IEEE Conference on Computer Vision and Pattern Recognition 2000, Volume 1, pages 18--25, June 2000.

Alpha Estimation in Natural Images

Mark A. Ruzon
Carlo Tomasi
Computer Science Department
Stanford University
Stanford, CA 94305
{ruzon,tomasi}@cs.stanford.edu
Color Images: http://vision.stanford.edu/~ruzon/alpha/

Abstract

*Many boundaries between objects in the world project onto curves in an image. However, boundaries involving natural objects (*e.g.*, trees, hair, water, smoke) are often unworkable under this model because many pixels receive light from more than one object. We propose a technique for estimating alpha, the proportion in which two colors mix to produce a color at the boundary. The technique extends blue screen matting to backgrounds that have almost arbitrary color distributions, though coarse knowledge of the boundary's location is required. Results show a number of different objects moved from one image to another while maintaining naturalism.*

1. Introduction

The popularity of image-based rendering techniques has led to increased interest in extracting objects from one image to be placed in another. When boundaries are in focus and are well modeled by a set of edges meeting at corners, a reasonable effect can be obtained by cutting and pasting followed by a smoothing operation along the boundary. In many cases, however, this approach is insufficient.

Natural objects are particularly difficult to extract because parts of the object can exist at scales below the resolution of the camera. Leaves and strands of hair are often much smaller than one pixel, resulting in large numbers of pixels receiving light from more than one object. The output of an edge detector or a segmentation algorithm is usually a poor representation of such boundaries.

We examine a model in which a pixel belongs to two regions whenever its color was formed by light directly reflecting from two separate objects. The colors and the relative amounts of each are determined by examining the colors of nearby pixels that receive light from only one object. The percentage that an object contributes to the color of a pixel is referred to as its *alpha value*.

Of course, we have little hope of finding out in general whether a pixel is gathering light from one object or two. Since traditional edge models break down, we require additional input in the form of a segmentation of an image into regions that are definitely an object versus regions that contain a boundary. These boundary regions need not conform exactly to a boundary because extra pixels can be reclassified as belonging to an object.

The following sections explain the details surrounding this procedure: the history and rationale of previous approaches, the constraints on specifying boundary and object regions, and the estimation of alpha values and the "unmixed" colors that formed the color of a boundary pixel. Results demonstrate the algorithm's effectiveness on boundaries that cannot otherwise be adequately captured.

2. The history of alpha estimation

The concept of alpha was invented separately by three different communities: computer graphics, film and television, and remote sensing. The assumptions, methods, and applications of the three groups are quite different, however, so it is worthwhile to examine them separately.

The original application of alpha in graphics was for *soft filling*, or changing the color of an antialiased region such as an edge. Fishkin and Barsky [2] published the most comprehensive technique for soft filling when alpha values were unknown but the original foreground and background colors were known exactly. The color of an edge pixel was presumed to fall into the vector subspace in color space spanned by the original colors. This technique works for up to four colors between the two objects [3].

Mitsunaga *et al.* [4] developed a more robust system for estimating alpha which assumed that the gradient of alpha across a boundary is proportional to the multidimensional gradient magnitude. Projecting image gradient vectors onto a reference vector connecting the average colors of the foreground and background increased the signal-to-noise ratio. Objects such as hair and water, however, do not follow this assumption.

For a long time now the film and television industry has used *blue screen matting* to perform image-based rendering. An actor could be filmed against a blue or other color screen, after which a matte could be extracted and the blue screen replaced. Smith and Blinn [7] analyzed the matting problem in great detail, noting that a unique solution exists only for the easiest cases. They showed that a unique solution can be found in the general case if the foreground object is filmed against two backgrounds that differ in every pixel. However, this approach is useful only in studios where non-moving objects can be photographed twice. They provided bounds on alpha for the general case.

