Layer extraction from multiple images containing reflections and transparency

Richard Szeliski, Shai Avidan, P. Anandan

Microsoft Research,
One Microsoft Way,
Redmond, WA 98052, USA,

E-mail: {szeliski,avidan,anandan}@microsoft.com

Abstract

Many natural images contain reflections and transparency, i.e., they contain mixtures of reflected and transmitted light. When viewed from a moving camera, these appear as the superposition of component layer images moving relative to each other. The problem of multiple motion recovery has been previously studied by a number of researchers. However, no one has yet demonstrated how to accurately recover the component images themselves. In this paper we develop an optimal approach to recovering layer images and their associated motions from an arbitrary number of composite images. We develop two different techniques for estimating the component layer images given known motion estimates. The first approach uses constrained least squares to recover the layer images. The second approach iteratively refines lower and upper bounds on the layer images using two novel compositing operations, namely minimum- and maximum-composites of aligned images. We combine these layer extraction techniques with a dominant motion estimator and a subsequent motion refinement stage. This results in a completely automated system that recovers transparent images and motions from a collection of input images.

1 Introduction

Reflections and transparency are about as ubiquitous as images themselves. Many natural images will typically contain one or both, i.e., contain mixtures of reflected and transmitted light. For example, any shiny or glass-like surface will create a reflected image of other surfaces in its immediate environment. Also, surfaces like glass and water are (at least partially) transparent, and hence will transmit the light from the surfaces behind it[1]. Thus, many natural images are composed of reflected and transmitted images which are super-imposed on each other. When viewed from a moving camera, these component *layer* images appear to move relative to each other.

The reflection and transmission of light on surfaces in visual images has been carefully studied in physics-based vision [15, 13]. Likewise, a number of techniques for recovering multiple motions from image sequences have been developed [16, 6, 12, 5, 8, 18, 11, 10, 14, 19, 3]. These techniques can recover multiple motions even in the presence of reflections and transparency. Some of these techniques also extract the individual component layer image from the input composite sequence [18, 10, 14, 19, 3], but only in the *absence* of reflections and transparency – i.e., all the layers must be opaque.[2] The detection of transparency in single images has been studied by [2, 1], but neither of these studies provides a complete algorithm for layer extraction from general images. Thus, the extraction of component layers images in the presence of reflections and transparency remains an open problem.

In this paper, we develop an optimal approach to recovering layer images and their associated motions from an arbitrary number of composite images. We develop two different techniques for estimating the component layer images given known motion estimates. The first approach uses constrained least squares estimation to optimally recover the layer images. The second approach iteratively refines lower and upper bounds on the layer images using two novel compositing operations, namely minimum- and maximum-compositing of aligned images. We combine these layer extraction techniques with a dominant motion estimator and a subsequent motion refinement stage. This results in a completely automated system that recovers transparent images and motions from a collection of input images.

The remainder of this paper is organized as follows. Section 2 presents the general problem formulation, including

[1] The transmitted light is usually attenuated to some degree by the glass (or frontal surface). Hence, the notion of partial transparency or "translucency" is more general. However, following common usage in the field, we will use the term "transparency" to indicate both complete transparency and translucency.

[2] Note that for the purpose of locking onto each component motion, [11] actually creates a "reconstructed" image of each layer through temporal integration. However, these fall short a being a proper extraction of the component layers, since the other layers are not fully removed, but rather appear as blurred streaks.

the image formation equations. Section 3 presents the constrained least squares algorithm we use to recover the component images. Section 4 presents our novel algorithm for estimating upper and lower bounds on the solution using min- and max-composites. Section 5 presents our motion refinement algorithm. Section 6 summarizes the overall layer extraction system. Section 7 presents our experiments with real image sequences. Finally, we close with a discussion of our results and ideas for future work in this area.

2 Problem formulation

In [2] Adelson and Anandan proposed the following recursive process as the generative model for obtaining a composite image from component layers. At each pixel, assuming a given spatial ordering of layers relative to the viewpoint, each layer partially attenuates the total amount of light coming from all the layers "behind it" and adds its own light to give an output signal. The final composite image is the result of applying this process to all layers in a back to front fashion. This process can be summarized in terms of the following modified form of the *over* operator used in image compositing [7],

$$F \wedge B \;\equiv\; F + (1 - \alpha_F)B, \tag{1}$$

where F and B denote the colors of the foreground and the background images[3].

For the purposes of this paper, we assume that each component layer (indexed by $l = 0, \ldots L-1$) is defined by a signal or 2D image $f_l(x)$, (we will use x to index both 1-D signals and 2-D images), which is warped to the current image (indexed by k) coordinate system via a warping operator W_{kl}, which resamples the pixels. Let $W_{kl} \circ f_l$ denote the warped image. Then, the composite image is given by the equation

$$m_k \;=\; W_{k0} \circ f_0 \wedge \cdots W_{k(L-1)} \circ f_{L-1}. \tag{2}$$

We assume in this paper that W_{kl} is an *invertible* global parametric motion, such as translation, rotation, affine, or perspective warp. For now, we also assume that the W_{kl} are known (we will remove this assumption in Sections 5 and 6).

