Single Image Spatially Variant Out-of-focus Blur Removal

Stanley H. Chan[a] and Truong Q. Nguyen[b]

[a]SEAS, Harvard University, 33 Oxford Street, Cambridge, MA 02138
[b]Dept of ECE, UCSD, 9500 Gilman Drive, La Jolla, CA 92093

ABSTRACT

This paper addresses the problem of two-layer out-of-focus blur removal from a single image, in which either the foreground or the background is in focus while the other is out of focus. To recover details from the blurry parts, the existing blind deconvolution algorithms are insufficient as the problem is spatially variant. The proposed method exploits the invariant structure of the problem by first predicting the occluded background. Then a blind deconvolution algorithm is applied to estimate the blur kernel and a coarse estimate of the image is found as a side product. Finally, the blurred region is recovered using total variation minimization, and fused with the sharp region to produce the final deblurred image.

Keywords: Blind deconvolution, spatially variant, total variation, alpha-matting, kernel estimation, depth.

1. INTRODUCTION

1.1 Two-layer Blur

For an image consisting of multiple layers of depth – objects at different depth locations – only one of them can be in focus using a standard camera. Those not being focused are blurred, and this type of blur is referred to as the out-of-focus blur. In terms of wave optics, out-of-focus blur is the result of additional (or insufficient) phase propagation from the desired image plane to the actual image plane [1, Ch. 6.4]. Recovering an out-of-focus blurred image using post-processing computational techniques is an ill-posed non-linear problem in general, and seeking a universal solution is almost impossible. However, in a simplified scenario where there is only one foreground object and a background scene, recovering the image becomes a more tractable problem. The goal of this paper is to present a method that restores both the foreground and background, using a single image at a low computational cost.

Although the two-layered problem is a special case of the general out-of-focus blur problem, some challenges remain. The first challenge is the *spatially variant* property due to different blurs occuring in the foreground and background. Spatially variant blur is computationally difficult because Fourier Transforms cannot be applied. The second challenge is the need for *blind* deconvolution as the blur kernel is unknown. Blind deconvolution is difficult because simultaneous recovery of image and kernel is a nonlinear ill-posed problem.

1.2 Related Works

Most of the existing image deblurring literature assumes that the blur is spatially invariant so that the observed image is related to the input image as

$$\mathbf{g} = \mathbf{h} * \mathbf{f} + \boldsymbol{\eta},$$

where $\mathbf{g}$ is the observed blur image, $\mathbf{f}$ is the unknown sharp image, $\mathbf{h}$ is the blur kernel, $\boldsymbol{\eta}$ is the noise and $*$ denotes the 2D convolution.

If $\mathbf{h}$ is known, the problem of recovering $\mathbf{f}$ from $\mathbf{g}$ can be done using classical methods such as Wiener deconvolution,[2] Lucy-Richardson deconvolution,[3] or regularized least-squares deconvolution.[4] Better approaches such as total variation minimization[5] and its variations[6,7] can also be used. With the new implementation by Chan et al.,[8] solving a TV problem can be done in less than a second for a moderate sized image in MATLAB. Advanced image deblurring algorithms[9] can be used if high quality recovery results are required.

Further author information: Stanley Chan: schan@seas.harvard.edu; Truong Nguyen: nguyent@ece.ucsd.edu

If $\mathbf{h}$ is unknown, blind deconvolution methods are needed to repeatedly estimate the blur kernel and predict the underlying image in an alternating minimization procedure. Cho and Lee[10] proposed a fast and reliable blind deconvolution algorithm using image gradients. Later, Xu and Jia[11] improved this method by selecting strong gradients. Note, however, that these methods are only applicable to invariant blurs.

The problem that interests us is the case where $\mathbf{h}$ is spatially variant. However, only few papers discuss spatially variant deblurring. Among these, perhaps the most widely used approach is by Nagy and O'Leary,[12] which suggests that spatially variant blur could be modeled as:

$$\mathbf{g} = \sum_{i=1}^{p} \boldsymbol{\alpha}_i \cdot (\mathbf{h}_i * \mathbf{f}), \tag{1}$$

where p is the number of blur kernels, $\mathbf{h}_i$ is the i-th blur kernel, $\boldsymbol{\alpha}_i$ is a binary mask indicating the contribution of the i-th kernel, and "$\cdot$" denotes element-wise multiplication. The problem of (1) is that it is inadequate to model a two-layered blur. Jia[13] reported that ringing artifact is generated by a moderately advanced deconvolution algorithm even if the true blur kernel is known. Similar observations are found in Dai and Wu,[14] suggesting that some fundamental issues are present.

The most relevant work to us is a global minimization method proposed by Dai and Wu.[15] In their work, they considered the two-layer blur problem as a giant global minimization and proposed an iterative reweighted least squares (IRLS) algorithm to solve the problem. While their approach gives satisfactory results, the computation time is large.

The goal of this paper is to achieve the following two objectives. First, we show that although the two-layer blur is spatially invariant, it can be transformed to two invariant sub-problems so that fast algorithms can be applied. Second, we propose a new blur kernel estimation method that uses information from both the image content and alpha-matte edges.

2. IMAGING MODEL

The imaging model of a two-layered blur is the foundation of all subsequent analysis discussed in this paper. Therefore, in this section, we provide justifications to our model. More details can be found in the literature.[15–17]

2.1 Limitation of Classical Model

Figure 1. Limitation of the Classical Model. The right image is a simulation of (2) in an extreme situation where $\mathbf{h}_F$ is a delta function and $\mathbf{h}_B$ is a "disk" function with large radius. Unwanted color bleeding is observed around the object boundary, which is wrong because the foreground color should not contribute to the background blur. The background color should be sky blue in the circled region. The left image is the result of spatially variant TV minimization.[8]

As we discussed in the introduction, the classical model (1) is widely used but there are fundamental problems. To understand the problem, we consider the formation of a sharp foreground and blurred background image. This image is formed according to (1) by

$$\mathbf{g} = \boldsymbol{\alpha} \cdot (\mathbf{h}_F * \mathbf{f}) + (1 - \boldsymbol{\alpha}) \cdot (\mathbf{h}_B * \mathbf{f}), \tag{2}$$

where $\mathbf{h}_F$ is the delta function, $\mathbf{h}_B$ is the blur for the background, and $\boldsymbol{\alpha}$ is the alpha matte that indicates the location of foreground pixels.[18] The interpretation of (2) is that the image $\mathbf{f}$ is first blurred using $\mathbf{h}_F$ and $\mathbf{h}_B$, and then cropped and combined according to $\boldsymbol{\alpha}$.

If the classical model *were* valid for the formation of a sharp foreground and blurred background image, then one should be able to recover the image (reasonably well) by using methods such as,[8][13] or.[14] However, even with a good estimate of the blur kernel and a fine-tuned algorithm, ringing artifacts still appear at the foreground object boundary as shown in the left image of Fig. 1.

2.2 Model Used in This Paper

Our model was used by McGuire *et al.*,[17] which suggested that the image $\mathbf{f}$ can be expressed as $\mathbf{f} = \boldsymbol{\alpha} \cdot \mathbf{f}_F + (1 - \boldsymbol{\alpha}) \cdot \mathbf{f}_B$, where $\mathbf{f}_F$ denotes the foreground and $\mathbf{f}_B$ denotes the background. Consequently, the observed image is

$$\mathbf{g} = \mathbf{h}_F * (\boldsymbol{\alpha} \cdot \mathbf{f}_F) + (1 - \boldsymbol{\alpha} * \mathbf{h}_F) \cdot (\mathbf{h}_B * \mathbf{f}_B). \tag{3}$$

Note that the order of convolution "$*$" and the element-wise multiplication "$\cdot$" cannot be switched. Also, it is assumed that $\mathbf{h}_F$ has a small support.

Observing (3), we find that there is no cross convolution terms such as $\mathbf{h}_F * \mathbf{f}_B$ or $\mathbf{h}_B * \mathbf{f}_F$, which would appear in (2) if we substitute $\mathbf{f} = \boldsymbol{\alpha} \cdot \mathbf{f}_F + (1 - \boldsymbol{\alpha}) \cdot \mathbf{f}_B$. For example, for the case where $\mathbf{h}_F$ is the δ-function, our model (3) implies

$$\mathbf{g} = (\boldsymbol{\alpha} \cdot \mathbf{f}_F) + (1 - \boldsymbol{\alpha}) \cdot (\mathbf{h}_B * \mathbf{f}_B), \tag{4}$$

whereas Nagy and O'leary's model (2) implies

$$\begin{aligned} \mathbf{g} &= \boldsymbol{\alpha} \cdot \mathbf{f} + (1 - \boldsymbol{\alpha}) \cdot (\mathbf{h}_B * \mathbf{f}) = \boldsymbol{\alpha} \cdot (\boldsymbol{\alpha} \cdot \mathbf{f}_F + (1 - \boldsymbol{\alpha}) \cdot \mathbf{f}_B) + (1 - \boldsymbol{\alpha}) \cdot (\mathbf{h}_B * (\boldsymbol{\alpha} \cdot \mathbf{f}_F + (1 - \boldsymbol{\alpha}) \cdot \mathbf{f}_B)) \\ &= \boldsymbol{\alpha} \cdot \mathbf{f}_F + (1 - \boldsymbol{\alpha}) \cdot (\mathbf{h}_B * (\boldsymbol{\alpha} \cdot \underline{\mathbf{f}_F} + (1 - \boldsymbol{\alpha}) \cdot \mathbf{f}_B)), \end{aligned} \tag{5}$$

where we used the facts $\boldsymbol{\alpha} \cdot \boldsymbol{\alpha} = \boldsymbol{\alpha}$ and $\boldsymbol{\alpha} \cdot (1 - \boldsymbol{\alpha}) = 0$. It can be seen that (4) and (5) coincides if the underlined term $\mathbf{f}_F$ in (5) is replaced by $\mathbf{f}_B$, which suggests that the cross convolution term $\mathbf{h}_B * \boldsymbol{\alpha}\mathbf{f}_F$ causes the color bleeding shown in Fig. 1.

