Image Matting via LLE/iLLE Manifold Learning

David Tien, Member, IEEE, Junbin Gao, and Jim Tulip

***Abstract*—Accurately extracting foreground objects is the problem of isolating the foreground in images and video, called image matting which has wide applications in digital photography. This problem is severely ill-posed in the sense that, at each pixel, one must estimate the foreground and background pixels and the so-called alpha value from only pixel information. The most recent work in natural image matting rely on local smoothness assumptions about foreground and background colours on which a cost function has been established. In this paper, we propose an extension to the class of affinity based matting techniques by incorporating local manifold structural information to produce both a smoother matte based on the so-called improved Locally Linear Embedding. We illustrate our new algorithm using the standard benchmark images and very comparable results have been obtained.**

***Index Terms*—Image Matting, Locally Linear Embedding, Manifold Learning, Alpha Matte, Nesterov's Method**

I. Introduction

Image matting is the problem of isolating the foreground in a single image. These kind of problems are important in both computer vision and graphics applications as well image/video editing. There is a large amount of research interest in the field and a variety of techniques that have been developed to solve this problem. We refer readers to a 2007 survey article [1] in which a comprehensive review of existing image and video matting algorithms and systems was presented with an emphasis on the advanced techniques that have been recently proposed.

Most common techniques rely on the so-called alpha matte model. The model specifies the image composition process from alpha matte, mathematically defined as follows,

$$\mathbf{I}_i = \alpha_i \mathbf{F}_i + (1 - \alpha_i)\mathbf{B}_i \tag{1}$$

where $\mathbf{I}_i$, α_i, $\mathbf{F}_i$ and $\mathbf{B}_i$ are image colour value, the alpha matte value, the foreground image colour value and the background colour value, respectively, at a given pixel i. We assume that the alpha matte value α_i lies in the range between 0 (for true background pixels) and 1 (for true foreground pixels). This kind of alpha matte are called soft matte, while the hard matte values take either 0 or 1 to clearly distinguish foreground and background pixels.

Matting requires that the α_i, $\mathbf{F}_i$ and $\mathbf{B}_i$ are solved simultaneously with the only available image information $\mathbf{I}_i$. Obviously this is a severely ill-posed problem. In practice, matting requires user interaction to produce acceptable results. One of common approaches relies on the user supplying a trimap, which is an input image which labels some pixels as definite foreground, definite background and some unknown regions. Usually the unknown regions are around the boundary of the objects to be extracted. To infer the alpha matte value in the regions, Bayesian matting [2] uses a progressively moving window marching inward from the known regions. Bai and Sapiro [3] proposed calculating alpha matte through the geodesic distance from a pixel to the known regions. Among the propagation-based approaches, the Poisson matting algorithm [4] assumes the foreground and background colours are smooth in a narrow band of unknown pixels, then solves a homogenous Laplacian matrix; A similar algorithm is proposed in [5] based on Random Walks.

In the closed-form matting approach [6], Levin *et al.* exploited learning approach to fit a linear model for foreground and background colours in each local window, thus globally a matting Laplacian matrix can be formulated to impose a natural smoothness constraint and finally a quadratic cost function over alpha matte can be minimized to solve the matting problem. As minimizing a quadratic cost function is equivalent to solving a linear system, a closed-form solution can be achieved. One of benefits offered by this approach is that user input can be reduced to several scribbles because of smoothness specified by the Laplacian matrix in the technique. When the image size is large, the resulting linear system is huge and it is time-consuming to solve. In a recent work [7], He et al. proposed a fast matting scheme by using large kernel matting Laplacian matrices. Although the closed-form matting takes care of global smoothness of entire matte, it pays less attention to edge discontinuity. In our recent research we introduced zero-one penalty [8] and total variation regularisation [9] to enforce edge preservation. This has added better improvement over the closed-form matting and other similar learning-based approaches.

A similar approach [10] has been proposed by making use of the Local Tangent Space Alignment (LTSA) [11]. LTSA, as a manifold learning approach, is an unsupervised learning algorithm that computes low-dimensional, neighbourhood-preserving embeddings of high-dimensional inputs. The argument used in [10] is to apply the LTSA algorithm to capture local linear structure among the colour information over a local window and link that to the matting value space. This kind of observation is intuitively vert encouraging for manifold learning methods to be successful in image matting.

