A Sparse Control Model for Image and Video Editing

Li Xu Qiong Yan Jiaya Jia*

Department of Computer Science and Engineering
The Chinese University of Hong Kong

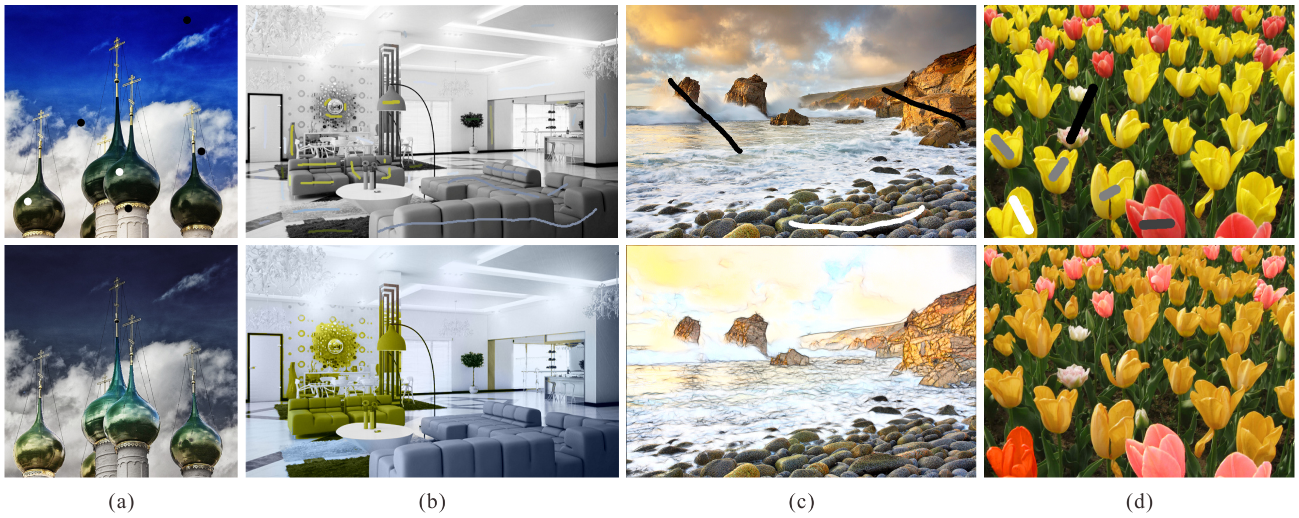


Figure 1: *Very sparse control samples for image and video editing. Applications that can be benefitted include, but are not limited to, (a) tone adjustment, (b) colorization, (c) non-photorealistic rendering, and (d) re-colorization. In the two rows, we show input images with editing samples and our corresponding results respectively. Input image (c) courtesy of Patrick Smith.*

Abstract

It is common that users draw strokes, as control samples, to modify color, structure, or tone of a picture. We discover inherent limitation of existing methods for their implicit requirement on where and how the strokes are drawn, and present a new system that is principled on minimizing the amount of work put in user interaction. Our method automatically determines the influence of edit samples across the whole image jointly considering spatial distance, sample location, and appearance. It greatly reduces the number of samples that are needed, while allowing for a decent level of global and local manipulation of resulting effects and reducing propagation ambiguity. Our method is broadly beneficial to applications adjusting visual content.

Keywords: image editing, video editing, image processing, matting, smoothing, cutout, colorization, segmentation

Links: DL PDF WEB CODE

*e-mail: {xuli, qyan, leojia}@cse.cuhk.edu.hk

1 Introduction

Many recent image and video processing methods are performed with the input of user set control samples for spatially-variant editing. For example, colorization [Levin et al. 2004] reconstructs chrominance channels based on a few color strokes. Interactive tone adjustment is achieved in [Lischinski et al. 2006] in a similar manner. Other representative tools include natural image matting [Wang and Cohen 2005; Levin et al. 2008a; Levin et al. 2008b], material editing [Pellacini and Lawrence 2007; An and Pellacini 2008], and white balance correction [Boyadzhiev et al. 2012]. All these methods share the characteristics of making results comply with image structures, in order to preserve edges in regional adjustment. The control sample strategy eschews blind parameter tuning employed in early image processing.

Albeit important and general, optimal adaptive editing from very sparse control points remains a major challenge due to the ambiguity for pixels *distant from* or *in between* the samples. The simple example in Fig. 1(d) fully exhibits the difficulty, where the user-drawn sparse strokes are expected to turn the left-bottom yellow tulip to red and all other yellow ones to orange.

Intriguingly, previous global and local methods have their respective limitations to produce correct results for this simple example. All-pixel-pair (a.k.a. global) methods [An and Pellacini 2008; Xu et al. 2009a; Xu et al. 2009b] that relate each pixel to all samples can quickly propagate edit across the image. But they suffer from mixing the two types of edits (white and gray) on many pixels. Inevitably, unpleasant color blending is produced, as will be detailed in Section 3. Local-pixel-pair (a.k.a. local) approaches [Levin et al. 2004; Lischinski et al. 2006], contrarily, need to edit tulips densely, which involve much more user interaction.

Apparently, this problem is about density and distribution of user input in different regions. But it, in fact, corresponds to the principle to design optimal strategies to propagate sparse edits. In this paper, we provide a new understanding from a feature space perspective and present a solver scrupulously designed to enable effective optimization, aware of image structures and user expectation.

Our method is based on iterative feature discrimination and relates each pixel to only part of the control samples. It is solved by global optimization. For the examples in Fig. 1, our method automatically balances edits, making these samples adaptively propagated to different groups of pixels. In terms of generality, our method can incorporate a variety of smoothing schemes, such as weighted least square regularizer and matting Laplacian, to exhibit various edge-preserving properties. A spectrum of features can also be employed to enlist new applications. The profit in long-range propagation within and across frames fits this framework to video editing.

2 Related Work

Propagating user drawn stokes to all pixels was used in colorization [Levin et al. 2004]. It is archived by constructing a sparse affinity matrix for neighboring pixel pairs, which was found effective also for local tone adjustment [Lischinski et al. 2006], material editing [Pellacini and Lawrence 2007] and intrinsic image computation [Bousseau et al. 2009]. The neighbors can be in spatial or feature space combining color and location. A typical strategy is to construct the affinity matrix with a finite number of neighbors, making it sparse and solvable. To enable long-range propagation, all-pair constraints were employed [An and Pellacini 2008], which were further accelerated by clusters with K-D trees [Xu et al. 2009a]. The affinity matrix is dense but with a low rank, which could be approximated by the Nyström extension [Fowlkes et al. 2004]. However, the low-rank condition is valid only when the interaction range is sufficiently large.

Various feature space or distance measures were also studied in the literature. To cross texture and fragmented regions, geodesic distance [Criminisi et al. 2010] and diffusion distance [Farbman et al. 2010] were be applied. Li et al. [2008] advocated the use of pixel classification based on user input.

Matting techniques aim to generate soft alpha mattes for the foreground region [Wang and Cohen 2005; Levin et al. 2008a; Levin et al. 2008b; Rhemann et al. 2009] with the input of a trimap or strokes. Most matting approaches bear a resemblance to local edit propagation and differ on the way to deal with intricate boundaries. For example, matting Laplacian [Levin et al. 2008a] can handle fuzzy boundaries. Nonlocal matting [Lee and Wu 2011] and KNN matting [Chen et al. 2012a] exploited long-range relationship among pixels, approximating all-pair affinity propagation. Locally-linear embedding propagation preserved the manifold structure [Chen et al. 2012b] to tackle color blending. Our framework can similarly generate soft mattes when foreground and background labels are set. The main contribution is on the principled adaptation to relate pixels considering sparse strokes.

Edit propagation is also related to edge-preserving smoothing, in the sense that edge-aware interpolation can be achieved. Representative methods include bilateral filtering [Durand and Dorsey 2002; Paris and Durand 2006; Kopf et al. 2007; Chen et al. 2007], tone manipulation [Bae et al. 2006; Fattal et al. 2007], optimization with a weighted least square (WLS) regularizer [Farbman et al. 2008], edge-avoid wavelet [Fattal 2009], histogram-based filtering [Kass and Solomon 2010], local Lapalacian [Paris et al. 2011], L_0 gradient smoothing [Xu et al. 2011], and texture-aware separation [Subr et al. 2009; Xu et al. 2012]. These approaches preserve strong edges while suppressing details. But they are not designed for efficient edit propagation due to the use of local smoothness constraints. Recently, high-dimensional filtering achieves real-time performance [Adams et al. 2009; Adams et al. 2010; Krähenbühl and Koltun 2011; Gastal and Oliveira 2011; Gastal and Oliveira 2012] on GPU or even with optimized C code [Ragan-Kelley et al. 2012].

