MITSUBISHI ELECTRIC RESEARCH LABORATORIES
http://www.merl.com

Exploring Defocus Matting: Nonparametric Acceleration, Super-Resolution and Off-Center Matting

Neel Joshi, W. Matusik, S. Avidan, H. Pfister, W.T. Freeman

TR2007-052 March 2007

Abstract

Defocus matting is a fully automatic and passive method for pulling mattes from video captured with coaxial cameras that have different depths of field and planes of focus. Nonparametric sampling can accelerate the video-matting process from minutes to seconds per frame. In addition, a super-resolution technique efficiently bridges the gap between mattes from high-resolution video cameras and those from low-resolution cameras. Off-center matting pulls mattes for an external high-resolution camera that doesn't share the same center of projection as the low-resolution cameras used to capture the defocus matting data. In this article, we address these limitations and extend defocus matting in several important ways.

IEEE Computer Graphics and Applications

Exploring Defocus Matting

Nonparametric Acceleration, Super-Resolution, and Off-Center Matting

Neel Joshi
University of California, San Diego

Wojciech Matusik, Shai Avidan, and Hanspeter Pfister
Mitsubishi Electric Research Labs

William T. Freeman
Massachusetts Institute of Technology

Defocus matting is a fully automatic and passive method for pulling mattes from video captured with coaxial cameras that have different depths of field and planes of focus. Nonparametric sampling can accelerate the video-matting process from minutes to seconds per frame. In addition, a super-resolution technique efficiently bridges the gap between mattes from high-resolution video cameras and those from low-resolution cameras. Off-center matting pulls mattes for an external high-resolution camera that doesn't share the same center of projection as the low-resolution cameras used to capture the defocus matting data.

Image matting and compositing are important operations in image editing, photography, and film production. Matting separates a foreground element from an image by estimating a color, F, and an opacity, α, for each foreground pixel. The set of all α values is the *alpha matte*. Compositing blends the extracted foreground element, F, on top of an opaque background image, B, using linear blending: $I[x;y] = \alpha F + (1 - \alpha)B$. Given an image, I, matting solves the inverse problem with seven unknowns (α, F_r, F_g, F_b, B_r, B_g, B_b) and three constraints (I_r, I_g, I_b).

To make the matting problem tractable, most commercial matting approaches use a background with known, constant color. This is called *blue screen matting* (see the "Previous Work in Matting and Compositing" sidebar), even though green is preferable when shooting with digital cameras. Unfortunately, blue screen matting is an intrusive, expensive process unavailable to amateur users. The ideal matting approach works for scenes with arbitrary, unknown, and possibly dynamic backgrounds. This *natural image matting* typically requires substantial manual labor. However, two fully automatic natural image matting solutions have recently been developed. They acquire additional data during scene capture using special imaging devices.

In previous work, we (Joshi, Matusik, and Avidan) describe using a camera array to create a synthetic aperture image that focuses on the foreground object.[1] We estimate the variance of foreground and background pixels and compute a high-quality alpha matte at several frames per second (fps). We also show how to transfer the computed alpha matte to one of the cameras in the array using image-based rendering. Although camera arrays might be feasible for professional use, they can be impractical for certain applications.

McGuire et al.[2] developed a fully automated method that computes alpha mattes using three video cameras that share a common center of projection but vary in depth of field and focal plane. The additional defocus information constrains the original ill-posed matting problem. This *defocus matting* approach can compute high-quality mattes from natural images without user assistance. The special-purpose defocus camera uses three imagers that share an optical axis using beam splitters. Defocus matting is fully automatic, passive, and could be implemented compactly using methods similar to those used for constructing 3-charge-coupled-device (CCD) cameras. However, the approach also has several limitations:

- Each camera's resolution limits the alpha matte's resolution. All three must have the same resolution, so producing high-resolution results requires that you use three high-resolution cameras.
- Matte computation is time consuming, taking several minutes per frame. Consequently, it's somewhat impractical for long video clips.
- Defocus matting is impractical for photographers, who might be reluctant to place a beam splitter between their high-definition, high-quality cameras and the scene.

In this article, we address these limitations and extend defocus matting in several important ways.

Improvements on defocus matting

Our improved hardware setup allows more accurate alignment of cameras and beam splitters and more efficient light splitting, thus improving the input images' noise characteristics.

Our approach also addresses the speed of McGuire et al.'s optimization procedure. Their process requires many minutes of computation per frame and, when used for processing videos, runs independently on each video frame. We accelerate the method for video

 0272-1716/07/$25.00 © 2007 IEEE

Previous Work in Matting and Compositing

Matting and compositing are important tasks in television, film production, and publishing and have attracted researchers' attention since at least the late 1950s. Vlahos's initial work on matting and compositing led to the development of the UltiMatte system. Wallace[1] and Smith and Blinn[2] formalized digital compositing for film production mathematically, including the invention of two-background matte extraction.

Two-background matte extraction shows that by imaging a foreground object against two backgrounds with different intensity (or color), you can derive an expression for the α and foreground color. Zongker et al.[3] introduced environment matting, an extension of alpha matting to cover more complex light transport effects (such as specular reflection or refraction).

Estimating α and foreground color for scenes in which the background can't be controlled is often referred to as natural image matting. This task is much more difficult because the solution is typically underconstrained. To constrain the problem, most methods make assumptions about the frequency content of background, foreground, and alpha.[4-6] Furthermore, they rely on a user-specified trimap that segments the image into definitely foreground, definitely background, and unknown regions. These algorithms analyze unknown pixels using local color distributions. Natural image matting methods have been extended to video by propagating user-specified trimaps with optical flow.[7]

The Zcam (www.3dvsystems.com) and the work of Yasuda et al.[8] follow some of the same principles as our method in that they film additional data during capture to aid in matting. Both approaches also include a beam splitter so the extra camera shares the same optical axis as the main camera; however, the Zcam uses active illumination, which is undesirable for many applications, while the work of Yasuda et al. focuses on segmenting people using passive IR.

