Digital Compositing

Martin Berlakovich

Abstract

It is often intended to merge two or more digital images into one image, an example would be special effects in films or photomontages. The process of assembling these images into one final image is called digital compositing.

While the first intention might be to just overlay the images the problem is far more complicated. Several techniques have been introduced to increase the quality and the realism of the created images. While the first steps were to fight aliasing by using the so called alpha channel and looking in the problem of depth other techniques concentrated on the realism of an image, caused by interaction with the environment. For example a newly added object in an image might throw a shadow or reflect the environment. These techniques are called shadow and environment matting.

This article should be a review about history and applications of digital compositing as well as providing an overview of the current state of the art.

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—

Keywords: digital compositing, alpha channel, shadow matting, environment matting

1 Introduction

The problem of compositing of images is not limited to digital images. In fact the problem as well as some techniques already existed for non-digital film so it was no suprise that when film and images went digital, compositing did the same thing. The most common way to do composition of images was to create one image which is transparent where the second image should be visible, called a matte. The first techniques of digital compositing were just digital representations of existing techniques for non-digital film. However digital compositing offered more possibilities to improve the quality and realism of images and it did not take long until new techniques evolved.

The first and maybe one of the most important attemps to improve the quality of composited images was to store more information in a picture. The transparency-value of a pixel, the so called alpha channel, was introduced by Catmull and Smith in the late 70s which extended the RGB (red green blue) to the RGBA (red, green, blue, alpha) representation. By using the alpha channel the borders between two pictures become smooth and the visible edges called jaggies disappeared. Another attempt to store more information was the RGBAZ representation which additionally stored the depth value. Further techniques like shadow and environment matting followed in the next years. Since in most cases object throw shadows a picture must consider that the fact that shadows in one picture might look different in another picture. Also the environment conditions might be different, for example a reflecting plate will look different in a blue than in a green background. All of these techniques will be discussed in this article.

2 Compositing

In this section the basics of compositing images will be discussed. The most common way to make composites of images is to use an extra channel, called the alpha channel, which stores their transparency value. By comparing this value a decision is made which picture is visible at each pixel.

2.1 The Alpha Channel

The basic idea of an additional channel which contains the transparency information, or short alpha, was first introduced by Catmull and Smith in the late 70s. Really developed, however, it was by Duff and Porter in 1984.

The main motivation was to combine two images. However if two one image is just displayed over another you sharp borders will occur on the edges of the top picture, this aliasing effects are called jaggies. The effect can be reduced by using very high resolutions, but this leads to other problems like memory intensive pictures and does not really solve the problem of the aliasing effect, just makes it less visible.

Another method to combine two images, without having the problems of visible silouhettes at the borders of the matte, was needed, which leaded to the development of alpha. The main idea behind alpha is not to decide whether a pixel is visible or transparent but choose a degree of transparency. In other words a pixel is not only 0 or 100 percent transparent, but 0 to 100 percent. The transparency value of a pixel can vary between 0, which means totally transparent, and 1 which would be full coverage. For example an alpha value of 0.5 would mean that the pixel is half transparent. Considering this an example for a RGBA color representation would be (1,0,0,0.5) for a red half transparent color. However in order to display the color the red would have to be multiplied with the alpha value. And not only the red color, but the green and blue color too, and this for every pixel in the picture, and this for all pictures and this for every process. This problem leads to the idea of premultiplied alpha.

As described above the simple RGBA representation leads to many multiplications which would have to be done for every pixel. However exactly the fact that is has to be done every time leads to the idea of storing the already multiplied values for the red, green and blue color channels. This RGBA representation is called premultiplied alpha. Not the color values, but the color values multiplied by the alpha value are stored in the color channels, and additionally the alpha value is stored in the alpha channel.

To illustrate what is meant, imagine a red object which is half transparent. The RGB representation of red would be (1,0,0) and the alpha value of half transparent 0.5. So the RGBA representation of the half transparent red object would be (0.5,0,0,0.5). Due to the multiplication it is not possible that any of the color values is different than zero if the object is transparent (alpha is 0).

With the problem of representing colors and alpha solved and the idea premultiplied alpha introduced the next step ist to combine two images, or in other words determine what the final picture looks like. The whole process is called blending and due to alpha, alpha blending. In the paper of Porter and Duff from 1984 a list of binary operators for computing two images are introduced. They are based on the idea that if the alpha value of image A is α_A, (1-α_A) of the background image is seen through, or in another way $\alpha_A(1-\alpha_B)$, with α_B being the alpha value of the second image, is being blocked by alpha.

This leads to the representation in subpixel areas. Imagine two pictures A and B which divide a pixel in 4 subpixel areas, depending which areas of the pixel are covered by each of the two pictures. Table 1 illustrates what is meant in detail.

A	B	name	description	choices
0	0	0	$\overline{A}\bigcap\overline{B}$	0
1	0	A	A$\bigcap\overline{B}$	0, A
0	1	B	$\overline{A}\bigcap B$	0, B
1	1	AB	A$\bigcap B$	0, A, B

Table 1: Possible situations and handling.

These four subpixel areas can be represented by a quadruple (0,A,B,AB) which is affected by the way the two pictures are computed. As an example let us assume that picture A lies over picture B, which means that in the area where both pictures could be chosen we will now choose A. This leads to the quadruple (0,A,B,A). Figure 1 shows other binary operations.

These operators show how specific situations are treated. The over operator represents the foreground-background relation, the in operator defines which picture is dominant where, and only where, both picture could be seen, while A out B refers only to the part of A which lies outside of B. The use of the stop and xor operators are very limited and are only introduced for completeness.

The most important operator is the over operator. A over B can be interpreted as A being the foreground object over a background object B. This is the main operation used when doing compositing, since usally one picture is considered foreground and the other the background picture. The colums F_A and F_B indicated the fraction of the original picture covered by it in the output picture. As you can see in the example A over B, F_B is now reduced by α_A.

These fractions are relevant in order to composite two images. For each output pixel we want to compute the contribution of the input pixels. If we consider that each input picture contributes to the output by its alpha value in relation to the fraction covered we can derive that the total area covered can be computed by adding α_A times F_A to α_B times F_B.

We can now compute the color of the composited picture by calculating each color channel (red, green and blue) separately by using the following formula

$$C_O = c_A F_A + c_B F_B \tag{1}$$

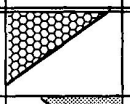
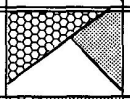
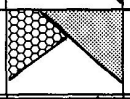
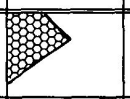
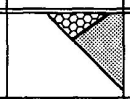
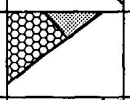
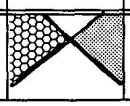
operation	quadruple	diagram	F_A	F_B
clear	(0,0,0,0)		0	0
A	(0,A,0,A)		1	0
B	(0,0,B,B)		0	1
A **over** *B*	(0,A,B,A)		1	1–α_A
B **over** *A*	(0,A,B,B)		1–α_B	1
A **in** *B*	(0,0,0,A)		α_B	0
B **in** *A*	(0,0,0,B)		0	α_A
A **out** *B*	(0,A,0,0)		1–α_B	0
B **out** *A*	(0,0,B,0)		0	1–α_A
A **atop** *B*	(0,0,B,A)		α_B	1–α_A
B **atop** *A*	(0,A,0,B)		1–α_B	α_A
A **xor** *B*	(0,A,B,0)		1–α_B	1–α_A

Figure 1: Binary compositing operations from [PorterDuff84].

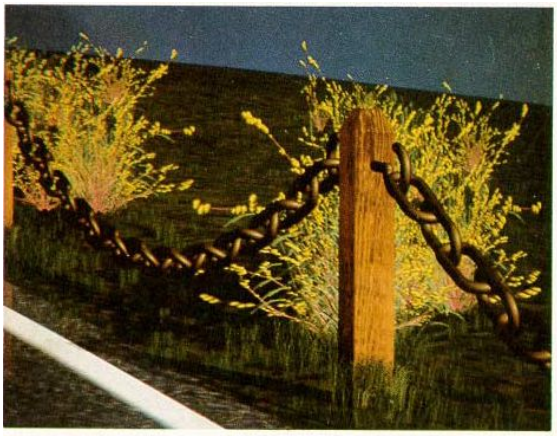


Figure 2: Output picture composed by 19 input pictures from [PorterDuff84].

with C_O being the output color and c_A and c_B being the input colors. The fractions F_A and F_B can be looked up in the table above.

This example will help illustrating what is meant. Let us imagine a kind of purple color which is partly transparent (0.5,0,1,0.75) which we lay over a white, totally opaque background (1,1,1,1). Since this is an A over B operation we can look up F_A as 1 and F_B as (1-α_A) which leads to the formula

$$C_O = c_A + c_B(1-\alpha_A) \tag{2}$$

for the colors and

$$\alpha_O = \alpha_A + \alpha_B(1-\alpha_A) \tag{3}$$

for the transparency of the output color, with α_O being the alpha of the output color. Doing these calculations will provide the RGBA color representation of the pixel as (0.625,0.25,1,1).

In addition to these binary operators also some unary were introduced. The first operator is named darken, although it can also be used to brighten the elements. When darkening (or brightening) the red, green and blue are just multiplicated by and ϕ ¡ 1, while 1 means no change, while 0 would be complete black. If ϕ is chosen bigger than 1 the elements appear brighter than before.

The second unary operator is called dissolve. When dissolving the whole RGBA representation is multiplied by an δ which is the factor with which the element is fading.

