LLE Algorithm in Natural Image Matting

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

Abstract—Accurately extracting foreground objects in images and video has wide applications in digital photography. These kind of problems are referred to as image matting. The most recent work [1] in natural image matting relies on local smoothness assumptions about foreground and background colours on which a cost function has been established. The closed-form solution has been derived based on a certain degree of user inputs. In this paper, we present a framework for formulating a new cost function from the manifold learning perspective based on the so-called Locally Linear Embedding [2] where the local smoothness assumptions have been replaced by an implicit manifold structure defined in local colour spaces. 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

As usual, we refers to the problem of softly extracting the foreground object from a single image as image matting. These kind of problems are important in both computer vision and graphics applications. Image matting has been widely used in film production applications and is a key technique in image/video editing. The past twenty years have witnessed rapid development in digital matting approaches. A 2007 survey article [3] provided a comprehensive review of existing image and video matting algorithms and systems, with an emphasis on the advanced techniques that have been recently proposed.

Most of existing digital matting techniques depend on the so-called alpha matte. Mathematically, the alpha matte is based on the following model assumption

\[ I = \alpha F + (1 - \alpha)B \]

where \( \alpha \), \( I \), \( F \) and \( B \) are the alpha matte value, image colour value, the foreground image colour value and the background colour value, respectively, at any given pixel. The alpha matte value \( \alpha \) is assumed to be 0 or 1 (hard matte value) which clearly distinguish foreground and background pixels. However it would be convenient to assume that \( \alpha \) lies between 0 and 1 as soft matte.

In a matting problem, the goal is to estimate \( \alpha \), \( F \) and \( B \) simultaneously, given the image pixel information \( I \) for all the pixels. Obviously this is a severely ill-posed problem. Many existing matting algorithms and systems require a certain degree of user interaction to extract a good matte. For example, a so-called trimap is usually supplied to matting algorithms or systems, which indicates definite foreground, definite background and 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 [4] uses a progressively moving window marching inward from the known regions. Bai and Sapiro [5] proposed calculating alpha matte through the geodesic distance from a pixel to the known regions. Among the propagation-based approaches, the Poisson matting algorithm [6] 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 [7] based on Random Walks. The Closed-form matting approach was proposed by Levin et al. [1] via introducing the matting Laplacian matrix under the assumption that foreground and background colours can be fit with linear models in local windows, which leads to a quadratic cost function in alpha that can be minimized globally. Minimizing the quadratic cost function is equivalent to solving a linear system which is often time-consuming when the image size is large. In a recent work [8], He et al. proposed a fast matting scheme by using large kernel matting Laplacian matrices. Review and assessment of more matting techniques can be found at the alpha matting evaluation website http://www.alphamatting.com/.

In this paper, we consider using traditional manifold learning algorithms [9], [10] such as Locally Linear Embedding (LLE) [2] to solve the problem of image matting. As has been done in [1], we still use the so-called user-specified constraints (such as scribbles or a bounding rectangle). Similar approaches can also be found in [1], [11], [12]. In deriving the matting Laplacian matrices [1], the authors have used the assumption that the foreground (or background) colours in a local window lie on a single line in the RGB 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) manifold learning
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 [1]. We make our conclusions in Section V with suggestions for further exploring the application of dimensionality reduction algorithms in image matting.

II. LOCALLY LINEAR EMBEDDING ALIGNMENT MATRIX

The locally linear embedding (LLE) algorithm has been recently proposed as a powerful eigenvector method for the problem of nonlinear dimensionality reduction [2]. The key assumption related to LLE is that, even if the manifold embedded in a high-dimensional space is nonlinear when considered as a whole, it still can be assumed to be locally linear if each data point and its neighbours lie on or close to a locally linear patch of the manifold. LLE maps its inputs into a single global coordinate system of lower dimensionality, and its optimizations do not involve local minima. In the last decade, many LLE based algorithms have been developed and introduced in the machine learning community: kernelized LLE (KLLE) [13], Laplacian Eigenmap (LEM) [14], Hessian LLE (HLLE) [15], robust LLE [16], weighted LLE [17] enhanced LLE [18] and supervised LLE [19].

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 [2], classification [17], [19] and fault detection [20].

