High Resolution Matting via Interactive Trimap Segmentation

Christoph Rhemann[1]*, Carsten Rother[2], Alex Rav-Acha[3], Toby Sharp[2]

[1]Institute for Software Technology and Interactive Systems, Vienna University of Technology, Austria
[2]Microsoft Research Cambridge, Cambridge, UK
[3]The Weizmann Institute of Science, Rehovot, Israel

Abstract

We present a new approach to the matting problem which splits the task into two steps: interactive trimap extraction followed by trimap-based alpha matting. By doing so we gain considerably in terms of speed and quality and are able to deal with high resolution images. This paper has three contributions: (i) a new trimap segmentation method using parametric max-flow; (ii) an alpha matting technique for high resolution images with a new gradient preserving prior on alpha; (iii) a database of 27 ground truth alpha mattes of still objects, which is considerably larger than previous databases and also of higher quality. The database is used to train our system and to validate that both our trimap extraction and our matting method improve on state-of-the-art techniques.

1. Introduction

Natural image matting addresses the problem of extracting an object from its background by recovering the opacity and foreground color of each pixel. Formally, the observed color C is a combination of foreground (F) and background (B) colors:

$$C = \alpha F + (1 - \alpha)B \qquad (1)$$

interpolated by the opacity value α. (This simplified model will be reconsidered later). Matting is a highly underconstrained problem and hence user interaction is essential.

In this introduction we will first consider the user aspects of the matting problem and then compare the different matting approaches themselves. Previous work in this domain can be broadly classified into three types of user interfaces.

The first class of interface is based on trimaps [3, 17, 22, 20, 4]. First the user paints a trimap by hand as accurately as possible, i.e. each pixel is assigned to one of three classes: foreground (F), background (B) or unknown (U) (e.g. fig. 1(middle)). In a perfectly tight trimap the α values in U are above 0 and below 1 and F and B regions have only α values which are exactly 0 and 1 respectively. The information from the known regions (F, B) is used to predict for each unknown pixel the values for F, B and α. It has been shown [22], and we will confirm it, that if the trimap is perfect (or nearly perfect), the resulting matte is of very high quality. The recent soft scissors approach [20] is probably the most sophisticated trimap “paint tool”. It builds on the intelligent scissors approach [13] which gives a hard segmentation, i.e. only F,B labels. In soft scissors the user traces the boundary with a “fat brush”. The brush size is adapted according to the underlying data and intermediate results of the matte are shown, enhancing the user experience. The main drawback of such a brush tool is that objects with a long boundary or complicated boundary topology are very tedious to trace, e.g. a tree with many foreground holes or the example in fig. 1(left).

Mainly due to this drawback, the trend of hard segmentation has been to move from *boundary* selection tools like intelligent scissors [13] to scribble-based *region* selection tools [2, 16]. This second class of interfaces is more user-friendly since only a few pixels have to be assigned to F or B, which are ideally far away from the boundary. Impressive results were achieved for hard segmentation [2, 16, 1] and also to some extent for matting [9, 21, 11, 5]. Note, a simple approach to obtain a soft matte from a hard segmentation is to run existing trimap-based matting techniques in a band of constant width around the hard segmentation, as done in e.g. [16, 1]. In contrast to this, our work computes an *adaptive* band which respects the underlying data.

We now review scribble-based matting methods. In [9, 11] a pure local “propagation” approach is taken and no global color information (i.e. outside a small window) is used. If the assumption holds that all colors within a small window around any unknown pixel lie on a line in color space then this approach obtains the ground truth matte. We observed in our experiments that this approach obtains good results for relatively tight trimaps (or scribbles), but it performs quite poorly on sparse scribble inputs. An example is shown in fig. 2(a) and fig. 7 in [9], where even relatively tight scribbles cause problems. A plausible explanation is that global color models do help considerably to overcome local ambiguities. In contrast, [21, 5] use both global color models and local propagation. In [21] an iterative approach is suggested where a global color model is used only for pixels with low confidence. In [5] the global color model of

*This work was supported in part by Microsoft Research Cambridge through its PhD Scholarship Programme and a travel sponsorship.

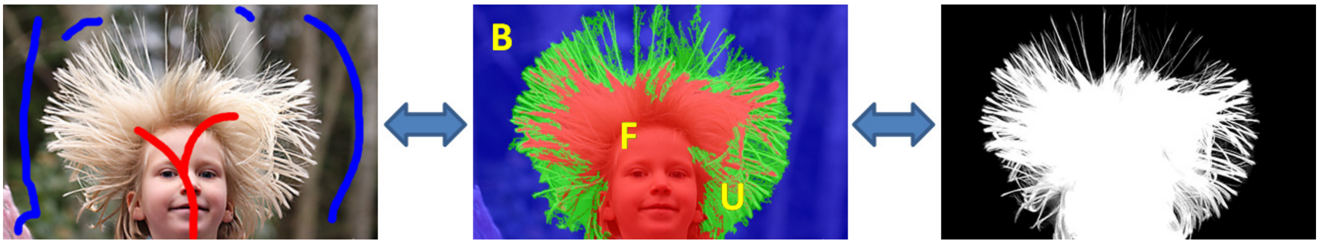


Figure 1. **Our interactive matting framework.** Given an input image (left) the user interactively creates a trimap (middle). From the trimap an α matte (right) is computed, first in low and then in high resolution, together with the true fore- and background colors. The user interactions consist of three types of brushes (F-foreground (red), B-background (blue), and U-unknown (green)) and a slider for the trimap size (interactive due to the recently developed parametric maxflow technique [8]). We believe that this approach has several benefits over existing ones: Speed, quality, and user friendliness (see text for details).