Finally, the remote sensing community has long been interested in unmixing pixels, because each pixel from a satellite image receives light from many different materials. A simple yet effective implementation of this idea came from Adams *et al.* [1], who deduced the composition of rocks and soil in an image of the Martian surface and estimated the amount of each everywhere in the image while also accounting for illumination effects. Such techniques usually involve much information not present in the image, however, such as laboratory reference spectra of materials, heuristics for ranking candidates, and other analyses of the data. The problem we consider uses only the image data.

3. Object and boundary regions

Since the algorithm we present here is more a tool than a system, the user must specify more than the input image. We restrict our attention to images in which there are only two regions, foreground and background, since an image with multiple objects can be decomposed one at a time. Images must be partitioned into not two but three regions, the third being the boundary region. This section details two alternatives for specifying these regions using the tree example in Figure 1.

A chain of pixels separating the two objects can be dilated to form a boundary region. This chain can be constructed using the edges found by an edge detector, the boundary found by a region segmentation algorithm, or a hand-drawn boundary using a paint program or a boundary-finding tool such as Intelligent Scissors [5].

Also, a paint program can be used to specify parts of the image as "pure," or consisting only of pixels belonging to one of the two objects. In our example magenta pixels mark the sky and yellow pixels delineate the tree (the two colors should not exist in the image already, of course).

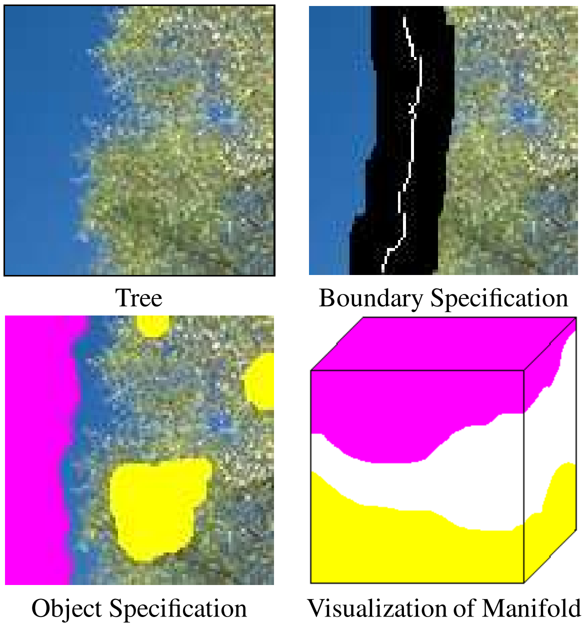


Figure 1. Computing alpha values along a boundary requires specifying the object or boundary regions.

Two methods are necessary because there are many images where one is preferable to the other. The object specification method is superior in this example because there are many pixels well inside the outer boundary of the tree that contain blue. The boundary specification method is advantageous in images where the colors of one or both objects change as we move along the boundary, however. If we have a pixel chain we can form many pairs of local color distributions instead of one global pair that may not be well separated in color space.

The purpose of specifying object and boundary regions is twofold: it labels each pixel for proper use in the computation, and it partitions color space as well. The final illustration of Figure 1 is a conceptualization of the colors of the tree and sky mapped into color space. Pixels from the boundary region lying between these two regions of color space are assigned fractional alpha values. Thus, the specification of the boundary region need not be precise so long as it actually contains the boundary, and so long as it leaves enough of a color in an object that it can be represented by one or more clusters.

4. Estimating alpha

Our algorithm for alpha estimation for the most basic cases is similar to the algorithms of Fishkin and Barsky. The complexities come from the fact that we have noise

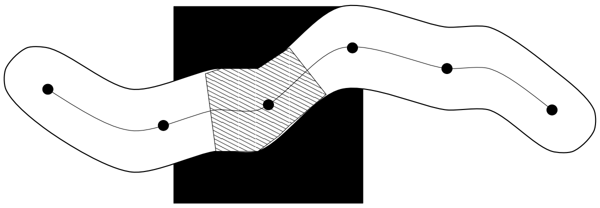


Figure 2. **An example of the roles of anchor points. All boundary pixels in the shaded region have their alpha values computed from the two distributions of pixels in the square.**

and other sources of variance in the data, potentially many more than four colors, and unmixed colors that need not correspond to modes of the color distributions. This section describes the mechanisms for dealing with this complexity.