In this paper, we restrict our attention to the problem of pure additive mixing of images

$$m_k(x) = \sum_{l=0}^{L-1} W_{kl} \circ f_l(x). \tag{3}$$

This corresponds to portions of the scene where pure reflection/transmission is occurring, e.g., windows or glass in picture frames. An alternative way of writing the image formation equations is to look at the discrete image pixels written in (rasterized) vector form,

$$\mathbf{m}_k = \sum_{l=0}^{L-1} \mathbf{W}_{kl}\mathbf{f}_l. \tag{4}$$

[3]The standard definition of the over operator uses foreground colors that are *premultiplied* by the opacities of the foreground layer; hence, the R, G, and B values that must be $\leq \alpha$. In our case, this restriction is removed in areas of reflection, in order to handle additive composition.

This formula is equivalent to the first (continuous) formula if the images are sampled without aliasing (below their Nyquist frequency) and the warping does not unduly compress the layer images (thereby causing aliasing). The $\mathbf{W}_{kl}$ matrices are very sparse, with only a few non-zero coefficients in each row (i.e., the interpolation coefficients for a given pixel sample).

In addition to the image formation equations, we also know that the original layer images must be non-negative, i.e., $f_l(x) \geq 0$. As we will see shortly, this provides very important (and useful) constraints on the solution space.

Since we are working with real images, there is also a good chance that the images may be *saturated* (i.e., $m_k(x) = 255$ for 8-bit images) in some regions. For a truly accurate model of the mixing process, we really need to be working with photometrically calibrated camera, i.e., cameras where the radience to pixel-value transfer curve is known [9]. For this paper, however, we will assume that the mixing process is truly linear, but that the observed mixed signal values $\mathbf{m}_k$ are clipped to 255. The extension to a truly calibrated camera is straightforward, but may require a level-dependent noise process to be added.

3 Constrained least squares

Given a set of images $\mathbf{m}_k$, how do we recover the layer images $\mathbf{f}_l$? Since the image formation equations are linear, constrained least squares,

$$\min \sum_k \| \sum_{l=0}^{L-1} \mathbf{W}_{kl}\mathbf{f}_l - \mathbf{m}_k\|^2 \;\text{ s.t. }\; \mathbf{f}_l \geq 0, \tag{5}$$

is a good choice. Such a least squares estimator is statistically optimal if the measured input images are corrupted by uniform independent (white) Gaussian noise. The least squares problem is constrained, since we require that all of the elements in the $\mathbf{f}_l$ images be non-negative. Also, for any pixel in $\mathbf{m}_k$ that is saturated (255), we only penalize the mismatch between $\mathbf{m}_k$ and the mixed layers if the predicted value is *below* 255.

This least squares problem is very large (one term or linear equation per measured input pixel), and very sparse (only a few non-zero coefficients per equation). Iterative techniques, such as variants of gradient descent or conjugate gradient, will therefore have to be used.

For our current implementation, we have used a two stage approach for solving the constrained least-squares problem. We first solve the problem without constraints using a Preconditioned Conjugate Gradient method (using standard MATLAB function). Using this as an initial estimate, we then use a Quadratic Programming algorithm with the positivity constraints enabled (again using a standard MATLAB function) to obtain the constrained optimal solution.

3.1 Uniqueness of a solution

The positivity constraints on the component signals (images) restrict the solution to be in a *convex* subspace. Therefore, the quadratic programming program posed in Equation 5 does not

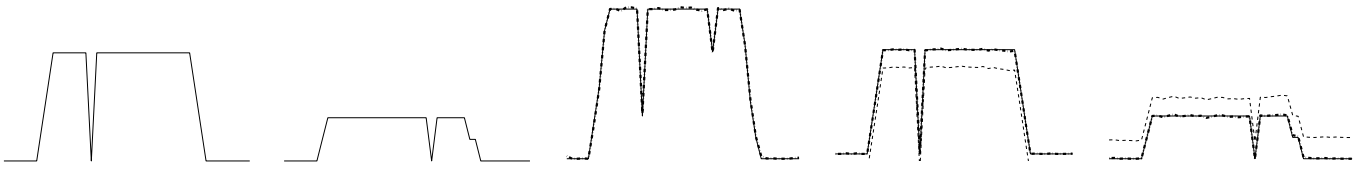


Figure 1. 1D example of constrained least squares: (a) background signal, (b) foreground signal, (c) noisy mixed signal, (d) and (e) the reconstructed background and foreground signals. The solid curves represent the input data, the thin dashed curves show the results of solving the least-squares problem without the positivity constraints, and the thick dash-dot curves show the final result after incorporating the constraints.

suffer from multiple local minima. However, is the solution unique?

It is easy to show that (without the constraints) the solution is not unique. We can see this even without analyzing the particular structure of the $\mathbf{W}_{kl}$ matrices, based on the following reasoning. Let $\{\hat{f}_l\}$ be a set of component layer signals (images) that minimizes the least-squares error defined in Equation 5. Since each input image is simply a sum of warped and resampled versions of these components, we can subtract a constant image from one of the layers and distribute (add) this amount among the other layers without changing the sum. The new set of layer thus obtained is also a valid solution to the *unconstrained* minimization problem posed in Equation 5. This implies that the system of equations is degenerate.