References

1. B. Wallace, "Automated Production Techniques in Cartoon Animation," master's thesis, Cornell Univ., 1982.
2. A.R. Smith and J.F. Blinn, "Blue Screen Matting," *Computer Graphics*, Proc. Siggraph, vol. 30, ACM Press, 1996, pp. 259–268.
3. D.E. Zongker et al., "Environment Matting and Compositing," *Proc. 26th Ann. Conf. Computer Graphics and Interactive Techniques*, ACM Press/Addison-Wesley, 1999, pp. 205-214.
4. M. Ruzon and C. Tomasi, "Alpha Estimation in Natural Images," P*roc. IEEE Computer Soc. Conf. Computer Vision and Pattern Recognition* (CVPR), IEEE CS Press, 2000, pp. 18-25.
5. Y.-Y. Chuang et al., "A Bayesian Approach to Digital Matting," *Proc. IEEE Computer Soc. Conf. Computer Vision and Pattern Recognition* (CVPR), IEEE CS Press, 2001, pp. 264-271.
6. J. Sun et al. "Poisson Matting," *ACM Trans. Graphics*, vol. 23, no. 3, 2004, pp. 315-321.
7. Y.-Y. Chuang et al., "Video Matting of Complex Scenes," *ACM Trans. Graphics*, vol. 21, no. 3, 2002, pp. 243-248.
8. K. Yasuda, T. Naemura, and H. Harashima, "Thermo-key: Human Region Segmentation from Video," *IEEE Computer Graphics and Applications*, vol. 24, no. 1, 2004, pp. 26-30.

sequences by using a nonparametric sampling technique that relies on the optimization result from previous frames.

Third, we consider a setup consisting of a defocus camera where the primary camera is high resolution and the other two cameras are low resolution. We show how to handle the resolution difference by adapting an image-restoration technique to perform super-resolution to create alpha mattes that match the primary camera's resolution.

This setup gives good results, but it requires placing beam splitters in front of all the cameras, which might be undesirable in some situations. Thus, we extend defocus matting to handle the case where an off-center camera doesn't share a center of projection with the low-resolution cameras. In such a setup, we attach the three defocus cameras next to the off-center camera and align them in software instead of hardware.

The A-Cam

The A-Cam consists of a compact collection of three video cameras, each with a different plane of focus and depth of field, that share a single center of projection (Figure 1). The A-Cam is an improved version of McGuire et al.'s camera for defocus matting.[2]

The three cameras are axis-aligned and image the scene through a tree of beam-splitters. We focus one camera, I_F, on the foreground, another camera, I_B, on the background, and a pinhole camera, I_P, on the entire scene. The foreground and background cameras have a large aperture with a narrow depth of field, whereas the pinhole camera has a small aperture with a large depth of field. Figures 2a to 2c give examples of these images.

The defocus in the foreground and background cameras occurs because the cone of rays from a point in the scene intersects the image plane at a disk. We describe the resulting point-spread function (PSF) or circle of confusion as:

$$r = \frac{f}{2\sigma\phi}\left|\frac{z_R\left(z_F - f\right)}{z_F\left(z_R - f\right) - 1}\right|$$

where the camera is focused at depth z_F, the point is at z_R, ϕ is the f-number, f is the focal length, and σ is the width of a pixel.[2] Depths are positive distances in front of the lens. To a good approximation, we can express images from the three cameras, I_P, I_B, and I_F, using the following equations:

$$I_P = \alpha F + (1 - \alpha)B \quad (1)$$

$$I_B = (\alpha F) \otimes disk(r_F) + (1 - \alpha \otimes (disk(r_F))B \quad (2)$$

$$I_F = \alpha F + (1 - \alpha)(B \otimes disk(r_B)) \quad (3)$$

Equations 1, 2, and 3 give us seven unknowns (α, F_r, F_g, F_b, B_r, B_g, B_b) and nine constraints (three per equation). The convolutions with $disk(r_F)$ and $disk(r_B)$ suggest that we can't solve for each pixel independently. Instead, we search for a global solution that minimizes this error function:

$$J = (I_P - \hat{I}_P)^2 + (I_B - \hat{I}_B)^2 + (I_F - \hat{I}_F)^2 \quad (4)$$

where I_P, I_F, and I_B are the observed images, and $\hat{I}_P$, $\hat{I}_F$, and $\hat{I}_B$ are the reconstructed images. We find the minimum using McGuire et al.'s sparse nonlinear optimization procedure.[2] This procedure finds a solution by iteratively updating initial values for the unknowns, taking steps along the error function's gradient toward a minimum value. Because the convolutions in Equations 2 and 3 are linear operations and Equation 1 is also linear, we can efficiently compute these equations' derivatives and then easily compute the error function's gradient. The optimization procedure also includes several regularization terms, such as terms enforcing spatial smoothness for the recovered α, F, and B.

We've improved the defocus camera hardware in two ways.

First, we constructed a rigid case to hold the cameras and beam splitters such that the optical centers are the same. This improves the cameras' overall alignment. In addition, the defocus camera is now self-contained, and the entire unit is much more portable and can be mounted on a tripod, as Figure 1 shows.

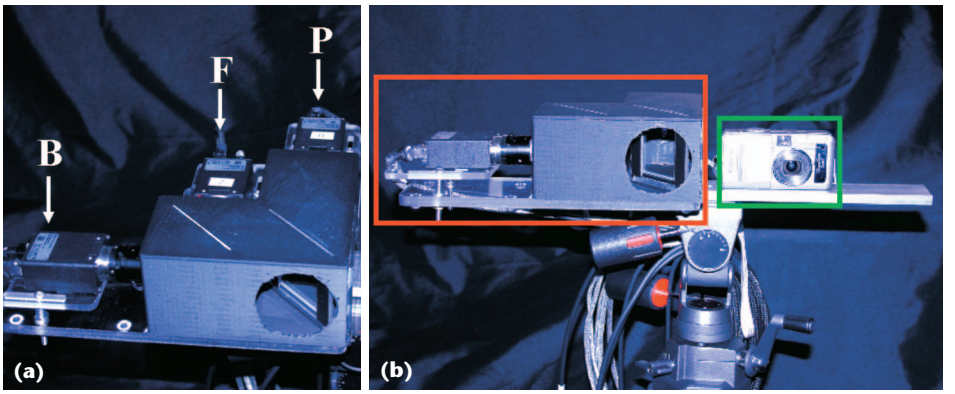
We also modified the beam splitters. The original design uses two 50/50 beam splitters. The pinhole camera receives half the light and the foreground- and background-focused cameras each receive one-quarter. However, because the camera apertures don't reflect this division of light, the pinhole doesn't receive enough light and the other cameras receive more than they need. As a result, the pinhole camera's images suffer from noise because of lack of light, and the other two cameras tend to be overexposed as they receive too much light. We modified the design to use a 90/10 beam splitter to send 90 percent of the light to the pinhole with the remaining 10 percent split equally between the other two cameras. The ratio of the apertures now matches these splitting ratios so that each sensor receives the same amount of light.