The last unary operator is called opaque. Here only the alpha component is changed by an ω between 0 and 1. Since alpha indicates the transparency value the element of course gets more and more transparent as the ω gets closer to zero.

The last operator which was introduced by Porter and Duff is the plus operator. Here both components are simply added in the area covered by both pictures. By dissolving the participation of the input pictures can be controlled.

By using only these operators the first digital compositing was done. The results are pictures which look as if they are one picture, not composited, since the object boundaries can not be seen and therefore the picture appears natural. Figure 2 for example was composed of 19 pictures.

2.2 Including Depth

Figure 2 shows that with the operations from the last section it is already possible to achieve natural looking pictures when compositing many images into one composite. However since every part of the picture (the fence, the plants, the smaller plants, etc.) was an input picture of its own it is easy to assume that the picture would look different if the pictures were composed in a different order. In other words, to get a correctly composited picture the front-to-back order of the pictures must be known.

An approach made by Duff in [Duff85] was to combine the RGBA representation, only containing the color with the alpha value, with a Z-buffer. A Z-buffer decides which color is seen in a composited image by computing the z value.

$$rgb_c = (if\ z_f < z_b\ then\ rgb_f\ else\ rgb_b) \tag{4}$$

Here rgb_c, rgb_f and rgb_b mean the composited, the foreground and the background color and z_f and z_b are the depth values of the pixels of the foreground and the background picture. An interpretation of this would be an operator zmin which computes two pictures at each pixel. This operator is associative and commutative. The over operator of the previous section unfortunately is not commutative. Of course it does not matter whether two elements are composited front-to-back or back-to-front, but they must be adjacent in depth when they are combined.

The rgbaz algorithm developed by Duff now introduces a comp operator which combines the actions of zmin and over. All pixels of a picture contain the rgb values for the color, the alpha values for transparency and a z value for depth at each corner of the pixel. If a corner is not covered its z value is set to infinity, leaving it larger than any legitimate z value.

If two pictures f and b are computed, in formula f comp b, the z values at the four corners of every pixel are compared. There are 16 possible outcomes of this comparison, but only two of them will have the same outcome at all four corners of the pixel. In the other 14 cases, if at some corners of the picture the z values of f are smaller (or bigger) than the ones of b while at some this is not the case, we call the pixel confused. If this is the case the z values are linearly interpolated to find the point at which they are equal at each of the four edges of the pixel. Figure 3 shows the 16 cases.

This leads to the following formula for the rgba values

$$rgb\alpha_c = \beta(f\ over\ b) + (1-\beta)(b\ over\ f) \tag{5}$$

with β being the fraction of f which is in front of b. The new z value is the minimum of the both z values.

$$z_c = min(z_f, z_b) \tag{6}$$

The comp operator is commutative, because if f and b are interchanged so is β and (1-β). The operation is only associative for unconfused pixels, since then only f over b, or b over f, is done. Since linearly separable objects are free of confused pixels the comp operator performs just as the over operator here. However small errors can occur if comp is applied in a completely arbitrary order. These errors can be eliminated if the elements are sorted by z coordinates before the comp operation is done. Another problem are small objects that fall between pixels, as they might be lost during the process.

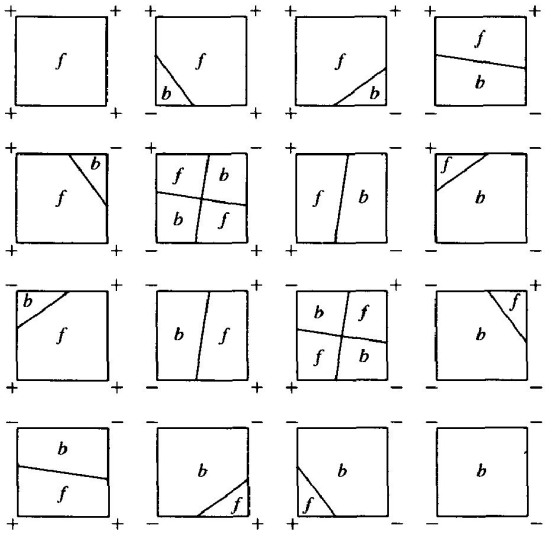


Figure 3: The 16 possible cases for the comparison of two pictures f and b.

Figure 4 shows an example picture created the comp algorithm. A terrain is created, a flying saucer was placed in the picture and fog was added.

However due to the fact that using A over B over C and so on was often more convenient than storing the additional z value this approach is not used in praxis in the area of digital compositing, although it had an impact in other areas. There are cameras which also store the depth values which use this approach, however this will not be discussed in this article.

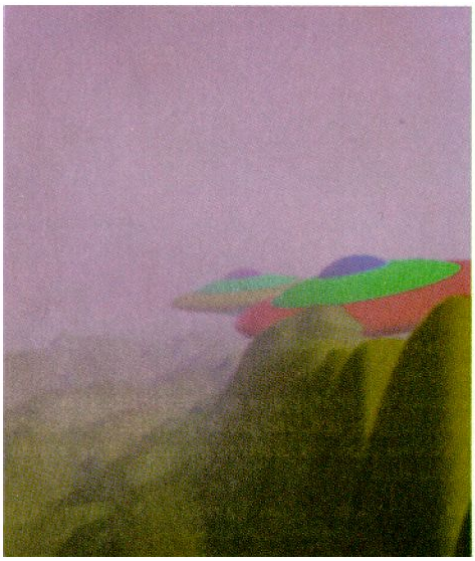


Figure 4: Flying saucer in a terrain behind fog.

3 Matting

So far we discussed how to do the compositing part. However, to do compositing at least a background and a foreground image is needed. While the background part is quite trivial, the foreground must be extracted from a picture first. The problem of separating the foreground image, is called the matting problem. This means the separation of a usally non-rectangular foreground form a background image.

3.1 Bluescreen Matting

In this section bluescreen matting will be discussed. In this case the background is just a constant color, for example blue, which gives the matting process the name bluescreen matting. This technique is widely used by film and video. An everyday seen application of blue screen matting is for example the weather forecast.

Now let us discuss the problem in a more formal way. The color C, with its RGBα representation, at each pixel of the image we want to composite a function of the color C_f for the foreground picture and the color C_b of the newly applied background. All colors C_i consist of the RGBAα representation [R_i G_i B_i α_i].

The foreground element C_f can be interpreted as a composite of a constant colored background C_k and a foreground C_o. C_o represents the object which is intended to be isolated form any background. It is often referred to as the uncomposited foreground color.

Figure 5: A scene from the tv series My name is earl: A blue shirt in weather forecast.

The Matting Problem

Given C_f and C_b at corresponding points, and C_k a known backing color, and assuming $C_f = C_o + (1 - \alpha_o)C_k$, determine C_o which then gives composite color $C = C_o + (1 - \alpha_o)C_b$ at the corresponding point, for all points that C_f and C_b share in common.

Figure 6: The matting problem as stated in Blue screen matting by Smith and Blinn.

In other words C_f is the composite f of C_o, C_k. Let us assume that f is the over function introduced before. We can now formalize the matting problem.

C_o is considered a solution of the matting problem. This uncomposited foreground object is sometimes called an image sprite or just sprite. However the color C_o consists of red, green, blue and alpha values. This leads to four equations. The alpha equation is trivial, which leaves only three equations (for red, green and blue) but four unknowns.

$$R_f = R_o + (1 - \alpha_o) R_k \tag{7}$$

$$G_f = G_o + (1 - \alpha_o) G_k \tag{8}$$

$$B_f = B_o + (1 - \alpha_o) B_k \tag{9}$$

This fact leaves the matting problem with an intrinsic difficulty. Due to this the matting problem theoretically got an infinite number of solutions, while 3 of them will be discussed here.

3.1.1 Solution 1: No Blue

The first solution is called the no blue solution. Assuming that c_o contains no blue color, $c_o = [R_o\ G_o\ 0]$, and the background c_k contains only blue c_k = [0 0 B_k] a new formula for c_f can be found.

$$c_f = c_o + (1 - \alpha_o) c_k = [\, R_o \;\; G_o \;\; (1 - \alpha_o) B_k \,] \tag{10}$$

Solving the equitation for α_o results in

$$C_o = \left[\, R_f \;\; G_f \;\; 0 \;\; 1 - (B_f / B_k) \,\right] \tag{11}$$

However eliminating the blue color from the foreground object reduces the number of available hues by about two-thirds, since all hue which require blue are no longer available.

3.1.2 Solution 2: Gray or Flesh

The second solution which is discussed here is called gray or flesh. Here the assumption is made that c_o is grey, or in other words if the red part R_o or the green part G_o equals the blue color B_o. So there is a exists a solution for the matting problem if R_o or G_o = aB_o + bα_o, and if c_k is pure blue with aB_k + b $\neq 0$. These conditions rewritten in color primary coordinates:

$$c_f = [\, R_o \quad a B_o + b\alpha_o \quad B_o + (1 - \alpha_o) B_k \,] \tag{12}$$

Eliminating B_o from the expressions for G_f and B_f leads to a solution for α_o.

$$C_o = [\, R_f \quad G_f \quad B_\Delta + \alpha_o B_k \quad (G_f - a B_\Delta)/(a B_k + b) \,] \tag{13}$$

With C_Δ = C_f - C_k a very useful definition has been introduced. The special case C_o gray satifies the solution with a = 1 and b = 0 for both R_o and G_o. Therefore science fiction space movies use the blue screen process since the foreground objects in these movies are usally neutrally colored spacecrafts.

Another important foreground element in film and video is flesh, which is usally represented by a color [d 0.5d 0.5d]. This is a non-gray example satisfying the solution.