Figure 1 illustrates this degeneracy, using a one-dimensional example. Figures 1a and 1b show the plots of the two input component layers. Five mixed signals were created by shifting these two relative to each other by different (known) amounts and adding random Gaussian noise. As an example, one of these five mixed signals is shown in Figure 1c. The thin dashed curves in Figures 1d and 1e show the recovered component layers signals obtained by solving the unconstrained least-squares problem, using a "pseudo-inverse" (minimum norm) technique. Note that the recovered signal (in thin dashed curve) is offset from the true signal. (Similar results are obtained in the noise-free case as well.) For the two layer case, it is easy to show that the amount of this offset is equal to half the difference between the mean foreground and background layers values.

In practice, this degeneracy is not too critical, since it simply leads to a DC offset of the signals. Moreover, if each layer has at least one pixel that is black (i.e., signal value of zero), this degeneracy can be removed using the positivity constraint. This is easy to see, because subtracting an offset from any of the layers will lead to at least one negative valued pixel, which violates the positivity constraint. The result of solving the constrained least-square problem is shown as thick dash-dot curves in Figures 1d and 1e. Observe that these reconstructed signals differ from the input signals only by small random noise. In other words, solving the optimization problem with constraints appears to fix the degeneracy in the system. It should be noted, however, if there is some layer that has no black pixel (i.e., $f_l \geq c$, where $c > 0$), the solution can only be determined up to an offset of c.

In practice, in 2D images, there may also be additional source of degeneracy or poor conditioning due to the structure of the warping matrix $\mathbf{W}_{kl}$. Consider the case when the relative motion between the component layers consists of shifts purely in the horizontal (or vertical) direction. In this case, the overall problem decouples into a set of independent problems corresponding to each row (or column). Each row will be determined only up to an arbitrary but different DC offset. To impose the positivity constraint and obtain a unique solution, each row in each layer must have a black pixel, which may be unrealistic. Hence, even the use of the positivity constraints may not guarantee the correct recovery of component layers.

This is illustrated using a synthetic example in Figure 2. Figures 2a and 2b show two input component layer images that were shifted relative to each other by different (known) amounts to produce a set of composite images, which were then used as the input data for our least-squares problem. A small amount of random noise was added to each composite image. Figure 2c shows one example of such a composite image.

We first conducted an experiment in which all the relative shifts were purely in the horizontal direction. The reconstructed images obtained by solving the constrained least-squares problem is shown in Figures 2d and 2e. Note the faint horizontal streaks in the reconstructed images. To highlight these streaks, in Figures 2f and 2g, we show the difference images obtained by subtracting the reconstructed images from the respective true input component layer images.

This is just an example of a more general class of motion degeneracies, where the layers break up into different *isolated regions*. For example, if all shifts are by even (horizontal and vertical) amounts, the image breaks up into four independent isolated regions (corresponding to the obvious 4-coloring of a checkerboard).

If the motion is not along one of the coordinate directions (or in a fixed integral shift pattern), this problem is somewhat reduced, because the resampling process during warping will combine pixels from different rows and columns. Figures 2h and 2i show the results of reconstruction from an input sequence in which the relative motions between the layers were

