Environment Matting and Compositing

Douglas E. Zongker[1]   *Dawn M. Werner*[1]   *Brian Curless*[1]   *David H. Salesin*[1,2]

[1]University of Washington   [2]Microsoft Research

Abstract

This paper introduces a new process, *environment matting*, which captures not just a foreground object and its traditional opacity matte from a real-world scene, but also a description of how that object refracts and reflects light, which we call an *environment matte*. The foreground object can then be placed in a new environment, using *environment compositing*, where it will refract and reflect light from that scene. Objects captured in this way exhibit not only specular but glossy and translucent effects, as well as selective attenuation and scattering of light according to wavelength. Moreover, the environment compositing process, which can be performed largely with texture mapping operations, is fast enough to run at interactive speeds on a desktop PC. We compare our results to photos of the same objects in real scenes. Applications of this work include the relighting of objects for virtual and augmented reality, more realistic 3D clip art, and interactive lighting design.

CR Categories: I.2.10 [**Artificial Intelligence**]: Vision and Scene Understanding – modeling and recovery of physical attributes; I.3.3 [**Computer Graphics**]: Picture/Image Generation – display algorithms; I.3.7 [**Computer Graphics**]: Three-Dimensional Graphics and Realism – color, shading, shadowing, and texture

Additional Keywords: environment matte, refraction, reflection, image-based rendering, environment map, augmented reality, interactive lighting design, clip art, alpha channel, blue-screen matting, blue spill, colored transparency

1 Introduction

Matting and compositing are fundamental operations in graphics. In the *matting* process, a foreground element of arbitrary shape is extracted from a background image. The *matte* extracted by this process describes the opacity of the foreground element at every point. In the *compositing* process, the foreground element is placed over a new background image, using the matte to hold out those parts of the new background that the foreground element obscures. Matting and compositing were originally developed for film and video production [11], where they are often used, for instance, to place the image of an actor, photographed in a studio in front of a controlled backdrop, into another environment. In 1984, Porter and Duff [20] introduced the digital analog of the matte — the *alpha channel* — and showed how synthetic images with alpha could be useful in creating complex digital images. In 1996, Smith and Blinn [25] described a mathematical framework for extracting alpha channels from real-world scenes, given either a pair of background images or certain assumptions about the colors in the foreground element.

Although matting and compositing have proven tremendously useful in film, video, and computer graphics production, they nevertheless fail to simulate two key effects that are essential for realism:

Figure 1 A water goblet, digitally composited onto background images, preserving the effects of refraction.

- *refraction*, exhibited by transparent objects; and
- *reflection*, exhibited by shiny objects and objects seen at a grazing angle.

Moreover, in the case of translucent or glossy materials, refraction and reflection are further coupled with various light scattering effects. For colored materials, refraction and reflection may also exhibit selective attenuation and scattering of light, according to wavelength.

In this paper, we generalize the traditional matting and compositing processes to incorporate all of these effects. The resulting processes, which we call *environment matting and compositing*, are capable of capturing not just a foreground object and its traditional matte, but a description of how that object refracts and reflects light, which we call an *environment matte*. The foreground object can then be placed in a new environment, where it will refract and reflect light from that scene. While the environment mattes that we capture provide only an approximation to the way an object truly refracts and reflects light, we have found them to produce convincing results in real scenes. Furthermore, our environment matte representation is such that environment compositing can be performed largely with simple texture mapping. Thus, for environments of reasonable size, this process can be performed at interactive speeds on inexpensive PCs. Figure 1 shows a water goblet captured in this way and composited onto two novel backgrounds.

1.1 Related work

The work described in this paper brings together strands of research from many different areas in computer graphics.

Researchers have augmented pixels in a number of ways to enable more flexible image synthesis. Porter and Duff's alpha at each pixel allows for images to be acquired or rendered in layers and then combined [20]. Layers may be modified independently and then simply recombined. As noted above, their compositing method does not account for reflections, refractions, or colored transparency.

Gershbein [12] augments pixels with depth, surface normal, and shading parameters and then performs per-pixel shading calculations for light sources that can be moved at interactive rates. Dorsey *et al.* [10] render a scene under various lighting conditions and then synthesize a new image by taking a linear combination of the ren-

derings. In effect, they store at each pixel the contributions from a set of light sources. Similarly, Nimeroff *et al.* [19] compute linear combinations of images that have been pre-rendered under daylight conditions. By suitably approximating the angular distribution of the daylight, they construct steerable basis functions that model daylight for arbitrary positions of the sun. This approach has been demonstrated on scenes with diffuse surfaces and smoothly varying lighting.

Several researchers have explored the idea of ray tracing a scene while caching the results for efficient re-rendering. Séquin and Smyrl [23] ray trace scenes and store a shading expression at each pixel. After modifying some simple shading parameters for objects in the scene, the shading expression is re-evaluated to generate an image without casting any new rays. Briére and Poulin [3] extend this idea by storing geometric information about the ray paths in a ray tree. Re-shading can be done as before, but changes in scene geometry are handled rapidly through efficient updating of the ray tree. Miller and Mondesir [18] ray trace individual objects and store a ray tree at each pixel for fast compositing into environments consisting of the front and back faces of an environment mapped cube. Each of these methods benefits from the generality of ray tracing and models phenomena such as colored reflection and refraction. However, the scenes are synthetic, and the general effects of glossy and translucent surfaces must be modeled using methods such as distributed ray tracing [7], requiring multiple ray trees per pixel.

Rendering synthetic scenes to augment pixel color is straightforward when compared to the task of acquiring the information from real images. Blue screen matting, pioneered by Vlahos [25], relies on a single-color background sufficiently different from foreground objects. As noted above, Smith and Blinn [25] use two backdrops to lift restrictions on the color of foreground objects. Reflection from the background onto the foreground, called *blue spill*, however, remains troublesome even with two backdrops and results in incorrect object transparency after matte extraction. Our method attempts to model this reflection correctly as a redirection of light from the backdrop.