3.1.3 Solution 3: Triangulation

The last solution discussed here is called triangulation. In difference to the last two solutions for triangulation two different shades of backing color are required and it is assumed that c_o is known against these. Then a complete solution exists which does not require any special information about c_o.

The two shades of the backing color are named B_{k1} and B_{k2}, with B_{k1} = cB_k and B_{k2} = dB_k and c,d lying in the interval [0,1]. Since we assume that c_o is known against the two shades $c_o B_{k1}$ and B_{k2} we can form the following.

$$c_{f1} = [\, R_o \quad G_o \quad B_o + (1 - \alpha_o) B_{k1} \,] \tag{14}$$

$$c_{f2} = [\, R_o \quad G_o \quad B_o + (1 - \alpha_o) B_{k2} \,] \tag{15}$$

By combining the expressions for B_{f1} and B_{f2} we can eliminate the B_o to show that

$$\alpha_o = 1 - (B_{f1} - B_{f2}) / (B_{k1} - B_{k2}) \tag{16}$$

and since the two backing colors are different the denominator is never 0. The following equitations complete the solution.

$$R_o = R_{f1} = R_{f2} \tag{17}$$

$$G_o = G_{f1} = G_{f2} \tag{18}$$

$$B_o = (B_{f2} B_{k1} - B_{f1} B_{k2}) / (B_{k1} - B_{k2}) \tag{19}$$

As mentioned before triangulation requires the foreground object to be shot against two different backgrounds. For nonuniform backing even four passes are required to get a solution. Figure 7 shows steps of triangulation and example pictures.

Another matting approach exists which can handle known, but not constant background. Since this approach demands that the foreground is photographed against a known background there is a difference in the those pixels of the picture where the foreground is, in other words where the α of the foreground is not 0. So knowing where the pixels are different allows the algorithm to determine which pixels belong to the foreground and which are background. Since it is the difference which allows this approach to do the matting, this approach is called difference matting. However this approach leads to sharp edges, resulting in aliasing and as mentioned above demands knowledge of the background.

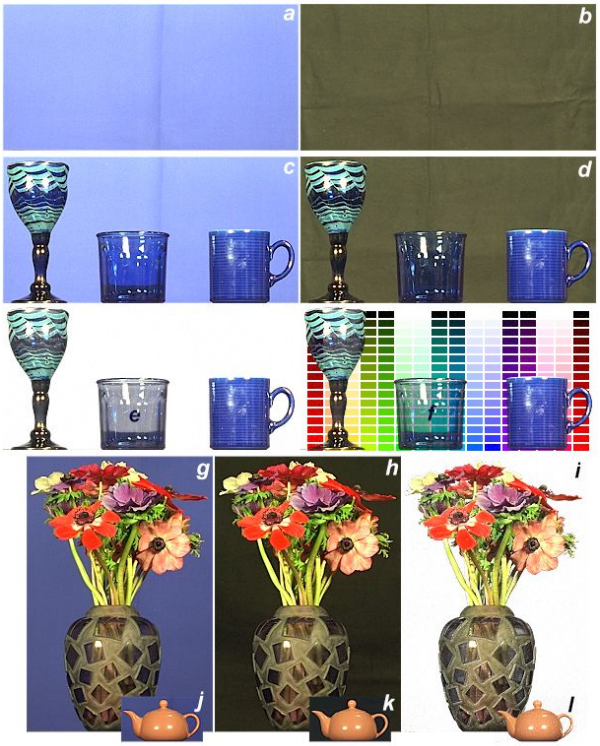


Figure 7: A figure showing practical triangulation matting from Blue screen matting by Smith and Blinn. (a-b) Two different backings. (c-d) Objects against the backings. (e) Pulled. (f) New composite. (g-i) and (j-l) Same triangulation process applied to two other objects (backing shots not shown). (l) Object composited over another.

3.2 Natural Image Matting

The blue screen matting described above is only able to retrieve foreground elements if the background consists of a known color only. However it is not always possible or at least not reasonable to get the desired foreground element in a scene with such a background. This leads to the problem of how to extract the desired foreground element from a natural image. This problem is called natural image matting.

A common approach is to separate the picture in three regions, the one which is definately foreground, the one which is definately background and a third region which is unknown and has yet to be decided. With the picture divided in these three regions the task to be done is to determine whether the pixels in the unknown region are foreground or background. I will briefly introduce two older approaches to solve this problem and after that two more recent which are called bayesian matting and poisson matting in detail.

3.2.1 Mishima Algorithm

First I want to briefly discuss the Mishima algorithm, which is not really an approach for natural image matting. However the algorithm uses samples to form a global distribution which might be interesting since the following algorithms work in a more or less similar way.

Mishima [Mishima] developed a blue screen matting technique based on representative foreground and background samples. The algorithm starts with two identical polyhedral approximations of a sphere in RGB space which are centered at the average value of the background samples. Then the vertices of one of the polyhedra which is considered the background polyhedron are repositioned by moving them along lines radiating from the center until the polyhedron is the smallest possible which still contains all the background samples. The vertices of the other polyhedron, which is considered the foreground polyhedron are adjusted in a similar way to give a polyhedron which is the largest that contains no foreground pixels from the provided sample. Then a new composite color C is casted via a ray from B through C which defines the intersections with the background and foreground polyhedra which are B and F, respectively. The position of C along the line between B and F is the α. See figure 8(e) to make things clearer.

3.2.2 Knockout

The first approach for natural image matting discussed here is the Knockout approach. Here, after the segmentation of the picture in the three regions mentioned above, the next step is to extrapolate the known foreground and background colors into the unknown region. For a point in the unknown region of the picture the foreground F is calculated as a weighted sum of the pixels on the perimeter of the known foreground region. While the weight for the nearest known pixel is set to 1, the weight is reduced linearly by distance and is 0 for pixels which are twice as distant as the nearest known pixel. To initially estimate the background B' the same is done with the nearby known background pixels. Figure 8(b) shows the pixels contributing to the calculation of F and B' for some unknown pixel. B', which is the estimated background color, then gives B by establishing a plane through the estimated background color B' with a normal parallel to the line B'F. After that the pixel color in the unknown region is projected along the normal on that plane, which represents B. After that α is calculated by

$$\alpha = \frac{(f(C) - f(B)}{f(F) - f(B)} \tag{20}$$

The function f projects a color on an axes in rgb space. In figure 8(f) you can see how alpha is computed using the r and g axes.

3.2.3 Ruson and Tomasi

The second approach i want to discuss briefly was made by Ruzon and Tomasi [RusonTomasi]. Again the picture is separated in three regions, the foreground, the background and the unknown region. The first step is to part the unknown region into sub-regions, not only bordering both foreground and background but also including some of them, see Figure 8(c). The foreground and background pixels encompassed by these boxes are regarded as samples of distributions P(F) and P(B), in color space. Then the foreground pixels are split into coherent clusters. These clusters are fitted with Gaussians, which are axis-aligned in color space, with F being the mean and a diagonal covariance matrix $\sum_F$. Then the foreground distribution is treated as a sum of Gaussians. The same is done with the background pixels yielding Gaussians, with B being the mean and a diagonal covariance $\sum_B$. After that the foreground clusters are paired with the background clusters. An example is shown in figure 8(g) for such a pairing. Many of the pairings are rejected based on various intersection and angle criteria.

After these paired Gaussians are constructed, the algorithm indicates that the color C comes from an intermediate distribution P(C), which lies somewhere between the foreground and background distributions. Like for the foreground and background before this intermediate distriubtion is treated as a sum of Gaussians, which have a mean C which lies between, depending on the alpha, the mean values of the foreground and background cluster pairs. The covariance $\sum_C$ is determined in a similar way, again see Figure 8(g). The optimum for the α is the one which causes the maximum probability. After that the F and B, the means of the foreground and background Gaussians, are computed as weighted sums of their cluster means using the pairwise distribution probabilities as weights.

3.2.4 Bayesian Matting

The first approach I want to discuss in detail is the so called bayesian matting. While Knockout and Ruzon-Tomasi already are working algorithms for natural image matting they still fail in various situations, for example if the constrast between foreground and background image is weak. Although there is no perfect solution for all situations the bayesian approach was developed to achieve better solutions overall. Bayesian matting is similar to the Ruzon-Tomasi approach, but differs in some key aspects. For example a MAP is used to optimize α, F and B. Another difference is the use of oriented Gaussian covariances to better modell the color distribution and a sliding window is used to construct neighborhood color distributions.

Like in the algorithms before the picture is separated in a foreground, background and unknown region. The goal for bayesian matting is to solve for a foreground color F, a background color B and an α given the observed color C for each pixel of the picture which lies in the unknown region. Since the colors F, B and C all have three color channels we can derive three equations with seven unknowns.