In this paper, we consider using traditional manifold learning algorithms [12], [13] such as Locally Linear Embedding (LLE) [14] and its improved version iLLE [15] to solve the

David Tien is with School of Computing and Mathematics, Charles Sturt University, Bathurst, NSW 2795, Australia; email: dtien@csu.edu.au

Junbin Gao is with School of Computing and Mathematics, Charles Sturt University, Bathurst, NSW 2795, Australia; email: dtien@csu.edu.au

Jim Tulip is with School of Computing and Mathematics, Charles Sturt University, Bathurst, NSW 2795, Australia; email: jtulip@csu.edu.au

  DOI:

problem of image matting. As has been done in [6], we still use the so-called user-specified constraints (such as scribbles or a bounding rectangle). Similar approaches can also be found in [6], [16], [17]. In deriving the matting Laplacian matrices [6], the authors have used the assumption that the foreground (or background) colours in a local window lie on a single line in the RBG colour space. This assumption is going to be naturally replaced with the smooth manifold assumption which is purely a mathematical assumption in learning a smooth manifold. Hence, a locally linear learning model such as LLE can be utilized to capture such linear information which in turn assists matte information learning. One of innovations offered by our approach is to take a natural definition of pixel neighborhood on the colour manifold as the neighborhood used in most standard manifold learning algorithms such as LLE. It is hoped that the idea of using manifold learning or a dimensionality reduction approach for image matting as offered in this paper will inspire more exploration in this direction.

The paper is organized as follows. Section II simply introduces the Locally Linear Embedding (LLE) [13], [14] and iLLE [15] manifold learning algorithms. Section III is dedicated to formulating the new algorithms based on LLE. In Section IV, we present several examples of using the new image matting approach over the benchmark images, see [6]. We make our conclusions in Section V with suggestions for further exploring the application of dimensionality reduction algorithms in image matting.

II. LLE/iLLE Alignment Matrices

A. LLE and Its Alignment Matrix

The locally linear embedding (LLE) algorithm has been recently proposed as a powerful eigenvector method for the problem of nonlinear dimensionality reduction [14]. In the last decade, many LLE based algorithms have been developed and introduced in machine learning community: kernelized LLE (KLLE) [18], Laplacian Eigenmap (LEM) [19], Hessian LLE (HLLE) [20], robust LLE [21], weighted LLE [22] enhanced LLE [23] and supervised LLE [24].

By exploiting the local symmetries of linear reconstructions, LLE is able to learn the global structure of nonlinear manifolds, such as those generated by images of faces, or documents of text. The LLE method is also widely used for data visualization [14], classification [22], [24] and fault detection [25].

Because of the assumption that local patches are linear, that is, each of them can be approximated by a linear hyperplane, each data point can be represented by a weighted linear combination of its nearest neighbours (different ways can be used in defining the neighbourhood). Coefficients of this approximation characterize local geometries in a highdimensional space, and they are then used to find low-dimensional embeddings preserving the geometries in a low-dimensional space. The main point in replacing the nonlinear manifold with the linear hyperplanes is that this operation does not bring significant error, because, when locally analyzed, the curvature of the manifold is not large, i.e., the manifold can be considered to be locally flat. The result of LLE is a single global coordinate system.

The argument that we take for using the LLE for image matting is that, like the application of the Laplacian alignment matrix used in [6], local linear structure among the colour information over a local window can be learnt and transferred to the matting value space. This observation suggests intuitively that the LLE alignment matrix could be successful in image matting.

In the following sections, we still use $X_i = \{\mathbf{I}_{i_j} | i_j \in w_i, j = 1, ..., K\}$ to denote the subset of colour vectors over a local window pixels of pixel i. Note that the pixel $\mathbf{I}_i$ is contained in X_i. Under the LLE assumption, the colour vector $\mathbf{I}_i$ at pixel i can be approximated by a linear combination β_{ij} (the so-called reconstruction weights) of its $K-1$ nearest neighbours $X_i \backslash \mathbf{I}_i$. Hence, LLE fits a hyperplane through $\mathbf{I}_i$ and its nearest neighbors in the colour manifold defined over the image pixels. The fitting is achieved by solving the following optimal problem

$$\min_W \sum_i \|\mathbf{I}_i - \sum_{j=1}^{K} \beta_{ij} \mathbf{I}_{i_j}\|^2$$

under the condition $\sum_{j=1}^{K} \beta_{ij} = 1$. For the sake of simplicity, we assume that $\beta_{ij} = 0$ when $\mathbf{I}_{i_j} = \mathbf{I}_i$. Assuming that the manifold is locally linear, the local linearity may be preserved in the space of matting values. Once the weights W have been determined, the matting values $\boldsymbol{\alpha}$ can be determined by minimizing the following objective function

$$\min_{\boldsymbol{\alpha}} F(\boldsymbol{\alpha}) = \sum_i \|\alpha_i - \sum_{j=1}^{K} \beta_{ij} \alpha_{i_j}\|^2 = \boldsymbol{\alpha}^T R_{\text{LLE}} \boldsymbol{\alpha} \qquad (2)$$

where $R_{\text{LLE}} = (E - W)^T (E - W)$ is called LLE alignment matrix and E is the identity matrix.

