Motion-Aware KNN Laplacian for Video Matting*

Dingzeyu Li
dli@cs.columbia.edu
Columbia University

Qifeng Chen
cqf@stanford.edu
Stanford University

Chi-Keung Tang
cktang@cs.ust.hk
HKUST

Abstract

This paper demonstrates how the nonlocal principle benefits video matting via the KNN *Laplacian, which comes with a straightforward implementation using motion-aware K nearest neighbors. In hindsight, the fundamental problem to solve in video matting is to produce spatio-temporally coherent clusters of moving foreground pixels. When used as described, the motion-aware KNN Laplacian is effective in addressing this fundamental problem, as demonstrated by sparse user markups typically on only one frame in a variety of challenging examples featuring ambiguous foreground and background colors, changing topologies with disocclusion, significant illumination changes, fast motion, and motion blur. When working with existing Laplacian-based systems, our Laplacian is expected to benefit them immediately with improved clustering of moving foreground pixels.*

1. Introduction

The goal of video matting is to pull out a moving foreground matte from a single video:

$$I = \alpha F + (1 - \alpha)B \tag{1}$$

where I is the observed pixel color, F is the unknown foreground, B is the unknown background, and α is the unknown alpha matte which should be spatially and temporally coherent.

Successful works rely on generating dense trimaps or precise strokes in all frames to ensure good color samples for solving the alpha. On the other hand, if we can produce spatially and temporally coherent clusters of moving foreground pixels, then ideally the user only needs to specify a *single* pixel in each cluster to drive the automatic algorithm to produce a spatio-temporally coherent video matte.

Recent state of the arts center around the construction of graph Laplacians which have a direct impact on the quality of pixel clustering. The matting Laplacian has been widely adopted since closed-form matting and spectral matting [16, 17]. Its definition is based over a small local window, and it was shown in nonlocal image matting [14] that the matting Laplacian produces scattered matting components with the use of the local color line model. Later, nonlocal video matting [8] demonstrated good matting results, but the implementation is complicated involving several steps with specialized data structure and a matte regularization step in order to produce coherent video mattes.

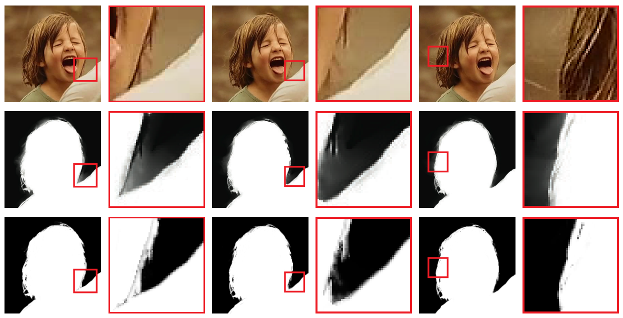
Figure 1. Comparison with nonlocal video matting [8] (middle row) on *rabbit*. KNN video matting (bottom row) produces significantly better results in the presence of ambiguous background and foreground colors. The use of optical flow as motion cue is particularly helpful in disambiguating complex situations where texture information is similar. (see electronic version)

This paper contributes to video matting by incorporating motion information in the so-called *KNN Laplacian* to make it *motion-aware*. This is the first attempt to empirically show this simple strategy is effective in producing spatio-temporally coherent pixel clusters of moving pixels. In principle, unlike nonlocal movie denoising [6] which argued against motion information, we utilize optical flow results when computing the motion-aware KNN Laplacian. Compared with other Laplacians that use motion, motion-aware KNN Laplacian has a simpler implementation and produces better quantitative results. When used on its own, it allows for sparse user markups and alpha constraints to be incorporated in a closed-form solution to produce competitive matting results, as shown in our qualitative as well as quantitative evaluation. The simplicity of motion-aware KNN Laplacian should make it easy to be incorporated into

*The research was supported by the Hong Kong Research Grant Council under grant number 619313.

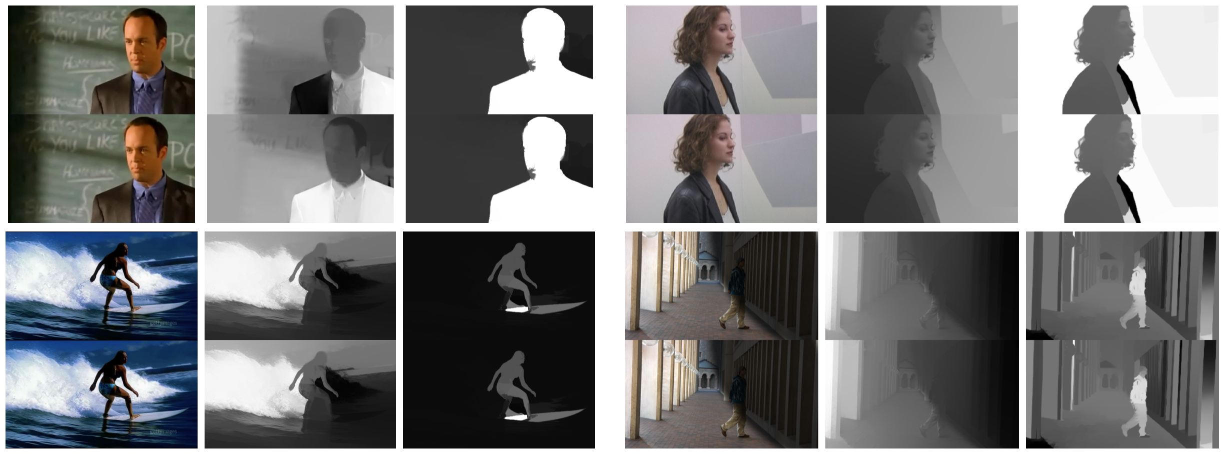

Figure 2. Two-frame motion-aware KNN Laplacian achieves excellent clustering results compared with the widely adopted matting Laplacian with the local color line model [17] which makes it difficult to incorporate bidirectional motion information to enhance temporal consistency, where optical flows often straddle beyond a local (3×3) window. This figure shows the representative eigenvectors of the respective Laplacian matrix.

existing video matting systems based on graph Laplacian, thus benefiting them immediately with improved clustering of moving foreground pixels.