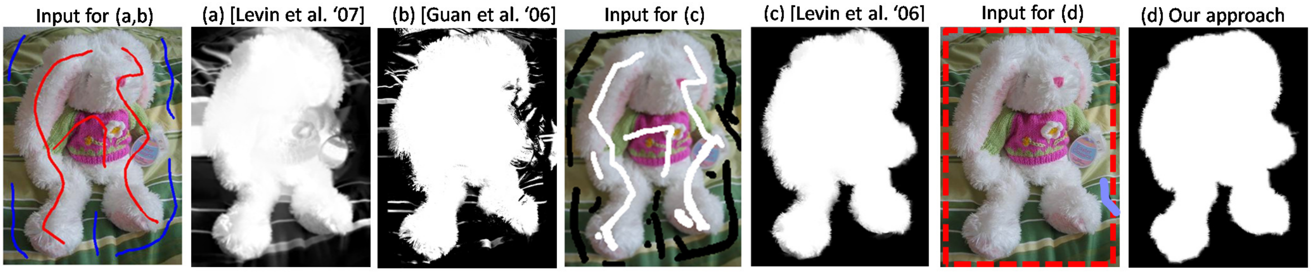


Figure 2. **Comparison of scribble-based matting approaches** (see text for discussion and more results in [15]). Our result was achieved with a single bounding box selection, inspired by [16], and one additional background brush stroke. Note, our approach can also handle more challenging α mattes, e.g. fig. 1. All results we show were either taken from the original papers or created with the original implementation of the respective authors.

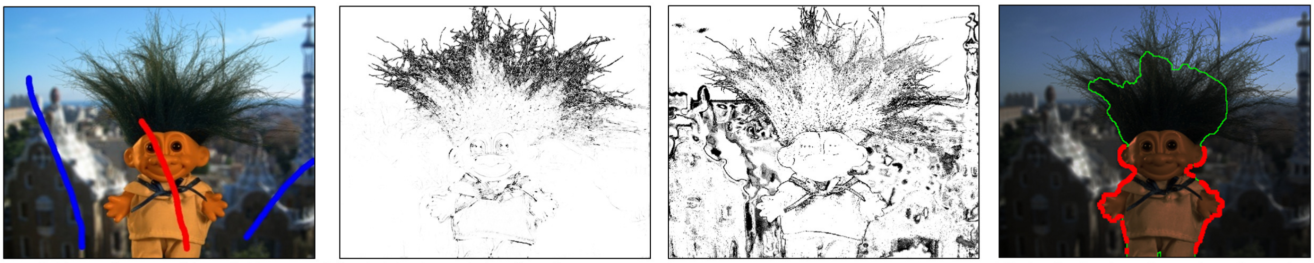
the brush strokes is used to reason about all unknown pixels (very similarly to [22]). However, even [21, 5] do not perform well for the example in fig. 2(b) and fig. 5b in [21] using scribbles. In order to achieve a good result for this example, previous results have required either many scribbles [9] (fig. 2(c)) or a very tight trimap [21] (fig. 5e in [21]).

Recently an intermediate solution has been suggested by Juan and Keriven [7]: The user interactively creates a trimap using a scribble-based interface (see fig. 1). In our work we use the same approach since we believe it to be an intuitive interface and it has an advantage in speed and quality compared to the approaches above. While the user creates the trimap, a preview of the matte is computed (fig. 1 right) and shown to the user. But even before a preview is available, the user can interactively adjust the trimap and remove obvious mistakes in order to simplify the matting task. In this process all images are scaled down to typically 0.3 Mpix. In a final step a high resolution matte, e.g. 6 Mpix, is computed, which in our case is a slow process. It is important to note that in contrast to the multi-resolution approach for hard segmentation [12], sub-pixel structure is captured in the unknown trimap region. Additionally, the user has a slider for controlling the trimap size which is interactive due to parametric max-flow [8]. The main benefit in speed comes from the fact that in a typical image most pixels belong solely to either fore- or background. For these pixels expensive matting algorithms which recover the full range of fractional α should not be invoked. For example, for a typical small resolution image (0.3 Mpix), [21] and [9] have reported a runtime of about $20sec$, [5] of about $200sec$, and we achieve with a tight trimap a runtime of 3.5sec using [22]. Moreover, not only speed but also the quality of the matte is improved, as shown in fig. 2(d). Our advantage over scribbled-based systems and [7] is that we exploit the result of a hard segmentation system such as [16] to build better global color models, and to detect and model physically sharp boundaries. Furthermore, we use our large ground truth dataset to build a classifier for potential trimap pixels, and to train parameters of our energy (in particular we learned a predictor for the trimap size λ).

For the second task of trimap-based alpha matting we concentrate on two challenges, which we believe have not yet been solved: (i) working with high resolution images and (ii) finding a good prior for α. The novel ideas are an edge-preserving sparsity prior for α and the use of the camera's point spread function to model most fractional α values in high resolution images.

Finally, using our new ground truth database we are able to show that we outperform both existing trimap creation approaches and trimap-based matting methods.

The paper is organized as follows. Section 2 explains our trimap extraction method, and section 3 the trimap-based matting approach, and finally section 4 introduces our database and describes experiments.



(a) Input Image with scribbles (b) sub-blur kernel structure (unary energy) (c) Color (unary energy) (d) Physically sharp boundary detection

Figure 3. **Unary terms for trimap segmentation**. (a) Input image with user scribbles (red-foreground, blue-background). (b) Unary energy for the sub-blur kernel structure term and (c) color term. Dark indicates low energy and white high energy. (d) Pixels in a small band around the F', B' transition of GrabCut [16] (green line) are classified into physically sharp boundary pixels indicated in bright red (the image was darkened for better visibility). The class prior is not visualized since it is constant over the whole image.