For video and multi-frame editing, feature clustering [Xu et al. 2009a], fast interpolation [Li et al. 2010], and optical flow [Lang et al. 2012] were studied. Our method with controllable interaction can be naturally employed in video editing.

3 Background

We show in this section that both local and global methods to propagate edits actually have their respective implicit requirements on how users draw editing strokes.

A grayscale image in Fig. 2(a) is overlaid with colorization samples. Three points A, B, and C are on two different strokes that are marked in orange and green, and on a region far from the strokes. Each pixel corresponds to a 5D feature, detailed below. (e) illustrates the relationship in the dominant 2D of the feature space.

For the methods proposed for local appearance adjustment [Levin et al. 2004; Lischinski et al. 2006], a general data cost, without involving the smoothness constraint, can be expressed as

$$\sum_{i\in\Omega}(s_i - g_i)^2, \tag{1}$$

where s is the output editing action label and g_i denotes an exemplar action label, such as color change or tone adjustment, for pixel i where i is in the user-drawn stroke set Ω. This term has definition only on editing samples. For pixels afar, data costs do not exist; so their values have to be determined through an extra smoothness constraint, which propagates confidence to neighbors, as shown in Fig. 2(f). In the corresponding result shown in (b), points A and B are naturally assigned with different colors. Because C is far from all samples, its final color is hardly predictable before optimization because slightly varying parameters or optimization procedure may make it altered. The user thus has to draw dense samples/strokes to make sure all pixels are processed in accordance to expectation.

All-pair constraints [An and Pellacini 2008; Xu et al. 2009a], on the contrary, allow for a large propagation range, where the data term can be expressed as

$$\sum_i \sum_{j\in\Omega} k_{ij}(s_i - g_j)^2, \tag{2}$$

where k_{ij} is an attenuation function, generally Gaussian-like:

$$k_{ij} = \exp\{-\|\mathbf{f}_i - \mathbf{f}_j\|^2/\sigma_{\mathbf{f}}\}. \tag{3}$$

Eq. (2) relates each pixel i to many samples j as shown in Fig. 2(g). k_{ij} is to reduce influence when pixels have different feature vectors $\mathbf{f}$. A common form of features $\mathbf{f}$ is concatenation of normalized color vectors and spatial coordinates in a total of 5 dimensions. $\sigma_{\mathbf{f}}$ in Eq. (3) is for normalization.

The final objective similarly incorporates a smoothness term. After optimization, pixels around A and B are affected by both samples according to the data term, incurring color mixing as shown in Fig. 2(c). Adding more strokes worsens this problem owing to the inherent long-range influence.

The above analysis reveals the fact that prior methods require samples to be drawn in their respective preferred manners. Unexpected results could be produced if the rules are not followed. This introduces extra difficulty and demand for common users to understand how propagation is achieved.

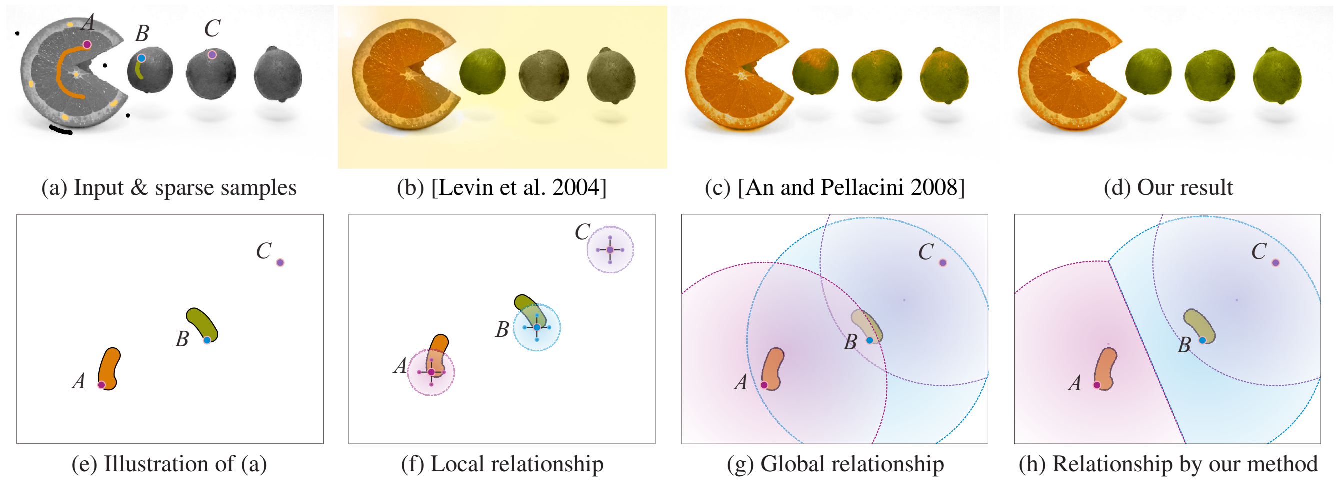


Figure 2: *Difficulty of edit propagation from sparse samples. (a) Input. (b)-(d) Three results by local, global, and our methods. (e)-(h) Illustrations of the data term influence in the dominant two dimensions of the feature space.*

In what follows, we present a new model to overcome these inherent limitations. Our method automatically balances the rival influence of two strokes for A and B and yields a visually pleasing result, shown in Fig. 2(d), even with a small amount of user input. Point C is colorized naturally.

4 Our Model

Our general *one-sample principle* to deal with edit propagation is on *optimally* relating resulting editing label for each pixel to the most confident sample, or one cluster of samples undergoing similar adjustment. The benefit is twofold. On the one hand, unlike global methods that connect one pixel to all samples in Eq. (2), this principle prevents color mixing. On the other hand, when multiple samples are with different functions – for instance, one sample turns a red pixel to green and another turns a red pixel to blue – this strategy is particularly helpful to find which sample to trust when handling another red pixel in the vicinity.

Our objective to adaptively determine impact of each control sample depends on control location, image structure, and spatial distance. We denote a user-provided editing label as g_j for pixel j, which could be a color vector in colorization, a binary label in matting, or a vector of binary labels when there are multiple types of tasks. g_j is non-zero only when $j \in \Omega$, where Ω is the set containing locations of sparse inputs. We denote by s_i the resulting editing label for each pixel i in the same value or vector form as g_j.

We regard s_i as being generated from a distribution determined by input g. It yields the following mixture model for each pixel i:

$$p(s_i|g) \propto \sum_{j\in\Omega} \pi_{ij} \exp\left\{-\frac{\|s_i - g_j\|^2}{\sigma_s}\right\}, \tag{4}$$

where σ_s controls the standard deviation. π_{ij} is the mixture coefficient corresponding statistically to the portion of samples in the category, defined based on the affinity in feature space, i.e.,

$$\pi_{ij} = \frac{k_{ij}}{\sum_{j\in\Omega} k_{ij}}, \tag{5}$$

where k_{ij} is the affinity between pixels i and j, defined in Eq. (3) as $\exp\{-\|\mathbf{f}_i - \mathbf{f}_j\|^2/\sigma_{\mathbf{f}}\}$. We use the same 5D feature $\mathbf{f}$ here for simplicity's sake, which will be flexibly changed to other more complicated forms to suit various applications, detailed later.

In Eq. (5), if two pixels are similar in color and spatially close, their affinity k_{ij}, as well as the mixture coefficient π_{ij}, must be large.

Why This Model? The local data cost in Eq. (1) can be viewed as a single Gaussian. For pixels without user specified labels, the data cost does not exist and the result relies on smoothness propagation, which is difficult to cross strong edges. The all-pair data cost defined in Eq. (2) is a linear combination of quadratic penalty functions. Because of it, estimate for one pixel could be a weighted average of control samples, even if these controls are contradictive, as illustrated in Fig. 2. Our mixture model, during energy minimization, associates each pixel to only one component by nature. It is vital to guarantee optimal edit propagation.

More importantly, this model, after necessary derivations presented below, can generally represent a new feature space involving $\mathbf{f}$ and updated estimates in iterations, quickly enhancing edit propagation.

It is notable the Gaussian Mixture Model (GMM) used in matting [Rother et al. 2004; Wang and Cohen 2005] defines local color distribution, while our mixture model differentiate user editing actions $\|s_i - g_j\|$ rather than image color. This marks the essential difference in formulation, making our method employable for many operations beyond matting.