Even though the three cameras are better aligned and matched in light exposure, we still need to colorimetrically and geometrically align them. We can calibrate color through simple histogram matching to match the A-Cam's foreground- and background-focused cameras to the pinhole camera. We compute the histogram remapping on smoothed version images to limit the effect of noise in the calibration process. The cameras' geometric alignment is straightforward. Because the cameras are coaxial, we need to correct only for each camera sensor's possible rotation and translation, which we do using one homography per camera. We do this by placing a grid in the scene, detecting feature points on the grid, and computing a homography for the foreground and background cameras to align them to the pinhole camera.

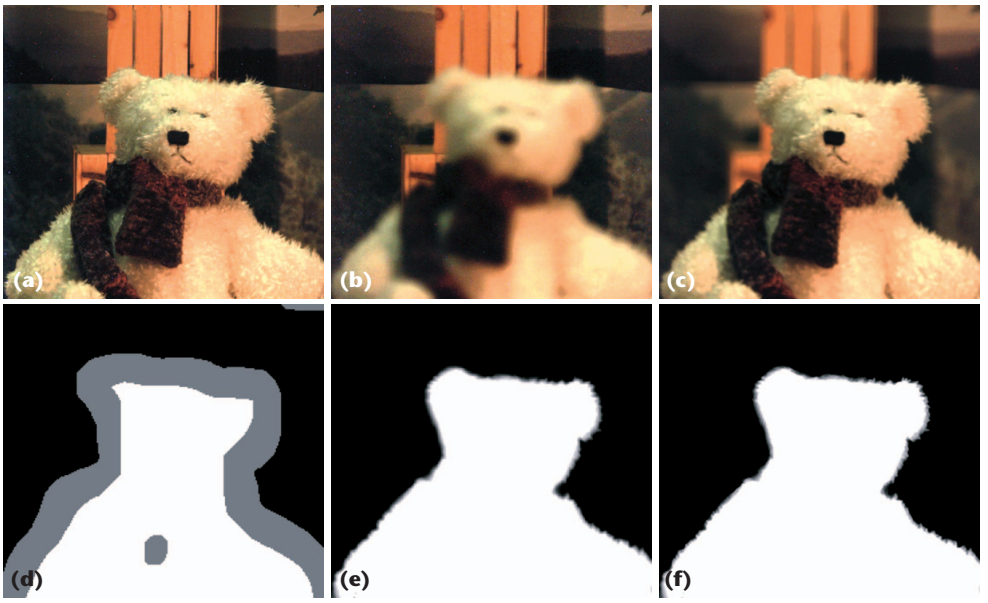
Nonparametric video matting

The original defocus matting work makes no distinction between still image matting and video matting, and creates mattes for video by processing each frame independently. With many minutes of CPU time per frame and video rates of 30 frames per second (fps), processing video clips quickly becomes impractical. To address this issue, we accelerate the video-matting process using a nonparametric model built on-the-fly from our optimization results. For videos with strong temporal consistency, where "past predicts future" is a reliable assumption, this process provides a significant speedup. Our method starts with a training phase in which we individually process the first k frames of a video sequence using our full optimization procedure (we use $k = 5$). Then, during the processing phase, we use the results of these k frames to predict the result for a subsequent frame i using a method similar to image analogies.[3]

Specifically, we create an input feature vector, ψ, for each pixel. This vector contains the corresponding pixel values from I_P, I_B, and I_F and the $disk(r_B)$ and $disk(r_F)$ size neighborhoods of pixels from I_B and I_F, respectively. (The sizes of $disk(r_B)$ and $disk(r_F)$ are a function of the camera lens settings and the foreground and background depths. These disks model the lens' PSF at the foreground and background depths[2]—as a rule of thumb, they're generally on the order of two to 10 pixels wide for common defocus matting setups.) We don't use a neighborhood from I_P because surrounding pixels don't affect the pinhole image constraint. For the neighborhood of pixels, instead of using pixel values directly, we use a rotationally invariant transform of the pixel data.[4] This transform collapses the feature space significantly, allowing our model to generalize to a larger range of input data with many fewer samples. We populate a



1 The A-Cam. (a) The A-Cam is a collection of three video cameras that share a single center of projection by using a tree of beam splitters. The black case houses the cameras and beam splitters. (b) The A-Cam (red rectangle) is attached to a standard consumer-level camera (green rectangle) for off-center matting.



2 Results from the new A-Cam: (a) pinhole image of the A-Cam, (b) background-focused image, (c) foreground-focused image, (d) automatically computed trimap, (e) the alpha matte at the A-Cam resolution, (f) the super-resolution alpha matte, and (g) super-resolution alpha premultiplied foreground image composited with a new background.

lookup table with pairs $(\psi(p), [\alpha(p), B(p), F(p)])$ for each pixel, p, for each of the training frames 1 through k. We then predict values for a subsequent frame i using a process similar to that used by Hertzmann et al.[3] In scanline order, for pixel q in frame i, we

1. Construct input feature vector $\psi(q)$.
2. Find five nearest-neighbor pixels (the five pixels closest to q in terms of distance, $d_j = \|(\psi(q) - \psi(p_j))^2\|$).
3. From these nearest-neighbor pixels, select the smoothest pixel, p_{smooth}—that is, the pixel whose sum of squared difference between its corresponding α and the α values for the already computed neighboring pixels in the alpha matte is minimal.
4. Set $d_{smooth} = \|(\psi(q) - \psi(p_{smooth}))^2\|$.
5. From the nearest-neighbor pixels, find the nearest pixel, p_n—that is, the pixel whose distance, $d_n = \|(\psi(q) - \psi(p_n))^2\|$ is smallest.
6. If $d_{smooth} < \gamma d_n$,
   $[\alpha(q), B(q), F(q)] \leftarrow [\alpha(p_{smooth}), B(p_{smooth}); F(p_{smooth})]$
   else
   $[\alpha(q), B(q), F(q)] \leftarrow [\alpha(p_n), B(p_n), F(p_n)]$.

Intuitively, we construct the result for frame i pixel-by-pixel, where we set the result equal to the corresponding result for either the closest match, or equal to the result for a close match whose corresponding alpha is similar to those already computed in the image. The γ parameter lets us adjust the result's smoothness. We've empirically found that $\gamma = 2$ works well. For efficiency, instead of finding exact nearest neighbors, we use an approximate nearest-neighbor algorithm with a small tolerance. Once we've predicted the entire frame, we evaluate the error function in Equation 4 with this newly created result. For any pixels whose error is greater than some ε tolerance, we refine the estimate by running the optimization. We then add these newly optimized pixels back into our lookup table so we can continually update our nonparametric model.