3. EXPLOITING INVARIANT STRUCTURES

By definition of a two-layered out-of-focus blur problem, the observed image consists of two regions in which each region is homogeneously blurred by a blur kernel. Assuming that the blur kernels are known (estimation of kernels are discussed in Section 4), recovering for each region is a classical invariant deconvolution problem. However, the question is how to partition a variant problem into two invariant sub-problems.

3.1 Background Blur / Foreground Sharp Case

A background blur is characterized by setting $\mathbf{h}_F = \delta$-function in (3) so that the observed image is given in (4). With the assumption that the ground truth alpha-matte $\boldsymbol{\alpha}$ is available, the foreground component $\boldsymbol{\alpha} \cdot \mathbf{f}_F$ equals to $\boldsymbol{\alpha} \cdot \mathbf{g}$, and hence $\mathbf{g} - \boldsymbol{\alpha} \cdot \mathbf{f}_F = (1 - \boldsymbol{\alpha}) \cdot \mathbf{g}$. Since it is also true that $\mathbf{g} - \boldsymbol{\alpha} \cdot \mathbf{f}_F = (1 - \boldsymbol{\alpha}) \cdot (\mathbf{h}_B * \mathbf{f}_B)$, we have

$$(1 - \boldsymbol{\alpha}) \cdot \mathbf{g} = (1 - \boldsymbol{\alpha}) \cdot (\mathbf{h}_B * \mathbf{f}_B). \tag{6}$$

Our goal is to determine $\mathbf{f}_B$. Solving (6) would be a standard deconvolution problem if the term $(1 - \boldsymbol{\alpha})$ were invertible, which is not possible due to the binary nature of $\boldsymbol{\alpha}$. However, the singularity of $(1 - \boldsymbol{\alpha})$ implies the existence of many $\widetilde{\mathbf{g}}$ such that $\widetilde{\mathbf{g}} \neq \mathbf{g}$ but $(1 - \boldsymbol{\alpha}) \cdot \widetilde{\mathbf{g}} = (1 - \boldsymbol{\alpha}) \cdot \mathbf{g}$. In fact, any $\widetilde{\mathbf{g}}$ in the following form would satisfy (6):

$$\widetilde{\mathbf{g}}(i,j) = \begin{cases} (\mathbf{h}_B * \mathbf{f}_B)(i,j), & \text{if } (i,j) \in \Omega_B, \\ \text{any function } \psi(i,j), & \text{if } (i,j) \in \Omega_F, \end{cases}$$

where $\widetilde{\mathbf{g}}(i,j)$ denotes the (i,j)-th pixel of $\widetilde{\mathbf{g}}$, $\Omega_B = \{(i,j) \,|\, \boldsymbol{\alpha}(i,j) = 0\}$ is the set of background pixel coordinates, and $\Omega_F = \{(i,j) \,|\, \boldsymbol{\alpha}(i,j) = 1\}$ is the set of foreground pixel coordinates. The arbitrary function $\psi(i,j)$ is chosen such that the deconvolution problem $\widetilde{\mathbf{g}} = \mathbf{h}_B * \mathbf{f}_B$ yields good results. Choosing a meaningful $\psi(i,j)$ is equivalent to *inpainting* the pixels for Ω_F.

Figure 2. [Left] The background component with occluded region unfilled (i.e., $\psi(i,j) = 0$). [Right] Deblurring result of the left image.

A naive choice of $\psi(i,j)$ is that $\psi(i,j) = 0$ for any $(i,j) \in \Omega_F$, which means that no inpainting is performed. As one can expect, ringing artifacts will appear. Fig. 2 shows an example where we use $\mathbf{h}_B$ = Gaussian blur kernel ($\sigma = 2$). The deconvolution is performed using a regularized Wiener filter.

To reduce oscillation, we must fill Ω_F carefully so that the transient between Ω_F and Ω_B is smooth. This can be achieved by minimizing the image gradient near the boundary of Ω_F and Ω_B. Specifically, we initially set $\widetilde{\mathbf{g}}(i,j) = (\mathbf{h}_B * \mathbf{f}_B)(i,j)$ for $(i,j) \in \Omega_B$ and $\widetilde{\mathbf{g}}(i,j) = 0$ for $(i,j) \in \Omega_F$. Then for any pixel (i,j) at the boundary of Ω_F and Ω_B, we define $\mathcal{A}$ as the set of non-zero pixels of neighborhood of (i,j):

$$\mathcal{A} = \{(p,q) \,|\, \widetilde{\mathbf{g}}(i+p, i+q) \neq 0, |p| \leq 1, |q| \leq 1\},$$

and define the function $\psi(i,j)$ as

$$\psi(i,j) = \frac{1}{|\mathcal{A}|} \sum_{(p,q)\in\mathcal{A}} \left(1 + \frac{1}{k}\right) \widetilde{\mathbf{g}}(i+p, i+q) - \frac{1}{k}\widetilde{\mathbf{g}}(i+2p, i+2q), \tag{7}$$

where k is the shortest distance from the unknown pixel (i,j) to the known set Ω_B. The motivation behind this $\psi(i,j)$ is that we want to fill the missing pixel such that the image gradient across the boundary is minimized. Full derivation can be found in the Supplementary material. The outline of the algorithm is given in Algorithm 1.

Algorithm 1 Proposed Inpainting Algorithm

Given $\widetilde{\mathbf{g}}$, Ω_F and Ω_B.
// Outward Version
Partition Ω_F into K rings from the outermost to the innermost, with each ring one pixel in width.
// Inward Version
Partition Ω_B into K rings from the innermost to the outermost, with each ring one pixel in width.
for $k = 1 : K$ **do**
  for each (i,j) on the k-th ring **do**
    Determine $\mathcal{A} = \{(p,q) \,|\, \widetilde{\mathbf{g}}(i+p, i+q) \neq 0, |p| \leq 1, |q| \leq 1\}$.
    Calculate $\psi(i,j) = \frac{1}{|\mathcal{A}|} \sum_{(p,q)\in\mathcal{A}} \left(1 + \frac{1}{k}\right) \widetilde{\mathbf{g}}(i+p, i+q) - \frac{1}{k}\widetilde{\mathbf{g}}(i+2p, i+2q)$,
  end for
end for

Asymptotically, Algorithm 1 becomes a moving average when $k \to \infty$, because $\widetilde{\mathbf{g}}(i,j) \to \frac{1}{|\mathcal{A}|} \sum_{(p,q)\in\mathcal{A}} \widetilde{\mathbf{g}}(i+p, i+q)$. In fact, except for small k, experimentally the difference between Algorithm 1 and the simple moving average is almost visually indistinguishable. Fig. 3 shows the result of applying Algorithm 1 to fill Ω_F.

As shown in Fig. 3, the central region of $\widetilde{\mathbf{g}}$ does not seem visually pleasing. However, we argue that the goal is to fill Ω_F so that there are less ringing artifacts for the deconvolution step. As a comparison, we applied a state-of-the-art exemplar-based inpainting algorithm by Criminisi *et at.*[19] Shown in Fig. 4 are the inpainting results of Algorithm 1 and the method by Criminisi *et al.*. In terms of visual quality, it is clear that the method by Criminisi *et al.* is significantly better than Algorithm 1. However, since Criminisi *et al.*[19] did not impose

Figure 3. Filling Ω_F for Image No.5 . From Left to Right: The intermediate result of inpainting at iteration 0, 20, 40 and final respectively. Left: Ω_F. When inpainting starts, the algorithm fills the occluded region from outside to inside. Right: the inpainted $\widetilde{\mathbf{g}}$.

(a) Proposed inpainting method (b) Inpainting method by Criminisi et al.[19]

Figure 4. Comparisons between the proposed inpainting method and the exemplar-based inpainting method by Criminisi et al.[19] Top: inpainting results and zoom-in. Bottom: deblurring results and zoom-in.

smoothness criteria at the boundary, ringing artifacts are present. In contrast, the smoothness condition at the boundary of Ω_F is explicitly enforced by Algorithm 1. There are artifacts occurring in the center part of Ω_F, but these can be covered by $\mathbf{g}_F$ as they are far from the boundary. Note also that the computational complexity of Algorithm 1 is significantly lower than the method by Criminisi *et al.*.