More Derivations Pixels have different probability distributions owing to the involvement of the mixture coefficients. The optimal s_i for each pixel i is yielded as

$$s_i = \arg\max_{s_i} p(s_i|g) = \arg\min_{s_i} (-\ln p(s_i|g)). \tag{6}$$

Taking the derivative w.r.t. s_i, we get

$$\begin{aligned}-\frac{\partial \ln p(s_i|g)}{\partial s_i} &= \frac{\sum_{j\in\Omega} \pi_{ij} \exp\left\{-\frac{\|s_i-g_j\|^2}{\sigma_s}\right\} 2\sigma_s^{-1}(s_i - g_j)}{\sum_{j\in\Omega} \pi_{ij} \exp\left\{-\frac{\|s_i-g_j\|^2}{\sigma_s}\right\}}, \\ &\propto s_i - \frac{\sum_{j\in\Omega} k_{ij} \exp\{-\|s_i-g_j\|^2/\sigma_s\}\, g_j}{\sum_{j\in\Omega} k_{ij} \exp\{-\|s_i-g_j\|^2/\sigma_s\}}. \end{aligned} \tag{7}$$

We use symbol $\propto$ in Eq. (7) because constant $2\sigma_s^{-1}$ is omitted. Setting Eq. (7) to zero yields a per-pixel constraint for each s_i.

To further count in structure information, we incorporate a simple weighted least square smoothness term [Lischinski et al. 2006; Farbman et al. 2010] that enforces label similarity for four neighboring pixels according to color.

We write the whole objective in a matrix form. We denote by s the output vector, g the user input vector, and by $\mathbf{W}$ the weight matrix. Each matrix element is

$$\mathbf{W}_{ij} = \frac{m_j k_{ij} \exp(-\|s_i - g_j\|^2/\sigma_s)}{\sum_j m_j k_{ij} \exp(-\|s_i - g_j\|^2/\sigma_s)}, \tag{8}$$

where m is a binary mask to mark the position of user-provided control pixels. It has value 1 for $j \in \Omega$ and 0 otherwise. The objective function with respect to s is therefore written as

$$\min E(s) = \min \left\{ (s - \mathbf{W}g)^T (s - \mathbf{W}g) + \lambda s^T \mathbf{L} s \right\}, \tag{9}$$

where $\mathbf{L}$ is the sparse Laplacian for regularization and $s^T \mathbf{L} s$ is the local smoothness term [Lischinski et al. 2006]. Each element $\mathbf{L}_{ij} = -k_{ij}$ for $i \neq j$ and $j \in N(i)$, and $\mathbf{L}_{ii} = \sum_{j \in N(i)} k_{ij}$. $N(i)$ includes four nearest neighboring pixels around i, making the resulting $\mathbf{L}$ a five-point sparse Laplacian. λ controls the smoothness strength.

Note that our framework allows $\mathbf{L}$ to be defined in other forms, such as Matting Laplacian [Levin et al. 2008a] to generate softer boundary, which still results in a sparse linear system.

5 Solver

Computing s by minimizing Eq. (9) is non-trivial. $\mathbf{W}$ depends on s, making the derivative of $E(s)$ nonlinear to s. We adopt a fixed point iteration strategy to address the non-linearity. In each iteration $t+1$, s is initialized as s^t, estimate from iteration t. Then we minimize

$$E(s^{t+1}) = (s^{t+1} - \mathbf{W}^t g)^T (s^{t+1} - \mathbf{W}^t g) + \lambda s^{t+1,T} \mathbf{L} s^{t+1}. \tag{10}$$

Now the major difficulty is on the evaluation of dense affinity matrix $\mathbf{W}$. Direct computation of $\mathbf{W}$ is difficult since it is not sparse. In addition, the Nyström extension [An and Pellacini 2008; Farbman et al. 2010] that is used to approximate dense affinity cannot be applied here because $\mathbf{W}$ is not necessarily with a low rank. We make $\mathbf{W}^t g$ trackable and further enable efficient computation through a filtering process.

Claim 1. *If g_j exists for pixel j, replacing s_j by it in each iteration enables Shepard's interpolation to efficiently compute $\mathbf{W}^t g$.*

Proof. The replacement procedure can be expressed as

$$\tilde{s}^t_i = \begin{cases} g_i, & \text{if } i \in \Omega \\ s^t_i, & \text{otherwise} \end{cases} \tag{11}$$

Because $\|\tilde{s}^t_i - \tilde{s}^t_j\|^2 = \|\tilde{s}^t_i - g_j\|^2$ for $j \in \Omega$, each element $\mathbf{W}_{ij}$ in $\mathbf{W}$ in iteration t is updated to

$$\mathbf{W}^t_{ij} = \frac{m_j k_{ij} \exp(-\|\tilde{s}^t_i - \tilde{s}^t_j\|^2/\sigma_s)}{\sum_j m_j k_{ij} \exp(-\|\tilde{s}^t_i - \tilde{s}^t_j\|^2/\sigma_s)}. \tag{12}$$

The matrix-vector product $\mathbf{W}^t g$ can be further expanded as

$$\begin{aligned} (\mathbf{W}^t g)_i &= \sum_j \frac{k_{ij} \exp(-\|\tilde{s}^t_i - \tilde{s}^t_j\|^2/\sigma_s) m_j}{\sum_j k_{ij} \exp(-\|\tilde{s}^t_i - \tilde{s}^t_j\|^2/\sigma_s) m_j} g_j \\ &= \frac{\sum_j k_{ij} \exp(-\|\tilde{s}^t_i - \tilde{s}^t_j\|^2/\sigma_s) g_j}{\sum_j k_{ij} \exp(-\|\tilde{s}^t_i - \tilde{s}^t_j\|^2/\sigma_s) m_j} \\ &:= \frac{\sum_j w^t_{ij} g_j}{\sum_j w^t_{ij} m_j} = \frac{\bar{g}^t_i}{\bar{m}^t_i}, \end{aligned} \tag{13}$$

where $w^t_{ij} = k_{ij} \exp(-\|\tilde{s}^t_i - \tilde{s}^t_j\|^2/\sigma_s)$, which is in the same form for both the denominator and nominator.

$$\bar{g}^t_i = \frac{\sum_j w^t_{ij} g_j}{\sum_j w^t_{ij}} \quad \text{and} \quad \bar{m}^t_i = \frac{\sum_j w^t_{ij} m_j}{\sum_j w^t_{ij}}$$

are two intermediate maps. Since m is the characteristic function given its binary indicator form for user input g, Eq. (13) nicely corresponds to Shepard's interpolation [Shepard 1968], which can be efficiently estimated using two filtering passes. □

Claim 2. *Weights w^t_{ij} in Eq. (13) has a similar form as k_{ij} in Eq. (3), but measures a new feature distance.*

Proof. In Eq. (13), weights are expressed as

$$\begin{aligned} w^t_{ij} &= \exp\left(-\|\mathbf{f}_i - \mathbf{f}_j\|^2/\sigma_{\mathbf{f}} - \|\tilde{s}^t_i - \tilde{s}^t_j\|^2/\sigma_s\right) \\ &= \exp\left(-\|\mathbf{f}^t_i - \mathbf{f}^t_j\|^2/\sigma_{\mathbf{f}}\right), \end{aligned} \tag{14}$$

which is in the form of a high-dimensional Gaussian, with the new feature vector $\mathbf{f}^t$ constructed by concatenating $\mathbf{f}$ and $\tilde{s}^t \sqrt{\sigma_{\mathbf{f}}}/\sqrt{\sigma_s}$. □

In Eq. (14), $\mathbf{f}^t$ has more dimensions than the original $\mathbf{f}$, and thus is regarded as a new feature formed in the duration of our optimization. It carries vital information to enhance the selection ability, which will be elaborated on in the next section.

According to Claims 1 and 2, both the nominator $\bar{g}^t$ and denominator $\bar{m}^t$ in Eq. (13) can be computed using high-dimensional Gaussian filtering passes on g and m, respectively. Fast computation can be achieved by adaptive manifold filtering [Gastal and Oliveira 2012], which guarantees smoothly constructed manifolds and avails effective control propagation.

We note that the substitution in Eq. (11) from s_i to g_i only affects a very small set of pixels, due to the sparse nature of user provided samples. Additionally, this type of change tends to enforce $s_j = g_j$ during optimization. This effect is generally desired.

Data Cost Confidence We finally address a data confidence problem in minimizing $E(s^{t+1})$ in Eq. (10). It happens ubiquitously that the denominator $\bar{m}$ contains small values; some are near zeros. For these pixels, we cannot apply division directly. Further, pixels along object boundary typically have noisy $\mathbf{W}^t g$, due to possibly color blending or blurriness. It is not effective to only rely on local smoothness to ameliorate results. We instead introduce a confidence map to attenuate sensitivity to problematic inputs.

