GEODESIC MATTING: A FRAMEWORK FOR FAST INTERACTIVE IMAGE AND VIDEO SEGMENTATION AND MATTING

By

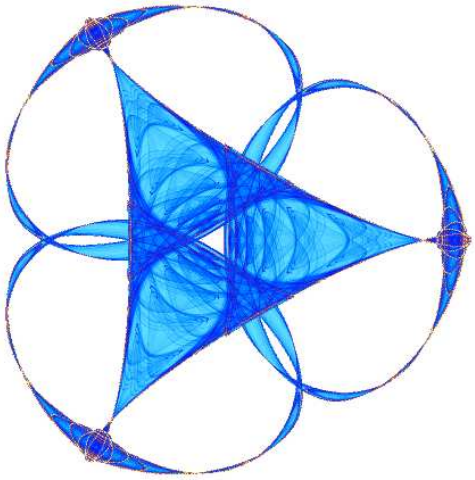
Xue Bai

and

Guillermo Sapiro

IMA Preprint Series # 2185

(January 2008)



INSTITUTE FOR MATHEMATICS AND ITS APPLICATIONS

UNIVERSITY OF MINNESOTA

400 Lind Hall
207 Church Street S.E.
Minneapolis, Minnesota 55455–0436

Phone: 612-624-6066 Fax: 612-626-7370
URL: http://www.ima.umn.edu

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
JAN 2008		**00-00-2008 to 00-00-2008**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Geodesic Matting: A Framework for Fast Interactive Image and Video Segmentation and Matting (PREPRINT)	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **University of Minnesota,Institute for Mathematics and its Applications,207 Church Street SE,Minneapolis,MN,55455-0436**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

An interactive framework for soft segmentation and matting of natural images and videos is presented in this paper. The proposed technique is based on the optimal, linear time, computation of weighted geodesic distances to user-provided scribbles, from which the whole data is automatically segmented. The weights are based on spatial and/or temporal gradients, considering the statistics of the pixels scribbled by the user, without explicit optical flow or any advanced and often computationally expensive feature detectors. These could be naturally added to the proposed framework as well if desired, in the form of weights in the geodesic distances. An automatic localized refinement step follows this fast segmentation in order to further improve the results and accurately compute the corresponding matte function. Additional constraints into the distance definition permit to efficiently handle occlusions such as people or objects crossing each other in a video sequence. The presentation of the framework is complemented with numerous and diverse examples, including extraction of moving foreground from dynamic background in video, natural and 3D medical images, and comparisons with the recent literature.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			
unclassified	**unclassified**	**unclassified**	**Same as Report (SAR)**	**27**	

Geodesic Matting: A Framework for Fast Interactive Image and Video Segmentation and Matting

Xue Bai and **Guillermo Sapiro**
Electrical and Computer Engineering
University of Minnesota
Minneapolis, MN 55455
{baixx015,guille}@umn.edu

January 3, 2008

Abstract

An interactive framework for soft segmentation and matting of natural images and videos is presented in this paper. The proposed technique is based on the optimal, linear time, computation of weighted geodesic distances to user-provided scribbles, from which the whole data is automatically segmented. The weights are based on spatial and/or temporal gradients, considering the statistics of the pixels scribbled by the user, without explicit optical flow or any advanced and often computationally expensive feature detectors. These could be naturally added to the proposed framework as well if desired, in the form of weights in the geodesic distances. An automatic localized refinement step follows this fast segmentation in order to further improve the results and accurately compute the corresponding matte function. Additional constraints into the distance definition permit to efficiently handle occlusions such as people or objects crossing each other in a video sequence. The presentation of the framework is complemented with numerous and diverse examples, including extraction of moving foreground from dynamic background in video, natural and 3D medical images, and comparisons with the recent literature.

1 Introduction

The segmentation of natural images and videos is one of the most fundamental and challenging problems in image processing. One of its applications is to extract the foreground object (or object of interest) out of the cluttered background, and, for example composite it onto a new background without visual artifacts (see also [5] for additional applications in video). For complex images, as well as subjective applications, there can be more than one interpretation of the foreground or objects of interest (in absence of higher level knowledge), thus making the task ill-posed and ambiguous. It is often imperative then to incorporate some user intervention, which encodes prior information, into the process. Specifically, the user can draw rough scribbles labelling the regions of interest and then the image/video is automatically segmented. The user is allowed to then iteratively add more scribbles to achieve the desired result, although of course, the goal is to minimize as much as possible the user effort.

Closely connected to the segmentation of objects of interest, image and video matting refers to the process of reconstructing the foreground/background components and the alpha value (transparency) of each pixel. This is important for applications such as extracting hair strands or blurry edges, as well as for compositing. Being inherently under-constrained (solving for three components, F (foreground), B (background), and α transparency, with only the observed color), the matting problem also requires priors, such as user interactions, which could be in the form of scribbles as in the segmentation task, or a complete trimap.

In addition to the standard high quality and accuracy, the overall requirements of an interactive segmentation and matting framework include the ease of scribble marking and adding, with as minimal as possible user input, as well as real-time processing for making the system truly interactive. These requirements are addressed in this paper. In particular, we propose a fast weighted-distance-based technique for image and video segmentation and matting from very few and roughly placed user scribbles (often just one scribble for the foreground and one for the background). The distance (geodesic) computation is linear in time, and thereby optimal (with minimal memory requirements as well). The weights are based on simple properties such as spatial and temporal gradients which consider the statistics of the user marked pixels, while more sophisticated features can be naturally included as well. The proposed framework can handle diverse data, including natural and 3D images, dynamic background, moving cameras, and objects crossing each other in the video.

Following a brief literature review, Section 2, we describe the proposed framework for segmenting and matting still images, Section 3. This includes both global segmentation and local refinements, all automatically obtained in linear time from a few scribbles provided by the user. We also describe how the user can add scribbles to further improve the results in localized regions. Numerous examples and comparison with the literature are presented in this section as well. Then, we extend the proposed approach to video applications, where a video can be processed with little user interaction, Section 4. We explain how to add constraints to the distance computation to handle moving objects occluding each other, e.g., people/objects crossing each other. We illustrate how the framework can be readily extended to 3D segmentation in Section 5, and conclude and discuss future research in Section 6.

Before proceeding, let us explicitly present some of the key attributes of the proposed framework:
1. It is based on weighted distance functions (geodesics), thereby solving a first order geometric Hamilton-Jacobi equation in computationally optimal linear time. This makes the proposed framework natural for user-interactive processing of images and videos.
2. It produces very good, state-of-the-art results, with very few and intuitive user provide scribbles and very simple attributes defining the weights in the distance computation. We often use just a couple of rough scribbles for still images (one for the foreground and one for the background), and scribble very few key frames for our tested video segments of 70-90 frames in this paper.
3. It exploits both the scribbles locations and the statistics of the pixels marked by the user, thereby utilizing all the information provided by him/her.
4. It applies to a large class of natural data, and since it avoids off-line learning, it is not limited to pre-observed and classified classes and to the availability of ground-truth and hand segmented data.
5. It can handle dynamic background in video as well as crossing objects of interest.
6. The framework is general so that additional attributes can be naturally included in the weights for the geodesic distances if so required for a particular type of data.

2 Related work

One important class of related works is based on energy formulations which are minimized via discrete optimization techniques. The pioneering graph cuts technique, [8], addresses the foreground/background interactive segmentation in still images via max-flow/min-cut energy minimization. The energy balances between the probability of pixels belonging to the foreground (likelihood) and the edge contrast, imposing regularization. The user-provided scribbles collect statistical information on pixels and also serve as hard constraints. The Grabcut algorithm, [21], further simplifies the user interaction. Scribbles can be interactively added to improve the initial segmentation. Full color statistics are used, modeled as mixtures of Gaussians (here, in contrast, we use fast kernel density estimation), and these are updated as the segmentation progresses. This can help but also hurt by propagating segmentation errors. Very good and fast results were demonstrated with

this technique. A number of methods have been proposed extending this framework, aiming at devising more sophisticated energy formulations and at extending it to higher dimensions (video). The Bilayer approach, [12], segments videos with basically static background. It incorporates an additional second order temporal transition prior term and a motion likelihood term. Each frame is segmented via graph cuts, conditionally dependent on the previous two frames. Although excellent results are reported for a particular type of videos, this method makes assumptions about the different behaviors of foreground and background pixels and deals with videos with mostly static backgrounds (they do permit a moving object in the background as long as it is different enough from the foreground). Moreover, it needs to learn the motion statistics, which is very useful as they have cleverly incorporated in their system, but requires the availability of pre-segmented ground-truth training data and of video classes (to train and apply with videos having the same type of motion).

Interactive video cutout, [28], presents a system where the user draws scribbles in 3D space. A hierarchical mean-shift preprocess is employed to cluster pixels into super-nodes, which greatly reduces the computation of the min-cut problem.

In [15], the author uses random walks for soft image segmentation. Each pixel is assigned the label with maximal probability that a random walker reaches it when starting from the corresponding scribbles. The authors of [30] propose an MRF framework to solve segmentation and matting simultaneously. The basic idea is to minimize the fitting error of the matte while maintaining its smoothness. The uncertainties (0 for the scribbles and 1 for all unknown pixels) are propagated to the rest of the image using belief propagation. Once the alpha values are found, the F and B components are estimated. In [17], a local linear relation between the alpha values and image intensities is assumed, that is, the pixel's alpha value can be immediately determined in a local region if its intensity is known. The matting problem is solved by minimizing a cost function combining the prediction error, the regularization of alpha values, and the user-supplied scribbles which indicate constraints to the optimization problem.

Poisson matting, [24], and Bayesian matting, [11], are two important matting techniques that use trimaps as inputs. Poisson matting computes the alpha matte by solving the second order Poisson equation with Dirichlet boundary conditions.[1] An assumption is made by neglecting the gradients of F and B, considering the matte gradient proportional to the image gradient. Additional operations are performed to adjust to local regions. Bayesian matting simultaneously estimates F, B, and α by maximizing a posterior probability. For each pixel in the trimaps region, it models the known F and B colors around as mixture of oriented Gaussians in color space (again, we use fast kernel densities instead). An (F, B, α) triplet is computed as the one that most probably generates the observed color of that pixel. This technique is applied to videos in [10], where the trimap is temporally propagated using optical flow and the matte is pulled out individually in each frame by the Bayesian matting algorithm. Explicit optical flow is not used in our method, although it could be incorporated as part of the weights in the geodesic computation.

