Automated Removal of Partial Occlusion Blur

Scott McCloskey, Michael Langer, and Kaleem Siddiqi

Centre for Intelligent Machines, McGill University
{scott,langer,siddiqi}@cim.mcgill.ca

Abstract. This paper presents a novel, automated method to remove partial occlusion from a single image. In particular, we are concerned with occlusions resulting from objects that fall on or near the lens during exposure. For each such foreground object, we segment the completely occluded region using a geometric flow. We then look outward from the region of complete occlusion at the segmentation boundary to estimate the width of the partially occluded region. Once the area of complete occlusion and width of the partially occluded region are known, the contribution of the foreground object can be removed. We present experimental results which demonstrate the ability of this method to remove partial occlusion with minimal user interaction. The result is an image with improved visibility in partially occluded regions, which may convey important information or simply improve the image's aesthetics.

1 Introduction

Partial occlusions arise in natural images when an occluding object falls nearer to the lens than the plane of focus. The occluding object will be blurred in proportion to its distance from the plane of focus, and contributes to the exposure of pixels that also record background objects. This sort of situation can arise, for example, when taking a photo through a small opening such as a cracked door, fence, or keyhole. If the opening is smaller than the lens aperture, some part of the door/fence will fall within the field of view, partially occluding the background. This may also arise when a nearby object (such as the photographer's finger, or a camera strap) accidentally falls within the lens' field of view.

Whatever its cause, the width of the partially-occluded region depends on the scene geometry and the camera settings. Primarily, the width increases with increasing aperture size (decreasing f-number), making partial occlusion a greater concern in low lighting situations that necessitate a larger aperture.

Fig. 1 (left) shows an image with partial occlusion, which has three distinct regions: complete occlusion (outside the red contour), partial occlusion (between the green and red contours), and no occlusion (inside the green contour). As is the case in this example, the completely occluded region often has little high-frequency structure because of the severe blurring of objects far from the focal plane. In addition, the region of complete occlusion can be severely underexposed when the camera's settings are chosen to properly expose the background.

In [7], it was shown that it is possible to remove the partial occlusion when the location and width of the partially occluded region are found by a user. Because

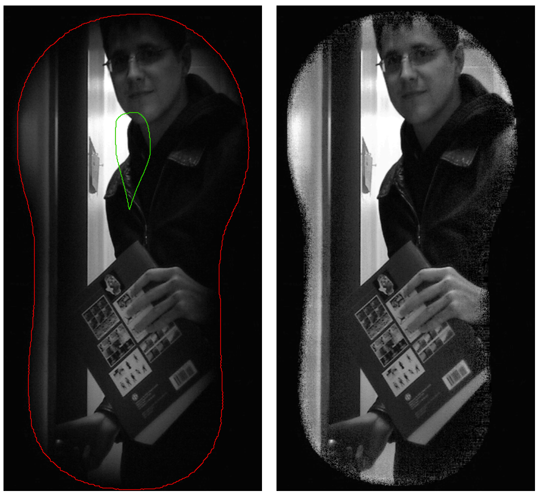

Fig. 1. (Left) Example image taken through a keyhole. Of the pixels that see through the opening, more then 98% are partially occluded. (Right) The output of our method, with improved visibility in the partially-occluded region.

of the low contrast and arbitrary shape of the boundary between regions of complete and partial occlusion, this task can be challenging, time consuming, and prone to user error. In the current paper we present an automated solution to this vision task for severely blurred occluding objects and in doing so significantly extend the applicability of the method in [7]. Given the input image of Fig. 1 (left), the algorithm presented in this paper produces the image shown in Fig. 1 (right). The user must only click on a point within each completely-occluded region in the image, from which we find the boundary of the region of complete occlusion. Next, we find the width of the partially occluded band based on a model of image formation under partial occlusion. We then process the image to remove the partial occlusion, producing an output with improved visibility in that region. Each of these steps is detailed in Sec. 4.

