EUROGRAPHICS 2013 / I. Navazo, P. Poulin
(Guest Editors)

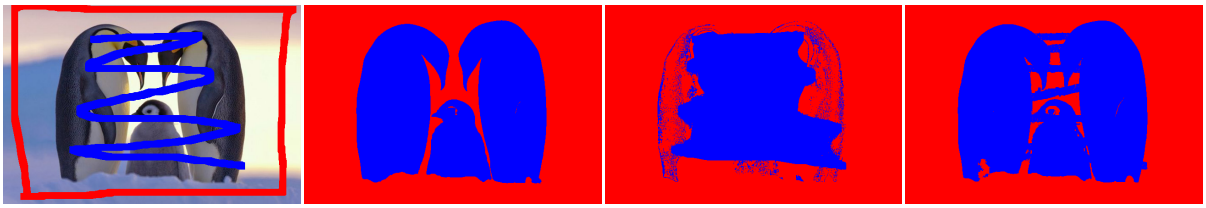
Volume 32 (2013), Number 2

Accurate Binary Image Selection from Inaccurate User Input

Kartic Subr[1], Sylvain Paris[2], Cyril Soler[3] and Jan Kautz[1]

[1]University College London [2]Adobe Research [3]INRIA Grenoble

Author's Version



(a) input image + scribbles (b) our result (c) using diffusion maps (d) using KNN matting

Figure 1: *We present a technique to make binary selections in images, such as to select the three penguins, using inaccurate scribbles to indicate foreground (blue) and background (red). Unlike existing approaches, our approach does not assume that the indications are* 100% *accurate. Since the related work, diffusion maps [FFL10] as well as KNN matting [CLT12], produce fuzzy selections, we manually thresholded their results to achieve the best possible selections.*

Abstract

Selections are central to image editing, e.g., they are the starting point of common operations such as copy-pasting and local edits. Creating them by hand is particularly tedious and scribble-based techniques have been introduced to assist the process. By interpolating a few strokes specified by users, these methods generate precise selections. However, most of the algorithms assume a 100% accurate input, and even small inaccuracies in the scribbles often degrade the selection quality, which imposes an additional burden on users. In this paper, we propose a selection technique tolerant to input inaccuracies. We use a dense conditional random field (CRF) to robustly infer a selection from possibly inaccurate input. Further, we show that patch-based pixel similarity functions yield more precise selection than simple point-wise metrics. However, efficiently solving a dense CRF is only possible in low-dimensional Euclidean spaces, and the metrics that we use are high-dimensional and often non-Euclidean. We address this challenge by embedding pixels in a low-dimensional Euclidean space with a metric that approximates the desired similarity function. The results show that our approach performs better than previous techniques and that two options are sufficient to cover a variety of images depending on whether the objects are textured.

1. Introduction

Marking specific pixels in an image as selected is an indispensable task, serving as the starting point for many image-editing operations such as background replacement, colour and tone manipulation, and copy-pasting. Obtaining a binary mask that is accurate at the pixel level by manual selection is an arduous task. Several techniques have been proposed to assist this process. For instance, the Magnetic Lasso [MB95] "snaps" the user input to the closest edge as users delineate the object. However, this still imposes much tedium on users who have to carefully mark the object boundaries. Scribble-based approaches have been proposed to alleviate this difficulty, e.g., [BJ01, ADA*04, LSS09]. Users perform a few strokes over the object of interest, a.k.a. foreground, and a few more strokes to indicate the background. The system then solves a classification problem to mark each pixel as foreground or background. The advantage of this approach is that it requires a much simpler input from users. They do not have to carefully paint the selection or delineate its boundary. Instead, they only indicate a few regions on the foreground and background, such as in Fig. 1a, and the algorithm infers an accurate selection. However, most existing techniques assume that the user scribbles are perfectly accurate, that is, that they cover only foreground pixels or only

background pixels. When this is not the case, i.e., if users introduce a scribble that crosses the foreground-background boundary, the quality of the produced selection degrades significantly (see Fig. 1c and 1d). This imposes a strong constraint on the user input to be 100% accurate in order to get the best possible result. This can be a serious impediment in several cases. For instance, objects of interest with thin parts may be difficult to mark accurately; or touch interfaces and mobile devices can make it difficult to obtain accurate scribbles. To remedy this problem, we introduce an algorithm that generates accurate binary selections even from inaccurate user inputs, as in Fig. 1b.

Our approach consists of three main components. First, we remark that in many existing techniques, user scribbles are used as boundary conditions when solving the foreground-background classification problem [BJ01, ADA*04,LSS09]. That is, the user-provided labels cannot be changed by the solver, which makes the process rely on the user ability to make perfectly accurate markings and gives it no chance to recover from erroneous scribbles. In contrast, our approach builds upon a dense conditional random field (CRF) that uses the user input as an initial guess while still allowing for corrections based on the image content (§ 2). Second, we observe that simple pixel-to-pixel color similarity only weakly discriminates between different regions. We address this shortcoming with improved similarity functions that consider pixels and their neighborhoods. We show that the latter are useful in discriminating regions of similar colors but different textures. However, because distance functions based on large pixel neighborhoods manipulate high-dimensional data, they make solving a dense CRF problem impractical. Furthermore, some of these functions are not based on a Euclidean metric, e.g., the χ^2 test between patch histograms, which precludes the use of efficient algorithms [KK11]. We overcome this difficulty by efficiently embedding the pixel data into a low-dimensional Euclidean space with a metric that approximates the desired similarity functions (§ 2).

We validate our approach on a variety of images (§ 3). We show that our algorithm consistently produces accurate selections even when the input scribbles are partly inaccurate. In comparison, the accuracy of the selections produced by previous work quickly degrades as soon as some regions are incorrectly labeled. We confirm this trend using ground truth data that allows us to quantify the impact of inaccurate scribbles on the output selection.

Contributions The main contribution of this work is an algorithm to produce binary image selections that is robust to input accuracies. To achieve this goal, we make the following technical contributions:

- We characterize the use of user scribbles as hard constraints as the source of the inability of most existing techniques to cope with inaccurate input. We overcome this issue by solving a dense CRF.
- We study several similarity functions and show that the ones based on patches yield more accurate selections.
- We enable the use of high-dimensional, non-Euclidean distance metrics by efficiently embedding them into a low-dimensional, Euclidean space.

The problem of generating binary selections is related to the *matting* problem that seeks to estimate the transparency of each pixel [RRW*09]. In our result section, we show that even state-of-the-art matting algorithms perform poorly at binary selection when the input is not accurate. Conversely, we acknowledge that our approach, as all binary selection techniques, is not appropriate to select transparent objects.