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 high-dimensional 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 [1], 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 $X_i = \{I_{ij}|i_j \in w_i, j = 1, ..., K\}$ to denote the subset of colour vectors over a local window pixels of pixel $i$. Note that the pixel $I_i$ is contained in $X_i$. Under the LLE assumption, the colour vector $I_i$ at pixel $i$ can be approximated by a linear combination $w_{ij}$ (the so-called reconstruction weights) of its $K - 1$ nearest neighbours $X_i \setminus I_i$. Hence, LLE fits a hyperplane through $I_i$ and its nearest neighbors in the colour manifold defined over the image pixels. The fitting is achieved by solving the following optimal problem

$$\min_W \sum_i \|I_i - \sum_{j=1}^K w_{ij}I_{ij}\|^2$$

under the condition $\sum_{j=1}^K w_{ij} = 1$. For the sake of simplicity, we assume that $w_{ij} = 0$ when $I_{ij} = I_i$. 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 $\alpha$ can be determined by minimizing the following objective function

$$\min_\alpha F(\alpha) = \sum_i \|\alpha_i - \sum_{j=1}^K w_{ij}\alpha_j\|^2 = \alpha^T R \alpha$$

where $R = (E - W)^T (E - W)$ is called LLE alignment matrix and $E$ is the identity matrix.

In the standard LLE algorithm, (2) is usually solved with additional constrained conditions like $\|\alpha\|_2 = 1$ 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.

III. ALGORITHM FORMULATION BASED ON LLE

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 $\Omega$ be a subset of pixels and $\Gamma_\Omega$ be the operator mapping alpha matte $\alpha$ to the given alpha matting values $\alpha_0$ (trimap and user-scribbles) over $\Omega$, then the matting problem is to minimize the following objective function with respect to $\alpha$ such that $\Gamma_\Omega(\alpha) = \alpha_0$

$$\min_\alpha F(\alpha) = \alpha^T R \alpha.$$

where $R$ is the LLE alignment matrix.

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

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

$$\min_\alpha \alpha^T R \alpha + \lambda(\alpha - \alpha_\Omega)D_\Omega(\alpha - \alpha_\Omega^T)$$

where $\lambda$ is some large regularizer, $D_\Omega$ is a diagonal matrix whose diagonal elements are one for constrained pixels (in trimap or user-scribbles) and zero for all other pixels, and $\alpha_\Omega$ 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 $\alpha$ to its soft range $0 \leq \alpha_i \leq 1$

$$\min_{0 \leq \alpha \leq 1} \alpha R \alpha^T + \lambda (\alpha - \alpha_{\Omega}) D_{\Omega} (\alpha^T - \alpha_{\Omega}^T)$$

where $0 \leq \alpha \leq 1$ means the elements of the vector $\alpha$ are between 0 and 1.

B. Algorithms

It is easy to solve the optimization problem (3) for the scribble smoothing as it is an unconstrained quadratic programming problem. To reenforce the user scribble conditions $\alpha_{\Omega}$ over pixels $\Omega$, the algorithm in [1] takes a larger regularizer such as $\lambda = 100$. Thus the objective function can be optimized by solving a sparse linear system:

$$(R + \lambda D_{\Omega}) \alpha = \lambda \alpha_{\Omega}. \quad (5)$$

However solving the optimization problem defined in (4) is much harder as it is a quadratic programming problem with a set of linear constraints. One of approaches to solve a quadratic programming problem is to use an interior 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-like algorithm as proposed in [21] can be used for the optimization problem defined in (4). The algorithm has a fast convergence rate of $O(1/k^2)$ where $k$ is the number of iterations. However, the trust-region-like 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 [22][24], to achieve a convergence rate of $O(1/k^2)$. We first construct the following model for approximating the objective function $F(\alpha)$ in (4) at the point $\alpha$,

$$h_{C, \alpha}(\tilde{\alpha}) = F(\alpha) + F'(\alpha)(\tilde{\alpha} - \alpha)^T + \frac{C}{2}\|\tilde{\alpha} - \alpha\|^2, \quad (6)$$

where $C > 0$ is a constant.