2 Previous Work

The most comparable work to date was presented by Favaro and Soatto [3], who describe an algorithm which reconstructs the geometry and radiance of a scene, including partially-occluded regions. While this restores the background, it requires *several* registered input images taken at different focal positions.

Tamaki and Suzuki [12] presented a method for the detection of completely occluded regions in a single image. Unlike our method, they assume that the occluding region has high contrast with the background, and that there is no adjacent region of partial occlusion.

A more distantly related technique is presented by Levoy *et al.* in [5], where synthetic aperture imaging is used to see around occluding objects. Though this ability is one of the key features of their system, no effort is made to identify or remove partial occlusion in the images.

Partial occlusion also occurs in matte recovery which, while designed to extract the foreground object from a natural scene, can also recover the background in areas of partial occlusion. Unlike our method, matte recovery methods require either additional views of the same scene [8,9] or substantial user intervention [1,11]. In the latter category, users must supply a *trimap*, a segmentation of the image into regions that are either definitely foreground, definitely background, or unknown/mixed. Our method is related to matte recovery, and can be viewed as a way of automatically generating, from a single image, a trimap for images with partial occlusion due to blurred foreground objects.

3 Background and Notation

In [7], it was shown that the well-known matting equation,

$$R_{input}(x,y) = \alpha(x,y)R_f + (1-\alpha(x,y))R_b(x,y), \tag{1}$$

describes how the lens aperture combines the radiance R_b of the background object with the radiance R_f of the foreground object. The blending parameter α describes the proportion in which the two quantities combine and R_{input} is the observed radiance. Notionally, the quantity α is the fraction of the pixel's viewing frustum that is subtended by the foreground object. Since that object is far from the plane of focus, the frustum is a cone and α is the fraction of that cone subtended by the occluding object.

In order to remove the contribution of the occluding object, the values of α and R_f must be found at each pixel. Given the location of the boundary between regions of complete and partial occlusion, the distance d between each pixel and the nearest point on the boundary can be found. From d and the width w of the partially occluded region, the value of α is well-approximated [7] by

$$\alpha = \frac{1}{2} - \frac{l\sqrt{1-l^2} + arcsin(l)}{\pi}, \text{ where } \ l = min\left(2, \frac{2d}{w}\right) - 1. \tag{2}$$

This can be done if the user supplies both w and the boundary between regions complete and partial occlusion, as in [7]. Unfortunately, this task is time consuming, difficult, and prone to user error. In this paper, we present an automated solution to this vision problem, from which we compute the values α and R_f.

4 Method

To state the vision problem more clearly, we refer to the example[1] in Fig. 2. In this example, the partial occlusion is due to the handle of a fork directly in front

[1] The authors of [7] have made this image available to the public at http://www.cim.mcgill.ca/∼scott/research.html

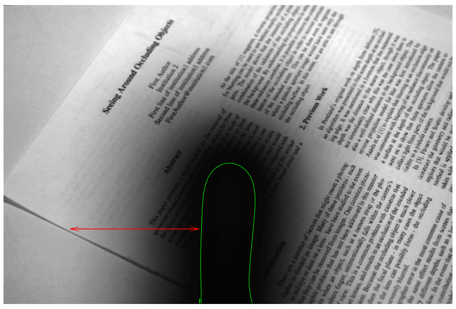

Fig. 2. To remove partial occlusion from a foreground object, the vision problem is to determine the boundary of the completely occluded region (green curve) and the width of the partially-occluded region (the length of the red arrow)

of the lens. In order to remove the contribution of the occluding object, we must automatically find the region of complete occlusion (outlined in green) and the width of the partially occluded band (the length of the red arrow).

In order to find the region of complete occlusion within the image, we assume that the foreground image appears as a region of nearly constant intensity. Note that this does *not* require that the object itself have constant radiance. Because the object is far from the plane of focus, high-frequency radiance variations will be lost due to blurring in the image. Moreover, when objects are placed against the lens they are often severely under-lit, as they fall in the shadow of the camera or photographer. As such, many occluding objects with texture may appear to have constant intensity in the image.

