Extracting Fluid from a Video for Efficient Post-Production

Makoto Okabe* Ken Anjyo† Rikio Onai‡
*,‡The University of Electro-Communications †OLM Digital, Inc./JST CREST *JST PRESTO

Figure 1: *Top row: the input video sequence of an explosion. Bottom row: the composite of the extracted explosion with another background image. Our method automatically extracted the explosion, which has considerable transparency, especially in the smoky regions of the 4th and 5th frames.*

Abstract

We propose a method to extract fluids from a video that is captured outside a special studio. Since such a video usually has a complex background and the fluids overlap with much transparency, it is a difficult, time-consuming task for a designer to extract them. Our goal is to develop an efficient method to solve the problem: we estimate the background of an input video, and then compute the foreground and alpha matte at each frame. Our method estimates the background by observing only pixels that have little motion at each frame. Given the estimated background, we estimate an initial alpha matte based on the color difference at every pixel between each frame and the estimated background. Since the initial alpha matte usually includes many artifacts, we employ the gradient-domain image processing approach to refine it: our method attenuates unrequired gradients adequately, and then integrate them to recover the refined alpha matte. The foreground, which explains about the color and texture pattern of the fluid itself, is also estimated in a similar manner. We demonstrate that our method enables to extract the fluids from a video, which were difficult to achieve using the previous methods.

CR Categories: I.4.8 [Image Processing and Computer Vision]: Scene Analysis—Motion—;

Keywords: video processing, alpha matte, post-production, fluid

1 Introduction

In computer graphics production, a video database of fluids is an important resource for a designer to create scenes with fluids related with smoke, fire, or an explosion. Although physics simulations of fluids have become powerful tools for such purposes, it is often difficult for a non-expert designer to understand and set all the physical parameters appropriately. Hence, constructing a composite with fluid videos is still a popular and reasonable way to create a high-quality fluid scene.

Most of the fluid videos in the current database are produced in a special studio, e.g., the explosion of a bomb, captured in front of a single-colored screen [Smith and Blinn 1996]. Such studios enable us to capture fluids with clean alpha mattes, which is important for the efficient creation of a composite. Nevertheless, we are also interested in fluid videos captured outside the studio, since they usually include larger-scale, more dynamic fluids, captured in a real environment. Also, such video materials have recently become more readily available on the web (the top row of Fig. 1 shows an example). However, since such fluid videos are usually captured in front of a complex background, rather than a single-colored screen, it is time-consuming to manually separate the fluid from the background. Our goal is to develop a system to support this process, allowing the designer to efficiently use the extracted fluids for post-production, as shown in the bottom row of Fig. 1.

Extraction of transparent fluids from a video is still a challenging problem. As far as we know, none of the previous methods (including those of the survey [Wang and Cohen 2007]) can solve the problem perfectly. One approach solves the problem efficiently by introducing a strong assumption to the fluid. For example, the Bayesian Matting method assumes that the fluid has only a single color [Chuang et al. 2002], which works well for faint smoke. However, when it is applied to a colorful explosion, the extraction result has only a flat color (Fig. 2-d). Ghanem et al. assume a texture rather than a single color [Ghanem and Ahuja 2008]. However, since the texture cannot change temporally, it is also inadequate for representing an explosion. There is a method for dehazing a video [Zhang et al. 2011], but it is likewise limited to fog of a single color. A related work involves video matting combined with transparency caused by a motion blur [Lin et al. 2011], but this technique has not been applied to the transparency of fluids.

Another approach is to apply the image matting method to each frame of the video [Sun et al. 2004; Levin et al. 2008; Gastal and Oliveira 2010]. In this approach, we create a *trimap* or *scribbles* for each frame, which partitions the image into three regions: 'defi-

*m.o@acm.org
†anjyo@olm.co.jp
‡onai@cs.uec.ac.jp

(a) Original (b) Background (c) Trimap (d) Bayesian Matting (e) Spectral Matting (f) Shared Matting (g) Our method

Figure 2: *We compare our method with previous methods. The trimap (c) is computed based on the color difference between the original (a) and the background (b) at each pixel. We use the trimap to compute the foreground color for Bayesian Matting (d). For Spectral Matting (e) and Shared Matting (f), we use the trimap as direct input to the algorithm.*