Like before in the approach of Ruzon and Tomasi [Ruzon and Tomasi], the problem will be solved partly by creating probability

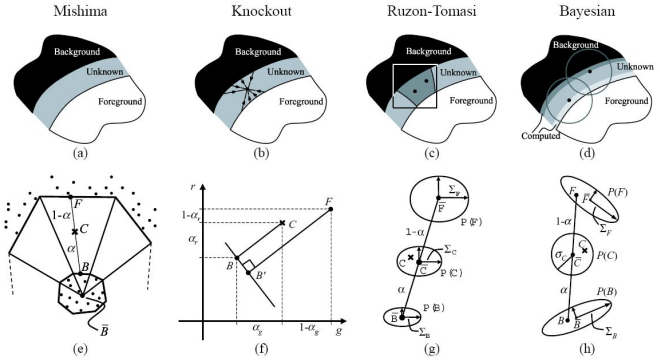


Figure 8: The four algorithm summarized. Each of the algorithms shown in this figure requires some specification of background and foreground pixels. Mishimas algorithm (a) uses these samples to form a global distribution, whereas Knockout (b), Ruzon-Tomasi (c), and the Bayesian approach (d) analyze unknown pixels using local distributions. The dark gray area in (c) corresponds to a segment within the unknown region that will be evaluated using the statistics derived from the square regions overlap with the labeled foreground and background. Figures (e)-(h) show how matte parameters are computed using the Mishima, Knockout, Ruzon-Tomasi, and the Bayesian approach, respectively.

distributions for the foreground and background in a given neighborhood of the pixel. In difference to Ruzon and Tomasi, bayesian matting uses continuously sliding windows for neighborhood definitions, which march inward form the foreground and background regions and, in addition to the values from known regions, make use of nearby computed F, B and α values when constructing oriented Gaussian distributions. This is shown in Figure 8(d). Additionally this approach formulates the problem of computing the parameters of a matte in a well-defined framework, the Bayesian framework. The MAP (maximum a posteriori) technique is used to solve the problem. A MAP estimation tries to find the most likely estimates for F, B and α given the observation C. This can be expressed as a maximization over a probability distribution P and afterwards the Bayes's rule is used to express the result as a maximization over a sum of log likelihoods:

$\arg\max_{F,B,\alpha}$ P(F,B,α —C) =
$\arg\max_{F,B,\alpha}$ P(C—F,B,α) $P(F)P(B)P(\alpha)$ / P(C) =
$\arg\max_{F,B,\alpha}$ L(C—F,B,α) + L(F) + L(B) + L(α)

The function L(x) = log P(x) and P(C) is dropped since it is a constant with respect to the optimization parameters. See Figure 8(h) for an illustration of the distributions which are used to solve the problem of the optimal F, B and α. Now the problem is reduced to defining the logarithms of the likelihoods which are L(C—F,B,α), L(F), L(B) and L(α).

The first term can now be modelled by quantifying the difference between the observed color C and the color which would be predicted by the estimations for F, B and α.

$$L(C|F,B,\alpha) = -\|C - \alpha F - (1-\alpha)B\|^2 / \sigma_C^2 \tag{21}$$

The function L models the error in the measurement of C and corresponds to a Gaussian probability distribution centered at $\overline{C}$ = αF + (1 - α)B with standard derivation σ_C.

To estimate the foreground term L(F) the spatial coherence of the

image is used. The color probability distribution is made using the known and estimated foreground colors within the neighborhood of each pixel. To improve the distribution a weight $w_i = \alpha_i^2 g_i$,with $\sigma = 8$ for the g_i,is applied. The α_i^2 will give more weigth to pixels with high opagueness while the g_i will give more relevance to the pixels which are close than to those further away.

With a given set of foreground colors and their corresponding weights, the colors first are partitioned clusters. For each cluster the weighted mean color F and the weighted covariance matrix F is calculated, using the following equations.

$$\overline{F} = \frac{1}{W} \sum_{i \in N} w_i F_i \tag{22}$$

$$\sum F = \frac{1}{W} \sum_{i \in N} w_i (F_i - \overline{F})(F_i - \overline{F})^T \tag{23}$$

with W = $\sum_{i \in N} w_i$ Using this the L(F), the log likelihoods for the foreground, can be modeled as being derived form an oriented elliptical Gaussian distribution by using the weighted covariance matrix $\sum_F$.

$$L(F) = -(F - \overline{F})^T \sum \overline{F}^{-1} (F - \overline{F})/2 \tag{24}$$

Solving the matting problem for the likelihood for the background L(B) works quiet analogous for natural image matting. The only change, except from the notation of course, is that the weight w_i which was used improve the distribution is now (1 - α_i)$^2 g_i$. For constant or known background colors the problem of course is much easier.

The next step is taking care of the likelihood for the opacity, L(α). Assuming that L(α) is constant we can use the can use the maximization introduced earlier. Because of the multiplications of α with F and B in L(C—F,B,α) the function is not quadratic equation in its unknowns if maximized. In order to solve this equation efficiently the problem is divided in two quadratic sub-problems. For the first sub-problem α is considered constant. Making this assumption and taking the partial derivatives of the maximization with respect to F and B and considering them 0 leads to the following equation.

$$\begin{bmatrix} \Sigma_F^{-1} + I\alpha^2/\sigma_C^2 & I\alpha(1-\alpha)/\sigma_C^2 \\ I\alpha(1-\alpha)/\sigma_C^2 & \Sigma_B^{-1} + I(1-\alpha)^2/\sigma_C^2 \end{bmatrix} \begin{bmatrix} F \\ B \end{bmatrix} = \begin{bmatrix} \Sigma_F^{-1}\overline{F} + C\alpha/\sigma_C^2 \\ \Sigma_B^{-1}\overline{B} + C(1-\alpha)/\sigma_C^2 \end{bmatrix},$$

Here is a 3x3 identity matrix. With α constant, the best parameters F and B can be found solving the 6x6 linear equation.

For the second sub-problem we assume that F and B are constant which leads to a quadratic equation in α This problem can be solved by projecting C, the observed color, on the between F and B in color space.

$$\alpha = \frac{(C-B)\cdot(F-B)}{\|F-B\|^2} \tag{25}$$

With the two sub-problems solved the next task is to optimize the overall maximization by alternatively assuming that either α is constant and using the equation from the first sub-problem or assuming that B and F are constant and using the equation from the second sub-problem. For the start of the optimization α is initialized with the mean α of the nearby pixels. If there are more clusters for foreground and background for each of them pairs have to be made and the pair with the maximum likelihood is chosen.

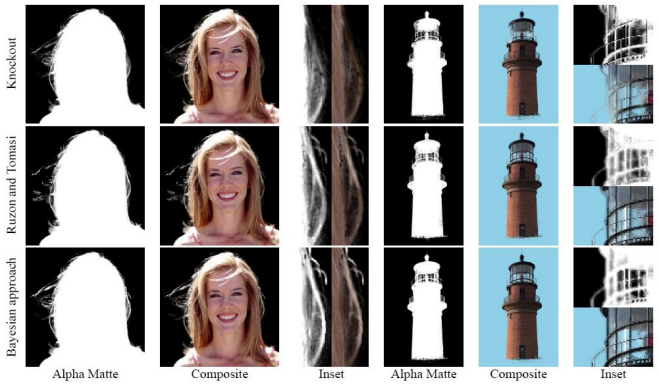


Figure 9: Samples of natural image matting. The insets of these two photographs show both a close-up of the alpha matte and the composite image. For the womans hair, Knockout loses strands in the inset, whereas Ruzon-Tomasi exhibits broken strands on the left and a diagonal color discontinuity on the right, which is enlarged in the inset. Both Knockout and Ruzon-Tomasi suffer from background spill as seen in the lighthouse inset, with Knockout practically losing the railing.

In Figure 5 comparisons of the Knockout, Ruzon and Tomasi and Bayesian methods can be seen for two natural images. There are some missing strands of hair in the close-up for the Knockout results, while the Ruzon and Tomasi result has a discontinuous hair strand on the left side of the image, as well as a color discontinuity near the center of the inset. In the lighthouse example, both, the Knockout as well as Ruzon-Tomasi, show background spill. For example, with Ruzon-Tomasi the background blends through the roof at the top center of the composite inset, while with Knockout the railing around the lighthouse is almost completely lost. The Bayesian results shows none of these artifacts.

3.2.5 Poisson Matting

The last approach i want to discuss is called Poisson matting. Since all approaches mentioned before rely on sampling pixels in the known background and foreground area they all rely on a carefully specified trimap. This leads to the problem that for pictures like in Figure 10 where foreground hairs and background branches might be confused, especially in low contrast regions, these approaches fail. Poisson matting uses a different method based on the gradient of the foreground element and therefore achieves better examples in such pictures.

While the previously discussed approaches optimize the foreground color, background color and the α statistically, this approach introduces methods which operate directly on the gradient of the element which should be retrieved, from now on called a matte. In complex scenes this reduces the error made by mis-classification of the color samples. The gradients of the matte of the image are estimated and afterwards the matte is reconstructed by solving Poisson equations, which leads to the name Poisson matting. For Poisson matting it is assumed that the intensity changes in foreground and background are smooth. The first step is global Poisson matting, while if the

Figure 10: From left to right: a complex natural image for existing matting techniques where the color background is complex, a high quality matte generated by Poisson matting, a composite image with the extracted koala and a constant-color background, and a composite image with the extracted koala and a different background.

quality of the global approach produces not good enough quality for the mattes local Poisson matting is introduced which works on a continuous gradient field in a local region. However user interaction is required to distinguish the image gradients caused by foreground and background colors locally. The information from the user interaction is integrated into Poisson matting with the help of tools which work on the gradient field of the matte. This allows Poisson matting to maintain the continuities of thin long shapes in foreground objects, like the Koala bear in Figure 10.

Generally Poisson matting consists of two steps. The first step is to compute an approximate gradient field of matte from the imput image. The second step is to retrieve the matte from the the gradient field using the Poisson equations.