1.1. Related Work

Several techniques have been proposed to improve over the simple brush and lasso. For instance, the magnetic lasso snaps the selection to the nearest edge [MB95] and the edge-aware brush stops at color discontinuities [CPD07, OH08]. However, these tools still require tedious work from users to completely mark the selected region or its boundary. Selection techniques for multi-touch screens [BWB06] improve accuracy when clicking interface elements such as buttons and check-boxes are available.

An alternative workflow is to let users make a few discrete marks and have an algorithm interpolate these indications to infer a complete selection. For instance, GrabCut [RKB04] generates a selection from a user-provided bounding box. While such minimal input is sufficient in some cases, it is often useful to have users also indicate where the background is. This is typically done with scribbles, that is, users make strokes to mark a few foreground and background regions. Our paper focuses on this scenario. Among these techniques, we distinguish two categories. Most methods consider the user scribbles fully reliable and use them as hard constraints, e.g., [BJ01, ADA*04, LSS09, LAA08], which is also commonly used in the context of tonal adjustments [LFUS06] or colorization [LLW04]. As we shall see, this strategy yields good results as long as the input scribbles are accurate but the selection quality degrades when it is not the case. In contrast, scribbles in AppProp [AP08] and Instant Propagation [LJH10] are soft constraints and the optimization can override the user-specified labels, which allows it to correct input errors. However, these techniques target soft selections and do not perform as well in the binary case as we will see. In comparison, we formulate the problem as a dense CRF and uses the recent method developed by Krähenbühl and Koltun [KK11] to solve it efficiently, which produce significantly better selections as shown in our experiments.

A few techniques account for the possibility of errors. For instance, Tao et al. [TJP10] propose an algorithm for seamless compositing that is robust to erroneous selections. In comparison, our approach seeks to avoid such selection at the first place. Lazy Snapping [LTS04] describes tools

to correct the selection if the results from scribbles only is not accurate. This is complementary to our approach that focuses on producing a good selection from the scribbles even if they are not accurate.

Recent work by Şener et al. [ŞUA12] proposes an error correction procedure and combines this with a dynamic and iterative graph-cut algorithm for interactive segmentation. Superpixels that do not conform to a dynamically-learned single Gaussian color model are removed from the graph.

The problem of accommodating inaccuracies in user-indications shares some commonality with problems in background subtraction [SZTS06, SJK09]. Although we share the need to handle an inaccurate data term, our scenario has its own specificities. We deal with static images, not video. We cannot observe what is behind the selected object. Also, our data term is sparse and user-provided and, for example, errors tend to be spatially coherent.

2. Robust and Accurate Selections

In this section, we describe the main components of our approach. We start by showing a selection model based on a dense CRF that is robust to scribble inaccuracies. Then, we show how to further improve the selection accuracy with patch-based similarity functions. Finally, we explain how to embed the data into an Euclidean space such that we can efficiently solve the dense CRF even for high-dimensional and possibly non-Euclidean distance functions.

2.1. Error-tolerant Scribble Expansion

Most previous work assume perfectly accurate user input. In this scenario, using the scribbles as hard constraints is a sensible choice. However, in our context where the scribbles may be partly inaccurate, such constraints are detrimental since any error in the scribbles becomes an error in the selection and in addition, because it is used as a hard constraint, it is likely to "contaminate" its neighborhood. We address this issue by using soft constraints instead. This gives the algorithm the ability to ignore the errors where there is sufficient evidence that the other label is more appropriate. Because inaccurate scribbles create spatially consistent errors, considering only a small neighborhood around a pixel, e.g., its 4 adjacent pixels [LTS04], is unlikely to be enough to detect errors. In many cases, a compact group of pixels is mislabeled altogether and with such limited interactions, they would reinforce each other.

The solution is to have each pixel gather evidence from the entire image, i.e., from all other pixels [KLT09]. In our work, we use the efficient dense CRF technique of Krähenbühl and Koltun [KK11] that minimizes the following functional:

$$E(\mathbf{x}) \equiv \sum_i \psi_u(x_i) + \sum_{i<j} \psi_p(x_i, x_j), \qquad (1)$$

with

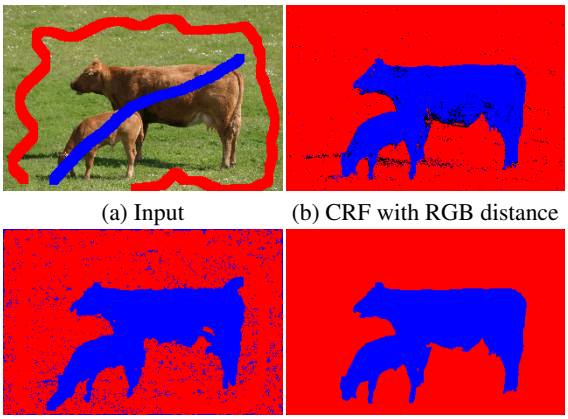
$$\psi_p(x_i, x_j) = \mu(x_i, x_j) \sum_{m=1}^{K} k^{(m)}(\mathbf{f}_i, \mathbf{f}_j). \qquad (2)$$

Here, $x_i \in \{\text{foreground}, \text{background}\}$ is the output label at pixel i. $\psi_u(x_i)$ is the unary potential that is computed independently for each pixel; in our case it is a simple foreground/background classifier based on the pixel's feature vector $\mathbf{f}_i$ (more details in next subsection). The term $\psi_p(x_i, x_j)$ denotes the pair-wise term, where $\mu(x_i, x_j)$ is a simple Potts model and introduces a penalty for pixels being assigned different labels ($\mu(x_i, x_j) = 1$ if $x_i \neq x_j$ and 0 otherwise); $k^{(m)}(\mathbf{f}_i, \mathbf{f}_j)$ is a kernel weighting how likely it is to have the same label x_i and x_j given their corresponding feature vectors.

It is important to note that the method of Krähenbühl and Koltun *only* supports Euclidean distances between features, as the operand to their Gaussian kernel. This is often insufficient for clean selections (see Fig. 2b) for an example). Although we also use a Gaussian kernel $k^{(m)}$ over the (weighted) L_2 norm between the feature vectors, *we can accommodate* other similarity measures based on non-Euclidean distances by first computing an approximate Euclidean embedding of the pixels. That is, rather than packing a sophisticated similarity metric into the kernel, we use a simple Gaussian kernel and, instead, automatically adjust the feature space using to approximate the given similarity measure.