2. Interactive Trimap Segmentation

In the following we extend the approach of Juan and Keriven [7]. We denote the color of pixel i as c_i, its label as x_i, and its α value as α_i. Let I be the set of all pixels. Formally, the three subsets F, B and U (see fig. 1) are defined as $B = \{i|\alpha_i < \epsilon\}$, $F = \{i|\alpha_i > 1 - \epsilon\}$ and $U = I \backslash (F \cup B)$, where we choose $\epsilon = \frac{5}{255}$. (For simplicity, sets and labels have the same name, e.g. F.) We also introduce two extra subsets F', B' where $B' = \{i|\alpha_i \leq 0.5\}$ and $F' = \{i|\alpha_i > 0.5\}$. The transition from F' to B' is the 0.5 level-line of a hard segmentation. Obviously, it is $F \subset F', B \subset B'$ and $F' \cup B' = F \cup B \cup U = I$.

In [7], an energy for the three labels F, B, U was defined and optimized globally. Ideally we would like to define an energy for all 5 labels: F, F', B, B', U. It has the main advantage that the transition F', B' is modelled to coincide with an image edge (as in [2, 16]) whereas other transitions, e.g. B, U, are not data dependent (e.g. Potts model in [7]). However, instead of optimizing an energy for all 5 labels, we employ a 2 step process that allows a more expressive model and higher speed. First, we obtain a hard binary segmentation into the sets F' and B' using GrabCut [16]. The energy and parameter settings are as defined in [16] and the interested reader is refereed to the respective paper for details. Following the hard binary segmentation we compute a trimap segmentation with labels F, B and U (sec. 2.1). The energy function considers several image cues, and four different types of priors are used to regularize the result (visualized in fig. 3). We show that the trimap segmentation can be formulated as binary classification problem and minimized with graph cut. Finally, in sec. 2.2, we show how to learn the parameters for trimap segmentation from a training data set. Please note that in this section we assume the image to be of a small size, typically 0.3 Mpix.

2.1. Trimap Segmentation - Model

In this step all pixels will be assigned to one of the three labels F, B or U. We assume that each pixel has been already classified into F' or B' using GrabCut [16]. Since $F \subset F'$ and $B \subset B'$ a *binary* classification into two labels U and $\bar{U}$ is sufficient, where $\bar{U} = F \cup B$. (Each pixel in $\bar{U}$ is uniquely specified to be in either F or B given F', B'.) Note that α should only be fractional $(0 < \alpha < 1)$ at the boundary of the hard segmentation (F' to B' transition) and not necessarily exactly 0.5.

We define the binary energy E for the trimap extraction as:

$$E(\mathbf{x}, \theta) = \sum_{(i,j)\in\mathcal{N}} \theta_b V^b_{ij}(x_i, x_j) + V^s_{ij}(x_i, x_j)$$

$$+ \sum_i U^c_i(x_i) + U^p_i(x_i) + \theta_{b'} U^b_i(x_i) + \theta_t (U^s_i(x_i))^{\theta_{s'}} \quad (2)$$

where $\mathcal{N}$ is the set of neighboring pixels (8-neighborhood), and θ comprises of all model parameters. The energy can be locally optimized using graph cut [16] or parametric maxflow [8] depending on the choice of the free parameters during test time (see below). The individual terms are defined as follows.

Color (c) The color unary term for $\bar{U}$ is modeled as $U^c_i(\bar{U}) = -logP(c_i|\theta_{GF})$ if $x_i = F'$, and $U^c_i(\bar{U}) = -logP(c_i|\theta_{GB})$ if $x_i = B'$. Here θ_{GF} and θ_{GF} are the Gaussian Mixture Models (GMM) of fore- and background respectively (see sec. 2.2 for computational details). In the unknown region the color distribution of $U^c_i(U)$ is represented by a third GMM θ_{GU} by blending all combinations of fore- and background mixtures of the respective GMMs as in [7] (see example in fig. 3(c)). Motivated by [23], and in contrast to [7], the distribution of the blending coefficients is modeled as a beta distribution whose two free parameters were derived as $(0.25, 0.25)$ from our training set of ground truth alpha mattes (see sec. 4).

Class prior (p) The probability of a pixel belonging to the class U or $\bar{U}$ is image dependent. For instance, an image where the foreground object has been tightly cropped has a different proportion of U versus $\bar{U}$ pixels than the original image. We model this ratio by an unary term as:

$$U^p_i(x_i) = \lambda[x_i \neq U] \, , \quad (3)$$

i.e. a larger λ gives a larger U region. We show that predicting λ during test time improves the performance considerably. The learning of the predictor is discussed below (sec. 2.2). Furthermore, the parameter λ is also exposed to the user as a slider. Due to the nestedness property [8] of the solutions for all λ's, the λ slider corresponds to the size of the trimap. Note, the solution of our energy for *all* λ's can be efficiently optimized using parametric maxflow [8].

Sub-blur kernel structure (s) There are many different reasons for a pixel to have fractional α values (i.e. belong to U): Motion blur, optical blur, sub-pixel structure, transparency, or discretization artifacts. Here we concentrate on optical blur. The goal is to detect thin structures which have a width that is smaller than the size of the camera's point spread function (PSF). These structures give rise to fractional α values but they may not be close to a physically sharp boundary (e.g. the hair in fig. 3(b)). See explanation in fig. 4(right). The pixels belonging to thin structures are encoded in U^s (see [15] for more details).

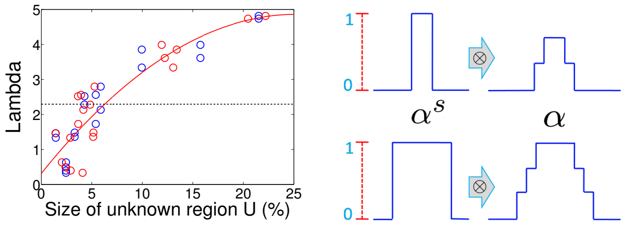


Figure 4. **Left:** Visualizing the correlation between λ and the size of the U region (see text for details). **Right:** A 1-D example of a thin (top,left) and thick (bottom,left) hard boundary (sparse α^s), which is either 0 or 1. It is convolved with the camera's PSF, here a box filter, which gives α. In the bottom case two smooth boundaries appear, i.e. some α values remain 1, which ideally should be detected by the hard segmentation (and then handled by the sharp boundary term). We want to build a detector for the top case where the thin structure is smaller than the size of the blur kernel. Roughly speaking for thin structure the magnitude of the first derivative should be low and the magnitude of the second derivative should be high.