Our map d is constructed as follows. The value for each pixel i is set to

$$d^t_i = \ominus \{\bar{m}^t_i > \varepsilon \;\&\; e^t_i = 0\}, \tag{15}$$

where ε is a pre-defined small positive number, e^t is the binary edge map of $\bar{g}^t$, and $\ominus$ is the morphological erosion operator that removes region boundaries for robustness. d^t makes us only use confident non-boundary and numerically reliable pixels. For the remaining ones, we count on local smoothness to correct them.

The practical function to solve in iteration $t+1$ is thus

$$E(s^{t+1}) = (s^{t+1} - \mathbf{W}^t g)^T \mathbf{D}^t (s^{t+1} - \mathbf{W}^t g) + \lambda s^{t+1,T} \mathbf{L} s^{t+1}, \tag{16}$$

where $\mathbf{D}^t$ is a diagonal matrix with $\mathbf{D}^t_{ii} = d^t_i$. The resulting linear systems are obtained by taking derivatives on s^{t+1} and setting them to zeros, expressed as

$$(\mathbf{D}^t + \lambda \mathbf{L}) s^{t+1} = \mathbf{D}^t \mathbf{W}^t g. \tag{17}$$

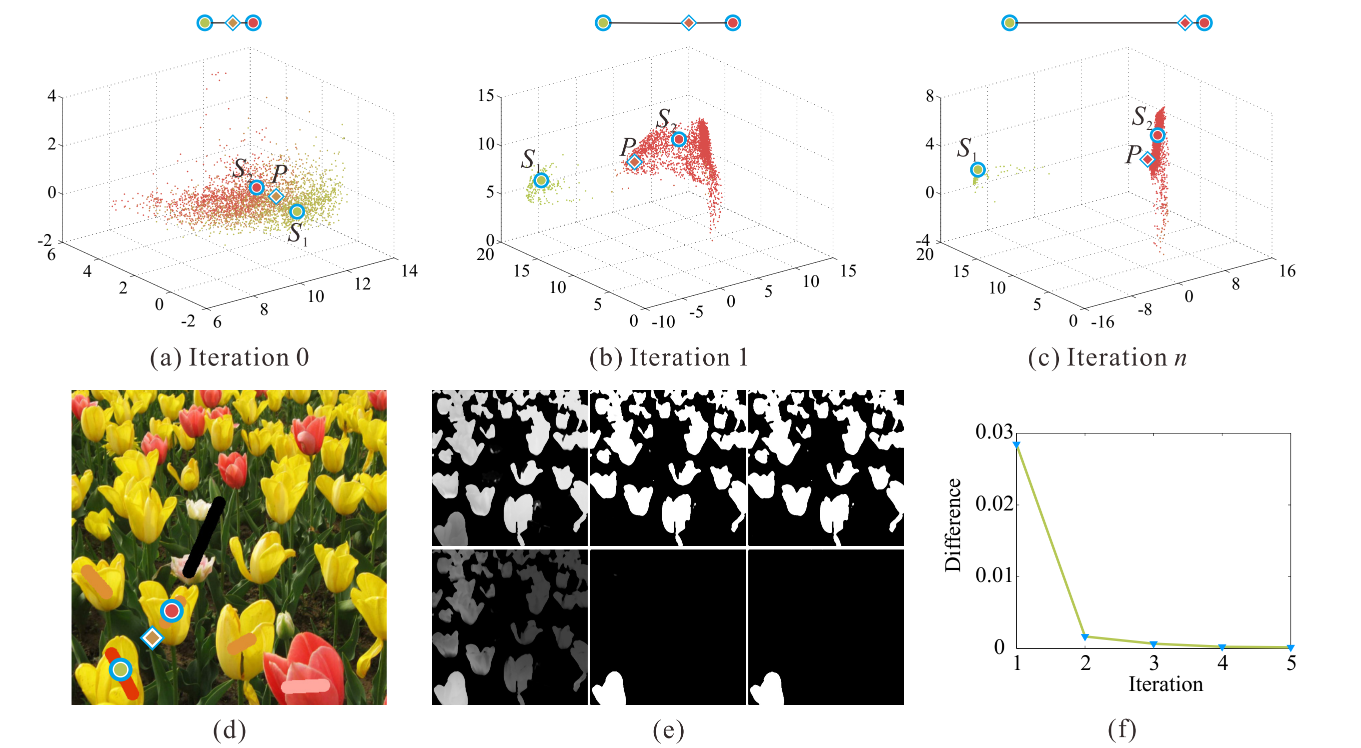


Figure 3: *Feature space illustration. For all pixels in (d), our feature construction actually takes the control samples into consideration, which quickly improves feature discrimination in only three iterations, as shown in (a)-(c). (e) Propagation of two s-regions (in two rows); results in three iterations are shown from left to right. (f) Result difference in every continuous two iterations versus iteration number.*

Algorithm 1 Adaptive Edit Propagation

```
1: initialization: $s^0 \leftarrow 0$, compute $\mathbf{L}$
2: for t=0:2 do
3:     Compute $\bar{g}^t$ and $\bar{m}^t$ using Gaussian filtering.
4:     Apply Shepard interpolation: $\mathbf{W}^t g = \bar{g}^t / \bar{m}^t$.
5:     Construct diagonal matrix $\mathbf{D}^t$ based on $\bar{m}^t$ and $\bar{g}^t$.
6:     Solve linear system Eq. (17) to obtain $s^{t+1}$.
7: end for
8: output: $s^3$
```

As $\mathbf{D}^t$ is diagonal and $\mathbf{L}$ is sparse, $(\mathbf{D}^t + \lambda \mathbf{L})$ is computed directly. $\mathbf{D}^t \mathbf{W}^t g$ is estimated by Gaussian filtering. The overall linear system is also sparse and can be solved using standard approaches. We adopt the pre-conditioned conjugate gradient (PCG). The procedure is sketched in Algorithm 1.

6 Understanding Adaptive Feature Space

Why the proposed method works can be understood from an adaptive feature space point of view. Looking at the algorithm again, in Claim 2, the final effective affinity is actually constructed in feature space $\mathbf{f}^t = (\mathbf{f}, \sqrt{\sigma_\mathbf{f}}/\sqrt{\sigma_s}\tilde{s}^t)^T$. Each pixel i corresponds to a feature $\mathbf{f}^t_i$, which expands $\mathbf{f}_i$ introduced in Eq. (3). The extra dimension $\sqrt{\sigma_\mathbf{f}}/\sqrt{\sigma_s}\tilde{s}^t$ enhances the ability to correctly construct the resulting map from only sparse samples because it involves both user input control confidence and propagation result in previous iterations.

In the beginning, all $\tilde{s}$ elements are initialized to zeros; so the feature elements in $\mathbf{f}$ dominate computation. Since iteration 1, the feature space evolves with the involvement of estimates and samples $\tilde{s}^t$, quickly shaping the solver to differentiate edits.

We give a demonstration in Fig. 3, which contains features drawn from the image in (d). This example is also included in Fig. 1. The 3D space in (a)-(c) is created by performing PCA and selecting only dominant dimensions. Initially in (a), points are not separable although S_1 and S_2 are samples with different labels. It is because all these three pixels originally have very similar yellow colors. In iteration one (b), due to incorporation of elements $\tilde{s}$ into the feature vector $\mathbf{f}^t$, two clusters are rapidly formed. Only three iterations make the system converge where S_2 and P are formed in the same red cluster and S_1 is in another. The large distance between the two feature clusters manifests the usefulness of our new feature space construction and evolvement. Although the original color P is similar to both S_1 and S_2, it does not fall blending of controls on S_1 and S_2 due to the adaptive feature space as well as spatial consideration. Instead, it finds the optimal color on S_2 and updates its result automatically. This goal cannot be accomplished by previous approaches with a similar level of controls.

The influence maps s for red and orange controls in iterations are shown respectively in the two rows in (e). The bottom left tulip is quickly separated from others due to its unique adjustment to turn red. The difference of estimates in consecutive iterations is plotted in (f), bearing out fast convergence.