4.1. Building a manifold in color space

Alpha values are measured along a manifold connecting the "frontiers" of each object's color distribution. Each distribution is represented as a set of point masses found through vector quantization, for which we use Orchard and Bouman's binary split algorithm [6] and the CIE-Lab color space [8]. We denote the two distributions as $X = \{(x_j, \mathbf{u}_j, \sigma^2_{\mathbf{u}_j})\}$ ($j = 1, \ldots, M$) and $Y = \{(y_k, \mathbf{v}_k, \sigma^2_{\mathbf{v}_k})\}$ ($k = 1, \ldots, N$), where the x_j's and y_k's are percentages of each color specified by $\mathbf{u}_j$ and $\mathbf{v}_k$ respectively. The variance of each cluster is recorded as well.

One manifold is constructed for each pair of distributions. If the boundary region was initially specified, we must decide which pixels to use for each pair of distributions. We divide the chain of pixels into intervals, defining the endpoints of the intervals as *anchor points*. The length of each interval is equal to three times the amount of dilation performed. Since different parts of the chain can be dilated by different amounts, the length of the intervals can also differ. The anchor points serve two purposes: (1) they become the center of a window defining the pixels in each object region that will form the local color distributions, and (2) they divide the boundary region into pieces through a Voronoi diagram. Each piece uses the color distributions specified by the corresponding anchor point for computing alpha. Figure 2 illustrates these two functions.

The set of line segments connecting one point mass from each signature can be represented as the Cartesian product $\{1, \ldots, M\} \times \{1, \ldots, N\}$. A subset of this product is used to create a flow that constructs the manifold between the signatures. This flow maximizes the number of line segments on which a nonzero amount of mass is transported. Doing so assures us that as many boundary region pixels as possible are near a line segment, increasing the accuracy. It is important that the segments do not conflict, however, or ambiguity will enter the computation.

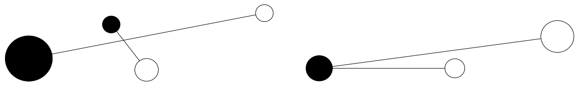


Figure 3. Conflicts between line segments cause ambiguities in the computation. We always choose the shorter segment.

Figure 3 illustrates two types of conflicts, "intersection" and "angle." A pixel near the intersection of two line segments could be a combination of either pair of colors with very different alpha values, so we must reject the longer segment. In three dimensions, two random line segments never intersect, so we declare an intersection conflict whenever the minimum distance between any pair of points from each segment is below a threshold (set at 5 CIE-Lab units). Intersection conflicts can occur only when the two segments do not share an endpoint.

Angle conflicts, on the other hand, can occur only when the two segments do share an endpoint. When the angle between two segments is small (less than $10°$), the three clusters are almost collinear. This is another source of ambiguity, noticed previously by Smith and Blinn. Again, we choose the smaller segment as being more likely.

Figure 4 shows a sample manifold. It is produced by a greedy algorithm that adds segments to the set of accepted segments in order of increasing length if they have no angle or intersection conflicts with segments already in the set. Note that it is possible that one or more clusters will not be an endpoint of any accepted line segment; any such clusters are excluded from the rest of the computation, and the x_j's or y_k's are renormalized. The n line segments chosen help define the manifold. Each segment is represented as an ordered pair (j, k), and we define two functions, $J(i) = j$ and $K(i) = k$, that return the index into X and Y, respectively, of the clusters marking the endpoints of the segment.