To acquire our environment matte, we could illuminate one point of light at a time and sweep over the environment around the object. To cover this area would require $O(n^2)$ images where n^2 is proportional to the area of an environment around the object. Instead, we take inspiration from the structured light range scanning literature. Using a swept plane of light, $O(n)$ images can give shape through optical triangulation [1]. By projecting a hierarchy of progressively finer stripe patterns, the required number of images can be reduced to $O(\log n)$ [21]. Using intensity ramps, the number of images can be reduced, in theory, to two [5]; however, such methods are highly susceptible to noise. Hybrid methods have been proposed to manage the trade-off between number of images and susceptibility to noise [6, 16]. Our environment matting approach is based on the hierarchical stripe methods. Note, however, that the structured light rangefinders assume a diffuse object and attempt to intersect a line of sight from the sensor with a line of sight from the projected illuminant. In our case, the surface is typically assumed to be relatively shiny, even refractive, and the line of sight of the sensor can be arbitrarily remapped onto a *diffuse* structured light pattern.

Finally, a number of methods have been developed to render reflective and refractive objects into environments. Blinn and Newell [2] introduced the environment (a.k.a. reflection) mapping to model reflections from geometric objects into environments stored as textures. Greene [14] refined the idea to model some limited shading effects. Voorhies and Foran [26] later demonstrated environment mapping at interactive rates using texture mapping hardware. In addition, Kay and Greenberg [17] developed an approximate method for fast rendering of refractive objects. Each of these methods is based on object geometry, and none, to our knowledge, has been demonstrated to represent refraction accurately or model spatially varying translucency and gloss at interactive rates.

Miller and Mondesir's hyper-sprite rendering [18] is closest to our rendering approach; however, it uses supersampling for antialiasing. With summed area tables [8], we can achieve texture antialiasing as well as substantial gloss or translucency at interactive rates.

1.2 Overview

The following three sections discuss, in order, our new representation for capturing refraction and reflection properties of a foreground element, the *environment matte* (Section 2); the *environment matting* process, in which an environment matte is extracted from a real-world scene (Section 3); and the *environment compositing* process, in which foreground elements are placed into new environments (Section 4). The discussion of our main results (Section 5) is followed by a sketch of some early results that extend our approach to allow the relative depth of the object and background to be set at compositing time (Section 6). We conclude with some ideas for future research (Section 7).

2 The environment matte

We begin with the traditional compositing equation and then augment it with a new structure, the environment matte, which captures how light in the environment is refracted and reflected by a foreground element.

In traditional digital compositing, the color C that results from placing a foreground element with color F and matte α over a background with color B is given by the "Matting Equation" [25], which is computed at each pixel:

$$C = F + (1 - \alpha) B$$

With the Matting Equation, an element's matte, or alpha, has traditionally played a dual role: it is used to represent, simultaneously, both the *coverage* of a pixel by a foreground element, and the *opacity* of that element. In our approach, we use alpha to describe coverage only. Thus, pixels with no foreground element have an alpha of zero, while pixels that are completely covered have an alpha of one—even if the foreground element covering that pixel is semi-transparent or opaque. Similarly, fractional values of alpha are used to represent partial coverage by a foreground element—again, regardless of that element's transparency.

Likewise, in traditional compositing, an element's color can be thought of as a conglomeration of several different components. These include:

1. any emissive component that the foreground object may have;

2. any reflections coming from light sources in the scene; and

3. any additional reflections or transmission of light from the rest of the environment in which the foreground object was photographed.

In our approach, an element's *foreground color* is used to characterize just the first two of these components.[1]

In addition, we use a new structure, the environment matte, to capture how light in an environment is refracted and reflected by a

---

[1] An alternative would be to represent just the emissive component in the foreground color, and to treat any light sources as just another part of the environment. However, because light sources are generally orders of magnitude brighter than their environments, we have found it to be more practical to separate out their contribution—in much the same way that shaders typically consider the contributions of light sources independently from those of the rest of the environment. (An interesting alternative would be to use the "high dynamic range images" introduced by Debevec and Malik [9].)

foreground element. Thus, the environment matte will capture any transmissive effects of the foreground element—in a manner independent of its coverage, or alpha. The resulting "Environment Matting Equation" will have the following form:

$$C = F + (1 - \alpha)B + \Phi$$

where Φ represents the contribution of any light from the environment that reflects from or refracts through the foreground element.

Our representation for the environment matte should satisfy two requirements. First, the representation should admit a fast compositing algorithm. We therefore choose to represent our environments as sets of texture maps (as in "environment mapping" [2]). Thus, our environment mattes contain indices into these texture maps, along with a small set of additional parameters. Second, we would like the representation to be reasonably concise—that is, to require only a small, constant amount of data per environment map.

We therefore begin with a general formulation for light transport from an environment through the foreground element at each pixel, and then derive an approximation that meets our requirements. This derivation will allow us to bring out each source of error and characterize it explicitly as the approximation is described.

To start, we will assume that the only light reaching the foreground object is light coming from distant parts of the scene. This is essentially the "environment mapping assumption." We use this assumption to create the following simplified model of light transport.

As in Blinn and Newell's original formulation [2], we can describe an environment as light $E(\omega)$, coming from all directions ω. The total amount of light Φ emanating from the portion f of a foreground element that is visible through a given pixel can then be described as an integral over f of all light from the environment that contributes to point $\boldsymbol{p}$ in the pixel, attenuated by some reflectance function $R(\omega \rightarrow \boldsymbol{p})$:

$$\Phi = \iint R(\omega \rightarrow \boldsymbol{p}) \, E(\omega) \, d\omega \, d\boldsymbol{p}$$

Note that the reflectance function $R(\omega \rightarrow \boldsymbol{p})$ includes the overall effect of all absorption and scattering as seen through the foreground element. In addition, Φ, R, and E all have an implicit wavelength dependence.

