012, pp. 293–296.

[28] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed. New York, NY, USA: Springer-Verlag, 2006.

Chen-Yu Tseng received the B.S. and M.S. degrees in electrical engineering from the National Chiao Tung University (NCTU), Hsinchu, Taiwan, in 2005 and 2007, respectively.

He is currently pursuing the Ph.D. degree in electrical engineering at NCTU. His research interests are image processing and image analysis.



Sheng-Jyh Wang (M'95) received the B.S. degree in electronics engineering from National Chiao Tung University (NCTU), Hsinchu, Taiwan, in 1984, and the M.S. and Ph.D. degrees in electrical engineering from Stanford University, Stanford, CA, USA, in 1990 and 1995, respectively.

He is currently a Professor with the Department of Electronics Engineering, NCTU. His research interests are image processing, video processing, and image analysis.