3.2 Foreground Blur / Background Sharp Case

In the case of foreground blur, we set $\mathbf{h}_B = \delta$-function in (3). Thus, the observed image is $\mathbf{g} = \mathbf{h}_F * (\boldsymbol{\alpha} \cdot \mathbf{f}_F) + (1 - \boldsymbol{\alpha} * \mathbf{h}_F) \cdot \mathbf{f}_B$. Our goal is to determine $\mathbf{f}_F$. Rearranging the terms we have

$$\mathbf{g}_F = \mathbf{g} - (1 - \boldsymbol{\alpha} * \mathbf{h}_F) \cdot \mathbf{f}_B = \mathbf{h}_F * (\boldsymbol{\alpha} \cdot \mathbf{f}_F). \tag{8}$$

Therefore, assuming that $\boldsymbol{\alpha}$, $\mathbf{h}_F$ are correctly estimated, solving for $\boldsymbol{\alpha} \cdot \mathbf{f}_F$ from $\mathbf{g}_F$ in (8) is a standard deconvolution once $\mathbf{f}_B$ is known. However, $\mathbf{f}_B$ is never known exactly because part of $\mathbf{f}_B$ is occluded by the blurring edge of $\mathbf{f}_F$ *. Thus, given an estimate $\widehat{\mathbf{f}}_B$, for example using Algorithm 1, there is an error term $\Delta\mathbf{f}_B$ so that

$$\widehat{\mathbf{f}}_B = \mathbf{f}_B + \Delta\widehat{\mathbf{f}}_B. \tag{9}$$

Substituting (9) into (8) yields

$$\mathbf{g}_F = \mathbf{g} - (1 - \boldsymbol{\alpha} * \mathbf{h}_F) \cdot (\widehat{\mathbf{f}}_B - \Delta\widehat{\mathbf{f}}_B) = \mathbf{g} - (1 - \boldsymbol{\alpha} * \mathbf{h}_F) \cdot \widehat{\mathbf{f}}_B + (1 - \boldsymbol{\alpha} * \mathbf{h}_F) \cdot \Delta\widehat{\mathbf{f}}_B. \tag{10}$$

In (10), only $\mathbf{g}$ and $(1 - \boldsymbol{\alpha} * \mathbf{h}_F) \cdot \widehat{\mathbf{f}}_B$ can be calculated. Multiplying $\boldsymbol{\alpha} * \mathbf{h}_F$ to both sides of (10) yields

$$(\boldsymbol{\alpha} * \mathbf{h}_F) \cdot [\mathbf{h}_F * (\boldsymbol{\alpha} \cdot \mathbf{f}_F)] = (\boldsymbol{\alpha} * \mathbf{h}_F) \cdot \left[\mathbf{g} - (1 - \boldsymbol{\alpha} * \mathbf{h}_F) \cdot \widehat{\mathbf{f}}_B + (1 - \boldsymbol{\alpha} * \mathbf{h}_F) \cdot \Delta\widehat{\mathbf{f}}_B\right]. \tag{11}$$

*The central interior region of $\mathbf{f}_F$ is not important because it is the same as the central interior region of $\mathbf{g}$.

It can be shown that

$$\|(\boldsymbol{\alpha} * \mathbf{h}_F) \cdot (1 - \boldsymbol{\alpha} * \mathbf{h}_F) \cdot \Delta\widehat{\mathbf{f}}_B\| \leq \|(1 - \boldsymbol{\alpha} * \mathbf{h}_F) \cdot \Delta\widehat{\mathbf{f}}_B\|,$$

implying that the effect of $\Delta\widehat{\mathbf{f}}_B$ is reduced by multiplying $(\boldsymbol{\alpha} * \mathbf{h}_F)$. Therefore, we want to approximate $(\boldsymbol{\alpha} * \mathbf{h}_F) \cdot (1 - \boldsymbol{\alpha} * \mathbf{h}_F) \cdot \Delta\widehat{\mathbf{f}}_B$ instead of $(1 - \boldsymbol{\alpha} * \mathbf{h}_F) \cdot \Delta\widehat{\mathbf{f}}_B$.

Now, two issues remain: (i) We need an approximation for $(\boldsymbol{\alpha} * \mathbf{h}_F) \cdot (1 - \boldsymbol{\alpha} * \mathbf{h}_F) \cdot \Delta\widehat{\mathbf{f}}_B$ because $\Delta\widehat{\mathbf{f}}_B$ is not known; (ii) Given the approximation and hence the right hand side of (11), we need to solve for $\boldsymbol{\alpha} \cdot \mathbf{f}_F$.

Let us consider the second question first. Given $\mathbf{g}_F$, we want to solve $\boldsymbol{\alpha} \cdot \mathbf{f}_F$ from the equation

$$(\boldsymbol{\alpha} * \mathbf{h}_F) \cdot \mathbf{g}_F = (\boldsymbol{\alpha} * \mathbf{h}_F) \cdot [\mathbf{h}_F * (\boldsymbol{\alpha} \cdot \mathbf{f}_F)]. \tag{12}$$

Solving (12) is similar to solving (6), which both requires inpainting and deconvolution. However, (12) is more difficult than solving (6) because $\boldsymbol{\alpha} * \mathbf{h}_F$ is not a binary mask. Here we propose two inpainting strategies. The first strategy inpaints the background using the background color, which can be accomplished using Algorithm 1 (inpainting *inwards*). The second strategy inpaints the background using the foreground color, which can also be accomplished using Algorithm 1 (inpainting *outwards*). In both methods, since there is no sharp cut off between foreground and background, the algorithm starts from some definite background (or foreground) pixels. In our method, we start from K pixels from the expected object boundary (estimated from $\boldsymbol{\alpha}$), where K is the one-sided width of the blur kernel.

Denoting $\overline{\mathbf{g}}_B$ the inpainted background, $\Omega_M = \{(i,j) \,|\, 0 < (\boldsymbol{\alpha} * \mathbf{h}_F)(i,j) < 1\}$, $\Omega_F = \{(i,j) \,|\, (\boldsymbol{\alpha} * \mathbf{h}_F)(i,j) = 1\}$, and $\Omega_B = \{(i,j) \,|\, (\boldsymbol{\alpha} * \mathbf{h}_F)(i,j) = 0\}$, we construct an approximately invariant blur image

$$\widetilde{\mathbf{g}}(i,j) = \begin{cases} \mathbf{g}_F(i,j), & (i,j) \in \Omega_F, \\ \big[(\boldsymbol{\alpha} * \mathbf{h}_F) \cdot \mathbf{g}_F + (1 - \boldsymbol{\alpha} * \mathbf{h}_F) \cdot \overline{\mathbf{g}}_B\big](i,j), & (i,j) \in \Omega_M, \\ \overline{\mathbf{g}}_B(i,j), & (i,j) \in \Omega_B, \end{cases} \tag{13}$$

Based on (13), we propose an approximation for $(\boldsymbol{\alpha} * \mathbf{h}_F) \cdot (1 - \boldsymbol{\alpha} * \mathbf{h}_F) \cdot \Delta\widehat{\mathbf{f}}_B$. Substituting (10) to the Ω_M case of (13), (13) becomes

$$\begin{aligned} \widetilde{\mathbf{g}} = & \underbrace{(\boldsymbol{\alpha} * \mathbf{h}_F) \cdot \mathbf{g}}_{\text{foreground with edge residue}} - \underbrace{(\boldsymbol{\alpha} * \mathbf{h}_F) \cdot (1 - \boldsymbol{\alpha} * \mathbf{h}_F) \cdot \widehat{\mathbf{f}}_B}_{\approx\text{edge residue}} \\ & + \underbrace{(1 - \boldsymbol{\alpha} * \mathbf{h}_F) \cdot \overline{\mathbf{g}}_B}_{\text{inpainted background}} + (\boldsymbol{\alpha} * \mathbf{h}_F) \cdot (1 - \boldsymbol{\alpha} * \mathbf{h}_F) \cdot \Delta\widehat{\mathbf{f}}_B. \end{aligned} \tag{14}$$

The four terms in (14) have individual meaning: $(\boldsymbol{\alpha} * \mathbf{h}_F) \cdot \mathbf{g}$ is the foreground component; $(\boldsymbol{\alpha} * \mathbf{h}_F) \cdot (1 - \boldsymbol{\alpha} * \mathbf{h}_F) \cdot \widehat{\mathbf{f}}_B$ is the foreground edge residue remaining in $(\boldsymbol{\alpha} * \mathbf{h}_F) \cdot \mathbf{g}$; $(1 - \boldsymbol{\alpha} * \mathbf{h}_F) \cdot \overline{\mathbf{g}}_B$ is the background component to be added. The sum of the first three terms is an image with a dark ring around the object boundary, because excessive boundary intensity is subtracted by $(\boldsymbol{\alpha} * \mathbf{h}_F) \cdot (1 - \boldsymbol{\alpha} * \mathbf{h}_F) \cdot \widehat{\mathbf{f}}_B$. Therefore, the fourth term must compensate for the presence of the dark ring. Hence, going back to the first question, we *choose*

$$(\boldsymbol{\alpha} * \mathbf{h}_F) \cdot (1 - \boldsymbol{\alpha} * \mathbf{h}_F) \cdot \Delta\widehat{\mathbf{f}}_B = (\boldsymbol{\alpha} * \mathbf{h}_F) \cdot (1 - \boldsymbol{\alpha} * \mathbf{h}_F) \cdot \overline{\mathbf{g}}_B.$$

The results are shown in Figs. 5 and 6.