With this approach, the number of pixels to be optimized depends on the video's nature. If the video exhibits slow transitions, the nonparametric method should suffice. In case of abrupt changes, say due to camera motion, we'll have to evaluate more pixels using the time-consuming optimization method. In this latter case, after we add the frame to the model, the system is bootstrapped and can return to computing subsequent frames primary from the model. By varying the ε tolerance, we can trade accuracy for speed, where a value of 0 would perform full optimization for each frame and a value of infinity would compute results using only the nonparametric model from the first k frames. An in-between value would be a mixture of the two, which we refer to as *mixed optimization*.

Alpha super-resolution

The resolution of the three cameras currently limits the resolution of the mattes pulled with defocus matting, so high-resolution results require three high-resolution cameras. Although using high-resolution cameras might seem reasonable, increases in computation time, memory and bandwidth usage, and camera cost can quickly make this option impractical. Using full optimization, the computation time for one 320×240 frame is about five to 10 minutes, which would put a 1-megapixel video at well over an hour per frame. Our nonparametric method will accelerate this process. However, when building a nonparametric model from high-resolution images, the model's size can become prohibitively large. Another concern is the limited bandwidth of cameras and their capture devices. Generally, cameras trade frame rate for increased resolution—a trade-off that's nice to avoid if possible.

To combat these issues, we propose a method that computes high-resolution alpha mattes by replacing the A-Cam's pinhole camera with a high-resolution camera, leaving the other two cameras the same. We downsample the high-resolution pinhole image, I_P^H, to get I_P to match the resolution of I_F and I_B. We then use I_P, I_F, and I_B as before to compute mattes at 320×240 resolution

using our nonparametrically accelerated optimization method. We then use a super-resolution technique as a postprocess to upgrade the alpha matte to the primary camera's native resolution.

Specifically, given α and F, as computed using the method described in the previous sections, and given a blur function f, which models the resolution gap, where $f(I_P^H)=I_P$, we want to find α^H and F^H such that $f(\alpha^H) = \alpha$, and $f(F^H) = F$. This is an underconstrained problem—there are many high-resolution α and F images that could blur to the low-resolution versions. Thus, we solve this problem using an edge-preserving regularization term that's based on anisotropic diffusion. Researchers have used similar techniques for image restoration, and the following derivation mirrors those used in these techniques (see Kornprobst et al.[5] for a more detailed explanation of this derivation).

Specifically, we want to optimize the following regularized error function: $J = (\boldsymbol{A}\alpha^H - \alpha)^2 + \lambda \int\int \phi(|\nabla(\alpha^H)|)dxdy$, where $\boldsymbol{A}$ is a matrix that encodes the blur function f, α^H and α are given in vector form, $\nabla(x)$ is the gradient of x, and $\phi(x)$ is an edge-preserving function, $\phi(x)=1/\sqrt{1+x^2}$. This regularization term lets us perform smoothing when gradient values are low; for high gradients, we can apply smoothing only along an edge and not across it.

We can minimize this partial differential equation (PDE) using a variational method. Applying the Euler-Lagrange equation shows that the error function is minimized when

$$\boldsymbol{A}^T\boldsymbol{A}\alpha^H + div\left(\lambda\phi'\left(\left|\nabla\left(\alpha^H\right)\right|\right)\frac{\nabla\left(\alpha^H\right)}{\left|\nabla\left(\alpha^H\right)\right|}\right) = \boldsymbol{A}^T\alpha$$

Imposing Neumann boundary conditions, which specify that values of the solution's gradient in the direction normal to the boundary are zero, lets us convert the divergence term to a Laplacian matrix, $\boldsymbol{B}$, that is spatially varying as a function of the gradients of α^H:

$$\boldsymbol{A}^T\boldsymbol{A}\alpha^H + \lambda\boldsymbol{B}\left(\phi'\left(\left|\nabla\left(\alpha^H\right)\right|\right)\right)\alpha^H = \boldsymbol{A}^T\alpha \quad (5)$$

We then assume $\nabla(\alpha^H)=\nabla(I_P^H)$. This enforces that edges in the original image are preserved in the sharpened alpha matte. This assumption is valid for depth edges, which dominate in the unknown region, because gradients due to depth discontinuities correspond to edges in the alpha matte. Although this assumption is invalid for gradients in I_P that don't appear in the alpha matte (for example, because of texture), this shouldn't cause any errors as our regularization term is edge-preserving but shouldn't introduce edges when none are present. We also note that our super-resolution method doesn't rely on any new information relative to our original matting construct. The gradients we use for super-resolution are the same needed for defocus matting to compute an alpha matte. We perform super-resolution only on alpha in the unknown region. Equation 5 now becomes:

$$\boldsymbol{A}^T\boldsymbol{A} + \lambda\boldsymbol{B}\left(\phi'\left(\left|\nabla\left(I_P^H\right)\right|\right)\right)\alpha^H = \boldsymbol{A}^T\alpha \quad (6)$$

We can then solve for α^H in one step using a sparse linear least-squares solver. Observe that our regularization depends on the edges of I_P^H and not the edges of α^H. This is where our method deviates from similar image restoration methods. In those methods, the regularization matrix $\boldsymbol{B}$ is a function of the edges of the image being solved for, which in this case is α^H. This gives rise to a nonlinear PDE that must be solved using an iterative approach. Our solution is linear, as $\boldsymbol{B}$ is a function of a known value I_P^H, and thus can be solved in one step using linear least-squares. We use the same process to obtain F^H, the high-resolution foreground. Performing super-resolution for the entire image at once can be problematic for large images—even though the matrices in Equation 6 are sparse, they can be still be quite large. However, because the blurring that we're trying to remove is a relatively local function, we can perform super-resolution on a block-by-block basis rather efficiently even for large images. We've empirically determined that 60 pixel subblocks that overlap by seven pixels work well for our scenes.

Off-center alpha matting

The A-Cam uses a collection of beam splitters placed between the cameras and the scene. Although this might be acceptable in some cases, it might not be desirable for high-end photography or for filming movies where the user might not want to let any optical device, such as a beam splitter, come between the camera and the scene. We propose a hybrid camera approach[6] in which we use the A-Cam as an accessory to an external off-center camera, and the cameras don't share the same center of projection, as Figure 1b shows. In this situation, we can't use a matte from the A-Cam directly as a matte for the off-center camera. Instead, we compute the alpha matte directly for the off-center camera and use the data from the A-Cam to regularize the solution.

The off-center camera and the A-Cam observe the same scene, albeit with potentially different camera settings. Still, we can assume that they're both focused on the foreground object and that we can colorimetrically and geometrically align the cameras. Given this information, how can we use the A-Cam data to regularize the alpha matte we compute for the off-center camera?