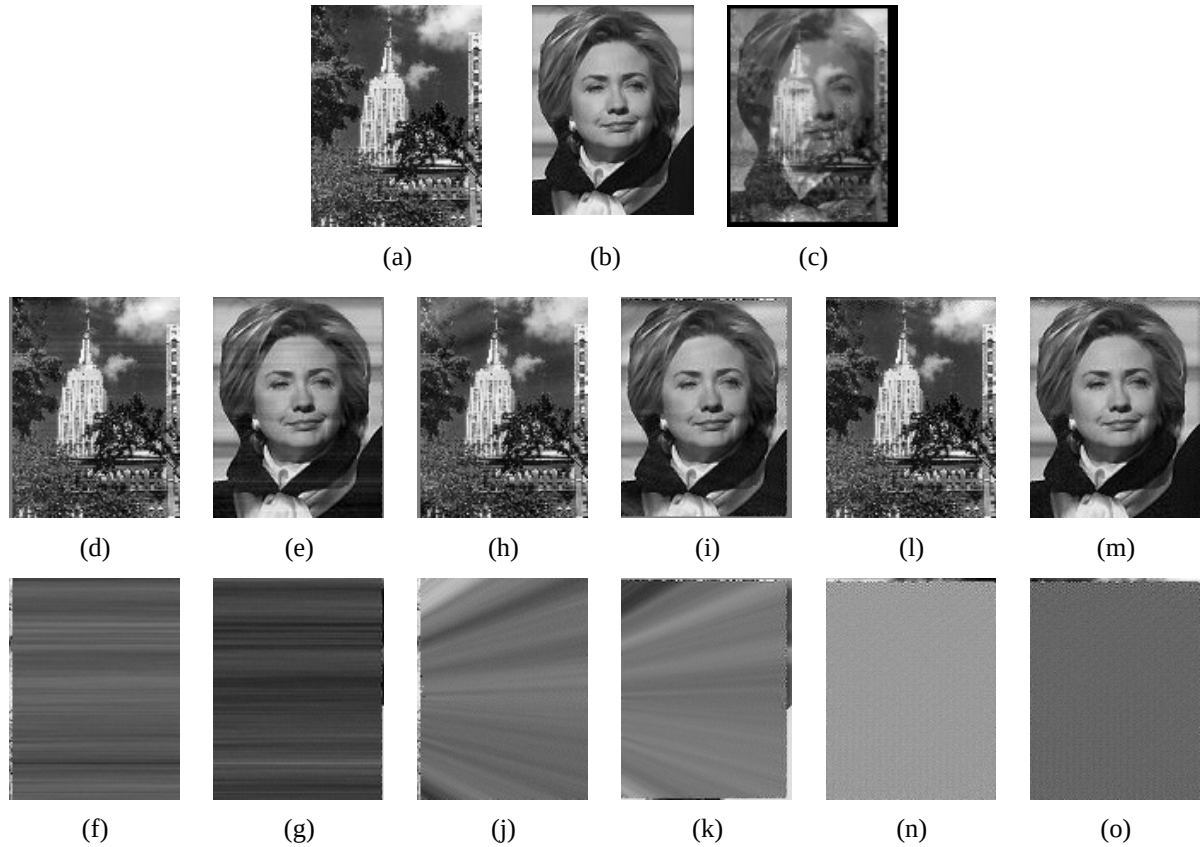


Figure 2. 2D example of constrained least squares: (a),(b) background and foreground images, (c) noisy mixed image, (d),(e) reconstructed background and foreground for horizontal motion , (f),(g) corresponding difference images wrt to the input "true" components, (h),(i),(j),(k) reconstructed images and the corresponding differences for diagonal motion, (l),(m),(n),(o) reconstructed images and the corresponding differences for general motion.

along a set of lines radially emanating out a single point in the 2D image plane (i.e. as in the case of a "zoom" motion).

If the set of motions is arbitrary and general (and not degenerate), the reconstruction can also be achieved without degeneracy. This is illustrated in Figures 2l and 2m, which show the reconstruction for a case when an arbitrary and general set of relative motions were used to create the input composites. The difference images (Figures 2n and and 2o) look like white noise.

We also tested the algorithm on a synthetic case of three layers with known motions and obtained similar results to the two layers case. For lack of space, we do not present the images for this experiment.

In summary, the constrained least-square problem posed in Equation 5 has a unique solution unless the set of relative motions between the component layers in the input composites is degenerate (or poorly conditioned) in some fashion. Under the general (non-degenerate) condition, given known motion, it should be possible to recover the component layers from the input composites.

Of course, in practice we do not assume that the motions are known – indeed the estimation of the motion is an important part of our overall algorithm. This will be discussed further in Sections 5 and 6.

4 Min/max alternation

In order to run the constrained least-squares algorithm, the motions for all of the layers must be known. Unfortunately, in many image sequences, only the dominant motion can be reliably estimated at first. Unless there is some way to estimate the non-dominant motion(s), we cannot solve the overall problem. In this section, we propose a novel algorithm that iteratively re-estimates upper and lower bounds on two component layers. This estimation can be interleaved with layer motion estimation, as explained in the Section 6.

Our algorithm is based on the following observation. Once the dominant motion has been estimated, an estimate for the layer corresponding to this motion can be obtain by forming a *mosaic* from the stabilized image sequence. However, unlike conventional mosaics, where either an average or median is used to form the estimate (sometimes with appropriate *feathering* near the edges [17]), we propose computing the *minimum* pixel value across all images in the stabilized sequence.

Why is this *min-composite* the right estimate to compute? Observe that the contributions from other layers can only add

to the intensity at a given pixel. Therefore, the min across all mixed images gives us an upper bound on the possible value for the dominant layer.

More formally, let

$$\mathbf{s}_k = \mathbf{W}_{k0}^{-1}\mathbf{m}_k = \mathbf{f}_0 + \sum_{l=1}^{L-1} \mathbf{W}_{k0}^{-1}\mathbf{W}_{kl}\mathbf{f}_l \tag{6}$$

be the set of images stabilized with respect to layer 0. Then,

$$\mathbf{f}_0^{\max} = \min_k \mathbf{s}_k = \mathbf{f}_0 + \sum_{l=1}^{L-1} \min_k \mathbf{W}_{k0}^{-1}\mathbf{W}_{kl}\mathbf{f}_l \tag{7}$$

is an upper bound on $\mathbf{f}_0$.

Once we have an estimate for layer 0, we can compute the difference images

$$\mathbf{d}_k = \mathbf{s}_k - \mathbf{f}_0^{\max}. \tag{8}$$

These difference images give us the luminance that must somehow be accounted for by the other layers.

At this point, it is hard to make further progress unless we know how to distribute this residual error among the remaining layers. For this reason, we will now restrict our attention to the two layer (foreground / background) case.