Finally, given the right hand side of (14), we solve the deconvolution problem $\widetilde{\mathbf{g}} = \mathbf{h}_F * \widetilde{\mathbf{f}}$. The solution $\widetilde{\mathbf{f}}$ is then combined with the estimated background $\widehat{\mathbf{f}}_B$ to give the final solution $\mathbf{f} = \boldsymbol{\alpha} \cdot \widetilde{\mathbf{f}} + (1 - \boldsymbol{\alpha}) \cdot \widehat{\mathbf{f}}_B$.

4. BLUR KERNEL ESTIMATION

In this section we discuss the proposed kernel estimation. Our method is an improved version of Xu and Jia,[11] Fergus *et al.*[20] and Cho and Lee.[10] Therefore, the focus of this section is on the modifications we make.

(a) (b) (c)

Figure 5. (a) Input blurred image $\mathbf{g}$. (b) Inpainted result $\widetilde{\mathbf{g}}$ using inpainting method 1. (c) Inpainted result $\widetilde{\mathbf{g}}$ using inpainting method 2

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

Figure 6. Illustration of (14) using inpainting method 2. (a) The observed image extracted by alpha-matte $(\boldsymbol{\alpha} * \mathbf{h}_F) \cdot \mathbf{g}$. (b) Subtract the result of (a) with the estimated background $(\boldsymbol{\alpha} * \mathbf{h}_F) \cdot \left[\mathbf{g} - (1 - \boldsymbol{\alpha} * \mathbf{h}_F) \cdot \widehat{\mathbf{f}}_B\right]$. (c) Add the result of (b) with the newly inpainted background $(\boldsymbol{\alpha} * \mathbf{h}_F) \cdot \left[\mathbf{g} - (1 - \boldsymbol{\alpha} * \mathbf{h}_F) \cdot \widehat{\mathbf{f}}_B + (1 - \boldsymbol{\alpha} * \mathbf{h}_F) \cdot \Delta\widehat{\mathbf{f}}_B\right]$. (d) Add the result of (c) with an approximation from the object boundary $(\boldsymbol{\alpha} * \mathbf{h}_F) \cdot \left[\mathbf{g} - (1 - \boldsymbol{\alpha} * \mathbf{h}_F) \cdot \widehat{\mathbf{f}}_B + (1 - \boldsymbol{\alpha} * \mathbf{h}_F) \cdot \Delta\widehat{\mathbf{f}}_B\right] + (1 - \boldsymbol{\alpha} * \mathbf{h}_F) \cdot \overline{\mathbf{g}}_B$.

4.1 Kernel Estimation by Strong Edges

A blurred image is the result of convolving a sharp image and the blur kernel. Thus, if one wants to estimate the blur kernel from a blurred image, a rough estimate of the sharp image must be used.[20]

To obtain such a sharp image estimate, we use the shock filter.[5,10] Without discussing the details of a shock filter, we consider the shock filter as a module that takes an image $\mathbf{g}$ and produces an output image $\mathbf{g}_S$:

$$\mathbf{g}_S = \text{shock filter}(\mathbf{g}).$$

Next, we use the fact that not all gradients of $\mathbf{g}_S$ are useful for blur kernel estimation.[11] Therefore, we multiply a mask $\mathbf{M}$ to the image gradient $\nabla\mathbf{g}_S$. We consider this edge selection process as a module that takes a shock filtered image $\mathbf{g}_S$ to produce an output image $\nabla^s\mathbf{g}_S$:

$$\nabla^s\mathbf{g}_S = \text{edge selection}(\mathbf{g}_S) = \mathbf{M} \cdot \nabla\mathbf{g}_S.$$

Therefore, $\mathbf{h}$ can be estimated from

$$\mathbf{h} = \underset{\mathbf{h}}{\text{argmin}} \ \|\nabla^s\mathbf{g}_S * \mathbf{h} - \nabla^s\mathbf{g}\|^2 + \lambda\|\mathbf{h}\|^2. \tag{15}$$

4.2 Kernel Estimation by Alpha-Mattes

A limitation of (15) is that (15) depends on the availability of strong edges in an image. In a two-layered blur image, it is possible that the foreground or background component does not have strong edges. For example, Fig. 7(b) shows an image where the cropped area of the foreground component does not have significant color variation.

For a two-layered blur problem, the alpha-matte provides useful information for blur kernel estimation because the alpha-matte is also blurred by the same blur kernel.[13] Therefore, by studying the transient characteristics of the alpha-matte, blur kernel can be estimated. For example in Fig. 7(a), while the cropped region in (b) does not have strong edges, the alpha-matte does. The estimation using alpha-matte is similar to (15):

$$\mathbf{h} = \underset{\mathbf{h}}{\text{argmin}} \ \|\nabla\boldsymbol{\alpha}_S * \mathbf{h} - \nabla\boldsymbol{\alpha}\|^2 + \lambda\|\mathbf{h}\|^2, \tag{16}$$

where $\boldsymbol{\alpha}$ is the estimated alpha-matte e, and $\boldsymbol{\alpha}_S = \text{shock filter}(\boldsymbol{\alpha})$.

Algorithm 2 Recovering from background blur

Given $\mathbf{g}$. Estimate $\mathbf{h}_B$ and $\boldsymbol{\alpha}$ (see Section 4).
Estimate $\widetilde{\mathbf{g}}$ using Algorithm 1.
Solve the deconvolution $\widetilde{\mathbf{g}} = \mathbf{h}_B * \mathbf{f}_B$.
Form the solution $\mathbf{f} = \boldsymbol{\alpha} \cdot \mathbf{g} + (1 - \boldsymbol{\alpha}) \cdot \mathbf{f}_B$.

Algorithm 3 Recovering from foreground blur

Given $\mathbf{g}$. Estimate $\mathbf{h}_F$ and $\boldsymbol{\alpha}$ (see Section 4).
Estimate $\widehat{\mathbf{f}}_B$ using Algorithm 1.
Estimate $\widetilde{\mathbf{g}}$ by (13).
Solve the deconvolution $\widetilde{\mathbf{g}} = \mathbf{h}_F * \widetilde{\mathbf{f}}$.
Form the solution $\mathbf{f} = \boldsymbol{\alpha} \cdot \widetilde{\mathbf{f}} + (1 - \boldsymbol{\alpha}) \cdot \widehat{\mathbf{f}}_B$.

4.3 Kernel Estimation using both Strong Edges and Alpha-Matte

The observation that alpha-matte can be used when edges in the foreground region are weak suggests that reversely the strong edges can be used when alpha-mattes are inaccurate. Show in Fig. 7(c)-(d) is an example where the alpha-matte of the hair regions cannot be estimated correctly, but there are strong edges in the cropped region. Therefore, to incorporate the strengths of both methods, we consider the estimation as

$$\mathbf{h} = \text{argmin } \mu\|\nabla^s \mathbf{g}_{IS} * \mathbf{h} - \nabla^s \mathbf{g}_I\|^2 + \gamma\|\nabla \boldsymbol{\alpha}_S * \mathbf{h} - \nabla \boldsymbol{\alpha}\|^2 + \lambda\|\mathbf{h}\|^2, \tag{17}$$

where μ, γ and λ are parameters to be discussed. Analytical solution for (17) exists, and is given by

$$\mathbf{h} = \mathcal{F}^{-1}\left\{\frac{\mu\overline{\mathcal{F}(\nabla^s \mathbf{g}_{IS})} \cdot \mathcal{F}(\nabla^s \mathbf{g}_I) + \gamma\overline{\mathcal{F}(\nabla \boldsymbol{\alpha}_S)} \cdot \mathcal{F}(\nabla \boldsymbol{\alpha})}{\mu|\mathcal{F}(\nabla^s \mathbf{g}_{IS})|^2 + \gamma|\mathcal{F}(\nabla \boldsymbol{\alpha}_S)|^2 + \lambda}\right\}, \tag{18}$$

where $\mathcal{F}$ is the Fourier Transform operator, $\overline{(\cdot)}$ denotes the complex conjugate over the argument and "$\cdot$" is the element-wise multiplication. Though not written explicitly in (17), the gradients $\nabla^s \mathbf{g}_{IS}$ (and similarly for $\nabla \boldsymbol{\alpha}_S$) are assumed to contain both horizontal and vertical directions, i.e., $\nabla^s \mathbf{g}_{IS} = [\partial_x^s \mathbf{g}_{IS},\ \partial_y^s \mathbf{g}_{IS}]$. Thus, $\overline{\mathcal{F}(\nabla^s \mathbf{g}_{IS})} \cdot \mathcal{F}(\nabla^s \mathbf{g}_I) = \overline{\mathcal{F}(\partial_x^s \mathbf{g}_{IS})} \cdot \mathcal{F}(\partial_x^s \mathbf{g}_I) + \overline{\mathcal{F}(\partial_y^s \mathbf{g}_{IS})} \cdot \mathcal{F}(\partial_y^s \mathbf{g}_I)$, and $|\mathcal{F}(\nabla^s \mathbf{g}_{IS})|^2 = |\mathcal{F}(\partial_x^s \mathbf{g}_{IS})|^2 + |\mathcal{F}(\partial_y^s \mathbf{g}_{IS})|^2$.