4.2. Computing alpha and unmixed colors

The two signatures X and Y serve as discrete representations of the colors from each object region. We must now form a relationship between these two distributions and an arbitrary pixel Q in color space. This task can be accomplished more naturally if we convert the color signatures to continuous probability distributions. We use a mixture of isotropic Gaussians, ensuring a simple formulation and nonzero responses at all points in color space. Comparing these two distributions at Q would be one way to estimate

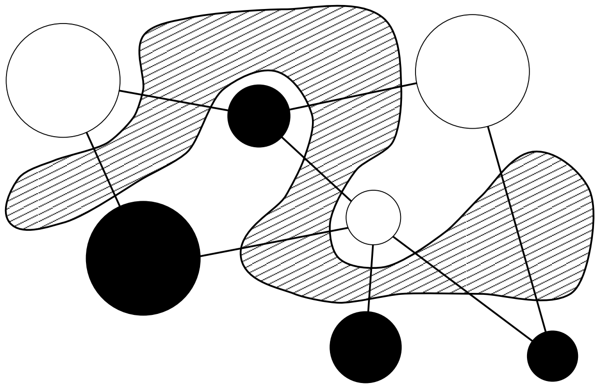


Figure 4. An example of a manifold. The line segments are those accepted by the algorithm, and pixels along a boundary should lie in the shaded region between the two distributions represented by point masses.

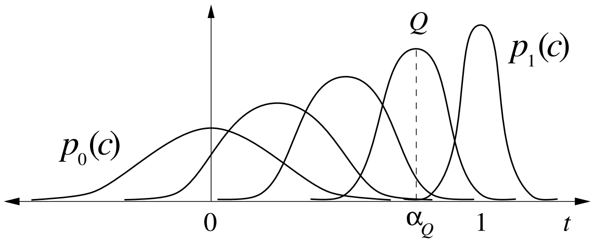


Figure 5. A 1-D example of interpolation. As t varies, the mean and variance of a Gaussian interpolates between $p_0(\mathbf{c})$ and $p_1(\mathbf{c})$. The value of t that maximizes the value at Q is the alpha value α_Q.

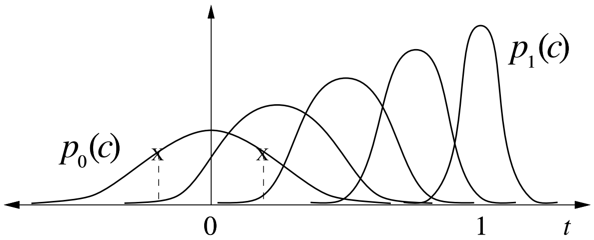


Figure 6. The two X's are equally likely to be drawn from $p_0(\mathbf{c})$, but these colors will have different alpha values if we do not give special treatment to the values 0 and 1.

α_Q, the alpha value of Q.

However, we presume Q to be a mixture of colors from X and Y, and so the probability of Q under either of these distributions is likely to be numerically meaningless. Traditional methods from decision theory are unhelpful because we are estimating the amount of mixture. We argue that Q is actually drawn from a probability distribution formed as the colors of X are "morphing" into the colors of Y across the boundary. This morphing can be modeled as an interpolation between the two probability distributions. Estimating alpha becomes a maximum likelihood estimation problem: find the interpolated probability density that maximizes the value at Q.

We start by defining a function $f(t)$ that produces a probability distribution for every value of t:

$$f(t) = p_t(\mathbf{c}),\ 0 \leq t \leq 1\ ,$$

where $\mathbf{c}$ is an arbitrary color vector. The values at $f(0)$ and $f(1)$ are the probability densities corresponding to X and Y, respectively. We model these distributions as mixtures of n isotropic Gaussians (denoted $G_i(\mathbf{c}; \mu_i, \sigma_i^2)$), each being interpolated along one of the line segments defining the manifold. The distributions for $t = 0, 1$ can be written as:

$$\begin{aligned} p_0(\mathbf{c}) &= \textstyle\sum_{i=1}^n a_i G_i(\mathbf{c}; \mathbf{u}_{J(i)}, \sigma^2_{\mathbf{u}_{J(i)}})\ , \\ p_1(\mathbf{c}) &= \textstyle\sum_{i=1}^n a_i G_i(\mathbf{c}; \mathbf{v}_{K(i)}, \sigma^2_{\mathbf{v}_{K(i)}})\ , \end{aligned}$$

where a_i is the amplitude of each Gaussian. Each a_i is proportional to the product $x_{J(i)} \cdot y_{K(i)}$, and the sum of the a_i's is normalized to 1. For any intermediate value of t, the distribution produced by $f(t)$ interpolates the mean and variance of the Gaussians:

$$p_t(\mathbf{c}) = \textstyle\sum_{i=1}^n a_i G_i(\mathbf{c}; \mu_i(t), \sigma_i^2(t))\ , \text{ where}$$

$$\begin{aligned} \mu_i(t) &= (1-t)\mathbf{u}_{J(i)} + t\mathbf{v}_{K(i)} \text{ and} \\ \sigma_i^2(t) &= (1-t)\sigma^2_{\mathbf{u}_{J(i)}} + t\sigma^2_{\mathbf{v}_{K(i)}}\ . \end{aligned}$$

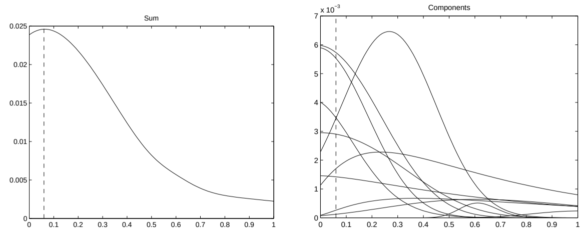
Computing alpha values becomes straightforward:

$$\alpha_Q = \arg\max_t f(t)|_Q\ .$$

In practice we discretize t at a resolution of 0.01 and evaluate Q for each set of Gaussians produced. Figure 5 shows a one-dimensional example of two unimodal distributions and the interpolated Gaussians.

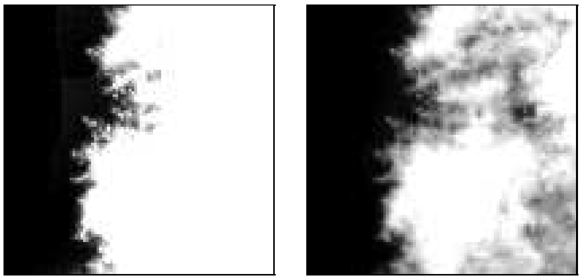
The sole exception to this rule is when Q appears to have been drawn from $p_0(\mathbf{c})$ or $p_1(\mathbf{c})$; Figure 6 illustrates this case. Both pixels marked with X's are equally likely to have been drawn from $p_0(\mathbf{c})$, but the directions of the noise vectors are different, resulting in two different alpha values. If $f(0)|_Q$ or $f(1)|_Q$ is above a threshold, the alpha value of Q is set to 0 or 1 accordingly.

We must also find the unmixed color from each object. These colors are not independent, because Q must lie on the line segment connecting them. We use the weights provided by each Gaussian component of the distribution to estimate the unmixed color of each side, followed by a



(a) $f(t)|_Q$ (b) Components of $f(t)|_Q$

Figure 7. The value α_Q is found by maximizing $f(t)|_Q$, which is the sum of many components, each corresponding to one interpolated Gaussian. The values of the component functions at α_Q (0.06 in this case) are used to compute two weighted averages to find the unmixed colors.



Boundary Specification Alpha Values | Object Specification Alpha Values

Figure 8. As predicted, the object specification results for the tree are more satisfying.

small perturbation so that Q properly divides this new line segment.