Sharp boundary (b) The F', B' transition determined by GrabCut [16] often coincides with a clean, physically sharp boundary. This means that in the vicinity of the detected boundary (defined by the PSF) there is no other boundary transition, hence no sub-blur kernel structure. An example is the body of the object in fig. 3(d). At such boundaries the width of U is equal to the width of the camera's PSF and thus is only a few pixels wide. We classify the alpha matte, computed with the method of [9] in a small band around the F', B' transition, to determine the physically sharp parts of the F', B' transition . The result of this classifier is used to model the boundary terms U^b and V^b (see [15] for details).

Smoothness (s) Following [16] the smoothness term is defined as

$$V^s_{ij}(x_i, x_j) = \frac{\delta(x_i = x_j)}{\text{dist}\,(i,j)} \left(\theta_\lambda \exp -\beta \left\|c_i - c_j\right\|^2\right), \quad (4)$$

where δ is the Kronecker delta, β is as in [16] and θ_λ is defined below. As we show in sec. 4, it nicely preserves thin structures, e.g. hair, inside the unknown region.

We also enforce the U region to be 4-connected, which is true for 98.6% of the pixels in the ground truth database. Since enforcing connectivity is NP-hard [19], we do it by a simple post-processing step.

2.2. Trimap Segmentation - Training

For training we have used the following heuristic error (loss) function, which counts false negatives twice compared to false positives: $error = \frac{100}{n}\sum_i 2[x_i^{true} = U \wedge x_i \neq U] + [x_i^{true} \neq U \wedge x_i = U]$, where x^{true} is the labeling of the ground truth trimap and n the number of pixels. This is motivated by the fact that a missed unknown (U) pixel in the trimap can *not* be recovered during alpha matting. We see in sec. 4 that it is indeed correlated to the error for α matting. Based on our training dataset of 20 images (see sec. 4) we have hand tuned all the parameters in θ, except of those discussed below, to $\{\theta_b, \theta_{b'}, \theta_s, \theta_{s'}, \theta_\lambda\} = \{2, 40, 1, 2, 0.1\}$.

We have observed that the initialization of the color models θ_{GF}, θ_{GB} is rather crucial, not discussed in [7]. The reason is that the energy has typically many local minima with a high error since e.g. a true F color can be modeled as the blend of a true B color with another true F. Surprisingly, even making the GMMs spatially dependent did not help to overcome these local minima. After initialization, the color models can be updated iteratively as in [16]. However, it is not guaranteed that the energy decreases due to the mixture term $U^c(U)$, and therefore we build the color models from a guessed trimap (see below), and do not iterate.

As discussed above, λ (eq. 3) corresponds to the size of the region U. Fig. 4(left) shows, in blue dots, the optimal λ wrt to the size of region U of our training set. We see a strong correlation. To exploit this, we have built a predictor for the size of U (see below). The red dots in fig. 4 show the predicted values of our training data, and the red line is a quadratic function (3 parameters) fitted to them. We see that the red(predicted) and blue(true) points are close-by. During test time the size of U is predicted, given the test image, and the quadratic function provides the corresponding λ. Note, in practice λ is predicted after the first two F and B brush strokes and not changed afterwards, i.e. we do not alter our energy during the optimization. The dashed line in fig. 4 shows the optimal average $\lambda = 2.3$ which is *independent* of the size of U. It performs less well as we see in sec. 4.

Finally, we use the following heuristic to guess the initial trimap. We use the data terms U^c, U^s, that are available at this point, and find the globally optimal trimap by simple thresholding the unary energy. Note, for U^c we initialize θ_{GF} and θ_{GB} with $\theta_{GF'}$ and $\theta_{GB'}$ as in [7]. On our training image set we have obtained an average prediction error for U of 1.5% relative to the image size.

3. High Resolution Alpha Matting

Given a trimap we describe now our approach to matting. We base it on the method of Wang & Cohen [22] which was shown to produce state-of-the-art results from trimaps. They first obtain a pixel-wise estimation of α from high confidence color samples collected from the fore- and background regions of the trimap. This estimation is translated into two data terms W_F and W_B which are combined with a pairwise smoothness term and solved using random walk. (see [22] for a more detailed description). In our implementation, we use the matting laplacian L of [9] as a smoothness term for the matting, as it has a better theoretical justification, giving the following objective function:

$$J(\alpha) = \alpha^T L\alpha + \theta_\alpha(\alpha - \vec{1})^T W_F(\alpha - \vec{1}) + \theta_\alpha \alpha^T W_B \alpha \quad (5)$$

where α is treated as a column vector, W_F, W_B are written as diagonal matrices and θ_α is the relative weighting of the data versus the smoothness terms (we use $\theta_\alpha = 10^{-3}$). This objective function is minimized by solving a set of sparse linear equations, subject to the input constraints.

Fig. 5(d) shows the result of this process for the input in (a), which is a crop of a 7.6Mpix image of hair. The result looks too blurry compared to the ground truth in (c)[1]. Note, the result of applying [9] directly, i.e. omitting the data terms W_F, W_B, gives a clearly inferior result. It was shown in [11] that an additional sparsity prior, i.e. pushing α towards 0 or 1, can solve some ambiguities in the estimation of α. However, [11] employs a simple *pixel independent* prior, and also the prior turns eq. 5 into a complicated non-linear system. Thresholding the initial α (fig. 5(f)) demonstrates the problem (loss of hair structure) of using a pixel independent prior on the blurry alpha. The result of [11] in fig. 5(g) is pretty poor. Additionally, in [11] matting was performed on low res images where α is even less sparse then in high res (compare fig. 5(b) and (c)).