In the two layer case, the difference images $\mathbf{d}_k$ are a partial estimate (lower bound) on the amount of light in layer 1. We can stabilize these images using a parametric motion estimator (assuming that the motion is not known *a priori*), and thereby compute $\mathbf{W}_{k1}$. Let

$$\mathbf{t}_k = \mathbf{W}_{k1}^{-1}\mathbf{W}_{k0}\mathbf{d}_k = \mathbf{f}_1 + \mathbf{W}_{k1}^{-1}\mathbf{W}_{k0}(\mathbf{f}_0 - \mathbf{f}_0^{\max}). \tag{9}$$

be the set stabilized of difference images. We can then compute a *max-composite* of the stabilized differences,

$$\mathbf{f}_1^{\min} = \max_k \mathbf{t}_k = \mathbf{f}_1 + \max_k \mathbf{W}_{k1}^{-1}\mathbf{W}_{k0}(\mathbf{f}_0 - \mathbf{f}_0^{\max}). \tag{10}$$

Since $\mathbf{f}_0 - \mathbf{f}_0^{\max} \leq 0$, each $\mathbf{t}_k$ is an underestimate of $\mathbf{f}_1$, and $\mathbf{f}_1^{\min}$ is the tightest lower bound on $\mathbf{f}_1$ we can compute.

With our improved lower bound estimate for $\mathbf{f}_1$ (remember that we started with $\mathbf{f}_1 \geq 0$), we can now re-compute a better estimate (tighter upper bound) for $\mathbf{f}_0$. Instead of stabilizing the orignal input images $\mathbf{m}_k$, we can instead stabilize the *corrected* images

$$\mathbf{c}_k = \mathbf{m}_k - \mathbf{W}_{k1}\mathbf{f}_1^{\min} \tag{11}$$

to obtain

$$\mathbf{s}_k = \mathbf{W}_{k0}^{-1}\mathbf{c}_k = \mathbf{f}_0 + \mathbf{W}_{k0}^{-1}\mathbf{W}_{k1}(\mathbf{f}_l - \mathbf{f}_1^{\min}). \tag{12}$$

The amount of overestimate in each stabilized image $\mathbf{s}_k$ is now proportional to the difference between the lower bound on $\mathbf{f}_1$ and its true value.

We can thus obtain an improved estimate for $\mathbf{f}_0^{\max}$, and use this to obtain an improved estimate for $\mathbf{f}_1^{\min}$. The question then is: does this iteration eventually lead to the correct solution, and if so, at what rate? The answer is in the following Theorem.

Theorem 1: Under ideal conditions (to be defined below), the min/max alternation algorithm described above will compute the correct estimates for $\mathbf{f}_0$ and $\mathbf{f}_1$. The time required to do so depends on the diameter of the largest non-zero region in the foreground layer ($\mathbf{f}_1$) divided by the diameter of the shifting operation seen in all input images (to be defined below).

Proof: First, we need to assume (as usual) that at least one pixel in the foreground layer is zero. If not, then min/max alternation will compute the best lower bound on $\mathbf{f}_1$ it can (which will contain at least one zero value) and stop. Also, we assume that there is only one isolated region (otherwise, the Theorem applies to each region independently).

The ideal conditions mentioned above come in two parts:

1. the entries in the the $\mathbf{W}_{kl}$ and $\mathbf{W}_{kl}^{-1}$ matrices are non-negative ;
2. there is no imaging noise .

The first condition is, in general, only attainable if the layers are shifted by integral amounts. The second condition is, of course, not attainable in practice. We will discuss how to compensate for these problems later. For now, let's finish the proof.

Let x be the coordinate of some pixel where $f_1(x) = 0$. Let $x' \in \mathcal{N}(x)$ be the *shift-induced neighborhood* of x, i.e., the set of pixels in the m_k images that are formed using $f_1(x)$. Then, since $\min_k W_{k1} f_1(x) = 0$ for any pixel in $\mathcal{N}(x)$, $f_0^{\max}(x') = f_0(x')$, i.e., the upper bound is exact at these pixels. Furthermore, the difference signals at these pixels is exact (the lower bound matches the true value of the shifted f_1 signal). Therefore, the pixels in f_1 where $x'' \in \mathcal{N}'(x')$, i.e., the pixels being re-estimated using at least one correct element in f_0, will have the correct estimated value, $f_1^{\min}(x'') = f_1(x'')$.

This process will grow regions of correct estimates out from pixels in the foreground that are black. How quickly do these regions grown and do they eventually cover the entire image? Think of $x' \in \mathcal{N}(x)$ as a morphological dilation operator that spreads good pixels (initially, the black ones) in f_1 into good estimates of f_0. Similarly, $x'' \in \mathcal{N}'(x')$ is the morphological dilation operator that spreads good pixels in f_0 into good pixels in f_1. Each dilation operation eats away at the borders of the regions that have potentially erroneous estimates of f_0 and f_1. The number of operations required is the (outside) diameter of the largest such region divided by the (inside) diameter of the dilation operator.

Figure 3c and 3d show the results of running our min/max algorithm on a simple 1-D signal with ± 1 shifts in x. The thin dashed curve shows the background (and foreground) signals after 1 iteration and the thick dash-dot curves show the background (and foreground) after 3 iterations. Note that convergence has already been achieved after 3 iterations.

Note that we have described the algorithm as computing upper bounds for one layer, and lower bounds for another.