A brief overview of the method is as follows. Given the location of a point p that is completely occluded (provided by the user), we use a geometric flow (Sec. 4.2) to produce a segmentation with a smooth contour such as the one outlined in Fig. 2, along which we find normals facing outward from the region of complete occlusion. The image is then re-sampled (Sec. 4.3) to uniformly-spaced points on these normals, reducing an arbitrarily-shaped occluding contour to a linear contour. Low variation rows in the resulting image are averaged to produce a profile from which the blur width is estimated (Sec. 4.4). Once the blur width is estimated, the method of [7] is used to remove the partial occlusion (Sec. 4.5).

4.1 Preprocessing

Two pre-processing steps are applied before attempting segmentation:

1. Because our model of image formation assumes that the camera's response is linear, we use the method of [2] to undo the effects of its tone mapping function, transforming camera image I_{input} to a radiance image R_{input}.
2. Before beginning the segmentation, we force the completely occluded region to be the darkest part of the image by subtracting R_p, the radiance of the user-selected point, and taking the absolute value. This gives a new image

$$R = \|R_{input} - R_p\|, \tag{3}$$

which is nearly zero at points in the region of complete occlusion, and higher elsewhere. As a result of this step, points on the boundary between the regions of partial and complete occlusion will have gradients ∇R that point out of the region of complete occlusion. This property will be used to find the boundary between the regions of complete and partial occlusion.

4.2 Foreground Segmentation

While the region of complete occlusion is assumed to have nearly constant intensity, segmenting this region is nontrivial due to the extremely low contrast at the boundary between complete and partial occlusion. In order to produce good segmentations in spite of this difficulty, we use two cues. The first cue is that pixels on the boundary of the region of complete occlusion have gradients of R that point into the region of partial occlusion. This is assured by the pre-processing of Eq. 3, which causes the foreground object to have the lowest radiance in R. The second cue is that points outside of the completely occluded region will generally have intensities that differ from the foreground intensity.

To exploit these two cues, we employ the flux maximizing geometric flow of [13], which evolves a 2D curve to increase the outward flux of a static vector field through its boundary. Our cues are embodied in the vector field

$$\overrightarrow{\mathcal{V}} = \phi \frac{\nabla R}{|\nabla R|}, \quad \text{where} \quad \phi = (1+R)^{-2}. \tag{4}$$

The vector field $\frac{\nabla R}{|\nabla R|}$ embodies the first cue, representing the direction of the gradient, which is expected to align with the desired boundary as well as be orthogonal to it[2]. The scalar field ϕ, which embodies the second cue, is near 1 in the completely-occluded region and smaller elsewhere. As noted in [6], an exponential form for ϕ can be used to produce a monotonically-decreasing function of R, giving similar results. The curve evolution equation works out to be

$$\mathcal{C}_t = div(\overrightarrow{\mathcal{V}})\overrightarrow{\mathcal{N}} = \left[\left\langle \nabla\phi, \frac{\nabla R}{|\nabla R|} \right\rangle + \phi \kappa_R \right] \overrightarrow{\mathcal{N}}, \tag{5}$$

where κ_R is the Euclidean mean curvature of the iso-intensity level set of the image. The flow cannot leak outside the completely occluded region since by construction both ϕ and $\nabla\phi$ are nearly zero there.

This curve evolution, which starts from a small circular region containing the user-selected point, may produce a boundary that is not smooth in the presence of noise. In order to obtain a smooth curve, from which outward normals can be robustly estimated, we apply a few iterations of the euclidean curve-shortening flow [4]. While it is possible to include a curvature term in the flux-maximizing flow to evolve a smooth contour, we separate the terms into different flows which are computed in sequence. Both flows are implemented using level set methods [10]; details are given in the Appendix.

[2] It is important to normalize the gradient of the image so that its magnitude does not dominate the measure outside of the occluded region.