Figure 7(a) shows a typical function consisting of the values at Q as t varies. This function is a sum of many smooth functions, each corresponding to one Gaussian as shown in Figure 7(b). The component functions are not Gaussians because the variance is a function of t. At the maximum, each of the n Gaussians contributes a weight w_i to $f(\alpha_Q)|_Q$. By mapping each weight back to the endpoints of the segment it came from, we form a weighted average of the colors of each signature to estimate the unmixed colors:

$$\hat{\mathbf{u}}_Q = \frac{\sum_{i=1}^{n} w_i \mathbf{u}_i}{\sum_{i=1}^{n} w_i} \ , \ \ \hat{\mathbf{v}}_Q = \frac{\sum_{i=1}^{n} w_i \mathbf{v}_i}{\sum_{i=1}^{n} w_i} \ .$$

Normally, no more than a few nearby Gaussians significantly influence the unmixed color of a pixel.

If we denote the point on the line formed by $\hat{\mathbf{u}}_Q$ and $\hat{\mathbf{v}}_Q$ that divides it in the ratio $\alpha : 1 - \alpha$ as Q', we can compute the final colors $\mathbf{u}_Q$ and $\mathbf{v}_Q$ by perturbing $\hat{\mathbf{u}}_Q$ and $\hat{\mathbf{v}}_Q$ by

Figure 9. A tree branch is extracted from a tree background.

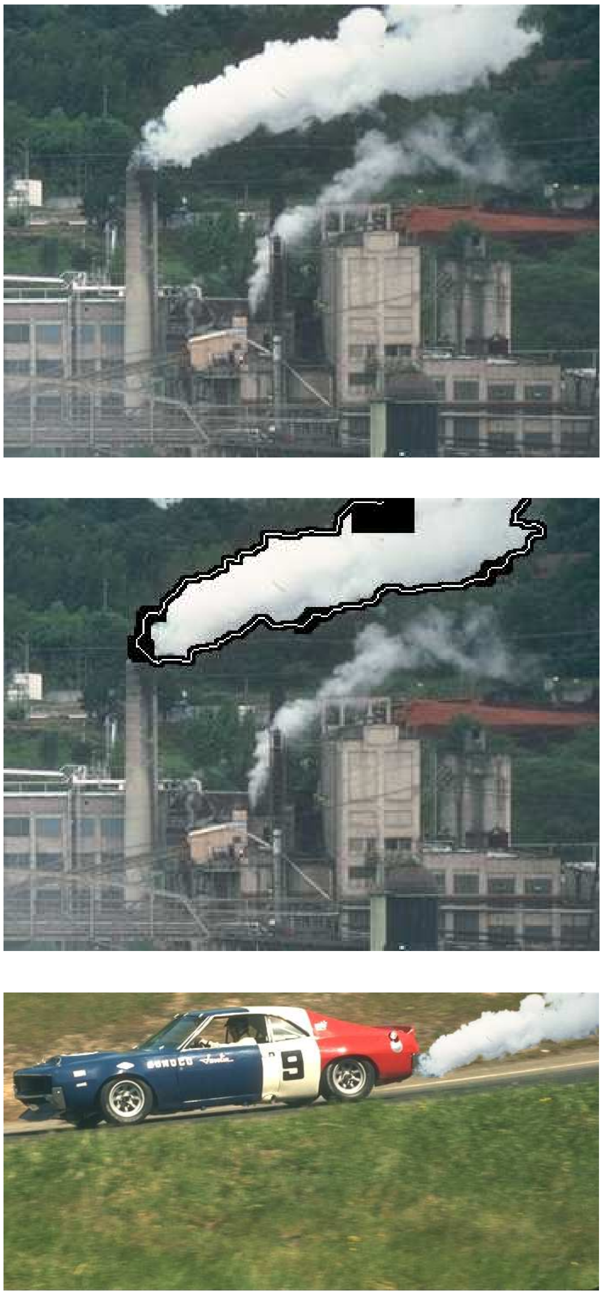


Figure 10. A plume of smoke causes a racing car to fail its emissions test.

the vector $\overrightarrow{Q'Q}$. The original image can now be faithfully reconstructed.