Our first approximation is to assume that the reflectance function $R(\omega \rightarrow \boldsymbol{p})$ is actually constant across the covered area of a given pixel, allowing us to write this formula in terms of a new function $R(\omega)$ that is independent of position within the pixel:

$$\Phi = \int R(\omega) \, E(\omega) \, d\omega$$

Next, we break the integral over the environment into a summation over a set of m texture maps $T_i(\boldsymbol{x})$, where each texture map represents light coming from a different part of the environment (for example, from different sides of a cube).

$$\Phi = \sum_{i=1}^{m} \int R_i(\boldsymbol{x}) \, T_i(\boldsymbol{x}) \, d\boldsymbol{x}$$

Here, the integral is taken over the entire area of each texture map, and $R_i(\boldsymbol{x})$ is a new reflectance function, which describes the contribution of light emanating from a point $\boldsymbol{x}$ on texture map T_i to the pixel p.

Finally, we make one more simplifying assumption: that the contribution from a texture map T_i can be approximated by some constant K_i times the total amount of light emanating from some axis-aligned
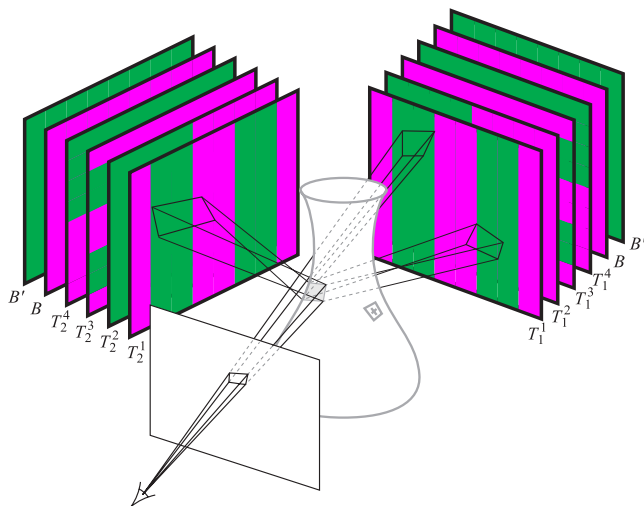

Figure 2 The environment matting process uses structured textures to capture how light is reflected and refracted from a backdrop (right shaft), as well as from various sidedrops (left shaft). The process also captures light coming from the backdrop that is seen through uncovered portions of a pixel (center shaft).

rectangular region A_i in that texture map. This approximation is reasonable in practice for many specular surfaces. We discuss its limitations in Section 5.

Most standard texture-mapping methods actually compute the *average* value of an axis-aligned region of a texture, so we'll let $R_i = K_i A_i$. Letting $\mathcal{M}(T, A)$ be a texture-mapping operator that returns the average value of an axis-aligned region A of the texture T, we have:

$$\Phi = \sum_{i=1}^{m} K_i \int_{A_i} T_i(\boldsymbol{x}) \, d\boldsymbol{x}$$

$$= \sum_{i=1}^{m} K_i A_i \, \mathcal{M}(T_i, A_i)$$

$$= \sum_{i=1}^{m} R_i \, \mathcal{M}(T_i, A_i)$$

We use the above approximation for Φ. Thus, our overall "Environment Matting Equation" becomes:

$$C = F + (1 - \alpha)B + \sum_{i=1}^{m} R_i \, \mathcal{M}(T_i, A_i) \qquad (1)$$

(Note that the reflectance coefficients R_i are essentially "premultiplied," in the sense that they do not need to be multiplied by the element's pixel coverage α. This result falls out of our earlier definition of Φ, in which we integrate over only the covered portion of the pixel.)

As mentioned above, light at each wavelength should ideally be treated independently. Instead, we use the standard computer graphics approximation of treating light as having just three color components: red, green, and blue. Thus, in our implementation, the quantities Φ, R_i, and L_i are treated as 3-component vectors, while T_i is a 2-D array of the same type. The additions and subtractions in the Environment Matting Equation above are treated as componentwise addition and subtraction.

Figure 3 A photograph of the experimental setup used to capture environment mattes. The camera in the foreground photographs objects surrounded by structured light patterns displayed on the computer monitors. The image is processed to extract only the area covered by the backdrop pattern, resulting in an image as shown in the inset.

3 Environment matting

Here, we consider the problem of extracting, from photos of a real-world object, a foreground color F, pixel coverage value α, and environment matte $\Phi = \{(R_1, A_1), \ldots, (R_m, A_m)\}$, at each pixel.

Our approach is motivated by the use of structured light systems for acquiring depth. However, unlike these previous systems, in which objects are sampled at individual points, we would like to capture how the collection of *all* rays passing through a pixel are scattered into the environment. (Really, we're interested in the opposite—how all rays in the environment are scattered through an individual pixel toward the eye—but it's easier to think in terms of this "backward ray tracing" approach [13].)

To this end, we use a number of different patterned textures, which we call *backdrops* and *sidedrops*, displayed on monitors behind and to the sides of the foreground object (see Figure 2). Each patterned backdrop is photographed both with and without the foreground object in front of it. In what follows, we will call the image of the backdrop alone the *reference image*, and the image of the foreground object in front of the backdrop the *object image*. We then solve a non-linear optimization problem to determine the set of parameters most consistent with all the image data. We use patterns that vary in only one dimension, thereby decomposing the problem of finding a rectangle into that of finding two one-dimensional intervals. In theory, any linearly-independent arrangement of horizontal and vertical stripes should work for our series of patterns. In practice, though, we felt that using coherent patterns would help reduce the chance that a slight misregistration would cause errors in our interval detection. We therefore chose to use striped patterns corresponding to one-dimensional Gray codes. We chose stripes of magenta and green, since these two colors are orthogonal in *RGB* space and have similar luminance. Our technique also requires views of the object photographed against two solid backdrops.

The dimensionality of the Environment Matting problem is unfortunately quite large: there are three degrees of freedom for the foreground color F and for each reflectance coefficient R_i (one for each color component), four more degrees of freedom for each area extent A_i, and one more for α. To make this extraction problem more

tractable, we separate its solution into four stages. We begin by considering only the backdrop—the face of the environment directly behind the object, which, by convention, we will number as the first in our list of environment textures. In the first stage, we use different backdrops to compute a coarse estimate for α. We then determine F and R_1 for any pixels covered by the foreground element. Next, we solve for A_1, along with a finer estimate of α along the element's boundary. Finally, once we have found F, α, R_1, and A_1, we can determine the R_i and A_i values for other faces of the environment.