7 Discussion

Main computation is spent on the high-dimensional Gaussian filtering by adaptive manifold and the sparse linear solver. The filtering complexity is $O(nZ)$, where n is the feature dimension and Z is the pixel number, linear to the size of input. The two operations to compute $\bar{w}^t$ and $\bar{g}^t$ can be achieved using one filtering pass by concatenating $\bar{g}^t$ and $\bar{w}^t$, since the filtering weights are the same for the two processes. The linear system solver complexity counting in

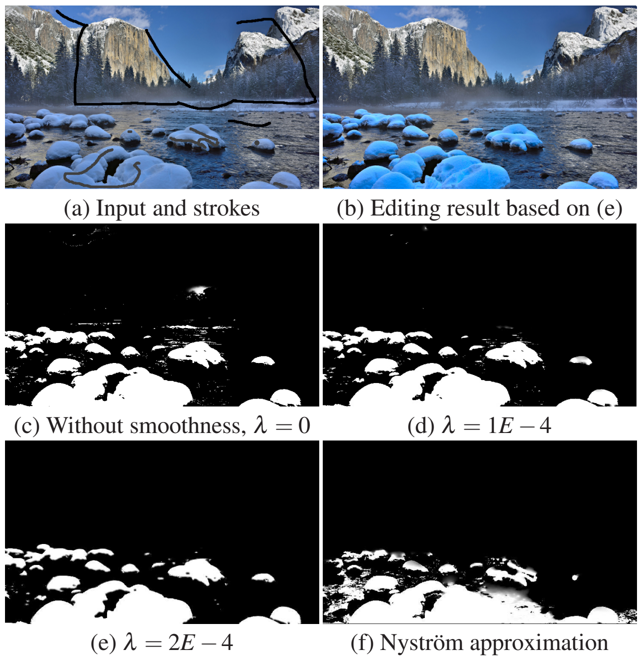

(a) Input and strokes
(b) Editing result based on (e)
(c) Without smoothness, $\lambda = 0$
(d) $\lambda = 1E-4$
(e) $\lambda = 2E-4$
(f) Nyström approximation

Figure 4: *Smoothness strength and comparison with Nyström approximation. (b) Editing based on s estimate in (e). (c)-(e) s result comparison with different smoothness strength. (f) The influence map with Nyström approximation.*

sparse Laplacian is also linear to the pixel number, making the proposed method flexibly scalable. Our Matlab implementation spends 10 seconds to process an 800×600 color image on a PC with an Intel i7 CPU and 8G memory. High-dimensional Gaussian filtering consumes most time (9 seconds), which can actually be much accelerated to achieve realtime performance using GPU [Gastal and Oliveira 2012]. So overall near-realtime processing can be expected using C or GPU acceleration.

Generalization of Previous Methods The proposed framework can be deemed as unification and generalization of previous local and all-pair propagation approaches. Specifically, when the influence control weight $\sigma_\mathbf{f}$ in Eq. (3) approaches zero, denominator $\bar{w}$ in Eq. (13) has its most elements approaching zeros except for the control samples. In sequel, only user specified inputs are kept in $\mathbf{D}$, which is exactly the case of local editing [Lischinski et al. 2006]. On the contrary, if σ_s approaches infinity, the influence of $\bar{s}^t$ is reduced. $\mathbf{W}^t g$ therefore carries the same information as all-pair propagation [An and Pellacini 2008]. Our method, in this regard, bridges originally different editing methods and generalizes them to count in both user input and image structure.

While global and local methods can either use sparse samples or impose accurate control, these benefits cannot be easily inherited in one system. Our strategy is the first attempt to achieve this goal with adaptive interaction range that discourages conflicting edits to interfere with each other in the efficient propagation.

Parameter Adjustment Basically, all propagation approaches share the similar batch of parameters in determining range/spatial influence and smoothness strength. If we do not incorporate the smoothness term, solving the linear system is no longer needed and our method simplifies to an iterative Shepard method. It is also a scatter data interpolation approach in the adaptive feature space. Fig. 4 shows the s estimates for one label using and without using the smoothness term given the input in (a). The improvement from (c) to (e) is due to the consideration of neighboring information. The final edit result is shown in (b).

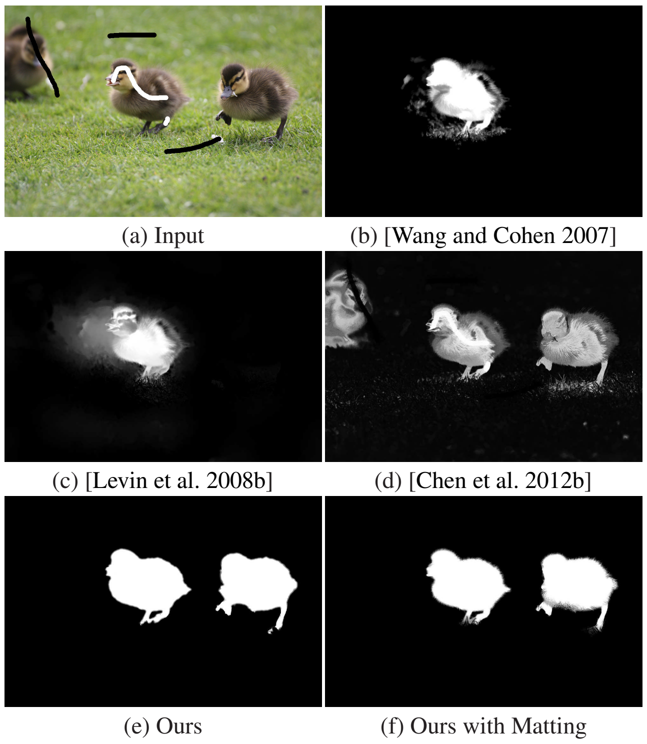


Figure 5: *Matting result comparison under sparse strokes. Input image courtesy of flicker user "f1uffster (Jeanie)". Our results in (e) and (f) are produced with smoothness terms set as WLS and matting Laplacian respectively. They only differ on boundary hairiness.*

Our framework is also general enough that it could be applied to image matting and many applications beyond it. First, our method is able to handle scalers, vectors, or combined labels in s, instead of only two values to indicate foreground and background. Second, our method is alterable to adopt various advanced smoothness terms to preserve edges [Levin et al. 2004; Lischinski et al. 2006] or generate soft mattes [Levin et al. 2008b]. Result comparisons are presented in Fig. 5, explained in Section 8.

Nyström Extension While Nyström extension [Fowlkes et al. 2004] was adopted to efficiently evaluate the affinity matrix [An and Pellacini 2008; Farbman et al. 2010], it is not suitable for computing our $\mathbf{W}$ that is not with a low rank. We have experimented with Nyström extension to approximate our $\mathbf{W}$. The result quality is reduced. An example is shown in Fig. 4(f) using 500 samples.

Relation to ScribbleBoost ScribbleBoost [Li et al. 2008] relies on explicit pixel classification using Gentle Boost. It in essence trains a boosting classifier using a small number of samples in the original feature space. Contrarily, our method iteratively adapts the feature space, which leverages structural and user editing information. Comparisons are given in Section 8.

8 Experiments

We first compare our influence map results with other alternatives based on sparse editing samples and then show a few effects created on images and videos.

Comparison with Matting Several matting methods accept sparse strokes to mark foreground and background colors. Recent approaches, such as KNN matting, can also extract multiple

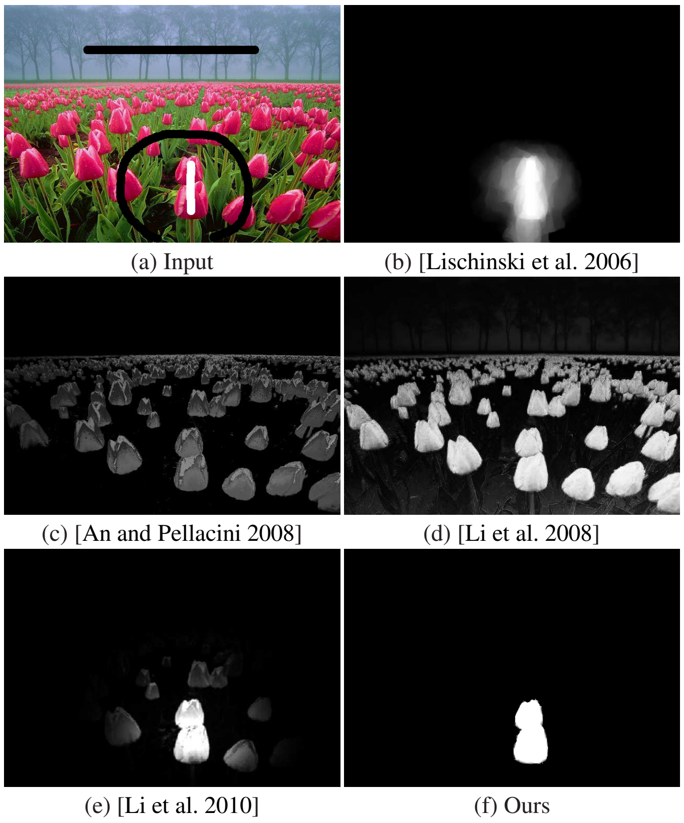


Figure 6: *Comparison with other edit propagation approaches. Image and strokes are provided in [Li et al. 2010].*