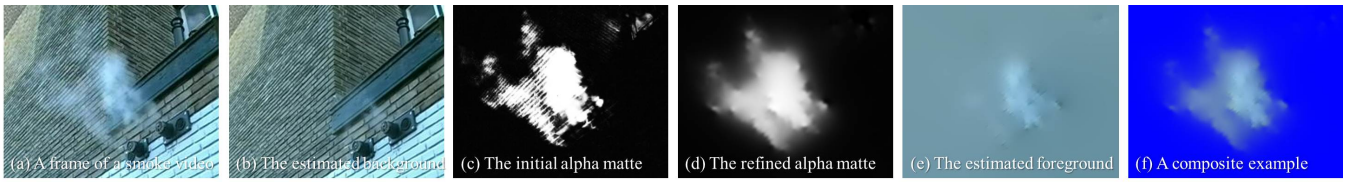


Figure 3: *An overview of our method.*

nitely foreground,' 'definitely background,' and 'unknown.' This type of procedure is effective for estimating the alpha matte on small transparent regions around an opaque object. However, since a fluid has transparency over a large area, it is difficult to apply such a method directly to fluid videos. Fig. 2-e shows the result of Spectral Matting [Levin et al. 2008]. This procedure extracts not only the fluids, but also the background image over a large area. Its computational cost is an additional disadvantage; it took 214 seconds to process a single frame. Fig. 2-f shows the result of Shared Matting [Gastal and Oliveira 2010]. This method processes each video frame in real time, successfully removes the background, and extracts the fire regions. However, there are many artifacts, appearing as blobs in the faint smoke regions.

To address these issues, we develop a novel method for extracting fluids from videos with reasonable computational expense. In the example of Fig. 1, our method automatically extracted the explosion together with the faint smoke (Fig. 2-g). Fig. 3 shows an overview of our approach. Given a fluid video captured with a fixed, monocular camera, we first estimate its background, and then compute the foreground and alpha matte for each frame. We estimate the background by averaging throughout the entire sequence only pixels that have little motion (Fig. 3-b). Most of the smoke disappears, lingering at some pixels where it is particularly dense; i.e., it always has significant motion.

More specifically, given the estimated background, our key idea is that the user creates an initial, rough alpha matte by manipulating several parameters, and then our system refines it. Because it is difficult for a computer to automatically estimate the alpha matte, the user provides a rough alpha matte. On the other hand, because it is difficult for the user to remove the artifacts from the rough alpha matte, our system refines it.

We compute an initial alpha matte by calculating color differences (Fig. 3-c). It roughly represents the area of the smoke, but the texture patterns of blocks of background can be seen in it. To remove these texture patterns, we employ gradient-domain image processing to refine the alpha matte (Fig. 3-d) and estimate the foreground (Fig. 3-e), which can be used to create a composite (Fig. 3-f). We compute horizontal and vertical image gradients for each frame, attenuate the gradients independently of the smoke, and then integrate them to obtain the refined alpha matte. The technique of attenuation is our major technical contribution. The computational time is several seconds per frame for 640×360 resolution.

2 Our method

We model the ith frame of a video sequence as a standard alpha blending:

$$I_i(\vec{x}) = F_i(\vec{x})\alpha_i(\vec{x}) + B(\vec{x})(1 - \alpha_i(\vec{x})), \tag{1}$$

where $\vec{x}$ denotes a pixel position, and I_i, F_i, B and α_i denote the original frame, its foreground, background, and alpha matte. Note that, because we assume that the input video is captured with a monocular camera, B is single through the entire sequence. In all of the following discussion, the range of a pixel value is from 0 to 1 (not 255). Given an input video, our algorithm begins by estimating the background.

2.1 Background Estimation

We estimate the background by observing only pixels that have insignificant motion. More sophisticated approaches to background estimation also exist (e.g., [Apostoloff and Fitzgibbon 2004]), but our approach is also useful for videos like the ones used in this paper and our supplementary video.

We compute the optical flow using the *OFLib* package [Zach et al. 2007], which is implemented via the graphics processing unit (GPU). It is computationally efficient, requiring about 0.1 second to compute the optical flow between temporally neighboring frames at 640×480 resolution. Then we compute the color of the background $B(x)$ as the following weighted average:

$$B(\vec{x}) = \frac{\sum_i I_i(\vec{x}) W_i(\vec{x})}{\sum_i W_i(\vec{x})}, \tag{2}$$

$$W_i(\vec{x}) = e^{-\beta |V_i(\vec{x})|}, \tag{3}$$

where V_i is the optical flow computed between the ith and $i+1$th frames, $V_i(\vec{x})$ is the velocity at the pixel position $\vec{x}$, W_i is the weight, and β is set equal to 40 throughout our experiments. Fig. 4 shows the background estimation process for a smoke video. We show the original frames I_i in the top row, the magnitude of the corresponding optical flow V_i in the middle row, and the corresponding weights W_i in the bottom row. Intuitively speaking, we