In the standard LLE algorithm, (4) is usually solved with additional constrained conditions like $\|\boldsymbol{\alpha}\|_2 = 1$ which results in an eigenvector problem. Instead of such standard constraints, in image matting we would like to formulate a matting solution by including appropriate constraints. We will discuss this issue in the next section.

B. iLLE and Its Alignment Matrix

Recently Xiang et al [15] re-explained the LLE as well as LTSA (Local Tangent Space Alignment) [11] under the framework of local linear transformation, and a close link between LLE and LSTA has been established. Based on this, Xiang *et al*. [15] proposed a new model of improved LLE (iLLE).

We still use $X_i = \{\mathbf{I}_{i_j} | i_j \in w_i, j = 1, ..., K\}$ to denote the subset of colour vectors over a local window of pixel i. Note that the pixel $\mathbf{I}_i$ is contained in X_i. In LLE, a linear regression was established to regress $\mathbf{I}_i$ by the rest of pixels $X_i \backslash \mathbf{I}_i$ in X_i. In iLLE, we can construct K linear regressions. That is, we can linearly approximate each pixel $\mathbf{I}_{i_j}$ with the rest of pixels

  DOI:

in X_i. Specifically, we have

$$\mathbf{I}_{i_j} = \sum_{l \neq j} \beta_{i,j}^{(l)} \mathbf{I}_{i_l} + \epsilon_{i_j}, \quad j = 1, 2, ..., K$$

Denote $\boldsymbol{\beta}_{i,j} = [\beta_{i,j}^{(1)}, ..., \beta_{i,j}^{(j-1)}, \beta_{i,j}^{(j+1)}, ..., \beta_{i,j}^{(K)}]^T$ a $(K-1)$-dimensional vector. Then the weight vector $\boldsymbol{\beta}_{i,j}$ can be solved for each j respectively by

$$\min_{\boldsymbol{\beta}_{i,j}} \|\mathbf{I}_{i_j} - \sum_{l \neq j} \beta_{i,j}^{(l)} \mathbf{I}_{i_l}\|^2 + \lambda \|\boldsymbol{\beta}_{i,j}\|_2^2$$

under the condition $\sum_{l \neq j}^{K-1} \beta_{i,j}^{(l)} = 1$. A closed-form solution for the weight vector $\boldsymbol{\beta}_{i,j}$ can be given, please refer to [15] for more details. For each $j = 1, ..., K$, let define an extended K-dimensional vector by $\boldsymbol{\beta}_j^{(i)} = [-\beta_{i,j}^{(1)}, ..., -\beta_{i,j}^{(j-1)}, 1, -\beta_{i,j}^{(j+1)}, ..., -\beta_{i,j}^{(K)}]^T$, i.e., we insert 1 at the location j in $\boldsymbol{\beta}_{i,j}$ and negate all its components. Then define K small matrice by

$$C_j^{(i)} = \boldsymbol{\beta}_j^{(i)} \boldsymbol{\beta}_j^{(i)^T}, \quad \text{for } j = 1, 2, ..., K$$

Now consider the matte value space, i.e., all the matte values $\boldsymbol{\alpha} = [\alpha_1, ..., \alpha_N]^T$ over all N pixels. In the local window w_i at pixel i, we wish to find the suitable matte value $\boldsymbol{\alpha}_i = [\alpha_{i_1}, ..., \alpha_{i_K}]^T$ such that under the learned locally linear structures the following error is as small as possible

$$\sum_{j=1}^{K} \|\alpha_{i_j} - \sum_{l \neq j} \beta_{i,j}^{(l)} \alpha_{i_l}\|^2 = \boldsymbol{\alpha}_i C_i \boldsymbol{\alpha}_i^T \tag{3}$$

where $C_i = \sum_{j=1}^{K} C_j^{(i)}$ is a $K \times K$ matrix.