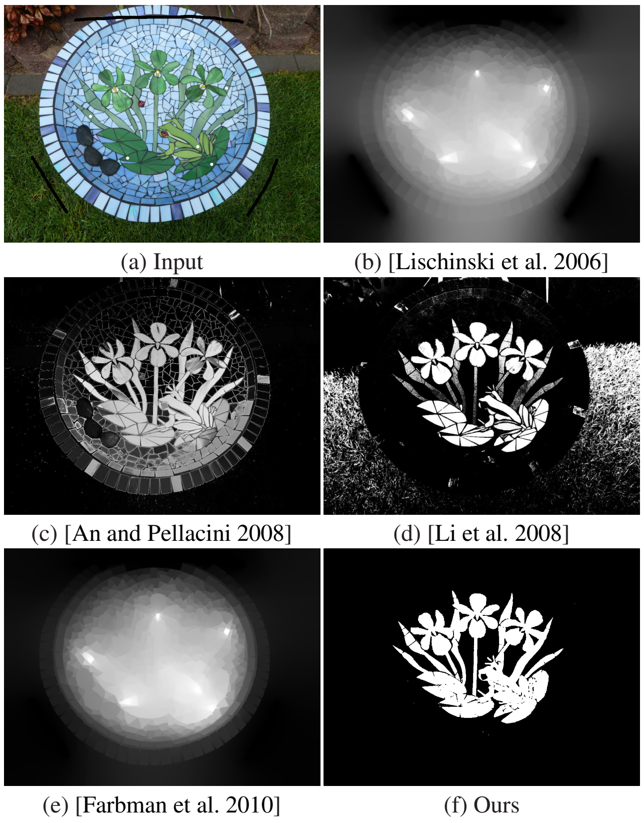

(a) Input
(b) [Lischinski et al. 2006]
(c) [An and Pellacini 2008]
(d) [Li et al. 2008]
(e) [Farbman et al. 2010]
(f) Ours

Figure 7: *Another image example for edit propagation comparison. Input image courtesy of flicker user "Joolz21 (julie)".*

foreground regions. In Fig. 5, we compare our result with those produced by robust matting [Wang and Cohen 2007] (b), spectrum matting [Levin et al. 2008b] (c), and KNN matting [Chen et al. 2012b] (d), which have their implementation publicly available. Because most matting methods are local-sample based, to produce decent results, a few more strokes should be drawn close to the object boundary. KNN matting produces the result in (d), extracting multiple objects simultaneously. It however shares the same limitation as all-pair edit propagation – the similar appearance makes the user indication to exclude the left-most chicken be ignored.

Our results are shown in (e) and (f), respectively produced incorporating the simple weighted least square (WLS) Laplacian and matting Laplacian as smoothness constraints. Matting Laplacian creates hairier boundary. More comparisons with state-of-the-art matting methods are provided in the project website.

Comparison with Edit Propagation We compare our strategy also with representative local and all-pair edit propagation algorithms [Lischinski et al. 2006; An and Pellacini 2008], ScribbleBoost [Li et al. 2008], fast propagation with interpolation [Li et al. 2010], and diffusion distance method [Farbman et al. 2010]. The results are shown in Figs. 6-7.

Since local approaches employ sparse data constraints, results are affected by local smoothness constraints, yielding soft and smooth boundary when sparse strokes are provided, as shown in (b) in the two figures. All-pair approaches have different types of strokes that affect each other, as shown in (c). ScribbleBoost is effective to propagate strokes across texture and boundary. But positive and negative training samples with similar appearance confuse each other. Sparse training samples also increase the ambiguity.

Image Editing Colorization can be immediately benefitted from our system. Fig. 8 shows one example with inputs in (a). To colorize haystacks differently from the land, local adjustment must be applied, which however requires strokes on all trees to mark them green. Results of state-of-the-art methods are shown in (b)-(d), with close-ups in the second row. The first two approaches face the difficulty to propagate strokes across texture and among different trees. The global method can colorize all trees but experiences color mixing artifacts. Our result is presented in (e).

A few other image editing results by changing color and tone are shown in Fig. 9. As the quality of these image editing tasks depends primarily on the correctness of influence maps of user strokes, we compare our influence maps ((c), (g), and (k)) with those produced by other approaches ((b), (f), and (j)) based on the same input. We note that local methods can possibly produce similar results if dense strokes are provided.

Our method also applies to interactive intrinsic image decomposition. Let the user strokes specify seed pixels that have the same reflectance, optimizing Eq. (9) yields selections that group pixels under the similar reflectance condition. The final reflectance is then estimated for each selection group. With the input in Fig. 10(a), our method produces the result in (b), quality of which is comparable to that shown in (d) (result of Bousseau et al. [2009]). It is notable the latter method has incorporated more user strokes and a larger variety of constraints in estimating the intrinsic image.

Video Editing Our method greatly benefits video editing due to its scalability and effectiveness to use a small amount of user input. The adaptive manifold for high-dimensional Gaussian filtering can be separably applied for each dimension x, y, and t. Its extension

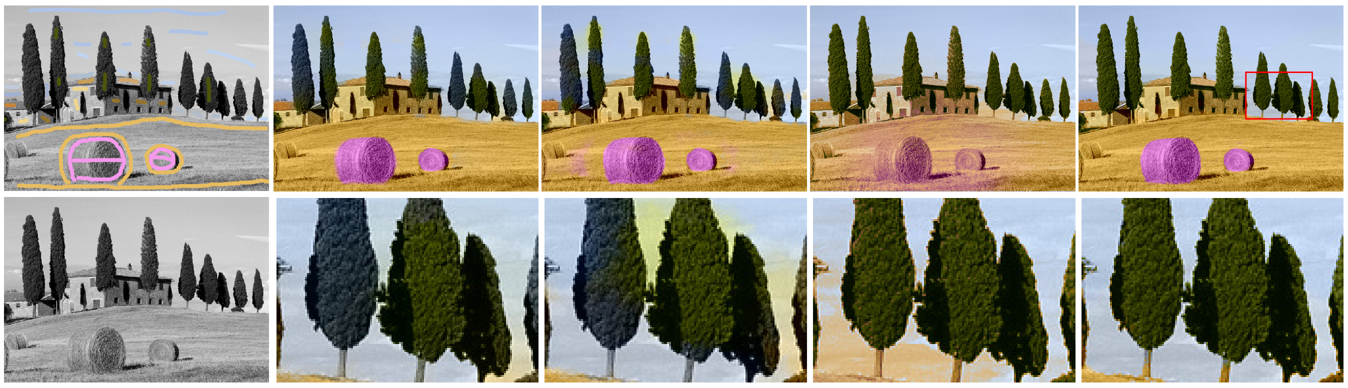
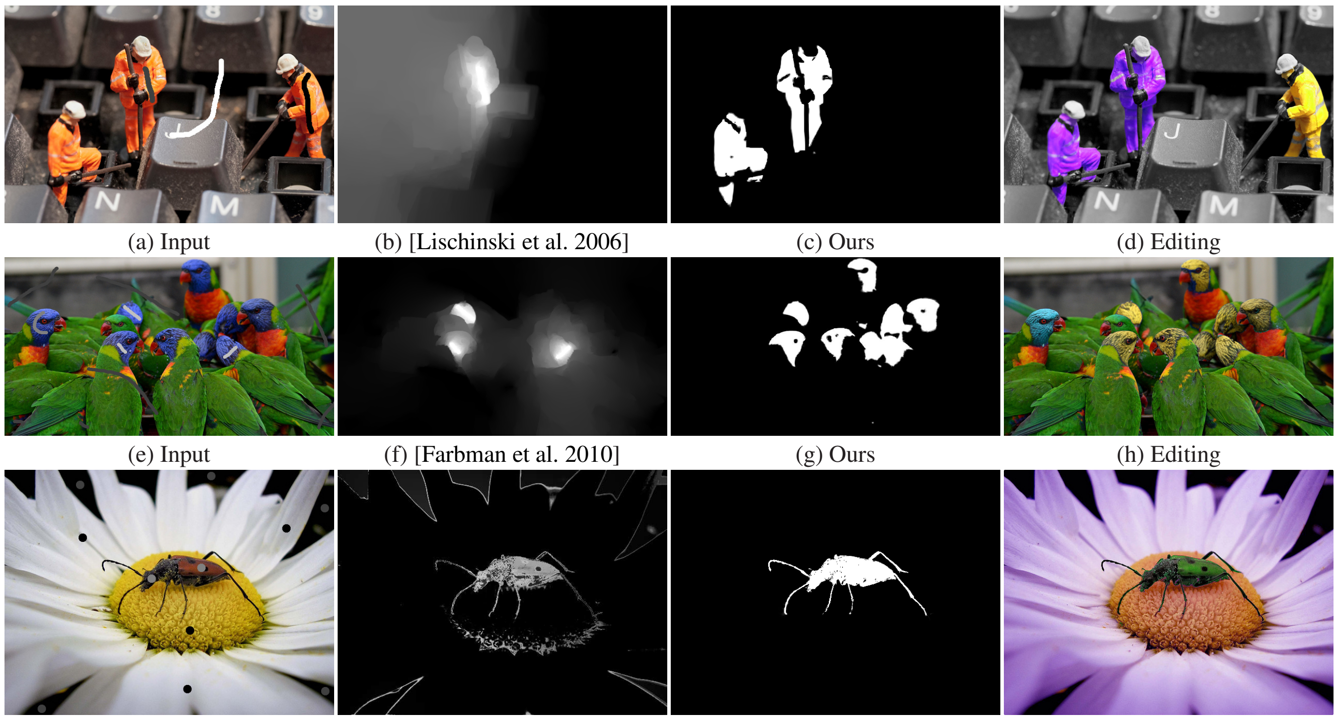