Even though the energy functional $E(\mathbf{x})$ depends on all pairs of pixels, i.e., each pixel i is connected to *all* other pixels j via the pair-wise term ψ_p, it can be solved very efficiently as shown by Krähenbühl and Koltun. In the following subsections, we detail the unary term and explain how to obtain the feature vectors $\mathbf{f}$ given a similarity measure.

Scribble Input & Per-Pixel Classifier The user supplies foreground and background scribbles as input. We use these to define probabilities, for each pixel being labeled foreground ($p_f \equiv P(x_i = \text{foregound})$), background ($p_b \equiv P(x_i = \text{background})$) and void ($p_v \equiv P(xi = void)$). That is, each pixel is assigned a 3-tuple of probabilities, $\langle p_f, p_b, p_v \rangle$ so that the three sum to one. We set these tuples to be $\langle 0.5, 0.25, 0.25 \rangle$ for pixels under the foreground scribble, $\langle 0.25, 0.5, 0.25 \rangle$ for pixels under the background scribble and $\langle 0.33, 0.33, 0.33 \rangle$ for all other pixels. This choice assumes an equal possibility of inaccuracies over the three labels, in the absence of extra information (besides the scribbles). For example, changing one of the 0.5 values to 0.6 will make it less robust to the corresponding input inaccuracy. Although we experimented with more sophisticated assignments, such as using distances to the foreground and background scribbles to derive the probabilities, they did not impact the results sufficiently enough to justify their complexity. We then compute the corresponding unary potentials



(a) Input (b) CRF with RGB distance

(c) nearest neighbor with χ^2 (d) CRF with χ^2

Figure 2: *Inaccurate foreground (blue) and background (red) user-scribbles in the input image (a) pose a challenge for selection. Applying an efficient and powerful segmentation technique such as with a dense CRF [KK11] with Euclidean distance in space (location) and color (RGB) works well (b) but is often insufficient. Using a sophisticated similarity measure to provide a distance, with simple classification such as a nearest-neighbor approach fails when the scribbles are inaccurate (c). Our approach is to enable a combination of both. A combination of the χ^2 distance, a cross-bin histogram similarity measure, with a dense conditional random field (CRF) binary classifier yields a clean solution (c) .*

from the probabilities, as the logarithms of the respective reciprocals. We rely on the fully connected CRF to propagate the label assignment across the rest of the image.

2.2. Pixel Similarity Functions

As discussed, the result of the dense CRF depends on how we compare pixels. In this section, we focus specifically on how to evaluate pixel similarity. We discuss how to incorporate it in the dense CRF model in the next subsection.

A simple option is to compare pixels using their colors, that is, for two pixels i and j, we estimate their similarity using the L_2 norm in a color space such as RGB or CIE Luv. As we shall see in the result section, this works well for images in which the foreground and background have different colors. However, this may not always be the case. When color is not sufficiently discriminative, a solution is to also observe the neighborhood around each pixel to account for the local texture. These patches can be compared in several ways. First, one can unroll the patches into large vectors and use the L_2 norm on these vectors. Standard options that compare the cross-bin distances between the histograms of the patches are more robust, yet discriminative. We experimented with two standard choices: the earth mover's distance (EMD) [RTG98] and the χ^2 distance, which is derived from the statistical χ^2 test to compare two distributions [GN96]. EMD is a common choice for vision applications [RTG98]. It interprets histograms as piles of dirt and estimates their differences as the amount of work needed to transform one pile into the other by moving dirt. Doing so takes into account how different the density peaks are as well as how distant they are in color space. We refer to [RTG98] for a formal definition. χ^2 distance is another common choice that normalizes the density differences by their amplitude to account for statistical variation [GN96]. For two histograms h_1 and h_2 with bins indexed by ℓ, we compute the distance as $\sqrt{\sum_\ell (h_{1\ell} - h_{2\ell})^2/(h_{1\ell} + h_{2\ell})}$.