5. Results

This section shows a set of examples on a variety of natural objects. Using both methods of specifying object and boundary regions, we create local or global color distributions of object regions to estimate the alpha values of pixels in the boundary region. The signatures we create have no more than 5 clusters each. An object is moved to a destination image by combining unmixed colors with the corresponding pixel colors from the new background according to the computed alpha values.

We follow up Figure 1 by displaying alpha values for the tree using both object and boundary specification (see Figure 8). This example is essentially blue screen matting. As predicted, boundary specification cannot retrieve the blue from the middle of the tree. The result using object specification is more natural.

Of course, the algorithm is designed to handle more interesting backgrounds than sky. Figure 9 also shows a tree branch, but the background consists of trees. A conservative specification of the two object regions is enough to recover most of the branch. The twigs connecting the leaves are dark, and so they are lost, and some of the highlights from the background are brought into the foreground, but overall the new rendering retains naturalism.

Figure 10 displays a plume of smoke extracted using the boundary specification method. Since there is another plume in this image, specifying object regions would produce poor results. The boundary is extracted using Intelligent Scissors and dilated by different amounts to capture, among other things, the hole in the plume near one endpoint. The other plume does not have deleterious effects on the result.

Figure 11 shows how a non-convex waterfall can still be extracted. The interior region between the two halves of the waterfall is successfully removed. The colors are well localized spatially, but the algorithm creates global color distributions that are unable to take advantage of this fact.

The final example, Figure 12, is of a woman whose hair is being blown about by the wind. The boundary region has been specified using Intelligent Scissors and dilated by different amounts. The riverbank in the background has similar colors to the hair, so the boundary must be narrow in those areas to minimize artifacts. Mao's picture shows through her hair in the final image, but not every strand of hair is recovered.

6. Conclusions

The problem of extracting an image region from the background has no general solution. The film and video