We receive the approximate gradient field of matte, by taking the partial derivatives on both sides of the matting equation using the formula:

$$\nabla I = (F-B)\nabla\alpha + \alpha\nabla F + (1-\alpha)\nabla B \tag{26}$$

∇ is the gradient operator ($\partial \frac{1}{\partial x}$, $\partial \frac{1}{\partial y}$). This is the differential form of the matting equation, for the RGB channels. If the foreground F and the background B are smooth, for example if $\alpha\nabla F + (1-\alpha)\nabla B$ is small in relation to (F-B)$\nabla\alpha$, the gradient field can be approximated by

$$\nabla\alpha \approx \frac{1}{F-B}\nabla I \tag{27}$$

This means that the matte gradient is proportional to the image gradient. The approximation above was used in [Mitsunaga et al. 1995] to estimate the opacity around the boundaries for a solid object by integrating the gradient of matte along a 1D path perpendicular to the object boundaries. By using this approximation Poisson matting can reconstruct the matte more efficiently by solving Poisson equations in a 2D image space directly.

Global Poisson Matting

As in the other approaches before the image is divided in three regions, definately foreground, definately background and the unknown region. For each pixel p=(x,y) in the image, I_p is the intensity and F_p and B_p are the respective foreground and background intensities of the pixel, while N_p is the set of the four neighbours of the pixel.

$$\partial\Omega = p \in \Omega_F \cup \Omega_B | N_p \cap \Omega \neq \emptyset \tag{28}$$

With $\partial\Omega$ being the exterior boundary of Ω. In order to recover the matte Ω, the unknown region, with (F - B) and the image gradient ∇I given, the following variational problem must be minimized.

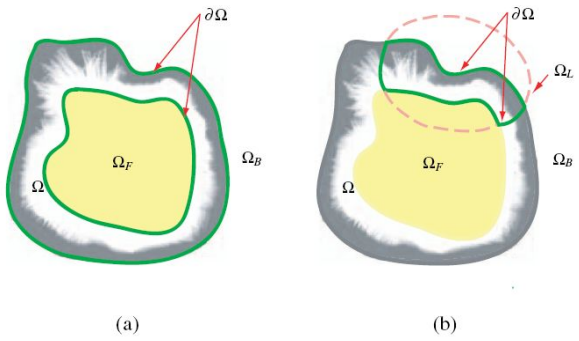


Figure 11: The left picture shows Global Poisson matting, the right describes Local Poisson matting.

$$\alpha^* = arg\min_{\alpha} \int\int_{p\in\Omega} \left\| \nabla\alpha_p - \frac{1}{F_p - B_p}\nabla I_p \right\|^2 dp \tag{29}$$

with Dirichlet boundary condition $\alpha|_{\partial\Omega} = \widehat{\alpha}|_{\partial\Omega}$, with $\widehat{\alpha}|_{\partial\Omega}$ = 1 for p $\in \Omega_F$ and $\widehat{\alpha}|_{\partial\Omega}$ = 0 for p $\in \Omega_B$.

This definition is similar to the supplied trimap which divides the picture in the three regions. The Poisson equations with the same boundary condition is:

$$\Delta\alpha = div(\frac{\nabla I}{F-B}) \tag{30}$$

with $\Delta = (\partial^2 \frac{1}{\partial x^2} + \partial^2 \frac{1}{\partial y^2})$ and div are Laplacian and Divergence operators respectively. In order to obtain the solution for the Poisson equations the Gauss-Seidel iteration with overrelaxation is used ([Perez et al. 2003]). For color images (F-B) and ∇I are measured in the grayscale channel.

The Global Poisson matting is an operative optimization process:

First (F-B) is initialized for each pixel p in the unknown region Ω, by taking into account the nearerst foreground pixel in Ω_F to approximate F_p and the nearest background pixel in Ω_B to approximate B_p. This way a (F-B) version of the image is constructed, which is smoothed by a Gaussian filter to eliminate most of noise and inaccurate estimations.

The second step is reconstructing α by solving the Poisson equation stated above by using (F-B) and ∇I.

The third step is called F and B refinement. We will use the following equation.

$$\Omega_F^+ = \{p \in \Omega | \alpha_p > 0.95, I_p \approx F_p\} \tag{31}$$

The conditions α_p greater than 0.95 and $I_p \approx F_p$ guarantee that the pixels in Ω_F^+ are mostly foreground. The same thing is with the background, however with slight changes.

$$\Omega_B^+ = \{p \in \Omega | \alpha_p < 0.05, I_p \approx B_p\} \tag{32}$$

Here F_p, B_p and I_p represent the color vectors at the pixel p. F_p and B_p are updated corresponding to the color of the nearest pixels

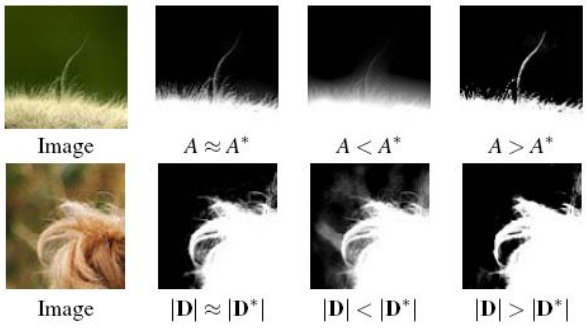


Figure 12: If A approximates A^* good the result is a correct matte. However if A is smaller than A^* the matte will appear smooth and if A is greater than A^* the matte will appear sharp. A similar behaviour can be seen for D.

in $\Omega_F \cup \Omega_F^+$ and $\Omega_B \cup \Omega_B^+$. Like before a Gaussian filter is applied for smoothing.

The second and third steps are iterated until the changes in the results are small enough or Ω_F^+ and Ω_B^+ are empty in the third step. In the usual case only a few iterations are needed.

The global form of Poisson matting gives satisfying results in scenes with both foreground and background being smooth. In complex images however Equation 27 is often a not good enough approximation of the matte gradient. This leads to Local Poisson matting, where the user can engage in the loop to locally refine the results.

Local Poisson Matting

First we bring up a different form of Equation 26:

$$\nabla\alpha = A(\nabla I - D) \tag{33}$$

In this equation A = 1/(F-B) and D = $[\alpha\nabla F + (1-\alpha)\nabla B]$. A affects the gradient scale of the matte, so increasing A the boundaries would be sharpened, while D is the gradient field caused by background and foreground. Because of that A and D must be estimated to approach the ground truth, A^* and D^*. For the global approach discussed before A is automatically estimated from the image and D is assumed to be zero. If the gradients of either background or foreground are strong the mattes achieved by using global Poisson matting are of poor qualitiy.

Now what local Poisson matting actually does is allowing the user to locally manipulate the gradient field. If A is smaller than A^* the matte received as a solution by Poisson matting appears to be smoother than it should be, while a difference in $|D|$ and $|D|$ also results in an erroneous matte. This is illustrated by Figure 12. The key thought for local Poisson matting is that the user can examine the recovered matte and adjust A and D to improve the result.

To do this the user can specify a region Ω_L which is not satisfacting and apply local Poisson matting for that region. Figure 12(b) shows that the integral region became $\Omega_L \cap \Omega$ and the boundary of the new integral region becomes

$$\partial\Omega = \left\{ p \in \overline{(\Omega_L \cap \Omega)} | N_p \cap (\Omega_L \cap \Omega) \neq \emptyset \right\} \tag{34}$$

The user selection Ω_L and the new boundary $\partial\Omega$ are illustrated in figure 12(b). The following equation defines the variational problem to be minimized by local Poisson matting.

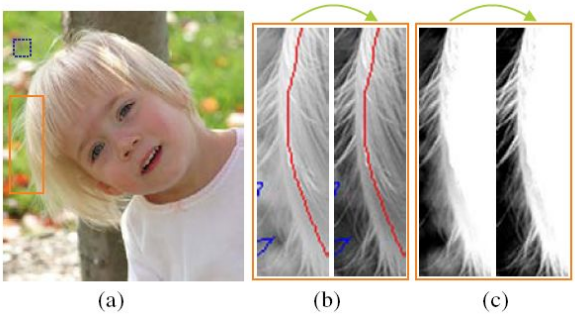


Figure 13: An illustration on how channel selection can improve the quality of the matte.

$$\alpha^* = arg\min_{\alpha} \int\int_{p\in\Omega_L\cap\Omega} \left\| \nabla\alpha_p - A_p(\nabla I_p - D_p) \right\|^2 dp \tag{35}$$

with Dirichlet boundary condition $\alpha|_{\partial\Omega} = \widehat{\alpha}|_{\partial\Omega}$, with $\widehat{\alpha}|_{\partial\Omega}$ = 1 for p $\in \Omega_F$ and $\widehat{\alpha}|_{\partial\Omega}$ = 0 for p $\in \Omega_B$ and $\widehat{\alpha}|_{\partial\Omega} = \alpha_g$ for p $\in \Omega$. Here α_g is the current matte value in an unknown region on the local boundary.

The local region is usally quite small in size (fewer than 200x200 pixels). A Poisson solver will very quickly generate a result for such local regions. Additionally a local operation can be integrated into the matte seamlessly because the boundary conditions exists.

As mentioned before the user can modify A and D in the selected region to receive a better approximation of $\nabla\alpha$. We can distinguish the operations in channel selection and local filtering, with channel selection reducing the error of D while local filtering directly manipulates A and D. The user can use these operations and in that way there is no need to optimize matte pixelwise and the results after a operation has been applied are produced quickly.