Optimization (3) can be further extended to the entire matte value space. This is to ask

$$\min_{\boldsymbol{\alpha}} F(\boldsymbol{\alpha}) = \sum_i \boldsymbol{\alpha}_i C_i \boldsymbol{\alpha}_i^T = \boldsymbol{\alpha}^T R_{\text{iLLE}} \boldsymbol{\alpha} \tag{4}$$

where R_{iLLE} is called iLLE alignment matrix and it can be constructed by $R_{\text{iLLE}} = \sum_i S_i C_i S_i^T$ where S_i is a column selection matrix such that only columns $\{i_1, i_2, ..., i_K\}$ are chosen.

The improved LLE can be considered as a collective version of the standard LLE [14] by taking a turn for each pixel to be linearly approximated by the rest pixels in a local window. This way specifies more constrains over the matte value space. We can demonstrate that the matting based on iLLE has some significant improvement over the matting with the conventional LLE.

Xiang et al [15] also proved that the improved LLE alignment matrix can be constructed according to the local linear transformation from the image color space to the matte value space. Thus the matting algorithm based on iLLE is closely related to the Laplacian alignment matting algorithm [6]. In the construction of the Laplacian alignment matrix, the linear transformation from the color space to the matte value space is constructed based on all the color information in a local window of one pixel to predict the signal matte value corresponding to the pixel, while the iLLE alignment matrix is based K local linear transformations each of which is built on color information over the rest of pixels to predict the matte value at a particular pixel in turn.

III. Algorithm Formulation

A. Formulation

In terms of machine learning terminology, the image matting is a unsupervised learning problem. As we pointed out in the introduction, one of major approaches in image matting is to use the so-called trimap or user-scribbles. The information given in a trimap or from user-scribbles is that for a group of pixels over an image, their alpha matte are known.

For this purpose, let Ω be a subset of pixels and Γ_Ω be the operator mapping alpha matte $\boldsymbol{\alpha}$ to the given alpha matting values $\boldsymbol{\alpha}_0$ (trimap and user-scribbles) over Ω, then the matting problem is to minimize the following objective function with respect to $\boldsymbol{\alpha}$ such that $\Gamma_\Omega(\boldsymbol{\alpha}) = \boldsymbol{\alpha}_0$

$$\min_{\Gamma_\Omega(\boldsymbol{\alpha}) = \boldsymbol{\alpha}_0} F(\boldsymbol{\alpha}) = \boldsymbol{\alpha}^T R \boldsymbol{\alpha}.$$

where R is either the LLE alignment matrix or the iLLE alignment matrix.

To get a smoother alpha matte $\boldsymbol{\alpha}$, we propose to formulate two optimization problems for our image matting

1) Scribble Smoothing [6]: Instead of using hard scribble constraints, we seek for $\boldsymbol{\alpha}$ by smoothing the matting value using

$$\min_{\boldsymbol{\alpha}} \boldsymbol{\alpha} R \boldsymbol{\alpha}^T + \lambda (\boldsymbol{\alpha} - \boldsymbol{\alpha}_\Omega) D_\Omega (\boldsymbol{\alpha}^T - \boldsymbol{\alpha}_\Omega^T) \tag{5}$$

where λ is some large regularizer, D_Ω is a diagonal matrix whose diagonal elements are one for constrained pixels (in trimap or user-scribbles) and zero for all other pixels, and $\boldsymbol{\alpha}_\Omega$ is the vector containing the specified alpha values for the constrained pixels and zero for all other pixels.

2) Constrained-Scribble Smoothing: In this formulation, we will constrain $\boldsymbol{\alpha}$ to its soft range $0 \leq \alpha_i \leq 1$

$$\min_{0 \preceq \boldsymbol{\alpha} \preceq 1} \boldsymbol{\alpha} R \boldsymbol{\alpha}^T + \lambda (\boldsymbol{\alpha} - \boldsymbol{\alpha}_\Omega) D_\Omega (\boldsymbol{\alpha}^T - \boldsymbol{\alpha}_\Omega^T) \tag{6}$$

where $0 \preceq \boldsymbol{\alpha} \preceq 1$ means the elements of the vector $\boldsymbol{\alpha}$ are between 0 and 1.