2. Related Work

See [29] for a comprehensive survey on image and video matting before 2008.

Nonlocal principle. The nonlocal principle was successfully applied to image and video denoising [6] where the authors argued against the use of motion in video denoising. Two recent contributions [7, 14] applied the nonlocal principle in natural image matting. For video matting, the first attempt tapping into the nonlocal principle is [8] which, similar to [6], does not employ explicit motion information. The method uses the multi-frame nonlocal matting Laplacian proposed in [14] defined over a nonlocal neighborhood in the spatio-temporal domain. Similar to [14] specialized implementation in [8] is needed to cope with the prohibitive memory and running time requirements, with an additional postprocessing step to clean up results.

Video matting. To produce a video matte, classical video matting [10] requires the user to paint a dense trimap to be propagated to all video frames before single image matting is applied on each of them. Bayesian video matting [1] extends Bayesian image matting [9] by defining proper priors using natural image statistics. Spatio-temporal information is integrated so as to produce temporally-coherent video mattes, but the method requires the background to be estimated reliably. Hardware-assisted systems [13, 19] automatically generate and propagate trimaps in all video frames before image matting is applied on each frame. Without using motion information or optical flows, their emphasis is on complete automation rather than temporal consistency of the resulting mattes. On the other hand, the following methods rely on accurate optical flows to enhance temporal consistency: spectral video matting [12] warps matting components using optical flows; robust matting [28] was extended in [15] to include optical flows in defining an anisotropic kernel for producing temporally-coherent video mattes; affine motion was used in [11] to extend Grabcut to video matting. Recent work [3] addresses temporally-coherent video matting by adaptive trimap propagation and matte filtering in the temporal domain. Since the matting Laplacian [17] was used, the trimap needs to be precise and dense to cluster relevant but scattered matting components.

Video segmentation. To maintain spatio-temporal consistency of the object cutout, 3D meanshift was employed in interactive video cut [27] to cluster relevant pixels. The geodesic framework was extended in [2] in spatio-temporal volume for video segmentation. Rather than early commitment to optical flow vectors, which may be inaccurate, multiple candidates were kept in [18] in their graph construction to embed temporal consistency without committing to any motion vectors. In LIVEcut [21] a learning-based video segmentation algorithm was proposed that weights a range of useful cues. But the cues can only be accumulated from previous frames and defined over local regions, such as color gradient and color adjacency. Motion vectors were used in [4] to shift local windows/classifiers which does not require highly accurate optical flow information. While we also use optical flows, we embed at each pixel several motion candidates (specifically, K of them) when encoding our affinity matrix.

3. Nonlocal Principle for Video Matting

Rather than sampling reliable albeit unknown foreground/background color pairs, we advocate good pixel clustering for video matting. Good spatio-temporal clusters for moving foreground is essential for high quality results when only sparse inputs are available.

To produce good clusters we leverage the nonlocal principle in video denoising [6] but argue for the use of motion information in video matting. For completeness, we include a concise summary of the nonlocal principle while highlighting its motion-awareness for video matting.

In [6], a denoised pixel i is a weighted sum of the pixels with similar appearance with the weights given by a kernel function $k(i,j)$ in a local search window:

$$\begin{aligned} E[X(i)] &\approx \sum_{j \in \mathcal{N}(i)} X(j) k(i,j) \frac{1}{\mathcal{D}_i} && (2) \\ k(i,j) &= \exp\left(-\frac{1}{h_1^2}\|X(i) - X(j)\|_g^2 - \frac{1}{h_2^2} d_{ij}^2\right) && (3) \\ \mathcal{D}_i &= \sum_{j \in \mathcal{N}(i)} k(i,j). && (4) \end{aligned}$$

where $X(i)$ is a feature vector computed using the RGB information at/around pixel i, $\mathcal{N}(i)$ is a local search window around pixel i, and d_{ij} is the distance between pixels i and j, $\|\cdot\|_g$ is a Gaussian-weighted norm, and h_1 and h_2 are empirical constants. By analogy of (2), the expected value of alpha matte:

$$E[\alpha_i] \approx \sum_j \alpha_j k(i,j) \frac{1}{\mathcal{D}_i} \quad \text{or} \quad \mathcal{D}_i \alpha_i \approx k(i,\cdot)^T \boldsymbol{\alpha} \qquad (5)$$

where $\boldsymbol{\alpha}$ is the vector of all α values over the input image. Notice the similarity between (5) and bilateral filtering [24] except that we find the neighbors nonlocally. In nonlocal image matting [14]:

- the nonlocal principle applies to $\boldsymbol{\alpha}$ as in (5);
- the conditional distribution $\boldsymbol{\alpha}$ given X is $E[\alpha_i | X(i) = X(j)] = \alpha_j$, that is, pixels with the same appearance are expected to share the same alpha value.