In this work we suggest a novel sparsity prior. It is based on a model of the imaging process, where the observed high resolution α is the result of blurring the true sparse alpha α^s with the camera's point spread function (PSF). Hence, the observed α has the form $\alpha = K \otimes \alpha^s$ as proposed in [6] for the case of motion blur, where $\otimes$ indicates convolution and the blur kernel K models the PSF. (Note that this approximates the real image composting eq. 1.) Fig. 5(h) shows α^s derived from the ground truth α in (c) and our kernel K (see details below). It is nearly a binary mask apart from sub-pixel structure, discretization artifacts, object motion, and semi-transparency (such as windows glass). Note, even a hair, as a physical entity, is opaque if the resolution is high enough and after deblurring. Here we assume that the object boundary is in-focus, i.e. we model the PSF of the in-focus area. Note, even in-focus pixels are slightly blurred due to imperfect camera optics. Note, Jia [6] computes K and α^s for motion blurred images given α by applying the method of [9]. In our work the "loop" is closed by improving α using α^s as a prior. We show that this works well for still images, and our framework is general enough to deal with motion blurred images, which we leave for future work.

Starting with the high resolution α we apply the following steps: (a) Initialize the PSF, (b) Deblur α to obtain sparse α^s using the PSF, (c) Estimate a binary sparse alpha α^{sb} from α^s while preserving edges, (d) Iterate (a-c) a few times, (e) Convolve the binary α^{sb} with the PSF and use it to re-estimate α.[2] Intermediate results of the process are in fig.5 (i-l). Each step is now described in detail:

Estimating the PSF. We model the PSF as a symmetric kernel K of size 9×9 with non-negative elements which sum up to 1. Motivated by [6] we use the following heuristic to initialize K. For all pixels in U we compute those which may belong to a physically sharp boundary (sec. 2.1). If the acceptance rate is above 10%, we obtain K by minimizing the linear system $||\delta(\alpha > 0.5) \otimes K - \alpha||^2$. Otherwise, a reliable initialization of K for the in-focus area is a Gaussian [14] (we use $\sigma = 1.5$). After the first iteration, the linear system is solved again using all in-focus pixels (see below), where $\delta(\alpha > 0.5)$ is replaced by α^{sb}.

Alpha Debluring and Binarization. To get the sparse alpha α^s from α and K we use the image-debluring implementation of [10] (see fig. 5(i)). From α^s we construct the binary sparse alpha α^{sb} as follows. We observed that applying a per-pixel sparsity such as thresholding α^s removes many features, such as hair, from the matte. A much better binarization can be obtained by preserving the edges of α^s (fig. 5(j)). To achieve this, we use the following MRF, and solve it with graph cut (since E is submodular):

$$E(\alpha^{sb}) = \textstyle\sum_i U_i(\alpha_i^{sb}) + \theta_{\alpha 1} \sum_{\{i,j\} \in \mathcal{N}} V_{ij}(\alpha_i^{sb}, \alpha_j^{sb}), \quad (6)$$

where α^{sb} is a binary labeling and $\mathcal{N}$ denotes an 8-connected neighborhood. The terms U_i and V_{ij} are given by

$$\begin{aligned} U_i(f_i) &= |\alpha_i^{sb} - \alpha_i^s| + \theta_{\alpha 2}|\alpha_i^{sb}|; \quad (7) \\ V_{ij}(f_i, f_j) &= \delta(a_i^{sb} \neq a_j^{sb}) + \theta_{\alpha 3}(\alpha_i^{sb} - \alpha_j^{sb})(\alpha_j^s - \alpha_i^s) \end{aligned}$$

with the constants $(\theta_{\alpha 1}, \theta_{\alpha 2}, \theta_{\alpha 3}) = (5, 0.2, 0.002)$. The data term encourages the labeling to be similar to α^s (with a small preference towards 0). The pair-wise term consists of both a regular smoothness and an edge-preserving term. Note that this edge-preserving smoothness is very different from the standard smoothness. Actually, with the edge-preserving term only, the global optimum contains typically thin (1-pixel wide) structures.

[1] The supplementary material shows the result of applying the original implementation of [22] which gives a more blurry result.

[2] In the formulation of [11] α can be replaced by α^s and K, however, we found it to be experimentally inferior to our approach.