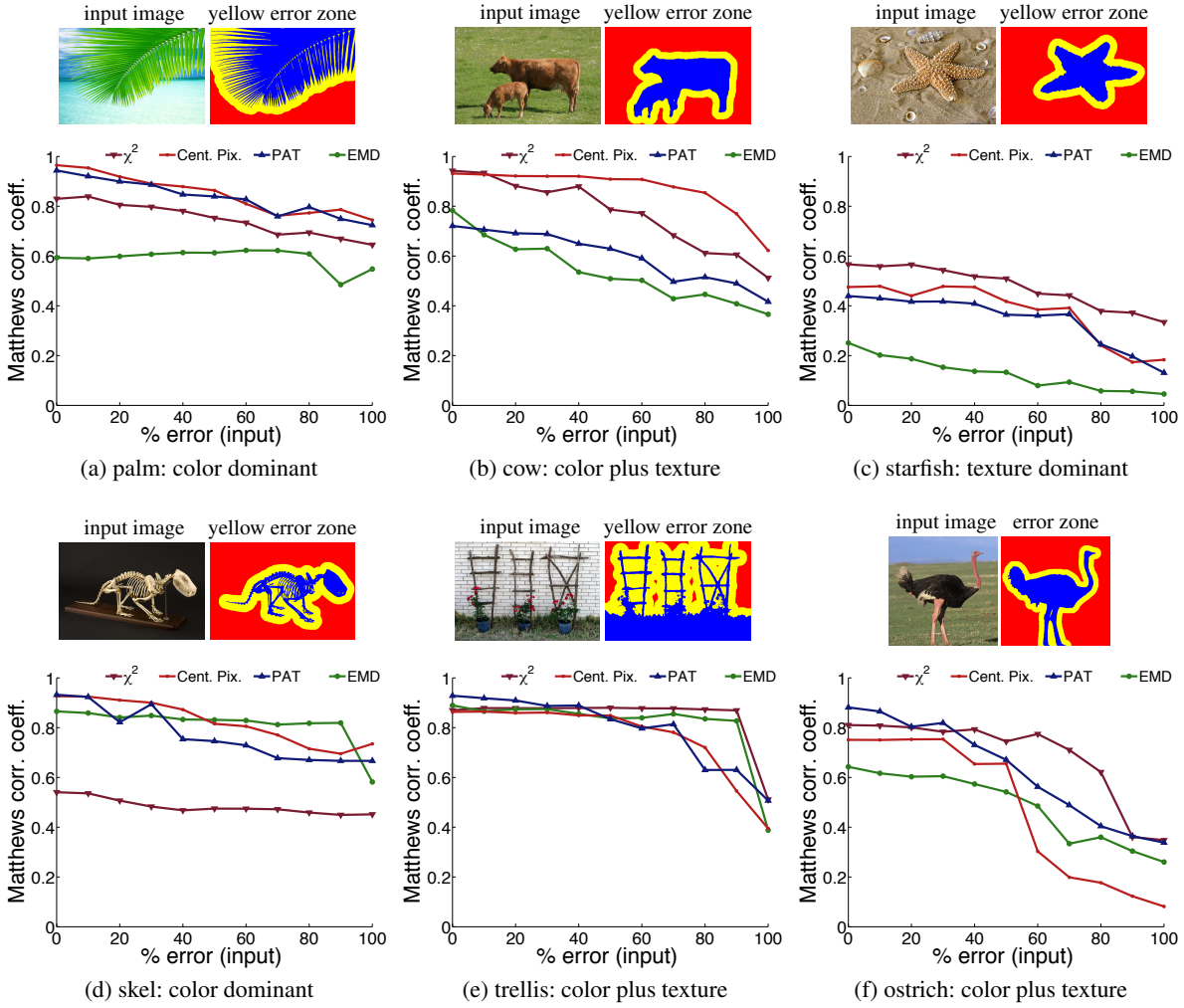
2.3. Manifold Embedding

The technique that we use to solve the dense CRF [KK11] requires the similarity between pixels to be expressed as the L_2 norm between feature vectors $\mathbf{f}$. Further, to obtain interactive running times, the $\mathbf{f}$ vectors need to be low-dimensional. Comparing the color of a single pixel is straightforward to use with this solver by using the 3 color channels of a pixel to define a 3D $\mathbf{f}$ vector. However, the patch-based similarity functions cannot be used as is. For the L_2 norm on patches, since it operates on a vector space, we could use PCA to reduce the dimensionality of the feature vectors to get an optimal approximation of the similarity function. However, it does not apply to the χ^2 and EMD functions that do not correspond to a Euclidean distance in a vector space. We overcome this problem by embedding the data in a space with a metric that approximates a given distance function.

We experimented with several variants. While nonlinear embeddings such as Isomaps [TdL00] and LLE [RS00] can represent more complex structures, they are also more computationally intensive and we did not see an improvement in performance that would warrant such additional cost. Instead, we opted for a linear embedding using Multi-Dimensional Scaling (MDS) [CC00] to estimate a linear embedding. We use the Landmark MDS algorithm [dST02] that achieves interactive performances by computing the embedding for a few points while taking into account the distances between all the data points. This is a type of Nyström method since it approximates the structure in a large matrix by sampling a small set of rows in the matrix. When the input metric is Euclidean, the result is equivalent to PCA, i.e., it produces an optimal embedding. We refer to [dST02] for the details of the Landmark MDS algorithm. The result is a set of low-dimensional feature vectors $\mathbf{f}$ that represent the image pixels, which pairwise L_2 distances is as close as possible to the input distance constraints. We feed these low-dimensional points into the pairwise term ψ_p. In practice, for the distances that we experimented with, we observed that three-dimensional vectors $\mathbf{f}$ were sufficient.

2.4. Synthetic Experiments

To find which option works best in our context, we use a synthetic experiment that allows us to control amount of er-



(a) palm: color dominant (b) cow: color plus texture (c) starfish: texture dominant

(d) skel: color dominant (e) trellis: color plus texture (f) ostrich: color plus texture

Figure 3: *Evaluation of the quality of the binary classifier by plotting the average Matthews correlation coefficient over many runs against percentage error in the input scribbles. A value of* 1 *on the Y-axis is achieved when the selection matches ground truth perfectly. For PAT (L2 distance between local patches),* χ^2*, and EMD, we use* 5×5 *patches. All distances are embedded in a 3-dimensional Euclidean space (even the RGB center pixel similarity, which is equivalent to PCA).*

rors in the input scribbles. We manually create ground-truth selections for a few images. For each image, we randomly pick 50 pixels in the foreground and 50 pixels in the background, and assign them the ground-truth label. We also define an "error zone" that comprises the background pixels that are less than a distance D from the foreground. In practice, we set D equal to 5% of the image diagonal. We randomly sample between 0 and 50 pixels in this region and assign them a foreground label, which is erroneous. This experiment represents the case where users have difficulties to mark the main object and their foreground marks "bleeds over the background". While this is clearly an approximation, it matches our own experience, i.e., we found the foreground often harder to label than the background – and most importantly, it allows us to study the impact of the amount of input error on the produced selection. We used this experiment to evaluate several factors:

- the color space: RGB or CIE Luv,
- the size of the patches: 3×3 and 5×5 (larger sizes were too time consuming).
- the similarity functions: L_2 norm on a single pixel, L_2 norm on patches, EMD, and χ^2.

Figure 3 shows sample plots. For the sake of clarity, we only report the best color space and patch size for each similarity function. We use Matthews correlation coefficient (MCC) to evaluate the quality of binary selections [Pow11]. It is defined as follows: $(pn - \bar{p}\bar{n})/\sqrt{(p+\bar{p})(p+\bar{n})(n+\bar{p})(n+\bar{n})}$ with p, n, $\bar{p}$, $\bar{n}$, the

numbers of true positives, true negatives, false positives, and false negatives. A value of 1 means a perfect segmentation and the score decreases as there are more errors in the selection. A random allocation of labels results in a value of 0 and a perfectly inverted classification yields a value of -1. For binary selection, we only expect values in $[0,1]$.

Two options stand out: the CIE Luv L_2 norm on a single pixel performs well on colorful images while χ^2 on the CIE Luv histograms of 5×5 patches yields satisfying results on challenging images with little color information for which texture differences are critical. By default, we use the L_2 norm on a single pixel, and users can switch to the "texture mode" that corresponds to χ^2 on 5×5 patch histograms.

3. Results

We validate our approach by comparing it to representative existing techniques. Diffusion Maps [FFL10] also use an embedding to better differentiate the pixels. It then applies the Matting Laplacian technique [LLW08], which assumes accurate scribbles; an assumption, as we shall see, that has a major impact on the produced selections. Hence, we regard this method as representative of all the methods that use the scribbles as hard constraints, e.g., [ADA*04]. We also compare to KNN Matting [CLT12] because it produces state-of-the-art mattes. Furthermore, we compare against Instant Propagation [LJH10], an efficient edit propagation method, which can tolerate some inaccuracies in the scribbles. All techniques generate soft selections. To compare them to our approach, we threshold their results to get binary selections. To make this step fair, we use an oracle to find the best possible threshold. For quantitative experiments, for each run, we test 10 regularly spaced threshold values and select the one that yields the closest result to ground truth. For visual results, we manually set the threshold to obtain the best looking selection.