3.1. KNN Laplacian

Applying the nonlocal principle in KNN video matting, we assume the alpha at pixel i is a weighted average of the alphas of its K nearest neighbors in the feature space which may not be necessarily spatially close to each other:

$$\begin{aligned} E[\alpha_i] &\approx \sum_{j \in \text{KNN}(i)} \alpha_j k(i,j) \frac{1}{\mathcal{D}_i} && (6) \\ k(i,j) &= 1 - \frac{\|X(i) - X(j)\|}{C} && (7) \\ \mathcal{D}_i &= \sum_{j \in \text{KNN}(i)} k(i,j) && (8) \end{aligned}$$

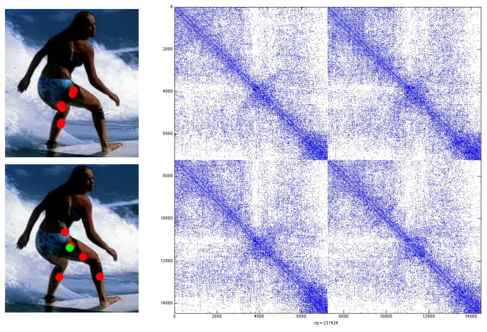
Figure 3. Left shows K nearest neighbors (red) of the selected point (green); note the nonlocal distribution of the neighbors; right shows a typical sparse nonlocal two-frame affinity matrix A in KNN video matting.

where $k(i,j)$ here uses the linear kernel which was shown in [7] to be conducive to soft segmentation. C is a normalization constant.

Following the derivation $\mathcal{D}\boldsymbol{\alpha} \approx \mathcal{A}\boldsymbol{\alpha}$ in (5), where $\mathcal{A} = [k(i,j)]$ is the *affinity matrix* and $\mathcal{D} = \text{diag}(\mathcal{D}_i)$ is an $2N \times 2N$ diagonal matrix, where N is the total number of pixels in one frame where two are used here. Thus, $(\mathcal{D} - \mathcal{A})\boldsymbol{\alpha} \approx \mathbf{0}$ or $\boldsymbol{\alpha}^T L \boldsymbol{\alpha} \approx 0$ where $L = (\mathcal{D} - \mathcal{A})^T(\mathcal{D} - \mathcal{A})$ is called the *KNN Laplacian* in this paper, noting that L is a positive semi-definite matrix.

3.2. Motion Awareness in Clustering

It remains to define the appropriate X for constructing the KNN Laplacian (section 4). In hindsight, the fundamental problem to solve is to cluster similar pixels together. In video matting, similar pixels should have similar appearance *and* motion, which agrees in principle with classical perceptual grouping or specifically, grouping by common fate [25].

Figure 2 shows that our KNN Laplacian is conducive to good graph clusters when motion information is encoded in feature vector X. We note for most video matting approaches, optical flow is almost exclusively used in trimap generation only. In contrast, as we will shortly see, optical flow is directly used in constructing our Laplacian, making our method fundamentally different because temporal consistency is considered in the matte optimization rather than the trimap generation stage.

Specifically, suppose we are given an image patch p and two candidate matches p_1 and p_2. They have identical texture; the only difference is that p_1 is moving but p_2 is not. We prefer to match p to the candidate with consistent (apparent) motion, since it is likely that both of them end up moving to (or remain stationary on) the same background in two consecutive frames, and thus likely to have the same alpha as they already have the same foreground colors.

In contrast to nonlocal denoising [6] where noise to be removed is white noise without temporal consistency, our goal is to pull out a temporally-coherent foreground matte and so motion information is considered in constructing a

	$x \times y \times t$	λ_s	λ_f	λ_p	time
kim	$720 \times 480 \times 111$	2	0.001	1	34.5
amira	$720 \times 480 \times 91$	2	0.001	1	34.1
surfer	$564 \times 320 \times 83$	3	1	3	12.2
talk	$504 \times 269 \times 70$	2	1.5	3	11.3
walk	$640 \times 360 \times 50$	3	0.6	3	15
rabbit	$800 \times 340 \times 33$	5	2	3	35.6
jurassic	$720 \times 480 \times 120$	3	1	3	28
waving	$720 \times 480 \times 35$	3	1	3	29.1

Table 1. Parameters and running times in secs for KNN video matting on a machine with an Intel i7 2.6GHz CPU. The running time includes collecting K nearest neighbors and solving the huge linear system per frame. We set $K = 15$ for all of the examples.

motion-aware KNN Laplacian. Despite that, it is not at odds with nonlocal video denoising (see Figure 8 of [6]): both nonlocal methods define proper feature vector and match similar and moving pixels to compute optimal solutions.

4. KNN Video Matting

This section applies motion-aware KNN Laplacian in what we call KNN video matting. With improved pixel clustering shown in Figure 2, we expect the motion-aware KNN Laplacian can immediately benefit existing state of the arts that are Laplacian-based.

We first describe the feature vector X which results in an asymmetric affinity $\mathcal{A}$ for embedding temporal bi-directional motion consistency, and a two-frame L for minimizing the Laplacian energy to compute an optimal video matte. KNN video matting has a straightforward implementation and produces comparable or at times better results than state-of-the-art approaches [2, 4, 8].

4.1. Feature Vector X

Our feature vector should be conducive to grouping similar pixels together, that is, pixels sharing similar appearance and similar motion should have similar α. The feature vector $X(i)$ at a pixel i in frame t is:

$$X_t(i) = (\lambda_s(x, y) \quad \lambda_f(u_f, v_f, u_b, v_b) \quad P(i, \lambda_p))_t \tag{9}$$

where (x, y) are spatial coordinates of pixel i, (u_f, v_f) and (u_b, v_b) are respectively the forward and backward motion vectors, $P(i, s)$ is an RGB image patch of size s centered at i in lexicographical order to make it one-dimensional. Thus it is easy to incorporate motion information in constructing motion-aware KNN Laplacian, not limited to trimap generation as done by many existing systems [3, 10, 12, 15].