3.1 A coarse estimate of coverage

We begin by computing a coarse estimate of coverage for each pixel. We first partition each pixel of the environment matte into two classes: covered and uncovered. A pixel is considered covered if, for any of the n background images, the colors of the reference and corresponding object images differ by more than a small amount ϵ. Next, we use morphological operations [24] (an *open* followed by a *close*, both with a 5×5 box as the structuring element) to clean up the resulting alpha channel, removing any stray isolated covered or uncovered pixels.

The uncovered, or *background*, pixels will be assigned an alpha of 0. The covered pixels need to be further partitioned into *foreground* pixels (which have an alpha of 1) and *boundary* pixels at the object silhouette (for which we must determine a fractional alpha). We perform this second partition by eroding the binary image obtained from the first step, and subtracting it from the uneroded image. Computation of the fractional alpha is done at the same time as the rectangle estimation, which is discussed below in section 3.3.

An alternative to this division of the image pixels into three classes ("background," "foreground," and "boundary") would be to independently determine alpha at each pixel from the patterned backdrops—in effect, to treat each pixel as a boundary pixel, and calculate the best alpha for that pixel along with its rectangle. This was indeed the approach we first took, but when working with real photographic data there were inevitably a few single-pixel errors. These errors are generally not noticeable in still composites, but they stand out dramatically when the composited object is in motion with respect to a background. They appear as a "speck of dust" moving in sync with the object, or as a tiny "hole" where the background shines through. Our morphological approach offers cleaner results at the expense of some mathematical elegance. In addition, since it inexpensively determines alpha for most of the pixels in the image, using the morphological approach speeds the acquisition process considerably.

3.2 The foreground color and reflectance coefficients

For covered pixels (pixels with $\alpha > 0$), we need to determine the foreground color F and reflectance coefficients R_i for each face of the environment. To do this, we first use photographs of the object in front of two different solid backdrops, and solve for R_1 and F.

For a given pixel, let B be the color of the first backdrop as seen from the camera, and B' the color of the second. Let C and C' be the colors of the foreground object over these two backdrops, respectively, as seen by the camera. These four colors are related by the Environment Matting Equation (1):

$$C = F + (1 - \alpha)B + R_1 B$$
$$C' = F + (1 - \alpha)B' + R_1 B'$$

We now have two equations in two unknowns, which are easily solved for R_1 and F, expressed here as functions of α:

$$R_1(\alpha) = (C - C')/(B - B') - (1 - \alpha) \qquad (2)$$
$$F(\alpha) = C - (1 - \alpha + R_1)B \qquad (3)$$

3.3 The area extents and a refined estimate of coverage

To refine the alpha values for pixels on the boundary, and to determine the axis-aligned rectangle A_1 of the background that best approximates the reflection and refraction in the scene, we minimize an objective function over the series of photographed images for each covered pixel in the scene:

$$E_1 = \sum_{j=1}^{n} \left\| C^j - F(\alpha) - (1-\alpha)B^j - R_1(\alpha)\mathcal{M}(T_1^j, A_1) \right\|^2$$

Here, B^j and C^j are the colors of the pixel in question in the reference image and object image, respectively, when the j-th pattern is displayed as a backdrop. Similarly, the texture map T_1^j is obtained by taking a reference photograph of the j-th pattern, displayed as a backdrop. The functions $F(\alpha)$ and $R_1(\alpha)$ are computed according to equations (2) and (3) above. Finally, the (squared) magnitude is computed as the sum of squared distances between colors in *RGB* space. Our aim will be to find the rectangular region A_1 that minimizes the objective function E_1.[2]

This optimization problem still has four degrees of freedom for the unknown rectangle A_1: left, right, top, and bottom (l, r, t, b). We can reduce the dimensionality further by using horizontal and vertical stripes for backgrounds. For horizontally striped backgrounds, the area determination is independent of l and r; similarly, for vertically striped backgrounds, the area determination is independent of t and b. Thus, the five-dimensional problem of minimizing E_1 over (α, l, r, t, b) can be effectively reduced to two three-dimensional problems: minimizing over (α, l, r) with one set of patterns, and over (α, t, b) with another.

We begin by assuming a value for α: for foreground pixels, $\alpha = 1$; while for boundary pixels we will try out multiple values. We then look for an interval $[l, r]$ that minimizes E_1 over the vertically striped patterns by testing a large number of candidate intervals. To speed this search, we apply a multiresolution technique, finding the best interval at some coarse scale, and then repeatedly subdividing to look for better approximations with narrower intervals. We use the same technique with the horizontally striped patterns to find the vertical extents $[t, b]$, and thus determine the rectangle. For boundary pixels, we repeat this search for multiple values of α and output the α and interval $A_1 = (l, r, t, b)$ that together minimize the objective function. This larger search is significantly slower, of course, but is required only at the object silhouette. Moreover a very coarse-grained search of α suffices—we've found that trying just nine values, $\alpha \in \{\frac{1}{8}(0, 1, 2, \cdots, 8)\}$ gives good results.

3.4 Sidedrops

The technique we have described allows us to capture information about how an object refracts and reflects light from a backdrop located directly behind the camera. To capture environment mattes for light coming from the other parts of the environment, we photograph the object while illuminating sidedrops, instead of the backdrop, with the same set of structured patterns.