The spectral matting technique, [18], automatically computes a set of soft matting components via a linear transformation of the smallest eigenvectors of the matting Laplacian matrix [17]. These components are then selected and grouped into semantically reasonable mattes either in an unsupervised or supervised fashion. The main drawback of this algorithm is its high computational cost – it takes several minutes to compute the matting components for small sized images. In addition, it is not intuitive where to place the constraints. The authors of [29] proposed an improved color sampling method for natural image matting, and demonstrated very good performance. The authors in [27] implemented an interface for interactive realtime matting. The user roughly tracks the boundary with a self-adjustable brush. Like in [29], the matte is pulled out in local regions, solving a soft graph-labelling problem. Flash cut, [25], extracts the foreground layers of flash/no-

[1] Note that in contrast with this, we solve a first order Hamilton-Jacobi equation, which is computationally more efficient.

flash image pairs, using the prior information that only the foreground is significantly brightened. This information is incorporated in an graph cut energy framework. The segmentation algorithm is shown to tolerate some amount of foreground motion and camera shake.

Recently, the authors of [23] unified the graph cuts and the random walker algorithms into a common energy minimization formulation, with l_1 and l_2 norms respectively. They then suggested a third algorithm by extending the framework to the l_∞ case, and demonstrated better segmentation results.[2] The new approach is intrinsically equivalent to a shortest path (geodesic) framework, as here proposed. Finally, fast distance computations for segmentation are also considered in [14] and related papers by the authors.

Our work is inspired by [33, 34], where the authors, following [16], show how to use distance functions for image colorization. As here, these distances are optimally computed in linear time [32]. This was then extended in [20] for segmentation. In contrast with this work, we use significantly less scribbles per image while at the same time producing more accurate results (thanks in part to a more efficient modelling of the corresponding probability distribution functions), see Figure 15; add local refinements via a sliding window and second (user-free) local iteration with a variable width band at the foreground/background boundary; extend the work to video and 3D; and also produce explicit mattes (F, B, and α).

A preliminary version of this work was introduced in [6]. We provide additional algorithmic and computational details, extend the framework considering local sliding windows and variable width bands at the critical foreground/background boundaries for accurate classification and matting, provide numerous additional examples, including comparisons with the literature and ground truth, as well as new three dimensional results.

3 General framework: Still images

In this section we present the core of the proposed framework. We start by describing the segmentation approach, and follow it by the matting technique. Numerous results and comparisons with the literature accompany the presentation of the framework.

3.1 Segmentation

As discussed in the introduction, our algorithm starts from two types of user-provided scribbles, $\mathcal{F}$ for foreground and $\mathcal{B}$ for background, roughly placed across the main regions of interest. Now the problem is how to learn from them and propagate this prior information/labeling to the entire image, exploiting both the marked pixel statistics and their positions. We address these issues next. We then detail how the user can add scribbles to further improve the results.

3.1.1 Feature distribution estimation

The role of the scribbles is two fold. The scribbles indicate spatial constraints, and also collect labelled information from $\mathcal{F}/\mathcal{B}$ regions. We use discriminative features to learn from the samples on the scribbles (pixels marked by the user via the scribbles), and to classify all the remaining pixels in the image. For most of our examples, we use the *Lab* color vector to provide the basic features learned from the scribbles, though this framework is not limited to color features. The user can select other types of features (or their combination) depending on the particular application.[3] For example, we can use the frequency response of Gabor filters for texture images, as in [20], or

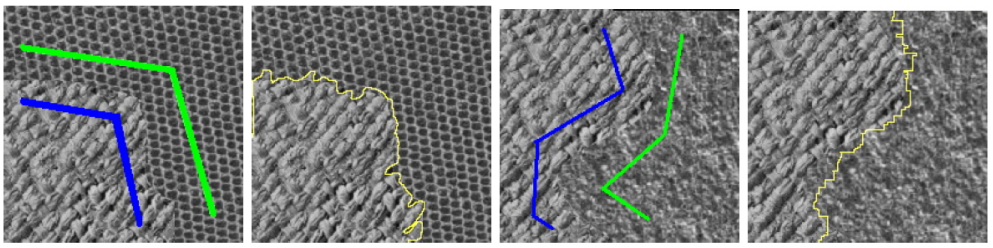
[2] The l_∞ norm has been shown to be critical for many image processing applications, in particular interpolation, following a very elegant axiomatic approach [9].

[3] Initial considerations on how to automatically learn the most relevant features were presented in [20], while future work in this direction needs to be pursued.

optical flow for image sequences of moving objects (see figures 1 and 2).[4] The actual segmentation algorithm that exploits these features will be introduced in the following sections.

(a) (b) (c) (d)

Figure 1: *Segmentation obtained while using optical flow as feature. (a-b) Two consecutive frames from a video clip. Scribbles (blue for foreground and green for background) are placed on one frame. (c) Computed optical flow, obtained from the publicly available code [7]. (d) Segmented foreground.*



(a) (b) (c) (d)

Figure 2: *Segmentation obtained using texture filter responses as features. (a) and (c) Gray scale texture images and user provided scribbles. (b) and (d) Segmentation results using Gabor filter responses.*

We estimate the probability density of the selected features, optical flow or filter responses in the previous examples and the 3D *Lab* color vector for most of the other examples presented in this paper, of the pixels covered by scribbles, via the fast kernel density estimation [31]. Let $\Omega_{\mathcal{F}}$ be the set of pixels with label/scribble $\mathcal{F}$ and $\Omega_{\mathcal{B}}$ those corresponding to the background scribble $\mathcal{B}$. The likelihood of a pixel x with color $\vec{c_x}$ to belong to the foreground becomes

$$P_{\mathcal{F}}(\vec{c_x}) = \frac{Pr(\vec{c_x}|\mathcal{F})}{Pr(\vec{c_x}|\mathcal{F}) + Pr(\vec{c_x}|\mathcal{B})}, \tag{1}$$

where $Pr(\vec{c_x}|\mathcal{F})$ is the color PDF, estimated via the fast kernel method, of $\Omega_{\mathcal{F}}$ (same process for the background PDF). See Figure 3 for an illustration.

In our experiments, we found that a highly accurate probability estimation is usually not completly necessary in this step. Thus we decompose the color space and estimate the probabilities in three independent channels, further speeding-up the already efficient KDE algorithm. The full color model is used in the refinement stage, Section 3.2. A median filter is optional to smooth the estimated probabilities.

3.1.2 Geodesic distance

We use the geodesic distance from these user-provided scribbles to classify the pixels x in the image (outside of the scribbles), labelling them $\mathcal{F}$ or $\mathcal{B}$. The geodesic distance $d(x)$ is simply the smallest integral of a weight function over all possible paths from the scribbles to x (in contrast with the

[4] All the figures in this paper are color figures.

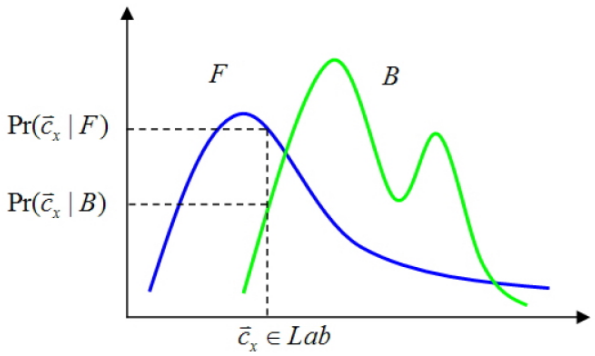


Figure 3: *Estimated probability densities of foreground samples (blue) and background samples (green), shown in one dimension for best visualization.*

"average" distance as used in random walks or diffusion/Laplace based frameworks). Specifically, the weighted distance (geodesic) from each of the two scribbles for every pixel x is

$$D_l(x) := \min_{s \in \Omega_l} d(s, x), \quad l \in \{\mathcal{F}, \mathcal{B}\}, \tag{2}$$

where

$$d(s_1, s_2) := \min_{C_{s_1,s_2}} \int_{s_1}^{s_2} |W(x) \cdot \dot{C}_{s_1,s_2}(x)| dx, \tag{3}$$

where $C_{s_1,s_2}(x)$ is a path connecting the pixels s_1, s_2. In this way we obtain the distance from each user-provided scribble to every image pixel, Figure 4.

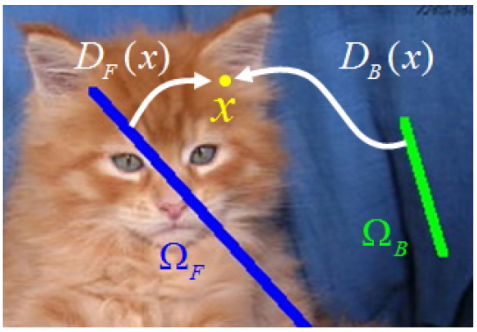


Figure 4: *Weighted distance from pixel x to scribbles.*

The weights W are here selected as the gradient of the likelihood that a pixel belongs to the foreground (resp. background), i.e., $W(x) = \nabla P_{\mathcal{F}}(x)$ (recall that as mentioned above, such weights can depend on other features). Note how in this case we are exploiting from the user-provided scribbles both their actual position and the statistics of the pixel colors marked by these scribbles.[5]

In discrete form and assuming a 4-neighborhood stencil,[6] the image is modelled as four-neighbor-

[5] Although we use in our implementation fix width scribbles, these can vary as well, as it is standard in most commercial packages dealing with computational photography, where the radius of the brush can be selected by the user. This can help in collecting additional pixels for the statistics.