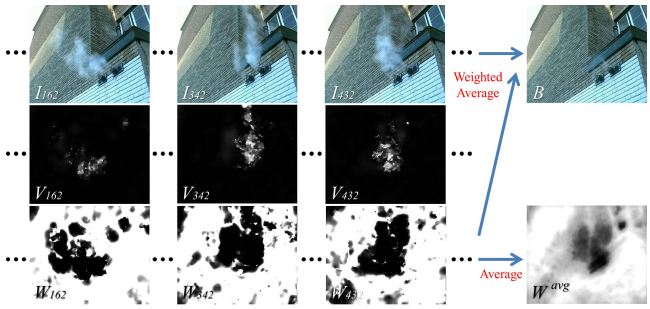


Figure 4: *Background estimation.*

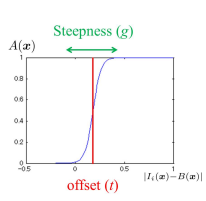
assume that at a pixel with low magnitude $|V_i(x)|$, $W_i(x)$ assures us a high probability of obtaining the correct background color. The rightmost part of Fig. 4 shows the estimated background B and the average of W_i; i.e., $W^{avg} = \sum_i^M W_i/M$, where M is the number of frames. We can see some smoke remaining in B, and W^{avg} is dark around this area; i.e., it has low values.

2.2 Estimation of the Initial Alpha Matte

Given the estimated background B, we estimate the initial alpha matte A_i^{init} by calculating the color difference between each frame I_i and B at every pixel. We assume that a three-dimensional (3D) vector of red, green, and blue channels is assigned to each pixel of I_i and B, and a single scalar value for the luminance channel is assigned to each pixel of A_i^{init}. We compute the initial alpha matte of I_i as follows:

$$A_i^{init}(\vec{x}) = \frac{1}{1 + e^{-g(|I_i(\vec{x}) - B(\vec{x})| - t)}}, \tag{4}$$

which is a sigmoid function of color difference. g and t control the steepness and the offset of the function. Fig. 5-a shows the sigmoid function obtained by setting g and t equal to $10^{1.4}$ and 0.18, respectively. Before calculating the color difference, we apply a Gaussian blur to remove the noise from I_i and B. The size of the Gaussian kernel is set at 5.0 throughout our experiments. Figs. 5-b, c, and d show the resulting initial alpha matte. Note that there are many artifacts; e.g., the texture patterns of the blocks of the background wall can be seen in the figure.



(a) The sigmoid function

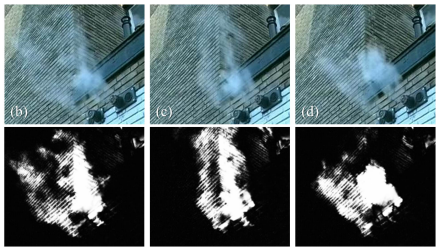


Figure 5: *The sigmoid function and the initial alpha matte.*

We design Eq. 4 as a heuristic with two requirements: user controllability and nonlinear smoothness. Most of the existing image-matting methods use a trimap or scribbles, designed via a user heuristic, as input [Sun et al. 2004; Levin et al. 2008; Gastal and Oliveira 2010]. Eq. 4 reflects this; it allows the user to design a rough hint for an initial alpha matte, similar to the way in which a trimap or scribbles are designed by user intuition, independently of any equation.

2.3 Refinement of the Alpha Matte

The initial alpha matte constructed by the user gives the overall region of the fluid. However, artifacts from the texture patterns of the background remain, and we want to remove them.