The extraction process is nearly identical. We no longer need to compute α or F; these values are the same as before. The computation of R_i is also somewhat simpler. When two different solid colors S and S' are displayed on the sidedrop, the corresponding Environ-

---

[2]Note that if using an axis-aligned rectangle were an exact model of the light transport behavior, then minimizing this objective function would correspond to ascribing a Gaussian error model to the pixel values and solving for the optimal parameters in the "maximum likelihood" sense. In fact, the axis-aligned rectangle is only an approximate model, so we are actually finding a least-squares best fit of the model parameters to the data.

ment Matting Equations become:

$$C = F + (1-\alpha)B_1 + R_i S$$
$$C' = F + (1-\alpha)B_1 + R_i S'$$

Here B_1 is the color of the backdrop (typically very close to black). Subtracting these two equations gives a solution for R_i:

$$R_i = \frac{C - C'}{S - S'}$$

The objective functions E_i to minimize for each additional sidedrop i are then given by

$$E_i = \sum_{j=1}^{n} \left\| C^j - F(\alpha) - (1-\alpha)B_1 - R_i(\alpha)\mathcal{M}(T_i^j, A_i) \right\|^2$$

Since we cannot take reference photographs of each structured pattern on the sidedrop (the sidedrops are not visible to the camera) we instead use the photographs of the corresponding patterns on the backdrop to obtain B and B', as well as the texture maps T_i^j.

By placing sidedrops around the object in all directions and illuminating one sidedrop at a time, we can, in theory, obtain a description of all light rays that strike the camera. So far, though, for real objects we have captured environment mattes only for one backdrop and at most two sidedrops.

4 Environment compositing

Once we have an environment matte for a foreground object, we can composite it into a new environment in a way that preserves reflections and refractions.

This process, called environment compositing, is just an implementation of Equation (1). It involves summing the contributions of the foreground and background colors, along with a weighted contribution from each of the texture maps describing the environment.

As in any texture mapping process, we need to perform some kind of filtering to avoid aliasing. We use summed area tables [8], which allow us to compute the average value of an axis-aligned rectangle quickly. In order to handle lenticular objects, which magnify, we generally use a higher resolution for storing texture maps of background images than we use for displaying them as backdrops.

5 Results

We assembled an imaging system using a digital camera and several monitors to acquire environment mattes from a number of objects to illustrate the capabilities of our method. A photograph of the imaging system appears in Figure 3. We placed three monitors in positions consistent with portions of three faces of a cubical environment. We used identical 20" Trinitron monitors and adjusted the displays so that the magenta and green images were as close as possible on all monitors. We photographed each object with a Kodak DCS 520 digital camera and cycled the images on each monitor, one monitor at a time. Interreflections among monitors did not prove consequential. Enough stripe patterns were used so that the width of the smallest stripe corresponded to a pixel of the final matte. Thus, to extract a 512×512 matte, we would use 18 stripe images, nine horizontal and nine vertical. The monitors also displayed registration markers enabling the portion of the image containing the structured pattern to be extracted from each photograph. This extraction process includes a simple warp step that attempts to at least partially correct for monitor curvature and lens distortion.

The extraction process took on the order of 10 to 20 minutes per face of the environment map, running on an Intel Pentium II at

Figure 4 From left to right: an alpha matte composite, an environment matte composite, and a photograph of an object in front of a background image. The top row shows a ribbed glass candle holder; the bottom row shows a rough-surfaced glass bookend.

400MHz. Compositing is much faster, and can performed at speeds of 4 to 40 frames per second.

In order to obtain consistent colors in our side-by-side comparisons, we photographed the various backgrounds on the monitors and used these digital photos as the backgrounds in our composites. In Figures 4 and 5, the resulting composites are compared against photos of the actual objects against the same backgrounds. Within each row of these figures, the left picture shows the composite obtained using traditional alpha, obtained via the Smith-Blinn triangulation method (Theorem 4 of their paper [25]). The middle picture shows the result of compositing using the environment matte method, and the right picture is a photograph of the actual object in front of a monitor displaying the background image (taken at the same time the environment matte and triangulation data were obtained).

The first object is a glass candle holder. The two ribbed bulges produce a very complex refraction pattern, which is accurately recreated by the environment matte.

The second object is a glass bookend, with a rough textured surface that scatters light passing through it, producing a translucent effect. Note that this object simply disappears with the traditional alpha, since each pixel is judged to be transparent or nearly so.

The third set of images (top row of Figure 5) demonstrates the advantage of representing the reflection/refraction coefficient R_1 as an RGB triple. The objects are champagne glasses filled with water tinted red, green, and blue, respectively. The environment matte is able to more accurately reproduce the colors of the background as they appear filtered through the colored water. Note also the suc-

cess of the environment matte in capturing reflections off the base of each glass, especially as compared to the ordinary alpha image.

The fourth set of images (bottom row of Figure 5) provides an example of glossy reflection. The object is a metal pie tin tilted to a near-grazing angle with respect to the camera. For this object, we used a Gaussian-weighted texture extraction operator, as discussed for the next example in detail.

Figure 6 illustrates how our assumption that light reflecting from an object can be described as a constant-weighted average over a region of a texture map begins to break down when reflections are sufficiently diffuse. The leftmost image reveals noticeable banding due to inaccuracies in this approximation. To ameliorate this effect, we attempted to model the reflection at each pixel as an elliptical Gaussian-weighted average over the texture instead, in a model similar to Ward's [27]. To extract the matte, we first modified the texture operator $\mathcal{M}$ during the acquisition process so that it would return a Gaussian-weighted average of the texture map rectangle, rather than a box-filtered average. The Gaussian is chosen so that each side of the rectangle corresponds to a 3σ difference from the center. Fast rendering with summed area tables requires a box filter, so for rendering we take each rectangle acquired and shrink it so that it corresponds to a width of $\frac{3}{2}\sigma$, using the box filter as a very rough approximation of the original Gaussian. (A potentially better alternative would be to use an elliptical weighted average (EWA) filter [15], represented as an image pyramid constructed with Gaussian filters [4] for rendering.) Using this Gaussian-filter approximation for acquisition, followed by a box-filter approximation for rendering, improves the results to some extent. However, finding

Figure 5 From left to right: an alpha matte composite, an environment matte composite, and a photograph of an object in front of a background image. The top row shows three glasses of water tinted red, green, and blue; the bottom row shows a pie tin tilted to reflect light off the backdrop.

an approximation that is general enough to handle both diffuse and specular surfaces and also to provide efficient rendering remains a topic for future research.