[6] We found this discretization to be sufficient for obtaining high accuracy results, while more sophisticated numerical schemes, still with linear complexity, can be exploited if needed, see [32] for details. Note that with this particular discretization, we simply use the classical linear-time Dial's algorithm for weighted distance computation with integer weights [13, 26], while the general case with enhanced data structures has been developed in [32].

connected graph. The (discrete) geodesic distance then can be approximated as (see Figure 5)

$$d(s_1, s_2) := \min_{C_{s_1,s_2}} \sum_{x,y} W_{xy}, \quad W_{xy} = |P_{\mathcal{F}}(\vec{c_x}) - P_{\mathcal{F}}(\vec{c_y})|, \quad x, y \in C_{s_1,s_2}. \tag{4}$$

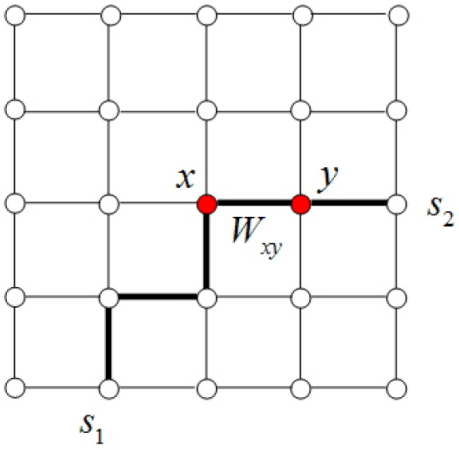


Figure 5: *In discrete form, the image can be modelled as a graph. The integration becomes the sum of edge weights along the path connecting s_1 and s_2.*

Based on this concept of geodesic distances, a pixel is close in this metric to a scribble in the sense that there exists a path along which the likelihood function does not change much, Figure 6(b). Following [32], we can efficiently compute the distances, in optimal linear time, and assign each pixel to the label with the shorter distance:

$$\Omega_{\mathcal{F}} = \{x | D_{\mathcal{F}}(x) \leq D_{\mathcal{B}}(x)\}.^7 \tag{5}$$

By the triangular inequality property of distance functions, we can prove that, under our metric definition, each stroke of scribble results in at most one connected component, regardless of how cluttered the image is (simply assume that a non-connected component is created, and it is easy to see that this will contradict the triangle inequality). Moreover, using once again the triangle inequality, the robustness with respect to the actual position of the scribbles can be explicitly computed as well [34]. This will be illustrated later in this paper.

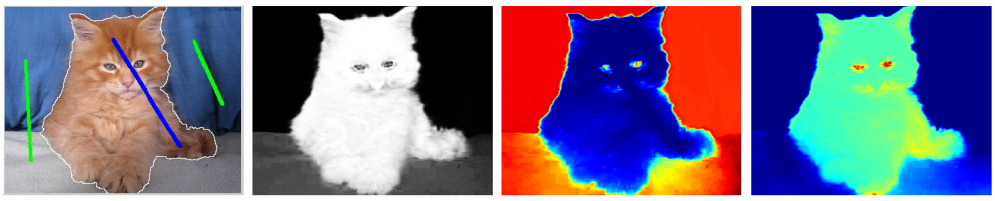


Figure 6: *(a) A binary segmentation (white curve) is quickly found by a few scribbles. (b) $P_{\mathcal{F}}(x)$. Dark indicates low probabilities and white high probabilities. (Note that this is not the final alpha matte.) (c) $D_{\mathcal{F}}(x)$. (d) $D_{\mathcal{B}}(x)$. Blue indicates low distances and red high distances. The binary segmentation is obtained by simply comparing these two weighted distances.*

[7] By a slight abuse of notation, we now use $\Omega_{\mathcal{F}}$ (respectively $\Omega_{\mathcal{B}}$) to denote both the user marked foreground $\mathcal{F}$ (background, $\mathcal{B}$) pixels and those classified as foreground (background) by the algorithm just described.

3.1.3 User interaction

The proposed algorithm allows the users to interactively add new scribbles to achieve the desired segmentation in a progressive fashion (the original graph cut approach and the extensions mentioned before also include the possibility of adding scribbles). Figure 7(a) shows the binary segmentation computed from the initially placed scribbles. $\partial\Omega_{\mathcal{F}}$ denotes the segmentation boundary, shown in a dashed line. Every time, a new scribble is added either to cut off a piece of foreground (meaning that the foreground is larger than desired), or to attach a piece of region to the foreground (meaning that the foreground is smaller than desired). In the former case (Figure 7(b)), a new background scribble B_1 is added to the region that needs to be removed. The weights are updated by computing the probabilities between B_1 and all previously marked foreground scribbles. Then the distances are re-computed using those new weights, but within the current foreground $\Omega_{\mathcal{F}}$. This is in part to protect the other regions from being affected by this local correction. The latter case, Figure 7(c), works in the same fashion, except that now the distance computation is confined within the background mask. These local operations give more accurate probability estimation (for the features) and provide the user with an intuitive tool to repair the segmentation results. Note once again that we exploit both the pixel features and the positions of the newly added scribbles, meaning the user input is helping both in improving and localizing the feature distribution estimation and the local region of intervention. Figure 8 shows an example of how the user removes the undesired regions from the foreground.

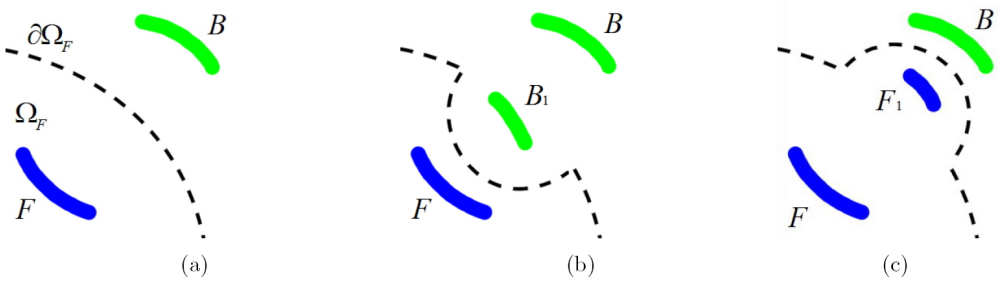


Figure 7: *User interaction, addition of local scribbles to improve local results.*

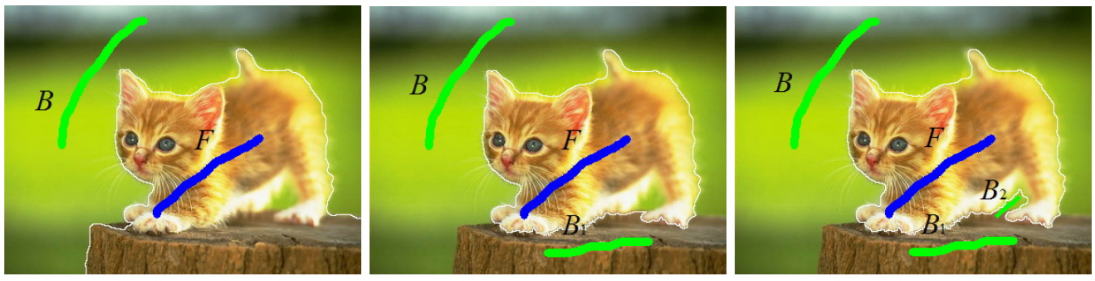


Figure 8: *Example of progressive segmentation. New scribbles B_1 and B_2 are progressively added to remove undesired regions from the foreground. Results are then further improved during the matting process described below.*

3.2 Matting

Image matting refers to the process of estimating the alpha value (transparency) for every pixel, which can be used for example to seamlessly merge the object into a new background. For the refinement of the binary segmentation described above, a narrow band is automatically spanned

across the current $\mathcal{F}/\mathcal{B}$ boundaries (see Figure 9(b)), and its borders serve as new $\mathcal{F}$ and $\mathcal{B}$ scribbles for a second iteration of the algorithm just described. This way the computational cost of the second stage is reduced to just a few pixels in the band, while at the same time refining the likelihood functions (and thereby the weights) and locally adapting them to the region of interest.[8]

Once this refined distance has been obtained following this second step (once again, note that the second step is automatic, directly obtained from the first segmentation), the alpha channel inside the band is explicitly computed as

$$\omega_l(x) = D_l(x)^{-r} \cdot P_l(x), \quad l \in \{\mathcal{F}, \mathcal{B}\}, \tag{6}$$

$$\alpha(x) = \frac{\omega_{\mathcal{F}}(x)}{\omega_{\mathcal{F}}(x) + \omega_{\mathcal{B}}(x)}, \tag{7}$$

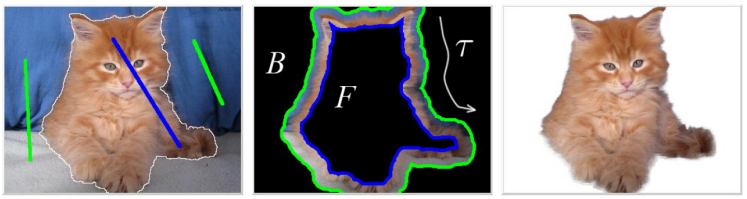
where $P_l(x)$ is locally recomputed using the feature vector (L, u, v, τ), $\tau \in [0, 1]$ parameterizes the band along the boundary (leading to local PDF estimations), and is periodic with period 1 if the curve is closed (see Figure 9(c)). r controls the smoothness of the edges. When $r = 0$, $\alpha(x) = P_{\mathcal{F}}(x)$; when $r \to \infty$, $\alpha(x)$ becomes hard segmentation (typically $0 \leq r \leq 2$ in our examples). This alpha matte combines the weighted distance ($D_l(x)$), measuring how 'close' the pixel is to the scribble, and the probability ($P_l(x)$), based on the fast kernel density estimation (measuring how probable is its color). Note that regularization, e.g., anisotropic diffusion of α, can be applied inside the band as well if needed. Since this is done locally, virtually no computational cost is added.

After the matte α is computed, we follow the method in [30] to estimate the F_x and B_x components (in *Lab* space) for each pixel x inside the band. We randomly sample the foreground and background colors in the neighborhood of x and use the pair that gives the minimal fitting error,

$$(F_x, B_x) = \arg\min_{F_i, B_j} \|F_i\alpha_x + B_j(1 - \alpha_x) - I_x\|, \tag{8}$$

where $i \in N(x) \cap \Omega_{\mathcal{F}}, j \in N(x) \cap \Omega_{\mathcal{B}}$, F_i, B_j are foreground and background colors sampled on the (band boundary) scribbles within the window $N(x)$ centered at x, and I_x is the observed color.

With these components, we can now paste the object onto a new background if desired, with no noticeable visual artifacts by the simple matting equation $C_x^* = F_x\alpha_x + B_x^*(1 - \alpha_x)$, where the composite color C_x^* is a linear combination of foreground color F_x and the new background color B_x^* for every pixel x in the image.



(a) (b) (c)

Figure 9: *(a) Original image and scribbles. (b) Automatically generated trimap, a narrow band around the white curve, and new automatically generated local scribbles (borders of the band). (c) Obtained segmentation and alpha matting.*

Figure 10 (and 9(c)) shows additional results for numerous still images. Note how simple scribbles can handle cluttered and diverse images.

[8]The use of a two-steps process, namely binary segmentation followed by local refinement, is common in matting algorithms, e.g., [21]. A unique characteristics of our approach is that both steps use the same underlying algorithm, the geodesic based segmentation, just with refined features. Additional unique features are the incorporation of the sliding window and the variable width band, inspired by [27], as described below.