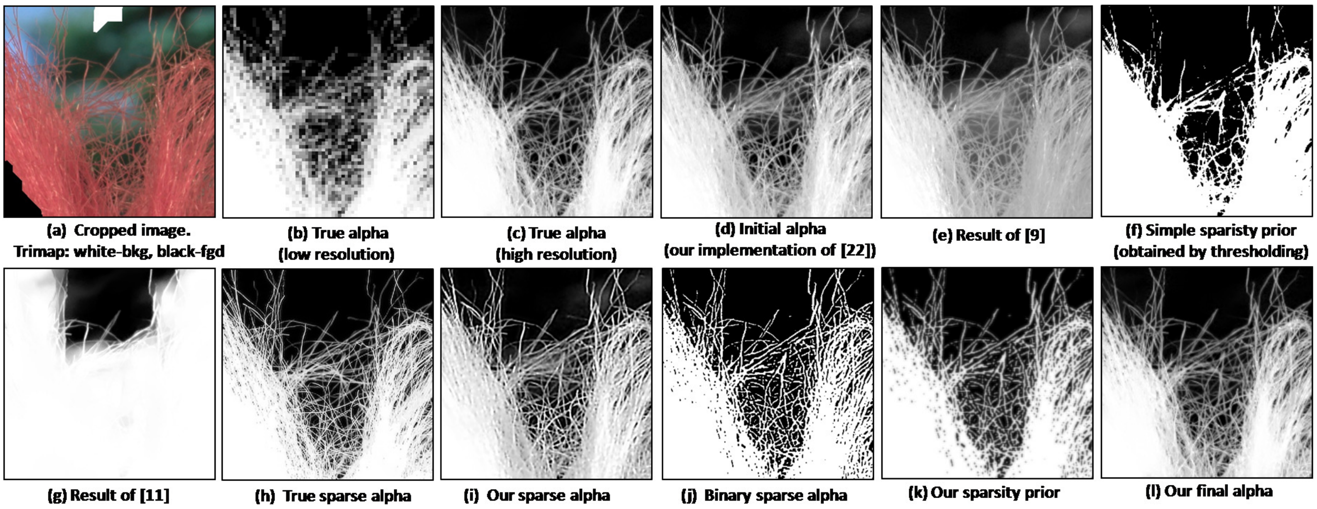


Figure 5. **The matting process.** (see detailed description in text). The key idea is that our final α matte (l) is derived from the initial matte of [22] (d) by imposing a new edge-preserving sparsity prior (k), which is better than a simple pixel independent prior (f).

Re-estimating α using the Sparsity Prior. By convolving the binary α^{sb} with K, we construct a new sparsity prior s_α on the values of α (fig. 5(k)). This prior is added simply by replacing the data terms in eq. 5 with the new terms:

$$W'_B(i) = W_B(i) + \theta_{\alpha 4} s_\alpha; W'_F(i) = W_F(i) + \theta_{\alpha 4}(1 - s_\alpha),$$

where $\theta_{\alpha 4} = 5$ is the relative weight of the new prior. The final alpha matte, shown in fig. 5(l), is less blurry than the initial matte in (d).

Handling Multiple PSFs. Due to depth variations there may be more than a single PSF along the object boundary. Ideally, this problem is handled by recovering depth at each pixel. In our work, however, we estimate a single PSF and assume it can describe well all in-focus pixels. The sparsity prior for other pixels is set to 0. First, we compute for each pixel the gradient of α^s, normalized by the range of values in a window (11×11) around each pixel. Then we compute the in-focus mask by thresholding this score (we used 0.4). Note that for out-of-focus regions our method is equivalent to regular matting methods and therefore can overcome multiple PSFs.

3.1. Multi-Resolution Estimation of the Matte

To obtain high quality alpha mattes of 6 Mpix images with reasonable time and memory requirements, we use a multi-resolution framework with three levels: 0.3 Mpix, 1.5 Mpix and 6 Mpix. The matte in lower resolutions is used as a week regularization for higher resolutions. At the higher resolution, α was solved by processing the image in overlapping windows. Using the low resolution matte as regularization has two advantages: (a) it encourages a smooth transition between windows (for that reason, this prior got a higher weight along window boundaries), (b) it pushes the solution towards the global optimum, which is essential for handling non-informative windows. Note, we computed the data terms of [22] in full resolution using the entire image since runtime and memory was reasonable.

4. Experimental results

We first introduce our database and then compare our approach to state-of-the-art methods. Note, more results are available in the supplementary material.

Ground Truth Database. Recently, two small databases were introduced [22, 11]. The database we use is considerably larger and, we believe, of higher quality. The data in [11] is of intermediate quality, probably due to noise[3]. In [22] most examples are natural (outdoor) images, which is a very realistic scenario, however the ground truth α was created using a variety of matting methods along with extensive user assistance. We believe that such a dataset is biased especially if used, as in our work, for training.

Our dataset has 27 α mattes obtained using triangulation matting [18] from the RAW sensor data (10.1 Mpix; Canon 1D Mark III). The objects have a variety of hard and soft boundaries (e.g. fig. 6) and different boundary lengths, e.g. a tree with many holes (see supplementary material). The final mattes, of average size 6 Mpix, consist of the cropped out objects. To create training and test images we composed the true foreground masks with different natural background images with varying degree of difficulty: color ambiguity of fore- and background and backgrounds with different degrees of focus. Then the set was split into 10 training and 17 test images. To obtain 20 training images, 2 different backgrounds were used. Finally, we created for each image a set of potential user inputs in the form of scribbles and trimaps. For each image we have casually drawn large scribbles that cover all major colors in the image but are not close to the object boundary.

[3] In the matte of fig. 8 in [11] 42% of true foreground pixels have an α value below 0.98, in our case this occurred only for 1% of pixels.

Method	av. error	worst 25%	time
Grady et al. '05 [4]	(24.3, 19.8)	(33.6, 28.6)	5
Levin et al. '07 [11]	(17.9;9.5)	(28.3;17.8)	20
Guan et al. '06 [5]	(13.4;9.0)	(22.7;16.5)	300
Levin et al. '06 [9]	(11.4;6.9)	(19.0;13.3)	18
Wang et al. '07 [22]	(11.0;8.4)	(22.5;19.0)	50
Juan et al. '05 [7]	(7.6;4.6)	(13.8;12.0)	1.5
Our (fixed $\lambda = 2.3$)	(2.5;1.2)	(4.9;2.3)	1
Our (predict λ)	(2.3;1.0)	(4.5;1.9)	1
Our (optimal λ)	(2.2;0.7)	(4.5;1.5)	1.8
True trimap	(0.0;0.4)	(0.0;0.8)	-

Table 1. **Comparison of trimap methods.** In brackets is our trimap error and our α matte error (definition in text). All numbers are averaged over all (worst 25%) test images. Times are in seconds and were measured on the same machine (2.2 GHz).

Method	Large	Small	Our	True
Our impl. Levin '06 [9]	1.5	1.2	1.3	0.71
Our impl. Wang '07 [22]	1.7	1.0	1.0	0.68
Ours	1.3	0.8	0.9	0.67

Table 2. **Comparison of trimap-based matting methods.** The average error over all test images for different trimaps (see text).