B. Algorithms

It is easy to solve the optimization problem (5) for the scribble smoothing as it is an unconstrained quadratic programming problem. To reenforce the user scribble conditions $\boldsymbol{\alpha}_\Omega$ over pixels Ω, the algorithm in [6] takes a larger regularizer such as $\lambda = 100$. Thus the objective function can be optimized by solving a sparse linear system:

$$(R + \lambda D_\Omega) \boldsymbol{\alpha} = \lambda \boldsymbol{\alpha}_\Omega. \tag{7}$$

However solving the optimization problem defined in (6) is much harder as it is a quadratic programming problem with a set of linear inequality constraints. One of approaches to solve a quadratic programming problem is to use an interior

  DOI:

point method that uses Newton-like iterations to find a solution of the Karush-Kuhn-Tucker conditions of the primal and dual problems. The computational complexity is polynomial time of the size of the matrix R. Even for a moderately sized image, it is intractable for one to use a standard quadratic optimization algorithm as the number of variables is equal to the number of pixels in the whole image. Fortunately given the special form of the constraints involved in the problem, a trust-region alike algorithm as proposed in [26] can be used for the optimization problem defined in (6). The algorithm has a fast convergence rate of $O(1/k^2)$ where k is the number of iterations. However, the trust-region alike algorithm is very expensive because the second order derivative of the objective function is needed.

However we also note that the problem is convex and smooth. In the following sections, we propose applying the optimal first-order black-box method for smooth convex optimization, i.e., Nesterov's method [27]–[29], to achieve a convergence rate of $O(1/k^2)$. We first construct the following model for approximating the objective function $F(\boldsymbol{\alpha})$ in (6) at the point $\boldsymbol{\alpha}$,

$$h_{C,\boldsymbol{\alpha}}(\widetilde{\boldsymbol{\alpha}}) = F(\boldsymbol{\alpha}) + F'(\boldsymbol{\alpha})(\widetilde{\boldsymbol{\alpha}} - \boldsymbol{\alpha})^T + \frac{C}{2}\|\widetilde{\boldsymbol{\alpha}} - \boldsymbol{\alpha}\|^2, \quad (8)$$

where $C > 0$ is a constant.

With model (8), Nesterov's method is based on two sequences $\{\boldsymbol{\alpha}_k\}$ and $\{s_k\}$ in which $\{\boldsymbol{\alpha}_k\}$ is the sequence of approximate solutions while $\{s_k\}$ is the sequence of search points. The search point s_k is the convex linear combination of $\boldsymbol{\alpha}_{k-1}$ and $\boldsymbol{\alpha}_k$ as

$$s_k = \boldsymbol{\alpha}_k + \beta_k(\boldsymbol{\alpha}_k - \boldsymbol{\alpha}_{k-1})$$

where β_k is a properly chosen coefficient. The approximate solution $\boldsymbol{\alpha}_{k+1}$ is computed as the minimizer of $h_{C_k,s_k}(\widetilde{\boldsymbol{\alpha}})$. It can be proved that

$$\boldsymbol{\alpha}_{k+1} = \underset{\mathbf{0}\preceq\widetilde{\boldsymbol{\alpha}}\preceq\mathbf{1}}{\operatorname{argmin}} \frac{C_k}{2}\left\|\widetilde{\boldsymbol{\alpha}} - \left(s_k - \frac{1}{C_k}F'(s_k)\right)\right\|^2 \quad (9)$$

where C_k is determined by line search according to the Armijo-Goldstein rule so that C_k should be appropriate for s_k, (see [29]). $\boldsymbol{\alpha}_{k+1}$ defined by (9) is actually the projection of the vector $s_k - \frac{1}{C_k}F'(s_k)$ over the convex set $\{\widetilde{\boldsymbol{\alpha}} | \mathbf{0} \preceq \widetilde{\boldsymbol{\alpha}} \preceq \mathbf{1}\}$. We can easily work out the projection given by the following formula

$$\boldsymbol{\alpha}_{k+1} = \max\left\{0, \min\left\{1, s_k - \frac{1}{C_k}F'(s_k)\right\}\right\}. \quad (10)$$

where both max and min operate over vectors componentwisely as the same meaning in Matlab.

An efficient algorithm for (6) is summarized in Algorithm 1.