Regularized alpha matting

The external off-center camera I_{OFF} gives us one equation with seven unknowns and three constraints:

$$I_{OFF} = \alpha_{OFF}F_{OFF} + (1 - \alpha_{OFF})B_{OFF}.$$

If the A-Cam and primary camera are colormetrically and geometrically aligned and the foreground object is in focus in both cameras, we can assume that $\alpha_{OFF} = \alpha$ and $F_{OFF} = F$. This reduces the number of unknowns by four. This gives us a set of four equations with 10 unknowns and 12 constraints:

- $I_P = \alpha F + (1 - \alpha)B$
- $I_B = (\alpha F) \otimes disk(r_F) + (1 - \alpha \otimes disk(r_F))B$
- $I_F = \alpha F + (1 - \alpha)(B \otimes disk(r_b))$
- $I_{OFF} = \alpha F + (1 - \alpha)B_{OFF}$

Just as before, we can't solve this problem on a pixel-by-pixel basis because we convolve the images I_B and I_F with a finite-size disk. Instead, we solve this system of equations using regularized optimization. Of all the solutions that satisfy the primary camera's matting equation, we choose the one that also satisfies the alpha matting equation of the A-Cam. Specifically, let $J = (I_P - \hat{I}_P)^2 + (I_B - \hat{I}_B)^2 + (I_F - \hat{I}_F)^2$ be the error term for the A-Cam, where $\hat{I}_P$, $\hat{I}_B$, and $\hat{I}_F$ are the pinhole, background, and foreground images recovered by the optimization, respectively, and let $J_{OFF} = (I_{OFF} - \hat{I}_{OFF})^2$ be the error term for the high-definition camera, where $\hat{I}_{OFF}$ is the recovered off-center camera image. We solve

$$\arg\min_{\alpha,F,B,B_{OFF}} \left\{ J_{OFF} + \lambda J \right\} \quad 0 \le \lambda \le 1, \tag{7}$$

where λ is the regularization parameter controlled by the user. We've empirically found that setting λ between 0.3 and 0.5 works well.

Although we assume that the off-center camera and the A-Cam can share α and F, this isn't the case with the background. This is because the background B and the background B_{OFF} are related by a homography if they're planar, or by pixel-wise correspondence if they aren't. In addition, they might be related by a convolution (if they're defocused differently because of different depths of field of the lenses in the A-Cam and off-center camera). We avoid estimating these relationships by directly solving both for B and B_{OFF}. We adapt McGuire et al.'s[2] minimization method to minimize the function in Equation 7 by adding the additional unknown, B_{OFF}, and error term, J_{OFF}, to the error function and gradient computation and by incorporating the λ regularization factor; the minimization method is otherwise unchanged. We've similarly adapted our nonparametric acceleration method to incorporate this additional data.

Off-center calibration

Colorimetrically and geometrically calibrating the primary camera and the A-Cam takes several steps.

We achieve color calibration through simple histogram matching, as described earlier. For geometric calibration, we assume that we know the foreground object's depth and place a grid at that location ahead of time. (We address this assumption later.) We use corresponding points from this grid to compute a homography between the primary camera and the A-Cam.

Because this initial homography aligns features at only one depth and a typical foreground object spans a range of depths, we must perform additional alignment between the off-center camera and the A-Cam. One approach is to use pixel-wise optical flow. However, because optical flow tends to fail along object borders, we instead align the foreground by computing homographies on a block-by-block basis. Because we solve for different backgrounds for the A-Cam and off-center camera, we need to align only the foreground pixels. We do this by computing homographies to best align only the foreground pixels, as determined by our trimap. We align the background pixels in a block with the same transform as the foreground pixels. This block-based alignment preserves the structure of the foreground object in the border region by providing a rigid transform across the boundary. Our method works well as long as the object boundary lies within the block. Small seams can exist if the object boundary lies across a block boundary, so we size our blocks to approximately match the unknown region's size to minimize this situation's occurrence. We've empirically determined that 60×60-pixel blocks work well for our scenes.

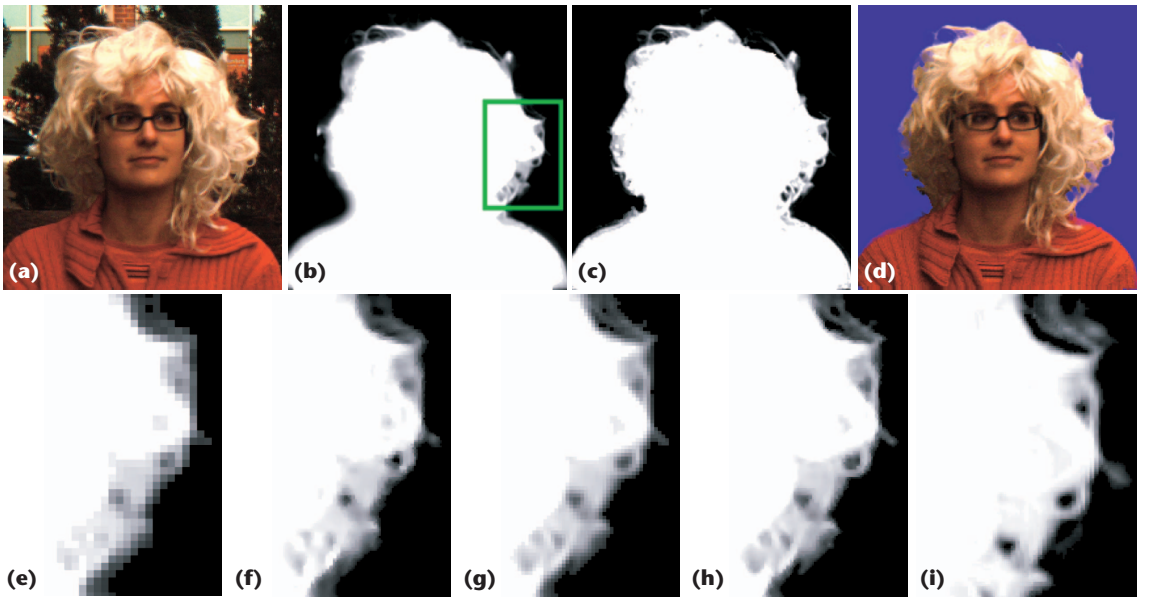
Computing these homographies requires estimating eight parameters per homography. Fortunately, we can rely on the fact that the space of all homographies between a pair of images is a 4D space.[7] Intuitively, this is true because a plane that defines the homography has four degrees of freedom. We therefore image a planar grid in 10 positions and compute the homography for each position. For these 10 homographies, we use principal components analysis (PCA) to find the four basis homographies spanning the space of all homographies between the two cameras. Then, for every pixel block, we find a homography that minimizes the sum-of-squared differences between the aligned pixel blocks. We need to solve for only four unknowns—the four coefficients of the basis homographies—which we do with an iterative Levenberg-Marquardt solver.