Our idea for refining the initial alpha matte is based on gradient-domain image processing. Fig. 6 shows an overview of the procedure. We begin by computing the horizontal and vertical image gradients (Figs. 6-b and c) of the initial alpha matte (Fig. 6-a). Then we process these so that only gradients related to the smoke are preserved, and the other gradients are attenuated (Figs. 6-d and e). Finally, we integrate the processed image gradients to create the refined alpha matte (Fig. 6-f). To obtain the image gradients,

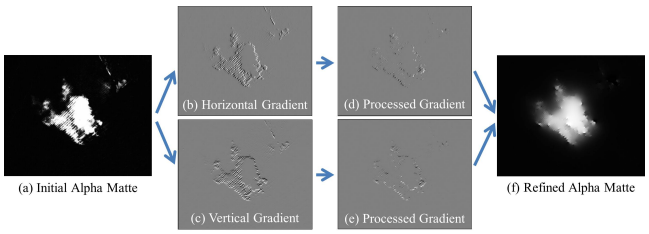


Figure 6: *An overview of the refinement of the alpha matte.*

we simply compute the horizontal and vertical differences between neighboring pixels. We formulate the integration as a constrained least-squares optimization:

$$\underset{a_1,\ldots,a_N}{\operatorname{argmin}} E = \sum_p \sum_{q \in \nu(p)} (a_p - a_q - g_{pq})^2 + \sum_p \lambda_p (a_p - a_p^{init})^2, \tag{5}$$

where $\{a_1, ..., a_N\}$ are all of the pixel values of the refined alpha matte, p denotes a pixel position, $\nu(p)$ denotes the set of pixel positions of the horizontal and vertical neighbors of p, g_{pq} is the processed gradient between p and q, corresponding to Figs. 6-d and e, λ_p is the weight of the constraint at p, and a_p^{init} is the pixel value of the initial alpha matte A_i^{init} at p. To minimize the energy function, we use the Poisson solver with the effective preconditioner [Szeliski 2006], which is implemented via the GPU. We set the maximum number of iterations of the conjugate gradient method at 80. To obtain the refined alpha matte, at every pixel we must adequately specify 1) the gradient g_{pq}, and 2) the weight of the constraint λ_p. We explain how to specify each of these.

To compute g_{pq} as shown in Figs. 6-d and e, we preserve the gradients found only in A_i^{init}, and attenuate the gradients shared by the frame A_i^{init} and the estimated background B. Our idea is similar to the "contrast attenuation" of Background Cut [Sun et al. 2006]. Background Cut attenuates a gradient in A_i^{init} when the magnitude of the gradient at the same pixel position in B is large. It successfully extracts only gradients found uniquely in A_i^{init}. However, attenuation based solely on the magnitude of the gradient is not sufficient to satisfy our goal. We process gradients more carefully; we attenuate the gradient at a pixel position around which a similar texture pattern is shared by A_i^{init} and B.

To describe the features of a texture pattern, we consider a patch of 5×5 resolution around a pixel position (Fig. 7-a). In this patch, we compute the oriented gradient at every pixel, and create a histogram of these oriented gradients. We include eight orientations, but the dimension of the histogram is only four, since an orientation and its reverse share a single bin, as shown in Fig. 7-a. Each bin contains the total of the magnitudes of all gradients whose orientations

correspond to the orientation the bin represents. The histogram is finally normalized so that the total of all bins is equal to one.

At every pixel position, we compute the Euclidean distance between the histograms of A_i^{init} and B. Fig. 7-d shows the result. The intensity at each pixel position describes the difference between the texture patterns of A_i^{init} (Fig. 7-b) and B (Fig. 7-c). A gradient g_{pq}^{init} in A_i^{init} is attenuated as follows:

$$g_{pq} = g_{pq}^{init}(1 - e^{-d_p^2/K^2}), \tag{6}$$

where d_p is the Euclidean distance (Fig. 7-d) at p, and K is set equal to 0.01 throughout our experiments. The attenuated image gradients g_{pq} are shown in Figs. 6-d and e. The texture patterns of the blocks of the background wall are removed from the image gradients, and also from the resulting refined alpha matte.

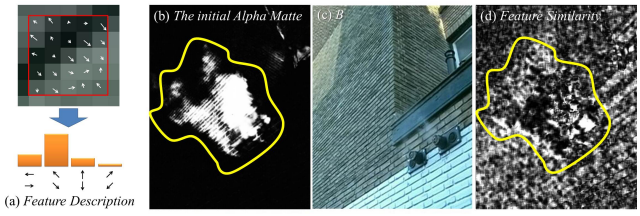


Figure 7: *(a) We compute oriented gradients in a 5×5 patch and create a four-dimensional histogram of these gradients to describe the features of a texture pattern. (b) The initial alpha matte A_i^{init}. (c) The background B. (d) Feature similarity as the Euclidean distance between A_i^{init} and B. Note that the yellow boundary is overlaid to show the corresponding region. The area of darker values contains strong similarities, where the gradients will be attenuated.*

Fig. 8 compares our attenuation method to that of Background Cut [Sun et al. 2006]. Background Cut attenuates a gradient simply when the corresponding background gradient is large. Because the brick wall is highly textured (i.e., has many large gradients), it attenuates the gradients in the smoke region. As a result, the extracted smoke becomes smooth and flat (Fig. 8-a). To solve this problem, we introduce texture pattern similarity. The gradients are attenuated only when the texture patterns match, which preserves the original smoke texture more faithfully (Fig. 8-b).