Comparison of Trimap Extraction Methods. We had to down-scale our 6 Mpix images to a size that most competitors can handle, which was 0.3 Mpix (e.g. 700 × 560) - the limit of the publicly available system of [9][4]. We compare our approach to six other methods [7, 22, 9, 11, 5, 4][5]. Note, with respect to [21] we only have results for two images, fig. 6 and 2 (see also fig. 5 in [21]). However [5], to which we compare to, has shown that they slightly outperform [21]. The reason for including matting systems [22, 9, 11, 5, 4] in this comparison is to show that we gain not only in speed but also in quality in terms of the final α matte. The trimap error rate was defined in sec. 2.2 (measured in percentage wrt image size), and the error for an α matte is defined below. The trimap error for systems which directly produce a matte was done by transforming the matte into a trimap. In order to compute a matte from our trimaps, and those of [7], we use our matting approach with low resolution input and without sparsity prior (essentially [22]). For this experiment the input was the set of user-defined scribbles.

Qualitative results are in fig. 1,2, and 6, and quantitative results are in table 1. We see that matting systems [22, 9, 11, 5] are obviously considerably slower. Note, our approach and [7] need for small resolution alpha matting additional 3.5sec on average (not reported in table 1). Also, we see that our method with optimal λ takes on average 0.8sec longer to compute all solutions for the range of $\lambda \in (0, 5)$, but obviously it improves the usability of our method. Considering error rates, we see a correlation between the trimap- and α matte error, which motivates our heuristically defined error functions. We see that our system clearly outperforms all other approaches both in terms of trimap error and α matte errors. Also, predicting λ in our system works better than using a fixed λ. As expected, choosing for each image the optimal λ gives best performance. Finally, ground truth trimaps (last row) give by far the lowest α matte errors, which shows that the problem of good trimap generation is vital for successful α matting. Note, the low rate of [11] might be explained by the fact that it was not designed for a scribble-based interface, but a matting component picking interface. It is not even guarantied that the scribbles will be assigned to the correct α.

Comparison of Trimap-based Matting Methods. We used the following error function for the α matte, which penalizes more heavily an over-estimation of α:

$$\frac{100}{n}\sum_i [1.5\delta(\alpha_i \geq \alpha_i^{true}) + 0.5\delta(\alpha_i < \alpha_i^{true})]|\alpha_i - \alpha_i^{true}|,$$

where α^{true} is the true α and n the number of pixels in U.

It has been shown in [22, 11] that [22, 9] are the best performing methods for this task. Fig. 7 and 5 show qualitative results of [22, 9, 11] on crops of high resolution images. Since we were not able to get all competitors working for our high resolution images, we adapted our method to simulate: (i) [22], by removing our sparsity prior, and (ii) [9], by setting W_F, W_B to $\mathbf{0}$ in eq. 5. Quantitative results for high resolution images are shown in table 2 for different trimaps: ground truth, our trimaps (optimal λ)[6], and small(large) trimaps (dilation of the ground truth trimap by 22(44) pixels). We see that we outperform other methods, more significantly for larger trimaps. The improvements might look small but it is very important to note that our results are overall considerably less blurry than others (e.g. fig. 7). This visual improvement is, however, not captured well in our error metric, which motivates new research in this field. Table 2 also shows that our scribble-based trimaps are better than large, hand drawn trimaps with a brush of radius 88 respectively. Note, the α mate error in table 1 and 2 can deviate due to differently sized input images. Finally, in our un-optimized implementation a full 7 Mpix α matte computation takes between $10 - 25$ minutes depending on the size of the unknown region.

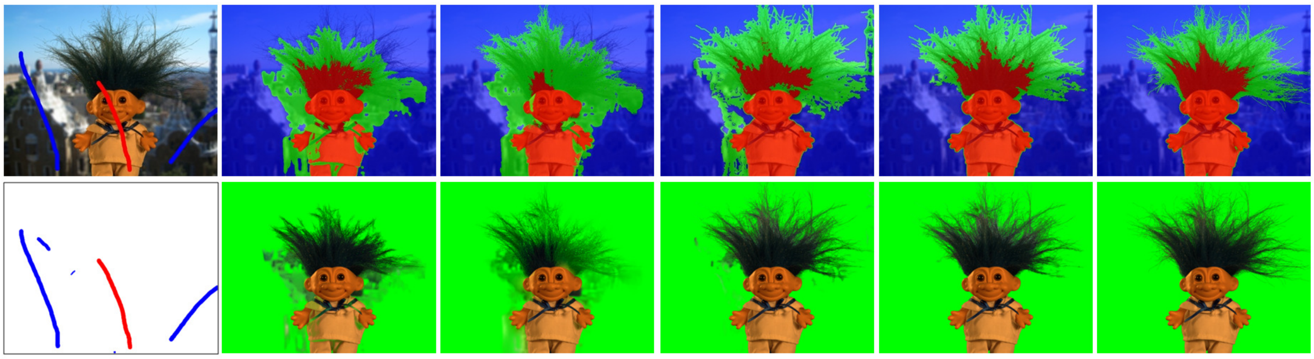
5. Conclusions

We have presented a new approach for alpha matting of high resolution images via interactive trimap extraction. Using our new database we could confirm that we improve on state-of-the-art methods for both trimap extraction and alpha matting. The main contribution for matting was an α

[4] For [11] we had to even scale down the images to 0.15 Mpix. For comparison, the obtained result was then up-scaled to 0.3 Mpix images.

[5] [22, 9, 11, 5, 4] was the authors implementation and [7] our own.

[6] For a fair comparison of our trimaps to dilated trimaps, we brushed additionally over false negative pixels, i.e. pixels marked as unknown in the ground truth trimap but not in our trimap, with a brush of radius 44. On average we had to brush only 0.5% of pixels in the image. Note, a small dilated trimap contains 15% of all image pixels.