(a) (b) [Levin et al. 2004] (c) [Fattal 2009] (d) [An and Pellacini 2008] (e) Ours

Figure 8: *A colorization example. (a) Grayscale input and color strokes. (b)-(d) Results of [Levin et al. 2004], edge-avoid wavelet (EAW) [Fattal 2009], and the global method [An and Pellacini 2008]. (e) Our result.*


(a) Input (b) [Lischinski et al. 2006] (c) Ours (d) Editing

(e) Input (f) [Farbman et al. 2010] (g) Ours (h) Editing

(i) Input (j) [An and Pellacini 2008] (k) Ours (l) Editing

Figure 9: *Comparisons and effects. Sparse control samples make the problem challenging to solve. Please see the images in their original resolutions. More comparisons are in our project website. Input images (e)(i) courtesy of flicker users "RUMTIME" and "photogirl7.1".*

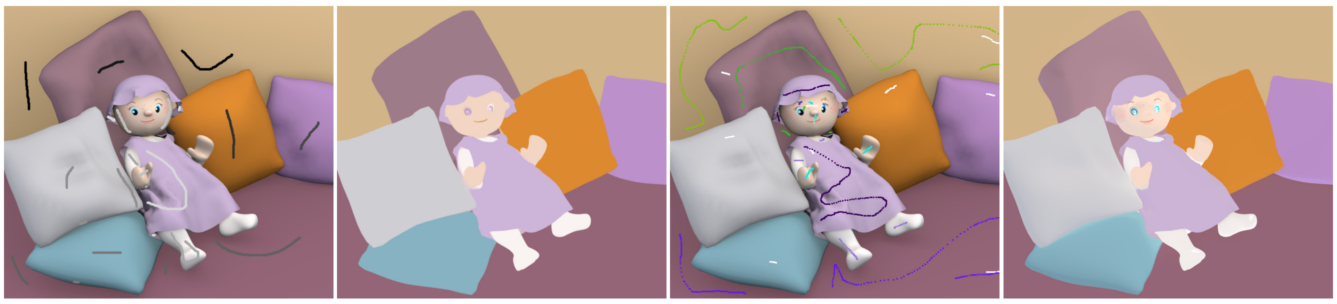

(a) Our input (b) Our result (c) Input [Bousseau et al. 2009] (d) Result [Bousseau et al. 2009]

Figure 10: *Intrinsic Image Decomposition. Our method can be used to find pixels with similar reflectance.*

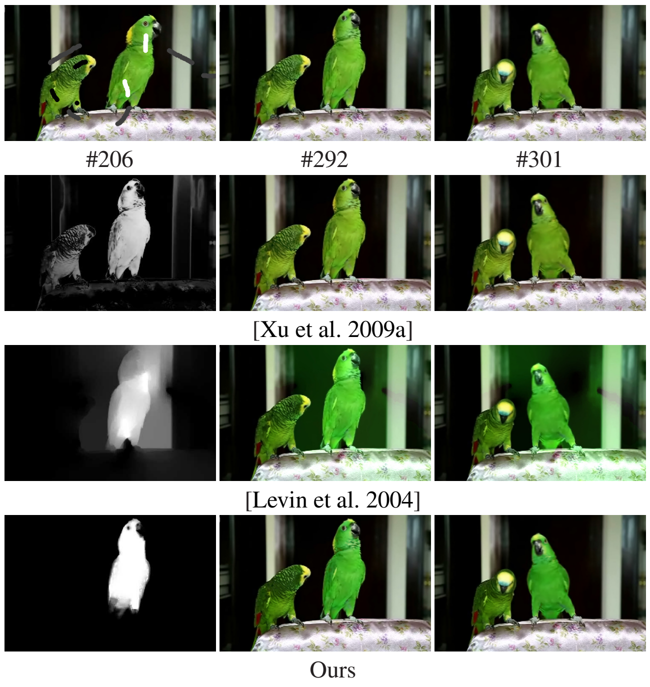


Figure 11: *A video editing example. We assign different colors to the two parrots. Strokes are only drawn on frame #206. The influence s maps for the right parrot in frame #292 are shown in the first column. Editing results are compared in the right two columns.*

to video is therefore straightforward. While there is no problem to solve a very large sparse linear system considering the 8-neighbors in the spatial-temporal space, to save memory, we only define a spatial 4-neighbor smoothness constraint, making the solver efficient.

We show in Fig. 11 a video editing example, in which we separately tune color of the two parrots with user control in only one frame #206. The full sequence is included in the project website. We compare our result (bottom) with all-pair propagation [Xu et al. 2009a] and local method [Levin et al. 2004] without estimating motion between frames for simplicity's sake. Note for video editing, intensive user control on many frames is generally hard to create. It is expected better results and more effects can be generated when temporal correspondence is taken into consideration.

Feature Space As elaborated on in previous sections, our method can flexibly employ features other than those only considering color and image coordinates. We present a focus-defocus example.

Fig. 12(a) is a heron crane image with defocused background. It is not easy to wholly select the crane due to its complex color, not to mention only using sparse strokes shown in (a). Two matting results by previous methods are shown in (b) and (c). We employ two metrics for distinguishing between blurry and non-blurry regions similar to those in [Liu et al. 2008]. The local power spectrum slope (LPSS) metric makes use of natural image statistics in Fourier domain. Its value for blurry regions is large, as illustrated in (d). Another metric is local Laplacian of Gaussian (LoG) response, as shown in (e). Sharper structures are with more high-frequency details and therefore yield larger LoG responses. For a non-blurry object, LoG response is large on edges while LPSS is small inside the body, complementing each other. Our final result is shown in (f), based on the new feature vector concatenating these two metrics and 2D spatial coordinates for each pixel.

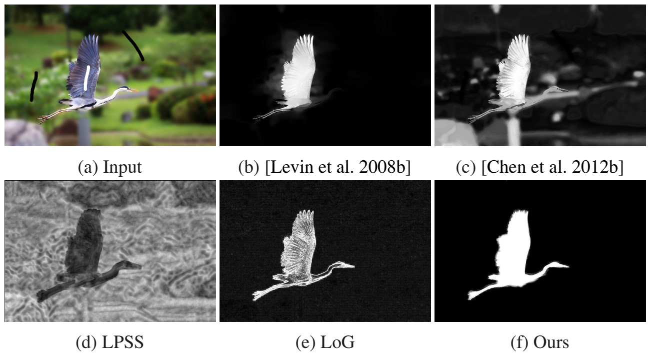


Figure 12: *Separating objects based on focused and out-of-focus pixel information. It is achieved in our method with the blur-aware features given the sparse editing samples.*

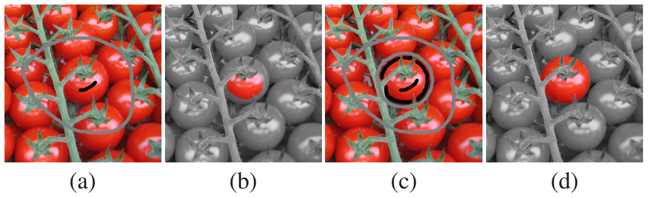


Figure 13: *A difficult example. Separating one tomato from others with similar structure and color appearance still needs deliberative stroke drawing.*

9 Concluding Remarks

We have presented a general control propagation framework only requiring a small number of input samples. It resolves the major ambiguity and possible conflict brought by sparse controls in an optimal way. Our solver companying the model achieves efficient and theoretically sound optimization. This method, as a fundamental step, can be substituted into many applications requiring guidance samples. In challenging sparse-sample cases, the advantage of our method is prominent.