Figure 10: *Left column: original images with user-provided scribbles. Blue for foreground and green for background. Middle column: Computed alpha matte. Right column: Foregrounds pasted on constant blue or new image backgrounds.*

While we only have a research code implementation,[9] and as mentioned above the algorithm is linear and thereby computationally optimal, we recorded the actual running time for a few examples in Table 1. The columns correspond to image dimensions, time for the color density estimation, binary segmentation first step, the proposed fix width band matting second step, and the total execution time. The research code is implemented in C++ and tested under Windows environment with 2GHz CPU and 2G RAM. Note that the time cost of matting also depends on the actual contour length and bandwidth (this time is further reduced by the variable width approach described below).

Table 1: *Computational time (in milliseconds).*

image size	density est.	segmentation	matting	total
255×308	31	78	156	256
481×321	16	79	234	329
640×480	32	156	484	672
1600×1200	109	1375	574	2031
2048×1365	172	2375	2282	4829

We should note that we can also use other matting algorithms inside the band, thereby keeping the first step of the framework (binary $\mathcal{F}/\mathcal{B}$ segmentation and band computation), while skipping the second. We now show how to actually further refine the results around this first-step $\mathcal{F}/\mathcal{B}$ boundary, including the use of alternative matting approaches.

3.2.1 Sliding window refinement

To get more accurate estimation inside the band, we propose to update the probabilities using locally sampled scribbles. We only use the $\mathcal{F}/\mathcal{B}$ scribbles (border of the band described above) inside a sliding window centered at the computed boundary $\partial\Omega_{\mathcal{F}}$. This refinement is repeated for equally spaced points $p_1, p_2, p_3, \cdots$ on the boundary until every pixel in the band is covered by those windows (see Figure 11). These refined probabilities can then be used in the matting algorithm just described, or as part of the variable width estimation introduced below.

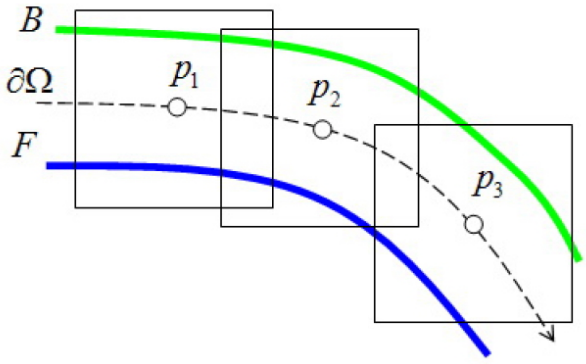


Figure 11: *Sliding window refinement.*

[9]The interested reader can refer to http://www.digitalfilmtools.com/powerstroke/ for a commercial implementation of the colorization work, [34], to get a glance on how fast the geodesic-based approach for image manipulation is.

3.2.2 Variable width band

The uniform band in the second step locates the fuzzy boundary region that needs further refinement. Since it only involves the additional processing of a small number of pixels, the computational time of the matting process (second step) is greatly reduced, while as mentioned above, the results are significantly improved as a result of the localized probability estimations. The authors of [27] devised a brush that automatically adjusts its size to cover this fuzzy regions when travelling along the object boundary (in their proposed framework, the user drags the mouse around this boundary). Following their idea, we propose a variable width band, to replace the fixed width band, adaptive to the boundary. Note that in our case, the boundary and band are automatically computed from the user-provided scribbles. These scribbles require, in most scenarios, significantly less user intervention that dragging the mouse around the possible boundary as in [27].

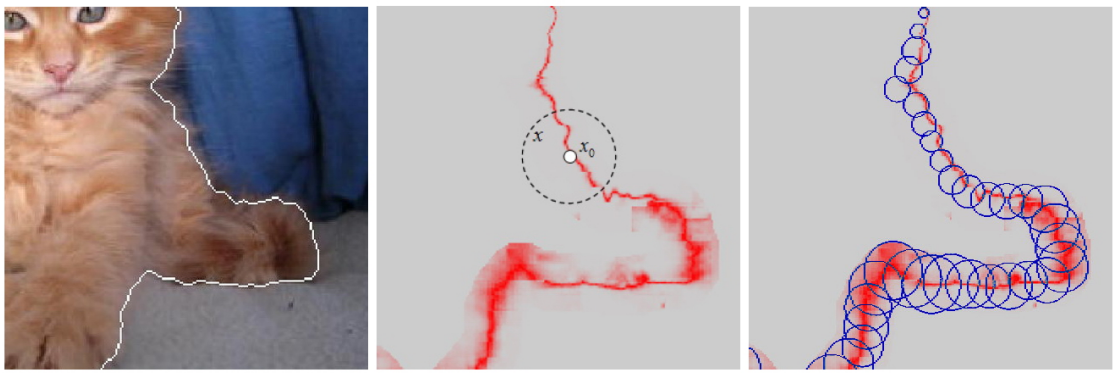
The variable width band, which also eliminates the need to guess the critical width parameter for the fix width one, is computed by

$$\omega(x) := (1 - 2|P(x) - 0.5|)^2, \tag{9}$$

$$\bar{L}(x_0) := \frac{\sum_{x \in N(x_0)} \omega(x) L(x)}{\sum_{x \in N(x_0)} \omega(x)}, \tag{10}$$

$$R(x_0) := 4\bar{L}(x_0) + 4\sqrt{\sum_{x \in N(x_0)} \omega(x)(L(x) - \bar{L}(x_0))^2 / \sum_{x \in N(x_0)} \omega(x)}, \tag{11}$$

where P is either $P_{\mathcal{F}}$ or $P_{\mathcal{B}}$ as computed from the previous sliding window step, $N(x_0)$ is a local window around the pixel x_0 on the boundary being considered, and $L(x)$ is the Euclidean distance from x to the binary segmentation boundary. The brush radius R at x_0 is then computed as a weighted distance inside the neighborhood. The union of all such circles (equally sampled along the boundary), forms the variable width band, see Figure 12 for an illustration. The band adapts to the edges, further reduces the number of pixels to be refined, and produces overall more accurate matting results.



(a) (b) (c)

Figure 12: *Variable band illustration. (a) Binary segmentation. (b) Local window and weights shown in red. 0.5 gray has the highest weight. (c) Circles of different sizes representing the automatically computed variable width band.*

As briefly mentioned above, the band gives the flexibility to plug in other matting algorithm (both scribble-based and trimap-based). This narrow variable width band, being quickly and automatically generated after the user finishes the scribbles, leaves very few pixels to the matting algorithm in the second step, virtually optimizing the computational cost for any natural image

matting algorithm. Figure 13 shows some examples of applying the matting algorithm proposed in [30] inside the variable width band. Figure 14 compares the results between the fixed and the variable width band.

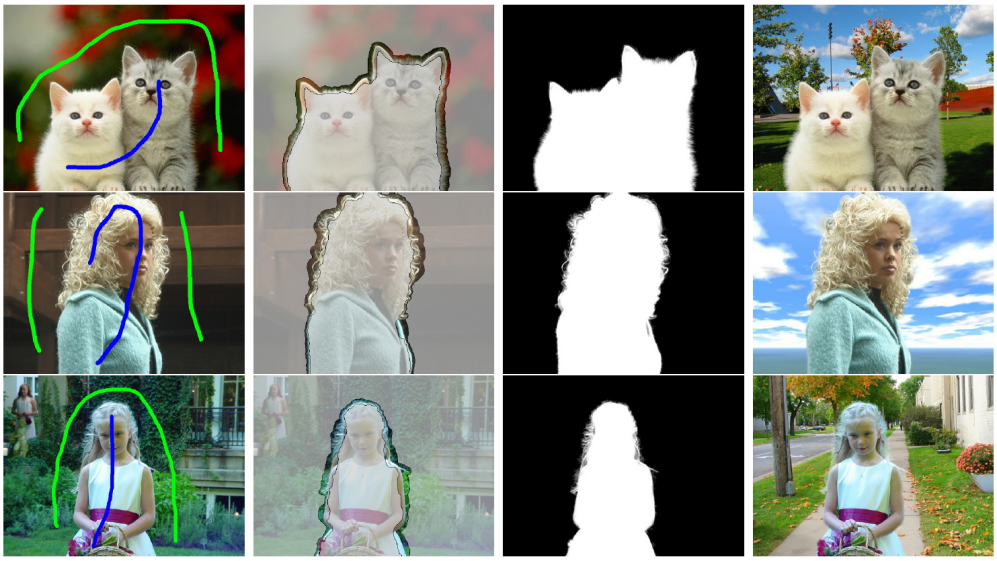


Figure 13: *Variable width band working in conjunction with [30]. First column: Original images and scribbles. Second column: variable width band, automatically computed from the scribbles in the previous column, superimposed on corresponding images. Third and fourth columns: mattes and compositions produced by the robust matting approach described in [30].*

3.3 Additional results and comparison

We now present additional examples and comparisons with a number of leading prior art techniques.[10] Overall, the proposed framework improves over previous work at a number of different levels, often requiring significant less scribbles, reducing the computational time, adapting to a wide range of natural images, and producing the explicit matte.

First, in Figure 15 we compare with the algorithm in [20], which inspired in part this work. Figure 16 then presents comparisons with the work in [30] , Photoshop Extract Filter [2], Photoshop CS3 Quick Selection & Refine Edge tools [4] , Corel Knockout2 [3], Spectral Matting [18],[11] and Power Mask (a commercial software version of the Soft Scissors [27]). Note how our proposed approach needs significantly less scribbles to achieve state of the art results, often also at significantly reduced computational cost.

Next, in Figure 17 we show results of our algorithm with some of the images used in [21] (to the best of our knowledge, not all the images used in that paper are publicly available). The results obtained with our proposed framework are comparable to those published in [21] (for these particular images, [21] marks scribbles on the background only).

We then compare with ground truth for some of the examples in [18], showing once again the high accuracy of the proposed approach.

A key question is how robust is the algorithm with respect to the exact position of the user-provided scribbles. This is tested in Figure 19. Recall that from the triangle inequality, this

[10] We compare both with commercial packages and recently published leading academic contributions. For this group, we only consider algorithms for which the authors have kindly released their code, often using their default parameters.

[11] Comparison with [18] is presented mainly for completeness, since the main goal of [18] is to present an interesting mathematical formulation of matting, extending spectral segmentation techniques.