The first kind of operation which will be discussed is the so called channel selection. When considering color images, the equation (27) can measured in different channels, one of the three RGB channels or the grayscale channel, which was used in global Poisson matting, $\nabla\alpha = A_g(\nabla I_g - Dg)$. Assuming that foreground and background are smooth makes $|D_g|$ small, which makes $A_g\nabla I_g$ a good approxaimation for $\nabla\alpha$. In a similar way it is intended to construct a new channel γ = aR + bG + cB with $|D_\gamma|$ is smaller than $|D_g|$. Therefore the variance of he foreground or background is minimized and the new channel γ is constructed in two steps:

In step one the user select a background or a foreground color sample (R_i, G_i, B_i) in the image. In the second step the weights of a, b and c are computed to minimize the sample variances in the channel. This leads to the linearly constrained quadratic optimization problem

$$\min_{a,b,c} \sum_i \left[(abc)\cdot(R_iG_iB_i)^T - (abc)\cdot\left(\overline{RGB}\right)^T \right]^2 \tag{36}$$

with a + b + c = 1 and $(\overline{RGB})$ being the mean color value of the samples. The weights (a b c) are obtained by solving an augmented linear system [P. Gill and Wright 1981]. The operation is illustrated in figure 13. The error of D is reduced and therefore the quality of the hair shape recovered is better.

As mentioned before there is a second kind of operations which directly operate on A and D, this is called local filtering. There are a number of local filters a user can use to manipulate the matte gradient field.

The first one is called boosting brush. If the matting result is too sharp or too smooth the user can use this filter to either decrease or increase A. This filter has a local Gaussian shape for each pixel p in the area it is applied. For A_p we receive a modified A'$_p$ after using the boosting brush filter on it.

$$A'_p = \left[1 + \gamma \exp(-\frac{\|p - p_0\|^2}{2\sigma^2})\right] \cdot A_p \tag{37}$$

with p_0 being the coordinate fo the brush center and σ and γ are parameters the user can define to control size and strength of the boosting effect. Therefore the user can boost the desired partial region, or even the whole region, by using brushes of the needed sizes. If γ ¿ 0 the filter will increase A around the brush center and if its ¡ 0 it will decrease it.

The next filter to be discussed is the highpass filter. Since channel selection generates smooth foreground or background this leads to low frequency gradients. Using this D can be estimated by using the low frequency part of the image gradient

$$D = K * \nabla I \tag{38}$$

with K being a Gaussian filter $N(p;p_0,\sigma^2)$ centered at p_0 and $*$ being the convolution operator.

The third filter is the diffusion filter. Since on the boundaries of solid objects the alpha matte changes quickly, $A\nabla I$ is already a good approxiamtion, but the gradient ∇I is sensitive to noise and blocking effects from JPEG images. Anisotropic diffusion [Perona and Malik. 1990] is used to diffuse the image. It is a blurring process, which preserves edges, that removes small scale noise. Afterwards the image gradient ∇I is re-computed from the diffused image.

The last filter mentioned here is the clone brush. Some situations may occur in which it is intended to copy the matte gradient $A(\nabla I - D)$ of a selected source to a target region via the clone brush. It can be seen as a copy paste method.

Figure 14 shows samples for the four filters:

The local operations mentioned above can be used by the user to refine the matte in the selected region. The global Poisson matting results as basis, the local ones are proceeded in four steps:

1. Channel selection is applied to reduce the errors in D. The diffusion filter is applied to remove possible noise for solid object boundaries.
2. The highpass filter is applied to obtain an approximation of D.
3. The boosting brush is applied to manipulate A.
4. The clone brush may be applied if gradients are indistinguishable.

There are to more brushes, an erase and an inverse brush, which may be used optionally. The generation of results is made very quickly at each step, so the user can observe the results and select the regions which do not satisfy him.

At last some some samples to compare Poisson matting with other approaches.

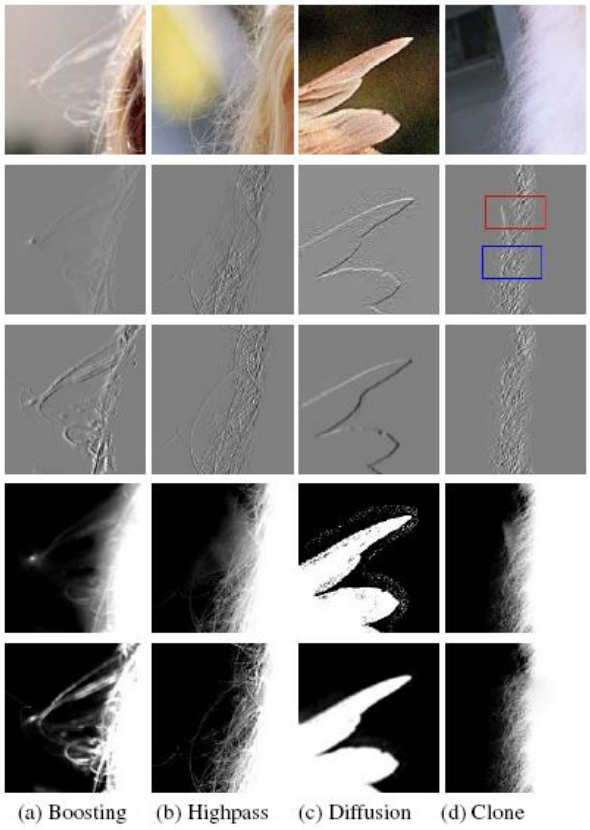


Figure 14: The first column shows the boosting brush procuding a sharper matte. The second shows a highpass filter recovering structures of the matte and the third one shows how the diffusion filter removes noise. The last one shows how the clone brush is used to copy paste the matte gradient from the red to the blue region.

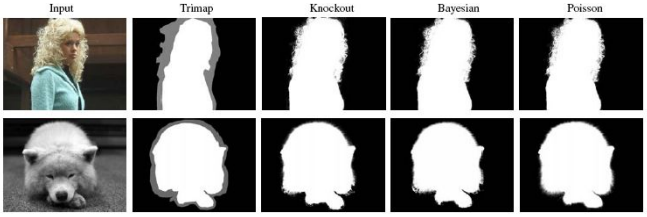


Figure 15: The performance of different matting techniques in comparison.

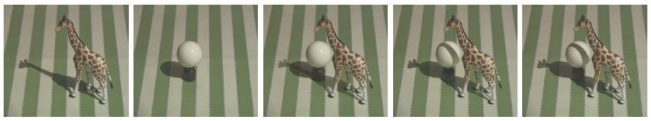


Figure 16: The first and the second picture are the foreground and background scene. The third picture shows the problem that occurs if shadow are considered as normal elements, while the fourth shows the result from shadow matting and compositing. The last picture is the real photograph, which just looks like the result of the fourth.

3.3 Shadow Matting and Compositing

The next problem with matting and compositing i want to discuss are shadows. Since shadows are an important part in the human perception of the world they should also be considered when compositing two images. If shadows are missing in a composited picture it just will not look realistic. It would be possible just to matte the shadows like other elements and inserting them in the composite. However since shadows can interact with the background and maybe other elements this will not be sufficient to get a realistic result. Therefore a method is needed to provide correct shadows for composited images. Such a method was introduced in [Chuang et al. 2003] and i will discuss this method here in detail.

3.3.1 Shadow Matting

The first step would be to retrieve, or matte, the shadows from an image. Since the ordinary compositing equation by Porter and Duff is not sufficient for shadow matting, a new equation, the so called shadow compositing equation has to be introduced. To determine an appropriate model it is assumed that a single point light source, sometimes called the key light, is the only lightsource that casts shadows in the scene. Secondary lighting is considered to be either dim or of such wide area that its shadows do not have to be considered relevant. Additionally we assume that there are no interreflections. This leads to the model of the observed color C for a pixel

$$C = S + \beta I \tag{39}$$

with S being the shadowed color, β the visibility of the lightsource and I the reflected contribution of the light source. If we assume that L is the color of the pixel without shadows we can substitute I by L - S, which leads to the shadow compositing equation:

$$C = \beta L + ((1-\beta)S \tag{40}$$

From this equation we can interpret that L is the unshadowed image, S is the shadowed image and β is the shadow matte, which represents the per-pixel visibility of the light source. β can be seen similar to α in the original composition equation. While β may be transferred itno different scene L and S are dependent on the lighting conditions, so for a new image L' and S' are required as the new lit and shadowed images.

Now how can the shadow matte β can be recovered with the observed color C given. First L, the lit image, and S, the shadowed image, must be estimated. This can be done using max and min compositing [Szeliski et al. 2000], which generally means to find the darkest and the brightest value of each pixel. First the video matting algorithm, Chuang et al. [2002], is used to extract the mattes of foreground objects and exclude them from the max/min compositing. For each pixel

$$S = min_f C_f \tag{41}$$

and

$$L = max_f C_f \tag{42}$$

is computed with min and max being independently computed across all frames. The color images C, L and S can be seen as 3-vectors at each pixel. The shadow matte β can be estimated as

$$\beta = \frac{(C-S)\cdot(L-S)}{\|L-S\|^2} \tag{43}$$

The observed color C can be seen as lying on a line between L and S computing the parametric distance of the projection along that line. If the estimations for L ans S are good this method works well, however if L and S get more similar the estimates for β become noisy. If they are similar this means that we can not recover β, but also states that the pixel is not effected by the shadow, since L = S, so the pixel is not required anyway.

3.3.2 Shadow Compositing

Now that we have separated the shadow matte from the input picture we can use this information to produce a composite picture. In order to do shadow compositing however first L', the lit image of the new background scene, and S', the new shadowed image are required. Assuming that the same camera and light source is used for the new background as for the old, the new composite color can be calculated using the following equation.

$$C' = \beta L' + (1-\beta)S' \tag{44}$$

While it is relatively easy to render scenes lit and shadowed for synthetic scenes, for natural scenes photosymetric shadow scanning must be done by moving an object between the light such that every part of the scene needed for compositing is shadowed once. As previously described the max/min compositing operations are used to recover the lit and shadow images of the scene. However the shadow cast by this method does not yet consider the geometry of the new background scene, as you can see in figure 17(c), which leads us to the next step.