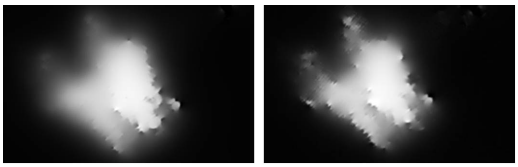


Figure 8: *Comparison of the attenuation methods.*

We specify a pixel as a constraint when the pixel value of the initial alpha matte is extremely large or small. When a_p^{init} is greater than 0.8 or less than 0.02, we set λ_p equal to 0.01. Otherwise, we set λ_p equal to zero.

2.4 Foreground Estimation

We estimate the foreground by almost the same method used for the refinement of the alpha matte. We process the red, green, and blue channels independently, solving the constrained least-squares optimization separately for each color channel. We compute the horizontal and vertical image gradients of I_i, attenuate them in the same manner as Eq. 6, and integrate them to estimate the foreground. In Eq. 5, we let a_i be the intensity of the foreground, and a_p^{init} be the intensity of I_i. For the attenuation of the image gradient, we use Eq. 6, letting g_{pq}^{init} be the gradient in I_i, and d_p be the Euclidean distance between the texture patterns I_i and B. As for constraints, we specify a pixel as a constraint only when the pixel value of the initial alpha matte is extremely large. When a_p^{init} is greater than 0.8, we set λ_p equal to 0.01, but this time we ignore pixels whose values are extremely small. Fig. 9 shows the resulting foreground.

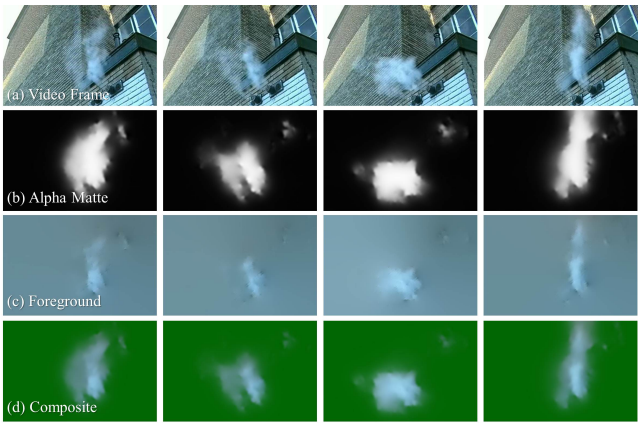


Figure 9: *(a) The original video frame, (b) the refined alpha matte, (c) the estimated foreground, and (d) the composite with a green background.*

3 Results and Discussion

We apply our method to various fluid videos containing smoke and explosions. The videos are selected from the DynTex database [Péteri et al.] and the internet. Each video shows a fluid flowing in front of a static background. The fluids have both sparse and dense parts, and the backgrounds include complex texture patterns. Information on each result is given in Table 1. All results, and comparisons with the extractions performed by the difference matte function of Adobe After Effects, are also given for our supplementary video. All computations are carried out using a notebook PC with an Intel(R) Core(TM) i7 1.87 GHz CPU and an NVIDIA Quadro FX 3800M GPU.

Employing our system, the user begins the process of fluid extraction by loading a video into the system. The system spends several minutes estimating the background, most of it on the computation of the optical flow. Then the user creates the initial alpha matte by manipulating the slider bars of t and g. This process is fully interactive; i.e., the user can check the alpha matte in real time. When a satisfactory initial alpha matte is obtained, the user pushes the button and the system proceeds to apply gradient-domain image-processing to every frame. Finally, when the system finishes processing the last frame, a video sequence of the extracted fluid is obtained. We found that $t = 0.18$ and $g = 10^{1.4}$ are good default settings for the parameters.

Figs. 9 and 10 show some successful results. In Fig. 9, because the smoke is relatively sparse, the background is successfully estimated (Fig. 3-b). A small amount of smoke remains, but it does not create a major problem for the smoke extraction. In Fig. 10, we compare the results obtained from the difference matte of Adobe After Effects and our method. The red arrows indicate the area where our method successfully removes the texture pattern derived from the background: the edges and tile patterns of the background.