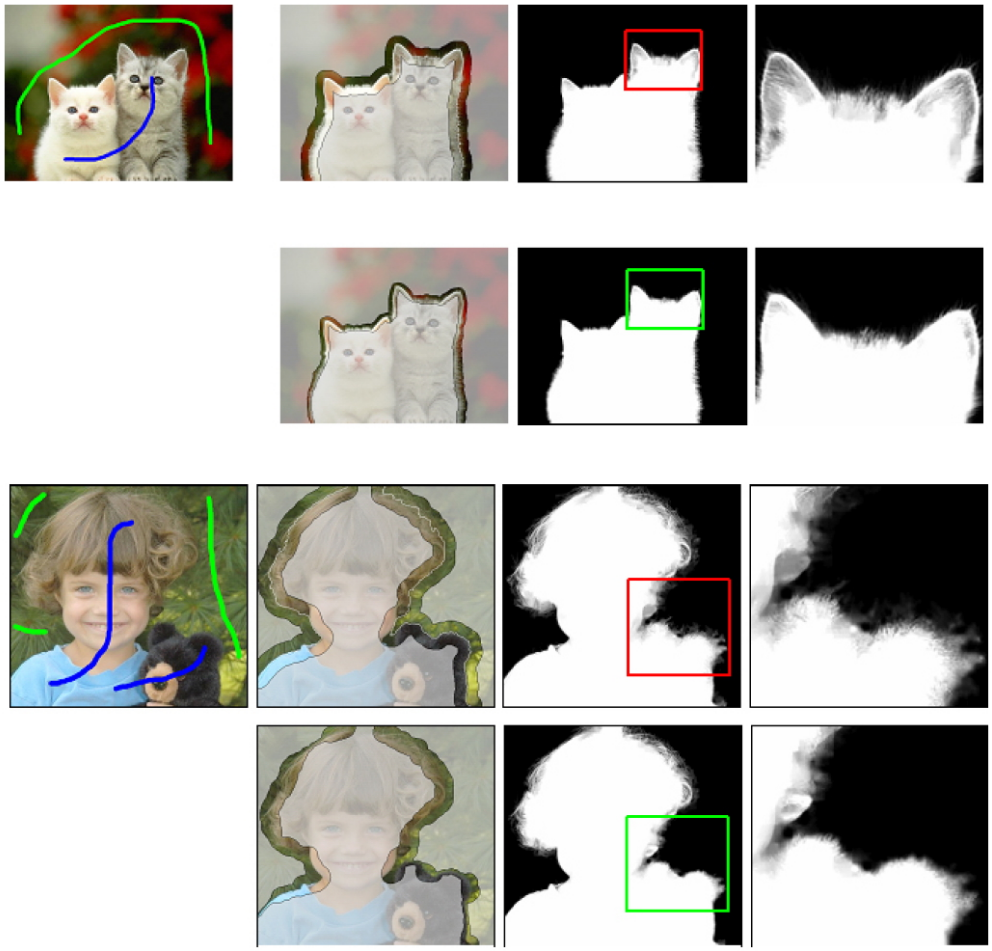


Figure 14: *Comparisons between the use of fix and variable width bands. For each example, the original image and user-provide scribbles is presented first on the left, followed by the obtained band, the computed matte, and a zoom-in image of the marked region (fix width on top and variable width on bottom). Note the significant improvements obtained wit the variable width band.*

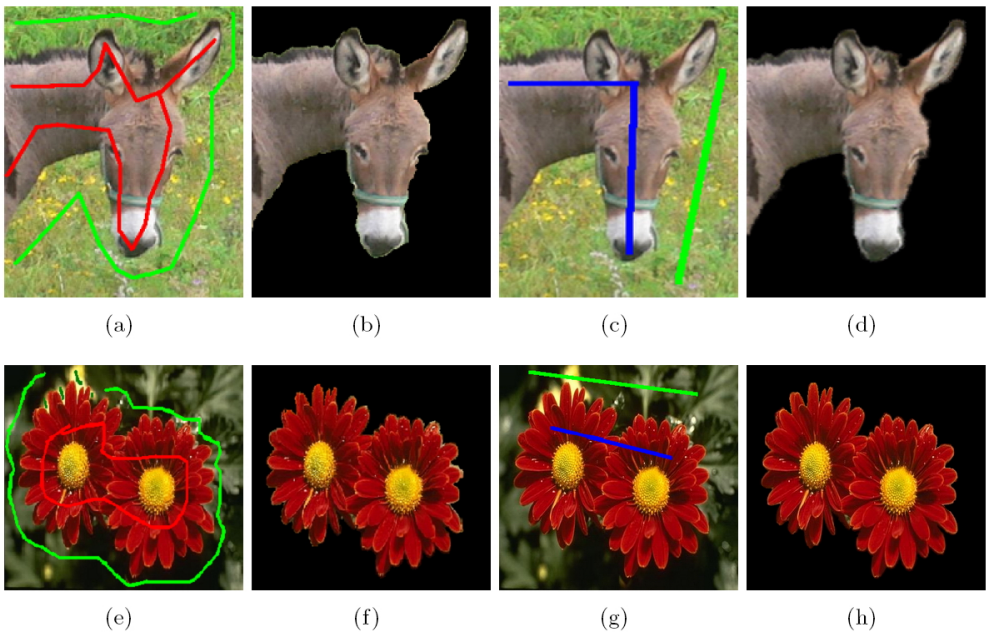


Figure 15: *Figures (a)(b)(e)(f) show the user inputs and results from [20]. Figures (c)(d)(g)(h) correspond to the new inputs and our results for the same images, leading to better performances with less scribbles.*

robustness can be explicitly studied [34].

4 Video segmentation

The above described framework is now extended to videos, modelled as 3D images, in which every pixel has six neighbors, four spatial and two temporal (except the ones on the frame borders). The scribbles, drawn on one or several frames, propagate throughout the whole video by weighted distances in spatial-temporal space, Figure 20. In particular, spatial and temporal gradients of the likelihood function are used to define the weight W in the geodesic computation in Equation (3). Note that there is no explicit use of optical flow in this case (or motion models as in the works described in Section 2), thereby not only simplifying the computations but also permitting to deal with dynamic background and not limiting the work to pre-specified motion classes. As we will see in the experimental section, this simple model is already very useful for numerous scenarios. We now introduce some additional extensions to make it more general.

4.1 Constrained spatial-temporal distance

In still images, a single $\mathcal{F}$ scribble and a single $\mathcal{B}$ scribble always return two connected components. This can be easily proved by the triangle inequality property of the distance function (this, as mentioned above, also helps to prove the robustness of the method with respect to the exact placement of the scribbles, see [34]). If the user marks a circle of $\mathcal{B}$ scribble around the object, all the exterior region will be classified as background. However, this is no longer guaranteed in the 3D spatial-temporal case. Consider the simple scenario in Figure 21. Two objects with similar color/feature distributions move toward each other, cross, and split apart. The inside of the tube has low distances to the $\mathcal{F}$ scribble (shown in red). The $\mathcal{F}$ scribble in object A propagates to the frames with occlusions, and then backwards to object B (B refers to the second object in Figure 21 and not to the background value). Although the user might intend to separate object A as foreground in the initial frame, object B is mistakenly cut out because of the connectivity in 3D

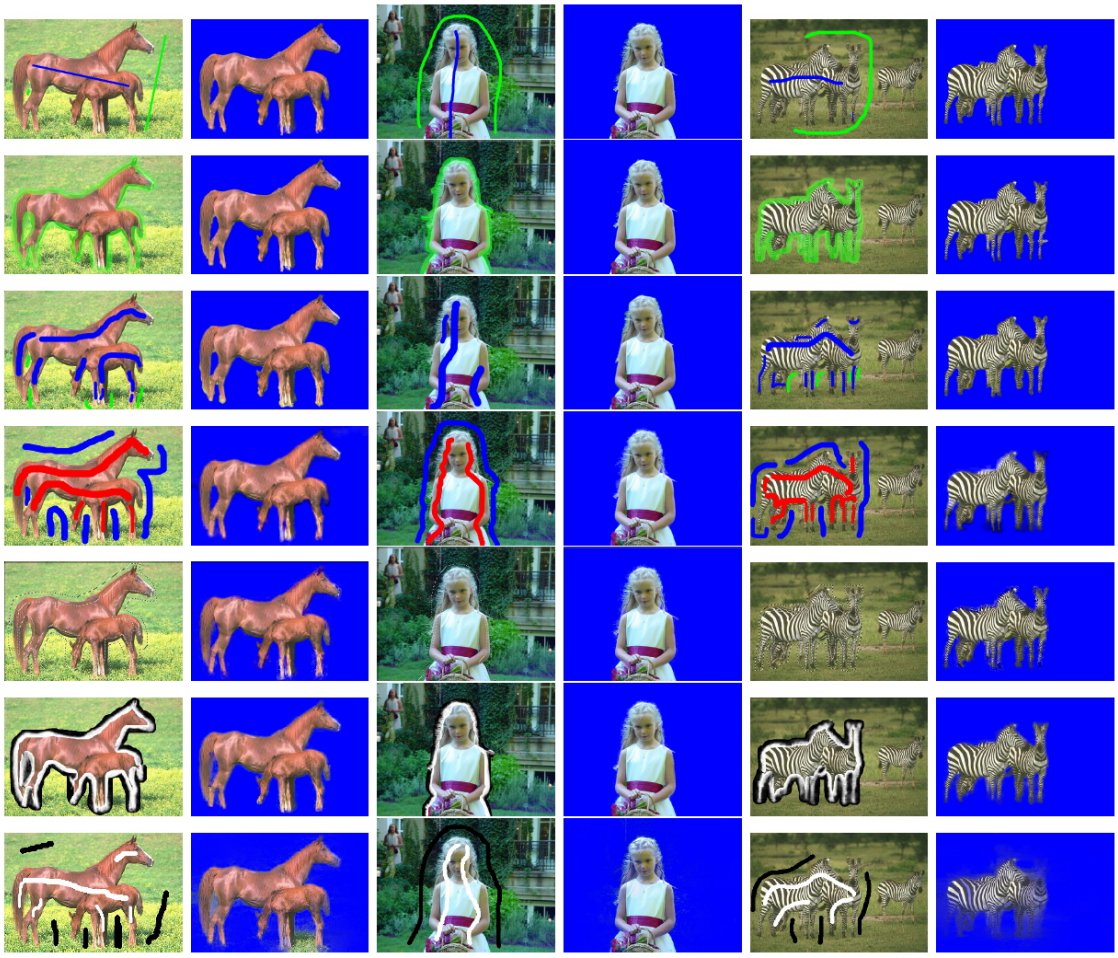


Figure 16: *Comparison with leading user-assisted segmentation and matting algorithms. For each example, we show the scribbles positions and the final matting result on a blue background. 1st row: Our results. 2nd row: Photoshop Extract Filter [2]. 3rd row: Photoshop CS3 Quick Selection & Refine Edge tools [4]. 4th row: Robust Matting [30]. 5th row: Corel Knockout2 [3]. 6th row: Power Mask [27]. 7th row: Spectral Matting [18].*