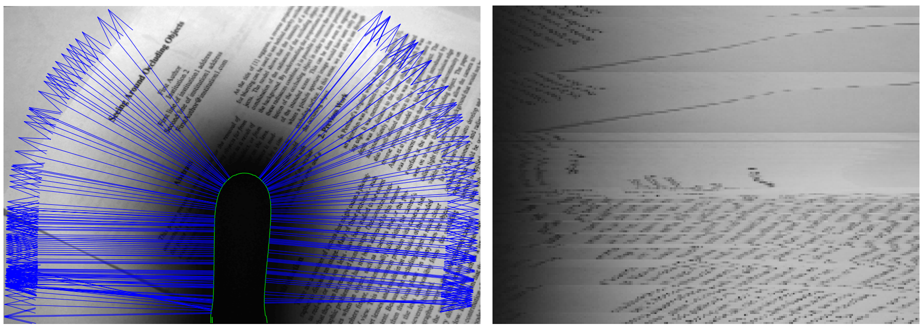

Fig. 3. (Left) Original image with segmentation boundary (green) and outward-facing normals (blue) along which the image will be re-sampled. (Right) The re-sampled image (scaled), which is used to estimate the blur width.

Once the curve-shortening flow has terminated, we can recover the radiance R_f of the foreground (occluding) object by simply taking the mean radiance value within the segmented (completely occluded) region. Note that we use this instead of R_p, the radiance of the user-selected point, as there may be some low-frequency intensity variation within the region of complete occlusion.

4.3 Boundary Rectification and Profile Generation

One of the difficulties in measuring the blur width is that the boundary of the completely occluded region can have an arbitrary shape. In order to handle this, we re-sample the image R along outward-facing normals to the segmentation boundary, reducing the shape of the occluding contour to a line along the left edge of a re-sampled image R^l. The number of rows in R^l is determined by the number of points on the segmentation boundary, and pixels in the same row of R^l come from points on the same outward-facing normal. Pixels in the same column come from points the same distance from the segmentation boundary on different normals and thus, recalling Eq. 2, have the same α value. The number of columns in the image depends on the distance from the segmentation boundary to the edge of the input image. We choose this quantity to be the largest value such that 80% of the normals remain within the image frame and do not re-enter the completely-occluded region (this exact quantity is arbitrary and the method is not sensitive to variations in this choice). Fig. 3 shows outward-facing surface normals from the contour in Fig. 2, along with the re-sampled image.

The task of measuring the width of the partially occluded region is also complicated by the generality of the background intensity. In the worst case, it is impossible (for human observers or our algorithm) to estimate the width if the background has an intensity gradient in the opposite direction of the intensity gradient due to partial occlusion. The measurement is straightforward if the background object had constant intensity, though this assumption is too strong. Given that the blurred region is a horizontal feature in the re-sampled image, we average rows of R^l in order to smooth out high-frequency background

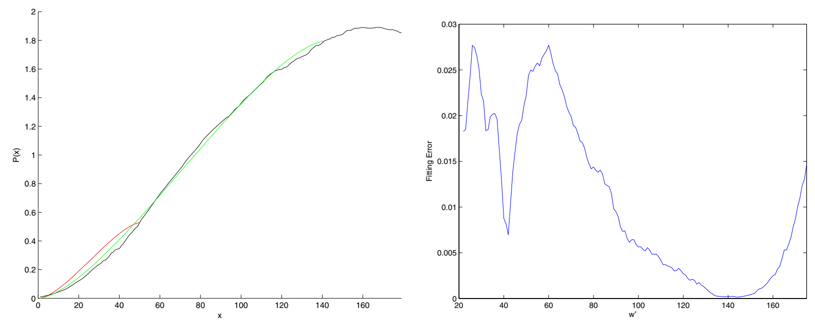

Fig. 4. [Left] Profile generated from the re-sampled image in Fig. 3 (black curve). Model profile P_m^{50} with relatively high error (red curve). Model profile P_m^{141} with minimum error (green curve). [Right] Fitting error as a function of w'.

texture. While we *do not assume* a uniform background, we have found it useful to eliminate rows with relatively more high-frequency structure before averaging. In particular, for each row of R^l we compute the sum of its absolute horizontal derivatives

$$\sum_x \|R^l(x+1,y) - R^l(x-1,y)\|. \tag{6}$$

Rows with an activity measure in the top 70% are discarded, and the remaining rows are averaged to generate the one dimensional blur profile P.

4.4 Blur Width Estimation

Given a 1D blur profile P, like the one shown in Fig. 4 (black curve), we must estimate the width w of the partially occluded region. We do this by first expressing P in terms of α. Recalling Eq. 3 and the fact that $R_f \approx R_p$, we rearrange Eq. 1 to get

$$R^l(x,y) = (1-\alpha(x,y))\|R_b^l(x,y) - R_f\|, \tag{7}$$

where R_b^l is the radiance of the background object defined on the same lattice as the re-sampled image. The profile $P(x)$ is the average of many radiances from pixels *with the same* α *value*, so

$$P(x) = (1-\alpha(x))\|\overline{R_b^l}(x) - R_f\|, \tag{8}$$

where $\overline{R_b^l}(x)$ is the average radiance of background points a fixed distance from the segmentation boundary (which fall in a column of the re-sampled image). As we have removed rows with significant high-frequency structure and averaged the rows of the re-sampled image, we assume that the values $\overline{R_b^l}(x)$ are relatively constant over the partially-occluded band, and thus

$$P(x) = (1-\alpha(x))\|\overline{R_b^l} - R_f\|. \tag{9}$$

Based on this, the blur width w is taken to be the value that minimizes the average fitting error between the measured profile P and model profiles. The model profile $P_m^{w'}$ for a given width w' is constructed by first generating a linear ramp l and then transforming these values into α values by Eq. 2.

An example is shown in Fig. 4, where the green curve shows the model profile for which the error is minimized with respect to the measured profile (black curve), and the red curve shows another model profile which has higher error. A plot of the error as a function of w' is shown in figure 4. We see that it has a well-defined global minimum, which is at $w = 141$ pixels.

4.5 Blur Removal

Once the segmentation boundary and the width w of the partially-occluded region have been determined, the value of α can be found using Eq. 2. In order to compute α at each pixel, we must find its distance to the nearest point on the segmentation boundary. We employ the fast marching method of [10].

Recall that the radiance R_f of the foreground object was found previously, so we can recover the radiance of the background at pixel (x, y) according to

$$R_b(x, y) = \frac{R_{input}(x, y) - \alpha(x, y)R_f}{1 - \alpha(x, y)}. \tag{10}$$

Finally, the processed image R_b is tone-mapped to produce the output image. This tone-mapping is simply the inverse of what was done in section 4.1.

5 Experimental Results

Fig. 5 shows the processed result from the example image in Fig. 2. The user-selected point was near the center of the completely occluded region though, in our experience, the segmentation is insensitive to the location of the initial point. We also show enlargements of a region near the occluding contour to illustrate the details that become clearer after processing. Near the contour, as $\alpha \to 1$, noise becomes an issue. This is because we are amplifying a small part of the input signal, namely the part that was contributed by the background.

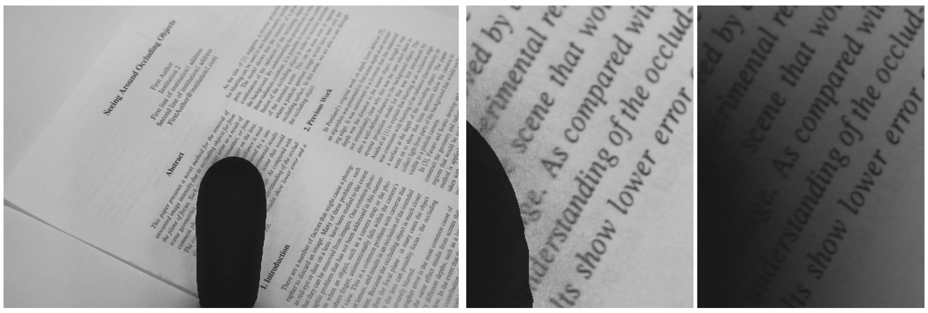

Fig. 5. (Left) Result for the image shown in Fig. 2. (Center) Enlargement of processed result. (Right) Enlargement of the corresponding region in the input image.

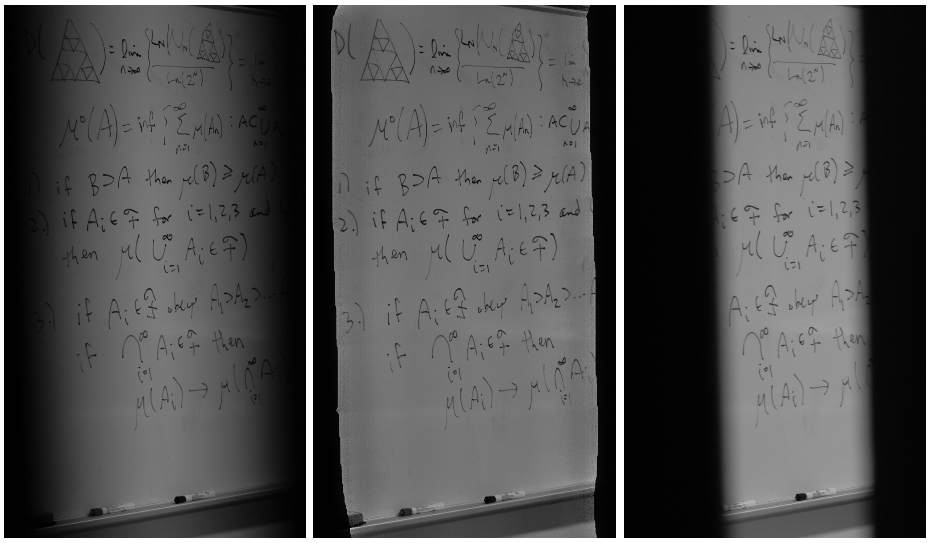

Fig. 6. Example scene through a small opening. (Left) Input wide-aperture image. (Middle) Output wide-aperture image. (Right) Reference small-aperture image. Notice that more of the background is visible in our processed wide-aperture image.

Fig. 6 shows an additional input and output image pair, along with a reference image taken through a smaller aperture. The photos were taken through a slightly opened door. It is important to note that processing the wide aperture photo reveals background detail in parts of the scene where a small aperture is completely occluded. Namely, all pixels where $\alpha > .5$ are occluded in a pinhole aperture image, though many of them can be recovered by processing a wide aperture image. In this scene, there are two disjoint regions of complete occlusion, each of which has an adjacent region of partial occlusion. This was handled by having the user select two starting points from which the segmentation flow was initialized, though the method could also have been applied separately to each occluded region.

The method described in this paper can also be extended to video processing. In the event that the location of the camera and the occluding object are fixed relative to one another, we need only perform the segmentation and blur estimation on a single frame of the video. The recovered value of α at each pixel (the matte) can be used to process each frame of the video separately. A movie, `keyholevideo.mpg`, is included in the supplemental material with this submission, and shows the raw and processed frames side-by-side (as in Fig. 1).

6 Conclusion

The examples in the previous section demonstrate how our method automatically measures the blur parameters and removes partial occlusion due to nearby objects. Fig. 6 shows that pictures taken through small openings (such as a fence, keyhole, or slightly opened door) can be processed to improve visibility. In this and the case of the text image shown in Fig. 5, this method reveals important image information that was previously difficult to see.

The automated nature of this method makes the recovery of partially-occluded scene content accessible to the average computer user. Users need only specify a single point in each completely occluded region, and the execution time of 10-20 seconds is likely acceptable. Given such a tool, users could significantly improve the quality of images with partial occlusions.

In order to automate the recovery of the necessary parameters, we have assumed that the combination of blurring and under-exposure produces a foreground region with nearly constant intensity. Methods that allow us to relax this assumption are the focus of ongoing future work, and must address significant additional complexity in each of the segmentation, blur width estimation, and blur removal steps.

References

1. Chuang, Y., Curless, B., Salesin, D.H., Szeliski, R.: A Bayesian Approach to Digital Matting. In: CVPR 2001, pp. 264–271 (2001)
2. Debevec, P., Malik, J.: Recovering High Dynamic Range Radiance Maps from Photographs. In: SIGGRAPH 1997, pp. 369–378 (1997)
3. Favaro, P., Soatto, S.: Seeing Beyond Occlusions (and other marvels of a finite lens aperture). In: CVPR 2003, pp. 579–586 (2003)
4. Grayson, M.: The Heat Equation Shrinks Embedded Plane Curves to Round Points. Journal of Differential Geometry 26, 285–314 (1987)
5. Levoy, M., Chen, B., Vaish, V., Horowitz, M., McDowall, I., Bolas, M.: Synthetic Aperture Confocal Imaging. In: SIGGRAPH 2004, pp. 825–834 (2004)
6. Perona, P., Malik, J.: Scale-Space and Edge Detection using Anisotropic Diffusion. IEEE Trans. on Patt. Anal. and Mach. Intell. 12(7), 629–639 (1990)
7. McCloskey, S., Langer, M., Siddiqi, K.: Seeing Around Occluding Objects. In: Proc. of the Int. Conf. on Patt. Recog. vol. 1, pp. 963–966 (2006)
8. McGuire, M., Matusik, W., Pfister, H., Hughes, J.F., Durand, F.: Defocus Video Matting. ACM Trans. Graph. 24(3) (2005)
9. Reinhard, E., Khan, E.A.: Depth-of-field-based Alpha-matte Extraction. In: Proc. 2nd Symp. on Applied Perception in Graphics and Visualization 2005, pp. 95–102 (2005)
10. Sethian, J.A.: Level Set Methods and Fast Marching Methods. Cambridge University Press, Cambridge (1999)
11. Sun, J., Jia, J., Tang, C., Shum, H.: Poisson Matting. ACM Trans. Graph. 23(3) (2004)
12. Tamaki, T., Suzuki, H.: String-like Occluding Region Extraction for Background Restoration. In: Proc. of the Int. Conf. on Patt. Recog. vol. 3, pp. 615–618 (2006)
13. Vasilevskiy, A., Siddiqi, K.: Flux Maximizing Geometric Flows. IEEE Trans. on Patt. Anal. and Mach. Intell. 24(12), 1565–1578 (2002)

Appendix: Implementation Details

For the experiments shown here, we down-sample the original 6MP images to 334 by 502 pixels for segmentation and blur width estimation. Blur removal is performed on the original 6MP images. Based on this, blur estimation and image processing takes approximately 10 seconds (on a 3 GHz Pentium IV) to produce the output in Fig. 5. Other images take more or less time, depending on the size of the completely-occluded region. Readers should note that some of the code used in this implementation was written in Matlab, implying that the execution time could be further reduced in future versions.

As outlined in section 4.2, we initially use a flux-maximizing flow to perform the segmentation, followed by a euclidean curve-shortening flow to produce a smooth contour. For the flux-maximizing flow, we evolve the level function with speed $\Delta t = 0.1$. This parameter was chosen to ensure stability for our 6MP images; in general it depends on image size. The evolution's running time depends on the size of the foreground region. The curve evolution is terminated if it fails to increase the segmented area by 0.01% over 10 iterations. As the flux-maximizing flow uses an image-based speed term, we use a narrow-band implementation [10] with a bandwidth of 10 pixels.