For our results, we evaluate both the "center pixel" option where we compare pixels using the L_2 norm on their colors, and the χ^2 test on patch histograms. Both are computed on a 3D feature space obtained with Landmark MDS. For the center pixel option, this aligns the color space axes with the color distribution, which produces better results in our tests. With χ^2, this generates a 3D space with a metric that approximates the test value. The CRF solver classifies pixels as "foreground", "background", and "unclassified". In most cases, there are no "unclassified" pixels but when there are, we count them as false positives or false negatives in our quantitative measure, i.e., we apply the worst scenario to prevent bias in our favor.

On an Intel Core i7-2630QM CPU @ 2.00GHz, our algorithm runs in about 2 seconds for χ^2 using 5×5 patches on images of size 800×600. Of this, about 35% is typical for computing the pairwise distances, about 50% for the embedding, and the rest for the dense CRF propagation. For central pixel only, our algorithm runs in under 1 second for the same image. The embedding is only performed once, while loading the image. For each set of scribbles, only the classification (using dense CRF) needs to be performed, which is at near-interactive rates.

Quantitative Results We proceed as in Section 2.4, i.e., we use random pixels as scribbles with a controlled number of errors and compare the results to manually created ground truth. In this test, in addition to Diffusion Maps and KNN Matting, we include the pixel similarity function based on the feature vectors $\mathbf{f}$ proposed by An and Pellacini [AP08]: they augment the pixel color with average and standard deviation of the colors in a 3×3 neighborhood. We use these $\mathbf{f}$ vectors in the same dense CRF as our approach. We also compare our approach to nearest-neighbor classifiers based on the pixel colors and the χ^2 distance between patch histograms.

The plots in Figure 5 show that when there is no error, all methods perform well. But as soon as inaccuracies corrupt the input scribbles, the accuracy of other techniques decreases quickly. For images where colors discriminate well the foreground from the background as in the palm example, the decrease is slower; whereas it is sharper with challenging images such as the manga. In comparison, with our approach, the selection quality is almost not impacted until high levels of errors, typically beyond 50%. Error values measured for partly inaccurate scribbles made by hand confirm that our approach performs better (Fig. 4).

Figure 3 compares four embedding dimensionalities with respect to selection accuracy for decreasingly accurate input. The data shown is for the palm leaf image, with the χ^2 distance between local histograms. Although there is a notable improvement in accuracy (for input errors of $< 50\%$), this comes at the cost of performance.

Qualitative Results Figure 6 shows actual selection produced by our approach and existing techniques. We manually created scribbles with various degrees of error. The results show that for moderate levels of error as in the palm and ostrich images, all methods produce usable selections. However, for more challenging cases, existing techniques generate inaccurate outputs. In all cases, the errors come directly from the input scribbles, that is, input errors appear in the output and often expand into their vicinity. In comparison, the result of our approach contain far fewer errors and are usable.

Figure 7 shows two examples using GrabCut [RKB04]. While it requires different user input (bounding box), the input does not need to be accurate, similar to our method. GrabCut works well on some images but tends to have difficulties in cases where the bounding box is not expressive enough. method

Figure 8 shows the results of error tolerant interactive segmentation using dynamic and iterated graph-cuts [ŞUA12].

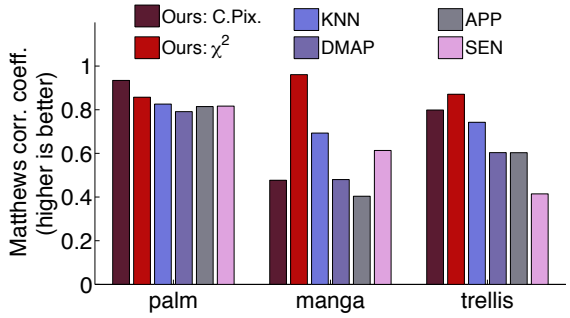


Figure 4: *Comparison of our selection, using χ^2 distance and center pixel distances, with competing techniques on three images with user-provided scribbles. KNN is k-nearest neighbour matting [CLT12]. DMAP is the matting Laplacian applied on top of diffusion map, as suggested by Farbman et al. [FFL10]. APP is the similarity measure proposed An and Pellacini [AP08] used with the same dense CRF as our approach. SEN is the method of Şener et al [ŞUA12]*

.

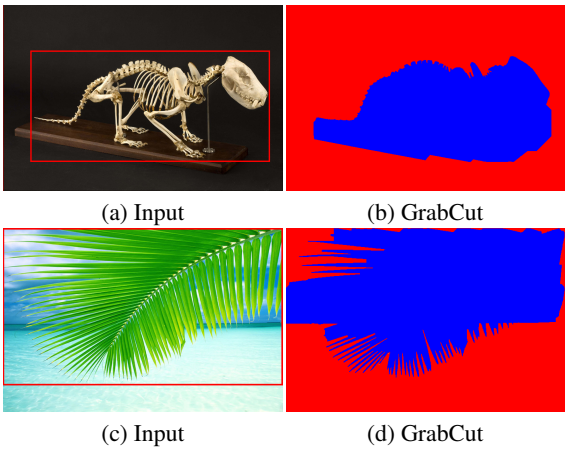


Figure 7: *A few examples showing results using GrabCut [RKB04]. Compare to our selections in Figure 6.*

They only accept foreground scribbles. Their results also depend on the order in which pixels are marked as foreground. The authors kindly shared their results. Note that their implementation has modified the aspect ratios. Their results were obtained by user scribbles as close to ours (in fig. 6) as possible, while maintaining an order favorable to their algorithm.