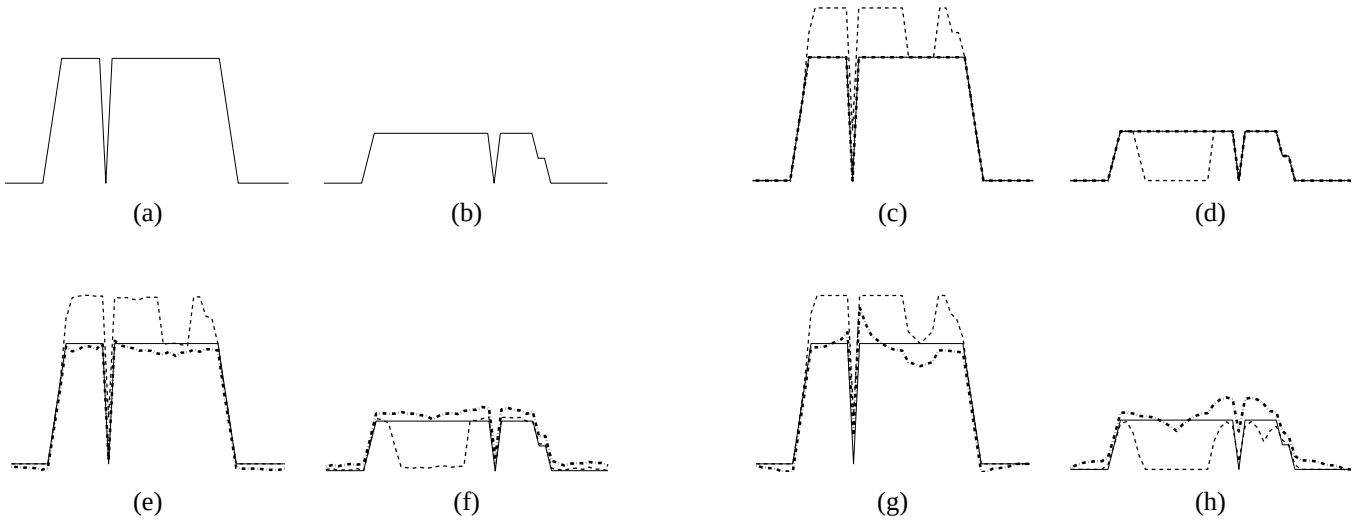


Figure 3. 1D example of min/max alternation: (a) background signal, (b) foreground signal, (c)–(d) convergence in ideal case (noise-free, integer shifts) thin dashed curve is the signal obtained after 1 iteration, the thick dash-dot curve is the signal obtained after the 3rd iteration, (e)–(f) non-convergence for noisy signals, same symbology conventions as before, (g)–(h) non-convergence for non-integer shifts.

The process could also be run the other way around (once motion estimates are known for both layers) to simultaneously compute upper and lower bounds.

4.1 Problems due to noise and resampling

The min-max algorithm is powerful in that it guarantees global convergence. Unfortunately, in order for the theorem to hold, the ideal conditions mentioned above must be strictly satisfied. When noise is present, the upper and lower bounds computed by min/max will be erroneous at each iteration, leading to a divergence away from the correct solution. This behavior can be seen in Figures 3e and 3f.

Similarly, the subpixel interpolation involved in the resampling process can also lead to a bad solution. There are two potential problems when resampling the images. The first is that some entries in the the $\mathbf{W}_{kl}$ and $\mathbf{W}_{kl}^{-1}$ matrices may be negative. (For a positive interpolants $\mathbf{W}_{kl}$ such as bilinear or B-splines, the inverse warp will have negative sidelobes.) In these cases, the upper/lower bound estimates $\mathbf{f}_0^{\max}$ and/or $\mathbf{f}_1^{\min}$ computed in Equations 7 and 10 may be invalid (too tight). These errors propagate from iteration to iteration, and eventually come up with global solutions that are invalid (do not satisfy the constraints).

The second potential problem is that we are using an approximation to $\mathbf{W}_{kl}^{-1}$. This happens quite often, for example when bi-linear or bi-cubic filtering is used in conjunction with a hardware or software perspective warping algorithm (in both directions). If in this case, while the entries in $\mathbf{W}_{kl}$ and $\mathbf{W}_{kl}^{*}$ (the approximate inverse) may be non-negative, Equation 7 is no longer valid. Instead, the equation should read

$$\mathbf{f}_0^{\max} = \mathbf{W}_{k0}^{*}\mathbf{W}_{k0}\mathbf{f}_0 + \sum_{l=1}^{L-1} \min_k \mathbf{W}_{k0}^{*}\mathbf{W}_{kl}\mathbf{f}_l. \qquad (13)$$

There is no longer any guarantee that the first term is not less than $\mathbf{f}_0$. In practice, we observe that the algorithm starts to diverge rather quickly (Figures 3g and 3h).