Limitations Complex and very detailed editing effects may increase the need to draw more strokes. We show one very challenging example in Fig. 13 where editing only one tomato in a group of them with similar appearance demands carefully drawn samples. Note that this example fails all global methods no matter how the samples are placed. With the strokes shown in (c), a visually plausible result is produced in (d) by our method.

Acknowledgements

We thank Shicheng Zheng, Qi Zhang, and Xiaoyong Shen for their help to generate several results for the image and video editing applications. We also thank flicker users "f1uffster (Jeanie)", "Joolz21 (julie)", "RUMTIME" and "photogirl7.1" for the pictures used in the paper. This work is supported by a grant from the Research Grants Council of the Hong Kong SAR (project No. 413110) and by NSF of China (key project No. 61133009).

References

ADAMS, A., GELFAND, N., DOLSON, J., AND LEVOY, M. 2009. Gaussian kd-trees for fast high-dimensional filtering. *ACM Trans. Graph. 28*, 3.

ADAMS, A., BAEK, J., AND DAVIS, M. A. 2010. Fast high-dimensional filtering using the permutohedral lattice. *Comput.*

Graph. Forum 29, 2, 753–762.

AN, X., AND PELLACINI, F. 2008. Appprop: all-pairs appearance-space edit propagation. *ACM Trans. Graph. 27*, 3.

BAE, S., PARIS, S., AND DURAND, F. 2006. Two-scale tone management for photographic look. *ACM Trans. Graph. 25*, 3, 637–645.

BOUSSEAU, A., PARIS, S., AND DURAND, F. 2009. User-assisted intrinsic images. *ACM Trans. Graph. 28*, 5.

BOYADZHIEV, I., BALA, K., PARIS, S., AND DURAND, F. 2012. User-guided white balance for mixed lighting conditions. *ACM Trans. Graph. 31*, 6, 200.

CHEN, J., PARIS, S., AND DURAND, F. 2007. Real-time edge-aware image processing with the bilateral grid. *ACM Trans. Graph. 26*, 3, 103.

CHEN, Q., LI, D., AND TANG, C.-K. 2012. Knn matting. In *CVPR*, 869–876.

CHEN, X., ZOU, D., ZHAO, Q., AND TAN, P. 2012. Manifold preserving edit propagation. *ACM Trans. Graph. 31*, 6, 132.

CRIMINISI, A., SHARP, T., ROTHER, C., AND PÉREZ, P. 2010. Geodesic image and video editing. *ACM Trans. Graph. 29*, 5, 134.

DURAND, F., AND DORSEY, J. 2002. Fast bilateral filtering for the display of high-dynamic-range images. *ACM Trans. Graph. 21*, 3, 257–266.

FARBMAN, Z., FATTAL, R., LISCHINSKI, D., AND SZELISKI, R. 2008. Edge-preserving decompositions for multi-scale tone and detail manipulation. *ACM Trans. Graph. 27*, 3.

FARBMAN, Z., FATTAL, R., AND LISCHINSKI, D. 2010. Diffusion maps for edge-aware image editing. *ACM Trans. Graph. 29*, 6, 145.

FATTAL, R., AGRAWALA, M., AND RUSINKIEWICZ, S. 2007. Multiscale shape and detail enhancement from multi-light image collections. *ACM Trans. Graph. 26*, 3, 51.

FATTAL, R. 2009. Edge-avoiding wavelets and their applications. *ACM Trans. Graph. 28*, 3.

FOWLKES, C., BELONGIE, S., CHUNG, F. R. K., AND MALIK, J. 2004. Spectral grouping using the nyström method. *IEEE Trans. Pattern Anal. Mach. Intell. 26*, 2, 214–225.

GASTAL, E. S. L., AND OLIVEIRA, M. M. 2011. Domain transform for edge-aware image and video processing. *ACM Trans. Graph. 30*, 4, 69.

GASTAL, E. S. L., AND OLIVEIRA, M. M. 2012. Adaptive manifolds for real-time high-dimensional filtering. *ACM Trans. Graph. 31*, 4, 33.

KASS, M., AND SOLOMON, J. 2010. Smoothed local histogram filters. *ACM Trans. Graph. 29*, 4.

KOPF, J., COHEN, M. F., LISCHINSKI, D., AND UYTTENDAELE, M. 2007. Joint bilateral upsampling. *ACM Trans. Graph. 26*, 3, 96.

KRÄHENBÜHL, P., AND KOLTUN, V. 2011. Efficient inference in fully connected crfs with gaussian edge potentials. In *NIPS*.

LANG, M., WANG, O., AYDIN, T., SMOLIC, A., AND GROSS, M. H. 2012. Practical temporal consistency for image-based graphics applications. *ACM Trans. Graph. 31*, 4, 34.

LEE, P., AND WU, Y. 2011. Nonlocal matting. In *CVPR*, 2193–2200.

LEVIN, A., LISCHINSKI, D., AND WEISS, Y. 2004. Colorization using optimization. *ACM Trans. Graph. 23*, 3, 689–694.

LEVIN, A., LISCHINSKI, D., AND WEISS, Y. 2008. A closed-form solution to natural image matting. *IEEE Trans. Pattern Anal. Mach. Intell. 30*, 2, 228–242.

LEVIN, A., RAV-ACHA, A., AND LISCHINSKI, D. 2008. Spectral matting. *IEEE Trans. Pattern Anal. Mach. Intell. 30*, 10, 1699–1712.

LI, Y., ADELSON, E. H., AND AGARWALA, A. 2008. Scribbleboost: Adding classification to edge-aware interpolation of local image and video adjustments. *Comput. Graph. Forum 27*, 4, 1255–1264.

LI, Y., JU, T., AND HU, S.-M. 2010. Instant propagation of sparse edits on images and videos. *Comput. Graph. Forum 29*, 7, 2049–2054.

LISCHINSKI, D., FARBMAN, Z., UYTTENDAELE, M., AND SZELISKI, R. 2006. Interactive local adjustment of tonal values. *ACM Trans. Graph. 25*, 3, 646–653.

LIU, R., LI, Z., AND JIA, J. 2008. Image partial blur detection and classification. In *CVPR*.

PARIS, S., AND DURAND, F. 2006. A fast approximation of the bilateral filter using a signal processing approach. In *ECCV (4)*, 568–580.

PARIS, S., HASINOFF, S. W., AND KAUTZ, J. 2011. Local laplacian filters: Edge-aware image processing with a laplacian pyramid. *ACM Trans. Graph. 30*, 4, 68.

PELLACINI, F., AND LAWRENCE, J. 2007. Appwand: editing measured materials using appearance-driven optimization. *ACM Trans. Graph. 26*, 3, 54.

RAGAN-KELLEY, J., ADAMS, A., PARIS, S., LEVOY, M., AMARASINGHE, S., AND DURAND, F. 2012. Decoupling algorithms from schedules for easy optimization of image processing pipelines. *ACM Trans. Graph. 31*, 4, 32.

RHEMANN, C., ROTHER, C., WANG, J., GELAUTZ, M., KOHLI, P., AND ROTT, P. 2009. A perceptually motivated online benchmark for image matting. In *CVPR*, 1826–1833.

ROTHER, C., KOLMOGOROV, V., AND BLAKE, A. 2004. "grabcut": interactive foreground extraction using iterated graph cuts. *ACM Trans. Graph. 23*, 3, 309–314.

SHEPARD, D. 1968. A two-dimensional interpolation function for irregularly-spaced data. In *ACM national conference*, 517–524.

SUBR, K., SOLER, C., AND DURAND, F. 2009. Edge-preserving multiscale image decomposition based on local extrema. *ACM Trans. Graph. 28*, 5.

WANG, J., AND COHEN, M. F. 2005. An iterative optimization approach for unified image segmentation and matting. In *ICCV*, 936–943.

WANG, J., AND COHEN, M. F. 2007. Optimized color sampling for robust matting. In *CVPR*.

XU, K., LI, Y., JU, T., HU, S.-M., AND LIU, T.-Q. 2009. Efficient affinity-based edit propagation using k-d tree. *ACM Trans. Graph. 28*, 5.

XU, K., WANG, J., TONG, X., HU, S.-M., AND GUO, B. 2009. Edit propagation on bidirectional texture functions. *Comput. Graph. Forum 28*, 7.

XU, L., LU, C., XU, Y., AND JIA, J. 2011. Image smoothing via L0 gradient minimization. *ACM Trans. Graph. 30*, 6.

XU, L., YAN, Q., XIA, Y., AND JIA, J. 2012. Structure extraction from texture via relative total variation. *ACM Trans. Graph. 31*, 6, 139.