With model (6), Nesterov’s method is based on two sequences $\{\alpha_k\}$ and $\{s_k\}$ in which $\{\alpha_k\}$ 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 $\alpha_{k-1}$ and $\alpha_k$ as

$$s_k = \alpha_k + \beta_k (\alpha_k - \alpha_{k-1})$$

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

$$\alpha_{k+1} = \arg \min_{0 \leq \tilde{\alpha} \leq 1} \frac{C_k}{2} \left\| \tilde{\alpha} - \left( s_k - \frac{1}{C_k} F'(s_k) \right) \right\|^2 \quad (7)$$

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 [24]). $\alpha_{k+1}$ defined by (7) is actually the projection of the vector $s_k - \frac{1}{C_k} F'(s_k)$ over the convex set $\{\tilde{\alpha} \mid 0 \leq \tilde{\alpha} \leq 1\}$. We can easily work out the projection given by the following formula

$$\alpha_{k+1} = \max \left\{ 0, \min \left\{ 1, s_k - \frac{1}{C_k} F'(s_k) \right\} \right\} \quad (8)$$

where both max and min operate over vectors component-wise as the same meaning in Matlab.

An efficient algorithm for (4) is summarized as

Algorithm 1 The Efficient Nesterov’s Algorithm

Input: $C_0 > 0$ and $\alpha_0$, $K$

Output: $\alpha_{k+1}$

1: Initialize $\alpha_1 = \alpha_0$, $\gamma_0 = 0$, $\gamma_0 = 1$ and $C = C_0$

2: for $k = 1$ to $K$

3: Set $\beta_k = \frac{2\gamma_k - 1}{\gamma_k - 1}$, $s_k = \alpha_k + \beta_k (\alpha_k - \alpha_{k-1})$

4: Find the smallest $C = C_{k-1}, 2C_{k-1}, ...$ such that

$$F(\alpha_{k+1}) \leq h_{C, s_k}(\alpha_{k+1}),$$

where $\alpha_{k+1}$ is defined by (8).

5: Set $C_k = C$ and $\gamma_k = \frac{1 + \sqrt{1 + 4\frac{C}{2}}}{2}$

6: end for

C. Reconstruction of Foreground and Background Images

After solving for the alpha values $\alpha$, we need to reconstruct foreground $F$ and background $B$. For this purpose, we take the same strategy as [1] to reconstruct $F$ and $B$ by using the composition equation (1) with certain smoothness priors on both $F$ and $B$. $F$ and $B$ are obtained from optimizing the following objective function

$$\min_{F, B} \sum_i \| \alpha_i F_i + (1 - \alpha_i) B_i - I_i \|^2 + \langle \partial \alpha_i, (\partial F_i)^2 + (\partial B_i)^2 \rangle$$

where $\partial$ 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 (5), 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 [1] for image matting, under the Scribble Smoothing formulation. Similar to the closed-form solution, we solve (5) for matte values.
For this experiment, two images from the original paper describing the closed form solution for image matting [1] 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 [1].

For this experiment, two images from the original paper describing the closed form solution for image matting [1] 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 [1].

B. Experiment II

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. 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) troll (Strongly Transparent); (C) donkey (Medium Transparent) and (D) pineapple (Little Transparent). The four original images and their strokes used in the experiment are displayed in Figure 3.

The results for the highly transparent image plastic bag and the strongly transparent image troll are shown in Figure 4 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 troll the results from both formulation schemes are comparable to each other.

Similarly we have shown the results for both the medium transparent donkey image and the little transparent pineapple image in Figure 5.

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 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 corner of the pineapple 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 pineapple image is quite similar. In this case, we note that the CSS formulation performs much better than the SS formulation.
scheme.

V. Conclusion

This paper proposes two formulations for image matting based on the classical 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 LLE alignment matrix is comparable to the closed-form solution of the Laplacian alignment matrix. A similar approach [25] has been used for the Local Tangent Space Alignment (LTSA) which is an unsupervised learning algorithm that computes low-dimensional, neighbourhood-preserving embeddings of high-dimensional inputs, see [26]. The argument used in [25] to apply the LTSA algorithm for image matting is that, like the application of the LLE approach, local linear structure among the colour information over a local window can be transferred to the matting value space. This kind of observation is intuitively very encouraging for manifold learning methods to be successful in image matting.

Acknowledgment

The authors are partially supported by Charles Sturt University Competitive Research Grants OPA 4818 and OPA 4356.

References