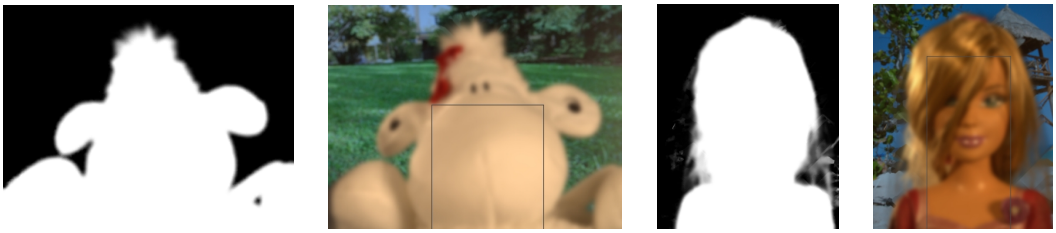
4.4 Choosing Parameters

The next question to ask is how to choose the parameters μ, γ and λ. Without loss of generality we set $\mu = 1$, as the minimizer of (17) is unchanged if we scale the objective function in (17) by $\frac{1}{\mu}$. Thus it remains to determine γ and λ. λ is the parameter for $\|\mathbf{h}\|^2$. Typically, meaningful results are found using a λ within the range $10^{-3} \leq \lambda \leq 10^{-2}$. In our experiments, λ is fixed at $\lambda = 10^{-2}$.

Choice of γ is a critical one as it sets relative emphasis between $\|\nabla \mathbf{g}_{IS} * \mathbf{h} - \nabla \mathbf{g}_I\|^2$ and $\|\nabla \boldsymbol{\alpha}_S * \mathbf{h} - \nabla \boldsymbol{\alpha}\|^2$. Our proposed method is based on the *confidence of* $\boldsymbol{\alpha}$. The confidence measure indicates the chance of getting an accurate alpha-matte. If it is likely that $\boldsymbol{\alpha}$ is a reliable estimate, then the weighing factor γ for $\|\nabla \boldsymbol{\alpha}_S * \mathbf{h} - \nabla \boldsymbol{\alpha}\|^2$ should be large. Otherwise, γ should be small.

γ can be evaluated from the performance across different methods. The intuition is that if an image is easy to be alpha-matted, then results of different alpha-matting methods should be similar. To verify our claim, we conducted an experiment by studying 15 alpha-matting methods available in.[21] Alpha-matting results (small tri-map case, 8 images per method), and the corresponding SAD scores are recorded. Discarding the worst performing method, we compute, for each pixel, the variance of alpha-matte values across the methods. That is, given an alpha-matte $\boldsymbol{\alpha}^k$ produced by the k-th method, we compute

$$\mathbf{V}(i,j) = \text{Var}\{\boldsymbol{\alpha}^1(i,j), \boldsymbol{\alpha}^2(i,j), \ldots, \boldsymbol{\alpha}^k(i,j)\}.$$



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

Figure 7. (a)-(b): An example where there is no strong edge in the cropped area, but the alpha-matte estimate is good. In this case, large γ can be used. (c)-(d): An example where the alpha-matte is not well estimated, but there is strong edge in the cropped area. In this case, large μ shall be used.

(a) avg SAD = 18, $\bar{v} = 0.0074$ (b) avg SAD = 10, $\bar{v} = 0.0054$ (c) avg SAD = 9, $\bar{v} = 0.0014$ (d) avg SAD vs $\bar{v}$ for 8 images.

Figure 8. Variance maps $\mathbf{V}$ for images "troll", "doll", and "elephant". White indicates variance = 0, and black indicates variance = maximized.

$V(i,j)$ is a map in which each pixel is the variance of the alpha-matting methods (see Fig. 8). It is a measure of agreement among the methods, for lower variance means the methods give similar alpha-matte values. Average of the variance over the entire image is then computed as

$$\bar{v} = \frac{1}{n} \sum_{i,j} \mathbf{V}(i,j),$$

where n is the number of pixels in $\mathbf{V}$. In Fig. 8(d), we show the relation between the average SAD score and the average variance for the 8 images provided in.[21] It can be seen that the variance and the SAD score shows a linear relation. This result implies that in the absence of ground-truth, the performance variance between methods is a good indicator of whether the image is easy to be alpha-matted.

The above experiment suggests that for each pixel of $\boldsymbol{\alpha}$, we can use a number of alpha-matting algorithms to measure the confidence. In our algorithm, we use shared matting, grow-cut[22] and closed-form matting.[23] More methods can be considered, but computing time will increase.

Finally, we define γ as

$$\gamma = C \exp\{-a\bar{v}\}, \tag{19}$$

where the constants $a = 1000$ and $C = 0.65$ are determined empirically.

4.5 Iterative Update of f and h

Our problem is a blind deconvolution problem. Therefore, intermediate update of the solution $\mathbf{f}_I$ is required for the iterative update of $\mathbf{h}$. Given the current estimate $\mathbf{h}$, $\mathbf{f}_I$ is updated by solving the following minimization problem

$$\mathbf{f}_I = \underset{\mathbf{f}_I}{\operatorname{argmin}} \ \ \|\mathbf{f}_I * \mathbf{h} - \mathbf{g}_I\|^2 + \kappa \|\nabla \mathbf{f}_I - \nabla^s \mathbf{g}_{IS}\|^2. \tag{20}$$

(20) is derived from an MAP framework consisting of a Gaussian fidelity term and a Gaussian prior. The Gaussian prior $\|\nabla \mathbf{f}_I - \nabla^s \mathbf{g}_{IS}\|^2$ measures the goodness of fit between the gradient of the unknown image $\mathbf{f}_I$ and the gradient of the shock filtered image $\mathbf{g}_{IS}$. The Gaussian prior $\|\nabla \mathbf{f}_I - \nabla^s \mathbf{g}_{IS}\|^2$ performs better than $\|\nabla \mathbf{f}_I\|^2$ in preserving edges and suppressing ringing artifacts.[11] The parameter κ is fixed at $\kappa = 10^{-2}$.

Algorithm 4 Blur Kernel Estimation

Given $\mathbf{g}$ and $\boldsymbol{\alpha}_0$. Crop an interior region $\mathbf{g}_I$ from the foreground of $\mathbf{g}$, and $\boldsymbol{\alpha}_I$ from $\boldsymbol{\alpha}_0$.
Initially set $\mathbf{f}_I = \mathbf{g}_I$. Let $\boldsymbol{\alpha} = \boldsymbol{\alpha}_0$.
while Not converge **do**
Compute $\mathbf{f}_{IS}$ = shock filter($\mathbf{f}_I$). Compute $\boldsymbol{\alpha}_S$ = shock filters($\boldsymbol{\alpha}$).
Compute $\nabla^s \mathbf{f}_{IS}$ = edge selection($\mathbf{f}_{IS}$).
Estimate μ, γ, λ, κ and ρ, then compute

$$\begin{aligned}
\mathbf{h} &= \underset{\mathbf{h}}{\operatorname{argmin}} \ \ \mu\|\nabla^s \mathbf{f}_{IS} * \mathbf{h} - \nabla^s \mathbf{f}_I\|^2 + \gamma\|\nabla \boldsymbol{\alpha}_S * \mathbf{h} - \nabla \boldsymbol{\alpha}\|^2 + \lambda\|\mathbf{h}\|^2 \\
\mathbf{f}_I &= \underset{\mathbf{f}_I}{\operatorname{argmin}} \ \ \|\mathbf{f}_I * \mathbf{h} - \mathbf{g}_I\|^2 + \kappa\|\nabla \mathbf{f}_I - \nabla^s \mathbf{g}_{IS}\|^2 \\
\boldsymbol{\alpha} &= \underset{\boldsymbol{\alpha}}{\operatorname{argmin}} \ \ \|\mathbf{h} * \boldsymbol{\alpha} - \boldsymbol{\alpha}_I\|^2 + \rho\|\boldsymbol{\alpha}\|_{TV}
\end{aligned}$$

end while

The updated solution $\mathbf{f}_I$ is then feedback to (17) by applying shock filter $\mathbf{f}_{IS}$ = Shock Filter($\mathbf{f}$) and replaces $\mathbf{g}_{IS}$ using $\mathbf{f}_{IS}$ in (17). Therefore, the iterative update of $\mathbf{f}_I$ and $\mathbf{h}$ is equivalent to solving

$$\begin{aligned}
\mathbf{f}_I &= \underset{\mathbf{f}_I}{\operatorname{argmin}} \ \ \|\mathbf{f}_I * \mathbf{h} - \mathbf{g}_I\|^2 + \kappa\|\nabla \mathbf{f}_I - \nabla^s \mathbf{f}_{IS}\|^2, \qquad \mathbf{f}_{IS} = \text{Shock Filter}(\mathbf{f}_I), \\
\mathbf{h} &= \underset{\mathbf{h}}{\operatorname{argmin}} \ \ \mu\|\nabla^s \mathbf{f}_{IS} * \mathbf{h} - \nabla^s \mathbf{f}_I\|^2 + \gamma\|\nabla \boldsymbol{\alpha}_S * \mathbf{h} - \nabla \boldsymbol{\alpha}\|^2 + \lambda\|\mathbf{h}\|^2.
\end{aligned}$$

The iteration repeats, until the relative change $\|\mathbf{f}_I^{(k+1)} - \mathbf{f}_I^{(k)}\|_2 / \|\mathbf{f}_I^{(k)}\|_2 \leq 10^{-3}$, where $\mathbf{f}_I^{(k)}$ is the solution at the k-th iteration.