Top: 3 Scribbles Bottom: 6 Scribbles

[Wang & Cohen '07] (14.4;8.2)

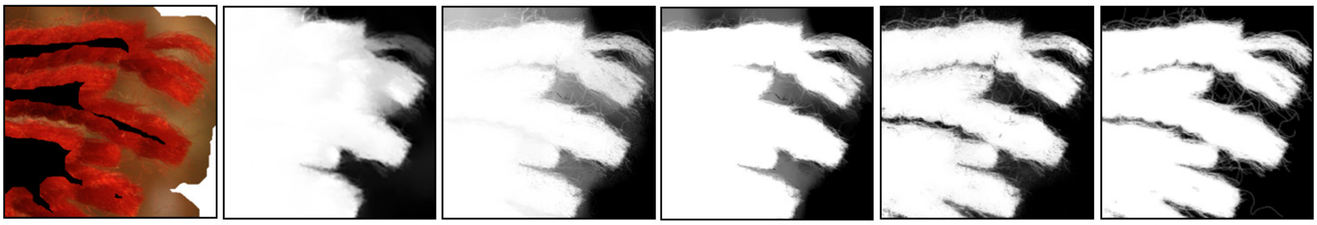
[Levin et al. '06] (13.1;7.4)

Our (pred. λ=3.5) (5.9;2.0)

Our (opt. λ=3 & 6 scribbles) (4.9;1.8)

From GT trimap (0.0;0.93)

Figure 6. **Comparison of trimap methods**. In this example the object has a combination of sharp edges and large fuzzy regions. The first and second row depict the trimaps and final composites respectively. All results were generated from the scribbles shown in the top left image, except of column 5 where we adjusted λ and used 3 extra scribbles (bottom left image) to demonstrate the capability of our method to easily generate an almost perfect result. In brackets is our trimap and α matting error. Our results outperform the competitors, where we show only the top 2, the others were worse, both in terms of error rates and visually (see supplementary material).



(a) Cropped image

(b) [Levin et al. '07] (25.4)

(c) [Levin et al. '06] (21.8)

(d) [Wang et al. '07] (15.1)

(e) Our result (4.2)

(f) Ground Truth

Figure 7. **Comparison of α matting methods**. (a) A crop of a 7.7 Mpix image of a region with a woollen scarf. The input trimap (small dilation of 22 pixels) is superimposed: black(inside) and white(outside). (b-f) Results of various methods with respective α matte error.

prior which imposes sparsity on α while preserving gradients.

References

[1] X.Bai and G.Sapiro. A geodesic framework for fast interactive image and video segmentation and matting.ICCV 07. 1

[2] Y. Boykov and M. Jolly. Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images. In *ICCV*, 2001. 1, 3

[3] Y. Chuang, B. Curless, D. Salesin, and R. Szeliski. A bayesian approach to digital matting. In *CVPR*, 2001. 1

[4] L. Grady, T. Schiwietz, S. Aharon, and R.Westermann. Random walks for interactive alpha-matting.VIIP 05. 1, 7

[5] Y. Guan, W. Chen, X. Liang, Z. Ding, and Q. Peng. Easy matting: A stroke based approach for continuous image matting. In *Eurographics*, 2006. 1, 2, 7

[6] J. Jia. Single image motion deblurring using transparency. In *CVPR*, 2007. 5

[7] O. Juan and R. Keriven. Trimap segmentation for fast and user-friendly alpha matting. In *VLSM*, 2005. 2, 3, 4, 7

[8] V. Kolmogorov, Y. Boykov, and C. Rother. Applications of parametric maxflow in computer vision. ICCV '07. 2, 3, 4

[9] A. Levin. D. Lischinski, Y. Weiss. A closed form solution to natural image matting. In *CVPR '06*. 1, 2, 4, 5, 7

[10] A. Levin, R. Fergus, F. Durand, and W. T. Freeman. Image and depth from a conventional camera with a coded aperture. In *SIGGRAPH*, 2007. 5

[11] A. Levin, A. Rav-Acha, and D. Lischinski. Spectral matting. In *CVPR*, 2007. 1, 5, 6, 7

[12] H. Lombaert, Y. Sun, L. Grady, and C. Xu. A multilevel banded graph cuts method for fast image segmentation. In *ICCV*, 2005. 2

[13] E. Mortensen and W. Barrett. Intelligent scissors for image composition. *SIGGRAPH*, 1995. 1

[14] A. Pentland. A new sense for depth of field. *PAMI*, 1987. 5

[15] C. Rhemann, C. Rother, A. Rav-Acha, and T. Sharp. High resolution matting via interactive trimap segmentation. Technical report, 2008. 2, 4

[16] C. Rother, V. Kolmogorov, and A. Blake. Grabcut - interactive foreground extraction using iterated graph cuts. *SIGGRAPH*, 2004. 1, 2, 3, 4

[17] M. Ruzon and C. Tomasi. Alpha estimation in natural images. In *CVPR*, 2000. 1

[18] A. Smith and J. Blinn. Blue Screen Matting. In *SIGGRAPH*, 1996. 6

[19] S. Vicente, V. Kolmogorov, and C. Rother. Graph cut based image segmentation with connectivity priors. CVPR '08. 4

[20] J. Wang, M. Agrawala, and M. F. Cohen. Soft scissors : An interactive tool for realtime high quality matting. *SIGGRAPH*, 2007. 1

[21] J. Wang and M. F. Cohen. An iterative optimization approach for unified image segmentation and matting. In *ICCV*, 2005. 1, 2, 7

[22] J. Wang and M. F. Cohen. Optimized color sampling for robust matting. In *CVPR*, 2007. 1, 2, 5, 6, 7

[23] Y. Wexler, A. Fitzgibbon, and A. Zisserman. Bayesian estimation of layers from multiple images. In *ECCV '02*. 3