Figure 7 shows objects captured using a backdrop and two side-drops. The objects are a shiny metal candlestick, and a metal vase turned on its side. These objects were chosen and positioned so that as much of their surface as possible was reflecting light off the limited sidedrop area available with our experimental setup. The figure shows the contributions of the foreground color and unoccluded background, followed by the separate contributions of each texture map (back, left, and right). The total composite (bottom left) is made by summing these four images, and is shown along with a photograph for comparison (bottom right). Note that even complex reflections are captured—in the base of the candlestick, we can see light reflecting from the backdrop off the side of the vase facing away from the camera.

Figure 8 illustrates the use of the environment matte technique for novel relighting of objects. Here the objects are shown against a background image, but now the sidedrops are synthetically generated from photographs of actual light sources. As these lights are moved around, the environment matte shows how a light at that position would reflect off the surface of the objects. The figure shows the objects with two different lighting configurations, with the texture maps used for the left and right sidedrops shown in inset.

Figure 9 illustrates some of the ways in which our technique can fail to accurately capture reality. The top row shows an environment matte for a stack of four reflective balls: (a) the composite generated with a butterfly image on each face of the environment, (b) the corresponding photograph. While the composite image recreates the photograph well, it highlights the fact that our current setup is able to capture only a small fraction of the object's environment matte—reflections of the camera and other equipment are visible, having been captured in the foreground color term F. This is more a failure of engineering than of theory; with a more elaborate system for surrounding the object with structured patterns a more complete matte could be obtained. The second row of Figure 9 illustrates a more fundamental failure: the result obtained when a single pixel sees two distinct regions of the same backdrop. Here a drinking glass has been tilted so that the single backdrop is visible both through the glass and as a reflection in its surface. This is evident in photograph (d), but the two images of the stripe patterns interfere during the acquisition process. The rectangles obtained are chosen almost arbitrarily, which lead to noise in the composite (c).

6 Depth correction

Thus far, we have presented a method that allows us to acquire environment mattes for objects at a fixed distance from the backdrop. As a result. all of the composites we have shown are also in front of backdrops at that same depth. In order to perform composites with a backdrop at an arbitrary distance, we need to capture information about how the light rays travel through space. In this section, we therefore sketch an extension to our method, which models the light refracted or reflected by the foreground object as a 3D beam with axis-aligned rectangles as cross-sections. The rectangular extents for a backdrop at an arbitrary depth can then be constructed

Figure 6 An object with glossy reflection, captured using an environment matte. From left to right: artifacts produced in composite image when the box filter approximation is used during acquisition, the improvement produced when a Gaussian approximation is used instead, and the actual photograph.

by taking the cross-section of the beam where it is intersected by the backdrop.

To construct a beam for each pixel, we extract two different environment mattes for the foreground object, using a backdrop placed at two different depths. As before, we consider each rectangle's horizontal extent independently of its vertical extent. For a given pixel, let $[l, r]$ and $[l', r']$ be the left and right endpoints of the rectangular extents in the two environment mattes. There are two ways in which these two extents can be connected to form a beam: either l is connected to l', and r to r'; or the two endpoints are flipped, and l is connected to r', and r to l'. The latter case corresponds to the presence of a focal point between the two backdrop planes. To disambiguate between the two cases, we could extract a third environment matte for a backdrop located in between the two original depths, and test to see whether the extents in that environment matte are most consistent with the straight-through connection or with the flipped connection. Instead, as a proof of concept, we have so far used just a simple flag, set by the user, that controls whether the connections for all the pixels are "straight through" or flipped. (Note that since the flag controls flipping of the beam only *within each pixel*, even if the flag is set incorrectly the refracted image appears in the correct orientation; however, the resulting image may be either too sharp or too blurry due to integration over either too small or too large an area when estimating the contributions of the environment to each pixel.)

Finally, to composite these objects in front of backdrops with arbitrary depth, we use linear interpolation to intersect the beam with a plane at that depth. The resulting rectangle is used as the area extent in the normal environment compositing operation. Figure 10 demonstrates some early results of an environment matte with depth, captured for a magnifying glass.

7 Conclusion

In this paper, we have introduced the environment matte and shown how it augments the alpha matte used in traditional image compositing. The environment matte models the effects of reflection, refraction, translucency, gloss, and interreflections. It also handles colored transparency in a manner that more closely approximates reality. We have demonstrated a novel method for acquiring environment mattes for real objects using structured diffuse lighting. To extract the matte from acquired images, we have developed a mathematical framework for analyzing the photographic data. This framework allows us to identify reflection and refraction intervals

in a manner that is statistically optimal in the least squares sense and has proven fairly robust for specular and near specular surfaces. Using summed area tables for fast integration over rectangles, we have developed a software rendering system that composites an environment-matted object onto novel backgrounds at interactive rates with modest hardware. By placing images of lights into the environment, we can use the environment compositor as an interactive lighting tool.

This research leads to many areas for future work. First, we have more work to do to accurately calibrate our acquisition process for non-linearities and to compensate for cross-talk in the color channels. Also in the area of acquisition, we would like to explore ways of reducing the number of images required to capture an environment matte. In addition, our method assumes that there is a single region per texture map onto which a pixel maps. This is not the case when there are abrupt changes in reflected and refracted ray directions that map onto the same backdrop. One could imagine increasing the dimensionality of the maximum likelihood problem to identify more than one region, though more backdrops may be necessary to correlate the sets of horizontal and vertical intervals correctly.

As noted in Section 5, we obtain better results for glossy surfaces when estimating axis-aligned elliptical Gaussians instead of rectangles. An area of future work is to find even better functions, such as oriented, possibly steerable, functions for estimating environment mattes. Similarly, at some expense in performance, we could explore more accurate weighted filtering methods when compositing the environment mattes.

To acquire environment mattes with depth, we would like to further investigate methods for extracting the 3D beam for refraction and reflection. As mentioned earlier, we could acquire information at a third depth, and use this to decide, for each pixel, whether there is a focal point between the captured depth extremes. Instead of extracting the rectangle extents independently for each depth, we could optimize over all three depths simultaneously.