In case of foreground blur, $\boldsymbol{\alpha}$ is a blurred alpha-matte which needs to be deblurred. To deblur $\boldsymbol{\alpha}$, we consider the following minimization

$$\boldsymbol{\alpha} = \underset{\boldsymbol{\alpha}}{\operatorname{argmin}} \ \ \|\mathbf{h} * \boldsymbol{\alpha} - \boldsymbol{\alpha}_I\|^2 + \rho\|\boldsymbol{\alpha}\|_{TV}, \tag{21}$$

where $\|\boldsymbol{\alpha}\|_{TV} = \sum_i \sqrt{[\nabla_x \boldsymbol{\alpha}]_i^2 + [\nabla_y \boldsymbol{\alpha}]_i^2}$ is the isotropic total variation norm, and ρ is a regularization parameter.

The blur kernel estimation step is outlined in Algorithm 4. The background blur case is similarly performed, where we set $\gamma = 0$, and crop an interior region from the background of $\mathbf{g}$.

5. RESULTS

A data set of 27 training images from `http://www.alphamatting.com` are downloaded for the comparisons. These images are all composed of a sharp foreground object and an out-of-focus blurred background scene. For images no.1-23, the object is placed in front of a high-definition (HD) monitor showing some background scenes, whereas for images no.24-27, the object is placed in front of real 3D scenes. Ground-truth alpha-mattes are available in this data set, but we will also test the proposed algorithm with estimated alpha-mattes later.

5.1 Blur Kernel Estimation

First, we compare the proposed kernel estimation method with the method by Fergus *et al.*[20] and the method by Xu and Jia.[11] Note that Xu and Jia supersedes Cho and Lee,[10] and the method by Fergus *et al.*[20] is used by Dai and Wu.[15]

We synthesize two foreground blur images using a Gaussian blur kernel of size 19×19 and variance $\sigma = 3$ (See Fig. 9 and Fig. 10). Shared matting[24] is applied to the blurred images so that alpha-mattes are estimated. Interior regions were cropped manually.

In Fig. 9, the cropped interior region does not have strong edges. Thus, applying[20] and[11] to the interior region does not produce good estimates (Fig. 9(a)-(b)). On the other hand, when the alpha-matte is poorly

estimated (Fig. 10), applying[20] and[11] to the alpha-matte does not give good estimates (Fig. 10(d)-(e)). The proposed method automatically weights the emphasis on the alpha-matte and the cropped region. Therefore, the kernel estimation result is better than the other two methods.

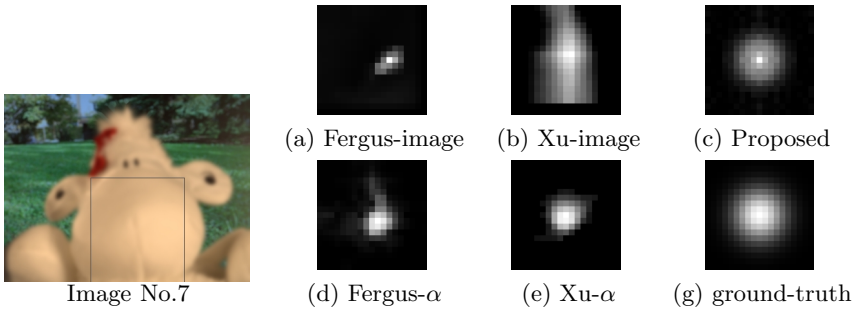


Image No.7 (a) Fergus-image (b) Xu-image (c) Proposed (d) Fergus-α (e) Xu-α (g) ground-truth

Figure 9. Kernel estimation for Image no. 7. (a) Fergus et al.[20] on cropped region. (b) Xu and Jia[11] on cropped region. (c) Proposed method. (d) Fergus et al.[20] on alpha-matte. (e) Xu and Jia[11] on alpha-matte. (f) Ground-truth kernel.

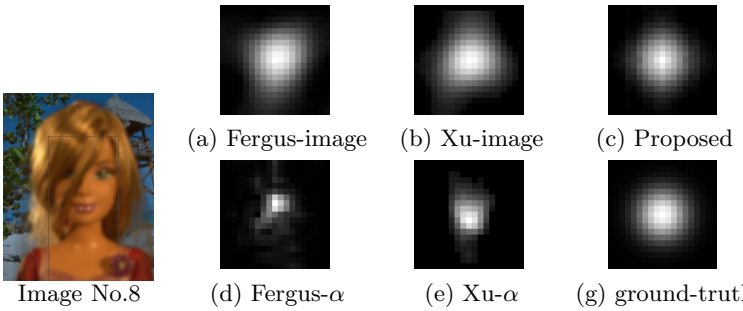


Image No.8 (a) Fergus-image (b) Xu-image (c) Proposed (d) Fergus-α (e) Xu-α (g) ground-truth

Figure 10. Kernel estimation for Image no. 8. (a) Fergus et al.[20] on cropped region. (b) Xu and Jia[11] on cropped region. (c) Proposed method. (d) Fergus et al.[20] on alpha-matte. (e) Xu and Jia[11] on alpha-matte. (f) Ground-truth kernel.

5.2 Real Background Blur

Next we compare the overall performance of the proposed method with three existing spatially variant deconvolution algorithms.

The first method to be compared is the spatially variant Lucy-Richardson (LR) algorithm.[13,14] In this method, the spatially variant blur $\mathbf{h}$ is expressed as a linear combination of invariant blurs. The deblurring step is performed via an iterative approach as

$$\mathbf{f}^{(k+1)} = \mathbf{f}^{(k)} \cdot \left[\mathbf{h}' * \left(\frac{\mathbf{g}}{\mathbf{h} * \mathbf{f}^{(k)}}\right)\right],$$

where $\mathbf{f}^{(k)}$ is the solution of the k-th iteration, $\mathbf{h}'$ is the flipped version of $\mathbf{h}$, i.e., $\mathbf{h}'(m,n) = \mathbf{h}(-m,-n)$. The multiplication "$\cdot$" and the division in the parenthesis are element-wise operations. The algorithm terminates when $\|\mathbf{f}^{(k+1)} - \mathbf{f}^{(k)}\|^2 / \|\mathbf{f}^{(k)}\|^2 \leq 10^{-3}$.

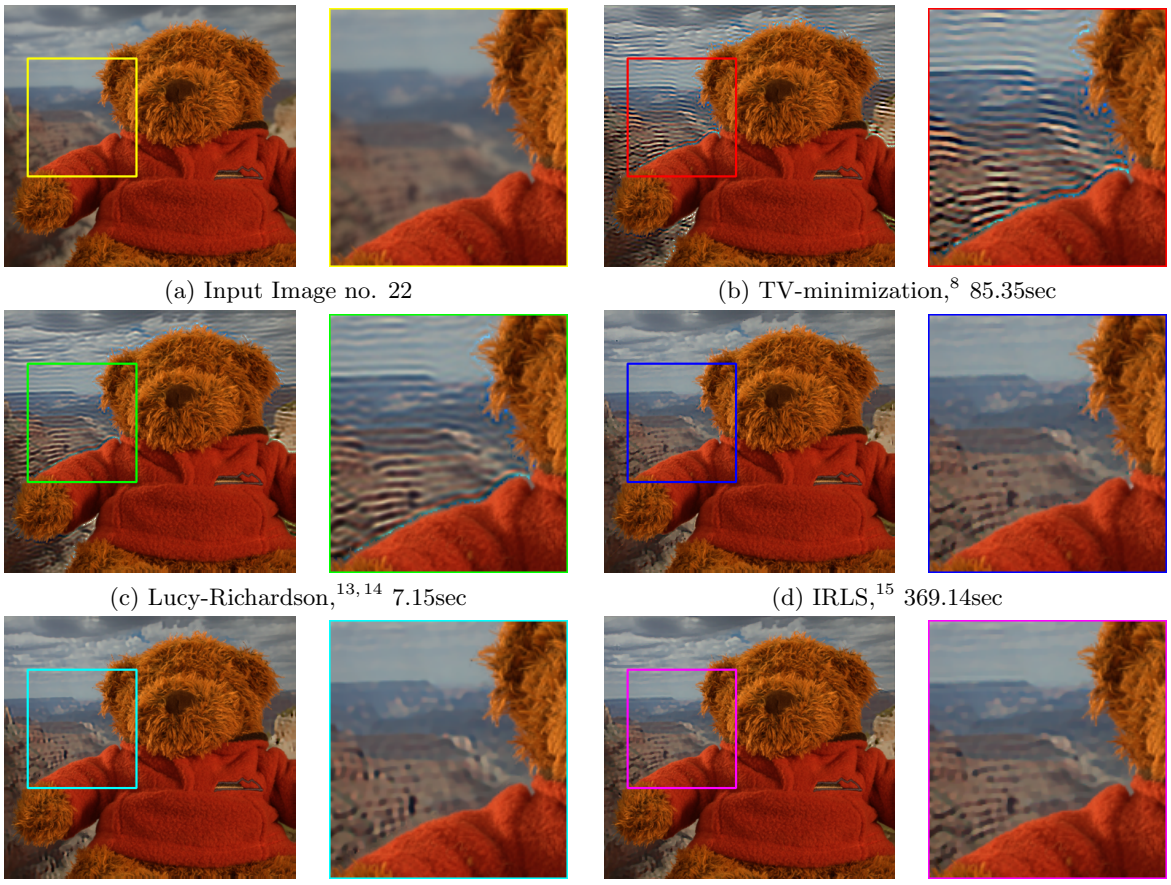
The second method is a modified version of the total variation (TV) minimization using augmented Lagrangian method.[8] We express the spatially variant operator $\mathbf{h}$ as a non circulant matrix $\mathbf{H}$. Then the $\mathbf{f}$-subproblem (Equation (14) in Chan *et al.*[8])

$$(\mu\mathbf{H}^T\mathbf{H} + \rho\mathbf{D}^T\mathbf{D})\mathbf{f} = \mu\mathbf{H}^T\mathbf{g} + \rho\mathbf{D}^T\mathbf{u} - \mathbf{D}\mathbf{y}$$

is solved using conjugate gradient iterations.