We return now to the assumption that we know the foreground object's depth ahead of time. This need not be the case. After we recover the four basis homography matrices between the off-center camera and the A-Cam, we can represent any homography, including the one caused by the plane going through the foreground object, as a linear combination of the basis matrices. We can estimate the foreground object's depth by estimating the plane that minimizes the image difference between the off-center camera and the A-Cam. The plane estimation involves estimating only four parameters (the coefficients of the basis homographies) and not the eight parameters of a homography. Note that even if the off-center camera moves, as long as the A-Cam remains rigidly attached to it, these basis homographies can continue to be used for alignment.

Experiments

We conducted several experiments to validate our methods. We first show results using our nonparametric acceleration method. We then show results that illustrate our super-resolution method and follow those with results from off-center matting. In addition to visual results, we provide an analysis of the running times of our extensions.

Figure 3 compares results from our nonparametric method with a result from the original defocus matting paper. We captured this data set, of a person with blond hair, using three 640×480 (Bayer pattern) video cameras. As in McGuire et al.,[2] we demosaiced the data to 320×240 resolution for computing mattes. For this sequence, we used the first five frames for training the model, and here show the result of the 22nd frame. For a comparison of the longer video sequence, see our



3 Nonparametrically accelerated optimization and super-resolution: (a) pinhole image of the A-Cam, (b) the alpha matte at 320 × 240 using full optimization (that is, McGuire et al.'s method[2]), (c) the alpha matte at 320 × 240 using nonparametrically accelerated optimization, (d) composite of 320 × 240 nonparametrically accelerated optimization result. Alpha mattes for the region shown in the green box computed at (e) 160 × 120 using full optimization, (f) 160 × 120 upgraded to 640 × 480, (g) 320 × 240 using full optimization, (h) 320 × 240 upgraded to 640 × 480, and (i) 640 × 480 using full optimization.

video results at http://graphics.ucsd.edu/papers/exploring_defocus_matting. Our mixed optimization approach produces a result with comparable quality to the original method in little less than half the time (3.5 minutes per frame compared to 8.5 minutes with the original optimization process).

Figure 3 also shows our super-resolution approach applied to this same data set. For the first result, we push our super-resolution method and downsample the A-Cam data to 160 × 120 and then compute the matte at this low resolution. We then upgrade these mattes to 640 × 480 resolution. In the second result, we upgrade a 320 × 240 result to 640 × 480. We show the original low-resolution versions and a result computed natively at 640 × 480 for comparison. Both super-resolution results look sharper and more detailed than the original optimization output, and the upgraded 160 × 120 result (Figure 3f) and the upgraded 320 × 240 result (Figure 3h) are on par with the result computed natively at 640 × 480 (Figure 3i).

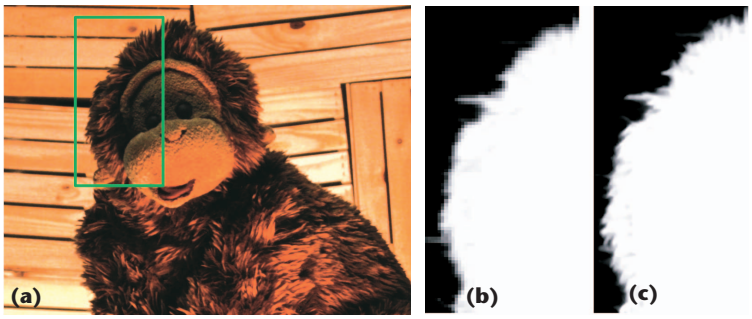
Although the results with this original data set are reasonable, some errors are still present. These errors are because of poor geometric alignment and saturation of the foreground- and background-focused cameras. Our nonparametric method doesn't introduce these errors—similar artifacts are present in McGuire et al.'s results.[2] It's the presence of these errors that lead us to change the beam splitters and build an accurately designed case for holding the camera and beam splitter assembly.

Figure 2 shows a result for a new data set acquired with our improved A-Cam setup. We show the A-Cam images, resulting matte and composite, and super-resolution from 320 × 240 to 640 × 480. The images show a stuffed bear in front of some wood boxes and artificial trees. The improved alignment from our new A-Cam hardware setup lets us pull a much cleaner and accurate matte, even though the background in this scene has complex structure and high-frequency content.

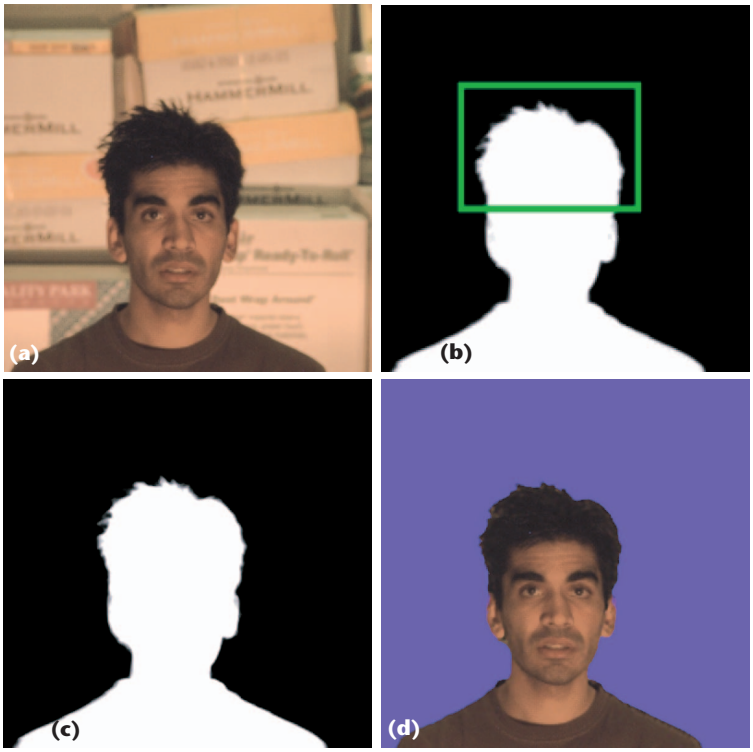
Our final set of results illustrates off-center matting. We attached the A-Cam to an external off-center camera, as Figure 1b shows. We captured images of a grid in several positions in the filming area and used them to compute the basis homographies between the off-center camera and the A-Cam, as we described earlier. Figure 4 shows an off-center matting result for a stuffed

4 Off-center matting: (a) original off-center camera image, (b) the alpha matte at the A-Cam resolution, (c) the super-resolution alpha matte, and (d) super-resolution alpha premultiplied foreground image composited with a blue background.

gorilla in front of some wood boxes. In this data set, there is a relatively large (5×) resolution difference between the off-center camera and the A-Cam because the off-center camera is a two-megapixel consumer camera. This object spans a relatively large depth range, thus our per-block geometric alignment was essential. We successfully pulled the alpha matte and, as Figure 5 shows, recovered a significant amount of detail using our super-resolution method.