3.3.3 Estimating Shadow Deformations

Since we want the shadows in the composited image to look realistic the geometry of the background scene of the compisited image must be taken into consideration. First we assume that there is a region in the target background that is planar to the source background, in other words a reference on the height within the image. Then we need to construct a displacement or warping map W which places each pixel p in the target image in correspondence with a point $W_{(p)}$ in the target image.

The shadow composition equation defined before can be used, with the exception that we now have a warped shadow matte:

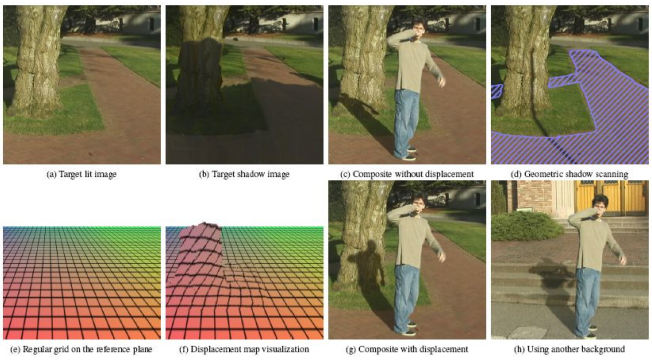


Figure 17: The whole process of shadow compositing: Picture a) and b) show the lit and shadow images recovered for the target geometry. In picture c) the result without considering the geometry can be seen. Picture d) to f) show the process of displacement and g) the result with displacement. Another example with a different background is shown in h).

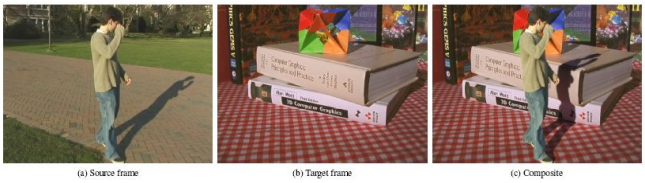


Figure 18: Another sample for a composite with correct shadows.

$$\beta' = \beta \left[W_{[p]} \right] \tag{45}$$

The method for estimating the shadow displacement map is based on Bouguet's shadow scanning approach [1998]. Shadow scanning is an active illuminatiom method for which the reconstruction cotains gaps where surfaces are occluded from the view of the lightsource or the camera. However these gaps are exactly what is needed for shadow compositing. The process used for shadow compositing is called geometric shadow scanning.

First it is required that the target background has planar regions, which are specified by the user, see 17(d). This region is called the reference plane region and the plane defined as π. Imagine a pixel p from which the point P in the target background can be seen. For P there is a projection Q on the reference plane that lies at the intersection of π and the shadow ray V which passed through P. Q can be computed as the intersection between two shadow lines, caused by a thin object (a stick) that casts a shadow on P, on the reference plane. With q being the image-space projection of Q the problem can be solved directly, since it is in fact the warping map $W_{[p]}$, by doing all computations in image coordinates.

3.4 Environment Matting

There are two more effects which increase realism in an image that are missing in ordinary compositions, refraction, by transparent objects, and reflection, by shiny objects. Additionally scattering effects may occur in the case of glossy materials. In this section i will discuss the process called environment matting and compositing, introduced in [Zongker et al. 1999], which expand traditional matting and compositing processes by the above mentioned effects.

Figure 19: A glass of water composited on two different backgrounds.

Environment matting and compositing is aimed to capture, additionally to the foreground object, a description of how the object refracts and reflects light from the scene. This information is called the environment matte. Using this environment matte good results for refraction and reflection of objects in the new scene can be produced. The following figure gives an example what environment matting and compositing is capable of.

3.4.1 The Environment Matte

We will start by expanding the traditional compositing equation

$$C = F + (1-\alpha)B \tag{46}$$

by the environment matte, which defines how the foreground element interacts with light in the environment. Although α refers to the coverage of pixel and the opacity as well in the traditional compositing equation, we assume that it will only refer to coverage now, so for a pixel with no foreground element α is zero and if it is completely covered α is 1, no matter whether the the element is semitransparent or opaque at that pixel. In a similar way an elements color is handled differently than in traditional compositing. An elements color characterized any emissive component of the foreground object and any reflections coming from lightsources in the scene, but not any additional reflections or transmission of light from the rest of the environment.

The environment matte, which captures any transmissive effects of the foreground element is added to the traditional equation.

$$C = F + (1-\alpha)B + \Phi \tag{47}$$

with Φ representing the contribution of light from the environment that refracts through or reflects from the foreground element.

There are two requirements the environment matte should meet. The first one is that compositing should be relatively fast, therefore the environments are represented by sets of texture maps, just like in environment mapping. The second requirement is that the representation should only require a small, constant amount of data for each environment map.

First we assume that only light coming from distant parts of the scene reaches the foreground object. Using this a simplified model of light transport can be created. An environment can be described as light $E(\omega)$ coming from directions ω. The total amount of light Φ can be described as following:

$$\Phi = \int\int R(\omega \to p)E(\omega)d\omega dp \tag{48}$$

This means that the total amount of light emanating form a portion f of the foreground element that is visible through a given pixel can be described as an integral over f of all light from the environment that contributes to the point p in the pixel, attenuated by the reflectance function $R(\omega \to p)$. The reflectance function describes the effect of all absorption and scattering by the foreground element.

The first approximation made is that the reflectance function is constant across the covered area of a pixel.

$$\Phi = \int R(\omega)E(\omega)d\omega \tag{49}$$

The next step is to break up the whole integral to a sum of m integrals representing the different parts of the environment, which means their texture maps which represent the light coming from those parts.

$$\Phi = \sum_{i=1}^{m} \int R_i(x)T_i(x)dx \tag{50}$$

The integral is taken over the whole area of the texture map for each texture map and $R_i(x)$ is the reflectance function for a point x on the texture map T_i.

The last assumption made is that the contribution of a texture map T_i can be approximated by a constant K_i multiplied by the total amount of light emanated by an axis-aligned rectangular region A_i of the texture map. The operator M(T,A) returns the average value of this rectangle, what leads to:

$$\Phi = \sum_{(i=1)}^{m} K_i \int_{A_i} T_i(x)dx = \sum_{(i=1)}^{m} K_i A_i M(T_i,A_i) = \sum_{(i=1)}^{m} R_i M(T_i,A_i) \tag{51}$$

Inserting the approximation for Φ in the first Environment Matting Equation leads to the overall Environment Matting Equation.

$$C = F + (1-\alpha)B + \sum_{(i=1)}^{m} R_i M(T_i,A_i) \tag{52}$$

3.4.2 Environment Matting

The next question is how we can do environment matting, which means extracting the foreground and background with the pixel coverage α as well as the environment matte Φ for each pixel. The idea is to capture how the collection of all rays passing through a pixel are scattered into the environment, some kind of backward ray tracing. As shown in Figure 20 some different patterned textures, the backdrops and sidedrops are displayed behind and at the sides of the foreground object. For each backdrop one image without the foreground, the reference image, and one image with the foreground, the object image, is created. This gives a non-linear optimization problem which solution is a set of parameters which are most consistent with all the image data. In order to split the problem of finding a rectangle into finding two one-dimensional intervals the patterns used in the textures vary in only one dimension.

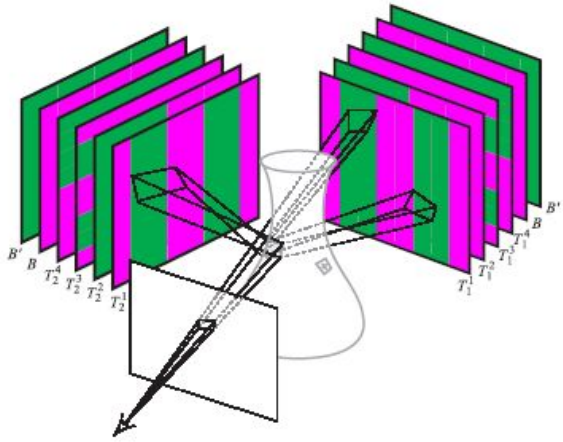


Figure 20: The environment matting process.

The colors green and magenta are used because they are orthogonal in RGB space and have a similar luminance. Additionally to the patterns two textures must be one colored.

A difficulty is the large dimensionality of the Environment Matting problem. Since we there are three color components there are three degrees of freedom for the foreground color F and for each R_i, the reflectance coefficient, in Φ. Additionally there are four degrees of freedom for each area extent A_i and one more for α. This fact requires that the problem is solved in four stages. In the first stage only the backdrop is considered and different backdrops are used to compute a coarse estimate for α. After that F and R_1 are determined for the pixels which are covered by the foreground element. The third stage is to solve for A_1 and make a finer estimate for α at the boundary of the foreground element. In the last step R_i and A_i are determined for other faces of the environment, the sidedrops.

Stage 1 - estimate the coverage

The first step is to make a coarse estimate of coverage for each pixel. We start by separating the pixels of the environment matte in two classes, the covered, which means the reference and object images differ, and the uncovered. The next step is clean up the resulting alpha channel by removing isolated covered or uncovered pixels, what is done with a morphological operations (a close and a open box), a operation used in image processing.

The uncovered pixels are considered the background, having an α of 0. The covered are separated into foreground, $\alpha = 1$, and boundary pixels having a fractional α.

Stage 2 - foreground color and reflectance coefficients

Now that we know the pixel which are considered covered the color of the foreground F and the reflectance coefficients R_i for the environment must be determined. Here the one colored, or solid, backdrops are needed, by which F and R_1 can be determined.