5 Re-estimating the layer motions

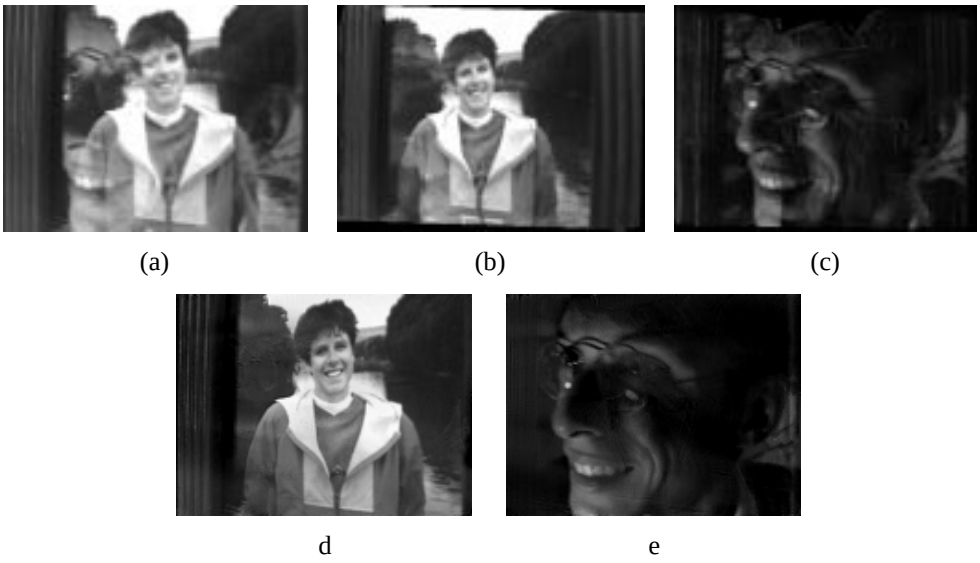
Once we have layer estimates (starting with one iteration of the min/max algorithm to compute the initial dominant and non-dominant motions, and optionally followed by an initial solution of the constrained least squares), we can refine our motion estimates.

The algorithm to do this is almost identical to the usual [4, 17] parametric motion estimator. Expanding (5) using a Taylor series in the motion parameters $\mathbf{p}_{kl}$, we obtain

$$\sum_k \sum_x \left[\sum_{l=0}^{L-1} f_l(x_{kl}(x;\mathbf{p}_{kl})) - m_k(x) \right]^2$$

$$\approx \sum_k \sum_x \left[e_k(x) + \sum_{l=0}^{L-1} \nabla f_l(x_{kl}(x;\mathbf{p}_{kl})) \frac{\partial x_{kl}}{\partial \mathbf{p}_{kl}} \Delta \mathbf{p}_{kl} \right]^2$$

The errors $e_k(x)$ are computed as usual (difference between predicted and observed signals). The gradients ∇f_l are computed for each layer separately, and used to compute that layer's motion.



(a) (b) (c)

d e

Figure 4. Experimental results for the *Michael and Lee* sequence [6]: (a) first input image, (b) dominant (picture) layer *min-composite*, (c) reflection layer *max-composite*, (d)-(e) final estimated picture and reflection layers.

6 Complete algorithm

The complete algorithm to estimate the component layer images and their associated motions is as follows:

1. Compute a dominant motion for the sequence using image alignment against the current min-composite $\mathbf{f}_0^{\max}$.
2. Compute the difference images $\mathbf{d}_k$ between the stabilized images and the min-composite $\mathbf{f}_0^{\max}$.
3. Compute the non-dominant motion by aligning the difference images $\mathbf{d}_k$ with a max-composite of these images.
4. Using the intial layer guesses, improve the motion estimates using the motion re-estimation algorithm.
5. Compute the unconstrained least-squares solution.
6. Using this result as the initial value, solve the quadratic-programming problem with the positivity constraints.
7. Optionally alternate the least-squares optimization of layer values with motion re-estimation.

In the experiments presented in the next section, we did not observe any improvement from performining step 7.

7 Experimental results

We show the results of applying our technique to some real image examples. Both the examples involve only two layers, although the technique described in this paper applies to an arbitrary number of layers. Note that in both of these cases (unlike the previous synthetic examples), neither the motion nor the component layers are known.

The first example consists of the reflection of a face in a photograph. This is the same sequence that was used in [6]. Figure 4a shows one image from the input sequence. The algorithm described in Section 6 was used, with all the steps, including the motion estimation and the layer extraction, being done automatically. Figures 4b shows the *min*-composite of the dominant layer (the picture) while Figure 4c shows the *max*- composite of the reflection layer. Figures 4d and 4e show the results obtained using the constrained least-squares algorithm.

The second real image example is shown in Figure 5. These images are in color (here we can only display the gray value version), so we had to extend our estimation framework to handle this case. Because the color channels do not interact in Equation 5, we can solve three independent constrained least squares problems. The motion estimation is performed using all three channels simultaneously.

The first input composite image is shown in Figure 5a. As in the previous example, the entire analysis was automatic. Figure 5b and 5c show the results of the *min*- and *max*-composites of the dominant layer and the reflection, while Figures 5d and 5e show the final reconstructions obtained by solving the constrained least-squares problem. While it is hard to interpret the reflected light in the original image sequence, it is clear to see that it is a bookshelf with labelled boxes of paper.