5 Gorilla with alpha super-resolution: (a) original image, (b) upsampling using bicubic interpolation, and (c) super-resolution using our edge-preserving method.



6 Off-center video matting with nonparametric acceleration and super-resolution: (a) original off-center camera image, (b) the alpha matte computed at 320 × 240 resolution using our mixed optimization method, (c) the 640 × 480 super-resolution alpha matte, and (d) super-resolution alpha-multiplied foreground image composited with a blue background.

7 Nonparametric video matting of the area in the green box in Figure 6b: (a) alpha matte computed using full optimization (5.5-minute computation time), (b) alpha matte computed using our mixed optimization method (1.5 minutes), and (c) alpha matte computed using only the nonparametric model (30 seconds).

Figure 6 shows a single frame of a video sequence of an off-center matting result for a person with black hair in an office supply room. We captured this data set using a 640 × 480 video camera mounted next to the A-Cam. We computed mattes at 320 × 240 resolution and then upgraded to 640 × 480 using our super-resolution method. Figure 7 shows a side-by-side comparison of a result using full optimization, mixed optimization, and the nonparametric model alone. We computed the fully optimized result in six minutes, the mixed optimization result in 1.5 minutes, and the nonparametric result in 30 seconds. We used the first five frames of a 60-frame sequence for training. Video results for this data set are available at http://graphics.ucsd.edu/papers/exploring_defocus_matting.

Table 1 compares running times of our two video data sets processed with and without our extensions. This data demonstrates that our pure nonparametric approach provides a reasonably significant speedup over the full optimization approach. A mixed optimization still produces a moderate speedup. The exact value to set for ε for a mixed approach depends on the desired balance of speed versus quality and also on the overall data quality—that is, some inherent lower bound on the error exists, given noise in the data, calibration errors, and so on. Thus, if we set ε below this lower bound, it's equivalent to specifying full optimization. The units of ε are mean-squared-difference of image intensity of the observed and predicted images. For the data sets shown here, we set ε equal to 0.003, which corresponds to around a 5-percent error in image intensity.

Although we initially envisioned the super-resolution method as a way to match mixed-resolution cameras, our method can also serve as an acceleration technique when all cameras have the same resolution. First, we downsample all videos to a low resolution, then compute an alpha matte, and then upgrade back to the original resolution. Although this might not provide exactly the same quality as optimizing at the A-Cam's native resolution, for some applications the speed increase might justify the slight loss of quality.

Discussion and future work

Our novel defocus matting extensions increase speed and resolution and can be used with traditional consumer cameras; however, some limitations remain. Our nonparametric method's two primary limitations are typical of many learning methods.

The first limitation is that errors present in the training set will propagate through to prediction. Currently, we pick our training set by simply using the first five frames in a sequence. By using this scheme, we let the quality of the optimization for these initial frames dictate our training data's quality, but we don't have any way to ensure that these first five frames are actually good examples. Furthermore, we pick entire frames for training. One straightforward modification would be to

Table 1. Per-frame computation times.

Data set	Method	Computation of alpha (minutes)	Super-resolution (minutes) 320 × 240	Super-resolution (minutes) 640 × 480	Total time for 640 × 480 (minutes)
Blond hair	160 × 120 (full optimization)	1	0.5	4	5
	320 × 240 (full optimization)	8.5	N/A	2	10.5
	320 × 240 (mixed optimization)	3.5	N/A	2	5.5
	320 × 240 (non-parametric only)	0.5	N/A	2	2.5
	640 × 480 (full optimization)	30	N/A	N/A	30
Black hair	320 × 240 (full optimization)	5.5	N/A	1.5	7
	320 × 240 (mixed optimization)	1.5	N/A	1.5	3
	320 × 240 (non-parametric only)	0.5	N/A	1.5	2

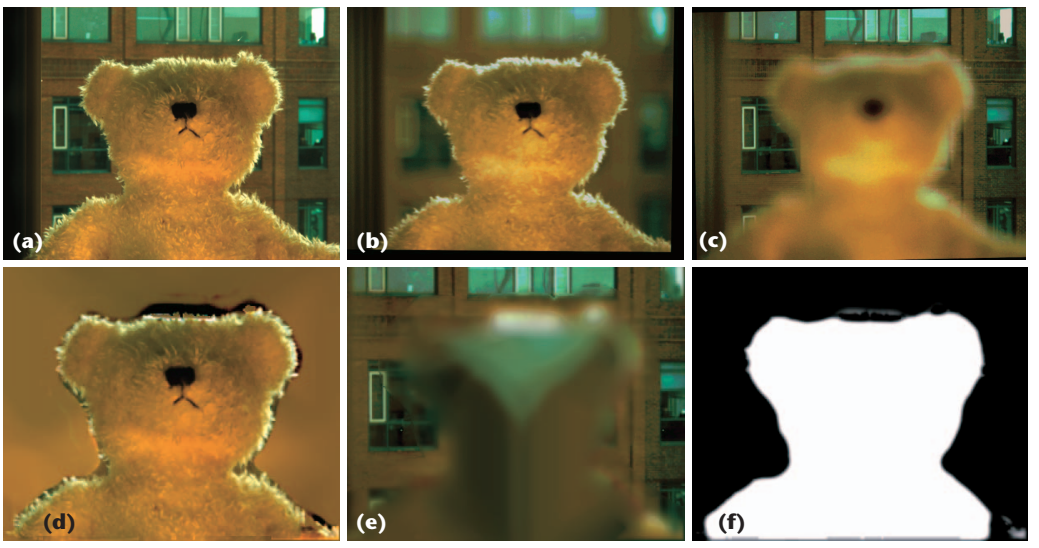
include training data on a per-pixel basis only where the absolute error from the optimization is low. A further modification would be to train a classifier to automatically label "good" and "bad" results.