3.1. Discussion and Limitation

Parameters We experimented with various choices for the similarity measure, along with its associated parameters such as patch size and color space, Nyström sampling rate, output dimension and the width of the CRF Gaussian kernel. All results in this paper were generated with 3D embeddings

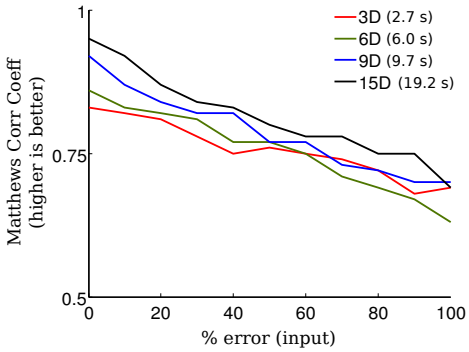
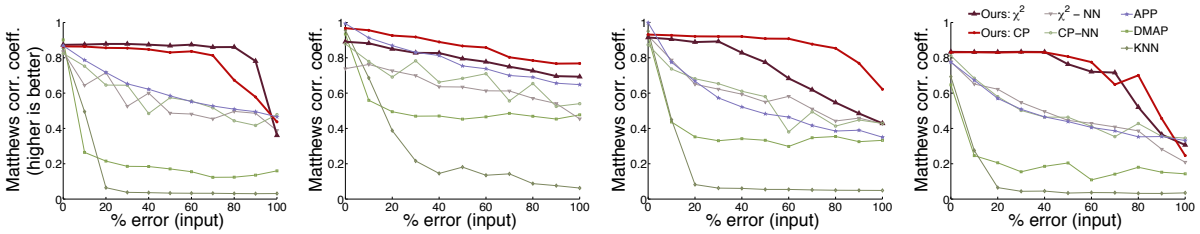


Figure 9: *Increasing the dimensionality of the embedded space improves selection accuracy at the cost of speed. The plot compares selection accuracy for embedding dimensionalities 3D, 6D, 9D and 15D on the palm leaf image.*

and a constant set of parameters. As mentioned earlier, we found that using the distance between RGB center pixels or χ^2 distances between the CIE Luv values of 5×5 patches mutually work well for a variety of images. For the Nyström approximation, we used 0.0025% of the number of pixels in the image, or 25 samples for a 1 megapixel image. We use a standard deviation of 1 for the CRF Gaussian kernel when using the patch-based similarities and 0.1 when using the center pixel comparison.

The Effect of Scribbles If one class of indications, say background scribbles, are accurate our selection is more robust to errors in the foreground scribbles. For instance, the wind turbines in Figure 10 are selected although the foreground scribbles actually cover more non-selected pixels than desired ones. Intuitively, our simple assignment of probabilities for computing the unary potential leads to a voting scheme. Since the background scribbles cover the sky sufficiently, the sky pixels in the foreground stroke are "outnumbered" and are less likely to confuse the classifier. Our algorithm still produces an accurate result despite the highly inaccurate foreground scribble. In our experiments, we observed that dense scribbles tend to overly constrain the CRF, leading to less accurate results. However, users typically make sparse scribbles, as in Figure 6, with which our approach works well.

Similarity measures In this paper, we have used center-pixel distances and χ^2 distances between local histograms since they compared favorably among the few similarity measures that we explored. Experimenting with more powerful similarity measures is an exciting avenue for future work. For example, the pixel position may be included in the computation of dissimilarities. We found that often, the objective is to select the main object in an image. In that case, pixel position does not help much but adds 2 dimensions to the feature vectors, which slows the process down without an evident improvement in accuracy. In general, a variety



(a) trellis (b) palm (c) cow (d) manga

Figure 5: *Comparison of our selection, using χ^2 distance and center pixel distances, with competing techniques on multiple images. χ^2-NN and CP-NN (center pixel) are nearest neighbor classification of pixels based on the respective distance metrics. APP is the similarity measure proposed in the AppProp paper [AP08]. DMAP is diffusion maps with closed form matting applied on top of the diffusion map [FFL10]. KNN is k-nearest neighbour matting [CLT12]. The reported quality, using the Matthews correlation coefficient (Y-axis), is the average coefficient produced over 10 iterations using random scribbles generated with increasing levels of inaccuracy (X-axis).*