There are three parameters: λ_s controls the amount of spatial coherence, λ_f controls the influence of optical flow [22], and λ_p controls the size of image patch, which is inspired by PatchMatch [5]. Note the difference with [8], their feature vector uses a constant 5×5 spatial patch where spatial and temporal distance are considered separately, which may not give optimally similar neighbors in spatio-temporal grouping.

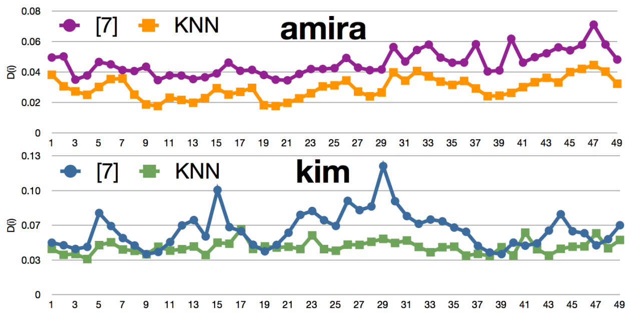

Figure 4. Temporal coherence: quantitative comparison with nonlocal video matting [8] which uses the multi-frame nonlocal Laplacian. KNN video matting uses the two-frame KNN Laplacian; our video mattes not only give smaller error between consecutive $\boldsymbol{\alpha}$ but also show a more stable temporal coherence over [8] particularly on *kim*.

4.2. Asymmetric Two-frame Affinity Matrix $\mathcal{A}$

Once X is defined we can compute $\mathcal{A}$ by (7). Mathematically, a fully connected affinity matrix $\mathcal{A}$ is symmetric provided that the distance metric is commutative. In KNN video matting, each pixel finds its K nearest neighbors in the feature space, thus the resulting $\mathcal{A}$ is not necessarily symmetric: pixel i finds pixel j as one of its nearest K neighbors, but on the other hand, pixel j may have its own K neighbors at smaller distance in the feature space than pixel i.

In [8], in order to preserve temporal coherence, three frames are used to construct their affinity matrix. Instead, we use only two frames for the following reasons. When integrating temporal information, the 2-frame affinity matrix with motion cues is more effective, since in defining X both forward and backward flows are considered, we effectively look into four frames at a time, while a smaller affinity matrix $\mathcal{A}$ of size $2N \times 2N$ is built, where N is the total number of pixels to be processed in one frame. This can drastically reduce the running time. The 2-frame affinity matrix is defined as:

$$\mathcal{A} = \begin{bmatrix} \mathcal{A}_{11} & \mathcal{A}_{12} \\ \mathcal{A}_{21} & \mathcal{A}_{22} \end{bmatrix}_{2N \times 2N} \tag{10}$$

where $\mathcal{A}_{11}$ and $\mathcal{A}_{22}$ are intra-frame affinity matrix, computed within frame 1 and frame 2 respectively, and $\mathcal{A}_{12}$ and $\mathcal{A}_{21}$ describe the inter-frame affinity information between the two frames under consideration. Figure 3 shows a typical two-frame affinity matrix. In general, to enhance temporal coherence by supplying more candidate nonlocal matches, a larger affinity matrix involving $n \geq 2$ frames can be defined in a similar manner. In practice, memory and running time will be an issue. To avoid kd-tree segmentation and other sophisticated post-processing as done in [8], we choose $n = 2$ in our implementation which produces competitive results in our experiments.

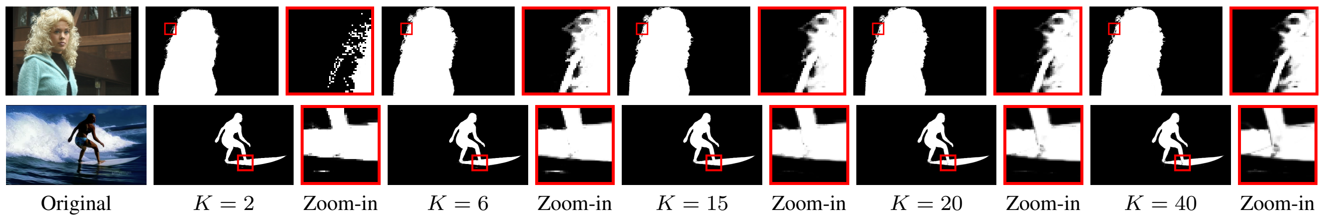

Figure 5. Analysis on different K. For small K, insufficient information from neighbors gives a poor estimation, while a huge K introduces speed and memory issues. We keep $K = 15$ constant in our experiments.

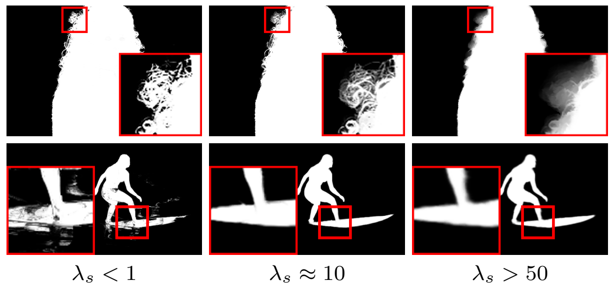

Figure 6. Analysis on parameter λ_s. When λ_s is too small, the matte becomes brittle since the affinity matrix is built using color and motion information which are ambiguous for small λ_s. Excessively large λ_s tends to over-smooth the matte.

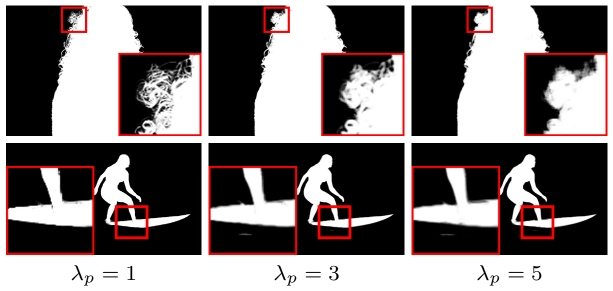

Figure 7. Analysis on parameter λ_p. Larger patch can collect useful textural information by concatenating the neighboring pixels' RGB value thus also making our method more robust to noise. One exception is hairy foreground, where smaller patch is preferred as most hair strand has width size of one pixel.