C. Reconstruction of Foreground and Background Images

After solving for the alpha values $\boldsymbol{\alpha}$, we need to reconstruct foreground $\mathbf{F}$ and background $\mathbf{B}$. For this purpose, we take the same strategy as [6] to reconstruct $\mathbf{F}$ and $\mathbf{B}$ by using the composition equation (1) with certain smoothness priors on

Algorithm 1 The Efficient Nesterov's Algorithm

Input: $C_0 > 0$ and $\boldsymbol{\alpha}_0$, K

Output: $\boldsymbol{\alpha}_{k+1}$

1: Initialize $\boldsymbol{\alpha}_1 = \boldsymbol{\alpha}_0$, $\gamma_{-1} = 0, \gamma_0 = 1$ and $C = C_0$.
2: **for** $k = 1$ to K **do**
3: Set $\beta_k = \frac{\gamma_{k-2}-1}{\gamma_{k-1}}$, $s_k = \boldsymbol{\alpha}_k + \beta_k(\boldsymbol{\alpha}_k - \boldsymbol{\alpha}_{k-1})$
4: Find the smallest $C = C_{k-1}, 2C_{k-1}, \ldots$ such that

$$F(\boldsymbol{\alpha}_{k+1}) \le h_{C,s_k}(\boldsymbol{\alpha}_{k+1}),$$

where $\boldsymbol{\alpha}_{k+1}$ is defined by (10).
5: Set $C_k = C$ and $\gamma_{k+1} = \frac{1+\sqrt{1+4\gamma_k^2}}{2}$
6: **end for**

Fig. 1. Images and Strokes

both $\mathbf{F}$ and $\mathbf{B}$. $\mathbf{F}$ and $\mathbf{B}$ are obtained from optimizing the following objective function

$$\min_{\mathbf{F},\mathbf{B}} \sum_i \|\alpha_i\mathbf{F}_i + (1-\alpha_i)\mathbf{B}_i - \mathbf{I}_i\|^2 + \langle \partial\alpha_i, (\partial\mathbf{F}_i)^2 + (\partial\mathbf{B}_i)^2\rangle$$

where ∂ is the gradient operator over the image grid. For a fixed A, the problem is quadratic and its minimum can be found by solving a set of linear equations.

IV. Experiment Results

All the algorithms are implemented using MATLAB on a small workstation machine with 32G memory. In all the calculations, we set $\lambda = 100$, see (7), in both algorithms.

A. Experiment I

In this experiment, we aim to compare the performance of our newly suggested LLE alignment matrix with the Laplacian alignment matrix introduced in [6] for image matting, under the Scribble Smoothing formulation. Similar to the closed-form solution, we solve (7) for matte values.

For this experiment, two images from the original paper describing the closed form solution for image matting [6] are used for tests. Figure 1 presents the images and their stroked images as used in the algorithm comparison.

Figure 2 presents matting results on the two images, respectively. In Figure 2, the first and third columns show the results from the closed-form solution while the second row and the fourth row show the results from the Scribble Smoothing formulation based on the LLE alignment matrix. The results given by the LLE alignment matrix look visually comparable to the results in [6].

  DOI:

Fig. 2. Mattes from strokes with the Closed-form solution for the Laplacian alignment matrix (the first and third columns) and the Scribble Smoothing for the LLE alignment matrix (the second and fourth columns): (a) Learned Masks (the first row). (b) Reconstructed foreground images (the second row). (c) Extracted background images (the third row).

B. Experiment II

In this experiment, we compare the performance of iLLE alignment matrix against the LLE alignment matrix based on two algorithm formulations. All the setting used in this experiment is exactly same as those used in Experiment I. Image and its scribbles used in this experiment are shown in Figure

Fig. 3. Images and Strokes

Figure 4 presents matting results on the plant image. In Figure 4, the first and second columns show the results from the Scribble Smoothing for the LLE and iLLE alignment matrices, respectively; and the third and fourth columns show that from the Constrained-Scribble Smoothing for both LLE and iLLE alignment matrices, respectively. The results given by the iLLE alignment matrix is slightly better than those given by the LLE alignment matrix.

C. Experiment III

In this experiment, the primary goal is to assess the performance of the two formulations: Scribble Smoothing (SS) and Constrained-Scribble Smoothing (CSS) presented in this paper. We use iLLE alignment matrix as an example. For this purpose, we use benchmark test images taken from the matting website http://www.alphamatting.com. On the website, there are four types of test images. The images that we are using are (A) `plastic bag` (Highly Transparent); (B) `net` (Strongly Transparent); (C) `elephant` (Medium Transparent) and (D) `peacock` (Little Transparent but with detailed structures). The four original images and their strokes used in the experiment are displayed in Figure 5.