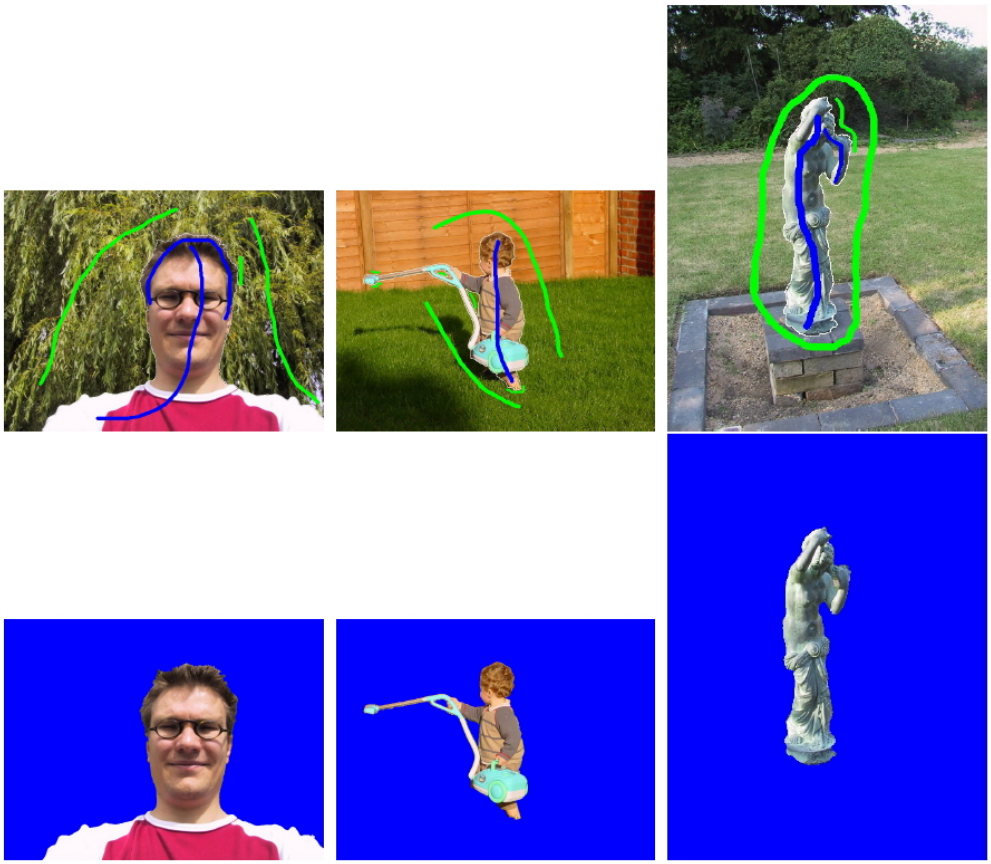


Figure 17: *Results with images from [21].*

Figure 18: *Comparison with ground truth data, obtained from [18]. First column: original images and scribbles. Second column: variable width band. Third column: alpha mattes, produced with [30] inside the computed band. Fourth column: ground truth mattes.*

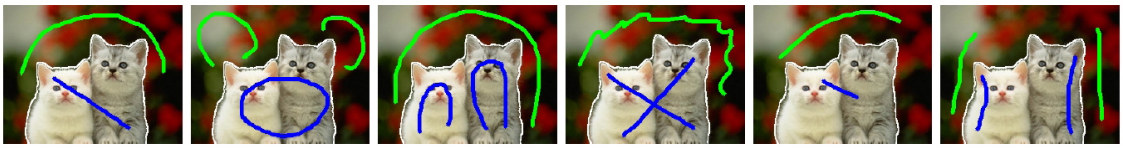


Figure 19: *Scribbles are placed in six different ways, all giving virtually the same segmentation, indicated by the white curves.*

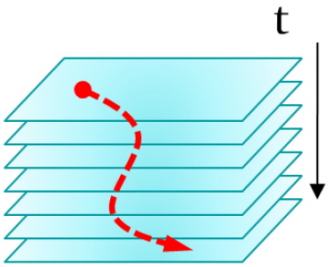


Figure 20: *Scribbles propagate in spatial-temporal space for video segmentation and matting.*

space (such connectivity does not occur in still images). This phenomenon happens when undesired objects in the background touch the foreground in a certain frame, and the error spreads temporally throughout all frames.

We address this problem with very limited extra computation. To eliminate the branch formed by the undesired object before occlusion, we simply constrain the propagation to be temporally non-decreasing, and Equation (3) is replaced by:

$$d(s_1, s_2) := \min_{C_{s_1,s_2}} \int_{s_1}^{s_2} W dx, \ s.t.\ t_1 \leq t_2 \ if \ x_1 \leq x_2, \tag{12}$$

where $x_1, x_2 \in C_{s_1,s_2}$ indicate any two positions on $C_{s_1,s_2}(x)$ and t_1, t_2 are their corresponding time coordinates. In other words, $d(s_1, s_2)$ is minimized among the paths that temporally go forwards. Of course we can also constrain the distance function in the opposite direction. However, it becomes the same definition if we let the video play reversely.

In the discrete scenario, the temporal links (the links that connect temporal neighbors) are replaced by directed links, and this new constrained propagation is simply obtained by setting to infinity the weights of going backwards in time. This simple modification leads to the correct segmentation before the occlusion, but confusion might still exists after the occlusion (Figure 21(c)). We can further remove the wrong branch using the same approach, but now in the opposite direction. This can be done by specifying a point in the desired tube at a latter time, letting it propagate backwards within the tubes, constrained to move only backwards. Figure 21 illustrates the process. As a result, the ambiguity is removed in frames where the objects are disconnected within the frame. The last point can be either manually specified or automatically detected, for example, by a SIFT-feature matching algorithm [19] (see Figure 22).

Figure 23 shows the example of two people walking. The user desires to segment the person initially on the right. The two people are merged as a single object when they cross each other (since they share the features that are used to compute the weighted distance). The second row shows the results using the distance function without the constraint. The wrong segment appears in every frame (again, see Figure 21(a)). The third row shows the result by the constrained distance function. We can see that the error is removed before and after the intersection. Adding scribbles in the intersection frames will manage to separate them also there, see below, but this is left without in this figure to illustrate the power of the "tubing" effect just described.

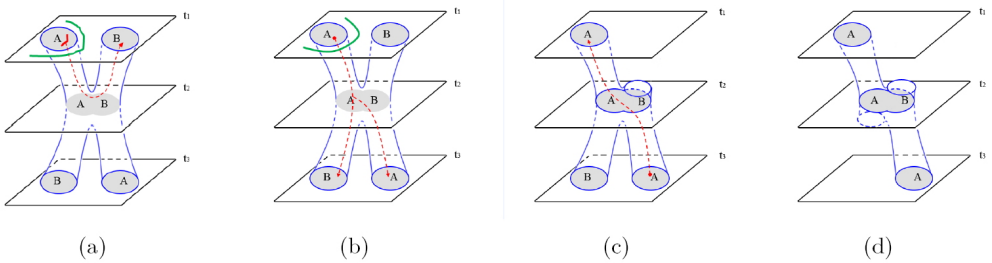


Figure 21: *Tubes in 3D space, where $t_1 < t_2 < t_3$ (a) Although the scribbles in the first frame intend to separate A, the $\mathcal{F}$ scribble (red) reaches the object B by a path in 3D space where both objects A and B overlap. (b) The scribble propagation is constrained to move forward and the branch between t_1 and t_2 is eliminated. (c) The user specifies a pixel in A at t_3 and lets it propagate backwards. The branch of B between t_2 and t_3 is removed. (d) Result with the proper separation of the object A.*

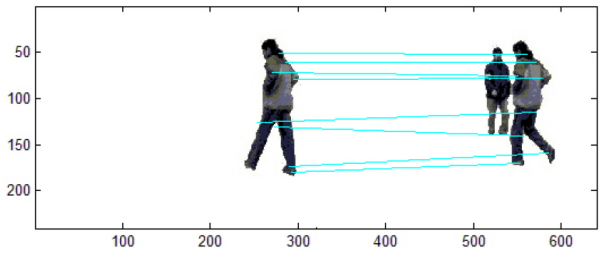


Figure 22: *Feature point matching for automatically constraining the background propagation, as illustrated in Figure 21(c), for the video in Figure 23.*

Figure 23: *A video example of two people crossing. First row: original video. Scribbles drawn on the first frame. Second row: Segmented results using unconstrained distance function. Third row: Segmented results using constrained distance function. See text for details.*

4.2 Interactive refinement

For individual frames where occlusion actually happens and can not be fixed by the "tubing" approach described above, the user simply provides extra scribbles to segment the object. Since the color distribution might be inadequate to differentiate the objects (this is what led to their merge in the first case), we switch to another contrast sensitive weight to be used for the geodesic distance computation in Equation (3). This shows the power of the framework, features can be adapted to the problem at hand. For discrete images, the new feature is defined as $W_{xy} := \| \bar{c_x} - \bar{c_y} \|$, where x and y are two adjacent pixels and $\bar{c}$ is the color vector in Lab space. Figure 24 shows how the user separates the two persons using the new weights.

(a) (b) (c)

Figure 24: *(a) Original segmentation obtained by gradients of the PDF. (b) The user adds new scribbles. (c) Segmentation results obtained with the new geodesic distance.*

4.3 Additional video experimental results

We test our algorithm on five additional video segments of 71, 79, 78, 44, and 94 frames respectively. We mark scribbles on very few frames (2, 1, 1, 3, and 4 frames respectively). The results are shown in figures 25-29 as image sequences sampled every few frames (please see videos uploaded with the supplementary material to appreciate the moving camera and dynamic background). The rows correspond to the original frames, alpha matte, composites on a white background, and composites on a new movie.