For rendering images with depth, we would like to look into methods for compositing on top of backgrounds with varying depth. Permitting the backdrop to rotate away from the camera might be useful for simulating depth of field effects. Also, the depth information might be used to composite onto arbitrary 3D scenes, rather than a 2D background. We could also investigate methods for compositing multiple environment "sprites" into one scene. Miller and Mondesir [18] note that the straightforward method of layering multiple

Figure 7 An environment matte captured using multiple sides. The top left image shows the effects of the foreground color and uncovered pixel terms. The next three images show the light contributed via reflection off the back, left, and right texture maps. The final composite is the sum of these four images, shown at lower left. This is compared with a photograph of the object surrounded by these images at the lower right.

Figure 8 Interactively relighting an object, using photographs of lights as sidedrops. The left and right sidedrops (shown in insets) are generated by moving a picture of a light source around within the texture map.
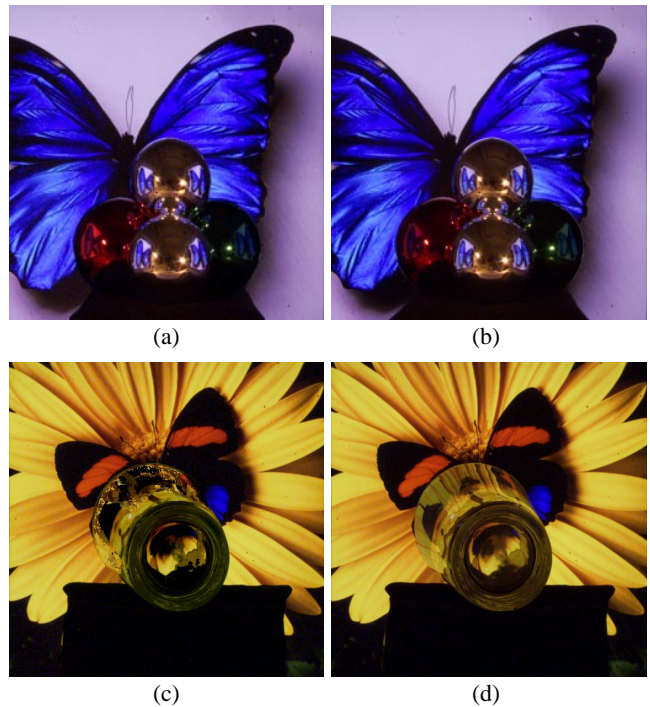

(a)   (b)

(c)   (d)

Figure 9 Failure cases.

sprites into a background, while not correct, still gives a compelling appearance. Nonetheless, some loss of "depth" must occur. Perhaps our method for capturing depth information could lead to more realistic composites. Indeed, it might be possible to acquire or render our depth environment mattes from many viewpoints, leading to a new rendering primitive: a specular transfer function or specular light field. For each incoming ray, we could quickly determine a set of outgoing rays or prisms.

In other rendering areas, we could explore methods like view morphing [22] to transition among environment mattes acquired from different viewpoints. In addition, images from other viewpoints could be used to cast rays through the object into the environment to create caustics and shadows.

Another interesting area to explore, suggested by this line of research, would include methods for creating and modifying environment mattes for real or synthetic objects, interactively or algorithmically. For instance, one could easily imagine a "paint program" in which the light scattering properties of a scanned object are interactively modified through various paint operations; or texture generation methods, in which its reflectance properties are modified algorithmically.
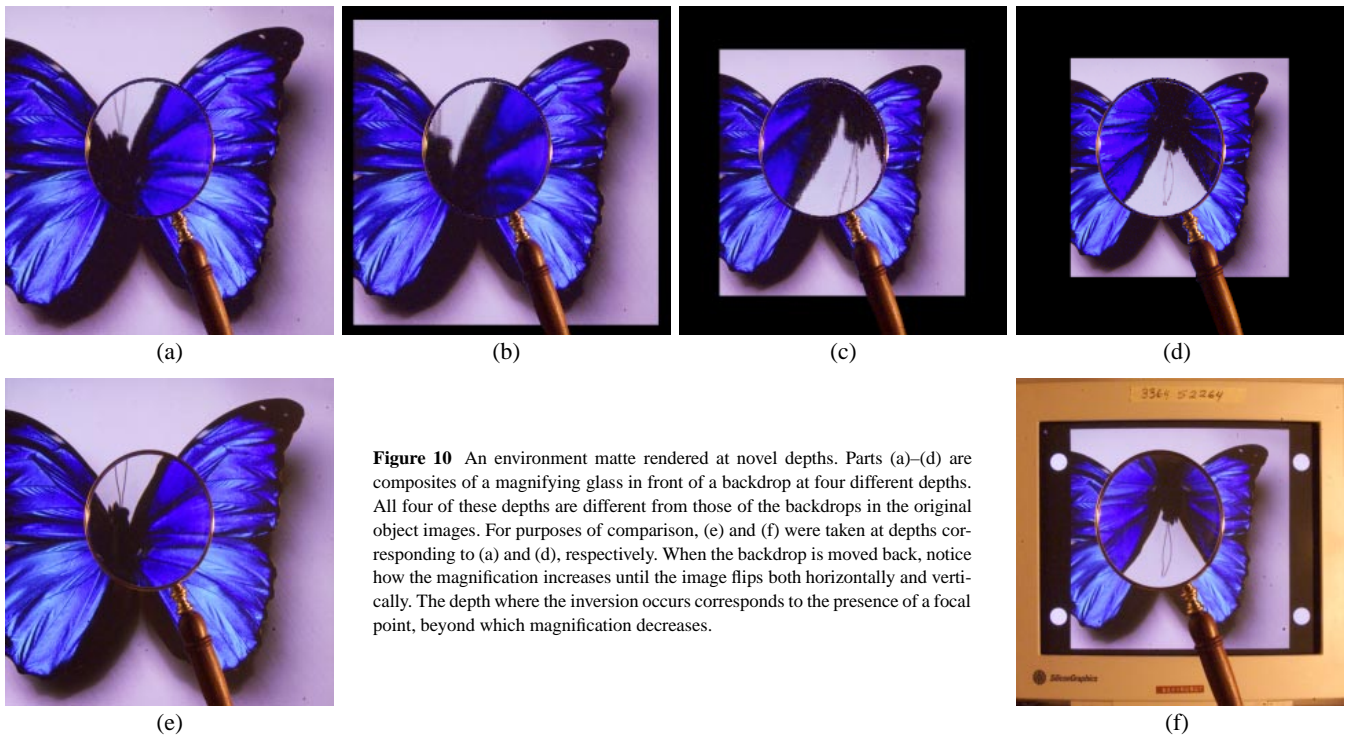
8   Acknowledgements