(a) Original (b) Estimated Background (c) Difference Matte (d) Our method

Figure 10: *The extracted fluids from Explosion2 and Explosion3.*

Fluid Video	Frames	Resolution	Time
Smoke1	883	352×288	2.92
Explosion1	209	480×360	6.31
Explosion2	947	640×360	5.0
Explosion3	609	640×360	6.0

Table 1: *For each fluid video, we list the number of the frames, the resolution, and the average time (seconds) per frame required to compute the extraction.*

3.1 Limitations and Future Work

Because our method attenuates image gradients, the results tend to be blurry, because of both the blurred foreground and the alpha matte. Fig. 1 illustrates the phenomenon; the contrast and texture patterns of the exploding fire are reduced in the result. This suggests that we must find a more delicate technique for selecting the gradients to be attenuated. This will be the subject of our future work.

Our method relies on the quality of the initial alpha matte, which is based on color differences. As a result, when a pixel in a fluid has a color very similar to that of the background, it is often estimated as a background pixel. Fig. 11 shows an example of smoke flowing slowly across the sky. The background (Fig. 11-b) is not successfully estimated because of the extremely dense smoke. In addition, the similarity in color between the smoke and the background is high, creating an undesirable hole in the initial alpha matte (Fig. 11-c). Our method attempts to remove the hole by attenuating the gradients around the hole. The boundary of the hole becomes blurry, but remains in the refined alpha matte (Fig. 11-d). To solve this problem, we plan to estimate the initial alpha matte by relying not only on color differences, but also on other information, such as fluid motion.

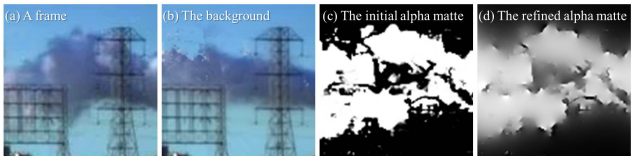


Figure 11: *Failure caused by small color differences.*

References

APOSTOLOFF, N. E., AND FITZGIBBON, A. W. 2004. Bayesian video matting using learnt image priors. In *Proc. of CVPR '04*.

CHUANG, Y.-Y., AGARWALA, A., CURLESS, B., SALESIN, D. H., AND SZELISKI, R. 2002. Video matting of complex scenes. 243–248.

GASTAL, E. S. L., AND OLIVEIRA, M. M. 2010. Shared sampling for real-time alpha matting. *Comput. Graph. Forum 29*, 2, 575–584.

GHANEM, B., AND AHUJA, N. 2008. Extracting a fluid dynamic texture and the background from video. In *Proc. of CVPR '08*, 1 –8.

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

LIN, H. T., TAI, Y.-W., AND BROWN, M. S. 2011. Motion regularization for matting motion blurred objects. *IEEE Trans. Pattern Anal. Mach. Intell. 33*, 11, 2329–2336.

PÉTERI, R., FAZEKAS, S., AND HUISKES, M. J. DynTex : a Comprehensive Database of Dynamic Textures. *Pattern Recognition Letters*. http://projects.cwi.nl/dyntex/.

SMITH, A. R., AND BLINN, J. F. 1996. Blue screen matting. In *Proc. SIGGRAPH '96*, 259–268.

SUN, J., JIA, J., TANG, C.-K., AND SHUM, H.-Y. 2004. Poisson matting. *ACM Trans. Gr. 23*, 3, 315–321.

SUN, J., ZHANG, W., TANG, X., AND SHUM, H. Y. 2006. Background cut. In *Europ. Conf. on Computer Vision*, 628–641.

SZELISKI, R. 2006. Locally adapted hierarchical basis preconditioning. In *SIGGRAPH2006*, 1135–1143.

WANG, J., AND COHEN, M. F. 2007. Image and video matting: a survey. *Found. Trends. Comput. Graph. Vis. 3*, 2, 97–175.

ZACH, C., POCK, T., AND BISCHOF, H. 2007. A duality based approach for realtime tv-l1 optical flow. In *Pattern Recognition (Proc. DAGM)*, 214–223.

ZHANG, J., LI, L., ZHANG, Y., YANG, G., CAO, X., AND SUN, J. 2011. Video dehazing with spatial and temporal coherence. *Vis. Comput. 27*, 749–757.