We compare our approach with the rotoscoping algorithm in [5] for the video in Figure 25 (we only refer to the segmentation/tracking part, which is the contribution of our paper, and not the very nice special effects they show after the segmentation is obtained). Our approach has a number of advantages over this work: *(a)* We need significantly less user interaction. In [5] the user basically needs to draw the boundaries for all key frames by hand (about every 10 frames for this video), while our method only requires very few rough scribbles, see Figure 25. *(b)* We explicitly compute the alpha matte, while [5] gives spline approximations of the detected boundaries (explicit computation of the matte was not in the original goals of [5] for their applications). *(c)* Our method can adapt to a wide variety of motions while the algorithm in [5] easily loses track of the object, especially when part of the object moves out of the frame, requiring further user intervention. To better illustrate the comparison, we generate the boundaries by thresholding and dilating the alpha matte obtained by our method. A few frames are shown in Figure 30.

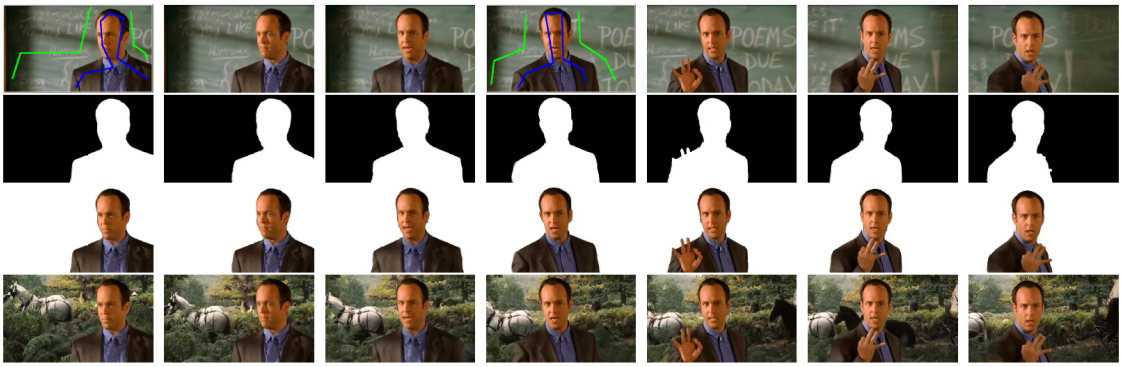


Figure 25: *Video example 1 (a total of 71 frames).*

Figure 26: *Video example 2 (a total of 79 frames).*

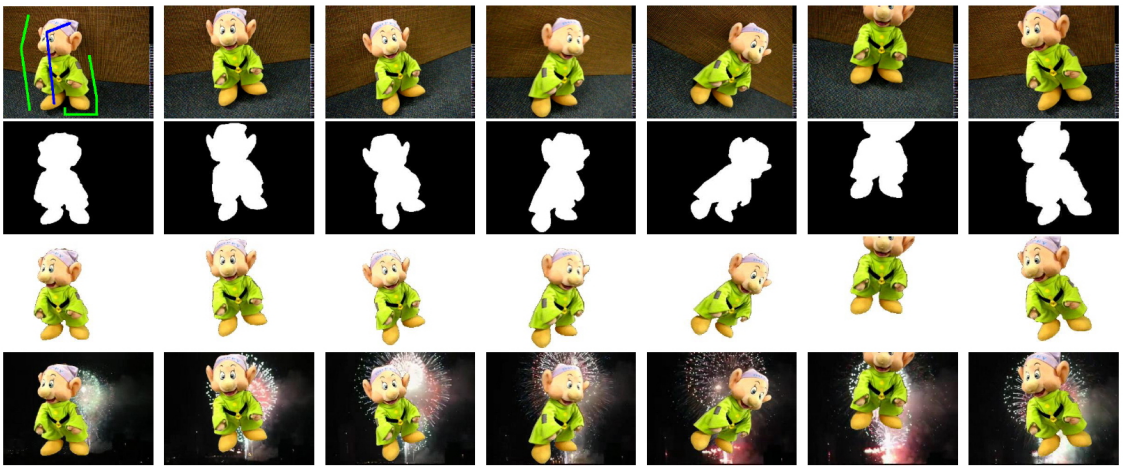


Figure 27: *Video example 3 (a total of 78 frames).*

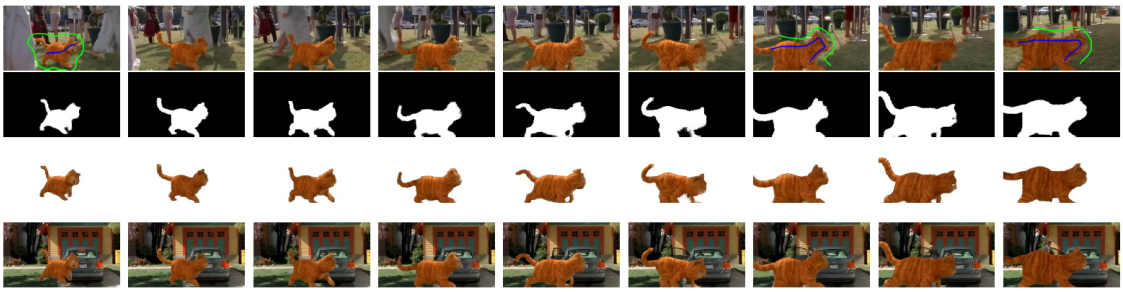


Figure 28: *Video example 4 (a total of 44 frames).*

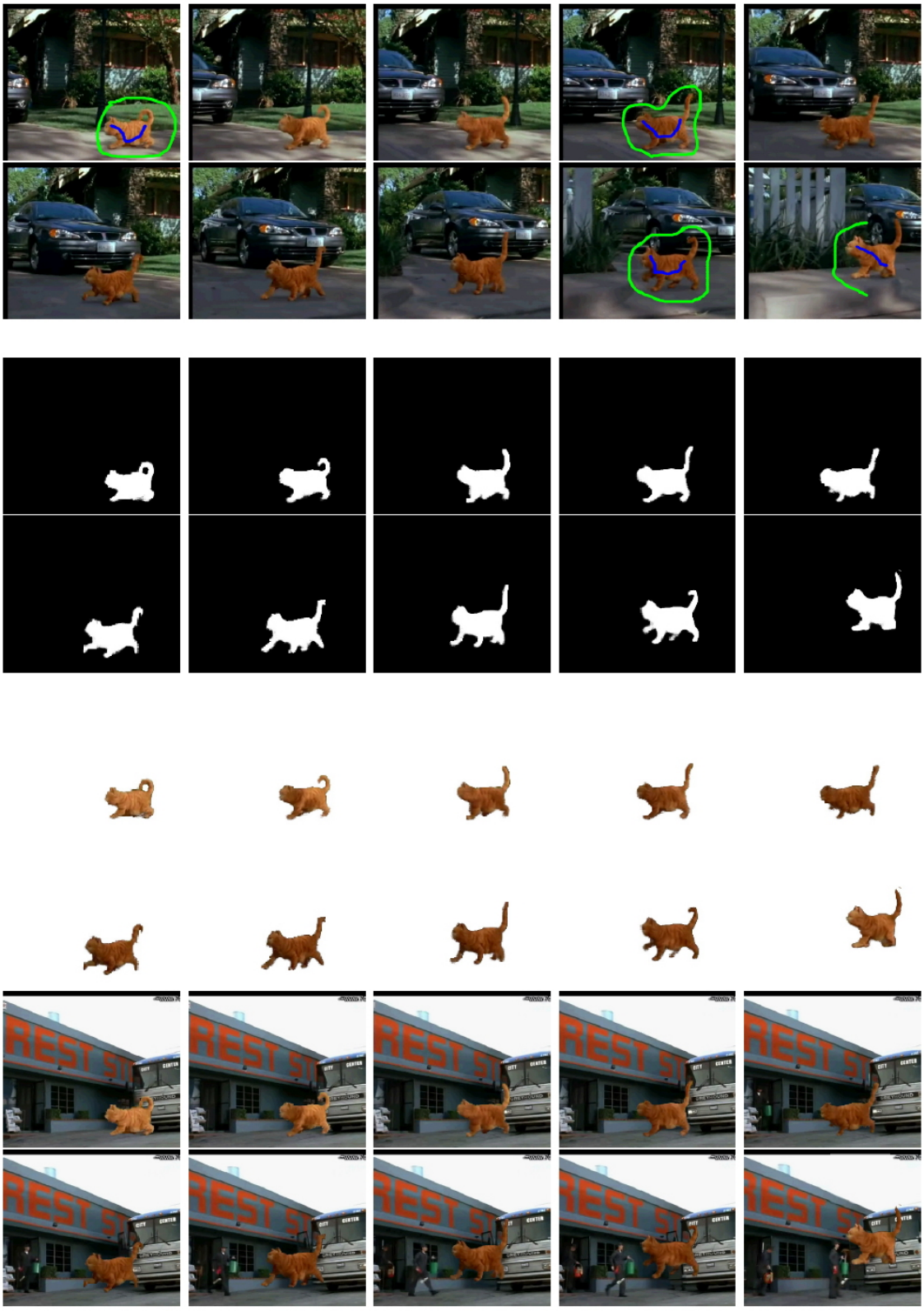


Figure 29: *Video example 5 (a total of 94 frames).*

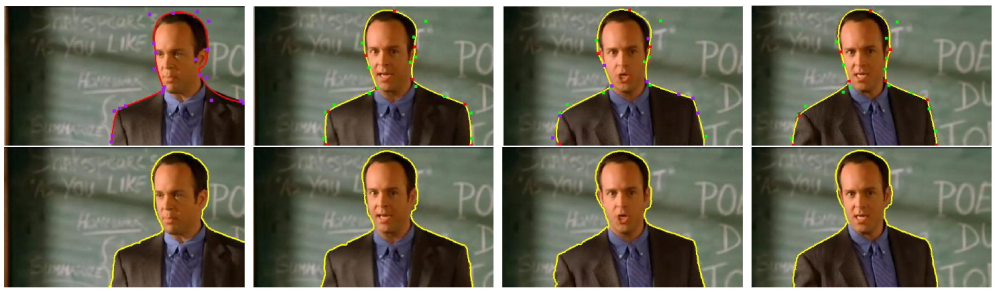


Figure 30: *Comparison with the rotoscoping algorithm in [5]. The curves indicate the boundaries. Top row: A few frames for the work in [5], obtained by their provided interface. The small squares are the control points of the splines. Bottom row: Results from our approach, obtaining similar segmentation with significantly less user intervention (see Figure 25).*

5 Three dimensional medical image segmentation

To conclude the extensions of the proposed framework, and the detailed experimentation with it, we demonstrate our framework in medical image application. Following the same idea as in video segmentation, we apply the algorithm in 3D CT colon images, obtained from [1]. From a pair of scribbles in one frame, the algorithm manages to successfully separates the homogeneous tissue of interest, Figure 31. The constraints on the third dimension we specified for the video data are not required in this case, since the 3D connectivity actually means that the tissue is connected.