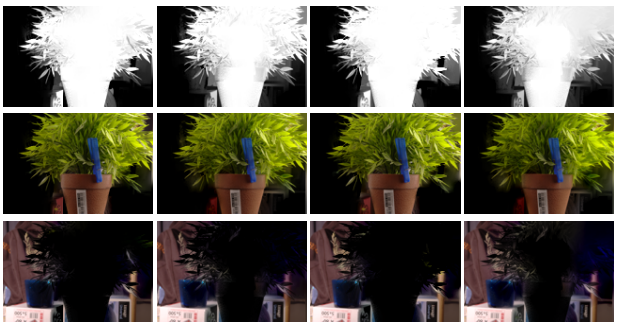
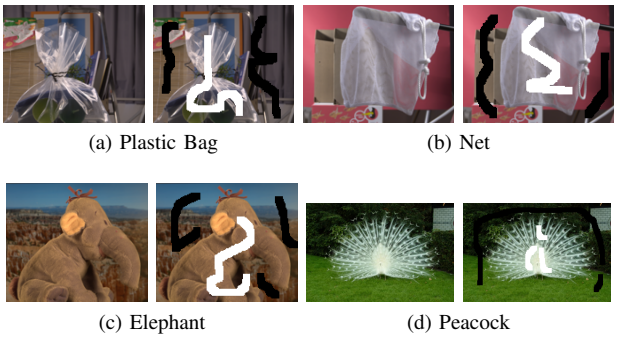


Fig. 4. Mattes from strokes with the LLE (the first and third columns) and iLLE (the second and fourth columns) alignment matrices with Scribble Smoothing (the first and second columns) and Constrained-Scribble Smoothing (the third and fourth columns) formulations: (a) Learned Masks (the first row). (b) Reconstructed foreground images (the second row). (c) Extracted background images (the third row).



(a) Plastic Bag (b) Net

(c) Elephant (d) Peacock

Fig. 5. Images and Strokes: (a) Highly Transparent. (b) Strongly Transparent. (c) Medium Transparent. (d) Little Transparent

The results for the highly transparent image `plastic bag` and the strongly transparent image `net` are shown in Figure 6 in which the first and third columns are for the SS formulation while the second and fourth columns are for the CSS formulations. The first row shows the learnt alpha matte for each image and formulation while the second row and the third row show the extracted foreground and background images.

It is obvious that, for the highly transparent plastic bag, the learnt alpha matte from the CSS formulation is much better than the one from the SS formulation while for the strongly transparent net the results from both formulation schemes are comparable to each other.

Similarly we have shown the results for both the medium transparent `elephant` image and the little transparent `peacock` image in Figure 7.

From this group of results, we can conclude that the performance of both formulation schemes are comparable to each other, but the CSS formulation slightly outperforms

  DOI:

Fig. 6. Learning Results for the Highly and Strongly Transparent Images

Fig. 7. Learning Results for the Medium and Little Transparent Images

the SS formulation as can be observed from the part of background that has been absorbed into the foreground by the SS formulation, e.g., the green spot on left upper region of the `peacock`'s plumage in the image.

Actually the matting problem for all the images used in this experiment is very challenging. For example, the colour information of the background and foreground in the `elephant` image is quite similar. In this case, we note that the SS formulation performs much better than the CSS formulation scheme.

V. Conclusion

This paper proposes two formulations for image matting based on the classical LLE alignment matrix and improve LLE alignment matrix. The experiments have demonstrated both formulations are comparable to each other while in many cases the CSS formulation is slightly better than the SS formulation. The experiment also demonstrated that the SS formulation based on the iLLE alignment matrix is comparable to both LEE and Laplacian alignment matrices. A similar approach [8] can be used for further edge preservation constraint. This kind of observation is intuitively encouraging for manifold learning methods to be successful in image matting.

Acknowledgment

The authors are partially supported by the Faculty of Business Research Compact Grants.

References

[1] J. Wang and M. F. Cohen, "Image and video matting: A survey," *Foundations and Trends in Computer Graphics and Vision*, vol. 3, no. 2, pp. 97–175, 2007.

[2] Y. Chuang, B. Curless, D. Salesin, and R. Szeliski, "A Bayesian approach to digital matting," in *Proceedings of CVPR*, vol. 2, 2001, pp. 264–271.

[3] X. Bai and G. Sapiro, "A geodesic framework for fast interactive image and video segmentation and matting," in *Proceedings of International Conference on Computer Vision*, 2007, pp. 1–8.

[4] J. Sun, J. Jia, C. K. Tang, and H. Y. Shum, "Poisson matting," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 315–321, 2004.

[5] L. Grady, T. Schiwietz, and S. Aharon, "Random walks for interactive alpha-matting," in *VIIP*, 2005.