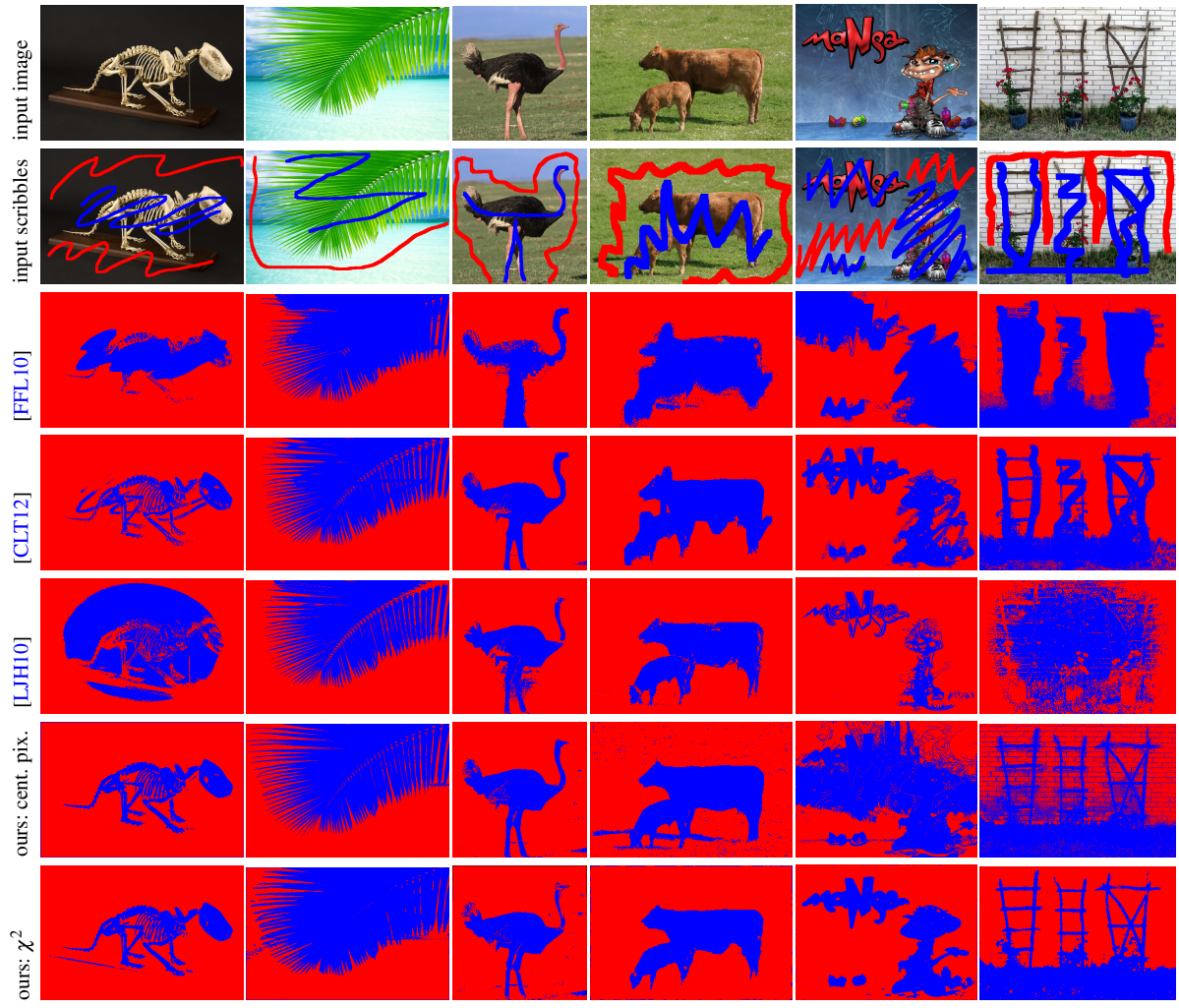


Figure 6: *Qualitative comparison of our technique against competing techniques: diffusion maps [FFL10], KNN matting [CLT12], and instant propagation [LJH10]. Blue is selected foreground and red is background.*

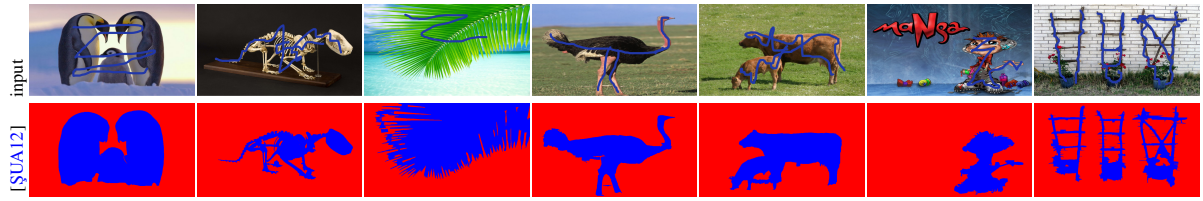


Figure 8: *A few examples showing results using error-tolerant interactive image segmentation using dynamic and iterated graph-cuts [ŞUA12]. This method only accepts foreground indications from the user. Compare to our selections in Figure 6.*

of feature vectors may be concatenated and dissimilarities computed by weighting their respective contributions. This would offer a fine-grained control over the color-vs-texture trade-off at the cost of an additional parameter.

Comparison with Related Work Our idea of operating in an embedded space is related to other methods such as the bilateral grid [CPD07], diffusion maps [FFL10], and the domain transform [GO11]. Similar to diffusion maps, we adaptively create the space based on the image content. Like AppProp [AP08], we perform a global embedding that considers all-pair distances. To leverage more efficient realization of all-pair interactions, we use the method of Krähenbühl and Koltun [KK11]. We KNN matting [CLT12] strikes a compromise between performance and long-range interactions, by reducing the interactions to a few nearest neighbors within large neighborhoods. In comparison, we improve performance by reducing the feature space dimensionality, which allows us to keep the interactions between all pairs. In addition to performance, our idea of first computing an approximately Euclidean embedding overcomes the major restriction in Krähenbühl and Koltun [KK11], that the feature space be Euclidean.

Limitations We currently rely on users to identify textured images and switch to χ^2 distances. An interesting direction would be to build an extension to make this choice automatically. Also, our method is designed for binary selections and would not perform as well on objects with transparent regions and/or thin features that are a few pixels wide or less. For instance, the spokes of the bike in Figure 11 are missing. Objects with similar color and texture as the background like the starfish are also challenging and the produced selection may require manual editing before being usable. Nonetheless, our approach provides a good starting point that is significantly more accurate than the output of other techniques. More intelligent image descriptors might help in this case, and are an interesting avenue of future research.

4. Conclusion

We have presented a method to produce accuracte image selections even when the user input is inaccurate. Our experiments show that our results are on par with existing techniques when there is no error in the input and that it performs significantly better when inaccuracies are present. We hope that our approach will lead to easier-to-use editing tools that tolerate some errors from users. We believe that this can be particularly useful with mobile devices that have small screens.

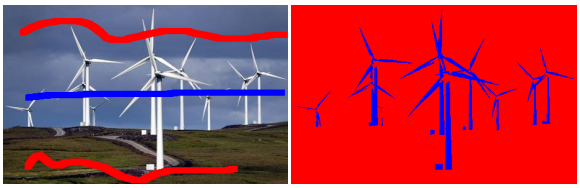


Figure 10: *Our approach relies on relative accuracy of foreground and background. For highly inaccurate scribbles, such as the blue one, the other (red) scribble is compensatory. Since the red scribble covers ample regions of the sky and ground, the classifier is reinforced about the background pixels and is less likely to be confused by an inaccurate foreground scribble.*

Acknowledgements

We thank the anonymous reviewers for their suggestions. We also thank the authors of [ŞUA12] for providing the input scribbles and segmentation results shown in fig. 8. Kartic Subr is supported by the Royal Society's Newton International Fellowship.

References

[ADA*04] AGARWALA A., DONTCHEVA M., AGRAWALA M., DRUCKER S., COLBURN A., CURLESS B., SALESIN D., COHEN M.: Interactive digital photomontage. *ACM Transactions on Graphics 23*, 3 (2004). 1, 2, 6

[AP08] AN X., PELLACINI F.: AppProp: all-pairs appearance-space edit propagation. *ACM Trans. Graph. 27*, 3 (2008). 2, 6, 7, 8, 9

[BJ01] BOYKOV Y., JOLLY M.-P.: Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images. In *International Conference on Computer Vision* (2001). 1, 2

[BWB06] BENKO H., WILSON A. D., BAUDISCH P.: Precise selection techniques for multi-touch screens. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2006), CHI '06, ACM, pp. 1263–1272. 2

[CC00] COX T. F., COX M. A. A.: *Multidimensional Scaling*. Chapman & Hall/CRC, 2000. ISBN: 1584880945. 4