The third method is the one by Dai and Wu,[15] which is the most relevant method to our approach. This method solves the minimization problem

$$\underset{\mathbf{f}_F,\,\mathbf{f}_B}{\text{minimize}} \quad \|\mathbf{g} - \alpha\mathbf{f}_F - (1-\alpha)(\mathbf{h}_B * \mathbf{f}_B)\|^2 + \lambda_1\|\mathbf{f}_F\|_{TV} + \lambda_2\|\mathbf{f}_B\|_{TV}, \tag{22}$$



(a) Input Image no. 22

(b) TV-minimization,[8] 85.35sec

(c) Lucy-Richardson,[13,14] 7.15sec

(d) IRLS,[15] 369.14sec

(e) Proposed (Ground-Truth α), 9.45sec

(f) Proposed (Estimated α), 9.62sec

Figure 11. Real image background deblurring for Image No. 22. In methods shown here, TV-minimization, Lucy-Richardson, IRLS and Proposed (Ground-Truth α) use the ground-truth alpha-matte for blur kernel estimation and deblurring. However, Proposed (Estimated α) uses the shared matting method for the same tasks.

using an iterative reweighted least-squares (IRLS) method. Here, we use the standard isotropic TV norm $\|\cdot\|_{TV}$ instead of the l_p-norm ($p = 0.8$) in,[15] because the goal of this paper is not to compare different TV norms. In solving (22), $\mathbf{f}_F$ and $\mathbf{f}_B$ are determined simultaneously. Since the linear operators in (22) are not block-circulant, Fourier Transform cannot be used. Therefore, the speed of solving (22) is expected to be slow.

Fig. 11 is one of the 27 images being tested. Referring to the image, the foreground is sharp and the background is blurred. Since the blur kernels are unknown, we applied the proposed algorithm to estimate the blur kernel. The estimated blur kernel is then applied to the three existing methods listed in Fig. 11.

Two versions of the proposed method are also tested. Proposed (Ground-Truth α) uses the ground-truth alpha-matte for kernel estimation, background inpainting and deblurring, whereas Proposed (Estimated α) uses the shared matting results for kernel estimation, background inpainting and deblurring.

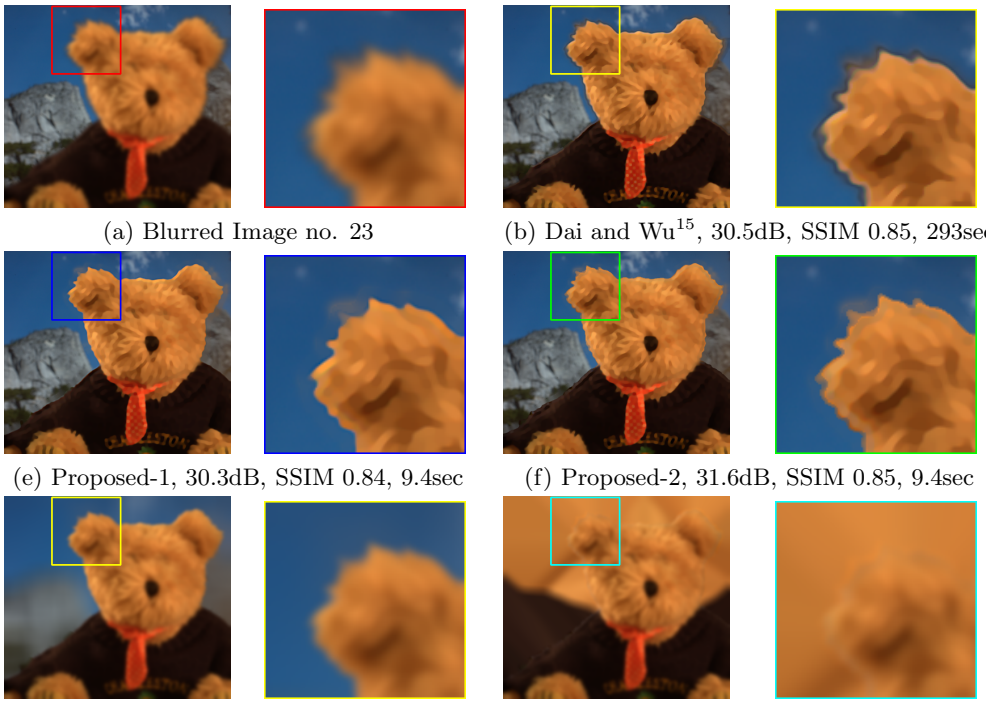
The run-time of these methods are recorded based on a desktop computer with Intel Quadcore Q9550 2.8GHz, 4GB DDR3, MATLAB/ Windows 7. It can be seen that the proposed method shows significantly faster speed than Dai and Wu's method,[15] and better recovery results than the method by Chan *et al.*[8]

5.3 Synthetic Foreground Blur

We now show the PSNR and SSIM comparisons between existing methods. To do so, the foreground of the 27 testing images are *synthetically* blurred. The blur kernel in this experiment is a Gaussian blur kernel with size 19×19 and variance $\sigma = 3$. Since the methods by Chan *et al.*,[8] Dai and Wu[14] and Jia[13] are evidently not able to handle the two-layer blur, comparing to Dai and Wu[15] is sufficient.

For Dai and Wu,[15] we first apply the proposed kernel estimation algorithm to estimate the blur kernel. Once the kernel is estimated, it is fixed in the iteration of the IRLS algorithm. For fairness alpha-matting is performed using shared matting,[24] same as the proposed method.

Some results are shown in Fig. 12. It can be observed that the proposed method generally produces similar image quality. However, the run time is significantly shorter. PSNR, SSIM and the run-times are listed in Table 1.



(a) Blurred Image no. 23 (b) Dai and Wu[15], 30.5dB, SSIM 0.85, 293sec

(e) Proposed-1, 30.3dB, SSIM 0.84, 9.4sec (f) Proposed-2, 31.6dB, SSIM 0.85, 9.4sec

(c) $\widetilde{\mathbf{g}}$ using Proposed-1 (d) $\widetilde{\mathbf{g}}$ using Proposed-2

Figure 12. Synthetic image foreground deblurring for Image No. 23. PSNR, SSIM, and Run-time can be referred to Table 1.

5.4 Real Foreground Blur

Finally, we applied the proposed algorithm to deblur real images with blurred foreground. The images were captured using a Canon ESO REBEL T2i camera. The focal length is 24mm, and ISO is 400. The images consist of a toy placed in front of posters with different background contents. The distance between the object and the background is approximately 30cm.

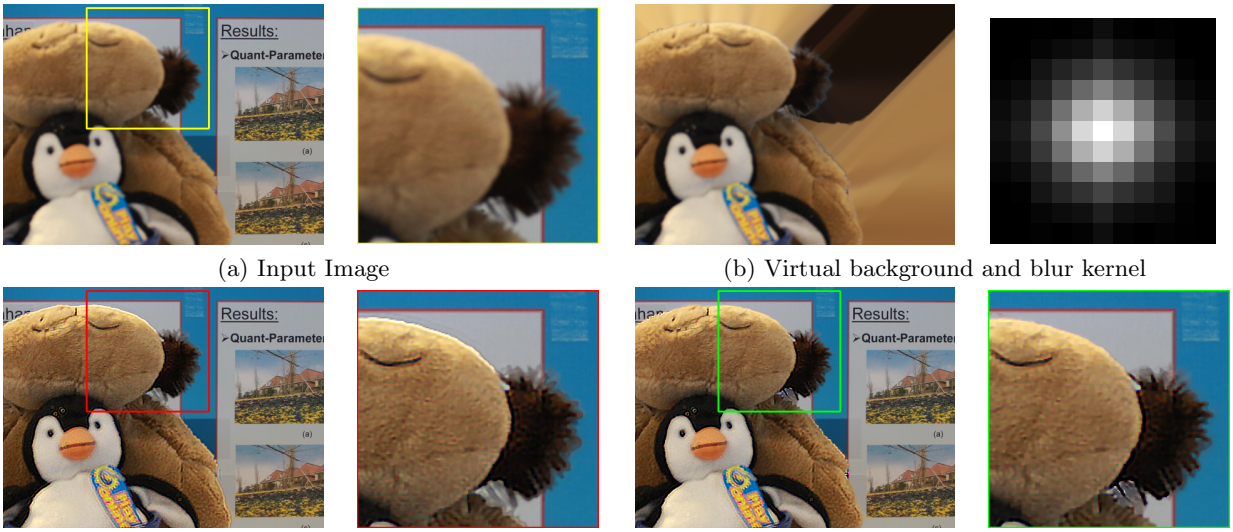
To recover the foreground image, we applied the shared matting algorithm to first extract the foreground object (with blurred edges). Then, the proposed blur kernel estimation method is run to determine the blur kernel, and consequently the deblurring step could be performed.