4.3. Alpha Constraint vs Trimap Propagation

Most recent methods [3, 8, 10, 13, 15, 19] require trimaps for each frame be explicitly available to compute $\boldsymbol{\alpha}$ in a video sequence, either manually drawn or computed via optical flow. In hindsight, trimap propagation has several disadvantages. Accurate trimap propagation requires reliable optical flow estimation, because the trimap propagated to the next frame is expected to be error free: wrongly-propagated definite foreground or hard constraint is often detrimental and hard to correct during optimization. However, this accuracy is not guaranteed even with the state-of-the-art optical flow algorithms. On the other hand, one may argue for trajectory estimation, which is more accurate since sophisticated motion models (such as affinity tensors [20]) are considered. However, the estimated trajectories are usually too sparse to be practical for trimap propagation, typically around 200 trajectories in a 100-frame video at 600×400 resolution as in [20].

In KNN video matting, we make use of $\boldsymbol{\alpha}_t$ as *soft* constraint to optimize $\boldsymbol{\alpha}_{t+1}$, which has the additional advantage of refining $\boldsymbol{\alpha}_t$ after the optimization. Details are in the following derivation.

4.4. Closed-Form Solution and Implementation

Denote $\mathbf{m}$ of size $2N \times 1$ as the binary indication vector for the pixels that have definite alpha values (1 or 0), where $\mathbf{m} = \mathbf{m}_f + \mathbf{m}_b$, $\mathbf{m}_f = \begin{bmatrix} \mathbf{v}_f \\ \mathbf{0} \end{bmatrix}$ $\mathbf{m}_b = \begin{bmatrix} \mathbf{v}_b \\ \mathbf{0} \end{bmatrix}$, where $\mathbf{v}_b$ and $\mathbf{v}_f$ are $N \times 1$ indication vectors respectively specifying definite foreground and background in frame t, and $\mathbf{0}$ is an $N \times 1$ zero vector since $\boldsymbol{\alpha}_{t+1}$ is unknown. In essence, we apply $\boldsymbol{\alpha}_t$ as a soft constraint when we solve for $\boldsymbol{\alpha}_{t+1}$ where *no* hard constraint (trimap or alpha) from frame t is propagated to frame $t+1$. Our energy function w.r.t. to $x = \begin{bmatrix} \boldsymbol{\alpha}_t \\ \boldsymbol{\alpha}_{t+1} \end{bmatrix}$ is defined as following:

$$\begin{aligned} g(x) &= x^T L x + \lambda \left[\sum_{i \in \mathbf{m}_b} x_i^2 + \sum_{i \in \mathbf{m}_f} (1 - x_i)^2 \right] \\ &= x^T L x + \lambda \left[\sum_{i \in \mathbf{m}} x_i^2 - 2\mathbf{m}_f^T x + |\mathbf{m}_f| \right] \\ &= x^T (L + \lambda D) x - 2\lambda \mathbf{m}_f^T x + \lambda |\mathbf{m}_f| \end{aligned}$$

where $D = \mathrm{diag}(\mathbf{m})$ and λ is a constant controlling our confidence on the previous computed alpha. Differentiating $g(x)$ w.r.t. x and equating the result to zero:

$$\frac{\partial g}{\partial x} = 2(L + \lambda D)x - 2\lambda \mathbf{m}_f^T = 0 \qquad (11)$$

The optimal solution is

$$x = (L + \lambda D)^{-1}(\lambda \mathbf{m}_f). \qquad (12)$$

After solving this linear system, we obtain not only $\boldsymbol{\alpha}_{t+1}$ but also the refined $\boldsymbol{\alpha}_t$ for free. When erroneous alphas are present, the K nearest neighbors are capable of nonlocally averaging the alpha to ameliorate their effect. Unlike [12] our alpha map on the previous frame is not motion-warped to the current frame where the current alpha is being optimized. For moderate motion, $\boldsymbol{\alpha}_t$ is sufficient for providing the necessary soft constraint. For fast and rapid motion, optical flows tend to be inaccurate, thus the incorrectly

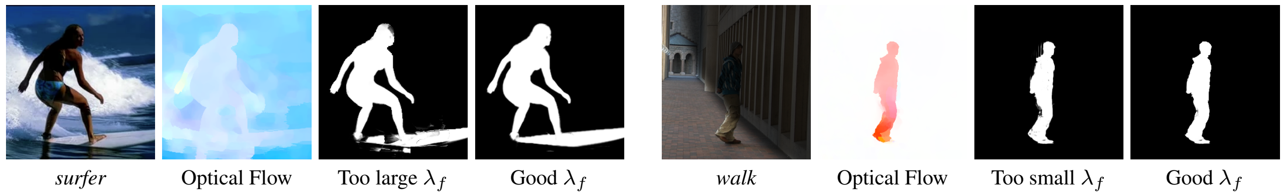

Figure 8. Analysis on parameter λ_f. This parameter controls the influence of motion vectors. In videos such as *surfer*, λ_f should not be too large because the optical flows are noisy and inaccurate, whereas in challenging example *walk*, larger λ_f can extract a clearer alpha matte, since optical flow gives good estimation on the man's movement.

warped alphas may introduce bad constraints, not to mention that alpha warping introduces complication as the mapping is seldom one-to-one. When motion is inaccurate, we can weaken the influence of optical flow (by adjusting λ_f) so that color/texture information can dominate.