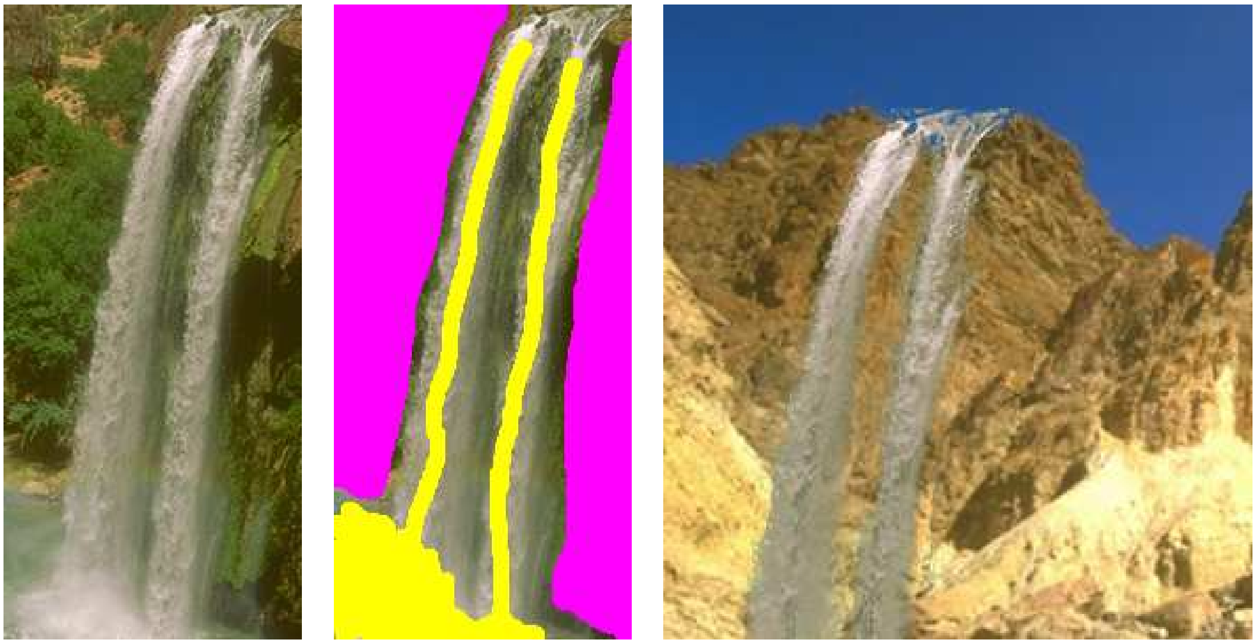


Figure 11. A waterfall is transported to arid Death Valley.

industries use equipment that provides manual control over the extraction process, but only when the background is a constant color. In contrast, we have presented a tool for extracting image regions from almost arbitrary backgrounds. It requires enough knowledge of a boundary's location to estimate the color distributions of the two image regions accurately, and the current implementation does not allow for much human intervention.

The results show that foreground objects can be moved to new images without appearing counterfeit (with the exception of illumination changes). A close examination shows some defects because the colors are not well separated, or because the color representation is not accurate enough. The second can be solved by increasing the size of the color signatures, but the first is still unapproachable.

When the boundary specification method can be used, the algorithm produces more accurate results and ensures that any errors are local. For many boundary regions, however, the topology makes this impossible. Specifying object regions directly solves this problem, but only when the color distributions are spatially uniform over the image. Preventing distributions from sharing colors is more difficult in this case, and a method of providing local correspondence for arbitrary region topologies would be helpful.

The fact that we are excluding colors in objects that are not close to the frontiers of the color distributions has the practical effect of producing biases in the alpha values. Angle conflicts in particular force us to overestimate values with respect to the signature with two of the three collinear clusters. Using information about the spatial distribution and relative amounts of colors might allow us to choose one color or the other at random.

Finally, we reiterate that this algorithm is indeed a tool, not a system. Without reliable user input as to the location of the boundary, the algorithm cannot succeed unless the boundary can be extracted through edge detection or segmentation algorithms, an unlikely prospect for the types of boundaries that benefit most from this algorithm. Nevertheless, it expands the power of image extraction techniques.

References

[1] J. Adams, M. Smith, and P. Johnson. Spectral mixture modeling: A new analysis of rock and soil types at the Viking 1 lander site. *J. of Geophys. Res.*, 91(B8):8098–8112, Jul. 1986.

[2] K. Fishkin and B. Barsky. A family of new algorithms for soft filling. *Comp. Graph.*, 18(3):235–244, Jul. 1984.

[3] J. Foley, A. van Dam, S. Feiner, and J. Hughes. *Computer Graphics: Principles and Practice*. Addison-Wesley, Reading, MA, 1990.

[4] T. Mitsunaga, T. Yokoyama, and T. Totsuka. AutoKey: Human assisted key extraction. In *SIGGRAPH95*, pages 265–272, Aug. 1995.

[5] E. Mortensen and W. Barrett. Interactive segmentation with intelligent scissors. *GMIP*, 60:349–384, Sep. 1998.

[6] M. Orchard and C. Bouman. Color quantization of images. *IEEE Trans. on Sig. Proc.*, 39(12):2677–2690, Dec. 1991.

[7] A. Smith and J. Blinn. Blue screen matting. In *SIGGRAPH96*, pages 259–268, Aug. 1996.

[8] G. Wyszecki and W. Stiles. *Color Science: Concepts and Methods, Quantitative Data and Formulae*. John Wiley and Sons, New York, NY, 1982.

Figure 12. A woman takes an instantaneous vacation to Beijing.