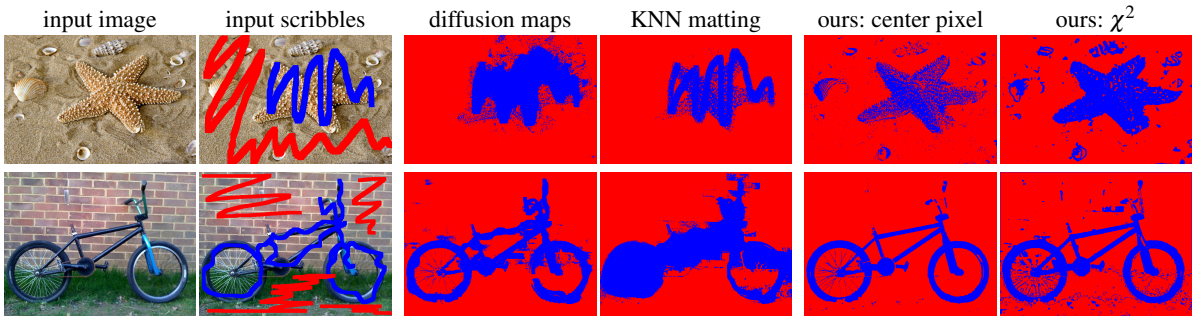


Figure 11: *The top example is an extremely challenging case because of the camouflage of the starfish. Although our result would need to be cleaned, it is more accurate than other techniques. The bottom example shows that, similar to other methods, our approach is not good at identifying thin features such as the spokes.*

[CLT12] CHEN Q., LI D., TANG C.-K.: KNN matting. In *IEE Computer Vision and Pattern Recognition* (2012). 1, 6, 7, 8, 9

[CPD07] CHEN J., PARIS S., DURAND F.: Real-time edge-aware image processing with the bilateral grid. In *ACM SIGGRAPH 2007 papers* (New York, NY, USA, 2007), SIGGRAPH '07, ACM. 2, 9

[dST02] DE SILVA V., TENENBAUM J. B.: Global versus local methods in nonlinear dimensionality reduction. *Advances in Neural Information Processing Systems 15.* (2002). 4

[FFL10] FARBMAN Z., FATTAL R., LISCHINSKI D.: Diffusion maps for edge-aware image editing. *ACM Transations on Graphics 29* (2010). 1, 6, 7, 8, 9

[GN96] GREENWOOD P., NIKULIN M.: *A guide to chi-squared testing*. J.Wiley, 1996. ISBN 0-471-55779-X. 4

[GO11] GASTAL E. S. L., OLIVEIRA M. M.: Domain transform for edge-aware image and video processing. *ACM Transactions on Graphics 30*, 4 (2011). 9

[KK11] KRÄHENBÜHL P., KOLTUN V.: Efficient inference in fully connected CRFs with Gaussian edge potentials. In *Advances in Neural Information Processing Systems* (2011). 2, 3, 4, 9

[KLT09] KOHLI P., LADICKÝ L., TORR P. H.: Robust higher order potentials for enforcing label consistency. *Internation Journal Computer Vision 82*, 3 (2009). 3

[LAA08] LI Y., ADELSON E., AGARWALA A.: Scribbleboost: adding classification to edge-aware interpolation of local image and video adjustments. In *Proceedings of the Nineteenth Eurographics conference on Rendering* (Aire-la-Ville, Switzerland, Switzerland, 2008), EGSR'08, Eurographics Association, pp. 1255–1264. 2

[LFUS06] LISCHINSKI D., FARBMAN Z., UYTTENDAELE M., SZELISKI R.: Interactive local adjustment of tonal values. *ACM Trans. Graph. 25*, 3 (July 2006), 646–653. 2

[LJH10] LI Y., JU T., HU S.-M.: Instant propagation of sparse edits on images and videos. *Computer Graphics Forum (Proceedings of Pacific Graphics 2010) 29*, 7 (2010), 2049–2054. 2, 6, 8

[LLW04] LEVIN A., LISCHINSKI D., WEISS Y.: Colorization using optimization. *ACM Trans. Graph. 23*, 3 (Aug. 2004), 689–694. 2

[LLW08] LEVIN A., LISCHINSKI D., WEISS Y.: A closed-form solution to natural image matting. *IEEE Transactions on Pattern Analysis and Machine Intelligence 30*, 2 (2008). 6

[LSS09] LIU J., SUN J., SHUM H.-Y.: Paint selection. *ACM Trans. Graph. 28*, 3 (2009). 1, 2

[LTS04] LI Y., 0001 J. S., TANG C.-K., SHUM H.-Y.: Lazy snapping. *ACM Transactions on Graphics 23*, 3 (2004). 2, 3

[MB95] MORTENSEN E. N., BARRETT W. A.: Intelligent scissors for image composition. In *ACM SIGGRAPH* (1995). 1, 2

[OH08] OLSEN JR. D. R., HARRIS M. K.: Edge-respecting brushes. In *ACM Symposium on User Interface Software and Technology* (2008). 2

[Pow11] POWERS D. M. W.: Evaluation: From precision, recall and F-factor to ROC, informedness, markedness and correlation. *Journal of Machine Learning Technologies 2*, 1 (2011). 5

[RKB04] ROTHER C., KOLMOGOROV V., BLAKE A.: "GrabCut": interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics 23*, 3 (2004). 2, 6, 7

[RRW*09] RHEMANN C., ROTHER C., WANG J., GELAUTZ M., KOHLI P., ROTT P.: A perceptually motivated online benchmark for image matting. In *Computer Vision and Pattern Recognition* (2009). 2

[RS00] ROWEIS S. T., SAUL L. K.: Nonlinear dimensionality reduction by locally linear embedding. *Science 290* (2000). 4

[RTG98] RUBNER Y., TOMASI C., , GUIBAS L. J.: A metric for distributions with applications to image databases. In *IEEE International Conference on Computer Vision* (1998). 4

[SJK09] SHEIKH Y., JAVED O., KANADE T.: Background Subtraction for Freely Moving Cameras. In *Computer Vision, 2009 IEEE 12th International Conference on* (2009), IEEE, pp. 1219–1225. 3

[ŞUA12] ŞENER O., UĞUR K., ALATAN A. A.: Error-tolerant interactive image segmentation using dynamic and iterated graph-cuts. *International workshop on Interactive Multimedia on Mobile and Portable Devices (in conjunction with ACM Multimedia).* (2012). 3, 6, 7, 9

[SZTS06] SUN J., ZHANG W., TANG X., SHUM H.-Y.: Background cut. In *Proceedings of the 9th European conference on Computer Vision - Volume Part II* (Berlin, Heidelberg, 2006), ECCV'06, Springer-Verlag, pp. 628–641. 3

[TdL00] TENENBAUM J. B., DE SILVA V., LANGFORD J. C.: A global geometric framework for nonlinear dimensionality reduction. *Science 290*, 5500 (2000). 4

[TJP10] TAO M. W., JOHNSON M. K., PARIS S.: Error-tolerant image compositing. In *European Conference on Computer Vision* (2010). 2