[6] A. Levin, D. Lischinski, and Y. Weiss, "A closed-form solution to natural image matting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 2, pp. 228–242, 2008.

[7] K. He, J. Sun, and X. Tang, "Fast matting using large kernel matting Laplacian matrices," in *Proceedings of CVPR 2010*, 2010, pp. 2165–2172.

[8] J. Gao, M. Paul, and J. Liu, "The image matting method with regularized matte," in *Proceedings of ICME*, 2012.

[9] S. Tierney and J. Gao, "Natural image matting with total variation regularisation," in *Proceedings of DICTA*, 2012, pp. 1–8.

[10] J. Gao, "Image matting via local tangent space alignment," in *Proceedings of the International Conference on Digital Image Computing: Techniques and Applications (DICTA)*, 2011.

[11] Z. Zhang and H. Zha, "Principal manifolds and nonlinear dimension reduction via local tangent space alignment," *SIAM Journal of Scientific Computing*, vol. 26, pp. 313–338, 2004.

[12] L. van der Maaten, E. O. Postma, and H. van den Herick, "Dimensionality reduction: A comparative review," Tilburg University, Technical Report TiCC-TR 2009-005, 2009.

[13] N. D. Lawrence, "A unifying probabilistic perspective for spectral dimensionality reduction," Sheffield Institute for Translational Neuroscience, University of Sheffield, Tech. Rep., 2010, arXiv:1010.4830v1.

[14] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, pp. 2323–2326, 2000.

[15] S. Xiang, F. Nie, C. Pan, and C. Zhang, "Regression reformulations of LLE and LTSA with locally linear transformation," *IEEE Trans on Systems, Man and Cybernetics, Part B: Cybernetics*, vol. 41, no. 5, pp. 1250–1262, 2011.

[16] Y. Li, J. Sun, C. K. Tang, and H. Y. Shum, "Lazy snapping," *ACM Transactiona on Graphics*, vol. 23, no. 3, pp. 303–308, 2004.

[17] C. Rother, V. Kolmogorov, and A. Blake, "Grabcut: Interactive foreground extraction using iterated graph cuts," *ACM Transactions on Graphics*, vol. 23, no. 3, pp. 309–314, 2004.

[18] D. DeCoste, "Visualizing mercer kernel feature spaces via kernelized locally-linear embeddings," in *Proc. of the Eighth International Conference on Neural Information Processing*, 2001.

[19] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural Computation*, vol. 15, pp. 1373–1396, 2003.

[20] D. L. Donoho and C. Grimes, "Hessian eigenmaps: locally linear embedding techniques for high-dimensional data," in *Proceedings of the National Academy of Arts and Sciences*, vol. 10, 2003, p. 55915596.

[21] H. Chang and D. Yeung, "Robust locally linear embedding," *Pattern Recognition*, vol. 39, pp. 1053–1065, 2006.

[22] Y. Pan, S. Ge, and A. Al-Mamun, "Weighted locally linear embedding for dimension reduction," *Pattern Recognition*, vol. 42, pp. 798–811, 2009.

[23] S. Q. Zhang, "Enhanced supervised locally linear embedding," *Pattern Recognition Letters*, vol. 30, no. 13, pp. 1208–1218, 2009.

[24] L. Zhao and Z. Zhang, "Supervised locally linear embedding with probability-based distance for classification," *Computers & Mathematics with Applications*, vol. 57, no. 6, pp. 919–926, 2009.

[25] W. L. Zheng, N. Ru, and F. H. Yao, "Locally linear embedding for fault diagnosis," *Advanced Materials Research*, vol. 139-141, pp. 2599–2602, 2010.

  DOI:

[26] C. J. Lin and J. J. Moré, "Newton's method for large bound-constrained optimization problems," *SIAM Journal on Optimization*, vol. 9, pp. 1100–1127, 1999.

[27] Y. Nesterov, "A method for solving a convex programming problem with convergence rate $o(1/k^2)$," *Soviet Math. Dokl.*, vol. 27, pp. 372–376, 1983. [Online]. Available: http://www.core.ucl.ac.be/~nesterov/Research/Papers/DAN83.pdf

[28] ——, "Gradient methods for minimizing composite objective function," Université catholique de Louvain, Belgium, CORE DISCUSSION PAPER 76, 2007. [Online]. Available: http://www.uclouvain.be/cps/ucl/doc/core/documents/Composit.pdf

[29] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM Journal of Imaging Sciences*, vol. 2, no. 1, pp. 183–202, 2009.

  DOI: