Boundary Matting for View Synthesis

Samuel W. Hasinoff[a,*], Sing Bing Kang[b], and Richard Szeliski[b]

[a]*Dept. of Computer Science, University of Toronto, Toronto, Canada*

[b]*Interactive Visual Media Group, Microsoft Research, Redmond, WA, USA*

Abstract

In the last few years, new view synthesis has emerged as an important application of 3D stereo reconstruction. While the quality of stereo has improved, it is still imperfect, and a unique depth is typically assigned to every pixel. This is problematic at object boundaries, where the pixel colors are mixtures of foreground and background colors. Interpolating views without explicitly accounting for this effect results in objects with a "cut-out" appearance.

To produce seamless view interpolation, we propose a method called *boundary matting*, which represents each occlusion boundary as a 3D curve. We show how this method exploits multiple views to perform fully automatic alpha matting and to simultaneously refine stereo depths at the boundaries. The key to our approach is the 3D representation of occlusion boundaries estimated to sub-pixel accuracy. Starting from an initial estimate derived from stereo, we optimize the curve parameters and the foreground colors near the boundaries. Our objective function maximizes consistency with the input images, favors boundaries aligned with strong edges, and damps large perturbations of the curves. Experimental results suggest that this method enables high-quality view synthesis with reduced matting artifacts.

Key words: Multi-view stereo, view synthesis, image matting, occlusion boundaries, sub-pixel reconstruction, 3D curves

1 Introduction

Although stereo correspondence was one of the first problems in computer vision to be extensively studied, automatically obtaining dense and accurate estimates of depth from multiple images remains a challenging problem [1].

[*] Corresponding author.
Email address: hasinoff@cs.toronto.edu (Samuel W. Hasinoff).

Preprint submitted to Elsevier Science *26 October 2005*

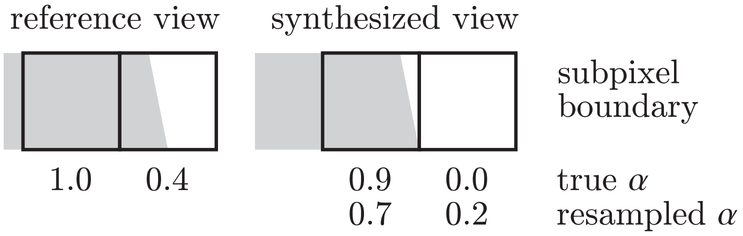

Fig. 1. *View synthesis with matting. The shaded polygon represents the foreground object, and the overlaid squares represent pixels. If the object is represented with an exact sub-pixel boundary model, the true distribution of α (per-pixel fraction foreground contribution) can be recovered by integration. By contrast, synthesizing new views from a pixel-level representation requires resampling α, which can lead to blurring artifacts at object boundaries. The synthesized view corresponds to a half-pixel translation, resampled with linear interpolation.*

Most stereo research has been concerned solely with methods for producing accurate depth maps, so interpolated views are rarely evaluated as results. By contrast, our explicit goal is superior view synthesis from stereo. Even for easy scenes in which all objects are opaque, diffuse, and well-textured, state-of-the-art stereo techniques often fail to generate high-quality interpolated views. Even if a perfect depth map were available, current methods for view interpolation share two major limitations:

- **Sampling blur.** There is an effective loss of resolution caused by resampling and blending the input views.
- **Boundary artifacts.** Foreground objects seem to pop out of the scene, as in bad blue-screen composites, because most current methods do not perform matting to resolve mixed pixels at object boundaries into their foreground and background components. (There are a few notable exceptions, as discussed in the next section.)

In this paper, we focus on the issue of boundary artifacts and propose a technique we call *boundary matting* to reduce such artifacts. Our technique, as outlined in Figs. 2–3, combines ideas from image matting and stereo to resolve mixed boundary pixels. Our approach consists of estimating 3D curves over multiple views and uses stereo data to bootstrap this estimation.

The key feature of our approach is that occlusion boundaries are represented in 3D. This results in several improvements over the state of the art. First, compared to video matting [2] and other methods that recover pixel-level mattes for the input views [3–6], our method is theoretically better suited to view synthesis, because it avoids the blurring associated with resampling those mattes (Fig. 1). Second, our method performs automatic matting from imperfect stereo data, fully incorporating multiple views, for large-scale opaque objects. Third, our method exploits information from matting to refine stereo disparities along occlusion boundaries. Fourth, our method estimates occlusion boundaries to sub-pixel accuracy, suitable for super-resolution or zooming. Fifth, our error metric is symmetric with respect to the input images, and so does not overly favor specific frames.

Our approach is based on several assumptions. First, we assume that the scene is made up of opaque Lambertian surfaces, i.e., surfaces that satisfy color constancy across the different input views. In practice, we can handle scenes that deviate somewhat from this assumption, treating non-Lambertian effects near object boundaries as noise. Moreover, we do not consider wide-baseline stereo configurations where these effects are most pronounced. Another important assumption is that the projected 2D boundaries correspond to the same 3D edge of an object. This is strictly true only for planar objects, however, this approximation is reasonable for small camera motion or relatively flat or distant objects (see Sec. 3.1).

2 Previous work

In their seminal blue screen matting paper, Smith and Blinn [7] review traditional film-based matting techniques and propose a *triangulation* method for matting static foreground objects using multiple images taken with different backgrounds (see Sec. 3). More recent matting research has focused on *natural image matting*, where the goal is to estimate the matte from a single image, given regions hand-labelled as completely foreground and background [8–12,4]. These methods operate by propagating statistics of the labelled color distributions throughout the unlabelled regions, yielding impressive results. Chuang *et al.* extend their approach from [9] using optical flow techniques to obtain a semi-automated method for matting video sequences [2]. Most recently, stereo data was used to automatically perform the initial labeling for natural image matting, but the matting was estimated independently in each view [5].

Several researchers have also investigated an additive transparent image formation model, useful for separating the reflections found on glass and specular surfaces [13,14]. Along the same lines, additive transparency has been decomposed based on parameterizing the dominant motions in the scene [15].

There has been some work done on estimating transparency from stereo data in general terms [6,16,17]. In [6], transparency was estimated in a volumetric fashion along with depth, using a plane-sweep algorithm, generalized to a four-dimensional $xyd\alpha$ search space. Results were mainly shown for synthetic problems, but even for those, the quality of interpolated views was limited. Similarly, the iterative voxel reconstruction approach presented in [16] gave results unsuitable for view synthesis, whereas [17] is more appropriate for volumetric scenes that are semi-transparent everywhere. Mixed pixels for stereo

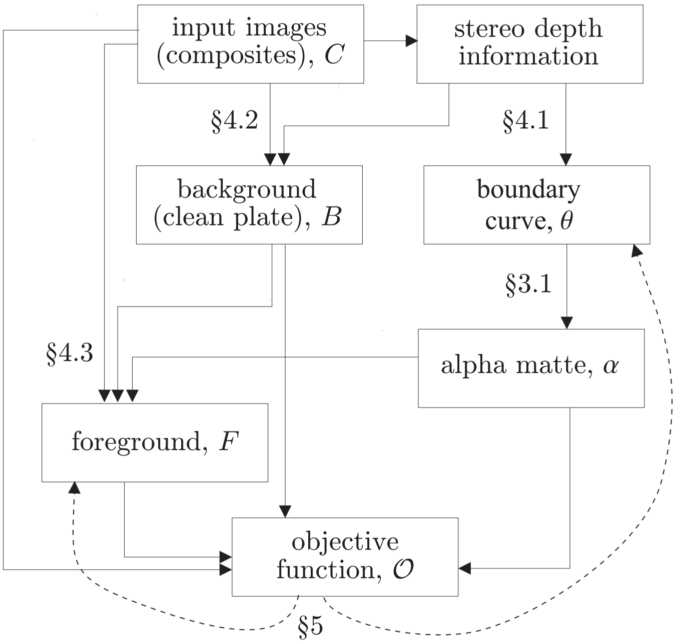

Fig. 2. *Block diagram describing the system architecture. The dashed lines indicate that the objective function is used to optimize the parameters of the boundary curve and the foreground colors.*

have also been examined as a consequence of using mixture models for estimating optical flow [18], and in developing matching metrics more robust to mixing [19].

Most closely related to our work is the method proposed by Wexler *et al.* [3], which also estimates matting by incorporating multiple views of a scene. However, as described in Sec. 3, this method effectively calculates alpha in the reference view only, and therefore requires resampling the mixed pixels (i.e., alpha values) from other views. As shown in Fig. 1, this can introduce undesirable blurring. Another basic limitation of this method is that high quality stereo data is required everywhere in the image, and its performance on inaccurate stereo is unclear. In practice both these problems were circumvented in [3] by considering scenes consisting of two planar structures, and demonstrating object insertion in the reference view rather than view synthesis. In contrast, since our method is based on a 3D curve representation (see Sec. 3), the alpha matte has a well-defined geometric interpretation that is consistent across arbitrary nearby views. Moreover, we tolerate some inaccuracy in the stereo data by simultaneously estimating the matting and refining the disparity estimates (i.e., by adjusting the boundary curve).

Our geometric view of α has precedence in work on (single view) user-assisted segmentation of opaque objects [20,21]. Here, α is estimated from the fractional pixel coverage given by a sub-pixel parametric edge model fit to the object boundaries. Both methods require manual interaction at key frames,

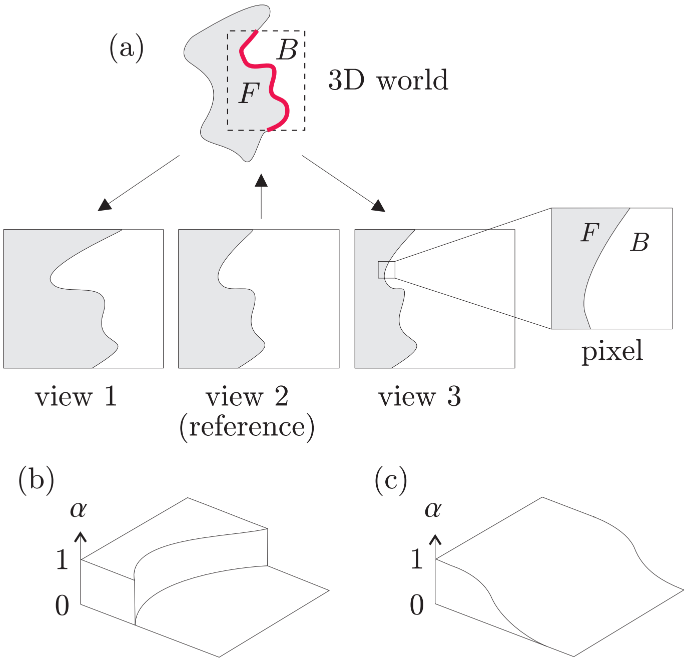

Fig. 3. *Geometric view of the system. (a) Stereo depth information is used to detect an occlusion boundary in the reference view, which is backprojected to 3D as our initial curve estimate. The 3D curve is refined, along with estimates for F color, by evaluating the projections of the curve in all input views. The value of α for a given pixel can be computed from the projected curve geometrically. (b) In the simplest case, α corresponds to the fraction of area on the F side of the curve. (c) Smoother blurring of the continuous alpha matte is a more realistic model.*

and neither extend readily to multiple views. By comparison, our method is automatic, and multiple views are fully incorporated. Along similar lines, subpixel edge geometry has been used to interpolate sparse point samples for rendering synthetic scenes, to better respect object and shadow boundaries [22]. In the recent matting literature [11,4], object boundary geometry has been represented implicitly using smoothness priors on the alpha profile, in order to regularize the matting. However, like the previous work, these methods require user interaction and only operate on individual images.

In one recent approach to view synthesis [23], the matting problem is handled implicitly, by incorporating an image-based prior that describes the quality of the final synthesized view in terms of how well it resembles exemplar patches from the input images. If there are enough input images to adequately sample alpha variation at the boundaries, this prior may indeed lead to plausible view synthesis at the boundaries. The lack of an explicit boundary model, however, means that compositing a new object into the scene, for example, would be problematic. This image-based exemplar approach was also used in [24], but they take the further step of estimating matting within each patch by extrapolating the occluded background colors.

3 Image formation model

To model the matting effects at occlusion boundaries, we use the well-known compositing equation [7,25]

$$C = \alpha F + (1 - \alpha)B \ , \tag{1}$$

which describes the observed composite color C as a blend of the foreground color F and the background color B according to opacity α. The alpha matte is typically given at the pixel level, so fractional α's may be due to partial pixel coverage of foreground objects at their boundaries or due to true semi-transparency. In this work, we focus exclusively on case where objects are opaque and alpha values are entirely due to the micro-geometry of partial pixel coverage.

Our method for inverting Eq. (1) exploits stereo information, and extends the *triangulation* method for matting [7]. In its classic form, the triangulation method operates by observing foreground objects in front of V known backgrounds, giving the linear system

$$\{ \ C_i = \alpha F + (1 - \alpha)B_i \ \}_{i=1}^{V} \ , \tag{2}$$

with $3V$ equations (one per RGB channel) in $3+1$ unknowns (F and α). This system is well-posed for $V \geq 2$, provided that the background colors for each pixel are different.

Instead of using a fixed camera and substituting different backgrounds behind the foreground objects, we use multiple views to provide us with images of the same foreground region against different backgrounds, as in [3]. This approach is valid under the assumption that foreground color varies little over nearby views, and provided that we can obtain the unoccluded background colors using stereo.

Unlike our method, [3] is based directly on the framework of Eq. (2), where α's for corresponding pixels are assumed not to vary across viewpoint, so in effect, α is estimated only in the reference view. By contrast, the 3D sub-pixel boundary curves in our method lead to different α's across viewpoint in general. We therefore obtain the revised linear system

$$\{ \ C_i = \alpha_i F + (1 - \alpha_i)B_i \ \}_{i=1}^{V} \ , \tag{3}$$

consisting is $3V$ equations in $3 + V$ unknowns (F and $\{\alpha_i\}$). Another consequence of viewpoint-varying alpha is that we can potentially resolve the standard ambiguity where background color is constant over all views. Note that *we do not restrict our calculations to a reference image*, as in [3].

3.1 Boundary curves in 3D with blurring

We model the occlusion boundary of a foreground object as a single (possibly open) 3D curve. For such a curve to be globally consistent with all of its projections, we assume that the occluding contours of the foreground objects are sufficiently sharp relative to both the closeness of the views and the standoff distance of the cameras (unlike, e.g., [26], which assumes that the object surface may be smoothly curved). Even for relatively smoother contours, although the boundary curve only approximates a path through the swept occlusion surface, this approximation may still be accurate enough to improve our estimation of α. After refinement, our method localizes this curve to sub-pixel precision.

In our approach, we model the 3D curve as a spline parameterized by control points, θ. For now we take this curve to be piecewise linear, parameterized using the (metric) 3D coordinates of the control points. While the extension to higher-order splines should be straightforward, using linear splines affords us ease of implementation and a simple way of modeling sharp corners. Moreover, linear splines can model arbitrarily complicated curves given enough control points. We can write the linear 3D spline in explicit parametric form as

$$S(t) = \sum_{p=1}^{n(\theta)} B_p(t)\theta_p \ ,$$

for $t \in [1, n(\theta)]$, where $n(\theta)$ is the number of control points, and the bases $B_p(t)$ are linear hat functions centered on each knot,

$$B_p(t) = \begin{cases} t-p+1, & t \in [p-1, p) \\ p+1-t, & t \in [p, p+1) \\ 0, & \text{otherwise} \ . \end{cases}$$

In practice, our control points are spaced several pixels apart and therefore cannot model such extremely fine-scale objects as hair or foliage. Rather, for such objects, splines can only approximate partial pixel coverage along occlusion boundaries.

Given the camera projection matrix, Π_i, for a particular view, we construct a signed distance function from the projected curve, $d(\Pi_i, \theta)$, defined to be positive on the foreground side and negative on the background side. In the ideal case, with a Dirac point spread function, the continuous alpha matte for the i-th view is

$$\alpha_i(\theta) = \begin{cases} 1, & d(\Pi_i, \theta) > 0 \\ 0, & \text{otherwise} \ . \end{cases} \tag{4}$$

This is a simple 2D step function of the curve parameters (Fig. 3(b)).

We simulate image blurring due to camera optics and motion by convolving α with an isotropic 2D Gaussian function $\mathcal{N}(0, \sigma)$:

$$\begin{aligned} \alpha_i(\theta, \sigma) &= \alpha_i(\theta) * \mathcal{N}(0, \sigma) \\ &= \tfrac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{d(\Pi_i, \theta)} \exp\left(\tfrac{-t^2}{2\sigma^2}\right) \, dt \ . \end{aligned} \tag{5}$$

This modified model gives us a smoothed step function for α (Fig. 3(c)), parameterized using a single additional variable σ.

For a given pixel j, we can generate the resulting pixel-level α-value by integrating one of the continuous α functions proposed in Eqs. (4–5) over the footprint of that pixel. For view i, this gives $\alpha_{ij} = \iint_j \alpha_i$. For the ideal case of Eq. (4), this is equivalent to computing the area on the foreground side of the projected curve, which has a simple form when the curve is piecewise linear. The blurred model of Eq. (5) is more complicated, so we approximate the integral by supersampling. More specifically, we supersample by a factor of 2 in the x- and y-dimensions.

3.2 Objective function

We formulate boundary matting as estimating the 3D boundary curve and foreground colors that best fit the V input images. Our primary goal is to minimize inconsistency with the images, according to the matting equation, Eq. (1). This leads to a basic objective function encoding the total cost of matting inconsistency:

$$\mathcal{O}(\theta, F) = \textstyle\sum_{i=1}^{V} \sum_{j=1}^{N_i} \left[\, C_{ij} - \ \alpha_{ij}(\theta) \, F_j - (1 - \alpha_{ij}(\theta)) \, B_{ij} \, \right]^2 \ , \tag{6}$$

where N_i is the number of pixels along the curve in view i. In practice, we evaluate this objective function over all pixels in a conservatively wide band around the boundary curve (see, for example, Fig. 6(b)), to ensure that every mixed pixel contributes to Eq. (6). Any pixel far enough from the boundary to be purely foreground ($\alpha = 1$) or background ($\alpha = 0$) will have no effect on the optimization, because the matting will have a trivial solution, namely $C = F$ or $C = B$.

If we are using the blurred image formation model from Eq. (5), we also need to determine the optimal value for the blur parameter σ. Currently, we estimate this parameter using a coarse exhaustive search, as an outer loop separate from the rest of the optimization (Sec. 5).

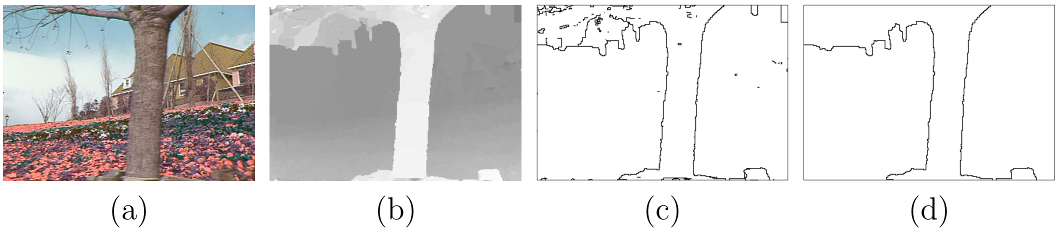

Fig. 4. *Boundary initialization. For a well-known stereo sequence, we show (a) the reference image, (b) the disparity map calculated using [27], (c) the* depth discontinuity map*, corresponding to the thresholded gradient of disparity, and (d) the initial occlusion boundaries extracted from (c).*

4 Initialization using stereo data

The starting point for boundary matting is an initialization derived from stereo and the attendant camera calibration. Boundary matting can use stereo data from any source; however, we chose to use results generated with [27] because its performance at occlusion boundaries was reasonable and an implementation was readily available. This method computes stereo by combining shiftable windows for matching with global minimization using graph cuts for visibility reasoning.

While initialization depends on the accuracy of the stereo data, the matting is later refined using an optimization based on Eq. (6). Moreover, view synthesis with boundary matting should always constitute an improvement over naïve view synthesis, regardless of the source of stereo data.

In this section, we describe how to extract initial occlusion boundaries θ^0 from the stereo data, how to estimate the *clean-plate* background B for pixels near the occlusion boundaries, how to initialize our estimate of foreground color F^0, and how to construct a prior favoring strong edges at the boundary that can be used to tweak the initial guess.

4.1 Boundary initialization and approximation

To extract the initial curves θ^0 corresponding to occlusion boundaries, we first form a *depth discontinuity map* by applying a manually-selected threshold to the gradient of the disparity map for the reference view (Fig. 4(b–c)). This threshold should be chosen conservatively to include all object boundaries of interest with some possible spurious structure, yet not so high as to identify many discontinuities across smooth surfaces. For all of our experiments, we used the same disparity gradient threshold of 2.0 pixels.

Next, we greedily remove the longest four-connected curves from the depth

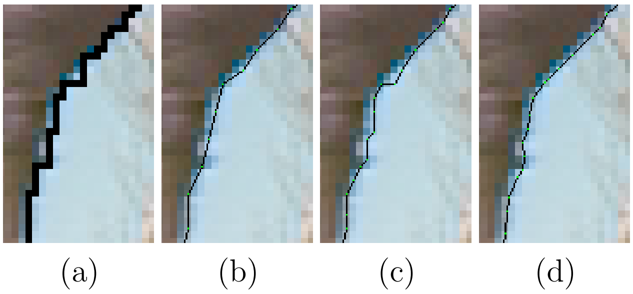

Fig. 5. *Spline fitting to an occlusion boundary. (a) Pixel-level occlusion boundary extracted from a region at the top-middle of Fig. 4(a). We fit a piecewise-linear 3D spline to this boundary and show it projected into the 2D image, overlaid at sub-pixel resolution (b–d). (b) Initial fit to the extracted boundary. (c) Adaptive subdivision in regions of poor matting. (d) Snapping to the strongest nearby edge within 1 pixel.*

discontinuity map until no curves longer than some minimum length remain (Fig. 4(d)):

(1) Partition the depth discontinuity map into four-connected components.
(2) Compute the diameter (the “longest shortest path”) of each component (e.g., using breadth-first search [28]).
(3) Greedily identify the boundary corresponding to the largest diameter, and remove it from the depth discontinuity map.
(4) Repeat Steps (1–3) until no diameter of some minimum length remains (we use a threshold of 70 pixels).

By growing the longest curves possible, we eliminate the small spurs and loops that are mainly due to inaccurate stereo.

This boundary extraction method is related to more sophisticated techniques for segmenting range images (see [29] for a review). However, for the purpose of reducing matting artifacts in view synthesis, our simpler method suffices. This is because matting artifacts will only occur when sufficient parallax causes some foreground object to be composited over a new background, i.e., exactly at the depth discontinuities identified by stereo.

To transform the extracted curves into 3D, we backproject the points along each curve using the (foreground-side) depth from stereo (Fig. 3). We then fit a 3D spline curve to these points (Fig. 5), simply by setting the control points θ^0 to be every fifth point along the four-connected curve (Fig. 5(b)).

After initial boundary extraction, we evaluate the curve for consistency with the matting equation (see Sec. 5 for more details). In regions with high matting error, we subdivide the curve once (Fig. 5(c)). While we have experimented with a general adaptive subdivision scheme, the four-connected boundary gives undesired staircase artifacts with tighter stopping criteria.

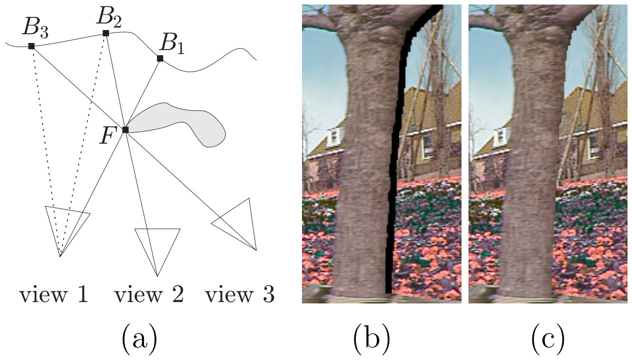

Fig. 6. *Estimation of the clean-plate background. (a) The region labelled F is a mixed pixel in all views. The background colors B_2 and B_3 can be obtained from view 1, by following the dashed lines. However, B_1 is occluded in all views. (b–c) A region of Fig. 4(a) is shown, (b) with pixels near the boundary highlighted in black, and (c) with these pixels filled in using our clean-plate background estimate.*

We also modify our initial guess to reflect the fact that occlusion boundaries tend to coincide with strong edges. To do this we perturb the control points in the reference image to the local peak of an edge potential field (Fig. 5(d)). We first apply a multiscale difference-of-Gaussians edge detector to each image, localizing edgels to sub-pixel precision and use this to pre-compute edge potential fields, $\{E_i\}$, quantized to 0.25 pixels. We define these fields as the sum of "forces" proportional to edgel strength and inversely proportional to squared edgel distance. Although edges are a strong cue for occlusion boundaries in many scenes, this heuristic can also be distracted by spurious internal texture, so we limit the perturbation to a one-pixel radius neighborhood.

4.2 Background (clean plate) estimation

As discussed in Sec. 3, using stereo data to triangulate the matting problem requires that the background B be known. A "clean plate" background refers to an image where foreground pixels are replaced with (unmixed) background colors, and is specified in many systems using manual interaction at keyframes [2,3]. However, this process can in theory be made automatic by exploiting stereo information to grab corresponding background colors from nearby frames in which the background is exposed (Fig. 6). Note that aside from specifying the initial 3D boundary curve, the only place our approach relies on accurate stereo is in warping the background from nearby views.

For a given boundary pixel, we find potentially corresponding background colors by forward-warping that pixel to all other views. This warping is performed according to the depth on the background side of the boundary, as given by stereo. If a forward-warped pixel has background depth in the new view, it becomes a candidate source from which to grab the background. We use nearest-neighbor sampling so that any mixed pixels falsely labeled as background can be more easily identified, and will not bias the background estimation.

Further to this end, we use a *color inconsistency measure* to select the corresponding background pixel most likely to consist of pure background color. For each candidate background pixel, we compute its "color inconsistency" as the maximum L_2 distance in RGB space between its color and any of its eight-neighbors that are also labeled at background depth. We then choose the background pixel with minimum color-inconsistency out of all views. This heuristic assumes slowly varying background texture, but seems to work well in most of our cases.

If a corresponding background pixel cannot be estimated (i.e., it is occluded by the foreground object in all views), it is marked as such and this pixel is not used directly in the optimization. In rendering the results, we either highlight these pixels as unknown, or use the naïve non-matting approach to determine color (i.e., $F = B = C$) but still estimate α from the curve.

4.3 Foreground estimation

Given an initial estimate for the curve parameters θ^0 (which determines α), along with the clean-plate background B and input images C, we can obtain a reasonable initialization for the foreground colors F^0 by simply inverting the matting equation, Eq. (1):

$$F^0(\alpha) = (C - (1 - \alpha)B)/\alpha \; . \tag{7}$$

The implied correspondence between foreground pixels is determined from stereo. Analogous to clean plate background estimation, we use the foreground-side depth from the boundary curve to warp the boundary pixels in the reference view to all other views.

For each pixel, we aggregate the foreground color estimates of Eq. (7) over all V views for robustness. To do this we take the weighted average,

$$F^0 = \frac{\sum_{i=1}^{V} \alpha_i^2 F^0(\alpha_i)}{\sum_{i=1}^{V} \alpha_i^2} \; , \tag{8}$$

with the weights constructed to favor foreground color information from pixels containing more foreground, based on the curve estimate. Note that this formula is also the statistically optimal least-squares estimate for F given the set of V i.i.d. noise-contaminated composite color pixels, $C_i = \alpha_i F + (1 - \alpha_i)B_i + \mathcal{N}(0, \sigma_{\text{noise}})$.

5 Parameter optimization

Now that we have constructed the clean-plate background, B (Sec. 4.2), and obtained initial estimates for the parameters of each boundary curve, θ^0 (Sec. 4.1), and the foreground colors, F^0 (Sec. 4.3), we are in a position to refine these estimates to better fit the images.

Note that the objective function, Eq. (6), is highly non-linear, with bilinearity in the variables, perspective projection, and a complicated form for alpha as the partial pixel coverage of a projected spline (possibly convolved some blurring). Therefore we resort to Levenberg-Marquardt nonlinear least-squares optimization [30] to refine the boundaries and foreground colors.

5.1 Two-stage estimation

In our experience, it is faster and more stable to first optimize the curve parameters only, dynamically updating our estimate of the foreground colors based on the alpha values derived from the curve, i.e., $F = F(\theta)$ (Sec. 4.3). We thus suggest a two-stage approach, where the optimized curve from the first stage is used as an initial guess for the joint estimation of both the curve and foreground colors. For each boundary curve, we do the following procedure:

(1) Refine the curve parameters by solving

$$\theta^1 = \arg\min_{\theta} \ \mathcal{O}(\theta, F(\theta)) \ ,$$

using Levenberg-Marquardt optimization, initialized with $\theta = \theta^0$ (Sec. 4.1) and $F = F(\theta^0)$ (Sec. 4.3).

(2) Jointly refine the curve parameters and the foreground colors, by solving

$$\{\theta, F\} = \arg\min_{\{\theta,F\}} \ \mathcal{O}(\theta, F) \ ,$$

using Levenberg-Marquardt optimization, initialized from Step (1) with $\theta = \theta^1$ and $F = F(\theta^1)$.

(3) (optional) Repeat Steps (1–2), for different values of the blur parameter σ, selecting the one that gives the lowest least-squares error.

We use an implementation of Levenberg-Marquart algorithm based on the Minpack library [31], where the step size and stopping criteria are both related to a parameter encoding the predicted accuracy of the objective function (we set this to 1.0×10^{-4}).

This optimization refines each curve and separates mixed pixels into background and foreground components. Note that if our initial estimate is more than one pixel away from the true boundary, we may get trapped in a local minimum, as differential changes to the curve parameters may not improve matting consistency. However, even for such gross stereo errors, the control points may have wide enough support that some pixels may gradually guide the curve closer to the true solution. The blurred image formation model of Eq. (5), i.e., $\sigma > 0$, is potentially more resilient to these errors than the basic model, because the control points have a larger support still.

While all gradients can be evaluated analytically, from an implementation standpoint it is more convenient to calculate the Jacobian for partial pixel coverage, $[\frac{\partial \alpha_{ij}}{\partial \Pi_i \theta_p}]_{ij,p}$, using a finite difference approximation. Note that this Jacobian is very sparse, as the locality of the spline ensures that each control point θ_p influences a limited number of pixels. For efficiency we therefore restrict gradient computation to these salient pixels.

5.2 Adding edge snapping and state damping

We also created a penalty function to bias the optimization to areas with stronger edges, so the overall optimization can be considered a kind of 3D snake [32]. This function reuses the edge potential fields, $\{E_i\}$, described in Sec. 4.1, normalized to have a maximum of one. We project all $n(\theta)$ control points, denoted $\{\theta_p\}$, into each view using the camera matrices $\{\Pi_i\}$, then calculate a penalty term proportional to inverse edge strength,

$$P_1(\theta) = \textstyle\sum_{i=1}^{V} \sum_{p=1}^{n(\theta)} \; [\, 1 - E_i(\Pi_i \theta_p)\,]^2 \;, \tag{9}$$

for the control points over all views.

An additional penalty function is used to discourage the control points from being displaced too far from their starting positions,

$$P_2(\theta) = \textstyle\sum_{i=1}^{V} \sum_{p=1}^{n(\theta)} \Big[\, \max\Big(0,\ ||\Pi_i\theta_p - \Pi_i\theta_p^0||^2 - 4.0\Big)\, \Big]^2 \;, \tag{10}$$

where θ_p^0 is the initial location of the p-th control point. The penalty is zero for displacements of 2.0 pixels or less, but increases rapidly after that. This function helps avoid degenerate configurations where the curve collapses on itself.

We add these penalty terms to the original objective function, Eq. (6), and express this succinctly as:

$$\mathcal{O}_{\text{new}}(\theta, F) = \mathcal{O}(\theta, F) + \lambda_1 P_1(\theta) + \lambda_2 P_2(\theta) \;. \tag{11}$$

Moderate values of λ_1 and λ_2 ensure that neither edges nor initial positions exert too much influence over the optimization. In practice, the optimization did not seem overly sensitive to these parameters, so the same setting was used across all datasets. These parameters were normalized by $k = N/n(\theta)$, and set to $\lambda_1 = 0.015k$ and $\lambda_2 = 0.053k$.

6 Results

For all datasets, we used five input views, with the middle view designated as the reference view for initialization. While our prototype system was not designed for efficiency, a typical run for a 300-pixel boundary in five views could take approximately five minutes to complete, converging within 20 iterations.

For our first experiment, we used a synthetic dataset (448 × 336 pixels), consisting of a planar ellipse-shaped sprite with pure translation relative to the background, in order to investigate the behavior of boundary matting under noise. Fig. 7 shows that boundary matting is visually indistinguishable from the ground truth in the noise-free case, where alpha values over the boundary pixels have an RMS error of 0.02. Boundary matting demonstrates further resilience to artificial noise in the input images, and the shape of the recovered boundary degrades gracefully as the noise level increases.

Next, we applied boundary matting to insert a new object between the foreground and background layers of three well-known stereo datasets (Fig. 8): the flower garden sequence (Fig. 4), the Middlebury sawtooth dataset [1] (Fig. 8, top middle), and the Samsung commercial sequence (Fig. 8, top right). These sequences are 344 × 240, 640 × 486, and 434 × 380 pixels respectively, with the calibration accurate to within about 0.5 pixels. For the flower garden and sawtooth datasets, haloing artifacts from the background layer are clearly reduced. Although the matting improvement is less dramatic for the Samsung sequence, this is mainly because the quality of naïve object insertion is relatively less objectionable, due to the high-quality initial stereo and the similar, desaturated colors of the foreground and background layers around the main subject's head. The similarity of colors between foreground and background layers is also a source of ambiguity for the matting, particularly in the self-shadowed regions of the hair.

Not only does boundary matting improve the composites, but the extracted boundaries can even be sharpened by rendering the curves at sub-pixel resolution (Fig. 8, bottom row). At close scales this sub-pixel rendering may appear less desirable for view synthesis due to the apparent resolution mismatch; however the exact sub-pixel nature of the boundaries allows us to reblur them by any amount.

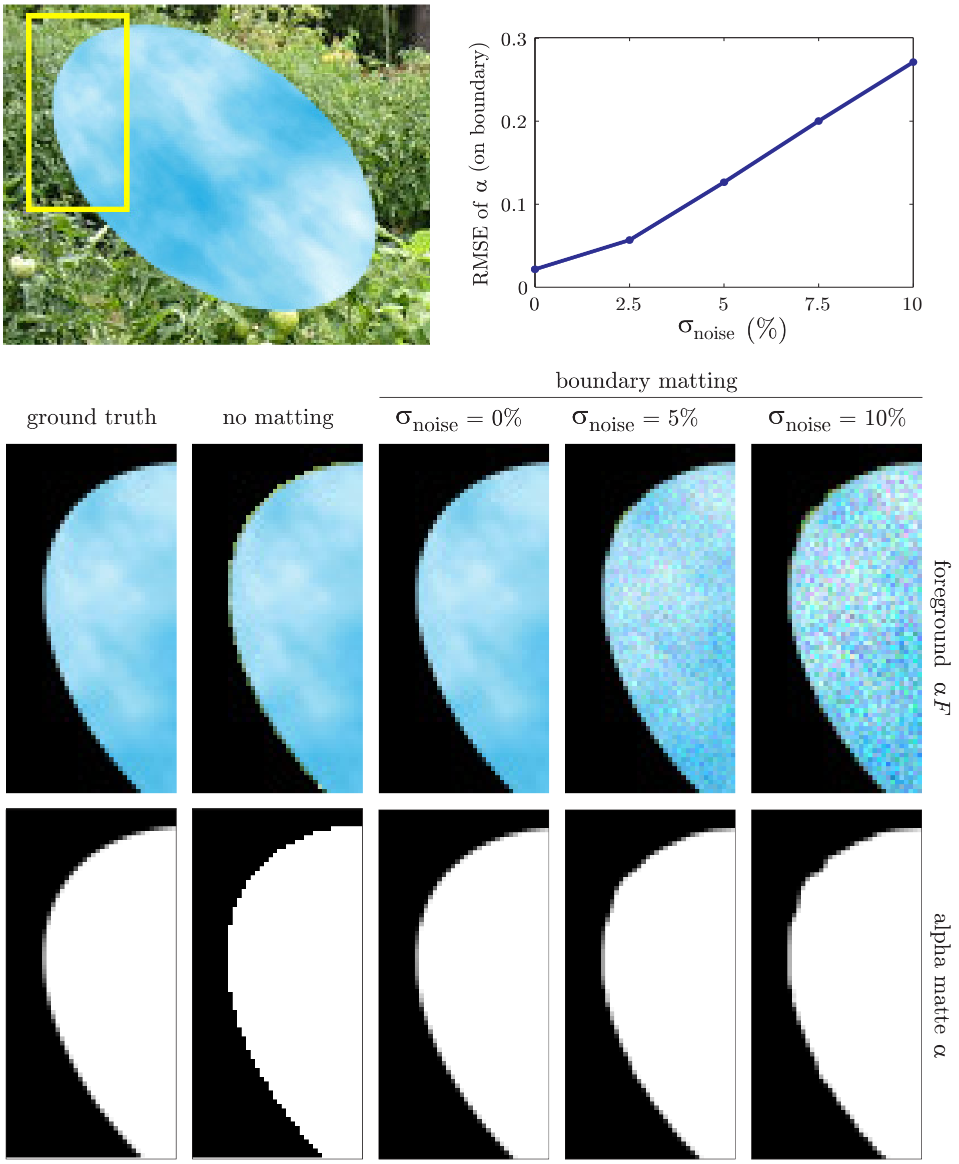

Fig. 7. *Boundary matting synthetic data, with the addition of zero-mean Gaussian noise. Top left: reference image of a planar textured ellipse, with zoomed region indicated. Top right: RMS error of α for pixels on the occlusion boundary. Bottom: visual comparison of the ground truth foreground and alpha matte with the boundary matting estimates given various levels of added noise. The naïve segmentation without matting is also shown for comparison.*

Finally, the flower garden dataset was also used for a view synthesis task, for matting both an input view and a novel interpolated view (Fig. 9). In both cases, boundary matting produces a significant improvement over naïve view synthesis (i.e., forward warping with a fixed footprint, then feathering between the warped images). These results also demonstrate some tolerance to inaccurate stereo, since the initial stereo estimate in the region shown was up to two pixels off. We also experimented with a variety of settings for the blur

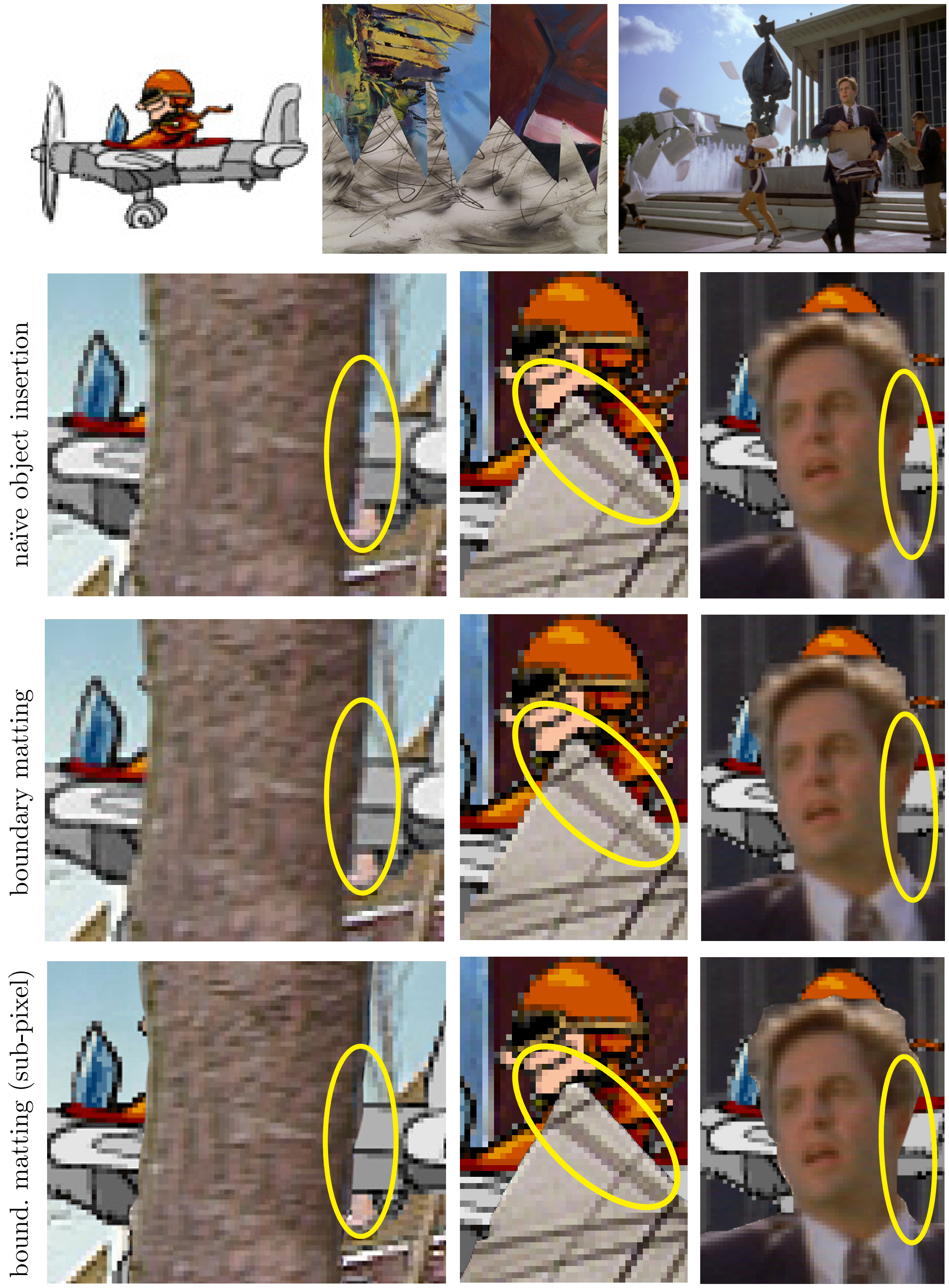

Fig. 8. *Object insertion. The airplane at top left is shown inserted behind the tree in Fig. 4(a), the foreground in the Middlebury sawtooth dataset (top middle), and the main figure in the Samsung sequence (top right). Naïve object insertion, without matting, leads to background spill and haloing artifacts. However, boundary matting significantly improves the composite. The underlying boundaries are also shown sharpened to sub-pixel resolution (although the mismatch in resolution may appear artificial). Regions of interest are highlighted for each dataset.*

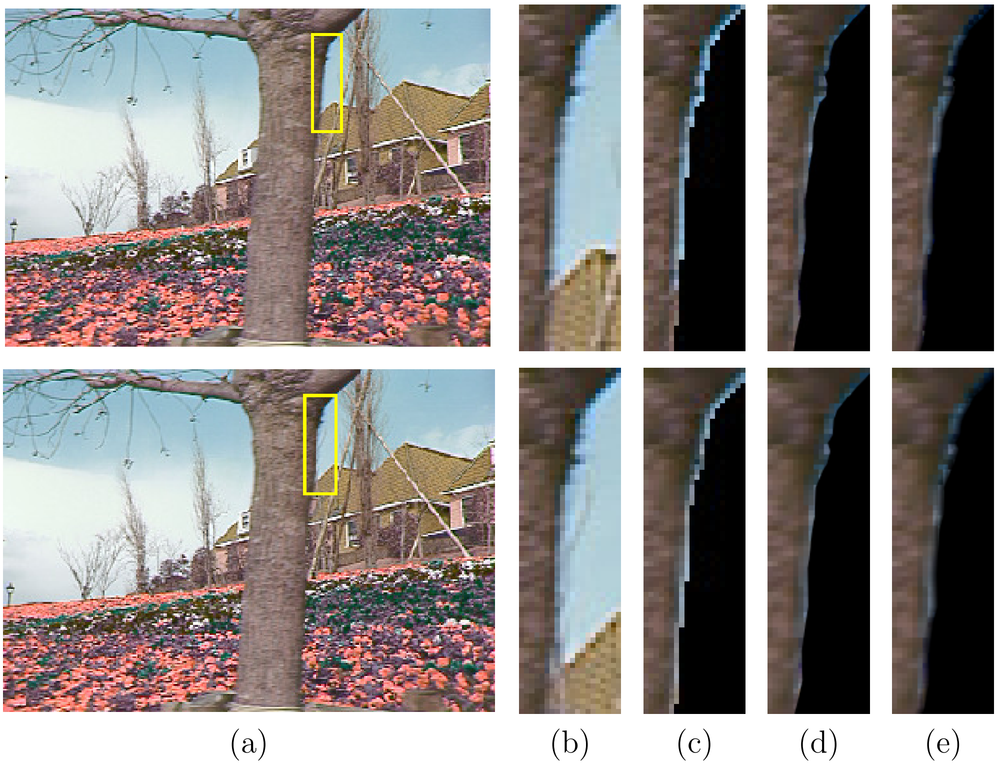

Fig. 9. *Boundary matting for view synthesis. The first row corresponds to the reference view, and the second row corresponds to an interpolated view. (a) Input image (ground truth interpolated view). (b) Zoomed-in region. (c) Naïve foreground separation without matting shows significant spill from the background layer. (d) Using the boundary matting method reduces this artifact. (e) Boundary matting with a blurred edge model (*$\sigma = 0.4$ *pixels) better matches the blur in the input images.*

parameter. While the addition of blur did not appear to improve the matting for this case, the optimal blurred boundary better matches the appearance of the input images.

For this portion of the dataset, some degree of blue spill from the sky remains even after performing boundary matting (Fig. 9(d–e)). This may be due to the optimization being trapped in a local minimum because of poor initialization. Another explanation is that the object curves smoothly enough that it cannot be accurately modeled using a single 3D boundary curve, and the boundary shown truly represents the global best-fit approximation.

Our method broke down completely for the upper-left region of the tree containing many twigs (Fig. 10), yet still performs no worse than naïve view synthesis ignoring matting. The reason for failure was not an inability to localize a consistent 3D curve, but rather that inaccurate stereo led to an initial boundary up to 30 pixels off. Without additional color-based priors, our matting method is content to accept depth discontinuities across untextured regions of sky, trapping the optimization in spurious local minima with $F = B$.

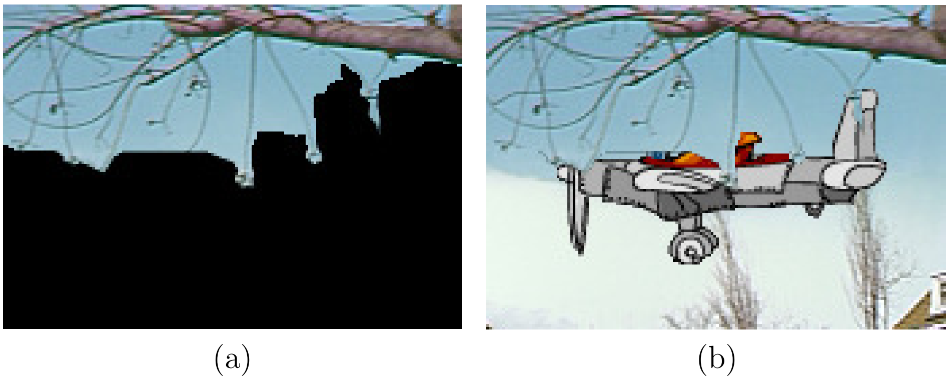

Fig. 10. *Boundary matting failure case. (a) Inaccurate stereo leads to a poor initial boundary estimate in a mainly untextured region, which cannot be refined successfully by locally optimizing the matting. Very thin structures such as the small branches may also pose difficulties. (b) Object insertion highlights the severity of the problem, although view synthesis is less objectionable in this region.*

7 Concluding remarks

For seamless view interpolation, mixed boundary pixels must be resolved into foreground and background components. Boundary matting appears to be a useful tool for addressing this problem in an automatic way. Using 3D curves to model occlusion boundaries is a natural representation that provides several benefits, including the ability to super-resolve the depth maps near occlusion boundaries.

A current limitation of our approach is its lack of reasoning about color statistics, which has proven very useful in natural image matting [9,2]. Such an ability might enable us to resolve boundaries even in areas where stereo gives grossly incorrect depths, as in the upper-left region of the tree in Fig. 4(a). By integrating boundary matting with complementary aspects of pixel-based matting methods [9,3], we hope to extend the generality of boundary matting while retaining its superior view synthesis.

In the future, we would also like to adapt boundary matting to a dynamic stereo framework, where disocclusions over time may reveal additional information to improve the matting.

References

[1] D. Scharstein, R. Szeliski, A taxonomy and evaluation of dense two-frame, stereo correspondence algorithms, Intl. Journal of Comp. Vision 47 (1) (2002) 7–42.

[2] Y.-Y. Chuang, A. Agarwala, B. Curless, D. H. Salesin, R. Szeliski, Video matting of complex scenes, in: Proc. ACM SIGGRAPH, 2002, pp. 243–248.

[3] Y. Wexler, A. W. Fitzgibbon, A. Zisserman, Bayesian estimation of layers from multiple images, in: Proc. ECCV, Vol. 3, 2002, pp. 487–501.

[4] H.-Y. Shum, J. Sun, S. Yamazaki, Y. Li, C.-K. Tang, Pop-up light field: An interactive image-based modeling and rendering system, in: Proc. ACM SIGGRAPH, 2004, pp. 143–162.

[5] C. L. Zitnick, S. B. Kang, M. Uyttendaele, S. Winder, R. Szeliski, High-quality video view interpolation using a layered representation, in: Proc. ACM SIGGRAPH, 2004, pp. 600–608.

[6] R. Szeliski, P. Golland, Stereo matching with transparency and matting, in: Proc. ICCV, 1998, pp. 517–526.

[7] A. R. Smith, J. F. Blinn, Blue screen matting, in: Proc. ACM SIGGRAPH, 1996, pp. 259–268.

[8] M. Ruzon, C. Tomasi, Alpha estimation in natural images, in: Proc. CVPR, 2000, pp. 18–25.

[9] Y.-Y. Chuang, B. Curless, D. H. Salesin, R. Szeliski, A Bayesian approach to digital matting, in: Proc. CVPR, 2001, pp. 264–271.

[10] P. Hillman, J. Hannah, D. Renshaw, Alpha channel estimation in high resolution image and image sequences, in: Proc. CVPR, 2001, pp. 1063–1068.

[11] C. Rother, A. Blake, V. Kolmogorov, "GrabCut" − Interactive foreground extraction using iterated graph cuts, in: Proc. ACM SIGGRAPH, 2004, pp. 309–314.

[12] J. Sun, J. Jia, C.-K. Tang, H.-Y. Shum, Poisson matting, in: Proc. ACM SIGGRAPH, 2004, pp. 315–321.

[13] R. Szeliski, S. Avidan, P. Anandan, Layer extraction from multiple images containing reflections and transparency, in: Proc. CVPR, 2000, pp. 246–253.

[14] Y. Tsin, S. Kang, R. Szeliski, Stereo matching with reflections and translucency, in: Proc. CVPR, 2003, pp. 702–709.

[15] M. Irani, B. Rousso, S. Peleg, Computing occluding and transparent motions., Intl. Journal of Comp. Vision 12 (1) (1994) 5–16.

[16] J. D. Bonet, P. Viola, Roxels: Responsibility weighted 3D volume reconstruction, in: Proc. ICCV, 1999, pp. 418–425.

[17] S. W. Hasinoff, K. N. Kutulakos, Photo-consistent 3D fire by flame-sheet decomposition, in: Proc. ICCV, 2003, pp. 1184–1191.

[18] S. Ju, M. J. Black, A. D. Jepson, Skin and bones: Multi-layer, locally affine, optical flow and regularization with transparency, in: Proc. CVPR, 1996, pp. 307–314.

[19] S. Birchfield, C. Tomasi, A pixel dissimilarity measure that is insensitive to image sampling, IEEE Trans. PAMI 20 (4) (1998) 401–406.

[20] T. Mitsunaga, T. Yokoyama, T. Totsuka, Autokey: Human assisted key extraction, in: Proc. ACM SIGGRAPH, 1995, pp. 265–272.

[21] E. N. Mortensen, W. A. Barrett, Toboggan-based intelligent scissors with a four parameter edge model, in: Proc. CVPR, 1999, pp. 2452–2458.

[22] K. Bala, B. Walter, D. Greenberg, Combining edges and points for interactive anti-aliased rendering, in: Proc. ACM SIGGRAPH, 2003, pp. 631–640.

[23] A. W. Fitzgibbon, Y. Wexler, A. Zisserman, Image-based rendering using image-based priors, in: Proc. ICCV, 2003, pp. 1176–1183.

[24] A. Criminisi, A. Blake, The SPS algorithm: Patching figural continuity and transparency by split-patch search, in: Proc. CVPR (1), 2004, pp. 342–349.

[25] T. Porter, T. Duff, Compositing digital images, in: Proc. ACM SIGGRAPH, 1984, pp. 253–259.

[26] J. J. Koenderink, What does the occluding contour tell us about solid shape?, Perception 13 (1984) 321–330.

[27] S. B. Kang, R. Szeliski, J. Chai, Handling occlusions in dense multi-view stereo, in: Proc. CVPR, 2001, pp. 103–110.

[28] D. B. West, Introduction to Graph Theory, 2nd Edition, Prentice-Hall Inc., 2001.

[29] A. Hoover, G. Jean-Baptiste, X. Jiang, P. J. Flynn, H. Bunke, D. B. Goldgof, K. K. Bowyer, D. W. Eggert, A. W. Fitzgibbon, R. B. Fisher, An experimental comparison of range image segmentation algorithms, IEEE Trans. PAMI 18 (7) (1996) 673–689.

[30] J. Nocedal, S. J. Wright, Numerical Optimization, Springer, 1999.

[31] J. Moré, B. Garbow, K. Hillstrom, Minpack, http://www.netlib.org/minpack.

[32] M. Kass, A. Witkin, D. Terzopoulos, Snakes: Active contour models, Intl. Journal of Comp. Vision 1 (4) (1988) 321–331.