According to the implementation of [7] which is publicly available, their affinity matrix is forced to be symmetric, so that they can solve the sparse system using the preconditioned conjugate gradient method, which runs about 5 times faster than the conventional conjugate method used in [16, 17]. We use the biconjugate gradients stabilized method (available in Matlab as `bicgstab`) to solve our system. It turns out that the speed is even faster, i.e. 15 seconds for two-frame Laplacian matrix on a computer with Intel i7 2.6GHz CPU, note that the huge Laplacian system is twice as large as the image matting Laplacian. We use the `VLFeat` [26] to compute K nearest neighbors running in a few seconds in total. As will be shown in the result section, unlike [8] where kd-tree segmentation is used to handle memory problem, our Laplacian matrix of smaller size produces competitive results, with a closed-form solution that can be solved directly in Matlab.

5. Experimental Results

Table 1 gives a summary of running times and the values of parameters used, which shows a variety of challenging examples also used in recent contributions. In all, with our simple KNN strategy comparable results are obtained. At times we are a bit worse as we used motion-aware KNN Laplacian and nothing else. We thus expect that, when our motion-aware KNN Laplacian is adopted in state-of-the-art Laplacian-based methods [3, 4, 8, 12, 14, 16, 17, 28, 23, 30], they can be immediately benefited from the improved pixel clustering as shown in Figure 2. Optical flow computation using [22] is around one minute per frame and is not included in the table.

Figure 4 shows the quantitative comparison on temporal coherence with nonlocal video matting which will be explained in the sequel.

5.1. Effect of Parameters

We use a furry example *kim* and an example with solid boundary *surfer* to evaluate the effect of different parameters. Recall from (9) that λ_s controls the amount of spatial coherence, λ_f the influence of optical flow, and λ_p the size of an image patch. K is the number of nearest neighbors for nonlocal matching.

Fixed λ_s, λ_f and λ_p, varying K. Figure 5 analyzes the effect of different values of K while fixing all of the λ parameters. This shows that K is not critical: although the results look similar, smaller K allows for faster running time while an overly large K produces irrelevant matches manifested as unsightly matting artifacts while the overall quality is still maintained. Using an excessively small K gives poor estimation. We keep $K = 15$ unless otherwise stated in the following experiments.

Fixed K, λ_f and λ_p, varying λ_s. Figure 6 shows the effect of different λ_s which controls the spatial inter-frame and intra-frame coherence. While a small λ_s produces a brittle matte and a large λ_s over-smooths the result, we found a wide range of λ_s between the two extremes produces visually good results.

Fixed K, λ_f and λ_s, varying λ_p. Figure 7 compares the effects using different patch sizes. In general, since a larger patch collects more textural information, a smoother boundary will be encouraged. However, for hairy objects (top row of Figure 7), a smaller patch can preserve details better, when most hair strands have width of one pixel or less. Textural information is less useful for hairy foreground *kim* and *amira* as the local texture at each hair strand is similar with a lot of color ambiguities. In the following experiments, we will keep $\lambda_p = 3$ unless otherwise stated.

Fixed K, λ_p and λ_s, varying λ_f. When $\lambda_f = 0$, no motion is considered in the feature vector, then the KNN Laplacian is similar to the multiframe nonlocal Laplacian [8], except in the number of frames used (three in [8]) and in the affinity matrix construction (asymmetric here). Figure 8 shows two typical scenarios where improper λ_f can considerably deteriorate the alpha mattes: optical flow serves as a beneficial motion cue for grouping relevant pixels under color ambiguities, but since inaccurate flow is sometimes inevitable, a large λ_f in such case will give undesired results.

5.2. Baseline and Challenging Examples

Please view the supplemental material for full video results. Unless otherwise stated, only one trimap on a single frame is given for KNN video matting. We first tested on baseline examples *kim* and *amira* from [10]. Comparison on *kim* with the latest nonlocal video matting [8] is shown in Figure 9, which demonstrates that our motion-aware feature

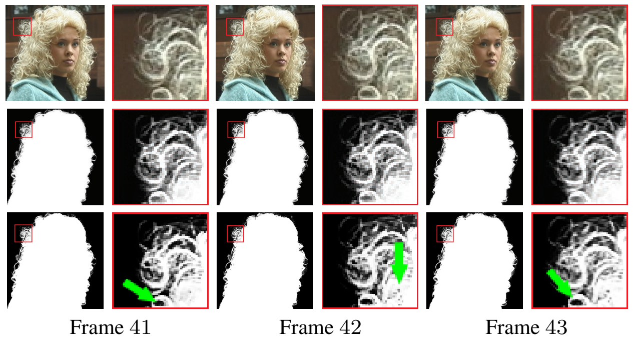

Figure 9. Comparison with nonlocal video matting [8] on *Kim*. Second row is from [8], bottom row is ours. Based on the same trimaps, we produce more accurate alpha mattes *without* additional post-processing and matte regularization. We use $\lambda_p = 1$ to capture fine hairy details.

vector is effective in discerning ambiguous texture information. We also perform quantitative comparison on temporal coherence, by measuring the difference in alpha values between successive frames to quantify the amount of temporal flicker [15]: $D(i) = \frac{\alpha_i(t+1)-\alpha_i(t)}{I_i(t+1)-I_i(t)}$. Figure 4 shows that motion-aware KNN Laplacian compares favorably to the multi-frame Laplacian used in [8], given our straightforward implementation and that no matte regularization as postprocessing is done in KNN video matting. This measure is by no means perfect, for example, in the case where consecutive frames stay the same. Nonetheless, it is still adopted here for simplicity and fairness. Better means for quantitative comparison will be future work.