6 Conclusions and future work

We presented a geodesics-based algorithm for (interactive) natural image, 3D, and video segmentation and matting. We introduced the framework for still images and extended it to video segmentation and matting, as well as to 3D medical data. We added constraints to the distance function in order to handle objects that cross each other in the video temporal domain. We showed examples illustrating the application of this framework to very different images and videos, including videos with dynamic background and moving cameras. Another application of our approach is to speed up available image matting algorithms (e.g., [30]). A variable width band is quickly generated from a few scribbles, and then a different matting algorithm is applied inside this band. Similarly, if segmentation is the main goal, the very fast results of the first step can be used as input for geometric active contours, [22], which will then need to run for very few iterations for local adjustments only, thereby leading to a very fast and accurate segmentation algorithm.

Although the proposed framework is general, we mainly exploited weights in the geodesic computation that depend on the pixel value distributions. As such, in this form the algorithm works best when these distributions do not significantly overlap. In principle, this can be solved with enough user interactions, but could be tedious, and would be better to solve this by enhancing the features used in deriving the weights. Our current efforts are concentrated on enhancing the features we currently use for weighting the geodesic. Related to this, it is important to being able to automatically select the best features from a bank of possibilities. We are also investigating how to naturally add a regularization term into the model, without having to perform this as a post-processing step as currently done. Results in these directions will be reported elsewhere.

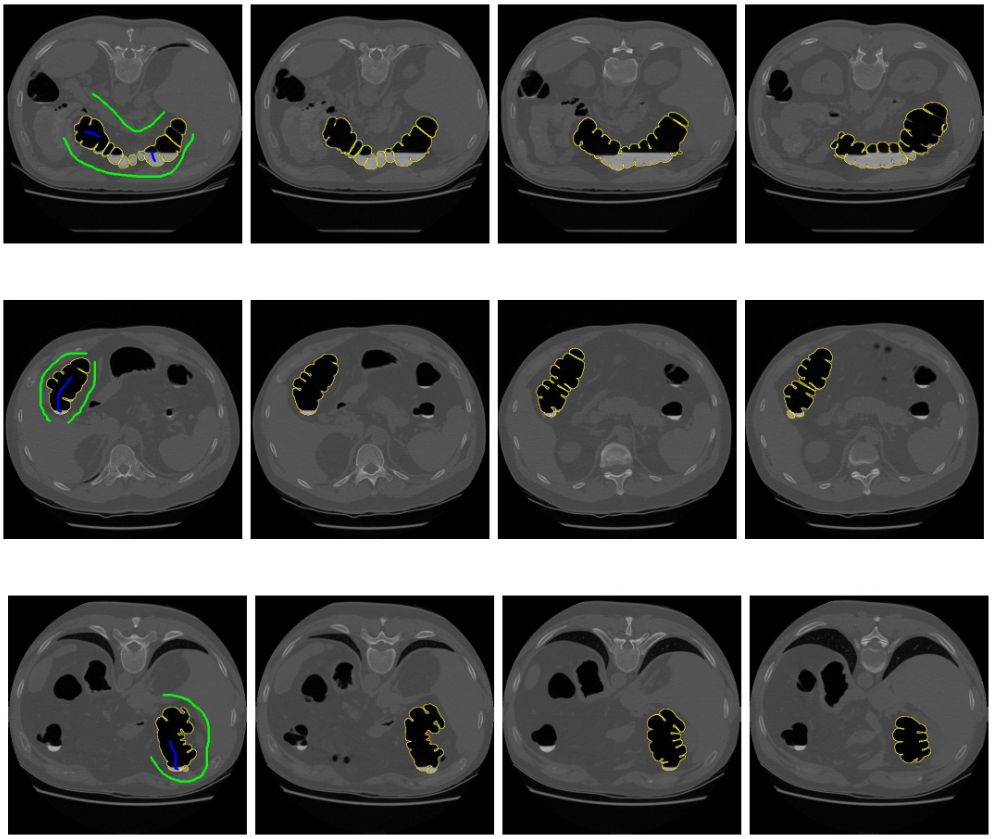


Figure 31: *3D colon segmentation. The rows show slices from three different data sets. First column: scribbles are placed in the region of interest. Second to fourth columns: segmentation results.*

Acknowledgments

We thank David Simons, Scott Cohen, Alex Rav-Acha, Alexandre X. Falcao, Jue Wang (who also motivated the nickname of the algorithm), Aseem Agarwala, and the members of our research group for very important feedback on the work here reported. This work is partially supported by ONR, NGA, NSF, DARPA, NIH, and ARO.

References

[1] *Virtual Colonoscopy Screening Resource Center*. http://www.vcscreen.com.

[2] *Adobe Photoshop User Guide*. ADOBE SYSTEMS INCORP, 2002.

[3] *Knockout User Guide*. COREL CORPORRATION, 2002.

[4] *Adobe Photoshop CS3 New Features*. http://www.adobe.com/products/photoshop/photoshop, 2007.

[5] A. Agarwala, A. Hertzmann, D. Salesin, and S. Seitz. Keyframe-based tracking for rotoscoping and animation. *Proceedings of SIGGRAPH'04*, 2004.

[6] X. Bai and G. Sapiro. A geodesic framework for fast interactive image and video segmentation and matting. In *Proc. International Conference Computer Vision*, October 16-19, Rio de Janeiro, Brazil, 2007.

[7] M. J. Black and P. Anandan. The robust estimation of multiple motions: parametric and piecewise-smooth flow fields. *Comput. Vis. Image Underst.*, 63(1):75–104, 1996.

[8] Y. Y. Boykov and M.-P. Jolly. Interactive graph cuts for optimal boundary and region segmentation of objects in n-d images. *IEEE ICCV 2001*, 01:105, 2001.

[9] V. Caselles, J.-M. Morel, and C. Sbert. An axiomatic approach to image interpolation. *IEEE Trans. Image Processing*, 7:3:376–386, 1998.

[10] Y.-Y. Chuang, A. Agarwala, B. Curless, D. H. Salesin, and R. Szeliski. Video matting of complex scenes. In *SIGGRAPH '02*, pages 243–248, 2002.

[11] Y.-Y. Chuang, B. Curless, D. H. Salesin, and R. Szeliski. A bayesian approach to digital matting. In *Proceedings of IEEE CVPR 2001*, volume 2, pages 264–271, December 2001.

[12] A. Criminisi, G. Cross, A. Blake, and V. Kolmogorov. Bilayer segmentation of live video. In *Proceedings of IEEE CVPR 2006*, pages 53–60, 2006.

[13] R. Dial. Shortest path forest with topological ordering. *Communications of the ACM*, 12:632–633, 1969.

[14] A. X. Falcao, J. Stolfi, and R. de Alencar Lotufo. The image foresting transform: Theory, algorithms, and applications. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(1):19–29, 2004.

[15] L. Grady. Random walks for image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(11):1768–1783, 2006.

[16] A. Levin, D. Lischinski, and Y. Weiss. Colorization using optimization. *SIGGRAPH'04*, 23(3):689–694, 2004.

[17] A. Levin, D. Lischinski, and Y. Weiss. A closed form solution to natural image matting. In *Proceedings of IEEE CVPR 2006*, pages 61–68, 2006.

[18] A. Levin, A. Rav-Acha, and D. Lischinski. Spectral matting. In *Proceedings of IEEE CVPR 2007*, June 2007.

[19] D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 2:91–110, 2004.

[20] A. Protiere and G. Sapiro. Interactive image segmentation via adaptive weighted distances. *IEEE Trans. Image Processing*, 16:1046–1057, 2007.

[21] C. Rother, V. Kolmogorov, and A. Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. *SIGGRAPH'04*, 2004.

[22] G. Sapiro. *Geometric Partial Differential Equations and Image Processing*. Cambridge University Press, January 2001.

[23] A. K. Sinop and L. Grady. A seeded image segmentation framework unifying graph cuts and random walker which yields a new algorithm. In *Proc. International Conference Computer Vision*, October 16-19, Rio de Janeiro, Brazil, 2007.

[24] J. Sun, J. Jia, C.-K. Tang, and H.-Y. Shum. Poisson matting. In *SIGGRAPH'04*, pages 315–321, 2004.

[25] J. Sun, S. Kang, Z. Xu, X. Tang, and H.-Y. Shum. Flash cut: Foreground extraction with flash and no-flash image pairs. In *Proceedings of IEEE CVPR 2007*, June 2007.

[26] M. Thorup. Undirected single source shortest paths in linear time. In *Proceedings IEEE Symposium on Foundations of Computer Science*, pages 12–21, 1997.

[27] J. Wang, M. Agrawala, and M. F. Cohen. Soft scissors: An interactive tool for realtime high quality matting. In *SIGGRAPH'07*, 2007.

[28] J. Wang, P. Bhat, R. A. Colburn, M. Agrawala, and M. F. Cohen. Interactive video cutout. *SIGGRAPH'05*, 24(3):585–594, 2005.

[29] J. Wang and M. Cohen. Optimized color sampling for robust matting. In *Proceedings of IEEE CVPR 2007*, June 2007.

[30] J. Wang and M. F. Cohen. An iterative optimization approach for unified image segmentation and matting. In *Proceedings of IEEE ICCV 2005*, pages 936–943, 2005.

[31] C. Yang, R. Duraiswami, N. Gumerov, and L. Davis. Improved fast Gauss transform and efficient kernel density estimation. In *Proceedings IEEE ICCV 2003, Nice, France*, pages 464–471, 2003.

[32] L. Yatziv, A. Bartesaghi, and G. Sapiro. O(n) implementation of the fast marching algorithm. *Journal of Computational Physics*, 212:393–399, 2006.

[33] L. Yatziv and G. Sapiro. Image and video data blending using intrinsic distances. *Patent pending*, 2005.

[34] L. Yatziv and G. Sapiro. Fast image and video colorization using chrominance blending. *IEEE Trans. on Image Processing*, 15:5:1120–1129, 2006.