Imagine two colors B and C with B being the color of a backdrop and C being the color of the foreground before this backdrop. We use a second backdrop with the color B' and get the color C' which is the color of the foreground of the second backdrop B'. By inserting these four colors in the Environment matting equation we receive the two equations

$$C = F + (1-\alpha)B + R_1 B \tag{53}$$

and

$$C' = F + (1-\alpha)B' + R_1 B' \tag{54}$$

which are tow equations with two unknowns, which can be solved by replacing R_1 and F by functions of α. This leads to

$$R_1(\alpha) = \frac{C - C'}{B - B'} - (1-\alpha) \tag{55}$$

and

$$F(\alpha) = C - (1 - \alpha + R_1)B \tag{56}$$

Stage 3 - the area extents and a refined estimate of coverage

Now that R_1 and F have been determined, we can go on to the third stage, where a refined alpha for the boundary pixels and the axis-aligned rectangle A_1 which approximates the reflections and refraction of the background in the best way are determined. To do this the objective function over all backdrops for each covered pixel is made

$$E_1 = \sum_{j=1}^{n} \left\| C^j - F(\alpha) - (1-\alpha)B^j - R_1(\alpha)M(T_1^j, A_1) \right\|^2 \tag{57}$$

with B^j and C^j being the colors of the pixel in the referenced and object image in the j-th backdrop, T_1^j being the texture map of the j-th pattern. $F(\alpha)$ and R_1^α are of course the functions from the last stage. What we try to do is find the rectangular A_1 which minimizes this function.

There are still four degrees of freedom for the rectangle, namely left, right, top, bottom, or (l,r,t,b). Since we stated that the patterns in the backdrops are only one dimensional we know that l and r are independent for horizontally striped backgrounds and t and b are independent for vertically striped. Knowing this we can break down the problem to two three dimensional problems, since we must consider α, which are (α,l,r) and (α,t,b). Since we can assume α being 1 in the foreground pixels, we are looking for an interval [l,r] which minimizes E_1 over the vertically striped patterns by testing a number of candidates and we do the same thing for [t,b] and the horizontal patterns. At the border area where α is between 0 and 1 we need to test for multiple α values of the interval A_1.

Stage 4 - the sidedrops

Now that we have acquired A_1 we know the object refracts and reflects light from the backdrop, now we must capture the same information for the sidedrops. We already know α and F as well and B_1 as the color of the backdrop, while S and S' are considered two solid colors of the sidedrop. This leads to the following equations:

$$C = F + (1-\alpha)B_1 + R_i S \tag{58}$$

$$C' = F + (1-\alpha)B_1 + R_i S' \tag{59}$$

Using these two equations we can extract $R_i by$

$$R_i = \frac{C - C'}{S - S'} \tag{60}$$

Figure 21: The first column shows the composited pictures without considering the environment, while the second shows the results using environment matte compositing. The last pictures in the rows are real photographs.

what leads to the objective function E_i

$$E_i = \sum_{j=1}^{n} \left\| C^j - F(\alpha) - (1-\alpha)B_1 - R_i(\alpha)M(T_i^j, A_i) \right\|^2 \tag{61}$$

to minimize for each sidedrop i. Due to the fact that the sidedrops are not visible to the camera we cannot take reference photographs for the sidedrops and use the ones from the backdrop as well as the texture maps. By placing the sidedrops around the object we can simulate light coming from these directions.

3.4.3 Environment compositing

Now that we have an environment matte for our foreground object it can be composited in a new environment with regard to the refractions and reflections. We just need to implement the Environment Matting Equation, equation (52), which is called environment compositing. The contributions of the foreground and background color as well as all weighted contribution from the texture maps which describe the environment are considered. In order to prevent aliasing a filter can be applied. The next figure will show some samples made with environment compositing.

4 User interfaces

In this chapter i will discuss what kind of software concerning digital compositing exists. Basically there are two different workflows, the node-based and layer-based, for digital compositing.

Node-based means that the composite is represented as a tree, with the nodes being images or effects. This is the way compositing applications are handled internally. An example is the software Nuke.

Layer-based means that each image or effect in a composited image is represented as a layer within a timeline. They are just stacked, one above the next, in the order the user desires, with the bottom layer rendered first. Layer-based compositing works well for motion graphics and simple compositing projects, but it is problematic to use it for more complex composites. An example for layer-based software would be Adobe After Effects.

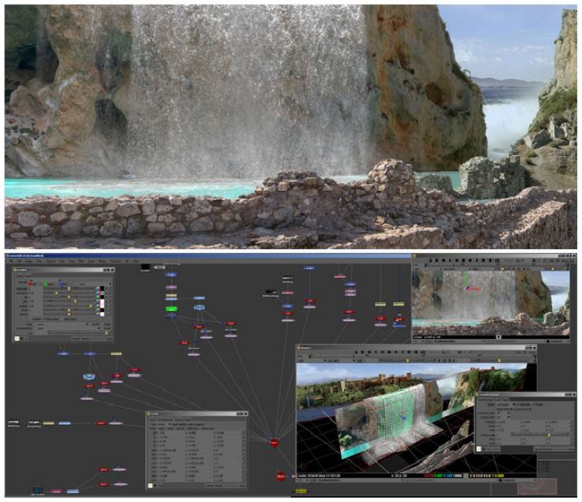


Figure 22: The user interface of nuke, a node-based digital compositing software.

Adobe After Effects is also capable of natural image matting. Although it is not explicitely mentioned which algorithm is used, the matting has problems with low contrast. Since other software shows the same problems it seems that methods like bayesian or poisson matting is not being used in commercial software so far.

Additionally, at least for my knowledge, no known digital compositing software is able to do real shadow or environment matting. There are two ways the known software handles these problems. While the first is just ignoring it, most programs offer a list of tools and effects to edit the images, so a skilled user may hide these deficiencies. In any case shadow and environment matting is not yet implemented in any known software by now.

5 Conclusion

In this paper the background behind digital compositing and the associated matting processes has been shown. We started with the basic idea of compositing of two images using the alpha channel to prevent aliasing. The next step was to show how matting is actually done, since usally elements which we want to composite must be retrieved first. Here we separted in techniques which require a certain background, for example a blue screen, and techniques which retrieve elements from natural images, like bayesian and poisson matting. Afterwards we looked into techniques to improve the realism of composited images. The first one was shadow matting and compositing which showed up a way to create realistic looking shadows in composited images. Afterwards we looked into environment matting and compositing what is a technique that tries to consider refractions and reflections when compositing images in order to improve realism.

References

BLINN, J. F., AND NEWELL, M. E. 1976. Texture and reflection in computer generated images. *Commun. ACM 19*, 10, 542–547.

CHUANG, Y.-Y. New models and methods for matting and compositing.

CHUANG, Y., CURLESS, B., SALESIN, D., AND SZELISKI, R. A bayesian approach to digital matting.

CHUANG, Y.-Y., ZONGKER, D. E., HINDORFF, J., CURLESS, B., SALESIN, D. H., AND SZELISKI, R. 2000. Environment matting extensions: towards higher accuracy and real-time capture. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 121–130.

CHUANG, Y.-Y., AGARWALA, A., CURLESS, B., SALESIN, D. H., AND SZELISKI, R. 2002. Video matting of complex scenes. In *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, ACM, New York, NY, USA, 243–248.

CHUANG, Y.-Y., GOLDMAN, D. B., CURLESS, B., SALESIN, D. H., AND SZELISKI, R. 2003. Shadow matting and compositing. In *SIGGRAPH '03: ACM SIGGRAPH 2003 Papers*, ACM, New York, NY, USA, 494–500.

CROW, F. C. 1984. Summed-area tables for texture mapping. In *SIGGRAPH '84: Proceedings of the 11th annual conference on Computer graphics and interactive techniques*, ACM, New York, NY, USA, 207–212.

DUFF, T. 1985. Compositing 3-d rendered images. In *SIGGRAPH '85: Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, ACM, New York, NY, USA, 41–44.

GREENE, N. 1986. Environment mapping and other applications of world projections. *IEEE Comput. Graph. Appl. 6*, 11, 21–29.

MATSUSHITA, Y., KANG, S. B., LIN, S., SHUM, H.-Y., AND TONG, X. 2002. Lighting interpolation by shadow morphing using intrinsic lumigraphs. In *PG '02: Proceedings of the 10th Pacific Conference on Computer Graphics and Applications*, IEEE Computer Society, Washington, DC, USA, 58.

PORTER, T., AND DUFF, T. 1984. Compositing digital images. In *SIGGRAPH '84: Proceedings of the 11th annual conference on Computer graphics and interactive techniques*, ACM, New York, NY, USA, 253–259.

RUZON, M., AND TOMASI, C. Alpha estimation in natural images. 18–25.

SMITH, A. R., AND BLINN, J. F. 1996. Blue screen matting. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, ACM, New York, NY, USA, 259–268.

SUN, J., JIA, J., TANG, C.-K., AND SHUM, H.-Y. 2004. Poisson matting. In *SIGGRAPH '04: ACM SIGGRAPH 2004 Papers*, ACM, New York, NY, USA, 315–321.

WU, T.-P., TANG, C.-K., BROWN, M. S., AND SHUM, H.-Y. e 07. Natural shadow matting. *ACM Trans. Graph. 26*, 2, 8.

ZONGKER, D. E., WERNER, D. M., CURLESS, B., AND SALESIN, D. H. 1999. Environment matting and compositing. In *SIGGRAPH '99: Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 205–214.