Color ambiguity and low contrast. When the foreground and background have similar colors thus producing low contrast edges, and in situations where texture falls short of being discriminating, motion cues are useful in extracting good nonlocal matches. Figure 1 shows that our motion-aware feature vector works well in this complex situation. In [8] their multi-frame nonlocal Laplacian does not explicitly consider optical flow information, which results in blurry and unclear boundary.

Disocclusion and changing topology. Figure 10 demonstrates that our method can naturally handle disocclusion and changing topology via KNN search for nonlocal neighbors, while video snapcut [4] produces a hard segmentation.

Sparse inputs. Figure 11 compares with geodesic video matting [2] which also only needs sparse scribbles from the user. Since motion-aware KNN Laplacian produces good clusters, only sparse user markups are needed on the first frame. The results in subsequent frames are automatically computed via our closed-form solution.

Significant illumination changes. Figure 12 demonstrates the robustness of our soft alpha constraints and motion-aware feature vector in KNN video matting. Since the shading condition is changing over the sequence, color information alone gives an inaccurate estimation on the foreground, whereas the combination with motion vector is shown to be effective in matting out the walking man, without the additional strokes needed in [4].

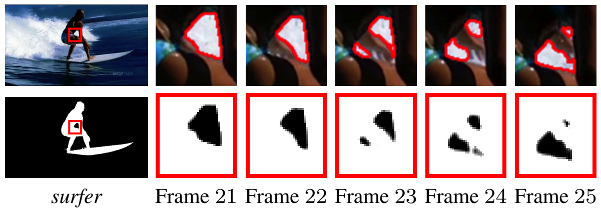

Figure 10. Comparison with video snapcut [4] on *surfer*. This example demonstrates that KNN video matting can handle disocclusion and topological changes. Similar to [4], we do not apply any user input on Frame 21 to 27; specifically, the only trimap provided is on the first frame. Notice the fine details (e.g. Frame 24) we recovered compared to the hard segmentation in [4].

Fast motion and motion blur. We show in Figure 13 our method's ability to handle fast moving foreground where the man stands up and then walks briskly away. With only motion-aware KNN Laplacian, while our result is not visually as good as that in [10], no trimap propagation is done in KNN video matting: only several trimaps on the keyframes are supplied. This figure also shows a failure mode using a challenging example on motion blur from [4]. A blurry image/video in general is modeled by image convolution rather than the image compositing equation (1) assumed in alpha matting. Our system degrades gracefully although it is hard to delineate the boundary between the foreground and background.

6. Conclusion

We study the nonlocal principle applied to video matting and use motion to disambiguate complex situations where colors and/or texture alone would fail. Consequently, our motion-aware KNN Laplacian produces improved pixel clustering. This allows for less user input (or one trimap) and simple alpha constraint being incorporated in the closed-form solution to handle significant illumination changes among other challenging cases. We analyzed a handful of parameters which are quite easy to set while we are looking for effective ways to automatically optimize them. With its flexibility in defining feature vectors, KNN Laplacian can be easily extended to include other useful information (e.g., depth) to improve results. With its simple implementation, we expect that motion-aware KNN Laplacian can be readily incorporated into Laplacian-based video matting systems to benefit them with better moving pixel clustering.

Acknowledgments. Input and output of [2, 4] are courtesy of X. Bai. Input and output of [8] are courtesy of Y.-W. Tai.

References

[1] N. Apostoloff and A. Fitzgibbon. Bayesian video matting using learnt image priors. In *CVPR*, pages I:407–414, 2004.

[2] X. Bai and G. Sapiro. A geodesic framework for fast interactive image and video segmentation and matting. In *ICCV*, 2007.

[3] X. Bai, J. Wang, and D. Simons. Towards temporally-coherent video matting. In *MIRAGE*, volume 6930, pages 63–74. Springer, 2011.

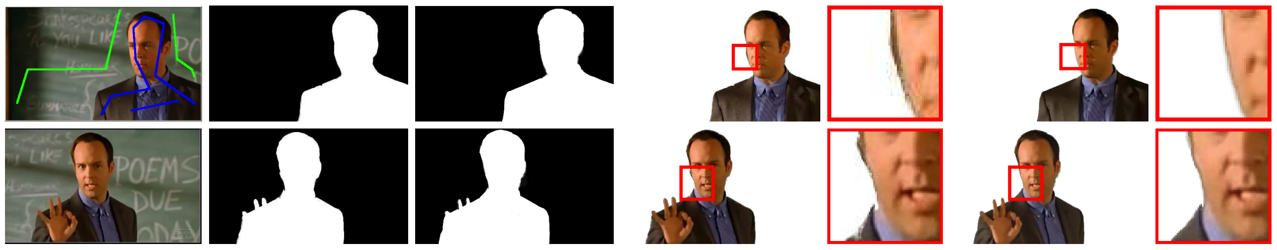

Figure 11. Comparison with geodesic matting [2] on *talk* using sparse strokes. Only strokes on the first frame are given and all the αs are computed using our closed-form solution. While the α results look similar, αI shows our method extracts a better foreground.