Fig. 13 shows one of the results. Similar to the case of synthetic blur, the proposed method is able to recover the image giving comparable quality with Dai and Wu's method.[15] However, the computation time of the proposed method is significantly shorter.

More results are available at `http://videoprocessing.ucsd.edu/~stanleychan`.

6. CONCLUSION

This paper has two main contributions. First, we proposed a new blur kernel estimation algorithm for the two-layer out-of-focus blur problem. The new algorithm encapsulates the strength of two existing classes of methods by utilizing both the alpha-matte transient and image gradient. Experimental results showed that the new



(a) Input Image (b) Virtual background and blur kernel

(c) Dai and Wu[15] (d) Proposed

Figure 13. Real image foreground deblurring (1). (a) The input image is captured using Canon ESO T2i camera. The background is about 30cm from the foreground. (b) The virtual background created by the proposed algorithm and the blur kernel estimated using the proposed algorithm. (c) Deblurring results by Dai and Wu.[15] (d) Deblurring results by the proposed method. In this image, alpha-matte is estimated using shared-matting.

algorithm is more robust than existing blur kernel estimation methods. Second, we proposed a new method to transform the spatially variant blur problem to two spatially invariant blur problems so that fast deconvolution algorithms can be used. The new method predicts the occluded background for the background blur case, and creates virtual background for the foreground blur case. Experimental results showed that the proposed method produces better recovery results than existing methods while at a significantly faster speed.

The proposed method is limited by a number of factors: accuracy of the initial alpha-matte estimation, signal-to-noise ratio of the image, degree of blurriness of the image, and presence of object motion. Future research shall be focused on overcoming these issues.

REFERENCES

[1] Goodman, J., [*Introduction to Fourier Optics*], Roberts & Company Publishers, 4 ed. (2004).
[2] Gonzalez, R. and Woods, R., [*Digital Image Processing*], Prentice Hall (2007).
[3] Lucy, L., "An iterative technique for the rectification of observed distributions," *Astronomical Journal* **79**, 745–754 (1974).
[4] Mesarovic, V., Galatsanos, N., and Katsaggelos, A., "Regularized constrained total least-squares image restoration," *IEEE TIP* **4**, 1096–1108 (1995).
[5] Rudin, L., Osher, S., and Fatemi, E., "Nonlinear total variation based noise removal algorithms," *Physica D* **60**, 259–268 (1992).
[6] Chambolle, A., "An algorithm for total variation minimization and applications," *J. of Math. Imaging and Vision* **20**(1-2), 89–97 (2004).
[7] Huang, Y., Ng, M., and Wen, Y., "A fast total variation minimization method for image restoration," *SIAM Multiscale model and simulation* **7**, 774–795 (2008).
[8] Chan, S., Khoshabeh, R., Gibson, K., Gill, P., and Nguyen, T., "An augmented Lagrangian method for total variation video restoration," *IEEE TIP* **20**, 3097–3111 (Nov 2011).
[9] Shan, Q., Jia, J., and Agarwala, A., "High-quality motion deblurring from a single image," in [*ACM SIGGRAPH*], (2008).
[10] Cho, S. and Lee, S., "Fast motion deblurring," in [*ACM SIGGRAPH*], (2009).
[11] Xu, L. and Jia, J., "Two-phase kernel estimation for robust motion deblurring," in [*ECCV*], (2010).

Image No.	Size		PSNR (dB)			SSIM			Run-time (sec)		
	rows	cols	DW[15]	Proposed-1	Proposed-2	DW[15]	Proposed-1	Proposed-2	DW[15]	Proposed-1	Proposed-2
1	249	400	31.3	29.5	30.9	0.86	0.84	0.85	831.44	7.53	7.53
2	262	400	27.5	26.8	26.9	0.83	0.82	0.82	783.42	7.25	7.28
3	400	320	31.2	33.7	33.9	0.86	0.88	0.88	955.35	8.15	8.21
4	282	400	25.2	28.6	28.9	0.76	0.81	0.82	109.18	7.59	7.52
5	276	400	29.4	30.0	32.7	0.91	0.92	0.93	816.58	7.62	7.61
6	339	400	31.6	31.5	33.7	0.92	0.92	0.93	1230.53	9.58	9.57
7	309	400	31.9	29.3	32.6	0.92	0.90	0.92	92.34	8.54	8.55
8	400	324	26.8	28.9	29.9	0.80	0.84	0.84	217.53	8.61	8.61
9	400	322	29.9	30.7	32.1	0.83	0.86	0.86	293.05	8.75	8.76
10	286	400	31.7	31.2	32.2	0.85	0.85	0.86	805.46	7.54	7.54
11	311	400	27.4	27.3	27.5	0.76	0.75	0.76	148.23	8.24	8.25
12	264	400	30.6	31.0	32.3	0.86	0.87	0.87	102.81	6.84	6.84
13	298	400	27.4	27.9	28.5	0.74	0.76	0.76	1561.58	7.81	7.81
14	265	400	31.8	30.3	32.3	0.93	0.93	0.94	109.45	7.26	7.28
15	245	400	31.3	32.2	34.9	0.93	0.94	0.94	78.68	6.89	6.87
16	268	400	30.5	28.8	31.1	0.88	0.87	0.88	113.42	7.19	7.20
17	283	400	32.1	31.8	33.4	0.89	0.90	0.90	296.12	8.05	8.08
18	323	400	30.5	29.9	31.4	0.90	0.90	0.91	1173.86	8.42	8.42
19	290	400	27.5	27.3	27.4	0.82	0.81	0.81	831.24	7.72	7.71
20	288	400	32.3	33.1	35.8	0.90	0.90	0.92	175.64	7.81	7.80
21	332	400	29.6	29.5	30.3	0.83	0.84	0.85	2188.35	8.39	8.40
22	360	400	29.7	30.2	30.2	0.78	0.78	0.78	1308.15	9.14	9.15
23	358	400	30.5	30.3	31.6	0.85	0.84	0.85	293.24	9.41	9.41
24	272	400	28.9	28.5	29.3	0.85	0.86	0.86	85.31	7.45	7.45
25	266	400	23.8	23.2	23.3	0.83	0.82	0.82	250.61	7.16	7.18
26	302	400	21.4	21.1	21.6	0.75	0.76	0.76	2598.53	7.98	7.92
27	304	400	22.7	22.1	22.7	0.74	0.74	0.74	201.42	8.12	8.15

Table 1. PSNR, SSIM and run-time comparisons among Dai and Wu,[15] Proposed method 1 and Proposed method 2 on synthetic foreground blurred images

[12] Nagy, J. and O'Leary, D., "Restoring images degraded by spatially variant blur," *SIAM Journal on Scientific Computing* **19**(4) (1998).

[13] Jia, J., "Single image motion deblurring using transparency," in [*IEEE CVPR*], (2007).

[14] Dai, S. and Wu, Y., "Motion from blur," in [*IEEE CVPR*], (2008).

[15] Dai, S. and Wu, Y., "Removing partial blur in a single image," in [*IEEE CVPR*], (2009).

[16] Asada, N., Fujiwara, H., and Matsuyama, T., "Seeing behind the scene: Analysis of photometric properties of occluding edges by the reversed projection blurring model," *IEEE TPAMI* **20**(2), 155–167 (1998).

[17] McGuire, M., Matusik, W., Pfister, H., Hughes, J., and Durand, F., "Defocus video matting," in [*ACM SIGGRAPH*], (2005).

[18] Wang, J. and Cohen, M., "Image and video matting: A survey," *Foundations and Trends in Computer Graphics and Vision* **3**(2) (2007).

[19] Criminisi, A., Perez, P., and Toyama, K., "Object removal by exemplar-based inpainting," in [*IEEE CVPR*], (2003).

[20] Fergus, R., Singh, B., Hertzmann, A., Roweis, S. T., and Freeman, W., "Removing camera shake from a single photograph," *ACM Transactions on Graphics* **25**(3), 787–794 (2006).

[21] Rhemann, C., Rother, C., Wang, J., Gelautz, M., Kohli, P., and Rott, P., "A perceptually motivated online benchmark for image matting," in [*IEEE CVPR*], (June 2009).

[22] Vezhnevets, V. and Konouchine, V., "Grow-cut - interactive multi-label n-d image segmentation," in [*Proceedings of Graphicon*], 150156 (2005).

[23] Levin, A., Lischinski, D., and Weiss, Y., "A closed form solution to natural image matting," *IEEE TPAMI* **30**(2), 228–242 (2008).

[24] Gastal, E. and Oliveira, M., "Shared sampling for real-time alpha matting," *Computer Graphics Forum* **29**, 575–584 (May 2010). Proceedings of Eurographics.