8 Conclusion

We have investigated the problem of extracting a set of component layers from a collection of composite images. While the problem of recovering the multiple motions from such sequences has been extensively studied (at least when the motions are parametric), the problem of extracting the layer images in the presence of reflections and transparency has not been adequately treated until now. Here, we have described an algorithm for recovering the layer images and their motions from the input sequence. When the input composite images can be modeled as an additive mixture of the component layers (such a model applies when the light from one surface is

(a) (b) (c) (d) (e)

Figure 5. Experimental results for the *Anne* sequence: (a) first input image, (b) dominant (picture) layer *min-composite*, (c) reflection layer *max-composite*, (d)-(e) final estimated picture and reflection layers. Note that the reflected layers in (c) and (e) have been doubled in intensity to better show their structure.

reflected by another), we have described a constrained least-square technique to recover the layers from known motions. We have further described a complete algorithm that combines the layer extraction step together with an automatic multiple motion technique to recover the layers and their motions from the input images.

There are several logical next steps for our work. Previous work in layer extraction has addressed the case of opaque layers, whereas this paper has focused on dealing with reflections and transparency. In a real image, both phenomena will be simultaneously present. Therefore, one logical extension is to handle cases where both opaque and transparent layers are present. Once the opacity (α) values are thrown in as unknowns, the formulation becomes a non-linear least squares problem (the opacities and colors form bi-linear measurement equations). A more complete description of such a formulation can be found in [3].

We have also restricted our attention to parametric motion models such as homographies. While these are adequate when the scene can be approximated as a collection of planar layers, to deal with more general scenes, we must also handle parallax. (It is worth noting, however, with a few exceptions [3, 10], previous work on layer extraction has also focused only on parametric motions.) Therefore another logical extension of our work is to handle scenes containing planar parallax.

We began this paper by noting that reflections and transparency are ubiquitous in images. While the literature on the recovery of camera and scene geometry from multiple images is well-developed, almost none of the current work can deal with images containing mixtures of transmitted and reflected light. The work described in this paper is our first step towards enabling vision-based scene modeling to deal with such complex scenes and images.

References

[1] E. H. Adelson. Perceptual organization and the judgement of brightness. *Science*, 262:2042–2044, 24 Dec. 1993.

[2] E. H. Adelson and P. Anandan. Ordinal characteristics of transparency. In *AAAI-90 Work. Qualitative Vision*, pp. 77–81, 1990.

[3] S. Baker, R. Szeliski, and P. Anandan. A layered approach to stereo reconstruction. In *CVPR'98*, pp. 434–441, June 1998.

[4] J. R. Bergen, P. Anandan, K. J. Hanna, and R. Hingorani. Hierarchical model-based motion estimation. In *ECCV'92*, pp. 237–252, Italy, May 1992.

[5] J. R. Bergen, P. J. Burt, R. Hingorani, and S. Peleg. A three-frame algorithm for estimating two-component image motion. *IEEE Trans. Patt. Anal. Mach. Intel.*, 14(9):886–896, Sept. 1992.

[6] M. J. Black and P. Anandan. The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields. *Comp. Vis. Image Understanding*, 63(1):75–104, 1996.

[7] J. F. Blinn. Jim Blinn's corner: Compositing, part 1: Theory. *IEEE Comp. Gr. and Appl.*, 14(5):83–87, Sept. 1994.

[8] T. Darrell and E. Simoncelli. "Nulling" filters and the separation of transparent motion. In *CVPR'93*, pp. 738–739, 1993.

[9] Paul E. Debevec and Jitendra Malik. Recovering high dynamic range radiance maps from photographs. *SIGGRAPH'97*, pp. 369–378, Aug. 1997.

[10] P. R. Hsu, P. Anandan, and S. Peleg. Accurate computation of optical flow by using layered motion representations. In *ICPR'94*, pp. 743–746, Oct. 1994.

[11] M. Irani, B. Rousso, and S. Peleg. Computing occluding and transparent motions. *Int. J. Comp. Vis.*, 12(1):5–16, Jan. 1994.

[12] S. X. Ju, M. J. Black, and A. D. Jepson. Skin and bones: Multi-layer, locally affine, optical flow and regularization with transparency. In *CVPR'96*, pp. 307–314, June 1996.

[13] S.K. Nayar, X. S. Fang, and T. Boult. Separation of reflectance components using color and polarization. *Int. J. Comp. Vis.*, 21:163–186, 1997.

[14] H. S. Sawhney and S. Ayer. Compact representation of videos through dominant multiple motion estimation. *IEEE Trans. Patt. Anal. Mach. Intel.*, 18(8):814–830, Aug. 1996.

[15] S. A. Shafer, G. Healey, and L. Wolff. *Physics-Based Vision: Principles and Practice*. Jones & Bartlett, 1992.

[16] M. Shizawa and K. Mase. A unified computational theory of motion transparency and motion boundaries based on eigenenergy analysis. In *CVPR'91*, pp. 289–295, June 1991.

[17] H.-Y. Shum and R. Szeliski. Construction and refinement of panoramic mosaics with global and local alignment. In *ICCV'98*, pp. 953–958, Jan. 1998.

[18] J. Y. A. Wang and E. H. Adelson. Representing moving images with layers. *IEEE Trans. Im. Proc.*, 3(5):625–638, Sept. 1994.

[19] Y. Weiss. Smoothness in layers: Motion segmentation using nonparametric mixture estimation. In *CVPR'97*, pp. 520–526, June 1997.