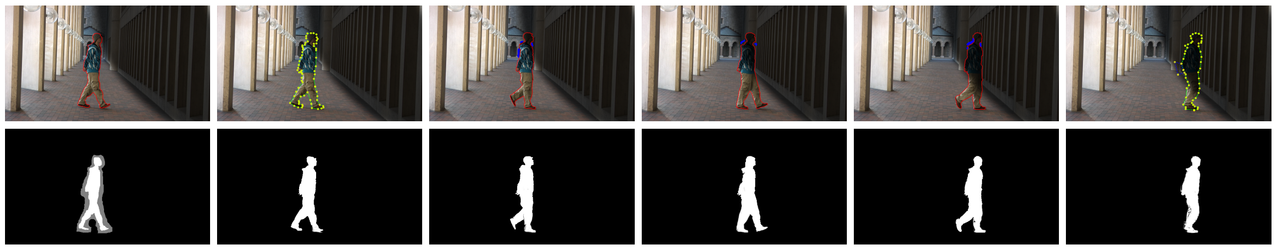

Figure 12. Comparison with video snapcut [4] on *walk*. Our results (bottom) are robust to stark illumination changes given only a *single* input trimap (Frame 12). The shading on the walking man is constantly changing. In video snapcut (top), the user needs to supply quite a number of additional strokes to achieve a comparable segmentation, for example, by carefully drawn control points on Frame 15 and 51 as well as blue strokes on the intermediate frames.

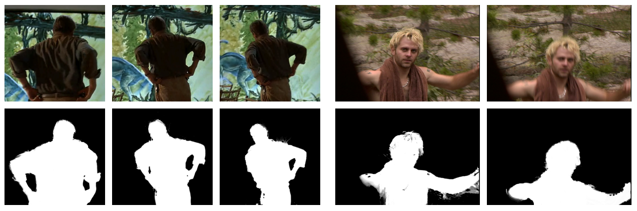

Figure 13. Motion-aware KNN Laplacian degrades gracefully in fast and complex motion in front of a background with ambiguous colors (left, *jurassic*), and in presence of motion blur (right, *waving*).

[4] X. Bai, J. Wang, D. Simons, and G. Sapiro. Video snapcut: robust video object cutout using localized classifiers. *ACM Trans. Graph.*, 28(3), 2009.

[5] C. Barnes, E. Shechtman, D. B. Goldman, and A. Finkelstein. The generalized patchmatch correspondence algorithm. In *ECCV'10*, pages 29–43, 2010.

[6] A. Buades, B. Coll, and J.-M. Morel. Nonlocal image and movie denoising. *IJCV*, 76(2):123–139, 2008.

[7] Q. Chen, D. Li, and C.-K. Tang. KNN matting. In *CVPR*, pages 869–876, 2012.

[8] I. Choi, M. Lee, and Y.-W. Tai. Video matting using multi-frame nonlocal matting laplacian. In *ECCV*, pages 540–553, 2012.

[9] Y. Chuang, B. Curless, D. H. Salesin, and R. Szeliski. A bayesian approach to digital matting. *CVPR'01, II:264-271*, 2001.

[10] Y.-Y. Chuang, A. Agarwala, B. Curless, D. H. Salesin, and R. Szeliski. Video matting of complex scenes. *ACM Trans. on Graph.*, 21(3):243–248, July 2002.

[11] D. Corrigan, S. Robinson, and A. Kokaram. Video matting using motion extended grabcut. In *Visual Media Production (CVMP 2008), 5th European Conference on*, pages 1 –9, nov. 2008.

[12] M. Eisemann, J. Wolf, and M. Magnor. Spectral video matting. In *Vision, Modeling and Visualization*, 2009.

[13] N. Joshi, W. Matusik, and S. Avidan. Natural video matting using camera arrays. *ACM Trans. Graph.*, 25(3):779–786, July 2006.

[14] P. Lee and Y. Wu. Nonlocal matting. In *CVPR*, pages 2193–2200, 2011.

[15] S.-Y. Lee, J.-C. Yoon, and I.-K. Lee. Temporally coherent video matting. *Graphical Models*, 72(3):25 – 33, 2010.

[16] A. Levin, D. Lischinski, and Y. Weiss. A closed-form solution to natural image matting. *IEEE TPAMI*, 30(2):228–242, 2008.

[17] A. Levin, A. Rav-Acha, and D. Lischinski. Spectral matting. *IEEE TPAMI*, 30:1699–1712, October 2008.

[18] Y. Li, J. Sun, and H.-Y. Shum. Video object cut and paste. *ACM Trans. Graph.*, 24(3):595–600, jul 2005.

[19] M. McGuire, W. Matusik, H. Pfister, J. F. Hughes, and F. Durand. Defocus video matting. *ACM Trans. Graph.*, 24(3):567–576, 2005.

[20] P. Ochs and T. Brox. Higher order motion models and spectral clustering. In *CVPR*, pages 614–621, 2012.

[21] B. L. Price, B. S. Morse, and S. Cohen. Livecut: Learning-based interactive video segmentation by evaluation of multiple propagated cues. In *ICCV*, pages 779–786, 2009.

[22] D. Sun, S. Roth, and M. J. Black. Secrets of optical flow estimation and their principles. In *CVPR*, pages 2432–2439, 2010.

[23] J. Sun, J. Jia, C.-K. Tang, and H.-Y. Shum. Poisson matting. *ACM Trans. Graph.*, 23:315–321, August 2004.

[24] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *ICCV*, pages 839–846, 1998.

[25] A. Treisman. Perceptual grouping and attention in visual search for features and for objects. *Journal of experimental psychology. Human perception and performance*, 8(2):194–214, Apr 1982.

[26] A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms, 2008.

[27] J. Wang, P. Bhat, R. A. Colburn, M. Agrawala, and M. F. Cohen. Interactive video cutout. *ACM Trans. Graph.*, 24(3):585–594, 2005.

[28] J. Wang and M. F. Cohen. Optimized color sampling for robust matting. *IEEE CVPR*, 2007.

[29] J. Wang and M. F. Cohen. *Image and Video Matting*. Now Publishers Inc., Hanover, MA, USA, 2008.

[30] S. K. Yeung, C.-K. Tang, M. S. Brown, and S. B. Kang. Matting and compositing of transparent and refractive objects. *ACM Trans. Graph.*, 30(1):2, 2011.