Figure 10 An environment matte rendered at novel depths. Parts (a)–(d) are composites of a magnifying glass in front of a backdrop at four different depths. All four of these depths are different from those of the backdrops in the original object images. For purposes of comparison, (e) and (f) were taken at depths corresponding to (a) and (d), respectively. When the backdrop is moved back, notice how the magnification increases until the image flips both horizontally and vertically. The depth where the inversion occurs corresponds to the presence of a focal point, beyond which magnification decreases.

References

[1] Paul Besl. Active optical range imaging sensors. In Jorge L.C. Sanz, editor, *Advances in Machine Vision*, chapter 1, pages 1–63. Springer-Verlag, 1989.

[2] J. F. Blinn and M. E. Newell. Texture and reflection in computer generated images. *Communications of the ACM*, 19:542–546, 1976.

[3] Normand Briére and Pierre Poulin. Hierarchical view-dependent structures for interactive scene manipulation. In *Proceedings of SIGGRAPH 96*, pages 83–90, August 1996.

[4] Peter J. Burt. Fast filter transforms for image processing. *Computer Graphics and Image Processing*, 16:20–51, 1981.

[5] Brian Carrihill and Robert Hummel. Experiments with the intensity ratio depth sensor. In *Computer Vision, Graphics, and Image Processing*, volume 32, pages 337–358, 1985.

[6] G. Chazan and N. Kiryati. Pyramidal intensity ratio depth sensor. Technical Report 121, Center for Communication and Information Technologies, Department of Electrical Engineering, Technion, Haifa, Israel, October 1995.

[7] Robert L. Cook, Thomas Porter, and Loren Carpenter. Distributed ray tracing. *Computer Graphics*, 18(3):137–145, July 1984.

[8] Franklin C. Crow. Summed-area tables for texture mapping. In *Proceedings of SIGGRAPH 84*, volume 18, pages 207–212, July 1984.

[9] Paul E. Debevec and Jitendra Malik. Recovering high dynamic range radiance maps from photographs. In *Proceedings of SIGGRAPH 97*, pages 369–378, August 1997.

[10] Julie Dorsey, James Arvo, and Donald Greenberg. Interactive design of complex time dependent lighting. *IEEE Computer Graphics and Applications*, 15(2):26–36, March 1995.

[11] R. Fielding. *The Technique of Special Effects Cinematography*, pages 220–243. Focal/Hastings House, London, third edition, 1972.

[12] Reid Gershbein. *Cinematic Lighting in Computer Graphics*. PhD thesis, Princeton University, 1999. Expected.

[13] Andrew S. Glassner. An overview of ray tracing. In Andrew S. Glassner, editor, *An Introduction to Ray Tracing*, chapter 1. Academic Press, 1989.

[14] Ned Greene. Environment mapping and other applications of world projections. *IEEE Computer Graphics and Applications*, 6(11), November 1986.

[15] Ned Greene and Paul S. Heckbert. Creating raster omnimax images from multiple perspective views using the elliptical weighted average filter. *IEEE Computer Graphics and Applications*, 6(6):21–27, June 1986.

[16] Eli Horn and Nahum Kiryati. Toward optimal structured light patterns. In *Proceedings of the International Conference on Recent Advances in Three-Dimensional Digital Imaging and Modeling*, pages 28–35, 1997.

[17] Douglas S. Kay and Donald P. Greenberg. Transparency for computer synthesized images. In *Proceedings of SIGGRAPH 79*, volume 13, pages 158–164, August 1979.

[18] Gavin Miller and Marc Mondesir. Rendering hyper-sprites in real time. In G. Drettakis and N. Max, editors, *Proceedings 1998 Eurographics Workshop on Rendering*, pages 193–198, June 1998.

[19] Jeffry S. Nimeroff, Eero Simoncelli, and Julie Dorsey. Efficient re-rendering of naturally illuminated environments. In *Fifth Eurographics Workshop on Rendering*, pages 359–373, June 1994.

[20] Thomas Porter and Tom Duff. Compositing digital images. In *Proceedings of SIGGRAPH 84*, volume 18, pages 253–259, July 1984.

[21] K. Sato and S. Inokuchi. Three-dimensional surface measurement by space encoding range imaging. *Journal of Robotic Systems*, 2:27–39, 1985.

[22] Steven M. Seitz and Charles R. Dyer. View morphing: Synthesizing 3D metamorphoses using image transforms. In *Proceedings of SIGGRAPH 96*, pages 21–30, August 1996.

[23] Carlo H. Séquin and Eliot K. Smyrl. Parameterized ray tracing. In *Proceedings of SIGGRAPH 89*, volume 23, pages 307–314, July 1989.

[24] Jean Paul Serra. *Image Analysis and Mathematical Morphology*. Academic Press, 1982.

[25] Alvy Ray Smith and James F. Blinn. Blue screen matting. In *Proceedings of SIGGRAPH 96*, pages 259–268, August 1996.

[26] Douglas Voorhies and Jim Foran. Reflection vector shading hardware. In Andrew Glassner, editor, *Proceedings of SIGGRAPH 94*, pages 163–166, July 1994.

[27] Gregory J. Ward. Measuring and modeling anisotropic reflection. In Edwin E. Catmull, editor, *Computer Graphics (SIGGRAPH '92 Proceedings)*, volume 26, pages 265–272, July 1992.