The nonparametric method's second limitation is the memory used by the model, which grows as we update it with new frames. Our current Matlab implementation has an upper limit of about 60 added frames. This number varies, however, with the ε threshold and the extent of the video's temporal coherence—that is, the more the frames are alike, the smaller the model can be. One solution to this problem is to compress the model. Recently, researchers have used canonical correlation analysis to reduce nonparametric model size. We're interested in exploring this as an alternative to the traditional PCA-based techniques.

A limitation of our super-resolution method is that it can enhance noise in the input low-resolution alpha matte. Occasionally a small, relatively low, errant alpha value becomes more prominent after super-resolution. In practice, a small amount of postprocessing[2] will often remove these errors before super-resolution. Decoupling our super-resolution method from the matte-optimization procedure might also introduce errors relative to computing the matte at high resolution with an integrated super-resolution method. Our motivation was speed, and we considered this additional source of error acceptable. However, one future area of work is to merge these processes without the performance hit of full optimization at high resolution. Another approach for improving our method is to combine it with a nonparametric approach similar in principle to our acceleration method. This seems like a promising direction because example-based methods have been used successfully for super-resolution of traditional images.

Our off-center method is currently limited to work with scenes containing predominantly planar foreground and background objects. Although we can handle some amount of depth variation with our alignment, this is, in general, a limitation. One way to address this is by using more sophisticated alignment techniques that leverage the fact that the off-center primary camera and the pinhole camera essentially provide a stereo pair. Also, although the resolution gap is the most obvious distinction between the consumer-level camera and the A-Cam, other camera setup gaps might be considered, such as a gap in the dynamic range of the two cameras.

While we've addressed several limitations of defocus matting, other inherent limitations remain. The primary limitation is the presence of local minima in the error-function minimized by nonlinear optimization. The final results can be sensitive to the initial guess and the choice of the regularization parameters involved in the process. Thus, the process can often terminate in a local minimum. Figure 8 shows an example of this type of failure. In this result, the optimization terminated with an incor-



8 A failure case: (a) pinhole image, (b) foreground focused image, (c) background focused image, (d) the foreground color recovered by the optimization, (e) the recovered background, and (f) the computed alpha matte. Because of the large amount of defocus and lack of significant gradients in the background, there is foreground pollution in the background and vice versa, resulting in an overly smooth outline for the matte and incorrect values above the bear's head and on its left shoulder.

rect low-frequency blur of the foreground color polluting the background color. Because of the large amount of defocus and the relatively smooth background, this error really only violates the pinhole constraint, but is low error given the remaining constraints.

Our nonparametric acceleration method provides a better initial guess than the original process of inpainting the unknown areas and estimating the initial alpha, but it doesn't remove the need for some form of initial guess for generating the training frames. Improving the initial guess (for example, by using single-frame methods such as Poisson matting) could be a fruitful direction for future work. ■

Acknowledgments

We thank John Barnwell for his help building the A-Cam and Amit Agrawal for his insight on edge-preserving regularization. We also thank the anonymous reviewers for their comments. We completed the majority of this work while Neel Joshi was an intern at Mitsubishi Electric Research Labs; he was additionally funded by US National Science Foundation grant DGE-0333451.

References

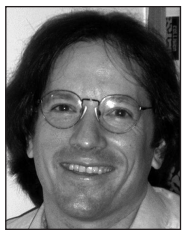
1. N. Joshi, W. Matusik, and S. Avidan, "Natural Video Matting Using Camera Arrays," *ACM Trans. Graphics*, vol. 25, no. 3, July 2006, pp. 779-786.
2. M. McGuire et al., "Defocus Video Matting," *ACM Trans. Graphics*, vol. 24, no. 3, Aug. 2005, pp. 567-576.
3. A. Hertzmann et al., "Image Analogies," *Proc. Computer Graphics* (Siggraph), ACM Press, 2001, pp. 327-340.
4. B.S. Reddy and B.N. Chatterji, "An FFT-based Technique for Translation, Rotation, and Scale-Invariant Image Registration," *IEEE Trans. Image Processing*, vol. 5, no. 8, 1996, pp. 1266-1271.
5. P. Kornprobst, R. Deriche, and G. Aubert, "Nonlinear Operators in Image Restoration," *Proc. IEEE Computer Soc. Conf. Computer Vision and Pattern Recognition* (CVPR), IEEE CS Press, 1997, pp. 325-330.
6. H.S. Sawhney et al., "Hybrid Stereo Camera: An IBR Approach for Synthesis of Very High Resolution Stereoscopic Image Sequences," *Siggraph: Proc. 28th Ann. Conf. Computer Graphics and Interactive Techniques*, ACM Press, 2001 pp. 451-460.
7. A. Shashua and S. Avidan, "The Rank-4 Constraint in Multiple View Geometry," *Proc. European Conf. Computer Vision* (ECCV), Springer-Verlag, 1996, pp. 196-206.

Neel Joshi *is a PhD student in the Computer Science and Engineering Department at the University of California, San Diego. His research interests include computer graphics and computer vision, specifically, data-driven graphics, image-based rendering, inverse rendering, appearance modeling, and computational photography and video. Joshi has an MS in computer science from Stanford University. Contact him at njoshi@cs.ucsd.edu.*

Wojciech Matusik *is a research scientist at Mitsubishi Electric Research Labs. His primary research lies in computer graphics, data-driven modeling, computational photography, and new display technologies. Matusik has a PhD in electrical engineering and computer science from the Massachusetts Institute of Technology. Contact him at matusik@merl.com.*

Shai Avidan *is a research scientist at Mitsubishi Electric Research Labs. His research interests are in the computer vision field with occasional detours into computer graphics and machine learning. Avidan has a PhD in computer science from the Hebrew University, Jerusalem. Contact him at avidan@merl.com.*

Hanspeter Pfister *is a senior research scientist at Mitsubishi Electric Research Labs. His research is at the intersection of visualization, computer graphics, and computer vision. Pfister has a PhD in computer science from the State University of New York at Stony Brook. He is chair of the IEEE Visualization and Graphics Technical Committee (VGTC). Contact him at pfister@merl.com.*



William T. Freeman *is a professor of electrical engineering and computer science at the Massachusetts Institute of Technology's Artificial Intelligence Lab. His research interests include machine learning applied to computer vision and computer graphics, Bayesian models of visual perception, and interactive applications of computer vision. Freeman has a PhD from the Massachusetts Institute of Technology, where he studied computer vision. Contact him at billf@mit.edu.*